Geometric and Semantic Scene Understanding

by

Yingze Bao

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

For many people.

ACKNOWLEDGEMENTS

When I am about to finish writing this dissertation, I cannot stop recalling what happened in the past few years. I feel very lucky to be a PhD student who can dive into his favorite research topic in a lovely campus. I feel even more lucky to have met many people who helped me, encouraged me, and advised me.

First and foremost, I would like to express my deepest gratitude to my adviser Prof. Silvio Savarese, who is an exceptional researcher and an encouraging teacher. Prof. Savarese has made his students to have broad mathematics background, solid computer programing skills, and accurate understanding of the latest computer-vision technology. Prof. Savarese is always selfless to guide and support his students. I am greatly impressed by his acute insights during our academic discussions, his patience at me for overcoming technical challenges, and his every efforts for funding me and other students to choose our favorite research directions. I also owe many thanks to Prof. Savarese for his precious help for finding internship opportunities and deterministic advice on choosing future career.

I am also grateful to my mentors and colleagues in my internship companies. I feel fortunate to work with and learn from these world-class researchers: Radek Grzeszczuk, Manmohan Chandraker, Yuanqing Lin, Kari Pulli and Kihwan Kim. Special thanks to Dr. Chandraker and Dr. Kim for their day-to-day mentorship, through which I learned the steps for converting brilliant research ideas into robust solutions to practical problems.

Finally, this dissertation would not be possible without other lab members: Wongun

Choi, Byungsoo Kim, Yu Xiang, Jingen Liu, Yu-Wei Chao, Shyam Kumar, Mohit Bagra, Murali Telaprolu, Ishan Mittal, Anush Mohan, Ryan Tokola, Changkyu Song, Jie Li, Zhen Zeng, Ziang Xie, and many. I would like to particularly thank Min Sun, who devoted much time into helping and guiding me as I initially started my PhD life. I also owe gratitude to the wife of my adviser, Prof. Fei-fei Li, who always gives me instructive research inspirations and important career advice.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

Geometric and Semantic Scene Understanding

by

Yingze (Sid) Bao

Silvio Savarese and Jeffrey A Fessler, Co-Chairs

Estimating the 3D structure of a scene and recognizing scene elements are two kernel functions supporting many artificial intelligence applications. The ability to achieve these two goals only using RGB images is very valuable to a low-cost system but also extremely challenging. A scene may comprise a large number of different points, regions, and objects. Identifying their existence and distinguishing their semantic properties using RGB images are related to two research topics in computer vision: geometric scene understanding and semantic scene understanding. Over the past decades, many researchers were devoted into solving the problem of geometric scene understanding such as the works in camera calibration, structure-from-motion, and dense reconstruction. Meanwhile, numerous other researchers studied the problem of semantic scene understanding including the works in object recognition, region segmentation, and layout estimation. However, these efforts of disjointly solving the geometric or the semantic understanding problem usually lead to limited estimation capability and recognition accuracy.

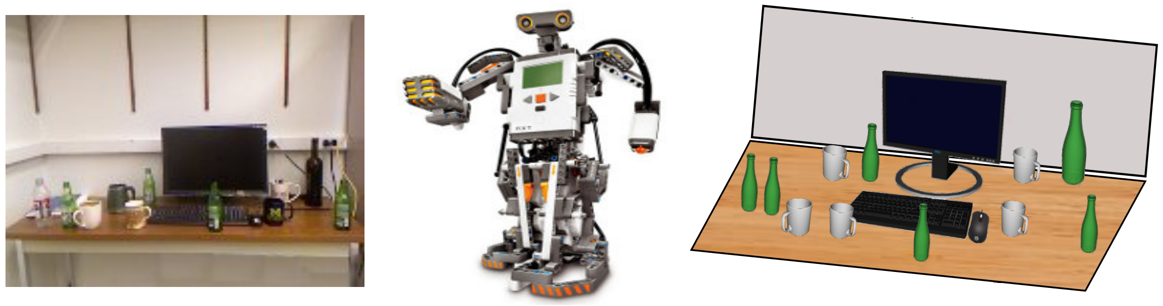
In this thesis, I will propose a novel image-based framework to jointly solve the geometric and semantic scene understanding problems, which includes the complete process of recognizing elements in a scene, estimating their spatial properties, and identifying their mutual relationships. Recognizing components in a scene provides constraints to estimate the geometric structure of the scene, while the estimated geometric structure in turn greatly helps the recognition task by providing contextual information and pruning out impossible configurations of scene components. Experiments proved that, by jointly solving the geometric understanding and semantic understanding problems, the two can be solved with an accuracy significantly higher than solving them separately. Several parts of this thesis were previously published in *Bao et al.* (2010, 2011b); *Bao and Savarese* (2011a); *Bao et al.* (2011a, 2012a, 2013).

CHAPTER I

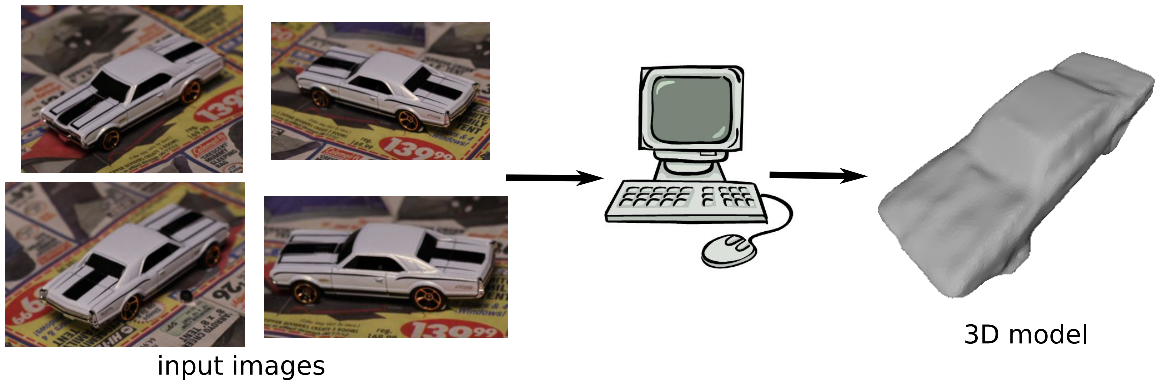
Introduction

Designing a computer system that can understand scenes in our world has been the motivation behind the efforts of many generations of engineers and scientists. On the road toward ultimately building such a system, a key challenge lies in understanding visual cues in order to interpret the 3D structure of a scene world and determine the existence of different scene components. For example, in robotic manipulation tasks, a robotic agent must recognize the target object and localize it within the application environment (Figure 1.1a). In autonomous driving tasks, an unmanned car needs to detect and localize pedestrians and other vehicles on the road (Figure 1.1b). In automatic 3D modeling tasks, a computer system needs to distinguish the object of interest from irrelevant background and reconstruct the 3D shape (Figure 1.1c). Different scene understanding systems may be equipped with different types of sensors such as the depth scanner, multi-spectrum camera, and regular RGB camera. Among these, the most economical solution is to only use RGB images. However, understanding a scene from RGB images alone is much more difficult than if other modalities are given as well.

This thesis provides a novel framework for understanding scenes only from RGB images. A complete and effective scene understanding system should solve two sub problems: (1) understanding the semantic properties of a scene, and (2) understand-


(a) Robotic manipulation. In order to execute any physical operations, an agent needs to understand the surrounding scene (working environment).

(b) Autonomous Driving. The autonomous vehicle (white truck) has to accurately estimate the 3D structure of the scene and recognize other vehicles in the scene.



(c) Automatic image-based modeling. A modeling algorithm recognizes interesting object from an input image, then estimates its 3D shape.

Figure 1.1: Applications of scene understanding

ing the geometric properties of a scene. These two sub-problems are illustrated in Figure 1.2. The inputs to the proposed scene understanding framework are images, two examples of which are shown in Figure 1.2(a). Figure 1.2(b) illustrates results of semantic scene understanding, i.e. assigning semantic labels (e.g. object categories) to individual components of a scene. The objects are localized using either bounding boxes or polygons in images. The recognized semantic properties of these scene components are labeled by color. On the other hand, understanding a scene's geometric properties indicates the process of inferring the 3D configurations of the components constituting the scene. Figure 1.2(c) shows the results of geometric scene understanding, which consists of two goals: estimating the positions, poses, and shapes of the components in the scene; estimating the locations and poses of the cameras corresponding to each of the images. Notice that geometric understanding alone only recovers the 3D configuration of a scene, without semantic labels for the components. A joint solution to these two sub-problems, (i.e., geometric and semantic scene understanding) recovers the 3D configuration of the scene components, while also estimating their semantic labels, as Figure 1.2(d) illustrates.

Achieving the goal of semantic and geometric scene understanding is never an easy task. Many problematic factors make it complicated. Typically, a scene consists of a large number of components. The attempt to individually recognize any single scene component without considering contextual information can be difficult due to intra-class variation and occlusion. Intra-class variation indicates the phenomenon that the set of objects classified as the same semantic category may have different appearances. Occlusion indicates the phenomenon that an object may be partially or fully invisible in images due to other objects lying between it and the camera. In contrast to detecting objects individually, objects can be identified more robustly if contextual information and 3D structural information are provided. For example, we are very unlikely to observe a cup hovering in the air, or a car parked in a tree.

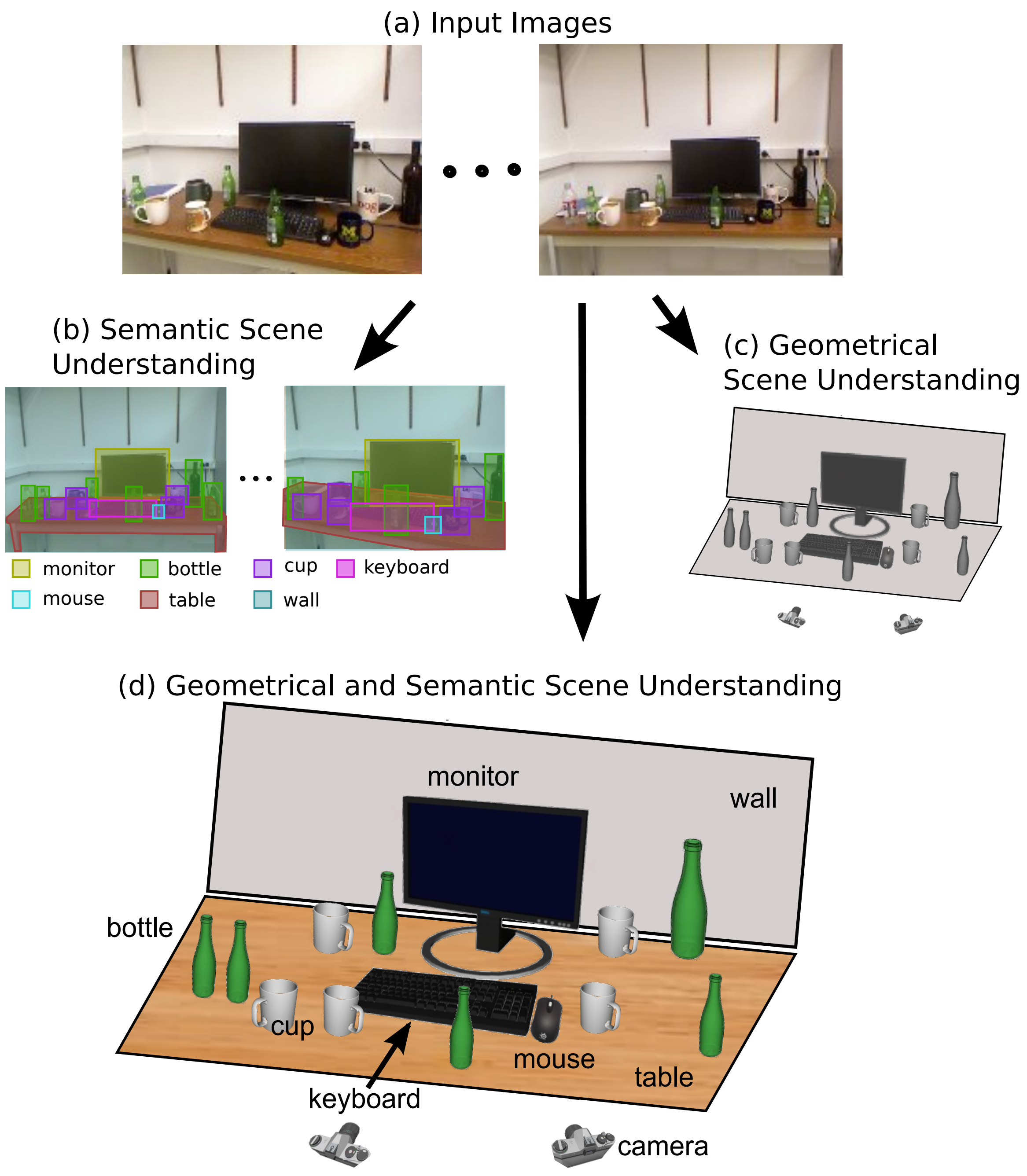

(a) Input Images
(b) Semantic Scene Understanding
monitor
bottle
cup
keyboard
mouse
table
wall
(c) Geometrical Scene Understanding
(d) Geometrical and Semantic Scene Understanding
monitor
wall
bottle
cup
mouse
table
keyboard
camera

Figure 1.2: Semantic and geometric scene understanding

The relative geometric poses of scene components can help eliminate infeasible object configurations. The desired contextual and geometric information of objects is thus related to the results of geometric scene understanding.

Nevertheless, geometric scene understanding is also nontrivial. Most conventional mechanisms for estimating the 3D structure of a scene are based on detecting and matching low-level local image features. Such process highly depends on the quantities and qualities of feature matches. Unfortunately, the process of matching local scene features is fragile in many situations such as wide camera baseline, shading variation, and illumination change. Local feature matching can be made much more robust if high-level object information is provided as guidance. Indeed, the object-level recognition and matching information per se can serve as a stable cue for geometry estimation. Numerous efforts have been made toward solving the geometric and semantic scene understanding problem. However, most of previous works either focus on the semantic recognition perspective or on the geometric estimation perspective. Hence, they cannot leverage the two related estimation problems to help solve one another.

The scene understanding framework proposed in this thesis bears significant novelty compared to most previous works, in that the geometric understanding and the semantic understanding problems are solved coherently in this framework. The results of recognizing scene components including points, regions, and objects impose strong constraints on the geometric understanding process, whose results in turn can help solve the semantic understanding problem by pruning out geometrically impossible configurations. It can be demonstrated that a joint semantic and geometric solution not only gives a more complete scene understanding result, but also yields higher robustness and accuracy than if semantic understanding and geometric understanding are solved independently. Joint geometric and semantic understanding can be accomplished if only one image is provided, or multiple images capturing the same scene are provided. We will propose a method for the scene understanding using a single image in Chapter II, and a method for the scene understanding using multiple images in Chapter III.

In the case when multiple images are provided as input, detecting and matching objects across images is the first step in robustly inferring their 3D configurations which will further assist the camera pose estimation. However, the task of detecting and matching the objects from multiple images largely remains an open problem. While numerous researchers have proposed methods for detecting and matching points or regions, little existing literature discusses the problem of effectively detecting and matching objects. The challenges in this problem lie in many aspects, especially the fact that it is difficult to predict the appearance change of a complex object given large camera viewpoint change. Notice that matching points or regions usually does not suffer from this problem, because the appearance variation of points or regions can be modeled by a homography transformation which is a function of only the camera view point change. In Chapter IV, we introduce a novel and effective approach to detecting and matching rigid objects. By effectively measuring object self-occlusions and viewpoint transformations, the proposed method is able to obtain a higher object detection and pose estimation accuracy than if objects were to be recognized from each image individually.

The goal of geometric and semantic scene understanding not only includes recognizing scene components and identifying their 3D locations, but also involves with recovering detailed 3D shapes of each individual object. Accurately localizing objects in 3D space and associating their identities to image measurements provides necessary but not sufficient conditions for densely reconstructing individual 3D shapes. Conventional dense 3D reconstruction methods heavily rely on local feature matching, which tends to fail given objects which are specular or lack texture. Unfortunately, many commonly observed objects such as cars or fruits are such challenging objects. In

Goals	Chapter	Related Publications
Understanding a scene's geometric and semantic properties from a single uncalibrated image.	II	*Bao et al.* (2010, 2011b)
Understanding a scene's geometric and semantic properties from a set of images, whose intrinsics are provided but extrinsics are unknown.	III	*Bao and Savarese* (2011a); *Bao et al.* (2011a, 2012a); *Bao and Savarese* (2013)
Detecting and matching identical object instances from multiple images.	IV	*Bao et al.* (2012b)
Reconstructing dense 3D object shape.	V	*Bao et al.* (2013)
An application of recovering a room's layout.	VI	In preparation for future submission.

Table 1.1: Components of the proposed scene understanding system.

Chapter V, a new technique for densely reconstructing 3D rigid objects will be introduced. The new method overcomes the drawbacks of traditional multi-view stereo by incorporating semantic information in the form of learned category-level shape priors and object detections. Extensive evaluations show that this method can produce more accurate reconstructions than alternative state-of-the-art 3D reconstruction systems.

Applying the proposed scene understanding framework in solving practical problems may require certain modifications. In Chapter VI, an example of such an adaption will be given for estimating the 3D layout and foreground objects of a room from a small number of images. Multiview geometry constraints and region recognition results will be jointly used to identify the best 3D layout configuration of a room. Extensive experimental evaluation demonstrates that the results of the new room layout estimation algorithm are more complete and accurate than alternative state-of-the-art algorithms.

The following chapters detail our approach for solving the aforementioned problems. An overview of different parts of this PhD dissertation is listed in Table. 1.1.

CHAPTER II

Scene Understanding From A Single Image[1]

2.1 Introduction

When we observe a complex scene such as an office or a street, it is easy for our visual system to recognize the scene components and infer their spatial organization in the environment. In this chapter, the scene components that we focus on are objects and supporting surfaces. Objects do not appear in arbitrary locations: it is very unlikely to observe a monitor floating in the air or a car hanging from a building. Objects are rather organized in the physical space in consistent geometrical configurations: their locations and poses obey the law of physics (objects lie on supporting planes in stable configurations) and follow the conventions of human behavior. It is clear that when humans observe the environment, such constraints will help reinforce the process of joint recognition and scene layout recovery *Palmer* (1999); *Biederman et al.* (1982). The recognition of objects with the estimate of their locations, scales and poses helps infer the spatial properties of the environment (e.g., the location and orientation of the surface where objects lie), and in turn the scene layout provides strong spatial contextual cues as for where and how objects are expected to be found. Contributions in computer vision for the past decade have followed the common paradigm of recognizing objects in isolation **?***Fergus et al.* (2003);

[1]This chapter was partially previous published in *Bao et al.* (2010, 2011b).

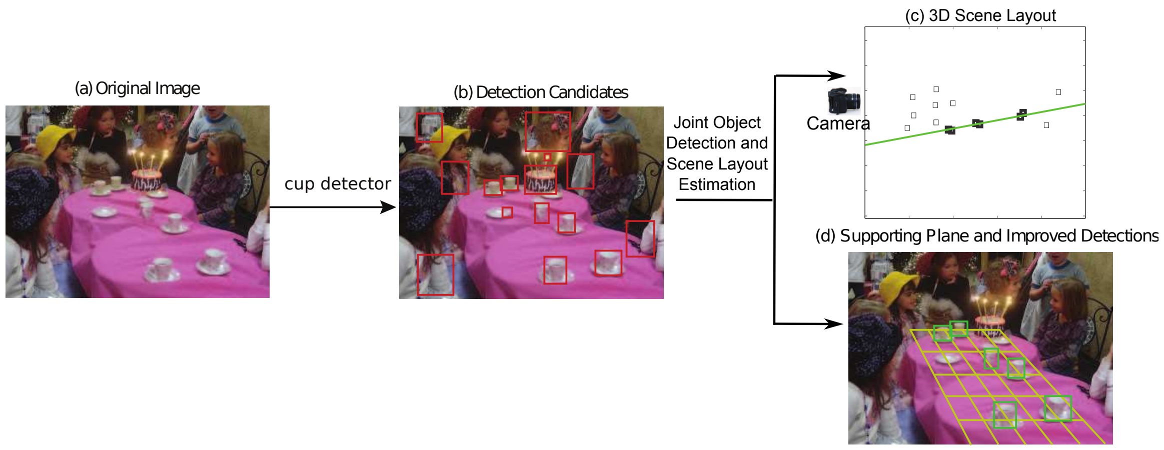

(a) Original Image
cup detector
(b) Detection Candidates
Joint Object Detection and Scene Layout Estimation
(c) 3D Scene Layout
Camera
(d) Supporting Plane and Improved Detections

Figure 2.1: A conceptual illustration of the flowchart of our algorithm. (a) Original input image with unknown camera parameters; (b) Detection candidates provided by a baseline "cup" detector; (c) The 3D layout. The figure shows the side view of the 3D reconstructed scene. The supporting plane is shown in green. Dark squares indicate the objects detected and recovered by our algorithm; light squares indicate objects detected by the baseline detector and identified as false alarms by our algorithm; (d) Our algorithm detects objects and recovers object locations and supporting plane (in gold color) orientations and locations within the 3D camera reference system from one single image. We show only a portion of the recovered supporting plane for visualization purposes.

Felzenszwalb and Huttenlocher (2000); *Leibe and Schiele* (2004); *Fei-Fei et al.* (2007), regardless of the geometrical contextual cues. It is indeed true that objects can be in general recognized even when no information about the scene layout is provided. However, we claim that joint object recognition and scene reconstruction are critical if one wants to obtain a coherent understanding of the scene as well as minimize the risk of detecting false positive examples or missing true positive ones. This ability is crucial for enabling higher level visual tasks such as event or activity recognition and in applications such as robotics, autonomous navigation, and video surveillance.

The intuition that recognition and reconstruction are mutually beneficial has been initially explored in early works of computer vision *Ohta* (1985); *Barrow and Tenenbaum* (1978); *Biederman* (1981); *Brooks* (1981); *Forsyth et al.* (1994); *Hanson and Riseman* (1978), and recently revitalized in *Hoiem et al.* (2006); *Hedau et al.* (2009, 2010); *Gould et al.* (2009); *Li et al.* (2009); *Brostow et al.* (2008); *Lee et al.* (2009);

Cornelis et al. (2008); *Saxena et al.* (2009); *Gupta et al.* (2010); *Payet and Todorovic* (2011a); *Sudderth et al.* (2006). In Hoiem et al. *Hoiem et al.* (2006), the process of detecting objects in a complex scene is enhanced by introducing the geometrical contextual information of the scene layout *Hoiem et al.* (2005) (e.g., vertical surfaces, ground horizontal planes, etc.). More explicit reasoning on the relationship between supporting planes and objects has been investigated in Hoiem et al. *Hoiem et al.* (2008b) and Hedau et al. *Hedau et al.* (2009, 2010). Hedau et al. *Hedau et al.* (2009, 2010) introduced a flexible methodology for estimating the layout of indoor scenes by modeling the scene using a 3D cube representation. Following our preliminary study *Bao et al.* (2010), we too advocate the importance of geometrical contextual reasoning for object recognition and focus on demonstrating that the contextual cues provided by object location and pose can be used, in turn, to reinforce the detection and prune out false alarms. Our key idea is that objects' locations and poses in the 3D space are not arbitrarily distributed but rather constrained by the fact that objects must lie on one or multiple supporting surfaces. We model such supporting surfaces by hidden parameters (i.e. not explicitly observed) and seek a configuration of objects and supporting surfaces in the 3D space that best explains the observations, including the estimation of each object's location, scale and pose. To this end, we leverage on recent methods for detecting multi-category objects and estimating their poses accurately from a single image *Savarese and Fei-Fei* (2007); *Liebelt et al.* (2008); *Su et al.* (2009); *Ozuysal et al.* (2009); **?**); *Farhadi et al.* (2009). Unlike *Hoiem et al.* (2006), where contextual information was partially provided by the explicit estimation of surface orientation using the geometric context operator *Hoiem et al.* (2005), we only use the objects *themselves* for extracting contextual cues. Thus, we do not require supporting planes or other scene surfaces to be visible or detectable in order to perform the joint recognition and reconstruction. Rather, supporting planes are implicitly estimated from the observation of the object location and pose in the image. Moreover, our camera representation is general: We only hypothesize that the camera has zero skew and unit pixel ratio (but unknown focal length). Most importantly, we do not make the assumption that the camera is at fixed distance from the ground plane and has a fixed view angle. Because of these properties, our algorithm can be successfully applied in both outdoors and indoors scenarios. Notice that Hedau et al. *Hedau et al.* (2009, 2010) use cues such as vanishing lines that are complementary to ours and could be nicely integrated into our framework. Also notice that physics-based constraints such as those introduced in Gupta et al *Gupta et al.* (2010) enable different ways for modeling the interaction between scene and objects wherein, in this case, objects are mostly identified as urban elements (i.e., buildings and houses). Finally, in Payet et al. *Payet and Todorovic* (2011a) the analysis of textures is introduced to provide scene-specific constraints among objects.

The main contributions of our work include: 1. A novel representation that can jointly model 3D objects locations and 3D supporting surfaces (planes) from the observations in a single image. Concretely, the problem of joint scene reconstruction and object recognition is formulated as finding a set of parameters that maximize the joint probability of having a number of detected objects on K supporting planes given the observations (Sec.2.2). 2. A relationship that connects the 2D image observation of object location and zenith angle pose with the normals of the supporting planes and with the camera focal length parameter. We prove that this relationship yields necessary conditions for estimating the camera focal length and the supporting planes' 3D orientations and locations (in the camera reference system) from the locations and zenith poses of at least 3 objects in the image. The relationship is general in that objects do not necessarily need to lie on the *same* supporting plane as long as their supporting planes are parallel with respect to each other and the objects are not collinear (Sec.2.3.1). 3. A framework that uses the above relationships and a novel probabilistic formulation to jointly detect objects (so as to reduce false alarm and

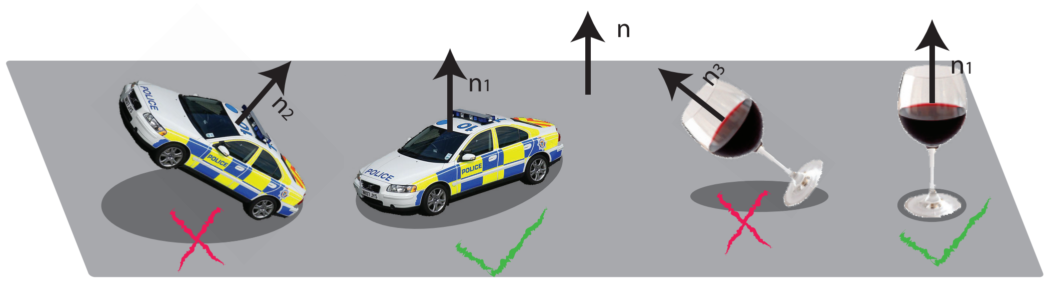


Figure 2.2: If the normal of a plane is n, objects lying on this plane tend to share the same normal direction $n_1//n$. Objects whose normal is not parallel to n (e.g. n_2 and n_3) are unlikely to sit on that supporting plane.

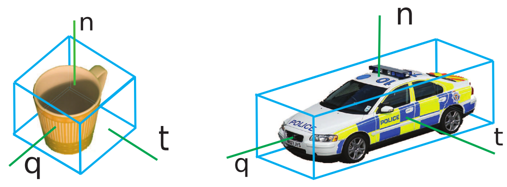
false negative rates) and recover (within the camera reference system) the objects' 3D locations, the 3D supporting planes, and the camera focal length parameter. All of the outcomes mentioned above are merely based on one single semi-calibrated image (Sec.2.2). Experimental evaluation on two datasets (desk-top dataset*Sun et al.* (2010) and the car and pedestrian Label-Me dataset*Russell et al.* (2005)) demonstrates our theoretical claims (Sec.2.4).

2.2 Modeling Scene Layout

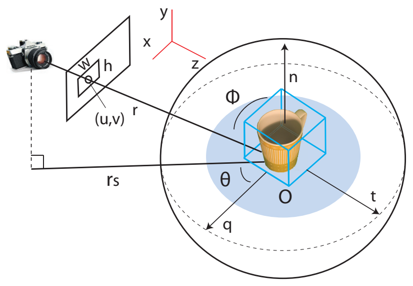
Given an image portraying a number of objects, our work proposes a new model for jointly recognizing objects in the scene and recovering the scene layout that best "explains" the evidence measured in the image. In this paper, the term "scene layout" indicates: i) the objects' 3D locations and poses in the camera reference system; ii) the 3D location and orientation of their supporting planes in the camera reference system; iii) the camera focal length. In this section we will first introduce notations and assumptions and then formulate the problem.

2.2.1 Assumptions and Notations

We assume that each object lies on a supporting plane at an up-right pose. This assumption is satisfied in most real world scenes. For example, a car is usually touch-



(a) Object in 3D



(b) Object's pose in view sphere

Figure 2.3: (a): Three perpendicular directions characterize the pose of an rigid object in a given reference system. n is defined as the object's normal. (b): Definition of zenith angle ϕ and azimuth angle θ, given the object's pose in the camera reference coordinates.

ing the ground by four wheels rather than only two and a wineglass is usually standing vertically rather than obliquely (Fig.2.2). This is consistent with the common observation that objects rarely float in the air or appear in unstable poses. Furthermore, if multiple supporting planes co-exist in one image, we assume that these planes are all parallel to each other. This parallel relationship of planes holds for most daily-life scenes. Notice that we are *not* assuming *the camera* must be "up-right" with respect to the supporting surfaces.

Plane in 3D. A plane in 3D has three degrees of freedoms. Hence, it can be parametrized by its surface normal n (Fig.2.4) and its distance h to the origin of the coordinate system (i.e. the camera).

Object in 3D. We define the parametrization to identify an object's location and pose in 3D coordinate system. Assuming that an object is enclosed by the tightest bounding cube lying on the supporting plane (Fig.2.3a), the object 3D location O can be specified by knowing the centroid of the 3D bounding box. Furthermore the object's pose can be defined by the three bounding box's perpendicular surfaces' normal n, q and t (Fig.2.3a). As discussed above, we assume all objects' up direction n should be the same as the normal of the supporting plane. Let the unit view sphere associated to an object be the collection of viewpoints equally distant from the object.

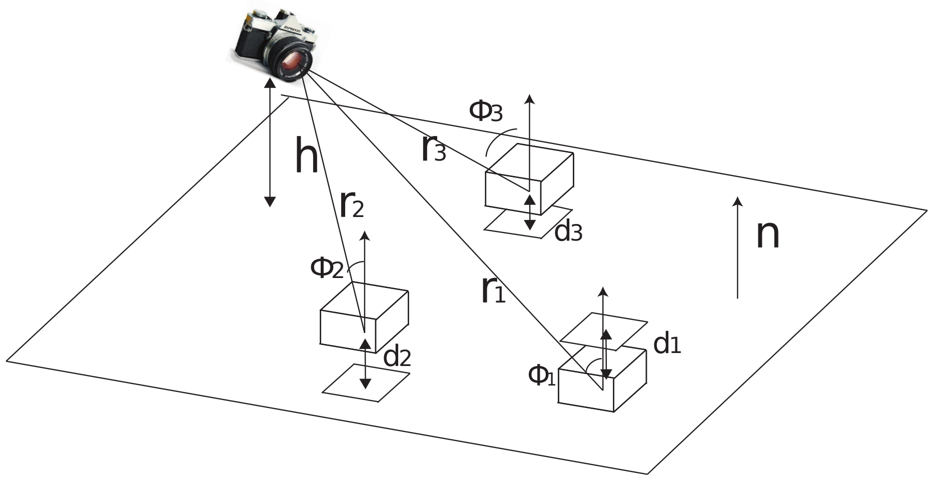


Figure 2.4: Geometric relationships of ϕ, r, d, h and n.

In the view sphere of an object, let r be the ray that connects O and the camera center (Fig.2.3b). Let *zenith* angle ϕ be the angle between the ray r and n (Fig.2.3b and Fig.2.4). Define *azimuth* angle θ be the angle formed by object's frontal vector q and a vector r_s. r_s is the projection of the ray r onto the plane perpendicular to n (i.e. supporting plane). We denote by ϕ the measured zenith pose from image, and by $\widehat{\phi}$ the estimated zenith pose consistent with the underlying surface layout. We will explain in details how to compute $\widehat{\phi}$ and measure ϕ in Sec.2.3.1.

Object in 2D. An object in the image plane is uniquely identified by a bounding box *bbox* tightly enclosing the object in 2D. We define *bbox* by its center (u, v), the height h, and width w in image coordinate (Fig.2.3b and Fig.2.7).

Candidate Detection. We assume a number of object class detectors are available and each detector returns a number of candidate detections m, where each m is defined by a bounding box *bbox* and the estimated object pose represented by the zenith angle ϕ and azimuth angle θ. Thus, $m = \{bbox, \phi, \theta\}$ (Fig.2.3b and Fig.2.7).

True-Positive Flag. We assign a true-positive flag t to each detection result. $t = 1$ if a candidate detection is associated to the true object category, and $t = 0$ if a candidate detection indicates the presence of an object from an incorrect category or just background. Given an image measurement (i.e. portion of the image that is

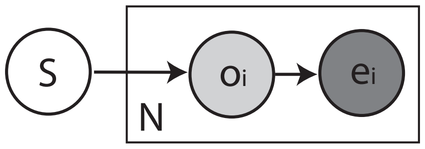


Figure 2.5: Graphical model of conditional independence for supporting plane parameters and detection result, where o_i is partially observed and e_i fully observed. Details are in Sec.2.2.2.

used by detector to assess whether an object class has been detected and may yield a detection m or not), the detector returns a confidence score indicating how likely a detection is a true positive, i.e. $t = 1$.

2.2.2 Joint Model of Objects and Supporting Planes

We propose a probabilistic model which incorporates the interaction between objects and supporting planes. The key idea is that the estimation of both candidate detections and the underlying geometry is more accurate than estimating each term independently. For simplicity, we denote scene information $S = \{n, h, f\}$ where n and h are supporting plane's parameters and f is the camera focal length. We formulate the joint probability of the candidate detections $o = \{o_i\} = \{m_i, t_i\}$, image evidence $E = \{e_i\}$, and scene information S following the graphical model in Fig.2.5 as

$$p(o, E, S) = p(S) \prod_{i=1}^{N} p(o_i|S)p(e_i|o_i)$$

$p(o_i|S)$ is the probability of an object given scene information. $p(o_i|S)$ can be further decomposed as $p(o_i|S) = p(t_i|m_i, S)p(m_i|S) \propto p(t_i|m_i, S)$ because the probability of a bounding box given only geometrical constraint $p(m_i|S)$ is a constant. Consequently,

$$p(o, E, S) \propto p(S) \prod_{i=1}^{N} p(t_i|m_i, S)p(e_i|m_i, t_i)$$

Each term is described as follows:

$\mathbf{p(S)}$ is the scene prior which can be modeled as uniform distribution within a range of n, h and f. Details of the selection of range values for these parameters are in Sec. 2.4.

$\mathbf{p(e|t, m)}$ is related to the detection result's confidence. Assume $p(m, t)$ and $p(e)$ follow a uniform distribution, we have

$$p(e|t, m) = p(t, m|e)p(e)/p(t, m) \propto p(t, m|e)$$

where $p(t, m|e)$ is the detection's confidence returned by the detector.

$\mathbf{p(t|m, S)}$ is the probability that the detection is a true positive, given the candidate detection m and the scene information S.

One contribution of our work is to estimate $p(t|m, S)$ with the help of two geometrical relationships: 1. Relationship between focal length f, zenith angle ϕ and supporting plane normal n. 2. Relationship between the object-to-plane distance d, the object's 3D coordinates O, the plane's normal n, and the camera-to-plane distance h (Fig.2.4). In Sec.2.3 we will explain in details these relationships. Here, we formulate

$$p(t = 1|m, S) \propto p(t = 1|d)p(t = 1|\phi - \hat{\phi}) \tag{2.1}$$

That is to say rather than using S directly, we use d and $\hat{\phi}$ to determine if the candidate detection m is true. We assume Gaussian distribution $p(t = 1|d) = N(d; 0, \sigma_d)$, and $p(t = 1|\phi - \hat{\phi}) = N(\phi - \hat{\phi}; 0, \sigma_\phi)$, where $\hat{\phi}$ is the inferred zenith and ϕ is the measured zenith from image. Thus, $t_i = 1$ is highly likely when the scale of the bounding box and the predicted pose by detector are consistent with the supporting plane.

To simultaneously estimate the scene information S, and the true-positive flag $\{t_i\}$ of each candidate detection, we want to find S and $\{t_i\}$ such that the joint probability $p(o, E, S)$ is maximized. Unknowns are $\{t_i\}, S$, and measurements are $\{m_i\}$ and $\{p(e_i|o_i)\}$ given by detector. The problem is equivalent to find S and $\{t_i\}$

by means of the following optimization problem:

$$\{S, \{t_i\}\} = \arg\max_{S,\{t_i\}} \ln p(S) + \sum_{i=1}^{N} [\ln p(t_i|m_i, S) + \ln p(e_i|t_i, m_i)] \tag{2.2}$$

2.2.3 Solving the Optimization

In this section we solve the optimization problem of Eq.(2.2) in Sec.2.2.2. Define $z(S)$ as

$$\begin{aligned} z(S) &= \max_{\{t_i\}} \sum_{i=1}^{N} [\ln p(t_i|m_i, S) + \ln p(e_i|t_i, m_i)] \\ &= \sum_{i=1}^{N} \left\{ \max_{t_i} [\ln p(t_i|m_i, S) + \ln p(e_i|t_i, m_i)] \right\} \end{aligned}$$

For a fixed value of S, the value of each term in the above summation can be calculated independently. Hence, the best global configuration of $\{t_i\}$ given S can be found after N comparisons of $\ln p(t_i = 1|m_i, S) + \ln p(e_i|t_i = 1, m_i)$ with $\ln p(t_i = 0|m_i, S) + \ln p(e_i|t_i = 0, m_i)$. Therefore, $\{t_i\}$ can be computed as a function of S:

$$t_i^* = \arg\max_{\mathbf{t}_i} \ln p(t_i|m_i, S) + \ln p(e_i|t_i, m_i) \tag{2.3}$$

Having estimated t_i, to find S is equivalent to

$$S = \arg\max_{S} [\ln p(S) + z(S)] \tag{2.4}$$

We propose to solve Eq.(2.4) by searching on a large but finite set $\mathbf{S}$ to find the optimal S^*. This can be computed almost in real-time by CUDA parallel programming. Let $\mathbf{F} \in R$ be the set of all the possible values of the focal length f. Let $\mathbf{N} \in R^3$ be the set of all possible values of the plane normal n. Let $\mathbf{H}$ be the set of all possible values of the plane height h. The search space is $\mathbf{S} = \mathbf{F} \times \mathbf{N} \times \mathbf{H}$. The

Algorithm II.1 Estimating scene layout from images

```
1. Set the number of already estimated planes $K = 0$. Set the active object detection set $A$ to be the set of all object detection candidates.
2. If $K = 0$, enumerate $S^j \in \mathbf{F} \times \mathbf{N} \times \mathbf{H}$; else enumerate $S^j \in f \times n \times \mathbf{H}$ where $f$ and $n$ are the already estimated focal length and plane normal.
3. For each enumeration $S^j$, estimate the flag $t_i^j$ for all objects $o_i \in A$ by Eq. (2.3)
4. Given the estimated $\{t_i^j\}$ for all enumerations $\{S^j\}$, find $S^* \in \{S^j\}$ by Eq. (2.4).
5. Take $S^*$ as one estimated supporting surface. Remove the objects that have true flag $t_i^* = 1$ from $A$. Set $K = K + 1$.
6. if $K$ is larger than a predefined threshold, then stop. Otherwise goto 2.
```

details can be found in Algorithm (II.1).

2.2.4 Extension to Multiple Planes

The above approach estimates the single most likely supporting plane by obtaining the highest log likelihood score. This approach can be extended to handle the case of multiple supporting planes by using an iterative procedure. Denote by K the number of already estimated planes. Denote by A the set of active object detection candidates. At the begining, $K = 0$ and A is all object detection candidates. First, we employ this approach to find the best plane configuration S. Then we determine the objects that sits on the plane S and remove them from A. Next, the algorithm processes the remaining detection candidates. The algorithm ceases after K is larger than a predefined threshold. Notice that, since all the supporting surfaces are assumed to have the same normals, the "at least three objects" requirement (Sec.2.3.1) is no longer necessary for other planes except the first one. The procedure is described in Algorithm (II.1)

2.3 Relating Camera Measurements and Supporting Planes

In this section we derive the relationship among the estimated zenith angle pose ϕ_i of an object in the image plane, the supporting plane normal n and the camera focal length f. We show that by measuring ϕ_i for at least three non-collinear objects, we can estimate f and n from a single image. Notice that in order for this result to be true, objects are not necessarily required to lie on a single supporting plane, but each object can have its own supporting plane as long as all the planes are parallel. This result is one of the main contributions of our paper and provides sufficient conditions for estimating $p(t_i|m_i, S)$. In Sec.2.3.2, we will explain how to locate an object O in 3D and establish a relationship between O, h, d and n.

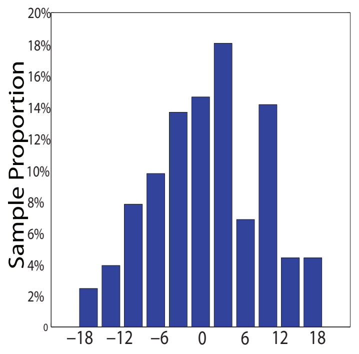
2.3.1 Relationship Between Focal Length and Supporting Plane Normal

We adopt homogeneous coordinates to represent objects in 3D and in the image plane coordinates. Let $(\widetilde{u}, \widetilde{v}, 1)$ be the homogeneous coordinates of the object projection in the image plane. We assume that the camera is semi-calibrated. That is, we assume that the camera center (u_0, v_0) is known, the pixel ratio $\alpha = 1$ and the camera has zero-skew. These are reasonable assumptions that hold in most practical cases. By translating any point in the image plane by $(u_i, v_i) = ((\widetilde{u}_i, \widetilde{v}_i) - (u_0, v_0))^T$, we write the camera intrinsic parameter matrix as $K = diag(f, f, 1)$.

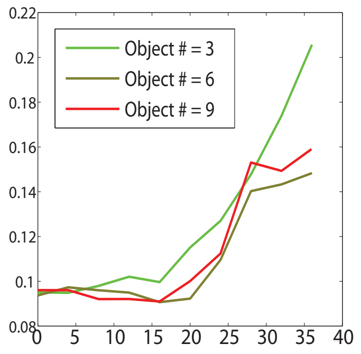
Let r_i be the line of sight connecting the camera center and an object O_i, which passes through an object's location (u_i, v_i, f) in the image. Then the direction of the line of sight r_i in camera coordinates is $(u_i/f, v_i/f, 1)$. Let $n = (n_1, n_2, n_3)$ denote the normal of the supporting plane in camera coordinates. s_i and n are shown is Fig. 2.4. If we enforce n to have unit norm, then $n_1^2 + n_2^2 + n_3^2 = 1$. Thus:

$$(u_i, v_i, 1) \begin{pmatrix} n_1 \\ n_2 \\ n_3 f \end{pmatrix} = -\cos\phi_i \sqrt{u_1^2 + v_1^2 + f^2} \tag{2.5}$$

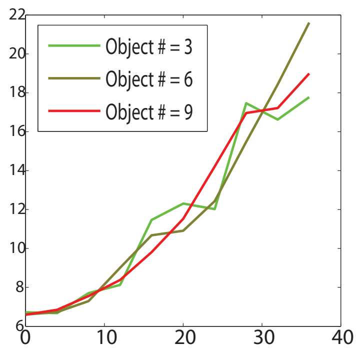
Using Eq.(2.5), the key term $\widehat{\phi}$ in Eq.(2.1) can be computed given n_1, n_2, n_3, and f, i.e. part of S.



(a) zenith angle



(b) Effect of noise in solving Eq. 2.6



(c) Effect of noise in solving Eq. 2.6

Figure 2.6: (a) Histogram of the actual error of the measurement of object zenith angle ϕ. The Y axis is the fraction of testing samples that fall in each error bin. The X axis are error bins in degree. The standard deviation of zenith angle measurement is 8.4°. (b)(c) Error analysis of Eq. 2.6. X axis is the variance of Gaussian noise in degree. (b) Y axis is $e_f = |(f - \widehat{f})/f|$. (c) Y axis is $e_n = |arccos(n \cdot \widehat{n})|$ in degree. This figure is best viewed in color.

2.3.1.1 Measuring Zenith Angle From Single Image

It is clear that our formulation relies on the measurement of the objects' zenith angles in the image plane. Recently, a number of techniques such as *Liebelt et al.* (2008); *Savarese and Fei-Fei* (2007); **?**); *Su et al.* (2009) have been proposed to estimate object pose from single images. We used an adapted version of *Su et al.* (2009) to measure zenith angles ϕ from the image. Quantitative experimental analysis on our in-house dataset shows that our detector is capable of generating zenith pose classification results that are compatible with our sensitivity analysis (Sec. 2.3.1.3 and Fig.2.6).

2.3.1.2 Estimating 3D Plane Orientation via Object Zenith Angles

In this section, we show that the normal of the supporting planes and the focal length of the camera can be estimated from the objects' zenith angles phi_i and their locations from just one single image. If a total number of N measurements ϕ_i, u_i, v_i

$(i = 1...N)$ are available, following Eq.(2.5) we obtain:

$$\begin{bmatrix} u_1 & v_1 & f \\ u_2 & v_2 & f \\ u_3 & v_3 & f \\ & \vdots & \\ u_N & v_N & f \end{bmatrix} \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} = \begin{pmatrix} -\cos\phi_1\sqrt{u_1^2+v_1^2+f^2} \\ -\cos\phi_2\sqrt{u_2^2+v_2^2+f^2} \\ -\cos\phi_3\sqrt{u_3^2+v_3^2+f^2} \\ \vdots \\ -\cos\phi_N\sqrt{u_N^2+v_N^2+f^2} \end{pmatrix} \tag{2.6}$$

This equation allows us to solve $\{f, n_1, n_2, n_3\}$ from the objects' measurements ϕ_i, u_i, v_i (i=1. . . N) in just one single image. The following proposition gives the conditions for the existence of a solution of Eq.(2.6).

Proposition 1. Eq. (2.6) admits one or at most two non-trivial solutions for $\{f, n_1, n_2, n_3\}$ if at least three non-aligned observations (u_i, v_i) (i.e. non-collinear in the image) are available. If the observations are collinear, then Eq.(2.6) has an infinite number of solutions.

Proof. Suppose at least three objects are not collinear in an image, then the rank of the left matrix on the left-hand side of Eq. (2.6) is 3. Therefore Eq. (2.6) provides 3 independent constraints. The unknowns in Eq.(2.6) are n_1, n_2, n_3, f. With these constraints, each of n_1, n_2, n_3 can be expressed as a function of f, i.e. $n_i = n_i(f)$. Because $\|n\| = 1$, we obtain an equation about f:

$$\sum_{i=1...3} n_i^2(f) = 1$$

In the above equation, f appears in the form of f^2 and f^4. Therefore, there are at most two real positive solutions for f. Given f, $\{n_1, n_2, n_3\}$ can be computed as $n_i = n_i(f)$.

If all objects are collinear in the image, then an infinite number of solutions exist for Eq.(2.6). If all objects are collinear, the rank of the left matrix in the left-hand side of Eq.(2.6) is 2. Without loss of generality, assume $(u_1, v_1) \neq 0$. In such a case, after using Gaussian elimination, Eq.(2.6) will be in the following form:

$$\begin{bmatrix} \alpha & \beta & f \\ \gamma & \epsilon & 0 \\ 0 & 0 & 0 \\ & \vdots & \end{bmatrix} \begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} = \begin{pmatrix} \zeta \\ \eta \\ 0 \\ \vdots \end{pmatrix} \tag{2.7}$$

If $\widehat{f}, \widehat{n1}, \widehat{n2}, \widehat{n3}$ is a solution, then $\widehat{f}, \widehat{n1} + km_1, \widehat{n2} + km_2, \widehat{n3} + km_3$ is also a solution of Eq. 2.7, where (m_1, m_2, m_3) is the non-trivial solution the following equation:

$$\begin{bmatrix} \alpha & \beta & f \\ \gamma & \epsilon & 0 \end{bmatrix} \begin{pmatrix} m_1 \\ m_2 \\ m_3 \end{pmatrix} = 0$$

Hence, Eq. (2.6) admits an infinite number of solutions.

Eq. (2.6) guarantees that as long as at least 3 objects do not lie on the same line in the image, it is possible to express the focal length of the camera and the normal of the supporting planes as a function of the objects' locations and zenith pose measurements in the image. Notice that this equation does not assume that all objects are placed on one unique plane and it also does not make the assumption that the camera has no in-plane-rotation (tilt).

2.3.1.3 Error Analysis

We use a numerical simulation to analyze the robustness of the estimation of f and n in solving Eq.(2.6) as a function of noise in the measurements ϕ. For a total number N of objects, first a random set of object's bounding box $\{u_i, v_i\}$, plane's normal n and focal length f are synthetically generated. Then the corresponding object's zenith angle ϕ_i is computed by Eq.(2.5). Next we add Gaussian noise w of variance σ to the object's zenith $\widetilde{\phi_i} = \phi_i + w$. Consequently, given $\{\widetilde{\phi_i}\}$ and $\{u_i, v_i\}$,

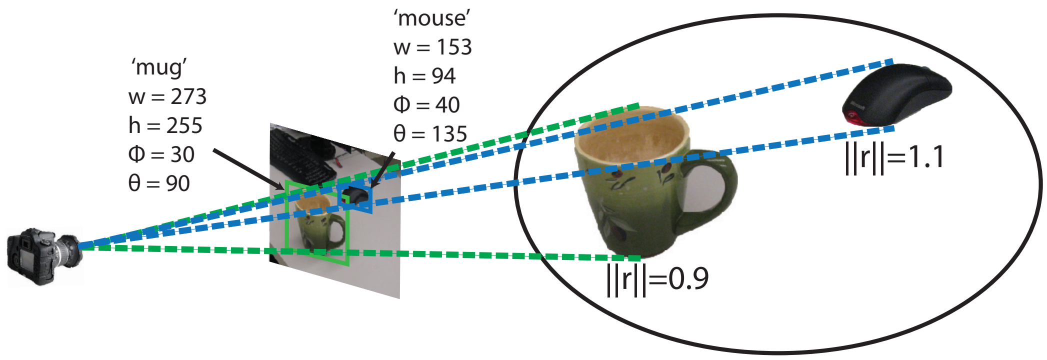


Figure 2.7: An illustration of a detected object and its corresponding 3D pose. Given object's image bounding box and estimated pose, its distance to the camera can be estimated using the procedure in Sec.2.3.2.

we compute the normal of the plane $\widehat{n}$ and the focal length $\widehat{f}$, by solving Eq.(2.6) using the Levenberg-Marquardt method. Fig.2.6b and Fig.2.6c show the mean value of the absolute errors v.s. the number of objects and the noise level (see figure captions for details). These plots relate the accuracy in estimating n and f as a function of the error in measuring the zenith angle ϕ. Given that our detector returns ϕ with an error of about $10°$ (Fig.2.6a), Fig.2.6b and Fig.2.6c show that the corresponding error in estimating n and f is reasonably low.

2.3.2 Locating Objects in 3D

In this section, we explain the relationship between S and d and how to locate objects in the 3D camera reference system. Denote by $\|r\|$ the distance between the object location O and the camera. It is impossible to estimate $\|r\|$ without any prior knowledge about the camera or the object if only a single image is available. However, assuming that we have some prior knowledge about the real size of the 3D object, the object distance $\|r\|$ can be estimated from the object scale in the image by means of an inversely proportional relationship. Specifically, if an object's image bounding box's height and width are h and w, its category is c, and its estimated pose is θ and

ϕ, we approximate its distance $\|r\|$ by the following linear combination in $\frac{1}{w}$ and $\frac{1}{h}$

$$\|r\| \simeq (\alpha(\theta,\phi,c)\frac{1}{w} + \beta(\theta,\phi,c)\frac{1}{h}) \cdot f \tag{2.8}$$

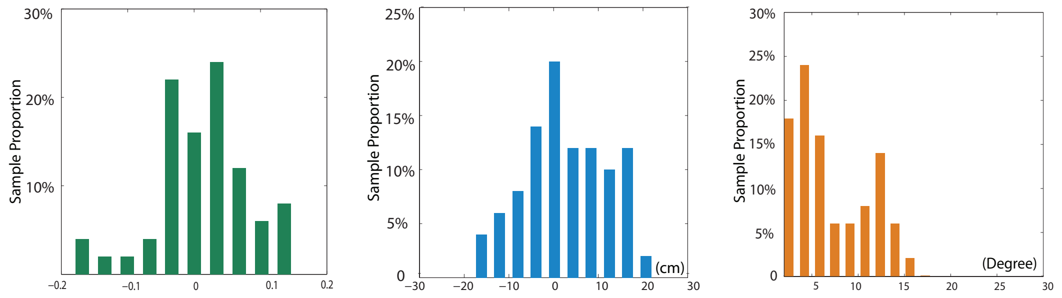
where α and β are function of the object's pose and class label and f is the focal length. α and β are related to the physical 3D shape of the object category. A more precise modeling of this relationship goes beyond the scope of this paper. We instead use linear regression to learn α and β for each set of θ, ϕ, c in the training set where ground truth pose and distance $\|r\|$ are available (Fig.2.7). As a result, given candidate object $m = \{bbox, \theta, \phi\}$ and its category c, its 3D coordinates can be estimated in the camera coordinates as follows:

$$O \backsimeq \frac{\|r\|}{\sqrt{(u/f)^2 + (v/f)^2 + 1}} \begin{pmatrix} u/f \\ v/f \\ 1 \end{pmatrix}$$

This allows us to relate the 3D coordinates of candidate object O, the supporting plane parameters (n, h), and the distance d between object and the supporting plane as $d = O^T n + h$ (Fig.2.4).

2.4 Evaluation

In this section we qualitatively demonstrate the ability of our framework to jointly estimate the scene layout (camera location, supporting plane orientation and object location in the 3D space) as well as improve the accuracy in detecting objects. We test our algorithm on a novel indoor desk-top database *Sun et al.* (2010) as well as on the LabelMe *Russell et al.* (2005) outdoor pedestrian and cars dataset. We use the Graphic Processor Unit to implement the optimization procedure. In our implementation of the optimization function, the range of values for each unknown



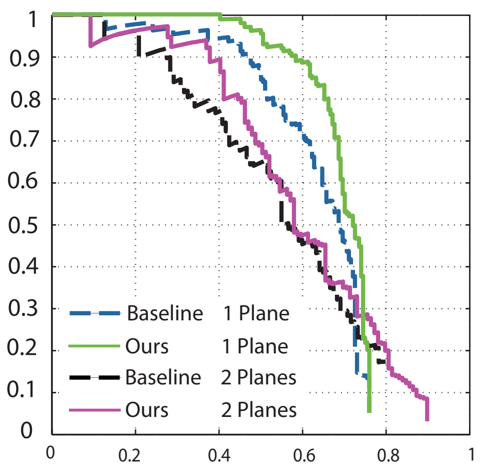
(a) Focal Length Est. Error (b) Height Est. Error (c) Plane Normal Est. Error

Figure 2.8: Experimental results on our desk-top dataset. Y axis is the proportion of test images associated to an error interval (X axis). (a) shows the error when estimating the focal length on 50 test images: the ground-truth focal length f^i_{gt} is known and the f^i_{est} is the estimated value. The error is computed as $e^i_f = (f^i_{est} - f^i_{gt})/f^i_{gt}$. (b) is the error when estimating the camera height on 50 test images. The ground truth value of camera height h^i_{gt} ranges from 35cm to 60cm, and the estimated value is h^i_{est}. The error is computed as $e^i_h = h^i_{est} - h^i_{gt}$. (c) shows the error when estimating the plane normal on 50 test images. The ground truth normal is n^i_{gt} and the estimated value is n^i_{est}. The error is defined as $e^i_n = arccos(n^i_{est} n^i_{gt})$.

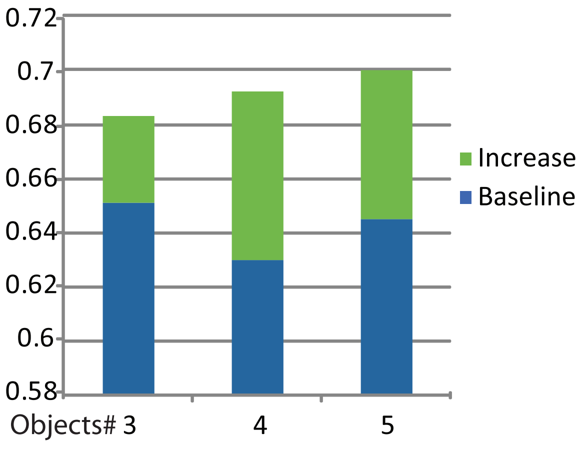
parameter is set as follows: i) plane normal has 20 discretized values for tilt direction from 15° to 75° and 5 discretized values for camera-rotation from −10° to 10°, ii) plane height has 20 discretized values from $30cm$ to $80cm$ for office dataset and from $1.5m$ to $2m$ for street dataset. iii) camera focal length has 20 discretized values from 0.8 to 1.25 fraction of the initial value of the camera focal length. The average optimization time for one 640×480 image is 0.2 seconds. Using the LabelMe dataset, we compare our algorithm with Hoiem et al. *Hoiem et al.* (2006). The comparison indicates that our method achieves competitive results in pruning out false positives and estimating layout properties such as the horizon line. We also show successful anecdotal results on a number of images downloaded from the web.

2.4.1 Desk-Top Scenario

We test our framework on a novel desk-top database *Sun et al.* (2010) where ground truth information about the geometry of the scene is available. This dataset comprises three object categories (computer mouse, mug and stapler). Each image



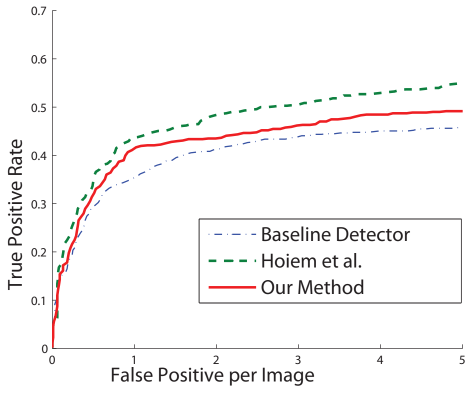
(a) Precision-recall curve



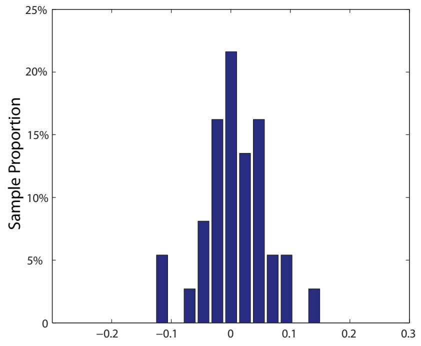
(b) AP v.s. number of objects

Figure 2.9: Experimental results on our desk-top dataset. (a) reports precision-recall curves by the base line detector (dash) and our algorithm (solid). Precision-recall curves are shown for one and two planes separately. (b) reports the average precision as the number of objects increases.

in the dataset portrays 3 to 5 object instances located at randomly selected positions and with random poses on one (or two) supporting plane(s) (Fig.2.11). Training and testing sets contain 80 and 75 images respectively. For each image we have the available ground truth values for the camera focal length and the normal of the supporting plane in the camera reference system as well as the ground truth locations of the objects in the image. These are used for training the distance function (Eq.(2.8)) and for evaluating our algorithm's performance. We learn our modified version of the object detector and pose estimator in *Su et al.* (2009) on the 3-object category training set. We apply the learnt detector to the testing set and obtain a number of detected objects. This provides the baseline object detection result (e.g. "baseline" in Fig. 2.9a and 2.9b). For each detection we also estimate the azimuth and zenith pose of the object. Examples of detections are in Fig.2.11. Among these detections we can find a number of false alarms. So we run our algorithm and use such detections (along with pose measurements) to jointly estimate the supporting plane normal, the camera focal length and the locations of the objects (among all



(a) Detection ROC curve



(b) Horizon line error.

Figure 2.10: Result on LabelMe dataset. (a) Car and Pedestrian detection. (b) The histogram of the horizontal vanishing line estimation error. The Y axis is the fraction of the number of testing images (samples) that have certain error.

detections returned by the detector) that are consistent with the estimated scene layout. Results are shown in Fig.2.8 and 2.9. We tested our algorithm on images where one plane or two planes exist in the scene. Our testing set contains 50 images of one-plane case and 25 images of two-planes case. Fig. 2.9a shows the object detection precision-recall curve. In the one-plane case, the baseline detector average precision is 64% compared to 70% with our method. In the two-planes case, the baseline detector average precision is 56% compared to ours 61%. Furthermore, we evaluate the detection accuracy as function of the number of instances appearing in the scene per test image. We show our results in Fig. 2.9b. The object detection performance improvement is obtained by using the estimated supporting plane to prune out false alarms and recover missed positives. The estimation of the supporting plane is affected by the observation noise (location and pose) associated to each object instance. As the number of observations increases, the contribution of the noise is averaged out which explains the reason the object detection performance increases with the number of instances.

2.4.2 Experiments on LabelMe Dataset

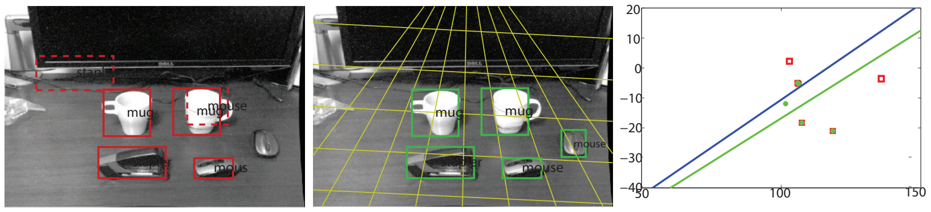
We compare our algorithm with another state-of-the-art method that uses geometrical contextual reasoning for improving object detection rates and estimating scene geometric properties such as the horizon line *Hoiem et al.* (2006). We use the LabelMe database on cars and pedestrians to compare the algorithms. Since one necessary condition for our algorithm to work is that at least three objects coexist in the same image, we use a subset of the dataset provided by *Hoiem et al.* (2006). We remove images containing less than three instances (pedestrians or cars). We test our algorithm on 100 randomly selected images and compare our method with *Hoiem et al.* (2006) by using the same baseline pedestrian and car detector as in *Hoiem et al.* (2006). Examples of detections are in Fig.2.12. Fig.2.10a compares the ROC curve for car and pedestrian detection by our algorithm to that of *Hoiem et al.* (2006). Fig.2.10b shows the histogram of the relative error of our algorithm in estimating the horizontal vanishing line. Notice the median absolute error in estimating the horizontal vanishing line reported in *Hoiem et al.* (2006) is 0.038. Detection rate and accuracy in estimating the horizon line are comparable between ours and *Hoiem et al.* (2006). However, notice that *Hoiem et al.* (2006) heavily relies on: i) estimating surface geometry *Hoiem et al.* (2005) by determining ”ground”, ”vertical” and ”sky” regions in the image; ii) assuming that the camera has a fixed distance from the ground plane (the distance is roughly the height of a person); iii) assuming that no multiple ground planes (at different heights) are present in the image. On the contrary, our algorithm: i) does not rely on estimating horizontal or vertical regions as it extracts spatial contextual information from the objects themselves (thus, our algorithm works even if the ground region is not visible at all); ii) does not assume fixed distance from the ground plane which can be located anywhere in the 3D space; iii) it works even if objects are supported by multiple planes located at different heights. For that reason our algorithm is particularly suitable to work in indoor settings where most of the assumptions of *Hoiem et al.* (2006) are violated.

2.4.3 Anecdotal Detections and Reconstructions

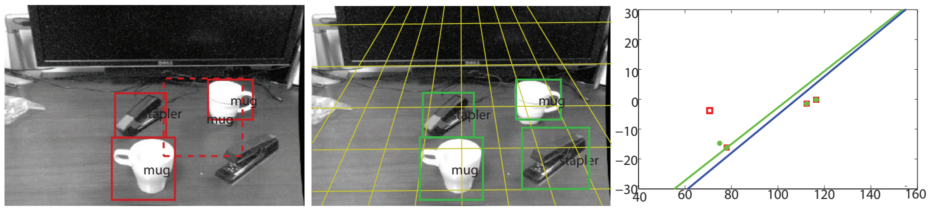
We conclude this section by presenting a number of anecdotal examples. Fig.2.13 shows joint detection and scene layout estimation on images taken from various sources including ETHZ *Ferrari et al.* (2008) and the Internet.

2.5 Conclusions

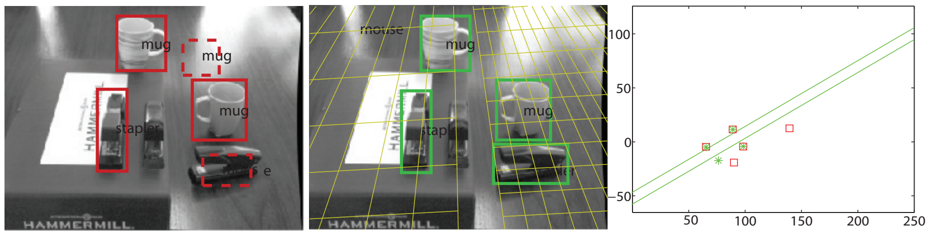
We have presented a novel scene understanding method that can jointly model object locations and supporting surfaces (planes) in the 3D space along with camera focal length in a single camera. We have modeled the problem of joint scene reconstruction and object recognition as the one of finding the set of parameters that maximizes the joint probability of detecting objects on several supporting planes. Experimental results have demonstrated the validity of our intuitions and assumptions.



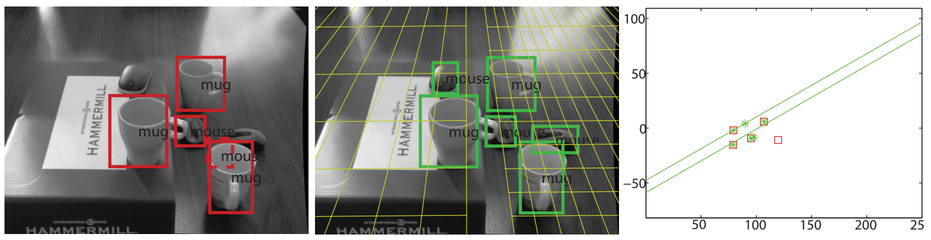
(a) One Plane Example



(b) One Plane Example

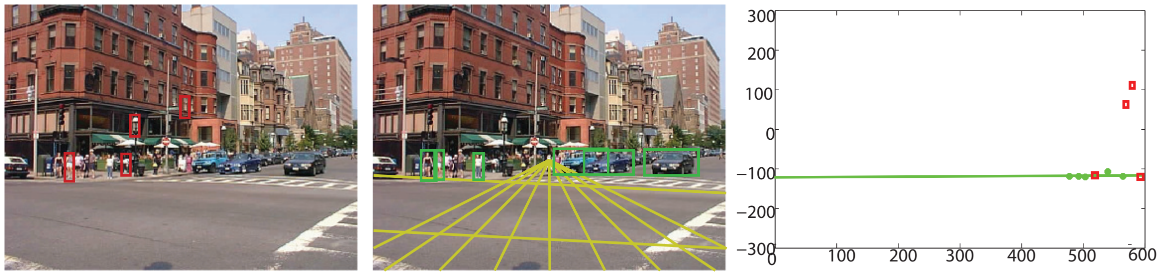


(c) two Plane Example



(d) Two Plane Example

Figure 2.11: Desk-top Dataset: In each sub-figure we show the baseline detector results on the left; our algorithm's object detection and support plane estimation results in the middle; our algorithm's 3D scene layout reconstruction on the right. Baseline detection results are in red; dashed red boxes indicate false alarms. Our improved detection results are in green; dashed green boxes indicate false alarms. Our estimated supporting plane is superimposed in yellow. Notice that most of the supporting planes estimations are visually convincing. The 3D layout shows the side view of the 3D reconstructed scene (the camera is located at $(0, 0)$ pointing to the right). The estimated supporting plane is in green and the ground truth supporting plane in blue. Green dots are the objects detected and recovered by our algorithm (in the 3D camera reference system); red squares are objects detected by the baseline detector. Notice that our algorithm works even when there are multiple supporting planes existing in a scene.


(a) Example in LabelMe Dataset

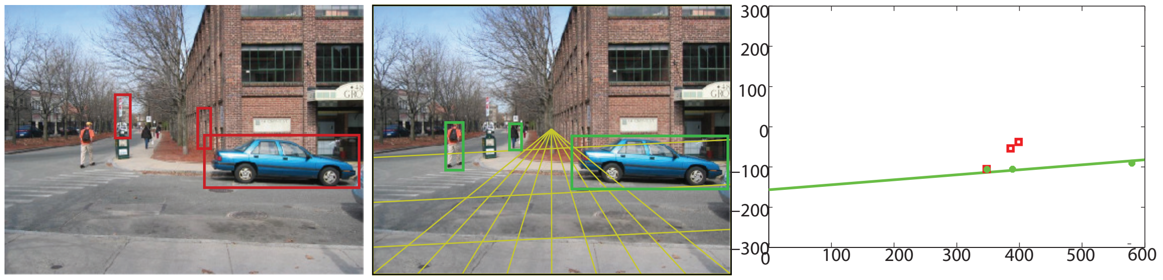
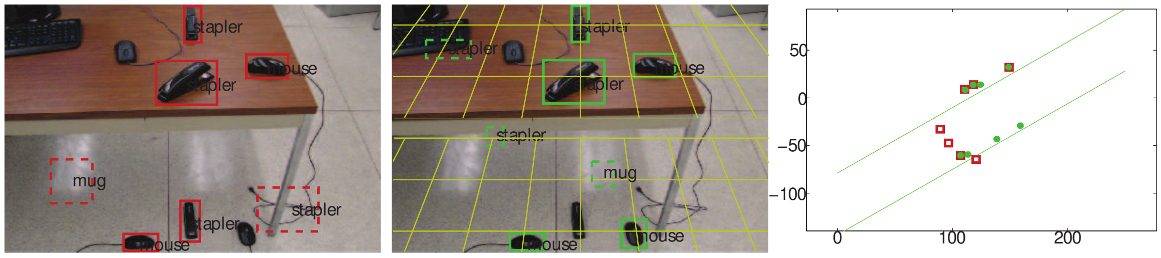
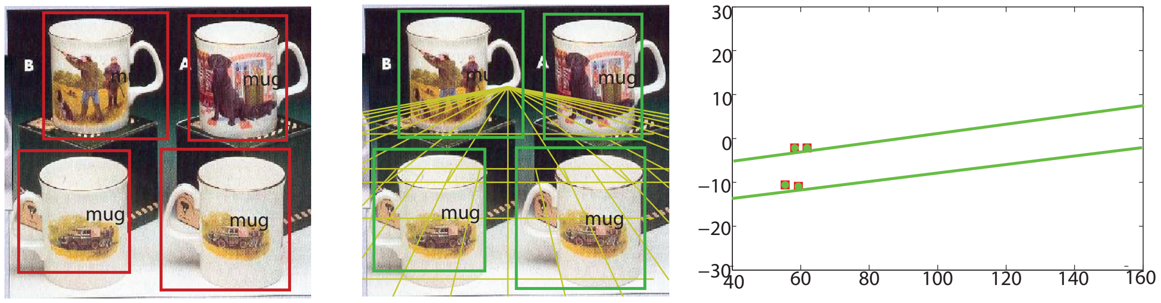

(b) Example in LabelMe Dataset

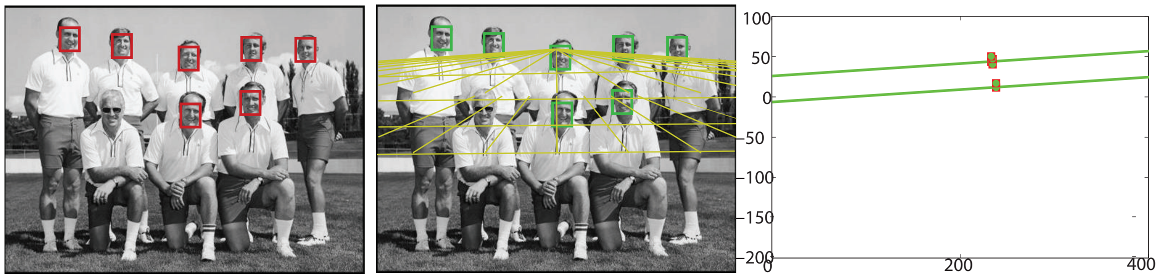
Figure 2.12: LabelMe Dataset: Please refer to the caption of Fig.2.11 for the meaning of the symbols.



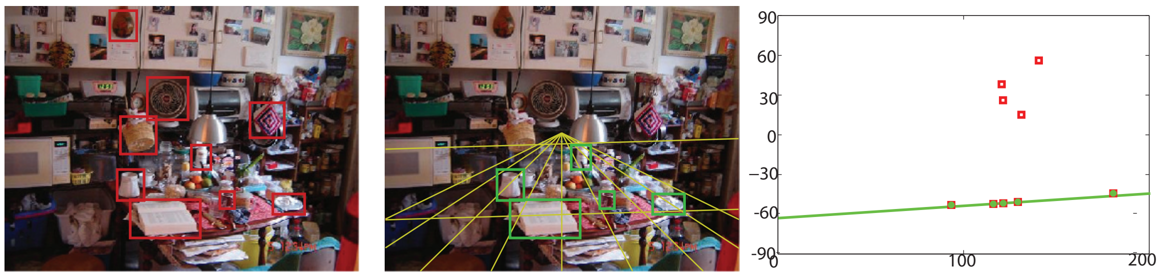
(a) Example of Internet Image



(b) Example in ETHZ Dataset



(c) Example using Face Detector on Internet Image



(d) Example of Internet Image (detections are manually identified)

Figure 2.13: Anecdotal scenarios: Please refer to the caption of Fig.2.11 for the meaning of the figure notations. In Fig. 2.13c, we use a detector to detect faces and use these (along with the fact that faces are estimated frontally) to estimate different hypothetical supporting planes. In Fig. 2.13d, we show that our algorithm can potentially recover the supporting plane and perform contextual reasoning even when the scene is highly cluttered (here detections in red were manually identified, but successfully pruned out by our algorithm in green). This figure is best viewed in color.

CHAPTER III

Scene Understanding From Multiple Images [1]

3.1 Introduction

Understanding the 3D spatial and semantic structure of a complex scene from images is a critical capability of an intelligent visual system. Consider the photographs in Fig. 3.1(a). These show the same environment observed from a handful of viewpoints. For a human observer, it is not difficult to infer: i) the spatial structure of the scene and the arrangement of objects in the 3D physical space; ii) the semantic content of the scene and its individual components. Methods for object recognition *Felzenszwalb et al.* (2010); *Leibe et al.* (2004); *Fergus et al.* (2003); *Lazebnik et al.* (2006) typically describe the scene with a list of class labels (e.g., a chair, a monitor, etc...) along with their 2D location and scale, but are unable to account for their configuration in 3D space (Fig. 3.1(b)). Algorithms for image segmentation *Ren and Malik* (2003); *Ladicky et al.* (2010); *Varma and Garg* (2007) identify regions of interests in the image and associate semantic labels to them (e.g. walls, desks, etc), but are unable to estimate their 3D spatial structure (Fig. 3.1(c)). On the other hand, 3D reconstruction methods (e.g. those based on SFM) *Pollefeys and Gool* (2002); *Dick et al.* (2004); *Snavely et al.* (2008); *Nister* (2004); *Soatto and Perona* (1998); *Agarwal*

[1]This chapter was partially previous published in *Bao and Savarese* (2011a); *Bao et al.* (2011a, 2012a); *Bao and Savarese* (2013).

et al. (2009) are capable of recovering the scene 3D structure (3D point clouds) but are mostly unable to infer the semantic content of its components (Fig. 3.1(d)).

In this chapter we seek to fill this representation gap. We propose a novel framework, called *semantic structure from motion* (SSFM), to jointly estimate the structure of the scene and the pose of cameras from a few uncalibrated (extrinsic parameters are unknown, but internal parameters are given) images (Fig. 3.1 (e)). By estimating the "structure of the scene" we refer to the ability to estimate the location of points, as well as location and pose of objects (e.g. monitors and chairs) and regions (e.g. desks and walls) in the scene. We refer to a point, a region, or an object as a *scene component*. The key concept we explore in this work is that measurements of scene components from images are semantically and geometrically consistent across views. For instance, in Fig. 3.1(a), the chair appears in two views and its location, scale and pose variation across the two views must be consistent with the view point transformation. Similarly, as Fig. 3.1(a) shows, the desk surface's location, scale and orientation variation across the two views should be consistent with the view point transformation.

In addition to considering scene components in isolation, we propose to take advantage of the *interactions* among points, objects, and regions. An *interaction* models the relationship between pairs of scene components in terms of location, pose, and semantics. For example, a bottle typically sits on a surface in an up-right pose. Interactions between scene components help regularize the solution and enhance accuracy and robustness in estimating their geometry and semantic properties. We summarize the interactions in Tab. 3.1.

We formulate the relationships between scene components and measurements using a factor graph representation. In such representation, each node captures different scene components or cameras. Each factor node captures the interaction between scene components or the relationship between scene components and their image

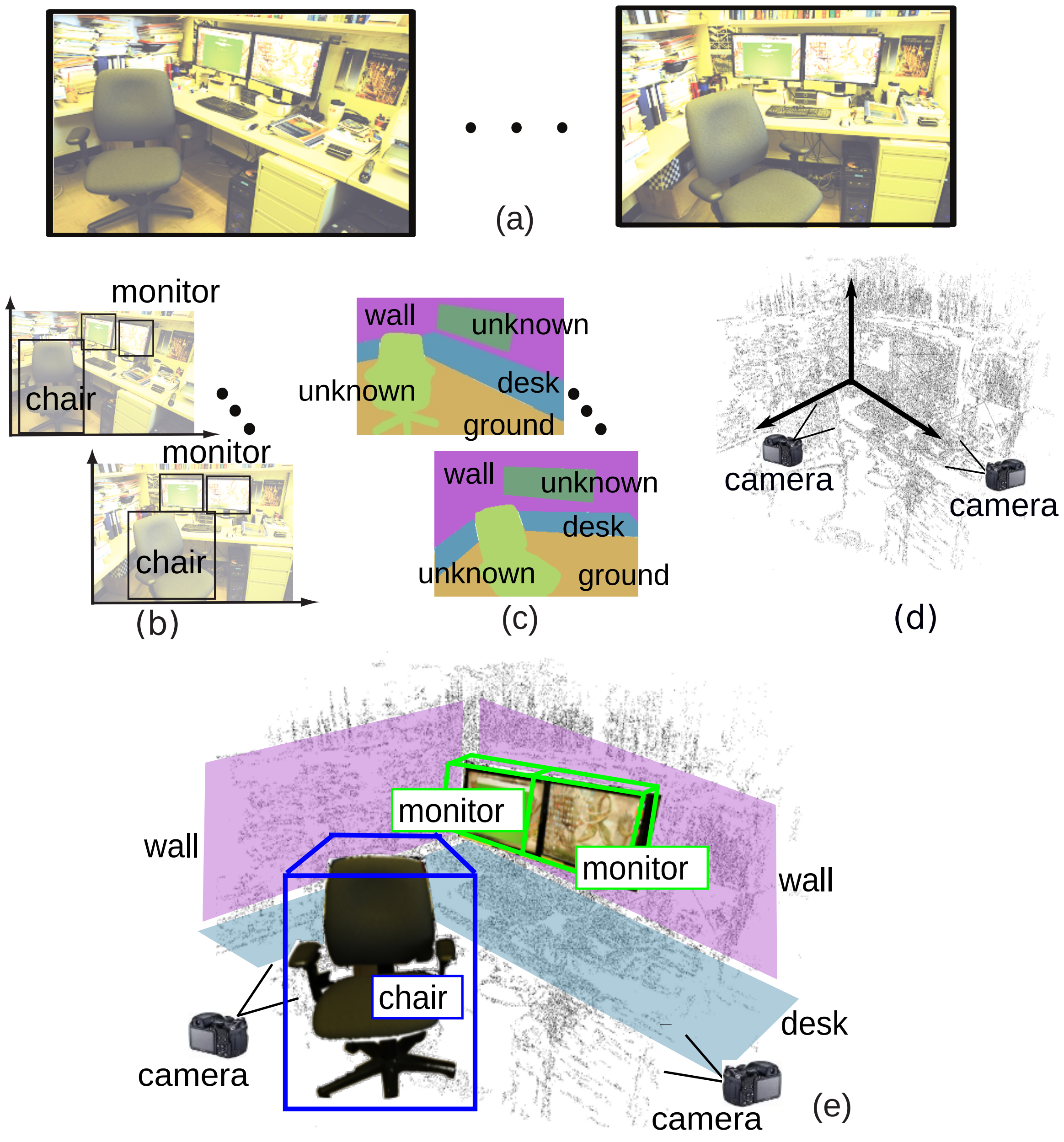


Figure 3.1: Main objective of our proposed semantic structure from motion (SSFM) framework. (a) Input photos showing the same environment observed from a handful of viewpoints. (b) Conventional object recognition algorithms identify objects in 2D without reasoning about the 3D geometry. (c) Conventional segmentation algorithms estimate region labels without determining their 3D configuration. (d) SFM returns 3D scene reconstruction (3D point clouds) with no semantic information attached to it. (e) SSFM aims to jointly recognize objects and reconstruct the underlying 3D geometry of the scene (cameras, points and objects).

Type	Across views	Single view	Pointers
Object-Point	X		Sec. 3.6.1. Fig. 3.9
Object-Region		X	Sec. 3.6.2. Fig. 3.12
Point-Region	X	X	Sec. 3.6.3. Fig. 3.13

Table 3.1: Types and properties of interactions.

measurements. This allows to formulate the problem using a novel energy-based framework and solve the semantic structure from motion problem in a coherent and principal fashion.

Our proposed method has the merit of enhancing both 3D reconstruction and visual recognition capabilities. **Enhancing 3D reconstruction**: our framework can help overcome a crucial limitation of scene / object modeling methods. State-of-the-art SFM techniques (e.g. *Pollefeys and Gool* (2002); *Dick et al.* (2004); *Snavely et al.* (2008); *Nister* (2004); *Soatto and Perona* (1998)) mostly fail when dealing with challenging camera configurations (e.g., when the views are too few and the view baseline is too large). This failure occurs as it is hard to establish correct feature correspondences for widely separated views. For instance, the 3D reconstruction in Fig. 3.1(d) was obtained using a state-of-the-art SFM algorithm *Golparvar-Fard et al.* (2009); *Snavely et al.* (2008) using 43 densely-sampled pictures of an office. The same algorithm would likely not work if we just used the two images in Fig. 3.1(a) for the reasons mentioned above. By reasoning at the semantic level, and by establishing object and region correspondences across views, our framework creates the conditions for overcoming this limitation. We show that our method can significantly outperform across-view feature matching SFM algorithms such as *Snavely et al.* (2008); *Lowe* (2004) (Tab. 3.10). Moreover, we show that our framework has the ability to estimate camera poses from object detections only. **Enhancing visual recognition**: conventional object recognition and region segmentation methods (e.g. *Ren and Malik* (2003); *Ladicky et al.* (2010); *Varma and Garg* (2007)) are typically prone to produce false alarms when appearance cues are not discriminative enough and no contextual information about the scene is available. For instance, the cabinet in Fig. 3.1(a) can be easily confused with a monitor as they both share similar appearance characteristics. By reasoning at the geometrical level, our framework is able to identify those hypotheses that are not consistent with the underlying geometry and reduce their confidence score accordingly. Our model leads to promising experimental results showing improvements in object detection accuracy and region classification compared with the state-of-the-art methods such as *Felzenszwalb et al.* (2010); *Ladicky et al.* (2010); *Varma and Garg* (2007).

3.2 Related Works

A number of recent works address the problem of joint geometry estimation and semantic understanding. Most of these works (e.g. *Gupta et al.* (2010); *Hoiem et al.* (2008a); *Gould et al.* (2009); *Payet and Todorovic* (2011b); *Lee et al.* (2009); *Hedau et al.* (2010)) either assume that only a single image is available, or make restrictive hypotheses on the camera and scene configuration. Ladicky et al. *Ladicky et al.* (2011) show promising results on joint reconstruction and segmentation from stereo pairs, but assume that cameras are calibrated with small baselines. Cornelis et al. *Cornelis et al.* (2008) propose a calibrated stereo system with a small baseline and known camera height. Khan et al. *Khan and Shah* (2006) address the problem of detecting and tracking objects using a multi-view camera system. A similar strategy is also introduced by Garg et al. *Garg et al.* (2011) and Helmer et al. *Helmer et al.* (2010). Unlike our framework, where the geometry and semantic information are jointly estimated, these methods require that the underlying scene geometry is available or estimated before the semantic recognition step.

A large literature focuses on solving the problem of interpreting complex scenes from 3D data *Frome et al.* (2004); *Huber* (2001); *Rusu et al.* (2008); *Reynolds et al.*

(2011) or a combination of 3D data and imagery *Brostow et al.* (2008); *Koppula et al.* (2011); *Lai et al.* (2011); *Silberman et al.* (2012). In most of these works, 3D information is either provided by external devices (e.g. 3D scanning systems such as LiDAR or a depth sensor such as Kinect) or using traditional SFM techniques. In either case, unlike our framework, the recognition and reconstruction steps are disjoint and cannot mutually benefit each other.

The idea that the interaction between scene components helps the estimation of 3D geometry has been explored by some previous works to obtain a more robust and coherent reconstruction of the scene. For example, Schaffalitzky et al. *Schaffalitzky and Zisserman* (2001) use point-region interactions, Lazebnik et al. *Lazebnik et al.* (2004) use point-object interactions, and *Gupta et al.* (2010); *Hoiem et al.* (2008a); *Bao et al.* (2011b); *Hedau et al.* (2010) use object-region interactions. Our framework incorporates all types of interactions (point-region, point-object, object-region) and use them jointly.

3.3 Semantic Structure From Motion Framework

In this section, we first introduce the unknowns and measurements in our framework. We next introduce our proposed framework and cast our problem as an energy maximization problem. We conclude this section with the inference algorithm for solving the maximization problem.

3.3.1 Measurements and Unknowns

In the semantic structure from motion (SSFM) framework, the measurements are the detected points, regions, and objects in images. The unknowns are points, regions, objects in 3D and the camera configuration. Their definitions and notations are introduced next, and summarized in Tab. 3.2.

Images. The input is a set of *images* $\mathbf{I} = \{I^1 \cdots I^k \cdots I^{N_c}\}$, where I^k is the k^{th}

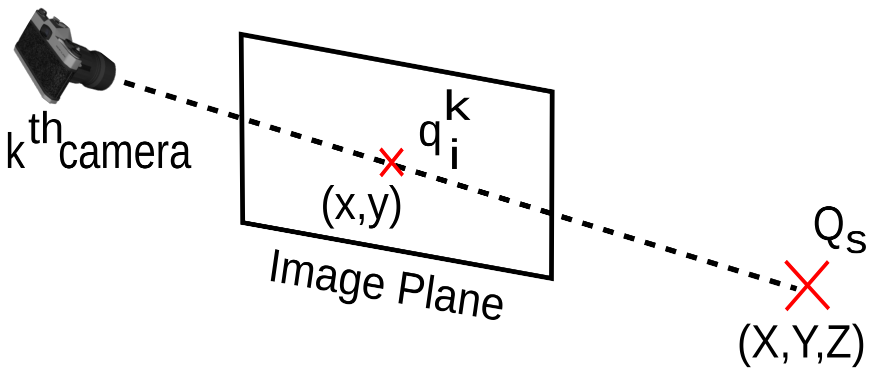


Figure 3.2: 3D point and point measurement

input image. N_c (≥ 2) is the total number of input images.

Cameras. A *camera* is described by its calibration parameters including internal parameter (known), rotation matrix (unknown), and translation vector (unknown) w.r.t world reference system. Let C^k be the calibration parameters of the k^{th} camera that captures image I^k.

Points. The *point measurements* are detected interest points (e.g. by detectors such as *Lowe* (2004); *Tuytelaars and Van Gool* (2000); *Rosten et al.* (2010)). Denote by q_i^k the i^{th} interest point in image I^k (Fig. 3.2). A point measurement q_i^k is described by its location (x, y). Let Q_s be the s^{th} 3D point within the scene (Fig. 3.2). A 3D point Q_s is described by its 3D location (X, Y, Z) in the world reference system. The correspondence between Q_s and point measurements is denoted by u_s, where $u_s = \{i^1, i^2, \cdots\}$ if Q_s corresponds to $q_{i^1}^1, q_{i^2}^2, \cdots$ respectively. Q_s and u_s are unknown.

Objects. The *object measurements* are detected 2D objects (e.g. by detectors such as *Felzenszwalb et al.* (2010); *Leibe et al.* (2004); *Xiang and Savarese* (2012b)). Denote by o_j^k the j^{th} detected object in image I^k (Fig. 3.3). An object measurement o_j^k is described by its location (x, y), bounding box scale h, w, pose θ, ϕ, and category label c. Let O_t be the t^{th} 3D object within the scene (Fig. 3.3). A 3D object O_t is described by 3D centroid (X, Y, Z), normal direction n, pose (Θ, Φ), and category

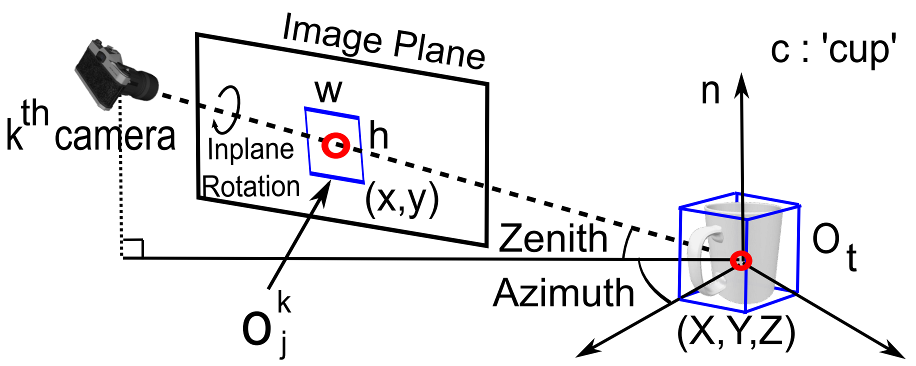


Figure 3.3: 3D object and object measurement

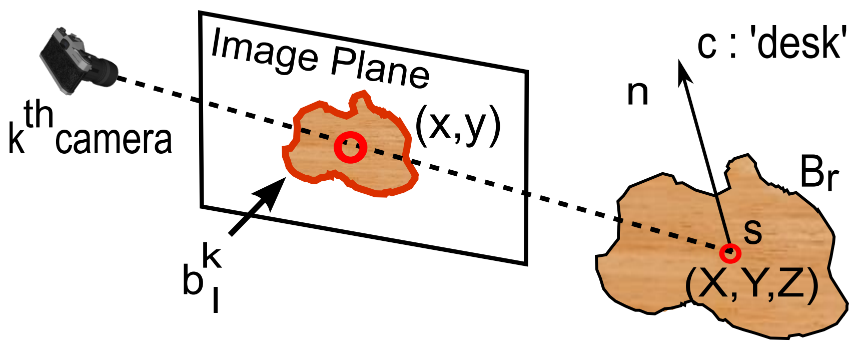


Figure 3.4: 3D region and region measurement

label c (e.g, cup or bottle). The correspondence between O_t and object measurements is denoted by v_t, where $v_t = \{j^1, j^2, \cdots\}$ if O_t corresponds to $o^1_{j^1}, o^2_{j^2}, \cdots$ respectively. O_t and v_t are unknown.

Regions. The *region measurements* are segmented regions (e.g. by segmentation algorithms such as *Ren and Malik* (2003); *Zitnick et al.* (2005)). Denote by b^k_l the l^{th} region in image I^k (Fig. 3.4). A region measurement b^k_l is described by its location (x, y), area s in the image plane, normal direction n, category label c, and appearance descriptor a (e.g. color histogram and texture). s is defined by the boundary of b^k_l in the image plane (red boundary in Fig. 3.4). n can be estimated based on the appearance of b^k_l from a single image by methods such as *Zhang et al.* (2010); *Hoiem et al.* (2007); *Lee et al.* (2009). Let B_r be the r^{th} 3D region within the scene (Fig. 3.4). We assume that 3D regions are planar. A 3D region B_r is described by the 3D centroid (X, Y, Z), the normal direction n, the area s in 3D space, and the category label c

	Camera	Point	Object	Region
Measurements	-	q_i^k (Fig.3.2)	o_j^k (Fig.3.3)	b_l^k (Fig.3.4)
Set Notation	-	$\mathbf{q} : \{q_i^k\}$	$\mathbf{o} : \{o_j^k\}$	$\mathbf{b} : \{b_l^k\}$
3D Unknowns	C^k	Q_s (Fig.3.2)	O_t (Fig.3.3)	B_r (Fig.3.4)
Set Notation	$\mathbf{C} : \{C^k\}$	$\mathbf{Q} : \{Q_s\}$	$\mathbf{O} : \{O_t\}$	$\mathbf{B} : \{B_r\}$
Correspondences	-	u_s	v_t	g_r
Pair Notation	$\mathcal{C}^k : \{C^k, I^k\}$	$\mathcal{Q}_s : \{Q_s, u_s\}$	$\mathcal{O}_t : \{O_t, v_t\}$	$\mathcal{B}_r : \{B_r, g_r\}$
Set Notation	$\mathbb{C} : \{\mathcal{C}^k\}$	$\mathbb{Q} : \{\mathcal{Q}_s\}$	$\mathbb{O} : \{\mathcal{O}_t\}$	$\mathbb{B} : \{\mathcal{B}_r\}$

Table 3.2: List of notations. Details are in Sec. 3.3.1. Sub-indices s, t, r indicate 3D points, 3D objects, and 3D regions respectively (e.g. $(X, Y, Z)_s$ is the 3D location of a 3D point Q_s, and $(X, Y, Z)_t$ is the 3D centroid of a 3D object O_t). Sub-indices i, j, l indicate 2D measurements of points, objects, and regions respectively (e.g. $(x, y)_i$ is the image location of a 2D point q_i, and $(x, y)_l$ is the image location of the centroid of a 2D region b_l). The super-index (usually k) denotes which camera / image a variable is related to (e.g, q_i^k is the i^{th} 2D point in the k^{th} image).

(e.g, sky, road, and wall). The correspondence between B_r and regions measurements is denoted by g_r, where $g_r = \{l^1, l^2, \cdots\}$ if B_r corresponds to 2D regions $b^1_{l^1}, b^2_{l^2}, \cdots$ respectively. B_r and g_r are unknown.

Objects v.s. Regions. Two properties differentiate objects from regions. The first property is 3D volume. An object occupies a certain 3D volume and is bounded by a 3D cube (Fig. 3.3); conversely a region is a planar portion of a surface that does not have a 3D volume. The second property is whether or not the 3D location is predictable from one internally calibrated image. The 3D location of an object can be (roughly) predicted from one detected 2D object in an image, if prior knowledge about the typical 3D object size is available *Bao et al.* (2011b); *Hoiem et al.* (2008a). The 3D location of a region cannot be predicted from only one image, since a region (e.g. a piece of sky) does not have "typical" size. Examples of objects include cars, bottles, persons. Examples of regions include the surface of a road or the sky.

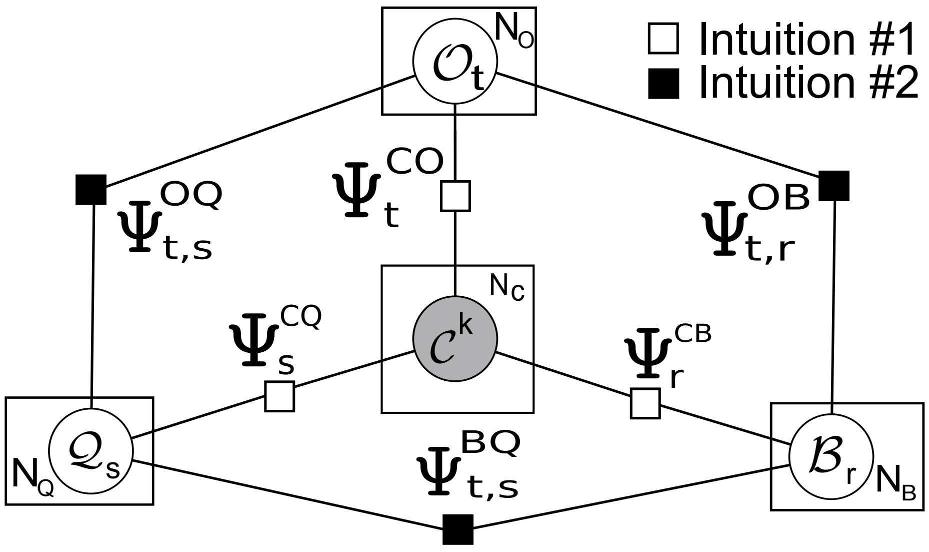


Figure 3.5: Factor node graph. The camera node $\mathcal{C}^k = \{C^k, I^k\}$ is partially observed because the rotation and translation of C^k are unknown and I^k are input images. The object node $\mathcal{O}_t = \{O_t, v_t\}$, point nodes $\mathcal{Q}_s = \{Q_s, u_s\}$, and region nodes $\mathcal{B}_r = \{B_r, g_r\}$ are associated to unknown quantities.

3.3.2 Energy Maximization Framework

Given a set of input images, the goals of SSFM are: i) recognizing objects and classifying regions in both 2D images and 3D space; ii) estimating 3D geometry (locations, scales, and poses) of points, regions, and objects; iii) estimating camera extrinsic parameters. The SSFM framework is formulated following two intuitions.

Intuition #1. The image projection of hypothesized scene components (objects, regions, and points) should be consistent with their image measurements (Fig. 3.6). Such consistency is measured with respect to location, scale, and pose.

Intuition #2. The interactions among estimated scene components should be consistent with the interactions learned from the training set (see Tab. 3.1).

These consistency constraints can be encoded using the factor graph in Fig. 3.5. The nodes (potential terms) Ψ_t^{CO}, Ψ_s^{CQ}, Ψ_r^{CB} are constructed following the intuition #1, and they evaluate how likely the image projections of objects, points, or regions are consistent with the corresponding 2D image measurements. We refer to Ψ_t^{CO}, Ψ_s^{CQ}, Ψ_r^{CB} as *projection potentials*, and they are explained in Sec. 3.4.

The nodes (potential terms) $\Psi^{OB}_{t,r}$, $\Psi^{OQ}_{t,s}$, $\Psi^{BQ}_{r,s}$ are constructed following the intuition #2, and they evaluate how likely the interaction among scene components (objects, points, or regions) are consistent with the learned corresponding relationships. We refer to Ψ^{CO}_{t}, Ψ^{CQ}_{s}, Ψ^{CB}_{r} as *interaction potential*s, and they are explained in Sec. 3.6.

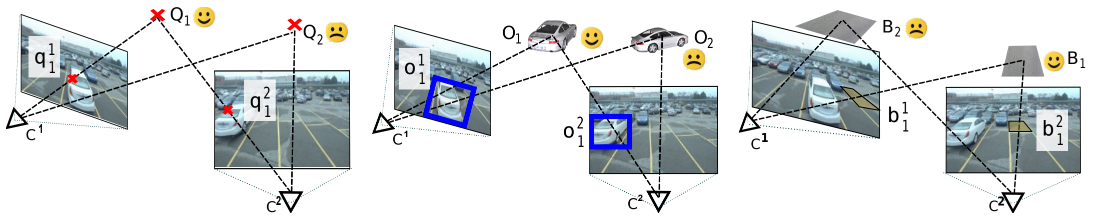
We formulate the SSFM estimation problem as the one of maximizing the overall energy expressed by the factor graph in Fig. 3.5:

$$\begin{aligned}\{\mathbb{Q},\mathbb{O},\mathbb{B},\mathbf{C}\} &= \arg\max_{\mathbb{Q},\mathbb{O},\mathbb{B},\mathbf{C}} \Psi(\mathbb{Q},\mathbb{O},\mathbb{B},\mathbf{C};\mathbf{I}) \\ &= \arg\max_{\mathbb{Q},\mathbb{O},\mathbb{B},\mathbf{C}} \prod_s \Psi^{CQ}_s \prod_t \Psi^{CO}_t \prod_r \Psi^{CB}_r \\ & \prod_{t,s} \Psi^{OQ}_{t,s} \prod_{t,r} \Psi^{OB}_{t,r} \prod_{r,s} \Psi^{BQ}_{r,s} \qquad (3.1)\end{aligned}$$

The definitions of $\mathbb{Q},\mathbb{O},\mathbb{B},\mathbf{C}$ are in Tab. 3.2.

3.3.3 Solving the Energy Maximization Problem

Our goal is to estimate camera parameters, points, objects, and regions by maximizing the energy as in Eq. 3.1. Due to the high dimensionality of the parameter space corresponding to cameras, points, objects, and regions, we use sampling to solve this optimization problem similarly to *Kirkpatrick et al.* (1983); *Dellaert et al.* (2000). Algorithm III.1 summarizes the procedure for sampling the parameter space. After the sampling, the maximum can be identified among all the samples. The cameras, points, regions, objects described by the parameters associated to such a maximum are taken as the solution for Eq. 3.1 and, hence, the solution for the SSFM problem.



(a) Consistency between 3D points and measurements. Q_1 and Q_2 are candidate 3D points given measurements q_1^1 and q_1^2. Q_1 is a good candidate because its projection is consistent with the location associated to q_1^1 and q_1^2.

(b) Consistency between 3D objects and measurements. O_1 and O_2 are candidate 3D objects given measurements o_1^1 and o_1^2. O_1 is a good candidate because its projection is consistent with the pose, location, and scale associated to o_1^1 and o_1^2.

(c) Consistency between 3D regions and measurements. B_1 and B_2 are candidate 3D regions given measurements b_1^1 and b_1^2. B_1 is a good candidate because its projection is consistent with the pose, location, and scale of b_1^1 and b_1^2.

Figure 3.6: Consistency between scene components and measurements. Notice that the 3D car model in (b) is only shown for visualization purposes.

3.4 Projection Potential

In SSFM framework, the scene components (3D points, 3D objects, and 3D regions) are estimated by jointly maximizing their projection potential and interaction potential (Eq. 3.1). In this section, we explain how to compute projection potentials given arbitrary configuration of 3D components and cameras.

3.4.1 Projection Potential for Objects

Given an arbitrary configuration of cameras $\mathbf{C}$ and a configuration of a 3D object $\mathcal{O}_t$, the object projection potential $\Psi_t^{CO}(\mathcal{O}_t, \mathbf{C}; \mathbf{I})$ captures the likelihood of observing O_t in images $\mathbf{I}$. We approximate the potential term $\Psi_t^{CO}(\mathcal{O}_t, \mathbf{C}; \mathbf{I})$ as:

$$\Psi_t^{CO}(\mathcal{O}_t, \mathbf{C}; \mathbf{I}) \propto (1 - \prod_{k=1}^{N_c}(1 - \Psi_t^{CO}(O_t, C^k; I^k))) \tag{3.2}$$

where $\Psi_t^{CO}(O_t, C^k; I^k)$ is object projection potential for an individual camera C_k. I.e., $\Psi_t^{CO}(O_t, C^k; I^k)$ captures the likelihood of observing O_t in image I^k with camera configuration C^k. This decomposition accounts for the fact that we do not want to penalize the case where an object is only observed in a subset of the input images, because of limited field of view.

In order to compute the object projection potential $\Psi_t^{CO}(O_t, C^k; I^k)$, we 1) estimates the image projection of O_t in image I^k by a function $\omega^k(O_t) = o_t^k$ (Sec. 3.4.1.1); 2) obtain the object-detection likelihood of o_t^k as $\Xi^k(o_t^k)$ (Sec. 3.4.1.2); 3) compute the potential value as $\Psi_t^{CO}(O_t, C^k; I^k) = \Xi^k(o_t^k)$.

3.4.1.1 Image Projection of 3D objects

We denote by $\omega^k(O_t) = o_t^k$ the function that allows to estimate o_t^k – the image projection of O_t in image I^k. $\omega^k(O_t)$ allows to estimate the object's location x_t^k, y_t^k, pose ϕ_t^k, θ_t^k, and scale w_t^k, h_t^k in image I^k. Next, we explain how to construct the function ω^k.

Given an hypothesis for an object O_t in a world-reference system, the object's 3D location X_t^k, Y_t^k, Z_t^k and 3D pose Θ_t^k, Φ_t^k in the reference system of camera C^k are available. By using the camera projection matrix *Hartley and Zisserman* (2000), the location x_t^k, y_t^k of O_t in image I^k can be obtained as:

$$[x_t^k, y_t^k, 1]' = K^k[X_t^k, Y_t^k, Z_t^k]'/Z_t^k \tag{3.3}$$

By construction, the pose ϕ_t^k, θ_t^k of the object in the image plane is equal to the pose of 3D object O_t in camera C^k (Fig. 3.3):

$$[\phi_t^k, \theta_t^k] = [\Phi_t^k, \Theta_t^k] \tag{3.4}$$

Accurate estimation of the 2D object scale w_t^k, h_t^k from a 3D object may require a complex geometrical derivation that goes beyond the scope of this paper. We introduce an approximated mapping between 3D scale and 2D scales:

$$\begin{cases} w_t^k = f_k \cdot W(\Theta_t^k, \Phi_t^k, c_t)/Z_t^k \\ h_t^k = f_k \cdot H(\Theta_t^k, \Phi_t^k, c_t)/Z_t^k \end{cases} \tag{3.5}$$

where w_t^k, h_t^k denote the estimated object 2D scale (See Fig. 3.3), f_k is the focal length of the k^{th} camera. $W(\Theta_t^k, \Phi_t^k, c_t)$ or $H(\Theta_t^k, \Phi_t^k, c_t)$ is a (scalar) mapping that describes the typical relationship between the physical object bounding cube and object image bounding box. Notice that, this mapping is a also function of the object pose Θ_t^k, Φ_t^k and category c_t. Given the ground truth 3D object bounding cubes and corresponding image observations in the training set, we use a max likelihood estimator to learn the mapping coefficients W, H for every object poses and categories.

The function $\omega^k(O_t) = o_t^k$ is readily available by combining Eq. 3.3, 3.4, and 3.5.

3.4.1.2 Detection Likelihood for 2D Object

The likelihood that an object of certain class, scale, and pose is found at certain location in an image can be computed using multi-view object detectors such as *Su et al.* (2009); *Savarese and Fei-Fei* (2007), or using object detectors such as *Felzenszwalb et al.* (2010) by treating each object view point as a different class. Let us denote by the tensor Ξ^k such likelihood value, which effectively captures the likelihood of detecting objects from the image I^k.

Specifically, Ξ^k is a set of M likelihood maps, wherein each map captures the likelihood of observing an object of category c with scale w, h and pose θ, ϕ at location x, y in the image. Thus, we can interpret Ξ^k as one $L_x \times L_y \times M$ tensor, where L_x and L_y are the image width and height and M adds up to the number of object categories, scales and poses. Fig. 3.7 shows an example of a set of 15 likelihood maps (3 scales and 5 poses) for only one object category (*car*). Each likelihood map of Ξ^k

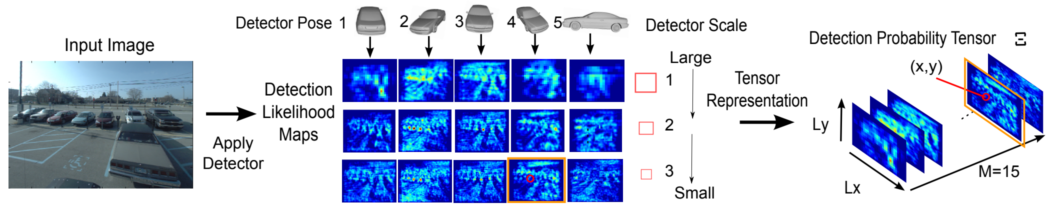


Figure 3.7: Illustration of the construction of the detection likelihood tensor Ξ. A set of "likelihood maps" are obtained by applying a car detector such as *Felzenszwalb et al.* (2010) on the left image. This detector is tuned to detect the object car at 3 scales and 5 poses (thus $M = 15$). The color from red to deep blue indicates the detector response from high to low. In this example, Ξ has dimensions $L_x \times L_y \times 15$. Such detector can be easily used to evaluate the likelihood of an object hypothesis o. Take the object corresponding to the red dot as an example. This car is of location=(x, y), scale=3, and pose=4, which corresponds to the 14^{th} probability map (the orange rectangle). The detection likelihood of this car is returned by $\Xi(x, y, 14)$.

can be computed during a pre-process phase where a set of multi-class and multi-view object detectors are applied to the image I^k. Since Ξ^k is fixed after this pre-process phase, the detection likelihood value $\Xi^k(o)$ can be obtained in constant time for any 2D object hypothesis o.

3.4.2 Projection Potential for Points

Given an arbitrary configuration of cameras $\mathbf{C}$ and a configuration of a 3D point $\mathcal{Q}_s$, the projection potential for the point $\Psi_s^{CQ}(\mathcal{Q}_s, \mathbf{C}; \mathbf{I})$ captures the likelihood of observing 3D point Q_s in images $\mathbf{I}$. This potential term can be estimated by computing the *agreement* between projections of Q_s and its actual measurements. Projections of a 3D point can be obtained by applying the camera projection transformation. Such agreement can be quantified using the epipolar constraints which regulate the relationship between points in 3D, measurements and cameras. Specifically, given a pair of cameras C^l and C^k, we can estimate the fundamental matrix $F_{l,k}$. Suppose q_i^k and q_j^l are a pair of corresponding points related to Q_s in image I^k and I^l respectively. $F_{l,k}$ allows to compute the epipolar line $\xi_i^{l,k}$ (or $\xi_j^{k,l}$) for point q_i^k (or q_j^l) in image I^l (or I^k). If we assume the distance $d_{j,i}^{l,k}$ between $\xi_i^{l,k}$ and q_j^l follows a zero-mean Gaussian distribution with variance σ_u, we can compute the potential as:

$$\Psi_s^{CQ}(\mathcal{Q}_s, \mathbf{C}; \mathbf{I}) = \prod_{k=1}^{N_C} \mathcal{Y}(\exp(-(d_{j,i}^{l,k})^2/\sigma_u)) \tag{3.6}$$

where $\mathcal{Y}(x)$ is a robust estimator that is capable of accommodating outliers. We can learn the variance σ_u using an ML estimator on a validation set.

3.4.3 Projection Potential for Regions

Given an arbitrary configuration of cameras $\mathbf{C}$ and a configuration of a 3D region $\mathcal{B}_r$, the projection potential for the region $\Psi_r^{CB}(\mathcal{B}_r, \mathbf{C}; \mathbf{I})$ captures the likelihood of observing 3D region B_r in images $\mathbf{I}$. This potential Ψ_r^{CB} can be decomposed as a product of two potential terms: $\bar{\Psi}_r^{CB}$ and $\widetilde{\Psi}_r^{CB}$. $\bar{\Psi}_r^{CB}$ captures how likely corresponding across-view measurements of B_r have similar appearance. $\widetilde{\Psi}_r^{CB}$ captures how likely the location, scale, and orientation of the projection of B_r is consistent with its corresponding measurements.

Appearance Potential $\bar{\Psi}_r^{CB}$. The measurements of a 3D region (Sec. 3.3.1) in different images should share similar appearance. However, the degree of across-view appearance similarity of a region depends on the region class. For example, the appearance of a portion of road will not change much as a function of view point transformations. Thus,

$$\bar{\Psi}_r^{CB} \propto \sum_c \prod_{k=1}^{N_C} N(a_{l^k}^k - \bar{a_r}; 0, \Sigma^c) \Pr(c_r = c) \tag{3.7}$$

where $a_{l^k}^k$ is the appearance of $b_{l^k}^k$, $\bar{a_r} = \frac{1}{N_c}\sum_k a_{l^k}^k$ is the mean of appearances, $N(\bullet; a, \Sigma)$ is a Gaussian distribution whose mean is a and covariance is Σ, $\Pr(c_r = c)$ is the probability that B_r is of class c. We learn Σ^c from our training set using a max-likelihood estimator.

Geometry Potential $\widetilde{\Psi}_r^{CB}$. The location and scale of the projection of B_r should be consistent with its corresponding image measurements. Given C^k and B_r, let b_r^k be the image projection of B_r. Let $b_{l^k}^k$ be the corresponding image measurement of B_r in image I^k. We model the potential $\widetilde{\Psi}_r^{CB}$ by evaluating if b_r^k and $b_{l^k}^k$ have: 1) similar image areas; 2) similar normal vectors computed w.r.t the camera reference system; 3) close locations in the image. Thus,

$$\widetilde{\Psi}_r^{CB} \propto \sum_c \prod_{k=1}^{N_C} N(\log \frac{s_r^k}{s_{l^k}^k}; 0, \sigma_{s,c}) N(n_{l^k}^k; n_r^k, \sigma_{n,c}) \tag{3.8}$$

$$N([x_{l^k}^k, y_{l^k}^k]; [x_r^k, y_r^k], \sigma_{xy,c}) \Pr(c_r = c) \tag{3.9}$$

where $s_{l^k}^k$ (or s_r^k) is the image area of $b_{l^k}^k$ (or b_r^k), $n_{l^k}^k$ (or n_r^k) is the region normal vector of $b_{l^k}^k$ (or b_r^k) w.r.t. C^k, and $x_{l^k}^k, y_{l^k}^k$ (or x_r^k, y_r^k) is centroid of the of $b_{l^k}^k$ (or b_r^k) in the image. We chose to use log difference for the area term in Eq. 3.8 since it best fits the real data distribution.

Overall Region Potential. Combining Eq. 3.7 and 3.8, projection potential of regions Ψ_r^{CB} can be computed as:

$$\Psi_r^{CB} = \bar{\Psi}_r^{CB} \widetilde{\Psi}_r^{CB} = \sum_c \prod_{k=1}^{N_C} N(a_{l^k}^k - \bar{a}_r; 0, \Sigma^c)$$

$$N(\log \frac{s_r^k}{s_{l^k}^k}; 0, \sigma_{sc}) N(n_{l^k}^k; n_r^k, \sigma_{nc}) \Pr(c_r = c)$$

3.5 Initializing Cameras and 3D Scene Components

In this section we explain how to initialize cameras and scene components – a very critical step in the sampling algorithm (Algorithm III.1) in the SSFM framework.

3.5.1 Initializing Camera

We follow two strategies for initializing the cameras. One is based on using point correspondences; the other leverages object detections.

3.5.1.1 Initialization by Points

Matched feature points between images (e.g. *Lowe* (2004); *Tuytelaars and Van Gool* (2000); *Rosten et al.* (2010)) can be used to estimate camera poses *Hartley and Zisserman* (2000). In our experiment, we use RANSAC, the five-points algorithm *Nister* (2004), and bundle adjustment *Triggs et al.* (1999) to obtain the camera pose initializations using matched feature points. Due to the metric reconstruction ambiguity, we scale the estimated camera translation with random values to obtain a number of camera pose initializations.

3.5.1.2 Initialization by Objects

Object detections can also serve for initializing cameras. We use a standard object detector (e.g. *Felzenszwalb et al.* (2010); *Savarese and Fei-Fei* (2007)) to detect 2D objects and estimate object pose and scale. Next, we use such object detections to establish possible object correspondences and use these to estimate initial camera configurations. Assume that image I^k has a set of object detections $\mathbf{o}^k = \{o_t^k\}$. The t^{th} detected 2D object o_t^k is parametrized by location x_t^k, y_t^k, bounding box scale w_t^k, h_t^k, and the pose ϕ_t^k, θ_t^k (depending on whether or not the object detector has the ability to predict object pose). Based on these hypotheses, we consider three approaches to initializing the camera rotation R^k and translation T^k (notice that the intrinsic parameter K^k is known): 1) initializing cameras using only the object scale mapping W, H; 2) initializing cameras using only the object pose ϕ, θ; 3) initializing cameras using both 3D scale W, H and pose ϕ, θ. These approaches leverage the following propositions which provide necessary conditions for estimating the camera parameters using object detections.

Proposition 1. Assume that at least 3 objects can be detected in the k^{th} image. Assume that the detector returns object image coordinates $x_t^k, y_t^k (t = 1, 2, 3)$, scales w_t^k, h_t^k, and category c_t. Assume that the mappings W_t and H_t are available for each detected object. Then extrinsic camera parameters R^k, T^k can be calculated.

Proof. We demonstrate proposition 1 for 3 objects but the proof can be extended if more than 3 objects are available. Let O_1, O_2, O_3 be the 3D objects in the world reference system, and O_1^k, O_2^k, O_3^k be their locations, poses, scales in the k^{th} camera reference system. We define the world reference system based on the first camera: location of O_1^1 is the origin; the vector from O_2^1 to O_1^1 is the X-axis; and the locations of O_1^1, O_2^1, O_3^1 (3 points) characterize the X-Y plane. The object coordinate in camera reference system is $[X_t^k, Y_t^k, Z_t^k] = Z_t^k (K^k)^{-1} [x_t, y_t, 1]'$, where Z_t^k can be computed from w_t^k, h_t^k with the mappings W and H. Therefore, we have the camera translation as $T^k = [X_1^k, Y_1^k, Z_1^k]$. Since $[x_t, y_t, 1]' = K^k (R^k [X_t, Y_t, Z_t]' + T_k)/Z_t^k (t = 1, 2, 3)$ and the degree of freedom of R^k is 3, the camera rotation matrix R^k can be solved accordingly.

Proposition 2. Assume that at least 2 objects can be detected in all the images. Assume that from image k the detector returns object image coordinates x_t^k, y_t^k, pose θ_t^k, ϕ_t^k, and category c_t. The camera extrinsic parameters R^k, T^k can be calculated up to a scale ambiguity (metric reconstruction).

Proof. We demonstrate proposition 2 for 2 objects but the proof can be extended if more than 2 objects are available. Let O_1, O_2 be the 3D objects in the world reference system, and O_1^k, O_2^k be their locations, poses, scales in the k^{th} camera reference system. We define the world reference system based on the first camera: the location O_1^1 is the origin; and the normals (q,t,n) of the bounding cube of O_1^1 (Figure 3.3) are the X,Y,Z axes. To address the ambiguity of the metric construction, we assume the distance between O_1 and O_2 is unit length. By the observed pose of O_1^k and O_1^1, the rotation of the k^{th} camera R^k can be computed, and its translation T^k is unknown up to 1 degree of freedom which is the distance of C^k to O_1. Since we assume the distance between O_1, O_2 is unit length, the 3D location of O_2 (in the world system) becomes a function of T^k, denote which by $X_2(T^k), Y_2(T^k), Z_2(T^k)$. Given all the cameras $C^1 \cdots C^{N_k}$, we have equations $[X_2(T^1), Y_2(T^1), Z_2(T^1)] = [X_2(T^2), Y_2(T^2), Z_2(T^2)] = \cdots = [X_2(T^{N_k}), Y_2(T^{N_k}), Z_2(T^{N_k})]$. These equations provide $3 \times (N_k - 1)$ constraints. Since the degree of freedom of T^k is 1, the number of unknowns are N_k. Therefore $\{T^k\}$ can be jointly solved if more than two cameras are available. Notice that the $\{R^k, T^k\}$ are estimated by assuming O_1, O_2 has the unit length. However, the real distance of O_1, O_2 is unknown and therefore the estimation of cameras is up to a metric reconstruction.

Proposition 3. Assume that at least 1 object can be detected in all the images. Assume on image k the detector returns object image coordinates x_t^k, y_t^k, pose θ_t^k, ϕ_t^k, scales w_t^k, h_t^k, and category c_t. Assume that the mapping W_t and H_t are available for each detected object. Then the camera extrinsic parameters R^k, T^k can be calculated.

Proof. We demonstrate proposition 3 for 1 object but the proof can be extended if more than 1 object is available. Let O_1 be a 3D object in the world reference system, and O_1^k be its location, pose, and scale in the k^{th} camera reference system. We define the world reference system based on the first camera: the location of O_1^1 is the origin and the normals (q,t,n) of the 3D cube of O_1 (Figure 3.3) are the X,Y,Z axes. Hence, the pose of the object Θ_1, Φ_1 (in the world system) is the same as the observed Θ_1^1, Φ_1^1 . Object camera coordinate is $[X_1^k, Y_1^k, Z_1^k] = Z_1^k (K^k)^{-1} [x_1, y_1, 1]'$. Therefore, the translation of the k^{th} camera is $T^k = [X_1^k, Y_1^k, Z_1^k]$. Finally, R^k can be computed by θ_1^k, ϕ_1^k and Θ_1, Φ_1.

In our experiments, approach 1 is applied to the pedestrian dataset, as the pose cannot be robustly estimated for pedestrians; approach 2 is not applied to any of our experiments; approach 3 is applied to the Ford car dataset and the office dataset.

3.5.2 Initializing 3D Objects

3D objects are initialized following two steps: 1) obtaining a set of 3D objects by projecting detected 2D objects into 3D space (Sec. 3.5.2.1), 2) applying greedy search to obtain a set of optimized 3D objects based on the set given by the previous step (Sec. 3.5.2.2).

3.5.2.1 Projecting Detected 2D Objects to 3D

As demonstrated by *Hoiem et al.* (2008a); *Bao et al.* (2011b), given the location, pose, and scale of a detected 2D object, its corresponding 3D object can be estimated. Let us denote by $\overline{\omega}(o_t^k) = O_t$ the function that projects a 2D object o_t^k into 3D space and thereby estimates a 3D object O_t. Since $\overline{\omega}(o_t^k)$ is the inverse function of $\omega(O_t)$ introduced in Sec. 3.4.1, the form of $\overline{\omega}(o_t^k)$ can be easily obtained. From each image I^k, we use $\overline{\omega}$ to obtain a set of 3D objects based on the detected objects. Denote by $\mathbb{O}^*$ the set of all 3D objects directly derived from the 2D object detections in all images.

3.5.2.2 Finding Optimized 3D Objects

We use greed search to optimize each object in the set $\mathbb{O}^*$, and use these optimized objects to construct the final set $\mathbb{O}'$ of initialized objects (see Algorithm III.1). Since the objects are assumed to be independent if only the projection potential is considered, each object can be optimized independently from the other.

Given an estimated 3D object $O_t \in \mathbb{O}^*$ and an arbitrary camera configuration, we can propose a set of 3D object candidates (denoted by $\mathbf{O}_t$) around O_t. $\mathbf{O}_t$ is obtained by exhaustively enumerating (locations, poses, scales) in the neighborhood of the parameter space of O_t. Without loss of generality, assume that O_t has its 2D projection in all input images, and o_t^k is the projection of O_t in the k^{th} image. We further assume that, among all input images, the k^{th} image has the largest detection

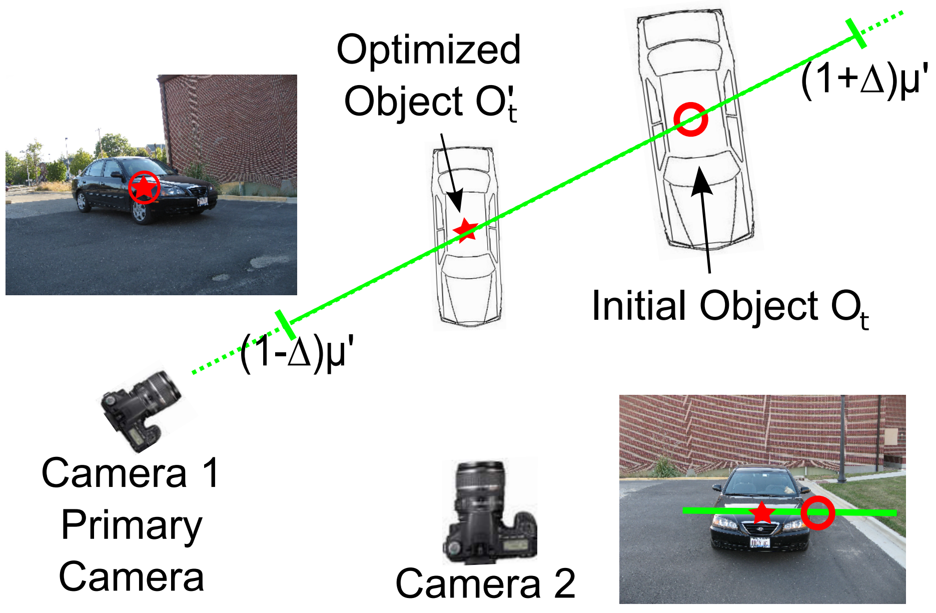


Figure 3.8: Proposing object candidates given the cameras. The red circle corresponds to the initial object estimate $O_t$is the result of the estimation from step i. The green line collects the set of 3D object candidates $\mathbf{O}_t$ given the primary camera (see text for details). O_t' is the location obtained solving the optimization expressed by Eq. 3.10 using all the remaining cameras.

score for O_t, i.e. $\Psi_t^{CO}(O_t, C^k; I^k) = \max_{h=1\cdots N_k} \Psi_t^{CO}(O_t, C^h; I^h)$. We define C^k as the "primary camera" of O_t (Figure 3.8). To limit the number of elements in $\mathbf{O}_t$ and increase the efficiency, for $\forall O_{t'} \in \mathbf{O}_t$, we enforce that: 1) $o_{t'}^k - o_t^k < \Delta o$ where $\Delta o = \{\Delta x, \Delta y, \Delta h, \Delta w, \Delta\theta, \Delta\phi\}$ is a threshold; 2) $Z_t^k/(1+\Delta Z) < Z_{t'}^k < Z_t^k/(1-\Delta Z))$ where $Z_{t'}^k$ is the depth between $O_{t'}$ and C^k, and ΔZ is a threshold. In Figure 3.8, the green line visualizes a range of location values for $\mathbf{O}_t$.

Given $\mathbf{O}_t$, the optimized 3D object O_t' can be selected so as to maximize the potential:

$$O_t' = \arg\max_{O_t' \in \mathbf{O}_t} \Psi_t^{CO}(\mathcal{O}_t, \mathbf{C}; \mathbf{I}) \tag{3.10}$$

Since the parameter space describing the objects is quantized and limited, exhaustive search is possible for solving Eq. 3.10. Finally, since one 3D object may be initialized multiple times from different images, we use 3D non-maximum suppression to remove overlapping 3D objects. We obtain the final set of optimized objects as

$\mathbb{O}' = \bigcup_t O'_t$.

3.5.3 Initializing 3D Points

Since the points are assumed to be independent if only the projection potential is considered, each point can be initialized independently from the other. We first generate the correspondences of the detected interest points across views. A correspondence u_s implies the existence of a 3D point Q_s. In practice, u_s is proposed by feature matching algorithm (e.g. *Lowe* (2004); *Tuytelaars and Van Gool* (2000); *Rosten et al.* (2010)). During each iteration in Algorithm III.1, given a match u_s, a 3D point Q_s can be initialized by triangulation *Hartley and Zisserman* (2000) and bundle adjustment *Triggs et al.* (1999) (Fig. 3.6a).

3.5.4 Initializing 3D Region

Since the regions are assumed to be independent if only the projection potential is considered, each region can be initialized independently from the other. Similar to initializing points, we first identify correspondences among 2D regions across views. The 2D regions can be obtained from each image independently *Ren and Malik* (2003) or concurrently *Zitnick et al.* (2005). We used *Ren and Malik* (2003) in our experiment. Similarly to points, we next establish the correspondences of region measurements across views. We use epipolar constraints and appearance matching (using color histograms and texture features) to find a set of potential matches. A correspondence g_r implies the existence of a 3D region B_r.

As visualized in Fig. 3.6c, given arbitrary camera configurations $\mathbf{C}$ and a correspondence g_r which indicates that a set of 2D regions $\{b^k_{l^k}\}$ are matched, we initialize a 3D region $B_r = (X, Y, Z, n, s, c)_r$ as follows.

- **Region Centroid** $(X, Y, Z)_r$**.** The region centroid is obtained by triangulating the centroids of $\{b^k_{l^k}\}$.

- **Region Normal** n_r**.** Using the appearance and location of the region b^k_{lk}, methods such as *Zhang et al.* (2010); *Lee et al.* (2009); *Hoiem et al.* (2007) can be employed to estimate the normal n^k_{lk} of b^k_{lk} from the camera C^k. For instance, *Zhang et al.* (2010) allows to estimate the region normal from the camera view point using properties of the rank of texture matrices. *Hoiem et al.* (2007) classifies regions into a few discretized normal orientation categories (flat, front, side, etc...). *Lee et al.* (2009) uses vanishing lines to estimate the region's normal w.r.t the camera. Given n^k_{lk} and C^k, we can estimate the normal n^k_r of b^k_{lk} in the world system. We obtain the normal n_r of B_r as $n_r = \frac{1}{N_c}\Sigma_k n^k_r$ where N_C is the number of images.
- **Region Area** s_r**.** Given $(X,Y,Z)_r$ and C^k, we can compute the distance d^k_r between B_r and C^k. Given n_r and C^k, we obtain the angle α^k_r between n_r and the camera C^k line of sight (Fig. 3.4). Each region b^k_{lk} has an image area s^k_{lk}. $s_r = \frac{1}{N_C}\sum_k \frac{d^k_r s^k_{lk}}{cos(\alpha^k_r)}$ where N_C is the number of images.
- **Region Class** c_r**.** The class of B_r is estimated based on the appearance properties of its measurements and its geometrical properties. Given a^k_{lk} (the appearance of b^k_{lk}), we use methods such as *Ladicky et al.* (2010); *Varma and Garg* (2007); *Berg et al.* (2007) to estimate the confidence $f_{app}(c_r = c; a^k_{lk})$ that B_r is of class c. The 3D geometry of B_r can be used as a cue for region classification. For example, most walls are vertical and the most desks are horizontal. Given the estimated geometry $\{n_r, s_r\}$, we learn a KNN classifier to estimate the confidence $f_{geo}(c_r = c; n_r, s_r)$ that B_r is of class c. We compute the probability $\Pr(c_r = c)$ that B_r is of class c as the weighted average of f_{geo} and f_{app}. Notice that estimating the class and geometry of 3D regions based on 2D measurements is usually a challenging problem. However, object-region and point-region interactions help us estimate 3D regions more accurately (Sec. 3.6).

3.6 Interaction Potentials

The concept of *interactions* among scene components (objects, points, and regions) originates from the observation that scene components are related following certain geometrical or physical rules in 3D. For example, 3D points lie on a physical 3D object or 3D region (Fig. 3.13a), and 3D objects lie on a 3D region (Fig. 3.12). Moreover, image cues can be used to validate the existence of interactions between scene components (Fig. 3.9 and Fig 3.13b). The summary of all interaction types is presented in Tab. 3.1. A pair of scene components that are hypothesized to interact can be associated to an *interaction potential* (i.e. $\Psi^{OQ}_{t,s}, \Psi^{OB}_{t,r}, \Psi^{BQ}_{r,s}$), which evaluates the likelihood that a pair of scene components do interact. If two scene components do not interact, their interaction potential is set to be 1.

One step in Algorithm III.1 is to search for the best configuration of $\mathbb{O}, \mathbb{Q}, \mathbb{B}$ so as to maximize interaction potential (i.e. $\{\mathbb{O}, \mathbb{Q}, \mathbb{B}\} = \arg\max \prod_{t,s} \Psi^{OQ}_{t,s} \prod_{t,r} \Psi^{OB}_{t,r} \prod_{r,s} \Psi^{BQ}_{r,s}$). This step refines the 3D locations and poses of all scene components, and is accomplished by a gradient descent method. The initial values $\mathbb{O}', \mathbb{Q}', \mathbb{B}'$ for the gradient descent algorithm are provided by individual initializations of every scene components, as will be discussed next in Sec. 3.5.2, 3.5.3, and 3.5.4.

3.6.1 Object-point Interaction Potential

A point Q_s and an object O_t interact if Q_s lies on the surface of O_t. The object-point interaction potential $\Psi^{OQ}_{t,s}$ captures the likelihood of the interaction between an object O_t and a point Q_s. If Q_s and O_t interact, the corresponding observations should be consistent across views (Fig. 3.9). We compute $\Psi^{OQ}_{t,s}$ under the assumption that the interaction likelihood for all input images can be decomposed into pair-wise terms:

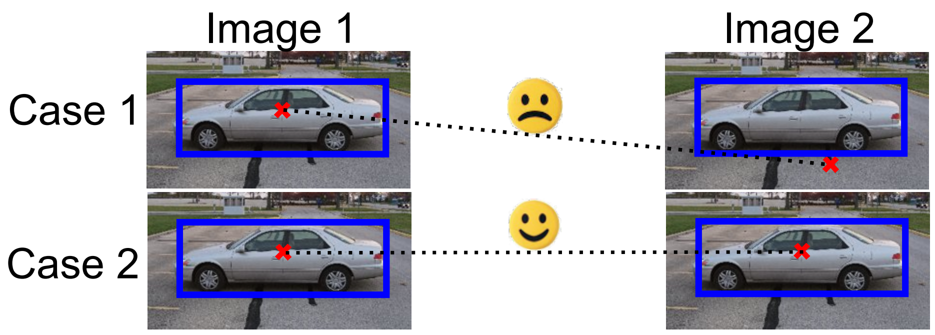


Figure 3.9: Object-point interactions across views. If an object O_t and point Q_s interact (i.e. the point lies on the object surface in 3D), then the observations of Q_s across views should also lie on the observations of O_t (blue bounding boxes). This is true for case 2 but not for case 1.

$$\Psi^{OQ}_{t,s} = \prod_{k_1 \neq k_2} \Psi^{OQ}(q^{k_1}_{i_{k_1}}, q^{k_2}_{i_{k_2}}, o^{k_1}_t, o^{k_2}_t)$$

where $q^k_{i_k}$ is the 2D point measurement corresponding to Q_s in image I^k, o^k_t is the 2D object measurement corresponding to O_t in image I^k, and $\Psi^{OQ}(q^{k_1}_{i_{k_1}}, q^{k_2}_{i_{k_2}}, o^{k_1}_t, o^{k_2}_t)$ is the potential (Eq. 3.11) evaluating the likelihood of the interaction between O_t and Q_s for 2 views I^{k_1} and I^{k_2}. Notice that here $\{q^1_{i_1} \cdots q^k_{i_k} \cdots\}$ is a set of across-view matched points, and $\{o^1_{i_1} \cdots o^k_{i_k} \cdots\}$ is a set of across-view matched objects. $\Psi^{OQ}(q^{k_1}_{i_{k_1}}, q^{k_2}_{i_{k_2}}, o^{k_1}_t, o^{k_2}_t)$ can be evaluated by measuring to what degree the image locations of $q^{k_1}_{i_{k_1}}, q^{k_2}_{i_{k_2}}$ are consistent with the image locations, scales, and poses of $o^{k_1}_t, o^{k_2}_t$. However, this is not a trivial task due to the fact that objects have a 3D shape which is complex in general and can lead to self occlusions as the viewpoint changes (Fig. 3.10).

Clearly, $\Psi^{OQ}(q^{k_1}_{i_{k_1}}, q^{k_2}_{i_{k_2}}, o^{k_1}_t, o^{k_2}_t)$ can be computed exactly if the 3D model of the object is available and the camera poses are known. In practice, however, the actual 3D shape is unknown (we only have an hypothesis of the object category label) and recovering it goes beyond the scope of this paper. In order to compute $\Psi^{OQ}(q^{k_1}_{i_{k_1}}, q^{k_2}_{i_{k_2}}, o^{k_1}_t, o^{k_2}_t)$ without relying on these assumptions, we introduce two functions L and U. Given the image observations of a 3D object O_t and the assumption

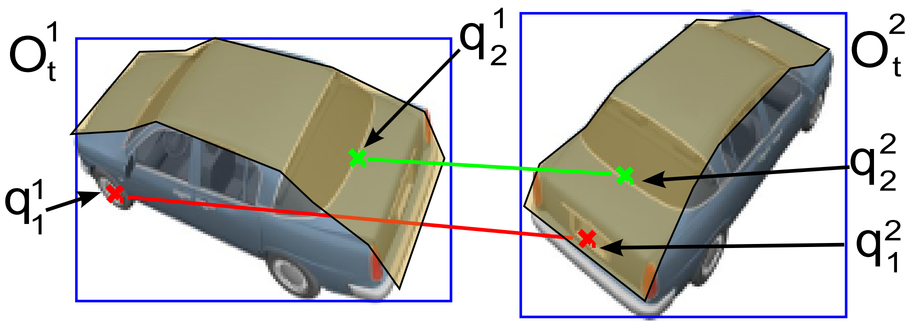


Figure 3.10: Visualization of the function L. Given an object and two (estimated) view points, L predicts a pair of regions (shown in yellow) that can potentially contain consistent feature correspondences. If a pair of points both fall into these regions (e.g. q_2^1 and q_2^2), it is likely to be a true match (so that $L = true$). Otherwise, it (e.g. q_1^1,q_1^2) is likely to be a false match (so that $L = false$).

a 3D point Q_s interacts with O_t, the function L predicts if a pair of image points is compatible with Q_s in the image, and U provides the predicted locations of Q_s. Please refer to Sec. 3.6.1.1 and Sec. 3.6.1.2 for more details regarding L and U. By using the functions L and U, $\Psi^{OQ}(q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}, o_t^{k_1}, o_t^{k_2})$ can be computed based on the image observations of the interacting object and point, which will be explained in Sec. 3.6.1.3.

3.6.1.1 Identifying Object-Point Interactions

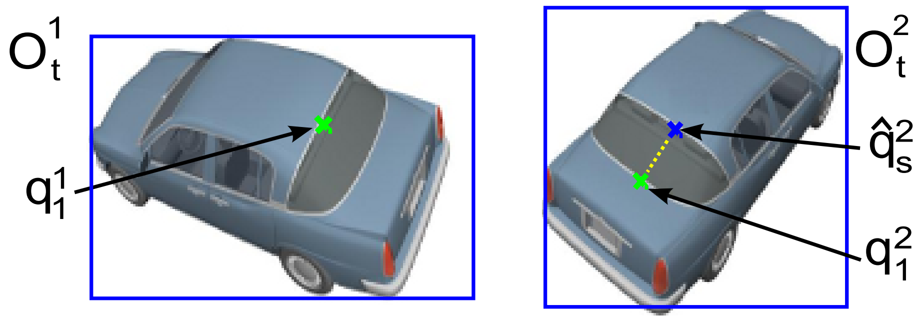
Suppose a point Q_s and an object O_t interact. The corresponding 2D feature point of Q_s in image I^k must fall into the 2D bounding box of o_t^k (the object projection of O_t in image I^k). However, notice that not all the 2D points within the bounding box of o_t^k interact with O_t – e.g. the bounding box of o_t^k may include features from the background which are not part of the object foreground O_t. With that in mind, we introduce a function $L = \{true, false\}$ to estimate whether or not two points (in different images) can correspond a 3D point interacting with an object. L returns "true" if a pair of matched feature points $q_{i_1}^1, q_{i_2}^2$ correspond to two observations of the same 3D point Q_s lying on the 3D physical object O_t (points and objects interacts). L returns "false" otherwise. Notice that, $L(q_{i_1}^1, q_{i_2}^2) =' false'$ suggests that either the points do not belong to the same object or the two points cannot be observed simultaneously. Therefore, in such a case, we set the potential $\Psi^{OQ}(q^1_{i_1}, q^2_{i_2}, o^1_t, o^2_t) = 0$.

We propose to learn this mapping L implicitly from a set of training data. Specifically, L is modeled as a classifier which is learned to associate input data points (i.e., the locations of the matched features in each image pair) to a "true" or "false" label. This association is learned for each (discretized) object pose and object class label. This association can be made *quasi* independent from the camera configurations by normalizing the feature coordinates with respect to the detected object bounding box. The classifier will return "true" labels if matched features are geometrically consistent with the object class and the object pose transformation; whereas it will associate "false" labels to "inconsistent" configurations which typically stems from self-occlusions or background regions. The classifier L can be expressed as follows:

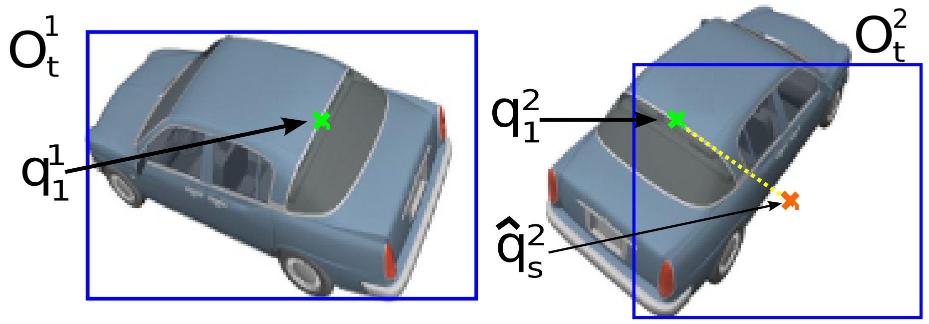
$$L_{\gamma^1_t,\gamma^2_t,c}(\frac{x^1_{i_1} - x^1_t}{w^1_t}, \frac{y^1_{i_1} - y^1_t}{h^1_t}, \frac{x^2_{i_2} - x^2_t}{w^2_t}, \frac{y^2_{i_2} - y^2_t}{h^2_t}) = \{true, false\}$$

where the coordinates $\frac{x^1_{i_1}-x^1_t}{w^1_t}, \frac{y^1_{i_1}-y^1_t}{h^1_t}, \frac{x^2_{i_2}-x^2_t}{w^2_t}, \frac{y^2_{i_2}-y^2_t}{h^2_t}$ are normalized with respect to the detected object bounding box (whose size scale are w_t and h_t) in each image, the variables associated with the 2D object projection are sub-indexed by t, and L's sub-indexes γ^1_t,γ^2_t,c capture the dependency of L on the object poses (γ^1_t,γ^2_t) and class labels (c). We implement L using a non-linear SVM. The classifier predicts a pair of decision regions (in the image plane) that may contain consistent feature correspondences (i.e. those labeled as "true"). As Fig. 3.10 shows, regions associated to a "true" label are highlighted in yellow. If $L_{\gamma^1_t,\gamma^2_t,c}(\frac{x^1_{i_1}-x^1_t}{w^1_t}, \frac{y^1_{i_1}-y^1_t}{h^1_t}, \frac{x^2_{i_2}-x^2_t}{w^2_t}, \frac{y^2_{i_2}-y^2_t}{h^2_t}) = false$, the match between $q^1_{i_1}$ and $q^2_{i_2}$ will result in $\Psi_{O,Q} = 0$. To simplify the notation, we use $L(q^{k_1}_{i_{k_1}}, q^{k_2}_{i_{k_2}})$ to denote the function L.

We learn the parameters of L (i.e the coefficient of the SVM classifier) from a training set. In our training set, images and corresponding depth maps are available. From these we can easily obtain ground truth feature matches on 3D objects across view points. We learn L for different pose pairs and for different object classes.



(a) U predicts the matched point of q_1^1 to be $\hat{q}_s^2$. The yellow dashed line indicates the distance d between $\hat{q}_s^2$ and q_1^1's actual matched point q_1^2. d should be zeros ideally. We use d to evaluate the likelihood of whether or not a pair of matched points (e.g. q_1^1 and q_1^2) is correct.



(b) If the object detection hypothesis is wrong (right blue rectangle), the prediction of U will be deviated from the matched point. As a result, even if the correspondence between q_1^1 and q_1^2 is correct, the interaction likelihood will drop due to the incorrect object detection.

Figure 3.11: Predicting Matching Point Location by U

3.6.1.2 Predicting Matching Point Location

If a pair of matched points $q^1_{i_1}$, $q^2_{i_2}$ both fall into the visible regions of 2D objects ($L =' true'$), the likelihood of such match can be measured based on their image locations. In order to measure such a likelihood, we introduce a function U which is capable of predicting the location of matched feature points (lying on the object O_t) across views given the object 2D projections in two images, and the 2D image location of one of the two matched points. See Fig. 3.11a and 3.11b as examples. This prediction can be made deterministic given the object 3D shape and camera configurations. As discussed earlier, however, we assume the object 3D shape is not available and we rather aim at learning this prediction using a training set. Similarly to L, by normalizing the feature coordinates with respect to the detected object bounding box, we can make U to be a function of the object class and pose transformation only and express U as:

$$U_{\gamma_t^1,\gamma_t^2,c}(\frac{(x_{i_1}^1 - x_t^1)}{w_t^1}, \frac{(y_{i_1}^1 - y_t^1)}{h_t^1}) = (\frac{\hat{x_i^2} - x_t^2}{w_t^2}, \frac{\hat{y_i^2} - y_t^2}{h_t^2})^T$$

where γ_t^1, γ_t^2 are 2D object poses, x_t^1,y_t^1,x_t^2,y_t^2 are 2D object projections, w_t^1,h_t^1,w_t^2,h_t^2 are 2D object projection scales, $x_{i_1}^1$,$y_{i_1}^1$ are the observed point in the 1st image, and $\hat{x}_i^2$,$\hat{y}_i^2$ are the predicted image point in the 2nd image. In order to limit the number of parameters required to learn U, we use a second order Taylor expansion to approximate U. This makes the learning tractable and the prediction more robust to data noise. Assume the exact prediction has Taylor expansion $\xi_{\gamma^1,\gamma^2,c}$ so that $U_{\gamma^1,\gamma^2,c}(x, y) = \xi_{\psi^1,\psi^2,c} \cdot (1, x, y, xy, x^2, y^2...)^T$. We use the quadratic terms to approximate U, i.e. $U_{\gamma^1,\gamma^2,c}(x, y) \approx \alpha_{\gamma^1,\gamma^2,c} \cdot (1, x, y, xy, x^2, y^2)^T$. To simplify the notation, we use $U(q_{i_{k_1}}^{k_1})$ to denote the function U that predicts the location of $\hat{q}_s^{k_2}$ (i.e. the matched point of $q_{i_{k_1}}^{k_1}$ in image k_2). We perform linear regression to learn the parameters of U (i.e. α) for different object pose pairs and categories.

3.6.1.3 Computing object-point interaction potential

Suppose that $q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}$ are the observations of Q_s which interacts with O_t. If $q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}$ are predicted by L to be both visible (i.e. $L(q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}) =' true'$) given the observations of O_t ($o_t^{k_1}$ and $o_t^{k_2}$), the location of $q_{i_{k_1}}^{k_1}$ in image I^{k_2} (denoted by $\hat{q}_s^{k_2}$) predicted by U should ideally overlap with $q_{i_{k_2}}^{k_2}$. We use the distance (denoted by $d_{i_{k_1},i_{k_2}}^{k_1,k_2}$) between $q_{i_{k_2}}^{k_2}$ and $\hat{q}_s^{k_2}$ to evaluate the potential $\Psi^{OQ}(q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}, o_t^{k_1}, o_t^{k_2})$. Assuming that $d_{i_{k_1},i_{k_2}}^{k_1,k_2}$ satisfies a zero-mean Gaussian distribution, the potential $\Psi^{OQ}(q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}, o_t^{k_1}, o_t^{k_2})$ can be computed as:

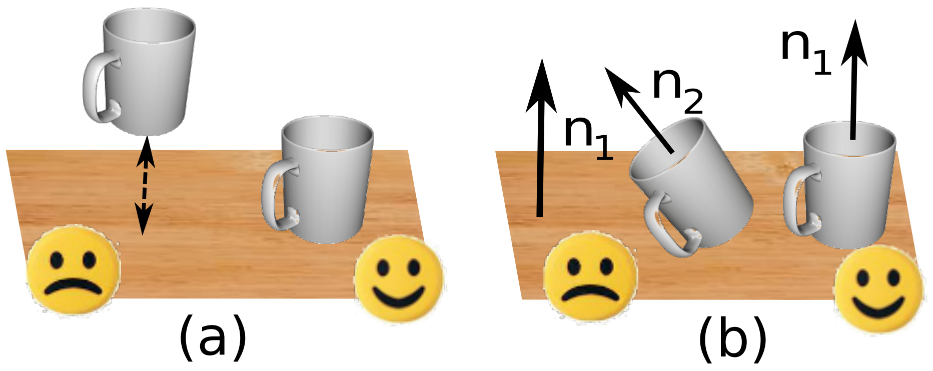


Figure 3.12: Object-region interaction. The estimated object should lie on the support region (a) and be in its up-right pose (b).

$$\Psi^{OQ}(q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}, o_t^{k_1}, o_t^{k_2}) = \begin{cases} 0 & \text{if } L(q_{i_{k_1}}^{k_1}, q_{i_{k_2}}^{k_2}) = false \\ N(d_{i_{k_1}, i_{k_2}}^{k_1, k_2}; 0, \Sigma_{c,\gamma^{k_1},\gamma^{k_2}}) & \text{o.w.} \end{cases} \tag{3.11}$$

where $N(\cdot; 0, \Sigma_{c,\gamma^{k_1},\gamma^{k_2}})$ is a Gaussian density function whose mean is zero and covariance matrix is $\Sigma_{c,\gamma^{k_1},\gamma^{k_2}}$.

3.6.2 Object-region Interaction Potential

An object O_t and a region B_r interact if B_r is the supporting region for O_t, i.e. O_t physically sits on the surface defined by B_r, and O_t sits in an up-right pose (Fig. 3.12). The object-region interaction potential $\Psi_{t,r}^{OB}$ captures the likelihood that the interaction between an object O_t and a region B_r conforms to the two conditions indicated above. As shown by *Hoiem et al.* (2008a); *Bao et al.* (2011b), supporting regions (surfaces) can be used to add constraints on object detection task, in turn, detected objects help the estimation of the geometry of supporting regions.

3.6.2.1 Identifying Object-region Interactions

In order to identify object-region interactions in a scene, we construct a set of candidate pairs of object and region that are hypothesized to interact. Each element (e.g. $\{O_t, B_r\}$) in this set is selected following three criteria: i) the bottom side of the bounding box of o_j^k (the image measurement of O_t) lies within b_l^k (the image measurement of B_r), ii) the region class of B_r and object class of O_t are compatible. This compatibility is expressed by an indicator function $I_{t,r}^{OB}(c_t, c_r) = \{0, 1\}$, which returns 1 (compatible) if an object of class c_t and a region of class c_r are very likely to interact, and returns 0 (incompatible) otherwise. For instance, $I_{t,r}^{OB}(c_t, c_r) = 0$ if $c_t =$ 'car' and $c_r =$ 'tree', because it is very unlikely for a car to "sit" on a tree. The indicator function $I_{t,r}^{OB}(c_t, c_r)$ is learned using a training set.. If both criteria are met, then we say that a region B_r and an object O_t do interact and we evaluate their corresponding interaction potential.

3.6.2.2 Computing Object-region Interaction Potential

After that the object-region interactions are identified, we evaluate their potential values $\Psi_{t,r}^{OB}$ by using the location and the pose consistencies. Specifically,$\Psi_{t,r}^{OB}$ is the product of the location consistency potential $\bar{\Psi}_{t,r}^{OB}$ and the pose consistency potential $\widetilde{\Psi}_{t,r}^{OB}$.

Location Consistency. If O_t lies on B_r, the bottom face of O_t touches the surface of B_r (Fig. 3.12.a). Denote by $d_{t,r}$ the point-to-plane distance from the bottom of the bounding cube of O_t to the surface of B_r. Assuming a Gaussian measurement noise, $d_{t,r}$ follows a zero-mean Gaussian distribution controlled by the variance σ^{OB}. In general, σ^{OB} is a function of object category c_t and region class c_r. $d_{t,r}$ can be used to evaluate $\bar{\Psi}_{t,r}^{OB}$:

$$\bar{\Psi}_{t,r}^{OB} \propto \sum_c [I_{t,r}^{OB} N(d_{t,r}; 0, \sigma_d^{OB}(c_t, c_r)) + (1 - I_{t,r}^{OB})] \Pr(c_r = c) \tag{3.12}$$

where $\Pr(c_r = c)$ is the confidence that B_r is of class c (Sec. 3.5.4).

Pose Consistency. If O_t lies on B_r, the pose of O_t and B_r should be consistent (Fig. 3.12.b): n_r (the normal of B_r) should be equal to n_t (the normal of O_t). The angle between n_r and n_t can also be used to evaluate $\widetilde{\Psi}^{OB}_{t,r}$:

$$\widetilde{\Psi}^{OB}_{t,r} \propto \sum_c [I^{OB}_{t,r} N(acos(n_r' n_t); 0, \sigma^{OB}_n(c_t, c_r)) + (1 - I^{OB}_{t,r})] \Pr(c_r = c) \tag{3.13}$$

where $\sigma^{OB}_n(c_t, c_r)$ is the variance of the angle.

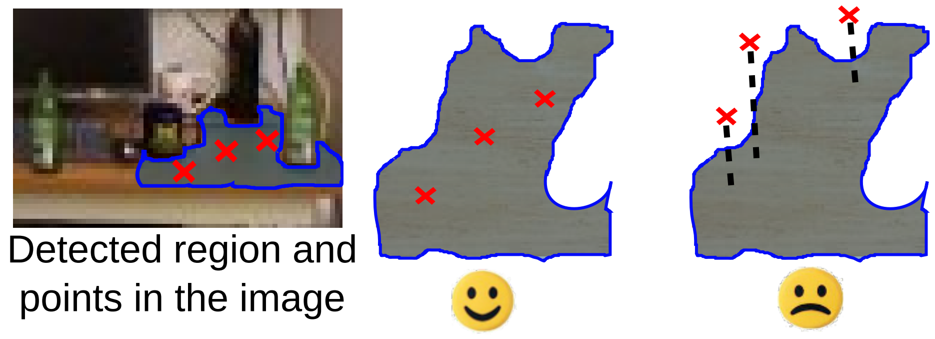
Interaction potential. Given Eq. 3.12 and 3.13, $\Psi^{OB}_{t,r}$ is computed as:

$$\begin{aligned}\Psi^{OB}_{t,r} = \bar{\Psi}^{OB}_{t,r}\widetilde{\Psi}^{OB}_{t,r} = \sum_c [I^{OB}_{t,r} N(d_{t,r}; 0, \sigma^{OB}_d(c_t, c_r))\\ N(acos(n_r' n_t); 0, \sigma^{OB}_n(c_t, c_r)) + (1 - I^{OB}_{t,r})] \Pr(c_r = c)\end{aligned} \tag{3.14}$$

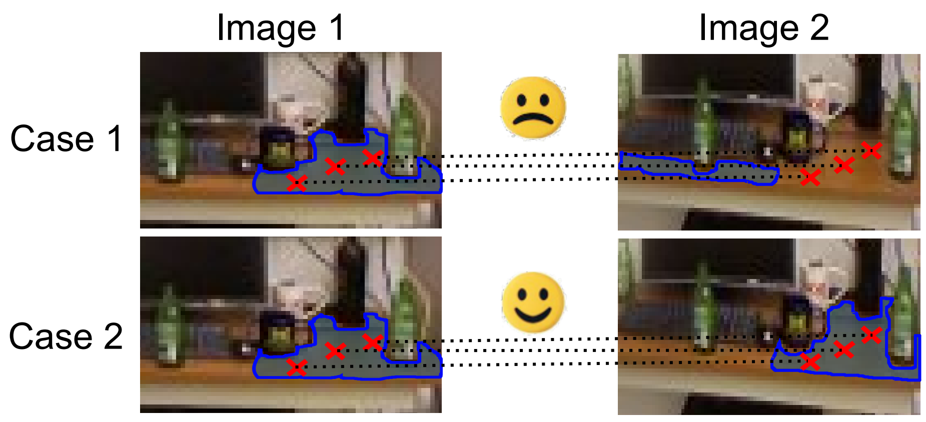
3.6.3 Point-region Interaction Potential

A point Q_s and a region B_r interact if Q_s lies on the surface of B_r. This can be verified by the fact that that: i) the image measurements of Q_s and B_r are consistent across views (Fig. 3.13b); ii) the 3D locations of Q_s and B_r are close to each other (Fig. 3.13a). The point-region interaction potential $\Psi^{QB}_{s,r}$ captures the likelihood that the interaction between Q_s and B_r conform to the two conditions indicated above. As shown in works such as *Schaffalitzky and Zisserman* (2001); *Ferrari et al.* (2003), point-region interactions help improve the accuracy in matching feature elements across views, as well as help the estimation of 3D locations and poses of both points and regions.

Similarly to the object-region interaction, we construct a set of candidate point-region interactions. A pair $\{Q_s, B_r\}$ will be selected as an element in this set if q_i^k (image measurement of Q_s) lies within b_l^k (image measurement of B_r). If this criterion is met, we say that Q_s and B_r do interact, and evaluate their interacting potential $\Psi^{QB}_{s,r}$. Such potential will have a large value if Q_s is close to B_r in 3D (wrong match



(a) The estimated 3D points (red crosses) should lie on the estimated 3D region (a portion of desk).



(b) Point-region interaction. Case 1: matched points are not associated to the same region across views. Case 2: matched points are associated to the same region across views.

Figure 3.13: Point-region interaction.

of points or regions may cause the estimated 3D point Q_s and region B_r to be far apart). Denote by $d_{s,r}$ the point-to-plane distance between Q_s to B_r. We assume the measurement noise makes $d_{s,r}$ obey a zero-mean Gaussian distribution . Thus, we have:

$$\Psi_{s,r}^{QB} = \sum_{c} N(d_{s,r}; 0, \sigma^{QB}(c_r)) \Pr(c_r = c) \tag{3.15}$$

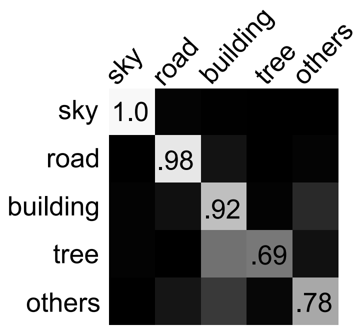
3.7 Evaluation

In this section, we qualitatively demonstrate the ability of the SSFM model to jointly estimate the scene components and camera configurations. We test SSFM on three datasets: the publicly available Ford Campus Vision and LiDAR Dataset*Pandey et al.* (2010), a novel Kinect office dataset*Bao and Savarese* (2011b), and a novel street-view pedestrian stereo-camera dataset. Fig. 3.18 shows anecdotal examples from these datasets. Although the SSFM does not use any information from 3D points, the calibrated 3D points from LiDAR and Kinect allow us to easily obtain ground truth information. From the training set, we learn the variances in Eq. 3.7 - Eq. 3.15 using maximum likelihood estimation. Benchmark comparisons with several state-of-the-art algorithms demonstrate that our method achieves significant improvement on object detection and camera pose estimation results.

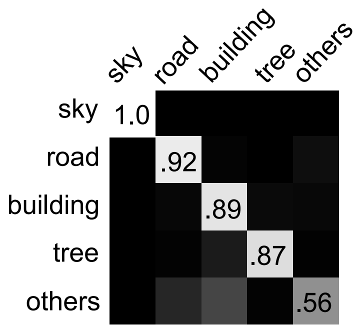
3.7.1 Datasets

We investigate the performance of SSFM using three datasets.

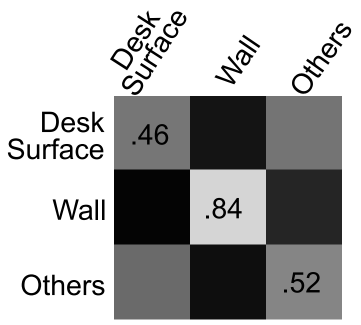
Ford Campus Vision and LiDAR Dataset *Pandey et al.* (2010). This dataset consists of images of cars aligned with 3D scans obtained using a LiDAR system. Ground truth camera poses and parameters are also available. Our training and testing set contain 4 and 5 different static scenarios respectively. For experiments in



(a) Car. Ours w/ *Ladicky et al.* (2010).



(b) Person. Ours w/ *Ladicky et al.* (2010).



(c) Office. Ours w/ *Varma and Garg* (2007).

Figure 3.14: Confusion table for region classification by our framework. "w/" = "with". Our framework uses the appearance-based classification confidence (i.e. f_{app} in Sec. 3.5.4) produced by either *Ladicky et al.* (2010) or *Varma and Garg* (2007).

Sec. 3.7.2 and Sec. 3.7.3.3, we randomly selected 200 and 350 groups of images out of the training and testing images respectively, with the rule that images in one group must capture the same scene. The number of images in one group may be 2, 3, or 4 depending on the experimental setup. The training set for the car detector is the 3D object dataset *Savarese and Fei-Fei* (2007). This training set consists of 8 poses.

Kinect office dataset *Bao and Savarese* (2011b). We use Microsoft's Kinect to collect images and corresponding 3D range data of several static indoor office environments. The ground truth camera parameters are obtained by aligning range data across different views. We manually identify the locations of ground truth 3D object bounding cubes. The dataset includes static office desktop environments. Object categories in this dataset are monitors, keyboards, and mice. The testing and training sets contain 5 different scenarios respectively and each scenario has ~50 images. As for experiments in Sec. 3.7.2, we randomly select 500 image pairs for the testing and the training respectively (100 pairs in each scenario).

Street-view pedestrian stereo-camera dataset. This dataset is collected by using a pair of pre-calibrated cameras, which are parallel and bear a relative distance of $4m$. Images portray moving pedestrians (objects) in streets. However, since each

e_T / e_R	Car	Person	Office
Snavely et al. (2008)	$26.5°/< 1°$	$27.1°/21.1°$	$8.5°/9.5°$
Bao and Savarese (2011a)	$19.9°/< 1°$	$17.6°/3.1°$	$4.7°/3.7°$
Bao and Savarese (2011a) + Regions	$18.0°/< 1°$	$15.7°/3.3°$	$4.9°/4.1°$
This paper	$\mathbf{12.1}°/< 1°$	$\mathbf{11.4}°/\mathbf{3.0}°$	$\mathbf{4.2}°/\mathbf{3.5}°$

Table 3.3: The error for camera pose estimation. The reported errors are computed in the second camera, given that the first camera is at the canonical position.

pair of images are captured simultaneously, the scene contained in every pair of images can be considered to be static. The training set of the object detector is the INRIA pedestrian dataset *Dalal and Triggs* (2005) without pose labels. The typical object-to-camera distance is $5-10m$. The training set contains 200 image pairs in 5 different scenarios. The testing set contains 200 image pairs in 6 other scenarios.

3.7.2 Quantitative Evaluation

3.7.2.1 Camera Pose Estimation

We evaluate the ability of SSFM to estimate the camera poses from input images. We assume that internal parameters are known. The results are reported in Tab. 3.3. The two numbers in each cell are translation errors e_T / rotation errors e_R, following the criterion in *Nister* (2004). Let R_{gt} and T_{gt} be the ground truth camera rotation and translation, and let R_{est} and T_{est} be the estimated camera rotation and translation. The translation error e_T is the angle (in degree) between the estimated baseline and the ground truth baseline, i.e. $e_T = acos(\frac{T_{gt}^T R_{gt}^{-T} R_{est}^{-1} T_{est}}{|T_{gt}| \cdot |T_{est}|})$. The rotation error e_R is the minimal rotation angle (in degree) of $R_{gt} R_{est}^{-1}$. As our framework jointly uses points, objects and regions to estimate the camera poses, it shows superior performances over our baseline algorithms. Our baseline methods are Snavely et al. *Snavely et al.* (2008) (which only uses points) and Bao and Savarese *Bao and Savarese* (2011a) (which only uses points and objects).

Percentage %	Car	Person	Office
Ladicky et al. (2010)	88.9	82.9	50.8
Varma and Garg (2007)	78.2	74.6	54.7
Ours with *Ladicky et al.* (2010)	90.2	84.4	51.2
Ours with *Varma and Garg* (2007)	80.0	75.4	59.0

Table 3.4: Average accuracy for region classification tasks. The numbers capture the percentage of the total pixels correctly labeled / total pixels in the reconstructed regions. Our framework uses the appearance-based classification confidence (i.e. f_{app} in Sec. 3.5.4) produced by either *Ladicky et al.* (2010) or *Varma and Garg* (2007).

3.7.2.2 Estimating Regions

Region Classification. The average accuracy in region classification task is reported in Tab. 3.4. For car / person / office set, we train the region classifier using the car *Bao and Savarese* (2011a) / Camvid *Brostow et al.* (2008) / office *Bao and Savarese* (2011a) dataset. Fig. 3.14 shows the confusion table for the three datasets. We report the classification results using all the 2D regions that our framework is capable to match and reconstruct in 3D. In car / person / office dataset, our framework is able to match and reconstruct the regions that account for 14.4% / 9.6% / 11.3% of total area of all input images. Examples of classifying and matching regions across views shown in Fig. 3.15a and 3.18.

Region 3D Orientation. Tab.3.5 shows the average error in estimating region orientations. Let n_{gt} be the ground truth normal of a region which is computed using the range data. If the estimated region normal is n_{est}, the error is defined as $acos(|n'_{gt}n_{est}|)$. We cannot evaluate orientations in the person dataset since it does not include range data. The baseline algorithms are Zhang et al. *Zhang et al.* (2010) and Hoiem et al. *Hoiem et al.* (2007). *Zhang et al.* (2010) estimates the region orientation by its texture, and it tends to fail when the region has little texture or irregular texture. *Hoiem et al.* (2007) estimates the region orientation by its appearance and its location in an image. Since our framework estimates the region orientation by taking advantage of object-region and point-region interactions in addition to its

	Zhang et al. (2010)	*Hoiem et al.* (2007)	This Paper
Car	69.2°	27.0°	**26.1**°
Office	45.2°	40.5°	**37.9**°

Table 3.5: Average error in estimating the region normals.

with / without interaction	$median(e_d)$	$var(e_d)$
Car	0.281 / **0.175**	0.54 / 0.44
Office	0.033 / **-0.011**	0.182 / 0.189

Table 3.6: Average error in estimating region 3D locations. The two numbers in each cell are our results estimated with / without interactions. Given a detected region, let d_{est} be its distance to camera, and let d_{gt} be its ground truth distance from the range data. We define the error as $e_d = \log \frac{d_{est}}{d_{gt}}$. $e_d = 0$ if $d_{est} = d_{gt}$. Since the person dataset does not include range data, we do not evaluate e_d for this dataset.

appearance, it shows better performance in estimating region orientations.

Region 3D Localization. Tab. 3.6 shows the average error for 3D region localization tasks. It demonstrates that interactions do help improve the accuracy in localizing regions in 3D.

3.7.2.3 Estimating Objects

Object 2D Detection. Tab. 3.7 shows the average precision for object detection in the image. We follow the criteria used in the PASCAL VOC challenge. In the office dataset, we compute the overall average precision for 5 categories: monitors, mice, keyboards, bottles, and cups. The baseline algorithms are Felzenszwalb et al. *Felzenszwalb et al.* (2010) and Bao and Savarese *Bao and Savarese* (2011a). Since our framework detects objects from multiple images and leverages the interaction between scene components, it enables better object detection performance than *Felzenszwalb et al.* (2010) (which detect objects from each single image separately) and *Bao and Savarese* (2011a) (which does not consider the interactions). Examples of detecting objects are shown in Fig. 3.18.

Object 3D Detection. Tab. 3.8 shows the average precision for object detection in 3D space. Ground truth 3D objects are manually labeled from range data. If the distance between the centroid of a detected 3D object and the centroid of a ground truth object is less than δ (we set $\delta = 2.5/0.1$ meter for car / office dataset), the detected 3D object is counted as true. Since every detected 3D object can be associated to a confidence value, we can generate precision-recall curves and compute the average precision. Due to the metric reconstruction ambiguity, we use ground truth camera poses in this experiment. Our baselines are Hoiem et al. *Hoiem et al.* (2008a) and Bao and Savarese *Bao and Savarese* (2011a). *Hoiem et al.* (2008a) uses the 2D bounding box to estimate the location of an object in 3D. *Bao and Savarese* (2011a) estimates 3D object locations using SSFM without interactions. Our framework shows much better performance over *Hoiem et al.* (2008a) and *Bao and Savarese* (2011a). This demonstrates that the importance of modeling interaction to solve the camera and structure estimation problems.

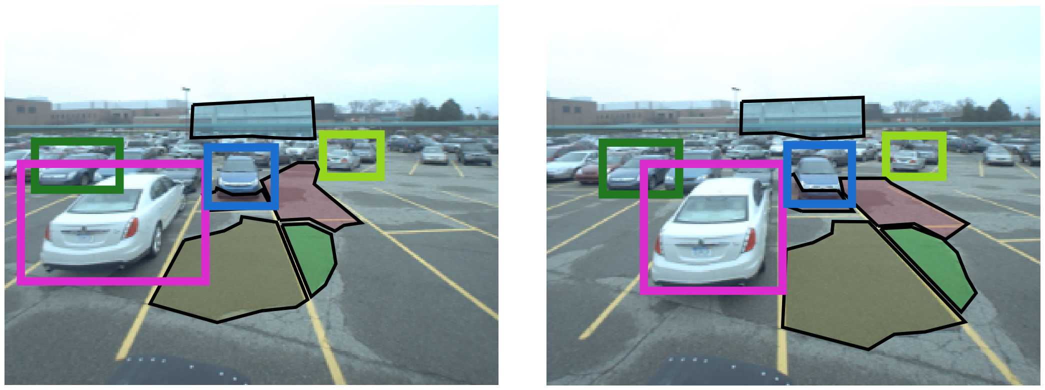
3.7.3 System Analysis

3.7.3.1 System Running Time

We report the running time of our Matlab single-thread implementation in Tab. 3.9. The reported time does not include the time for object detection, feature point detection, and region segmentation, which vary based on the actual algorithms and implementations.

	Felzenszwalb et al. (2010)	*Bao and Savarese* (2011a)	Ours
Car	54.5%	61.3%	**62.8%**
Person	70.1%	75.1%	**76.8%**
Office	42.9%	45.0%	**45.7%**

Table 3.7: 2D object detection average precision.


(a) Detection and matching results of objects and regions. The color of the bounding boxes and regions shows the correspondences of objects and regions.

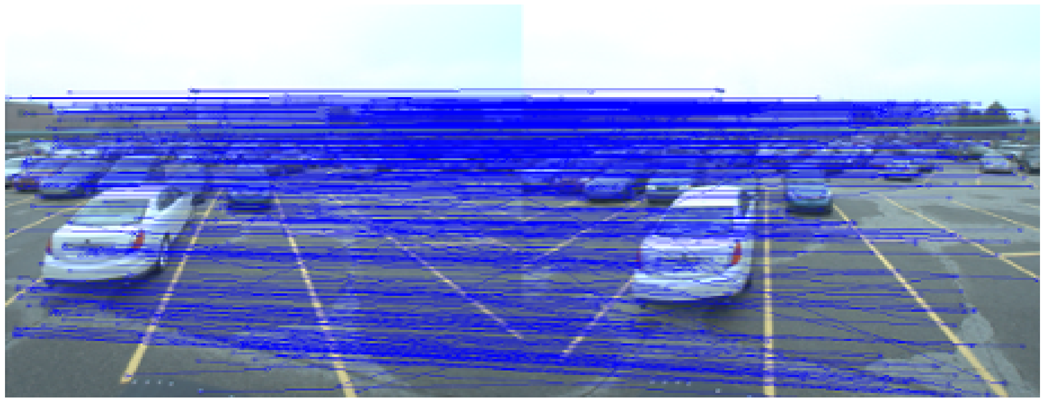

(b) Point matches. Since these images contain many replicate patterns, many matches are wrong.

Figure 3.15: A random image pair in the car dataset.

3.7.3.2 Values of the Potential Functions v.s. Camera Errors

One key implicit assumption of the SSFM model is that the potential terms reach their maximum values when camera configurations and scene components approach their ground truth values. In this analysis, we show the sensitivity of the energy terms in deviating from their maximum values as the camera parameters deviate from their ground truth values. This can be done by plotting average values of several potential functions as function of the camera translation error. Such average values are obtained from potential values calculated from 50 image pairs randomly selected from the car dataset (Fig. 3.15).

3.7.3.3 Estimation Accuracy v.s. Number of Images

SSFM can, in theory, work with an arbitrarily large number of input images. Tab. 3.10 shows the camera pose estimation error and the object detection AP as a function of the number of views (cameras) used to run SSFM. As more cameras are used, SSFM tends to achieve better object detection results and camera translation estimation. Tab. 3.10 also indicates that, given the same number of images, SSFM achieves better results than *Bao and Savarese* (2011a) wherein no interactions are used. This suggests that the ability to model interactions among scene components become particularly useful when more images are used.

	Single image	Without interactions	With interactions
Car	21.4%	32.7%	**43.1%**
Office	15.5%	20.2%	**21.6%**

Table 3.8: Average precision for 3D object detection. “Single image” shows the result of *Hoiem et al.* (2008a). “Without interactions” shows the result of *Bao and Savarese* (2011a). “With interactions” shows the result of our proposed method.

Image Number	2	3	4
Inference Time (min)	~10	~15	~20

Table 3.9: System running time.

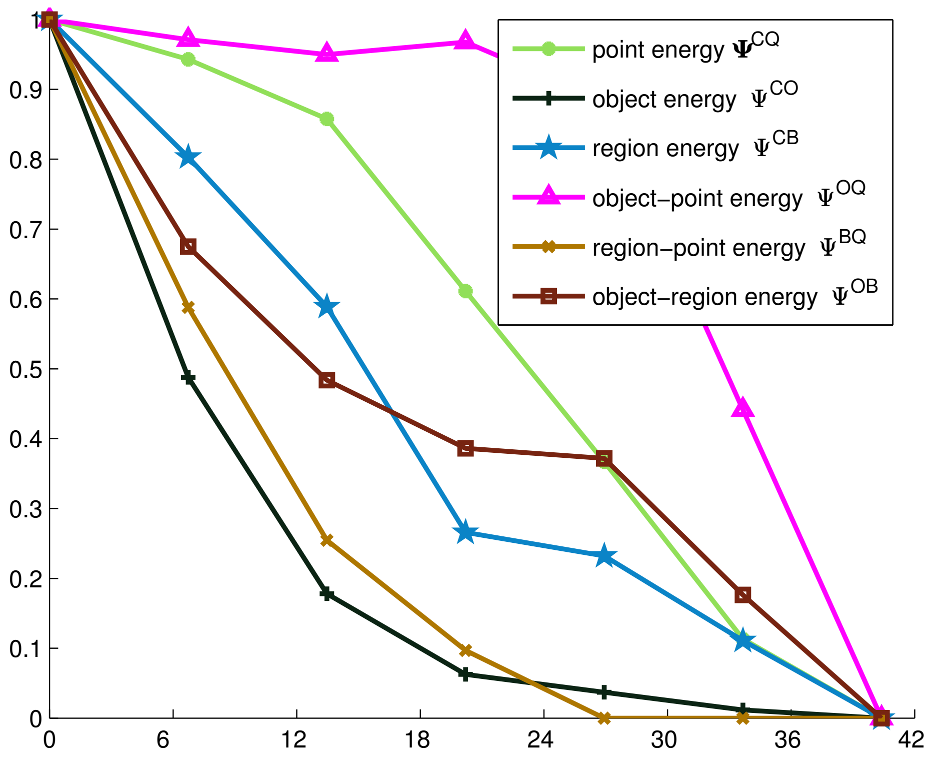


Figure 3.16: Average potential values v.s. camera translation error. The camera rotation is fixed as ground truth value. The X axis is the camera translation e_t in degree (defined in Sec. 3.7.2.1). For visualization purposes, we normalize the potential values to range between 0 and 1. The Y axis shows the normalized values. Notice that, the same translation error value may correspond to different camera configurations. The value for each translation error is the average of 50 random trials.

3.7.3.4 Camera Pose Estimation v.s. Camera Baseline

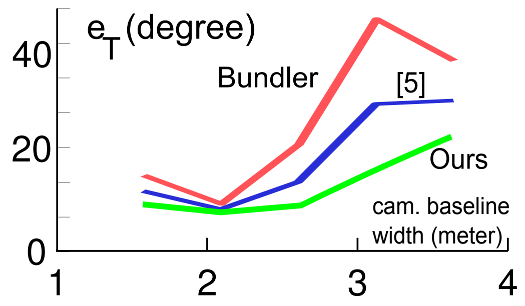
We analyze the effect of the distance between cameras (baseline) in estimating the camera poses. Since the camera rotation estimation of both Bundler and SSFM is already fairly accurate, we only show the translation estimation error v.s. camera baseline width (Fig. 3.17). As Fig. 3.17 and Tab. 3.3 both show, SSFM has a great advantage over point-based SFM algorithm *Snavely et al.* (2008). In Fig. 3.17, we see that this advantage grows as the camera baseline width increases. This fact reinforces our intuition that high-level semantic information is less affected than low-level features by the appearance variation caused by wide camera baseline. As the baseline width between input cameras increases, the matching process of point features becomes less robust and reliable, and hence the point-based SFM starts losing estimation accuracy dramatically. However, our SSFM framework does not surrender much accuracy since it leverages semantic information, including object poses or region orientations, which can still be acquired robustly even if the camera baseline width is high.

3.8 Conclusion

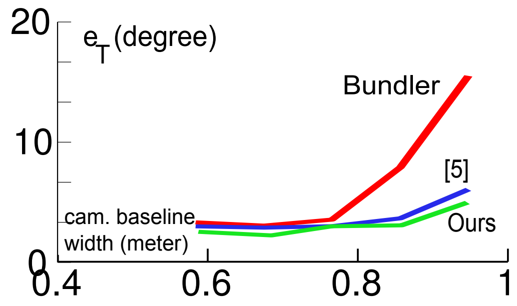
We have proposed a novel scene understanding framework for jointly estimating camera poses and detecting objects, points, and regions from two or multiple semi-calibrated images. We have demonstrated that by modeling the interaction among

Camera #		2	3	4
Det. AP	*Bao and Savarese* (2011a)	61.3%	61.7%	62.6%
	Ours	**62.8%**	**63.8%**	**66.5%**
$\bar{e}_T$	*Bao and Savarese* (2011a)	19.9°	16.2°	13.9°
	Ours	**12.1°**	**10.4°**	**9.2°**

Table 3.10: Camera pose estimation errors and object detection AP v.s. numbers of cameras on the Ford-car dataset. The baseline detector AP is 54.5%.



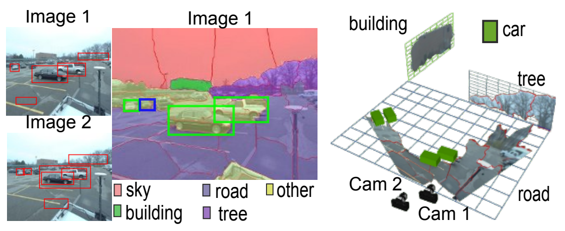
(a) Car dataset



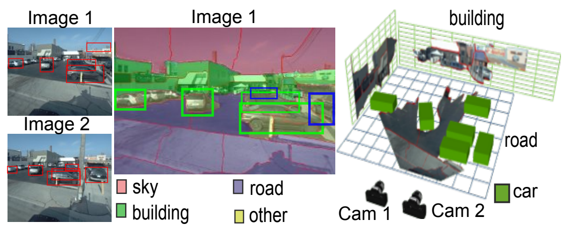
(b) Kinect dataset

Figure 3.17: Camera translation estimation error v.s. baseline width.

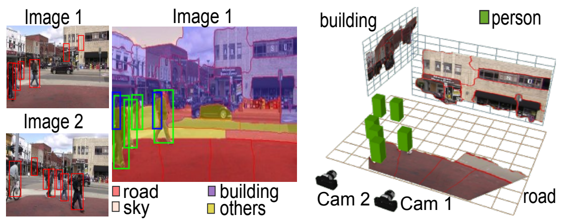
points, regions, and objects, we obtain more accurate results than if these scene elements are estimated in isolation.



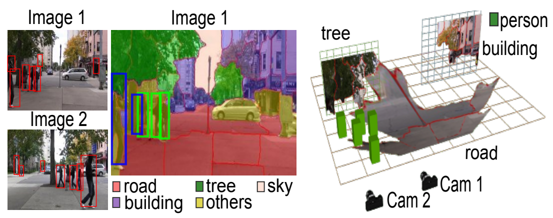
(a) Car



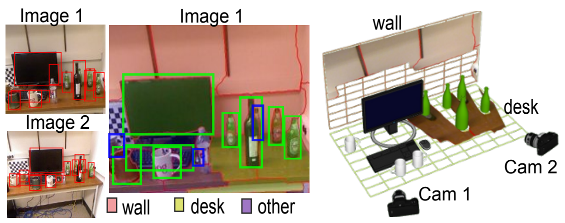
(b) Car



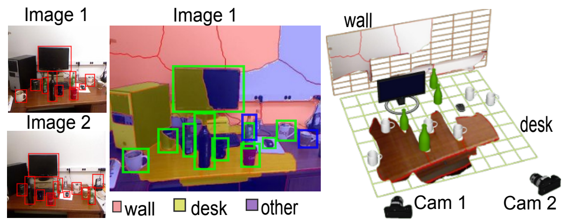
(c) Person



(d) Person



(e) Office



(f) Office

Figure 3.18: Each panel shows result examples of our framework. **Left**: Input image pair and baseline object detection by *Felzenszwalb et al.* (2010). **Middle**: Segmented regions by *Ren and Malik* (2003) (delimited by red boundary), region classes, and improved object detection (green and blue bounding boxes). Notice that many false alarms are removed and missed positive are recovered (blue bounding box). **Right**: Reconstructed 3D scene with objects and regions along with the estimation of the location and pose of cameras. Estimated planes appear perpendicularly w.r.t others because we use *Hoiem et al.* (2007) to initialize 3D plane orientations.

Algorithm III.1 Sampling parameters for searching the solution for Eq. 3.1.

```
1:  Initialize camera configurations (Sec.  3.5.1).
2:  FOR $\mathbf{C} \in$ the set of initial camera configurations
3: $\mathbf{C}_0 = \mathbf{C}$;
4: FOR $n = 1 : M$ ($M$ is predefined)
5: $\mathbf{C}_n = \mathbf{C}_{n-1} + \mathbf{C}'$ where $\mathbf{C}'$ is 0-mean Gaussian r.v.
6: $\mathbb{O}'_n \Leftarrow \arg\max_{\mathbb{O}=\{\mathcal{O}_t\}} \Pi_t \Psi_t^{CO}(\mathcal{O}_t, \mathbf{C}_n; \mathbf{I})$; (Sec.  3.5.2)
7: $\mathbb{Q}'_n \Leftarrow \arg\max_{\mathbb{Q}=\{\mathcal{Q}_s\}} \Pi_s \Psi_s^{CQ}(\mathcal{Q}_s, \mathbf{C}_n; \mathbf{I})$; (Sec.  3.5.3)
8: $\mathbb{B}'_n \Leftarrow \arg\max_{\mathbb{B}=\{\mathcal{B}_r\}} \Pi_r \Psi_t^{CB}(\mathcal{B}_r, \mathbf{C}_n; \mathbf{I})$; (Sec.  3.5.4)
9: $\{\mathbb{O}_n, \mathbb{Q}_n, \mathbb{B}_n\} \Leftarrow \arg\max \prod_{t,s} \Psi_{t,s}^{OQ} \prod_{t,r} \Psi_{t,r}^{OB} \prod_{r,s} \Psi_{r,s}^{BQ}$ by
10: gradient descent starting from $\{\mathbb{O}'_n, \mathbb{Q}'_n, \mathbb{B}'_n\}$ (Sec.  3.6)
11: $\alpha = \frac{\Psi(\mathbb{O}_n, \mathbb{Q}_n, \mathbb{B}_n; \mathbf{C}_n, \mathbf{I})}{\Psi(\mathbb{O}_{n-1}, \mathbb{Q}_{n-1}, \mathbb{B}_{n-1}; \mathbf{C}_{n-1}, \mathbf{I})}$;
12: IF $\alpha < \varrho$ where $\varrho \sim U(0,1)$ (uniform random variable)
13: $\{\mathbb{O}_n, \mathbb{Q}_n, \mathbb{B}_n\} = \{\mathbb{O}_{n-1}, \mathbb{Q}_{n-1}, \mathbb{B}_{n-1}\}$;
14: END
15: END
16:END
```

CHAPTER IV

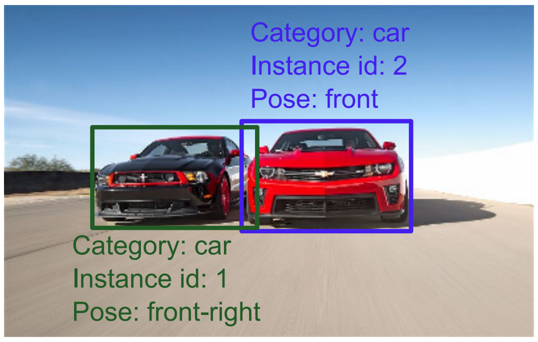
Object Co-detection[1]

4.1 Introduction

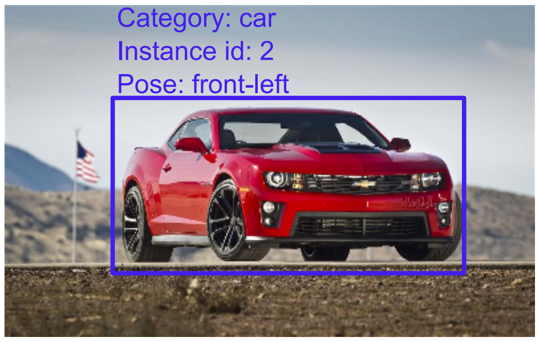
In Chapter III we introduced a scene understanding system, where object information can be used to help solve the camera motion estimation problem. In such a system, detecting and pairing objects from different images is a critical step. In this chapter, we will introduce a framework for solving the problem of simultaneously identifying and matching objects from multiple images. We name this novel problem as *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. 4.1 illustrates co-detection in two images. Fig. 4.1a shows two instances of the car category: a black Ford Mustang and a red Chevy Camaro. Fig. 4.1b also contains a red Camaro, which is considered to be the matching object to the Camaro in Fig. 4.1a. 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

[1]This chapter was partially previous published in *Bao et al.* (2012b).



(a) Input Image #1



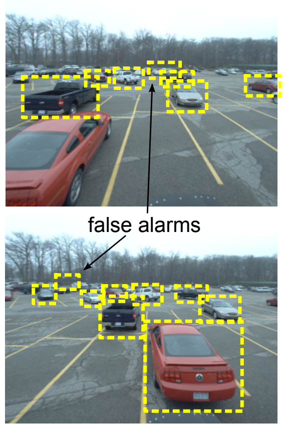
(b) Input Image #2

Figure 4.1: 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.

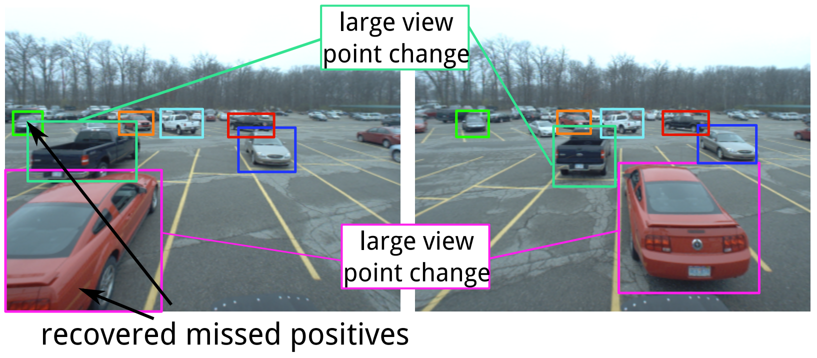
as a detection, but it has no matched object in the other image.

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. 4.2a, 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.4.2a-bottom, but it is not in Fig.4.2a-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-



(a)



(b)

Figure 4.2: Object co-detection improves object detection and matches 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 (a).

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* (2012a)). One possible object representation is shown in Fig. 4.4a. We measure appearance consistency between objects by matching their parts (Fig. 4.4b). 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* (2011a), 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.

4.2 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); *Felzenszwalb et al.* (2010); *Gu and Ren* (2010); *Xiang and Savarese* (2012a); *Su et al.* (2009) 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* (2007); *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* (2011a); *Bao et al.* (2012a); *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); *David* (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.

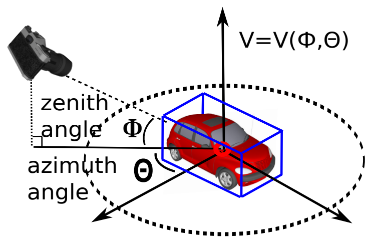
Region matching. Methods such as *Matas et al.* (2004); *Toshev et al.* (2007) match features or regions across views of the same scene. Co-detection is fundamentally different in that it works with high level semantics (i.e. objects). However, co-detection can be helpful for those algorithms since it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

4.3 Object Co-detection Model

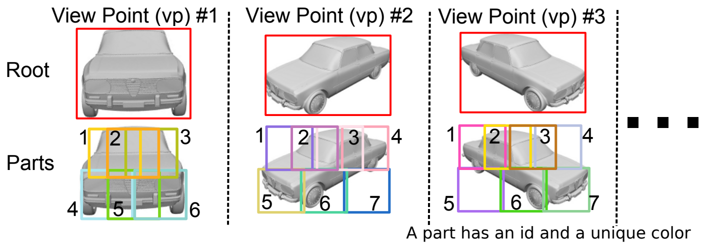
In an object co-detection problem, we are given a total number of K input images $\mathcal{I} = \{I^1, \ldots, I^K\}$. The goal of the co-detector is to detect the matching instances $\mathcal{O} = \{O^1, \ldots, O^K\}$ that simultaneously appear in each of the input image, where O^k is an object instance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

4.3.1 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

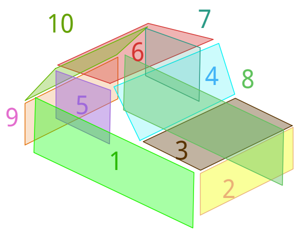


(a)

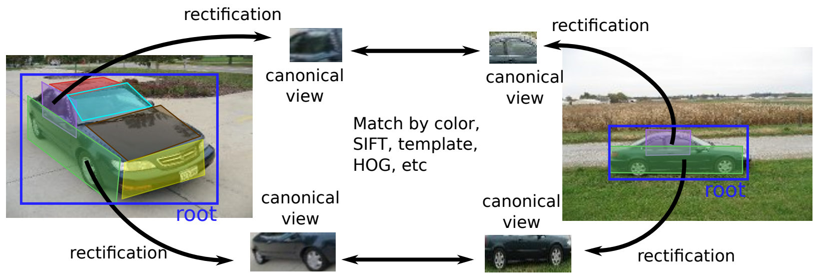


(b)

Figure 4.3: Viewpoint and 2D part representation. (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.



(a)



(b)

Figure 4.4: An example of 3D object 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. 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. 4.3.4).

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. 4.3b). Since different parts are defined for different viewpoints independently, no explicit part correspondence can be established across different viewpoints (Fig. 4.3b). Thus, a 2D representation is only suitable for matching objects observed from very similar viewpoints (e.g. if images are captured by small-baseline stereo cameras). In such a case, parts association can be easily established.

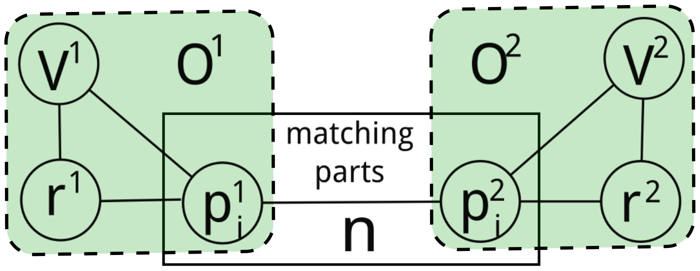


Figure 4.5: Object co-detection model when two images are considered. The dashed green box measures the compatibility between an object and its image (E_{unit}). The middle rectangle measures the similarity of parts of different objects (E_{match}).

In a 3D representation such as *Xiang and Savarese* (2012a); *Su et al.* (2009); *Savarese and Fei-Fei* (2007), the root is specified by a rectangle in the image, and parts are associated to 3D flat surfaces that make up an object (Fig. 4.4a). The viewpoint is denoted by the azimuth and zenith angle of object pose (Fig. 4.3a). The canonical view of a part (Fig. 4.4b) 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. 4.4b). Moreover, a 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.

4.3.2 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. 4.5 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. 4.5 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 $\{I^1 \cdots I^K\}$:

$$E(\mathcal{O},\mathcal{I}) = \sum_{k=1}^{K} E_{\text{unit}}(O^k, I^k) + \sum_{i=1}^{n} E_{\text{match}}(\{p_i^k\}_{k=1}^{K}, \{V^k\}_{k=1}^{K}, \mathcal{I}), \tag{4.1}$$

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:

$$E_{\text{unit}}(O^k, I^k) = E_{\text{root}}(r^k, V^k, I^k) + \sum_{i=1}^{n} E_{\text{part}}(p_i^k, V^k, I^k) + \sum_{i=1}^{n} E_{\text{rp}}(r^k, p_i^k, V^k, I^k), \tag{4.2}$$

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. 4.3.3.

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

$$E_{\text{match}}(\{p_i^k\}_{k=1}^{K}, \{V^k\}_{k=1}^{K}, \mathcal{I}) = \frac{1}{C_K^2} \sum_{k_1,k_2} M(p_i^{k_1}, p_i^{k_2}, V^{k_1}, V^{k_2}, I^{k_1}, I^{k_2}), \tag{4.3}$$

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. 4.4) 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. 4.3.4. 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. 4.1, 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).

4.3.3 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* (2012a). 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* (2012a)). In such models, the category-level detection templates, which encode the visual features (e.g. *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[2]. Sec. 4.3.6 explains how we learn these parameters.

[2]E.g. if the model in *Felzenszwalb et al.* (2010) is applied, we have $\beta_{root} = F_0'$, $\beta_{part}^i = F_i'$, and $\beta_{rp}^i = d_i$ for each part i, where the right-hand terms are defined in Eq. 2 and 3 in paper *Felzenszwalb et al.* (2010).

4.3.4 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* (2012a)) 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 *David* (2004) and pixel intensity values within image patches. If a 3D part representation is applied, we extract such features after rectifying the part into its canonical view (Fig. 4.4b).

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 *David* (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[3]. After considering the part visibility, we use the following vector to represent the similarity between two parts:

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

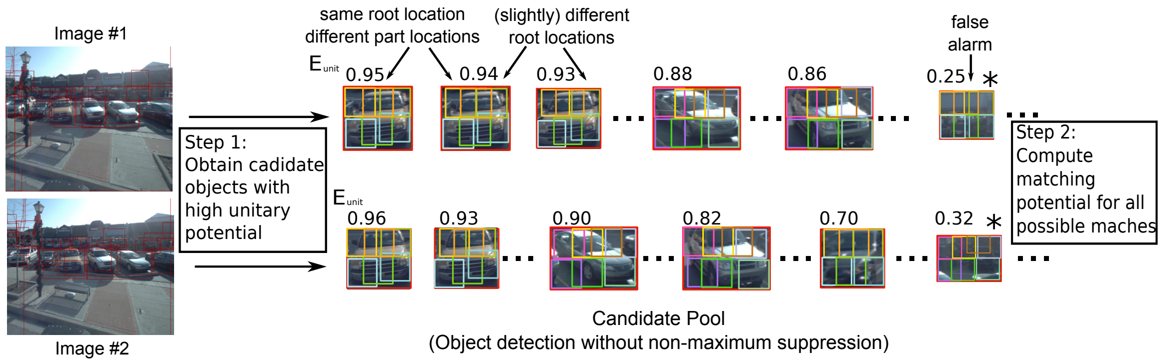


Image #1
Image #2
Step 1: Obtain cadidate objects with high unitary potential
same root location different part locations
(slightly) different root locations
false alarm
E_unit
0.95
0.94
0.93
0.88
0.86
0.25
0.96
0.93
0.90
0.82
0.70
0.32
Step 2: Compute matching potential for all possible maches
Candidate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

Figure 4.6: 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.

$$\mathbf{d}_i^{k_1k_2} = [v_i^{k_1}v_i^{k_2}\mathbf{s}(\phi_i^{k_1}, \phi_i^{k_2})^T, 1-(1-v_i^{k_1})(1-v_i^{k_2})]^T.$$

Note that $\mathbf{d}_i^{k_1k_2}$ is a function of part locations, viewpoints and images. The last term of $\mathbf{d}_i^{k_1k_2}$ accommodates the bias in the case where either part is not visible. We compute the similarity score as

$$M(p_i^{k_1}, p_i^{k_2}, V^{k_1}, V^{k_2}, I^{k_1}, I^{k_2}) = \mathbf{w}_i^T\mathbf{d}_i^{k_1k_2}, \tag{4.4}$$

where $\mathbf{w}_i$ is the matching weight to be learned from a training set. Since $\mathbf{d}_i^{k_1k_2}$ encodes the visibility information, we can learn a universal set of weights $\mathbf{w}_i$ for all the parts under different viewpoints. The procedure for learning $\mathbf{w}_i$ is explained in Sec. 4.3.6.

4.3.5 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.4.1. 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. 4.6 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.

4.3.6 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):

$$\begin{aligned} & \{\beta_{\text{root}}, \beta_{\text{part}}, \beta_{\text{rp}}, \mathbf{w}\} \\ = \; & \arg \min_{\beta_{\text{root}}, \beta_{\text{part}}, \beta_{\text{rp}}, \mathbf{w}} \frac{1}{2}(\|\beta_{\text{root}}\|^2 + \|\beta_{\text{part}}\|^2 + \|\beta_{\text{rp}}\|^2 + \|\mathbf{w}\|^2) + \\ & \lambda \sum_t \max(0, 1 - y_t \max_{\mathcal{P}^t} E(\mathcal{O}^t, \mathcal{I}^t)), \end{aligned} \tag{4.5}$$

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. 4.5 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. 4.5, 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 **w** can be estimated using a standard support vector machine.

4.4 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 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.

4.4.1 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* (2011a) (see Fig. 4.8a), a pedestrian dataset *Ess et al.* (2007) (see Fig. 4.8b), and a 3D object dataset *Savarese and Fei-Fei* (2007) (see Fig. 4.8c and 4.8d). To evaluate object detection accuracy, we follow the criteria in the PASCAL VOC challenge[4] and report average precision (AP). To evaluate pose estimation accuracy, we follow the criteria in *Savarese and Fei-Fei* (2007). Tab. 4.1 shows the object detection results on the car and pedestrian

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

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 4.1: Object detection results using the car dataset *Bao and Savarese* (2011a) and the person dataset *Ess et al.* (2007). DPM indicates the method proposed by *Felzenszwalb et al.* (2010). "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 / person dataset.

		Iron	Mouse	Shoe	Car	Cellphone	Stapler	Bike	Toaster	Mean
Object Detection	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 estimation	AML	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 4.2: Object detection (measured by average precision) and pose estimation (measured by accuracy) results using the 3D object dataset *Savarese and Fei-Fei* (2007). ALM indicates the method proposed in *Xiang and Savarese* (2012a). The reported numbers are percentage.

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. 4.1 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. 4.1 also shows that, if random pairs of images are considered, object co-detection performs on par with single-image

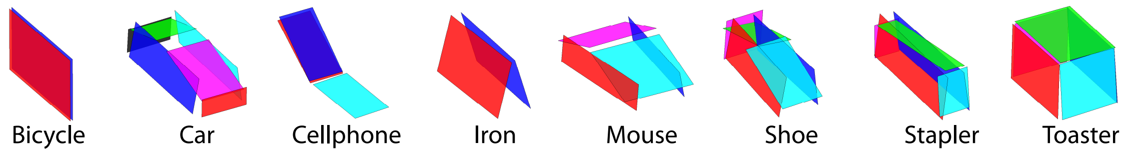


Figure 4.7: The 3D part representation for eight categories in *Xiang and Savarese* (2012a).

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. 4.2 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* (2012a). Examples of a 3D object representations in *Xiang and Savarese* (2012a) are shown in Fig. 4.7. As Tab. 4.2 shows, object co-detection outperforms *Xiang and Savarese* (2012a) 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* (2012a) already shows very strong performance.

4.4.2 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 *David* (2004), which uses generalized Hough voting and homography validation to detect objects. Tab. 4.3 shows the detection accuracy for detecting a labeled instance. Notice that our method achieves a significant improvement over *David* (2004) in that it leverages the learnt categorical structure of object as opposed to *David* (2004) which only relies on low level features and a subsequent geometrical validation step. Tab. 4.4 summarizes the overall accuracy in detecting objects and estimating their pose. The comparison between Tab. 4.4 and Tab. 4.2 allows us to appreciate the superior performance of the co-detector when the object position is available in one of the two images (Tab. 4.4), as opposed to be unknown in both images (Tab. 4.2).

4.4.3 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 4.5 shows, the co-detector obtains the best performances in all the experiments.

Average Precision (%)		Iron	Mouse	Shoe	Car	Cellphone	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 4.3: Single instance detection result using the 3D object dataset. Same / Different Pose: the azimuth angle (Fig. 4.3a) of an object in a query image is the same / different as the the azimuth angle of the labeled object. SIFT indicates the method proposed by *David* (2004).

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

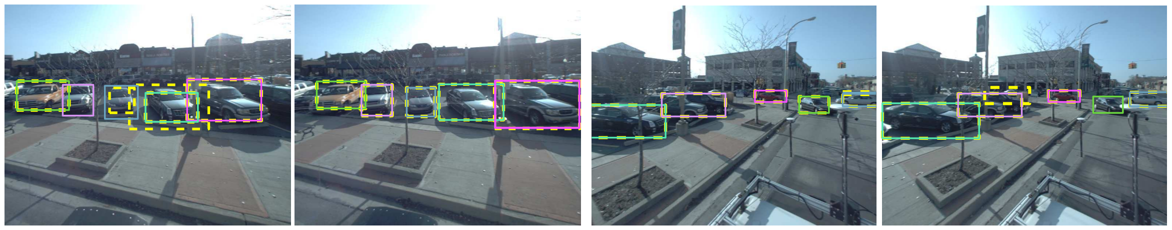
Table 4.4: Single instance detection results. See Tab. 4.2 for a comparison.

4.4.4 Matching Images by Objects

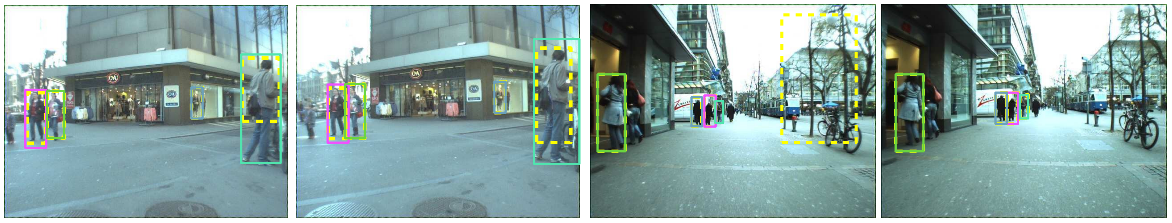
In this experiment, the goal is to match images if they contain the same object instance. Unlike the previous experiment, the locations of objects are not given in any of the images. For each trial, we have 5 candidate images and 1 target image. Each image contains one object. The goal is to find among 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. 4.6. We also apply image matching methods to match the bounding box of the most likely detection returned by *Xiang and Savarese* (2012a), 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. 4.4.3. Our co-detection model achieves superior performance in all the experiments.

4.5 Conclusion

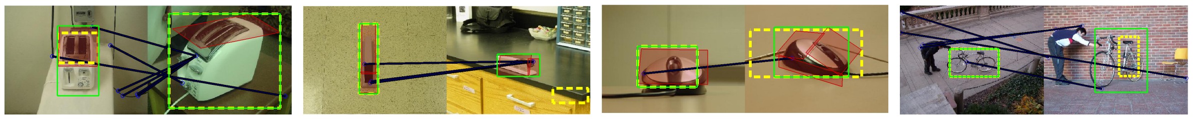
In this chapter, we introduced the problem of object co-detection, solving which is greatly helpful for solving the scene understanding problem introduced in Chapter III. We proposed a novel framework for solving it, and shown that our framework, by leveraging state-of-the-art part-based object representations, 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.



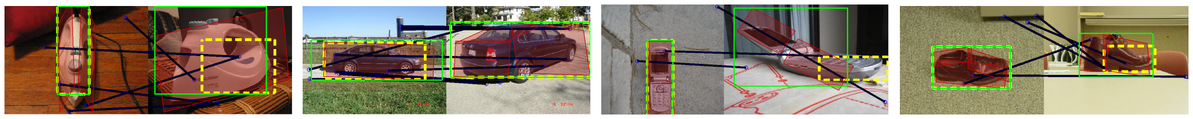
(a) Car dataset *Bao and Savarese* (2011a).



(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 4.8: 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. 4.8c and 4.8d: detected parts are highlighted in red. The blue lines are SIFT matches obtained by threshold test where the threshold is 0.7.

Accuracy %		Iron	Mouse	Shoe	Car	Cellphone	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**
Diffe-rent 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 4.5: 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*David* (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.

Accuracy %		Iron	Mouse	Shoe	Car	Cellphone	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**
Diffe-rent 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 4.6: 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*David* (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* (2012a). See Tab. 4.5 for a comparison.

CHAPTER V

Dense Object Reconstruction[1]

5.1 Introduction

Recent years have seen rapid strides in 3D shape reconstruction, with multiview stereo (MVS) systems capable of reconstructing entire monuments *Furukawa et al.* (2010); *Goesele et al.* (2010). Despite this progress, MVS has remained largely applicable only in favorable imaging conditions. Lack of texture leads to extended troughs in photoconsistency-based cost functions, while specularities violate inherent Lambertian assumptions. Diffuse photoconsistency is not a reliable metric with wide baselines in scenarios with few images, leading to sparse, noisy MVS outputs. Under these circumstances, MVS reconstructions often display holes or artifacts (see Figure 5.1 dashed box).

On the other hand, there have been crucial developments in two seemingly disjoint areas of computer vision. With the advent of cheap commerical scanners and depth sensors, it is now possible to easily acquire 3D shapes. Concurrently, the performance of modern object detection algorithms *Dalal and Triggs* (2005); *Felzenszwalb et al.* (2010); *Leibe et al.* (2006); *Xiang and Savarese* (2012b) has rapidly improved to allow inference of reliable bounding boxes in the presence of clutter, especially when information is shared across multiple views. This chapter presents a framework for dense

[1]This chapter was partially previous published in *Bao et al.* (2013).

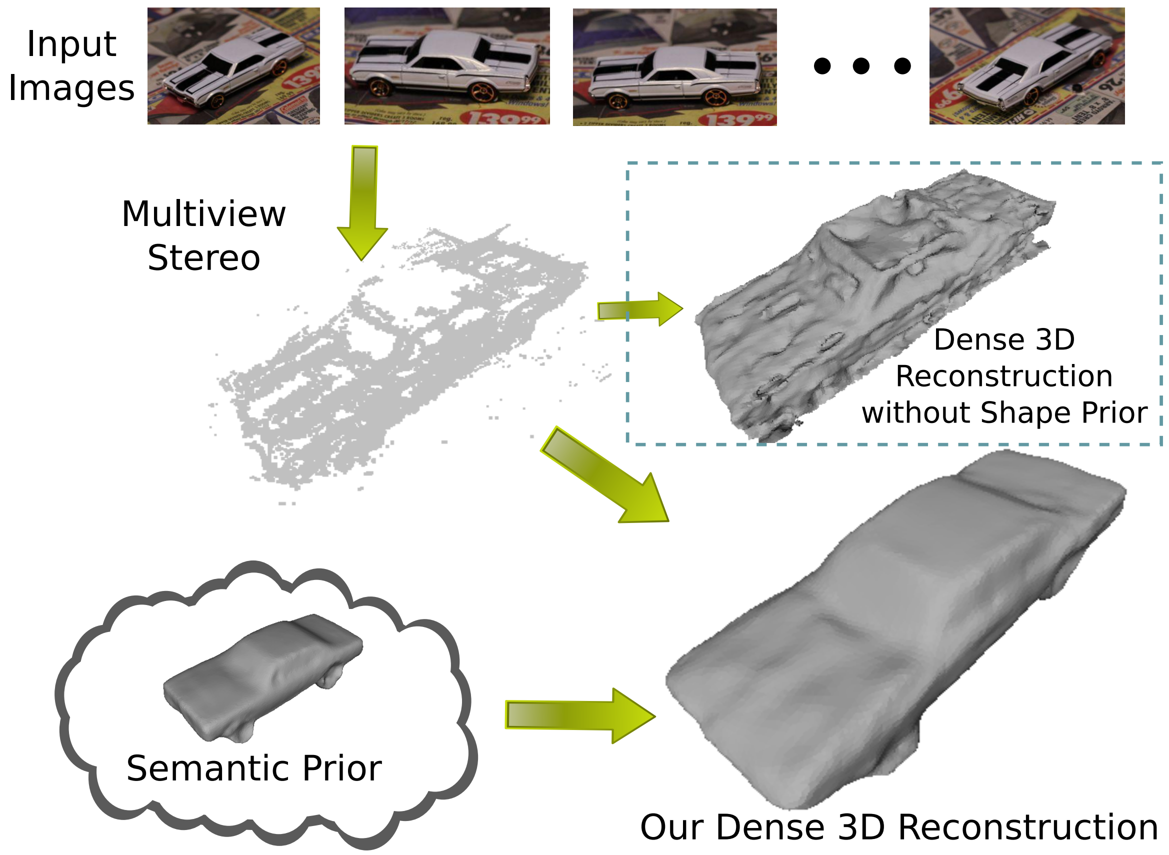

Input Images
Multiview Stereo
Dense 3D Reconstruction without Shape Prior
Semantic Prior
Our Dense 3D Reconstruction

Figure 5.1: Traditional multiview stereo faces challenges due to lack of texture, wide baselines or specularities. We propose a framework for semantic dense reconstruction that learns a category-level shape prior, which is used with weighted warping and refinement mechanisms to reconstruct regularized, high-quality 3D shapes.

3D reconstruction that overcomes the drawbacks of traditional MVS by leveraging semantic information in the form of object detection and shape priors learned from a database of training images and 3D shapes.

The aforementioned drawbacks of MVS have been widely recognized and several prior works share our philosophy of augmenting reconstruction with prior knowledge. For instance, *Furukawa et al.* (2009a) reconstruct indoor environments by incorporating Manhattan priors, *Gallup et al.* (2010) recover urban façades with a piecewise planar assumption and *Wu et al.* (2012) recover building models with a prior derived from architectural schematic curves. All the above approaches use application-specific information to provide the shape priors.

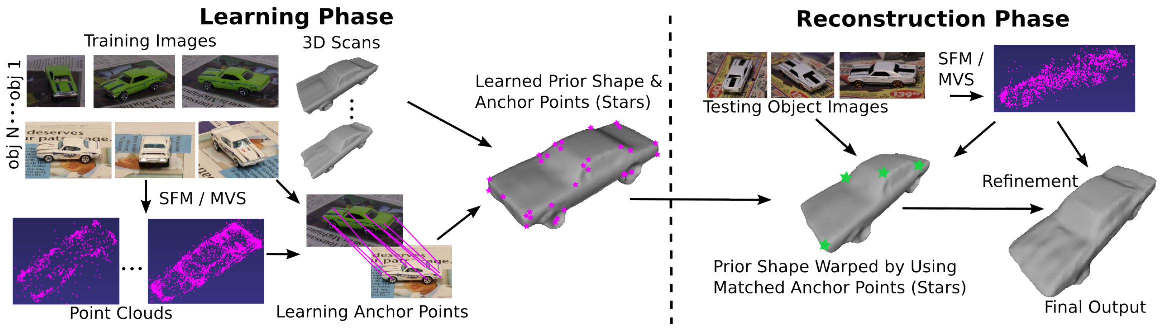


Figure 5.2: Outline of our semantic dense reconstruction framework. Please see Section 5.1 for an overview.

In contrast, our priors are far more general – they are category-level and learned from training data. An overview of our reconstruction framework is shown in Figure 5.2. We postulate in Section 5.3 that while object instances within a category might have very different shapes and appearances, they share certain similarities at a semantic level. For example, both sedans and sports cars have bonnets and wheels. We model semantic similarity as a shape prior, which consists of a set of automatically learned anchor points across several instances, along with a learned mean shape that captures the shared commonality of the entire category. Our experiments demonstrate that this novel representation can successfully achieve the balance between capturing semantic similarities and shape variation across instances.

In the learning phase (Section 5.4), the anchor points encode attributes such as frequency, appearance and location similarity of features across instances. The associated weights aid in discarding spurious texture matches, while determining a weighted regularization for both mean shape learning and reconstruction. Based on matched anchor points, the shape prior for a category is determined by a series of weighted thin-plate spline (TPS) warps over the scans of training objects.

Our reconstruction phase (Section 5.5) starts with a point cloud obtained by applying a structure-from-motion (SFM) or MVS system to images of an unseen instance (with a shape different from training objects). Bounding boxes from object detection in individual images are collated using the SFM camera poses and used to localize and orient the object in the point cloud. This guides the process of matching anchor points – shown by green stars in right panel in Figure 5.2 – between the learned prior and the test object's SFM point cloud, followed by a warping of the prior shape in order to closely resemble the true shape. Finer details not captured by the shape prior may be recovered by a refinement step, using guidance from SFM or MVS output. The refinement combines confidence scores from anchor points and photoconsistency in order to produce a regularized, high quality output shape. Not only are our reconstructions visually pleasant, they are also quantitatively closer to the ground truth than other baselines (Section 5.6).

5.2 Relation to Prior Work

Our comprehensive reconstruction pipeline relates to several areas of computer vision, as briefly explored in this section.

Multiview Stereo. This paper provides a framework to augment traditional multi-view stereo (MVS) reconstruction methods with semantic information. Broadly, MVS approaches in computer vision may be categorized as patch-growing, depth-map based and volumetric methods. The former uses a locally planar patch model to perform a succession of expansion steps to maximize photoconsistency and filtering steps to remove inaccurate patches *Furukawa and Ponce* (2010). Depth map-based methods seek a labeling from the space of pixels to a set of discrete depth labels *Kolmogorov and Zabih* (2002). Volumetric methods, on the other hand, seek a binary partitioning of 3D space into object and non-object *Hernández and Vogiatzis* (2010); *Vogiatzis et al.* (2007). We choose the patch-based system *Furukawa and Ponce* (2010) for demonstration, but our framework can be generalized to other approaches too.

Example-Based Reconstruction. A set of example shapes is used by active shape models (ASM) to encode patterns of variability, thereby ensuring a fitted shape consistent with deformations observed in training *Cootes et al.* (1995). However, it requires heavy manual annotation and only models linear variations. While reasonable in 2D, it is arguably not well-suited for the far higher shape and appearance variations in general 3D scenes. Subsequent works on statistical shape analysis *Dryden and Mardia* (1998) allow non-rigid TPS warps between shapes *Bookstein* (1989), but often require landmark identification and initial rigid alignment based on point distributions, which is not feasible for general scenes *Munsell et al.* (2008). We use semantic information, namely object detection for localization and anchor point matching, to overcome those drawbacks. Learned anchor points yield confidence scores, which guide our deformation process through a weighted TPS *Rohr et al.* (1996).

Morphable models in 3D demonstrate realistic shape recovery, but are limited to categories like faces with low shape variation that can be accurately modeled with a linear PCA basis *Blanz and Vetter* (1999). Pauly et al. propose a framework for example-based 3D scan completion, but require dense 3D scans *Pauly et al.* (2005). By exploiting semantics in the form of object detection and anchor point matching, we handle both greater shape variation and noisy, incomplete, image-based MVS inputs.

Shape Matching. Determining correspondence across instances with varying shape is a key step in shape matching. Belongie et al. pose correspondence search as a bipartite matching problem with shape context descriptors *Belongie et al.* (2002), Berg at al. find points with similar geometric blur descriptors by solving an integer quadratic program*Berg et al.* (2005), while Chui and Rangarajan's TPS-RPM determines matches with a soft assign *Chui and Rangarajan* (2003). A 3D CAD model is aligned to images in *Leotta and Mundy* (2009), but the model and features are manually defined. The demands on correspondences for 3D reconstruction are far higher than 2D shape matching – competing factors like high localization accuracy, stringent outlier rejection and good density are all crucial to obtaining a high quality dense reconstruction. Algorithms V.1 and V.2 are designed to robustly meet these challenges.

Object Detection and 3D Information. The mutual benefit of combining object detection and SFM is demonstrated in *Bao and Savarese* (2011a). The flexibility of implicit shape-based detection frameworks *Leibe et al.* (2006) is used to transfer depth annotations from training images to test objects in *Thomas et al.* (2007); *Sun et al.* (2010). TPS-RPM is combined with Hough voting to localize object boundaries in *Ferrari et al.* (2007). Object recognition is improved in *Jiang et al.* (2009) by computing deformation priors directly in transformation space. However, the complexity of 3D shapes and the accuracy demands of 3D reconstruction necessitate far greater control over the deformation process, so we consider it advantageous to compute priors in the mesh space.

5.3 Our Model

We assume that for each object category, there exists a prior that consists of a 3D *mean shape* $\mathbf{S}^*$ that captures the commonality of shapes across all instances and a set of *anchor points* $\mathbf{A}$ that captures similarities between subsets of instances. The shape of any particular object $\mathbf{S}^i$ is a transformation of $\mathbf{S}^*$, plus specific details Δ^i not shared by other instances:

$$\mathbf{S}^i = T(\{\mathbf{S}^*, \mathbf{A}\}, \theta^i) + \Delta^i, \tag{5.1}$$

where T is a warping (transformation) function and θ^i is the warping parameter that is unique to each object instance. In the following, we briefly explain the various aspects of our model.

Anchor points. The key to reconstructing an object instance is to estimate the warping parameters θ^i. We leverage on certain reliable features associated with the shape prior, which we call anchor points. Anchor points form the backbone of our framework, since they are representative of object shape and the relative importance of different object structures. Anchor points with high weights, ω, are considered stable in terms of location and appearance, and thus, more representative of object shape across instances. They guide the learning of the mean shape for a category, as well as the deformation processes during actual 3D reconstruction. In Section 5.4.1, we detail the mechanism of learning anchor points from training data.

Warping function. We assume that the functional form of T is known. In particular, prior work on shape matching *Belongie et al.* (2002); *Jiang et al.* (2009) has demonstrated inspiring results using regularized thin-plate spline (TPS) transformations *Bookstein* (1989) to capture deformations. Let $\{\mathbf{x}_i\}$ and $\{\mathbf{x'}_i\}$, $i = 1, \cdots, n$, be two sets of anchor points for object instances O and O'. The TPS mapping T is given by

$$T(\mathbf{x}, \{\boldsymbol{\alpha}, \boldsymbol{\beta}\}) = \sum_{j=0}^{3} \alpha_j \phi_j(\mathbf{x}) + \sum_{i=1}^{n} \beta_i U(\mathbf{x}, \mathbf{x}_i) \tag{5.2}$$

where $\phi_0(\mathbf{x}) = 1$, $\phi_j(\mathbf{x}) = x_j$ and $U(\mathbf{x}, \mathbf{x}_i) = \|\mathbf{x} - \mathbf{x}_i\|$. Note that our TPS representation is in 3D, instead of the more common 2D representation in traditional shape matching. The solution for the parameters $\theta = \{\boldsymbol{\alpha}, \boldsymbol{\beta}\}$ in a regularized framework is given by the system of equations:

$$(\mathbf{K} + n\lambda\mathbf{I})\boldsymbol{\beta} + \boldsymbol{\Phi}\boldsymbol{\alpha} = \mathbf{x}', \quad \boldsymbol{\Phi}^\top \boldsymbol{\beta} = \mathbf{0}, \tag{5.3}$$

where $K_{ij} = U(\mathbf{x}_i, \mathbf{x}_j)$, $\boldsymbol{\Phi}_{ij} = \phi_j(\mathbf{x}_i)$ and λ is a regularization parameter. Regularized TPS yields a solution that interpolates between two point sets and is sufficiently smooth. However, greater control is required for 3D reconstruction applications, since the extent of deformations must be determined by the local level of detail. Semantic information of this nature is determined automatically in our framework by the anchor point learning mechanism. To incorporate semantic information from anchor points, in the form of a weight matrix $\mathbf{W} = \mathrm{diag}(\omega_1, \cdots, \omega_n)$, we use an extension of TPS *Rohr et al.* (1996):

$$(\mathbf{K} + n\lambda\mathbf{W}^{-1})\boldsymbol{\beta} + \boldsymbol{\Phi}\boldsymbol{\alpha} = \mathbf{x}', \quad \boldsymbol{\Phi}^\top\boldsymbol{\beta} = \mathbf{0}, \tag{5.4}$$

which is again solvable analytically like regularized TPS.

Unique Details. Details specific to each object that are not captured in the shape prior are recovered by a refinement step. This refinement is used in both mean shape learning and during reconstruction of a particular test object.

To refine a shape $\mathbf{S}^i$ (a mesh) towards shape $\mathbf{S}^j$, we compute displacements for vertices in $\mathbf{S}^i$. For a vertex $\mathbf{p}_k^i$ in $\mathbf{S}^i$, we estimate the surface normal $\mathbf{n}_k^i$ by a local tangent space computation. The vertex $\mathbf{p}_k^i$ is matched to $\mathbf{p}_k^j$ in $\mathbf{S}^j$ if $\|\mathbf{p}_k^j - \mathbf{p}_k^i\| < \tau_1$ and $|(\mathbf{p}_k^j - \mathbf{p}_k^i)^\top \mathbf{n}_k^i| < 1 - \tau_2$, where τ_1, τ_2 are predefined thresholds. Let $\mathcal{P}^i$ be the set of vertices in $\mathbf{S}^i$ that can be matched as above to the set $\mathcal{P}^j$ in $\mathbf{S}^j$ and $\mathcal{N}_k^i$ be the set of 1-nearest neighbors of $\mathbf{p}_k^i$ in $\mathcal{P}^i$. Then, the set of displacements, $\Delta^i = \{\mathbf{d}_k^i\}$, for $1 \leq k \leq |\mathcal{P}^i|$, are computed by minimizing:

$$\sum_{\mathbf{p}_k^i \in \mathcal{P}^i} \epsilon_k^i (\mathbf{d}_k^i - (\mathbf{p}_k^j - \mathbf{p}_k^i))^2 + \mu \sum_{\mathbf{p}_k^i \in \mathbf{S}^i} \sum_{\mathbf{p}_l^i \in \mathcal{N}_k^i} (\mathbf{d}_k^i - \mathbf{d}_l^i)^2, \tag{5.5}$$

where ϵ_k^i is a weight factor. The above cost function encourages the refined shape to lie closer to $\mathbf{S}^j$, while minimizing the local distortion induced by such displacement. The parameter μ is empirically determined for the training set. Note that (5.5) represents an extremely sparse linear system that can be solved efficiently. The vertices of the refined shape are obtained as $\mathbf{p}_k^i + \mathbf{d}_k^i$ and it inherits the connectivity of $\mathbf{S}^i$.

In the above, we are purposefully vague on the representation for the shape $\mathbf{S}^j$.

This is because the above mechanism can be used, with minor changes, for both mean shape learning with the shape $\mathbf{S}^j$ being a mesh and for reconstruction with $\mathbf{S}^j$ being the oriented point cloud output of MVS, as elaborated in Sections 5.4.2 and 5.5.2, respectively.

5.4 Learning Reconstruction Priors

For each object category, we use a set of object instances $\{O^n\}$ to learn a mean shape $\mathbf{S}^*$ and a set of anchor points $\mathbf{A}$. For each object instance O^i in this training set, we capture a set of images $\mathbf{I}^i$ and use a 3D scanner to obtain a detailed 3D shape $\mathbf{S}^i_{\text{scan}}$. Given $\mathbf{I}^i$, we use a standard SFM pipeline to reconstruct a point cloud $\mathbf{S}^i_{\text{sfm}} = \{\mathbf{p}^i_j\}$, where $\mathbf{p}^i_j$ is a 3D point. We manually label a small number of SFM points, $\mathbf{M}^i = \{\mathbf{p}^i_1, \mathbf{p}^i_2, \cdots, \mathbf{p}^i_m\}$ (see the stars in Figure 5.3 and 5.4). The labelled points $\mathbf{M}$ are used to align the scanned shapes $\{\mathbf{S}^i_{\text{scan}}\}$ and their reconstructed point clouds $\{\mathbf{S}^i_{\text{sfm}}\}$ in our training dataset. They also serve as the initialization for the anchor point learning, as described in the following.

5.4.1 Learning Anchor Points

An anchor point, $A = \{\mathbf{\Gamma}, \boldsymbol{\chi}, \omega\}$, consists of a feature vector $\mathbf{\Gamma}$ that describes appearance, the 3D location $\boldsymbol{\chi}$ with respect to the mean shape and a scalar weight ω. $\mathbf{\Gamma}$ is the aggregation of HOG features *Dalal and Triggs* (2005) in all images where A is visible and of every object where A exists. For an anchor point A, if $\mathcal{V}$ are the indices of objects across which the corresponding SFM points are matched and Ω^i are the indices of images of O^i where A is visible, the corresponding feature vector is:

$$\mathbf{\Gamma} = \{\{\mathbf{f}^i_{k^i}\}_{k^i \in \Omega^i}\}_{i \in \mathcal{V}}. \tag{5.6}$$

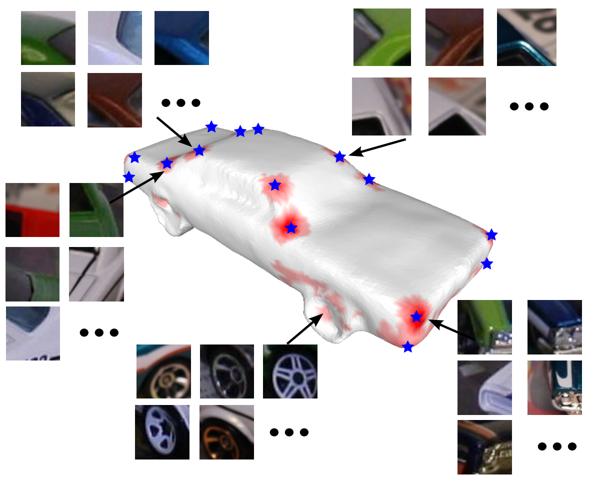
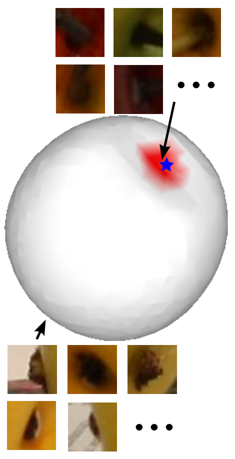

(a) Car (b) Fruit

Figure 5.3: Learned mean shape and anchor points density. Darker red indicates greater density of anchor points. For cars, most anchor points are located around wheels and body corners since those parts are shared across instances. For fruits, anchor points are distributed around the stem and bottom. Blue stars show initially labelled points and the rest are learned by the proposed method. We also show image patches associated with the features of a few example anchor points.



where $\mathbf{f}_{k^i}^i$ is the HOG feature of the image point associated with A in image $I_{k^i}^i$. Let $\mathbf{p}_j^i$ be the locations of the corresponding 3D points, normalized with respect to object centroid and scale. Then, the location for the anchor point is

$$\boldsymbol{\chi}_j = \frac{1}{|\mathcal{V}|} \sum_{i \in \mathcal{V}} \mathbf{p}_j^i. \tag{5.7}$$

The weight ω reflects "importance" of an anchor point. We consider an anchor point important if it appears across many instances, with low position and appearance variance. That is,

$$\omega = w_x \, w_a \, w_f \tag{5.8}$$

where $w_x = \exp(-\frac{\sum_{i \neq k} \|\mathbf{p}^i - \mathbf{p}^k\|}{\sigma_x N_2})$, $w_a = \exp(-\frac{\sum_{i \neq k} d^{i,k}}{\sigma_a N_2})$ and $w_f = \log|\mathcal{V}|$ encode location stability, appearance similarity and instance frequency, respectively. N_2 is the number of combinations. The coefficients σ_a and σ_x are determined empirically from training

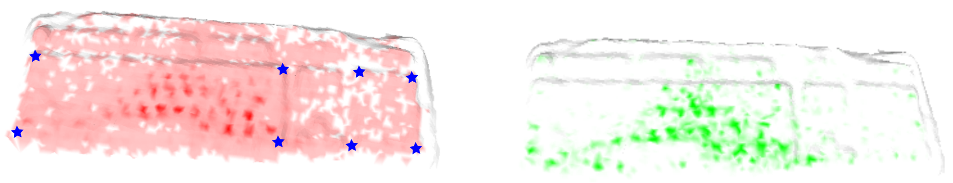

(a) Density (b) Weights

Figure 5.4: Learned shape prior and anchor points for keyboard category. (a) Density of anchor point distribution. Blue stars show the initially labelled anchor points. (b) Learned weights of anchor points. Deeper color means higher weight.

data for each category. In the above,

$$d^{i,k} = \min_{l^i \in \Omega^i, l^k \in \Omega^k} \left(|\mathbf{f}^i_{l^i} - \mathbf{f}^k_{l^k}| \right), \quad \text{for } i \neq k, \tag{5.9}$$

where Ω^i is the set of images of O^i where the point is visible.

In contrast to applications like shape matching, the quality of dense reconstruction is greatly affected by the order and extent of deformations. Thus, the learned anchor point weights ω are crucial to the success of dense reconstruction. Note that while ASM frameworks also associate a weight with landmark points, they are computed solely based on location uncertainty. By encoding appearance similarity and instance frequency, we impart greater semantic knowledge to our reconstruction stage.

The key precursor to learning anchor points is matching 3D points across instances, which is far from trivial. Besides within-class variation, another challenge is the fact that most SFM points correspond to texture. Such points usually dominate an SFM point cloud, but do not generalize across instances since they do not correspond to the object shape, thus, may not be anchor point candidates. Moreover, the density of anchor points cannot be too low, since they guide the deformation process that computes the mean shape and fits it to the 3D point cloud. To ensure the robustness of anchor point matching and good density, we propose an iterative algorithm, detailed in Algorithm V.1. The distribution and weights of the learned anchor points are visualized in Figure 5.3 and 5.4.

Algorithm V.1 Learning anchor points

```
Set Parameters $\delta_f$, $\delta_p$.
For objects $O^i$, $i \in [1, N]$, label $m$ points to get $\mathbf{M}^i$.
Use $\mathbf{M}^i$ to align $\mathbf{S}^i_{\text{sfm}}$ with $\mathbf{S}^i_{\text{scan}}$.
$\forall \mathbf{p}^i_j \subset \mathbf{M}^i$, find $A_j = \{\mathbf{\Gamma}_j, \boldsymbol{\chi}_j, \omega_j\}$ using (5.6), (5.7), (5.8).
Initialize $\mathbf{A} = \{A_j\}$, $j = 1, \cdots, m$.
while anchor point set $\mathbf{A}$ is updated do
    for $i = 1 : N$ do
        Solve $\theta = \arg\min \sum_k \|T(\mathbf{p}^i_k, \theta) - \boldsymbol{\chi}_k\|$.
        Warp SFM point cloud $\mathbf{S}^i_{\text{sfm}} \leftarrow T(\mathbf{S}^i_{\text{sfm}}, \theta)$.
    end for
    for all $\mathbf{p}^i_k \in \mathbf{S}^i_{\text{sfm}}$ do
        for all $\mathbf{p}^j_l \in \mathbf{S}^j_{\text{sfm}}$, where $j \neq i$ do
            if $d(\mathbf{f}^i_k, \mathbf{f}^j_l) < \delta_f$ and $\|\mathbf{p}^i_k - \mathbf{p}^j_l\| < \delta_p$ then
                Match $\mathbf{p}^i_k$ to $\mathbf{p}^j_l$.
            end if
        end for
    end for
    Filter conflicting matches.
    Identify sets of matched SFM points $\mathbf{B}_h$, $h \in [1, H]$.
    for $h = 1 : H$ do
        Find $A_h = \{\mathbf{\Gamma}_h, \boldsymbol{\chi}_h, \omega_h\}$ using (5.6), (5.7), (5.8).
    end for
    Update $\mathbf{A} = \mathbf{A} \cup \{A_h\}$, for $h = 1, \cdots, H$.
end while
Output: denser anchor point set $\mathbf{A}$.
```

5.4.2 Mean Shape Construction

The learned anchor points are used to compute a mean shape for an object category. Recall that we have a mapping from the set of anchor points to each instance in the training set. Thus, we can warp successive shapes closer to a mean shape using the anchor points. The mean shape is constructed by combining these aligned and warped shapes of different instances. Since there are multiple shape instances, the order of combining them is a critical design issue, because improperly combining dissimilar shapes may introduce severe artifacts. To determine the order for combining shapes, we first measure the pairwise similarity between all pairs of training instances. In our experiments, we use the weighted number of commonly matched anchor points as the similarity cue. Given the pairwise similarities, we use hierarchical clustering to group the shapes. The similarity relationships can be represented as a binary tree where each leaf node is an object. We combine the warped shapes $T(\mathbf{S}^i_{\text{scan}})$ following the order of merging successive branches, to eventually obtain a single shape $\mathbf{S}^*$, which represents the commonality of all training instances. We use $\mathbf{S}^*$ as the mean shape. The mean shape learning procedure is shown for a subset of the car dataset

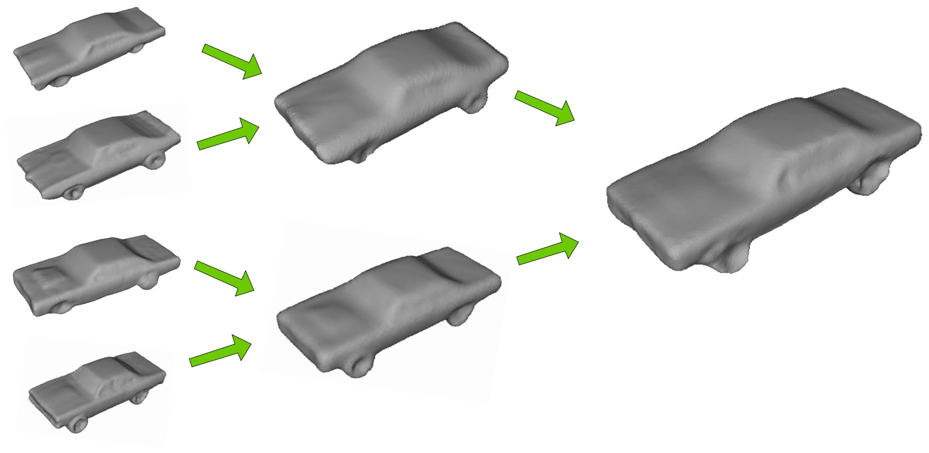


Figure 5.5: The mean shape computation proceeds by systematic combination of training instances, based on a binary tree traversal. The leaf nodes of the tree are the individual training instances, with assignments based on a pairwise shape similarity computation followed by hierarchical clustering. Note that unique details are lost, while features representative of the entire class are preserved.

in Fig. 5.5. Note that $\mathbf{S}^*$ is computed by using the warped training examples, where the warping maps the 3D locations of learned anchor points. Thus, the prior shape is always aligned with the anchor points.

In the above, the warp $T(\mathbf{S}^i_{\text{scan}}) \rightarrow \mathbf{S}^j_{\text{scan}}$, with $i < j$ according to the above defined ordering, is computed as the weighted thin plate spline transformation given by (5.4). Two shapes aligned by anchor points are eventually combined into a single one using displacement vectors computed by minimizing (5.5). The learned mean models for car, fruit and keyboard categories are shown in Figs. 5.3 and 5.4.

5.5 Semantic Reconstruction with Shape Priors

Given a number of images of an object O, we can reconstruct its 3D shape by warping the learned prior shape $\mathbf{S}^*$ based on the estimated θ and by recovering Δ in (5.1) subsequently. The reconstruction consists of three steps: matching anchor points, warping by anchor points, and refinement. Accurately recovering warp parameters θ requires accurate matches between anchor points in $\mathbf{S}^*$ and SFM points in $\mathbf{S}_{\text{sfm}}$. This is facilitated by an initial coarse alignment between $\mathbf{S}^*$ and $\mathbf{S}_{\text{sfm}}$.

5.5.1 Initial Alignment

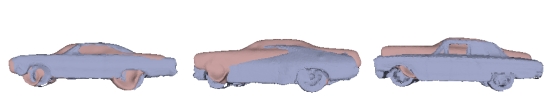
It is conventional in shape modeling literature to compute shape alignments using Procrustes analysis or ICP *Cootes et al.* (1995). However, reconstructed SFM point clouds are typically sparse, contain several outliers and the point set of the object of interest might be dominated by background clutter. The second semantic component of our framework, object detection, is used to alleviate these issues for initial alignment.

State-of-the-art object detectors like *Felzenszwalb et al.* (2010) can detect objects in an image with cluttered background, with reasonably accurate estimates of object pose. Further, as demonstrated by *Bao and Savarese* (2011a), multiple images can significantly improve detection accuracy in both image and 3D space. In image I_j, the detector returns the confidence value $p_i(\mathbf{u}, s, \pi)$ of a detection hypothesis which appears in image location $\mathbf{u}$, with scale (height and width) s and pose π. Given the estimated camera poses, a hypothesized 3D object O can be projected to each image I_j at location $\mathbf{u}_j$, scale s_j and pose π_j. Thereby, the object O in 3D space may be estimated as

$$O = \arg\max_{O} \sum p_j(\mathbf{u}_j, s_j, \pi_j). \tag{5.10}$$

Please refer to *Bao and Savarese* (2011a) for details. This allows approximate estimation of the centroid, 3D pose and scale of an object. Since we also know those for the shape prior, we can use a rigid transformation to coarsely align the prior shape and its anchor points to fit the SFM point cloud of the object. The initial alignment for a car reconstruction is shown in Figure 5.6.

Note that unlike Procrustes alignment, this detection-based alignment does not rely on any SFM points (only camera poses), thus, it is robust to the sparsity and noise that pervade SFM point clouds obtained from few images.


(a) Side view. Car 1 2 3

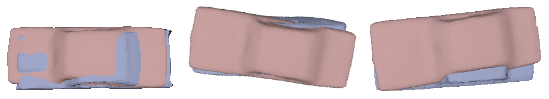
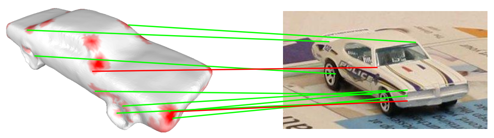

(b) Top view. Car 1 2 3

Figure 5.6: Initial alignment using object detection. Blue shows ground truth position of the object to be reconstructed. Red shows object position and orientation estimated from detection *Felzenszwalb et al.* (2010) across 15 views.


(a) Car 1

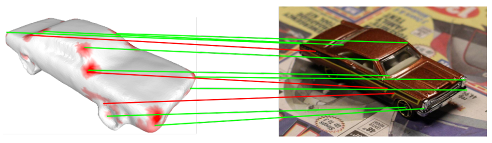

(b) Car 2

Figure 5.7: Matching anchor points from learned model (left) to new object (right). We show the high confidence matches visible under the displayed viewpoint. The green/red lines show the good/bad matches.

5.5.2 Reconstruction

Given a set of images $\mathbf{I}$ of an object with unknown shape $\mathbf{S}$, we use standard SFM to recover the 3D point cloud $\mathbf{S}_{\text{sfm}}$. Our goal is to use the mean shape $\mathbf{S}^*$ to produce a dense reconstruction that closely resembles $\mathbf{S}$.

Matching Anchor Points. Since the initial alignment uses the object's location, pose and scale, anchor points are likely to be aligned to 3D locations in the vicinity of their true matches. Thus, the burden of identifying the point in $\mathbf{S}_{\text{sfm}}$ that corresponds to an anchor point in $\mathbf{S}^*$ is reduced to a local search. We use HOG features to match anchor points to SFM points. To further improve the robustness, Algorithm V.2 proposes an iterative matching scheme. Examples of robust anchor point matches from our algorithm are shown in Figure 5.7.

Warping Based on Anchor Points. Assume $\mathbf{S}^*$ is the shape prior after the initial alignment of Section 5.5.1. We use the above matches between anchor points in $\mathbf{S}^*$ and SFM points in $\mathbf{S}_{\text{sfm}}$ to estimate parameters θ for the weighted TPS warping (5.4)

Algorithm V.2 Matching anchor points

```
Set parameters $\delta_1$ $\delta_2$ $\eta$.
for $k = 1 : K$ (total number of iterations) do
    Initialize match set $\mathbf{B}_k = \{\}$.
    for all $A_i = \{\mathbf{\Gamma}_i, \boldsymbol{\chi}_i, \omega_i\} \in \{\mathbf{A}\}$ do
        Define $P = \{\mathbf{p}_k \in \mathbf{S}_{\text{sfm}} : \|\mathbf{p}_k - \boldsymbol{\chi}_i\| < \delta_1\}$.
        Find $\mathbf{p}_j \in \mathbf{S}_{\text{sfm}}$ s.t. $\mathbf{p}_j = \arg\min_P d^{i,j}$ (Eq. 5.9)
        If $d(\mathbf{f}_j, \mathbf{f}_i) < \delta_2$, match $(A_i, \mathbf{p}_j)$, $\mathbf{B}_k = \mathbf{B}_k \cup \{\mathbf{p}_j\}$.
        Record 3D distance $r_i = \|\boldsymbol{\chi}_i - \mathbf{p}_j\|$.
    end for
    Solve $\theta'_k = \arg\min \|T(\mathbf{A}, \theta) - \mathbf{B}_k\|$.
    for all $A_i \in \mathbf{A}$ do
        if $\|T(\boldsymbol{\chi}_i, \theta'_k) - \mathbf{b}_i\| > r_i$ then
            Discard match $(A_i, \mathbf{b}_i)$, $\mathbf{B}_k = \mathbf{B}_k \backslash \{\mathbf{b}_i\}$.
        end if
    end for
    Solve $\theta_k = \arg\min \|T(\mathbf{A}, \theta) - \mathbf{B}_k\|$.
    $\forall A_i \in \mathbf{A},\ \boldsymbol{\chi}_i \leftarrow T(\boldsymbol{\chi}_i)$.
    $\delta_1 \leftarrow \eta\delta_1$.
end for
Output: the set of matches $\mathbf{B}_K$.
```

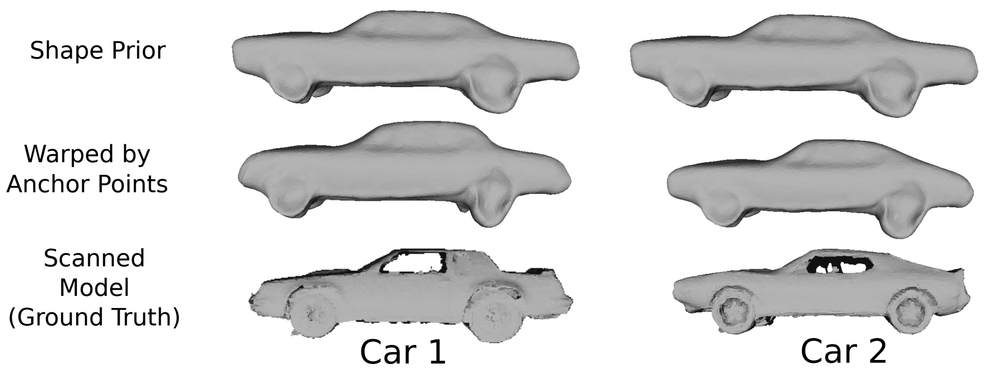


Figure 5.8: Warping of the shape prior with the learned anchor points matched to SFM points using Algorithm V.2. Note that while the shape prior represents the commonality of all instances, anchor point-based warping recovers coarse aspects of instance-specific shape, such as the back geometry of Car 2.

and obtain $\mathbf{S}' = T(\mathbf{S}^*, \theta)$ that further approaches the actual shape. Notice that, this warping not only reduces the alignment error from the initial detection-based alignment, it also deforms the prior to fit the actual shape of the object. See Figure 5.8.

Refinement. The final step in the reconstruction process is to recover the unique details of the object. These unique details cannot be learned a priori, so they may not be captured by the warped shape $\mathbf{S}'$. We use the output of an MVS algorithm *Furukawa and Ponce* (2010), $\mathbf{S}_{\text{mvs}}$, to supply these details. While MVS may have several missing regions and outliers for the object we consider, it may reconstruct

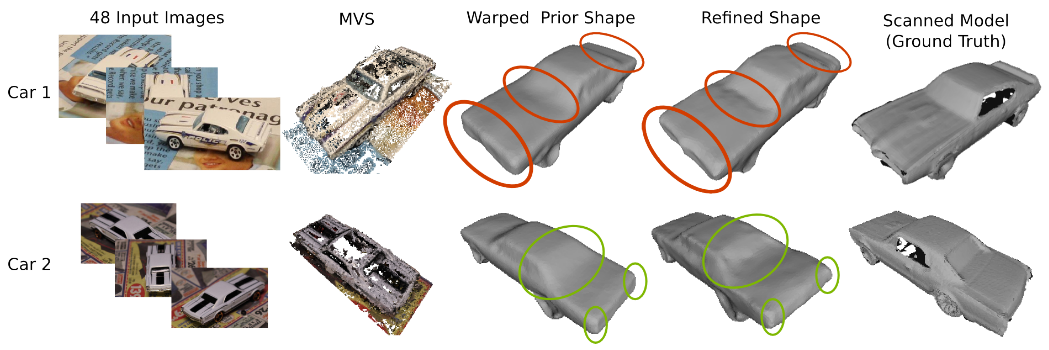


Figure 5.9: Refinement recovers unique details of an instance that are lost during mean shape learning. Examples such as the rear spoiler of Car 1 and the inset rear window of Car 2 are highlighted.

accurate oriented patches in textured or Lambertian regions where diffuse photoconsistency is a reliable metric. Using the refinement process governed by (5.5), we move the vertices of $\mathbf{S}'$ closer to $\mathbf{S}_{\text{mvs}}$. The weights ϵ_k now incorporate the confidence in the corresponding matched MVS point, which is encoded by the normalized cross-correlation photoconsistency.

The effect of refinement is shown in Figure 5.9. Note that not only are the holes and outliers of traditional MVS eliminated in our reconstruction, but fine details that are missing in the warped prior shape are also recovered by refinement – see the front bonnet and rear spoiler of Car 1, or the inset rear window edges and the protruding trunk of Car 2. This refined shape is the final output of our dense reconstruction framework.

5.6 Experiments

We evaluate our method on three categories: car, fruit and keyboard. We use a structured light 3D scanner to acquire ground truth shapes for learning and evaluation. Our testing is leave-one-out, that is, to reconstruct one instance, we train our model on all the rest. The model parameters are obtained by cross-validation in the training set. We compare against state-of-the-art MVS methods, show reconstruction results in Figure 5.11 and report quantitative evaluation results for the car dataset in Tables 5.1 and 5.2. Example results from individual stages of our framework are also depicted in Figures 5.3–5.9.

The car dataset comprises ten instances with lengths between 65 – 73mm. Using the detection-based initial alignment (Section 5.5.1), the estimated centroids of test objects are localized within 20% of object length and the orientation estimation error is within 10°, as shown in Figure 5.6. The fruit dataset consists of life-size models for twelve fruits of varying shapes and sizes. The keyboard dataset consists of seven keyboards. Centroid localization error (relative to object length) and orientation estimation error are within 5% and 40° for the fruits and within 10% and 30° for the keyboards.

To quantitatively demonstrate the efficacy of our framework, we perform a rigorous evaluation against ground truth. Reconstruction error (relative to ground truth scan) is computed using the metric in *Cignoni et al.* (1998) (other metrics such as *Seitz et al.* (2006) are equally applicable). For each test instance of the car category, we perform reconstructions using 48, 15 and 5 images. The baseline method is MVS *Furukawa and Ponce* (2010), with the reconstructed patches meshed using Poisson Surface Reconstruction (PSR) *Kazhdan et al.* (2006). We also evaluate errors for intermediate results of our pipepine. See Table 5.1. It is clear that each stage of our framework leads to significant improvement, with an over 40% improvement in final quality over traditional MVS. Also note that our reconstruction error in the challenging situation of 5 images is even lower than the baseline method with 15 images.

The efficacy of using anchor points and their learned weights can be demonstrated by Table 5.2. Using anchor points can greatly reduce the reconstruction error compared to only using object detection for alignment. Learning anchor point weights further enhances the reconstruction accuracy.

# img	Base %	RGD %	WP %	Full %
48	1.22	1.00	0.88	0.71
15	2.72	2.39	2.29	1.88
5	4.66	2.91	2.86	2.47

Table 5.1: Reconstruction error in car dataset. Base: *Furukawa and Ponce* (2010)+*Kazhdan et al.* (2006). RGD: Rigidly align mean shape to test object using matched anchor points. WP: Align and warp mean shape using matched anchor points (without refinement). Full: Our complete algorithm. Errors are reported in the metric of *Cignoni et al.* (1998). Note a 40% improvement between Base and Full.

Base	IA+RF	RGD+RF	WP (No ω)+RF	WP+RF
1.22%	1.94%	0.85%	0.75%	0.71%

Table 5.2: Reconstruction error of alternative designs of our pipeline. Base: *Furukawa and Ponce* (2010)+*Kazhdan et al.* (2006). IA: Initial alignment using object detection (Section 5.5.1). RF: Refinement (Section 5.5.2). RGD: Rigidly align the mean shape to a test object by using matched anchor points. WP: Align and warp the mean shape by using matched anchor points (Section 5.5.2). No ω: Using anchor points with equal weights. Errors are computed by using the car dataset with 48 images available for each car.

We also use our reconstruction method for scenes with multiple objects in a cluttered environment (Figure 5.10). The method of *Bao and Savarese* (2011a) is used to detect multiple objects in the 3D scene and our framework is individually applied to each object. Note that our reconstructed objects are aligned in the same coordinate system as the SFM point cloud of the scene. This allows us to automatically overlay the 3D objects reconstructed using our method with the point cloud of the background.

In Figure 5.11, we show several comparisons of our reconstructions against state-of-the-art MVS *Furukawa and Ponce* (2010); *Kazhdan et al.* (2006). Note the lack of texture and specularities in the sample images shown in (a). Diffuse photo-consistency is not a metric well-suited to these situations, so the MVS output in (b) is visibly noisy and contains a large number of artifacts in the form of holes and outliers. Consequently, the resulting PSR mesh in (c) is distorted. In contrast, we successfully learn meaningful semantic priors across shape variations and use them in our reconstruc-

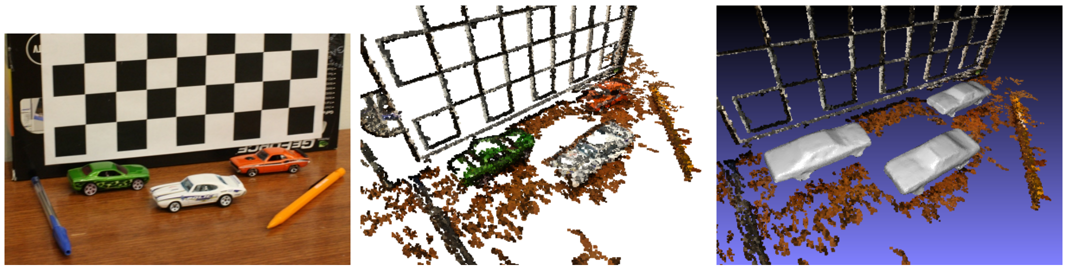


Figure 5.10: Reconstruction of multi-object scenes. (Left) 1 out of 10 input images. (Middle) MVS *Furukawa et al.* (2010). (Right) Our reconstruction.

tion, to produce the much higher quality reconstructions in (d), that closely resemble the ground truth (e).

5.7 Discussion

We have presented a comprehensive framework for dense object reconstruction that uses data-driven semantic priors to recover shape in situations unfavorable to traditional MVS. Our learned priors, combined with robust anchor point matching and refinement mechanisms, are shown to produce visually high quality and quantitatively accurate results.The success of this framework also opens up directions for future research. While semantic information for objects such as cars is easily correlated to shape, many categories such as chairs show shape variation at finer granularities. Thus, ongoing research efforts in fine-grained recognition and detection of object parts may also benefit our semantic reconstruction framework.

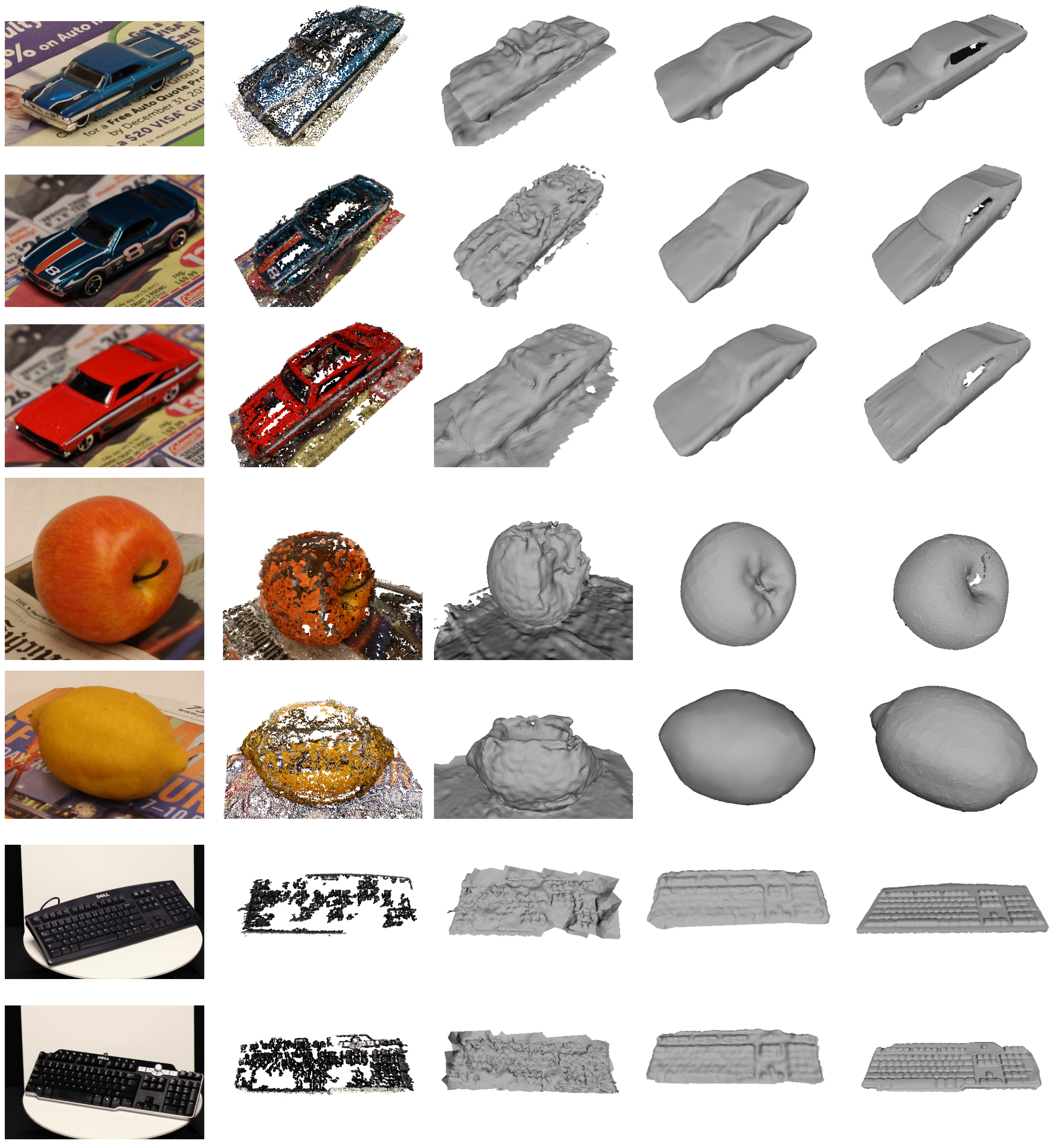


Figure 5.11: Examples of reconstructed objects. Notice the lack of texture and presence of specularities in sample images (a). MVS reconstruction from 48 images using the method of *Furukawa et al.* (2010) produces clearly visible holes and extremely noisy reconstructed patches (b). Poisson surface reconstruction (*Kazhdan et al.* (2006)) fails to produce a reasonable mesh under such scenarios (c). Our semantic framework, on the other hand, yields a high quality reconstruction (d), which closely resembles the ground truth (e), both visually and quantitatively. The results are obtained by using 48 images for cars and fruits, and 5 images for keyboards.

CHAPTER VI

Scene Understanding Application in Indoor Environment

6.1 Introduction

Chapter II - V present methods for understanding a scene from images. In this chapter, we will introduce an application of our scene understanding framework, with the goal of understanding an indoor environment from multiple images. The objectives of indoor room understanding involves estimating 3D layout (e.g. floor, walls, ceiling) of the indoor environment as well as identifying the objects within it. Using images to understand the layout of a cluttered room is a great challenge in computer vision research. A room may be occupied by objects that are not necessarily observed in a training set. The room walls may be occluded and cannot be observed directly (Fig. 6.1). Solving the room layout understanding problem is beneficial in many applications including autonomous navigation, manipulation, augmented reality, architecture CAD, and mobile vision.

In the past few decades, researchers proposed numerous remarkable methods*Hartley and Zisserman* (2000); *Snavely et al.* (2008); *Furukawa et al.* (2009b) focusing on obtaining metric reconstructions of an unknown environment. These methods can accurately recover the 3D geometry of an environment given enough quantity of images.

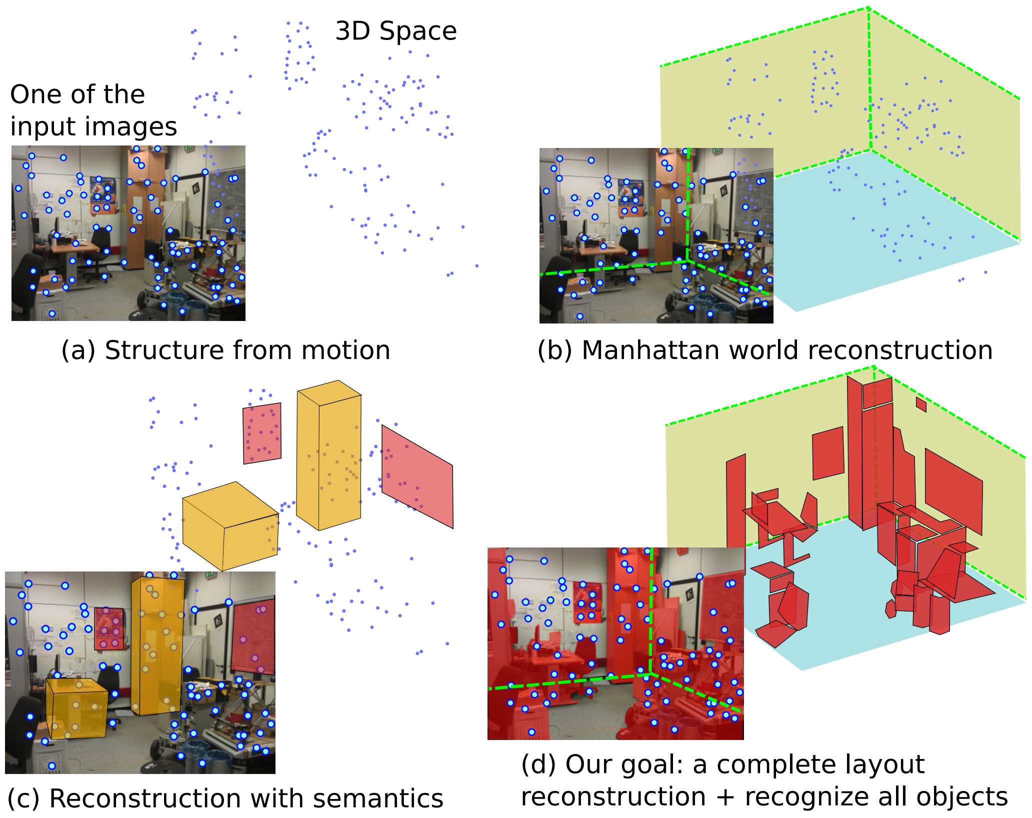

3D Space
One of the input images
(a) Structure from motion
(b) Manhattan world reconstruction
(c) Reconstruction with semantics
(d) Our goal: a complete layout reconstruction + recognize all objects

Figure 6.1: Understanding a cluttered room from a few images. (a) Structure from motion techniques (e.g. *Snavely et al.* (2008)) can only understand the geometry of the room as a sparse set of 3D points. (b) Layout estimation methods (e.g. *Tsai et al.* (2011); *Flint et al.* (2011)) may recover the wall structure without reasoning about objects. (c) Joint geometric and semantic reconstruction methods (e.g. *Bao et al.* (2012a)) can recognize a few objects (the yellow boxes), estimate their positions in 3D, as well as estimate the 3D layout as a sparse set of elements (the red regions). (d) Our goal is to estimate the complete 3D layout of the room (floor, walls, ceiling) and identify all the foreground objects. Notice that we aim at differentiating objects v.s. walls, rather than distinguishing different object entities / categories.

However, they cannot identify the key semantic phenomena inside the environment (Fig. 6.1a). Meanwhile, researchers *Brostow et al.* (2008); *Xiao and Furukawa* (2012); *Silberman et al.* (2012) also looked at estimating scene semantics from 3D points. Nevertheless, these methods usually require very dense and accurate reconstructions obtained using 3D scanners or from a very large number of images. Such requirement limits the scope of their applications. Moreover, *Tsai et al.* (2011); *Flint et al.* (2011) leverage the Manhattan world assumption to estimate room walls from a sequence of images, but they cannot handle objects in a scene (Fig. 6.1b).

Recently, *Bao et al.* (2012a) proposed an approach to jointly estimating the geometric and semantic properties of a scene. Using a small set of images, *Bao et al.* (2012a) shows better 3D geometry estimation and object recognition results than the geometry estimation methods or the semantic reasoning methods that work in isolation. Unfortunately, one of its shortcomings is that it can only produce a very sparse reconstruction of a scene (Fig. 6.1c), which is not desirable for the aforementioned applications.

Another noticeable series of works concentrate on parsing the room layout from a single image *Hedau et al.* (2009, 2010, 2012); *Lee et al.* (2010, 2009); *Pero et al.* (2012); *Schwing and Urtasun* (2012); *Wang et al.* (2010); *Jiang et al.* (2012); *Fouhey et al.* (2012). However, their accuracy in estimating the 3D scene layout is limited mostly due to the fact that 3D perception from a single view is essentially an ill-posed problem, and the room structure may not be uniquely inferred from a single image. An illustrative example is shown in Fig. 6.2.

Understanding the room layout from multiple images is far from being trivial. We need an effective and efficient algorithm to jointly reason about the content in multiple images. On the other hand, although we can infer certain 3D geometry information, e.g. structure-from-motion (SFM) points, to help room layout estimation, the 3D cues inferred from a few input images are usually very sparse and noisy. Experiment

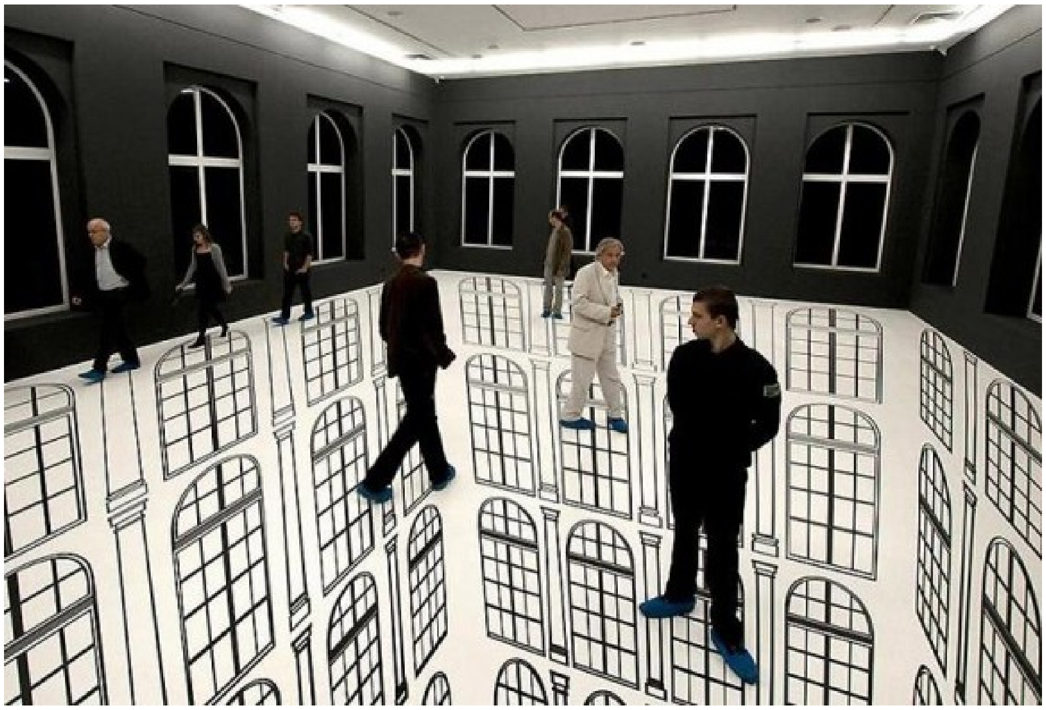


Figure 6.2: Using a single image to understand room layout may suffer from the intrinsic ambiguity of a single image. This photo may be interpreted in two ways: 1) the floor is painted artificially to create the illusion; 2) the room is hollow and the people are floating. If we are given another photo from a different view point, this ambiguity will naturally dissolve.

results proved that simply relying on the SFM points from a small set of images (~10) will yield very unstable and inaccurate layout estimation results. In order to address these challenges, we propose a new room understanding framework bearing the following contributions.

Accuracy. We can achieve higher accuracy in layout estimation and object recognition tasks than pure geometry-based methods or single-image methods. We estimate 3D room layout (walls, floor, ceiling) jointly using geometric and semantic cues, which play complementary roles in helping recover the geometry of the scene. When a room is very cluttered, there will usually exist a large set of characteristic feature points, which can yield a SFM point cloud with reasonable density (geometric cue). SFM points can help us reason about the extent of the room and thereby tackle the adversary that wall boundaries are occluded by the foreground objects. As the opposite, when a room is comparatively clean, we can exploit image line segments and region segmentation results (semantic cue) to obtain a good estimation of the room's walls. Meanwhile, by jointly using multiple images to reason about the existence of objects, our object recognition accuracy can be demonstrated to be significantly higher than single-image methods.

Completeness. We seek for a complete reconstruction of the room layout in 3D including objects. In contrast, many aforementioned methods can only reconstruct the room layout as a set of points *Hartley and Zisserman* (2000); *Snavely et al.* (2008) or a sparse set of regions *Bao et al.* (2012a). Moreover, different from many previous works *Lee et al.* (2010); *Hedau et al.* (2010); *Pero et al.* (2012) that only consider box-like objects, our model can accommodate objects with more complex shapes. We propose a surface-based object representation (Fig. 6.5), which greatly expands the types of recognizable objects compared to a box-based representation. Notice that, our goal is to recognize objects apart from room layouts, rather than recognizing object categories. Compared to recognizing an object as a whole (as in *Lee et al.*

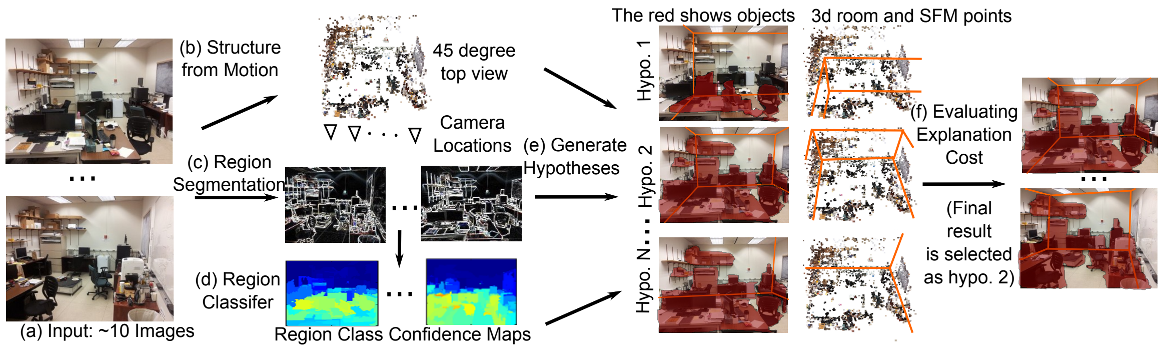


Figure 6.3: Multi-image room layout understanding framework.

(2010); *Hedau et al.* (2010); *Pero et al.* (2012)), our surface-based representation also enhances the chance of recognizing an unknown object (an object appears in testing but not in the training set) by using parts (surfaces) that are shared by other objects in the training set. For example, a wooden desk may share similar texture and legs as a wooden chair. Hence, even if our training set does not contain the desk category, the desk may still be successfully recognized (as an object) provided that the training set contains a chair with similar parts and texture. Notice that, we use a generic object segmentation algorithm (Sec. 6.2.1) to decompose objects into surfaces, rather than using a pre-trained model for each object category.

We conducted numerous experiments using a novel dataset containing 50 various room scenes with 10 images in each scene. Various experiments demonstrate that our framework can achieve better estimation accuracy and higher reconstruction completeness than alternative state-of-the-art approaches.

6.2 Problem Definition

6.2.1 Inputs and Measurements

We are provided a total number of N unordered images $I^1 \cdots I^N$ (Fig. 6.3a). In each image I^i we can detect a set of feature points (e.g. *Lowe* (2004)) $\mathbf{p}^i$, as well as a set of segmented regions $\mathbf{b}^i$ (Fig. 6.3c and 6.3d). In the following text we will also refer to line segments in images. Line segments are essentially the boundaries of regions. For the sake of simplicity, we do not introduce additional symbols for line segments.

The feature points play a number of different roles in our framework. One role is to create the 3D reconstruction of the points in rooms and help estimate camera parameters. Since the target scenario of our algorithm is a cluttered room, we assume these input images contain features points sufficient to be matched across each other, and therefore a structure-from-motion (SFM) pipeline can use these images to estimate a set of 3D points $\mathbf{P}$ in the scene, as well as the camera parameters $\mathbf{C}$ (Fig. 6.3b). Let $\mathbf{C} = \{C^i\}$ be the camera parameters where C^i indicates the rotation, translation, and intrinsics of image I^i. The extrinsics are estimated using a SFM pipeline (e.g. *Snavely et al.* (2008)), while the intrinsics may be provided as input or estimated using auto-calibration *Hartley and Zisserman* (2000).

The region segments are critical to our framework for evaluating the possibility of a room hypothesis (layout + objects). Since our framework is designed to use multiple input images, the region segments should be matched across images. We apply a multi-image segmentation algorithm (e.g. *Toshev et al.* (2007)), which not only automatically matches across-image regions covering the same objects (see colored regions in Fig. 6.4b), but also simultaneously guarantees the matched regions similar shapes and appearances (see region shapes in Fig. 6.4b). The k^{th} region segment in image I^i is denoted as b^i_k, whose appearance can be described by a vector concatenating multiple cues (e.g. cues proposed by *Hoiem et al.* (2007)). Given the appearance vector and a pre-trained region classifier, a confidence can be calculated that b^i_k belongs to class label l (e.g. walls, floors, ceilings, or other objects). See Fig. 6.4c for examples.

6.2.2 Unknown Parameters

The unknowns are the 3D layout and the configuration of objects in the room.

The 3D layout can be described by a set of room surfaces (walls, floors, ceilings) $\mathbf{S} = \{S_1 \cdots S_{N_S}\}$. A surface S_i is parametrized by its centroid, orientation, and extent in 3D. In our experiment, we follow previous works *Hedau et al.* (2009); *Lee et al.* (2009); *Pero et al.* (2012) which hold the assumption that the room layout is a 3D box. See orange lines in Fig. 6.3e and 6.3f.

We model 3D objects as a set of 3D planar surfaces (we also refer to as regions). See Fig. 6.5 for examples. In our framework, we do not model objects as each single entities. Instead, we assign to each surface a single class label which is *object* v.s. *non-object*. Non-object means that a surface belongs to the room layout which can be further classified into floor, wall, or ceiling. Object means that a surface belongs to one of the foreground objects (though we do not distinguish which one). Let $\mathbf{O} = \{O_1 \cdots \cdots O_{N_o}\}$ represent the collection of all objects in a room environment, where O_i is a planar 3D surface which belongs to an object in the scene. A surface O_i captures the location, orientation, and extent of a component of an object in 3D. Although such modeling approximates every surface as flat, it allows to accommodate arbitrarily complicated object configurations.

6.3 Model Formulation

Our goal is to estimate a room layout $\mathbf{R} = \{\mathbf{S}, \mathbf{O}\}$ from measurements by minimizing a cost function E:

$$\mathbf{R} = \arg\min_{\mathbf{R}} E(\mathbf{R}; \mathbf{P}, \mathbf{C}, \mathbf{b}) \tag{6.1}$$

where E evaluates the likelihood of $\mathbf{R}$ given SFM points $\mathbf{P}$, estimated camera parameters $\mathbf{C}$, and region measurements in every image $\mathbf{b} = \{\mathbf{b}^i\}$. In order to compute the cost of a given hypothesis with respect to the measurements, we consider the cost of their geometric compatibility in 3D space (E_G) and the cost of semantic interpretation in images (E_M):

$$E(\mathbf{R}; \mathbf{P}, \mathbf{C}, \mathbf{b}) = E_G(\mathbf{R}; \mathbf{P}) + E_M(\mathbf{R}; \mathbf{C}, \mathbf{b}) \tag{6.2}$$

Notice that, if only one image is given, we cannot evaluate 3D geometry cost, the overall cost degenerates into evaluating semantic cost in a single image, which is related to the energy function proposed in most single-image methods *Hoiem et al.* (2007); *Hedau et al.* (2009).

6.3.1 Geometric Cost

A good layout estimation should be compatible with the estimated SFM points. However, the criteria of evaluating such compatibility should be carefully selected since SFM points may contain many outliers and only sparsely represent the 3D layout of a room. We use the following criteria to calculate E_G:

- The inner space enclosed by $\mathbf{S}$ should contain all scene points $\mathbf{P}$. Let $\Omega(\mathbf{P}; \mathbf{S})$ be the function computing the percentage of points in $\mathbf{P}$ not enclosed by $\mathbf{S}$. The cost of the points excluded from a room structure can be computed as $E_G^I = \Omega^2(\mathbf{P}; \mathbf{S})/\sigma_\Omega^2$.

- The 3D walls / floors/ ceilings defined by $\mathbf{S}$ should be supported by points in $\mathbf{P}$. S_i is the i^{th} 3d surface in $\mathbf{S}$. Let $\tau_i^S \subset \mathbf{P}$ be the indices of the 3D points whose image projections fall into the image projection of S_i excluding the part occluded by object surfaces in $\mathbf{O}$. Denote by $\Lambda(S_i, p_j)$ the function computing the 3D distance from S_i to 3D point p_j. The cost of unsupported 3D walls can be computed as $E_G^S = \sum_i \sum_{j \in \tau_i^S} \Lambda^2(S_i, p_j)/\sigma_S^2$.

- Similarly, the 3D objects (i.e. a set of 3D regions) should also be supported by points in $\mathbf{P}$. Let $\tau_i^O \subset \mathbf{P}$ be the indices of the 3D points whose image projections fall into the image projection of O_i. Let The cost of unsupported 3D objects can be written as $E_G^O = \sum_i \sum_{j \in \tau_i^O} \Lambda^2(O_i, p_j)/\sigma_O^2$.

The overall geometric cost is a summation:$E_G = E_G^I + E_G^S + E_G^O$. The variance terms $\sigma_\Omega, \sigma_S, \sigma_O$ are learned using a max-likelihood approach.

6.3.2 Semantic Cost

The sophisticated content carried by images can be used to verify the possibility of a room hypothesis. We project $\mathbf{O}$ and $\mathbf{S}$ into each image using the estimated camera parameters $\mathbf{C}$. Denote by s_i^k / o_j^k the image projection of $S_i \in \mathbf{S}$ / $O_j \in \mathbf{O}$ in the k^{th} image considering their occlusion relationships. Once S_i or O_j is projected into an image, we can transfer their labels to corresponding image regions. The possible labels include left wall, front wall, right wall, ceiling, floor, objects. Correct 3D layout will lead to labels reinforced by image evidence. We use segmented regions to check the likelihood that a projection with certain inferred label is correct. A projected 3D region (s_i^k / o_j^k) may overlap with a number of image regions (object region only overlaps with one). Denote by θ_i^k the indices of the elements in $\mathbf{b}^k$ (regions in the k^{th} image) which overlap with s_i^k. The semantic cost for one projected wall region in image I^k can be computed as:

$$E_M^k(S_i) = -\frac{1}{|\theta_i^k|} \sum_{j \in \theta_i^k} c(b_j^k \in l_i^k)$$

where $c(\cdot)$ is the label confidence function defined by a classifier learned from a training set. The semantic cost for object regions $E_M^k(O_i)$ can be easily written in a similar fashion. Given multiple images and all the elements in $\mathbf{S}$ and $\mathbf{O}$, the semantic cost can be written as:

$$E_M(\mathbf{R};\mathbf{C},\mathbf{b}) = \sum_k (\sum_i E_M^k(S_i) + \sum_j E_M^k(O_j))$$

6.4 Solving the Estimation Problem

We solve the room estimation problem by identifying the room layout $\mathbf{S}$ and objects $\mathbf{O}$ minimizing Eq. 6.1. Due to the high dimensionality of the unknown parameter space, we adopt an approach that is based on proposing hypotheses and evaluating them using the cost function E (Eq. 6.2). We first propose a set of hypotheses $\{\mathbf{R}_n\}$ (Sec. 6.4.1), and next identify among these proposals the best layout configuration which yields the minimum cost (Sec. 6.4.2). Our framework can be summarized as the flowchart shown in Fig. 6.3.

6.4.1 Generating Hypotheses

Effectively proposing room hypotheses is the key to this estimation process. The room proposal process consists of four steps.

6.4.1.1 Estimating Dominant Directions

We adopt the Manhattan world assumption that the walls of a room must be perpendicular to one of three mutually perpendicular directions (dominant directions). We adopt *Lee et al.* (2009) to estimate dominant directions from the line segments (e.g. boundaries of regions or detected using methods such as *von Gioi et al.* (2012)) in each input image. The dominant direction in the world coordinate system can be calculated by averaging the dominant directions in all images considering their relative camera poses.

6.4.1.2 Triangulating Room Corners

In order to generate room hypotheses, we first estimate a set of possible 3d locations of room corners. A room corner is a 3D point where three walls intersect. A room's layout can be defined by its corners. In order to locate room corners in 3D space, we first identify them in each image (Fig. 6.6b), and use estimated camera poses to triangulate their 3D position (Fig. 6.6c). This is not a trivial task since wall corners may not be directly observable, due to occlusion or weak corner detector response. We leverage on line segments to infer the existence and locations of room corners in an image (Fig. 6.6a). Given the estimated dominant directions, each line segment can be labeled as "bottom-up", "left-right", "back-front", or "random". Two different types (except random lines) of line segments may intersect and form a corner. In one image, by pairing line segments and inferring their 2D intersections, we can obtain a (large) set of image points among which a few represent true room corners. We can obtain the 3D location of a room corner candidates q_i by triangulating a pair of image corner candidates that satisfy epipolar constraint. Triangulating every pair of 2D corner candidates may generate a very large set of 3D points $\mathbf{Q} = \{q_i\}$, among which only a few are true room corners.

6.4.1.3 Generating Room Hypotheses

Room hypotheses are generated from the corner candidate set $\mathbf{Q}$. In our experiment, we assume a room layout is a cuboid, hence a layout hypothesis can be uniquely proposed using a number of corners. We randomly sample points in $\mathbf{Q}$ and obtain the set of room layout hypotheses. In order to confine the total number of layout hypotheses within a tractable range (at most 300 in our experiment), we use K-means algorithm to cluster similar room layout and only keep significantly different room layout hypotheses as $\{\mathbf{S}_l\}$.

We hold the assumption that a room layout is a box. The three orthogonal directions of the box can be estimated using line segments in images (Sec. 4.1.1 in the main submission). We further assume that the cameras only point to one side of a room (Fig. 6.7). Hence, a room layout can be defined by four corners (1,2,3,4 in the Fig. 6.7). From room corners in images, we can obtain a large set of candidate 3D room corners $\mathbf{Q}$ (Sec. 4.1.1 and 4.1.2 in the main submission). However, which corners are actually present in the input images are unknown. We enumerate every possible combination of 3D room corners in $\mathbf{Q}$ to generate room layout hypotheses. As the dominating directions of the room are estimated and fixed, we can generate room layout from 3 types of corner combinitions: (a) if two diagonal corners ($(1,3)$ or $(2,4)$ in Fig. 6.7)) simultaneously occur in the candidate set, a room layout candidate can be uniquely identified; (b) if two corners belonging to one wall ($(1,4)$, or $(3,4)$, or $(2,3)$, or $(1,2)$) simultaneously occur, one side of the room can be identified. The unobserved side is assumed to be located at infinite; (c) each single corner (1, or 2, or 3, or 4) in the candidate set also defines a room layout candidate. In this case, the remaining three corners are assumed to be at infinite.

By enumerating every possible combination of the room corners in the candidate set $\mathbf{Q}$, we can obtain a set of room layout candidates $\mathbf{L} = \{L_i\}$. Notice that, many elements in $\mathbf{L}$ are almost identical, since they are produced by the observations of the same physical room corners in different images. For this reason as well as for the sake of limiting the estimation complexity, we group similar room layouts in $\mathbf{L}$ into single ones. A layout L_i can be parametrized by a vector recording the 3D locations of its four corners. We apply K-means *Lloyd* (1982) to every element in $\mathbf{L}$, and obtain $\mathbf{L}'$ which is the set of the means produced by the K-means clustering process. In our experiment, we set $K = 300$. We use $\mathbf{L}'$ as our set of room layout hypotheses.

6.4.1.4 Generating Object Hypotheses

After a layout hypothesis $\mathbf{S}_l$ is generated, we next generate its compatible object configuration $\mathbf{O}_l$. In order to minimize the overall cost, the object hypotheses are generated from two clues: 1) 3D SFM points that are not close to the room walls (to minimize E_G^O), 2) image regions that are assigned with a high score by object classifier (to minimize E_M^k). Please see Fig. 6.8 and its caption for the details regarding generating object hypotheses. In our experiment, we find these two types of clues complimentary. An object (e.g. a book with unique cover) may not share similar appearance with other objects in a training set, and therefore a, appearance-based classifier may fail to detect it. However, its triangulated 3D location can help infer that it does not belong to rooms walls (hence it must be an object). On the other hand, an object (e.g. a table surface) may have simple and clean appearance which does not carry sufficient features for SFM, but it's simple appearance pattern may be easily recognized by a classifier. A room layout hypothesis $\mathbf{S}_l$ and its corresponding object hypotheses $\mathbf{O}_l$ constitute a room hypothesis $\mathbf{R}_l$.

6.4.2 Evaluating Hypotheses

Given a layout hypothesis $\mathbf{R}_l$, we can evaluate its cost as $e_l = E(\mathbf{R}_l; \mathbf{P}, \mathbf{C}, \mathbf{b})$. The final estimation of the room layout is obtained by selecting the hypothesis with the lowest cost. In our experiments, we exploit parallel computing technique to efficiently evaluate all layout hypotheses. As our future work, we will adopt faster inference algorithm such as branch-and-bound *Schwing and Urtasun* (2012) to accelerate the hypotheses generation and evaluation process.

	SFM	Img. Seg.	Hypo.	Total
Sec.	18.5	27.3	36.8	82.6

Table 6.1: Average time consumption for estimating one scene from 10 images. **SFM** includes sift *Lowe* (2004) feature detection (CUDA), feature matching(CUDA), RANSAC essential matrix estimation (C), and bundle adjustment (C). **Img. Seg.** indicates image segmentation including superpixel generation (C) and classification (multi-thread Matlab). **Hypo.** indicates hypotheses generation (multi-thread Matlab) and evaluation (Matlab+C) process described in Sec. 6.4.1 and Sec. 6.3 .

6.5 Evaluation

We conduct experiments in a novel dataset which contains 50 different room scenes each of which include 10 images. We would like to release this new kind of multi-image dataset to the community for future research. Example figures and results are shown in Fig. 6.9 and 6.10. Using this dataset, we compare our method against other state-of-the-art methods. Since our proposed method requires multiple images, we cannot evaluate on single image datasets such as the one proposed in *Hedau et al.* (2009). However, we use the labeled data in *Hedau et al.* (2009) to train region classifiers for our method and competing methods.

6.5.1 Algorithm Speed

We report the time consumption of each step in our framework in Tab. 6.1. Our unpolished implementation mixes the usage of matlab, C, and CUDA. The implementation detail is listed in the table caption. The experiment is conducted using a 4-core 2.8GHz CPU.

6.5.2 3D Reconstruction Completeness

We show the 3D reconstruction completeness in Tab. 6.2. Our model aims at estimating the 3D information of every pixel in an image. In contrast, many alternative room reconstruction methods can only recover the 3D information for a set

	Snavely et al. (2008)	*Bao et al.* (2012a)	Ours
Objects	1.2%	77.5%	**86.0**%
All	0.69%	46.0%	**91.4**%

Table 6.2: 3D reconstruction completeness. The numbers are the percentage of image pixels whose 3D information can be estimated. Objects: only count the pixels belonging to non-wall objects. All: count every pixel. Notice that our completeness is not 100%, because we cannot recover the 3D location of the object surfaces that do not contain SFM points.

of points (e.g. *Snavely et al.* (2008)) or a set of regions + points (e.g. *Bao et al.* (2012a)). *Snavely et al.* (2008); *Bao et al.* (2012a) both show higher completeness level in reconstructing non-wall objects than reconstructing both objects and walls. The reason is that *Snavely et al.* (2008); *Bao et al.* (2012a) rely on matched features (points / regions) to create 3D elements. Non-wall objects usually carry more features than walls, and therefore they are more likely to be reconstructed than walls. Notice that our method does not suffer from this condition in that we can infer the existence of walls even if they are not directly observable.

6.5.3 Layout Estimation Accuracy

In order to evaluate the accuracy for estimating room layout, we adopt the criterion commonly used in other works *Hedau et al.* (2009); *Schwing and Urtasun* (2012). We project the estimated room layout into each image, and label every pixel into wall, ceiling, or floor. The percentage of correctly labeled pixels is shown in Tab. 6.3. Due to the code unavailability of other works, we cannot evaluate them in our dataset. We also compare with a baseline geometry-based approach (Plane Fitting in Tab 6.3), which uses vanishing lines to estimate dominant directions and uses RANSAC to fit a box-like room based on SFM points. This approach is equivalent to a degenerated version of our method which only minimizes the geometry cost term.

	Home	Office	Other	Overall
Image#	300	110	90	500
Hedau et al. (2009)	79.8	79.0	81.5	79.9
Lee et al. (2009)	73.5	67.7	71.7	71.9
Plane Fitting	71.6	76.0	68.4	72.0
Ours	**92.7**	**96.7**	**92.3**	**93.5**

Table 6.3: Room layout estimation accuracy. The number is percentage number averaged on 500 images in our dataset.

	Hoiem et al. (2007)	*Hedau et al.* (2009)	Ours
Precision	38.8%	52.2%	**58.1%**
Recall	50.0%	55.4%	**59.0%**

Table 6.4: Object Estimation Accuracy. We provide ground truth labels (objects / walls) to segments in images. The precision is the percentage of images pixels that can be correctly classified. The recall is the percentage of correctly-identified pixels that belong to objects.

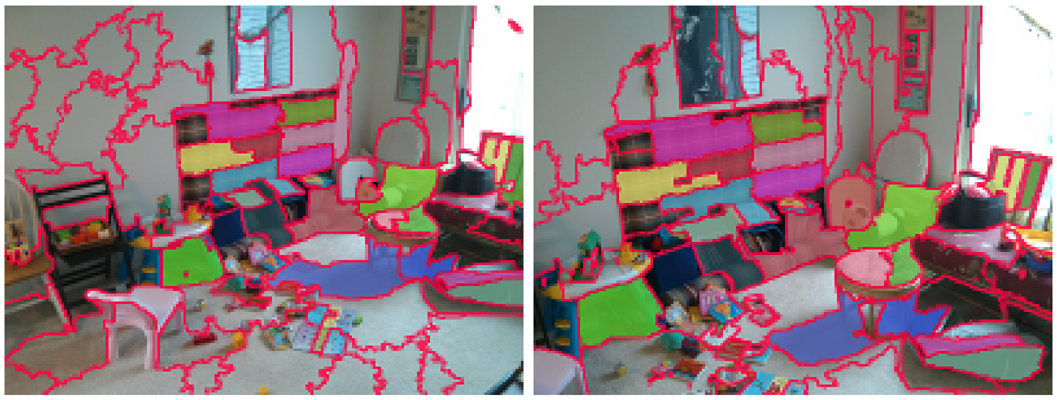
6.5.4 Object Estimation Accuracy

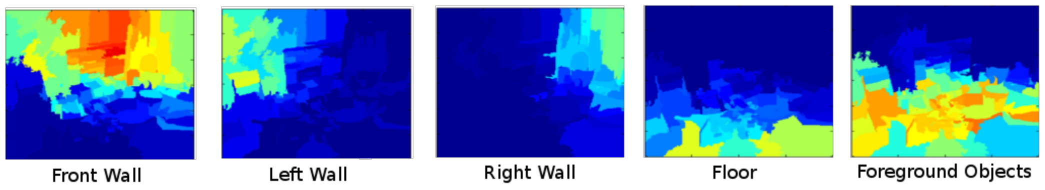
Our proposed framework can estimate non-wall objects in 3D space and in 2D images. We show example estimations in Fig. 6.9. We evaluate the accuracy of detecting object regions in images. The accuracy for estimating objects can by evaluated by examining every pixel label against ground truth. The result is shown in Tab. 6.4. Our proposed method shows significant advantage over rival methods, since it can effectively use multiple images which carry greater information than only a single image.

6.6 Conclusion

In this chapter, we proposed a multiview framework to solve the cluttered room understanding problem. Our solution can be executed efficiently using a standard computer system. Experiment results demonstrate that our method produces more complete and accurate result in estimating room layout and foreground objects than alternative state-of-the-art methods.

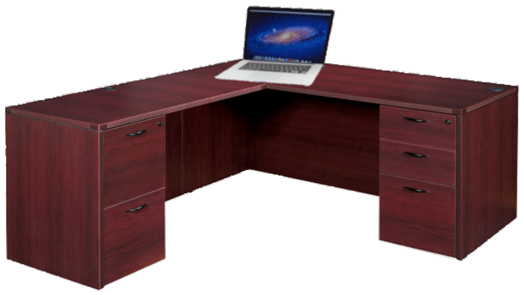
(a) Two input images


(b) Region segments and matched regions (indicated by color).

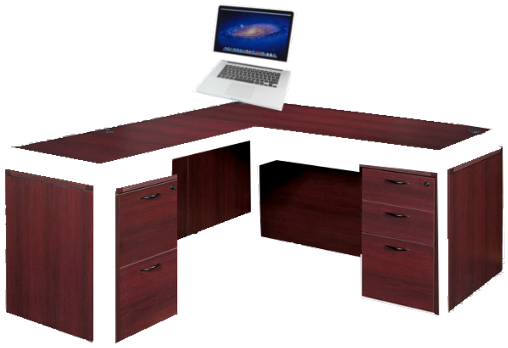


(c) Response map of region classifiers. (The left image in the image pair)

Figure 6.4: Co-segmentation. This example shows the result of using two images. In our experiment the co-segmentation is applied to ~10 images of the same room.

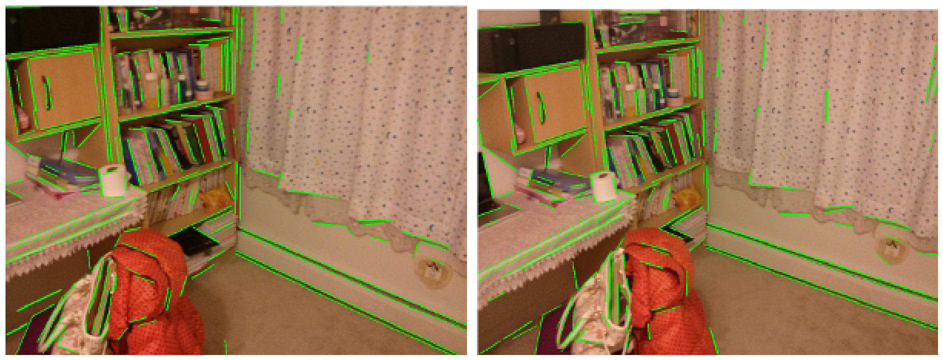


(a) Objects in a scene

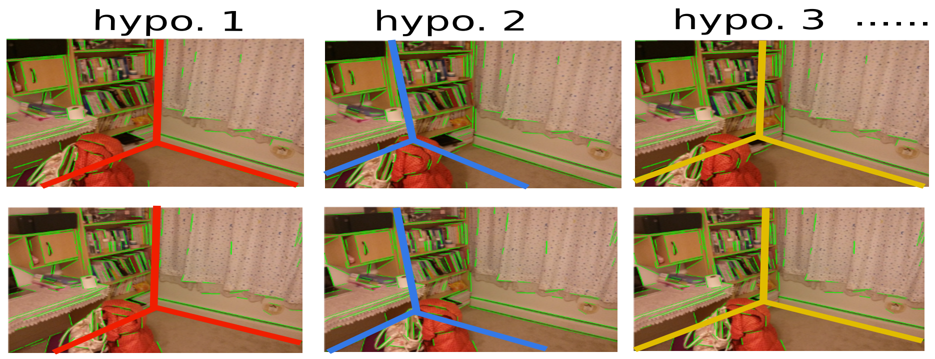


(b) Surface Decomposition

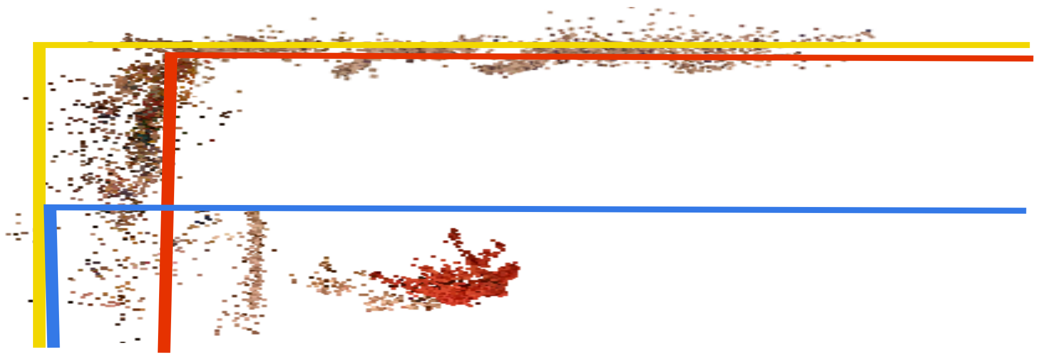
Figure 6.5: Object Representation. (a) Objects in images. These two objects cannot be effectively represented using bounding cubes as proposed by *Hedau et al.* (2010); *Lee et al.* (2010); *Pero et al.* (2012) (b) One possible region decomposition for the objects. In our experiment, the decomposition is the result generated from region segmentation algorithm (e.g. *Toshev et al.* (2007)), not from a pre-trained model. The 3D locations and orientations of object surfaces are estimated using the SFM points attached to the surfaces.



(a) Input images and detected line segments



(b) Three wall hypotheses among many proposals.



(c) SFM points and hypotheses in 3D (top view)

Figure 6.6: Proposing wall layout candidates. (a) Detected line segments. Line segments allow to estimate the dominant direction of the room. (b) Wall layout candidates are generated by enumerating pairs of line segments. (c) Triangulation of 2D wall layouts provides their configurations in 3D. By comparing with the SFM points, it is easy to see only hypothesis 3 (the yellow one) is compatible with the SFM points. Notice that most single-image methods suffer from accurately choosing the best layout among the three candidates shown in (b).

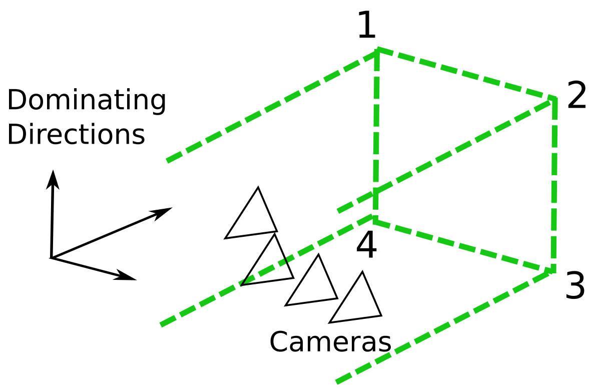


Figure 6.7: Proposing room structure hypotheses. The corners behind cameras are assumed to be located at infinity.

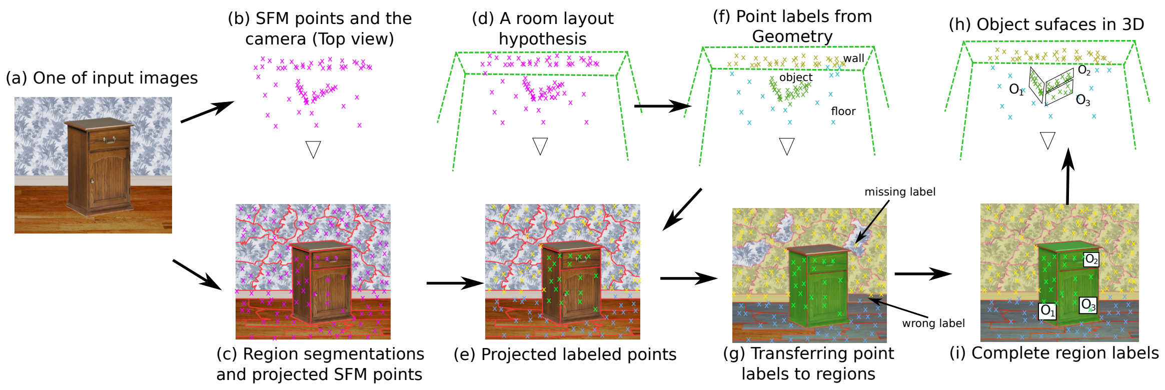

(a) One of input images
(b) SFM points and the camera (Top view)
(c) Region segmentations and projected SFM points
(d) A room layout hypothesis
(e) Projected labeled points
(f) Point labels from Geometry
wall
object
floor
(g) Transferring point labels to regions
missing label
wrong label
(h) Object sufaces in 3D
O1
O2
O3
(i) Complete region labels

Figure 6.8: Generating object hypotheses. (a) An example image from a set of input images. (b) Top view of the SFM points and the camera (the triangle). (c) Region segments in this image and the location of projected SFM points. (d) A given room hypothesis (Sec. 6.4.1.3) overlaid with SFM points. (f) We can identify the points close to the walls and assign labels to SFM points (yellow and blue). The points that do not belong to any walls will be labeled as non-room (green) i.e. objects. (e)(g) the SFM point labels can be transferred to regions. Notice that there are missing (transparent) or wrong region labels, since regions may not carry sufficient SFM points and point labels may be noisy. (i) Based on the labels initialized from SFM points, we obtain a complete region classification by minimizing E_M. Notice that the missing labels are inferred and the wrong labels are corrected by enforcing appearance consistency. (h) The region labeled as objects can be back-projected into 3D space if they carry sufficient SFM points. In this case, we can generate an object configuration hypothesis containing O_1,O_2, and O_3. Notice that the top part of the cabinet does not correspond to an object surface in 3D since it does not carry SFM points. For such surfaces of objects, our framework can infer their existence in images but not in 3D.

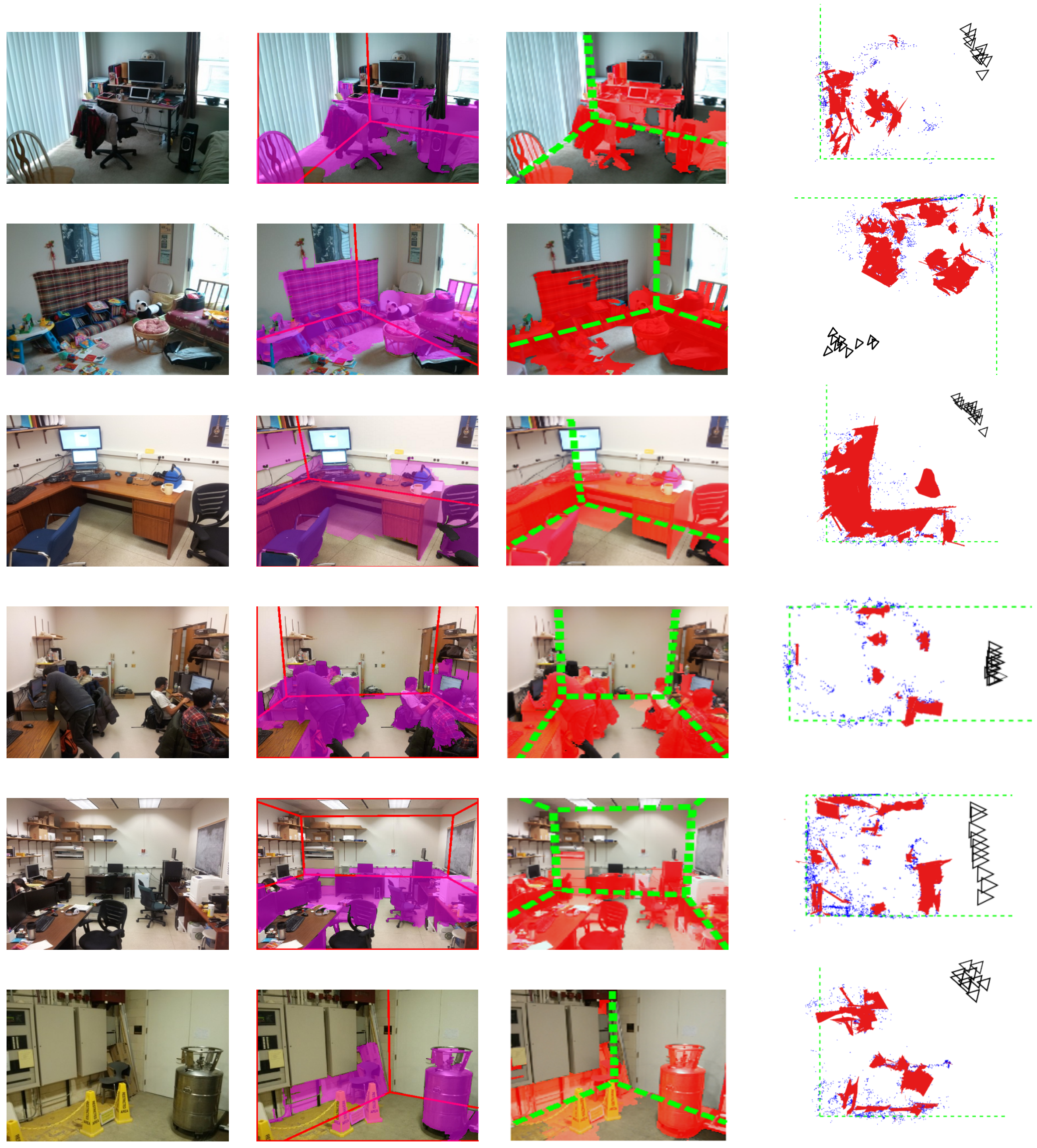


Figure 6.9: Example results. First column: one of the 10 input images of the scene. Second column: result of *Hedau et al.* (2009). The pink region is recognized as objects. The red lines show the estimated room layout. Third column: our result. The red region is recognized as objects. The green lines indicate the estimated room. Fourth column: floor occupancy map shown from the top view of the scene. The green dashed lines show the extent of the room. The blue points are SFM points. The red show regions in 3D. The triangles visualize the camera locations.

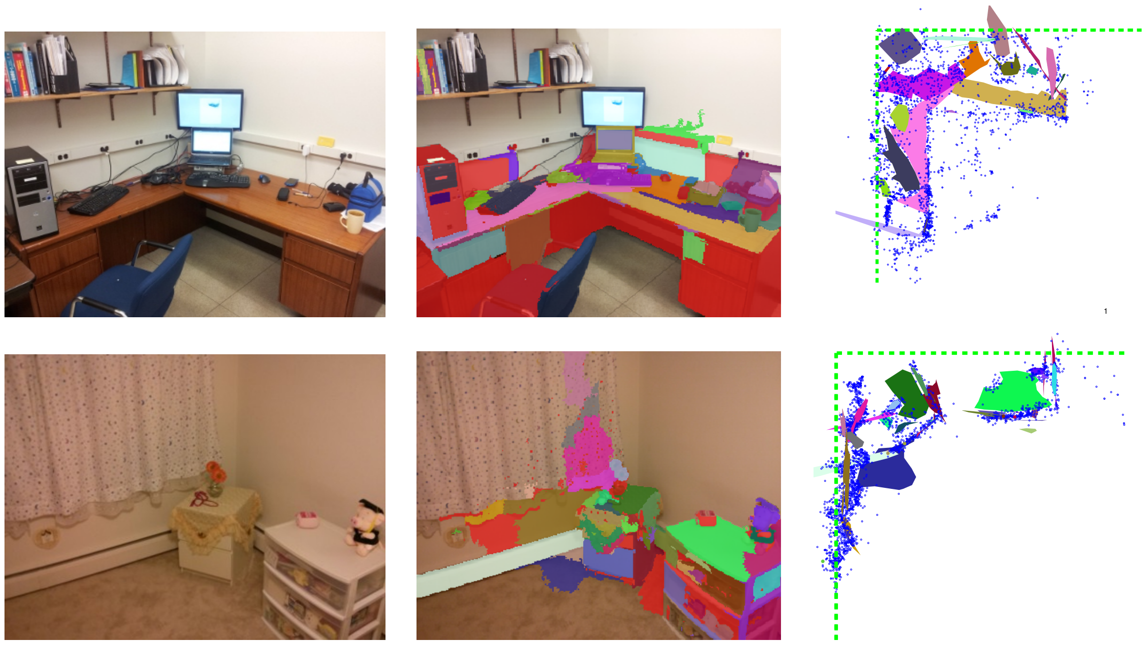


Figure 6.10: 3D reconstruction of objects. **First column**: an input image (among 10 input images in total). **Second column**: region segments in this images. The colored regions are the non-wall objects estimated by our framework. The red indicates the object surfaces whose 3D positions cannot be estimated (see Sec. 4.1.4 for more details). Other colors indicate the object surfaces that can be recovered in 3D space. **Third column**: top view of the reconstructed scene. The green dashed lines show the estimated boundary of the room. The blue dots are SFM points. The colored regions are the object surfaces in 3D space. The colored regions in the second column and the third column are in correspondence. Note that the colored regions in the third column are not all the surfaces we can reconstruct. The third column only shows the regions corresponding to the image in the 2nd column.

CHAPTER VII

Conclusion and Future Work

The proposed scene understanding framework significantly advances the performance of the state-of-the-art semantic recognition and geometric reconstruction algorithms. In the future, one of our focuses is on enhancing the scalability of the framework. The current MATLAB implementation of the scene understanding method consumes tens of minutes in a general setup given a few images (Chapter III) and ~10 minutes for a room layout estimation application (Chapter VI). The majority of the estimation time is taken by obtaining measurements (feature point matching, object detection, region classification) and joint energy optimization. Recent progress on fast object detection (e.g. *Pedersoli et al.* (2011); *Song et al.* (2012); *Dubout and Fleuret* (2012); *Dean et al.* (2013)) demonstrate the possibility of obtaining image measurements within a much shorter time compared to conventional approach implemented with a single CPU thread. On the other hand, to shorten the time of joint energy optimization process, other types of optimization may be adopted to replace the current sampling method (Chapter III). The sampling method is able to overcome local optima but intrinsically slow. If the initialization of the optimization process can be obtained with a higher accuracy, e.g., by means of providing more accurate object detection results, a greedy search approach can be chosen, which may be significantly faster than the sampling method.

Another limitation of the current framework is its inability to handle dynamic scenes. Many scenes in the world are dynamic, i.e., they contain moving objects. Moving objects violate the static scene assumption held by our framework and many other reconstruction methods (*Nister* (2004); *Snavely et al.* (2008); *Furukawa and Ponce* (2010); *Triggs et al.* (1999)). There are existing research topics related to dynamic scene reconstruction. Non-rigid structure-from-motion tackles of the problem of reconstructing deformable objects (*Xiao et al.* (2004); *Bregler et al.* (2000); *Torresani et al.* (2001, 2008); *Rabaud and Belongie* (2008); *Gotardo and Martinez* (2011)). Multi-body structure-from-motion reconstructs a scene that consists of multiple rigid parts (*Webb and Aggarwal* (1982); *Costeira and Kanade* (1998); *Schindler et al.* (2008); *Fitzgibbon and Zisserman* (2000); *Ozden et al.* (2010)). However, most non-rigid SFM and multi-body SFM algorithms solve the reconstruction problem only from the geometric perspective without leveraging recognition results. One key challenge in reconstructing dynamic scenes is separating non-static objects from static background. Conventional methods identify the non-static objects using geometric cues (*Webb and Aggarwal* (1982); *Costeira and Kanade* (1998); *Schindler et al.* (2008); *Fitzgibbon and Zisserman* (2000); *Ozden et al.* (2010)) which is proved to be very sensitive to noise and outliers. In addition to geometric cues, the results of object recognition can be used to guide the motion identification process. For example, if people and cars can be detected in an urban scene, we can confidently treat the rest part of the scene as static background. Another challenge in reconstructing dynamic scenes is choosing appropriate motion or deformation priors for estimating the dynamic parts. Conventional non-rigid SFM methods usually apply a global prior. Instead, by using the object category information extracted from recognition results, individual priors can be learned and applied for each object category.

In conclusion, this thesis proposes new geometric and semantic scene understanding methods. Chapter II tackles this problem if a single image is given as input, while Chapter III tackles this problem if multiple images are given as input. To relate objects from different images, which is a critical step in the scene understanding process, a new approach to matching objects across images is introduced in Chapter IV. A new algorithm for reconstructing 3D object shapes is proposed in Chapter V, which recovers the details of scenes on top of the results of reconstructing a scene as a blocks world. Finally, the functionality and the robustness of the proposed framework are demonstrated in an application for estimating indoor room layouts (Chapter VI). In the future, open challenges to be addressed include scalability, efficiency, and the ability to cope with dynamic scenes.

BIBLIOGRAPHY

BIBLIOGRAPHY

Agarwal, S., N. Snavely, I. Simon, S. Seitz, and R. Szeliski (2009), Building Rome in a day, in *International Conference on Computer Vision*, pp. 72–79.

Bao, S. Y., M. Sun, and S. Savarese (2010), Toward Coherent Object Detection And Scene Layout Understanding, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 65–72.

Bao, S. Y., M. Bagra, and S. Savarese (2011a), Semantic Structure From Motion with Object and Point Interactions, in *IEEE Workshop on Challenges and Opportunities in Robot Perception (in conjunction with ICCV)*, pp. 982–989.

Bao, S. Y., M. Sun, and S. Savarese (2011b), 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 (2012a), Semantic Structure from Motion with Points, Regions, and Objects, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2703–2710.

Bao, S. Y., Y. Xiang, and S. Savarese (2012b), Object Co-detection, in *Proc. of European Conference of Computer Vision*, pp. 86–101.

Bao, S. Y.-Z., and S. Savarese (2011a), Semantic Structure From Motion, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2025–2032.

Bao, Y., and S. Savarese (2011b), Webpage of Semantic Structure From Motion, http://www.eecs.umich.edu/vision/projects/ssfm/index.html.

Bao, Y., and S. Savarese (2013), *Outdoor and Large-Scale Real-World Scene Analysis*, chap. Semantic Structure from Motion: a Novel Framework for Joint Object Recognition and 3D Reconstruction, Springer.

Bao, Y., M. Chandraker, Y. Lin, and S. Savarese (2013), Dense Object Reconstruction Using Semantic Priors, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1264–1271.

Barrow, H., and J. Tenenbaum (1978), Recovering intrinsic scene characteristics from images, *Computer Vision Systems*.

Batra, D., A. Kowdle, D. Parikh, J. Luo, and T. Chen (2010), iCoseg: Interactive co-segmentation with intelligent scribble guidance, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Belongie, S., J. Malik, and J. Puzicha (2002), Shape Matching and Object Recognition Using Shape Contexts, *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Berg, A., T. Berg, and J. Malik (2005), Shape matching and object recognition using low distortion correspondences, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Berg, A., F. Grabler, and J. Malik (2007), Parsing Images of Architectural Scenes, in *International Conference on Computer Vision*.

Biederman, I. (1981), On the semantics of a glance at a scene, in *Perceptual Organization*, edited by M. Kubovy and J. Pomerantz, chap. 8.

Biederman, I., R. Mezzanotte, and J. Rabinowitz (1982), Scene perception: Detecting and judging objects undergoing relational violations, *Cognitive Psychology*, *14*(2), 143–177.

Blanz, V., and T. Vetter (1999), A morphable model for the synthesis of 3D faces, in *Proceedings of SIGGRAPH*.

Bookstein, F. L. (1989), Principal Warps: Thin-Plate Splines and the Decomposition of Deformations, *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Bregler, C., A. Hertzmann, and H. Biermann (2000), Recovering non-rigid 3D shape from image streams, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 690–696, IEEE.

Brooks, R. A. (1981), Model-based three dimensional interpretations of two dimensional images, in *IJCIA*, pp. 619–624, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Brostow, G. J., J. Shotton, J. Fauqueur, and R. Cipolla (2008), Segmentation and Recognition Using Structure from Motion Point Clouds, in *Proceedings of European Conference on Computer Vision*.

Choi, W., and S. Savarese (2010), Multiple Target Tracking in World Coordinate with Single, Minimally Calibrated Camera, in *Proceedings of European Conference on Computer Vision*.

Chui, H., and A. Rangarajan (2003), A new point matching algorithm for non-rigid registration, *Computer Vision and Image Understanding*.

Cignoni, P., C. Rocchini, and R. Scopigno (1998), Metro: Measuring Error on Simplified Surfaces, *CGF*.

Cootes, T., C. Taylor, D. Cooper, and J. Graham (1995), Active Shape Models - Their Training and Application, *Computer Vision and Image Understanding*.

Cornelis, N., B. Leibe, K. Cornelis, and L. Gool (2008), 3D Urban Scene Modeling Integrating Recognition and Reconstruction, *IJCV*, *78*(2-3), 121–141.

Costeira, J. P., and T. Kanade (1998), A multibody factorization method for independently moving objects, *International Journal of Computer Vision*, *29*(3), 159–179.

Dalal, N., and B. Triggs (2005), Histograms of Oriented Gradients for Human Detection, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

David, G. (2004), Distinctive image features from scale-invariant keypoints, *IJCV*.

Dean, T., M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, and J. Yagnik (2013), Fast, Accurate Detection of 100,000 Object Classes on a Single Machine, in *IEEE Conference on Computer Vision and Pattern Recognition*.

Dellaert, F., S. Seitz, S. Thrun, and C. Thorpe (2000), Feature correspondence: A markov chain monte carlo approach, in *Proceedings of Advances in Neural Information Processing System*.

Dick, A. R., P. H. S. Torr, and R. Cipolla (2004), Modelling and Interpretation of Architecture from Several Images, *IJCV*, *60*(2), 111–134.

Dryden, I., and K. Mardia (1998), *Statistical Shape Analysis*.

Dubout, C., and F. Fleuret (2012), Exact acceleration of linear object detectors, in *Computer Vision–ECCV 2012*, pp. 301–311, Springer.

Ess, A., B. Leibe, and L. V. Gool (2007), Depth and Appearance for Mobile Scene Analysis, in *International Conference on Computer Vision*.

Ess, A., B. Leibe, K. Schindler, and L. van Gool (2008), A Mobile Vision System for Robust Multi-Person Tracking, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Farhadi, A., M. K. Tabrizi, I. Endres, and D. A. Forsyth (2009), A Latent Model of Discriminative Aspect, in *International Conference on Computer Vision*.

Fei-Fei, L., R. Fergus, and A. Torralba (2007), Recognizing and Learning Object Categories.

Felzenszwalb, P., and D. Huttenlocher (2000), Pictorial structures for object recognition., in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2066–2073.

Felzenszwalb, P., R. Girshick, D. McAllester, and D. Ramanan (2010), Object Detection with Discriminatively Trained Part-Based Models, *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

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

Ferrari, V., T. Tuytelaars, and L. V. Gool (2003), Wide-baseline muliple-view Correspondences, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Ferrari, V., T. Tuytelaars, and L. V. Gool (2006), Simultaneous Object Recognition and Segmentation from Single or Multiple Model Views, *IJCV*.

Ferrari, V., F. Jurie, and C. Schmid (2007), Accurate Object Detection with Deformable Shape Models Learnt from Images, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Ferrari, V., L. Fevrier, F. Jurie, and C. Schmid (2008), Groups of Adjacent Contour Segments for Object Detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(1), 36–51, doi:10.1109/TPAMI.2007.1144.

Fitzgibbon, A. W., and A. Zisserman (2000), Multibody structure and motion: 3-d reconstruction of independently moving objects, in *Proceedings of European Conference on Computer Vision*.

Flint, A., D. Murray, and I. Reid (2011), Manhattan scene understanding using monocular, stereo, and 3D features, in *International Conference on Computer Vision*.

Forsyth, D. A., J. L. Mundy, A. Zisserman, and C. A. Rothwell (1994), Using global consistency to recognise euclidean objects with an uncalibrated camera, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Fouhey, D. F., V. Delaitre, A. Gupta, A. A. Efros, I. Laptev, and J. Sivic (2012), People Watching: Human Actions as a Cue for Single-View Geometry, in *Proceedings of European Conference on Computer Vision*.

Frome, A., D. Huber, R. Kolluri, T. Bulow, and J. Malik (2004), Recognizing Objects in Range Data Using Regional Point Descriptors, *ECCV*.

Furukawa, Y., and J. Ponce (2010), Accurate, Dense and Robust Multiview Stereopsis, *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Furukawa, Y., B. Curless, S. Seitz, and R. Szeliski (2009a), Manhattan-World Stereo, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Furukawa, Y., B. Curless, S. M. Seitz, and R. Szeliski (2009b), Reconstructing Building Interiors from Images, in *International Conference on Computer Vision.*

Furukawa, Y., B. Curless, S. Seitz, and R. Szeliski (2010), Towards Internet-scale multi-view stereo, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Gallup, D., J.-M. Frahm, and M. Pollefeys (2010), Piecewise planar and non-planar stereo for urban scene reconstruction., in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Garg, R., S. Seitz, D. Ramanan, and N. Snavely (2011), Where's Waldo: Matching people in images of crowds, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Goesele, M., J. Ackermann, S. Fuhrmann, R. Klowsky, F. Langguth, P. Müandcke, and M. Ritz (2010), Scene Reconstruction from Community Photo Collections, *IEEE Computer.*

Golparvar-Fard, M., F. Pena-Mora, and S. Savarese (2009), D4AR- A 4-Dimensional Augmented Reality Model for Automating Construction Progress Data Collection, Processing and Communication, in *TCON Special Issue: Next Generation Construction IT.*

Gotardo, P. F., and A. M. Martinez (2011), Kernel non-rigid structure from motion, in *International Conference on Computer Vision*, pp. 802–809, IEEE.

Gould, S., R. Fulton, and D. Koller (2009), Decomposing a Scene into Geometric and Semantically Consistent Regions, in *International Conference on Computer Vision.*

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

Gupta, A., A. Efros, and M. Hebert (2010), Blocks world revisited: Image understanding using qualitative geometry and mechanics, in *Proceedings of European Conference on Computer Vision.*

Hanson, A. R., and E. M. Riseman (1978), Visions: A computer system for interpreting scenes, *Computer Vision Systems, 1978.*

Hartley, R. I., and A. Zisserman (2000), *Multiple View Geometry in Computer Vision*, Cambridge University Press.

Hedau, V., D. Hoiem, and D. Forsyth (2009), Recovering the Spatial Layout of Cluttered Rooms, in *International Conference on Computer Vision*.

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

Hedau, V., D. Hoiem, and D. Forsyth (2012), Recovering Free Space of Indoor Scenes from a Single Image, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Helmer, S., D. Meger, M. Muja, J. J. Little, and D. G. Lowe (2010), Multiple Viewpoint Recognition and Localization, in *ACCV*.

Hernández, C., and G. Vogiatzis (2010), Shape from Photographs: A Multi-view Stereo Pipeline, in *Computer Vision*, Studies in Computational Intelligence.

Hochbaum, D., and V. Singh (2009), An efficient algorithm for Co-segmentation, in *International Conference on Computer Vision*.

Hoiem, D., A. Efros, and M. Hebert (2005), Geometric Context from a Single Image., in *Int. Conf. on Computer Vision*.

Hoiem, D., A. A. Efros, and M. Hebert (2006), Putting Objects in Perspective, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 2137–2144.

Hoiem, D., A. Efros, and M. Hebert (2007), Recovering surface layout from an image, *IJCV*.

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

Hoiem, D., A. A. Efros, and M. Hebert (2008b), Closing the Loop on Scene Interpretation, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Hsiao, E., A. Collet, and M. Hebert (2010), Making specific features less discriminative to improve point-based 3D object recognition, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Huber, D. (2001), Automatic 3D modeling using range images obtained from unknown viewpoints, in *Int. Conf. on 3-D Digital Imaging and Modeling*.

Jiang, T., F. Jurie, and C. Schmid (2009), Learning shape prior models for object matching, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Jiang, Y., M. Lim, and A. Saxena (2012), Learning Object Arrangements in 3D Scenes using Human Context, in *ICML*.

Kazhdan, M., M. Bolitho, and H. Hoppe (2006), Poisson surface reconstruction, in *Proceedings of Symposium on Geometry Processing.*

Khan, S. M., and M. Shah (2006), A Multi-view Approach to Tracking People in Dense Crowded Scenes using a planar Homography Constraint, in *Proceedings of European Conference on Computer Vision.*

Kirkpatrick, S., C. Gelatt, and M. Vecchi (1983), Optimization by simulated annealing, *Science.*

Kolmogorov, V., and R. Zabih (2002), Multi-camera Scene Reconstruction via Graph Cuts, in *Proceedings of European Conference on Computer Vision.*

Koppula, H., A. Anand, T. Joachims, and A. Saxena (2011), Semantic labeling of 3d point clouds for indoor scenes, in *Proceedings of Advances in Neural Information Processing System.*

Ladicky, L., C. Russell, P. Kohli, and P. Torr (2010), Graph Cut based Inference with Co-occurrence Statistics, in *Proceedings of European Conference on Computer Vision.*

Ladicky, L., P. Sturgess, C. Russell, S. Sengupta, Y. Bastanlar, W. Clocksin, and P. Torr (2011), Joint optimization for object class segmentation and dense stereo reconstruction, *IJCV.*

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

Lazebnik, S., C. Schmid, and J. Ponce (2004), Semi-local Affine Parts for Object Recognition, in *BMVC.*

Lazebnik, S., C. Schmid, and J. Ponce (2006), Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in *CVPR.*

Lee, D. C., M. Hebert, and T. Kanade (2009), Geometric reasoning for single image structure recovery, in *In proc. CVPR.*

Lee, D. C., A. Gupta, M. Hebert, and T. Kanade (2010), Estimating Spatial Layout of Rooms using Volumetric Reasoning about Objects and Surfaces, in *Proceedings of Advances in Neural Information Processing System.*

Leibe, B., and B. Schiele (2004), Scale Invariant Object Categorization Using a Scale-Adaptive Mean-Shift Search., in *DAGM Annual Pattern Recognition Symposium.*

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

Leibe, B., A. Leonardis, and B. Schiele (2006), An Implicit Shape Model for Combined Object Categorization and Segmentation, in *Toward Category-Level Object Recognition.*

Leotta, M. J., and J. L. Mundy (2009), Predicting High Resolution Image Edges with a Generic, Adaptive, 3D Vehicle Model, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Li, L.-J., R. Socher, and L. Fei-Fei (2009), Towards Total Scene Understanding:Classification, Annotation and Segmentation in an Automatic Framework, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Liebelt, J., C. Schmid, and K. Schertler (2008), Viewpoint-independent object class detection using 3D Feature Maps, *CVPR.*

Lloyd, S. (1982), Least squares quantization in PCM, *IEEE Transactions on Information Theory*, *28*(2), 129–137.

Lowe, D. (1999), Object recognition from local scale-invariant features, in *International Conference on Computer Vision.*

Lowe, D. (2004), Distinctive image features from scale-invariant keypoints, *IJCV.*

Matas, J., O. Chum, M. Urban, and T. Pajdla (2004), Robust wide-baseline stereo from maximally stable extremal regions, *Image and Vision Computing.*

Munsell, B., P. Dalal, and S. Wang (2008), Evaluating Shape Correspondence for Statistical Shape Analysis: A Benchmark Study, *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Nister, D. (2004), An efficient solution to the five-point relative pose problem, *IEEE Transactions on Pattern Analysis and Machine Intelligence.*

Nister, D., and H. Stewenius (2006), Scalable recognition with a vocabulary tree, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Ohta, Y. (1985), *Knowledge-based interpretation of outdoor natural color scenes*, Pitman Publishing, Inc., Marshfield, MA, USA.

Ozden, K. E., K. Schindler, and L. Van Gool (2010), Multibody structure-from-motion in practice, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(6), 1134–1141.

Ozuysal, M., V. Lepetit, and P. Fua (2009), Pose Estimation for Category Specific Multiview Object Localization, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Palmer, S. (1999), *Vision Science: Photons to Phenomenology*, The MIT Press.

Pandey, G., J. McBride, S. Savarese, and R. Eustice (2010), Extrinsic calibration of a 3d laser scanner and an omnidirectional camera, in *7th IFAC Symposium on Intelligent Autonomous Vehicles*.

Pauly, M., N. J. Mitra, J. Giesen, M. Gross, and L. J. Guibas (2005), Example-based 3D scan completion, in *Proceedings of Symposium on Geometry Processing*.

Payet, N., and S. Todorovic (2011a), Scene Shape from Texture of Objects, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Payet, N., and S. Todorovic (2011b), Scene shape from texture of objects, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Pedersoli, M., A. Vedaldi, and J. Gonzalez (2011), A coarse-to-fine approach for fast deformable object detection, in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1353–1360, IEEE.

Pero, L. D., J. Bowdish, D. Fried, B. Kermgard, E. L. Hartley, and K. Barnard (2012), Bayesian geometric modeling of indoor scenes, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Pollefeys, M., and L. V. Gool (2002), From images to 3D models, *Commun. ACM*, *45*(7), 50–55.

Rabaud, V., and S. Belongie (2008), Re-thinking non-rigid structure from motion, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Ren, X., and J. Malik (2003), Learning a classification model for segmentation, in *International Conference on Computer Vision*.

Reynolds, M., J. Doboš, L. Peel, T. Weyrich, and G. J. Brostow (2011), Capturing Time-of-Flight Data with Confidence, in *CVPR*.

Rohr, K., H. S. Stiehl, R. Sprengel, W. Beil, T. M. Buzug, J. Weese, and M. H. Kuhn (1996), Point-Based Elastic Registration of Medical Image Data Using Approximating Thin-Plate Splines, in *Int. Conf. on Visualization in Biomedical Computing*.

Rosten, E., R. Porter, and T. Drummond (2010), FASTER and better: A machine learning approach to corner detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Rother, C., V. Kolmogorov, T. Minka, and A. Blake (2006), Cosegmentation of Image Pairs by Histogram Matching, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

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, *IJCV*.

Russell, B. C., A. Torralba, K. P. Murphy, and W. T. Freeman (2005), Labelme: A database and web-based tool for image annotation.

Rusu, R., Z. Marton, N. Blodow, M. Dolha, and M. Beetz (2008), Towards 3D Point cloud based object maps for household environments, *Robotics and Autonomous Systems*, *56*(11).

Savarese, S., and L. Fei-Fei (2007), 3D generic object categorization, localization and pose estimation., in *International Conference on Computer Vision*.

Saxena, A., M. Sun, and A. Y. Ng (2009), Make3D: Learning 3D Scene Structure from a Single Still Image, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(5), 824–840, doi:10.1109/TPAMI.2008.132.

Schaffalitzky, F., and A. Zisserman (2001), Viewpoint invariant texture matching and wide baseline stereo, in *International Conference on Computer Vision*.

Schindler, K., D. Suter, and H. Wang (2008), A model-selection framework for multibody structure-and-motion of image sequences, *International Journal of Computer Vision*, *79*(2), 159–177.

Schwing, A., and R. Urtasun (2012), Efficient Exact Inference for 3D Indoor Scene Understanding, in *Proceedings of European Conference on Computer Vision*.

Seitz, S. M., B. Curless, J. Diebel, D. Scharstein, and R. Szeliski (2006), A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Silberman, N., D. Hoiem, P. Kohli, and R. Fergus (2012), Indoor Segmentation and Support Inference from RGBD Images, in *Proceedings of European Conference on Computer Vision*.

Snavely, N., S. M. Seitz, and R. S. Szeliski (2008), Modeling the World from Internet Photo Collections, *International Journal of Computer Vision*, (2).

Soatto, S., and P. Perona (1998), Reducing "Structure from Motion": a general framework for dynamic vision. Part 1: modeling., *International Journal of Computer Vision*, *20*.

Song, H. O., S. Zickler, T. Althoff, R. Girshick, M. Fritz, C. Geyer, P. Felzenszwalb, and T. Darrell (2012), Sparselet models for efficient multiclass object detection, in *Computer Vision–ECCV 2012*, pp. 802–815, Springer.

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.

Sudderth, E. B., A. Torralba, W. T. Freeman, and A. S. Willsky (2006), Depth from Familiar Objects: A Hierarchical Model for 3D Scenes, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2410–2417, IEEE Computer Society, Washington, DC, USA, doi:10.1109/CVPR.2006.97.

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 *Proceedings of European Conference on Computer Vision*.

Thomas, A., V. Ferrari, B. Leibe, T. Tuytelaars, and L. Van Gool (2007), Depth-From-Recognition: Inferring Meta-data by Cognitive Feedback, in *International Conference on Computer Vision*.

Torresani, L., D. B. Yang, E. J. Alexander, and C. Bregler (2001), Tracking and modeling non-rigid objects with rank constraints, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, pp. I–493, IEEE.

Torresani, L., A. Hertzmann, and C. Bregler (2008), Nonrigid structure-from-motion: Estimating shape and motion with hierarchical priors, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(5), 878–892.

Toshev, A., J. Shi, and K. Daniilidis (2007), Image Matching via Saliency Region Correspondences, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*.

Triggs, B., P. McLauchlan, R. Hartley, and A. Fitzgibbob (1999), Bundle adjustment: a modern synthesis, in *Vision Algorithms: Theory and Practice*.

Tsai, G., C. Xu, J. Liu, and B. Kuipers (2011), Real-time indoor scene understanding using Bayesian filtering with motion cues, in *International Conference on Computer Vision*.

Tuytelaars, T., and L. Van Gool (2000), Wide baseline stereo matching based on local, affinely invariant regions, in *British Machine Vision Conference*.

Varma, M., and R. Garg (2007), Locally invariant fractal features for statistical texture classification, in *International Conference on Computer Vision*.

Vogiatzis, G., C. Hernandez, P. Torr, and R. Cipolla (2007), Multiview Stereo via Volumetric Graph-Cuts and Occlusion Robust Photo-Consistency, *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

von Gioi, R. G., J. Jakubowicz, J.-M. Morel, and G. Randall (2012), LSD: a Line Segment Detector, *Image Processing On Line*.

Wang, H., S. Gould, and D. Koller (2010), Discriminative Learning with Latent Variables for Cluttered Indoor Scene Understanding, in *Proceedings of European Conference on Computer Vision.*

Webb, J. A., and J. K. Aggarwal (1982), Structure from motion of rigid and jointed objects, *Artificial Intelligence*, *19*(1), 107–130.

Wu, B., and R. Nevatia (2007), Detection and Tracking of Multiple, Partially Occluded Humans by Bayesian Combination of Edgelet based Part Detectors, *IJCV.*

Wu, C., S. Agarwal, B. Curless, and S. Seitz (2012), Schematic Surface Reconstruction, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Xiang, Y., and S. Savarese (2012a), Estimating the Aspect Layout of Object Categories, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Xiang, Y., and S. Savarese (2012b), Estimating the Aspect Layout of Object Categories, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.*

Xiao, J., and Y. Furukawa (2012), Reconstructing the World's Museums, in *Proceedings of European Conference on Computer Vision.*

Xiao, J., J.-x. Chai, and T. Kanade (2004), A closed-form solution to non-rigid shape and motion recovery, in *Proceedings of European Conference on Computer Vision*, pp. 573–587, Springer.

Zhang, Z., A. Ganesh, X. Liang, and Y. Ma (2010), TILT: Transform Invariant Low-rank Textures, *International Journal of Computer Vision.*

Zia, M. Z., M. Stark, B. Schiele, and K. Schindler (2011), Revisiting 3D Geometric Models for Accurate Object Shape and Pose, in *ICCV Workshop on 3D representation and recognition (3dRR-11).*

Zitnick, C., N. Jojic, and S. Kang (2005), Consistent segmentation for optical flow estimation, in *International Conference on Computer Vision.*