3D Object Representations for Recognition

by

Yu Xiang

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Electrical Engineering-Systems)
in The University of Michigan
2016

Doctoral Committee:

Assistant Professor Silvio Savarese, Co-Chair, Stanford University
Professor Alfred Hero, Co-Chair
Associate Professor Jason Corso
Assistant Professor Jia Deng

© Yu Xiang 2016

All Rights Reserved

For my family and friends

ACKNOWLEDGEMENTS

First of all, I would like to thank my PhD advisor, Prof. Silvio Savarese, for all the advices and support during my PhD. I have learned the way of conducting research from Silvio. I wish to thank my co-advisor, Prof. Alfred Hero, for his valuable suggestions and support during my PhD.

Second, I would like to thank all the lab members for their help in the past: Min Sun, Wongun Choi, Byungsoo Kim, Yingze Bao, Jingen Liu, Ryan Tokola, Yu-Wei Chao, Zhen Zeng, Changkyu Song, Jie Li at University of Michigan, and Roozbeh Mottaghi, Tian Lan, Hyun Oh Song, David Held, Christopher B. Choy, Alexandre Alahi, Kuan Fang, Kevin Chen at Stanford University.

Finally, I wish to thank my family and friends for their support and encouragement throughout my study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

3D Object Representations for Recognition

by

Yu Xiang

Co-Chairs: Silvio Savarese and Alfred Hero

Object recognition from images is a longstanding and challenging problem in computer vision. The main challenge is that the appearance of objects in images is affected by a number of factors, such as illumination, scale, camera viewpoint, intra-class variability, occlusion, truncation, and so on. How to handle all these factors in object recognition is still an open problem. In this dissertation, I present my efforts in building 3D object representations for object recognition. Compared to 2D appearance based object representations, 3D object representations can capture the 3D nature of objects and better handle viewpoint variation, occlusion and truncation in object recognition.

I introduce three new 3D object representations: the 3D aspect part representation, the 3D aspectlet representation and the 3D voxel pattern representation. These representations are built to handle different challenging factors in object recognition. The 3D aspect part representation is able to capture the appearance change of object categories due to viewpoint transformation. The 3D aspectlet representation and the 3D voxel pattern representation are designed to handle occlusions between objects in addition to viewpoint change. Based on these representations, we propose new object recognition methods and conduct experiments on benchmark datasets to verify the advantages of our methods.

Furthermore, we introduce, PASCAL3D+, a new large scale dataset for 3D object recognition by aligning objects in images with 3D CAD models. We also propose two novel methods to tackle object co-detection and multiview object tracking using our 3D aspect part representation, and a novel Convolutional Neural Network-based approach for object detection using our 3D voxel pattern representation. In order to track multiple objects in videos, we introduce a new online multi-object tracking framework based on Markov Decision Processes. Lastly, I conclude the dissertation and discuss future steps for 3D object recognition.

CHAPTER I

Introduction

Object recognition remains as one of the core problems in computer vision since the inception of the field. Inspired by the ability of human beings in recognizing objects in our daily life, pioneers in computer vision aimed at building computational models that are capable of recognizing objects from digital images. Remarkable progress in object recognition has been achieved nowadays due to endless efforts of computer vision researchers and significant advances in related fields, such as computer hardware, artificial intelligence, Internet, and so on. For example, one of the most successful achievements is human face recognition. A very recent paper in human face verification using a deep neural network achieves human-level performance *Taigman et al.* (2014). Although general object recognition beyond human face is still a grand challenge, recognition performances on benchmarking datasets, such as the PASCAL VOC Challenge *Everingham et al.* (b) and the ImageNet Challenge *Russakovsky et al.* (2015b) are improving year by year. Breakthroughs in object recognition will definitely bring significant impacts to the society since many applications are built upon object recognition techniques, such as road object recognition in autonomous driving, human recognition in surveillance and daily object recognition in robotics.

Object recognition is challenging due to the complexity of the objects themselves and the 3D world objects live in. A 2D image is a projection of the 3D world.

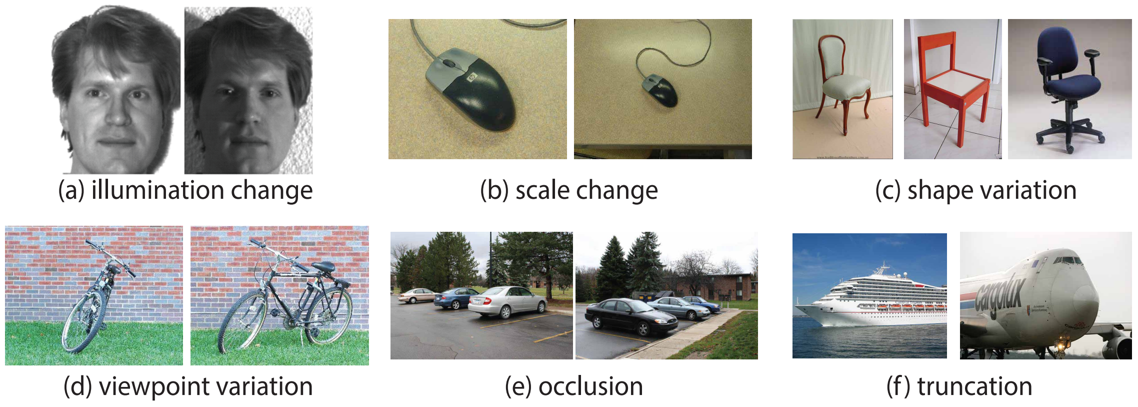


Figure 1.1: Illustration of six different challenging factors in object recognition. (a) The appearance of objects in images are different according to different illuminations. (b) The size of objects in images changes. (c) Objects in the same category can have very different 3D shapes. (d) The appearance of objects are different from different viewpoints. (e) Under occlusions, the appearance of occluded objects changes. (f) Objects are truncated due to limited field of view of the camera.

How to infer or understand the 3D world through 2D images is an ultimate goal of computer vision. In this sense, our visual processing system is amazing since it is able to recognize thousands of objects in our environment within a very short time period. In terms of designing computational models for object recognition, there are a number of challenging factors one needs to deal with, such as illumination change, scale change, shape variation, viewpoint variation, occlusion, truncation, and so on. Figure 1.1 illustrates these factors in more details. As we can see, the appearance of objects in images is effected by various aspects. In order to recognize objects in the real world, an object recognition system needs to handle all these aspects successfully.

In every object recognition method, object representation is a crucial component, which encompasses all the knowledge or information about the objects to be recognized. The object representation directs the visual processing pipeline in recognition. Although the mechanism about how human represent objects in our brains is still unclear, computer vision researchers have explored different ways in representing objects and utilized these representations in computational models to recognize objects from images. These representations could be designed manually or learned from data, and different representations have their own advantages in handling various challenging factors as described in Figure 1.1. For instance, in the early days of computer vision, objects are represented by 3D Computer-Aided Design (CAD) models, and recognition is performed by matching projections of the 3D CAD models to input images *Lowe* (1987). Recent progresses in object recognition leverage machine learning techniques and learn object representations from visual data *Fergus et al.* (2003); *Savarese and Fei-Fei* (2007); *Su et al.* (2009); *Felzenszwalb et al.* (2010). Overall, learning-based approaches are more robust than feature matching. According to the degree that an object representation captures the 3D nature of objects, we can categorize object representations into three classes: 2D representation, 2.5D representation and 3D representation.

2D representations are built to capture the 2D image appearance of objects without explicitly modeling the 3D properties of objects such as 3D pose or 3D shape. Representative approaches include the Deformable Part Model (DPM) for object detection *Felzenszwalb et al.* (2010) and deep neural network-based approaches *Krizhevsky et al.* (2012); *Girshick et al.* (2014). 2D representations are suitable for image-level recognition tasks, such as object classification or 2D localization. However, they cannot be used to recognize 3D properties of objects including 3D pose and 3D geometry, or perform scene-level understanding such as occlusion reasoning between objects. 2.5D representations *Savarese and Fei-Fei* (2007); *Su et al.* (2009); *Gu and Ren* (2010) encode the 2D appearance of objects in different viewpoints, and these appearance models can be connected according to their viewpoints. As a result, 2.5D representation can be used to recognize objects from various viewpoints. However, it is challenging for 2.5 representation to recognize objects from unseen viewpoints, and they cannot capture the 3D shape of object. In 3D representations, 3D models of objects are designed *Liebelt et al.* (2008); *Xiang and Savarese* (2012) or learned from visual data *Yan et al.* (2007) to represent the objects. The 3D models capture the 3D shape of object and can be used to recognize objects from different viewpoints. In addition, 3D representations are useful in scene understanding, such as estimating the 3D spatial layout of objects or reasoning about occlusions between objects *Zia et al.* (2013); *Xiang and Savarese* (2013); *Zia et al.* (2014). It is clear that 3D representations are able to recognize the 3D properties of objects compared with 2D representations. But the challenge for 3D representation based methods is to achieve the same generalization and discrimination power as 2D representations.

In this dissertation, I present our efforts on designing 3D object representations for object recognition. Inspired by the aspect graph theory *Koenderink and van Doorn* (1979) for multi-view object representation, we propose a novel 3D aspect part representation for recognizing objects from different viewpoints *Xiang and Savarese* (2012), which is described in Chapter 2. In this work, an object category is represented by a set of 3D aspect parts which are built from 3D CAD models, and we propose a new aspect layout model which leverages the 3D representation to localize 3D aspect parts in images. Compared to 2D object representations in the literature *Dalal and Triggs* (2005); *Felzenszwalb et al.* (2010), our method is able to jointly detect the objects and estimate the pose of the objects. Different from the previous 3D object representations *Savarese and Fei-Fei* (2007); *Su et al.* (2009) that mainly focus on viewpoint estimation, our 3D aspect part representation enables us to localize the 3D aspect parts of the object under different viewpoints.

In Chapter 3, I introduce the 3D aspectlet representation to recognize multiple objects from a single input image and reason about occlusions between objects *Xiang and Savarese* (2013). Handling occlusion in object detection is a challenging problem. Most previous 3D object representations *Liebelt et al.* (2008); *Xiang and Savarese* (2012) cannot model occlusion explicitly. The 3D aspectlet representation is a generalization of the 3D aspect part representation in order to handle occlusions between objects.

In Chapter 4, I present the 3D voxel pattern representation which is built in a data-driven way to deal with a number of challenging factors in object recognition including viewpoint variation, occlusion and truncation *Xiang et al.* (2015b). Unlike previous 3D object representations *Zia et al.* (2013); *Xiang and Savarese* (2013); *Zia et al.* (2014), 3D voxel pattern jointly encodes 3D shape, viewpoint, occlusion and truncation into a uniform 3D space, and we discover 3D voxel patterns with a data-driven approach. These properties enable us to improve the performance on object detection and pose estimation in complex scenes.

In Chapter 5, I present our efforts on building a large scale 3D object recognition dateset, PASCAL3D+ *Xiang et al.* (2014a), where we provide 3D annotations to 12 rigid categories in PASCAL VOC 2012 *Everingham et al.* (b) by aligning 2D objects in these images with 3D CAD models. PASCAL3D+ is useful for the community to study different problems such as object detection and pose estimation, object keypoint localization and 3D shape reconstruction. It can also be helpful in benchmarking the performance of 3D object recognition methods.

In the following chapters, I describe different methods that apply the introduced 3D object representations to real world problems. Chapter 6 and Chapter 7 describe two applications based on our 3D aspect part representation: object co-detection *Bao et al.* (2012c) and multiview object tracking *Xiang et al.* (2014b). Chapter 8 presents a novel Convolutional Neural Network-based approach for object detection using our 3D voxel pattern representation, and Chapter 9 describes a new multi-object tracking framework based on Markov Decision Processes *Xiang et al.* (2015a). Finally, Chapter 10 concludes the dissertation and discusses future steps.

CHAPTER II

3D Aspect Part Representation

2.1 Introduction

In most traditional object recognition methods, object categories are represented as 2D flat entities. The focus lies more on taming the intra-class variability within each category (indeed a very challenging problem) rather than seeking to model the intrinsic 3D nature of the object. Also, most of the methods aim at detecting objects in images and identifying them using a bounding box rather than estimating their geometrical properties such as the object 3D pose or the 3D layout configuration of their parts. While the 2D object detection problem is very useful in many applications such as Internet-based image search (and impressive results have been obtained), it is less so in applications such as robotics, autonomous navigation and manipulation. In such applications it is critical not only to recognize objects in 2D but also to estimate their locations and poses in 3D (Fig. 2.1). Moreover, the ability to parse the object layout and identify object functional elements such as the back or the seat of a sofa is crucial for enabling an agent to effectively interact with the objects in the scene (Fig. 2.1).

In this paper, we address the problem of detecting object categories, determining their 3D poses and estimating the objects' 3D layout from a single image. By object's layout we mean the configuration of object parts in 3D (Fig. 2.1). Instead of

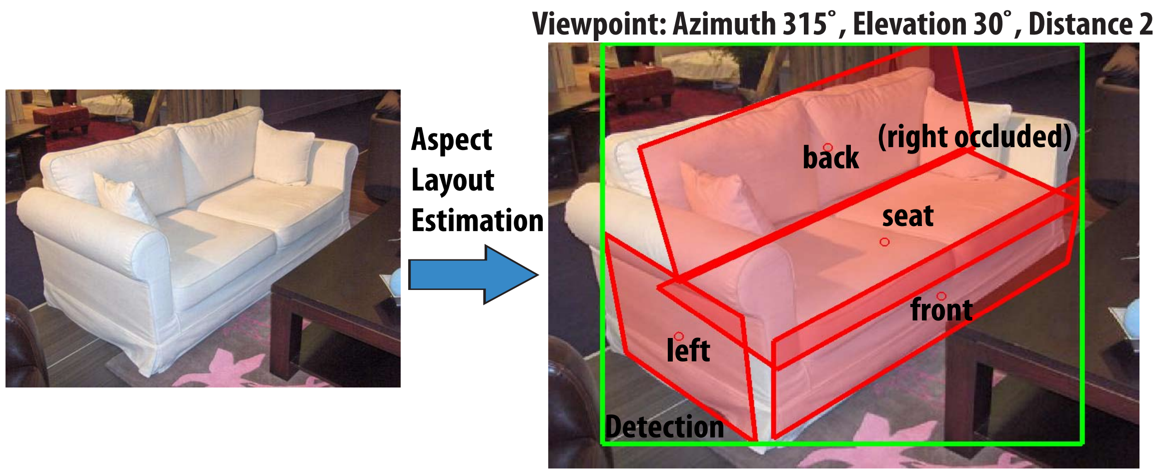


Figure 2.1: Illustration of aspect layout estimation of a sofa. Left: input image with a sofa. Right: the estimation result given by our method: the sofa is detected by the green bounding box, its viewpoint is estimated and its aspect parts are either located by a red quadrilateral or determined as self-occluded.

considering an arbitrary definition of *object part*, we seek to identify parts that have geometrical and topological relevance. We call these parts *aspect parts*. An aspect part can be defined as a portion of the object whose entire 3D surface is approximately either entirely visible from the observer or entirely non-visible (i.e., occluded). The seat and the back of a sofa are two examples of approximated aspect parts. The combination of the seat and the back of the sofa is *not* an aspect part as there are certain viewpoints from which either the back is visible and the seat is not, or, conversely, the seat is visible and the back is not. A planar surface is an ideal aspect part. The concept of aspect part is related to that of aspect graph which was introduced in the pioneering work by Koenderink and Doorn *Koenderink and van Doorn* (1979).

The ability to estimate the pose and the 3D layout of an object is connected to several key computer vision problems. An aspect part can be related to the concept of object affordance or functional part such as the seat or back of a sofa, thus our work is critical in object affordance estimation problems such as these addressed in *Stark et al.* (2008). Also, it allows us to characterize the object with geometrical attributes

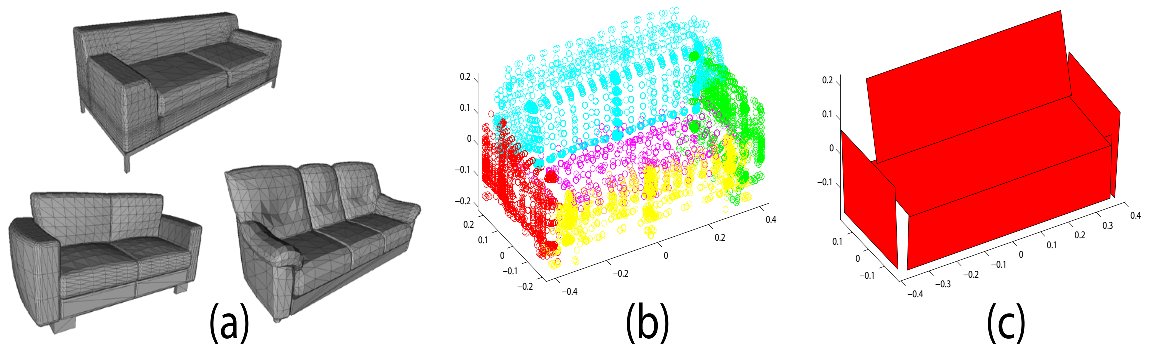


Figure 2.2: Overview of the training steps to build the 3D object model $O = (o_1, o_2, \ldots, o_n)$. We illustrate an example from the sofa category. i) Collect 3D CAD models of *sofa*, rescale the CAD models to fit into a unit sphere and orient them along their dominant dimension. Fig. 2.2(a) shows three 3D CAD models of *sofa* we collected from Google 3D Warehouse *Trimble*. ii) Identify aspect parts, segment 3D points in each CAD model according to the aspect parts using manual annotations and aggregate all the 3D points from the CAD models. Fig. 2.2(b) shows the 3D point cloud after segmentation and aggregation, where different colors represent different aspect parts. iii) Fit a rectangle to the 3D points belonging to each aspect part. First, fit a 2D plane to these 3D points, and then project the 3D points onto the plane. Finally, draw a bounding box of the projected points in the plane to obtain the rectangle for the aspect part. Fig. 2.2(c) shows the 3D model we built for *sofa*.

such as "it has an horizontal support surface" or "it has a back surface" which are suitable for fine-grained object recognition, zero-shot learning or transfer learning problems *Farhadi et al.* (2009a). Our work provides tools for effectively modeling object-scene interactions *Yao and Fei-Fei* (2010) and for scene layout understanding *Hoiem et al.* (2008); *Hedau et al.* (2010); *Bao et al.* (2011); *Bao and Savarese* (2011). Finally, it can be useful for automatic 3D object reconstruction or rough 3D shape prototyping from a single image *Thomas et al.* (2007); *Arie-Nachimson and Basri* (2009); *Sun et al.* (2010).

In this work, we propose a new model for jointly solving the object detection, pose classification and layout estimation problem. We call this model the Aspect Layout Model (ALM). ALM is constructed as follows. Aspect parts and their 3D configuration are automatically learnt from a set of 3D CAD models from which the aspect parts are manually identified for each object category (see Fig. 2.2 for details). The relationship between the 3D configuration of aspect parts and their corresponding projections (observations) in the images are modeled using a discriminative framework based on Conditional Random Fields (CRFs) *Lafferty et al.* (2001) with maximal margin parameter estimation. The unary potential of the CRF captures appearance and shape properties of each projected aspect part in the image. Projected aspect parts are shared across views and their appearances and shapes are rectified to their most frontal poses in order to guarantee view invariance. As a result, only one 2D part template is trained for each aspect part regardless of the number of viewpoints in the dataset. The pairwise potential is used to enforce spatial constraints to the relative 2D locations of aspect parts.

To summarize, our work has the following key contributions:

- Object detection, viewpoint classification and aspect layout estimation are jointly solved using a rigorous coherent formulation. Our method allows us to accurately estimate each aspect part's 3D location and orientation in the object reference system as well as reason about which aspect part is visible or occluded from the estimated viewpoint.
- The learnt aspect part templates are made view invariant by injecting a rectification process into inference.
- We significantly outperform state-of-the-art methods in estimating object pose using three public datasets as well as demonstrate the ability of accurately recovering the aspect layout of an object category from a single image.

The rest of the chapter is organized as follows: Section 2 reviews related works. Section 3 describes our aspect layout model including parameter estimation and model inference. Section 4 presents the experimental evaluation and Section 5 concludes the chapter.

2.2 Related Work

Part-based object representations have been widely used in computer vision (e.g., *Fergus et al.* (2003); *Felzenszwalb and Huttenlocher* (2005)). *Felzenszwalb et al.* (2010) utilize a part-based representation for general object detection and achieve remarkable detection results. Gu and Ren *Gu and Ren* (2010) extend *Felzenszwalb et al.* (2010) for viewpoint classification by discriminatively training mixture of templates of object viewpoints. However, both *Felzenszwalb et al.* (2010) and *Gu and Ren* (2010) only train independent models for a small number of discrete viewpoints, and the 3D spatial relationships between parts are not modeled.

Various approaches have been recently proposed that explicitly take into account the 3D nature of object categories *Thomas et al.* (2006); *Savarese and Fei-Fei* (2007); *Hoiem et al.* (2007a); *Chiu et al.* (2007); *Su et al.* (2009); *Arie-Nachimson and Basri* (2009); *Farhadi et al.* (2009b); *Liebelt and Schmid* (2010); *Stark et al.* (2010); *Lopez-Sastre et al.* (2011); *Payet and Todorovic* (2011); *Glasner et al.* (2011). These methods can be roughly classified into two main categories. Methods in the first category represent object as collections of parts or features which are connected across views *Thomas et al.* (2006); *Savarese and Fei-Fei* (2007); *Su et al.* (2009); *Arie-Nachimson and Basri* (2009); *Farhadi et al.* (2009b); *Payet and Todorovic* (2011). Methods in the second category represent objects using an explicit 3D model on top of which features or parts are associated *Hoiem et al.* (2007a); *Chiu et al.* (2007); *Liebelt and Schmid* (2010); *Stark et al.* (2010); *Glasner et al.* (2011). *Hoiem et al.* (2007a) proposes a CRF built on top of a rough 3D object model. The approach can be used for both object detection and segmentation. Similar to our model, Chui et al. *Chiu et al.* (2007) propose a 3D object representation which consists of planar parts. However, *Chiu et al.* (2007) mostly uses such 3D representation to generate virtual training

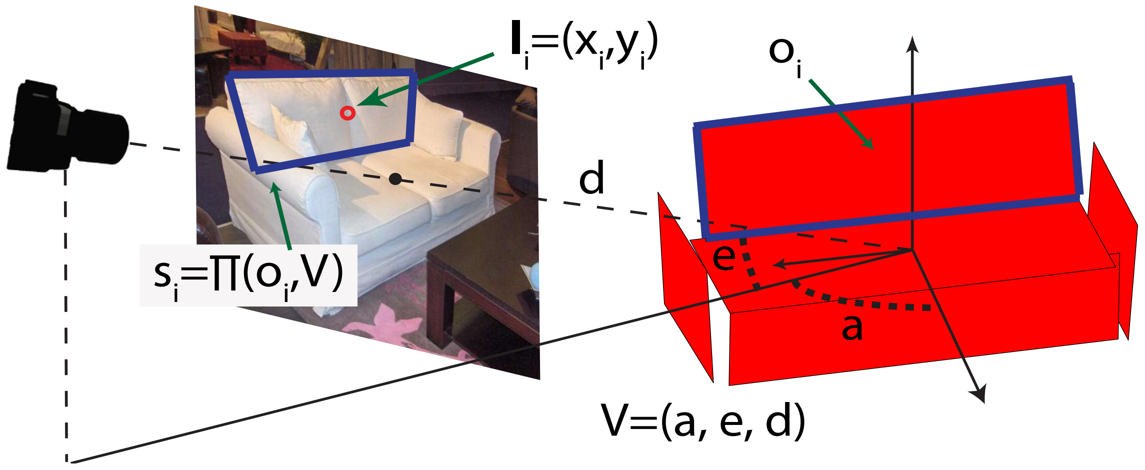


Figure 2.3: Illustration of viewpoint representation and part shape from 3D. The viewpoint V is represented by azimuth a, elevation e and distance d of the camera pose. 2D part shape s_i is determined by the viewpoint transformation $\Pi(o_i, V)$ with o_i be the ith 3D aspect part (*back* of the sofa in the figure). The part center location $\mathbf{l}_i$ is also shown.

examples. Unlike *Liebelt and Schmid* (2010); *Stark et al.* (2010), where 2D object detectors and 3D models are independent, our approach is based on the interaction between 3D object representation and 2D part detectors to guide the process of locating aspect parts and estimating object poses. Unlike *Farhadi et al.* (2009b), where object aspects are treated as latent variables, we relate our definition of aspect parts to 3D topological properties of the object category.

2.3 Aspect Layout Model

We propose a novel Aspect Layout Model (ALM) for estimating the 3D aspect layout of object categories. Suppose that each object in a category consists of n aspect parts. Let $O = (o_1, o_2, \ldots, o_n)$ denote the object in 3D, where $o_i, i = 1, \ldots, n$ represents the ith aspect part. Fig. 2.2 illustrates the training steps to construct the 3D object O from a set of 3D CAD models. Given an input image I, ALM predicts the object label $Y \in \{+1, -1\}$ indicating the presence or absence of an object instance in the image, and the part configuration $C = (\mathbf{c}_1, \dots, \mathbf{c}_n)$ if $Y = +1$. The state of part i is given by $\mathbf{c}_i = (x_i, y_i, s_i)$, x_i and y_i are the part center coordinates in the image coordinate system, and s_i represents the part shape in the image. Based on the observation that a 2D part shape is jointly determined by the 3D geometry of the part and the viewpoint, the part shape s_i is given by the viewpoint transformation $\Pi(o_i, V), i = 1, \dots, n$, where V denotes the viewpoint. Suppose that the 3D object is positioned at the world coordinate origin and the camera always looks at the origin without in-plane rotation. Then the viewpoint can be parameterized by $V = (a, e, d)$ with a, e, d being azimuth, elevation and distance of the camera pose. Fig. 2.3 illustrates the viewpoint representation and the 2D part shape generated by the viewpoint transformation. The posterior distribution of object label and part configuration can be written as

$$\begin{aligned} P(Y, C|I) &= P(Y, \mathbf{c}_1, \dots, \mathbf{c}_n|I) \\ &= P(Y, x_1, y_1, s_1, \dots, x_n, y_n, s_n|I) \\ &= P(Y, x_1, y_1, \dots, x_n, y_n, O, V|I) \\ &= P(Y, L, O, V|I), \end{aligned} \tag{2.1}$$

where $L = (\mathbf{l}_1, \dots, \mathbf{l}_n)$ and $\mathbf{l}_i = (x_i, y_i), i = 1, \dots, n$ denotes the 2D part center coordinates. In the third line of Eq. (2.1), we replace $s_i, i = 1, \dots, n$ with O and V, since the part shape s_i in the image is completely specified by the viewpoint transformation $\Pi(o_i, V)$. Then, the part configuration is given by L, O and V. Inference is achieved by maximizing the posterior distribution $P(Y, L, O, V|I)$.

2.3.1 Discriminative Modeling

We model ALM discriminatively using a Conditional Random Field (CRF) *Lafferty et al.* (2001) formulation. The posterior distribution of object label and part

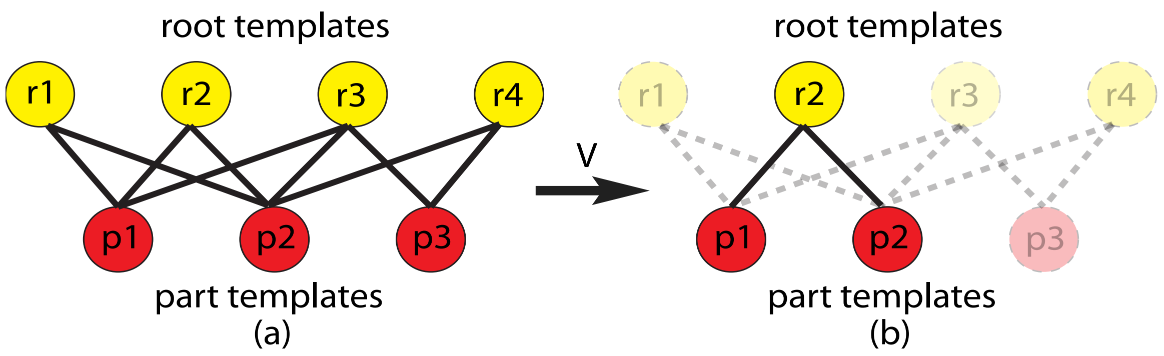


Figure 2.4: (a) An example of the bipartite graph structure in our model. A root template is connected to all the visible part templates in its view section. (b) Under a specific viewpoint V, the graph reduces to a tree with the root template as the root node and all the visible part templates under the viewpoint as its children.

configuration is

$$P(Y, L, O, V|I) \propto \exp\big(E(Y, L, O, V, I)\big), \tag{2.2}$$

where $E(Y, L, O, V, I)$ is the energy function. By imposing a graph structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ over parts as described below, the energy function can be decomposed as

$$E(Y, L, O, V, I) = \begin{cases} \sum_{i \in \mathcal{V}} V_1(\mathbf{l}_i, O, V, I) + \sum_{(i,j) \in \mathcal{E}} V_2(\mathbf{l}_i, \mathbf{l}_j, O, V), \text{ if } Y = +1 \\ 0, \text{ if } Y = -1, \end{cases} \tag{2.3}$$

where V_1 and V_2 are the unary potential and pairwise potential respectively. The unary potential captures the visual appearances of parts, while the pairwise potential encodes the spatial relationships between parts. The energy of a negative sample is set to zero.

Graph Structure. In our model, the unary potential is designed as a 2D part template. We use one part template for each aspect part in 3D. Moreover, we introduce root templates which are associated with the whole object from different viewpoints. Specifically, we divide the viewing sphere into a fixed number of view sections (e.g., 8 view sections with each covering 45° azimuth). For each view section, we add one 2D root template into ALM. The root template is activated if the object is viewed inside its view section. All the other root templates are considered to be occluded. Then we impose a bipartite graph structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ between the root templates and the part templates. A root template is connected to all the visible part templates in its view section, but there is no link between two root templates or two part templates. An important property of the bipartite graph structure is that, under a specific viewpoint, the graph reduces to a tree formed by all the visible templates. So we can have a local tree structure for each viewpoint and solve the inference problem efficiently. Fig. 2.4 illustrates the graph structure in our model.

Viewpoint Invariant Unary Potential. The unary potential is modeled with a linear discriminative model as

$$V_1(\mathbf{l}_i, O, V, I) = \begin{cases} \mathbf{w}_i^T \phi(\mathbf{l}_i, O, V, I), \text{ if unoccluded} \\ \alpha_i, \text{ if occluded,} \end{cases} \tag{2.4}$$

where $\mathbf{w}_i$ is the weight of the linear model, α_i is the weight for part i if it is occluded under viewpoint V, and $\phi(\mathbf{l}_i, O, V, I)$ represents the feature vector which consists of HOG features *Dalal and Triggs* (2005) in our implementation. Unlike previous methods *Felzenszwalb et al.* (2010); *Gu and Ren* (2010) which train multiple independent object templates for different viewpoints, ALM only trains one template for each part across all viewpoints. Similar to *Savarese and Fei-Fei* (2007), the template corresponds to the frontal view of the part. This is achieved by rectifying the part appearance using an homographic transformation H that transforms a part to its frontal view, where H can be obtained from the 3D model given V. Then HOG features are extracted from the rectified part. A reliable rectification process is also proposed in *Hedau et al.* (2010). Consequently, ALM is able to estimate fine-grained viewpoints and capture the relationships between viewpoints in a compact form. Fig.

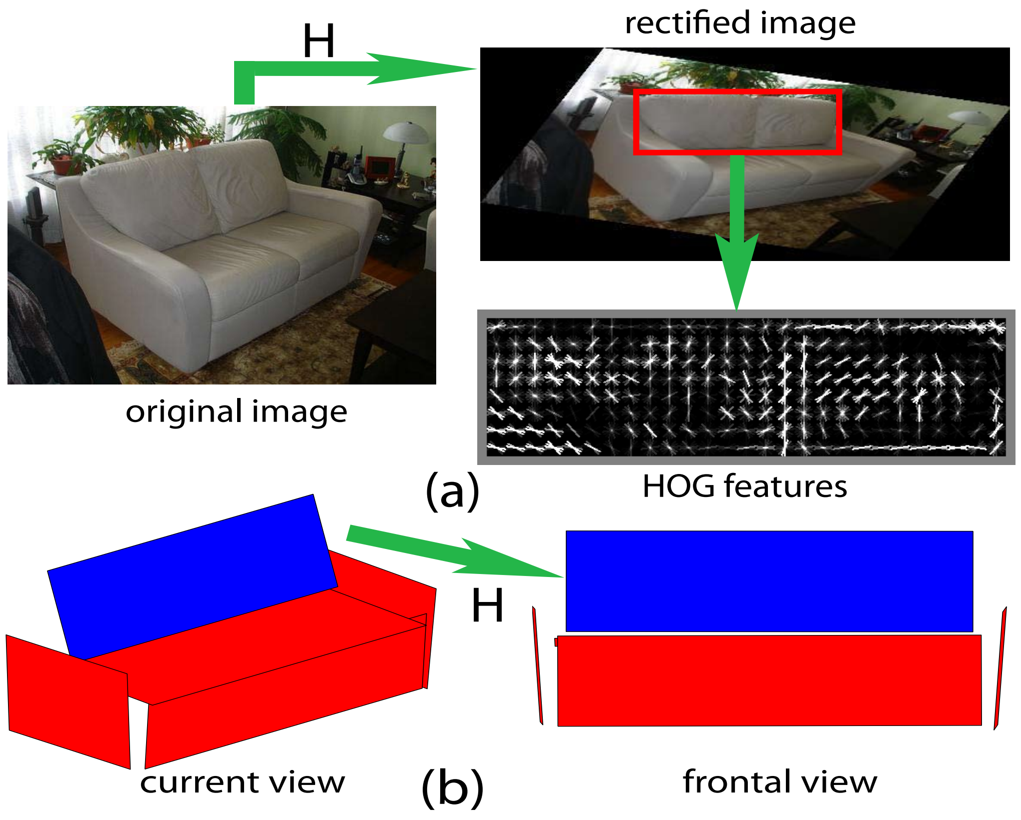


Figure 2.5: Illustration of rectified HOG features for the *back* of the *sofa* object category. (a) The original image is rectified to the frontal view of the aspect part *back* of the sofa using the homographic transformation H. Rectified HOG features for *back* are extracted from the red bounding box which delimits the transformed image of the *back* part to its frontal view. (b) The homographic transformation H between *back*'s current view and its frontal view is used for rectification.

2.5 illustrates an example of rectified HOG features.

Pairwise Potential. The pairwise potential captures the relationship between relative part locations and orientations in the image. In the ideal case, the relative locations given by projecting the 3D object O onto the image according to the viewpoint V and the corresponding observed relative locations should be equal. We design the pairwise potential so as to penalize deviation of the observed relative part locations from the ideal ones. Let (x'_i, y'_i) and (x'_j, y'_j) be the positions of the joints between part i and part j in the image coordinates (see *Felzenszwalb and Huttenlocher* (2005) for the definition of joint), $d_{ij,O,V}$ be the learnt distance between part i and part j given by projecting the 3D object O according to the viewpoint V to the image and $\theta_{ij,O,V}$ be the learnt relative orientation between part i and part j. Then the joint coordinates are given by

$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \frac{1}{2} d_{ij,O,V} \begin{bmatrix} cos(\theta_{ij,O,V}) \\ sin(\theta_{ij,O,V}) \end{bmatrix}, \tag{2.5}$$

$$\begin{bmatrix} x'_j \\ y'_j \end{bmatrix} = \begin{bmatrix} x_j \\ y_j \end{bmatrix} + \frac{1}{2} d_{ji,O,V} \begin{bmatrix} cos(\theta_{ji,O,V}) \\ sin(\theta_{ji,O,V}) \end{bmatrix}, \tag{2.6}$$

where $d_{ij,O,V}$, $d_{ji,O,V}$, $\theta_{ij,O,V}$, and $\theta_{ji,O,V}$ are computed from the 3D model. The pairwise potential is the negative squared distance between the two joints. Since $d_{ij,O,V} = d_{ji,O,V}$ and $\theta_{ij,O,V} = \theta_{ji,O,V} + \pi$, we have the following pairwise potential

$$V_2(\mathbf{l}_i, \mathbf{l}_j, O, V) = -w_x\big(x_i - x_j + d_{ij,O,V} cos(\theta_{ij,O,V})\big)^2 - w_y\big(y_i - y_j + d_{ij,O,V} sin(\theta_{ij,O,V})\big)^2, \tag{2.7}$$

where w_x and w_y are the parameters controlling the strength of the pairwise constraints.

Energy Function. Since both the unary and pairwise potentials are linear with respect to its own parameters, we can aggregate all the model parameters into one parameter vector $\theta = (\mathbf{w}_{i,\forall i}, \alpha_{i,\forall i}, w_x, w_y)$, and aggregate all the corresponding energy components into one feature vector $\Psi(Y, L, O, V, I)$. Then the energy function is

$$E(Y, L, O, V, I|\theta) = \theta^T \Psi(Y, L, O, V, I). \tag{2.8}$$

2.3.2 Maximal Margin Parameter Estimation

The most widely used technique for parameter estimation in CRFs is maximum likelihood, which requires proper normalization of the probabilities. However, normalization is not necessary in discriminative modeling. Consider the following inference problem:

$$(Y^*, L^*, O^*, V^*) = \arg \max_{Y,L,O,V} E(Y, L, O, V, I|\theta). \tag{2.9}$$

We note that only the "relative energy" values matter. By relative energy we refer to the difference between two energy values as opposed to the energy values themselves. From the point of view of energy based learning *LeCun et al.* (2006), the aim of parameter estimation in our model is to find an energy function which outputs the maximal energy value for the correct label configuration of an object in the image.

To train the model, we are given a set of training samples $\mathcal{T} = \{(I^t, Y^t, L^t, O^t, V^t), t = 1, \ldots, N\}$, where each sample is an image with the object label, 2D part center locations, learnt 3D model and viewpoint. Then a loss function is defined to evaluate the quality of a specific energy function. Finally, the parameters are estimated by minimizing the loss on the training set $\mathcal{T}$. If hinge loss is used in combination with a quadratic regularizer, the parameter estimation problem is equivalent to the following structural SVM optimization problem *Tsochantaridis et al.* (2004):

$$\min_\theta \frac{1}{2}\|\theta\|^2 + \lambda \sum_{t=1}^{N} \Big[\max_{Y,L,O,V} \big[\theta^T \Psi_{t,Y,L,O,V} + \Delta_{t,Y,L,O,V}\big] - \theta^T \Psi_{t,Y^t,L^t,O^t,V^t} \Big], \tag{2.10}$$

where λ is a fixed penalty parameter, $\Psi_{t,Y,L,O,V} = \Psi(Y, L, O, V, I^t)$, $\Psi_{t,Y^t,L^t,O^t,V^t} = \Psi(Y^t, L^t, O^t, V^t, I^t)$ and $\Delta_{t,Y,L,O,V} = \Delta(Y, L, O, V, Y^t, L^t, O^t, V^t)$ is the loss function measuring the difference between two sets of labels. We use the weighted 0-1 loss, i.e., $\Delta_{t,Y,L,O,V} = \beta \mathbf{I}(Y \neq Y^t)$, where β is a predefined constant and $\mathbf{I}$ is the indicator function. The above optimization problem can be solved efficiently using the cutting plane training method *Joachims et al.* (2009). We choose λ and β using a validation procedure.

2.3.3 Model Inference

Model inference aims to predict the object label and part configuration of an object. The inference problem is already given by Eq. (2.9). Viewpoints are discretized by sampling the viewing space defined by the azimuth, elevation and distance of the camera pose. Inference is then performed independently for different combinations of O and V.

Given O and V, Belief Propagation (BP) *Yedidia et al.* (2003) can be utilized to infer the 2D part center locations when $Y = +1$. Since the bipartite graph $\mathcal{G}$ reduces to a tree under a specific view, the inference for part location is optimal. BP works in a message passing fashion. The message that part i sends to its parent j is defined as

$$m_{ij}(\mathbf{l}_j) = \max_{\mathbf{l}_i} \Big(V_1(\mathbf{l}_i) + V_2(\mathbf{l}_i, \mathbf{l}_j) + \sum_{k \in \text{kids}(i)} m_{ki}(\mathbf{l}_i) \Big), \tag{2.11}$$

where V_1 and V_2 are the unary potential and pairwise potential respectively, and kids(i) denotes the children of part i. Messages are passed in the direction from the leaves to the root. Thus, we can obtain the belief vector at the root

$$b_i(\mathbf{l}_i) = V_1(\mathbf{l}_i) + \sum_{j \in \text{kids}(i)} m_{ji}(\mathbf{l}_i). \tag{2.12}$$

The location which maximizes the above belief is the optimal location for the root.

By keeping track of the argmax indices in Eq. (2.11), we can backtrace to find all the optimal locations of the other parts. After performing BP for all the combinations of O and V, we can obtain the energy value $E(Y = +1, L^*, O^*, V^*)$. The object label $Y^* = +1$ if and only if $E(Y = +1, L^*, O^*, V^*) > \gamma$, where γ is the detection threshold. To generate multiple detections in image I, we can threshold the belief at the root (Eq. (2.12)) and apply non-maximum suppression.

2.4 Experiments

2.4.1 Datasets

We evaluate our method for object aspect layout estimation on three public datasets: the 3DObject dataset *Savarese and Fei-Fei* (2007), the VOC2006 Car dataset *Everingham et al.* (c) and the EPFL Car dataset *Ozuysal et al.* (2009), and a new challenging dataset we extracted from ImageNet *Deng et al.* (2009). The 3DObject dataset is a standard benchmark for object pose estimation. It consists of 10 categories, each containing 10 different object instances observed from different viewpoints. We exclude the Head and the Monitor categories as they are not evaluated in previous work. The VOC2006 Car dataset consists of 921 car instances with viewpoint labels (Frontal, Rear, Left and Right). The EPFL Car dataset consists of 2,299 images of 20 car instances covering $360°$ azimuth in $3° - 4°$ steps with nearly the same elevation and distance. The new ImageNet dataset consists of four categories: Bed (400 images), Chair (770 images), Sofa (800 images) and Table (670 images). We manually annotated each object in the four datasets with azimuth, elevation, distance and part center locations following the structure of our 3D models unless the annotations were already available.

For each category in the 3DObject dataset, we use 5 instances for training and the other 5 instances for testing. Negative samples are randomly selected from the

VOC2007 dataset *Everingham et al.* (a). For the VOC2006 Car dataset, we train on the training and validation sets and test on the test set. For the EPFL Car dataset, we use the same training and testing partition as in *Ozuysal et al.* (2009). For each category in the ImageNet dataset, we use 50% images for training and test on the other 50% images, where we randomly separate the set of images under the same viewpoint into training images and test images.

2.4.2 Evaluation Measures

Object aspect layout estimation involves object detection, viewpoint estimation and part localization. We use Average Precision (AP) to measure the detection performance. The standard 50% bounding box overlap criteria of PASCAL VOC *Everingham et al.* (c) is used. For viewpoint estimation, we use the average viewpoint accuracy as performance measure, which is the average of the elements on the main diagonal of the viewpoint confusion matrix. As in all previous work, the viewpoint accuracy is computed among the true positives. To see how the viewpoint estimation is related to detection, we report the viewpoint accuracy as a function of the recall. For part localization, we use the Percentage of Correct Parts (PCP) in true positives as the evaluation measure. A predicted part is considered to be correct if the overlap between the predicted part and ground truth part is larger than 50%. Because part localization is evaluated only when the object is correctly detected, we plot PCP as a function of the recall. Then the area under the PCP-Recall curve is used as the quantitative measure for part localization. In the evaluation, we account for occlusion between parts, i.e., an occluded part that is predicted as being visible is considered to be incorrect.

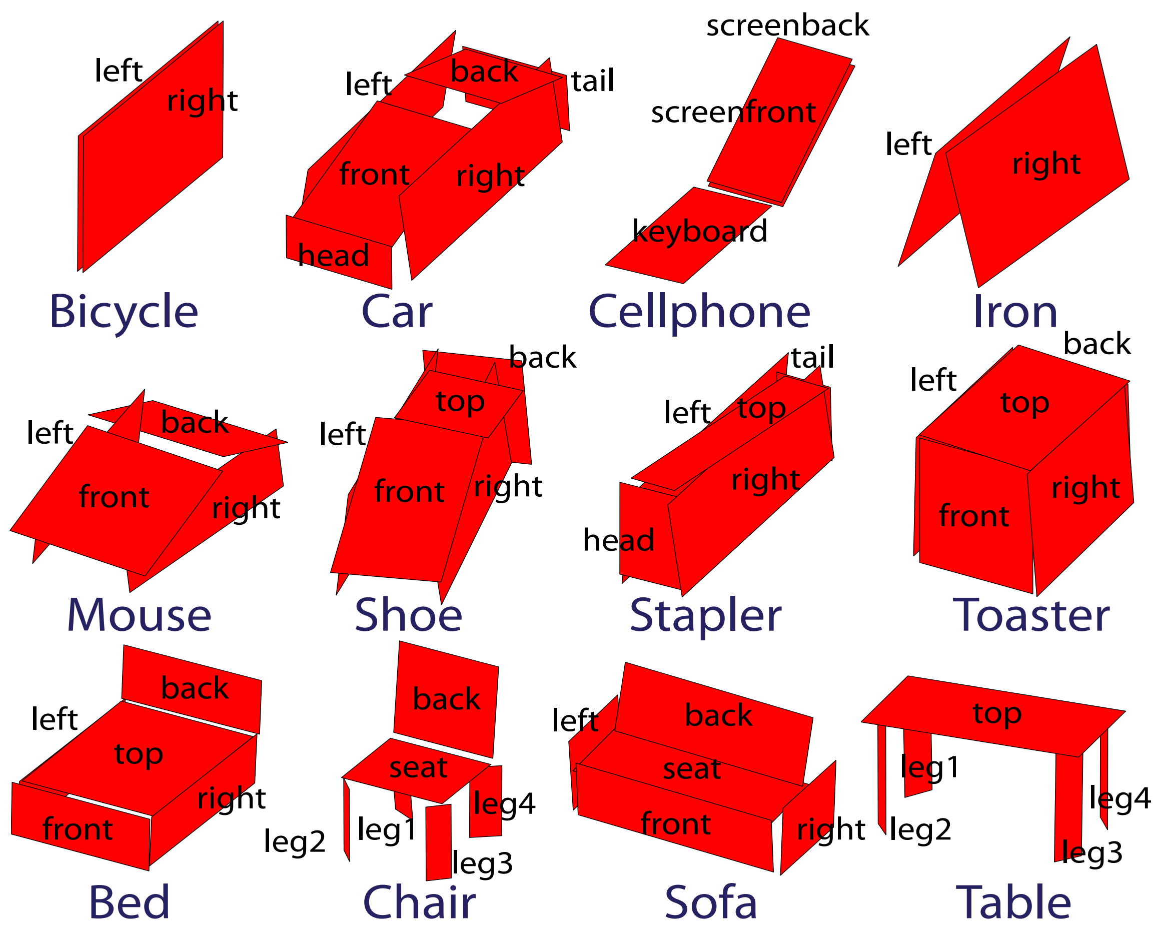


Figure 2.6: Our 3D object models for the 12 categories in our experiments. Each aspect part is associated to a part label.

Dataset	3DObject (8 views)		
Method	ALM	*Gu and Ren* (2010)	*Savarese and Fei-Fei* (2007)
Viewpoint	**80.7**	74.2	57.2
Detection	81.8	N/A	N/A

Table 2.1: Results on the 3DObject dataset.

Bicycle		
Method	Viewpoint	Detection
ALM	**91.4**	**93.0**
Payet and Todorovic (2011)	80.8	N/A
Liebelt and Schmid (2010)	75.0	69.8

Car		
Method	Viewpoint	Detection
ALM	**93.4**	98.4
Payet and Todorovic (2011)	85.4	N/A
Glasner et al. (2011)	85.3	**99.2**
Stark et al. (2010)	81.0	89.9
Liebelt and Schmid (2010)	70.0	76.7
Su et al. (2009)	67.0	55.3
Arie-Nachimson and Basri (2009)	48.5	N/A

Table 2.2: Results on the Bicycle and Car categories in the 3DObject dataset.

2.4.3 Results

3DObject Dataset. We first evaluate the performance of ALM for aspect layout estimation using portion of the 3DObject dataset. The first two rows of Fig. 2.6 show our 3D object models for the 8 categories of the 3DObject dataset. Table 2.1 shows the overall viewpoint estimation and detection results averaged on the 8 categories. Our model achieves 80.7% average viewpoint accuracy over 8 viewpoints, which is higher than 74.2% of the state-of-the-art *Gu and Ren* (2010). *Gu and Ren* (2010) and *Savarese and Fei-Fei* (2007) do not report the detection AP. Most of the previous works mainly conducted experiments on the Bicycle and Car categories. We also compare with the state-of-the-art methods on these two categories and present the results in Table 2.2. Our approach achieves the best performance.

More detailed viewpoint estimation results on the 3DObject dataset are presented in Table 2.3. In Fig. 2.7, we report the viewpoint accuracy as a function of the recall. Table 2.4 presents the detailed detection results on the 3DObject dataset. We compare our full model with our root model and the state-of-the-art object detector DPM *Felzenszwalb et al.* (2010), where the root model is trained only with root

Method	DPM *Felzenszwalb et al.* (2010)	ALM Root	ALM Full
Bicycle	88.4	**92.5**	91.4
Car	85.0	89.2	**93.4**
Cellphone	62.1	83.4	**85.0**
Iron	82.7	**86.0**	84.6
Mouse	40.0	58.7	**66.5**
Shoe	71.7	82.7	**87.0**
Stapler	58.5	69.2	**72.8**
Toaster	55.0	59.6	**65.2**
Mean	67.9	77.7	**80.7**

Table 2.3: Average viewpoint accuracy on the 3DObject dataset.

Category	DPM *Felzenszwalb et al.* (2010)	ALM Root	ALM Full
Bicycle	**95.1**	93.5	93.0
Car	98.2	**99.5**	98.4
Cellphone	73.1	77.4	**79.2**
Iron	**83.1**	75.8	80.7
Mouse	**64.0**	48.8	50.7
Shoe	**95.7**	85.6	84.2
Stapler	65.0	**73.4**	70.5
Toaster	96.7	96.5	**97.4**
Mean	**83.9**	81.3	81.8

Table 2.4: Average precision on the 3DObject dataset.

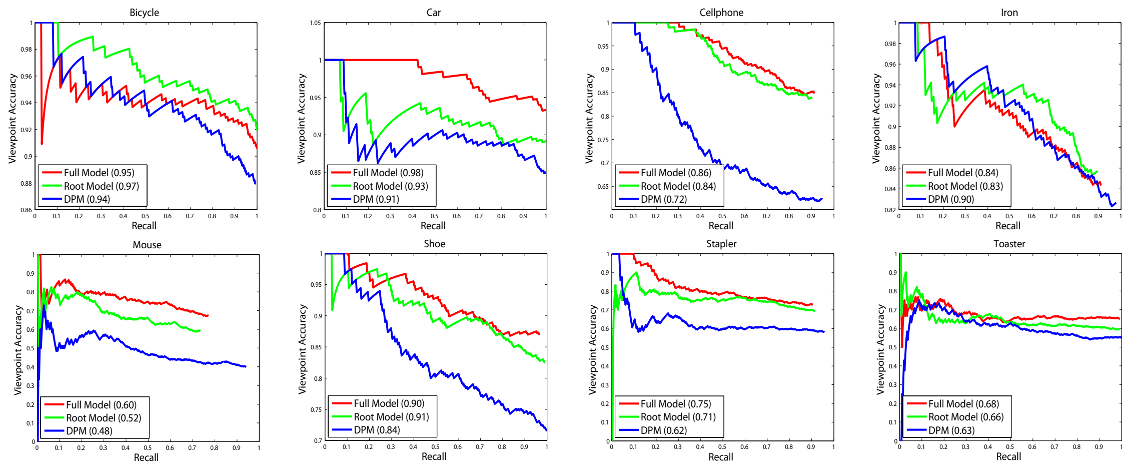


Figure 2.7: Viewpoint accuracy-recall curves for the eight categories in the 3DObject dataset.

Training Set Size	1	2	3	4	5
DPM *Felzenszwalb et al.* (2010)	69.2	81.9	84.5	84.6	85.0
ALM Root	**80.6**	**88.5**	90.5	91.7	89.2
ALM Full	76.3	85.1	**92.7**	**92.6**	**93.4**

Table 2.5: Average viewpoint accuracy on the 3DObject Car dataset with different training set sizes (number of instances).

templates. We train and test DPM with the same training and test sets as ALM. Eight root templates with parts are trained for DPM according to the 8 viewpoints. Our full model achieves the best viewpoint estimation among the three models. This demonstrates that adding part templates plays an important role in obtaining high performances. To see more clearly the benefit of employing the relationship between views, we compare the average viewpoint accuracy of our full model, our root model and DPM on the 3DObject Car dataset with different training set sizes. The results are given in Table 2.5, where the training set size is varied from 1 to 5 instances. The full model and the root model obtain better results than DPM in all the settings. By using more than 3 instances, the full model achieves better performances than the root model.

We evaluate the ability to localize aspect parts by using the PCP-Recall curves. Fig. 2.8 reports the PCP-recall curves of parts for the 8 categories. If the area under the curve is close to one, then we have good localization performance for the part (i.e., the *left* and *right* of car). Note that for the toaster category, we only use the *top* aspect part. Since the other parts have nearly no texture, we find that it is almost impossible to locate these parts in a reliable fashion. Some anecdotal aspect layout estimation results for the 8 categories are shown in Fig. 2.9. Notice that ALM is robust to intra-class variability and viewpoint change.

VOC2006 Car Dataset. We also conducted experiments on the VOC2006 Car dataset. The results for viewpoint estimation and object detection are showed in Table 2.6. We achieve nearly the same results as *Gu and Ren* (2010) and better results than

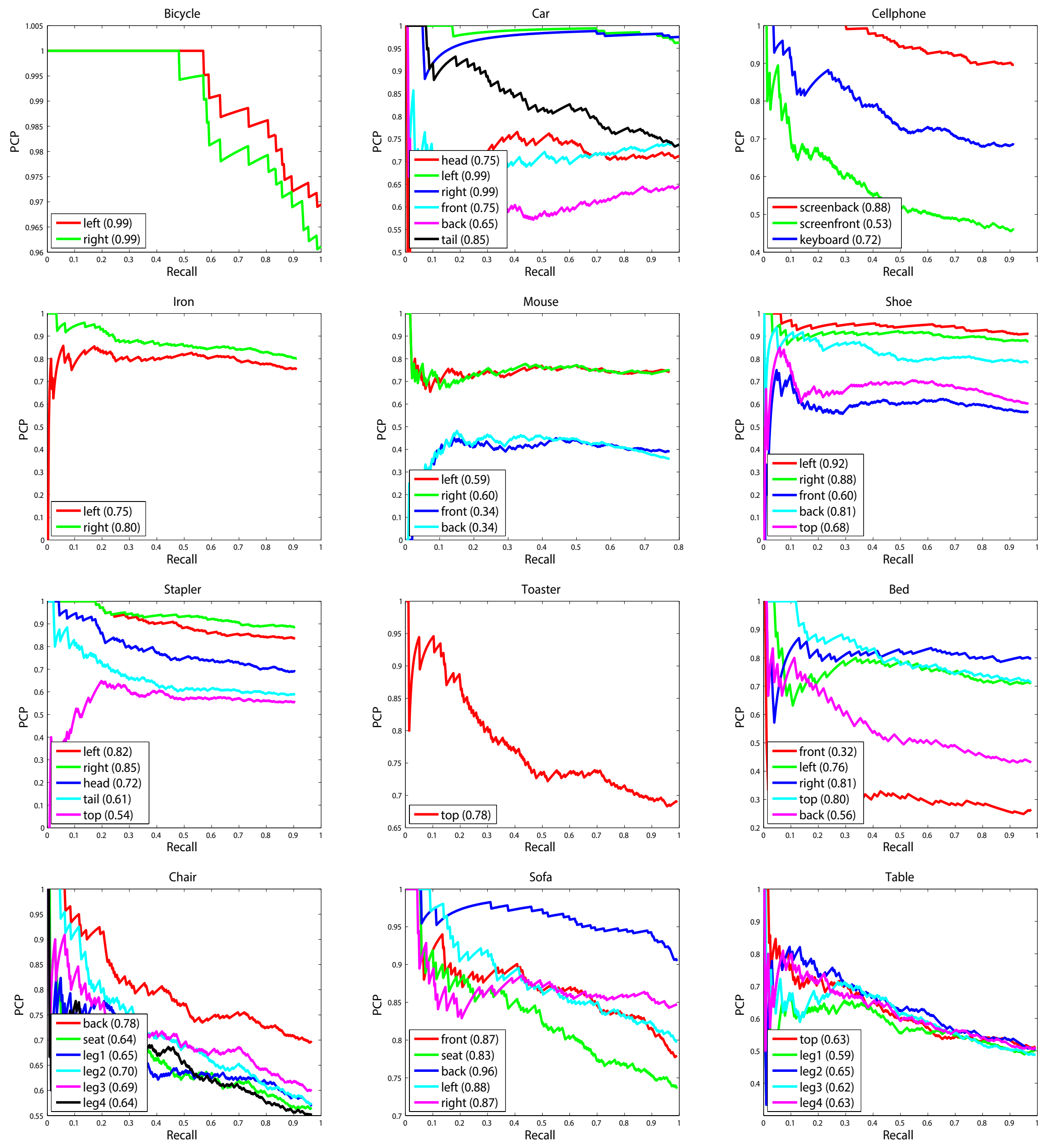


Figure 2.8: PCP-Recall curves for part localization on the 3DObject dataset (first two rows) and the ImageNet dataset (last row).

Dataset	VOC2006 Car (4 views)		
Method	ALM	*Gu and Ren* (2010)	*Su et al.* (2009)
Viewpoint	**85.9**	85.7	73.0
Detection	48.7	**51.0**	35.0

Table 2.6: Results on the VOC2006 Car dataset.

Method	Viewpoint	Detection
ALM Full	**64.8**	96.4
ALM Root	58.1	97.5
DPM *Felzenszwalb et al.* (2010)	56.6	**98.1**
Ozuysal et al. (2009)	41.6	85.4

Table 2.7: Results on the EPFL Car dataset (16 views).

Category	Bed	Chair	Sofa	Table	*Mean*
3 views					
DPM *Felzenszwalb et al.* (2010)	84.1	**88.6**	90.1	75.6	84.6
ALM Root	84.7	60.2	91.0	**80.0**	79.0
ALM Full	**90.0**	87.7	**92.4**	76.0	**86.5**
7 views					
DPM *Felzenszwalb et al.* (2010)	56.2	41.2	44.0	**56.4**	49.5
ALM Root	37.5	23.4	39.6	35.4	34.0
ALM Full	**62.7**	**73.1**	**65.0**	52.6	**63.4**

Table 2.8: Average viewpoint accuracy on the ImageNet dataset.

Su et al. (2009). Our method is less effective if the viewpoint distribution in training and testing is too coarse. There are only 4-view labels in the VOC2006 Car dataset.

EPFL Car Dataset. In order to compare the performance of our algorithm with *Ozuysal et al.* (2009), we bin our viewpoint estimation into 16 bins (22.5° azimuth degree). DPM is trained with 16 templates according to the 16 views. The results on this dataset are presented in Table 2.7. Notice that as the number of viewpoints increases, the full model achieves significant improvement on viewpoint accuracy over the root model and DPM.

ImageNet Dataset. The last row of Fig. 2.6 shows our 3D models for the 4 categories in the dataset. Most of the objects in the dataset are viewed from their

Category	DPM *Felzenszwalb et al.* (2010)	ALM Root	ALM Full
Bed	**94.0**	83.5	89.4
Chair	**95.4**	78.4	89.3
Sofa	**97.6**	93.7	92.8
Table	**95.1**	81.2	90.1
Mean	**95.5**	84.2	90.4

Table 2.9: Average precision on the ImageNet dataset.

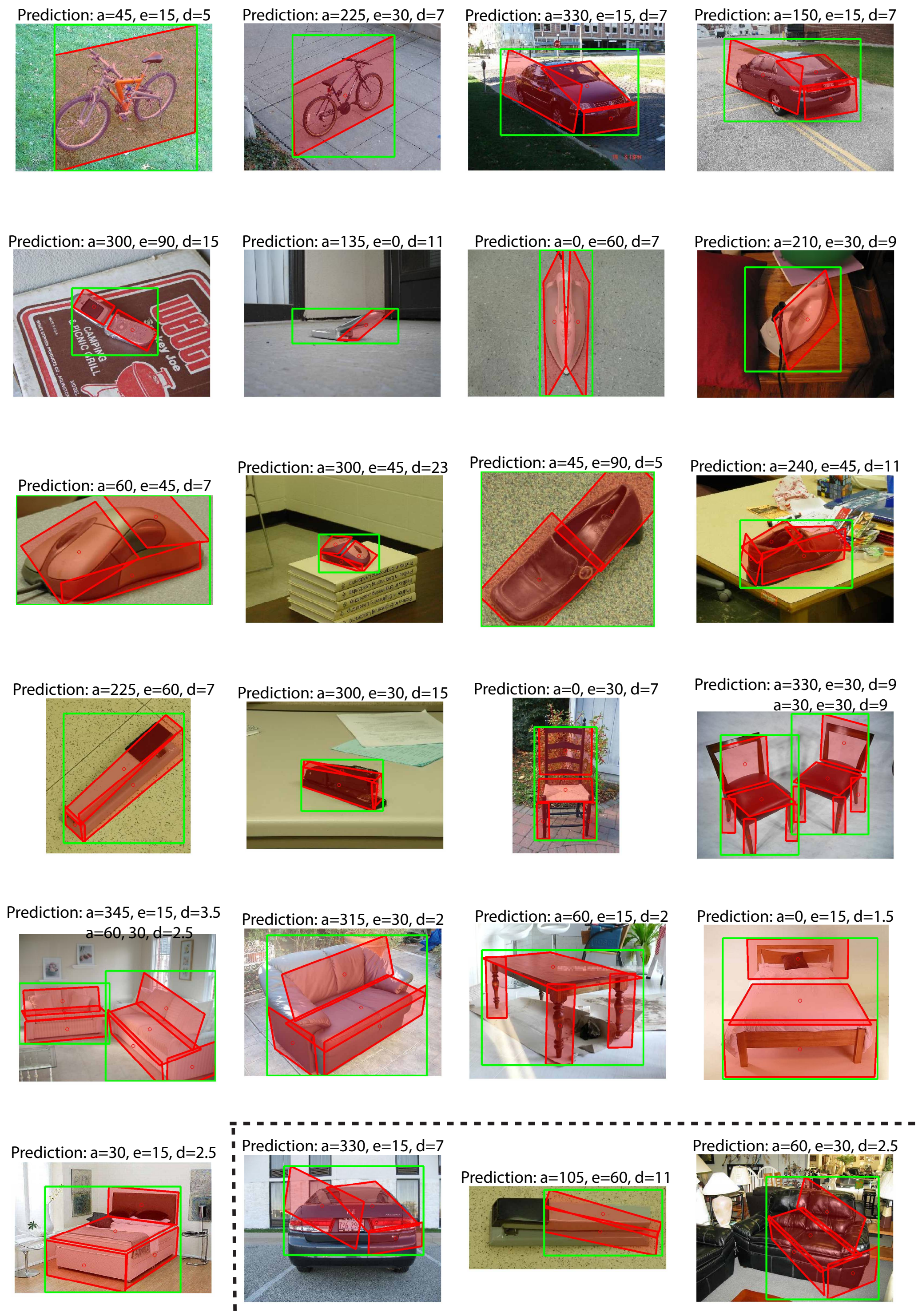

Prediction: a=45, e=15, d=5
Prediction: a=225, e=30, d=7
Prediction: a=330, e=15, d=7
Prediction: a=150, e=15, d=7
Prediction: a=300, e=90, d=15
Prediction: a=135, e=0, d=11
Prediction: a=0, e=60, d=7
Prediction: a=210, e=30, d=9
Prediction: a=60, e=45, d=7
Prediction: a=300, e=45, d=23
Prediction: a=45, e=90, d=5
Prediction: a=240, e=45, d=11
Prediction: a=225, e=60, d=7
Prediction: a=300, e=30, d=15
Prediction: a=0, e=30, d=7
Prediction: a=330, e=30, d=9
a=30, e=30, d=9
Prediction: a=345, e=15, d=3.5
a=60, 30, d=2.5
Prediction: a=315, e=30, d=2
Prediction: a=60, e=15, d=2
Prediction: a=0, e=15, d=1.5
Prediction: a=30, e=15, d=2.5
Prediction: a=330, e=15, d=7
Prediction: a=105, e=60, d=11
Prediction: a=60, e=30, d=2.5

Figure 2.9: Anecdotal aspect layout estimation results on the 3DObject dataset and the ImageNet dataset. The last row shows some wrong estimations. (Please zoom in to see details.)

front. So we evaluate the viewpoint estimation on 3 views (front, front-left, front-right) as well as 7 views (azimuth 0°, 15°, 30°, 45°, 315°, 330° and 345°) respectively. The results are shown in Table 2.8. Our full model achieves significant improvements on viewpoint estimation over the root model and DPM when 7 views are considered. The full model leverages the ability to handle few training samples by sharing part across views. Our full model achieves average detection AP 90.4% on the 4 categories, which is almost on par to 95.5% of DPM (See Table 2.9 for details). We show the PCP-Recall curves for part localization of the 4 categories in the last row of Fig. 2.8. Anecdotal aspect layout estimation results are shown in Fig. 2.9.

2.5 Conclusion

We have proposed a new model (called ALM) for jointly detecting objects as well as estimating their poses and the layout of their parts (aspect parts). ALM is capable of handling a large number of views, locating parts with approximately correct aspect orientations and reasoning about occlusions among parts. We have conducted extensive experiments to demonstrate the ability of our model to jointly solve these three tasks. We show high precision in detecting aspect parts using the 3DObject dataset *Savarese and Fei-Fei* (2007), the EPFL Car dataset *Ozuysal et al.* (2009) and a subset of the ImageNet dataset *Deng et al.* (2009). These results indicate that our method can be useful in problems where functional parts or affordances are to be estimated.

CHAPTER III

3D Aspectlet Representation

3.1 Introduction

The traditional object detection methods (e.g., *Viola and Jones* (2004), *Dalal and Triggs* (2005) and *Felzenszwalb et al.* (2010)) detect each object in an input image independently without considering the environment of the object. However, objects are not isolated in the real world. The contextual information around the objects plays an important role in object recognition *Oliva et al.* (2007). Recently, different types of contextual information have been utilized to help object detection, such as 3D scene geometry *Hoiem et al.* (2008) and 2D object co-occurrence *Desai et al.* (2011). Despite these efforts, the contextual cues that arise by considering object occlusions have not been fully explored yet. When objects occlude each other or are truncated by other scene elements, only limited portions of the objects are visible and some of the cues which we typically use to recognize the objects may not be available (e.g., the wheels of the blue car in Fig. 3.1(a)). In these cases, detecting each object independently is likely to fail (the detection score of the blue car in Fig. 3.1(a) would be low).

Detecting objects under occlusions is challenging due to various occlusion patterns in the image that can take place between objects. These occlusion patterns depend on the relative locations of objects in 3D with respect to the camera and also the shape

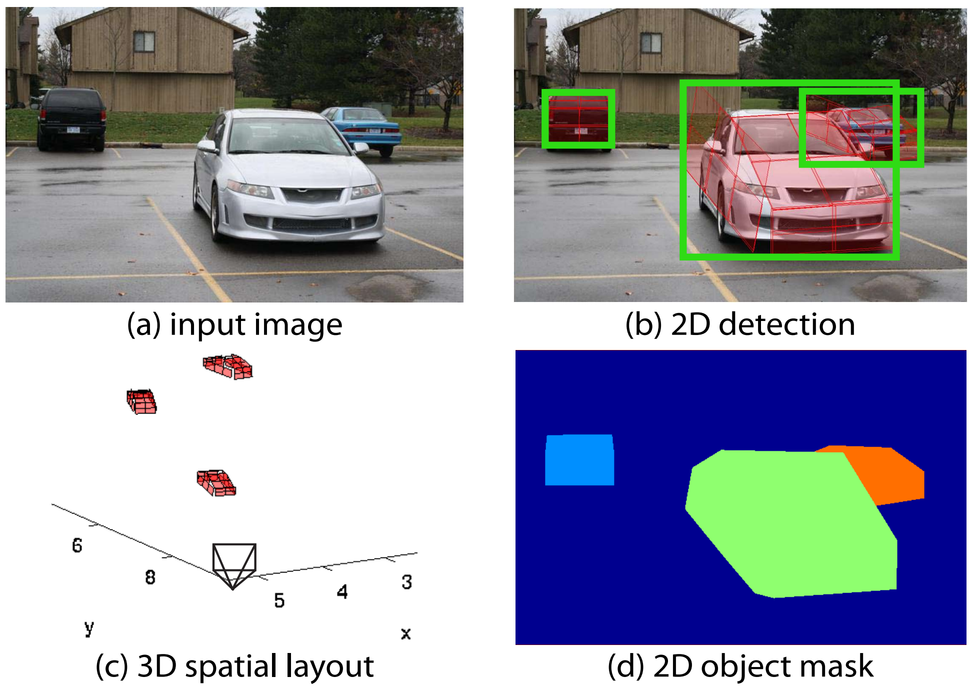


Figure 3.1: Illustration of our spatial layout model. Given an input image (a), our model detects the objects in the image (b), estimates their 3D spatial layout (c), and predicts the 2D object mask (d) which shows the occlusion order between objects.

and pose of the objects. Without considering these factors, methods which reason about occlusions based on 2D image features only, such as *Wang et al.* (2009) and *Gao et al.* (2011), are fragile to the uncertainty of the image evidence. In this paper, we handle occlusions in object detection from a 3D perspective. We design a novel framework that, from just one single image (Fig. 3.1(a)), is capable to jointly detect objects (Fig. 3.1(b)), determine their 3D spatial layout (Fig. 3.1(c)) and interpret which object occludes which (Fig. 3.1(d)). We call this model the Spatial Layout Model (SLM). First, inspired by the aspect part representation in *Xiang and Savarese* (2012), we propose a new 3D object representation using piecewise planar parts. These parts are fine-grained and suitable for occlusion reasoning in the sense that they can be approximated as either visible or non-visible. Second, inspired by the poselet framework for human detection *Bourdev and Malik* (2009), we group the planar parts in 3D to represent portions of the object. We call each group a "3D aspectlet", which is generated automatically. 3D aspectlets are able to provide more robust evidence of partial observations as opposed to the planar parts themselves. Finally, we generate hypotheses of the locations and poses of objects and camera in 3D (Fig. 3.1(c)), and then verify these hypotheses by combining prior knowledge and evidence from 3D aspectlets. This is achieved by a Markov Chain Monte Carlo (MCMC) sampling strategy, where different kinds of moves are designed to explore the hypothesis space efficiently. In this process, 3D aspectlets are weighted according to the occlusion patterns induced by the 3D hypotheses (Fig. 3.1(d)). Consequently, we combine the bottom-up evidence from 3D aspectlets and the top-down occlusion reasoning to help object detection. Experiments are conducted on two new challenging datasets, i.e., an outdoor-scene dataset with cars and an indoor-scene dataset with furniture, where multiple objects are observed under various degrees of occlusions. We demonstrate that our method is able to obtain competitive detection results even in the presence of severe occlusions. Besides, our method has the ability to estimate the spatial layouts of objects in 3D and predict the occlusion order between objects in images.

3.2 Related Work

Recently, the use of context for object detection has received increasing attention. Desai et al. *Desai et al.* (2011) formulate the multiple object detection as a structured labeling problem, where spatial interactions between objects in 2D are modeled. Hoiem et al. *Hoiem et al.* (2008) introduce the idea of using 3D scene geometry to help 2D object detection, where objects are supposed to be on the ground plane with certain heights. The ground plane constraint is generalized to supporting planes of objects by Bao et al. *Bao et al.* (2011). Richer geometrical and physical constraints are also explored by different works. Hedau et al. *Hedau et al.* (2010) detect indoor-scene objects by considering the room layout. Choi et al. *Choi et al.* (2013a) propose 3D Geometric Phases to capture the semantic and geometric relationships between co-occurring objects in 3D. In this work, we demonstrate that by modeling the spatial context of objects in 3D, we can successfully enhance object detection and reason about occlusions between objects.

Previous works that reason about occlusions have mostly focused on image segmentation *Winn and Shotton* (2006); *Hoiem et al.* (2007b), object tracking *Wojek et al.* (2011), single object instance recognition *Lowe* (1999) and category-level object detection *Wu and Nevatia* (2005); *Wang et al.* (2009); *Gao et al.* (2011); *Zia et al.* (2013); *Pepikj et al.* (2013). Methods for object detection have leveraged on 2D image features to predict whether an object is occluded or not, such as *Wang et al.* (2009) and *Gao et al.* (2011). Very few works have addressed the problem from a 3D perspective. Two exceptions are *Wu and Nevatia* (2005) and *Wojek et al.* (2011), which reason about occlusions between humans by generating hypotheses of humans in 3D and verifying these hypotheses using part-based human detectors. Different from these, we do not model occlusions with a simplified 2.5D structure of depth layers, but rather a true 3D representation to predict occlusion patterns. Recently, *Zia et al.* (2013) uses 2D masks to represent occlusion patterns, while *Pepikj et al.* (2013) learns the occlusion patterns from training data. In both methods, the occlusion patterns are view-specific, and only limited number of occlusion patterns can be modeled. Our method infers the occlusion patterns from the 3D spatial layout of objects, which is general to handle various occlusion patterns.

3.3 Spatial Layout Model

We propose a novel Spatial Layout Model (SLM) which is able to model the interactions between objects, 3D scene and camera viewpoint, especially the occlusions between objects. Given an input image I, SLM predicts a set of objects $\mathbf{O} = \{O_1, \ldots, O_M\}$ in the 3D world, their projections in the image plane $\mathbf{o} = \{o_1, \ldots, o_M\}$ and the camera C, where M is the number of objects in the scene. SLM models the posterior probability distribution of 2D projections $\mathbf{o}$, 3D objects $\mathbf{O}$ and camera C as

$$\begin{aligned} P(\mathbf{o}, \mathbf{O}, C|I) &= P(C)P(\mathbf{O})P(\mathbf{o}|\mathbf{O}, C, I) \\ &\propto P(C)P(\mathbf{O}) \prod_{i=1}^{M} P(o_i|\mathbf{O}, C, I) \prod_{(i,j)} P(o_i, o_j|\mathbf{O}, C, I), \end{aligned} \tag{3.1}$$

where $P(C)$ and $P(\mathbf{O})$ are the prior distributions over camera and 3D objects respectively, $P(o_i|\mathbf{O}, C, I)$ is the unary likelihood of 2D projection o_i given all the 3D objects, the camera and the image, and $P(o_i, o_j|\mathbf{O}, C, I)$ is the pairwise likelihood of a pair of 2D projections. Note that each 2D projection o_i depends on the configuration of all the 3D objects $\mathbf{O}$. This is because occlusions between objects in 3D affect the appearances of projections in 2D. SLM explicitly models the occlusions between objects.

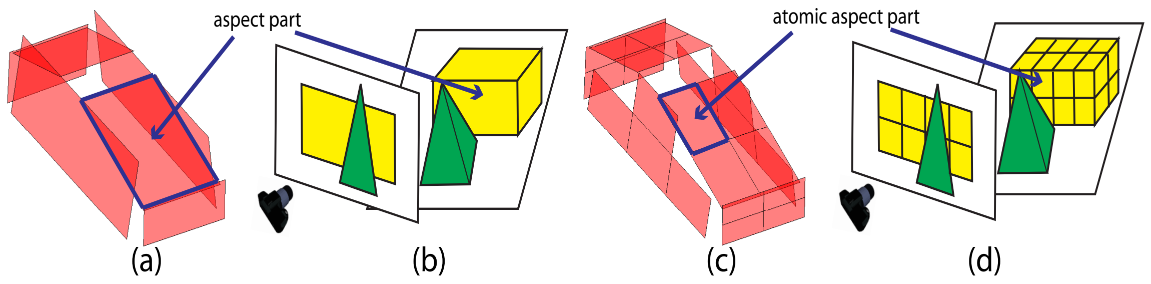


Figure 3.2: (a) Aspect part representation of car in *Xiang and Savarese* (2012) (b) A toy example shows that an AP is partially visible due to occlusion. (c) AAP representation of car in our model. (d) A toy example shows that an AAP can be approximated as either visible or non-visible.

3.3.1 3D Object Representation

We represent the 3D objects inspired by the piecewise planar representation introduced in the Aspect Layout Model (ALM) *Xiang and Savarese* (2012). In ALM, a 3D object consists of a set of Aspect Parts (APs). An aspect part is defined as "a portion of the object whose entire 3D surface is approximately either entirely visible from the observer or entirely non-visible" (Fig. 3.2(a)). While this definition is suitable for modeling object self-occlusions (akin to those used in aspect graph representations), they are not flexible enough to handling occlusions caused by other objects in the scene (as we seek to do). For instance, it is very unlikely that an AP is entirely occluded by another object - most likely just a portion of it is occluded (Fig. 3.2(b)). So we propose to represent a 3D object as a collection of Atomic Aspect Parts (AAPs) which are obtained by decomposing the original APs into smaller planar parts (Fig. 3.2(c)). Each AAP is approximated to be either visible or non-visible (Fig. 3.2(d)). This approximation is less coarse if AAPs are used as opposed to APs. As we can see, smaller AAPs are better for modeling occlusions. However, smaller AAPs are harder to detect due to the lack of visual features. So there is a trade-off between the ability of AAPs to model occlusions and the reliability to detect them in the image.

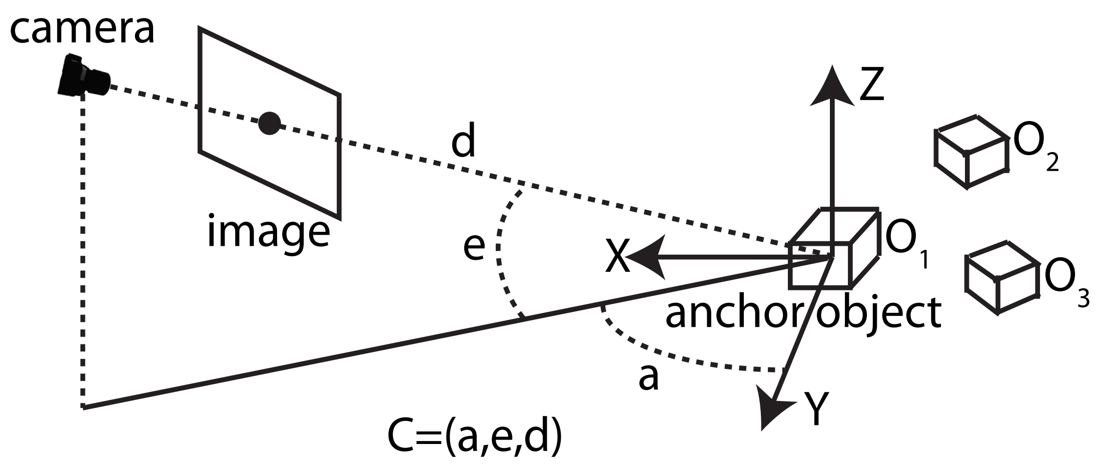


Figure 3.3: Camera and world coordinate system in our model.

3.3.2 Camera Prior

In SLM, 3D objects are rendered using the same internal virtual camera calibration matrix. As a result, the unknown camera parameters are the external camera matrix with respect to the world coordinate system. To define the world coordinate system, we choose one 3D object in the scene as the "anchor object", and define the world coordinate origin as the center of the anchor object. The axes of the world coordinate system are aligned with the dominating directions of the anchor object. Then the camera location in the world coordinate system can be specified by its azimuth a, elevation e and distance d. By assuming the camera is always looking at the world coordinate origin, the unknown camera parameters to be estimated are the azimuth, elevation and distance of the camera pose, i.e., $C = (a, e, d)$. A 3D object O_i can be represented by its coordinates in the world coordinate system (X_i, Y_i, Z_i) and its relative orientation in the X-Y plane with respect to the anchor object Θ_i, i.e., $O_i = (X_i, Y_i, Z_i, \Theta_i)$. Fig. 3.3 illustrates the camera representation and the world coordinate system in our model. Note that different anchor objects result in different coordinates of the camera and the 3D objects. The locations of the 2D projections in the image, however, are not affected. So we can choose an arbitrary 3D object as

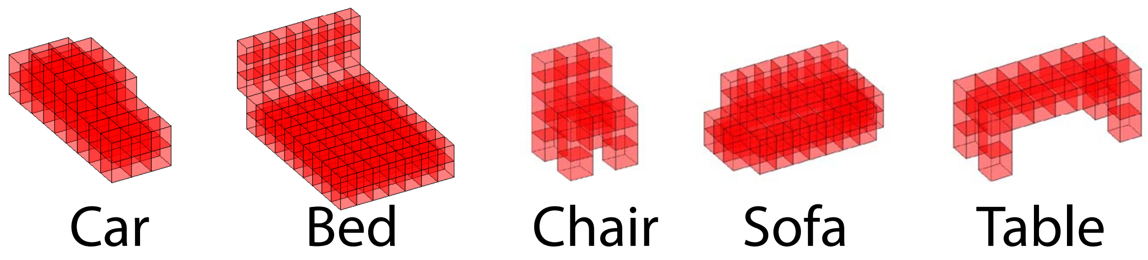


Figure 3.4: Voxel representations of the five categories in our experiments, which are used in computing the intersection and union of two volumes.

the anchor object. We define the camera prior as

$$P(C) = P(a)P(e)P(d), \tag{3.2}$$

where $P(a)$, $P(e)$ and $P(d)$ are the prior distributions for the azimuth, elevation and distance respectively. We assume uniform priors for the three variables:

$$a \sim \mathcal{U}(0, 2\pi), e \sim \mathcal{U}(0, \pi/2), d \sim \mathcal{U}(d_{\min}, d_{\max}), \tag{3.3}$$

where $d_{\min}$ and $d_{\max}$ are the minimum and maximum distances of the camera we considered in the model.

3.3.3 3D Objects Prior

We design the following prior to impose two constraints to a set of M objects in 3D: i) all the objects lie on the "ground plane"; ii) two objects can not occupy the same space in 3D. We model the prior distribution of 3D objects using a Markov Random Field (MRF):

$$P(\mathbf{O}) \propto \exp\Big(\sum_{i=1}^{M} V_1(O_i) + \sum_{(i,j)} V_2(O_i, O_j)\Big), \tag{3.4}$$

where V_1 and V_2 are the unary potential and pairwise potential respectively. Recall that the world coordinate system is defined on one of the 3D objects. If all the 3D objects lie on the "ground plane", their Z-coordinates should be close to zero (Fig. 3.3). By assuming a Gaussian distribution for the objects' Z-coordinates, we design the unary potential as

$$V_1(O_i) = -\frac{Z_i^2}{2\sigma^2}, \tag{3.5}$$

where σ is the standard deviation of the Gaussian distribution. Note that we do not estimate the real ground plane of the scene. The unary potential constrains that the 3D objects are all at similar heights. The pairwise potential penalizes overlapping between two 3D objects, which is defined as

$$V_2(O_i, O_j) = -\rho \frac{O_i \bigcap O_j}{O_i \bigcup O_j}, \tag{3.6}$$

where ρ is the parameter controlling the strength of the penalty, $\bigcap$ and $\bigcup$ denote the intersection and union between the volumes of two 3D objects. We represent the 3D objects using voxels, based on which we compute the intersection and union of two volumes (Fig. 3.4).

3.3.4 3D Aspectlets

In order to obtain evidence of partial observations of objects, we introduce the concept of "3D aspectlet" inspired by the poselet framework for human detection *Bourdev and Malik* (2009). A 3D aspectlet is defined as a portion of the 3D object, which consists of a set of the AAPs in our case. Not all the combinations of AAPs can form 3D aspectlets. We require the AAPs of a 3D aspectlet to have the two properties: i) they are geometrically close to each other in 3D; ii) there exists at least one viewpoint from which all the AAPs are visible, i.e., not self-occluded. If property ii) is not satisfied, we can represent the set of AAPs by smaller 3D aspectlets. To

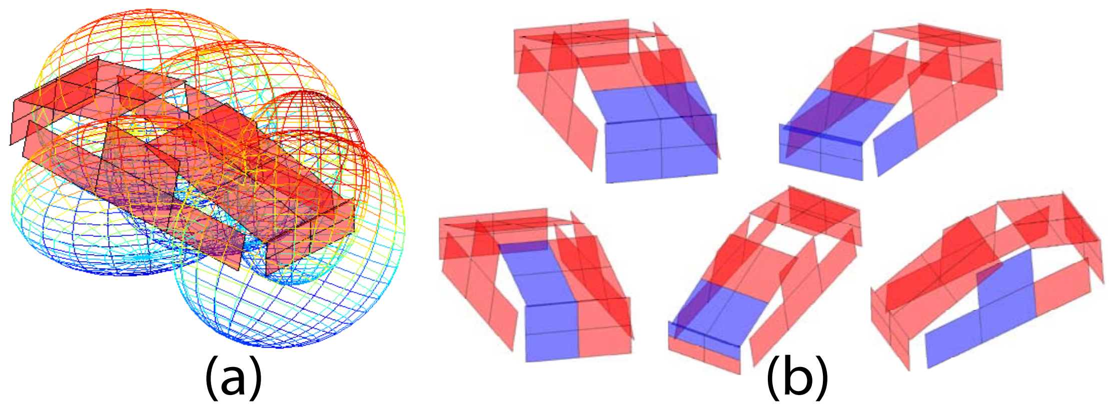


Figure 3.5: (a) Generating 3D aspectlet candidates by sampling ellipsoids in the space of the 3D object. (b) Examples of 3D aspectlets generated, where blue AAPs belong to the 3D aspectlets.

generate a 3D aspectlet with the two properties, we first randomly sample an ellipsoid in the 3D space of the 3D object (Fig. 3.5(a)), and select the AAPs inside the ellipsoid to form the 3D aspectlet. Then we check whether property ii) is satisfied. If not, we keep sampling ellipsoids until it is satisfied. Fig. 3.5(b) shows some 3D aspectlets of car generated in this way, where the blue AAPs belong to the 3D aspectlets.

To obtain evidence of objects from the image, we propose to represent the whole object and the 3D aspectlets an ensemble of tree models $\{\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_N\}$. Fig. 3.6 illustrates the graph structures of the trees. One of the tree models $\mathcal{T}_0$ represents the whole object, which is called the full-object model. The other N tree models $\{\mathcal{T}_1, \ldots, \mathcal{T}_N\}$ correspond to N 3D aspectlets, which represent portions of the object. The full-object model has a three-level tree structure which consists of the root level, the 3D aspectlet level and the AAP level. The root connects to all the 3D aspectlets in the mid-level, while a 3D aspectlet connects to all the AAPs it contains. By introducing 3D aspectlets as the mid-level, the full-object model is more robust to noises in the image. In theory, all the 3D aspectlets can be placed in the mid-level level. However, this would produce a complicated tree structure which makes the training and inference infeasible. Instead, 3D aspectlets which are not in the

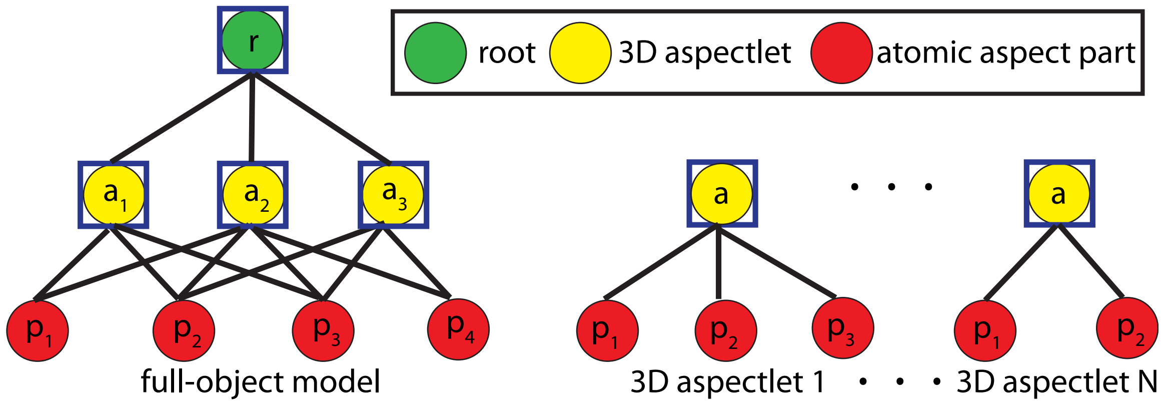


Figure 3.6: The graph structures of the full-object model and the 3D aspectlets, where the blue squares indicate that the bounded nodes can contain more than one template.

full-object model are represented by independent two-level tree structures. In our experiments, the 3D aspectlets in the full-object model correspond to the original APs in *Xiang and Savarese* (2012).

In the tree models, the AAPs are view-invariant, which means we only need to train one part template for each AAP regardless of the number of viewpoints. This is achieved by using rectified HOG features as in *Xiang and Savarese* (2012). But the root and the 3D aspectlets are viewpoint dependent. We train multiple templates for them, where each template captures the visual appearance of the object from a specific view section. For example, we train eight templates for the root with each template covering 45° azimuth. The number of templates for a 3D aspectlet depends on the range of its visible view section (i.e., not self-occluded). The blue squares in Fig. 3.6 indicate that there are multiple templates in these nodes. During inference, given a specific viewpoint hypothesis, only one template for each node is activated according to whether the given viewpoint hypothesis is inside its view section or not.

3.3.5 2D Projection Likelihood

The 2D projection likelihood measures the compatibility between the hypothesis of the locations and poses of objects and camera in 3D and the image evidence. Let the 2D projection o_i denote the 2D location of the ith object in the image plane, i.e., $o_i = (x_i, y_i)$. We model the unary 2D projection likelihood as

$$P(o_i|\mathbf{O}, C, I) \propto P_0(o_i|O_i, C, I) + \sum_{k=1}^{N} w_k(\mathbf{O}, C) P_k(o_i|O_i, C, I), \text{ s.t. } \sum_{k=1}^{N} w_k(\mathbf{O}, C) = 1, \tag{3.7}$$

where $P_0(o_i|O_i, C, I)$ is the likelihood of object O_i's 2D location from the full-object model, $P_k(o_i|O_i, C, I)$ is the likelihood of object O_i's 2D location from the kth 3D aspectlet, and $w_k(\mathbf{O}, C)$ is the weight of the kth 3D aspectlet. The weights measure the reliability of the 3D aspectlets, which relates to the visibility of the 3D aspectlets. Based on the observation that 3D aspectlets with more visible AAPs are more reliable, we set the weight of a 3D aspectlet proportional to the number of visible AAPs in it and constrain that all the weights sum to one. To test the visibility of AAPs, we project the 3D objects $\mathbf{O}$ to the image plane in the order of increasing distances of the objects from the camera. During the projection, the visibility test can be performed by checking whether the 2D regions of the AAPs are occupied by some frontal objects or not (refer to the 2D object mask in Fig. 3.1). Consequently, different occlusion patterns between objects result in different likelihoods. Note that in the unary 2D projection likelihood, the full-object model contributes equally with all the 3D aspectlets.

To define the likelihood of a 3D aspectlet for the object's 2D location, we perform a Hough transform from the 3D aspectlet's 2D location to the object's 2D location.

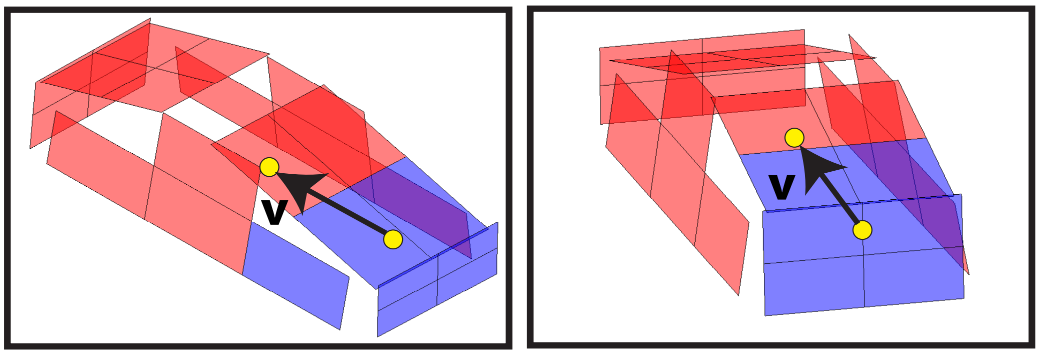


Figure 3.7: Illustration of the transform from the 3D aspectlet's 2D location to the object's 2D location, where the 2D projections of the two 3D aspectlets are shown in blue, and the yellow dots denote the 2D locations of the 3D aspectlets/objects in the projection.

Let $o_{ik} = (x_{ik}, y_{ik})$ be the 2D location of the kth 3D aspectlet. Then

$$P_k(o_i|O_i, C, I) = \sum_{o_{ik}} P(o_i|o_{ik}, O_i, C) P_k(o_{ik}|O_i, C, I), \tag{3.8}$$

where $P(o_i|o_{ik}, O_i, C)$ is the probability distribution of the object's 2D location conditioned on the 3D aspectlet's 2D location, the 3D geometry of the object and the camera viewpoint, and $P_k(o_{ik}|O_i, C, I)$ is the likelihood of the 3D aspectlet's 2D location. $P(o_i|o_{ik}, O_i, C)$ is defined as a delta function induced from the 3D-2D projection:

$$P(o_i|o_{ik}, O_i, C) = \begin{cases} 1, \text{ if } o_i = o_{ik} + \mathbf{v}_{ik}(O_i, C) \\ 0, \text{ otherwise,} \end{cases} \tag{3.9}$$

where $\mathbf{v}_{ik}(O_i, C)$ denotes the vector from the 3D aspectlet's 2D location to the object's 2D location in the projection of the 3D object O_i according to the camera C. Fig. 3.7 illustrates the transform. In practice, the equality test in Eq. (3.9) is performed by partitioning the image into grids and testing for inside the same grid.

The likelihood of the object's 2D location from the full-object model in Eq. (3.7) and the likelihoods of the 3D aspectlets' 2D locations in Eq. (3.8) are all modeled with the same type of Conditional Random Fields (CRFs) *Lafferty et al.* (2001) on their own tree structures (Fig. 3.6):

$$P_k(o_{ik}|O_i, C, I) \propto \exp\Big(\sum_{p\in\mathcal{T}_k} V_1(o_{ik}^p, O_i, C, I) + \sum_{(p,q)\in\mathcal{T}_k} V_2(o_{ik}^p, o_{ik}^q, O_i, C)\Big), k = 0, 1, \dots, N, \tag{3.10}$$

where p and q index nodes in the kth tree, (p, q) indicates an edge in the kth tree. $P_0(o_i|O_i, C, I) = P_0(o_{i0}|O_i, C, I)$, since there is no transform needed for the full-object model. $o_{ik}^p = (x_{ik}^p, y_{ik}^p)$ denotes the 2D location of the pth node, i.e., the 2D location of the root, the 3D aspectlet or the AAP depending on the type of the node. V_1 is the unary potential modeling 2D visual appearance and V_2 is the pairwise potential which constrains the 2D relative locations between two nodes. We utilize the unary and pairwise potentials used in *Xiang and Savarese* (2012). The unary potential is defined as

$$V_1(o_{ik}^p, O_i, C, I) = \begin{cases} \mathbf{w}_k^{pT}\phi(o_{ik}^p, O_i, C, I), \text{ if node } p \text{ visible} \\ \alpha_k^p, \text{ if node } p \text{ self-occluded}, \end{cases} \tag{3.11}$$

where $\mathbf{w}_k^p$ is the template for node p, $\phi(o_{ik}^p, O_i, C, I)$ is the rectified HOG features for the node extracted from the 2D image, and α_k^p is the weight for node p if it is self-occluded. The pairwise potential is defined as

$$\begin{aligned} & V_2(o_{ik}^p, o_{ik}^q, O_i, C) \\ & = -w_x\big(x_{ik}^p - x_{ik}^q + d_{ik}^{pq}(O_i, C)cos(\theta_{ik}^{pq}(O_i, C))\big)^2 \\ & - w_y\big(y_{ik}^p - y_{ik}^q + d_{ik}^{pq}(O_i, C)sin(\theta_{ik}^{pq}(O_i, C))\big)^2, \end{aligned} \tag{3.12}$$

where w_x and w_y are the parameters controlling the strength of the pairwise constraints, $d_{ik}^{pq}(O_i, C)$ is the computed distance between the two nodes after projecting the 3D object to the 2D image according to the camera, and $\theta_{ik}^{pq}(O_i, C)$ is the relative orientation between the two nodes computed from the 2D projection. Combining Eq. (3.7)-(3.12), we can obtain the form of the unary 2D projection likelihood.

For the pairwise 2D projection likelihood, it measures how likely the occlusion between a pair of objects induced from 3D is compatible with the 2D image evidence. We design the pairwise 2D projection likelihood to reflect the observation that the occluding object usually has higher unary 2D projection likelihood than the occluded object:

$$P(o_i, o_j|\mathbf{O}, C, I) \propto \exp\Big(- \frac{P(o_j|\mathbf{O}, C, I)}{P(o_i|\mathbf{O}, C, I)}\Big) \tag{3.13}$$

if O_i occludes O_j and $P(o_i|\mathbf{O}, C, I)$ is larger than some threshold to make sure O_i is a confident occluder. As a result, if the occluded object has higher unary 2D projection likelihood than the occluding object, the occlusion pattern is unlikely to be correct.

3.3.6 Training

Training aims at learning the CRFs of our 3D object detector, which is composed of two tasks: learning 3D aspectlets and estimating the model parameters. Since it is not feasible to use all the 3D aspectlets, we select the "good" 3D aspectlets automatically. We set up the following three criteria to measure the quality of a set of 3D aspectlets. i) *Discriminative power*: the selected 3D aspectlets are discriminatively powerful. To achieve this goal, we first sample a large number of 3D aspectlets according to the sampling process described in Sec. 3.3.4. Then we train and test the CRFs of the 3D aspectlets on the training dataset by cross-validation. The parameter estimation of the CRFs can be performed by the structural SVM optimization *Tsochantaridis et al.* (2004) in *Xiang and Savarese* (2012), while the inference is conducted by Belief Propagation on the tree structure of the 3D aspectlet. The discriminative power is measured by their detection performance, based on which we select the 3D aspectlets. ii) *Viewpoint coverage*: for a specific viewpoint, there are at least K 3D aspectlets visible. Because it would be difficult to detect an object under the viewpoint if too few 3D aspectlets are available. iii) *Atomic aspect part coverage*: an AAP is contained at least in one 3D aspectlet. Otherwise, the AAP is useless. According to the three criteria, we employ an greedy algorithm to select the 3D aspectlets. The algorithm starts with an empty set of 3D aspectlets. Then it keeps adding highly discriminative 3D aspectlets into the set until the viewpoint coverage and the atomic part coverage are satisfied.

3.3.7 Inference

The inference problem of our spatial layout model is to search for the most compatible configuration of 2D projections, 3D objects and camera given an input image:

$$(\mathbf{o}^*, \mathbf{O}^*, C^*) = \arg\max_{\mathbf{o}, \mathbf{O}, C} P(\mathbf{o}, \mathbf{O}, C|I), \tag{3.14}$$

where $P(\mathbf{o}, \mathbf{O}, C|I)$ is the posterior distribution defined in Eq. (3.1). Due to the complexity of the posterior distribution, we resort to Markov Chain Monte Carlo (MCMC) simulation to solve the inference problem. MCMC generates samples from the posterior distribution using a Markov chain mechanism. Then, the mode of the distribution is approximated by the sample with the largest probability among all the generated samples. As in *Desai et al.* (2011), we compute the log-odds ratio from the maximum a posteriori estimation as the 2D detection scores. Specifically, we exploit the reversible jump MCMC (RJMCMC) algorithm *Green* (1995). In RJMCMC, new samples are proposed by different moves from the proposal distributions. The proposed samples are either accepted or rejected according to the acceptance probabilities. The reversible moves enable the algorithm to explore spaces of different number of objects.

Initialization. We initialize the MCMC sampler with high confidence detections in the image, which are obtained by evaluating the unary 2D projection likelihood

(Eq. (3.7)) without considering occlusions between objects. The 3D objects and the camera are initialized by back-projecting the 2D detections into 3D according to the internal virtual camera calibration matrix and the estimated viewpoints of the 2D detections. A candidate set of objects is also obtained by evaluating the unary 2D projection likelihood without considering occlusions, which is used in the add moves and delete moves described below.

Add moves. Add moves add a new object O_{M+1} to the scene, where M is the current number of objects. An object in the candidate set which has not been associated with any object in the scene is randomly chosen to be added. The proposal distribution is proportional to the unary 2D projection likelihood. Since the add moves change the dimension of the state variables, specific consideration needs to be taken when computing the acceptance ratio. We map the low dimensional distribution into high dimension by assuming a constant probability $P(O_{M+1})$ for the new object:

$$\hat{P}(\mathbf{o}, \mathbf{O}, C|I) = P(\mathbf{o}, \mathbf{O}, C|I)P(O_{M+1}), \tag{3.15}$$

where $\hat{P}$ denotes the expanded posterior distribution. In this way, distributions of different dimensions can be compared.

Delete moves. Delete moves are the reverse moves of add moves, which remove one object from the scene and return it back to the candidate set. We adopt a uniform proposal distribution for delete moves. Similar to add moves, we map the low dimension distribution into high dimension by using a constant probability for the deleted object.

Switch moves. The switch moves change the anchor object in the scene, which prevents the model from local maximums if the anchor object is badly chosen. For example, if an object which is at different height with the other objects is selected to be the anchor object, then the other objects are unlikely to be added to the scene.

Category	Car	Bed	Chair	Sofa	Table
# objects	659	202	235	273	222
# occluded	235	81	112	175	61
# truncated	135	86	41	99	80

Table 3.1: Statistics of the objects in our new datasets.

The proposal distribution for switch moves is a uniform distribution.

3.4 Experiments

3.4.1 Datasets and Evaluation Measures

As far as we know, there is no dataset designed to test the ability to reason about occlusions in object detection. So we collected a new outdoor-scene dataset with 200 images of cars and a new indoor-scene dataset with 300 images of furniture for experiments, where objects are observed under various degrees of occlusion. These images are collected from PASCAL VOC *Everingham et al.* (b), LabelMe *Russell et al.* (2008), ImageNet *Deng et al.* (2009) and our own photos. Table 3.1 shows the statistics of the objects in the two datasets, from which we can see they include a large number of occluded and truncated objects. The new datasets are used for testing only. To learn the 3D aspectlets and train the CRFs, we utilize the 3DObject dataset in *Savarese and Fei-Fei* (2007) for car and the ImageNet dataset in *Xiang and Savarese* (2012) for bed, chair, sofa and table. We use the detailed ground truth annotations in *Xiang and Savarese* (2012), where each object is annotated by discretized azimuth, elevation, distance, and AP locations. The ground truth locations of AAPs and 3D aspectlets can be computed accordingly. Negative samples are from PASCAL VOC *Everingham et al.* (b). The same training datasets are used for two baselines: Deformable Part Model (DPM) *Felzenszwalb et al.* (2010) and Aspect Layout Model (ALM) *Xiang and Savarese* (2012). To measure the object detection performance, we use Average Precision (AP), where the standard 50% bounding box overlap criteria

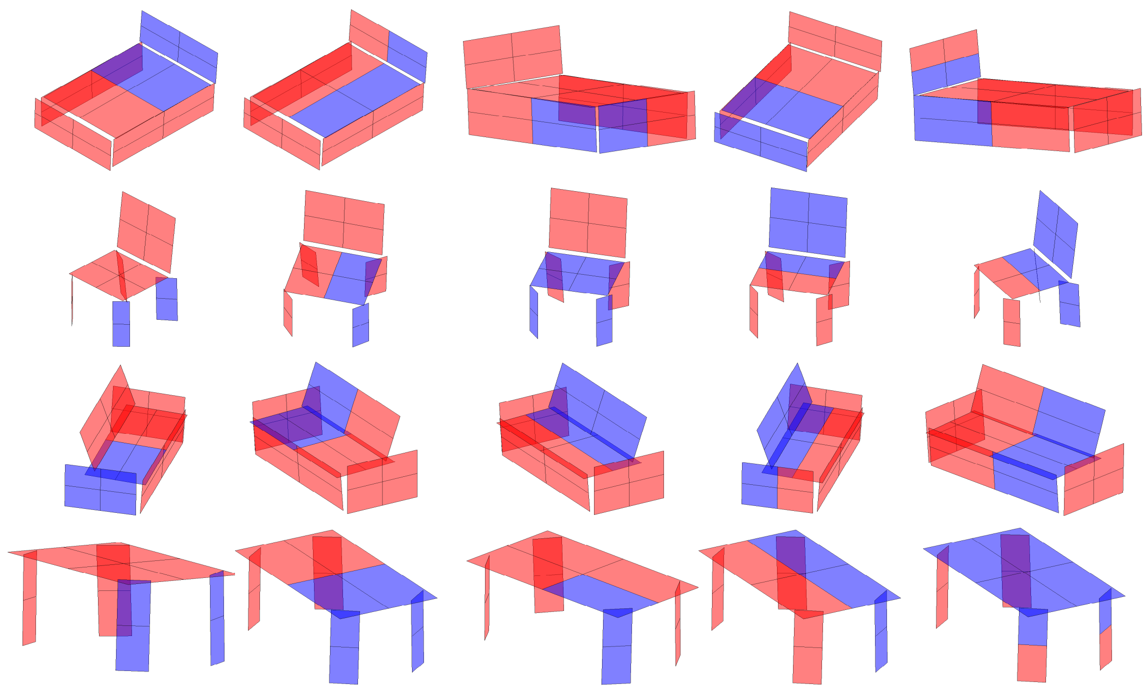


Figure 3.8: Sampled 3D aspectlets learnt in our experiments, where the blue AAPs belong to the 3D aspectlets.

Category	Car	Bed	Chair	Sofa	Table
ALM *Xiang and Savarese* (2012)	46.6	28.9	14.2	41.1	19.2
DPM *Felzenszwalb et al.* (2010)	57.0	34.8	14.4	38.3	15.1
SLM Aspectlets	59.2	35.8	15.9	45.5	24.3
SLM Full	**63.0**	**39.1**	**19.0**	**48.6**	**28.6**

Table 3.2: APs for the five categories in the two datasets.

of PASCAL VOC *Everingham et al.* (b) is used.

3.4.2 Results

After the learning of 3D aspectlets, we obtain 50 3D aspectlets for car, and 32, 46, 24 and 25 3D aspectlets for bed, chair, sofa and table respectively. Fig. 3.5(b) and Fig. 3.8 show some learnt 3D aspectlets in our experiments, where the blue AAPs belong to the 3D aspectlets. We compare the object detection performance of SLM with two baseline methods: the state-of-the-art object detector DPM *Felzenszwalb et al.* (2010) and the state-of-the-art object pose estimator ALM *Xiang and Savarese* (2012). Table

Dataset	Outdoor-scene			Indoor-scene		
% occlusion	<.3	.3-.6	>.6	<.2	.2-.4	>.4
# images	66	68	66	77	111	112
ALM *Xiang and Savarese* (2012)	72.3	42.9	35.5	38.5	25.0	20.2
DPM *Felzenszwalb et al.* (2010)	75.9	58.6	44.6	38.0	22.9	21.9
SLM Aspectlets	78.7	59.7	47.7	41.9	30.8	24.8
SLM Full	**80.2**	**63.3**	**52.9**	**45.9**	**34.5**	**28.0**

Table 3.3: APs/mAPs on the two datasets with different test image sets according to the degrees of occlusions.

Recall	54.8	64.6	76.8
ALM *Xiang and Savarese* (2012)	1.90	-	-
DPM *Felzenszwalb et al.* (2010)	2.07	2.39	-
SLM	**1.64**	**1.86**	2.33

Table 3.4: 3D localization errors on the outdoor-scene dataset according to best recalls of ALM, DPM and SLM respectively.

3.2 shows the average precisions of SLM and the two baseline methods on the two datasets. "SLM Aspectlets" only uses our unary 2D projection likelihood for detection without considering the occlusions between objects. By using 3D aspectlets, we are able to achieve better performance than the two baseline methods, which we attribute to the ability of 3D aspectlets to detect occluded or truncated objects. However, 3D aspectlets also produce more false alarms compared with the full object model since less visual features are available. By reasoning about occlusions, our full model "SLM Full" is able to increase the detection scores of truly occluded objects and penalize false alarms which introduce wrong occlusion patterns. As a result, "SLM Full" consistently achieves the best performance on the five categories in the two datasets.

To clearly see the advantage of SLM in handling occlusions, we partition the test images in the two datasets into three sets according to the degrees of occlusions respectively, and evaluate the detection performance of SLM on each of the three sets. For an occluded object, we define its occlusion percentage as the percentage of area occluded by other objects. Then the degree of occlusion for one image can be measured by the maximum occlusion percentage of the objects in the image. Table

3.3 shows the number of images in each set and the APs/mAPs of the three methods on different test sets. In all the settings, SLM achieves the best performance. Besides, the improvement for the large occlusion sets is significant, which demonstrates the ability of SLM to detect occluded and truncated objects.

In order to evaluate the 3D localization accuracy, we back-project the ground truth annotations and the 2D detections into 3D respectively and obtain two spatial layouts. Since their coordinate systems can be different, we first compute the pairwise distances among objects in each layout, and then evaluate the absolute error between two corresponding pairwise distances across the two layouts. Finally, we use the mean error in the dataset as the measure for 3D localization. Since the 3D location of an object is evaluated only if it is correctly detected, we present the mean pairwise distance error according to different recalls. Table 3.4 shows the errors according to the best recalls of ALM, DPM and SLM on the outdoor-scene dataset, where unit one is the length of the 3D car model. SLM achieves better 3D localization at the highest recalls of both ALM and DPM.

Fig. 3.9 and Fig.3.10 show some anecdotal detection results from our method. The 2D detections are high confidence detections in the MAP estimations from the MCMC inference. The 3D plots show the 3D spatial layout of the objects and the camera. Based on the detected AAPs, we are able to generate the 2D mask of an object. Then according to the inferred occlusions between objects, we can refine the 2D mask to only contain the visible region of the object, from which it is possible to clearly see which object occludes which.

3.5 Conclusion

We have proposed a novel Spatial Layout Model (SLM) for multiple object detection and occlusion reasoning. SLM contextualizes objects in their 3D geometric configuration with respect to the observer to help object detection. By combining

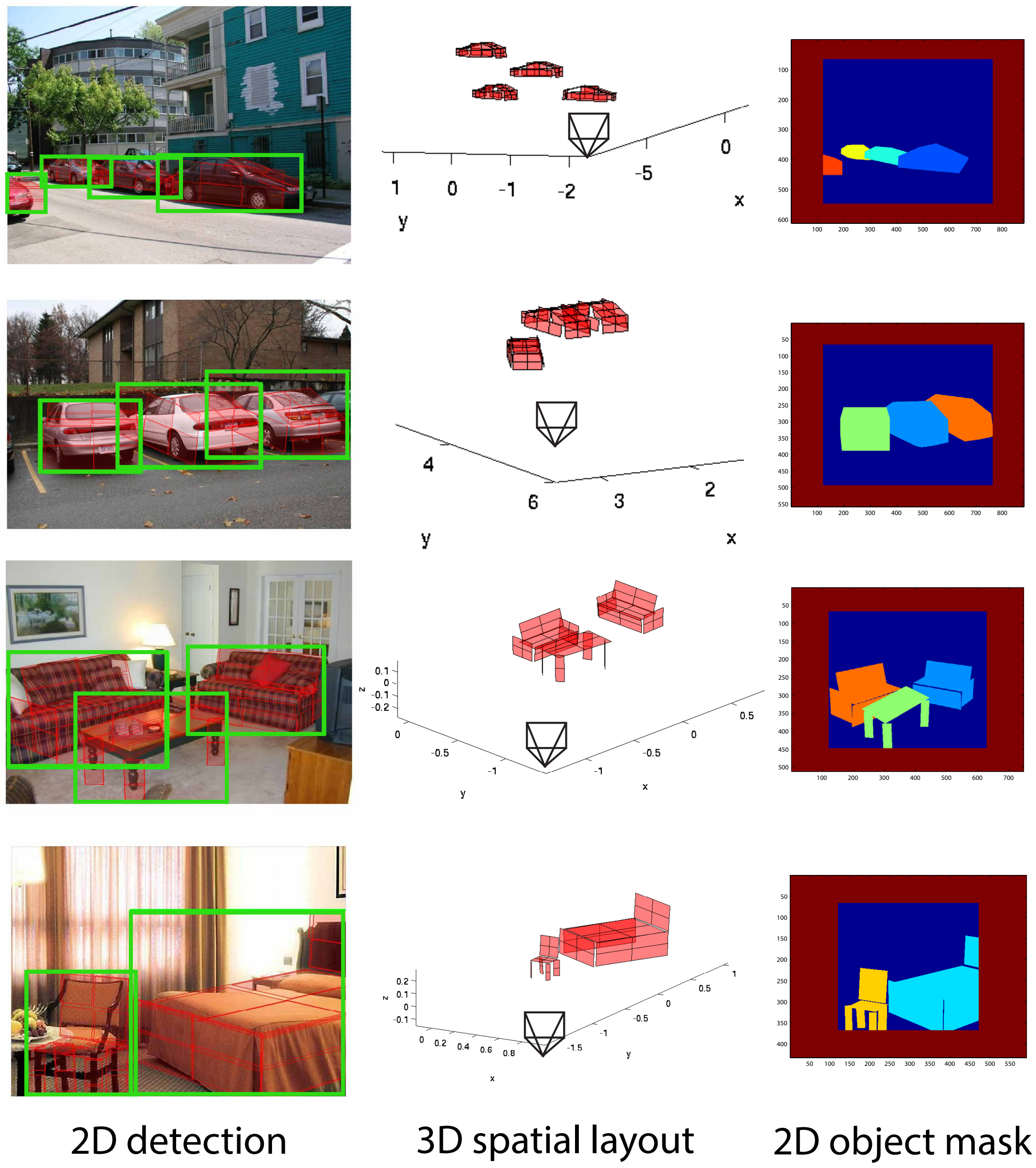


Figure 3.9: Anecdotal detection results on our datasets. The 2D detections show the detected objects in the images. The 3D plots show the spatial layout of objects and camera in 3D. The 2D object masks show the occlusion order in the images.

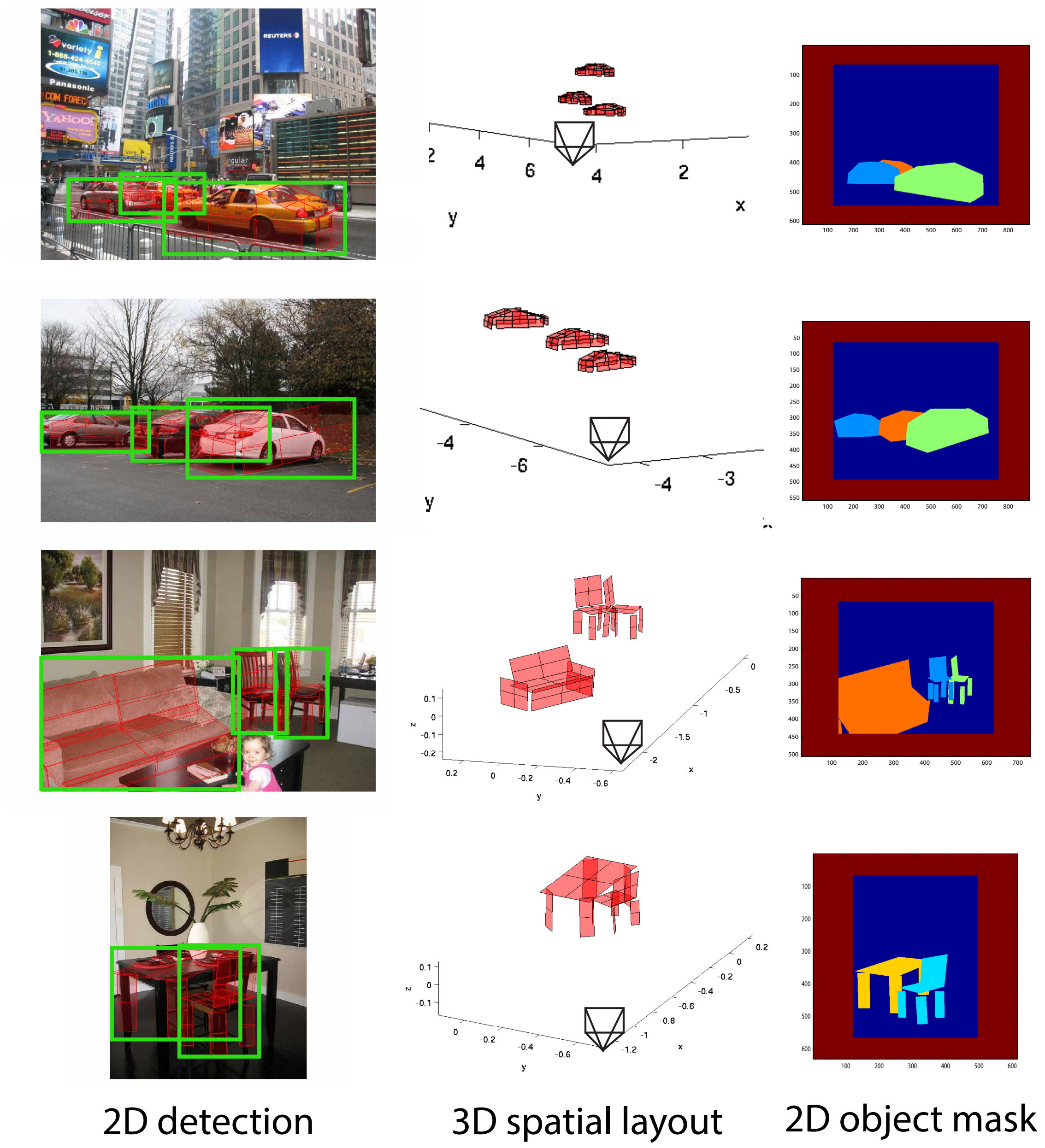


Figure 3.10: Anecdotal detection results on our datasets. The 2D detections show the detected objects in the images. The 3D plots show the spatial layout of objects and camera in 3D. The 2D object masks show the occlusion order in the images.

the bottom-up evidence from 3D aspectlets and the top-down occlusion reasoning, SLM is able to estimate the 3D spatial layout of objects and reason about occlusions between objects. Experiments on two new challenging datasets with various degrees of occlusions demonstrate the ability of our model to detect objects under severe occlusions and predict the occlusion patterns in images.

CHAPTER IV

3D Voxel Pattern Representation

4.1 Introduction

One of the major paradigms in modern object recognition consists of characterizing images with a list of 2D bounding boxes which correspond to the location and scale of the objects in the image. Recent methods have demonstrated that this task can be solved with a good degree of accuracy even when a large number of object categories is considered *Felzenszwalb et al.* (2010); *Krizhevsky et al.* (2012); *Girshick et al.* (2014). However, in many applications – autonomous driving is a notable example – recognizing objects as just 2D bounding boxes is not sufficient. In these applications, estimating the 3D object pose or figuring out the depth ordering of the objects from the observer is as important as (or even more important than) identifying the 2D locations of the objects. Moreover, in these scenarios, nuisances such as occlusions or truncation become dominant, and often one needs to recognize objects even when only a small portion of their surface is visible. The recently proposed KITTI benchmark *Geiger et al.* (2012) have been instrumental in highlighting the fact that object detection and 3D pose estimation tasks become extremely difficult when objects such as cars, bikes or trucks are to be recognized in the wild – that is within complex and cluttered urban scenes. Consider Fig. 4.1-top for instance, where cars occupy just a small portion of the image and most of them are heavily occluded by other cars.

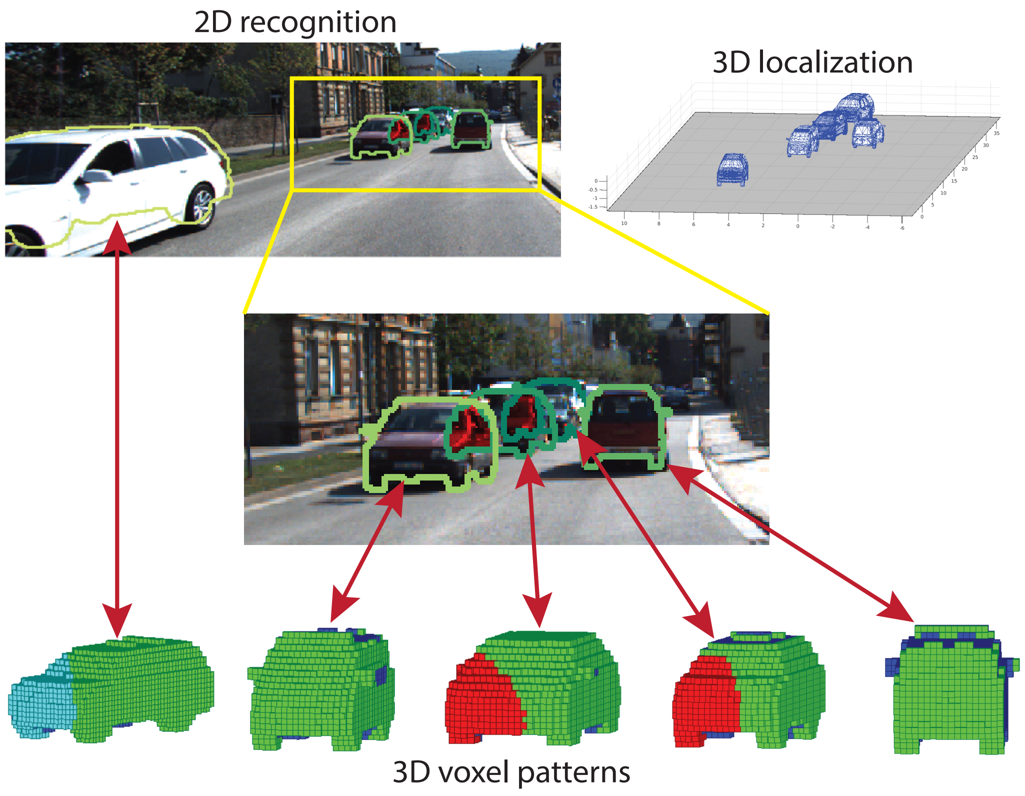


Figure 4.1: By introducing the 3D voxel patterns, our recognition framework is able to not only detect objects in images, but also segment the detected objects from the background, estimate the 3D poses and 3D shapes, localize them in the 3D space, and even infer the occlusion relationship among them. Green, red and cyan voxels are visible, occluded and truncated respectively.

Except for a few exceptions *Pepikj et al.* (2013); *Li et al.* (2014), most of the recent object detection methods have hard time in parsing out the correct configuration of objects from this kind of imagery.

In this paper, we present a novel recognition pipeline that addresses the key challenges above: i) it goes beyond 2D bounding box detection and is capable of estimating 3D properties of multiple detected objects such as 3D pose as well as their depth ordering from the observer; ii) it is designed to handle situations where objects are severely occluded by other objects or truncated because of a limited field of view; iii) it is capable of accurately estimating the occlusion boundaries of each objects as well as inferring which portions of the object are occluded or truncated and which are not (see Fig. 4.1).

At the foundation of our recognition pipeline is the newly proposed concept of *3D Voxel Pattern* (3DVP). A 3DVP is a novel object representation that jointly captures key object properties which relates: i) appearance – the RGB luminance values of the object in the image; ii) 3D shape – the 3D geometry of the object expressed as a collection of 3D voxels; iii) occlusion masks – the portion of the object that is visible or occluded because of self-occlusions, mutual occlusions and truncations (Fig. 4.3(d)). Our approach follows the idea that luminance variability of the objects in the image due to intra-class changes and occlusions can be effectively modeled by learning a large dictionary of such 3DVPs whereby each 3DPV captures a specific shared "signature" of the three properties listed above (appearance, 3D shape and occlusions). Examples of 3DVPs in the dictionary are shown in Fig. 4.6. Inspired by a recent body of work *Divvala et al.* (2012); *Chen et al.* (2014); *Divvala et al.* (2014); *Ohn-Bar and Trivedi* (2014) that proposes to learn object detectors using clusters of 2D images that share similar appearance properties, in our recognition pipeline we train a bank of detectors using our dictionary of 3DVPs whereby each detector is trained from the appearance information associated to a specific 3DVP. Thus, these detectors are designed to localize objects in the image even when they are observed from arbitrary viewpoints or visible under severe occlusions. Moreover, because the 3DVPs retain shared properties about the object (specifically, 3D shape and occlusion masks), these can be transferred during the detection regime so as to recover the 2D segmentation mask of the object, its 3D pose as well as which portions of the objects are occluded and which are visible. Finally, and most critically, we use these properties to reason about object-object interactions and infer which object is an "occluder" and which is an "occludee". This in turn helps adjusting the confidence values of the detectors (e.g., if we know that an object is occluded and we predict which portion is occluded, this can help reinforce the presence of the occluder and its location; vice versa, the occluder can help support the presence of the occludee and the portion of the object that is occluded).

We believe our approach is particularly valuable in an autonomous driving scenario where vehicles' locations must be detected from images as well as vehicles' precise depth ordering and pose configurations must be inferred in 3D. For that purpose, we trained and tested our approach using the KITTI detection benchmark *Geiger et al.* (2012) – a large dataset of videos of cars driving in challenging urban scenes – and focused on recognizing cars and estimating their 3D properties. We also evaluated our method using the outdoor-scene dataset proposed in *Xiang and Savarese* (2013) – a dataset that has been specifically designed to test object detectors in presence of severe occlusions. We note that even if we only tested our method on the "car" category, our approach is general and can be easily extended to other object categories. Our extensive experimental evaluation shows that: i) our approach based on 3D voxel patterns produces significant improvement over state-of-the-art results for car detection on KITTI ($\sim 6\%$ AP for the hard test set) and 3D pose estimation ($\sim 12\%$ AOS for the hard test set); ii) our approach allows us to accurately segment object boundaries and infer which areas of the objects are occluded and which are not; we demonstrate that our segmentations results are superior than several baseline methods; iii) our approach allows us to localize objects in 3D and thus infer the depth ordering of the object from the camera's viewpoint.

4.2 Related Work

We review representative techniques in handling different challenges in object category recognition, and relate our approach with them.

Shape variation. In order to handle the intra-class variability of shape, part-based object representations are introduced, such as the constellation model *Fergus et al.*

(2003) and pictorial structures *Felzenszwalb and Huttenlocher* (2005); *Felzenszwalb et al.* (2010). Another direction is to discover and learn appearance models for object subcategories *Divvala et al.* (2012); *Chen et al.* (2014); *Divvala et al.* (2014); *Ohn-Bar and Trivedi* (2014), where object instances in a subcategory share similar visual appearance. In our recognition framework, we discover 3D voxel patterns, where object instances in a 3DVP share similar visibility pattern.

Viewpoint. Recent progresses in multiview object recognition can be roughly classified according to their ways of representing the object category. In 2.5D object representation, object parts or features are connected across views *Thomas et al.* (2006); *Savarese and Fei-Fei* (2007); *Su et al.* (2009). While in 3D object representation, visual features are associated with explicit 3D models *Yan et al.* (2007); *Liebelt et al.* (2008); *Glasner et al.* (2011); *Xiang and Savarese* (2012). The 3D models can either be built from a set of 2D images in different views *Yan et al.* (2007); *Glasner et al.* (2011) or constructed using 3D CAD models *Liebelt et al.* (2008); *Xiang and Savarese* (2012). The new 3D object representation we introduce, i.e., 3D voxel pattern, utilizes 3D CAD models in the recognition pipeline.

Occlusion. In order to detect partially occluded objects, researchers have worked on training partial object detectors for visible parts of objects *Wu and Nevatia* (2005); *Wang et al.* (2009); *Gao et al.* (2011); *Wojek et al.* (2011); *Xiang and Savarese* (2013). Since partial object detectors are not very robust, *Wu and Nevatia* (2005); *Wojek et al.* (2011); *Xiang and Savarese* (2013) also jointly reason about the presence of multiple objects in the scene. *Zia et al.* (2013) and *Pepikj et al.* (2013) explicitly consider the occluder when detecting the occluded object by introducing occlusion masks and occlusion patterns respectively. In all the previous works, only limited number of occlusion patterns are modeled. In contrast, we propose a data-driven approach to handle a large number of occlusion patterns.

Truncation. Objects can be truncated by image borders due to the limited field of

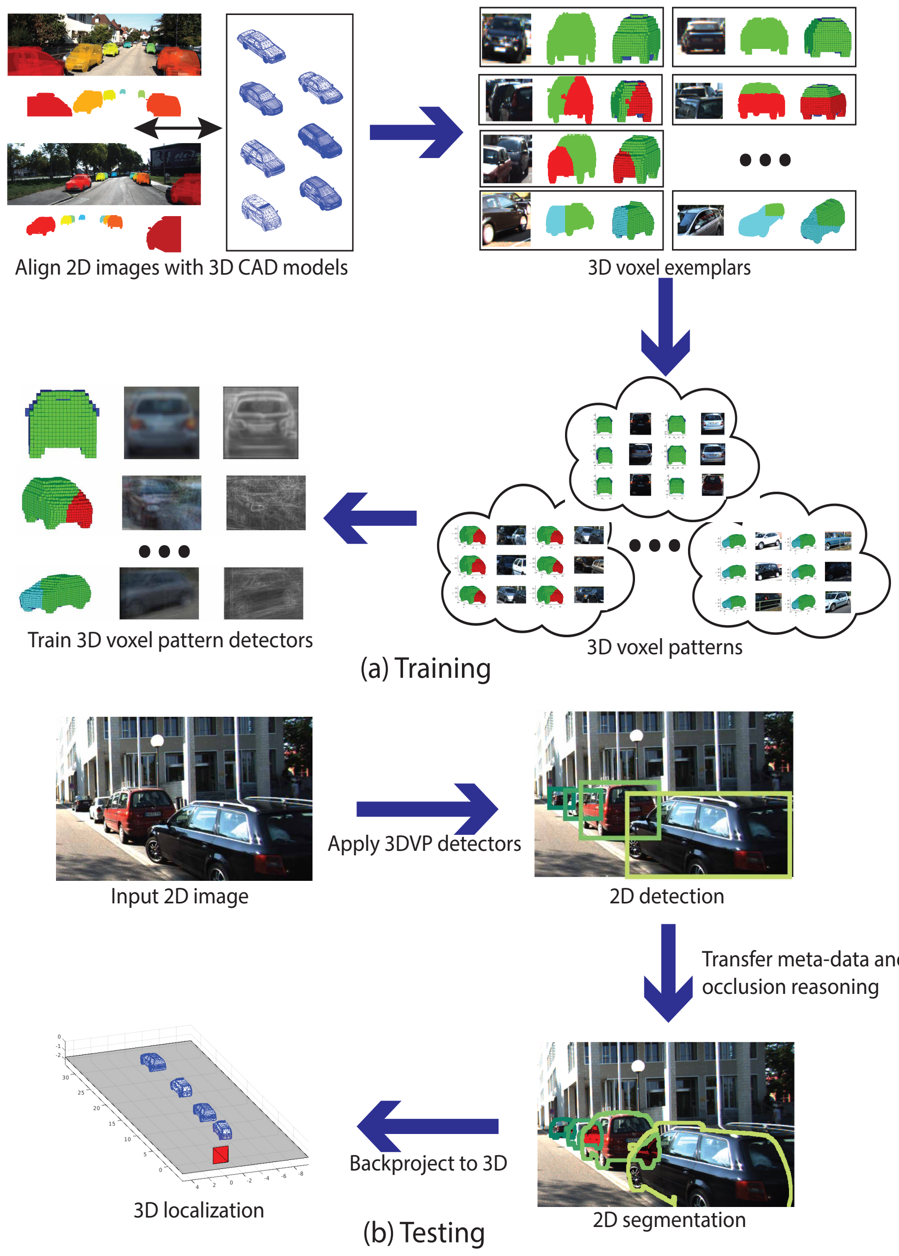


Figure 4.2: Overview of our object category recognition framework. (a) Training pipeline. (b) Testing pipeline.

view of the camera. Truncation is commonly handled by heuristics such as padding the image borders. An exception is *Vedaldi and Zisserman* (2009), which detected truncated objects with a structured output regression. In our work, we handle truncation by leveraging our 3DVP representation which can be used to characterize truncated objects.

Nearest neighbor and deep neural network. Nearest neighbor based methods *Malisiewicz et al.* (2011) and deep neural networks *Krizhevsky et al.* (2012); *Girshick et al.* (2014) handle the above factors in object category recognition implicitly. Nearest neighbor is able to transfer meta-data of the training examples to testing objects, such as 2D segmentation mask, 3D shape, and so on. We inherit this advantage in our recognition framework. In deep neural networks, millions of parameters are learned from training data which has the ability to handle different aspects in object recognition without explicit modeling them. However, deep neural networks cannot estimate explicit 3D geometrical properties, such as 3D pose or occlusion boundaries.

4.3 Object Category Recognition with 3DVPs

We propose a novel object recognition framework based on *3D Voxel Patterns* (3DVPs). 3DVPs are abstract 3D representations that capture patterns of visibility of an object category. The visibility of an object instance is represented by a *3D voxel exemplar*, which is a triplet of the 2D image of the object, its 2D segmentation mask and its 3D voxel model (see Fig. 4.4 for some examples).

In the training stage, we obtain 3D voxel exemplars of object instances in a data-driven approach (Sec. 4.3.1). Then we build a representative set of 3DVPs by clustering 3D voxel exemplars according to their visibility patterns (Sec. 4.3.2). Finally, we train an object detector for each 3DVP (Sec. 4.3.3), which is specialized to detect objects with specific visibility patterns. Fig. 4.2(a) illustrates our training pipeline. Our approach is similar in spirit to *Divvala et al.* (2012); *Chen et al.* (2014); *Divvala* *et al.* (2014); *Ohn-Bar and Trivedi* (2014) that build subcategories based on 2D appearance patterns. Unlike these works, however, we learn detectors on the 3DVPs which capture explicit information about the visibility patterns of objects.

In the testing phase, after applying 3DVP detectors to an input image, we can transfer the meta-data associated with the 3DVPs, such as 2D segmentation mask, 3D pose or 3D shape, to the detected objects. These transferred meta-data enables us to perform different recognition tasks beyond 2D detection, such as object segmentation, pose estimation, 3D localization and occlusion reasoning. Fig. 4.2(b) illustrates our testing pipeline.

4.3.1 3D Voxel Exemplars from Data

A 3D voxel exemplar captures the appearance, 3D shape and occlusion mask of an object. As a long as a method can produce the 2D segmentation mask and the 3D voxel model of an object in the image, it can be used to build 3D voxel exemplars. For example, one could collect data with depth sensors or 3D scanners. However, it is difficult to scale to a large number of objects. Our solution is to utilize 3D CAD models in repositories on the web, such as the Trimble 3D Warehouse *Trimble*, and register these 3D CAD models with 2D images to build 3D voxel exemplars. In this way, we can obtain 3D voxel exemplars for tens of thousands of objects. We illustrate how to build 3D voxel exemplars for cars using the KITTI detection benchmark *Geiger et al.* (2012) in Fig. 4.3: 1) For each image in the training set, an object in the image is registered with a 3D CAD model selected from a pre-defined collection of models, where the model which has the closest aspect ratio with the ground truth 3D cuboid of the object instance is selected. The KITTI dataset *Geiger et al.* (2012) provides ground truth 3D annotations (cuboids) and camera parameters. Then we register the chosen 3D CAD model to the ground truth 3D cuboid associated to the object instance (Fig. 4.3(a)). 2) We project all the registered 3D CAD models onto

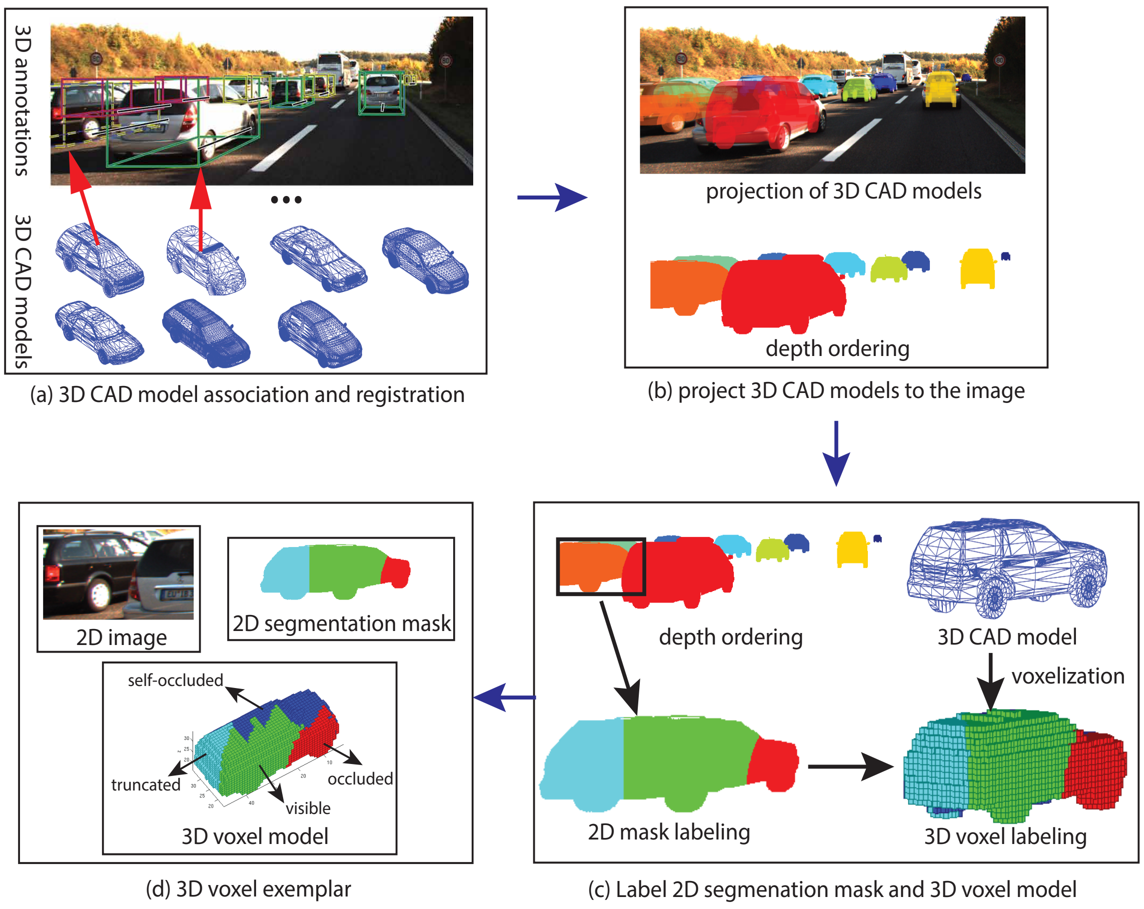


Figure 4.3: Illustration of generating 3D voxel exemplars from images and annotations available from the KITTI detection benchmark *Geiger et al.* (2012).

the image plane using the camera parameters and obtain the depth ordering mask (Fig. 4.3(b)). 3) The depth ordering mask determines which pixel of the projected 3D CAD model is visible, occluded, or truncated. So we can generate a 2D segmentation mask for each object associated with visibility labels. We use green to color "visible" pixels, red to color "occluded" pixels, and cyan to color "truncated" pixels in the segmentation mask. To build the 3D voxel model for the object, we first voxelize the associated 3D CAD model. Then we check the status of each voxel in the voxelize 3D CAD model. From the camera viewpoint and the geometry of the 3D CAD model, we can figure out which voxels are visible or self-occluded (blue). For each visible voxel, we project it onto the depth ordering mask to determine whether it is occluded or truncated (Fig. 4.3(c)). The result is a triplet called 3D voxel exemplar, which comprises the image of the object, the 2D segmentation mask of the object and the corresponding distribution of 3D voxels with associated visibility labels (Fig. 4.3(d)). More examples of the built 3D voxel exemplars are shown in Fig. 4.4.

The 3D voxel representation has several good properties. First, by encoding the 3D voxel space into empty or occupied voxels, 3D voxel exemplars can capture the 3D shape of objects. Second, viewpoint information is encoded by labeling the occupied voxels into visible or self-occluded voxels. Third, the visible voxels are further classified into truncated or occluded voxels by considering the image borders and other objects in the scene. As a result, 3D voxel exemplars are able to encode information about 3D shape, viewpoint, truncation and occlusion in a uniform 3D space.

4.3.2 Discovering 3DVPs

A 3DVP represents a group of 3D voxel exemplars which share similar visibility patterns encoded in their 3D voxel models. We discover 3DVPs by clustering 3D voxel exemplars in a uniform 3D space. To do so, we define a similarity score between two 3D voxel exemplars. Formally, a 3D voxel exemplar is represented by a feature vector

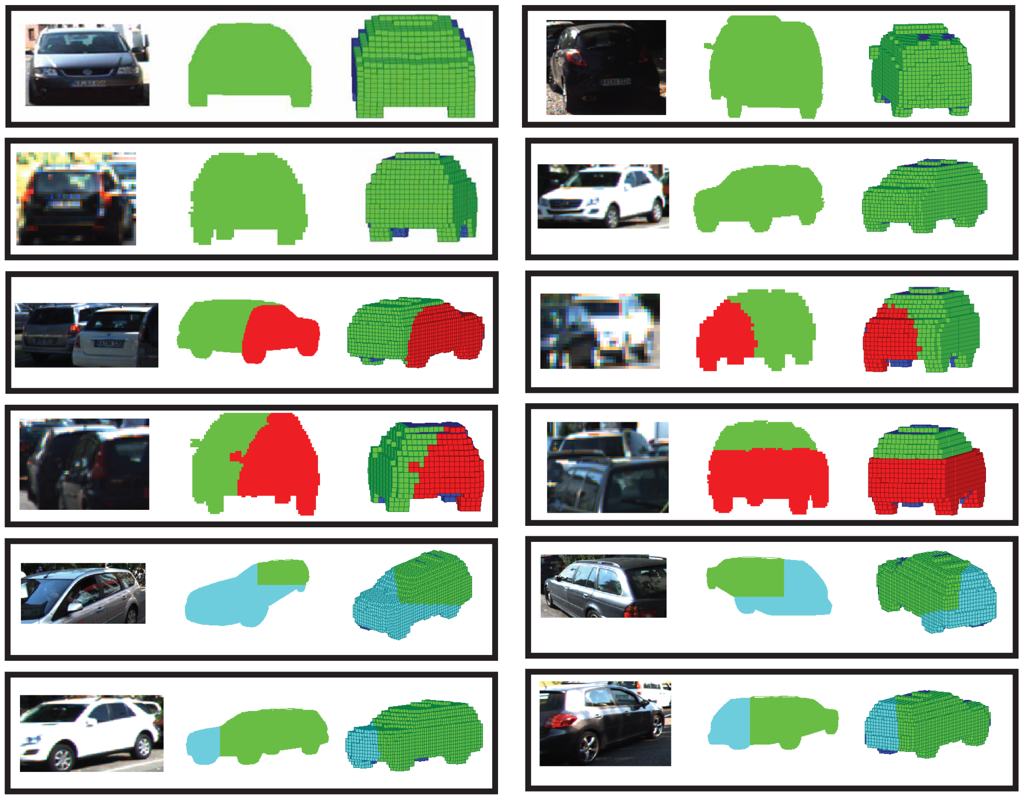


Figure 4.4: Examples of 3D voxel exemplars. Red indicates occlusion, and cyan indicates truncation.

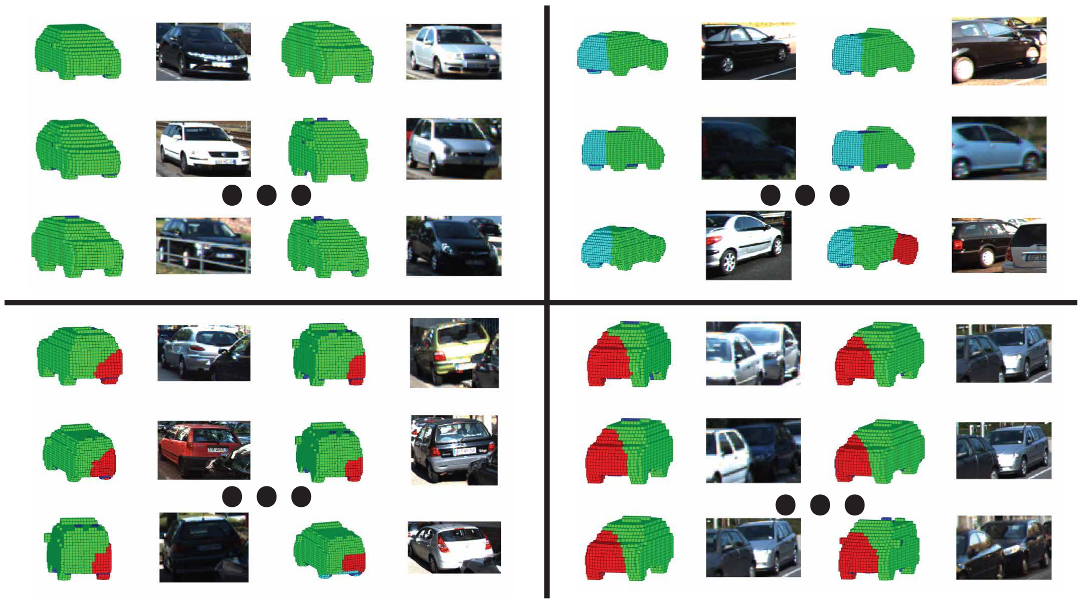


Figure 4.5: Examples of 3D clusters from the KITTI dataset.

$\mathbf{x}$ with dimension N^3, where N denotes the size of the 3D voxel space. The elements of the feature vector takes values from a finite set $\mathcal{S} = \{0, 1, 2, 3, 4\}$, which encodes the visibility of the voxels, i.e., 0 for empty voxels, 1 for visible voxels, 2 for self-occluded voxels, 3 for voxels occluded by other objects, and 4 for truncated voxels. Then the similarity metric between two feature vectors $\mathbf{x_1}$ and $\mathbf{x_2}$ is defined as:

$$\begin{aligned} s(\mathbf{x_1}, \mathbf{x_2}) &= \tfrac{|\mathcal{S}|}{N^3} \textstyle\sum_{i=1}^{N^3} \mathbb{1}(x_1^i = x_2^i) \cdot w(x_1^i), \\ \text{s.t.}, &\ \textstyle\sum_{i=0}^{|\mathcal{S}|-1} w(i) = 1, \end{aligned} \tag{4.1}$$

where x_1^i and x_2^i are the ith element of $\mathbf{x_1}$ and $\mathbf{x_2}$ respectively, $\mathbb{1}$ is the indicator function, and $w(i)$ is the weight for voxel status i. The definition in Eq. (4.1) is general such that the weights can be designed for different applications. For example, if we define all the weights $w(i)$ to $1/5$, the similarity metric in Eq. (4.1) simply computes the percentage of voxels with the same value. If we use a larger weight for occluded voxels, patterns with similar occluded regions are more likely to be grouped together (See supplementary material for our implementation details).

After defining the similarity metric between 3D voxel exemplars, we can employ different clustering algorithms in our framework, such as K-means or Affinity Propagation (AP) *Frey and Dueck* (2007). Fig. 4.5 shows several examples of 3D clusters from the KITTI dataset using AP clustering. With the 3D clustering algorithm, we are able to group cars from similar viewpoints and with similar occluded or truncated regions together. We visualize 3DVPs in Fig. 4.6. For each cluster, we show the 3D voxel model of the cluster center, the average RGB images of the 2D image patches in the cluster, and the average gradient image. Note that there is a high correlation between 3DVP and object appearance including relevant occlusions, which enable us to learn compact and accurate detectors for 3DVPs.

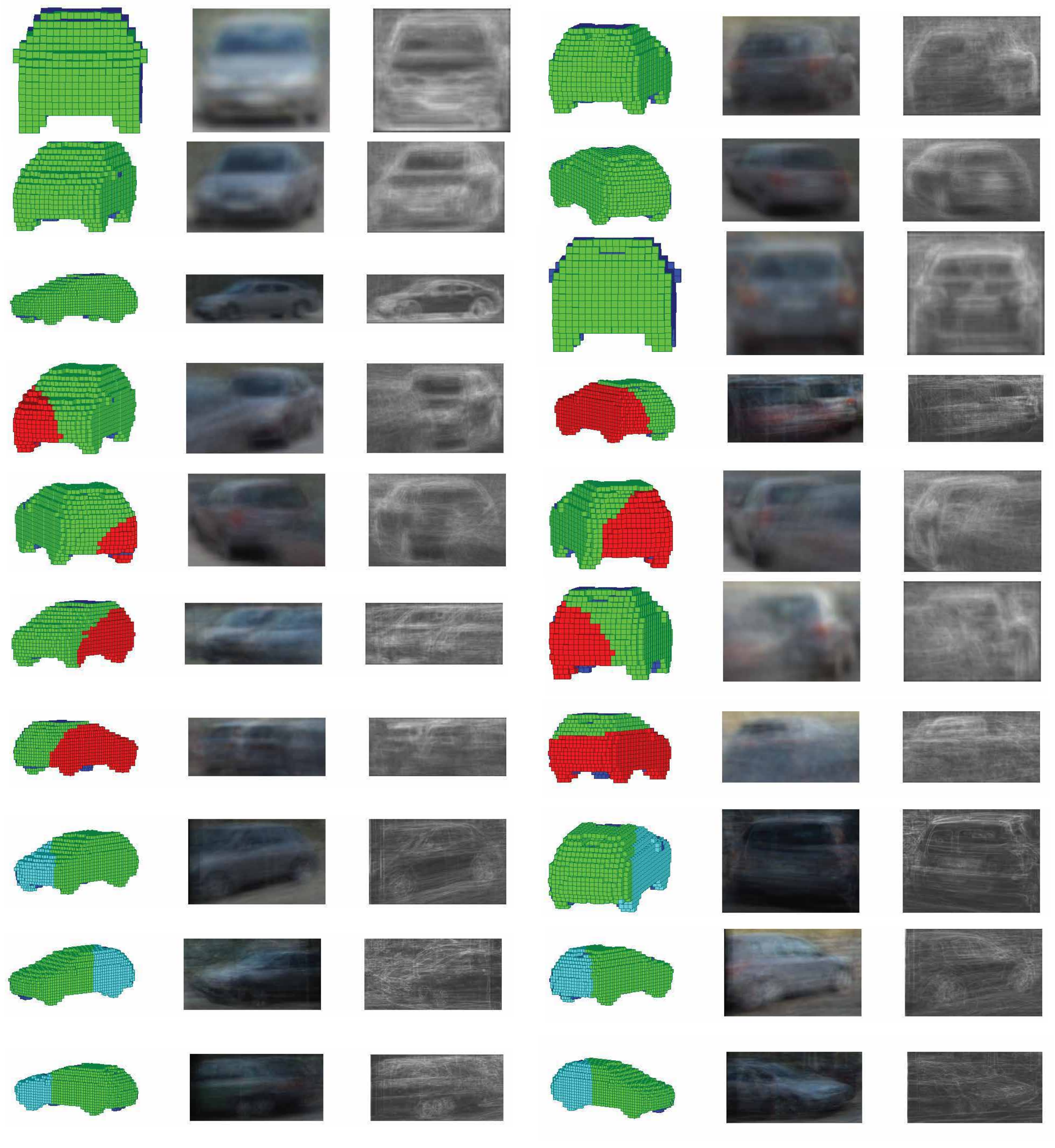


Figure 4.6: Visualization of selected 3DVPs. We show the 3D voxel model of the cluster center, the average RGB image, and the average gradient image of each 3DVP.

4.3.3 Learning 3DVP Detectors

We train a detector for each 3DVP with the associated 2D images. Our framework is general to integrate different classifiers in training the detectors, such as support vector machines or boosting. For example, in our experiments, we note that the Aggregated Channel Features (ACF) *Dollár et al.* (2014) is more suitable on the KITTI dataset compared to SVM-based detectors *Felzenszwalb et al.* (2010); *Malisiewicz et al.* (2011).

For an 3DVP that contains occlusion, we incorporate the appearance of the occluder which is inside the 2D bounding box of the occludee in training the 3DVP detector, where the 2D bounding box incorporates occluded area of the occludee. The observation behind it is that occlusions are likely to form certain types of patterns between the occluder and the occludee. For example, in street scenes, cars are likely to occlude each other within specific 3D spatial layout. Such cases include cars parking beside the street, cars lining up on the road, and so on. Incorporating the appearance of the occluder into modeling helps us to detect the occluded by leveraging these occlusion patterns. The 3DVPs we discover in the 3D clustering process capture these occlusion patterns. As we can see from Fig. 4.5 and Fig. 4.6, the included regions from the occluders in an occluded 3DVP also share similar appearance, which ensures us to train a reliable detector for the occluded 3DVP. For a truncated 3DVP, image patches corrsponding to the truncated objects are used to train the detector without padding.

4.3.4 Occlusion Reasoning with 3DVPs

After training all the 3DVP detectors, we can apply them to an input image and obtain the 2D detections. Then, we are able to transfer the meta-data from the 3DVP to the detected objects, which includes the 2D segmentation mask, the 3D pose and the 3D shape as shown in Fig. 4.2(b). These meta-data enable us to perform a global occlusion reasoning among all the detected objects in the scene, which outputs mutually consistent detections.

Let $\mathbb{D} = \{d_1, d_2, ...\}$ denote the detection hypotheses in an image I. We represent a detection d_i by its detection score s_i, and its 2D visibility mask m_i that are derived from the 3DVP. Specifically, we transfer the 2D segmentation mask associated with the cluster center of the 3DVP to the detection, and rescale it to fit the bounding box of the detection. m_i is composed of three components: m_i^v (visible region), m_i^o (occluded region), and m_i^t (truncated region) (refer to examples in Fig. 4.4). We design our occlusion reasoning model using an energy-based conditional random field model *LeCun et al.* (2006), which favors to have detections that are mutually consistent to each other. Underlying intuition is that 1) all the invisible regions of selected detections shall be explained either by another occluding object or by image truncation, and 2) visible regions of selected detections should not overlap with each other. The model is formulated as:

$$E(\hat{\mathbb{D}}) = \sum_{i \in \hat{\mathbb{D}}} \Big(\underbrace{w_d(s_i - b)}_{\text{detection score}} - \underbrace{w_o \frac{|m_i^o| + |m_i^t|}{|m_i|}}_{\text{invisibility penalty}} + \underbrace{w_o \frac{|m_i^t \not\subset I|}{|m_i|}}_{\text{truncation explained}} \Big) + \sum_{i,j \in \hat{\mathbb{D}}, i \neq j} \Big(\underbrace{w_o \frac{|m^o_{\text{far}(i,j)} \cap m^v_{\text{near}(i,j)}|}{|m_{\text{far}(i,j)}|}}_{\text{occlusion explained}} - \underbrace{w_p \frac{\sum_{k=v,o,t} |m_i^k \cap m_j^k|}{\min(|m_i|, |m_j|)}}_{\text{overlap penalty}} \Big) \tag{4.2}$$

where w_d, w_o, w_p and b are the model parameters, $|\cdot|$ operator measures the area of a region, far$(\cdot)$ and near$(\cdot)$ return far and near object based on the bottom position of a detection, and $\hat{\mathbb{D}} \subseteq \mathbb{D}$. Our model has a number of favorable properties. First, detection outputs that are associated with largely occluded patterns are penalized by the invisibility penalty term in Eq. (4.2) unless the occluded area is explained by other objects (see the "occlusion explained" term). Similarly, truncated detections are also penalized by the "invisibility penalty" term unless they are located in accordance with the image boundary (see the "truncation explained" term). Second, detections that overlap largely with other detections are penalized according to the overlap penalty term in Eq. (4.2), which implements a similar concept as non-maximum suppression, but our model is more fine-grained as it measures the overlap between visible areas.

Solving the exact inference problem of our occlusion reasoning model is infeasible as the graph is often very complex, i.e., there are many overlapping detections which create a locally fully connected graph. So we solve the MAP inference problem with a greedy algorithm. Starting from an empty set $\hat{\mathbb{D}}_0 = \emptyset$, we add one detection d_i to the set $\hat{\mathbb{D}}_k$ in each iteration k that maximizes the energy improvement $E(\hat{\mathbb{D}}_k \cup d_i) - E(\hat{\mathbb{D}}_k)$ until the energy improvement is smaller than zero. In order to rank detections, we compute the posterior marginals from the estimated MAP as in *Desai et al.* (2011), and use them as detection scores. We train the model parameters by grid search on the validation set.

4.4 Experiments

4.4.1 Datasets and Evaluation Metrics

Datasets. We apply our object recognition framework to the KITTI detection benchmark *Geiger et al.* (2012) and the outdoor-scene (OutdoorScene) dataset *Xiang and Savarese* (2013) for car detection. The KITTI dataset contains 7,481 images for training, and 7,518 images for testing. These are video frames from autonomous driving scenes. We focus on the car category in KITTI, since there are 28,612 cars in the training set which provides enough data to test our data-driven approach. Since the ground truth annotations of the KITTI test set are not released, we split the KITTI training images into train set and validation set to conduct analyses about our framework, which contain 3,682 images and 3,799 images respectively. Our splitting ensures that there is no images from the same video across the train and validation sets. We also evaluate our algorithm on the entire test set. The OutdoorScene dataset contains

200 images from various sources, which is designed to test object detectors in presence of severe occlusions and truncation. There are 659 cars in total, among which 235 cars are occluded and 135 cars are truncated, which are only used for testing.

Evaluation Metrics. We evaluate our recognition results at the three difficulty levels, easy, moderate, and hard, suggested by the KITTI benchmark *Geiger et al.*. To evaluate the object detection accuracy, the Average Precision (AP) *Everingham et al.* (b) is reported throughout the experiments. 70% overlap threshold is adopted in the KITTI benchmark for car. To evaluate jointly object detection and orientation estimation, *Geiger et al.* (2012) proposes a new metric called Average Orientation Similarity (AOS), which is defined as $AOS = \frac{1}{11}\sum_{r\in\{0,0.1,\ldots,1\}} \max_{\tilde{r}:\tilde{r}\geq r} s(\tilde{r})$, where r is the detection recall, and $s(r) \in [0,1]$ is the orientation similarity at r (see *Geiger et al.* (2012) for details). In addition, we propose two new evaluation metrics to measure the accuracy of 2D segmentation and 3D localization jointly with detection. For 2D segmentation, we define Average Segmentation Accuracy (ASA) by replacing the orientation similarity in AOS with the 2D pixel segmentation accuracy. For the 3D localization, we define Average Localization Precision (ALP) by replacing orientation similarity in AOS with localization precision, i.e., a 3D location is considered to be correct if its distance from the ground truth 3D location is smaller than certain threshold. For object detection evaluation on the OutdoorScene dataset, we use the standard 50% overlap criteria of PASCAL VOC *Everingham et al.* (b).

4.4.2 Analysis on KITTI Validation Set

In this section, we present the detailed analysis on our method using our validation split of the KITTI training set.

2D v.s. 3D Clustering. We show that the our method, which discovers 3DVPs with 3D clustering and trains detectors on 3DVPs, can improve the object detection performance compared with its 2D counterpart proposed in the literature *Divvala et al.*

2D K-means				3D K-means			
K	Easy	Moderate	Hard	K	Easy	Moderate	Hard
5	44.21	31.23	25.42	5	41.78	31.63	28.06
10	47.78	38.13	32.26	10	52.55	39.44	32.76
20	61.24	48.04	40.27	20	61.52	49.33	42.07
30	**67.83**	51.68	43.63	30	63.29	49.46	41.55
40	66.49	**53.18**	**45.96**	40	69.46	56.13	47.26
50	66.65	51.90	43.28	50	70.76	58.77	50.30
100	58.45	46.15	39.34	100	75.73	61.06	51.29
150	56.74	43.84	37.75	150	77.15	63.25	53.13
200	53.57	41.26	33.61	200	78.00	**64.81**	**54.30**
250	53.86	39.81	33.58	250	76.85	63.48	53.93
300	48.81	35.53	29.10	300	**78.10**	62.11	51.99
350	42.68	33.55	27.35	350	74.78	62.00	51.81

2D Affinity Propagation				3D Affinity Propagation			
K	Easy	Moderate	Hard	K	Easy	Moderate	Hard
137	**46.76**	**35.66**	**32.30**	87	74.28	62.54	52.87
156	46.12	34.44	30.35	125	78.28	**65.62**	**54.90**
189	44.97	34.88	31.53	135	78.13	65.44	54.79
227	39.66	31.67	29.62	152	77.96	64.45	53.93
273	36.52	28.51	27.08	180	79.02	65.55	54.72
335	27.96	22.74	22.22	229	79.94	64.87	53.53
				284	79.91	64.04	53.10
				333	**79.98**	63.95	52.99

Table 4.1: AP Comparison between 2D and 3D clustering with k-means and affinity propagation on our validation split. The table shows the average precision obtained by training ACF detectors in different settings.

Object Detection (AP)			
Methods	Easy	Moderate	Hard
DPM *Felzenszwalb et al.* (2010) NMS.5	54.91	42.49	32.73
DPM *Felzenszwalb et al.* (2010) INMS.6	44.35	36.49	28.87
Ours NMS.5	79.06	64.72	50.38
Ours INMS.6	78.28	65.62	54.90
Ours Occlusion	**80.48**	**68.05**	**57.20**

Orientation Estimation (AOS)			
Methods	Easy	Moderate	Hard
DPM *Felzenszwalb et al.* (2010) NMS.5	33.71	26.30	20.37
DPM *Felzenszwalb et al.* (2010) INMS.6	27.45	22.71	18.07
Ours NMS.5	77.65	62.75	48.57
Ours INMS.6	76.87	63.49	52.57
Ours Occlusion	**78.99**	**65.73**	**54.67**

Table 4.2: AP/AOS comparison between different detection/decoding methods on the validation set. We show the results of 3D AP with 125 clusters for **Ours**.

(2012); *Ohn-Bar and Trivedi* (2014) that discovers subcategories with 2D clustering and trains detectors on the subcategories. For 2D clustering, we extract visual features from the training instances, and cluster them by computing similarity between the 2D features similarly to *Divvala et al.* (2012); *Ohn-Bar and Trivedi* (2014). We experiment with two clustering algorithms, K-means and Affinity Propagation (AP) *Frey and Dueck* (2007), with different numbers of clusters. The control parameter in AP is varied to obtain different number of clusters. We train ACF detectors *Dollár et al.* (2014) for both 2D and 3D clusters. Table 4.1 shows the average precisions by applying the trained ACF detectors to the validation set. We can see from the table that 3D K-means and 3D AP outperform their 2D counterparts significantly. Our evaluation verifies that 3DVP driven detectors can better capture the appearance variation of an object category compared to the 2D driven detectors. We also observe that 3D AP is less susceptible to the choice of the cluster numbers. In the following analyses, we experiment with the 3DVP detectors trained on 125 clusters from 3D AP clustering.

Decoding Detection Hypotheses. Table 4.2 compares the detection and the orientation estimation accuracies on the validation set among DPM baselines and our 3DVP detectors using different decoding schemes. As the first decoding scheme, we adopt the popular Non-Maximum Suppression (NMS) implemented by *Felzenszwalb et al.* (2010). The method computes the overlap between two bounding boxes by $\frac{|o_i \cap o_j|}{|o_i|}$ and greedily suppresses detections that have larger than 0.5 overlap with already selected ones. Since this method (NMS.5) tends to suppress less confident occluded detections aggressively, which hurts the performance of 3DVP detectors in Hard case, we adopt another NMS method based on Intersection over Union (IoU) $\frac{|o_i \cap o_j|}{|o_i \cup o_j|}$ with 0.6 threshold (INMS.6). It performs the same suppression procedure as NMS.5, but using the 0.6 IoU threshold. INMS.6 tends to keep more occluded detection hypotheses and achieves better performance in moderate and hard cases compared to NMS.5. Finally, our occlusion reasoning method improves the detection and orientation estimation accuracies with significant margins in all difficulty levels. The superior results verifies that 3DVP detectors are able to learn accurate visibility patterns of the objects, which provides reliable cues to reason about the occlusion relationship between objects.

Joint 2D Detection and Segmentation Evaluation. We analyze the accuracy of the transferred 2D segmentation mask from 3DVP in terms of 2D segmentation accuracy. Since the KITTI dataset does not provide the ground truth segmentation masks of the objects, we use the 2D segmentation masks obtained by projecting registered 3D CAD models as the ground truth (Fig. 4.3). Since the registration is guided by the 3D annotations, the obtained masks are accurate. We use the ASA metric described in Sec. 4.4.1 for the evaluation. Table 4.3 shows the accuracies of different methods. As DPM *Felzenszwalb et al.* (2010) does not provide any segmentation information, we treat the whole region inside the bounding box as the segmentation mask (denoted as +box). As the results demonstrate, our 3DVP induced segmentations (+3DVP) improve 6%, 5% and 4% in each difficulty level compared to our own

Method	Easy	Moderate	Hard
Joint Detection and Segmentation (ASA)			
DPM *Felzenszwalb et al.* (2010)+box	38.09	29.42	22.65
Ours INMS.6+box	57.52	47.84	40.01
Ours Occlusion+box	59.21	49.74	41.71
Ours INMS.6+3DVP	63.88	52.57	43.82
Ours Occlusion+3DVP	**65.73**	**54.60**	**45.62**
Joint Detection and 3D Localization (ALP)			
DPM *Felzenszwalb et al.* (2010) < 2m	40.21	29.02	22.36
Ours INMS.6 < 2m	64.85	49.97	41.14
Ours Occlusion < 2m	**66.56**	**51.52**	**42.39**
DPM *Felzenszwalb et al.* (2010) < 1m	24.44	18.04	14.13
Ours INMS.6 < 1m	44.47	33.25	26.93
Ours Occlusion < 1m	**45.61**	**34.28**	**27.72**

Table 4.3: Comparison between different settings of our method and DPM for the 2D segmentation and 3D localization evaluation on our validation split, where 125 clusters from 3D AP clustering are used for **Ours**.

baselines (+box) and 17%, 25%, 23% compared to the DPM baseline.

Joint Detection and 3D Localization Evaluation. In Table 4.3, we also evaluate the 3D localization accuracy using the average localization precision (ALP). The 3D location of a 2D detection is computed by minimizing the re-projection error between a oriented mean 3D cuboid and the 2D bounding box of the detection, where the mean 3D cuboid is obtained by averaging the 3D dimensions of all the training objects, and the orientation is estimated by the detection. The re-projection error is the sum of squared errors in width and height between the projected 3D cuboid and the 2D bounding box. So accurate 2D bounding box and 3D pose produce precise 3D localization. We evaluate the performance using two 3D distance thresholds: 1 meter and 2 meters. In both experiments, **Ours** Occlusion achieves better 3D localization results than **Ours** INMS.6, and improves over the DPM baseline by more than 20% in 2-meter ALP and more than 10% in 1-meter ALP. We note that *Zia et al.* (2014) also evaluates 3D localization on KITTI images. However, the method is trained with external images and only tested on 260 KITTI images. We could not directly

Object Detection (AP)			
Methods	Easy	Moderate	Hard
DPM *Felzenszwalb et al.* (2010)	71.19	62.16	48.43
OC-DPM *Pepikj et al.* (2013)	74.94	65.95	53.86
AOG *Li et al.* (2014)	80.26	67.03	55.60
SubCat *Ohn-Bar and Trivedi* (2014)	81.94	66.32	51.10
Regionlets *Wang et al.* (2013)	84.27	75.58	59.20
Ours INMS.6	84.81	73.02	63.22
Ours Occlusion	**87.46**	**75.77**	**65.38**

Orientation (AOS)			
Methods	Easy	Moderate	Hard
DPM *Felzenszwalb et al.* (2010)	67.27	55.77	43.59
OC-DPM *Pepikj et al.* (2013)	73.50	64.42	52.40
AOG *Li et al.* (2014)	44.41	36.87	30.29
SubCat *Ohn-Bar and Trivedi* (2014)	80.92	64.94	50.03
Ours INMS.6	84.31	71.99	62.11
Ours Occlusion	**86.92**	**74.59**	**64.11**

Table 4.4: AP/AOS Comparison between different methods on the KITTI test set. We show the results of 3D AP with 227 clusters for **Ours**. More comparisons are available at *Geiger et al.*.

compare our results with *Zia et al.* (2014). Please see Fig. 4.7 for qualitative results using our method on the validation set.

4.4.3 KITTI Test Set Evaluation

To compare with the state-of-the-art methods on the KITTI dataset, we train 3DVP detectors with all the KITTI training data, and then test our method on the test set. We present the detection and the orientation estimation results in Table 4.4. The 3DVPs are obtained using AP clustering with 227 clusters. Each 3DVP detector is trained with the ACF detector *Dollár et al.* (2014). We evaluate the **Ours** INMS.6 and **Ours** Occlusion. Thanks to our 3DVP model, **Ours** INMS.6 already achieves the highest accuracies in most of the difficulty levels for both detection and orientation evaluation. Our full model achieves even further improvement (3.2%, 0.2% and 6.2% higher AP compared to the second best method and 6%, 9.7% and 11.7% higher AOS

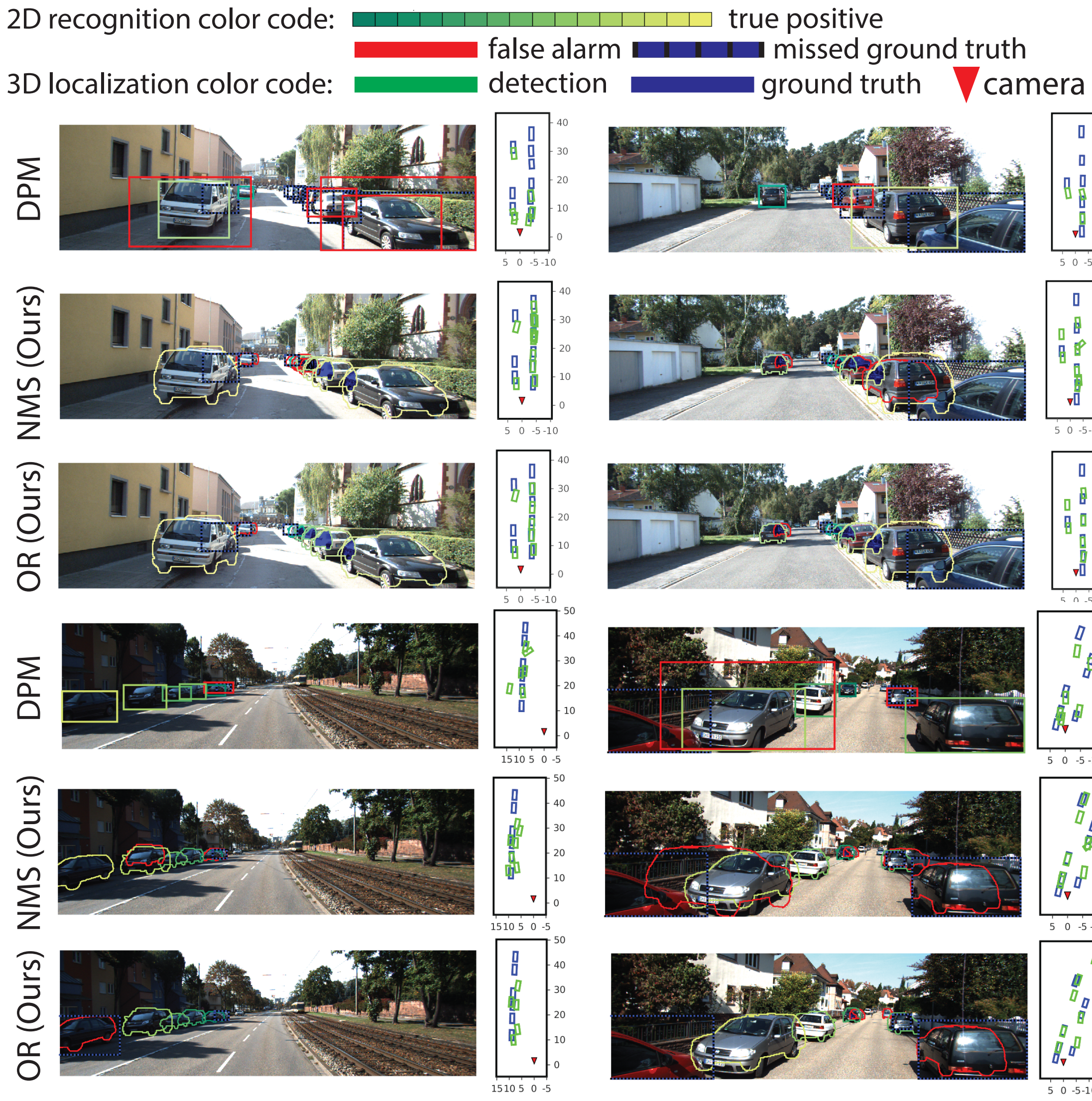


Figure 4.7: Car recognition results on the KITTI validation set. We compare our method w/wo occlusion reasoning and DPM *Felzenszwalb et al.* (2010). Detections at 1 false positive per image (fppi) for the three methods are shown. Blue regions in the images are the estimated occluded areas. Note that severe false alarms in NMS disappear with occlusion reasoning.

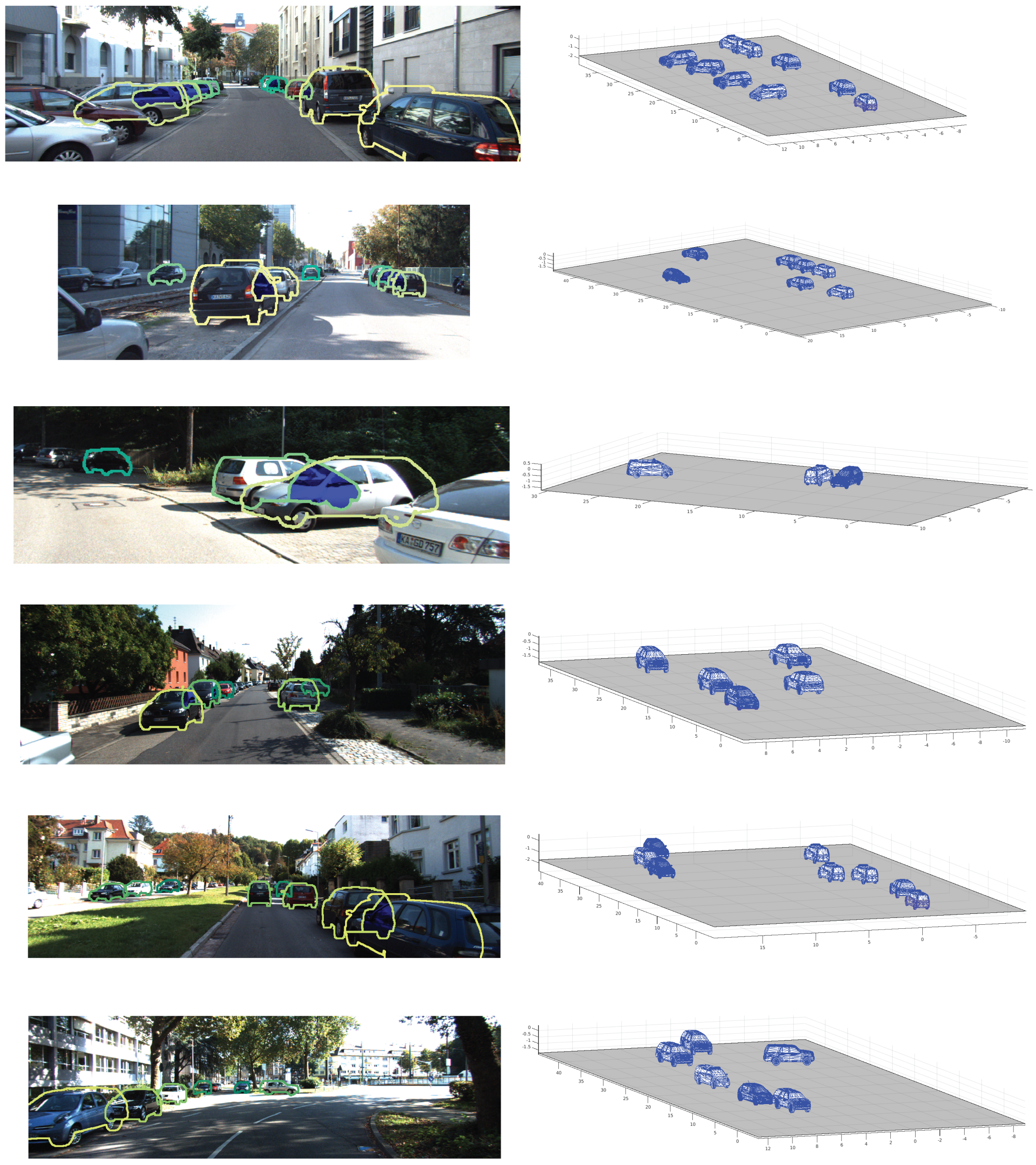


Figure 4.8: 2D recognition and 3D localization results on the KITTI test set. Blue regions in the images are the estimated occluded areas.

% occlusion	< 0.3	$0.3 - 0.6$	> 0.6
# images	66	68	66
ALM *Xiang and Savarese* (2012)	72.3	42.9	35.5
DPM *Felzenszwalb et al.* (2010)	75.9	58.6	44.6
SLM *Xiang and Savarese* (2013)	80.2	63.3	52.9
Ours NMS.5	89.7	76.3	55.9
Ours Occlusion	**90.0**	**76.5**	**62.1**

Table 4.5: AP of the car detection on the OutdoorScene dataset *Xiang and Savarese* (2013).

compared to the second best method) leveraging on the contextual occlusion relationship among objects. The large improvement in the *Hard* category verifies that our algorithm is capable of detecting challenging occluded objects. Notice that SubCat *Ohn-Bar and Trivedi* (2014) uses the same ensemble of ACF *Dollár et al.* (2014) detectors, but using 2D features in clustering. Fig. 4.8 shows some 2D recognition and 3D localization results on the KITTI test set (see the supplementary material for additional results).

4.4.4 Object Detection on the OutdoorScene Dataset

We apply our 227 3DVP detectors trained on the whole KITTI training set to the OutdoorScene dataset, and evaluate the object detection accuracy. Since the training and testing images are from different sources, we can test how well our 3DVP detectors trained on the KITTI dataset generalize to other scenarios, such as city and parking lot scenes in the OutdoorScene dataset. Table 4.5 shows the average precisions for car detection on the dataset, where the test images are partitioned into three different sets according to the amount of occlusion. Our 3DVP detectors outperform ALM *Xiang and Savarese* (2012), DPM *Felzenszwalb et al.* (2010) and SLM *Xiang and Savarese* (2013) on all the three partitions, which demonstrates the generalization capability of our 3DVP detectors. Similarly to our KITTI experiments, our occlusion reasoning algorithm further improves the detection accuracy in the largely occluded test set.

4.5 Conclusion

In this paper, we propose a novel 3D object representation, *3D Voxel Patterns*, that enables us to estimate detailed properties of objects beyond 2D bounding boxes, identify challenging occluded objects, and reason about the occlusion relationship between objects. The experimental evaluation demonstrates that our method can recognize objects in complex scenes with high accuracy, while providing detailed 2D/3D properties of the objects. The proposed occlusion reasoning method empowered by the properties further improves the recognition accuracy in various tasks. In addition, the experiment on OutdoorScene dataset confirms that our model generalizes well to different scenarios. Although the framework is evaluated on the "car" category, we believe that the idea of 3DVP is applicable to generic object categories. We consider generalize the method toward other object categories as a future direction.

CHAPTER V

PASCAL3D+: A Benchmark for 3D Object Recognition in the Wild

5.1 Introduction

In the past decade, several datasets have been introduced for classification, detection and segmentation. These datasets provide different levels of annotation for images ranging from object category labels *Fei-Fei et al.* (2007); *Deng et al.* (2009) to object bounding box *Ferrari et al.* (2010); *Everingham et al.* (2010); *Deng et al.* (2009) to pixel-level annotations *Shotton et al.* (2009); *Everingham et al.* (2010); *Xiao et al.* (2010). Although these datasets have had a significant impact on advancing image understanding methods, they have some major limitations. In many applications, a bounding box or segmentation is not enough to describe an object, and we require a richer description for objects in terms of their 3D pose. Since these datasets only provide 2D annotations, they are not suitable for training or evaluating methods that reason about 3D pose of objects, occlusion or depth.

To overcome the limitations of the 2D datasets, 3D datasets have been introduced *Savarese and Fei-Fei* (2007); *Ozuysal et al.* (2009); *Sun et al.* (2010); *Geiger et al.* (2012); *Matzen and Snavely* (2013). However, the current 3D datasets have a number of drawbacks as well. One drawback is that the background clutter is often limited

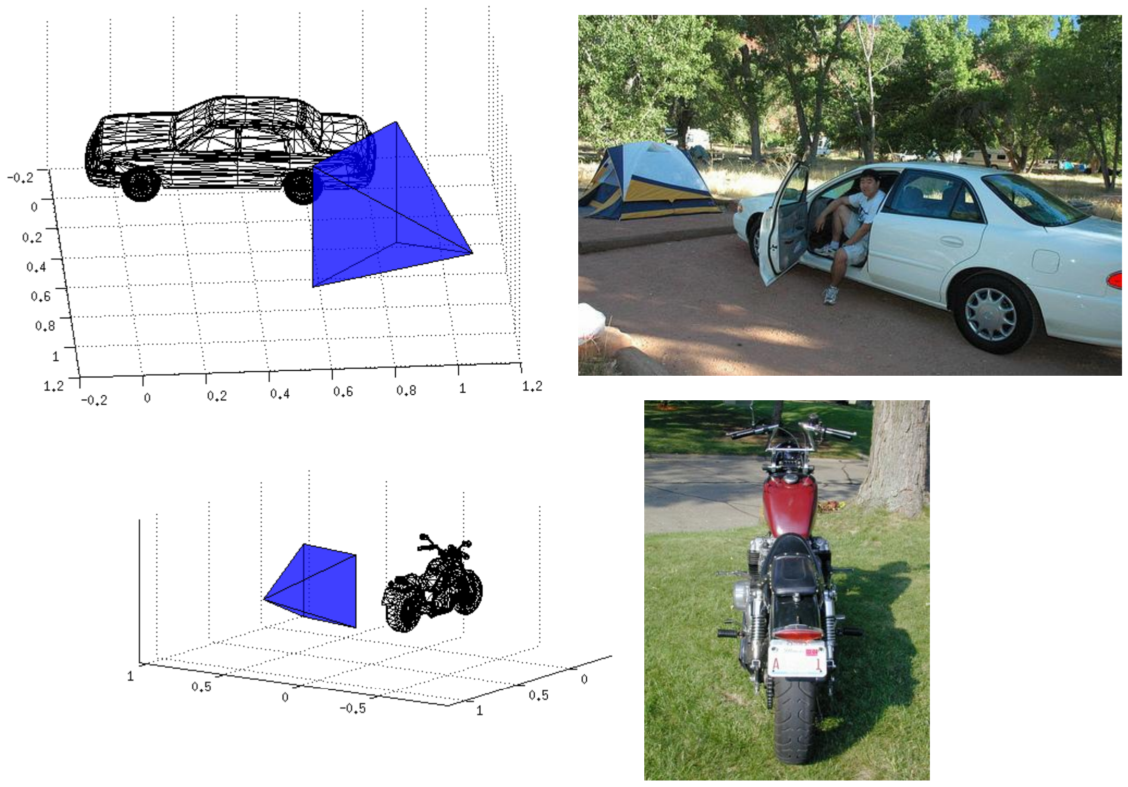


Figure 5.1: Example of annotations in our dataset. The annotators select a 3D CAD model from a pool of models and align it to the object in the image. Based on the 3D geometry of the model and the annotated 2D locations of a set of landmarks, we automatically compute the azimuth, elevation and distance of the camera (shown in blue) with respect to the object. Images are uncalibrated, so the camera can be at any arbitrary location.

	PASCAL3D+	ETH-80	MV	3DObject	EPFL Car
# of Categories	12	8	2	10	1
Avg. # Instances per Category	~3000	10	~140	10	20
Indoor(I) / Outdoor(O)	Both	I	Both	Both	I
Cluttered Background	✓	✗	✓	✗	✗
Non-centered Objects	✓	✗	✓	✗	✗
Occlusion Label	✓	✗	✗	✗	✗
Orientation Label	✓	✓	✓	✓	✓
Dense Viewpoint	✓	✗	✗	✗	✗
	ALM	KITTI	NYU Depth	NYC3DCars	IKEA
# of Categories	4	2	894	1	11
Avg. # Instances per Category	~660	80,000	39	3,787	~73
Indoor(I) / Outdoor(O)	Both	O	I	O	I
Cluttered Background	✓	✓	✓	✓	✓
Non-centered Objects	✗	✓	✓	✓	✓
Occlusion Label	✗	✓	✓	✓	✗
Orientation Label	✓	✓	✗	✓	✓
Dense Viewpoint	✓	✓	✗	✓	✓

Table 5.1: Comparison of our PASCAL3D+ dataset with some of the other 3D datasets: ETH-80 *Leibe and Schiele* (2003), MV *Thomas et al.* (2006), 3DObject *Savarese and Fei-Fei* (2007), EPFL Car *Ozuysal et al.* (2009), ALM *Xiang and Savarese* (2012), KITTI *Geiger et al.* (2012), NYU Depth *Silberman et al.* (2012), NYC3DCars *Matzen and Snavely* (2013), IKEA *Lim et al.* (2013).

and therefore methods trained on these datasets cannot generalize well to real-world scenarios, where the variability in the background is large. Another drawback is that some of these datasets do not include occluded or truncated objects, which again limits the generalization power of the relevant learnt models. Moreover, the existing datasets typically only provide 3D annotation for a few object classes and the number of images or object instances per category is usually small, which prevents the recognition systems from learning robust models for handling intra-class variations. Finally and most critically, most of these datasets supply annotations for a small number of viewpoints. So they are not suitable for object detection methods aiming at estimating continuous 3D pose, which is a key component in various scene understanding or robotics applications. In summary, it is necessary and important to have a challenging 3D benchmark which overcomes the above limitations.

Our contribution in this work is a new dataset, *PASCAL3D+*. Our goal is to overcome the shortcomings of the existing datasets and provide a challenging benchmark for 3D object detection and pose estimation. In PASCAL3D+, we augment the 12 rigid categories in the PASCAL VOC 2012 dataset *Everingham et al.* (2010) with 3D annotations. Specifically, for each category, we first download a set of CAD models from Google 3D Warehouse *Trimble*, which are selected in such a way that they cover the intra-class variability. Then each object instance in the category is associated with the closest CAD model in term of 3D geometry. Besides, several landmarks of these CAD models are identified in 3D, and the 2D locations of the landmarks are labeled by annotators. Finally, using the 3D-2D correspondences of the landmarks, we compute an accurate continuous 3D pose for each object in the dataset. As a result, the annotation of each object consists of the associated CAD model, 2D landmarks and 3D continuous pose. In order to make our dataset large scale, we add more images from ImageNet *Deng et al.* (2009) for each category. In total, more than 20,000 additional images with 3D annotations are included. Figure 5.1 shows some examples in our dataset. We also provide baseline results for object detection and pose estimation on our new dataset. The results show that there is still a large room for improvement, and this dataset can serve as a challenging benchmark for future visual recognition systems.

There are several advantages of our dataset: i) PASCAL images exhibit a great amount of variability and better mimic the real-world scenarios. Therefore, our dataset is less biased compared to datasets which are collected in controlled settings (e.g., *Savarese and Fei-Fei* (2007); *Ozuysal et al.* (2009)). ii) Our dataset includes *dense* and *continuous* viewpoint annotations. The existing 3D datasets typically discretize the viewpoint into multiple bins (e.g., *Leibe and Schiele* (2003); *Savarese and Fei-Fei* (2007)). iii) On average, there are more than 3,000 object instances per category. Hence, detectors trained on our dataset can have more generalization power. iv) Our dataset contains occluded and truncated objects, which are usually ignored in the current 3D datasets. v) Finally, PASCAL is the main benchmark for 2D object detection. We hope our efforts on providing 3D annotations to PASCAL can benchmark 2D and 3D object detection methods with a common dataset.

The next section describes the related work and other 3D datasets in the literature. Section 5.3 provides dataset statistics such as viewpoint distribution and variations in degree of occlusion . Section 5.4 describes the annotation tool and the challenges for annotating 3D information in an unconstrained setting. Section 5.5 explains the details of our baseline experiments, and Section 5.6 concludes the chapter.

5.2 Related Work

We review a number of commonly used datasets for 3D object detection and pose estimation. ETH-80 dataset *Leibe and Schiele* (2003) provides a multi-view dataset of 8 categories (e.g., fruits and animals), where each category contains 10 objects observed from 41 views, spaced equally over the viewing hemisphere. The background is almost constant for all of the images, and the objects are centered in the image. *Thomas et al.* (2006) introduces another multi-view dataset that includes *motorbike* and *sport shoe* categories in more challenging real-world scenarios. There are 179 images and 101 images corresponding to each category respectively. On average a motorbike is imaged from 11 views. For shoes, there are about 16 views around each instance taken at 2 different elevations. 3DObject dataset *Savarese and Fei-Fei* (2007) provides 3D annotations for 10 everyday object classes such as *car*, *iron*, and *stapler*. Each category includes 10 instances observed from different viewpoints. EPFL Car dataset *Ozuysal et al.* (2009) consists of 2,299 images of 20 car instances at multiple azimuth angles. The elevation and distance is almost the same for all of these instances. Table-Top-Pose dataset *Sun et al.* (2010) contains 480 images of 3 categories (*mouse*, *mug*, and *stapler*), where each consists of 10 instances under 16 different poses.

These datasets exhibit some major limitations. Firstly, most of them have more or less clean background. Therefore, methods trained on them will not be able to handle cluttered background, which is common in real-world scenarios. Secondly, these datasets only include a limited number of instances, which makes it difficult for recognition methods to learn intra-class variations. To overcome these issues, more challenging datasets have been proposed. ICARO *Lopez-Sastre et al.* (2010) contains viewpoint annotations for 26 object categories. However, the viewpoints are sparse and not densely annotated. *Xiang and Savarese* (2012) provides 3D pose annotations for a subset of 4 categories of the ImageNet dataset *Deng et al.* (2009): *bed* (400 images), *chair* (770 images), *sofa* (800 images) and *table* (670 images). Since the ImageNet dataset is mainly designed for the classification task, the objects in the dataset are usually not occluded and they are roughly centered. The KITTI dataset *Geiger et al.* (2012) provides 3D labeling for two categories (*car* and *pedestrian*), where there are 80K instances per category. The images of this dataset are limited to street scenes, and all of the images have been obtained by cameras mounted on top of a car. This may pose some issues concerning the ability to generalize to other scene types. More recently, NYC3DCars dataset *Matzen and Snavely* (2013) has been introduced, which contains information such as 3D vehicle annotations, road segmentation and direction of movement. Although the imagery is unconstrained for this dataset in terms of camera type or location, the images are constrained to street scenes of New York. Also, the dataset contains only one category. *Lim et al.* (2013) provides dense 3D annotations for some of the IKEA objects. Their dataset is also limited to indoor images and the number of instances per category is small.

Simultaneous use of 2D information and 3D depth makes the recognition systems more powerful. Therefore, various datasets have been collected by RGB-D sensors (such as Kinect). RGB-D Object Dataset *Lai et al.* (2011) contains 300 physically distinct objects organized into 51 categories. The images are captured in a controlled

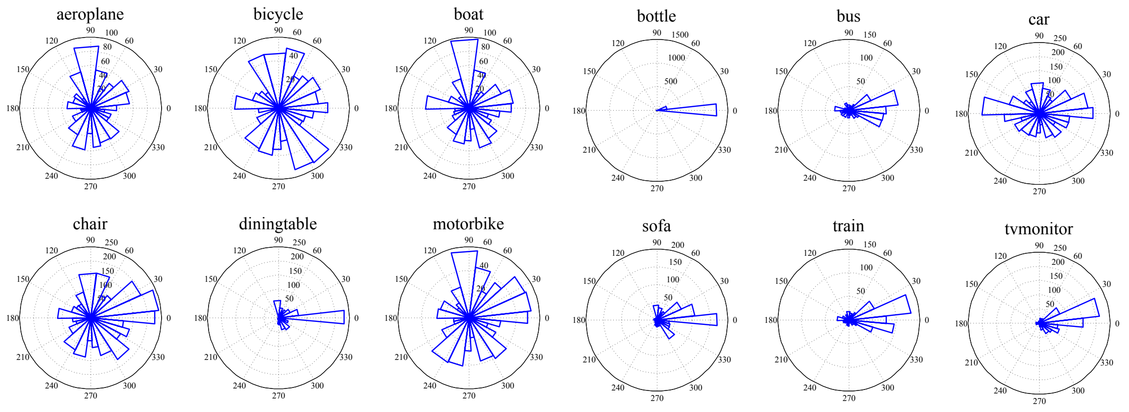


Figure 5.2: **Azimuth distribution.** Polar histograms show the distribution of azimuth among the PASCAL images for each object category.

setting and have a clean background. Berkeley 3-D Object Dataset *Janoch et al.* (2013) provides annotation for 849 images of over 50 classes in real office environments. NYU Depth *Silberman et al.* (2012) includes 1,449 densely labeled pairs of aligned RGB and depth images. The dataset includes 35,064 distinct instances, which are divided into 894 classes. SUN3D *Xiao et al.* (2013) is another dataset of this type, which provides annotations for scenes and objects. There are three limitations for these types of datasets that make them undesirable for 3D object pose estimation: i) They are limited to indoor scenes as the current common RGB-D sensors have a limited range. ii) They do not provide the orientation for objects (they just provide the depth). iii) Their average number of images per category is small.

Our goal for providing a novel dataset is to eliminate the mentioned shortcomings of other datasets, and enhance 3D object detection and pose estimation methods by training and evaluating them on a challenging and real world benchmark. Table 5.1 shows a comparison between our dataset and some of the most relevant datasets mentioned above.

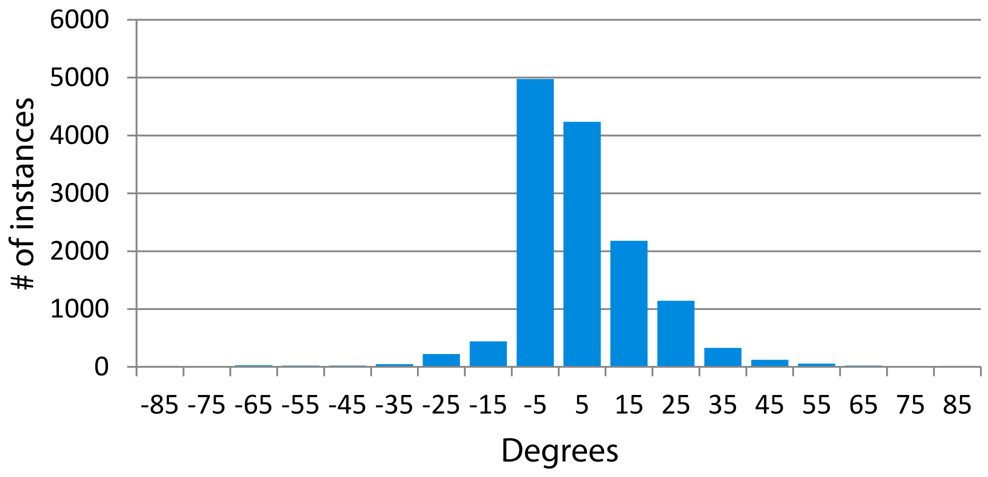


Figure 5.3: **Elevation distribution.** The distribution of elevation among the PASCAL images across all the categories.

5.3 Dataset Details and Statistics

We describe the details of our PASCAL3D+ dataset and provide some statistics. We annotated the 3D pose densely for all of the object instances in the `trainval` subset of PASCAL VOC 2012 detection challenge images (including instances labeled as 'difficult'). We consider the 12 rigid categories of PASCAL VOC, since estimating the pose of the deformable categories is still an open problem. These categories are *aeroplane*, *bicycle*, *boat*, *bottle*, *bus*, *car*, *chair*, *diningtable*, *motorbike*, *sofa*, *train* and *tvmonitor*. In total, there are 13,898 object instances that appear in 8,505 PASCAL images. Furthermore, we downloaded 22,394 images from ImageNet *Deng et al.* (2009) for the 12 categories. For the ImageNet images, the objects are usually centered without occlusion and truncation. On average, there are more than 3,000 instances per category in our PASCAL3D+ dataset.

The annotation of an object contains the azimuth, elevation and distance of the camera pose in 3D (we explain how the annotation is obtained in the next section). Moreover, we assign a visibility state to landmarks that we identify for each category: 1) **visible**: the landmark is visible in the image. 2) **self-occluded**: the landmark is

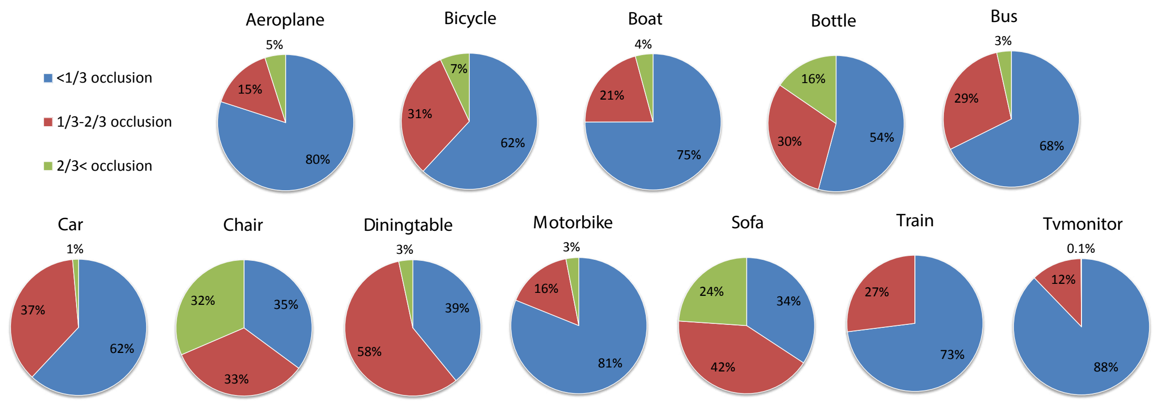
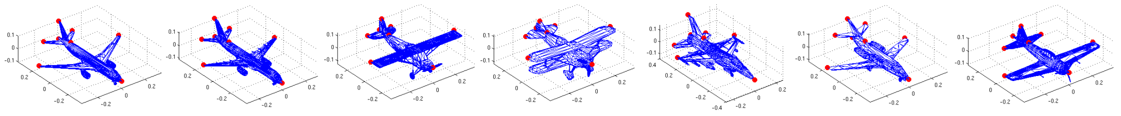


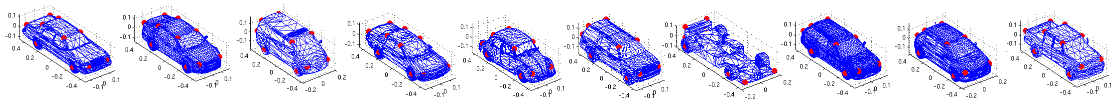
Figure 5.4: **Occlusion distribution.** The distribution of object instances based on the degree of occlusion in the PASCAL images.

not visible due to the 3D geometry and the pose of the object. 3) **occluded-by**: the landmark is occluded by an external object. 4) **truncated**: the landmark appears outside the image area. 5) **unknown**: none of the above four states. To ensure high quality labeling, we hired annotators for the annotation instead of posting the task on crowd-sourcing platforms.

Figure 5.2 shows the distribution of azimuth among the PASCAL images for the 12 categories, where azimuth 0° corresponds to the frontal view of the object. As expected, the distribution of viewpoints is biased. For example, very few images are taken from the back view (azimuth 180°) of *sofa* since the back of sofa is usually against a wall. For *tvmonitor*, there is also a high bias towards the frontal view. Since *bottles* are usually symmetric, the distribution is dominated by azimuth angles around zero. The distribution of elevation among the PASCAL images across all categories is shown in Figure 5.3. It is evident that there is large variability in the elevation as well. These statistics show that our dataset has a fairly good distribution in pose variation.

We also analyze the object instances based on their degree of occlusion. The statistics in Figure 5.4 show that PASCAL3D+ is quite challenging as it includes object instances with different degrees of occlusion. The main goal of most previous


(a) Aeroplane


(b) Car

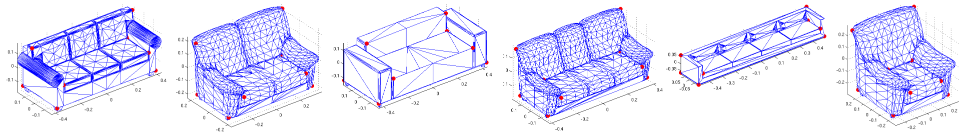

(c) Sofa

Figure 5.5: Examples of 3D CAD models used for annotation. To better capture intra-class variability of object categories, different types of CAD models are chosen. The red points represent the identified landmarks.

3D datasets was to provide a benchmark for object pose estimation. So they usually ignored occluded or truncated objects. However, handling occlusion and truncation is important for real world applications. Therefore, a challenging dataset like ours can be useful. In Figure 5.4, we divide the object instances into three classes based on the ratio of their externally occluded or truncated landmarks to all landmarks (0 to 1/3, 1/3 to 2/3 and above 2/3). The instances of some categories such as *chair* or *diningtable* are highly occluded, which poses a big challenge to the existing object detection and pose estimation methods.

5.4 3D Annotation

Providing 3D annotations for unconstrained images is not trivial since only a single image of a scene is available and the camera parameters are unknown. We explain the details of our annotation tool and the procedure for 3D annotation labeling.

For each category, we downloaded 3D CAD models from Google 3D Warehouse *Trimble*, which is a public repository for 3D CAD models. We select the CAD models in such a way that they represent intra-class variations of a particular category. For

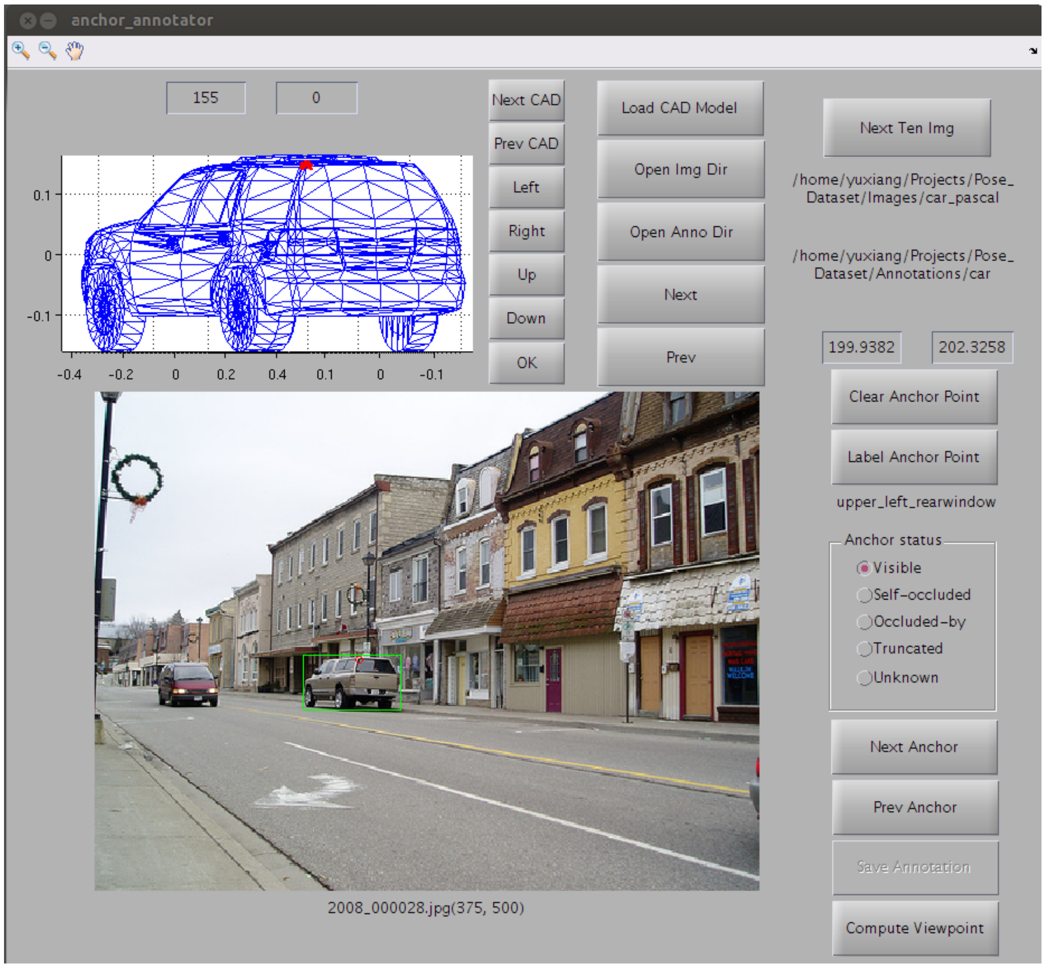


Figure 5.6: A snapshot of our annotation tool. The blue mesh is the 3D CAD model chosen by the annotator, and the red circle corresponds to one of the landmarks.

example, we select *SUV*, *sedan*, *hatchback*, etc., for the car category. For the aeroplane category, we choose *airliner*, *fighter*, *propeller*, and so on. The 3D CAD models for two example categories are shown in Figure 5.5. For a sub-category (e.g., propeller aeroplane), more than one CAD model can be selected to better capture the variations in the sub-category.

For each CAD model, we identify a set of landmarks, which are shown with red circles in Figure 5.5. The landmarks are chosen such that they are shared among instances in a category and can be identified easily in the images. Most of the landmarks correspond to the corners in the CAD models. The task of annotators is to select the closest CAD model for an object instance in terms of 3D geometry and label the landmarks of the CAD model on the 2D image. Then we use these 2D annotations of the landmarks and their corresponding locations on the 3D CAD models to find the azimuth, elevation and distance of the camera pose in 3D for each object instance. A visualization of our annotation tool is shown in Figure 5.6. The annotator first selects the 3D CAD model that best resembles the object instance. Then, he/she rotates the 3D CAD model until it is aligned with the object instance visually. The alignment provides us with rough azimuth and elevation angles, which are used as initialization in computing the continuous pose. Based on the 3D geometry and the rough pose of the CAD model (after alignment), we compute the visibility of the landmarks. After this step, we show the visible (not self-occluded) landmarks on the 3D CAD model one by one and ask the annotator to mark their corresponding 2D location in the image. For occluded or truncated landmarks, the annotator provides its visibility status as explained in Section 5.3.

As the result of the annotation, for each object instance in the dataset, we obtain the correspondences between 3D landmarks $\mathbf{X}$ on the CAD model and their 2D projection $\mathbf{x}$ on the image. By using a pinhole camera model, the relationship between the 2D and 3D points is given by: $\mathbf{x}_i = K[R|\mathbf{t}]\mathbf{X}_i$, where K is the intrinsic camera matrix, and R and $\mathbf{t}$ are the rotation matrix and the translation vector respectively. We use a virtual intrinsic camera matrix K, where the focal length is assumed to be 1, the skew is 0 and the aspect ratio is 1. We assume a simplified camera model, where the world coordinate is defined on the 3D CAD model and the camera is facing the origin of the world coordinate system. In this case, R and $\mathbf{t}$ are determined by the azimuth, elevation and distance of the camera pose in 3D. So we can minimize the re-projection error of the 3D landmarks to obtain the continuous pose of the object:

$$\min_{R,\mathbf{t}} \sum_{i=1}^{L} ||\mathbf{x}_i - \tilde{\mathbf{x}}_i||_2, \tag{5.1}$$

where L is the number of visible landmarks and $\tilde{\mathbf{x}}_i$ is the annotated landmark location in the image. By solving the minimization problem (5.1), we can find the rotation matrix R and the translation vector $\mathbf{t}$, which provide the azimuth, elevation and distance of the object pose. This is the well-studied Perspective-n-Points (PnP) problem for which various solutions (e.g., *Lu et al.* (2000); *Ansar and Daniilidis* (2003); *Lepetit et al.* (2009)) exist. We use the constrained non-linear optimization implementation of MATLAB to solve (5.1). For degenerate cases, where there are not enough landmarks visible to compute the pose (less than 2 landmarks), we use the rough azimuth and elevation specified by the annotator instead.

5.5 Baseline Experiments

In this section, we provide baseline results in terms of object detection, viewpoint estimation and segmentation. We also show that how well the baseline method can handle different degrees of occlusion. For all the experiments below, we use the `train` subset of PASCAL VOC 2012 (detection challenge) for training and the `val` subset for evaluation. We adapt DPM *Felzenszwalb et al.* (2010) (`voc-release4.01`) to joint object detection and viewpoint estimation.

	DPM	VDPM-4V	VDPM-8V	VDPM-16V	VDPM-24V
aeroplane	42.2 / –	40.0 / 34.6	39.8 / 23.4	43.6 / 15.4	42.2 / 8.0
bicycle	49.6 / –	45.2 / 41.7	47.3 / 36.5	46.5 / 18.4	44.4 / 14.3
boat	6.0 / –	3.0 / 1.5	5.8 / 1.0	6.2 / 0.5	6.0 / 0.3
bottle	49.6 / –	– / –	– / –	– / –	– / –
bus	54.1 / –	49.3 / 26.1	50.2 / 35.5	54.6 / 46.9	53.7 / 39.2
car	38.3 / –	37.2 / 20.2	37.3 / 23.5	36.6 / 18.1	36.3 / 13.7
chair	15.0 / –	11.1 / 6.8	11.4 / 5.8	12.8 / 6.0	12.6 / 4.4
diningtable	9.0 / –	7.2 / 3.1	10.2 / 3.6	7.6 / 2.2	11.1 / 3.6
motorbike	33.1 / –	33.0 / 30.4	36.6 / 25.1	38.5 / 16.1	35.5 / 10.1
sofa	18.9 / –	6.8 / 5.1	16.0 / 12.5	16.2 / 10.0	17.0 / 8.2
train	36.4 / –	26.4 / 10.7	28.7 / 10.9	31.5 / 22.1	32.6 / 20.0
tvmonitor	33.2 / –	35.9 / 34.7	36.3 / 27.4	35.6 / 16.3	33.6 / 11.2
Average	29.6 / –	26.8 / 19.5	29.9 / 18.7	30.0 / 15.6	29.5 / 12.1

Table 5.2: The results of DPM and VDPM are shown. The first number indicates the Average Precision (AP) for detection and the second number shows the Average Viewpoint Precision (AVP) for joint object detection and pose estimation.

5.5.1 Detection and Viewpoint Estimation

The original DPM method uses different mixture components to capture pose and appearance variations of objects. The object instances are assigned to these mixture components based on their aspect ratios. Since the aspect ratio does not necessarily correspond to the viewpoint, viewpoint estimation with the original DPM is impractical. Therefore, we modify DPM similar to *Lopez-Sastre et al.* (2011) such that each mixture component represents a different azimuth section. We refer to this modified version as Viewpoint-DPM (VDPM). In the original DPM, half of the mixture components are mirrored versions of the other half. So the training images are mirrored and assigned to the mirror mixture components. Similarly, we mirror the training images and assign them to the mirrored viewpoint components in VDPM. Another way to perform joint object detection and pose estimation is to treat it as a structure labeling problem. In Pepik et al. *Pepik et al.* (2012), they utilize structural SVM to predict the object bounding box and pose jointly, where the model is called DPM-VOC+VP. In our baseline experiments, we divide the azimuth angles into 4, 8,

	DPM-VOC+VP-4V	DPM-VOC+VP-8V
aeroplane	41.5 / 37.4	40.5 / 28.6
bicycle	46.9 / 43.9	48.1 / 40.3
boat	0.5 / 0.3	0.5 / 0.2
bottle	–/–	–/–
bus	51.5 / 48.6	51.9 / 38.0
car	45.6 / 36.9	47.6 / 36.6
chair	8.7 / 6.1	11.3 / 9.4
diningtable	5.7 / 2.1	5.3 / 2.6
motorbike	34.3 / 31.8	38.3 / 32.0
sofa	13.3 / 11.8	13.5 / 11.0
train	16.4 / 11.1	21.3 / 9.8
tvmonitor	32.4 / 32.2	33.1 / 28.6
Avarage	27.0 / 23.8	28.3 / 21.5

	DPM-VOC+VP-16V	DPM-VOC+VP-24V
aeroplane	38.0 / 15.9	36.0 / 9.7
bicycle	45.6 / 22.9	45.9 / 16.7
boat	0.7 / 0.3	5.3 / 2.2
bottle	–/–	–/–
bus	55.3 / 49.0	53.9 / 42.1
car	46.0 / 29.6	42.1 / 24.6
chair	10.2 / 6.1	8.0 / 4.2
diningtable	6.2 / 2.3	5.4 / 2.1
motorbike	38.1 / 16.7	34.8 / 10.5
sofa	11.8 / 7.1	11.0 / 4.1
train	28.5 / 20.2	28.2 / 20.7
tvmonitor	30.7 / 19.9	27.3 / 12.9
Avarage	28.3 / 17.3	27.1 / 13.6

Table 5.3: The results of DPM-VOC+VP *Pepik et al.* (2012) are shown. The first number indicates the Average Precision (AP) for detection and the second number shows the Average Viewpoint Precision (AVP) for joint object detection and pose estimation.

16 and 24 sections and train VDPM and DPM-VOC+VP models for each case.

To evaluate object detection, we use Average Precision (AP) as the metric and use the standard 50% overlap criteria of PASCAL VOC *Everingham et al.* (2010). For viewpoint estimation, the commonly used metric is the average over the diagonal of the viewpoint confusion matrix *Savarese and Fei-Fei* (2007). However, this metric only considers the viewpoint accuracy among the correctly detected objects, which makes it non-comparable for two detectors with different sets of detected objects. Since viewpoint estimation is closely related to detection, we need a metric for joint detection and pose estimation. We propose a novel metric called Average Viewpoint Precision (AVP) for this propose similar to AP in object detection. In computing AVP, an output from the detector is considered to be correct if and only if the bounding box overlap is larger than 50% *AND* the viewpoint is correct (i.e., the two viewpoint labels are the same in discrete viewpoint space or the distance between the two viewpoints is smaller than some threshold in continuous viewpoint space). Then we can draw a Viewpoint Precision-Recall (VPR) curve similar to the PR curve. Average viewpoint precision is defined as the area under the VPR curve. Therefore, AVP is the metric for joint detection and pose estimation. Note that detection PR curve is always an upper bound of the VPR curve. Small gap between AVP and AP indicates high viewpoint accuracy among the correctly detected objects.

The results of the original DPM with 6 mixture components, VDPM and DPM-VOC+VP *Pepik et al.* (2012) for different azimuth sections are shown in Table 5.2 and Table 5.3. Since the instances of the *bottle* category are often symmetric across different azimuth angles, it is ignored in VDPM and DPM-VOC+VP. The detection performance of VDPM is on par with DPM. Compared with VDPM, DPM-VOC+VP achieves better viewpoint estimation in a tradeoff of slightly lower detection performance. For most categories, as we increase the number of viewpoints, the viewpoint estimation task becomes harder and the AVP reduces, which is not surprising. We

	0–1/3	**1/3–2/3**	**2/3–max**
aeroplane	57.2	11.5	16.2
bicycle	70.6	30.4	8.7
boat	13.1	0.7	0.9
bus	77.4	35.7	4.1
car	55.3	12.3	3.4
chair	22.0	7.5	0.9
diningtable	33.3	19.9	7.8
motorbike	56.5	12.6	0.1
sofa	35.3	34.2	15.8
train	50.2	35.2	15.3
tvmonitor	58.0	8.1	2.2
Avg.	48.1	18.9	6.8

Table 5.4: The Normalized Average Precisions from VDPM with 8 views for object detection at different degrees of occlusion.

can see from Table 5.2 and Table 5.3 that there is still a large room for improvement both in detection and pose estimation on our dataset. Hence, our 3D annotations can be valuable for developing new 3D object detection methods.

5.5.2 Sensitivity of Detection to Occlusion

Since our dataset provides occlusion labels for landmarks, we can analyze the performance of detection at different degrees of occlusion. The occlusion of landmarks does not directly determine the degree of occlusion of the object, but it has a strong correlation with it. For example, all landmarks can be occluded while most of the object can be observed, but such a case does not happen in reality. Therefore, we use the ratio of externally occluded or truncated landmarks to all landmarks as a measure for the degree of occlusion. We refer to it as the "occlusion ratio". In this experiment, we analyze the detection performance of VDPM with 8 views in terms of different degree of occlusion. We partition the instances into three occlusion sets, i.e., the set with occlusion ratio between 0 and 1/3, the set with occlusion ratio between 1/3 and 2/3, and the set with occlusion ratio larger than 2/3. Since the number of instances in each occlusion set is different, we report Normalized Average Precision

	GT CAD	Random CAD	VDPM-4V	VDMP-8V	VDPM-16V	VDPM-24V
aeroplane	43.8	32.8±0.3	22.6	24.1	24.7	24.5
bicycle	28.7	29.2±0.5	16.1	16.6	16.6	16.9
boat	43.0	28.7±1.1	23.4	23.5	23.5	20.5
bottle	66.0	62.5±1.1	–	–	–	–
bus	78.4	67.2±0.8	50.7	52.7	57.8	57.1
car	67.3	61.8±0.5	51.2	51.2	51.9	50.9
chair	41.8	35.8±0.8	25.7	27.6	26.5	27.2
diningtable	28.0	21.3±0.6	12.4	10.8	10.1	11.5
motorbike	60.0	54.6±0.3	34.4	35.7	37.9	37.3
sofa	40.3	34.7±0.5	27.3	29.4	29.5	27.6
train	59.2	53.8±0.6	35.1	40.2	40.2	39.8
tvmonitor	72.3	60.5±2.8	56.6	55.0	55.9	54.7
Avg.	52.4	45.2	32.3	33.3	34.1	33.5

Table 5.5: Segmentation accuracy obtained by projecting the 3D CAD models onto the images. Please refer to the text for more details.

in Table 5.4 as suggested by *Hoiem et al.* (2012). It is evident that the detectors have difficulty in handling highly occluded objects. In order to achieve good performance in detection and pose estimation on our dataset, it is important to handle the occluded and truncated objects. Our dataset enables evaluation of occlusion reasoning as well.

5.5.3 Segmentation using 3D Pose

We show that estimating the viewpoint with the corresponding CAD model for an object enables object segmentation. To find the upper bound for segmentation in this way, we project the ground truth CAD model (the one that the annotator selected for the object instance) onto the image using the ground truth azimuth, elevation and distance. To evaluate the segmentation, we use the annotations provided by *Hariharan et al.* (2011). The first row of Table 5.5 shows the segmentation accuracy using the ground truth poses, where we use the standard PASCAL evaluation metric for segmentation. The accuracy is not 100% due to several reasons. First, we do not consider occlusion reasoning in the projection, and the ground truth mask from *Hariharan et al.* (2011) is just for the visible part of the object. Second, due to the

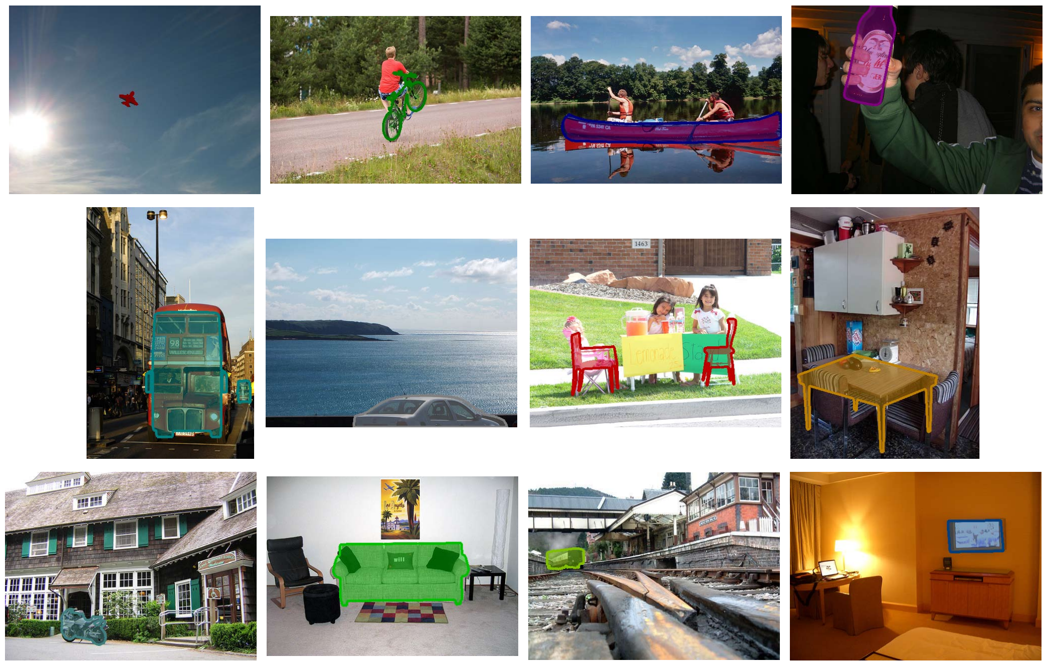


Figure 5.7: Segmentation results obtained by projecting the 3D CAD models to the images. Each figure shows an example for one of the 12 categories in our dataset.

simplified camera model in computing the continuous pose and the limited number of CAD models in our dataset, the projection matrix we use is an approximation to the real one. So we also include the re-projection error in our 3D annotation, which can be considered to be a measure for the quality of the annotation. Figure 5.7 shows segmentation examples for each category in our dataset using the ground truth pose. As an example of the re-projection error, the predicted legs of the diningtable are not precisely aligned with the object in the image, which results in a large penalty in the computing the segmentation accuracy. For the chairs, a large penalty is introduced due to occlusion. Occlusion reasoning is also important for segmentation.

To show the importance of using the right CAD model for annotation, instead of projecting the ground truth CAD model, we project a randomly chosen model (from the set of CAD models for a particular category) and evaluate the segmentation performance. As shown in the second row of Table 5.5, the average accuracy drops by about 7%. The shown accuracy is the average over 5 different random selections.

Note that the performance for *bicycle* with random models is higher than the case with the ground truth models. This is due to the inaccuracy in 2D segmentation annotation of bicycle. In most cases, the areas that correspond to the background are labeled as bicycle (e.g., around the spokes).

We also evaluate how well the automatic approaches can perform segmentation. In this experiment, we infer the azimuth automatically from VDPMs, but use the ground truth elevation, distance and CAD model in the projection. More specifically, for each detected object, we project the CAD model to the image. We consider an object as detected if there is a bounding box with more than 50% intersection over union overlap associated with it. The performance drops significantly for the automatic approach. Note that the segmentation performance becomes better as we use finer discretization of azimuth (with the exception of 24 viewpoints). The low performance with 24 views might be due to the low performance of VDPM in viewpoint estimation for 24 views as shown in Table 5.2.

5.6 Conclusion

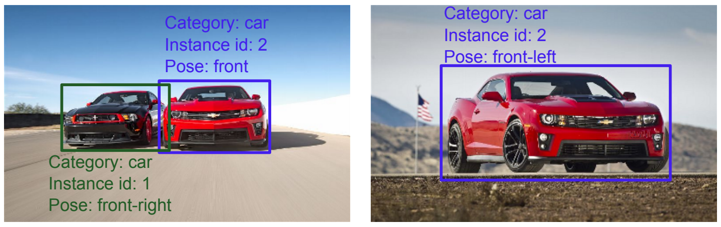
To further improve the development of 3D object detection and pose estimation methods, we provide a large scale benchmark PASCAL3D+ with 3D annotations of objects. PASCAL3D+ overcomes the limitations of the existing 3D datasets and better matches real-world scenarios. We developed an algorithm and annotation tool to provide the continuous 3D viewpoint annotations in unconstrained settings, where the camera parameters are unknown and only a single image of object instances is available. We also provide baseline results for object detection, viewpoint estimation and segmentation on our PASCAL3D+ dataset. The results illustrate that there is still a large room for improvement in all these tasks. We hope our dataset can push forward the research in 3D object detection and pose estimation.

CHAPTER VI

Application I: Object Co-detection with 3D Aspect Parts

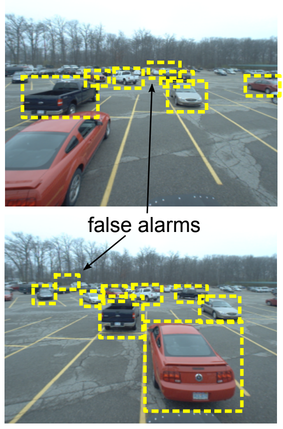
6.1 Introduction

We introduce a framework for solving a new problem called *object co-detection*. Given multiple images, each of which may contain object instances of a given category observed from different viewpoints, the goal of co-detection is to: 1) detect objects in all images; 2) recognize whether or not objects in different images correspond to the same instance – we refer to these object instances as *matching objects*; 3) estimate the viewpoint transformation between matching objects. Fig. 6.1 illustrates co-detection in two images. Fig. 6.1(a) shows two instances of the car category: a black Ford

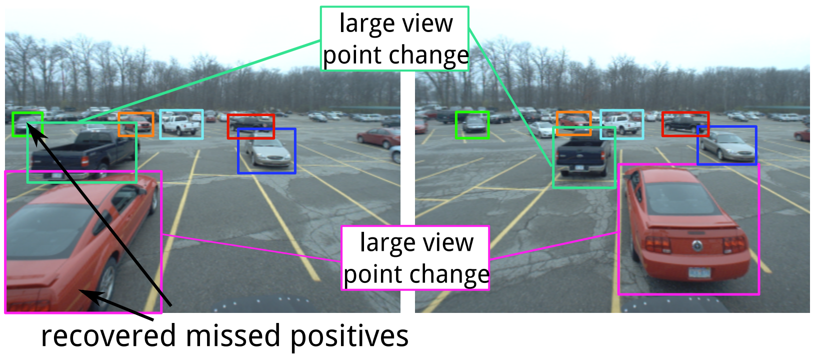


(a) Input Image #1 (b) Input Image #2

Figure 6.1: Object co-detection for two images. The goal is to i) detect objects; ii) identify which objects correspond to the same object instance (e.g. the red Camaro); we call these instances *matching objects*; iii) estimate the viewpoint transformation between matching objects.



(a) Single image object detection. Notice miss positives and false alarms.



(b) Object co-detection. Different colors correspond to different matching objects. Co-detection recovers missed positives and removes false alarms, compared to single image object detection (Fig. 6.2(a)).

Figure 6.2: Object co-detection improves object detection and matches objects.

Mustang and a red Chevy Camaro. Fig. 6.1(b) also contains a red Camaro, which is considered to be the matching object to the Camaro in Fig. 6.1(a). Through the process of co-detection, the two Camaro detections are matched and the viewpoint transformation between the two instances is estimated. The black Mustang is kept as a detection, but it has no matched object in the other image.

An important property that motivates the introduction of the co-detection paradigm is its ability to obtain superior detection results over conventional single-image detection schemes. We argue that, by leveraging on the fact that an object has consistent appearance when observed from the same or different viewpoints, a co-detector is capable of obtaining more accurate detection results than if objects were to be detected from each image individually. Consider the example in Fig. 6.2(a), the red car appears in both images. This car is successfully detected by a state-of-the-art detector *Felzenszwalb et al.* (2010) in Fig.6.2(a)-bottom, but it is not in Fig.6.2(a)-top. Our co-detector has the ability to recover the missed detection by leveraging the fact that the same car instance is detected in the other image, and that appearance and shape of the car must be consistent across the two images (up to a viewpoint transformation). If the car instance appears in only one of the images, the co-detector is equivalent to a single image detector. Notice that a co-detector can be applied to an arbitrary number of images (not just two).

Object co-detection is far from being a trivial problem. An object instance may have a dramatically varied appearance due to viewpoint transformations and self-occlusions (parts of the object are only visible from some viewpoints). Moreover, the background surrounding the object may also vary, which makes the naive object matching methods unstable (e.g. by matching bounding boxes via image features). Furthermore, object co-detection requires the simultaneous solution of two already difficult problems: object detection and pose estimation. State-of-the-art methods that address these problems still have much room for improvement.

In this work, we propose a novel framework for object co-detection. Our method jointly detects and matches objects by their parts. To represent an object category by parts, our model leverages existing part-based object representation models (e.g. *Felzenszwalb et al.* (2010); *Xiang and Savarese* (2012)). One possible object representation is shown in Fig. 6.4(a). We measure appearance consistency between objects by matching their parts (Fig. 6.4(b)). Compared with a holistic object representation *Dalal and Triggs* (2005), a part-based object representation is more robust to viewpoint changes and self-occlusions. We combine information from multiple images by introducing an energy based formulation that models both the object's category-level appearance similarity in each image and the instance's appearance consistency across images. We also propose a novel matching potential function to handle large viewpoint transformations and self-occlusions in the part matching process.

The main contributions of this paper include: 1) a general framework for object co-detection, which allows us to detect matching objects from two or multiple images without any knowledge on the viewpoint geometry; 2) a novel energy function and a matching potential function to model the object visual appearances both within images and across images; 3) extensive experimental evaluation on three public datasets – a car dataset *Bao and Savarese* (2011), a pedestrian dataset *Ess et al.* (2007), and a 3D object dataset *Savarese and Fei-Fei* (2007). Compared with alternative state-of-the-art methods, the proposed framework can improve both the detection and pose estimation accuracy, as well as match object instances more robustly.

6.2 Related Work

Co-detection is related with and potentially useful to several other problems in computer vision:

Object detection. Given an object category model, methods such as *Dalal and Triggs* (2005); *Leibe et al.* (2004); *Su et al.* (2009); *Felzenszwalb et al.* (2010); *Gu and Ren* (2010); *Xiang and Savarese* (2012) identify an object of such category from an input image. Co-detection is a generalization of standard object detection in that it handles multiple input images which contain the same objects. If an object instance is only present in one image, a co-detector degenerates into a standard object detector. Otherwise, a co-detector leverages object appearance and shape consistency to improve object detection accuracy. Furthermore, a co-detector can discover matching instances.

Single instance 3D object detection. Given a 2D or 3D model of an object instance, methods such as *Lowe* (1999); *Ferrari et al.* (2006); *Rothganger et al.* (2006); *Hsiao et al.* (2010) detect the same object instance from a query image. Particularly, in *Lowe* (1999); *Ferrari et al.* (2006), the object model is just a single training image and the object (which is possibly observed from a different viewpoint) is identified in the query image by matching features or aggregations of features. Object co-detection provides a framework for potentially incorporating the same appearance matching constraints as in *Lowe* (1999); *Ferrari et al.* (2006), and it does not require the identification of the object location in the training image (object locations can be unknown)

Image co-segmentation. Given multiple images containing similar foreground objects, methods such as *Rother et al.* (2006); *Batra et al.* (2010); *Hochbaum and Singh* (2009) perform pixel-level segmentation of the shared foreground objects. Most co-segmentation methods only depend on low-level image appearance information, and hence tend to fail if the object appearance changes because of viewpoint transformations. Furthermore, most co-segmentation methods do not attempt to recognize the object identity and cannot cope with multiple object instances in the same image. On the contrary, a co-detector is designed to detect an arbitrary number of object categories per image and associate a category label to each detection. Moreover, co-detection is designed to handle large viewpoint transformations across images.

Tracking by detection. To solve this problem *Wu and Nevatia* (2005); *Ess et al.* (2008); *Choi and Savarese* (2010), correspondences of object detections must be established across frames in order to form tracklets. Unlike co-detection, in these works detections are obtained independently from each frame and subsequently matched. By jointly detecting the same object instance from all the frames, a co-detection framework could potentially improve the tracklet quality and help make tracking by detection more robust.

Semantic structure from motion (SSFM). Given multiple views of a scene, SSFM methods such as *Bao and Savarese* (2011); *Bao et al.* (2012b); *Zia et al.* (2011) use high level semantic information to help estimate the camera viewpoint changes. In turn, object detection accuracy is improved by leveraging the estimated camera pose geometry. A co-detection method could play a critical role in a SSFM framework in that it can establish matches of objects across views without using camera information (external and internal parameters).

Single instance matching. Given an image of an object instance (e.g a music CD cover), the goal is to retrieve the same object instance from a large collection of images. Methods such as *Nister and Stewenius* (2006); *Berg et al.* (2005); *Lowe* (2004) usually evaluate the similarity based on the whole image and thus require that the image only contains one dominating object. Conversely, our object co-detection is capable of identifying and matching the objects of interest and discarding uninformative background clutter.

Region matching. Methods such as *Matas et al.* (2004); *Toshev et al.* (2007) match features or regions across views of the same scene. Co-detection is fundamentally different in that it works with high level semantics (i.e. objects). However, co-detection can be helpful for those algorithms since it provides high level contextual information for pruning out false feature or region matches.

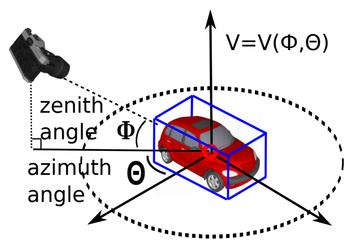
6.3 Object Co-detection Model

In an object co-detection problem, we are given a total number of K input images $\mathcal{I} = \{I^1, \ldots, I^K\}$. The goal of the co-detector is to detect the matching instances $\mathcal{O} = \{O^1, \ldots, O^K\}$ that simultaneously appear in each of the input image, where O^k is an object instance in image I^k.

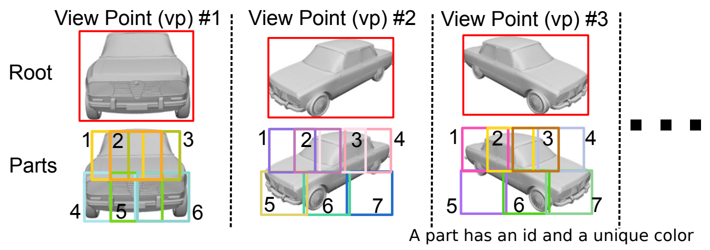
6.3.1 Object Representation

In our co-detection model, we adopt a part-based object representation. An object O in an image is represented by a root r, a number of n parts $\mathcal{P} = \{p_1, \ldots, p_n\}$, and a viewpoint V, i.e., $O = (r, \mathcal{P}, V)$. We explore two types of object representations: 2D part representation and 3D part representation.

In a 2D representation such as *Felzenszwalb et al.* (2010), the root and parts are specified by rectangles in the image (Fig. 6.3(b)). Since different parts are defined for different viewpoints independently, no explicit part correspondence can be established across different viewpoints (Fig. 6.3(b)). Thus, a 2D representation is only suitable for matching objects observed from very similar viewpoints (e.g. if images are captured

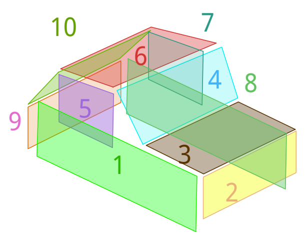


(a) The viewpoint V in a 3D part representation. Φ, Θ are zenith and azimuth angles

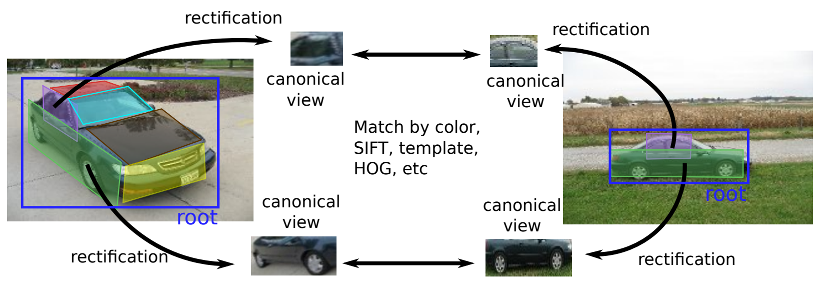


(b) A 2D part representation, where object parts are represented by 2D rectangles in the image plane. *Felzenszwalb et al.* (2010) uses 2D part representation.

Figure 6.3: Viewpoint and 2D part representation.



(a) A 3D part representation for a car.



(b) A 3D part representation allows to match objects across images by matching their parts after viewpoint rectification.

Figure 6.4: An example of 3D object part representation (a) and the matching process (b). The estimated viewpoint is the key to predicting self-occlusion and matching parts under different viewpoints. The similarity between parts is evaluated based on a bundle of features (Sec. 6.3.4).

by small-baseline stereo cameras). In such a case, parts association can be easily established.

In a 3D representation such as *Savarese and Fei-Fei* (2007); *Xiang and Savarese* (2012); *Su et al.* (2009), the root is specified by a rectangle in the image, and parts are associated to 3D flat surfaces that make up an object (Fig. 6.4(a)). The viewpoint is denoted by the azimuth and zenith angle of object pose (Fig. 6.3(a)). The canonical view of a part (Fig. 6.4(b)) is defined as the most frontal view of the part. If the pose of the object is available, any part in the 2D image can be rectified into its canonical view by using the homography transformation provided by the estimated viewpoint. Such rectification process allows us to compare the normalized appearance of two matching parts when observed from different viewpoints. (Fig. 6.4(b)). Moreover, a

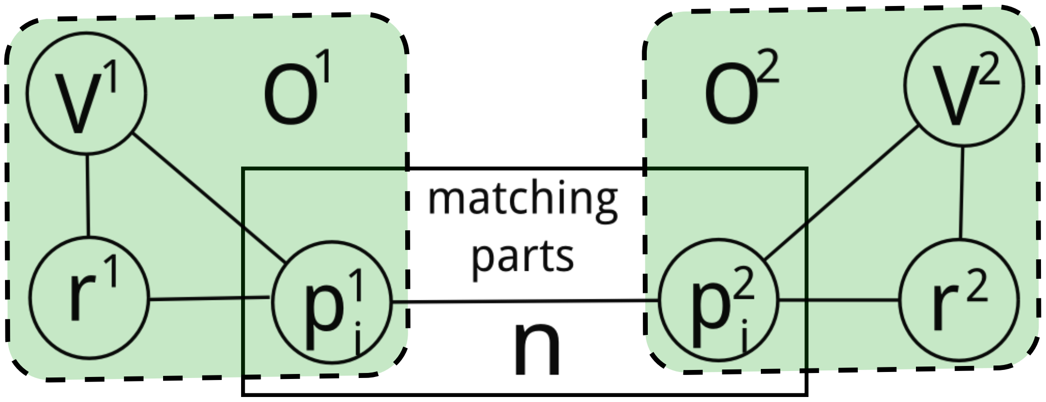


Figure 6.5: Object co-detection model when two images are considered. The dashed green box measures the compatibility between an object and its image (E_{unit}). The middle rectangle measures the similarity of parts of different objects (E_{match}).

3D part representation also enables us to predict if a certain part is occluded by other parts of the object (self-occlusion), which therefore prevents self-occluded parts from being erroneously matched. For all these reasons, a 3D representation is appropriate for matching objects observed from different viewpoints.

6.3.2 Energy Function for the Model

In formulating the co-detection framework, we follow the key intuition that objects across images are matched by associating corresponding parts. Fig. 6.5 shows the graphical representation of the model when two images are considered. The linkages between parts model the property that the corresponding parts must have similar appearance. Notice that, the model degenerates into a typical part-based object detection model (the green dashed box) if only one image is presented. The model in Fig. 6.5 can be generalized to the case of K input images and we define the following energy function to measure the likelihood of detecting the matching objects $\{O^1 \cdots O^K\}$ in different images $\{I^1 \cdots I^K\}$:

$$E(\mathcal{O},\mathcal{I}) = \sum_{k=1}^{K} E_{\text{unit}}(O^k, I^k) + \sum_{i=1}^{n} E_{\text{match}}(\{p_i^k\}_{k=1}^K, \{V^k\}_{k=1}^K, \mathcal{I}), \quad (6.1)$$

where E_{unit} measures the compatibility between the object O^k and the image I^k, and E_{match} models the constraint that the i^{th} part of a matching object should have similar appearance across images.

The term E_{unit} is the *unitary potential* and defined as:

$$E_{\text{unit}}(O^k, I^k) = E_{\text{root}}(r^k, V^k, I^k) + \sum_{i=1}^{n} E_{\text{part}}(p_i^k, V^k, I^k) + \sum_{i=1}^{n} E_{\text{rp}}(r^k, p_i^k, V^k, I^k), \quad (6.2)$$

where E_{root} and E_{part} are the unary potentials measuring the compatibility between image evidence and the root and the object part respectively; E_{rp} is the pairwise potential that measures the consistency between a part and its root. E_{rp} models the relative location between a root and the part, following a star-model representation. Details of computing E_{unit} are given in Sec. 6.3.3.

The term E_{match} is the *matching potential* and defined as:

$$E_{\text{match}}(\{p_i^k\}_{k=1}^{K}, \{V^k\}_{k=1}^{K}, \mathcal{I}) = \frac{1}{C_K^2} \sum_{k_1, k_2} M(p_i^{k_1}, p_i^{k_2}, V^{k_1}, V^{k_2}, I^{k_1}, I^{k_2}), \quad (6.3)$$

where $M(p_i^{k_1}, p_i^{k_2}, V^{k_1}, V^{k_2}, I^{k_1}, I^{k_2})$ is a matching function (Eq. 6.4) which measures the appearance similarity between the i^{th} part of object O^{k_1} and the i^{th} part of object O^{k_2}, and C_K^2 denotes the total number of possible object matches. Details of computing E_{match} are given in Sec. 6.3.4. Notice that the matching potential for multiple images is in practice expressed as a summation of pair-wise matching potentials.

By using the energy function defined in Eq. 6.1, a co-detector can boost the score (energy) of true positives if matching objects exist in other images. Therefore, a co-detector is capable of recovering true positives missed by a single-image detector (by threshold cutting).

6.3.3 Unitary Potential E_{unit}

The unitary potential E_{unit} measures the compatibility between object O^k and the evidence in image I^k. E_{unit} can be evaluated by retaining the score of a detection candidate returned by any standard object detector such as *Dalal and Triggs* (2005); *Leibe et al.* (2004); *Felzenszwalb et al.* (2010); *Gu and Ren* (2010); *Xiang and Savarese* (2012). In this paper, we adopt the energy formulation of a typical part-based object detection model (e.g. Sec. 3.1 in *Felzenszwalb et al.* (2010) and Sec. 3.1 in *Xiang and Savarese* (2012)). In such models, the category-level detection templates, which encode the visual features (e.g. HOG *Dalal and Triggs* (2005)), are trained for both root and parts. Relative locations between a root and parts are also encoded in the models. Given an input image, an object is detected by searching for the optimal locations of the root and parts so that their visual features fit the templates and their relative locations fit the shape model. We define β_{root}, β_{part}, and β_{rp} as the parameters in E_{root}, E_{part}, and E_{rp}. The form of these parameters varies according to the model applied[1]. Sec. 6.3.6 explains how we learn these parameters.

6.3.4 Matching Potential E_{match}

The matching potential E_{match} measures the similarity between two objects by matching their corresponding parts. If a part p_i is visible, we can extract its feature ϕ_i from the image. ϕ_i consists of a set of geometrical and visual features. In our experiment, the geometrical feature is: 1) the 3D location of this part w.r.t. the 3D object centroid if a 3D part representation (e.g. *Xiang and Savarese* (2012)) is applied, or 2) the 2D part location w.r.t. the 2D object centroid if a 2D part representation (e.g. *Felzenszwalb et al.* (2010)) is applied. The visual features include color histogram, point feature *Lowe* (2004) and pixel intensity values within image patches. If a 3D

[1]E.g. if the model in *Felzenszwalb et al.* (2010) is applied, we have $\beta_{root} = F'_0$, $\beta^i_{part} = F'_i$, and $\beta^i_{rp} = d_i$ for each part i, where the right-hand terms are defined in Eq. 2 and 3 in paper *Felzenszwalb et al.* (2010).

part representation is applied, we extract such features after rectifying the part into its canonical view (Fig. 6.4(b)).

If a part p_i is visible in both images $I^{k_1}$and I^{k_2}, we compute a vector $\mathbf{s}(\phi_i^{k_1}, \phi_i^{k_2})$ to measure the similarity between its features $\phi_i^{k_1}$ and $\phi_i^{k_2}$:

$$\mathbf{s}(\phi_i^{k_1}, \phi_i^{k_2}) = [s_1(\phi_i^{k_1}, \phi_i^{k_2}), s_2(\phi_i^{k_1}, \phi_i^{k_2}), s_3(\phi_i^{k_1}, \phi_i^{k_2}), s_4(\phi_i^{k_1}, \phi_i^{k_2})],$$

where s_1 is the negative value of the KL-distance between the color histograms, s_2 is the log value of the number of matched SIFT *Lowe* (2004) points, s_3 is the inner product of the normalized image patches, s_4 is the inverse value of the distance between their geometrical features. On the other hand, if either part is not visible (self-occluded), we set $\mathbf{s}(\phi_i^{k_1}, \phi_i^{k_2}) = \mathbf{0}$.

To handle object self-occlusions, we associate a visibility indicator v_i^k with part p_i^k, where $v_i^k = 1$ if p_i^k is visible in image I^k and vice versa. v_i^k is a function only of the object shape and viewpoint[2]. After considering the part visibility, we use the following vector to represent the similarity between two parts:

$$\mathbf{d}_i^{k_1 k_2} = [v_i^{k_1} v_i^{k_2} \mathbf{s}(\phi_i^{k_1}, \phi_i^{k_2})^T, 1 - (1 - v_i^{k_1})(1 - v_i^{k_2})]^T.$$

Note that $\mathbf{d}_i^{k_1 k_2}$ is a function of part locations, viewpoints and images. The last term of $\mathbf{d}_i^{k_1 k_2}$ accommodates the bias in the case where either part is not visible. We compute the similarity score as

$$M(p_i^{k_1}, p_i^{k_2}, V^{k_1}, V^{k_2}, I^{k_1}, I^{k_2}) = \mathbf{w}_i^T \mathbf{d}_i^{k_1 k_2}, \tag{6.4}$$

where $\mathbf{w}_i$ is the matching weight to be learned from a training set. Since $\mathbf{d}_i^{k_1 k_2}$ encodes the visibility information, we can learn a universal set of weights $\mathbf{w}_i$ for all the parts

[2]If a 2D part representation is applied, $v_i^k = 1$ for every parts of the object that is seen from the same viewpoint.

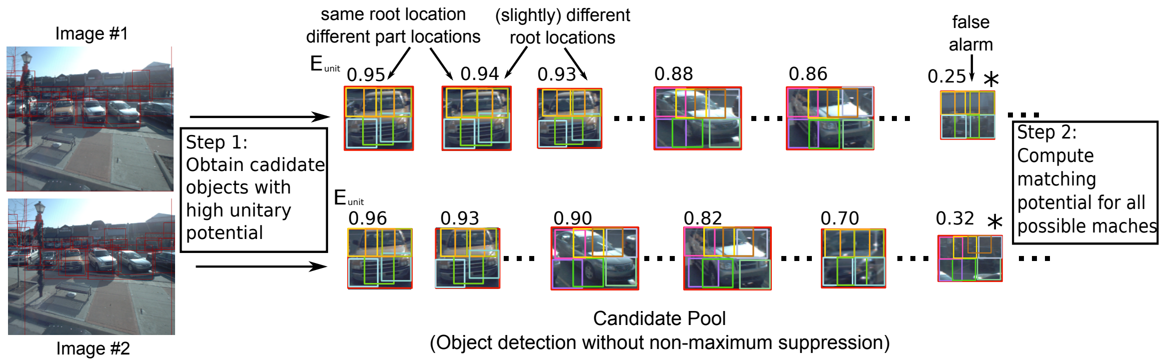


Image #1
Image #2
Step 1: Obtain cadidate objects with high unitary potential
same root location different part locations
(slightly) different root locations
false alarm
E_unit
0.95
0.94
0.93
0.88
0.86
0.25
0.96
0.93
0.90
0.82
0.70
0.32
Step 2: Compute matching potential for all possible maches
Candidate Pool (Object detection without non-maximum suppression)

Figure 6.6: Two-step inference. In this example, we apply *Felzenszwalb et al.* (2010) to compute E_{unit}. Two input images are displayed on the left. Each row on the right corresponds to a set of candidate detections extracted from the corresponding image on the left hand side.

under different viewpoints. The procedure for learning $\mathbf{w}_i$ is explained in Sec. 6.3.6.

6.3.5 Model Inference

The goal of the inference is to find the optimal matching instances $\mathcal{O}^*$ in the images $\mathcal{I}$ so that:

$$\mathcal{O}^* = \arg\max_{\mathcal{O}} E(\mathcal{O}, \mathcal{I}),$$

where $E(\mathcal{O}, \mathcal{I})$ is defined in Eq.6.1. The inference outputs the bounding box, part locations, viewpoint and instance ID (which defines matching objects correspondences across images) for each object in the images. Exactly solving the above optimization problem is intractable, since the model contains loops. We propose a two-step inference algorithm to make the problem computationally tractable.

The first step is to predict a candidate pool of object instances consisting of all objects whose unitary potential E_{unit} is larger than a threshold. Fig. 6.6 illustrates the candidate pool when *Felzenszwalb et al.* (2010) is applied. Since computing E_{unit} is equivalent to computing the potential score of an object detector, this candidate pool can be obtained by applying category level object detector without non-maximum suppression. Notice that, two resulting candidates may have the same root location but different part locations.

The second step is to identify the best set of co-detections by searching through all across-image matches in this candidate pool. Given K images, suppose the candidate pool of image I^k contains n_k objects ($k = 1 \cdots K$), then there will be $\prod_{k=1}^{K} n_k$ possible matching object candidates. We compute the joint energy $E(\mathcal{O}, \mathcal{I})$ for every matches. Since the unitary potential E_{unit} is already computed during the first step, the additional operation is just to compute the matching potential E_{match}, which is computationally cheap as it only requires the calculation of dot products. Finally, we apply non-maximum suppression to select among the $\prod_{k=1}^{K} n_k$ possible matches the best matching objects. Matching objects are selected based on their energy values – matching objects associated to high energy values are preferred over those associated with lower energy values. The result of this selection process is the output of the co-detector.

6.3.6 Model Learning

In order to learn the parameters of the co-detection model, we label the bounding boxes of objects and the ground truth matching objects across images. Given a set of T groups (a group consists of two or more images that include matching objects) of training images $\{\mathcal{I}^t\}$ with labeled matching objects $\{\mathcal{O}^t\}$, the goal is to learn β_{root}, β_{part}, β_{rp}, and $\mathbf{w} = (\mathbf{w}_1, \ldots, \mathbf{w}_n)$. Since the part locations are not labeled, learning can be solved following a latent SVM learning procedure (part locations are latent variables):

$$\begin{aligned}
&\{\beta_{\text{root}}, \beta_{\text{part}}, \beta_{\text{rp}}, \mathbf{w}\} \\
&= \arg \min_{\beta_{\text{root}}, \beta_{\text{part}}, \beta_{\text{rp}}, \mathbf{w}} \frac{1}{2}(\|\beta_{\text{root}}\|^2 + \|\beta_{\text{part}}\|^2 + \|\beta_{\text{rp}}\|^2 + \|\mathbf{w}\|^2) + \\
&\lambda \sum_t \max(0, 1 - y_t \max_{\mathcal{P}^t} E(\mathcal{O}^t, \mathcal{I}^t)),
\end{aligned} \tag{6.5}$$

where $\mathcal{P}^t$ represents all possible part locations for the objects $\mathcal{O}^t$, λ is the regularization constant, $y_t \in \{1, -1\}$ indicates if the t^{th} training group is positive or negative. However, exact learning using Eq. 6.5 is intractable due to the high dimensionality of the unknowns and the presence of loops in the model.

Instead of solving the problem in Eq. 6.5, we propose a two-step learning procedure. First, we only learn β_{root}, β_{part}, β_{rp} based on individual training images. This is equivalent to learning parameters of a traditional part-based detector (e.g. *Felzenszwalb et al.* (2010)). By using the learned β_{root}, β_{part}, β_{rp} and labeled root location r^k, the object parts in the training image I^k can be predicted as $\{\bar{p}_i^k\}_{i=1}^n$. Second, we learn $\mathbf{w}$ based on labeled matches, labeled viewpoints, and predicted parts:

$$\mathbf{w} = \arg\min_{\mathbf{w}} \frac{1}{2} \sum_i \|\mathbf{w}_i\|^2 + \lambda \sum_{t=1}^{T} \max(0, 1 - y_t[\sum_{i=1}^{n} E_{match}(\{\bar{p}_i^k\}_{k=1}^K, \{V^k\}_{k=1}^K, \mathcal{I})])$$

where $\mathbf{w}$ can be estimated using a standard support vector machine.

6.4 Experiments

The experiments are designed in order to demonstrate: 1) an object co-detector is capable of successfully detecting matching objects across images; 2) estimate the viewpoint transformation between matching objects; 3) achieve superior performances than traditional detection methods that work on individual images in isolation; 4) achieve similar performances to traditional detection methods if no matching objects are present in the images; 5) a co-detector can be successfully used to detect an object instance with just one training image (where the same object instance is observed from an unknown and arbitrary viewpoint) and obtain superior results than traditional single instance detectors. Moreover, we present experiments that demonstrate that our co-detection framework can be useful in a number of recognition scenarios so as

Average Precision (%)		Car (all)	Car ($h > 80$)	Person (all)	Person ($h > 120$)
Stereo Pair	DPM	49.8	47.1	59.7	55.4
	Co-detector	**53.5**	**55.5**	**62.7**	**63.4**
Random Pair	DPM	49.8	47.1	**59.7**	55.4
	Co-detector	**50.0**	**49.1**	58.1	**58.1**

Table 6.1: Object detection results using the car dataset *Bao and Savarese* (2011) and the pedestrians dataset *Ess et al.* (2007). "$h > X$" means we only count the objects with height more than X pixels. The image height of the car / pedestrian dataset is 600 / 480 pixels. "Stereo pair": testing image pairs are obtained from a stereo camera with small baseline; this implies that most images contain matching objects. "Random pair": testing image pairs are randomly selected from the whole data set; this implies that most of these images contain few or none matching objects. The number of testing image pairs are 300 / 200 for the car / pedestrian dataset.

		Iron	Mouse	Shoe	Car	Phone	Stapler	Bike	Toaster	Mean
Det.	ALM	82.2	52.2	84.1	**98.3**	80.2	**70.5**	**93.8**	97.5	82.3
	Ours	**82.5**	**54.5**	**85.5**	98.0	**81.0**	70.2	93.1	**98.2**	**83.0**
Pose	ALM	86.0	69.8	86.6	93.1	**86.3**	73.2	90.1	65.4	81.3
	Ours	**89.8**	**72.0**	**88.0**	**95.3**	86.0	**73.9**	**92.3**	**70.3**	**83.5**

Table 6.2: Object detection and pose estimation results using the 3D object dataset *Savarese and Fei-Fei* (2007).

to: 1) match the same object instances across images where the object location is known but the association and viewpoint transformation is unknown; 2) establish the correct correspondence between images that contain the same (but unknown) object instances seen from different (unknown) viewpoints.

6.4.1 Object Detection and Pose Estimation

The experiments on object detection and pose estimation are conducted on three publicly available datasets: a car dataset *Bao and Savarese* (2011) (see Fig. 6.8(a)), a pedestrian dataset *Ess et al.* (2007) (see Fig. 6.8(b)), and a 3D object dataset *Savarese and Fei-Fei* (2007) (see Fig. 6.8(c) and 6.8(d)). To evaluate object detection accuracy, we follow the criteria in the PASCAL VOC challenge[3] and report average precision

[3]`http://pascallin.ecs.soton.ac.uk/challenges/VOC/`

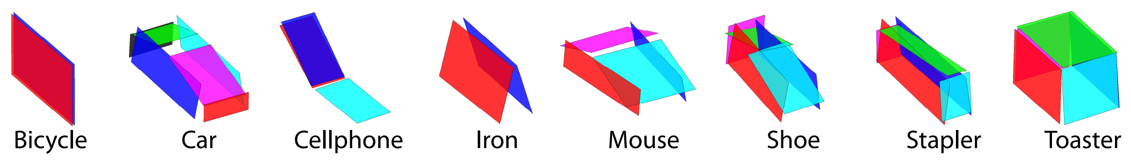


Figure 6.7: The 3D part representation for eight categories in *Xiang and Savarese* (2012).

(AP). To evaluate pose estimation accuracy, we follow the criteria in *Savarese and Fei-Fei* (2007). Tab. 6.1 shows the object detection results on the car and pedestrian datasets. For both datasets we evaluate the co-detector on image pairs with either small baseline (indicated by stereo pairs) or with large baseline or with no overlap at all (indicated as random pairs). In the former case, the object viewpoint change is not significant, and we apply the model in *Felzenszwalb et al.* (2010) (which uses a 2D part representation) to represent objects and compute E_{unit}. Tab. 6.1 shows that, object co-detector achieves higher detection accuracy than a traditional object detector such as *Felzenszwalb et al.* (2010) when it is applied on each image in isolation. This advantage grows if we only count the large objects in images, since these contain better identifiable parts than small scale objects. Tab. 6.1 also shows that, if random pairs of images are considered, object co-detection performs on par with single-image detection (e.g. *Felzenszwalb et al.* (2010)). This result validates the property that if no matching objects are present in the images, a co-detector degenerates into a traditional part-based detector.

Tab. 6.2 shows the object detection and pose estimation results on the 3D object dataset *Savarese and Fei-Fei* (2007), where significant object viewpoint changes exist. In the following experiments, we use 5 object instances for testing in each category. We enumerate all pairs of images containing matching objects to generate the testing image list. We apply the model in *Xiang and Savarese* (2012). Examples of a 3D object representations in *Xiang and Savarese* (2012) are shown in Fig. 6.7. As Tab.

AP (%)		Iron	Mouse	Shoe	Car	Phone	Stapler	Bike	Toaster
Same Pose	SIFT	25.4	15.2	37.6	43.2	30.7	25.6	24.6	15.2
	Ours	**90.8**	**56.5**	**86.6**	**98.4**	**88.5**	**72.6**	**93.7**	**98.2**
Different Pose	SIFT	2.5	2.2	6.0	3.3	5.6	1.2	5.0	1.3
	Ours	**81.8**	**54.8**	**86.3**	**98.1**	**81.1**	**71.4**	**94.5**	**97.9**

Table 6.3: Single instance detection result using the 3D object dataset. Same / Different Pose: the azimuth angle (Fig. 6.3(a)) of an object in a query image is the same / different as the the azimuth angle of the labeled object.

6.2 shows, object co-detection outperforms *Xiang and Savarese* (2012) in detecting the objects and estimating their pose. The gain may not be substantial for those categories for which the baseline method *Xiang and Savarese* (2012) already shows very strong performance.

6.4.2 Detecting Single Object Instances

In this experiment, we demonstrate the ability of the co-detector to detect an object instance from a testing image under the assumption that the same object instance is observed and labeled in one of the training images. The object poses in testing and training are in general different. We compare against a single instance detection method *Lowe* (2004), which uses generalized Hough voting and homography validation to detect objects. Tab. 6.3 shows the detection accuracy for detecting a labeled instance. Notice that our method achieves a significant improvement over *Lowe* (2004) in that it leverages the learnt categorical structure of object as opposed to *Lowe* (2004) which only relies on low level features and a subsequent geometrical validation step.

Tab. 6.4 summarizes the overall accuracy in detecting objects and estimating their pose. The comparison between Tab. 6.4 and Tab. 6.2 allows us to appreciate the superior performance of the co-detector when the object position is available in one of the two images (Tab. 6.4), as opposed to be unknown in both images (Tab. 6.2).

AP (%)	Iron	Mouse	Shoe	Car	Phone	Stapler	Bike	Toaster	Mean
Det.	84.8	55.3	86.3	98.2	83.6	71.7	94.2	98.0	84.0
Pose	93.2	76.7	90.1	97.9	89.3	79.0	92.1	87.3	88.2

Table 6.4: Single instance detection results. See Tab. 6.2 for a comparison.

Accuracy %		Iron	Mouse	Shoe	Car	Phone	Stapler	Bike	Toaster
Same Pose	Color	55.4	55.4	40.8	39.2	48.7	53.0	26.8	54.4
	SIFT	46.6	43.7	47.7	58.9	44.9	43.3	40.5	43.2
	SP	46.8	**58.7**	49.2	39.5	42.7	41.3	34.9	66.0
	Ours	**60.0**	55.6	**66.8**	**64.5**	**67.0**	**59.2**	**57.6**	**86.5**
Different Pose	Color	50.1	43.8	38.4	38,3	27.9	43.1	30.2	52.7
	SIFT	26.1	33.4	34.7	27.3	26.2	30.9	27.6	32.4
	SP	29.6	44.8	44.1	29.2	21.3	31.2	30.0	44.5
	Ours	**56.1**	**52.6**	**63.1**	**46.2**	**56.5**	**55.3**	**62.3**	**83.5**

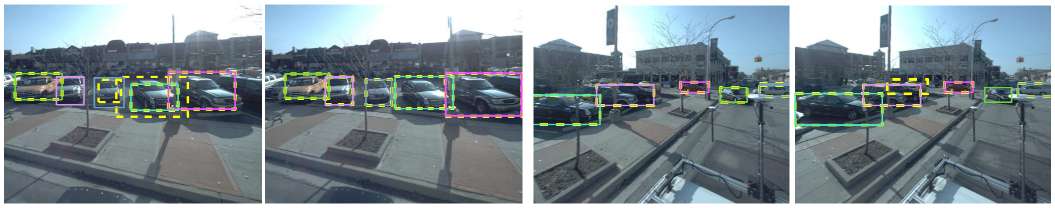
Table 6.5: Accuracy in matching object instances. Different baseline methods are compared using two different settings: the matching objects have the same / different azimuth pose. In Color, color histograms within the object bounding box (BB) are compared. In SIFT*Lowe* (2004), the number of matched SIFT features within the object BB is used. In SP, a spatial pyramid matching method *Lazebnik et al.* (2006) within the object BB is used.

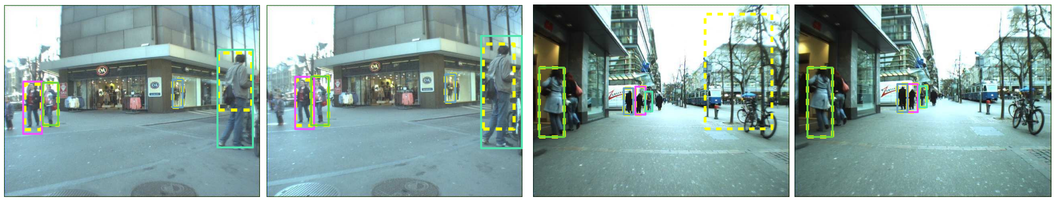
6.4.3 Matching Objects

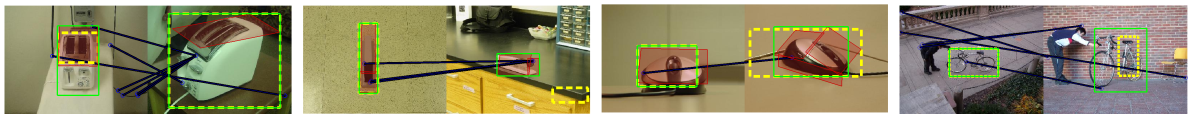
In this experiment, we demonstrate the ability of the co-detector to discover matching objects. We assume that objects are already correctly detected (i.e., the object bounding box is given for all the images) and the task consists of establishing the correct match between bounding boxes corresponding to same object instances. For each trial, we have 5 candidate object instances and 1 target object instance of the same object category. The goal is to find among the 5 candidates the one that corresponds to the target. We compare the co-detector against a number of baseline methods that are capable of estimating if two object bounding boxes correspond to the same instance or not. These methods use different strategies to compute the matching score. As Tab 6.5 shows, the co-detector obtains the best performances in all the experiments.

Accuracy %		Iron	Mouse	Shoe	Car	Phone	Stapler	Bike	Toaster
Same Pose	BoW	42.2	31.2	37.1	30.7	54.9	31.2	26.9	26.6
	SP	42.7	31.9	39.3	34.1	**56.7**	32.5	31.0	28.6
	Color+Det	52.7	35.5	35.1	39.0	40.8	40.1	26.9	39.6
	SP+Det	40.2	36.3	41.0	38.1	40.5	31.7	32.5	53.9
	SIFT+Det	41.9	39.3	46.4	59.5	40.9	38.5	39.9	41.3
	Ours	**53.6**	**47.6**	**55.1**	**64.7**	53.9	**50.6**	**58.3**	**66.0**
Different Pose	BoW	35.3	32.1	36.6	35.8	30.0	30.3	30.1	31.1
	SP	41.7	33.0	37.1	37.5	29.1	30.5	34.4	31.3
	Color+Det	42.6	36.0	34.6	34.4	20.7	37.6	29.7	40.5
	SP+Det	33.2	29.6	32.3	27.0	22.5	26.6	30.8	39.0
	SIFT+Det	35.8	28.6	33.2	28.1	26.8	27.1	27.3	31.0
	Ours	**48.3**	**44.1**	**45.9**	**44.2**	**40.3**	**44.3**	**64.8**	**59.4**

Table 6.6: Accuracy in matching images that contain the same object instance. Different baseline methods are compared using two different settings: the matching objects have the same / different azimuth pose. In BoW, bag-of-words model *Fei-Fei et al.* (2007) is used to compare images. In SP, a spatial pyramid matching method *Lazebnik et al.* (2006) is used. In Color, color histogram is used. In SIFT*Lowe* (2004), the number of matched SIFT features is used. X+Det: matching images by applying method X to match the first detected object by *Xiang and Savarese* (2012). See Tab. 6.5 for a comparison.


(a) Car dataset *Bao and Savarese* (2011).


(b) Pedestrian dataset *Ess et al.* (2007).


(c) The toaster, stapler, mouse, and bike in 3D object dataset *Savarese and Fei-Fei* (2007).

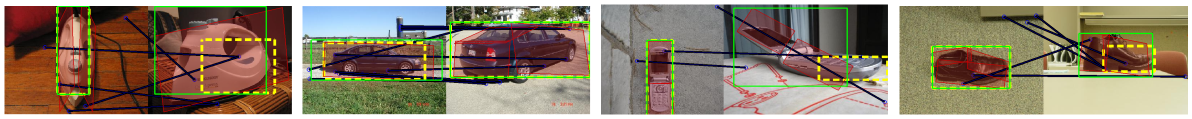

(d) The iron, car, cellphone, and shoe in 3D object dataset *Savarese and Fei-Fei* (2007).

Figure 6.8: Anecdotal results on different datasets. Solid bounding boxes: detection results by our object co-detector applied on the image pair. Detected matching instances are shown in different colors. Dashed yellow bounding boxes: detection results by state-of-the-art detector *Felzenszwalb et al.* (2010) applied on each image individually. Fig. 6.8(c) and 6.8(d): detected parts are highlighted in red. The blue lines are SIFT matches obtained by threshold test where the threshold is 0.7.

6.4.4 Matching Images by Objects

In this experiment, the goal is to match images if they contain the same object instance. Unlike the previous experiment, the locations of objects are not given in any of the images. For each trial, we have 5 candidate images and 1 target image. Each image contains one object. The goal is to find among all the image candidates the one that contains the same object instance as in the target image. We compare the co-detector against several possible image matching methods and report the matching accuracy in Tab. 6.6. We also apply image matching methods to match the bounding box of the most likely detection returned by *Xiang and Savarese* (2012), and we denote these results as "+Det". If we apply matching methods to match the ground truth bounding boxes of objects, the result will be identical to the experiment reported in Sec. 6.4.3. Our co-detection model achieves superior performance in all the experiments.

6.5 Conclusion

In this paper, we have introduced the problem of object co-detection and proposed a novel framework for solving it. We have shown that our framework, by leveraging state-of-the-art part-based object representations, is capable of successfully addressing the co-detection problem in presence of large viewpoint changes and object self-occlusions. We have conducted extensive experimental evaluation on three challenging datasets to demonstrate properties and strengths of our co-detection approach.

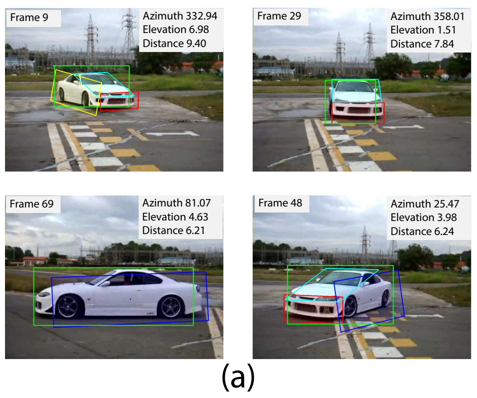
CHAPTER VII

Application II: Multiview Object Tracking with 3D Aspect Parts

7.1 Introduction

Traditional object tracking methods focus on accurately identifying the 2D location of objects in the image and associating those locations across frames. While this capability is a critical ingredient in many application scenarios, it is often not sufficient. There are numerous situations (e.g., in autonomous driving) where not only does one need to track the location of an object (e.g., a car) but also infer its 3D pose in time – for instance, if one needs to predict a potential collision, estimating other cars' pose and angular velocities is crucial. Moreover, there are situations (e.g., in robotics or augmented reality) where one needs to identify portions of the object such as its aspects or affordance. For instance, this is critical when an autonomous agent needs to interact with, say, a car and wants to figure out where a door or a window is.

Unfortunately, most of the existing tracking methods are not capable of (or at least not designed for) estimating the 3D object pose nor tracking portions of the target. In this paper, we seek to address this limitation and propose a new tracking framework that not only tracks the object in 2D, as most the state-of-the-art methods



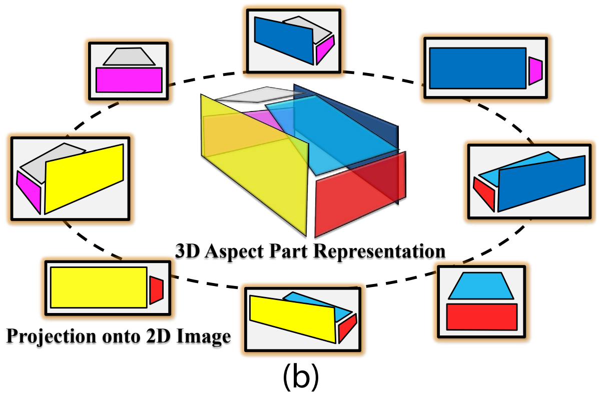


Figure 7.1: (a) An example output of our tracking framework. Our multiview tracker provides the estimates for continuous pose and 3D aspect parts of the object. (b) An example of the 3D aspect part representation of a 3D object (car) and the projections of the object from different viewpoints.

do, but also returns, as part of a joint inference problem, a continuous estimation of the viewpoint in time. Moreover, it is also able to identify and track portions of the object such as its aspects, in time (see Fig. 7.1(a)).

Our proposed tracker follows and generalizes the philosophy of "tracking by detection" (whereby a track is inferred by using detection hypotheses as observations) and leverages existing 3D (multiview) object representations *Thomas et al.* (2006); *Savarese and Fei-Fei* (2007); *Liebelt et al.* (2008); *Su et al.* (2009); *Xiang and Savarese* (2012); *Pepik et al.* (2012); *Fidler et al.* (2012); *Lim et al.* (2013) for detecting and estimating the 3D pose of object categories. Unlike traditional tracking by detection methods, however, that just focus on tracking the 2D or 3D location of the object, our approach also "tracks" the 3D pose and parts of the target. We leverage the 3D aspect part representation (see Fig. 7.1(b)) and use it in a novel particle filtering framework for multiview tracking, where combining viewpoint estimation and the 3D aspect parts enables us to predict the visibility and shape of each 3D aspect part. In particular, we leverage two state-of-the-art object detectors to train the category-level part templates in our multiview tracking framework: Deformable Part Model (DPM) *Felzenszwalb et al.* (2010) and Aspect Layout Model (ALM) *Xiang and*

Savarese (2012). We believe these are reasonable choices in that: i) DPM achieves state-of-the-art object detection performance and it is suitable for "tracking by detection" implementation as shown in *Choi et al.* (2013b); *Pirsiavash et al.* (2011) ii) ALM achieves state-of-the-art pose estimation results and provides a good platform for injecting 3D information to the 3D pose tracking problem; iii) ALM can recover the object layout in term of the distribution of object aspects in 3D.

Moreover, in order to increase the robustness of our tracker to viewpoint changes as well as occlusions, we propose to inject to our tracker the ability to learn the appearance of the object in an online learning fashion, similar to *Babenko et al.* (2011); *Hare et al.* (2011); *Kalal et al.* (2012); *Bao et al.* (2012a); *Supancic III and Ramanan* (2013); *Yao et al.* (2013). Unlike traditional online learning tracking methods, however, which focus on learning a holistic description of the entire object as the tracking goes by (an exception is the recent work by *Yao et al.* (2013)), we propose to update the appearance model only for the *visible* parts of the object. Part visibility is readily available as a result of the fact that we also estimate the 3D pose of the object in time. A key strength of our approach is that we combine tracking by detection and online learning in a coherent probabilistic framework.

In our experiments, we provide results for viewpoint estimation and 3D aspect part localization. Besides, to demonstrate the usefulness of 3D pose and viewpoint during tracking, we compare our method with some of the state-of-the-art online learning methods that do not use 3D information and show significant improvement. Furthermore, we illustrate that our framework is effective in leveraging temporal information to provide continuous estimates for the object pose with and without online learning. Finally, we show that in the presence of occlusions, online learning helps increase the robustness and accuracy.

Since the current benchmark datasets for online object tracking *Wu et al.* (2013) are not designed to test the ability of the trackers on handling topological appearance changes and do not show significant viewpoint variations, we collected a new challenging dataset with 9 multiview car video sequences from YouTube for experiments. We also test our method on a subset of the KITTI dataset *Geiger et al.* (2012) which comprises videos with significant viewpoint changes. Furthermore, we evaluate our method on a standard sequence for car tracking without viewpoint variations *Kalal et al.* (2012). We demonstrate the ability of our method to accurately track viewpoints and 3D aspect parts in videos. Fig. 7.1(a) shows the tracking results of our method.

Contributions. 1) We propose a multiview tracker to handle the topological appearance change of rigid objects during tracking, which estimates continuous 3D viewpoint in a monocular setting. 2) Our multiview tracker is able to track the 3D aspect parts of an object. 3) We combine category-level pre-trained 3D object detectors and instance-level online-learned part appearance models in a principled way. 4) We contribute a new dataset with 9 car video sequences for multiview object tracking, and show promising tracking results on it.

7.2 Related Work

Tracking by Detection. Our approach falls in the category of tracking by detection methods *Breitenstein et al.* (2011); *Butt and Collins* (2013); *Choi et al.* (2013b); *Pirsiavash et al.* (2011); *Yang and Nevatia* (2012), where category-level detectors are utilized to track the target of interest. However, in contrast to these methods, our focus is on tracking continuous 3D pose and 3D aspect parts.

Online Object Tracking. Online trackers focus on constructing appearance models which adapt to appearance changes during tracking *Babenko et al.* (2011); *Hare et al.* (2011); *Kalal et al.* (2012); *Bao et al.* (2012a); *Yao et al.* (2013); *Supancic III and Ramanan* (2013). By leveraging online learning techniques, such as online multiple instance learning *Babenko et al.* (2011), online structural learning *Yao et al.*

(2013) and self-paced learning *Supancic III and Ramanan* (2013), these methods have achieved robust tracking results on benchmark datasets *Wu et al.* (2013). Since they are able to track generic objects, they are referred to as model-free trackers. However, as shown in our experiments, they cannot handle the topological appearance change of objects caused by severe viewpoint transformations. An exception is the recent work by *Oron et al.* (2014) which extends the Lucas-Kanade algorithm *Lucas et al.* (1981) with pixel object/background likelihoods. It shows competitive performance on a vehicle tracking dataset with severe viewpoint changes.

Multiview Object Recognition. Our tracker builds upon the idea of multiview recognition. The goal of multiview object recognition is to recognize objects from arbitrary viewpoints, which dates back to the early works in computer vision (e.g., *Lowe* (1987); *Dickinson et al.* (1991)). Recent works in multiview object recognition either represent objects as collections of parts or features which are connected across views *Thomas et al.* (2006); *Savarese and Fei-Fei* (2007); *Su et al.* (2009), or utilize explicit 3D models with associated visual features to represent objects *Liebelt et al.* (2008); *Xiang and Savarese* (2012); *Pepik et al.* (2012); *Fidler et al.* (2012); *Lim et al.* (2013). Our method benefits from the 3D aspect part representation introduced in *Xiang and Savarese* (2012). While *Xiang and Savarese* (2012) focuses on object detection and pose estimation from single images in a discretized viewpoint space, we show that the 3D aspect part representation can be utilized to estimate continuous object pose and 3D aspect part locations in multiview object tracking.

3D Model-based Tracking. Multiview object recognition methods have been extended and applied to 3D tracking *Roller et al.* (1993); *Drummond and Cipolla* (2002); *Lepetit and Fua* (2005); *Choi and Christensen* (2010); *Prisacariu and Reid* (2012); *Pauwels et al.* (2013). Most of the previous works aim at tracking the 3D pose of an object *instance* using its 3D CAD model, e.g., *Drummond and Cipolla* (2002); *Choi and Christensen* (2010); *Prisacariu and Reid* (2012). In contrast, we focus on

3D tracking of object *categories* with a 3D object category representation, which is able to handle the intra-class variability among object instances in the same category.

Monocular vs. Multi-Camera Multiview Object Tracking. An alternative way to achieve multiview object tracking is to utilize multi-camera settings, where the target is observed from multiple cameras simultaneously *Khan and Shah* (2009); *Leal-Taixé et al.* (2012); *Hofmann et al.* (2012). Tasks such as occlusion reasoning *Khan and Shah* (2009) and 3D reconstruction *Hofmann et al.* (2012) which are challenging in monocular settings can be solved efficiently in multi-camera environments. Since multiple cameras are only available in specific scenarios, we focus on monocular multiview tracking in this work.

3D Tracking and Reconstruction In contrast to methods that track targets in 3D (e.g., *Petrovskaya and Thrun* (2008); *Kaestner et al.* (2012); *Feldman et al.* (2012)), we have access only to videos and do not use other sensor modalities such as range data. Compared with methods that perform joint 3D reconstruction and tracking (e.g., *Huang et al.* (2007); *Held et al.* (2013)), we are interested mainly in estimating the 3D pose and shape extent of the target in terms of its part layout.

7.3 Multiview Tracking Framework

The primary goal of multiview object tracking is to estimate the posterior distribution of the target's state $P(X_t, V_t|Z_{1:t})$ at the current time step t given all observations $Z_{1:t}$ up to that time step, where X_t and V_t denote the location and viewpoint of the target at time t respectively. Instead of tracking the object as a whole, which cannot handle the topological appearance change of object, we propose to track the 3D aspect parts of the object and its viewpoint jointly while modeling the relationship between these parts. By using a 3D aspect part representation of the object (Fig. 7.1(b)), we can predict the visibility and shape of the parts in arbitrary viewpoints. In this way, the tracking framework is able to handle the appearance change introduced by

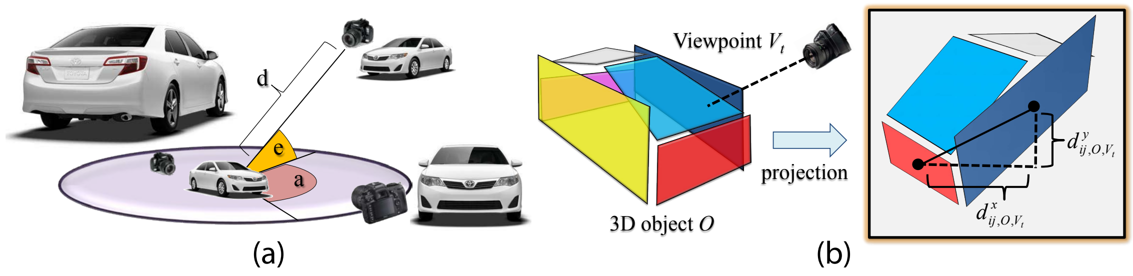


Figure 7.2: (a) The viewpoint of the object is represented by the azimuth, elevation, and distance of the camera pose in 3D, $V = (a, e, d)$. (b) Illustration of the relative distance between two parts by projecting the 3D object onto a 2D image.

viewpoint transitions, especially in cases when a part disappears or reappears due to self-occlusion. Consequently, the location of the object at time t is determined by the locations of the 3D aspect parts, i.e., $X_t = \{X_{it}\}_{i=1}^n$, where n is the number of parts and X_{it} denotes the location of part i at time t. The viewpoint V_t is represented by the azimuth a_t, elevation e_t and distance d_t of the camera position in 3D with respect to the object, i.e., $V_t = (a_t, e_t, d_t)$ as shown in Fig. 7.2(a).

By applying Bayes rule, the posterior distribution can be decomposed as

$$P(X_t, V_t|Z_{1:t}) \propto \underbrace{P(Z_t|X_t, V_t)}_{\text{likelihood}} \int \underbrace{P(X_t, V_t|X_{t-1}, V_{t-1})}_{\text{motion prior}} \underbrace{P(X_{t-1}, V_{t-1}|Z_{1:t-1})}_{\text{posterior at time t-1}} dX_{t-1}dV_{t-1}, \tag{7.1}$$

where the likelihood $P(Z_t|X_t, V_t)$ measures the probability of observing measurement Z_t given the state of the target (X_t, V_t) at time t, the motion prior $P(X_t, V_t|X_{t-1}, V_{t-1})$ predicts the state of the target at time t given its previous state, and $P(X_{t-1}, V_{t-1}|Z_{1:t-1})$ is the posterior at time $t - 1$.

7.3.1 Likelihood

The likelihood $P(Z_t|X_t, V_t)$ measures the compatibility between the state of the target (X_t, V_t) with the observation Z_t at time t. Since we track an object by its 3D aspect parts, the likelihood of the object is decomposed as the product of the likelihoods of the 3D aspect parts:

$$P(Z_t|X_t, V_t) = \prod_{i=1}^{n} P(Z_t|X_{it}, V_t), \tag{7.2}$$

where $P(Z_t|X_{it}, V_t)$ denotes the appearance likelihood of part i. The likelihood is measured based on category-level pre-trained part appearance models. To make the likelihood more robust in some difficult scenarios (e.g., occlusion), we also use instance-level online-learned part appearance models in computing the likelihoods for 3D aspect parts. In traditional online object tracking, the likelihood of a part is computed using the appearance model of that part learned online, where the assumption is that the part is always visible during tracking. However, this is not necessarily true when the viewpoint changes. When parts with learned appearance models disappear and unseen parts become visible, the tracker loses the target. In our case, when new parts appear, if no online appearance models have been learned for them before, we resort to the category-level part templates to compute the likelihood. Subsequently, the online appearance models for the new parts are initialized according to the tracking output and updated afterwards. The online appearance model is updated according to the 3D pose, i.e., we only update the model for the visible parts. Specifically, we define the likelihood as:

$$P(Z_t|X_{it}, V_t) \propto \exp\Big(\Lambda_{\text{category}}(Z_t, X_{it}, V_t) + \Lambda_{\text{online}}(Z_t, X_{it}, V_t)\Big), \tag{7.3}$$

where $\Lambda_{\text{category}}(Z_t, X_{it}, V_t)$ is the potential from the category-level part template for part i, and $\Lambda_{\text{online}}(Z_t, X_{it}, V_t)$ is the potential from the online appearance model for

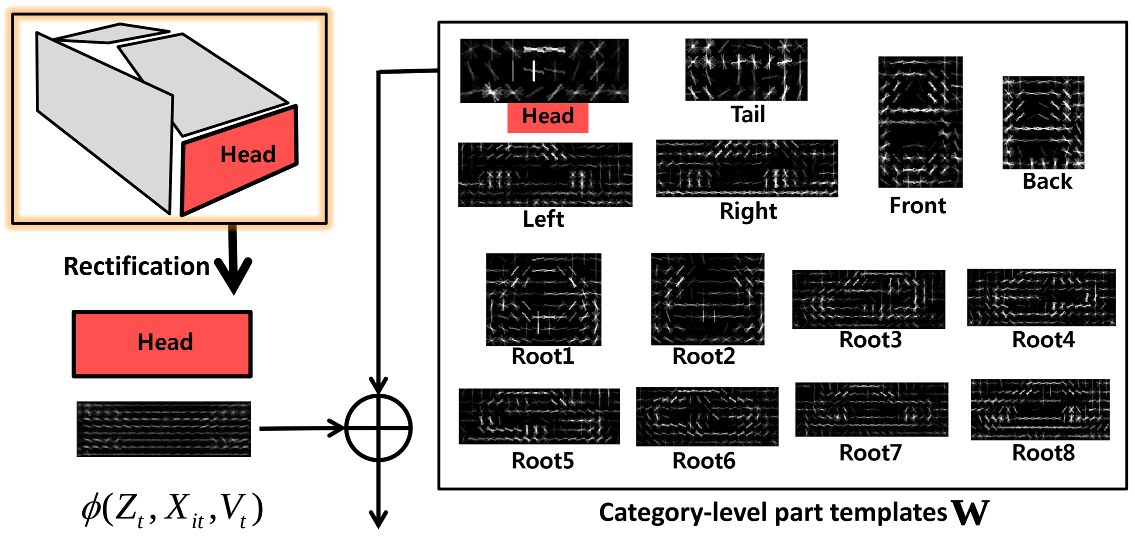


Figure 7.3: Illustration of the category-level part templates and the computation of the potential for the Head part, where rectified HOG features are used.

part i.

A category-level part template is trained with various instances in the same category, which captures the general shape of the part. We define the potential from the category-level part template as

$$\Lambda_{\text{category}}(Z_t, X_{it}, V_t) = \begin{cases} \mathbf{w}_i^T \phi(Z_t, X_{it}, V_t), & \text{if visible} \\ \alpha_i, & \text{if self-occluded,} \end{cases} \tag{7.4}$$

where $(\mathbf{w}_i, \alpha_i)$ denotes the weights of the part template, and $\phi(Z_t, X_{it}, V_t)$ is the feature vector. The part template $\mathbf{w}_i$ is applied only if the part is visible. Otherwise, an occlusion weight α_i is assigned to the part. We use rectified HOG features as $\phi(Z_t, X_{it}, V_t)$, where HOG features *Dalal and Triggs* (2005) are extracted after rectifying the image into the frontal view of the part according to the viewpoint V_t. Therefore, the part template $(\mathbf{w}_i, \alpha_i)$ corresponds to the frontal view of the part. This property is critical for continuous viewpoint estimation. In learning the part template from training images, the viewpoint space is discretized. During tracking, we can always first rectify the image into the frontal view of the part from arbitrary continuous viewpoint, and then apply the learned template. In this way, we are able to compute the likelihoods for continuous viewpoints during the Bayesian filtering tracking. All the part templates for 3D aspect parts are jointly learned from training images using a Structural SVM optimization as in *Xiang and Savarese* (2012). Fig. 7.3 illustrates the learned category-level part templates and the rectified HOG features. Note that, besides training part templates for 3D aspect parts, we also introduce root templates which correspond to the whole object in different view sections and are obtained from DPM *Felzenszwalb et al.* (2010).

The online appearance models capture instance-level characteristics of part appearance, which are specialized to the current target. Moreover, the models are updated during tracking to accommodate appearance change. The potential of the online appearance model in Eq. (7.3) is defined as

$$\Lambda_{\text{online}}(Z_t, X_{it}, V_t) = \begin{cases} \mathbf{H}_i(\psi(Z_t, X_{it}, V_t)), & \text{if visible} \\ \lambda_0, & \text{if self-occluded,} \end{cases} \tag{7.5}$$

where $\mathbf{H}_i$ is the classifier for part i, $\psi(Z_t, X_{it}, V_t)$ is the feature vector and λ_0 is a constant assigned to the part if it is self-occluded. We utilize the multiple instance boosting algorithm *Babenko et al.* (2011) for training and updating the classifier $\mathbf{H}_i$ during tracking. The classifier is applied and updated only if the part is visible under the predicted viewpoint, which prevents the classifier from learning with incorrect appearance features. Similar to the rectified HOG features used in constructing the category-level part templates, we rectify the image to the frontal view of the part according to V_t before extracting Haar-like features as in *Viola and Jones* (2001) for $\psi(Z_t, X_{it}, V_t)$. In this way, the online appearance model is robust to viewpoint distortions, and we can compute part likelihoods for continuous viewpoints.

7.3.2 Motion Prior

The motion prior $P(X_t, V_t|X_{t-1}, V_{t-1})$ predicts the current state of the target based on its previous state. We decompose the motion prior according to part location and viewpoint:

$$\begin{aligned} & P(X_t, V_t|X_{t-1}, V_{t-1}) \\ = \; & P(X_t|X_{t-1}, V_{t-1}, V_t)P(V_t|X_{t-1}, V_{t-1}) \\ = \; & P(X_t|X_{t-1}, V_t)P(V_t|V_{t-1}), \end{aligned} \tag{7.6}$$

where $P(X_t|X_{t-1}, V_t)$ models the change in location, and $P(V_t|V_{t-1})$ is the viewpoint motion. Note that in Eq. (7.6), two assumptions of conditional independence are imposed to simplify the motion prior. Inspired by *Khan et al.* (2005) which uses a Markov Random Field (MRF) motion prior to capture the interaction between targets, we model the change in location using an MRF that is able to capture the relationships between parts:

$$P(X_t|X_{t-1}, V_t) \propto \prod_{i=1}^{n} P(X_{it}|X_{i(t-1)}) \prod_{(i,j)} \Lambda(X_{it}, X_{jt}, V_t), \tag{7.7}$$

where $P(X_{it}|X_{i(t-1)})$ is the motion model for part i and $\Lambda(X_{it}, X_{jt}, V_t)$ is the pairwise potential which constrains the relative location of two parts according to the 3D aspect part representation and the viewpoint.

In order to handle abrupt location and viewpoint changes or occlusion, we do not impose a strong motion prior such as the constant velocity motion prior in our multiview tracker. The location motion of a part in Eq. (7.7) and the viewpoint motion in Eq. (7.6) are both modeled with Gaussian distributions centered on the previous location and the previous viewpoint respectively:

$$\begin{aligned} P(X_{it}|X_{i(t-1)}) &\sim \mathcal{N}(X_{i(t-1)}, \sigma_x^2, \sigma_y^2) && (7.8) \\ P(V_t|V_{t-1}) &\sim \mathcal{N}(V_{t-1}, \sigma_a^2, \sigma_e^2, \sigma_d^2), && (7.9) \end{aligned}$$

where σ_x^2, σ_y^2, σ_a^2, σ_e^2 and σ_d^2 are the variances of the Gaussian distributions for 2D part center coordinates, azimuth, elevation and distance respectively.

To define the pairwise potential between part locations in Eq. (7.7), we utilize the 3D aspect part representation (Fig. 7.1(b)). Let O denote the 3D object representation. Given the viewpoint V_t at time t, we can project the 3D object onto the image according to V_t. Then we obtain the ideal relative distance d_{ij,O,V_t} between part i and part j as shown in Fig. 7.2(b). We define the pairwise potential to penalize large deviations between the observed relative part locations from the ideal ones with Gaussian priors:

$$\begin{gathered} \Lambda(X_{it}, X_{jt}, V_t) = P(\Delta_t(x_i, x_j)|V_t) P(\Delta_t(y_i, y_i)|V_t), \\ P(\Delta_t(x_i, x_j)|V_t) \sim \mathcal{N}(d^x_{ij,O,V_t}, \sigma^2_{dx}), \\ P(\Delta_t(y_i, y_j)|V_t) \sim \mathcal{N}(d^y_{ij,O,V_t}, \sigma^2_{dy}), \end{gathered} \tag{7.10}$$

where $X_{it} = (x_{it}, y_{it})$ and $X_{jt} = (x_{jt}, y_{jt})$ denote the 2D center coordinates of the two parts, $\Delta_t(x_i, x_j) = |x_{it} - x_{jt}|$, $\Delta_t(y_i, y_j) = |y_{it} - y_{jt}|$, d^x_{ij,O,V_t} and d^y_{ij,O,V_t} are the ideal relative distances between the two parts in the x and y directions respectively (Fig. 7.2(b)), and σ^2_{dx} and σ^2_{dy} are the variances of the Gaussian distributions for 2D relative distances, which are set proportionally to the size of the part in the image. The pairwise potential (7.10) allows the 3D shape of the target to deviate from the 3D object model with some deformation cost. Note that we use a general 3D aspect part representation for an object category and apply it to different instances of that category.

7.3.3 Particle Filtering Tracking

In order to track the continuous pose of the target, we employ the particle filtering technique to infer the posterior distribution in Eq. (7.1). We use Markov Chain Monte Carlo (MCMC) sampling, where the posterior $P(X_{t-1}, V_{t-1}|Z_{1:t-1})$ at time $t-1$ is represented as a set of N unweighted samples $P(X_{t-1}, V_{t-1}|Z_{1:t-1}) \approx (X_{t-1}^{(r)}, V_{t-1}^{(r)})_{r=1}^{N}$. So we obtain the following Monte Carlo approximation to the Bayesian filtering distribution:

$$P(X_t, V_t|Z_{1:t}) \propto P(Z_t|X_t, V_t) \sum_{r=1}^{N} P(X_t, V_t|X_{t-1}^{(r)}, V_{t-1}^{(r)}), \tag{7.11}$$

where $P(Z_t|X_t, V_t)$ is the likelihood and $P(X_t, V_t|X_{t-1}^{(r)}, V_{t-1}^{(r)})$ is given by the motion prior. At time t, we obtain a set of new samples by sampling from Gaussian proposal distributions on both part locations and viewpoint centered on samples at time $t-1$. Then the state of the target at time t, i.e., 3D aspect part locations and viewpoint, is predicted as the MAP of the posterior at time t, which is given by the sample with the largest posterior probability in Eq. (7.11). By sampling new viewpoints, we are able to predict the topological appearance change of the target, so as to apply and update the part templates accordingly. To initialize the tracker, we use the ground truth viewpoint in the first frame of the video, and aspect parts are initialized automatically by projecting the 3D aspect part model according to the viewpoint. Algorithm 1 summarizes our multiview tracking method using Bayesian particle filtering.

7.4 Experiments

We evaluate the performance of our multiview tracker on car tracking, since the ability to track cars is critical for various real world applications and it represents an

```
input  : A video sequence $Z_{1:T}$, initial 3D aspect parts and viewpoint $(X_1, V_1)$
output: 3D aspect parts and viewpoints for the target in the video $(X_t, V_t)_{t=1}^{T}$
1  Initialize samples $(X_1^{(r)}, V_1^{(r)})_{r=1}^{N}$ for the first frame by sampling viewpoints and
   part locations according to the motion prior (7.6) based on $(X_1, V_1)$;
2  for $t \leftarrow 2$ to $T$ do
3      Initialize the MCMC sampler: randomly select a sample $(X_{t-1}^{(r)}, V_{t-1}^{(r)})$ as the
       initial state of the $(X_t, V_t)$ Markov chain;
4      repeat
5          Sample a new viewpoint from the Gaussian proposal density $Q(V_t'; V_t)$;
6          Compute the visibility of 3D aspect parts under viewpoint $V_t'$;
7          foreach part $i$ visible in both $V_t'$ and $V_t$ do
8              Sample its location from the Gaussian proposal density $Q(X_{it}'; X_{it})$;
9          end
10         foreach part $i$ visible in $V_t'$ but not in $V_t$ do
11             Compute its location $X_{it}'$ using the mean distance with respect to
               other visible parts according to the pairwise distributions (7.10);
12         end
13         Compute the acceptance ratio
               $a = \min\Big(1, \frac{P(X_t', V_t'|Z_{1:t})Q(X_t; X_t')Q(V_t; V_t')}{P(X_t, V_t|Z_{1:t})Q(X_t'; X_t)Q(V_t'; V_t)}\Big);$   (7.12)
14         Accept the sample $(X_t', V_t')$ with probability $a$. If accepted,
           $(X_t, V_t) \leftarrow (X_t', V_t')$. Otherwise, leave $(X_t, V_t)$ unchanged;
15     until $N$ samples are accepted;
16     Obtain the new sample set $(X_t^{(r)}, V_t^{(r)})_{r=1}^{N}$, and find the MAP among it as
       the tracking output for frame $t$;
17 end
```

Algorithm 1: Multiview particle filtering object tracking

informative case study in handling topological appearance change.

7.4.1 Datasets

The current benchmarks for evaluating trackers that handle appearance changes (e.g., *Wu et al.* (2013)) are not built to emphasize the 'topological' appearance change of the target. So they are not suitable for evaluating our method whose main goal is to handle the topological appearance changes. Hence, we collected a new car tracking dataset of 9 video sequences that contain significant viewpoint change from YouTube. Each video contains one car to be tracked. To provide ground truth annotations for viewpoints and 3D aspect parts, we use the pose annotation tool proposed in *Xiang et al.* (2014a), which computes accurate viewpoints and 3D aspect part locations of the targets using correspondences between 2D image points and 3D anchor points of CAD models. In order to test our multiview tracker in challenging real world scenarios, we also selected 11 sequences from the KITTI dataset *Geiger et al.* (2012) that contain significant viewpoint change. There can be multiple cars in each sequence, but we specify one car to track. In some sequences, the target is occluded temporarily which makes these sequences challenging. Finally, we evaluate our method on a standard sequence for car tracking from *Kalal et al.* (2012). Unfortunately, this sequence does not contain significant viewpoint variations. Refer to the technical report in *Xiang* (2014) for details of the annotation process and the statistics for the YouTube and the KITTI sequences.

7.4.2 Evaluation Measures

Our multiview tracker outputs not only the 2D bounding box of the target, but also its 3D pose and the 2D locations of the 3D aspect parts. So we evaluate the performance of our tracker on these three tasks and compare it with corresponding baselines. For 2D tracking, we report the Pascal VOC overlap ratio, which is defined as $R = Area(B_T \cap B_{GT})/Area(B_T \cup B_{GT})$, where B_T is the predicted bounding box of the target and B_{GT} is the ground truth bounding box.

For viewpoint estimation, we report two metrics. The first metric is the viewpoint accuracy, where an estimated viewpoint is considered to be correct if the deviation between the estimated azimuth and the ground truth azimuth is within $15°$. The second metric is the absolute difference in azimuth between the ground truth viewpoint and the estimated viewpoint. Since the elevation change is small in the sequences in our experiments, we do not present detailed evaluation in elevation estimation.

For 3D aspect part localization, we also use the Pascal VOC overlap ratio, where the intersection over union is computed between the predicted part shape and the ground truth part shape. If a visible part is predicted as self-occluded, the overlap ratio is zero. So we penalize incorrect aspect estimation of the target. We measure the viewpoint and part locations for the target in one frame only if the target is correctly tracked in the frame, i.e., its overlap ratio with ground truth bounding box is larger than 0.5.

7.4.3 Experimental Settings

The following parameters have been set experimentally and remain fixed for all of the experiments with different sequences. In the motion prior, the standard deviations of part center coordinates in Eq. (7.8) are set to $\sigma_x = 4 \cdot w$ and $\sigma_y = 4 \cdot h$, where w and h denote the width and height of the part respectively. The standard deviations of viewpoint in Eq. (7.9) are set to $\sigma_a = 135°$, $\sigma_e = 5°$ and $\sigma_d = 10$. We use large standard deviations for both part location and viewpoint in order to recover from tracking failures due to occlusions or noisy responses from part templates. In the pairwise potential, both the standard deviations in Eq. (7.10) are set to $\sigma_{dx} = \sigma_{dy} = h/4$. In Eq. (7.5), the constant λ_0 can be arbitrary since we only compare the common visible parts of two samples when selecting the MAP sample (Algorithm 1).

We compute 40 (viewpoints) $\times$ 200K (part locations) samples per frame since the joint space of viewpoint and all parts is huge. To train the templates for 3D aspect parts, we use the 3DObject dataset *Savarese and Fei-Fei* (2007). For the templates in DPM, we use the car model pre-trained on PASCAL'07 *Felzenszwalb et al.* (2010).

7.4.4 Results

2D Object Tracking. Tab. 7.1 shows the 2D object tracking results in terms of average bounding box overlap ratio on our new car tracking dataset, the KITTI sequences and the 06_car sequence from *Kalal et al.* (2012), where we compare our multiview tracker with several baselines. First, four state-of-the-art online tracking methods, MIL *Babenko et al.* (2011), L1 *Bao et al.* (2012a), TLD *Kalal et al.* (2012) and Struct *Hare et al.* (2011), perform poorly on our new dataset and the KITTI sequences. Their mean overlap ratios are below 0.5. This is mainly because these online tracking methods cannot handle the topological appearance change of the cars. When the viewpoint changes, the online trackers keep tracking just a single portion of the object or even lose the target (Fig. 7.4).

It is evident that the category-level part templates contribute significantly in the multiview tracking setting. In Tab. 7.1, "Category Model" column shows the case that we use only the category-level part templates in our particle filtering framework without using online learning (refer to Eq. (7.3)). We can see that "Category Model" improves over the best online tracker by 30% on the new dataset and 19% on the KITTI sequences in terms of mean overlap ratio. By leveraging the 3D aspect part representation and estimating the viewpoint, our "Category Model" is able to predict the aspect change of the target and track the target in different views.

Our full model takes advantages of both category-level part templates and online-learned part appearance models, and it achieves the best mean overlap ratio on the YouTube dataset and the KITTI sequences. The highest improvement is for Race5

Video	MIL	L1	TLD	Struct	DPM+PF	Category	Full
Race1	0.34	0.39	0.20	0.36	0.68	0.68	**0.69**
Race2	0.49	0.49	0.28	0.50	**0.74**	**0.74**	0.73
Race3	0.36	0.26	0.25	0.44	0.74	0.74	**0.77**
Race4	0.53	0.56	0.47	0.63	**0.76**	**0.76**	**0.76**
Race5	0.29	0.54	0.28	0.26	0.63	0.63	**0.68**
Race6	0.27	0.53	0.48	0.29	0.76	0.76	**0.77**
SUV1	0.58	**0.81**	0.56	0.60	0.78	0.78	0.78
SUV2	0.18	0.12	0.53	0.24	**0.77**	**0.77**	**0.77**
Sedan	0.26	0.23	0.33	0.30	**0.78**	**0.78**	**0.78**
Mean	0.37	0.44	0.38	0.40	0.74	0.74	**0.75**
KITTI01	0.20	0.40	0.44	0.33	0.65	0.64	**0.69**
KITTI02	0.28	0.18	0.20	0.12	0.26	0.26	**0.32**
KITTI03	0.37	**0.59**	0.42	0.36	0.20	0.19	0.50
KITTI04	0.31	0.12	**0.36**	0.34	0.67	0.33	0.33
KITTI05	0.40	0.32	0.51	0.41	0.54	**0.73**	0.72
KITTI06	0.64	0.21	0.54	**0.65**	**0.65**	**0.65**	0.56
KITTI07	0.12	0.33	0.03	0.28	**0.66**	0.65	**0.66**
KITTI08	0.58	0.13	0	0.66	**0.74**	**0.74**	0.72
KITTI09	0.18	0.15	0	0.17	0.18	0.51	**0.52**
KITTI10	0.33	0.46	0.41	0.35	**0.68**	**0.68**	**0.68**
KITTI11	0.28	0.23	0.24	0.28	**0.71**	**0.71**	0.68
Mean	0.34	0.28	0.29	0.36	0.54	0.55	**0.58**
06_car	0.19	0.52	**0.85**	0.48	0.70	0.67	0.70

Table 7.1: 2D object tracking performance using average bounding box overlap ratio. Trackers: MIL *Babenko et al.* (2011), L1 *Bao et al.* (2012a), TLD *Kalal et al.* (2012) and Struct *Hare et al.* (2011)

and KITTI03, where “Category Model” fails to track the car due to occlusion by smoke and another car, respectively. By combining online appearance models, the full model can recover from occlusion and track the car by adapting its appearance models. Fig. 7.4 shows some tracking outputs from our multiview tracker on SUV1 and Race1. Fig. 7.5 displays some tracking results on KITTI03, where our full Model recovers from occlusion, but the “Category Model” switches to the occluder.

We also compare our method with a tracking-by-detection baseline, which applies particle filtering to the output of a detector (DPM *Felzenszwalb et al.* (2010)). Our result is on par with this baseline for 2D object localization in the YouTube and 06_car sequences, and we provide 4% improvement on the KITTI dataset. However, note that this baseline and the online trackers baselines are not able to provide the estimates for the viewpoint and aspect part locations.

The results on the 06_car sequence from *Kalal et al.* (2012) demonstrate that our multiview tracker can handle the degenerate case where the viewpoint of the target does not change. MIL, L1 and Struct drift due to occlusion by trees, while TLD is well designed to recover from occlusion and achieves the best performance on this sequence. Our method also recovers from occlusion but obtains lower average overlap ratio than TLD. One main reason is that the elevation angle of the car in this sequence is totally different from that of the instances we used for training the category-level part templates (see *Xiang* (2014) for tracking videos on these datasets).

Continuous Viewpoint Estimation. The left half of Tab. 7.2 shows the viewpoint accuracy and the mean absolute difference in azimuth for viewpoint estimation on our new car dataset and the KITTI sequences. We compare our “Full Model” and “Category Model” with the state-of-the-art object pose estimator ALM *Xiang and Savarese* (2012). Since ALM does not output tracks of targets, we compare the three models on the commonly tracked frames between the “Full Model” and the “Category Model”, where we use the most confident detection with overlap ratio larger than 0.5

	Viewpoint Estimation			3D Aspect Part Localization		
Video	Full	Category	ALM	Full	Category	ALM
Race1	**0.67/18.73°**	0.59/22.88°	0.52/42.62°	**0.40**	0.39	0.35
Race2	**0.77/10.83°**	0.60/12.65°	0.53/44.30°	**0.45**	0.38	0.34
Race3	**0.83**/9.28°	**0.83/7.79°**	0.64/46.08°	0.45	**0.48**	0.31
Race4	0.69/15.83°	0.68/14.67°	**0.79/13.37°**	**0.48**	0.47	0.42
Race5	0.71/**10.75°**	**0.74**/11.78°	0.54/57.79°	**0.44**	0.42	0.28
Race6	**0.43/18.47°**	0.40/21.34°	0.31/37.08°	**0.35**	**0.35**	0.29
SUV1	**0.82/7.81°**	0.75/8.52°	0.47/78.38°	**0.42**	0.40	0.24
SUV2	**0.57/19.56°**	0.45/56.33°	0.39/63.41°	**0.30**	0.23	0.18
Sedan	0.76/9.87°	0.78/**9.50°**	**0.79**/20.84°	0.44	**0.45**	0.43
Mean	**0.69/13.46°**	0.65/18.38°	0.54/47.24°	**0.41**	0.40	0.30
KITTI01	**0.95/6.54°**	0.74/8.53°	0.57/44.46°	**0.49**	0.41	0.37
KITTI02	**1.00/5.40°**	0.20/30.06°	0.33/119.54°	**0.60**	0.15	0.13
KITTI03	**0.42**/15.64°	**0.42/15.14°**	0.50/15.99°	**0.33**	**0.33**	0.24
KITTI04	0.22/27.05°	**0.25/26.03°**	0.17/58.42°	**0.22**	**0.22**	0.14
KITTI05	0.36/23.59°	0.40/**22.17°**	**0.64**/23.65°	0.23	**0.25**	**0.25**
KITTI06	0.31/21.63°	0.29/21.58°	**0.59/20.29°**	0.21	0.21	**0.23**
KITTI07	**0.96/6.86°**	0.89/7.92°	0.70/24.50°	**0.48**	**0.48**	0.39
KITTI08	0.57/**15.61°**	0.48/23.84°	**0.67**/23.26°	**0.37**	0.29	0.26
KITTI09	**0.50**/21.63°	0.42/78.67°	**0.50/17.60°**	**0.28**	0.16	0.23
KITTI10	**0.81/7.99°**	0.79/9.44°	0.44/56.78°	**0.39**	**0.39**	0.21
KITTI11	**0.88/9.33°**	0.78/11.80°	0.68/12.29°	0.39	0.40	**0.41**
Mean	**0.63/14.66°**	0.51/23.20°	0.53/37.89°	**0.36**	0.30	0.26

Table 7.2: Viewpoint accuracy/mean absolute difference in azimuth and average overlap ratio of 3D aspect part on our new car dataset and the KITTI sequences.

from ALM as its output. It is clear that "Category Model" outperforms ALM in viewpoint estimation significantly. By utilizing the temporal information from videos, our multiview tracker estimates continuous viewpoints in the particle filtering framework and smoothes the viewpoint estimation via the motion prior. ALM discretizes the viewpoint space into 24 azimuth angles (i.e., 15° interval) and it does not use the temporal information. By combining online appearance models for 3D aspect parts, our full model improves over the "Category Model" by 4%/5° and 12%/9°, and over ALM by 15%/34° and 10%/23° in terms of mean accuracy/mean absolute difference in azimuth on the two datasets respectively. Online appearance models help 2D localization of 3D aspect parts, which in turn benefits viewpoint estimation. Our full model achieves 4.6° mean absolute difference in elevation on the YouTube sequences. Fig. 7.4 also shows some viewpoint estimation results from our multiview tracker and ALM.

3D Aspect Part Localization. The right half of Tab. 7.2 shows the 3D aspect part localization performance in terms of PASCAL VOC overlap ratio on our new car dataset and the KITTI sequences. Compared with ALM *Xiang and Savarese* (2012), "Category Model" achieves much better mean overlap ratio. Since part locations and viewpoint are jointly optimized in our multiview tracking framework, the category-level part templates and the motion prior result in accurate viewpoint and 2D part locations. Consequently, the 2D part shapes can be estimated more accurately. By introducing online appearance learning, our full model further improves the 3D aspect part localization, where it outperforms or is on par with the "Category Model" in 7 of the 9 YouTube sequences and in 9 of the 11 KITTI sequences. In Fig. 7.4, we can see that the 3D aspect parts from our tracker are more accurate than those obtained by ALM.

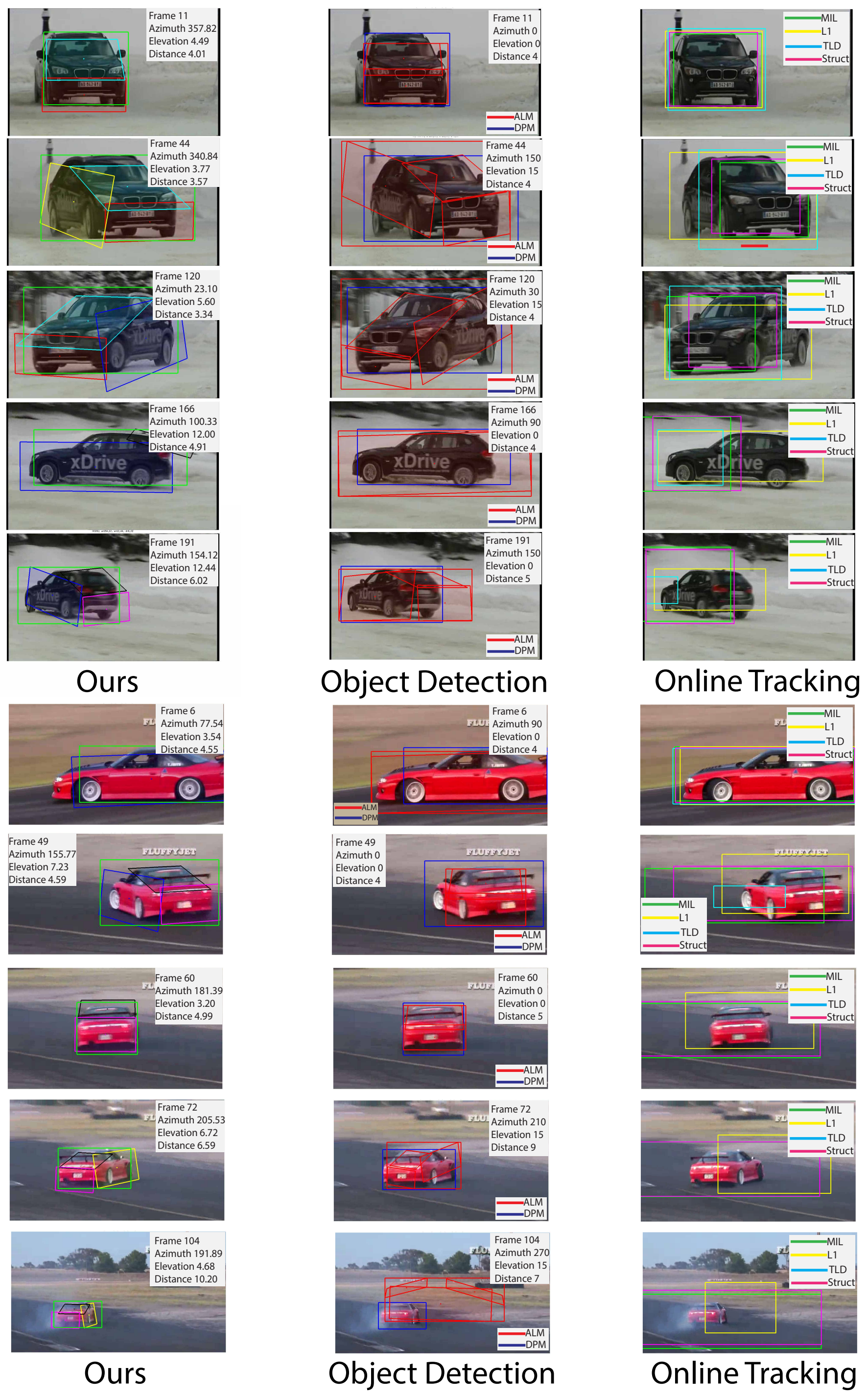


Figure 7.4: Tracking/Detection outputs from different methods on SUV1 and Race1. "Ours" are the tracking outputs from our multiview tracker. "Object Detection" shows the detection results from DPM *Felzenszwalb et al.* (2010) and ALM *Xiang and Savarese* (2012). "Online Tracking" shows the tracking results of four state-of-the-art online tracking methods: MIL *Babenko et al.* (2011), L1 *Bao et al.* (2012a), TLD *Kalal et al.* (2012) and Struct *Hare et al.* (2011).

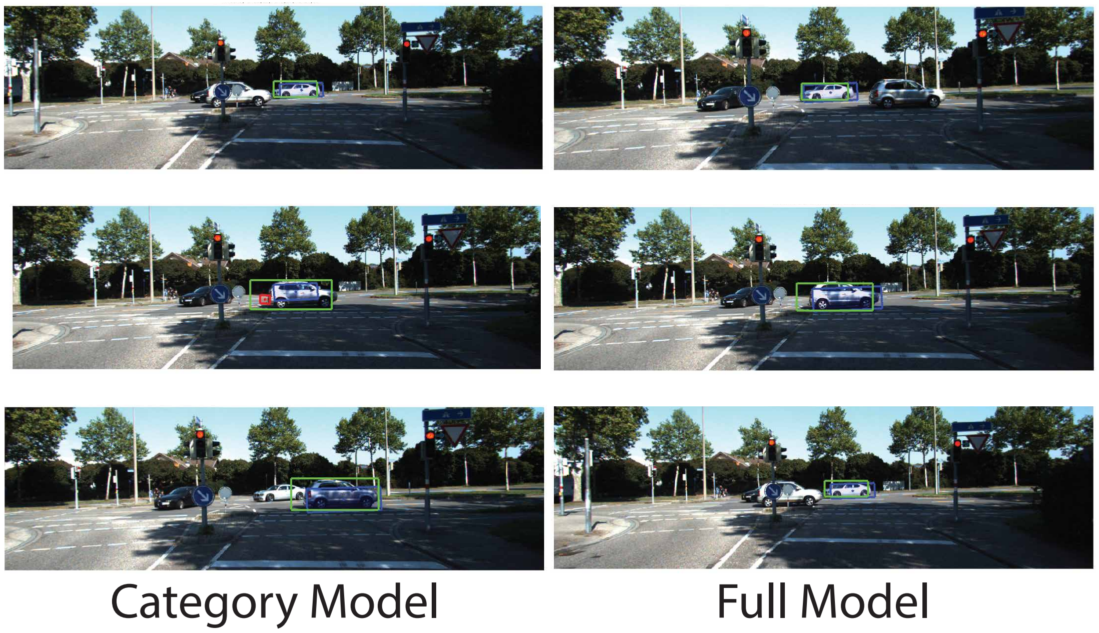


Figure 7.5: The tracking results on KITTI03. "Category Model" fails to track the target and switches to the occluder, while our full model is able to recover from occlusion and track the correct target.

7.5 Conclusion

We proposed a novel multiview rigid object tracking framework to handle the topological appearance change of objects caused by viewpoint transitions. Our multiview tracker is able to predict the aspect change of the target, and track the continuous pose and the 3D aspect parts of the target. We conducted experiments on a new challenging car dataset and a set of KITTI sequences with large viewpoint variations, as well as on a standard sequence for car tracking. We demonstrated that our method is effective in tracking continuous 3D pose and aspect part locations, and it is able to handle the changes in viewpoint robustly.

CHAPTER VIII

Application III: CNN-based Object Detection with 3D Voxel Patterns

8.1 Introduction

Convolutional Neural Networks (CNNs) have become dominating in solving different recognition problems recently, such as image classification *Krizhevsky et al.* (2012), object detection *Girshick et al.* (2014), and image description generating *Karpathy and Fei-Fei* (2014). CNNs are powerful due to their capability in both representation and learning. With millions of weights in the contemporary CNNs, they are able to learn much richer representations from data compared to traditional "non-CNN" methods. On the other hand, we believe the intuitions used for designing the traditional methods can also be applied to help us invent new CNN architectures that can effectively solve different problems. In this paper, we explore how subcategory information, which is widely exploited in traditional object detection methods, can help CNN-based object detection.

One main challenge in object category detection is how to handle the appearance change of objects in images due to different factors, such as intra-class variability, object pose variation, scale change, occlusion, and truncation. In traditional object detection methods, training a holistic model to handle all these challenging factors

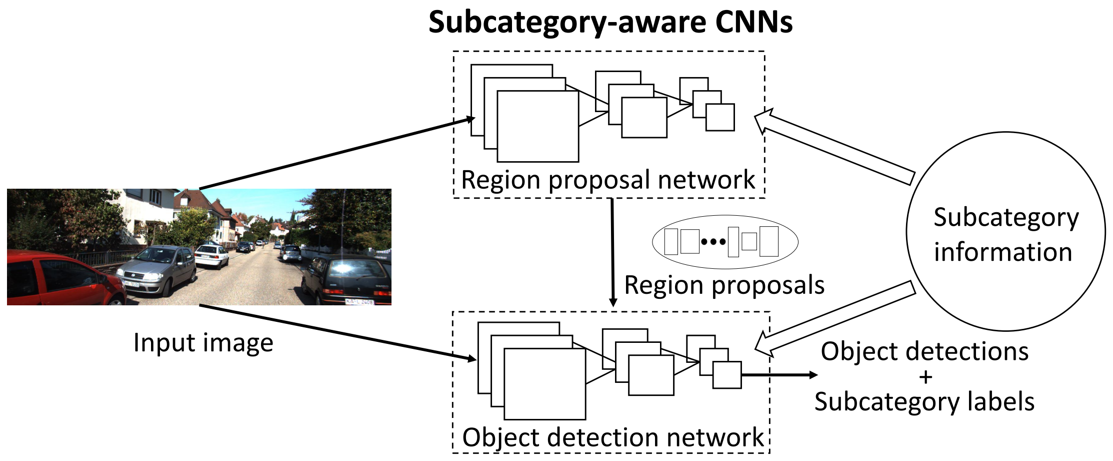


Figure 8.1: Overview of our object detection framework. By exploiting subcategory information, we propose a new CNN architecture for region proposal and a new object detection network for joint detection and subcategory classification.

is overwhelming. So the concept of subcategory is introduced. Instead of building one model for a category, often times a mixture of models is constructed, with each model capturing a subcategory. For example, in the Deformable Part Model (DPM) *Felzenszwalb et al.* (2010), a mixture of HOG *Dalal and Triggs* (2005) templates is trained for each category, where each template captures objects with a specific range of aspect ratios. In the recently introduced 3D Voxel Pattern (3DVP) representation *Xiang et al.* (2015b), each 3DVP captures objects with similar pose, occlusion and truncation, and a detector is trained for each 3DVP. As we can see from these examples, the subcategory concept has been widely utilized and can be generalized beyond nameable subcategories. For instance, a subcategory can be objects with similar 2D appearance, or objects with similar 3D pose or 3D shape. However, the subcategory concept has not been fully explored in CNN-based object detection methods *Girshick et al.* (2014); *Girshick* (2015); *Ren et al.* (2015).

In this work, we propose subcategory-aware CNNs for object detection. We introduce a novel CNN architecture that uses subcategory information to generate region proposals for object detection, as well as a new network for joint detection and subcategory classification. Specifically, our detection method operates in the two-stage pipeline proposed in *Girshick et al.* (2014). In the first stage, a set of region proposals are generated from an image. In the second stage, these region proposals are classified and their locations are refined to achieve better detection results. Fig. 8.1 illustrates our object detection framework.

For generating region proposals, bottom-up segmentation-based methods are widely used *Uijlings et al.* (2013); *Zitnick and Dollár* (2014); *Arbelaez et al.* (2014). However, these methods are not able to handle objects in complex scenes with significant scale variations such as in autonomous driving *Geiger et al.* (2012); *Chen et al.* (2015a). We propose a novel Region Proposal Network (RPN) which utilizes subcategory information to guide the region generating process. Motivated by the traditional detection methods that train a template or a detector for each subcategory, we introduce a *subcategory convolutional (conv) layer* in our RPN, where each filter in the conv layer is trained discriminatively for subcategory detection. The subcategory conv layer outputs heat maps about the presence of certain subcategories at a specific location and scale. Using these heat maps, our RPN is able to output confident subcategory detections as proposals.

For classifying region proposals and refining their locations, we introduce a new object detection network by injecting subcategory information into the network proposed in Fast R-CNN *Girshick* (2015). Our detection network is able to perform object detection and subcategory classification jointly. In addition, in both our RPN and our detection CNN, we use image pyramids as input, and we introduce a new *feature extrapolating layer* to efficiently compute conv features in multiple scales. In this way, our method is able to detect object categories with large scale variations.

We conduct experiments on the KITTI detection benchmark *Geiger et al.* (2012) and the PASCAL3D+ dataset *Xiang et al.* (2014a). By discovering subcategories related to object pose, our method is able to jointly detect objects and estimate their poses. Comparisons with the state-of-the-art detection methods on the two benchmarks demonstrate the advantages of our subcategory-aware CNNs for object detection.

8.2 Related Work

Subcategory in Object Detection. Subcategory has been widely utilized to facilitate object detection, and different methods of discovering object subcategories have been proposed. In DPM *Felzenszwalb et al.* (2010), subcategories are discovered by clustering objects according to the aspect ratio of their bounding boxes. *Gu and Ren* (2010) performs clustering according to the viewpoint of the object to discover subcategories. Visual subcategories are constructed by clustering in the appearance space of object *Divvala et al.* (2012); *Ohn-Bar and Trivedi* (2015); *Chen et al.* (2014); *Divvala et al.* (2014). 3DVP *Xiang et al.* (2015b) performs clustering in the 3D voxel space according to the visibility of the voxels. Unlike previous works, we utilize subcategory to improve CNN-based detection, and our framework is general to employ different types of object subcategories.

CNN-based Object Detection. We can categorize the state-of-the-art CNN-based object detection methods into two classes: one-stage detection and two-stage detection. In one-stage detection, such as the Overfeat *Sermanet et al.* (2013) framework, a CNN directly processes an input image, and outputs object detections. In two-stage detection, such as R-CNNs *Girshick et al.* (2014); *Girshick* (2015); *Ren et al.* (2015), region proposals are first generated from an input image, where different region proposal methods can be employed *Uijlings et al.* (2013); *Zitnick and Dollár* (2014); *Arbelaez et al.* (2014). Then these region proposals are fed into a CNN for classification and location refinement. It is debatable which detection paradigm is better. We adopt the two-stage detection framework in this work, and consider the region proposal process to be the coarse detection step in coarse-to-fine detection *Viola and*

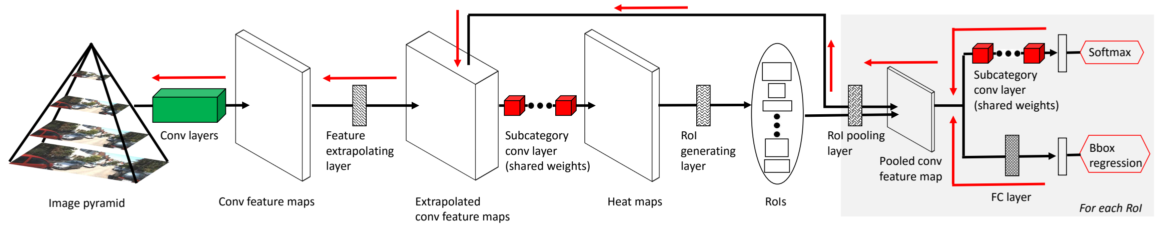


Figure 8.2: Architecture of our region proposal network. Red arrows indicate the route of derivatives in back-propagation training.

Jones (2004). We propose a novel region proposal network motivated by *Ren et al.* (2015), and demonstrate its advantages over the previous region proposal methods by injecting subcategory information into CNNs.

8.3 Subcategory-aware RPN

Ideally, we want to have a region proposal approach that can cover objects in an input image with as few proposals as possible. Since objects in images appear at different locations and different scales, region proposal itself is a challenging problem. Recently, *Ren et al.* (2015) proposed to tackle the region proposal problem with CNNs, demonstrating the advantages of using CNNs over traditional approaches for region proposal. In this section, we describe our subcategory-aware Region Proposal Network (RPN).

8.3.1 Network Architecture

We introduce a novel network architecture for generating object proposals from images. The architecture is inspired by the traditional sliding-window-based object detectors, such as the Aggregated Channel Feature (ACF) detector *Dollár et al.* (2014) and the Deformable Part Model (DPM) *Felzenszwalb et al.* (2010). Fig. 8.2 illustrates the architecture of our region proposal network. i) To handle different scales of objects, we input into our RPN an image pyramid. This pyramid is processed by several convolutional (conv) and max pooling layers to extract the conv feature maps, with one conv feature map for each scale. ii) In order to speed up the computation of conv features on image pyramids, we introduce the *feature extrapolating layer*, which generates feature maps for scales that are not covered by the image pyramid via extrapolation. iii) After computing the extrapolated conv feature maps, we specifically design a conv layer for object subcategory detection, where each filter in the conv layer corresponds to an object subcategory. We train these filters to make sure they fire on correct locations and scales of objects in the corresponding subcategories during the network training. The *subcategory conv layer* outputs a heat map for each scale, where each value in the heat map indicates the confidence of an object in the corresponding location, scale and subcategory. v) Using the subcategory heat maps, we design a *RoI generating layer* that generates object candidates (RoIs) by thresholding the heat maps. vi) The RoIs are used in a *RoI pooling layer* Girshick (2015) to pool conv features from the extrapolated conv feature maps. vii) Finally, our RPN terminates at two sibling layers: one that outputs softmax probability estimates over object subcategories, and the other layer that refines the RoI location with a bounding box regressor.

8.3.2 Feature Extrapolating Layer

In our RPN, we use fixed-size conv filters in the subcategory conv layer to localize objects (e.g., 5×5 conv filters). In order to handle different scales of objects, we resort to image pyramids. An image pyramid consists of images with different resolutions obtained by rescaling the original image according to different sampled scales. After constructing the image pyramid for an input image, multi-resolution conv feature maps can be computed by applying several conv layers and max pooling layers to each image in the pyramid (Fig. 8.2). If we perform convolution on every scale explicitly, it is computationally expensive, especially when a finely-sampled image pyramid is needed as in the region proposal process. In *Dollár et al.* (2014), Dollár et al. demonstrate that multi-resolution image features can be approximated by extrapolation from nearby scales rather than being computed explicitly. Inspired by their work, we introduce a feature extrapolating layer to accelerate the computation of conv features on an image pyramid.

Specifically, a feature extrapolating layer takes as input N feature maps that are supplied by the last conv layer for feature extraction, where N equals to the number of scales in the input image pyramid. Each feature map is a multi-dimensional array of size $H \times W \times C$, with H rows, W columns, and C channels. The width and height of the feature map corresponds to the largest scale in the image pyramid, where images in smaller scales are padded with zeros in order to generate feature maps with the same size. The feature extrapolating layer constructs feature maps at intermediate scales by extrapolating features from the nearest scales among the N scales using bilinear interpolation. Suppose we add M intermediate scales between every ith scale and $(i+1)$th scale, $i = 1, \ldots, N-1$. The output of the feature extrapolating layer is $N' = (N-1)M + N$ feature maps, each with size $H \times W \times C$. Since extrapolating a multi-dimensional array is much faster than computing a conv feature map explicitly, the feature extrapolating layer speeds up the feature computation on image pyramids while using less memory.

8.3.3 Subcategory Conv Layer

After computing the conv feature maps, we design a subcategory conv layer for subcategory detection. Motivated by the traditional object detection methods that train a classifier or a template for each subcategory *Felzenszwalb et al.* (2010); *Malisiewicz et al.* (2011); *Xiang et al.* (2015b), we train a conv filter in the subcategory conv layer to detect a specific subcategory. Suppose there are K subcategories to be considered. Then, the subcategory conv layer consists of $K + 1$ conv filters with one additional conv filter for a special “background” category. For multi-class detection (e.g., car, pedestrian, cyclist, etc.), the K subcategories are the aggregation of all the subcategories from all the classes. These conv filters operate on the extrapolated conv feature maps and output heat maps that indicate the confidences of the presence of objects in the input image. We use fixed-size conv filters in this layer (e.g., $5 \times 5 \times C$ conv filters), which are trained to fire on specific scales in the feature pyramid. Sec. 8.3.5 explains how we back-propagate errors from the loss layer to train these subcategory conv filters.

8.3.4 RoI Generating Layer

The RoI generating layer takes as input N' heat maps and outputs a set of region proposals (RoIs), where N' is the number of scales in the feature pyramid after extrapolation. Each heat map is a multi-dimensional array of size $H \times W \times K$ for K subcategories (i.e., for RoI generating, we ignore the “background” channel in the heat map). The RoI generating layer first converts each heat map into a $H \times W$ 2D array by performing max operation over the channels for subcategory. Then, it thresholds the 2D heat map to generate RoIs. In this way, we measure the objectness of a region by aggregating information from subcategories. Different generating strategies are used in testing and training.

In testing, each location (x, y) in a 2D heat map with a score larger than a predefined threshold is used to generate RoIs. First, a canonical bounding box is centered on (x, y). The width and height of the box are the same as those of the conv filters (e.g., 5×5) in the subcategory conv layer, which have an aspect ratio one. Second, a number of boxes centered on (x, y) with the same areas as the canonical box (e.g., 25) but with different aspect ratios are generated. Finally, the RoI generating layer rescales the generated boxes according to the scale of the heat map, so these RoIs can cover objects in different scales and aspect ratios.

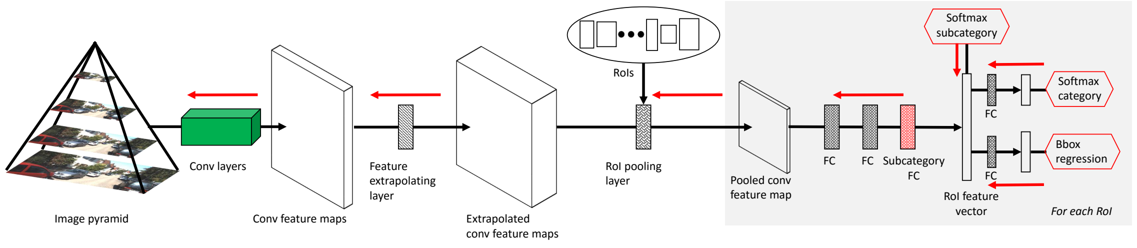


Figure 8.3: Architecture of our object detection network. Red arrows indicate the route of derivatives in back-propagation training.

In training, the RoI generating layer outputs hard positive RoIs and hard negative RoIs for training the subcategory conv filters, given a budget on batch size in stochastic gradient descent. First, we use the same procedure as described in testing to generate a number of bounding boxes for each location in each heat map. Second, according to the ground truth bounding boxes of objects in a training image, we compute the intersection over union (IoU) overlap between the generated boxes and the ground truth boxes. Bounding boxes with IoU overlap larger/smaller than some threshold (e.g., 0.5) are considered to be positive/negative. Finally, given the number of RoIs to be generated for each training image R (i.e., batch size divided by the number of images in a batch), the RoI generating layer outputs $R \times \alpha$ hard positives (i.e., $R \times \alpha$ positive bounding boxes with lowest scores in the heat maps) and $R \times (1-\alpha)$ hard negatives (i.e., $R \times (1-\alpha)$ negative bounding boxes with highest scores in the heat maps), where $\alpha \in (0, 1)$ is the percentage of positive examples.

8.3.5 Multi-task Loss

After generating RoIs, we apply the RoI pooling layer proposed in *Girshick* (2015) to pool conv features for each RoI. Then the pooled conv features are used for two tasks: subcategory classification and bounding box regression. As illustrated in Fig. 8.2, our RPN has two sibling output layers. The first layer outputs a discrete probability distribution $p = (p_0, \dots, p_K)$, over $K + 1$ subcategories, which is computed by applying a softmax function over the $K+1$ outputs of the subcategory conv layer. The second layer outputs bounding box regression offsets $t^{k'} = (t_x^{k'}, t_y^{k'}, t_w^{k'}, t_h^{k'}), k' = 0, 1, \dots, K'$ for K' object classes ($K' \ll K$). We parameterize $t^{k'}$ as in *Girshick et al.* (2014), which specifies a scale-invariant translation and log-space width/height shift relative to a RoI.

We employ a multi-task loss to train our RPN for joint subcategory classification and bounding box regression:

$$L(p, k^*, k'^*, t, t^*) = L_{\text{subcls}}(p, k^*) + \lambda[k'^* \geq 1] L_{\text{loc}}(t, t^*), \tag{8.1}$$

where k^* and k'^* are the truth subcategory label and the true class label respectively, $L_{\text{subcls}}(p, k^*) = -\log p_{k^*}$ is the standard cross-entropy loss, $t^* = (t_x^*, t_y^*, t_w^*, t_h^*)$ is the true bounding box regression targets for class k'^*, and $t = (t_x, t_y, t_w, t_h)$ is the prediction for class k'^*. We use the smoothed L_1 loss defined in *Girshick* (2015) for the bounding box regression loss $L_{\text{loc}}(t, t^*)$. The indicator function $[k'^* \geq 1]$ indicates that bounding box regression is ignored if the RoI is background (i.e., $k'^* = 0$). λ is a predefined weight to balance the two losses.

In training, derivatives from the loss function are back-propagated. Red arrows in Fig. 8.2 indicate the propagation route. The two subcategory conv layers in our RPN share their weights. These weights/conv filters are updated according to the derivatives from the softmax loss function for subcategory classification, so we are able to train these filters for subcategory detection. There is no derivative flow in computing heat maps using the subcategory conv layer and in the RoI generating layer. Finally, our RPN generates confident subcategory detections as region proposals.

8.4 Subcategory-aware Detection Network

After the region proposal process, CNNs are utilized to classify these proposals and refine their locations *Girshick et al.* (2014); *Girshick* (2015); *Ren et al.* (2015). Since region proposal significantly reduces the search space (e.g., several thousand regions per image), more powerful CNNs can be used in the detection step, which usually contain several fully connected layers with high dimensions. In this section, we introduce our subcategory-aware object detection network, where we use subcategory information to facilitate object detection and accomplish the task of joint detection and subcategory classification.

8.4.1 Network Architecture

Fig. 8.3 illustrates the architecture of our detection network. The network is constructed based on the Fast R-CNN detection network *Girshick* (2015) with a number of improvements. i) We use image pyramids to handle the scale variation of objects. After the last conv layer for feature extraction, we add the feature extrapolating layer to increase the number of scales in the conv feature pyramid. ii) Given the region proposals generated from our RPN, we employ a RoI pooling layer to pool conv features for each RoI. Each RoI is mapped to a scale in the conv feature pyramid such that smaller RoIs pool features from larger scales. iii) The pooled conv features are fed into three fully connected (FC) layers, where the last FC layer is designed for subcategory classification. For K subcategories, the "subcategory FC" layer outputs a $K + 1$ dimensional vector with one additional dimension for the background class. We consider the output, named *RoI feature vector*, to be an embedding in the subcategory space. iv) Finally, the network terminates at three output layers. The first output layer applies a softmax function directly on the output of the "subcategory FC" layer for subcategory classification. The other two output layers operate on the RoI feature vector and apply FC layers for object class classification and bounding

	#image	#car	#pedestrian	#cyclist
Train set	3,682	14,898	3,154	916
Validation set	3,799	13,714	1,333	711
Total	7,481	28,612	4,487	1,627

Table 8.1: Statistics on the KITTI training set.

box regression.

8.4.2 Multi-task Loss

We train our object detection network with a multi-task loss for joint object class classification, subcategory classification and bounding box regression:

$$\begin{aligned} &L(p, k^*, p', k'^*, t, t^*) = \\ &L_{\text{subcls}}(p, k^*) + \lambda_1 L_{\text{cls}}(p', k'^*) + \lambda_2 [k'^* \geq 1] L_{\text{loc}}(t, t^*), \end{aligned} \tag{8.2}$$

where $p = (p_0, \dots, p_K)$ is a probability distribution over $K + 1$ subcategories, $p' = (p'_0, \dots, p'_{K'})$ is a probability distribution over $K' + 1$ object classes, k^* and k'^* are the truth subcategory label and the true class label respectively, t and t^* are the predicted vector and the true vector for bounding box regression respectively, and λ_1 and λ_2 are predefined weights to balance the losses of different tasks. $L_{\text{subcls}}(p, k^*) = -\log p_{k^*}$ and $L_{\text{cls}}(p', k'^*) = -\log p'_{k'^*}$ are the standard cross-entropy loss, and $L_{\text{loc}}(t, t^*)$ is the smoothed L_1 loss as in our RPN. In back-propagation training, derivatives for the multi-task loss are back-propagated to the previous layers. Red arrows in Fig. 8.3 indicate the route of the derivative flow.

	#image	#aeroplane	#bicycle	#boat	#bottle	#bus	#car
Train	5,717	470	410	508	749	317	1,191
Test	5,823	484	380	491	733	320	1,173
	#chair	#table	#mbike	#sofa	#train	#monitor	
Train	1,457	373	375	399	327	412	
Test	1,449	374	376	387	329	414	

Table 8.2: Statistics on the PASCAL3D+ dataset.

8.5 Experiments

8.5.1 Experimental Settings

Datasets. We evaluate our object detection framework on the KITTI detection benchmark *Geiger et al.* (2012) and the PASCAL3D+ dataset *Xiang et al.* (2014a). i) The KITTI dataset consists of video frames from autonomous driving scenes, with 7,481 images for training and 7,518 images for testing. Car, pedestrian and cyclist are annotated and evaluated for object detection. Since the ground truth annotations of the KITTI test set are not released, we split the KITTI training images into a train set and a validation set to conduct analyses about our method. We follow the same splitting as in *Xiang et al.* (2015b). Table 8.1 summarizes the statistics on the KITTI training set. ii) The PASCAL3D+ dataset augments 12 rigid categories in the PASCAL VOC 2012 *Everingham et al.* (b) with 3D annotations. Each object in the 12 categories is registered with a 3D CAD model. The *train* set of PASCAL VOC 2012 is used for training (5,717 images), while the *val* set is used for testing (5,823 images). Table 8.2 summarizes the statistics on PASCAL3D+.

Evaluation Metrics. On KITTI, we evaluate our detection framework at three levels of difficulty as suggested by *Geiger et al.*, i.e., easy, moderate and hard, where the difficulty is measured by the minimal scale of object to be considered and the occlusion and truncation of the object. Average Precision (AP) *Everingham et al.* (b) is used to measure the detection performance, where 70%, 50%, and 50% overlap thresholds are adopted by the KITTI benchmark for car, pedestrian and cyclist respectively. To evaluate joint detection and orientation estimation on KITTI, *Geiger et al.* (2012) proposes a new metric called Average Orientation Similarity (AOS), which evaluates the orientation similarity between detections and ground truths at different detection recalls. On PASCAL3D+, the standard AP with 50% overlap ratio is adopted to evaluate object detection. For joint detection and pose estimation, we use the Average Viewpoint Precision (AVP) suggested by *Xiang et al.* (2014a), where a detection is considered to be a true positive if its location and viewpoint are both correct.

Subcategories. Different approaches can be utilized to discover subcategories, such as clustering based on object appearance *Ohn-Bar and Trivedi* (2015) or clustering based on aspect ratio of the object bounding box *Felzenszwalb et al.* (2010). Specifically, in our implementation, we adopt the 3D Voxel Pattern (3DVP) representation *Xiang et al.* (2015b) for rigid objects (i.e., car in KITTI and the 12 categories in PASCAL3D+), which jointly models object pose, occlusion and truncation in the clustering process. Each 3DVP is considered to be a subcategory. For pedestrian and cyclist in KITTI, we perform clustering according to the orientation of the object, and each cluster is considered to be a subcategory. In this way, we are able to estimate the orientation/pose of object by conducting subcategory classification, where we transfer the orientation/pose of the subcategory to the detected object. For validation on KITTI, we use 173 subcategories (125 3DVPs for car, 24 poses for pedestrian and cyclist each), while for testing on KITTI, we use 275 subcategories (227 3DVPs for car, 24 poses for pedestrian and cyclist each). 3DVPs are discovered with affinity propagation clustering *Frey and Dueck* (2007), which automatically discovers the number of clusters. For PASCAL3D+, 337 3DVPs are discovered among the 12 categories. Correspondingly, the output number of the subcategory conv layer in our RPN and that of the subcategory FC layer in our detection network equal to the number of subcategory plus one.

Region Proposal Network Hyper-parameters. In our RPN, we use 5 scales for KITTI in the input image pyramid $(0.25, 0.5, 1.0, 2.0, 3.0)$ and 4 scales for PASCAL3D+ $(0.25, 0.5, 1.0, 2.0)$, where each number indicates the rescaling factor with respect to the original image size. Objects in PASCAL3D+ have smaller scale variation compared to objects in KITTI. Adding larger scales for PASCAL3D+ only results in marginal improvement but significantly increases the computation. The feature extrapolating layer extrapolates 4 scales with equal intervals between every two input scales, so the final conv feature pyramid has 21 scales for KITTI and 16 scales for PASCAL3D+. In the RoI generating layer, each location in a heat map generates 7 boxes with 7 different aspect ratios $(3.0, 2.0, 1.5, 1.0, 0.75, 0.5, 0.25)$ for KITTI and 5 aspect ratios $(3.0, 2.0, 1.0, 0.5, 0.25)$ for PASCAL3D+, where each number indicates the ratio between the height and the width of the bounding box. In training the RPN, each SGD mini-batch is constructed from a single image, chosen uniformly at random. A mini-batch has size 128, with 64 positive RoIs and 64 negative RoIs, where the IoU threshold is 70% for both KITTI and PASCAL3D+.

Detection Network Hyper-parameters. In our detection network, we use 4 scales in the input image pyramid $(1.0, 2.0, 3.0, 4.0)$ for KITTI and 2 scales $(1.0, 2.0)$ for PASCAL3D+, both with 4 scales extrapolated between every two scales. Each SGD mini-batch is constructed from 2 images. A mini-batch has size 128, with 64 RoIs from each image. 25% of the RoIs are positive, where the IoU threshold is 70% for car in KITTI, and 50% for the other categories. The same SGD hyper-parameters are used as in *Girshick* (2015) for both region proposal and detection.

Fine-tuning Pre-trained Networks. We implement our detection framework in Caffe *Jia et al.* (2014). Instead of training our RPN and detection CNN from scratch, we initialize the conv layers for feature extraction in both networks and the two FC layers before subcategory FC layer in the detection network with pre-trained networks on ImageNet *Russakovsky et al.* (2015b). On KITTI, we fine-tune the AlexNet *Krizhevsky et al.* (2012) from R-CNN *Girshick et al.* (2014). On PASCAL3D+, we

Methods	Easy	Moderate	Hard	#boxes
Car				
Selective Search *Uijlings et al.* (2013)	71.91	56.96	51.41	6k
Edge Boxes *Zitnick and Dollár* (2014)	81.40	61.84	55.68	2k
RPN *Ren et al.* (2015)	98.84	**97.37**	**95.31**	2k
Ours	**99.27**	96.28	93.14	2k
Pedestrian				
Selective Search *Uijlings et al.* (2013)	80.28	69.76	63.70	6k
Edge Boxes *Zitnick and Dollár* (2014)	86.15	71.88	65.39	2k
RPN *Ren et al.* (2015)	98.88	91.69	88.64	2k
Ours	**99.44**	**93.46**	**91.02**	2k
Cyclist				
Selective Search *Uijlings et al.* (2013)	78.37	70.49	70.45	6k
Edge Boxes *Zitnick and Dollár* (2014)	56.11	46.52	45.72	2k
RPN *Ren et al.* (2015)	96.55	91.80	89.41	2k
Ours	**99.67**	**93.03**	**91.64**	2k

Table 8.3: Region proposal performance in terms of recall on the KITTI validation set.

fine-tune the deep VGG16 network from *Simonyan and Zisserman* (2014). Since we uses more scales on KITTI, we cannot use the VGG16 network due to GPU memory constraints.

8.5.2 Analysis on KITTI Validation Set

In this section, we perform detailed analyses on our detection framework by conducting experiments using the train-validation splitting of the KITTI training images.

Region Proposal Evalutaion on Recall. We evaluate the detection recall of our RPN and compare it with the state-of-the-art methods in Table 8.3 on the KITTI validation set. First, two popular methods that work well on PASCAL VOC *Everingham et al.* (b) for region proposal, Selective Search *Uijlings et al.* (2013) and Edge Boxes *Zitnick and Dollár* (2014), do not perform well on KITTI, mainly because objects in KITTI exhibit more significant scale variation, occlusion and truncation. It is challenging for a bottom-up proposal method to achieve high recall under a small budget (e.g., 2k boxes per image). For Selective Search, there is no direct way to con-

	Object Detection (AP)			Orientation (AOS)		
Methods	Easy	Moderate	Hard	Easy	Moderate	Hard
Car						
RPN *Ren et al.* (2015)+Ours (unshared)	89.29	82.58	70.12	87.70	80.47	67.83
RPN *Ren et al.* (2015)+Ours (shared)	87.67	82.21	70.10	86.58	80.27	67.90
Ours (unshared)	**95.77**	**86.64**	**74.07**	**94.55**	**85.03**	**72.21**
Pedestrian						
RPN *Ren et al.* (2015)+Ours (unshared)	83.07	69.32	63.46	71.43	58.67	53.58
RPN *Ren et al.* (2015)+Ours (shared)	82.73	68.28	62.30	70.31	56.94	51.87
Ours (unshared)	**86.43**	**69.95**	**64.03**	**73.91**	**58.91**	**53.79**
Cyclist						
RPN *Ren et al.* (2015)+Ours (unshared)	69.23	54.83	51.41	61.25	46.44	43.07
RPN *Ren et al.* (2015)+Ours (shared)	71.24	56.69	52.91	63.21	48.68	45.16
Ours (unshared)	**74.92**	**59.13**	**55.03**	**65.79**	**50.46**	**46.57**

Table 8.4: AP/AOS comparison between different detection methods on the KITTI validation set.

trol the number of proposals per image. Its "fast" mode generates around 6k boxes per image on KITTI. Second, the RPN in Faster R-CNN *Ren et al.* (2015) performs much better than Selective Search and Edge Boxes, which demonstrates the ability of discriminatively trained CNNs for region proposal. But we have to increase its parameter setting from 3 scales and 3 aspect ratios in *Ren et al.* (2015) to 10 scales and 7 aspect ratios in order to make it work on KITTI. Finally, our RPN performs on par with Faster R-CNN on car, and outperforms it on pedestrian and cyclist using the same number of proposals per image. The new architecture we introduce can better handle scale variation using image pyramid. It also benefits from data mining hard training examples in our RoI generating layer.

Region Proposal Evalutaion on Detection and Oritentaion Estimation. Detection recall measures the coverage of region proposals, which cannot demonstrate the quality of the region proposals for detection. In this experiment, we directly measure the detection and orientation estimation performance using different region proposals. Table 8.4 presents the detection and orientation estimation results using RPN in Faster R-CNN *Ren et al.* (2015) and the RPN we propose, while keeping the detection network the same as described in Sec. 8.4. We compare our RPN with two variations of the RPN in Faster R-CNN. For the first model, the RPN and the detection network are trained independently to each other ("unshared"). For the second model, the RPN and the detection network share their conv layers for feature extraction in order to save computation on convolution ("shared"). The sharing is achieved by the four-step alternating optimization training algorithm described in *Ren et al.* (2015). By comparing the two models in Table 8.4, we find that sharing conv layers hurts the performance on car and pedestrian, but improves the performance on cyclist. According to the statistics in Table 8.1, car and pedestrian have much more training examples available than cyclist. With enough training data, the RPN and the detection network trained independently can develop conv features suitable for its own task. In this case, shared conv features degrade the performance. However, when the training data is insufficient, sharing conv features can help.

In Table 8.4, by using region proposals from our RPN, we achieve better performance on detection and orientation estimation across all the three categories. The experimental results demonstrate the advantages of our RPN. We also tried to share the conv layers in our RPN and our detection network. However, since the architecture of our RPN after the conv layers for feature extraction is quite different from that of the detection network, we found that the training cannot converge, which verifies our observation that the RPN and the detection network have developed their own conv features that are suitable for its own task.

	Object Detection (AP)			Orientation (AOS)		
Methods	Easy	Moderate	Hard	Easy	Moderate	Hard
Car						
Faster R-CNN	82.91	77.83	66.25	N/A	N/A	N/A
Our RPN+Fast R-CNN	95.14	85.20	72.12	N/A	N/A	N/A
Ours w/o Extra	95.51	86.29	73.68	94.26	84.69	71.80
Ours Full	**95.77**	**86.64**	**74.07**	**94.55**	**85.03**	**72.21**
Pedestrian						
Faster R-CNN	83.31	68.39	62.56	N/A	N/A	N/A
Our RPN+Fast R-CNN	85.96	68.55	62.55	N/A	N/A	N/A
Ours w/o Extra	84.86	68.87	63.09	**74.05**	**59.06**	**54.05**
Ours Full	**86.43**	**69.95**	**64.03**	73.91	58.91	53.79
Cyclist						
Faster R-CNN	56.36	46.36	42.77	N/A	N/A	N/A
Our RPN+Fast R-CNN	71.00	55.88	51.72	N/A	N/A	N/A
Ours w/o Extra	71.23	55.56	51.61	61.89	47.30	43.69
Ours Full	**74.92**	**59.13**	**55.03**	**65.79**	**50.46**	**46.57**

Table 8.5: Comparison of different detection networks on the KITTI validation set.

Detection Network Evalutaion. In Table 8.5, we first show that our RPN achieves significantly better performance than the RPN in *Ren et al.* (2015) when the two RPNs are used with Fast R-CNN *Girshick* (2015) on the KITTI validation set respectively. Then, we use region proposals from our RPN and compare different network architectures for detection. Our detection network is based on the architecture proposed in *Girshick* (2015). By adding the "subcategory FC layer" (Fig. 8.3), our detection network is also able to estimate the orientation of the object, while Fast R-CNN cannot. "Ours w/o Extra" refers to a network without feature extrapolating. By augmenting the network with the feature extrapolating layer, our full model ("Ours Full" in Table 8.5) further boosts the detection and orientation estimation performance, except for a minor drop on orientation estimation of pedestrian.

8.5.3 KITTI Test Set Evaluation

To compare with the state-of-the-art methods on the KITTI detection benchmark, we train our RPN and detection network with all the KITTI training data, and then test our method on the KITTI test set by submitting our results to *Geiger et al.*. Table 8.6 presents the detection and orientation estimation results on the three categories, where we compare our method (SubCNN) with different methods evaluated on KITTI. Our method ranks on top among all the published methods. The experimental results demonstrate the ability of our CNNs in using subcategory information for detection and orientation estimation. We note that the very recent work 3DOP *Chen et al.* (2015b) achieves competitive performance on KITTI. However, 3DOP uses stereo image pairs as input, while our method only needs a monocular image as input. Fig. 8.4 presents some examples of our detection results on KITTI. Since we employ 3DVPs as subcategories for car, after subcategory/3DVP classification, we are able to transfer the segmentation mask carried by the 3DVP to the detected object, which enables us to segment the detected cars and estimate their occluded regions similar to *Xiang et al.* (2015b).

8.5.4 Detection and Pose Estimation on PASCAL3D+

We also evaluate our detection framework on the 12 categories in PASCAL3D+. Table 8.7 presents the detection results in AP and the joint detection and pose estimation results in AVP. After generating region proposals from our RPN, we experiment with our detection networks with and without feature extrapolation. First, in terms of detection, our method improves over R-CNN *Girshick et al.* (2014) on all 12 categories. Second, in terms of join detection and pose estimation, our method significantly outperforms two state-of-the-art methods: VDPM *Xiang et al.* (2014a) and DPM-VOC+VP *Pepik et al.* (2015). Third, feature extrapolation helps both detection and pose estimation on PASCAL3D+. It is worth mentioning that PASCAL3D+ has much fewer training examples in each subcategory compared to KITTI (Table 8.1 vs. Table 8.2). Our pose estimation performance is limited by the number of training examples available in PASCAL3D+. We also note that the two recent

	Object Detection (AP)			Orientation (AOS)		
Methods	Easy	Moderate	Hard	Easy	Moderate	Hard
Car						
ACF *Dollár et al.* (2014)	55.89	54.74	42.98	N/A	N/A	N/A
DPM *Felzenszwalb et al.* (2010)	68.02	56.48	44.18	67.27	55.77	43.59
DPM-VOC+VP *Pepik et al.* (2015)	74.95	64.71	48.76	72.28	61.84	46.54
OC-DPM *Pepikj et al.* (2013)	74.94	65.95	53.86	73.50	64.42	52.40
SubCat *Ohn-Bar and Trivedi* (2015)	84.14	75.46	59.71	83.41	74.42	58.83
Regionlets *Wang et al.* (2013)	84.75	76.45	59.70	N/A	N/A	N/A
AOG *Li et al.* (2014)	84.80	75.94	60.70	33.79	30.77	24.75
3DVP *Xiang et al.* (2015b)	87.46	75.77	65.38	86.92	74.59	64.11
3DOP *Chen et al.* (2015b)	**93.04**	**88.64**	**79.10**	**91.44**	86.10	76.52
SubCNN (**Ours**)	90.74	88.55	77.95	90.49	**87.88**	**77.10**
Pedestrian						
ACF *Dollár et al.* (2014)	44.49	39.81	37.21	N/A	N/A	N/A
DPM *Felzenszwalb et al.* (2010)	47.74	39.36	35.95	43.58	35.49	32.42
DPM-VOC+VP *Pepik et al.* (2015)	59.48	44.86	40.37	53.55	39.83	35.73
FilteredICF *Zhang et al.* (2015)	67.65	56.75	51.12	N/A	N/A	N/A
DeepParts *Tian et al.* (2015)	70.49	58.67	52.78	N/A	N/A	N/A
Regionlets *Wang et al.* (2013)	73.14	61.15	55.21	N/A	N/A	N/A
3DOP *Chen et al.* (2015b)	**81.78**	**67.47**	**64.70**	**72.94**	**59.80**	**57.03**
SubCNN (**Ours**)	79.13	66.13	61.27	72.61	59.40	54.78
Cyclist						
DPM *Felzenszwalb et al.* (2010)	35.04	27.50	26.21	27.54	22.07	21.45
DPM-VOC+VP *Pepik et al.* (2015)	42.43	31.08	28.23	30.52	23.17	21.58
Regionlets *Wang et al.* (2013)	70.41	58.72	51.83	N/A	N/A	N/A
3DOP *Chen et al.* (2015b)	**78.39**	**68.94**	**61.37**	**70.13**	**58.68**	**52.35**
SubCNN (**Ours**)	74.40	61.98	54.75	63.74	52.06	45.93

Table 8.6: AP/AOS Comparison between different methods on the KITTI test set. More comparisons are available at *Geiger et al.*.

Methods	aeroplane	bicycle	boat	bottle	bus	car	chair	table	mbike	sofa	train	monitor	Mean
Object Detection (AP)													
DPM	42.2	49.6	6.0	20.0	54.1	38.3	15.0	9.0	33.1	18.9	36.4	33.2	29.6
R-CNN	72.4	68.7	34.0	–	73.0	62.3	33.0	35.2	70.7	49.6	70.1	57.2	56.9
Ours w/o Extra	76.3	73.4	**43.4**	44.7	**74.5**	63.3	35.4	32.4	74.9	51.9	74.1	60.9	58.8
Ours Full	**76.5**	**74.0**	42.4	**47.0**	**74.5**	**64.7**	**38.5**	**38.6**	**76.7**	**55.1**	**74.8**	**65.3**	**60.7**
Joint Object Detection and Pose Estimation (4 Views AVP)													
VDPM	34.6	41.7	1.5	–	26.1	20.2	6.8	3.1	30.4	5.1	10.7	34.7	19.5
DPM-VOC+VP	39.4	43.9	0.3	–	49.1	37.6	6.1	3.0	32.2	11.8	12.5	33.2	24.5
Ours w/o Extra	**62.3**	56.6	18.0	–	62.0	40.9	19.3	14.9	**62.3**	44.1	**58.1**	58.5	45.2
Ours Full	61.4	**60.4**	**21.1**	–	**63.0**	**48.7**	**23.8**	**17.4**	60.7	**47.8**	55.9	**62.3**	**47.5**
Joint Object Detection and Pose Estimation (8 Views AVP)													
VDPM	23.4	36.5	1.0	–	35.5	23.5	5.8	3.6	25.1	12.5	10.9	27.4	18.7
DPM-VOC+VP	29.7	**42.6**	0.4	–	39.5	36.8	9.4	2.6	32.9	11.0	10.3	**28.6**	22.2
Ours w/o Extra	45.9	25.5	11.1	–	37.7	34.6	15.2	7.4	**37.1**	**33.0**	42.5	24.3	28.6
Ours Full	**48.8**	36.3	**16.4**	–	**39.8**	**37.2**	**19.1**	**13.2**	37.0	32.1	**44.4**	26.9	**31.9**
Joint Object Detection and Pose Estimation (16 Views AVP)													
VDPM	15.4	18.4	0.5	–	46.9	18.1	6.0	2.2	16.1	10.0	22.1	16.3	15.6
DPM-VOC+VP	17.0	**24.7**	1.0	–	49.0	30.1	6.6	3.0	17.2	7.7	20.4	20.2	17.9
Ours w/o Extra	23.3	19.2	8.4	–	**52.6**	27.0	9.9	5.1	**23.6**	**20.9**	27.4	**27.9**	22.3
Ours Full	**28.0**	23.7	**10.7**	–	50.8	**31.4**	**14.3**	**9.4**	23.4	19.5	**30.7**	27.8	**24.5**
Joint Object Detection and Pose Estimation (24 Views AVP)													
VDPM	8.0	14.3	0.3	–	39.2	13.7	4.4	3.6	10.1	8.2	20.0	11.2	12.1
DPM-VOC+VP	10.6	**16.7**	2.2	–	**43.5**	**25.4**	4.4	2.3	11.3	4.9	22.4	14.4	14.4
Ours w/o Extra	18.9	10.5	6.7	–	34.3	23.3	8.3	6.5	**20.6**	17.5	**33.8**	17.0	17.9
Ours Full	**20.7**	16.4	**7.9**	–	34.6	24.6	**9.4**	**7.6**	19.9	**20.0**	32.7	**18.2**	**19.3**

Table 8.7: AP/AVP Comparison between different methods on the PASCAL3D+ dataset.

methods *Tulsiani and Malik* (2014); *Su et al.* (2015) achieve very appealing pose estimation results on PASCAL3D+. However, both of them utilize additional training images (ImageNet images in *Tulsiani and Malik* (2014) and synthetic images in *Su et al.* (2015)) and conduct detection and pose estimation with separate CNNs, where a CNN is specifically designed for pose estimation. Our method is capable of simultaneous object detection and viewpoint estimation even in the presence of limited training examples per viewpoint. Fig. 8.5 shows some detection results from our method. We again transfer segmentation masks of 3DVPs to the detected objects according to the subcategory classification results. Please see supplementary material for more examples.

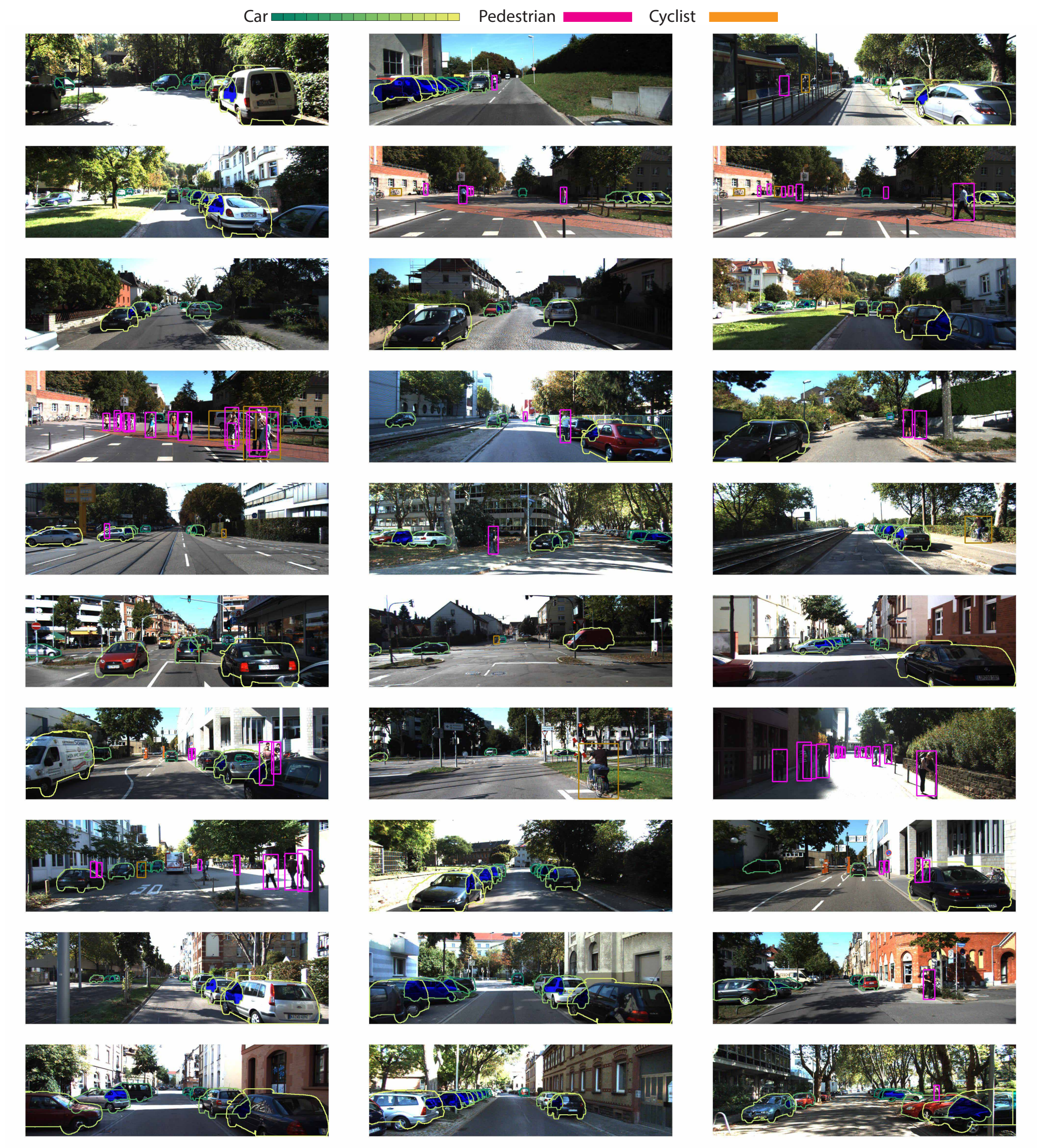


Figure 8.4: Examples of detections from our method on KITTI. By using 3DVP *Xiang et al.* (2015b) as subcategory, subcategory classification enables us to transfer a segmentation mask from a 3DVP to a detected object. Detections with score larger than 0.5 are shown.

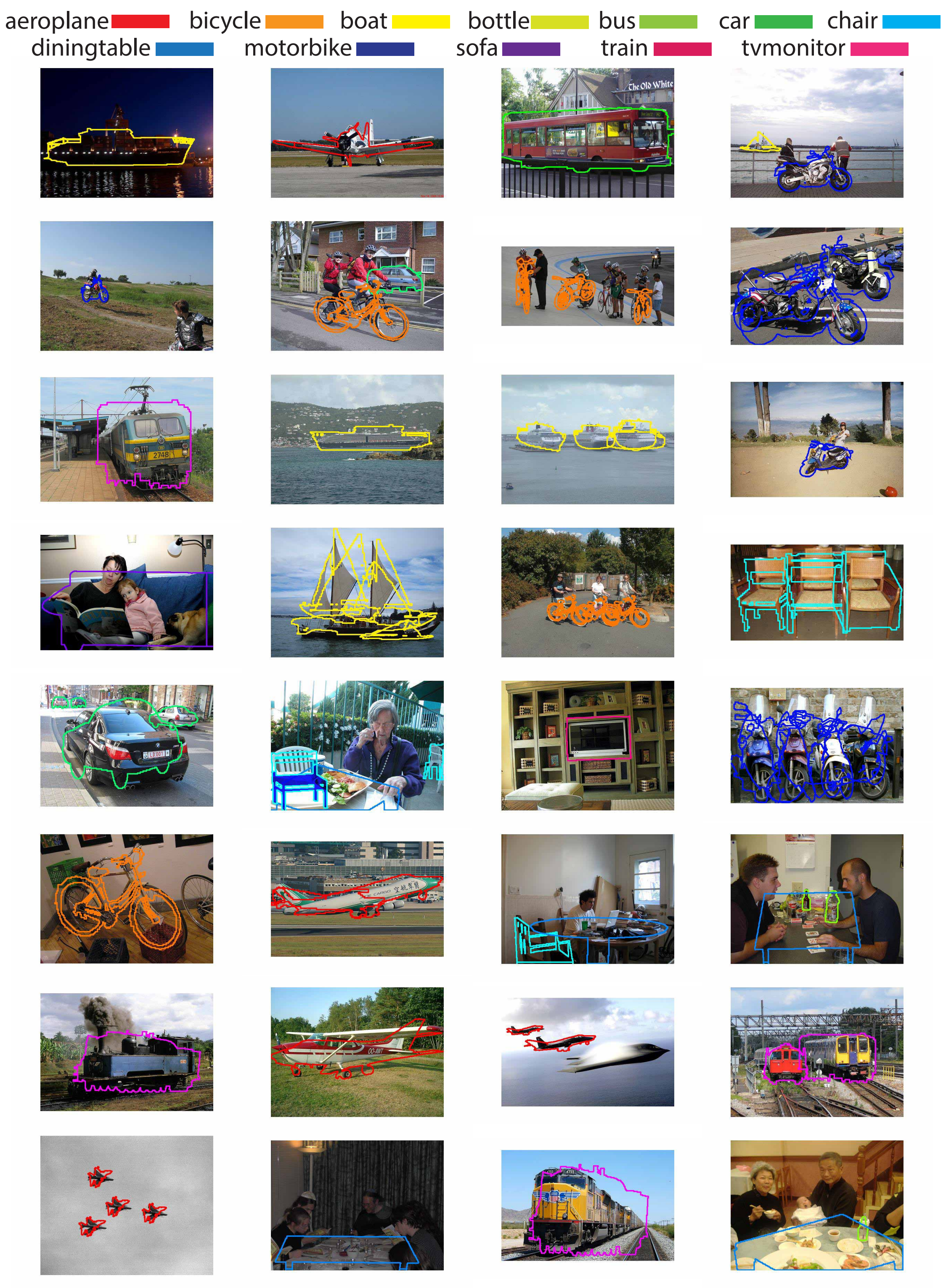


Figure 8.5: Examples of detections from our method on PASCAL3D+. By using 3DVP *Xiang et al.* (2015b) as subcategory, subcategory classification enables us to transfer a segmentation mask from a 3DVP to a detected object. Detections with score larger than 0.7 are shown.

8.6 Conclusion and Discussion

In this work, we explore how subcategory information can be exploited in CNN-based object detection. We have proposed a novel region proposal network, and a novel object detection network, where we explicitly employ subcategory information to improve region proposal, object detection and object pose estimation. Our subcategory-aware CNNs can also handle the scale variation of objects using image pyramids in an efficient way. We have conducted extensive experiments on the KITTI detection benchmark and the PASCAL3D+ dataset, and achieved the state-of-the-art results on both benchmarks.

Our subcategory-aware CNNs are able to utilize different types of subcategory representations. Specifically, in our implementation, we employ the 3D Voxel Patterns described in Chapter IV as subcategories for rigid object categories. Since 3DVP groups objects with similar visibility patterns, it enables us to learn good object subcategory detectors. In Chapter IV, we train a ACF detector *Dollár et al.* (2014) for each 3DVP. While in this work, we only need to train a CNN that is able to detect all 3DVPs jointly. Our subcategory-aware CNN is much more efficient than a set of ACF detectors, while achieving better detection performance.

CHAPTER IX

Application IV: Online Multi-Object Tracking by Decision Making

9.1 Introduction

Tracking multiple objects in videos is an important problem in computer vision which has wide applications in various video analysis scenarios, such as visual surveillance, sports analysis, robot navigation and autonomous driving. In cases where objects in a specific category are to be tracked, such as people or cars, a category detector can be utilized to facilitate tracking. Recent progress on Multi-Object Tracking (MOT) has focused on the tracking-by-detection strategy, where object detections from a category detector are linked to form trajectories of the targets. In order to resolve ambiguities in associating object detections and to overcome detection failures, most of these recent works *Berclaz et al.* (2011); *Butt and Collins* (2013); *Milan et al.* (2014); *Leal-Taixé et al.* (2014) process video sequences in a *batch mode* in which video frames from future time steps are also utilized to solve the data association problem. However, such non-causal systems are not suitable for online tracking applications like robot navigation and autonomous driving.

For tracking-by-detection in the *online mode*, the major challenge is how to associate noisy object detections in the current video frame with previously tracked

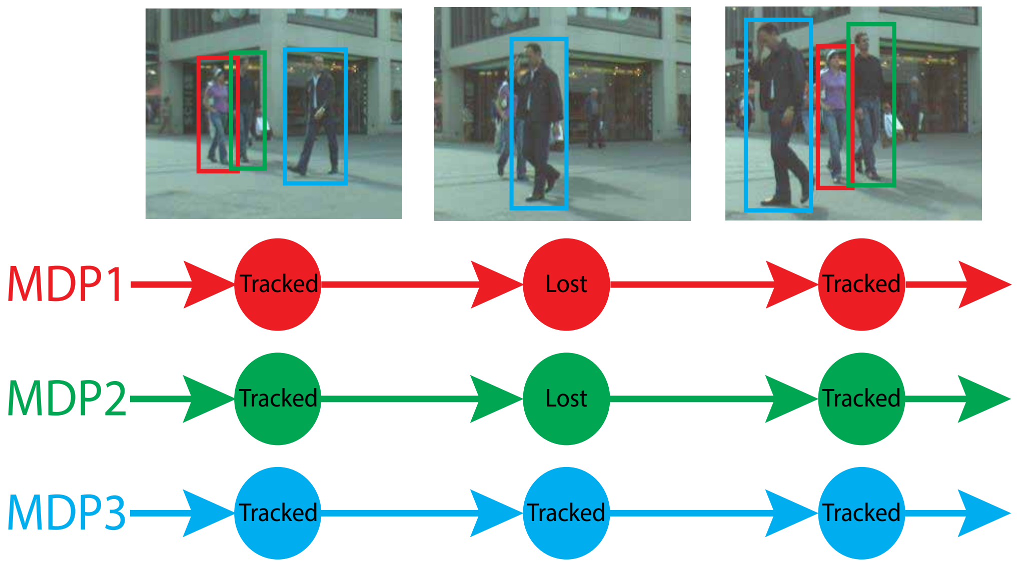


Figure 9.1: We formulate the online multi-object tracking problem as decision making in a Markov Decision Process (MDP) framework.

objects. The basis for any data association algorithm is a similarity function between object detections and targets. To handle ambiguities in association, it is useful to combine different cues in computing the similarity, such as appearance, motion, and location. Most previous works rely on heuristically selected parametric models for the similarity function and tune these parameters by cross-validation, which is not scalable to the number of features and does not necessarily guarantee generalization power of the model.

Recently, there is a trend on learning to track that advocates the concept of injecting learning capabilities to MOT *Song et al.* (2008); *Li et al.* (2009); *Kuo et al.* (2010); *Kim et al.* (2012); *Bae and Yoon* (2014). Based on their learning schemes, we can categorize these methods into *offline-learning* methods and *online-learning* methods. In offline-learning, learning is performed before the actual tracking takes place. For instance, *Li et al.* (2009); *Kim et al.* (2012) use supervision from ground truth trajectories offline to learn a similarity function between detections and tracklets for data association. As a result, offline-learning is static: it cannot take into account the dynamic status and the history of the target in data association, which is important to resolve ambiguities, especially when it needs to re-assign missed or occluded objects when they appear again. In contrast, online-learning conducts learning during tracking. A common strategy is to construct positive and negative training examples according to the tracking results, and then to train a similarity function for data association (e.g., *Song et al.* (2008); *Kuo et al.* (2010); *Bae and Yoon* (2014)). Online-learning is able to utilize features based on the status and the history of the target. However, there are no ground truth annotations available for supervision. So the method is likely to learn from incorrect training examples if there are errors in the tracking results, and these errors can be accumulated and result in *tracking drift*.

In this work, we formulate the online multi-object tracking problem (MOT in the online mode) as decision making in Markov Decision Processes (MDPs), where the lifetime of an object is modeled with a MDP, and multiple MDPs are assembled for multi-object tracking (Fig. 9.1). In our framework, learning a similarity function for data association is equivalent to learning a policy for the MDP. The policy learning is approached in a reinforcement learning fashion which benefits from advantages of both offline-learning and online-learning in data association. First, learning in our method is conducted offline so as to utilize supervision from ground truth trajectories. Second, learning in our method takes place while tracking objects in training sequences, so the MDP is able to make the decision based on both the current status and the history of the target. Specifically, given the ground truth trajectory of a target and an initial similarity function, the MDP attempts to track the target and collects feedback from the ground truth. According to the feedback, the MDP updates the similarity function to improve tracking. The similarity function is updated only when the MDP makes a mistake in data association, which enables us to collect hard training examples to learn the similarity function. Finally, training is finished when the MDP can successfully track the target.

In addition to the advantages of our learning strategy, our framework can naturally handle the birth/death and appearance/disappearance of targets by treating them as state transitions in the MDP. Our method also benefits from the strengths of online single object tracking methods *Babenko et al.* (2011); *Hare et al.* (2011); *Kalal et al.* (2012); *Bao et al.* (2012a), where we learn and update an appearance model for a target online in order to handle object detection failures. We conduct experiments on the recently introduced benchmark for multi-object tracking *Leal-Taixé et al.* (2015). Our extensive system analysis and comparison with the state-of-the-art tracking methods on the MOT benchmark demonstrate the superiority of our method.

9.2 Related Work

Multi-Object Tracking. Recent research in MOT has focused on the tracking-by-detection principal, where the main challenge is the data association problem in linking object detections. Majority of the batch methods (*Zhang et al.* (2008); *Li et al.* (2009); *Niebles et al.* (2010); *Berclaz et al.* (2011); *Pirsiavash et al.* (2011); *Butt and Collins* (2013); *Milan et al.* (2014)) formulates MOT as a global optimization problem in a graph-based representation, while online methods solve the data association problem either probabilistically *Okuma et al.* (2004); *Khan et al.* (2005); *Oh et al.* (2009) or determinatively (e.g., Hungarian algorithm *Munkres* (1957) in *Kim et al.* (2012); *Bae and Yoon* (2014) or greedy association *Breitenstein et al.* (2011)). A core component in any data association algorithm is a similarity function between objects. Both batch methods *Li et al.* (2009); *Kuo et al.* (2010) and online methods *Song et al.* (2008); *Kim et al.* (2012); *Bae and Yoon* (2014) have explored the idea of learning to track, where the goal is to learn a similarity function for data association from training data. Our main contribution in this work is a novel reinforcement learning algorithm for data association in online MOT.

Online Single Object Tracking. In single object tracking, the state-of-the-art trackers *Babenko et al.* (2011); *Hare et al.* (2011); *Kalal et al.* (2012); *Bao et al.* (2012a); *Yao et al.* (2013); *Supancic III and Ramanan* (2013); *Oron et al.* (2014); *Xiang et al.* (2014b) focus on how to learn a strong appearance model of the target online and use it for tracking. It is non-trivial to apply these trackers to MOT since they are not able to handle the entering/exiting of objects from the scene. The initial location of the target needs to be specified before the tracking starts, and they assume that the target exists in the whole video sequence. Additionally, online single object trackers are likely to drift if the appearance of the target changes significantly. Another contribution of our work is that by modeling the lifetime of an object with a MDP, we are able to take the advantages of existing online single object trackers to facilitate MOT, while overcoming their limitations by using object detection as additional cues.

MDP in Vision. Markov decision processes *Bellman* (1957) have been applied to different computer vision tasks, such as feature selection for recognition *Paletta et al.* (2005); *Karayev et al.* (2014), human activity forecasting *Kitani et al.* (2012), video game playing *Mnih et al.* (2013) and human-machine collaboration *Russakovsky et al.* (2015a). MDP is suitable for dynamic environments where an agent needs to perform certain tasks by making decisions and executing actions sequentially. In our framework, we consider a single object tracker to be an agent in MDP, whose task is to track the target. Then we learn a good policy for the MDP with reinforcement learning, and employ multiple MDPs to track multiple targets.

9.3 Online Multi-Object Tracking Framework

In Sec. 9.3.1 and Sec. 9.3.2, we introduce our Markov decision process formulation in modeling the lifetime of a single target in object tracking, then we present our method using multiple MDPs for online multi-object tracking in Sec. 9.3.3.

9.3.1 Markov Decision Process

In our framework, the lifetime of a target is modeled with a Markov Decision Process (MDP). The MDP consists of the tuple $(\mathcal{S}, \mathcal{A}, T(\cdot), R(\cdot))$:

- The target state $s \in \mathcal{S}$ encodes the status of the target.
- The action $a \in \mathcal{A}$ which can be performed to a target.
- The state transition function $T : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ describes the effect of each action in each state.
- The real-valued reward function $R : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ defines the immediate reward received after executing action a to state s.

States. We partition the state space in the target MDP into four subspaces, i.e., $\mathcal{S} = \mathcal{S}_{\text{Active}} \cup \mathcal{S}_{\text{Tracked}} \cup \mathcal{S}_{\text{Lost}} \cup \mathcal{S}_{\text{Inactive}}$, where each subspace contains infinity number of states which encode the information of the target depending on the feature representation, such as appearance, location, size and history of the target. Fig. 9.2 illustrates the transitions between the four subspaces. "Active" is the initial state for any target. Whenever an object is detected by the object detector, it enters an "Active" state. An active target can transition to "Tracked" or "Inactive". Ideally, a true positive from object detector should transition to a "Tracked" state, while a false alarm should enter an "Inactive" state. A tracked target can keep tracked, or transition to "Lost" if the target is lost due to some reason, such as occlusion, or disappearance from the field of view of the camera. Likewise, a lost target can stay as lost, or go back to "Tracked" if it appears again, or transition to "Inactive" if it has been lost for a sufficiently long time. Finally, "Inactive" is the terminal state for any target, i.e., an inactive target stays as inactive forever.

Actions and Transition Function. Seven possible transitions are designed between the states of a target, which correspond to seven actions in our target MDP.

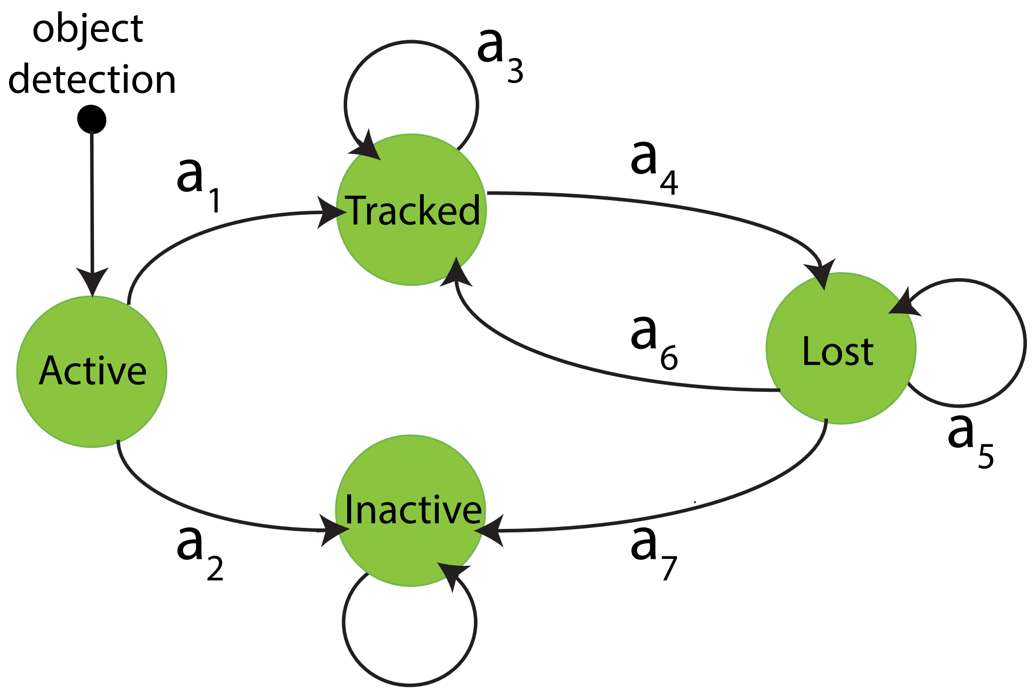


Figure 9.2: The target MDP in our framework.

Fig. 9.2 illustrate these transitions and actions. In the MDP, all the actions are deterministic, i.e., given the current state and an action, we specify a new state for the target. For example, executing action a_4 on a tracked target would transfer the target into a lost state, i.e., $T(s_{\text{Tracked}}, a_4) = s_{\text{Lost}}$.

Reward Function. In our MDP, the reward function is not given but needs to be learned from training data, i.e., an inverse reinforcement learning problem *Ng and Russell* (2000), where we use ground truth trajectories of the targets as supervision.

9.3.2 Policy

In MDP, a policy π is a mapping from the state space $\mathcal{S}$ to the action space $\mathcal{A}$, i.e., $\pi : \mathcal{S} \mapsto \mathcal{A}$. Given the current state of the target, a policy determines which action to take. Equivalently, the decision making in MDP is performed by following a policy. The goal of policy learning is to find a policy which maximizes the total rewards obtained. In this section, we first describe our policies designed for the Active

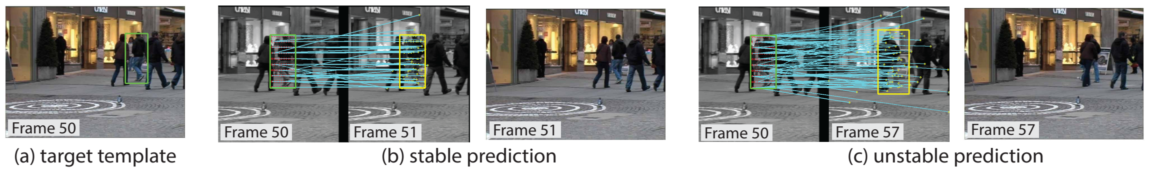


Figure 9.3: The appearance of the target is represented by a template in a video frame (a). We compute optical flow from densely sampled points inside the target template to a new frame. The quality of the flow is used as a cue to make the decision: (b) an example of stable prediction; (c) an example of unstable prediction due to partial occlusion, where we show both the cropped frames and the original frames. The yellow box is the predicted location of the target.

subspace and the Tracked subspace, then we present a novel reinforcement learning algorithm to learn a good policy for data association in the Lost subspace.

9.3.2.1 Policy in an Active State

In an Active state s, the MDP makes the decision between transferring an object detection into a tracked or inactive target to deal with noisy detections. This decision making can be considered to be a preprocessing step before tracking. Strategies such as non-maximum suppression or thresholding detection scores are usually used. In our implementation, we train a binary Support Vector Machine (SVM) *Boser et al.* (1992) offline to classify a detection into tracked or inactive using a normalized 5D feature vector $\phi_{\text{Active}}(s)$, i.e., 2D coordinates, width, height and score of the detection, where training examples are collected from training video sequences. This is equivalent to learning the reward function in Active:

$$R_{\text{Active}}(s,a) = y(a)\big(\mathbf{w}_{\text{Active}}^T \phi_{\text{Active}}(s) + b_{\text{Active}}\big), \tag{9.1}$$

where $(\mathbf{w}_{\text{Active}}, b_{\text{Active}})$ defines the hyperplane in SVM, $y(a) = +1$ if action $a = a_1$, and $y(a) = -1$ if $a = a_2$ in Fig. 9.2. Note that a false alarm from object detector can still be miss-classified and transfered to a tracked state, which will be handled by the

MDP in the tracked and lost states.

9.3.2.2 Policy in a Tracked State

In a Tracked state, the MDP needs to decide whether to keep tracking the target or to transfer it into a lost state. As long as the target is not occluded and is in the camera's field of view, we should keep tracking it. Otherwise, it should be marked as lost. This decision making is related to the goal of single object tracking in the literature *Babenko et al.* (2011); *Hare et al.* (2011); *Kalal et al.* (2012); *Bao et al.* (2012a). Inspired by these works, we build an appearance model for the target online and use it to track the target. If the appearance model is able to successfully track the target in the next video frame, the MDP leaves the target in a tracked state. Otherwise, the target is transferred to a lost state. Our framework is general to utilize different approaches in building the appearance model. We describe our implementation based on the TLD tracker *Kalal et al.* (2012) in this work.

Template Representation. The appearance of the target is simply represented by a template that is an image patch of the target in a video frame. Whenever an object detection is transferred to a tracked target, we initialize the target template with the detection bounding box. Fig. 9.3(a) illustrates a template for a pedestrian. When the target is being tracked, the MDP collects its templates in the tracked frames to represent the history of the target, which will be used in the lost state for decision making.

Template Tracking. In order to use the target template for tracking, we compute an optical flow from densely and uniformly sampled points inside the template to a new video frame. Specifically, given a point $\mathbf{u} = (u_x, u_y)$ on the target template I, we find its corresponding location $\mathbf{v} = \mathbf{u} + \mathbf{d} = (u_x + d_x, u_y + d_y)$ in the new frame J using the iterative Lucas-Kanade method with pyramids *Bouguet* (2001), where $\mathbf{d} = (d_x, d_y)$ is the optical flow at $\mathbf{u}$. After computing the optical flow of all the sampled points, we use the Forward-Backward (FB) error defined in *Kalal et al.* (2012) to measure how stable the predict is. Given the prediction $\mathbf{v}$ of point $\mathbf{u}$ on the target template, we can compute the backward flow of point $\mathbf{v}$ to the target template and obtain a new prediction $\mathbf{u}'$. If the optical flow is stable, $\mathbf{u}$ and $\mathbf{u}'$ should be close to each other. So FB error of a point is defined as the Euclidean distance between the original point and the forward-backward prediction: $e(\mathbf{u}) = \|\mathbf{u} - \mathbf{u}'\|^2$, and the stability of the tracking is measured using the median of the FB errors of all sampled points: $e_{\text{medFB}} = \text{median}(\{e(\mathbf{u}_i)\}_{i=1}^{n})$, where n is the number of points. If e_{medFB} is larger than some threshold, the tracking is considered to be unstable. Moreover, after filtering out unstable matches whose FB error is larger than the threshold, we can predict a bounding box for the target using the remaining matches, which is treated as the new location of the target. Fig. 9.3 (b) and (c) illustrate the optical flow in a stable case and an unstable case respectively. As we can see, the quality of the optical flow is an important cue to decide whether to keep tracking the target or not.

However, it is risky to make the decision based on optical flow only. Because the tracked target can be a false alarm from the object detector (see Sec. 9.3.2.1), whose appearance may not change, such as a detection on the background of the scene. In this case, the optical flow tracker will keep tracking the false alarm. To handle this case, we resort to the object detector. The intuition is that a false alarm cannot be consistently detected. If a tracked target does not encounter object detections for a while, it is likely to be a false alarm. So we examine the history of the target, and compute the bounding box overlap $o(t_k, \mathcal{D}_k)$ between the target t_k in k frames before and the corresponding detections $\mathcal{D}_k$. Then we compute the mean bounding box overlap for the past K tracked frames $o_{\text{mean}} = \text{mean}\big(\{o(t_k, \mathcal{D}_k)\}_{k=1}^{K}\big)$ as another metric to make the decision. Finally, we define the reward function in a tracked state

s with feature representation $\phi_{\text{Tracked}}(s) = (e_{\text{medFB}}, o_{\text{mean}})$ as

$$R_{\text{Tracked}}(s,a) = \begin{cases} y(a), \text{ if } e_{\text{medFB}} < e_0 \text{ and } o_{\text{mean}} > o_0 \\ -y(a), \text{ otherwise,} \end{cases} \tag{9.2}$$

where e_0 and o_0 are specified thresholds, $y(a) = +1$ if action $a = a_3$, and $y(a) = -1$ if $a = a_4$ in Fig. 9.2. So the MDP keeps the target in a tracked state if e_{medFB} is smaller but o_{mean} is larger than certain thresholds respectively. Otherwise, the target is transfered to a lost state.

Template Updating. The appearance model of the target needs to be updated in order to accommodate the appearance change. Online tracking methods *Babenko et al.* (2011); *Hare et al.* (2011); *Kalal et al.* (2012); *Bao et al.* (2012a) update the appearance model whenever the tracker tracks the target. As a result, they are likely to accumulate tracking errors during the update, and drift from the target. In our MDP, we adopt a "lazy" updating rule and resort to the object detector in preventing tracking drift. Specifically, the template used in tracking remains unchanged if it is able to track the target. Whenever the template fails to track the target due to appearance change, the MDP transfers the target into a lost state. The "tracking" template is replaced by the associated detection when the target transitions from lost to tracked (Sec. 9.3.2.3). Meanwhile, we store K templates as the history of the target being tracked. The "tracking" template is one of the K templates, but may not be the latest one due to our "lazy" updating rule. These K templates are used for data association in lost states. So we do not accumulate tracking errors, but reply on the data association to handle the appearance change and continue the tracking.

9.3.2.3 Policy in a Lost State

In a Lost state, the MDP needs to decide whether to keep the target as lost, transition it to a tracked state, or mark it as inactive. We simply mark a lost target as inactive and terminate the tracking if the target has been lost for more than T_{Lost} frames. The challenging case is to make the decision between tracking the target and keeping it as lost. We treat it as a data association problem: in order to transfer a lost target into a tracked state, the target needs to be associated with one of the detections from the object detector, otherwise, the target is kept as lost.

Data Association. Let t denote a lost target, and d be an object detection. Our goal is to predict the label $y \in \{+1, -1\}$ of the pair (t, d) indicating that the target is linked ($y = +1$) or not linked ($y = -1$) to the detection. We perform the binary classification using a real-valued linear function $f(t, d) = \mathbf{w}^T \phi(t, d) + b$, where $(\mathbf{w}, b)$ are the parameters that control the function, and $\phi(t, d)$ is the feature vector which captures the similarity between the target and the detection. The decision rule is given by $y = +1$ if $f(t, d) \geq 0$, otherwise $y = -1$. Consequently, the reward function for data association in a lost state s with feature representation $\phi_{\text{Lost}}(s) = \{\phi(t, d_k)\}_{k=1}^M$ is defined as

$$R_{\text{Lost}}(s, a) = y(a)\Big(\max_{k=1}^{M} \big(\mathbf{w}^T \phi(t, d_k) + b\big)\Big), \tag{9.3}$$

where $y(a) = +1$ if action $a = a_6$, $y(a) = -1$ if $a = a_5$ in Fig. 9.2, and k indexes M potential detections for association. The task of policy learning in the lost state reduces to learning the parameters $(\mathbf{w}, b)$ in the decision function.

Reinforcement Learning. We train the binary classifier with reinforcement learning in our MDP. Let $\mathcal{V} = \{v_i\}_{i=1}^N$ denote a set of video sequences for training, where N is the number of sequences. Suppose there are N_i ground truth targets $\mathcal{T}_i = \{t_{ij}\}_{j=1}^{N_i}$ in video v_i. Our goal is training the MDP to successfully track all these targets. We start training with an initial weights $(\mathbf{w}_0, b_0)$ and an empty training set $\mathcal{S}_0 = \emptyset$ for the binary classifier. Note that whenever the weights of the binary classifier are specified, we have a complete policy for the MDP which takes the action maximizing the reward in a given state. So the training algorithm loops over all the videos and all the targets, follows the current policy of the MDP to track the targets.

The binary classifier or the policy is updated only when the MDP makes a mistake in data association. In this case, the MDP takes a different action as indicated by the ground truth trajectory. Suppose the MDP is tracking the jth target t_{ij} in video v_i, and on the lth frame of the video, the MDP is in a lost state. Let's consider two types of mistakes that can happen. i) The MDP associates the target t_{ij}^l to an object detection d_k which is wrong according to the ground truth, i.e., the target is incorrectly associated to a detection. Then $\phi(t_{ij}^l, d_k)$ is added to the training set $\mathcal{S}$ of the binary classifier as a negative example. ii) The MDP decides to not associate the target to any detection, but the target is visible and correctly detected by a detection d_k according to the ground truth, i.e., the MDP missed the correct association. Then $\phi(t_{ij}^l, d_k)$ is added to the training set as a positive example. After the training set has been augmented, we update the binary classifier by re-training it on the new training set. Specifically, given the current training set $\mathcal{S} = \{(\phi(t_k, d_k), y_k)\}_{k=1}^M$, we solve the following soft-margin optimization problem to obtain a max-margin classifier for data association:

$$\begin{aligned} &\min_{\mathbf{w},b,\boldsymbol{\xi}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{k=1}^{M}\xi_k \\ &\text{s.t.} \quad y_k\big(\mathbf{w}^T\phi(t_k, d_k) + b\big) \geq 1 - \xi_k, \xi_k \geq 0, \forall k, \end{aligned} \tag{9.4}$$

where $\xi_k, k = 1, \ldots, M$ are the slack variables, and C is a regularization parameter. Once the classifier has been updated, we obtain a new policy which is used in the next iteration of the training process. We keep iterating and updating the policy until all the targets are successfully tracked. Algorithm 2 summarizes the policy learning algorithm.

Feature Representation. One advantage of our reinforcement learning algorithm is that it is general and enables us to design and utilize features which are based on the status and the history of the target. We describe our design of the

```
input  : Video sequences $\mathcal{V} = \{v_i\}_{i=1}^{N}$, ground truth trajectories $\mathcal{T}_i = \{t_{ij}\}_{j=1}^{N_i}$
         and object detection $\mathcal{D}_i = \{d_{ij}\}_{j=1}^{N_i'}$ for video $v_i, i = 1, \dots, N$
output: Binary classifier $(\mathbf{w}, b)$ for data association
1 Initialization: $\mathbf{w} \leftarrow \mathbf{w}_0$, $b \leftarrow b_0$, $\mathcal{S} \leftarrow \emptyset$
2 repeat
3     foreach video $v_i$ in $\mathcal{V}$ do
4         foreach target $t_{ij}$ in $v_i$ do
5             Initialize the MDP in Active ;
6             $l \leftarrow$ index of the 1st frame $t_{ij}$ correctly detected ;
7             Transfer the MDP to Tracked, and initial the target template ;
8             while $l \leq$ index of last frame of $t_{ij}$ do
9                 Follow the current policy and choose an action $a$ ;
10                Compute the action $a_{\text{gt}}$ indicated by the ground truth ;
11                if Current state is Lost and $a \neq a_{gt}$ then
12                    Decide the label $y_k$ of the pair $(t_{ij}^l, d_k)$ ;
13                    $\mathcal{S} \leftarrow \mathcal{S} \cup \{(\phi(t_{ij}^l, d_k), y_k)\}$ ;
14                    $(\mathbf{w}, b) \leftarrow$ solution of Eq. (9.4) on $\mathcal{S}$ ;
15                    break ;
16                else
17                    Execute action $a$ ;
18                    $l \leftarrow l + 1$ ;
19                end
20            end
21            if $l >$ index of last frame of $t_{ij}$ then
22                Mark target $t_{ij}$ as successfully tracked;
23            end
24        end
25    end
26 until all targets are successfully tracked;
```

Algorithm 2: Reinforcement learning of the binary classifier for data association

Type	Notation	Feature Description
FB error	$\phi_1, \cdots, \phi_5$	Mean of the median forward-backward errors from the entire, left half, right half, upper half and lower half of the templates in optical flow
NCC	ϕ_6	Mean of the median Normalized Correlation Coefficients (NCC) between image patches around the matched points in optical flow
	ϕ_7	Mean of the NCC between image patches of the detection and the predicted bounding boxes from optical flow
Height ratio	ϕ_8	Mean of the ratios in bounding box height between the detection and the predicted bounding boxes from optical flow
	ϕ_9	Ratio in bounding box height between the target and the detection
Overlap	ϕ_{10}	Mean of the bounding box overlaps between the detection and the predicted bounding boxes from optical flow
Score	ϕ_{11}	Normalized detection score
Distance	ϕ_{12}	Euclidean distance between the centers of the target and the detection after motion prediction of the target with a linear velocity model

Table 9.1: Our feature representation for data association.

feature vector $\phi(t, d)$ which encodes the similarity between a target t and a detection d. First of all, the history of the target is represented by K templates in the past K video frames when the target is being tracked before it transfers to the lost state. Second, given the object detection d, we compute optical flow from each template to the detection in the same way as described in Sec. 9.3.2.2 but constrain the destination of the optical flow inside a neighborhood around the bounding box of the detection. Then we measure the quality of the optical flow in different aspects and use these metrics as features. Finally, we add features based on the similarity between the bounding boxes of the target and the detection. Table 9.1 summaries our feature representation.

9.3.3 Multi-Object Tracking with MDPs

After learning the policy/reward of the MDP, we apply it to the multi-object tracking problem. We dedicate a MDP for each object, and the MDP follows the learned policy to track the object. Given a new input video frame, targets in tracked states are processed first to determine whether they should stay as tracked or transfer to lost states. Then we compute pairwise similarity between lost targets and object detections which are not covered by the tracked targets, where non-maximum suppression based on bounding box overlap is employed to suppress covered detections, and the similarity score is computed by the binary classifier for data association. After that, the similarity scores are used in the Hungarian algorithm *Munkres* (1957) to obtain the assignment between detections and lost targets. According to the assignment, lost targets which are linked to some object detections are transferred to tracked states. Otherwise, they stay as lost. Finally, we initialize a MDP for each object detection which is not covered by any tracked target. Algorithm 3 describes our multi-object tracking algorithm using MDPs in detail. Note that, tracked targets have higher priority than lost targets in tracking, and detections covered by tracked targets are suppressed to reduce ambiguities in data association.

```
input  : A video sequence $v$ and object detection $\mathcal{D} = \{d_k\}_{k=1}^{N}$ for $v$, binary
         classifier $(\mathbf{w}, b)$ for data association
output: Trajectories of targets $\mathcal{T} = \{t_i\}_{i=1}^{M}$ in the video
1  Initialization: $\mathcal{T} \leftarrow \emptyset$ ;
2  foreach video frame $l$ in $v$ do
       // process targets in tracked states
3      foreach tracked target $t_i$ in $\mathcal{T}$ do
4          Follow the policy, move the MDP of $t_i$ to the next state ;
5      end
       // process targets in lost states
6      foreach lost target $t_i$ in $\mathcal{T}$ do
7          foreach detection $d_k$ not covered by any tracked target do
8              Compute $f(t_i, d_k) = \mathbf{w}^T \phi(t_i, d_k) + b$ ;
9          end
10     end
11     Data association with Hungarian algorithm for the lost targets ;
12     foreach lost target $t_i$ in $\mathcal{T}$ do
13         Follow the assignment, move the MDP of $t_i$ to the next state ;
14     end
       // initialize new targets
15     foreach detection $d_k$ not covered by any tracked target in $\mathcal{T}$ do
16         Initialize a MDP for a new target $t$ with detection $d_k$ ;
17         if action $a_1$ is taken following the policy then
18             Transfer $t$ to the tracked state ;
19             $\mathcal{T} \leftarrow \mathcal{T} \cup \{t\}$ ;
20         else
21             Transfer $t$ to the inactive state ;
22         end
23     end
24 end
```

Algorithm 3: Multi-Object Tracking with MDPs

9.4 Experiments

Datasets. We test our tracking framework on the recently introduced Multiple Object Tracking Benchmark *Leal-Taixé et al.* (2015) for people tracking. The MOT Benchmark collects widely used video sequences in the MOT community and some new challenging sequences. These sequences are divided into a training set and a test set each with 11 sequences. Since the annotations of the test set are not released, we separate a validation set of 6 sequences from the 11 training sequences to conduct analysis about our framework. The training and testing splitting for validation and testing is shown in Table 9.2. Except for AVG-TownCentre in the test set, for each of the other test sequences, there are training sequences which are captured in similar scenario indicated by the naming of the sequences. This property enables us to learn meaningful characteristics from training sequences and use them for testing . The MOT benchmark also provides object detections from the ACF detector *Dollár et al.* (2014). By using the same object detection, we can make a fair comparison between different tracking methods.

Evaluation Metrics. We use multiple metrics to evaluate the multiple object tracking performance as suggested by the MOT Benchmark. These include Multiple Object Tracking Accuracy (MOTA) *Keni and Rainer* (2008), Multiple Object Tracking Precision (MOTP) *Keni and Rainer* (2008), Mostly Track targets (MT, percentage of ground truth objects who trajectories are covered by the tracking output for at least 80%), Mostly Lost targets (ML, percentage of ground truth objects who trajectories are covered by the tracking output less than 20%), the total number of False Positives (FP), the total number of False Negatives (FN), the total number of ID Switches (IDS), the total number of times a trajectory is Fragmented (Frag), and the number of frames processed in one second (Hz).

9.4.1 Analysis on Validation Set

Impact of the History. We first investigate the effect of the number of templates used in a lost state for data association (Sec. 9.3.2.3). Intuitively, the more templates we use, the longer history of the target is captured. Table 9.3 shows the tracking performance in terms of the number of templates on the validation set, where we

Training	**Testing**
Validation on MOT Benchmark	
TUD-Stadtmitte	TUD-Campus
ETH-Bahnhof	ETH-Sunnyday, ETH-Pedcross2
ADL-Rundle-6	ADL-Rundle-8, Venice-2
KITTI-13	KITTI-17
Testing on MOT Benchmark	
TUD-Stadtmitte, TUD-Campus	TUD-Crossing
PETS09-S2L1	PETS09-S2L2, AVG-TownCentre
ETH-Bahnhof, ETH-Sunnyday, ETH-Pedcross2	ETH-Jelmoli, ETH-Linthescher, ETH-Crossing
ADL-Rundle-6, ADL-Rundle-8	ADL-Rundle-1, ADL-Rundle-3
KITTI-13, KITTI-17	KITTI-16, KITTI-19
Venice-2	Venice-1

Table 9.2: Training and Testing sequences for validation and testing on the MOT Benchmark.

K	MOTA	MOTP	MT	ML	FP	FN	IDS	Frag
1	24.7	73.2	10.3	55.1	3,597	13,651	147	303
2	25.7	73.5	9.8	53.4	3,548	13,485	121	349
3	23.0	73.6	8.5	56.0	3,727	13,907	134	325
4	26.3	**73.9**	9.8	53.8	3,191	13,726	**91**	300
5	**26.7**	73.7	**12.0**	53.0	3,386	**13,415**	111	331
6	19.5	73.7	5.6	68.8	3,393	14,920	269	321
7	26.1	73.6	10.7	55.6	3,092	13,838	132	306
8	25.8	73.8	10.7	55.6	3,221	13,785	122	305
9	**26.7**	73.6	**12.0**	**51.7**	3,290	13,491	133	328
10	26.6	73.8	9.8	55.1	**2,691**	14,130	123	**276**
11	25.3	73.5	**12.0**	52.1	3,672	13,436	136	317
12	24.8	73.4	11.5	55.6	3,637	13,585	139	321

Table 9.3: Tracking performance in terms of the number of templates on the validation set.

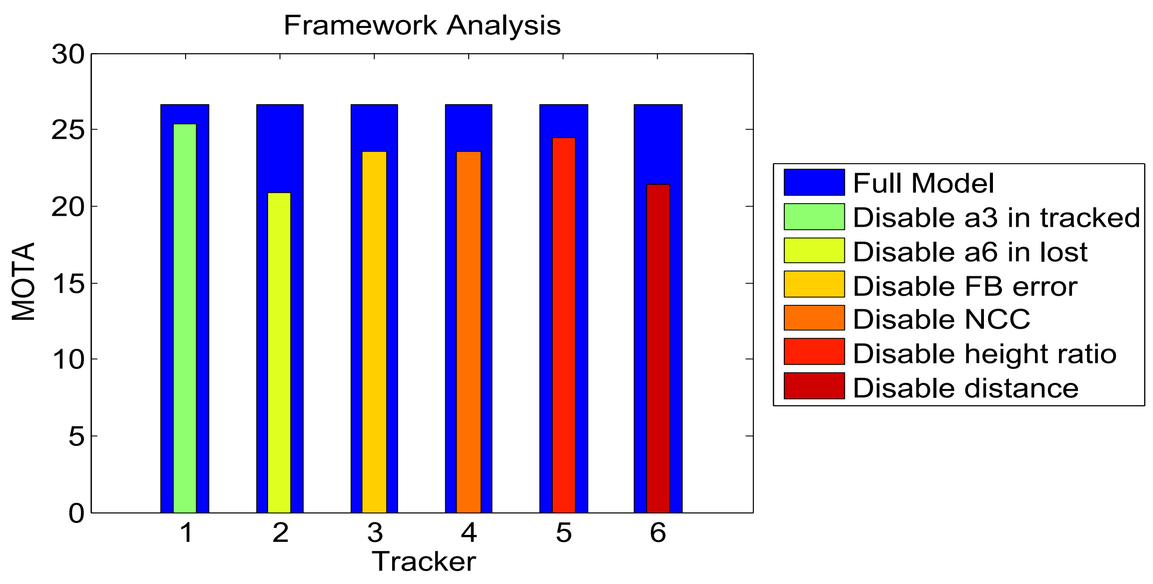


Figure 9.4: Analysis of our framework on the validation set by disabling different components.

accumulate the statistics across all the 6 testing sequences for evaluation. From the table, we observe two peaks for the tracking performance. One is around using 5 templates, and the other is around using 9 templates, which demonstrates that using multiple templates to capture the history of the object is helpful. With 9 templates, we see significant improvements in terms of mostly tracked (MT) and mostly lost (ML). This indicates that the tracker is able to generate long tracks to cover the target, which in turn reflects that the data association is more effective.

Contribution of Different Components. We investigate the contribution of different components in our framework by disabling a component at one time and then examining the performance drop in terms of MOTA on the validation set (Fig. 9.4). 1) We disable action a_3 in tracked states (Fig. 9.2). Then the template tracking is disabled and a tracked target directly transfers to a lost state. We do not see significant performance drop in this case, since the framework can still rely on data association in lost states to continue tracking. Template tracking is helpful when the detector misses the target. 2) We disable action a_6 in lost states (Fig. 9.2), i.e.,

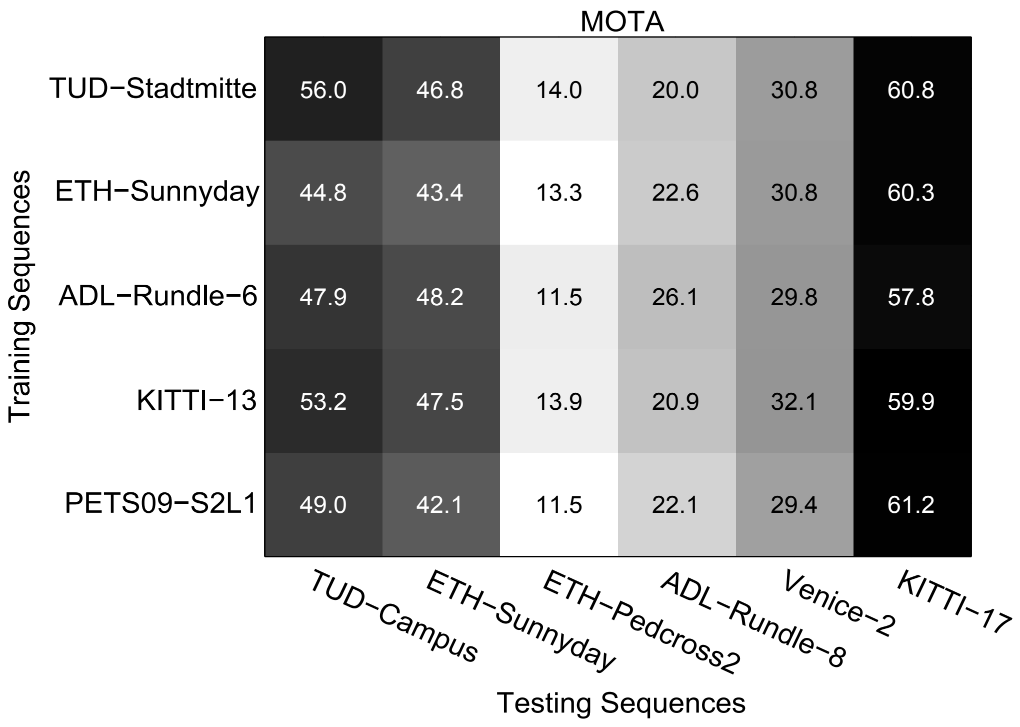


Figure 9.5: Tracking performance in MOTA with different pairs of training and testing sequences.

data association for lost targets is disabled. In this case, we see a significant loss in performance in Fig. 9.4. Especially, ID switches are more than 3 times compared to the full model. Data association is a crucial component in our framework. 3-6) Finally, we investigate the contribution of different features used in data association (Table 9.1). Fig. 9.4 shows the performance drop by disabling FB error in optical flow ($\phi_1, \cdots, \phi_5$), Normalized Correlation Coefficient (NCC, ϕ_6 and ϕ_7), ratio between the heights of bounding box (ϕ_8 and ϕ_9), and distance between the target and the detection (ϕ_{12}) respectively. As we can see, the four types of features all contribute, and distance is relatively more important than other features. In addition, we do not see performance drop by disabling bounding box overlap (ϕ_{10}) and detection score (ϕ_{11}) on the validation set.

Cross-domain Tracking. In order to test the generalization power of our method, we also conduct experiments by testing the trained tracker in different sce-

Tracker	Tracking	Learning	MOTA	MOTP	MT	ML	FP	FN	IDS	Frag	Hz
DP_NMS	Batch	N/A	14.5	70.8	6.0%	40.8%	13,171	34,814	4,537	3,090	**444.8**
TC_ODAL	Online	Online	15.1	70.5	3.2%	55.8%	12,970	38,538	**637**	1,716	1.7
TBD	Batch	Offline	15.9	70.9	6.4%	47.9%	14,943	34,777	1,939	1,963	0.7
SMOT	Batch	N/A	18.2	71.2	2.8%	54.8%	8,780	40,310	1,148	2,132	2.7
RMOT	Online	N/A	18.6	69.6	5.3%	53.3%	12,473	36,835	684	1,282	7.9
CEM	Batch	N/A	19.3	70.7	8.5%	46.5%	14,180	34,591	813	1,023	1.1
SegTrack	Batch	Offline	22.5	**71.7**	5.8%	63.9%	**7,890**	39,020	697	**737**	0.2
MotiCon	Batch	Offline	23.1	70.9	4.7%	52.0%	10,404	35,844	1,018	1,061	1.4
MDP OFL	Online	Offline	30.1	71.6	10.4%	41.3%	8,789	33,479	690	1,301	0.8
MDP REL	Online	Online	**30.3**	71.3	**13.0%**	**38.4%**	9,717	**32,422**	680	1,500	1.1

Table 9.4: Tracking performance on the test set of the MOT Benchmark. More comparisons are available at *Leal-Taixé et al.*. DP_NMS *Pirsiavash et al.* (2011), TC_ODAL *Bae and Yoon* (2014), TBD *Geiger et al.* (2014), SMOT *Dicle et al.* (2013), RMOT *Yoon et al.* (2015), CEM *Milan et al.* (2014), SegTrack *Milan et al.* (2015), MotiCon *Leal-Taixé et al.* (2014)

narios. The results are presented in Fig. 9.5. First, we can see from the table that performing training and testing in similar scenarios is beneficial. For example, the tracker trained on ADL-Rundle-6 achieves the best performance on ADL-Rundle-8. Second, trackers trained on the five training sequences perform reasonably well on all the test sequences. In some cases, cross-domain testing even improves the results. For instance, on the test sequence KITTI-17, the tracker trained on PETS09-S2L1 achieves better performance than the one trained on KITTI-13. Recall that our features used in data association are similarity metrics between targets and detections, which are not designed for specific scenarios. As a result, our method learns the similarity function which can be generalized across different sequences.

9.4.2 Evaluation on Test Set

After the analysis on the validation set, we perform training with all the training sequences, and test the trained trackers on the test set according to Table 9.2, where we use 10 templates in data association. We submitted our results to the MOT Benchmark website *Leal-Taixé et al.* for evaluation. Table 9.4 shows our tracking performance on the test set, where we compare our tracker (MDP REinforcement Learning, MDP REL) with the state-of-the-art methods tested on the MOT benchmark. As we can see from the table, our tracker improves 7% in MOTA compared with the second best published tracker, and achieves the best performance in terms of mostly tracked and mostly lost targets even though it works in the online mode. The superior performance demonstrates the advantages of our learning to track strategy with MDPs. Fig. 9.6 shows sampled tracking results on the 11 sequences in the test set (see *Xiang* (2015) for the technical report with evaluation on individual test sequences and the tracking videos).

We also evaluated a variation of our tracking method (MDP OFfline Learning, MDP OFL), where we construct training examples to learn the similarity function offline as in the traditional way. In order to use the same features as in MDP REL, we link true positive detections to form trajectory of the target using the ground truth annotations. Positive (Negative) examples are pairs of target and detection that should (not) be linked between adjacent video frames. We collect 45,005 examples to learn 6 similarity functions according to Table 9.2, and use them in our MDP framework for testing. As we can see in Table 9.4, MDP OFL also achieves very competitive performance compared to other methods, which verifies the robustness of our tracking framework. More importantly, MDP REL achieves better performance than offline training by using 1,397 training examples only in our experiments. With 3% of the training data as in offline learning but achieving similar or even better performance, we demonstrate the benefit of our reinforcement learning algorithm for multiple object tracking.

9.5 Conclusion

We have proposed a novel online multi-object tracking framework based on Markov decision processes, where the lifetime of an object is modeled with a MDP with four

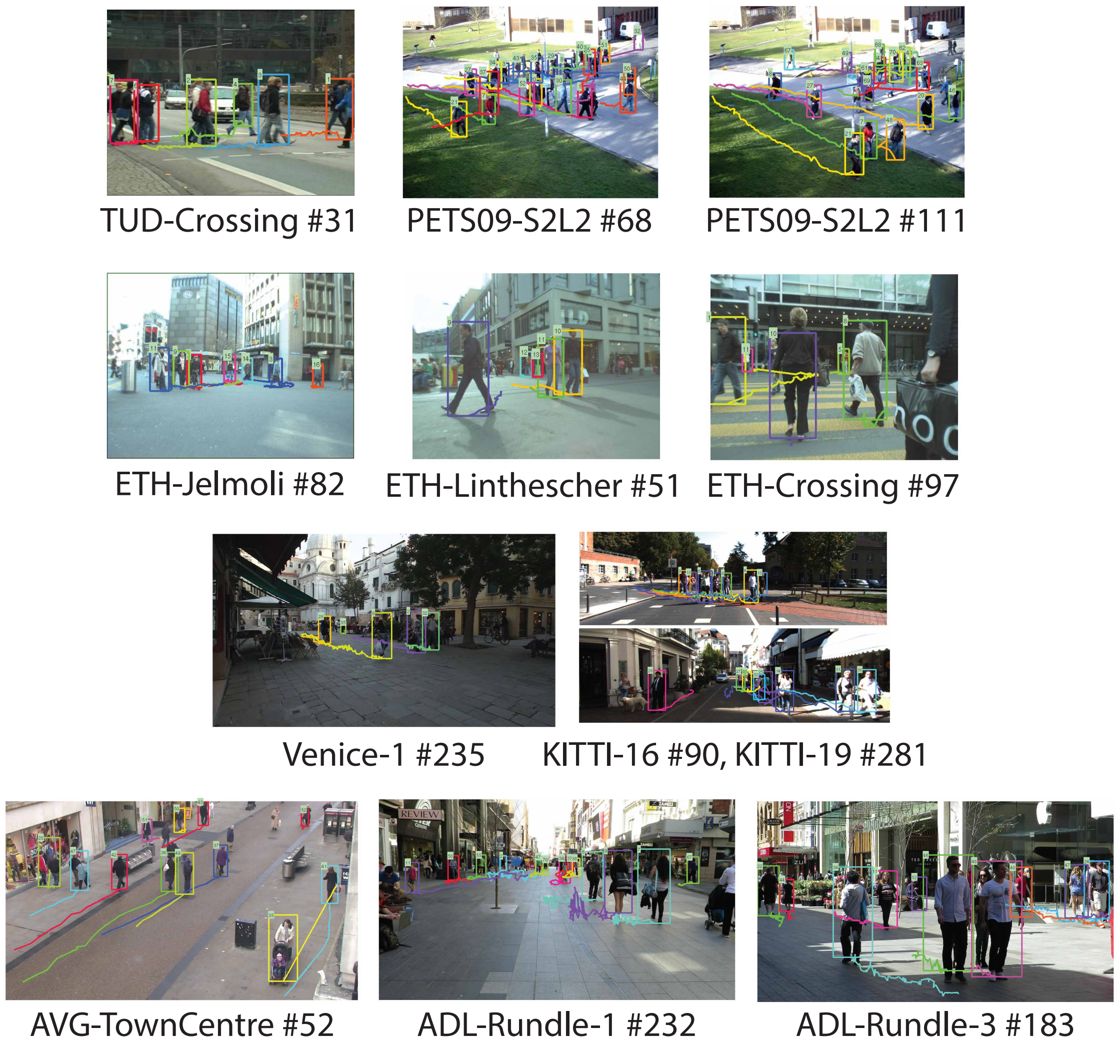


Figure 9.6: Tracking results on the test sequences in the MOT benchmark.

subspaces of states (Active, Tracked, Lost and Inactive). The state transitions in the MDP naturally handle the birth/death and appearance/disappearance of objects in tracking. A similarity function for data association is learned as part of the MDP policy with reinforcement learning. Our framework is general to be integrated with different techniques in object detection, single object tracking and data association by using them for MDP policy learning. We have tested our implementation of the tracking framework on the challenging MOT Benchmark, which outperforms the state-of-the-art methods tested on the benchmark by notable margins.

CHAPTER X

Conclusion and Future Work

I have introduced three 3D object representations for object recognition from a single image: the 3D aspect part representation *Xiang and Savarese* (2012), the 3D aspectlet representation *Xiang and Savarese* (2013) and the 3D voxel pattern representation *Xiang et al.* (2015b). These representations are designed to handle different challenges in object recognition, and have their own advantages.

The 3D aspect part representation is suitable for representing object categories whose 3D surfaces can be approximated by a set of 3D planes, such as cars, chairs, and so on. In these cases, our aspect layout model based on the 3D aspect part representation is capable of handling the appearance variation of object categories due to viewpoint transformations. The 3D aspectlet representation is designed to handle occluded or truncated objects in complex scenes. In these cases, only a portion of the object is visible due to occlusion or truncation. The 3D aspectlets are built to detect partial visible objects. Based on the 3D aspectlet representation, we have proposed a probabilistic model called spatial layout model to detect multiple objects from a single image and reason about occlusions between the objects in the scene. The 3D voxel pattern representation is designed to handle viewpoint variation, occlusion and truncation jointly in a data-driven manner. It can capture the distributions of viewpoint, occlusion and truncation in the data. We have shown that by training detectors for 3D voxel patterns, our method achieves the state-of-the-art recognition performance for car on the KITTI detection benchmark *Geiger et al.* (2012).

Furthermore, I have described our efforts on building a large scale 3D object recognition dataset *Xiang et al.* (2014a) by aligning 3D CAD models with 12 rigid categories in PASCAL VOC 2012 *Everingham et al.* (b). We hope our dataset can benchmark the 3D object recognition methods in the future. I have applied our 3D aspect part representation to tackle two challenging problems in computer vision: object co-detection *Bao et al.* (2012c) and multi-view object tracking *Xiang et al.* (2014b), and applied our 3D voxel pattern representation in Convolutional Neural Network-based object detection. Finally, I introduce a novel multi-object tracking framework based on Markov Decision Processes *Xiang et al.* (2015a).

For the future works, I am exploring three directions. The first one is to extend our PASCAL3D+ dataset with more categories. As we know, human are able to recognize thousands of objects in our daily life. The ability of scaling up with the number of categories is necessary for an object recognition method to be applied to real world applications. So we aim at building a new large scale 3D object recognition dataset with 100 categories to facilitate the research of object recognition. Besides increasing the number of categories, we also plan to collect more 3D CAD models for each category, and see how these 3D CAD models can benefit object recognition in images.

For the second direction, we focus on automatically building 3D CAD models of objects from web images. As we have seen, 3D CAD models can be useful for object recognition, and we benefit from the 3D modeling community in sharing their 3D CAD models on the web *Trimble*. However, compared with the number of images on the web, we only have a small number of 3D CAD models available. So we aim at designing an object recognition method which is able to utilize images from the web and build 3D CAD models of the objects in these images. In this way, we can largely increase the number of 3D CAD models for people to use. Besides, the problem of automatically building 3D CAD models requires a method to recognize the objects from images and reconstruct the objects in 3D, which is an important problem in computer vision and deserves more efforts in solving it.

For the last direction, I plan to explore deep learning techniques for object recognition while leveraging the advantages of 3D object representations. Recent progress on object recognition has demonstrated the ability of deep neural networks in image classification and 2D object detection. However, it is still an open question about how to apply deep learning in recognizing 3D properties of objects such as 3D pose or 3D shape. If we could combine deep learning with 3D object representations, we may further boost the performance on 3D object recognition. A straightforward way could be using features from deep neural networks to replace traditional features such as SIFT *Lowe* (2004) or HOG *Dalal and Triggs* (2005). For example, we could use features learned from deep neural networks in our aspect layout model instead of using HOG, which can improve the detection performance as shown in the literature *Girshick et al.* (2014). In addition to using more powerful features from deep learning, I also would like to explore different ways in building deep architectures that are able to capture the 3D nature of object intrinsically.

BIBLIOGRAPHY

BIBLIOGRAPHY

Ansar, A., and K. Daniilidis (2003), Linear pose estimation from points or lines, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *25*(5), 578–589.

Arbelaez, P., J. Pont-Tuset, J. Barron, F. Marques, and J. Malik (2014), Multiscale combinatorial grouping, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 328–335.

Arie-Nachimson, M., and R. Basri (2009), Constructing implicit 3d shape models for pose estimation, in *International Conference on Computer Vision (ICCV)*, pp. 1341–1348.

Babenko, B., M.-H. Yang, and S. Belongie (2011), Robust object tracking with online multiple instance learning, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *33*(8), 1619–1632.

Bae, S.-H., and K.-J. Yoon (2014), Robust online multi-object tracking based on tracklet confidence and online discriminative appearance learning, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1218–1225.

Bao, C., Y. Wu, H. Ling, and H. Ji (2012a), Real time robust l1 tracker using accelerated proximal gradient approach, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1830–1837.

Bao, S. Y., and S. Savarese (2011), Semantic structure from motion, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2025–2032.

Bao, S. Y., M. Sun, and S. Savarese (2011), Toward coherent object detection and scene layout understanding, *Image and Vision Computing*, *29*(9), 569–579.

Bao, S. Y., M. Bagra, Y.-W. Chao, and S. Savarese (2012b), Semantic structure from motion with points, regions, and objects, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2703–2710.

Bao, S. Y., Y. Xiang, and S. Savarese (2012c), Object co-detection, in *European Conference on Computer Vision (ECCV)*, pp. 86–101.

Batra, D., A. Kowdle, D. Parikh, J. Luo, and T. Chen (2010), icoseg: Interactive cosegmentation with intelligent scribble guidance, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3169–3176.

Bellman, R. (1957), A markovian decision process, *Journal of Mathematics and Mechanics*, *6*(5), 679–684.

Berclaz, J., F. Fleuret, E. Turetken, and P. Fua (2011), Multiple object tracking using k-shortest paths optimization, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *33*(9), 1806–1819.

Berg, A. C., T. L. Berg, and J. Malik (2005), Shape matching and object recognition using low distortion correspondences, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 26–33.

Boser, B. E., I. M. Guyon, and V. N. Vapnik (1992), A training algorithm for optimal margin classifiers, in *Proceedings of the fifth annual workshop on Computational learning theory*, pp. 144–152.

Bouguet, J.-Y. (2001), Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm, *Intel Corporation*, *5*, 1–10.

Bourdev, L., and J. Malik (2009), Poselets: Body part detectors trained using 3d human pose annotations, in *International Conference on Computer Vision (ICCV)*, pp. 1365–1372.

Breitenstein, M. D., F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool (2011), Online multiperson tracking-by-detection from a single, uncalibrated camera, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *33*(9), 1820–1833.

Butt, A. A., and R. T. Collins (2013), Multi-target tracking by lagrangian relaxation to min-cost network flow, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1846–1853.

Chen, C., A. Seff, A. Kornhauser, and J. Xiao (2015a), Deepdriving: Learning affordance for direct perception in autonomous driving, *arXiv preprint arXiv:1505.00256.*

Chen, X., A. Shrivastava, and A. Gupta (2014), Enriching visual knowledge bases via object discovery and segmentation, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2035–2042.

Chen, X., K. Kundu, Y. Zhu, A. Berneshawi, H. Ma, S. Fidler, and R. Urtasun (2015b), 3d object proposals for accurate object class detection, in *Advances in Neural Information Processing Systems (NIPS).*

Chiu, H., L. Kaelbling, and T. Lozano-Pérez (2007), Virtual training for multi-view object class recognition, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Choi, C., and H. I. Christensen (2010), Real-time 3d model-based tracking using edge and keypoint features for robotic manipulation, in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4048–4055.

Choi, W., and S. Savarese (2010), Multiple target tracking in world coordinate with single, minimally calibrated camera, in *European Conference on Computer Vision (ECCV)*, pp. 553–567.

Choi, W., Y.-W. Chao, C. Pantofaru, and S. Savarese (2013a), Understanding indoor scenes using 3d geometric phrases, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 33–40.

Choi, W., C. Pantofaru, and S. Savarese (2013b), A general framework for tracking multiple people from a moving camera, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *35*(7), 1577–1591.

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

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009), Imagenet: A large-scale hierarchical image database, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255.

Desai, C., D. Ramanan, and C. C. Fowlkes (2011), Discriminative models for multi-class object layout, *International Journal of Computer Vision (IJCV)*, *95*(1), 1–12.

Dickinson, S. J., A. P. Pentland, and A. Rosenfeld (1991), From volumes to views: An approach to 3-d object recognition, in *Workshop on Directions in Automated CAD-Based Vision*, pp. 85–96.

Dicle, C., O. I. Camps, and M. Sznaier (2013), The way they move: Tracking multiple targets with similar appearance, in *International Conference on Computer Vision (ICCV)*, pp. 2304–2311.

Divvala, S. K., A. A. Efros, and M. Hebert (2012), How important are deformable parts in the deformable parts model?, in *European Conference on Computer Vision Workshops (ECCVW)*, pp. 31–40.

Divvala, S. K., A. Farhadi, and C. Guestrin (2014), Learning everything about anything: Webly-supervised visual concept learning, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3270–3277.

Dollár, P., R. Appel, S. Belongie, and P. Perona (2014), Fast feature pyramids for object detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *36*(8), 1532–1545.

Drummond, T., and R. Cipolla (2002), Real-time visual tracking of complex structures, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *24*(7), 932–946.

Ess, A., B. Leibe, and L. V. Gool (2007), Depth and appearance for mobile scene analysis, in *International Conference on Computer Vision (ICCV)*, pp. 1–8.

Ess, A., B. Leibe, K. Schindler, and L. V. Gool (2008), A mobile vision system for robust multi-person tracking, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Everingham, M., L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (a), The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

Everingham, M., L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (b), The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results, http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Everingham, M., A. Zisserman, C. K. I. Williams, and L. Van Gool (c), The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results, http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf.

Everingham, M., L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman (2010), The pascal visual object classes (voc) challenge, *International journal of computer vision (IJCV)*, *88*(2), 303–338.

Farhadi, A., I. Endres, D. Hoiem, and D. Forsyth (2009a), Describing objects by their attributes, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1778–1785.

Farhadi, A., M. Tabrizi, I. Endres, and D. Forsyth (2009b), A latent model of discriminative aspect, in *International Conference on Computer Vision (ICCV)*, pp. 948–955.

Fei-Fei, L., R. Fergus, and P. Perona (2007), Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories, *Computer Vision and Image Understanding (CVIU)*, *106*(1), 59–70.

Feldman, A., M. Hybinette, and T. Balch (2012), The multi-iterative closest point tracker: An online algorithm for tracking multiple interacting targets, *Journal of Field Robotics*, *29*(2), 258–276.

Felzenszwalb, P. F., and D. P. Huttenlocher (2005), Pictorial structures for object recognition, *International Journal of Computer Vision (IJCV)*, *61*(1), 55–79.

Felzenszwalb, P. F., R. B. Girshick, D. McAllester, and D. Ramanan (2010), Object detection with discriminatively trained part-based models, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *32*(9), 1627–1645.

Fergus, R., P. Perona, and A. Zisserman (2003), Object class recognition by unsupervised scale-invariant learning, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 264–271.

Ferrari, V., T. Tuytelaars, and L. Van Gool (2006), Simultaneous object recognition and segmentation from single or multiple model views, *International Journal of Computer Vision (IJCV)*, *67*(2), 159–188.

Ferrari, V., F. Jurie, and C. Schmid (2010), From images to shape models for object detection, *International journal of computer vision (IJCV)*, *87*(3), 284–303.

Fidler, S., S. Dickinson, and R. Urtasun (2012), 3d object detection and viewpoint estimation with a deformable 3d cuboid model, in *Advances in Neural Information Processing Systems (NIPS)*, pp. 611–619.

Frey, B. J., and D. Dueck (2007), Clustering by passing messages between data points, *Science*, *315*(5814), 972–976.

Gao, T., B. Packer, and D. Koller (2011), A segmentation-aware object detection model with occlusion handling, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1361–1368.

Geiger, A., P. Lenz, C. Stiller, and R. Urtasun (), Kitti object detection benchmark, `http://www.cvlibs.net/datasets/kitti/eval_object.php`, accessed: 2014-11-12.

Geiger, A., P. Lenz, and R. Urtasun (2012), Are we ready for autonomous driving? the kitti vision benchmark suite, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3354–3361.

Geiger, A., M. Lauer, C. Wojek, C. Stiller, and R. Urtasun (2014), 3d traffic scene understanding from movable platforms, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *36*(5), 1012–1025.

Girshick, R. (2015), Fast r-cnn, *arXiv preprint arXiv:1504.08083*.

Girshick, R., J. Donahue, T. Darrell, and J. Malik (2014), Rich feature hierarchies for accurate object detection and semantic segmentation, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 580–587.

Glasner, D., M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich (2011), Viewpoint-aware object detection and pose estimation, in *International Conference on Computer Vision (ICCV)*, pp. 1275–1282.

Green, P. J. (1995), Reversible jump markov chain monte carlo computation and bayesian model determination, *Biometrika*, *82*(4), 711–732.

Gu, C., and X. Ren (2010), Discriminative mixture-of-templates for viewpoint classification, in *European Conference on Computer Vision (ECCV)*, pp. 408–421.

Hare, S., A. Saffari, and P. H. Torr (2011), Struck: Structured output tracking with kernels, in *International Conference on Computer Vision (ICCV)*, pp. 263–270.

Hariharan, B., P. Arbeláez, L. Bourdev, S. Maji, and J. Malik (2011), Semantic contours from inverse detectors, in *International Conference on Computer Vision (ICCV)*, pp. 991–998.

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

Held, D., J. Levinson, and S. Thrun (2013), Precision tracking with sparse 3d and dense color 2d data, in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1138–1145.

Hochbaum, D. S., and V. Singh (2009), An efficient algorithm for co-segmentation, in *International Conference on Computer Vision (ICCV)*, pp. 269–276.

Hofmann, M., D. Wolf, and G. Rigoll (2012), Hypergraphs for joint multi-view reconstruction and multi-object tracking, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3650–3657.

Hoiem, D., C. Rother, and J. Winn (2007a), 3d layoutcrf for multi-view object class recognition and segmentation, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Hoiem, D., A. Stein, A. Efros, and M. Hebert (2007b), Recovering occlusion boundaries from a single image, in *International Conference on Computer Vision (ICCV)*, pp. 1–8.

Hoiem, D., A. A. Efros, and M. Hebert (2008), Putting objects in perspective, *International Journal of Computer Vision (IJCV)*, *80*(1), 3–15.

Hoiem, D., Y. Chodpathumwan, and Q. Dai (2012), Diagnosing error in object detectors, in *European Conference on Computer Vision (ECCV)*, pp. 340–353.

Hsiao, E., A. Collet, and M. Hebert (2010), Making specific features less discriminative to improve point-based 3d object recognition, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2653–2660.

Huang, Q.-X., B. Adams, and M. Wand (2007), Bayesian surface reconstruction via iterative scan alignment to an optimized prototype, in *Symposium on Geometry Processing*, pp. 213–223.

Janoch, A., S. Karayev, Y. Jia, J. T. Barron, M. Fritz, K. Saenko, and T. Darrell (2013), A category-level 3-d object dataset: Putting the kinect to work, in *International Conference on Computer Vision Workshop on Consumer Depth Cameras in Computer Vision*, pp. 141–165.

Jia, Y., E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell (2014), Caffe: Convolutional architecture for fast feature embedding, *arXiv preprint arXiv:1408.5093*.

Joachims, T., T. Finley, and C.-N. J. Yu (2009), Cutting-plane training of structural svms, *Machine Learning*, *77*(1), 27–59.

Kaestner, R., J. Maye, Y. Pilat, and R. Siegwart (2012), Generative object detection and tracking in 3d range data, in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3075–3081.

Kalal, Z., K. Mikolajczyk, and J. Matas (2012), Tracking-learning-detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *34*(7), 1409–1422.

Karayev, S., M. Fritz, and T. Darrell (2014), Anytime recognition of objects and scenes, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 572–579.

Karpathy, A., and L. Fei-Fei (2014), Deep visual-semantic alignments for generating image descriptions, *arXiv preprint arXiv:1412.2306*.

Keni, B., and S. Rainer (2008), Evaluating multiple object tracking performance: the clear mot metrics, *EURASIP Journal on Image and Video Processing*, *2008*, 1:1–1:10.

Khan, S. M., and M. Shah (2009), Tracking multiple occluding people by localizing on multiple scene planes, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *31*(3), 505–519.

Khan, Z., T. Balch, and F. Dellaert (2005), Mcmc-based particle filtering for tracking a variable number of interacting targets, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *27*(11), 1805–1819.

Kim, S., S. Kwak, J. Feyereisl, and B. Han (2012), Online multi-target tracking by large margin structured learning, in *Asian Conference on Computer Vision (ACCV)*, pp. 98–111.

Kitani, K. M., B. D. Ziebart, J. A. Bagnell, and M. Hebert (2012), Activity forecasting, in *European Conference on Computer Vision (ECCV)*, pp. 201–214.

Koenderink, J. J., and A. J. van Doorn (1979), The internal representation of solid shape with respect to vision, *Biological cybernetics*, *32*(4), 211–216.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012), Imagenet classification with deep convolutional neural networks, in *Neural Information Processing Systems (NIPS)*, pp. 1097–1105.

Kuo, C.-H., C. Huang, and R. Nevatia (2010), Multi-target tracking by on-line learned discriminative appearance models, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 685–692.

Lafferty, J., A. McCallum, and F. Pereira (2001), Conditional random fields: Probabilistic models for segmenting and labeling sequence data, in *International Conference on Machine Learning (ICML)*, pp. 282–289.

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

Lazebnik, S., C. Schmid, and J. Ponce (2006), Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 2169–2178.

Leal-Taixé, L., A. Milan, I. Reid, S. Roth, and K. Schindler (), Multiple object tracking benchmark, `http://motchallenge.net`.

Leal-Taixé, L., G. Pons-Moll, and B. Rosenhahn (2012), Branch-and-price global optimization for multi-view multi-target tracking, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1987–1994.

Leal-Taixé, L., M. Fenzi, A. Kuznetsova, B. Rosenhahn, and S. Savarese (2014), Learning an image-based motion context for multiple people tracking, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3542–3549.

Leal-Taixé, L., A. Milan, I. Reid, S. Roth, and K. Schindler (2015), MOTChallenge 2015: Towards a Benchmark for Multi-Target Tracking, *arXiv:1504.01942 [cs]*.

LeCun, Y., S. Chopra, R. Hadsell, M. Ranzato, and F. J. Huang (2006), A tutorial on energy-based learning, in *Predicting Structured Data*, "MIT Press.

Leibe, B., and B. Schiele (2003), Analyzing appearance and contour based methods for object categorization, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. II–409.

Leibe, B., A. Leonardis, and B. Schiele (2004), Combined object categorization and segmentation with an implicit shape model, in *European Conference on Computer Vision Workshop on statistical learning in computer vision (ECCVW)*, vol. 2, p. 7.

Lepetit, V., and P. Fua (2005), Monocular model-based 3d tracking of rigid objects: A survey, *Foundations and Trends in Computer Graphics and Vision*, *1*(1), 1–89.

Lepetit, V., F. Moreno-Noguer, and P. Fua (2009), Epnp: An accurate o (n) solution to the pnp problem, *International journal of computer vision (IJCV)*, *81*(2), 155–166.

Li, B., T. Wu, and S.-C. Zhu (2014), Integrating context and occlusion for car detection by hierarchical and-or model, in *European Conference on Computer Vision (ECCV)*, pp. 652–667.

Li, Y., C. Huang, and R. Nevatia (2009), Learning to associate: Hybridboosted multi-target tracker for crowded scene, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2953–2960.

Liebelt, J., and C. Schmid (2010), Multi-view object class detection with a 3d geometric model, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1688–1695.

Liebelt, J., C. Schmid, and K. Schertler (2008), Viewpoint-independent object class detection using 3d feature maps, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Lim, J., H. Pirsiavash, and A. Torralba (2013), Parsing ikea objects: Fine pose estimation, in *International Conference on Computer Vision (ICCV)*, pp. 2992–2999.

Lopez-Sastre, R. J., C. Redondo-Cabrera, P. Gil-Jimenez, and S. Maldonado-Bascon (2010), ICARO: Image Collection of Annotated Real-world Objects, `http://agamenon.tsc.uah.es/Personales/rlopez/data/icaro`.

Lopez-Sastre, R. J., T. Tuytelaars, and S. Savarese (2011), Deformable part models revisited: A performance evaluation for object category pose estimation, in *International Conference on Computer Vision Workshop on Challenges and Opportunities in Robot Perception*, pp. 1052–1059.

Lowe, D. G. (1987), Three-dimensional object recognition from single two-dimensional images, *Artificial intelligence*, *31*(3), 355–395.

Lowe, D. G. (1999), Object recognition from local scale-invariant features, in *International Conference on Computer Vision (ICCV)*, pp. 1150–1157.

Lowe, D. G. (2004), Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision (IJCV)*, *60*(2), 91–110.

Lu, C.-P., G. D. Hager, and E. Mjolsness (2000), Fast and globally convergent pose estimation from video images, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *22*(6), 610–622.

Lucas, B. D., T. Kanade, et al. (1981), An iterative image registration technique with an application to stereo vision., in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 81, pp. 674–679.

Malisiewicz, T., A. Gupta, and A. A. Efros (2011), Ensemble of exemplar-svms for object detection and beyond, in *International Conference on Computer Vision (ICCV)*, pp. 89–96.

Matas, J., O. Chum, M. Urban, and T. Pajdla (2004), Robust wide-baseline stereo from maximally stable extremal regions, *Image and vision computing*, *22*(10), 761–767.

Matzen, K., and N. Snavely (2013), Nyc3dcars: A dataset of 3d vehicles in geographic context, in *International Conference on Computer Vision (ICCV)*, pp. 761–768.

Milan, A., S. Roth, and K. Schindler (2014), Continuous energy minimization for multitarget tracking, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *36*(1), 58–72.

Milan, A., L. Leal-Taixé, K. Schindler, and I. Reid (2015), Joint tracking and segmentation of multiple targets, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5397–5406.

Mnih, V., K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller (2013), Playing atari with deep reinforcement learning, *arXiv preprint arXiv:1312.5602*.

Munkres, J. (1957), Algorithms for the assignment and transportation problems, *Journal of the Society for Industrial & Applied Mathematics*, *5*(1), 32–38.

Ng, A. Y., and S. J. Russell (2000), Algorithms for inverse reinforcement learning, in *International Conference on Machine Learning (ICML)*, pp. 663–670.

Niebles, J. C., B. Han, and L. Fei-Fei (2010), Efficient extraction of human motion volumes by tracking, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 655–662.

Nister, D., and H. Stewenius (2006), Scalable recognition with a vocabulary tree, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2, pp. 2161–2168.

Oh, S., S. Russell, and S. Sastry (2009), Markov chain monte carlo data association for multi-target tracking, *IEEE Transactions on Automatic Control*, *54*(3), 481–497.

Ohn-Bar, E., and M. M. Trivedi (2014), Fast and robust object detection using visual subcategories, in *IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 179–184.

Ohn-Bar, E., and M. M. Trivedi (2015), Learning to detect vehicles by clustering appearance patterns, *IEEE Transactions on Intelligent Transportation Systems*, *16*(5), 2511–2521.

Okuma, K., A. Taleghani, N. De Freitas, J. J. Little, and D. G. Lowe (2004), A boosted particle filter: Multitarget detection and tracking, in *European Conference on Computer Vision (ECCV)*, pp. 28–39.

Oliva, A., A. Torralba, et al. (2007), The role of context in object recognition, *Trends in cognitive sciences*, *11*(12), 520–527.

Oron, S., A. Bar-Hille, and S. Avidan (2014), Extended lucas kanade tracking, in *European Conference on Computer Vision (ECCV)*, pp. 142–156.

Ozuysal, M., V. Lepetit, and P. Fua (2009), Pose estimation for category specific multiview object localization, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 778–785.

Paletta, L., G. Fritz, and C. Seifert (2005), Q-learning of sequential attention for visual object recognition from informative local descriptors, in *International Conference on Machine Learning (ICML)*, pp. 649–656.

Pauwels, K., L. Rubio, J. Diaz, and E. Ros (2013), Real-time model-based rigid object pose estimation and tracking combining dense and sparse visual cues, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2347–2354.

Payet, N., and S. Todorovic (2011), From contours to 3d object detection and pose estimation, in *International Conference on Computer Vision (ICCV)*, pp. 983–990.

Pepik, B., M. Stark, P. Gehler, and B. Schiele (2012), Teaching 3d geometry to deformable part models, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3362–3369.

Pepik, B., M. Stark, P. Gehler, and B. Schiele (2015), Multi-view and 3d deformable part models, *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, *37*(11), 2232–2245.

Pepikj, B., M. Stark, P. Gehler, and B. Schiele (2013), Occlusion patterns for object class detection, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3286–3293.

Petrovskaya, A., and S. Thrun (2008), Model based vehicle tracking for autonomous driving in urban environments, *Proceedings of Robotics: Science and Systems (RSS)*, *34*.

Pirsiavash, H., D. Ramanan, and C. C. Fowlkes (2011), Globally-optimal greedy algorithms for tracking a variable number of objects, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1201–1208.

Prisacariu, V. A., and I. D. Reid (2012), Pwp3d: Real-time segmentation and tracking of 3d objects, *International Journal of Computer Vision (IJCV)*, *98*(3), 335–354.

Ren, S., K. He, R. Girshick, and J. Sun (2015), Faster r-cnn: Towards real-time object detection with region proposal networks, *arXiv preprint arXiv:1506.01497*.

Roller, D., K. Daniilidis, and H.-H. Nagel (1993), Model-based object tracking in monocular image sequences of road traffic scenes, *International Journal of Computer Vision (IJCV)*, *10*(3), 257–281.

Rother, C., T. Minka, A. Blake, and V. Kolmogorov (2006), Cosegmentation of image pairs by histogram matching-incorporating a global constraint into mrfs, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. 993–1000.

Rothganger, F., S. Lazebnik, C. Schmid, and J. Ponce (2006), 3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints, *International Journal of Computer Vision (IJCV)*, *66*(3), 231–259.

Russakovsky, O., L.-J. Li, and L. Fei-Fei (2015a), Best of both worlds: human-machine collaboration for object annotation, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2121–2131.

Russakovsky, O., et al. (2015b), ImageNet Large Scale Visual Recognition Challenge, *International Journal of Computer Vision (IJCV)*, pp. 1–42.

Russell, B. C., A. Torralba, K. P. Murphy, and W. T. Freeman (2008), Labelme: a database and web-based tool for image annotation, *International journal of computer vision (IJCV)*, *77*(1-3), 157–173.

Savarese, S., and L. Fei-Fei (2007), 3d generic object categorization, localization and pose estimation, in *International Conference on Computer Vision (ICCV)*, pp. 1–8.

Sermanet, P., D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun (2013), Overfeat: Integrated recognition, localization and detection using convolutional networks, *arXiv preprint arXiv:1312.6229*.

Shotton, J., J. Winn, C. Rother, and A. Criminisi (2009), Textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context, *International Journal of Computer Vision (IJCV)*, *81*(1), 2–23.

Silberman, N., D. Hoiem, P. Kohli, and R. Fergus (2012), Indoor segmentation and support inference from rgbd images, in *European Conference on Computer Vision (ECCV)*, pp. 746–760.

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

Song, X., J. Cui, H. Zha, and H. Zhao (2008), Vision-based multiple interacting targets tracking via on-line supervised learning, in *European Conference on Computer Vision (ECCV)*, pp. 642–655.

Stark, M., P. Lies, M. Zillich, J. Wyatt, and B. Schiele (2008), Functional object class detection based on learned affordance cues, in *International Conference on Computer Vision Systems*, pp. 435–444.

Stark, M., M. Goesele, and B. Schiele (2010), Back to the future: Learning shape models from 3d cad data, in *British Machine Vision Conference (BMVC)*, pp. 106.1–106.11.

Su, H., M. Sun, L. Fei-Fei, and S. Savarese (2009), Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories, in *International Conference on Computer Vision (ICCV)*, pp. 213–220.

Su, H., C. R. Qi, Y. Li, and L. Guibas (2015), Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views, *arXiv preprint arXiv:1505.05641*.

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 *European Conference on Computer Vision (ECCV)*, pp. 658–671.

Supancic III, J. S., and D. Ramanan (2013), Self-paced learning for long-term tracking, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2379–2386.

Taigman, Y., M. Yang, M. Ranzato, and L. Wolf (2014), Deepface: Closing the gap to human-level performance in face verification, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1701–1708.

Thomas, A., V. Ferrari, B. Leibe, T. Tuytelaars, B. Schiele, and L. Van Gool (2006), Towards multi-view object class detection, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1589–1596.

Thomas, A., V. Ferrari, B. Leibe, T. Tuytelaars, and L. V. Gool (2007), Depth-from-recognition: Inferring metadata by cognitive feedback, in *International Conference on Computer Vision Workshop on 3D Representations for Recognition (3dRR)*, pp. 1–8.

Tian, Y., P. Luo, X. Wang, and X. Tang (2015), Deep learning strong parts for pedestrian detection, in *International Conference on Computer Vision (ICCV)*.

Toshev, A., J. Shi, and K. Daniilidis (2007), Image matching via saliency region correspondences, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Trimble (), Trimble 3d warehouse, `http://3dwarehouse.sketchup.com`.

Tsochantaridis, I., T. Hofmann, T. Joachims, and Y. Altun (2004), Support vector machine learning for interdependent and structured output spaces, in *International Conference on Machine Learning (ICML)*, pp. 104–111.

Tulsiani, S., and J. Malik (2014), Viewpoints and keypoints, *arXiv preprint arXiv:1411.6067*.

Uijlings, J. R., K. E. van de Sande, T. Gevers, and A. W. Smeulders (2013), Selective search for object recognition, *International Journal of Computer Vision (IJCV)*, *104*(2), 154–171.

Vedaldi, A., and A. Zisserman (2009), Structured output regression for detection with partial truncation, in *Neural Information Processing Systems (NIPS)*, pp. 1928–1936.

Viola, P., and M. Jones (2001), Rapid object detection using a boosted cascade of simple features, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. I–511.

Viola, P., and M. J. Jones (2004), Robust real-time face detection, *International Journal of Computer Vision (IJCV)*, *57*(2), 137–154.

Wang, X., T. X. Han, and S. Yan (2009), An hog-lbp human detector with partial occlusion handling, in *International Conference on Computer Vision (ICCV)*, pp. 32–39.

Wang, X., M. Yang, S. Zhu, and Y. Lin (2013), Regionlets for generic object detection, in *International Conference on Computer Vision (ICCV)*, pp. 17–24.

Winn, J., and J. Shotton (2006), The layout consistent random field for recognizing and segmenting partially occluded objects, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 37–44.

Wojek, C., S. Walk, S. Roth, and B. Schiele (2011), Monocular 3d scene understanding with explicit occlusion reasoning, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1993–2000.

Wu, B., and R. Nevatia (2005), Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 90–97.

Wu, Y., J. Lim, and M.-H. Yang (2013), Online object tracking: A benchmark, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2411–2418.

Xiang, Y. (2014), Multiview Tracking, `http://cvgl.stanford.edu/projects/multiview\_tracking`.

Xiang, Y. (2015), MDP Tracking, `http://cvgl.stanford.edu/projects/MDP_tracking`.

Xiang, Y., and S. Savarese (2012), Estimating the aspect layout of object categories, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3410–3417.

Xiang, Y., and S. Savarese (2013), Object detection by 3d aspectlets and occlusion reasoning, in *International Conference on Computer Vision Workshop (ICCVW)*, pp. 530–537.

Xiang, Y., R. Mottaghi, and S. Savarese (2014a), Beyond pascal: A benchmark for 3d object detection in the wild, in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 75–82.

Xiang, Y., C. Song, R. Mottaghi, and S. Savarese (2014b), Monocular multiview object tracking with 3d aspect parts, in *European Conference on Computer Vision (ECCV)*, pp. 220–235.

Xiang, Y., A. Alahi, and S. Savarese (2015a), Learning to track: Online multi-object tracking by decision making, in *International Conference on Computer Vision (ICCV)*.

Xiang, Y., W. Choi, Y. Lin, and S. Savarese (2015b), Data-driven 3d voxel patterns for object category recognition, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1903–1911.

Xiao, J., J. Hays, K. Ehinger, A. Oliva, and A. Torralba (2010), Sun database: Large-scale scene recognition from abbey to zoo, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3485–3492.

Xiao, J., A. Owens, and A. Torralba (2013), Sun3d: A database of big spaces reconstructed using sfm and object labels, in *International Conference on Computer Vision (ICCV)*, pp. 1625–1632.

Yan, P., S. Khan, and M. Shah (2007), 3d model based object class detection in an arbitrary view, in *International Conference on Computer Vision (ICCV)*, pp. 1–6.

Yang, B., and R. Nevatia (2012), An online learned crf model for multi-target tracking, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2034–2041.

Yao, B., and L. Fei-Fei (2010), Modeling mutual context of object and human pose in human-object interaction activities, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17–24.

Yao, R., Q. Shi, C. Shen, Y. Zhang, and A. van den Hengel (2013), Part-based visual tracking with online latent structural learning, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2363–2370.

Yedidia, J. S., W. T. Freeman, and Y. Weiss (2003), Understanding belief propagation and its generalizations, *Exploring artificial intelligence in the new millennium*, *8*, 236–239.

Yoon, J. H., M.-H. Yang, J. Lim, and K.-J. Yoon (2015), Bayesian multi-object tracking using motion context from multiple objects, in *IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 33–40.

Zhang, L., Y. Li, and R. Nevatia (2008), Global data association for multi-object tracking using network flows, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–8.

Zhang, S., R. Benenson, and B. Schiele (2015), Filtered channel features for pedestrian detection, *arXiv preprint arXiv:1501.05759*.

Zia, M. Z., M. Stark, B. Schiele, and K. Schindler (2011), Revisiting 3d geometric models for accurate object shape and pose, in *IEEE Conference on Computer Vision Workshops (ICCVW)*, pp. 569–576.

Zia, M. Z., M. Stark, and K. Schindler (2013), Explicit occlusion modeling for 3d object class representations, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3326–3333.

Zia, M. Z., M. Stark, and K. Schindler (2014), Are cars just 3d boxes?–jointly estimating the 3d shape of multiple objects, in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3678–3685.

Zitnick, C. L., and P. Dollár (2014), Edge boxes: Locating object proposals from edges, in *European Conference on Computer Vision (ECCV)*, pp. 391–405.