Learning Single-view 3D Reconstruction of Objects and Scenes

Shubham Tulsiani

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2018-93
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2018/EECS-2018-93.html

July 26, 2018

Copyright © 2018, by the author(s).
All rights reserved.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission.

Learning Single-view 3D Reconstruction of Objects and Scenes

By

Shubham Tulsiani

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Jitendra Malik, Chair
Professor Alexei A. Efros
Professor Bruno Olshausen

Summer 2018

Learning Single-view 3D Reconstruction of Objects and Scenes

Copyright 2018
by
Shubham Tulsiani

Abstract

Learning Single-view 3D Reconstruction of Objects and Scenes

by

Shubham Tulsiani

Doctor of Philosophy in Computer Science

University of California, Berkeley

Professor Jitendra Malik, Chair

We address the task of inferring the 3D structure underlying an image, in particular focusing on two questions – how we can plausibly obtain supervisory signal for this task, and what forms of representation should we pursue. We first show that we can leverage image-based supervision to learn single-view 3D prediction, by using geometry as a bridge between the learning systems and the available indirect supervision. We demonstrate that this approach enables learning 3D structure across diverse setups e.g. learning deformable models, predictive models for volumetric 3D, or inferring textured meshes. We then advocate the case for inferring interpretable and compositional 3D representations. We present a method that discovers the coherent compositional structure across objects in a unsupervised manner by attempting to assemble shapes using volumetric primitives, and then demonstrate the advantages of predicting similar factored 3D representations for complex scenes.

To my parents, for their boundless love and support

Contents

List of Figures

List of Tables

Acknowledgments

Although the title page of this thesis may identify me as the author, I cannot help but consider it a blatant misrepresentation of where the credit is actually due. I am fortunate to have been around some wonderful people throughout this journey, and none of this work would have been even a remote possibility if not for their continual support and guidance. I would like to take this opportunity to thank them for making this possible.

First, I am deeply indebted to my advisor Jitendra Malik for helping me find my feet, and supporting me through all the stumbles in between. Jitendra has taught me how to do research, how to choose a problem, the importance of being well-read, and considering the big picture (as well as Figure 1). Instead of making people around him work on his ideas, Jitendra enables them to pursue their own, and for this I feel extremely lucky to have been a part of his group.

While I'm recounting my good fortune, I am glad that Alexei (Alyosha) Efros decided to come back to Berkeley. Alyosha is an extraordinarily selfless person, and has generously spent long hours helping me over the course of various projects. He has inculcated in me the value of looking at the pixels, the courage to pursue different ideas, and introduced me to the joys of hiking. I am grateful to him for having positively impacted my research, education, and general life.

The environment in Berkeley is possibly unique in that it allows interacting and collaborating with faculty across areas and institutions. I have greatly benefited from this opportunity, and would like to thank Trevor Darrell and Bruno Olshausen for sharing their insights and valuable feedback over the years, Leo Guibas for taking the time to collaborate across the bay and ignite my interest in graphics and geometry, and Pieter Abbeel for an invaluable learning experience on how to organize a course. I spent a formative summer in Cambridge, learning about building systems that really work and the benefits of good optimization techniques, and would like to thank Jonathan Taylor, Jamie Shotton, Andrew Fitzgibbon, and Daniel Tarlow for the opportunity. I am also indebted to Noah Snavely and Richard Tucker for allowing me to spend three wonderful months in New York working with them, and helping me learn how to code better, make (arguably) good coffee, and make progress towards

larger goals by choosing smaller steps. While acknowledging people who've helped shape my research direction, I'd be remiss not to thank the anonymous reviewers for various submissions, in particular the ones who helped reject submissions that admittedly should have been, and helping me become a better researcher through their honest feedback.

The faculty and mentors helped me throughout in choosing research directions and ideas to pursue, but it has been my collaborators across these projects who really helped bring these ideas to life. Abhishek Kar has simultaneously been a great mentor, friend and co-author, and I am grateful for him helping me find my place in research (figuratively), and in Berkeley (literally). I am glad to have been able to collaborate with Tinghui Zhou, who made me more culturally literate by dragging me to Broadway shows in New York, and hope that some of his ability to have deep and simple insights into well-chosen problems rubs off on me. Saurabh Gupta has taught me the importance of considering the details, and often questioning the basic assumptions.

The vision group at Berkeley has also been home to numerous brilliant postdocs, some of whom I had the chance to work with, and while the departmental mailing lists may not acknowledge their existence, I certainly will. Joao Carreira instructed me on how to make good figures, and made working fun through his creativity and cheerfulness. Hao Su taught me about the insights to be gained via simple experiments, and I am glad that he bore the pains of a long commute across the bay for weekly meetings. I learned from Christian Häne about how to do sensible things, and patiently persist until things work, and am grateful for his endless supply of swiss chocolates. I'm thankful to David Fouhey for pushing me towards fruitful directions, and sharing his love of Pittsburgh and the keys to life over plenty of conversations. I am glad for the chance to have worked with Angjoo Kanazawa, whose boundless enthusiasm, an amazing ability to get things done and a knack for producing pretty results would make her the perfect collaborator (if not for her misguided love of emacs).

Everyone at Berkeley, though their comments, discussion, and willingness to be inclusive, has contributed to my stay being a great experience. I'd particularly like to thank people in Jitendra and Alyosha's groups over the years: Andrew, Ashish, Bharath, Deepak, Georgia, Judy, Jun-Yan, Katerina, Ke, Pablo, Panna, Philipp, Phillip, Pulkit, Richard, Ross, Shiry, Weicheng, Yong Jae and Zhe, for being there, and shaping my opinions and actions through various discussions. I am also thankful for the role Angie has played in making my stay comfortable, and acting as a shield against any logistical and bureaucratic issues.

Pursuing a PhD can sometimes become a lonely enterprise, and I have been fortunate to have an amazing group of friends around who have helped me in testing

times, celebrated with me in good ones, and most importantly, have tolerated me in normal times. I am thankful to Niharika for always being there over these years to support me, and giving me something to hold on to when the going was tough. Thank you Abhishek, Anurag, Bharath, Neeraja, Nikunj, Radhika, Saurabh, Shromona, Sreeta, Somil, Sunanda, Tejas and Vivek for making these years enjoyable. I am really grateful to my brother for his constant support, wisdom and help throughout these years, and for paving for himself a new path so that following it became much easier. Finally, I cannot adequately express in words my gratitude towards my parents who raised me to be inquisitive, supported me in all endeavors, gave me the freedom to pursue my dreams in a far-away country, and taught me all the good things I know.

Chapter 1

Introduction

We live in a complex world, relying on the 2D retinal projections of the 3D physical reality to understand and act in it. We can determine while driving whether one car is farther away than the other. When searching for a spot to sit, we can imagine the 3D shape of a chair, even though we may have had only a single glance at it. We understand the the floor, though hidden from view by a couch, continues beneath it. These are merely some of the endless list of judgments we can easily make from our visual inputs, all of which serve to highlight one aspect – that we humans have the remarkable capability to perceive 3D from 2D.

A long-standing goal in computer vision, dating back to the very first attempts [112], has been to build computational systems that have similar capabilities – that of being able to infer, from a single input image, the underlying 3D structure. Unfortunately, this task is mathematically ill-posed. As depicted in several excellent illustrations *e.g.* the workshop metaphor by Adelson and Pentland [2], or Sinha and Adelson's polyhedral line drawings [128], there are infinite possible 3D geometries than can explain a given 2D image. However, our world is structured, and not all of these are equally likely. We humans infer (the likely) 3D by relying on our knowledge about these regularities in the world, and our attempts at making computers understand 3D have similarly relied on incorporating prior knowledge for inference.

Initial attempts towards the goal of single-view 3D inference leveraged hand-designed priors, either in form of explicit constraints regarding what entities comprised the scene [54, 97, 112], or as statistical assumptions regarding the process of image formation [10, 67, 100]. An alternate approach has been to rely on training data, and implicitly learn these priors in a data-driven manner [65, 122]. With the recent advent of deep learning, the trend of increasingly relying on direct supervision for the end-task has further continued, yielding impressive results for various other tasks such as image recognition [84], object detection [47], segmentation [124] *etc.* by

relying on large amounts of annotations [114]. However, this paradigm of supervised learning is not easily applicable for learning 3D inference – it is costly and painstaking, if not impossible, to obtain such 3D supervision at a large scale. An important challenge towards learning single-view 3D reconstruction, therefore, is to enable learning without requiring such tedious explicit 3D supervision.

In addition to the forms of priors leveraged, another aspect that varies across 3D reconstruction approaches is the 3D representation pursued. A common practice is to directly infer, as a whole, the representation of the entire object or scene in the form of a volumetric or pixelwise representation. While easier to incorporate in prediction frameworks, this ignores an important aspect of the underlying world. The very complex and varied structures we see are actually comprised of simpler entities – a living room contains chairs, tables, walls, floor *etc.*, a chair, in turn, has legs and a seat. We humans leverage this aspect, reasoning about the world as being composed of separate objects, surfaces *etc.*, and we argue that our computational 3D inference systems should also strive for representations that reflect this compositional structure.

In this thesis, we address these two challenges regarding the forms of supervision required to learn 3D reconstruction, and the desired representations to be pursued. In Part I, we demonstrate that we can learn to predict 3D using only image based training data, and without requiring direct 3D supervision. In Part II, we show that we can discover and predict compositional 3D representations.

Part I: Learning 3D via Geometric Consistency. While acquiring direct supervision for the 3D structure underlying images is difficult, it pragmatically and ecologically more plausible to obtain indirect supervision in the form of (multiple of single) 2D views of various instances. In this part, we show that we can learn to predict 3D using such supervision, by allowing geometry to act as a bridge between the inferred 3D representations and the available 2D supervision. We can do so because we know precisely, in the form of concise geometric equations, the relationship between a 3D representation and the corresponding 2D projections. We can therefore learn to infer 3D structure by enforcing that our inferences are geometrically consistent with the available 2D training data.

We examine various scenarios where this idea of learning via geometric consistency is applicable. We first propose in Chapter 2 an approach to learn category-specific deformable 3D models by relying on an annotated image collection. This enables us to represent the space of possible shapes of categories using learned linear models, and using estimates from recognition systems, robustly fit these deformable models to obtain the shape for a new instance.

In Chapter 3, we pursue a more expressive, convolutional neural network (CNN)

based prediction model. Towards this, we study the notion of consistency between a volumetric 3D shape and a 2D observation and propose a differentiable formulation which allows computing gradients of the 3D shape given an observation from an arbitrary view. We show that this formulation can be incorporated in a learning framework to leverage different types of multi-view observations e.g. foreground masks, depth, color images, semantics etc. as supervision for learning single-view 3D prediction, and demonstrate its applicability across various setups.

In Chapter 4, we then aim to infer a textured 3D mesh representation using a CNN based prediction model, while only relying on an image collection as supervision. To enable this, we leverage the representations and objectives analogous to those used by deformable model learning methods, but incorporate these in a CNN based prediction framework. We demonstrate that this allows us to recover the 3D shape, camera, and texture of an object from a single image.

Part II: Inferring Compositional 3D Representation. While in Part I our aim is to enable learning 3D using plausible forms of supervision, in Part II we focus on the ability to predict compositional representations. Our goal is to represent and understand complex objects and scenes in terms of the underlying simpler entities. Towards this, we first address the question of how we can learn what these entities are, and then demonstrate the benefits of inferring these compositional representations.

In Chapter 5, we consider a collection of 3D shapes, and present an unsupervised learning framework for abstracting complex shapes by learning to assemble objects using 3D volumetric primitives. In addition to generating simple and geometrically interpretable explanations of 3D objects, our framework also allows us to automatically discover and exploit consistent structure in the data. We demonstrate that this discovered consistent compositional structure can be leveraged for obtaining a consistent parsing across the instances of a shape collection, constructing an interpretable shape similarity measure, or for shape manipulation. We also demonstrate that we can learn to infer similar primitive based representation from input RGB images.

In Chapter 6, we then focus on representing the 3D structure of scenes in terms of a small set of factors: a layout representing the enclosing surfaces as well as a set of objects represented in terms of shape and pose. We present a CNN based learning approach to predict this representation, and quantitatively and qualitatively demonstrate its merits compared to pursuing alternate representations.

We finally conclude in Chapter 7, and discuss possible directions for future research.

Part I

Learning via Geometric Consistency

Chapter 2

Category-Specific Deformable 3D Models

Consider the chairs in Figure 2.1. As humans, not only can we infer at a glance that the image contains three chairs, we also construct a rich internal representation of each of them such as their locations and 3D poses. Moreover, we have a guess of their 3D shapes, even though we might never have seen these particular chairs. We can do this because we do not experience this image *tabula rasa*, but in the context of our "remembrance of things past". Previously seen chairs enable us to develop a notion of the 3D shape of chairs, which we can project to the instances in this particular image. We also specialize our representation to these particular instances (e.g. any custom decorations they might have), signalling that both top-down and bottom-up cues influence our percept [104]. In this chapter, we incorporate these principles in a computational approach for reconstructing objects given a single image. Towards this, we propose a method to learn category-specific deformable models directly from 2D annotations available in object detection datasets, and demonstrate that these models can be robustly fitted to images based on noisy pose and silhouette estimates.

The task of reconstructing objects from a single image is a challenging one – a typical image depicts many objects, each possibly belonging to a different object category; an object category, in turn, comprises instances of varying shapes, textures, size *etc.* and any particular instance may be viewed from a different viewpoint. Previous approaches to this problem can be broadly grouped into two paradigms. The paradigm of model-based object reconstruction has reflected varying preferences on model representations. Generalized cylinders [105] resulted in very compact de-

This chapter is based on joint work with Abhishek Kar, João Carreira, and Jitendra Malik. An initial version of this work appeared in CVPR, 2015 [78], and subsequently in TPAMI, 2017 [141].

Figure 2.1: Example outputs of our system, given a single image of a scene having chairs, a class that the system was exposed to during training. The coloring on the right image signals object-centric depth (we do not aim for globally consistent depths across multiple objects). Blue means close to the camera, red means far from the camera.

scriptions for certain classes of shapes, and can be used for category level descriptions, but the fitting problem for general shapes is challenging. Polyhedral models [51, 159], which trace back to the early work of Roberts [112], and CAD models [94, 107, 119], cannot perfectly deform into shapes even slightly different from those in training data, but given a set of point correspondences can be quite effective for determining approximate instance viewpoints. Some recent methods have proposed using similar instances from a collection of CAD models [70, 132] for non-parametric reconstruction but their applications have been restricted to pre-segmented online product images or recovering 3D from 2.5D object scans [134]. Here we pursue more expressive basis shape models [3, 14, 172] which establish a balance between the two extremes as they can deform but only along class-specific modes of variation.

The alternate paradigm comprises of approaches that target the problem of object reconstruction in a class or object agnostic manner, either implicitly or explicitly using generic learned 3D shape cues [65, 122], or bottom-up cues and the physics of image formation [9, 79] building upon the long tradition of shape-from-X, which traces back to seminal work by Horn [67]. These methods, while quite general, have not yet been demonstrated for 3D reconstruction – as opposed to 2.5D – and typically assume known object segmentation [9]. Some recent approaches have demonstrated the use of supervised learning techniques to implcitly learn generic cues to predict depth maps [31] and surface normals [30, 151] but these have primarily focused on inferring scene-level information which differs from our goal of perceiving the shape of objects.

In this chapter, we combine both these reconstruction paradigms - we obtain

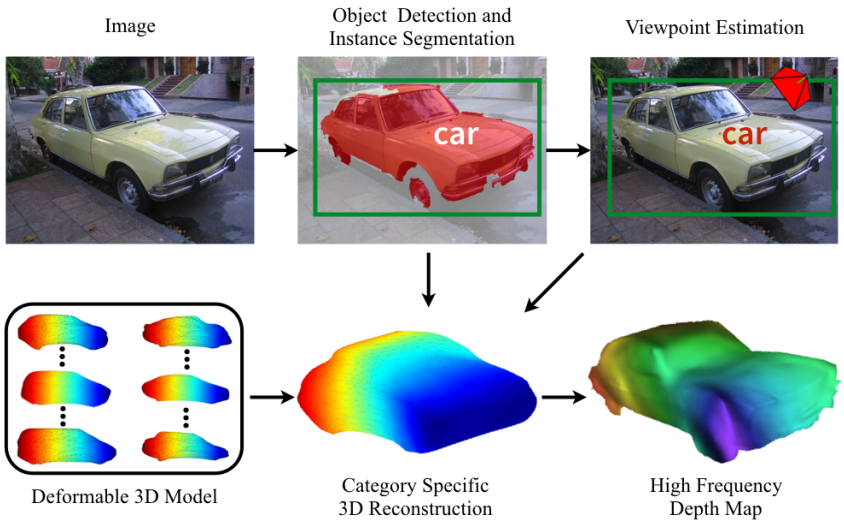

Figure 2.2: Overview of our full reconstruction method. We leverage estimated instance segmentations and predicted viewpoints to generate a full 3D mesh and a high frequency 2.5D depth map for each object in the image.

top-down shape information from our model-based reconstruction approach and complement it with bottom-up shape information obtained via an intrinsic image decomposition method. Crucially, in contrast to previous work (e.g. [9, 19, 148]), we do not require perfect knowledge of object localization and pose as our reconstruction is driven by automatic figure-ground object segmentations and viewpoint estimations.

The framework we propose to reconstruct the objects present in an image is outlined in Figure 2.2. As a first step, we leverage the recent progress made by the computer vision community in object detection [47], instance segmentation [58, 59] and viewpoint estimation [142] to identify, localize and estimate pose for the objects in the image. We then use our learned deformable 3D shape models in conjunction with the viewpoint and localization information to produce a "top-down" 3D reconstruction for the object guided primarily by category level cues. Finally, we infuse our 3D shape with high frequency local shape cues to obtain our end result - a rich 3D reconstruction of the object. We briefly outline each of the components required for the above proposed framework.

Learning Deformable 3D Models. As noted earlier, previously seen objects allow us to develop a notion of 3D shape which informs inference for new instances. We present an algorithm that can build category-specific deformable shape models from just images with 2D annotations (segmentation masks and a small set of keypoints) present in modern computer vision datasets (e.g. PASCAL VOC [34]). These learnt

shape models and deformations allow us to robustly infer shape while capturing intra-class shape variation.

Object Shape Recovery. Given an object's category, approximate localization and viewpoint, we obtain a 3D reconstruction for the corresponding object using the learned category-specific deformable shape model. We complement the top-down shape inferred via this inference with a bottom-up module that further refines our shape estimate for a particular instance. This framework allows us to capture the coarse as well as fine level shape details for objects from a single image.

This chapter is organized as follows: in Section 2.1 we describe our model learning pipeline where we estimate camera parameters for all training objects (Section 2.1.1) followed by our shape model formulation (Section 2.1.2) to learn 3D models. Section 2.2 describes our testing pipeline where we leverage our learnt models to reconstruct novel instances without assuming any annotations. We evaluate the various components of our approach in Section 2.3 and provide sample reconstructions in the wild.

2.1 Learning Deformable 3D Models

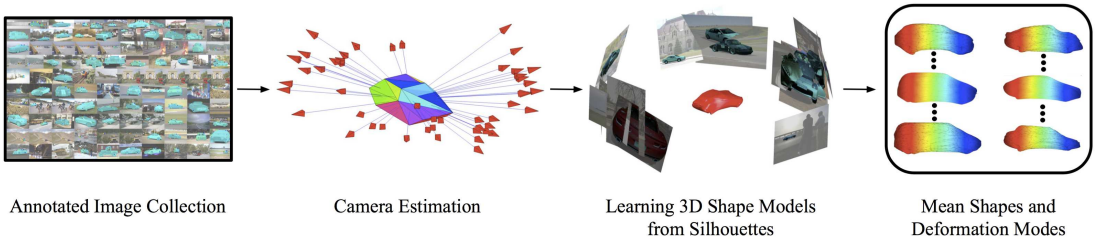

Figure 2.3: Overview of our training pipeline. We use an annotated image collection to estimate camera projection parameters which we then use along with object silhouettes to learn 3D shape models. Our learnt shape models, as illustrated in the rightmost figure are capable of deforming to capture intra-class shape variation.

We are interested in learning 3D shape models that can be robustly aligned to noisy object segmentations by incorporating top-down class-specific knowledge of how shapes from the class typically project onto the image. We want to learn such models from just 2D training images, aided by ground truth segmentations and a few keypoints, similar to [148]. Our approach operates by first estimating the projection parameters (camera) for all objects in a class using a structure-from-motion approach, followed by optimizing over a deformation basis of representative 3D shapes that

best explain all silhouettes, conditioned on the estimated cameras. We describe these two stages of model learning in the following subsections. Figure 2.3 illustrates this training pipeline of ours.

2.1.1 Camera Estimation

We use the framework of NRSfM [17] to jointly estimate the projection parameters (rotation, translation and scale) for all training instances in each class. Originally proposed for recovering shape and deformations from video [11, 17, 42, 138], NRSfM is a natural choice for camera estimation from sparse correspondences as intra-class variation may become a confounding factor if not modeled explicitly. However, the performance of such algorithms has only been explored on simple categories, such as SUV's [171] or flower petal and clown fish [109]. Closer to our work, Hejrati and Ramanan [63] used NRSfMon a larger class (cars) but need a predictive detector to fill-in missing data (occluded keypoints) which we do not assume to have here.

We closely follow the EM-PPCA formulation of Torresani *et al.* [138] and propose a simple extension to the algorithm that incorporates silhouette information in addition to keypoint correspondences to robustly recover cameras and shape bases. Energies similar to ours have been proposed in the shape-from-silhouette [149] and rigid structure-from-motion [148] literature but, to the best of our knowledge, not in conjunction with NRSfM.

NRSfM Model Formulation. We are provided with an annotated training set $T : \{(O_n, P_n)\}_{n=1}^{N}$, where O_n is the instance silhouette and $P_n \in \mathbb{R}^{2\times K}$ denotes the annotated keypoint coordinates, possibly with missing entries (occluded/truncated keypoints). The annotated keypoints P_n are projections of the underlying 3D points $W_n \in \mathbb{R}^{3\times K}$ via the projection function π_n. In the NRSfMmodel, the space of 3D keypoint locations W_n is parametrized linearly and the projection function is assumed to be weakly orthographic *i.e.* $\pi_n \equiv (c_n, R_n, T_n)$, where c_n represents scale, $R_n \in \mathbb{R}^{2\times 3}$ denotes rotation and $T_n \in \mathbb{R}^{1\times 2}$ corresponds to 2D translation. Our goal is to infer the camera parameters (c_n, R_n, T_n) as well as 3D keypoint locations W_n for all instances in the annotated training set.

Formally, our adaptation of the NRSfMalgorithm in [138] corresponds to maxi-

mizing the likelihood of the following model:

$$\begin{aligned} P_n &= c_n R_n W_n + 1^T T_n + N_n \\ W_n &= \bar{W} + \sum_{k=1}^{B} U_b z_{nb} \\ z_n &\sim \mathcal{N}(0, I), \quad N_n^k \sim \mathcal{N}(0, \sigma^2 I) \end{aligned} \tag{2.1}$$

$$\text{subject to:} \quad R_n R_n^T = I_2$$

$$\sum_{k=1}^{K} C_n^{mask}(p_{k,n}) = 0, \quad \forall n \in \{1, \cdots, N\} \tag{2.2}$$

Here, the (partially) observed keypoint locations P_n are assumed to be the projection under $\pi_n \equiv (c_n, R_n, T_n)$ of the 3D shape W_n with white noise N_n. The shape is parameterized as a factored Gaussian with a mean shape $\bar{W}$, B basis vectors $[U_1, U_2, \cdots, U_B] = U$ and latent deformation parameters z_n. Our key modification is constraint in Eq. 2.2 where C_n^{mask} denotes the Chamfer distance field of the n^{th} instance's binary mask and says that all keypoints $p_{k,n}$ of instance n should lie inside its binary mask. We observed that this results in more accurate cameras as well as more meaningful shape bases learnt from the data.

Learning. The likelihood of the above model is maximized using the EM algorithm. Missing data (occluded keypoints) is dealt with by "filling-in" the values using the forward equations after the E-step. The algorithm computes shape parameters $\{\bar{W}, U\}$, rigid body transformations $\{c_n, R_n, T_n\}$ as well as the deformation parameters $\{z_n\}$ for each training instance n. In practice, we augment the data using horizontally mirrored images to exploit bilateral symmetry in the object classes considered. We also precompute the Chamfer distance fields for the whole set to speed up computation. As shown in Figure 2.4, NRSfMallows us to reliably predict cameras while being robust to intraclass variations.

2.1.2 3D Basis Shape Model Learning

Equipped with camera projection parameters and keypoint correspondences (lifted to 3D by NRSfM) on the whole training set, we proceed to build deformable 3D shape models from object silhouettes within the same class. 3D shape reconstruction from multiple silhouettes projected from a single object in calibrated settings has been widely studied. Two prominent approaches are *visual hulls* [90] and variational methods derived from *snakes* e.g [33,117] which deform a surface mesh iteratively until

Figure 2.4: NRSfM camera estimation: Estimated cameras visualized using a 3D car wireframe.

convergence. Some interesting recent papers have extended variational approaches to handle categories [19, 22] but typically require some form of 3D annotations to bootstrap models. A recently proposed visual-hull based approach [148] requires only 2D annotations as we do for class-based reconstruction and it was successfully demonstrated on PASCAL VOC but does not serve our purpose as it makes strong assumptions about the accuracy of the segmentation and will in fact fill entirely any segmentation with a voxel layer. In contrast, we build parametric shape models for categories that compactly capture intra class shape variations. The benefits of having a model of 3D shape are manifold: 1) we are more robust to noisy inputs (silhouettes and pose) allowing us to pursue reconstruction in a fully automatic setting and 2) we can potentially sample novel shapes from an object category.

Shape Model Formulation. We model our category shapes as a deformable point cloud. As in the NRSfMmodel, we use a linear combination of basis vectors to model these deformations. Note that we learn such models from silhouettes and this is what enables us to learn deformable models without relying on point correspondences between scanned 3D exemplars [15].

The annotated training set $T : \{(O_n, P_n)\}_{n=1}^{N}$, where O_n is the instance silhouette and $P_n \in \mathbb{R}^{2\times K}$ denotes the annotated keypoint coordinates, is augmented after NRSfMto contain π_n (the projection function from world to image coordinates) and W_n (3D coordinates for a small set of keypoints). Our shape model $M = (\overline{S}, V)$ comprises of a mean shape $\overline{S}$ and deformation bases $V = \{V_1, ., V_K\}$ learnt from

the augmented training set $T : \{(O_n, \pi_n, W_n)\}_{n=1}^{N}$. Note that the π_i we obtain using NRSfM corresponds to orthographic projection but our algorithm could handle perspective projection as well.

In addition to the above, we use the following notations – $\pi(S)$ corresponds to the 2D projection of shape S, C^{mask} refers to the Chamfer distance field of the binary mask of silhouette O and $\Delta^k(p; Q)$ is defined as the squared average distance of point p to its k nearest neighbors in set Q.

Energy Formulation. We formulate our objective function primarily based on image silhouettes. For example, the shape for an instance should always project within its silhouette and should agree with the keypoints (lifted to 3D by NRSfM). We capture these by defining corresponding energy terms as follows:

Silhouette Consistency. Silhouette consistency simply enforces the predicted shape for an instance to project inside its silhouette. This can be achieved by penalizing the points projected outside the instance mask by their distance from the silhouette (*i.e.* squared distance to the closest silhouette point). In our Δ notation it can be written as follows:

$$E_s(S, O, \pi) = \sum_{C^{mask}(p)>0} \Delta^1(p; O) \tag{2.3}$$

Silhouette Coverage. Using silhouette consistency alone would just drive points projected outside in towards the silhouette. This wouldn't ensure though that the object silhouette is "filled" - i.e. there might be overcarving. We deal with it by having an energy term that encourages points on the silhouette to pull nearby projected points towards them. Formally, this can be expressed as:

$$E_c(S, O, \pi) = \sum_{p \in O} \Delta^m(p; \pi(S)) \tag{2.4}$$

Keypoint Consistency. Our NRSfMalgorithm provides us with sparse 3D keypoints along with camera projection parameters. We use these sparse correspondences on the training set to deform the shape to explain these 3D points. The corresponding energy term penalizes deviation of the shape from the 3D keypoints W for each instance. Specifically, this can be written as:

$$E_{kp}(S, W) = \sum_{\kappa \in W} \Delta^m(\kappa; S) \tag{2.5}$$

Local Consistency. In addition to the above data terms, we use a simple shape regularizer to restrict arbitrary deformations by imposing a quadratic deformation penalty between every point and its neighbors. We also impose a similar penalty on deformations to ensure local smoothness. The δ parameter represents the mean squared displacement between neighboring points and it encourages all faces to have similar size. Here V_{ki} is the i^{th} point in the k^{th} basis.

$$E_l(\bar{S}, V) = \sum_i \sum_{j \in N(i)} ((\|\bar{S}_i - \bar{S}_j\| - \delta)^2 + \sum_k \|V_{ki} - V_{kj}\|^2) \quad (2.6)$$

Normal Smoothness. Shapes occurring in the natural world tend to be locally smooth. We capture this prior on shapes by placing a cost on the variation of normal directions in a local neighborhood in the shape. Our normal smoothness energy is formulated as

$$E_n(S) = \sum_i \sum_{j \in N(i)} (1 - \vec{\mathcal{N}}_i \cdot \vec{\mathcal{N}}_j) \quad (2.7)$$

Here, $\vec{\mathcal{N}}_i$ represents the normal for the i^{th} point in shape S which is computed by fitting planes to local point neighborhoods. Our prior essentially states that local point neighborhoods should be flat. Note that this, in conjunction with our previous energies automatically enforces the commonly used prior that normals should be perpendicular to the viewing direction at the occluding contour [8].

Our total energy is given in equation Eq. 2.8. In addition to the above smoothness priors we also penalize the L_2 norm of the deformation parameters α_i to prevent unnaturally large deformations.

$$E_{tot}(\bar{S}, V, \alpha) = E_l(\bar{S}, V) + \sum_i (E_s^i + E_{kp}^i + E_c^i + E_n^i + \sum_k (\|\alpha_{ik} V_k\|_F^2)) \quad (2.8)$$

Learning. We solve the optimization problem in equation Eq. 2.9 to obtain our shape model $M = (\bar{S}, V)$. The mean shape and deformation basis are inferred via block-coordinate descent on $(\bar{S}, V)$ and α using sub-gradient computations over the training set. We restrict $\|V_k\|_F$ to be a constant to address the scale ambiguity between V and α in our formulation. In order to deal with imperfect segmentations and wrongly estimated keypoints, we use truncated versions of the above energies that reduce the impact of outliers. The mean shapes learnt using our algorithm for 9 rigid categories in PASCAL VOC are shown in Figure 2.5. Note that in addition to representing the coarse shape details of a category, the model also learns finer

structures like chair legs and bicycle handles, which become more prominent with deformations.

$$\begin{aligned} \min_{\bar{S},V,\alpha} \quad & E_{tot}(\bar{S}, V, \alpha) \\ \text{subject to:} \quad & S^i = \bar{S} + \sum_k \alpha_{ik} V_k \end{aligned} \tag{2.9}$$

Our training objective is highly non-convex and non-smooth and is susceptible to initialization. We follow the suggestion of [33] and initialize our mean shape with a soft visual hull computed using all training instances. The deformation bases and deformation weights are initialized randomly.

Implementation Details. The gradients involved in our optimization for shape and projection parameters are extremely efficient to compute. We use approximate nearest neighbors computed using k-d tree to implement the 'Silhouette Coverage', 'Keypoint Consistency' gradients and leverage Chamfer distance fields for obtaining 'Silhouette Consistency' gradients. Our overall computation takes only about 15 min to learn a deformable shape model for an object category with about 500 annotated examples.

2.2 Reconstruction in the Wild

Given an image, our goal is to reconstruct the depicted objects. As the initial step, we use existing state-of-the-art systems [58] to detect and segment the objects present in the image. We then proceed to individually reconstruct each of the detected objects. We approach the problem of reconstructing these objects from the big picture downward - like a sculptor first hammering out the big chunks and then chiseling out the details. We infer their coarse 3D poses and use these along with the predicted instance segmentations to fit our top-down shape models to obtain a coarse top-down shape (Section 2.2.1). Finally, we recover high frequency shape details from shading cues present in the image (Section 2.2.2).

2.2.1 Category Specific Shape Inference

We have at our disposal category-level deformable shape models which can be driven by data-specific and shape-prior based energy terms to infer an object's shape. Recall that the proposed energy terms (Section 2.1.2), in particular 'Silhouette Consistency' ($E_s(S, O, \pi)$) and 'Silhouette Coverage' ($E_c(S, O, \pi)$) depend on a known

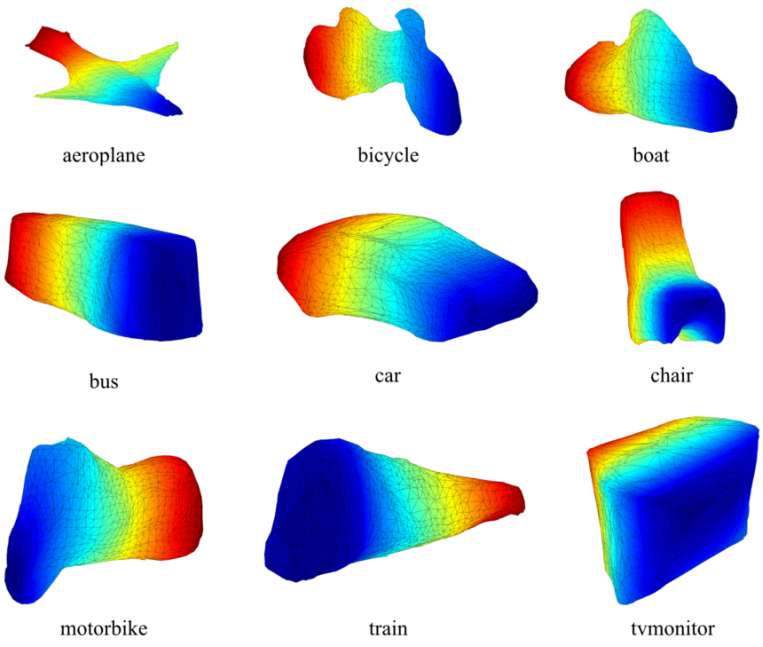

Figure 2.5: Mean shapes learnt for rigid classes in PASCAL VOC obtained using our basis shape formulation. Color encodes depth when viewed frontally.

object silhouette O and camera projection π. We first describe how we estimate O, π and then formulate an optimization problem to infer object shape S.

Initialization. Given an object detection along with its predicted instance segmentation, we use the largest connected component in the predicted segmentation to obtain the object silhouette O. We use an off-the shelf viewpoint prediction system [142] to predict the viewpoint for the detected object, thereby obtaining the camera rotation R. Our learnt models are at a canonical bounding box scale - all objects are first resized to a particular width during training. Given the predicted bounding box, we scale the learnt mean shape accordingly and obtain camera scale c. The translation T is initialized to be the center of the predicted bounding box. These provide us an initial estimate of the camera parameters $\pi_0 \equiv (c, R, T)$.

Formulation. We want to infer a shape that best explains the observed object silhouette, respects generic shape priors (smoothness, continuity) and lies on the linear manifold of category-level shapes. Note that, unlike model learning phase, we do not have access to annotated keypoint locations and thus do not enforce the reconstruction to explain any keypoint locations. These observations are incorporated

by the reconstruction energy defined in (using E_s, E_c, E_n defined in Section 2.1.2).

$$E_r = E_s + E_c + E_n \tag{2.10}$$

In addition to inferring the instance shape, we also observe that the initial camera estimate π_0 is only approximate as the R is predicted upto a dicretization and c, T are initialized coarsely. To alleviate this, we treat the camera parameters π as optimization variables. We further add regularizers to enforce the prior that shape deformation should be small and the the estimated camera should not deviate significantly from the initial camera estimate π_0 . Our final optimization for inferring the object reconstruction is given in Eq. 2.11.

$$\begin{aligned} \min_{\alpha,\pi} \quad & E_r(S,\pi) + \delta(\pi,\pi_0) + \sum_k (\|\alpha_k V_k\|_F^2)) \\ \text{subject to:} \quad & S = \bar{S} + \sum_k \alpha_k V_k \end{aligned} \tag{2.11}$$

Inference. In the above optimization, we first set the optimization variables α, π to $0, \pi_0$ respectively. We then solve the above minimization for the deformation weights α as well as all the camera projection parameters π (scale, translation and rotation) by optimizing Eq. 2.9 using block-coordinate descent (alternately optimizing π and α). The resulting output from the minimization provides us the projection parameters π as well as the inferred 3D shape $S = \bar{S} + \sum_k \alpha_k V_k$. We use the efficient implementations of energy gradients described earlier and consequently, our overall computation takes only about 2 sec to reconstruct a novel instance using a single CPU core.

2.2.2 Bottom-up Shape Refinement

The above optimization results in a top-down 3D reconstruction based on the category-level models, inferred object silhouette, viewpoint and our shape priors. We propose an additional processing step to recover high frequency shape information by adapting the intrinsic images algorithm of Barron and Malik [8, 9], SIRFS, which exploits statistical regularities between shapes, reflectance and illumination

Formally, SIRFS is formulated as the following optimization problem:

$$\underset{Z,L}{\text{minimize}} \quad g(I - S(Z,L)) + f(Z) + h(L)$$

		aero	bike	boat	bus	car	chair	mbike	sofa	train	tv	mean
Mesh	Ours	**1.72**	**1.78**	3.01	**1.90**	1.77	**2.18**	1.88	**2.13**	**2.39**	**3.28**	**2.20**
	Carvi [148]	1.87	1.87	**2.51**	2.36	**1.41**	2.42	**1.82**	2.31	3.10	3.39	2.31
	Puff [145]	3.30	2.52	2.90	3.32	2.82	3.09	2.58	2.53	3.92	3.31	3.03
Depth	Ours	**9.51**	**9.27**	17.20	**12.71**	9.94	**7.78**	9.61	**13.70**	31.58	8.78	**13.01**
	Carvi [148]	10.05	9.28	**15.06**	18.51	**8.14**	7.98	**9.38**	13.71	**31.25**	**8.33**	13.17
	SIRFS [9]	13.52	13.79	20.78	29.93	22.48	18.59	16.80	18.28	40.56	20.18	21.49

Table 2.1: Studying the quality of our learnt 3D models: comparison between our method and [145, 148] using ground truth keypoints and masks on PASCAL VOC.

where $R = I - S(Z, L)$ is a log-reflectance image, Z is a depth map and L is a spherical-harmonic model of illumination. $S(Z, L)$ is a rendering engine which produces a log shading image with the illumination L. g, f and h are the loss functions corresponding to reflectance, shape and illumination respectively.

We incorporate our current coarse estimate of shape into SIRFS through an additional loss term:

$$f_o(Z, Z') = \sum_i ((Z_i - Z'_i)^2 + \epsilon^2)^{\gamma_o}$$

where Z' is the initial coarse shape and ϵ a parameter added to make the loss differentiable everywhere. We obtain Z' for an object by rendering a depth map of our fitted 3D shape model which guides the optimization of this highly non-convex cost function. The outputs from this bottom-up refinement are reflectance, shape and illumination maps of which we retain the shape.

2.3 Experiments

We first examine the quality and expressiveness of our learned 3D models by evaluating how well they matched the underlying 3D shapes of the training data (Section 2.3.1). We then study their sensitivity of obtained reconstructions when fit to images using noisy automatic segmentations and pose predictions (Section 2.3.2) and finally present qualitative results for reconstructions from a single image (Section 2.3.3).

2.3.1 Quality of Learned 3D Models

The first question we address is whether the category-specific shape models we learn for each object class (Section 2.1) using an annotated image collection correctly explain the underlying 3D object shape for these annotated instances. Note that while it is not our final goal, this is itself a very challenging task - we have to obtain a dense 3D reconstruction for annotated images using just silhouettes and sparse keypoint correspondences. Recent work by Vicente *et al.* [148] addressed this task of 'lifting' an annotated image collection to 3D and we compare the performance of our model learning stage against their approach. We also incorporate category-agnostic shape inflation [145] and intrinsic image [8] methods as baselines. The evaluation metrics, dataset and results are described below.

Dataset. We consider images from the challenging PASCAL VOC 2012 dataset [34] which contain objects from the 10 rigid object categories (as listed in Table 2.1). We use the publicly available ground truth class-specific keypoints [16] and object segmentations [57] to learn category-specific shape models for each class. We learn and fit our 3D models on the whole dataset (no train/test split), following the setup of Vicente *et al.* [148].

Since ground truth 3D shapes are unavailable for PASCAL VOC and most other detection datasets, we evaluated the quality of our learned 3D models on the next best thing we managed to obtain: the PASCAL3D+ dataset [158] which has up to 10 3D CAD models for the rigid categories in PASCAL VOC. PASCAL3D+ provides between 4 different models for "tvmonitor" and "train" and 10 for "car" and "chair". The subset of PASCAL we considered after filtering occluded instances, which we do not tackle in this paper, had between 70 images for "sofa" and 500 images for classes "aeroplanes" and "cars".

Metrics. We quantify the quality of our 3D models by comparing against the PASCAL 3D+ models using two metrics - 1) a mesh error metric computed as the Hausdorff distance between the ground truth and predicted mesh after translating both to the origin and normalizing by the diagonal of the tighest 3D bounding box of the ground truth mesh [5] and 2) a depth map error to evaluate the quality of the reconstructed visible object surface, measured as the mean absolute distance between reconstructed and ground truth depth:

$$Z\text{-MAE}(\hat{Z}, Z^*) = \frac{1}{n \cdot \gamma} \min_{\beta} \sum_{x,y} |\hat{Z}_{x,y} - Z^*_{x,y} - \beta| \tag{2.12}$$

where $\hat{Z}$ and Z^* represent predicted and ground truth depth maps respectively. Analytically, β can be computed as the median of $\hat{Z} - Z^*$ and γ is a normalization

		aero	bike	boat	bus	car	chair	mbike	sofa	train	tv	mean
Mesh	KP+Mask	1.77	1.85	3.68	1.90	1.80	2.26	1.83	6.86	2.69	3.40	2.80
	KP+SDS	1.75	1.89	3.71	1.87	1.75	2.27	1.84	6.56	2.76	3.39	2.78
	PP+SDS	1.84	2.02	4.59	1.86	1.88	2.41	2.01	7.30	2.74	3.27	2.99
	Puff [145]	3.31	2.49	2.95	3.40	2.87	3.09	2.65	2.73	3.91	3.33	3.07
Depth	KP+Mask	9.83	9.95	21.07	12.80	10.07	9.10	9.98	29.39	25.70	9.85	14.77
	KP+SDS	9.95	10.35	20.11	13.06	10.49	9.24	10.61	27.94	26.13	10.10	14.80
	PP+SDS	11.42	11.25	21.93	22.04	13.69	10.27	11.71	26.76	34.92	9.88	17.39
	SIRFS [9]	13.58	14.48	19.64	30.14	22.60	20.12	16.81	21.54	41.40	23.67	22.40

Table 2.2: Ablation study for our method assuming/relaxing various annotations at test time on objects in PASCAL VOC. As can be seen, our method degrades gracefully with relaxed annotations. Note that these experiments are in a train/test setting and numbers will differ from Table 2.1. Please see text for more details.

factor to account for absolute object size for which we use the bounding box diagonal. Note that our depth map error is translation and scale invariant.

Results. We report the performance of our model learning approach in Table 2.1. Here, 'SIRFS' denotes a state-of-the art intrinsic image decomposition method and 'Puffball'l [145] denotes a shape-inflation method for reconstruction. 'Carvi' denotes the recent method by Vicente *et al.* [148] which is specifically designed for the task of reconstructing an annotated image collection as their visual hull based reconstruction technique makes strong assumptions regarding the accuracy of the object mask and predicted viewpoint.

We observe that category-agnostic methods – Puffball [145] and SIRFS [8, 9] – consistently perform worse on the benchmark by themselves as they use generic priors to reconstruct each image individually and cannot reason over the image collection jointly. Our model learning performs comparably to the specialized approach of Vicente *et al.*– we demonstrate competitive, if not better, performance on both benchmarks with our models showing greater robustnes to perspective foreshortening effects on "trains" and "buses". Certain classes like "boat" and "sofa" are especially hard because of large intra-class variance and data sparsity respectively.

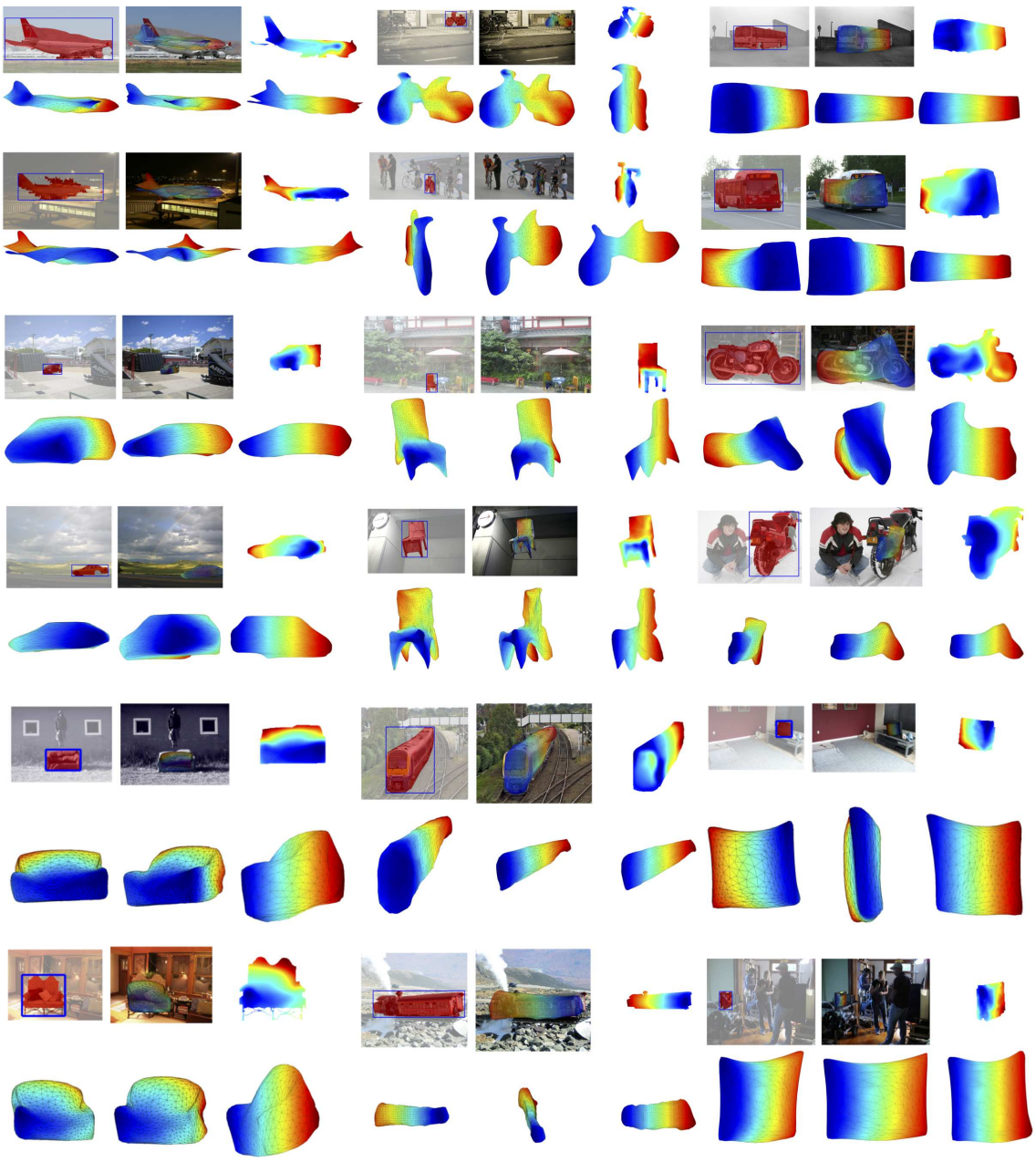

Figure 2.6: Fully automatic reconstructions on detected instances (0.5 IoU with ground truth) using our models on rigid categories in PASCAL VOC. We show our instance segmentation input, the inferred shape overlaid on the image, a 2.5D depth map (after the bottom-up refinement stage), the mesh in the image viewpoint and two other views. It can be seen that our method produces plausible reconstructions which is a remarkable achievement given just a single image and noisy instance segmentations. Color encodes depth in the image coordinate frame (blue is closer). More results can be found at https://goo.gl/MgVQzZ.

2.3.2 Sensitivity Analysis for Recognition based Reconstruction

Our primary goal is to reconstruct objects in an image automatically. Towards this goal, we study the performance of our system when relaxing the availability of various expensive annotations of the form of keypoint correspondences or instance segmentations.

Dataset and Metrics. The reconstruction error metrics for measuring mesh and depth error are the same as described previously (Section 2.3.1). The segmentation, keypoint annotations for learning and the mesh annotations for evaluation are also similarly obtained. However, for the sensitivity analysis, we introduce a train/test split since the recognition components used for instance segmentation and viewpoint estimation are trained on the PASCAL VOC train set. We therefore train our category-shape models on only the subset of the data corresponding to PASCAL VOC train set. We then reconstruct the held out objects in the PASCAL validation set and report performance for these test objects.

Results. In order to analyze sensitivity of our models to noisy inputs we reconstructed held-out test instances using our models given just ground truth bounding boxes. We compare various versions of our method using ground truth(Mask)/imperfect segmentations(SDS) and keypoints(KP)/our pose predictor(PP) for viewpoint estimation respectively. For pose prediction, we use a state-of-the-art CNN-based system [142]. To obtain an approximate segmentation from the bounding box, we use the refinement stage of the state-of-the-art joint detection and segmentation system proposed in [58].

Table 2.2 shows that our results degrade gracefully from the fully annotated to the fully automatic setting. Our method is robust to some mis-segmentation owing to our shape model that prevents shapes from bending unnaturally to explain noisy silhouettes. Our reconstructions degrade slightly with imperfect pose initializations even though our projection parameter optimization deals with it to some extent. With predicted poses, we observe that sometimes even when our reconstructions look plausible, the errors can be high as the metrics are sensitive to bad alignment. The data sparsity issue is especially visible in the case of sofas where in a train/test setting in Table 2.2 the numbers drop significantly with less training data (only 34 instances). Note we do not evaluate our bottom-up component as the PASCAL 3D+ meshes provided do not share the same high frequency shape details as the instance.

2.3.3 Fully Automatic Reconstruction

We qualitatively demonstrate reconstructions on automatically detected and segmented instances with 0.5 IoU overlap with the ground truth in whole images in PASCAL VOC using [58] in Figure 2.6. We can see that our method is able to deal with some degree of mis-segmentation. Some of our major failure modes include not being able to capture the correct scale and pose of the object and thus badly fitting to the silhouette in some cases.

2.4 Discussion

We proposed an approach to perform fully automatic object reconstruction from a single image on a large and realistic dataset. Critically, our deformable 3D shape model can be learned from easily acquired ground-truth 2D annotations, by enforcing consistency between the annotations and the model projections. This allows bypassing the need for a-priori manual mesh design or 3D scanning and making it possible for convenient use of these types of models on large real-world datasets (e.g. PASCAL VOC).

While this ability to learn and infer 3D using such supervision is certainly desirable, leveraging a linear morphable model does restrict expressivity *e.g.* topological changes across shapes are not easily modeled. Further, the reliance on a model fitting approach leads to a computationally expensive inference procedure. We address these challenges in Chapter 3 and Chapter 4, where we show that building upon similar ideas of enforcing geometric consistency, we can learn even more expressive prediction models that also allow efficient inference.

Chapter 3

Multi-view Supervised Single-view Reconstruction

Humans are moving organisms: our ecological supervision [44] comprises of observing the world and the objects in it from different perspectives. These multiple views inform us of the underlying geometry. In addition to allowing us to understand in 3D the particular instance(s) we observe from multiple views, this supervision also enables us to learn about the common structure in the world and reconstruct new objects even from a single view. In Chapter 2, we extracted this common structure in the form of a deformable 3D model, which allowed us to infer 3D shapes for novel instances. In this chapter, we pursue neural network based reconstruction models which are both, more expressive and more efficient for inference. We present an approach that allows learning these single-view prediction models using only the ecologically plausible multi-view supervisory data. As depicted in Figure 3.1, we can learn to infer the 3D shape from a single input image without relying on ground-truth 3D supervision during training.

The insight the multiple views inform us of the underlying geometry has been successfully leveraged by a long line of geometry-based reconstruction techniques. While these structure from motion or multi-view stereo methods work for specific instances, they do not, unlike humans, generalize to predict the 3D shape of a novel instance given a single view. Recent learning-based methods have attempted to address this single-view 3D inference task. However, these approaches rely on full 3D supervision and require known 3D shape for each training image. Not only is this form of supervision ecologically implausible, it is also practically tedious to acquire and difficult to scale. Instead, as depicted in Figure 3.1(b), our goal is to learn 3D

This chapter is based on joint work with Tinghui Zhou, Alexei A. Efros, and Jitendra Malik. The two papers corresponding to this chapter appeared in CVPR, 2017 [144] and CVPR, 2018 [139].

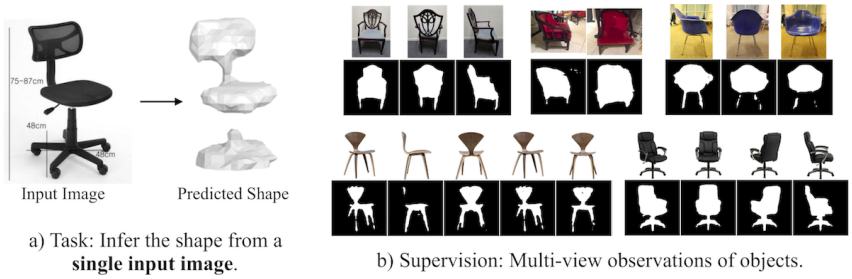

Figure 3.1: We learn to predict 3D shape from a single input view. Our framework can leverage training data of the form of multi-view observations, and learn 3D reconstruction despite the lack of any direct supervision.

prediction using the more naturally plausible multi-view supervision.

We therefore aim to combine aspects of classical multi-view reconstruction with learning based prediction. Akin to the classical geometry-based approaches, we rely on multi-view supervisory signal, while being able to generalize to novel instances and infer their 3D structure from a single view. Our approach is to learn shape prediction by enforcing *geometric consistency* between the predicted 3D and the available multi-view data. Concretely, given one image of an object instance, we predict a corresponding shape, and enforce that this predicted shape is consistent with the multiple views of this instance.

A central aspect of this approach is the notion of geometric consistency between a 3D shape and 2D image. In particular, our learning system requires signals for how to improve predicted shapes such that they become more consistent with the available observations. One way this problem has been traditionally addressed is by space carving [88]. Rays are projected out from pixels into the 3D space and each ray that is known not to intersect the object removes the volume in its path, thereby making the carved-out shape consistent with the observed image.

However, to leverage it in a learning-based system, we want to extend this notion of consistency to a differential setting. That is, instead of deleting chunks of volume all at once, we would like to compute incremental changes to the 3D shape that make it more consistent with the 2D image. In this chapter, we present a differentiable ray consistency formulation that allows computing the gradient of a predicted 3D shape of an object, given an observation (depth image, foreground mask, color image *etc.*.) from an arbitrary view. The differentiability of our consistency formulation is what allows its use in a learning framework, such as a neural network. Every new piece of evidence gives gradients for the predicted shape, which, in turn, yields incremental updates for the underlying prediction model. Since this prediction model is shared across object instances, it is able to find and learn from the commonalities across

different 3D shapes, requiring only sparse per-instance supervision.

We first describe in Section 3.2 the formulation of our geometric consistency loss, and then present our approach to leverage it for learning single-view reconstruction via multi-view supervision in Section 3.3. In Section 3.4 we demonstrate the applicability of our framework to learn 3D inference across various scenarios.

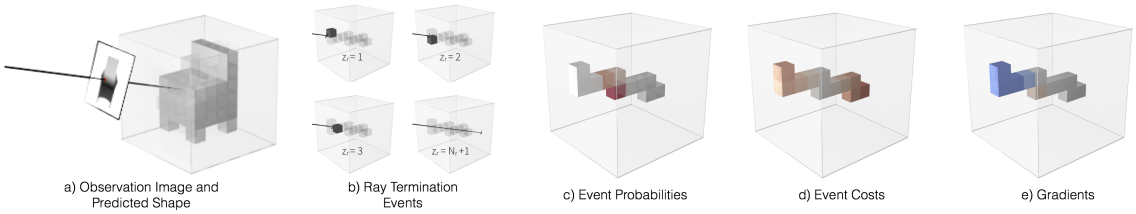

Figure 3.2: Visualization of various aspects of our Differentiable Ray Consistency formulation. a) Predicted 3D shape represented as probabilistic occupancies and the observation image where we consider consistency between the predicted shape and the ray corresponding to the highlighted pixel. b) Ray termination events (Section 3.2.2) – the random variable $z_r = i$ corresponds to the event where the ray terminates at the i^{th} voxel on its path, $z_r = N_r + 1$ represents the scenario where the ray escapes the grid. c) Depiction of event probabilities (Section 3.2.2) where red indicates a high probability of the ray terminating at the corresponding voxel. d) Given the ray observation, we define event costs (Section 3.2.3). In the example shown, the costs are low (white color) for events where ray terminates in voxels near the observed termination point and high (red color) otherwise. e) The ray consistency loss (Section 3.2.4) is defined as the expected event cost and our formulation allows us to obtain gradients for occupancies (red indicates that loss decreases if occupancy value increases, blue indicates the opposite). While in this example we consider a depth observation, our formulation allows incorporating diverse kinds of observations by defining the corresponding event cost function as discussed in Section 3.2.3 and Section 3.2.5. Best viewed in color.

3.1 Background

Object Reconstruction from Image-based Annotations. Blanz and Vetter [14] demonstrated the use of a morphable model to capture 3D shapes. Cashman and Fitzgibbon [19] learned these models for complex categories like dolphins using object silhouettes and keypoint annotations for training and inference. Wu *et al.* [156], using similar annotations, learned a system to predict sparse 3D by inferring parameters of a shape skeleton. However, since the use of such low-dimensional models restricts expressivity, Vicente *et al.* [148] proposed a non-parametric method by leveraging

surrogate instances – but at the cost of requiring annotations at test time. We leverage similar training data but using a CNN-based voxel prediction framework allows test time inference without manual annotations and allows handling large shape variations.

Object Reconstruction from 3D Supervision. The advent of deep learning along with availability of large-scale synthetic training data has resulted in applications for object reconstruction. Choy *et al.* [24] learned a CNN to predict a voxel representation using a single (or multiple) input image(s). Girdhar *et al.* [45] also presented similar results for single-view object reconstruction, while also demonstrating some results on real images by using realistic rendering techniques [133] for generating training data. Several approaches have further improved these voulmetric predictions [77, 155, 170], or pursued alternate 3D representations such as point clouds [36], octrees [56, 135], or meshes [75, 87, 89]. A crucial assumption in the procedure of training these models, however, is that full 3D supervision is available. As a result, these methods primarily train using synthetically rendered data where the underlying 3D shape is available.

While the progress demonstrated by these methods is encouraging and supports the claim for using CNN based learning techniques for reconstruction, the requirement of explicit 3D supervision for training is potentially restrictive. We relax this assumption and show that alternate sources of supervision can be leveraged. It allows us to go beyond reconstructing objects in a synthetic setting, to extend to real datasets which do not have 3D supervision.

Multi-view Instance Reconstruction. Perhaps most closely related to our work in terms of the proposed formulation is the line of work in geometry-based techniques for reconstructing a single instance given multiple views. Visual hull [90] formalizes the notion of consistency between a 3D shape and observed object masks. Techniques based on this concept [18, 101] can obtain reconstructions of objects by space carving using multiple available views. It is also possible, by jointly modeling appearance and occupancy, to recover 3D structure of objects/scenes from multiple images via ray-potential based optimization [28, 95] or inference in a generative model [43, 154]. Ulusoy *et al.* [146] propose a probabilistic framework where marginal distributions can be efficiently computed. More detailed reconstructions can be obtained by incorporating additional signals *e.g.* depth or semantics [86, 120, 121].

The main goal in these prior works is to reconstruct a specific scene/object from multiple observations and they typically infer a discrete assignment of variables such that it is maximally consistent with the available views. Our insight is that similar cost functions which measure consistency, adapted to treat variables as continuous probabilities, can be used in a learning framework to obtain gradients for the current

prediction. Crucially, the multi-view reconstruction approaches typically solve a (large) optimization to reconstruct a particular scene/object instance and require a large number of views. In contrast, we only need to perform a single gradient computation to obtain a learning signal for the CNN and can even work with sparse set of views (possibly even just one view) per instance.

Multi-view Supervision for Single-view Depth Prediction. While single-view depth prediction had been dominated by approaches with direct supervision [30], recent approaches based on multi-view supervision have shown promise in achieving similar (and sometimes even better) performance. Garg *et al.* [41] and Godard *et al.* [48] used stereo images to learn a single image depth prediction system by minimizing the inconsistency as measured by pixel-wise reprojection error. Zhou *et al.* [168] further relax the constraint of having calibrated stereo images, and learn a single-view depth model from monocular videos. The motivation of these multi-view supervised depth prediction approaches is similar to ours, but we aim for 3D instead of 2.5D predictions and address the related technical challenges in this work.

3.2 Formulation

In this section, we formulate a differentiable 'view consistency' loss function which measures the inconsistency between a (predicted) 3D shape and a corresponding observation image with an associated known (or predicted) camera viewpoint. We first formally define our problem setup by instantiating the representation of the 3D shape and the observation image with which the consistency is measured.

Shape Representation. Our 3D shape representation is parametrized as occupancy probabilities of cells in a discretized 3D voxel grid, denoted by the variable x. We use the convention that x_i represents the probability of the i^{th} voxel being empty (we use the term 'occupancy probability' for simplicity even though it is a misnomer as the variable x is actually 'emptiness probability'). Note that the choice of discretization of the 3D space into voxels need not be a uniform grid – the only assumption we make is that it is possible to trace rays across the voxel grid and compute intersections with cell boundaries.

Observation and Camera. We aim for the shape to be consistent with some available observation O from a camera C. This 'observation' can take various forms *e.g.* a depth image, an object foreground mask, a color image *etc.*– these are treated similarly in our framework. Concretely, we have a observation-camera pair (O, C) where the 'observation' O is from a view defined by (known or predicted) camera C. The camera C is defined via an intrinsic matrix and the extrinsics specifying its

rotation and translation in the world coordinate frame.

Our view consistency loss, using the notations mentioned above, is of the form $L(x; (O, C))$. Towards defining this loss, in Section 3.2.1 we reduce the notion of consistency between the 3D shape and an observation image to consistency between the 3D shape and a ray with associated observations. We then present a differentiable formulation for ray consistency, the various aspects of which are visualized in Figure 3.2. In Section 3.2.2, we examine the case of a ray travelling though a probabilistically occupied grid and in Section 3.2.3, we instantiate costs for each probabilistic ray-termination event. We then combine these to define the consistency cost function in Section 3.2.4. While we initially only consider the case of the shape being represented by voxel occupancies x, we show in Section 3.2.5 that it can be extended to incorporate optional per-voxel predictions p. This generalization allows us to incorporate other kinds of observation *e.g.* color images, pixel-wise semantics *etc.*. The generalized consistency loss function is then of the form $L(x, [p]; (O, C))$ where $[p]$ denotes an optional argument. The view consistency loss formulation we present, while differentiable w.r.t the shape x, is not differentiable w.r.t the camera C. In Section 3.2.6 we present an alternative formulation of this loss that, using a simple re-parametrization, is also differentiable w.r.t C.

3.2.1 View Consistency as Ray Consistency

Every pixel in the observation image O corresponds to a ray with a recorded observation (depth/color/foreground label/semantic label). Assuming known camera intrinsic parameters (f_u, f_v, u_0, v_0), the image pixel (u, v) corresponds to a ray r originating from the camera centre travelling in direction $(\frac{u-u_0}{f_u}, \frac{v-v_0}{f_v}, 1)$ in the camera coordinate frame. Given the camera extrinsics, the origin and direction of the ray r can also be inferred in the world frame.

Therefore, the available observation-camera pair (O, C) is equivalently a collection of arbitrary rays $\mathcal{R}$ where each $r \in \mathcal{R}$ has a known origin point, direction and an associated observation o_r *e.g.* depth images indicate the distance travelled before hitting a surface, foreground masks inform whether the ray hit the object, semantic labels correspond to observing category of the object the ray terminates in.

Analogous to the common practice in classical multi-view reconstruction approaches [95, 120, 121, 146] which formulate objectives using a set of ray potentials, we can similarly formulate the view consistency loss $L(x; (O, C))$ using per-ray based consistency terms $L_r(x)$. Here, $L_r(x)$ captures if the inferred 3D model x correctly explains the observations associated with the specific ray r. Our view consistency

loss is then just the sum of the consistency terms across the rays:

$$L(x;(O,C)) \equiv \sum_{r\in\mathcal{R}} L_r(x) \tag{3.1}$$

Our task for formulating the view consistency loss is simplified to defining a differentiable ray consistency loss $L_r(x)$.

3.2.2 Ray-tracing in a Probabilistic Occupancy Grid

With the goal of defining the consistency cost $L_r(x)$, we examine the ray r as it travels across the voxel grid with occupancy probabilities x. The occupancy probabilities in this grid (instantiated by the shape parameters x) induce a distribution over possible terminations for a ray which can be efficiently computed [18, 154]. We denote the various likely ray terminations as *events* that can occur to ray r, and we can define $L_r(x)$ by seeing the incompatibility of these events with available observations o_r.

Ray Termination Events. Since we know the origin and direction for the ray r, we can trace it through the voxel grid - let us assume it passes though N_r voxels. The *events* associated with this ray correspond to it either terminating at one of these N_r voxels or passing through. We use a random variable z_r to correspond to the voxel in which the ray (probabilistically) terminates - with $z_r = N_r + 1$ to represent the case where the ray does not terminate. These events are shown in Figure 3.2.

Event Probabilities. Given the occupancy probabilities x, we want to infer the probability $q(z_r = i)$. The event $z_r = i$ occurs iff the previous voxels in the path are all unoccupied and the i^{th} voxel is occupied. Assuming an independent distribution of occupancies where the prediction x_i^r correspnds to the probability of the i^{th} voxel on the path of the ray r as being *empty*, we can compute the probability distribution for z_r.

$$q(z_r = i) = \begin{cases} (1 - x_i^r) \prod_{j=1}^{i-1} x_j^r, & \text{if } i \leq N_r \\ \prod_{j=1}^{N_r} x_j^r, & \text{if } i = N_r + 1 \end{cases} \tag{3.2}$$

3.2.3 Event Cost Functions

Note that each event $(z_r = i)$, induces a prediction *e.g.* if $z_r = i$, we can geometrically compute the distance d_i^r the ray travels before terminating. We can

define a cost function between the induced prediction under the event $(z_r = i)$ and the available associated observations for ray o_r. We denote this cost function as $\psi_r(i)$ and it assigns a cost to event $(z_r = i)$ based on whether it induces predictions inconsistent with o_r. We now show some examples of event cost functions that can incorporate diverse observations o_r and used in various scenarios.

Object Reconstruction from Depth Observations. In this scenario, the available observation o_r corresponds to the observed distance the ray travels d_r. We use a simple distance measure between observed distance and event-induced distance to define $\psi_r(i)$.

$$\psi_r^{depth}(i) = |d_i^r - d_r| \tag{3.3}$$

Object Reconstruction from Foreground Masks. We examine the case where we only know the object masks from various views. In this scenario, let $s_r \in \{0, 1\}$ denote the known information regarding each ray - $s_r = 0$ implies the ray r intersects the object *i.e.* corresponds to an image pixel within the mask, $s_r = 1$ indicates otherwise. We can capture this by defining the corresponding cost terms.

$$\psi_r^{mask}(i) = \begin{cases} s_r, & \text{if } i \leq N_r \\ 1 - s_r, & \text{if } i = N_r + 1 \end{cases} \tag{3.4}$$

We note that some concurrent approaches [111, 160] have also been proposed to specifically address the case of learning object reconstruction from foreground masks. These approaches, either though a learned [111] or fixed [160] reprojection function, minimize the discrepancy between the observed mask and the reprojected predictions. Our ray consistency based approach effectively minimizes a similar loss using a geometrically derived re-projection function, while also allowing us to handle more general observations.

3.2.4 Ray-Consistency Loss

We have examined the case of a ray traversing through the probabilistically occupied voxel grid and defined possible ray-termination events occurring with probability distribution specified by $q(z_r)$. For each of these events, we incur a corresponding cost $\psi_r(i)$ which penalizes inconsistency between the event-induced predictions and available observations o_r. The per-ray consistency loss function $L_r(x)$ is simply the expected cost incurred.

$$L_r(x) = \mathbb{E}_{z_r}[\psi_r(z_r)] \tag{3.5}$$

$$L_r(x) = \sum_{i=1}^{N_r+1} \psi_r(i)\ q(z_r = i) \tag{3.6}$$

Recall that the event probabilities $q(z_r = i)$ were defined in terms of the voxel occupancies x predicted by the CNN (Eq. 3.2). Using this, we can compute the derivatives of the loss function $L_r(x)$ w.r.t the CNN predictions.

$$\frac{\partial\ L_r(x)}{\partial\ x_k^r} = \sum_{i=k}^{N_r} (\psi_r(i+1) - \psi_r(i)) \prod_{1 \le j \le i, j \ne k} x_j^r \tag{3.7}$$

The ray-consistency loss $L_r(x)$ completes our formulation of view consistency loss as the overall loss is defined in terms of $L_r(x)$ as in Eq. 3.1. The gradients derived from the view consistency loss simply try to adjust the voxel occupancy predictions x, such that events which are inconsistent with the observations occur with lower probabilities.

3.2.5 Incorporating Additional Labels

We have developed a view consistency formulation for the setting where the shape representation is described as occupancy probabilities x. In the scenario where alternate per-pixel observations (*e.g.* semantics or color) are available, we can modify consistency formulation to account for per-voxel predictions p in the 3D representation. In this scenario, the observation o_r associated with the ray r includes the corresponding pixel label and similarly, the induced prediction under event $(z_r = i)$ includes the auxiliary prediction for the i^{th} voxel on the ray's path – p_i^r.

Inspired by Savinov *et al.* [120,121] who address a similar challenge for multi-view reconstruction, we incorporate consistency between these by extending $L_r(x)$ to $L_r(x, [p])$ by using a generalized event-cost term $\psi_r(i, [p_i^r])$ in Eq. 3.5 and Eq. 3.6. Examples of the generalized cost term for two scenarios are presented in Eq. 3.9 and Eq. 3.10. The gradients for occupancy predictions x_i^r are as previously defined in Eq. 3.7, but using the generalized cost term $\psi_r(i, [p_i^r])$ instead. The additional per-voxel predictions can also be trained using the derivatives below.

$$\frac{\partial\ L_r(x, [p])}{\partial\ p_r^i} = q(z_r = i) \frac{\partial\ \psi_r(i, [p_r^i])}{\partial\ p_r^i} \tag{3.8}$$

Note that we can define any event cost function $\psi(i, [p_i^r])$ as long as it is differentiable w.r.t p_i^r. We can interpret Eq. 3.8 as the additional per-voxel predictions p being updated to match the observed pixel-wise labels, with the gradient being weighted by the probability of the corresponding event.

Scene Reconstruction from Depth and Semantics. In this setting, the observations associated with each ray correspond to an observed depth d_r as well as semantic class labels c_r. The event-induced prediction, if $z_r = i$, corresponds to depth d_i^r and class distribution p_i^r and we can define an event cost penalizing the discrepancy in disparity (since absolute depth can have a large variation) and the negative log likelihood of the observed class.

$$\psi_r^{sem}(i, p_i^r) = |\frac{1}{d_i^r} - \frac{1}{d_r}| - log(p_i^r(c_r)) \tag{3.9}$$

Object Reconstruction from Color Images. In this scenario, the observations c_r associated with each ray corresponds to the RGB color values for the corresponding pixel. Assuming additional per voxel color prediction p, the event-induced prediction, if $z_r = i$, yields the color at the corresponding voxel *i.e.* p_i^r. We can define an event cost penalizing the squared error.

$$\psi_r^{color}(i, p_i^r) = \frac{1}{2}\|p_i^r - c_r\|^2 \tag{3.10}$$

In addition to defining the event cost functions, we also need to instantiate the induced observations for the event of ray escaping. We define $d_{N_r+1}^r$ in Eq. 3.3 and Eq. 3.9 to be a fixed large value, and $p_{N_r+1}^r$ in Eq. 3.9 and Eq. 3.10 to be uniform distribution and white color respectively.

3.2.6 Pose-Differentiable Ray Consistency

The loss formulation presented above is differentiable w.r.t the shape x, but not w.r.t the camera parameters C. This is because $\{x_i^r\}$, which represents the occupancy probability of the i^{th} voxel in the ray's path, is not a differentiable function of the camera (since the ordering of voxels on a ray's path is a discrete function). However, in certain scenarios *e.g.* when leveraging predicted camera parameters instead of known ones, it would be necessary to have a formulation where the loss is differentiable w.r.t both, shape and pose.

In order to overcome this, we use an alternate pose-differentiable ray consistency loss formulation, with the corresponding view loss denoted as $\tilde{L}(x; (O, C))$. We do so by redefining the variable $\{x_i^r\}$ to correspond to the occupancy at the i^{th} sample

along the ray. Therefore, instead of using probabilities of voxels along a ray, we consider probabilities at point samples along a ray. Concretely, we sample points at a fixed set of $N_r = 80$ depth values $\{d_i | 1 \leq i \leq N\}$ along each ray.

To determine x_i^r, we look at the 3D coordinate of the corresponding point (determined using camera parameters), and trilinearly sample the shape x to determine the occupancy at this point.

$$l_i \equiv (\frac{u - u_0}{f_u} d_i, \frac{v - v_0}{f_v} d_i, d_i) \tag{3.11}$$

$$x_i^r = \mathcal{T}(x, R \times (l_i + t)\ ) \tag{3.12}$$

As the trilinear sampling function $\mathcal{T}$ is differentiable w.r.t its arguments, the sampled occupancy x_i^r, and consequently the alternate view consistency loss $\tilde{L}(x; (O, C))$, is differentiable w.r.t the shape x and the camera C.

We note that although this idea of using samples instead of voxels (similar to [160]) is less physically grounded, it provides us a convenient tool to obtain gradients for the predicted cameras. While we primarily use the original loss formulation for most of our experiments, we leverage this pose-differentiable loss in some scenarios where the associated cameras for observations are also predicted.

3.3 Learning Single-view Reconstruction

We aim to learn a function f – modeled as a parameterized CNN f_θ, which given a single image I corresponding to a novel object, predicts its shape as a voxel occupancy grid. A straightforward learning-based approach would require a training dataset $\{(I_i, \bar{x}_i)\}$ where the target voxel representation $\bar{x}_i$ is known for each training image I_i. However, we are interested in a scenario where the ground-truth 3D models $\{\bar{x}_i\}$ are not available for training f_θ directly, as is often the case for real-world objects/scenes. While collecting the ground-truth 3D is not feasible, it is relatively easy to obtain 2D or 2.5D observations (e.g. depth maps) of the underlying 3D model from other viewpoints. In this scenario we can leverage the 'view consistency' loss function described in Section 3.2 to train f_θ.

We consider two supervision scenarios for learning f_θ. We first examine the setting where the available multi-view observations have known associated camera poses (*e.g.* as possible for a moving agent that knows its egomotion), and then address the scenario where even the camera poses associated are unknown.

3.3.1 Learning with Pose Supervision

Training Data. As our training data, corresponding to each training (RGB) image I_i in the training set, we also have access to one or more additional observations of the same instance from other views. The observations, as described in Section 3.2, can be of varying forms. Concretely, corresponding to image I_i, we have *one or more* observation-camera pairs $\{O_k^i, C_k^i\}$ where the 'observation' O_k^i is from a view defined by camera C_k^i. Note that these observations are required only for training; at test time, the learned CNN f_θ predicts a 3D shape from only a single 2D image.

Predicted 3D Representation. The output of our single-view 3D prediction CNN is $f_\theta(I) \equiv (x, [p])$ where x denotes voxel occupancy probabilities and $[p]$ indicates optional per-voxel predictions (used if corresponding training observations *e.g.* color, semantics are leveraged).

To learn the parameters θ of the single-view 3D prediction CNN, for each training image I_i we train the CNN to minimize the inconsistency between the prediction $f_\theta(I_i)$ and the one or more observation(s) $\{(O_k^i, C_k^i)\}$ corresponding to I_i. This optimization is the same as minimizing the (differentiable) loss function $\sum_i \sum_k L(f_\theta(I_i); (O_k^i, C_k^i))$ *i.e.* the sum of view consistency losses (Eq. 3.1) for observations across the training set. To allow for faster training, instead of using all rays as defined in Eq. 3.1, we randomly sample a few rays (about 1000) per view every SGD iteration.

3.3.2 Learning without Pose Supervision

In this supervision scenario, we do not assume known camera poses associated with the multiple views. Instead, we assume that we have an RGB image I_k^i associated with each observation image, and leverage a predicted camera to enforce geometric consistency. To operationalize this setup, in addition to learning the single-view 3D prediction CNN f_θ, we also jointly learn a pose prediction CNN g_ϕ. We first describe the training data and representations predicted, and then summarize the learning process.

Training Data. Similar to the setup in Section 3.3.1, we rely on multi-view training data, but without camera pose annotations. Corresponding to image I_i, we have observation-image pairs $\{O_k^i, I_k^i\}$ where the 'observation' O_k^i is associated with an RGB image I_k^i (which we use to predict pose).

Predictions. The output of the shape prediction CNN f_θ is a voxel occupancy grid as in Section 3.3.1. The pose prediction CNN g_ϕ predicts, from a single input image, the corresponding camera extrinsic parameters: a quaternion to instantiate the rotation, and a translation $\in \mathbb{R}^3$. We assume known camera intrinsics (although

these can also be predicted), and therefore the predictions of g_ϕ suffice to instantiate the associated camera.

To jointly learn the shape and pose prediction CNNs f_θ and g_ϕ, we train these CNNs to minimize the inconsistency between a predicted shape prediction $f_\theta(I_i)$ and available observations with their corresponding predicted cameras $\{(O_k^i, g_\phi(I_k^i)\}$. As we also want the consistency loss to be differentiable w.r.t the predicted cameras, we use the formulation defined in Section 3.2.6, and minimize the loss function $\sum_i \sum_k \tilde{L}(f_\theta(I_i); (O_k^i, g_\phi(I_k^i)))$.

We empirically observe that training both f_θ and g_ϕ jointly from scratch, and without any direct supervision, is challenging. The optimization often gets stuck at a local minima for the camera pose prediction and only predicts a restricted range of poses *e.g.* conflating front and back facing chairs. To overcome this, we incorporate a pose prior as well as allow g_ϕ to predict a distribution of pose hypotheses instead of a single one.

3.4 Experiments

We consider various scenarios where we can learn single-view reconstruction using our differentiable ray consistency (DRC) formulation. First, we examine the ShapeNet dataset where we use synthetically generated images and corresponding multi-view observations to study our framework. We then demonstrate applications on the PASCAL VOC dataset where we train a single-view 3D prediction system using only one observation per training instance. We then explore the application of our framework for scene reconstruction using short driving sequences as supervision. We also show qualitative results for using multiple color image observations as supervision for single-view reconstruction. While these scenarios assume known camera pose for training, we finally examine two settings where we demonstrate that we can learn 3D prediction even without pose supervision.

3.4.1 Empirical Analysis on ShapeNet

We study the framework presented and demonstrate its applicability with different types of multi-view observations and also analyze the susceptibility to noise in the learning signal. We perform experiments in a controlled setting using synthetically rendered data where the ground-truth 3D information is available for benchmarking.

Setup. The ShapeNet dataset [20] has a collection of textured CAD models and we examine 3 representative categories with large sets of available models : airplanes,

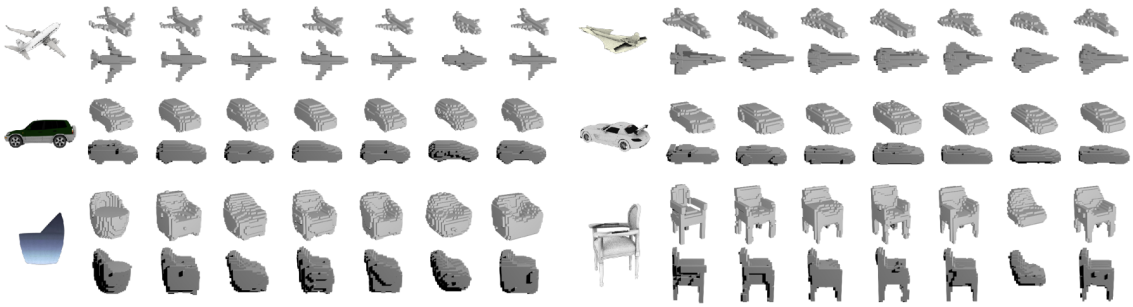

Figure 3.3: Reconstructions on the ShapeNet dataset visualized using two representative views. Left to Right : Input, Ground-truth, 3D Training, Ours (Mask), Fusion (Depth), DRC (Depth), Fusion (Noisy Depth), DRC (Noisy Depth).

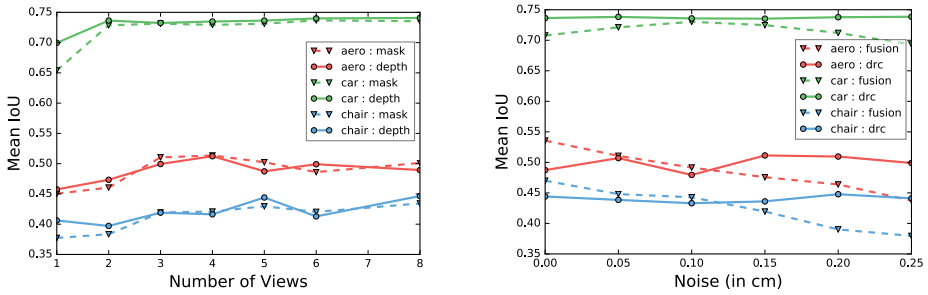

Figure 3.4: Analysis of the per-category reconstruction performance. a) As we increase the number of views available per instance for training, the performance initially increases and saturates after few available views. b) As the amount of noise in depth observations used for training increases, the performance of our approach remains relatively consistent.

cars, and chairs . We create random train/val/test splits and use rendered images with randomly sampled views as input to the single-view 3D prediction CNNs.

Our CNN model is a simple encoder-decoder which predicts occupancies in a voxel grid from the input RGB image. To perform control experiments, we vary the sources of information available (and correspondingly, different loss functions) for training the CNN. The various control settings are briefly described below:

Ground-truth 3D. We assume that the ground-truth 3D model is available and use a simple cross-entropy loss for training. This provides an upper bound for the performance of a multi-view consistency method.

DRC (Mask/Depth). In this scenario, we assume that (possibly noisy) depth images (or object masks) from 5 random views are available for each training CAD model and minimize the view consistency loss.

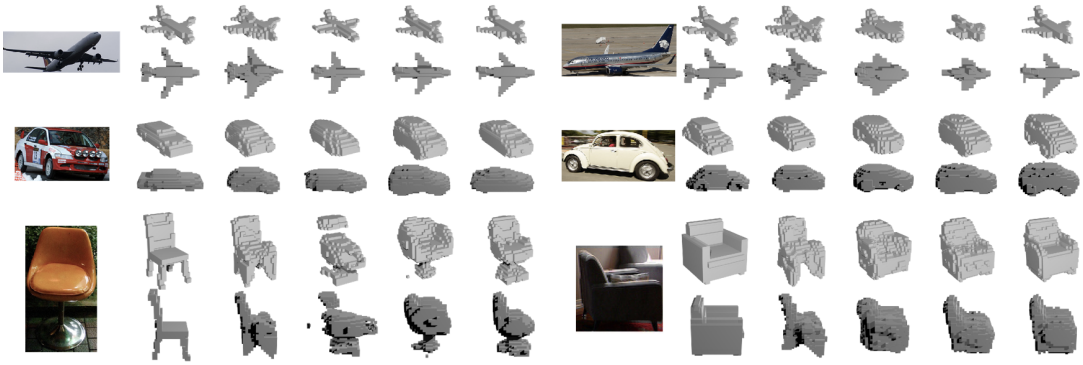

Figure 3.5: PASCAL VOC reconstructions visualized using two representative views. Left to Right : Input, Ground-truth (as annotated in PASCAL 3D), Deformable Models [141], DRC (Pascal), Shapenet 3D, DRC (Joint).

Training Data	3D	Mask		Depth		Depth (Noisy)	
class		Fusion	DRC	Fusion	DRC	Fusion	DRC
aero	0.57	-	0.50	0.54	0.49	0.46	0.51
car	0.76	-	0.73	0.71	0.74	0.71	0.74
chair	0.47	-	0.43	0.47	0.44	0.39	0.45

Table 3.1: Analysis of our method using mean IoU on ShapeNet.

Depth Fusion. As an alternate way of using multi-view information, we preprocess the 5 available depth images per CAD model to compute a pseudo-ground-truth 3D model. We then train the CNN with a cross-entropy loss, restricted to voxels where the views provided any information. Note that unlike our method, this is applicable only if depth images are available and is more susceptible to noise in observations.

Evaluation Metric. We use the mean intersection over union (IoU) between the ground-truth 3D occupancies and the predicted 3D occupancies. Since different losses lead to the learned models being calibrated differently, we report mean IoU at the optimal discretization threshold for each method (the threshold is searched at a category level).

Results. We present the results of the experiments in Table 3.1 and visualize sample predictions in Figure 3.3. In general, the qualitative and quantitative results in our setting of using only a small set of multi-view observations are encouragingly close to the upper bound of using ground-truth 3D as supervision. While our approach and the alternative way of depth fusion are comparable in the case of perfect depth

information, our approach is much more robust to noisy training signal. This is because of the use of a ray potential where the noisy signal only adds a small penalty to the true shape unlike in the case of depth fusion where the noisy signal is used to compute independent unary terms. We observe that even using only object masks leads to comparable performance to using depth but is worse when fewer views are available (Figure 3.4) and has some systematic errors *e.g.* the chair models cannot learn the concavities present in the seat using foreground mask information.

Ablations. When using muti-view supervision, it is informative to look at the change in performance as the number of available training views is increased. We show this result in Figure 3.4 and observe a performance gain as number of views initially increase but see the performance saturate after few views. We also note that depth observations are more informative than masks when very small number of views are used. Another aspect studied is the reconstruction performance when varying the amount of noise in depth observations. We observe that our approach is fairly robust to noise unlike the fusion approach.

3.4.2 Object Reconstruction on PASCAL VOC

We demonstrate the application of our DRC formulation on the PASCAL VOC dataset [34] where previous 3D supervised single-view reconstruction methods cannot be used due to lack of ground-truth training data. However, available annotations for segmentation masks and camera pose allow application of our framework.

Training Data. We use annotated pose (in PASCAL 3D [158]) and segmentation masks (from PASCAL VOC) as training signal for object reconstruction. To augment training data, we also use the Imagenet [114] objects from PASCAL 3D (using an off-the shelf instance segmentation method [92] to compute foreground masks on these). These annotations effectively provide an orthographic camera C_i for each training instance. Additionally, the annotated segmentation mask provides us with the observation O_i. We use the proposed view consistency loss on objects from the training set in PASCAL3D – the loss measures consistency of the predicted 3D shape given training RGB image I_i with the single observation-camera pair (O_i, C_i). Despite only one observation per instance, the shared prediction model can learn to predict complete 3D shapes.

Benchmark. PASCAL3D also provides annotations for (approximate) 3D shape of objects using a small set of CAD models (about 10 per category). Similar to previous approaches [24, 141], we use these annotations on the test set for benchmarking purposes. Note that since the same small set of models is shared across training and test objects, using the PASCAL3D models for training is likely to bias the

Method	aero	car	chair	mean
CSDM	0.40	0.60	0.29	0.43
DRC (PASCAL)	0.42	0.67	0.25	0.44
Shapenet 3D	0.53	0.67	0.33	0.51
DRC (Joint)	**0.55**	**0.72**	**0.34**	**0.54**

Table 3.2: Mean IoU on PASCAL VOC.

evaluation. This makes our results incomparable to those reported in [24] where a model pretrained on ShapeNet data is fine-tuned on PASCAL3D using shapes from this small set of models as ground-truth.

Setup. The various baselines/variants studied are described below. Note that for all the learning based methods, we train a single category-agnostic CNN.

Category-Specific Deformable Models (CSDM). We compare to [141] in a setting where, unlike other methods, it uses ground-truth mask, keypoints to fit deformable 3D models.

ShapeNet 3D (with Realistic Rendering). To emulate the setup used by previous approaches *e.g.* [24, 45], we train a CNN on rendered ShapeNet images using cross entropy loss with the ground-truth CAD model. We attempt to bridge the domain gap by using more realistic renderings via random background/lighting variations [133] and initializing the convolution layers with a pretrained ResNet-18 model [61].

DRC (Pascal). We only use the PASCAL3D instances with pose, object mask annotations to train the CNN with the proposed view consistency loss.

DRC (Joint : ShapeNet 3D + Pascal). We pre-train a model on ShapeNet 3D data as above and finetune it using PASCAL3D using our view consistency loss.

Results. We present the comparisons of our approach to the baselines in Table 3.2 and visualize sample predictions in Figure 3.5. We observe that our model when trained using only PASCAL3D data, while being category agnostic and not using ground-truth annotations for testing, performs comparably to [141] which also uses similar training data. We observe that using the PASCAL data via the view consistency loss in addition to the ShapeNet 3D training data allows us to improve across categories as using real images for training removes some error modes that the CNN trained on synthetic data exhibits on real images. Note that the learning signals used in this setup were only approximate – the annotated pose, segmentation masks computed by [92] are not perfect and our method results in improvements

despite these.

3.4.3 3D Scene Reconstruction from Ego-motion

The problem of scene reconstruction is an extremely challenging one. While previous approaches, using direct [30], multi-view [41, 48] or even no supervision [40] predict detailed 2.5D representations (pixelwise depth and/or surface normals), the task of single image 3D prediction has been largely unexplored for scenes. A prominent reason for this is the lack of supervisory data. Even though obtaining full 3D supervision might be difficult, obtaining multi-view observations may be more feasible. We present some preliminary explorations and apply our framework to learn single image 3D reconstruction for scenes by using driving sequences as supervision.

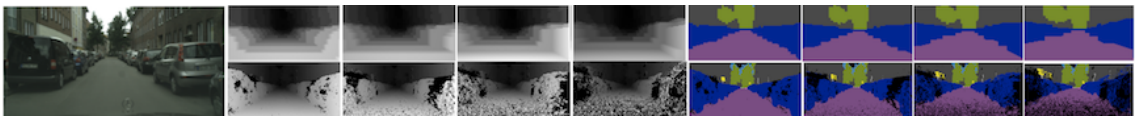

Figure 3.6: Sample results on Cityscapes using ego-motion sequences for learning single image 3D reconstruction. Given a single input image (left), our model predicts voxel occupancy probabilities and per-voxel semantic class distribution. We use this prediction to render, in the top row, estimated disparity and semantics for a camera moving forward by 3, 6, 9, 12 metres respectively. The bottom row renders similar output but using a 2.5D representation of ground-truth pixel-wise disparity and pixel-wise semantic labels inferred by [162].

We use the cityscapes dataset [26] which has numerous 30-frame driving sequences with associated disparity images, ego-motion information and semantic labels[1]. We train a CNN to predict, from a single scene image, occupancies and per-voxel semantic labels for a coarse voxel grid. We minimize the consistency loss function corresponding to the event cost in Eq. 3.9. To account for the large scale of scenes, our voxel grid does not have uniform cells, instead the size of the cells grows as we move away from the camera.

We show qualitative results in Figure 3.6 and compare the coarse 3D representation inferred by our method with a detailed 2.5D representation by rendering inferred disparity and semantic segmentation images under simulated forward motion. The 3D representation, while coarse, is able to capture structure not visible in the original image (*e.g.* cars occluding other cars). While this is an encouraging result that demonstrates the possibility of going beyond 2.5D for scenes, there are several

[1] while only sparse frames are annotated, we use a semantic segmentation system [162] trained on these to obtain labels for other frames

challenges that remain *e.g.* the pedestrians/moving cars violate the implicit static scene assumption, the scope of 3D data captured from the multiple views is limited in context of the whole scene and finally, one may never get observations for some aspects *e.g.* multi-view supervision cannot inform us that there is road below the cars parked on the side.

3.4.4 Object Reconstruction from RGB Supervision

We study the setting where only 2D color images of ShapeNet models are available as supervisory signal. In this scenario, our CNN predicts a per-voxel occupancy as well as a color value. We use the generalized event cost function from Eq. 3.10 to define the training loss. Some qualitative results are shown in Figure 3.7. We see the learned model can infer the correct shape as well as color, including the concavities in chairs, shading for hidden parts *etc.*.

Figure 3.7: Sample results on ShapeNet dataset using multiple RGB images as supervision for training. We show the input image (left) and the visualize 3D shape predicted using our learned model from two novel views. Best viewed in color.

3.4.5 ShapeNet Reconstruction without Pose Supervision

We again consider the ShapeNet dataset, but in this scenario to demonstrate our ability to learn without requiring known poses associated with the available observations. We use the loss formulation defined in Section 3.2.6 and show that we can learn both shape and pose prediction without direct supervision for either.

Dataset. We use the same splits as in Section 3.4.1. We render the training objects under two settings - a) origin centred (as in Section 3.4.1), or b) randomly translated around the origin. As the camera is always at a fixed distance away from the origin, the first setting corresponds to training with a known camera translation, but

Training Data	Multi-view & GT Pose		Multi-view w/o Rot		Multi-view w/o Rot & Trans	
class	Mask	Depth	Mask	Depth	Mask	Depth
aero	0.55	0.43	0.52	0.44	0.38	0.37
car	0.75	0.69	0.74	0.71	0.48	0.68
chair	0.42	0.45	0.40	0.43	0.35	0.37
mean	0.57	0.52	0.55	0.53	0.40	0.47

Table 3.3: Analysis of the performance for single-view shape prediction. We report the mean IoU on the test set using various supervision settings.

Training Data	GT Pose		MV w/o Rot				MV w/o Rot & Trans			
			Mask		Depth		Mask		Depth	
class	Acc	Err	Acc	Err	Acc	Err	Acc	Err	Acc	Err
aero	0.79	10.7	0.69	14.3	0.60	21.7	0.53	26.9	0.63	12.3
car	0.90	7.4	0.87	5.2	0.85	4.9	0.53	24.8	0.56	20.6
chair	0.85	11.2	0.81	7.8	0.83	8.6	0.55	24.0	0.62	19.1
mean	0.85	10.0	0.79	9.0	0.76	11.7	0.54	25.1	0.61	17.4

Table 3.4: Analysis of the performance for single-view pose prediction.We report the Pose Accuracy/Error: $\text{Acc}_{\frac{\pi}{6}}$ and Med-Err across different supervision settings.

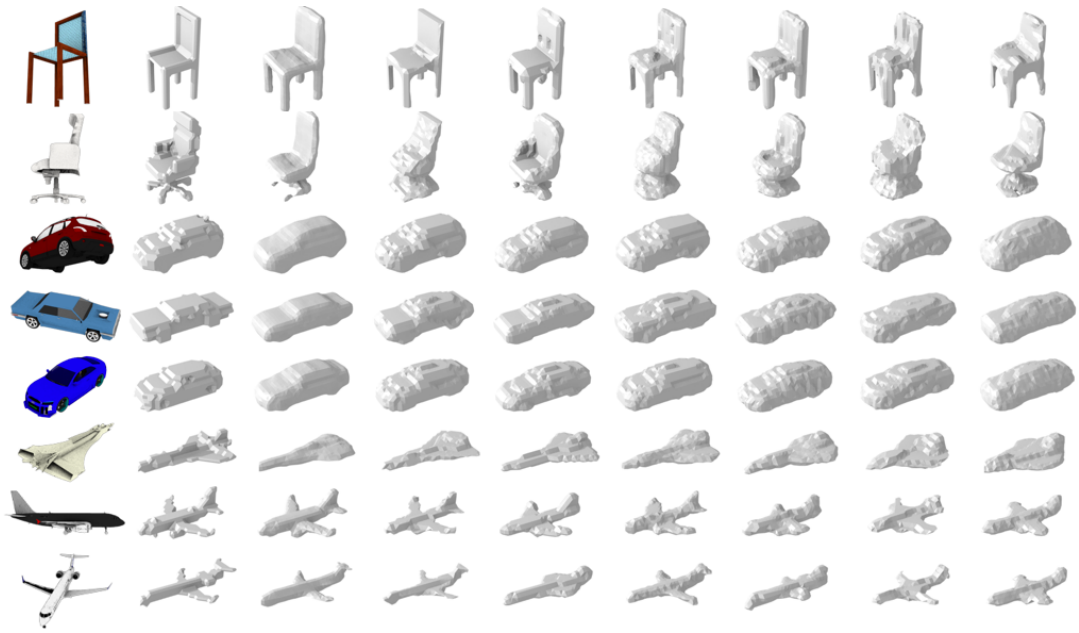

Figure 3.8: Shape predictions on the validation set using a single RGB input image. We visualize the voxel occupancies by rendering the corresponding mesh (obtained via marching cubes) from a canonical pose. Left to Right: a) Input Image b) Ground-truth c) 3D Supervised Prediction d,e) Multi-view & Pose Supervision (Mask, Depth) f,g) Mult-view w/o Rotation Supervision (Mask, Depth), and h,i) Mult-view w/o Rotation and Translation Supervision (Mask, Depth)

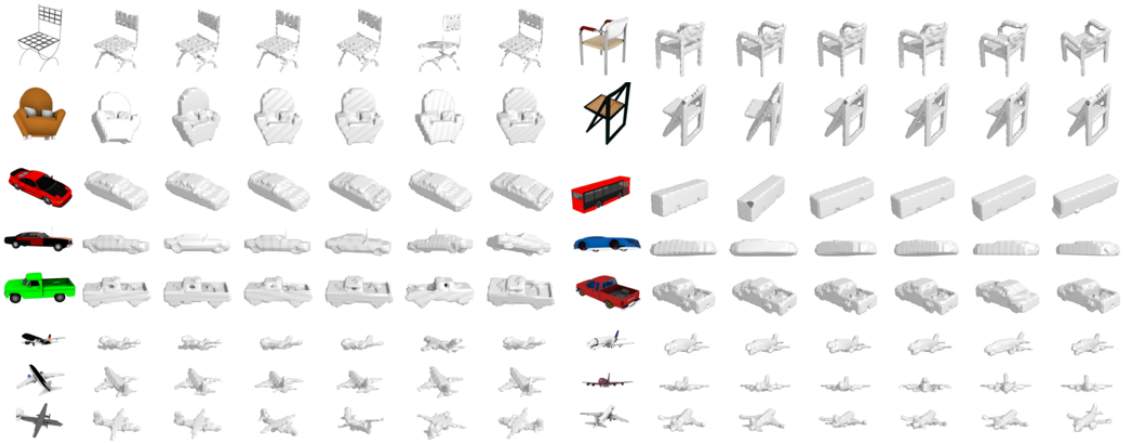

Figure 3.9: Rotation predictions on a random subset of the validation images. For visualization, we render the ground-truth voxel occupancies using the corresponding rotation. Left to Right: a) Input Image b) Ground-truth Rotation c) GT Supervised Prediction d,e) Multi-view w/o Rot Supervision (Mask, Depth), and f,g) Multi-view w/o Rot and Trans Supervision (Mask, Depth)

unknown rotation. The second corresponds to training with both translation and rotation unknown. To have a common test set across various control setting, we use the origin centered renderings for our validation and test sets.

Setup. We use the same evaluation setup, hyperparameters, and network architectures as used in Section 3.4.1, and additionally train a pose CNN which predicts the (unknown) associated camera poses. As we jointly learn both shape ans pose prediction, the obtained reconstructions are in some arbitrary canonical frame different from the ShapeNet canonical frame. Therefore, before evaluating our results, we compute an optimal rotation to best align the predictions to the canonical ShapeNet frame.

In addition to evaluating the target setting where we learn without pose supervision, we also report control settings regarding training with known camera poses. This is similar to the setup in Section 3.4.1, with the difference that we instead use the pose-differentiable loss defined in Section 3.2.6.

Shape Prediction Results. Our results and the performance under various control settings with stronger supervision is reported in Table 3.3 and visualized in Figure 3.8. In general, we observe that the performance degrades gracefully as the amount of supervision available is reduced. This clearly indicates that our approach is able to learn single-view shape prediction despite the lack of either shape or pose information during training. As expected, we also observe that we cannot learn about concavities in chairs via consistency against mask validation images, though we can do so using depth images. We observe a noticeable performance drop in case of mask supervision with unknown translation, as this setting results in scale ambiguities which our evaluation does not account for *e.g.* we learn to predict larger cars, but further away, and this results in a low empirical score.

Pose Estimation Results. The results of our approach are reported in Table 3.4 and visualized in Figure 3.9. We report performance using the metrics used in [142] – median angular error and the fraction of instances with error less than a threshold of 30 degrees. We observe a similar trend for the task of pose prediction – that our approach performs comparably to directly supervised learning using ground-truth pose supervision. Interestingly, we often get lower median errors than the supervised setting. We attribute this to the different topologies of the loss functions. The squared L2 loss used in the supervised setting yields small gradients if the pose is almost correct. Our consistency loss however, would want the observation image to perfectly align with the shape via the predicted pose.

Figure 3.10: Visualization of predictions using the Stanford Online Product Dataset. (Top) Input image. (Middle) Predicted shape in the emergent canonical pose. (Bottom) Predicted shape rotated according to the predicted pose.

3.4.6 Learning from Online Product Images

Online images of products are a natural source of multi-view observations. While no associated shape or pose supervision is available in such setting, we demonstrate that we can learn 3D prediction systems using such data.

Dataset. We examined the 'chair' object category from the Stanford Online Products Dataset [129] which comprises of automatically downloaded images from eBay.com [1]. Since multiple images (views) of the same product are available, we can leverage our approach to learn from this data. As we also require associated foreground masks for these images, we use an out-of-the-box semantic segmentation system [21] to obtain these. However, the obtained segmentation masks are often incorrect. Additionally, many of the product images were not suited for our setting as they only comprised of a zoom-in of a small portion of the instance (*e.g.* chair wheel). We therefore manually selected images of unoccluded/untruncated instances with a reasonably accurate (though still noisy) predicted segmentation. We then used the object instances with at least 2 valid views for training. This results in a filtered dataset of 282 instances with 3.65 views on average per instance.

Results. We can apply our approach to learn from this dataset comprising of multiple views with associated (approximate) foreground masks. Since the camera intrinsics are unknown, we assume a default intrinsic matrix. We then learn to predict the (unknown) translation and rotation via the pose CNN g_ϕ and the (unknown) shape via the shape CNN f_θ using the available multi-view supervision. Note that the learned CNNs are trained from scratch, and that we use the same architecture/hyperparameters as in the ShapeNet experiments.

Some results (on images of novel instances) using our learned CNN are visualized in Figure 3.10. We see that we can learn to predict meaningful 3D structure and infer the appropriate shape and pose corresponding to the input image. Since only foreground mask supervision is leveraged, we cannot learn to infer the concavities in shapes. We also observe confusion across poses which result in similar foreground masks. However, we feel that this result using training data derived from a challenging real world setting, concretely demonstrates our method's ability to learn despite the lack of direct shape or pose supervision. To the best of our knowledge, this is the first such result and it represents an encouraging step forward.

3.5 Discussion

We have presented a differentiable formulation for consistency between a 3D shape and a 2D observation and demonstrated its applications for learning single-view reconstruction in various scenarios using multi-view supervision. These are, however, a number of challenges yet to be addressed. Our formulation is applicable to voxel-occupancy based representations and an interesting direction is to extend these ideas to alternate representations which allow finer predictions *e.g.* meshes or octrees. Finally, while our approach allows us to bypass the availability of ground-truth 3D information for training, we still rely on multi-view training data, which may be tedious to acquire in certain scenarios. We address these challenges in Chapter 4.

Chapter 4

Learning Mesh Reconstruction from Image Collections

How can we learn to model the 3D structure for all object classes? In Chapter 3, we demonstrated that leveraging multiple views per instance of a category is a possibility. This supervision allows us to learn a CNN that can predict a volumetric 3D representation for new instances. While this is encouraging, volumetric representations are fundamentally limited in their ability to represent finer details and additional surface properties *e.g.* texture. Further, the reliance on multiple views per instance is also restrictive – while we can obtain these for inanimate objects, it is rather difficult to conceive of a similar setup being applicable for animate categories *e.g.* birds.

In this chapter, we address these challenges, and present a computational model that can similarly infer a mesh based 3D representation given just a single image. Additionally, as illustrated in Figure 4.1, the learning only relies on an annotated 2D image collection of a given object category, comprising of foreground masks and semantic keypoint labels. Our training procedure, depicted in Figure 4.2, forces a common prediction model to explain all the image evidences across many examples of an object category. This allows us to learn a meaningful 3D structure despite only using a single-view per training instance, without relying on any ground-truth 3D data for learning.

At inference, given a single unannotated image of a novel instance, such as the one depicted in Figure 4.1, our learned model allows us to infer the shape, camera pose, and texture of the underlying object. We represent the shape as a 3D mesh in a

This chapter is based on joint work with Angjoo Kanazawa, Alexei A. Efros, and Jitendra Malik, which will appear in the proceedings of ECCV, 2018 [76].

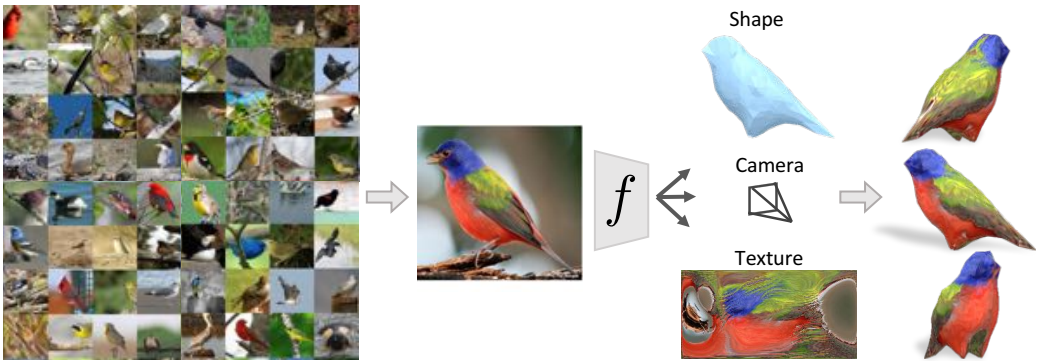

Figure 4.1: Given an annotated image collection of an object category, we learn a predictor f that can map a novel image I to its 3D shape, camera pose, and texture.

canonical frame, where the predicted camera transforms the mesh from this canonical space to the image coordinates. The particular shape of each instance is instantiated by deforming a learned category-specific mean shape with instance-specific predicted deformations. The use of this shared 3D space affords numerous advantages as it implicitly enforces correspondences across 3D representations of different instances. As we detail in Section 4.1, this allows us to formulate the task of inferring mesh texture of different objects as that of predicting pixel values in a common texture representation. Furthermore, we can also easily associate semantic keypoints with the predicted 3D shapes.

Our shape representation is an instantiation of deformable models, the history of which can be traced back to D'Arcy Thompson [137], who in turn was inspired by the work of Dürer [29]. Thompson observed that shapes of objects of the same category may be aligned through geometrical transformations. Cootes and Taylor [25] operationalized this idea to learn a class-specific model of deformation for 2D images. Pioneering work of Blanz and Vetter [14] extended these ideas to 3D shapes to model the space of faces. These techniques have since been applied to model human bodies [3, 96], hands [81, 136], and more recently on quadruped animals [174]. Unfortunately, all of these approaches require a large collection of 3D data to learn the model, preventing their application to categories where such data collection is impractical. In contrast, our approach is able to learn using only an annotated image collection.

Sharing our motivation for relaxing the requirement of 3D data to learn morphable models, some related approaches have examined the use of similarly annotated image collections. Cashman and Fitzgibbon [19] use keypoint correspondences and segmentation masks to learn a morphable model of dolphins from images. Kar *et al.* [78] extend this approach to general rigid object categories. Both approaches

follow a *fitting-based* inference procedure, which relies on mask (and optionally keypoint) annotations at test-time and is computationally inefficient. We instead follow a *prediction-based* inference approach, and learn a parametrized predictor which can directly infer the 3D structure from an unannotated image. Moreover, unlike these approaches, we also address the task of texture prediction which cannot be easily incorporated with these methods.

While deformable models have been a common representation for 3D inference, the recent advent of deep learning based prediction approaches has resulted in a plethora of alternate representations being explored using varying forms of supervision. Relying on ground-truth 3D supervision (using synthetic data), some approaches have examined learning voxel [24, 45, 155, 170], point cloud [36] or octree [56, 135] prediction. While some learning based methods do pursue mesh prediction [75, 87, 89], they also rely on 3D supervision which is only available for restricted classes or in a synthetic setting. Reducing the supervision to multi-view masks [55, 111, 144, 160] or depth images [144] has been explored for voxel prediction, but the requirement of multiple views per instance is still restrictive. While these approaches show promising results, they rely on stronger supervision (ground-truth 3D or multi-view) compared to our approach.

In the context of these previous approaches, the proposed approach differs primarily in three aspects:

- *Shape representation and inference method.* We combine the benefits of the classically used deformable mesh representations with those of a learning based prediction mechanism. The use of a deformable mesh based representation affords several advantages such as memory efficiency, surface-level reasoning and correspondence association. Using a learned prediction model allows efficient inference from a single unannotated image

- *Learning from an image collection.* Unlike recent CNN based 3D prediction methods which require either ground-truth 3D or multi-view supervision, we only rely on an annotated image collection, with only one available view per training instance, to learn our prediction model.

- *Ability to infer texture.* There is little past work on predicting the 3D shape and the texture of objects from a single image. Recent *prediction-based* learning methods use representations that are not amenable to textures (*e.g.* voxels). The classical deformable model *fitting-based* approaches cannot easily incorporate texture for generic objects. An exception is texture inference on human faces [14, 118], but these approaches require a large-set of 3D ground truth data

with high quality texture maps. Our approach enables us to pursue the task of texture inference from image collections alone, and we address the related technical challenges regarding its incorporation in a learning framework.

4.1 Approach

We aim to learn a predictor f_θ (parameterized as a CNN) that can infer the 3D structure of the underlying object instance from a single image I. The prediction $f_\theta(I)$ is comprised of the 3D shape of the object in a canonical frame, the associated texture, as well as the camera pose. The shape representation we pursue in this work is of the form of a 3D mesh. This representation affords several advantages over alternates like probabilistic volumetric grids *e.g.* amenability to texturing, correspondence inference, surface level reasoning and interpretability.

The overview of the proposed framework is illustrated in Figure 4.2. The input image is passed through an encoder to a latent representation that is shared by three modules that estimate the camera pose, shape deformation, and texture parameters. The deformation is added to the learned category-level mean shape to obtain the final predicted shape. The objective of the network is to minimize the corresponding losses when the shape is rendered onto the image. We train a separate model for each object category.

We first present the representations predicted by our model in Section 4.1.1, and then describe the learning procedure in Section 4.1.2. We initially present our framework for predicting shape and camera pose, and then describe how the model is extended to predict the associated texture in Section 4.1.3.

4.1.1 Inferred 3D Representation

Given an image I of an instance, we predict $f_\theta(I) \equiv (M, \pi)$, a mesh M and camera pose π to capture the 3D structure of the underlying object. In addition to these directly predicted aspects, we also learn the association between the mesh vertices and the category-level semantic keypoints. We describe the details of the inferred representations below.

Shape Parametrization. We represent the shape as a 3D mesh $M \equiv (V, F)$, defined by vertices $V \in \mathbb{R}^{|V|\times 3}$ and faces F. We assume a fixed and pre-determined mesh connectivity, and use the faces F corresponding to a spherical mesh. The vertex positions V are instantiated using (learned) instance-independent mean vertex locations $\bar{V}$ and instance-dependent predicted deformations Δ_V, which when added,

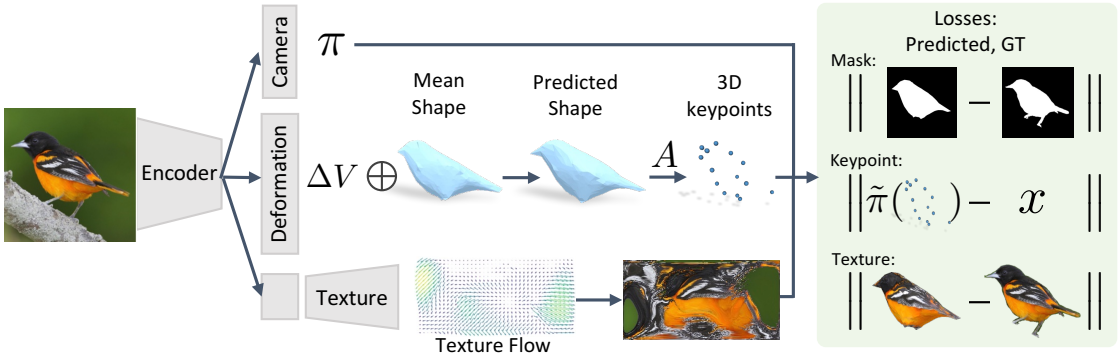

Figure 4.2: **Overview of the mesh prediction framework.** An image I is passed through a convolutional encoder to a latent representation that is shared by modules that estimate the camera pose, deformation and texture parameters. Deformation is an offset to the learned mean shape, which when added yield instance specific shapes in a canonical coordinate frame. We also learn correspondences between the mesh vertices and the semantic keypoints. Texture is parameterized as an UV image, which we predict through texture flow (see Section 4.1.3). The objective is to minimize the distance between the rendered mask, keypoints and textured rendering with the corresponding ground truth annotations. We do not require ground truth 3D shapes or multi-view cues for training.

yield instance vertex locations $V = \bar{V} + \Delta_V$. Intuitively, the mean shape $\bar{V}$ can be considered as a learnt bias term for the predicted shape V.

Camera Projection. We model the camera with weak-perspective projection and predict, from the input image I, the scale $s \in \mathbb{R}$, translation $\mathbf{t} \in \mathbb{R}^2$, and rotation (captured by quaternion $\mathbf{q} \in \mathbb{R}^4$). We use $\pi(P)$ to denote the projection of a set of 3D points P onto the image coordinates via the weak-perspective projection defined by $\pi \equiv (s, \mathbf{t}, \mathbf{q})$.

Associating Semantic Correspondences. As we represent the shape using a category-specific mesh in the canonical frame, the regularities across instances encourage semantically consistent vertex positions across instances, thereby implicitly endowing semantics to these vertices. We can use this insight and learn to explicitly associate semantic keypoints *e.g.*, beak, legs *etc.* with the mesh via a keypoint assignment matrix $A \in \mathcal{R}_+^{|K|\times|V|}$ s.t. $\sum_v A_{k,v} = 1$. Here, each row A_k represents a probability distribution over the mesh vertices of corresponding to keypoint k, and can be understood as approximating a one-hot vector of vertex selection for each keypoint. As we describe later in our learning formulation, we encourage each A_k to be a peaked distribution. Given the vertex positions V, we can infer the location v_k for the k^{th} keypoint as $v_k = \sum_v A_{k,v} v$. More concisely, the keypoint locations

induced by vertices V can be obtained as $A \cdot V$. We initialize the keypoint assignment matrix A uniformly, but over the course of training it learns to better associate semantic keypoints with appropriate mesh vertices.

In summary, given an image I of an instance, we predict the corresponding camera π and the shape deformation Δ_V as $(\pi, \Delta_V) = f(I)$. In addition, *we also learn* (across the dataset), instance-independent parameters $\{\bar{V}, A\}$. As described above, these category-level (learned) parameters, in conjunction with the instances-specific predictions, allow us to recover the mesh vertex locations V and coordinates of semantic keypoints $A \cdot V$.

4.1.2 Learning from an Image Collection

We present an approach to train f_θ without relying on strong supervision in the form of ground truth 3D shapes or multi-view images of an object instance. Instead, we guide the learning from an image collection annotated with sparse keypoints and segmentation masks. Such a setting is more natural and easily obtained, particularly for animate and deformable objects such as birds or animals. It is extremely difficult to obtain scans, or even multiple views of the same instance for these classes, but relatively easier to acquire a single image for numerous instances.

Given the annotated image collection, we train f_θ by formulating an objective function that consists of instance specific losses and priors. The instance-specific energy terms ensure that the predicted 3D structure is consistent with the available evidence (masks and keypoints) and the priors encourage generic desired properties *e.g.* smoothness. As we learn a common prediction model f_θ across many instances, the common structure across the category allows us to learn meaningful 3D prediction despite only having a single-view per instance.

Training Data. We assume an annotated training set $\{(I_i, S_i, x_i)\}_{i=1}^N$ for each object category, where I_i is the image, S_i is the instance segmentation, and $x_i \in \mathbb{R}^{2\times K}$ is the set of K keypoint locations. As previously leveraged by [78, 148], applying structure-from-motion to the annotated keypoint locations additionally allows us to obtain a rough estimate of the weak-perspective camera $\tilde{\pi}_i$ for each training instance. This results in an augmented training set $\{(I_i, S_i, x_i, \tilde{\pi}_i)\}_{i=1}^N$ which we use for training our predictor f_θ.

Instance Specific Losses. We ensure that the predicted 3D structure matches the available annotations. Using the semantic correspondences associated to the mesh via the keypoint assignment matrix A, we formulate a keypoint reprojection loss. This term encourages the predicted 3D keypoints to match the annotated 2D

keypoints when projected onto the image:

$$L_{\texttt{reproj}} = \sum_i ||x_i - \tilde{\pi}_i(AV_i)||_2. \tag{4.1}$$

Similarly, we enforce that the predicted 3D mesh, when rendered in the image coordinates, is consistent with the annotated foreground mask: $L_{\texttt{mask}} = \sum_i ||S_i - \mathcal{R}(V_i, F, \tilde{\pi}_i)||_2$. Here, $\mathcal{R}(V, F, \pi)$ denotes a rendering of the segmentation mask image corresponding to the 3D mesh $M = (V, F)$ when rendered through camera π. In all of our experiments, we use Neural Mesh Renderer [80] to provide a differentiable implementation of $\mathcal{R}(\cdot)$.

We also train the predicted camera pose to match the corresponding estimate obtained via structure-from-motion using a regression loss $L_{\texttt{cam}} = \sum_i ||\tilde{\pi}_i - \pi_i||_2$. We found it advantageous to use the structure-from-motion camera $\tilde{\pi}_i$, and not the predicted camera π_i, to define $L_{\texttt{mask}}$ and $L_{\texttt{reproj}}$ losses. This is because during training, in particular the initial stages when the predictions are often incorrect, an error in the predicted camera can lead to high errors despite accurate shape, and possibly adversely affect learning.

Priors. In addition to the data-dependent losses which ensure that the predictions match the evidence, we leverage generic priors to encourage additional properties. The prior terms that we use are:

Smoothness. In the natural world, shapes tend to have a smooth surface and we would like our recovered 3D shapes to behave similarly. An advantage of using a mesh representation is that it naturally affords reasoning at the surface level. In particular, enforcing smooth surface has been extensively studied by the Computer Graphics community [108, 131]. Following the literature, we formulate surface smoothness as minimization of the mean curvature. On meshes, this is captured by the norm of the graph Laplacian, and can be concisely written as $L_{\texttt{smooth}} = ||LV||_2$, where L is the discrete Laplace-Beltrami operator. We construct L once using the connectivity of the mesh and this can be expressed as a simple linear operator on vertex locations.

Deformation Regularization. In keeping with a common practice across deformable model approaches [14, 19, 78], we find it beneficial to regularize the deformations as it discourages arbitrarily large deformations and helps learn a meaningful mean shape. The corresponding energy term is expressed as $L_{\texttt{def}} = ||\Delta_V||_2$.

Keypoint association. As discussed in Section 4.1.1, we encourage the keypoint assignment matrix A to be a peaked distribution as it should intuitively correspond to a one-hot vector. We therefore minimize the average entropy over all keypoints: $L_{\texttt{vert2kp}} = \frac{1}{|K|} \sum_k \sum_v -A_{k,v} \log A_{k,v}$.

In summary, the overall objective for shape and camera is

$$L = L_{\texttt{reproj}} + L_{\texttt{mask}} + L_{\texttt{cam}} + L_{\texttt{smooth}} + L_{\texttt{def}} + L_{\texttt{vert2kp}}. \tag{4.2}$$

Symmetry Constraints. Almost all common object categories, including the ones we consider, exhibit reflectional symmetry. To exploit this structure, we constrain the predicted shape and deformations to be mirror-symmetric. As our mesh topology corresponds to that of a sphere, we identify symmetric vertex pairs in the initial topology. Given these pairs, we only learn/predict parameters for one vertex in each pair for the mean shape $\bar{V}$ and deformations Δ_V.

Initialization and Implementation Details. While our mesh topology corresponds to a sphere, following previous fitting based deformable model approaches [78], we observe that a better initialization of the mean vertex positions $\bar{V}$ speeds up learning. We compute the convex hull of the mean keypoint locations obtained during structure-from-motion and initialize the mean vertex locations to lie on this convex hull. As the different energy terms in Eq. 4.2 have naturally different magnitudes, we weight them accordingly to normalize their contribution.

4.1.3 Incorporating Texture Prediction

In our formulation, all recovered shapes share a common underlying 3D mesh structure – each shape is a deformation of the mean shape. We can leverage this property to reduce texturing of a particular instance to predicting the texture of the mean shape. Our mean shape is isomorphic to a sphere, whose texture can be represented as an image I^{uv}, the values of which get mapped onto the surface via a fixed UV mapping (akin to unrolling a globe into a flat map) [71]. Therefore, we formulate the task of texture prediction as that of inferring the pixel values of I^{uv}. This image can be thought of as a canonical appearance space of the object category. For example, a particular triangle on the predicted shape always maps to a particular

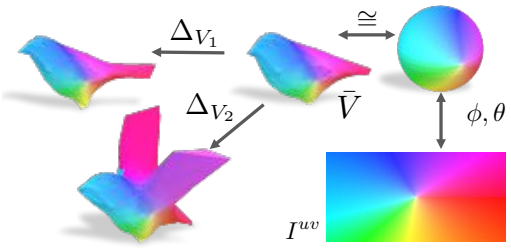

Figure 4.3: **Illustration of the UV mapping.** We illustrate how a texture image I^{uv} can induce a corresponding texture on the predicted meshes. A point on a sphere can be mapped onto the image I^{uv} via using spherical coordinates. As our mean shape has the same mesh geometry (vertex connectivity) as a sphere we can transfer this mapping onto the mean shape. The different predicted shapes, in turn, are simply deformations of the mean shape and can use the same mapping.

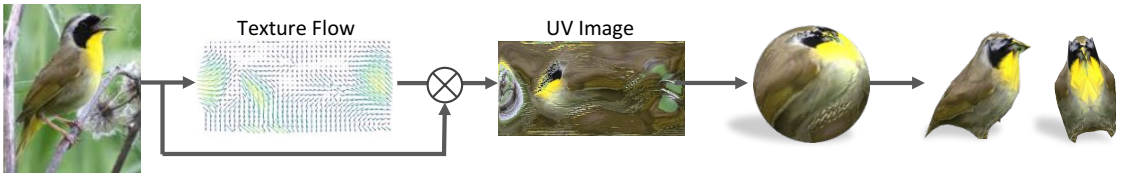

Figure 4.4: **Illustration of texture flow.** We predict a texture flow $\mathcal{F}$ that is used to bilinearly sample the input image I to generate the texture image I^{uv}. We can use this predicted UV image I^{uv} to then texture the instance mesh via the UV mapping procedure illustrated in Figure 4.3.

region in I^{uv}, irrespective of how it was
deformed. This is illustrated in Figure 4.3. In this texture parameterization, each pixel in the UV image has a consistent semantic meaning, thereby making it easier for the prediction model to leverage common patterns such as correlation between the bird back and the body color.

We incorporate texture prediction module into our framework by setting up a decoder that upconvolves the latent representation to the spatial dimension of I^{uv}. While directly regressing the pixel values of I^{uv} is a feasible approach, this often results in blurry images. Instead, we take inspiration from [169] and formulate this task as that of predicting the appearance flow. Instead of regressing the pixel values of I^{uv}, the texture module outputs where to copy the color of the pixel from the original input image. This prediction mechanism, depicted in Figure 4.4, easily allows our predicted texture to retain the details present in the input image. We refer to this output as 'texture flow' $\mathcal{F} \in \mathbb{R}^{H_{uv} \times W_{uv} \times 2}$, where H_{uv}, W_{uv} are the height and width of I^{uv}, and $\mathcal{F}(u, v)$ indicates the (x, y) coordinates of the input image to sample the pixel value from. This allows us to generate the UV image $I^{uv} = G(I; \mathcal{F})$ by bilinear sampling G of the original input image I according to the predicted flow $\mathcal{F}$. This is illustrated in Figure 4.4.

Now we formulate our texture loss, which encourages the rendered texture image to match the foreground image:

$$L_{\texttt{texture}} = \sum_i \text{dist}(S_i \odot I_i, S_i \odot \mathcal{R}(V_i, F, \tilde{\pi}_i, I^{uv})). \tag{4.3}$$

$\mathcal{R}(V_i, F, \tilde{\pi}_i, I_i^{uv})$ is the rendering of the 3D mesh with texture defined by I^{uv}. We use the perceptual metric of Zhang *et al.* [165] as the distance metric.

The loss function above provides supervisory signals to regions of I^{uv} corresponding to the foreground portion of the image, but not to other regions of I^{uv} corresponding to parts that are not directly visible in the image. While the common

patterns across the dataset *e.g.* similar colors for bird body and back can still allow meaningful prediction, we find it helpful to add a further loss that encourages the texture flow to select pixels only from the foreground region in the image. This can be simply expressed by sampling the distance transform field of the foreground mask $\mathcal{D}_S$ (where for all points x in the foreground, $\mathcal{D}_S(x) = 0$) according to $\mathcal{F}$ and summing the resulting image:

$$L_{\texttt{dt}} = \sum_i \sum_{u,v} G(\mathcal{D}_{S_i}; \mathcal{F}_i)(u, v). \tag{4.4}$$

In contrast to inferring the full texture map, directly sampling the actual pixel values that the predicted mesh projects onto creates holes and leaking of the background texture at the boundaries. Similarly to the shape parametrization, we also explicitly encode symmetry in our I^{uv} prediction, where symmetric faces gets mapped on to the same UV coordinate in I^{uv}. Additionally, we only back-propagate gradients from $L_{\texttt{texture}}$ to the predicted texture (and not the predicted shape) since bilinear sampling often results in high-frequency gradients that destabilize shape learning. Our shape prediction is therefore learned only using the objective in Eq. 4.2, and the losses $L_{\texttt{texture}}$ and $L_{\texttt{dt}}$ can be viewed as encouraging prediction of correct texture 'on top' of the learned shape.

4.2 Experiments

We demonstrate the ability of our presented approach to learn single-view inference of shape, texture and camera pose using only a category-level annotated image collection. As a running example, we consider the 'bird' object category as it represents a challenging scenario that has not been addressed via previous approaches. We first present, in Section 4.2.1, our experimental setup, describing the annotated image collection and CNN architecture used.

As ground-truth 3D is not available for benchmarking, we present extensive qualitative results in Section 4.2.2, demonstrating that we learn to predict meaningful shapes and textures across birds. We also show we capture the shape deformation space of the category and that the implicit correspondences in the deformable model allow us to have applications like texture transfer across instances.

We also present some quantitative results to provide evidence for the accuracy of our shape and camera estimates in Section 4.2.3. While there has been little work for reconstructing categories like birds, some approaches have examined the task of learning shape prediction using an annotated image collection for some rigid classes. In Section 4.2.4 we present our method's results on some additional representative

categories, and show that our method performs comparably, if not better than the previously proposed alternates while having several additional advantages *e.g.* learning semantic keypoints and texture prediction.

4.2.1 Experimental Setup

Dataset. We use the CUB-200-2011 dataset [150], which has 6000 training and test images of 200 species of birds. Each image is annotated with the bounding box, visibility indicator and locations of 14 semantic keypoints, and the ground truth foreground mask. We filter out nearly 300 images where the visible number of keypoints are less than or equal to 6, since these typically correspond to truncated close shots. We divide the test set in half to create a validation set, which we use for hyper-parameter tuning.

Network Architecture. A schematic of the various modules of our prediction network is depicted in Figure 4.2. The encoder consists of an ImageNet pretrained ResNet-18 [61], followed by a convolutional layer that downsamples the spatial and the channel dimensions by half. This is vectorized to form a 4096-D vector, which is sent to two fully-connected layers to get to the shared latent space of size 200. The deformation and the camera prediction components are linear layers on top of this latent space. The texture flow component consists of 5 upconvolution layers where the final output is passed through a *tanh* function to keep the flow in a normalized [-1, 1] space. We use the neural mesh renderer [80] so all rendering procedures are differentiable. All images are cropped using the instance bounding box and resized such that the maximum image dimension is 256. We augment the training data on the fly by jittering the scale and translation of the bounding box and with image mirroring. Our mesh geometry corresponds to that of a perfectly symmetric sphere with 642 vertices and 1280 faces.

4.2.2 Qualitative Results

We visualize the results and application of our learned predictor using the CUB dataset. We show various reconstructions corresponding to different input images, visualize some of the deformation modes learned, and show that the common deformable model parametrization allows us to transfer the texture of one instance onto another.

Single-view 3D Reconstruction. We show sample reconstruction results on images from the CUB test set in Figure 4.5. We show the predicted shape and texture from the inferred camera viewpoint, as well as from novel views.

Figure 4.5: **Sample results.** We show predictions of our approach on images from the test set. For each input image on the left, we visualize (in order): the predicted 3D shape and texture viewed from the predicted camera, and textured shape from three novel viewpoints.

We observe that our learned model can accurately predict the shape, estimate the camera and also infer meaningful texture from the corresponding input image. Our predicted 3D shape captures the overall shape (fat or thin birds), and even some finer details *e.g.* beaks or large deformations *e.g.* flying birds. Additionally, our learned pose and texture prediction are accurate and realistic across different instances. We observe that the error modes corresponds to not predicting rare poses, and inability to incorporate asymmetric articulation. However, we feel that these predictions learned using only an annotated image collection are encouraging.

Learned shape space. The presented approach represents the shape of an instance via a category-level learned mean shape and a per-instance predicted deformation Δ_V. To gain insight into the common modes of deformation captured via our predictor, obtained the principal deformation modes by computing PCA on the predicted deformations across all instances in the training set.

We visualize in Figure 4.6 our mean shape deformed in directions corresponding three common deformation modes. We note that these plausibly correspond to some of the natural factors of variation in the 3D structure across birds *e.g.* fat or thin birds, opening of wings, deformation of tails and legs.

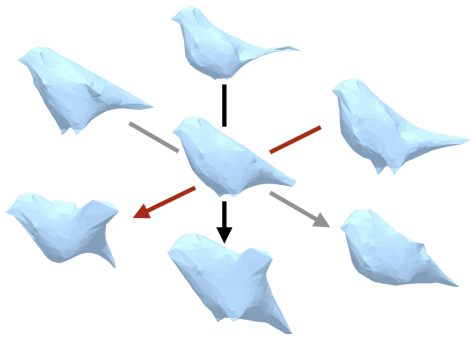

Figure 4.6: **Learned deformation modes.** We visualize the space of learned shapes by depicting the mean shape (centre) and three common modes of deformation as obtained by PCA on the predicted deformations across the dataset.

Texture Transfer. Recall that the textures of different instance in our formulation are captured in a canonical appearance space in the form of a predicted 'texture image' I_{uv}. This parametrization allows us to easily modify the surface appearance, and in particular transfer texture across instances.

We show some results in Figure 4.7 where we sample pairs of instances, and transfer the texture from one image onto the predicted shape of the other. We can achieve this by simply using the predicted texture image corresponding to the first when rendering the predicted 3D for the other. We note that even though the two views might be different, since the underlying 'texture image' space is consistent, the transferred texture is also semantically consistent *e.g.* the colors corresponding to the one bird's body are transferred onto the other bird's body.

Figure 4.7: **Texture Transfer Results.** Our representation allows us to easily transfer the predicted texture across instances using the canonical appearance image (see text for details). We visualize sample results of texture transfer across different pairs of birds. For each pair, we show (left): the input image, (middle): the predicted textured mesh from the predicted viewpoint, and (right): the predicted mesh textured using the predicted texture of the other bird.

4.2.3 Quantitative Evaluation

We attempt to indirectly measure the quality of our recovered reconstructions on the CUB dataset. As there is no ground-truth 3D available for benchmarking, we instead evaluate the mask reprojection accuracy. For each test instance in the CUB dataset, we obtain a mask prediction via rendering the predicted 3D shape from the predicted camera viewpoint. We then compute the intersection over union (IoU) of this predicted mask with the annotated ground-truth mask. Note that to correctly predict the foreground mask, we need both, accurate shape and accurate camera.

Our results are plotted in Figure 4.8. We compare the accuracy our full shape prediction (using learned mean shape $\bar{V}$ and predicted deformation Δ_V) against only using the learned mean shape to obtain the predicted mask. We observe that

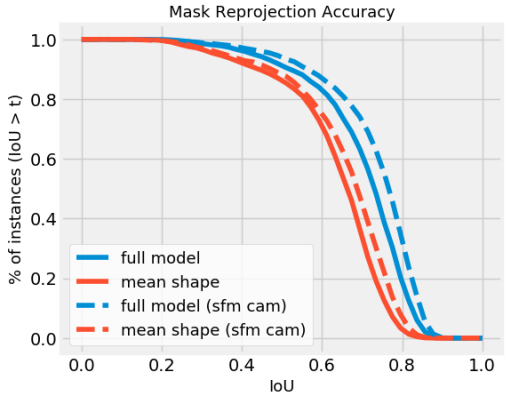

Figure 4.8: **Mask reprojection accuracy evaluation on CUB.** We plot the fraction of test instances with IoU between the predicted and ground-truth mask higher than different thresholds (higher is better) and compare the predictions using the full model against only using the learned mean shape. We report the reprojection accuracy using predicted cameras and cameras obtained via structure-from-motion based on keypoint annotation.

Method	Aeroplane	Car
CSDM [78]	0.40	0.60
DRC [144]	0.42	0.67
Ours	0.46	0.64

Table 4.1: **Reconstruction evaluation using PASCAL 3D+.** We report the mean intersection over union (IoU) on PASCAL 3D+ to benchmark the obtained 3D reconstructions (higher is better). We compare to previous deformable model fitting-based [78] and volumetric prediction [144] approaches that use similar image collection supervision. Note that our approach can additionally predict texture and semantics.

the predicted deformations result in improvements, indicating that we are able to capture the specifics of the shape of different instances. Additionally, we also report the performance using the camera obtained via structure from motion (which uses ground-truth annotated keypoints) instead of using the predicted camera. We note that comparable results in the two settings demonstrate the accuracy of our learned camera estimation. Lastly, we can also measure our keypoint reprojection accuracy using the percentage of correct keypoints (PCK) metric [161]. We similarly observe that our full predicted shape performs (slightly) better than only relying on the category-level mean shape – by obtaining a PCK (at normalized distance threshold 0.1) of 0.72 compared to 0.71. The improvement over the mean shape is less prominent in this scenario as most of the semantic keypoints defined are on the torso and therefore typically undergo only small deformations.

4.2.4 Evaluation on Other Object Classes

While our primary results focus on predicting the 3D shape and texture of birds using the CUB dataset, we note that some previous approaches have examined the task of shape inference/prediction using a similar annotated image collection as

Figure 4.9: **Pascal 3D+ results.** We show predictions of our approach on images from the test set. For each input image on the left, we visualize (in order): the predicted 3D shape viewed from the predicted camera, the predicted shape with texture viewed from the predicted camera, and the shape with texture viewed from a novel viewpoint.

supervision. While these previous methods do not infer texture, we can compare our shape predictions against those obtained by these techniques.

We compare to previous deformable model fitting-based [78] and volumetric prediction [144] methods using the PASCAL 3D+ dataset and examine the car and aeroplane categories. Both of these approaches can leverage the annotation we have available *i.e.* segmentation masks and keypoints to learn 3D shape inference (although [144] requires annotated cameras instead of keypoints). Similar to [144], we use PASCAL VOC and Imagenet images with available keypoint annotations from PASCAL3D+ to train our model, and use an off-the shelf segmentation algorithm [60] to obtain foreground masks for the ImageNet subset.

We report the mean IoU evaluation on the test set in Table 4.1 and observe that we perform comparably, if not better than these alternate methods. We also note that our approach yields additional outputs *e.g.* texture, that these methods do not. We visualize some predictions in Figure 4.9. While our predicted shapes are often reasonable, the textures have more errors due to shiny regions (*e.g.* for cars) or smaller amount of training data (*e.g.* for aeroplanes).

4.3 Discussion

We have presented a framework for learning single-view prediction of a textured 3D mesh using an image collection as supervision. However, we have by no means solved the problem in the general case, and a number of interesting challenges and possible directions remain. Our formulation addresses shape change and articulation via a similar shape deformation mechanism, and it would be beneficial to extend our deformable shape model to explicitly allow articulation. Additionally, while we presented a method to synthesize texture via copying image pixels, a more sophisticated mechanism that allows both, copying image content and synthesizing

novel aspects might be desirable. Finally, it would be desirable to relax the need of annotation even further, and investigate learning similar prediction models using unannotated image collections.

Part II

Inferring Compositional 3D Representations

Chapter 5

Unsupervised Learning of Shape Abstractions

> "Treat nature by means of the cylinder, the sphere,
> the cone, everything brought into proper perspective"
>
> *Paul Cezanne*

We demonstrated in Part I that we can learn to predict 3D structure using only image-based supervision. While we may have the capability to infer 3D, what representation should we pursue? Cezanne's insight is that an object can be conceived as assembled from a set of simpler underlying entities. In this chapter, we operationalize this insight, and present a learning framework for abstracting complex shapes using 3D volumetric primitives.

The idea of representing a complex shape in terms of simper entities has resurfaced multiple times in the vision and graphics literature. In computer vision, generalized cylinders were introduced by Binford back in 1971, where a cross-sectional area is swept along a straight or curved axis while possibly being shrunk or expanded during the process [13]. One of the key motivations was *parsimony of description* – an object could be described by relatively few generalized cylinders, each of which in turn requiring only a few parameters. Volumetric primitives remained popular through the 1990s as they provided a coherent framework for explaining shape inference from a single image, perceptual organization, as well as recognition of a 3D object from 2D views. However, fitting generalized cylinders to image data required considerable

This chapter is based on joint work with Hao Su, Leonidas J. Guibas, Alexei A. Efros, and Jitendra Malik, presented primarily as it appeared in the proceedings of CVPR, 2017 [143].

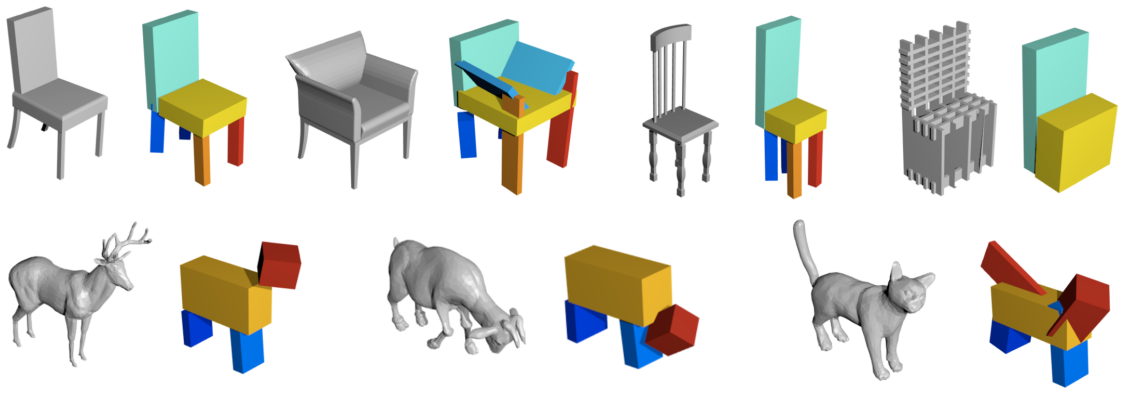

Figure 5.1: Examples of chair and animal shapes assembled by composing simple volumetric primitives (cuboids). The obtained reconstructions allows an interpretable representation for each object and provides a consistent parsing across shapes *e.g.* chair seats are captured by the same primitive across the category.

hand crafting, and as machine learning techniques for object recognition came to the fore in the 1990s, this paradigm faded from the main stage.

Of course, finding parsimonious explanations for complex phenomena lies at the core of learning-based visual understanding. Indeed, machine learning is only possible because our visual world, despite its enormous complexity, is also highly structured – visual patterns don't just happen once, but keep on repeating in various configurations. In contemporary computer vision, this structure is most often modeled via human supervision: the repeating patterns are labeled as objects or object parts, and supervised learning methods are employed to find and name them in novel imagery. However, it would seem more satisfying if complex structures could be explained in terms of simpler underlying structures.

In this chapter, we return to the classic problem of explaining objects with volumetric primitives, but using the modern tools of unsupervised learning and convolutional neural networks (CNNs). We choose the simplest possible primitives, rigidly transformed cuboids, and show how deep convolutional networks can be trained to assemble arbitrary 3D objects out of them (at some level of approximation). The main reason we succeed where the classic approaches failed is because we aim to explain the entire dataset of 3D objects jointly, allowing us to learn the common 3D patterns directly from the data.

While the representation of the 3D object shapes *e.g.* as meshes or voxel occupancies, is typically complex and high-dimensional, the resulting explanation in terms of basic primitives is parsimonious, with a small number of parameters. As examples of their applicability, we leverage the primitive based representation for

various tasks *e.g.* part discovery, image based abstraction, shape manipulation *etc.*.

5.1 Background

3D Representation and Reconstruction. The classic approaches for modeling objects and scenes dating to the very beginnings of the computer vision discipline, such as blocks world [112], generalized cylinders [13], and geons [12], emphasized the compactness of representation as the central goal. In a similar spirit, a few modern approaches have attempted to reconstruct objects/scenes using simple primitives, including Lego pieces [147] and qualitative 3D blocks [51]. Apart from these attempts, most mainstream methods for representing and reconstructing objects typically use much higher-dimensional representations *e.g.* objects as point clouds [78, 148] or exemplar CAD models [93, 94, 158]. The success of the latter set of approaches has been largely driven by the data-driven reasoning which the classical methods did not leverage. Our work aims to combine the two – we aim for a parsimonious representation but discover the underlying parsimony in a data-driven manner instead of relying on hand-crafted cues and priors. Similar to our approach, Yumer and Kara [163, 164] showed that parsimonious modelling with data-driven reasoning can allow consistent geometry simplifications or deformations in shape collections but our learning based approach allows efficient test time inference for novel shapes. An additional property of our approach, compared to classical methods, is the consistency of representation across instances. Classical approaches solve a per-instance optimization and obtain an *unordered set of primitives* whereas our our approach outputs a consistent *indexed set of primitives* – this allows several applications examined in Section 5.4.

Parsing Objects, Scenes and 3D Shapes. The idea of exploiting repeating structures in large datasets has been central to efforts on unsupervised object discovery and co-segmentation [113, 116]. Data-driven compositionality, in particular, has been used for co-segmentation [35], scene parsing and novel scene generation [72, 115]. In the domain of 3D shapes, the idea of exploiting compositionality has played a similarly important role for object representation, parsing, and manipulation. Pre-labeled, part-based shape representations were used for capturing the category-specific shape manifold [38], generating novel objects [70, 74] or recovering 3D from 2.5D data [134]. Other methods aim to automatically discover these components in 3D shape datasets [69], and their relative arrangements [167]. Similar to these shape and scene based methods, our framework can automatically discover consistent components and understand the structure of the data, but we do so by virtue of

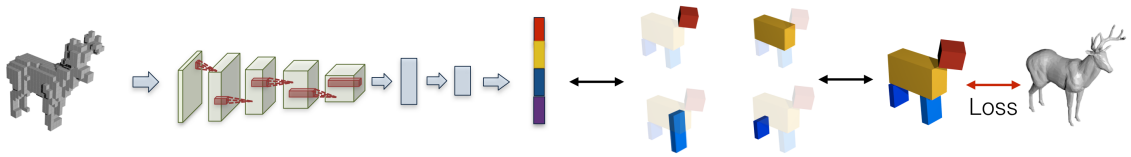

Figure 5.2: Overview of our approach. Given the input volume corresponding to an object O, we use a CNN to predict primitive shape and transformation parameters $\{(z_m, q_m, t_m)\}$ for each part (Section 5.2.1). The predicted parameters implicitly define transformed volumetric primitives $\{\bar{P}_m\}$ whose composition induces an assembled shape. We train our system using a loss function which attempts to minimize the discrepancy between the ground-truth mesh for O and the assembled shape which is implicitly defined by the predicted parameters (Section 5.2.2).

learning to generate parsimonious explanations.

Deep Generative Models. The rapid recent progress in supervised learning tasks by using deep learning techniques has been accompanied by a growing interest in leveraging similar methods to discover structure in the visual data. Using generative adversarial networks [49, 110] allows learning the data distribution but the underlying latent space lacks interpretability. Other generative methods aim to explicitly decouple the underlying factors of variation [23, 85] but rely on supervision for disentangling these factors. More closely related to our work, some recent approaches use recurrent networks to iteratively generate components to explain a simple 2D input scene [32, 50, 68]. Our work uses similar principles of learning component based explanations of complex shapes where the components are interpretable simple 3D primitives.

5.2 Learning Object Assembly

We formulate the problem of assembling a target object O, given input signal I as that of predicting (up to) M distinct parts which are then composed to output the final shape. Towards this, we learn a CNN h_θ parametrized by θ which outputs a primitive based representation. The task of learning this CNN is an unsupervised one – we do not have any annotations for the primitive parameters that best describe the target objects. However, even though there is no direct supervision, one can measure if a predicted primitive configuration is good by checking if the assembled object matches the target object. Using this insight, we formulate a loss function which informs us if the shape assembled using the predicted primitives matches the target shape and optimize this loss to train the CNN.

An overview of our approach is presented in Figure 5.2. Given a discretized representation of the target shape as input, we use a CNN to predict a primitive representation (described in Section 5.2.1). The predicted representation implicitly defines an assembled shape by composing the predicted primitives. Section 5.2.2 describes a differentiable loss function that allows using this representation in a learning framework. While the initial presentation assumes the use of a fixed number of primitives, Section 5.2.3 extends our approach to allow a variable number of primitives.

5.2.1 Primitive based Representation

We represent an assembled shape by composing the predicted simple transformed primitives. Each primitive is encoded in terms of a tuple (z, q, t) where z represents its shape in a canonical frame and (q, t) represent the spatial transformation (rotation and translation). The assembled shape predicted by the neural network h_θ can therefore be written as below.

$$\{(z_m, q_m, t_m)|m \ = \ 1, \cdots, M\} = h_\theta(I) \tag{5.1}$$

The motivation for this parametrization is to exploit the compositionality of parts as well as the independence of 'what' and 'where' (part shape and spatial transformation respectively). The representation of a shape as a set of parts allows independent reasoning regarding semantically separate units like chair legs, seat *etc.*. The decomposition in terms of part shape and transformation parameters further decomposes factors of variation like 'broad aeroplane wing' (captured by shape) and 'tilted chair back' (captured by transformation).

5.2.2 Loss Function for Assembled Shape

We want to define a differentiable loss function $L(\{(z_m, q_m, t_m)\}, O)$ between the CNN prediction $\{(z_m, q_m, t_m)\}$ and the target object O. This is a challenging task because the prediction and the groundtruth have different 3D representations – the prediction is a parametrized shape whereas the groundtruth is a mesh consisting of triangles. To overcome this, we leverage the fact that the parametrization in terms of simple primitives allows efficient computation of some properties of the shape induced by their composition. In particular, we can compute the *distance field* (Section 5.2.2) of the assembled shape as well as sample points on the surface of the primitives. These allow us to define two complimentary losses which together aim to minimize the discrepancy between the predicted and ground-truth shape. The *Coverage Loss* tries to enforce that the object O is subsumed by the predicted

assembled shape. The *Consistency Loss* enforces the other direction – that the object O subsumes the predicted shape. By optimizing these losses together, we ensure that the assembled shape tries to be maximally consistent with the target object.

Preliminaries

Notation. We represent by P_m, the *untransformed primitive* as predicted according to z_m and use $\bar{P}_m$ to denote the primitive P_m after rotation, translation according to (q_m, t_m). Therefore, the final shape induced by the composition of the predicted primitives is $\underset{m}{\cup}\bar{P}_m$.

We use the function $S(\cdot)$ to represent the surface of the argument and $p \sim S(\cdot)$ represents a random point sampled on it *e.g.* $p \sim S(\bar{P}_m)$ corresponds to a point sampled on the surface of m^{th} primitive. We also require notations for simple rigid transformations – we denote by $\mathcal{R}(p, q)$ result of rotating a point p according to rotation specified by quaternion q and similarly, $\mathcal{T}(p, t)$ denotes the result of translating a point p by t. Note that the operations $\mathcal{R}, \mathcal{T}$ are both differentiable.

Distance Field. A distance field $\mathcal{C}(\ \cdot\ ; O)$ corresponding to an object O is a function $\mathbb{R}^3 \rightarrow \mathbb{R}^+$ that computes the distance to the closest point of the object. Note that it evaluates to 0 in the object interior.

$$\mathcal{C}(p; O) = \underset{p' \in O}{min}\ \|p - p'\|_2 \tag{5.2}$$

Coverage Loss : $O \subseteq \underset{m}{\cup}\bar{P}_m$ **.**

We want to penalize the CNN prediction if the target object O is not completely covered by the predicted shape $\underset{m}{\cup}\bar{P}_m$. A sufficient condition to ensure this is that the distance field of the assembled shape evaluates to zero for all points on the surface of O.

$$L_1(\{(z_m, q_m, t_m)\}, O) = \mathbb{E}_{p \sim S(O)} \|\mathcal{C}(p; \underset{m}{\cup}\bar{P}_m)\|^2 \tag{5.3}$$

Computation can be simplified due to a nice property of distance fields. It is easy to show that the distance field of a composed shape equals to the pointwise minimum of the distance fields of all composing shapes:

$$\mathcal{C}(p; \underset{m}{\cup}\bar{P}_m) = \underset{m}{min}\ \mathcal{C}(p; \bar{P}_m) \tag{5.4}$$

This decomposition rule boils the distance field of a whole shape down to the distance field of a primitive. In the following, we show how to efficiently compute $\mathcal{C}$ for primitives as cuboids.

Distance field of Primitives. Given an origin-centred cuboid represented by $z \equiv (w, h, d)$ – its extent in the three dimensions, its distance field $\mathcal{C}_{cub}(\ \cdot\ ; z)$ can be computed as below (using $\max(0, x) \equiv x_+$):

$$\mathcal{C}_{cub}(p; z)^2 = (|p_x| - w)_+^2 + (|p_y| - h)_+^2 + (|p_z| - d)_+^2$$

Consider an object O (with an associated field $\mathcal{C}(\ \cdot\ ; O)$) undergoing a rotation R (parametrized by quaternion q) followed by a translation t. The distance field at a point p w.r.t. the transformed object is the same as the distance field at p' wrt. the canonical object where $p' = R^{-1}(p - t)$. This observations allows us to complete the formulation by defining $\mathcal{C}(p; \bar{P}_m)$ (required in Eq. 5.4) as below.

$$\mathcal{C}(p; \bar{P}_m) = \mathcal{C}(p'; P_m); p' = \mathcal{R}(\mathcal{T}(p, -t_m), \bar{q}_m) \tag{5.5}$$

$$\mathcal{C}(\ \cdot\ ; P_m) = \mathcal{C}_{cub}(\ \cdot\ ; z_m) \tag{5.6}$$

Consistency Loss : $\underset{m}{\cup}\bar{P}_m \subseteq O$.

We want to penalize the CNN prediction if the predicted shape $\underset{m}{\cup}\bar{P}_m$ is not completely inside the target object O. A sufficient condition is to ensure this is that the distance field of the object O shape evaluates to zero for all points on the surface of individual primitives $\bar{P}_m$.

$$L_2(\{(z_m, q_m, t_m)\}, O) = \sum_m \mathbb{E}_{p \sim S(\bar{P}_m)} \|\mathcal{C}(p; O)\|^2 \tag{5.7}$$

Additionally, we observe that to sample a point p on the surface of $\bar{P}_m$, one can equivalently sample p' on the surface of the untransformed primitive P_m and then rotate, translate p' according to (q_m, z_m).

$$p \sim S(\bar{P}_m) \equiv \mathcal{T}(\mathcal{R}(p', q_m), t_m); p' \sim S(P_m)$$

An aspect for computing gradients for the predicted parameters using this loss is the ability to compute derivatives for z_m given gradients for a sampled point on the canonical untransformed primitive $p' \sim S(P_m)$. We do so by using the *re-parametrization trick* [83] which decouples the parameters from the random sampling. As an example, consider a point being sampled on a rectangle extending from $(-w, -h)$ to (w, h). Instead of sampling the x-coordinate as $x \sim [-w, w]$, one can use $u \sim [-1, 1]$ and $x = uw$. This re-parametrization of sampling allows one to compute $\frac{\partial x}{\partial w}$.

5.2.3 Allowing Variable Number of Primitives

The framework we have presented so far reconstructs each instance in an object category using exactly M primitives. However, different instances in an object category can be explained by different number of primitives *e.g.* some chairs have handles, others don't. To incorporate this, in addition to predicting the shape and transformation of each primitive, we also predict the probability of its existence p_m. We first discuss the modified representation predicted by the CNN and discuss how the loss function can incorporate this.

Primitive Representation. As we mentioned above, the primitive representation has an added parameter p_m – the probability of its existence. To incorporate this, we factor the primitive shape z_m into two components – (z^s_m, z^e_m). Here z^s_m represents the primitive's dimensions (*e.g.* cuboid height, width, depth) as before and $z^e_m \sim Bern(p_m)$ is a binary variable which denotes if the primitive actually exists *i.e.* if $z^e_m = 0$ we pretend as if the m^{th} primitive does not exist. The prediction of the CNN in this scenario is as below.

$$\{(z^s_m, q_m, t_m, p_m) | m \ = \ 1 \cdots M\} = h_\theta(I) \tag{5.8}$$

$$\forall_m \ z^e_m \sim Bern(p_m); \ z_m \equiv (z^s_m, z^e_m) \tag{5.9}$$

Note that the CNN predicts p_m – the parameter of the Bernoulli distribution from which the part existence variable z^e_m is sampled. This representation allows the prediction of a variable number of parts *e.g.* if a chair is best explained using $k < M$ primitives, the network can predict a high p_m for only k primitives and a low p_m for the remaining $M - k$ primitives.

Learning. Under the reformulated representation of primitives, the CNN output does not induce a unique assembled shape – it induces a distribution of possible shapes where the m^{th} primitive stochastically exists with probability p_m. In this scenario, we want to minimize the expected loss across the possible assemblies. The first step is to modify the consistency and coverage losses to incorporate $z_m \equiv (z^s_m, z^e_m)$. Towards this, we note that the untransformed primitive P_m is either a cuboid (if $z^e_m = 1$) or empty (if $z^e_m = 0$). In case it is empty, we can simply skip it the the consistency loss (Section 5.2.2) for this primitive and can incorporate this in the coverage loss (Section 5.2.2) by modifying Eq. 5.6 as follows -

$$\mathcal{C}(\ \cdot\ ; P_m) = \begin{cases} \infty, & \text{if } z^e_m = 0 \\ \mathcal{C}_{cub}(\ \cdot\ ; z^s_m), & \text{if } z^e_m = 1 \end{cases} \tag{5.10}$$

We can now define the final loss $L(h_\theta(I), O)$ using the concepts developed. Note

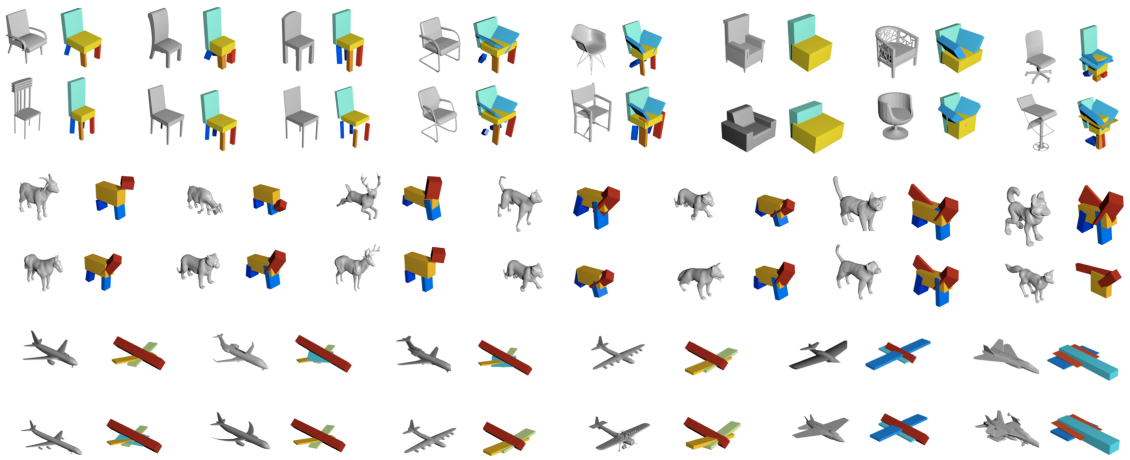

Figure 5.3: Final predictions of our method on chairs, animals and aeroplanes. We visualize the more commonly occurring modes on the left and progressively towards the right show rarer configurations predicted.

that this is simply the expected loss across possible samplings of z_m^e according to p_m.

$$L(\{(z_m, q_m, t_m)\}, O) \quad = \quad L_1(\{(z_m, q_m, t_m)\}, O) \; + \; L_2(\{(z_m, q_m, t_m)\}, O) \quad (5.11)$$

$$L(h_\theta(I), O) = \mathbb{E}_{\forall m \; z_m^e \sim Bern(p_m)} L(\{(z_m, q_m, t_m)\}, O)$$

Under this loss function, the gradients for the continuous variables *i.e.* $\{(z_m^s, q_m, t_m)\}$ can be estimated by averaging their gradients across samples. However, to compute gradients for the distribution parameter p_m, we use the REINFORCE algorithm [153] which basically gives positive feedback if the overall error is low (reward is high) and negative feedback otherwise. To further encourage parsimony, we include a small *parsimony reward* (reward for choosing fewer primitives) when computing gradients for p_m.

5.3 Experiments

Dataset. We perform our experiments primarily using the ShapeNet [20] dataset which has a large collection of 3D models. In particular, we use the 'airplane' and 'chair' object categories which have thousands of meshes available. The ShapeNet models are already aligned in a canonical frame and are of a fixed scale. Additionally, in order to demonstrate applicability beyond rigid objects, we also manually download

and similarly preprocess a set of around 100 models corresponding to four-legged animals.

Network Architecture and Training. The dataset described above gives us a set of 3D objects $\{O_i\}$. Corresponding to O_i, the input to our CNN is a discretized representation as a volumetric occupancy grid I_i of size 32∗32∗32 (we later experiment with rendered images as input in Section 5.4.3). The encoder used in our shape assembler, as shown in Figure 5.2, takes in as input an occupancy grid and passes it through 3D convolutional and fully connected layers with intermediate non-linearities to output the primitive parameters $\{(z^s_m, q_m, t_m, p_m)|m = 1 \cdots M\} \equiv h_\theta(I_i)$. In this work, we use cuboid primitives and z^s_m represents the width, height and thickness of cuboids. We use ADAM [82] to train our network according to the loss $L(h_\theta(I_i), O_i)$ described in Section 5.2 which aims to make the assembled shape predicted using I_i match to the target object O_i.

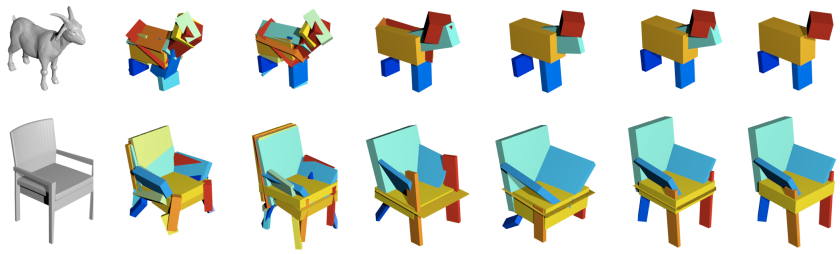

Figure 5.4: Visualization of the training progression. We visualize the prediction for two instances (shown in column 1) after every 10,000 iterations (left to right, in columns 2-6). The last column shows the result after post-processing to remove redundant parts that overlap significantly with others. The initial training stage (up to 20,000 iterations) uses all primitives but we later allow the network to learn to use fewer primitives and the predictions gradually become more parsimonious.

Implementation Details. The coverage and consistency loss functions are both defined using expectations over sampled points. In practice, we randomly sample 1000 points on $S(O)$ to implement Eq. 5.3 and 150 points from each $S(\bar{P}_m)$ to implement Eq. 5.7. To efficiently compute the distance field of the target object O at an arbitrary point p in Eq. 5.7, we precompute the distance field and its derivatives for samples in a dense regular grid and use it to obtain efficient but approximate gradients $\frac{\partial \mathcal{C}(p,O)}{\partial p}$.

Another practical difficulty is that the gradients for the primitive existence probabilities p_m are extremely noisy in the initial training stages – *e.g.* in the initial stages if a primitive is incorrectly placed, the CNN may learn to predict a very small p_m instead of learning to align the primitive correctly. To overcome this, we use a

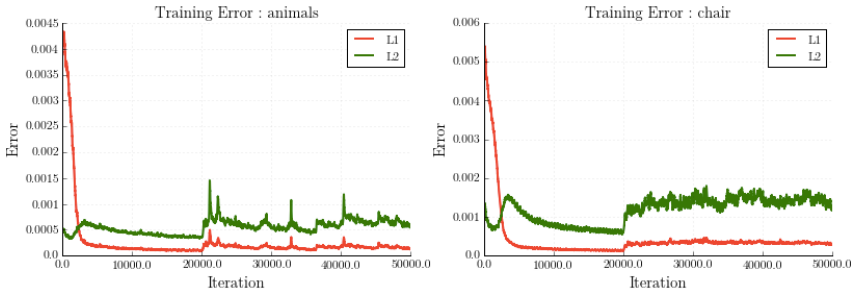

Figure 5.5: We plot the Coverage (L1) and Consistency (L2) losses over training iterations. The losses both decreases in the initial stage of training (up to 20,000 iterations) but when we allow the use of varying number of primitives along with parsimony reward, the losses initially increase. This reveals a tradeoff between representation parsimony and reconstruction accuracy.

two-stage training process. We first train the network using a fixed high value of p_m across primitives and later allow the network to also learn p_m while also encouraging simplicity by the external parsimony reward. As shown in Figure 5.5, this has the effect of first using a large number of primitives and in later stages, merging them together and using fewer primitives.

After the CNN has been trained, when computing the assembled representation for an object, we use MLE estimates instead of sampling *i.e.* $z_m^e = \mathbb{1}(p_m > 0.5)$. The final shape predictions using the CNN may still have redundant parts used and we use a simple post-processing step to refine the prediction by removing the parts which significantly overlap with others.

Results and Analysis. We show the results of our method for three object categories – chairs, aeroplanes and animals in Figure 5.3. We observe that the predictions successfully capture the coarse structure and are consistent across objects. The results indicate that the we can handle structural variations within a category *e.g.* the objects in the right side of Figure 5.3 have a different structure than those on the left which occur more commonly in the dataset.

We visualize in Figure 5.5 the training error across iterations. We observe that in the initial training stage (up to 20000 iterations), the loss rapidly decreases as the correct configuration is being learned. In the second stage of training, when we allow p_m to be learned, the error initially increases – this is because some primitives, encouraged by the parsimony reward, now start disappearing and the network eventually learns to use fewer primitives better. Even though the reconstruction error in the initial stages is lower, the reconstructions using fewer primitives, are more parsimonious. This provides an insight regarding the tradeoff between representation

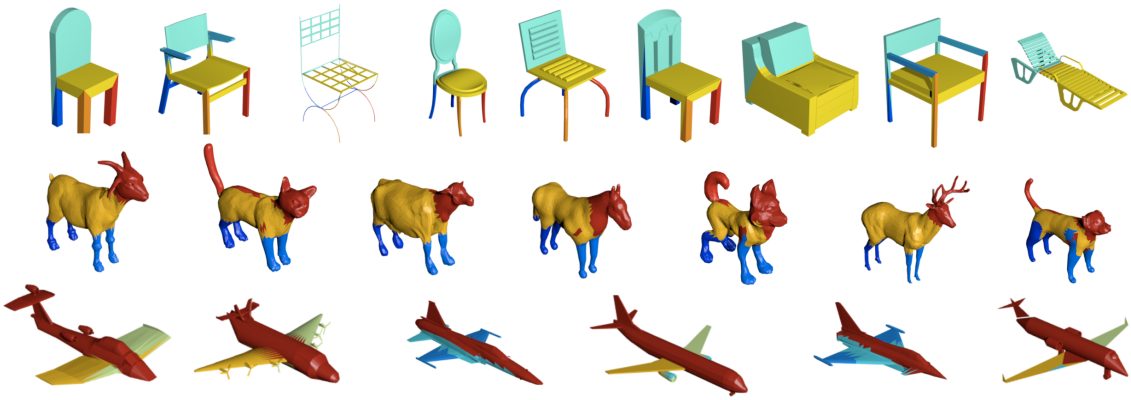

Figure 5.6: Projection of the predicted primitives onto the original shape. We assign each point p in the original shape to the corresponding primitive with lowest distance field $\mathcal{C}(p, \bar{P}_m)$. We visualize the parsing by coloring each point according to the assigned primitive. We see that similar parts *e.g.* aeroplane wings, chair seat, *etc.* are consistently colored.

parsimony and reconstruction accuracy – and that we should not judge the former by the latter.

5.4 Applications

We observe in Figure 5.1 and Figure 5.3 that the inferred representations are consistent across a category – chair seat is explained consistently using the same primitive. They are also descriptive of the underlying shape and are, by construction, interpretable. Therefore, our framework allows us to automatically discover descriptive, consistent and interpretable shape abstractions using a collection of 3D models. By virtue of these properties, our representation can enable several applications related to shape similarity, part discovery, perception and shape manipulation.

5.4.1 Unsupervised Parsing and Correspondence

The learned primitive decomposition is useful for obtaining part-level correspondences across instances. Since we use a common network across an object category, simple and consistent solutions are preferred to explain the data *i.e.* the same primitive explains the chair back across the category. We can leverage this observation to extract correspondences across the category by assigning labels to points according to the primitive that explains them – we assign each point to the primitive

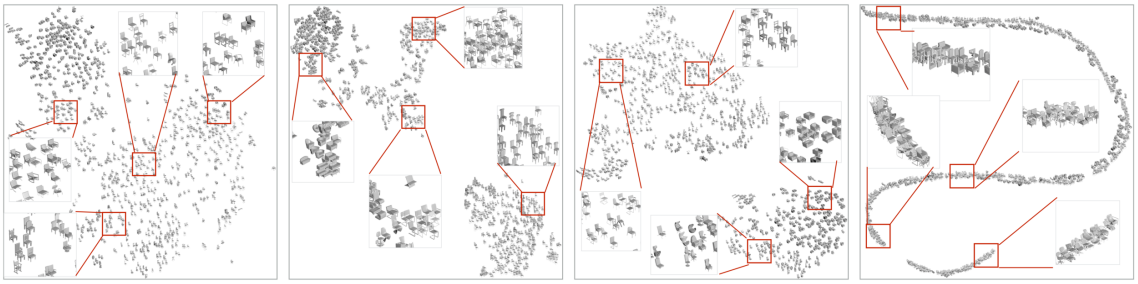

Figure 5.7: Embeddings computed using various distance measures - a) Voxel IoU based distance b) Ours (all primitives) c) Ours (chair back, seat primitives) d) Ours (chair back orientation). While the IoU based embedding conflates chairs different fine level structure (*e.g.* with/without handles), our embedding using all primitives encodes them separately. Additionally, unlike common shape representations, our inferred abstractions give us control over similarity measures – we can choose to consider only specific primitives if required *e.g.* chair back and seat which, as expected, results in ignoring existence of chair handles. We can also focus on specific properties *e.g.* chair back orientation and observe a 1D manifold emerge in this scenario.

that has the lowest $\mathcal{C}(p, \bar{P}_m)$, giving preference to larger primitives to break ties. We therefore obtain a consistent labelling of all points across instances using the predicted primitive decomposition – some examples are depicted in Figure 5.6.

We also evaluate this parsing on the Shape COSEG [152] dataset by measuring the accuracy using annotated ground-truth. While the ground-truth only has 3 clusters (chair back, seat, legs), our method as well as previous unsupervised approaches [126, 152] cluster shapes into a larger number of partitions (number of primitives in our case) and assign each partition a ground-truth label to evaluate. We obtain a mean accuracy of 89.0% whereas [126] reports 78.6% and 84.8% accuracy with initial and refined parsings respectively[1].

5.4.2 Interpretable Shape Similarity

The trained CNN of our shape assembler maps every 3D shape to corresponding primitive parameters $\{(z_m, q_m, t_m)\}$. These parameters succinctly capture the geometry of the underlying object. We find that a simple euclidean distance in the embedding space is a reliable measure of shape similarity. We use this distance

[1]Unfortunately, we found that [126] used a preliminary version of the Shape COSEG dataset [152]. We were unable to obtain this preliminary version, therefore the results are not exactly comparable. The algorithm in [152] does use the current dataset but reports no quantitative results.

to compute a t-sne [98] embedding of shapes and visualize 1000 random instances in Figure 5.7 . We observe that the automatically discovered structure captures similarity better than a simple voxel IoU based metric and that clusters correspond to natural sub-categories *e.g.* sofa *etc.*.

One aspect unique to our approach is that the shape embedding is interpretable and instead of using primitive parameters for all parts, we can modify the distance measure to focus on specifics of interest for the application. As an example, we show the resulting t-sne embedding if only 2 primitives, which correspond to back and seat, are used to compute the distance across shapes. We observe that the embedding reflects the desired similarity *e.g.* unlike in the case of using all primitives to measure shape similarity, chairs with and without handles are now embedded together. We also compute the embedding for the distance measure which only measures the difference in the orientation (q_m) for a specific part (chair back) and observe that this is a 1D manifold with the tilt increasing as we traverse it. Therefore, unlike common shape representations, our inferred abstractions give us control over similarity measures.

Figure 5.8: Inferred abstractions using real image inputs.

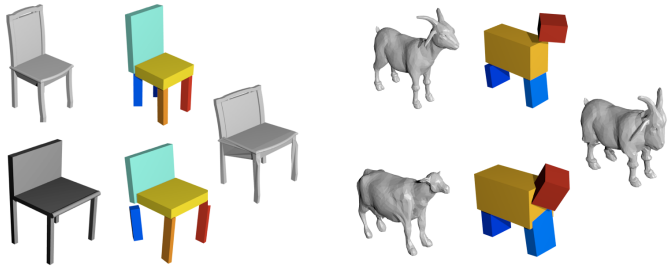

Figure 5.9: We deform the source mesh (top) to have a shape similar to the target mesh (bottom) by using the inferred primitive representation. Each source mesh point is assigned a local coordinate in the closest primitive's frame. A deformation of the primitives from the source to target configuration induces a deformed mesh (shown on right).

5.4.3 Image based Abstraction

Given our trained model h_θ which infers primitive representation using volume inputs, we can train an image based prediction model $g_{\theta'}$. We obtain volume-image pairs (V_i, I_i) by rendering ShapeNet models with random lighting and background (as suggested in [133]) and train the image based network to mimic the volume based network's predictions *i.e.* we train $g_{\theta'}$ to minimize $\|h_\theta(V_i) - g_{\theta'}(I_i)\|^2$. This distillation technique [64] for using paired data to train a model for predicting outputs similar to a pre-trained CNN is common [53] and has previously also been used for learning shape embeddings [45]. We find that we can successfully apply this to our scenario and learn an image-based prediction model that outputs the abstraction of the underlying shape given a single image. We show some results in Figure 5.8. This demonstrates that one can learn to predict shape abstractions using varying inputs and this might enable applications in robotics settings where such inference might help in grasping, planning *etc.*.

5.4.4 Shape Manipulation

The inferred primitive based shape abstractions can be used as a skeleton to guide manipulation of the underlying objects. We can assign each mesh point a local coordinate in the frame of its corresponding primitive (as computed in Section 5.4.1). A rotation, translation or scaling of the corresponding primitive can thereby induce a change in the global coordinates of the associated mesh points. We show some examples in Figure 5.9 where we deform a source mesh to have a similar configuration as a target mesh. While the transformation used in this example is defined using a target mesh, one can also use our representation for other transformation *e.g.* making the legs longer or tilting the back *etc.*

5.5 Discussion

In this chapter, we took an unsupervised, data-driven approach to explain visual information in terms of simpler primitives. Taking inspiration from the classic work on generalized cylinders [13] and geons [12], we demonstrated that we can learn to represent complex 3D structures in terms of simpler entities. However, unlike the earlier work in this area, we demonstrated the benefits of being data-driven and letting the data itself discover the best representation.

While we focused here on a relatively simple setting of isolated objects, we demonstrated the benefits of inferring the underlying compositional structure. This merely

represents an initial step towards our goal of discovering and inferring the compositional structure underlying complex scenes. It is an interesting and challenging to similarly discover, in an unsupervised manner, the notion of objects, planar surfaces *etc.* by reasoning about full scenes. Assuming these known factors, in Chapter 6 we examine the benefits of being able to infer a compositional representations for complex 3D scenes in terms of these factors.

Chapter 6

Factoring Shape, Pose and Layout

How should we represent the 3D structure of the scene in Figure 6.1? Most current methods for 3D scene understanding produce one of two representations: i) a 2.5D image of the scene such as depth [31, 122] or surface normals [7, 39]; or ii) a volumetric occupancy grid/voxels representation in terms of a single voxel grid [24, 45, 157]. Accordingly, they miss a great deal. First, all of these representations erase distinctions between objects and would represent Figure 6.1 as an undifferentiated soup of surfaces or volumes rather than a set of chairs next to a table. Moreover, the 2.5D representations intrinsically cannot say anything about the invisible portions of scenes such as the thickness of a table or presence of chair legs. While in principle voxel-based representations can answer these questions, they mix together beliefs about shape and pose and cannot account for the fact that it is easy to see that the chair has a thin back but difficult to determine its exact depth.

We demonstrated in Chapter 5 the benefits of inferring a compositional representation for objects. Motivated by similar insights, in this chapter, we present an alternative representation for 3D scenes: we should think of scenes as being composed of a distinct set of factors. One represents the *layout*, which we define as the scene surfaces that enclose the viewer, such as the walls and floor, represented in terms of their full, amodal extent *i.e.* what the scene would look like without the objects. The others represent a discrete set of objects which are in turn factored into *3D shape* (voxels) and *pose* (location and rotation). This representation solves a number of key problems: rather than a muddled mess of voxels, the scene is organized into discrete entities, permitting subsequent tasks to reason in terms of questions like "what would the scene be like if I moved that chair." In terms of reconstruction

This chapter is based on joint work with Saurabh Gupta, David Fouhey, Alexei A. Efros, and Jitendra Malik, presented primarily as it appeared in the proceedings of CVPR, 2018 [140].

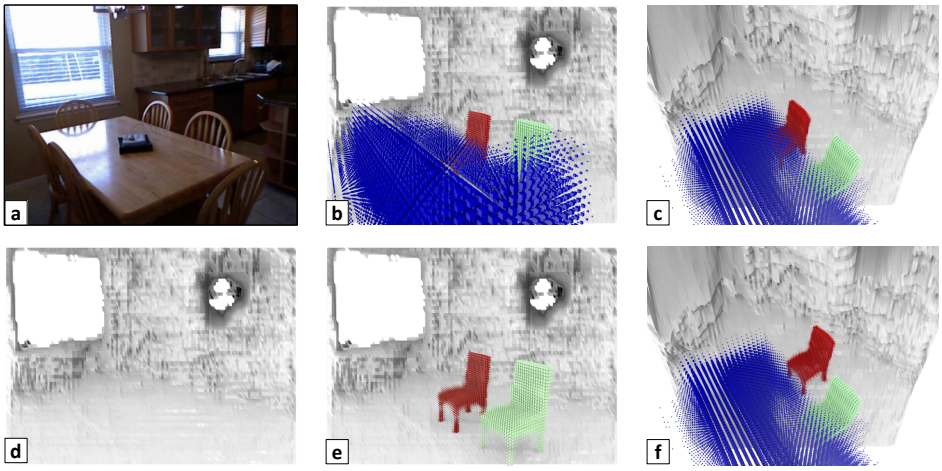

Figure 6.1: **Our 3D representation.** Given a single 2D image (a) we infer a 3D representation which is *factored* in terms of a set of objects inside an enclosed volume. We show it from the camera view in (b) and a novel view in (c). By virtue of being factored, our representation trivially enables answering questions that are impossible in other ones. For example, we can give the scene (d) without any objects; (e) without the table; or (f) answer "what would it be like if I moved the chair". These and all results best viewed in color on screen.

itself, the factored approach does not conflate uncertainties in pose and shape, and automatically allocates voxel resolution, enabling high resolution output for free.

One needs a way to infer this representation from single 2D images. We thus propose an approach in Section 6.2 which is summarized in Figure 6.2. Starting with an image and generic object proposals, we use convolutional neural networks (CNNs) to predict both the layout, *i.e.* amodal scene surfaces, as well as the underlying shape and pose of objects. We train this method using synthetic data [130], although we show it on both synthetic and natural data.

We investigate a number of aspects of our method in Section 6.3. Since our approach is the first method to tackle this task and many design decisions are non-obvious, we first present extensive ablations in Section 6.3.3. We then demonstrate that we can infer the full representation and find current performance limitations in Section 6.3.4. Next, since one might naturally wonder how the representation compares to others, we compare it to the more standard representations of a per-pixel depthmap and single voxel grid in Section 6.3.5. Figure 6.6 qualitatively and Figure 6.8 quantitatively demonstrate the benefits of our representation. We finally show qualitative results on the NYUv2 dataset.

6.1 Background

This chapter aims to take a single 2D image and factor it into a set of constituent 3D components, and thus touches on a number of topics in 3D scene understanding. This goal of recovering 3D properties from a 2D image has a rich history in computer vision starting from Robert's Blocks World [112]. In the learning-based era, this has mainly taken the form of estimating view-based per-pixel 3D properties of scenes such as depth [31, 122] or orientation information [39, 66]. These approaches are limited in the sense that they intrinsically cannot say anything about non-visible parts of the scene. This shortcoming has motivated a line of work aiming to infer volumetric reconstructions from single images [24, 45, 157], working primarily with voxels. These have been exclusively demonstrated with presegmented objects in isolation, and never with scenes: scenes pose the additional challenges of delineating objects, properly handling uncertainty in shape and pose, and scaling up resolution. Our representation automatically and naturally handles each of these challenges.

Our goal of a volumetric reconstruction of a scene has been tackled under relaxed assumptions that alleviate or eliminate the difficulty of handling either shape or pose. For example, with RGBD input, one can complete the invisible voxels from the visible ones as in [130]. Here, the problem of pose is eliminated: because of the depth sensor, one knows where the objects are, and the remaining challenge is inferring the missing shape. Similarly, in CAD retrieval scenarios, one assumes the object [6, 7, 52, 93, 94] or scene [73] can be represented in terms of a pre-determined dictionary of shapes; a great deal of earlier work [51, 91, 123] tackled this with a dictionary of box models. The challenge then is to detect these objects and figure out their pose. In contrast, we jointly infer both shape and pose. Our approach, therefore does not rely on privileged information such as the precise location of the visible pixels or is restricted to a set of pre-determined objects.

In the process of predicting our representation, we turn to tools from the object detection literature. There is, of course, a large body of work between classic 2D detection and full 3D reconstruction. For instance, researchers have predicted 3D object pose [106, 142], low-dimensional parametric shapes [37, 158], and surface normals [125]. Our representation is richer than this past work, providing detailed volumetric shape and pose, as well as the layout of the rest of the scene.

6.2 Approach

The goal of our method is to take a single 2D image and infer its 3D properties in terms of scene layout, object shape and pose. We attack this problem with two

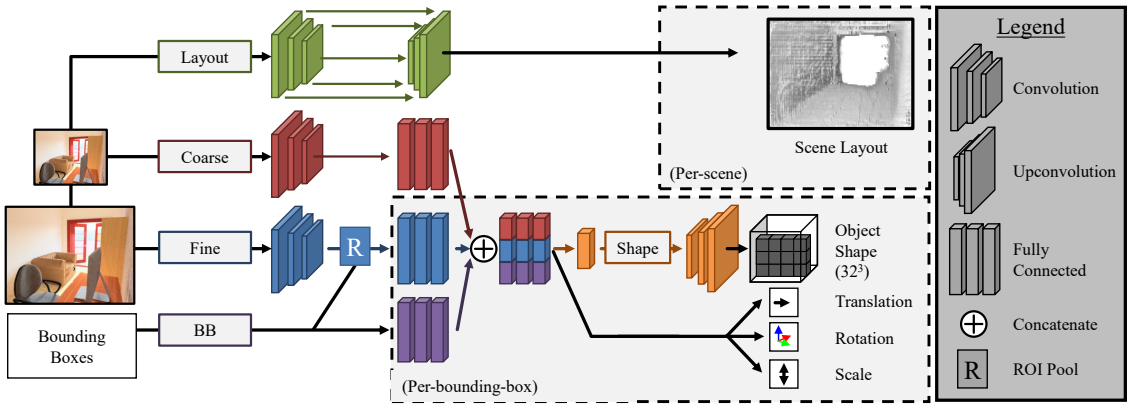

Figure 6.2: **Overview of our framework.** We take as input an image and set of bounding boxes. The scene layout **H** is predicted by the layout module, a skip-connected CNN. Each bounding box is then represented by features from three sources: ROI-pooled features extracted from a fine module that uses the original resolution, full image context features from the coarse module, and the bounding-box location. These features are concatenated and passed through several layers, culminating in the prediction of a shape code **s**, scale **c**, translation **t**, and rotation of the object **q**. The shape code is mapped to voxels **V** by the shape decoder.

main components, illustrated in Figure 6.2, that can be trained end-to-end. The first is a scene network that maps a full image to an amodal layout describing the scene minus the object. The second is an object-centric network that maps bounding boxes to the their constituent factors: shape and pose. We now describe the architectural details of each component, the loss functions learned to learn each, and the training and inference procedures.

6.2.1 Layout

We first predict the *layout*. This represents the enclosing surfaces of the scene, such as the walls and floor. Specifically, the layout is the amodal extent of these surfaces (*i.e.* the floor as it exists behind the objects of the scene). Past approaches to this [62, 123] have primarily posed this in terms of fitting a vanishing-point-aligned 3D box, which intrinsically cannot generalize to non-box environments. Here, we treat the more general case as 2.5D problem and propose to predict the layout as the disparity (*i.e.* inverse depth) map of the scene as if there were no objects.

We predict the layout using the layout module, a skip-connected network similar to [102]. The first half of the network takes the image and maps it to an intermediate representation, slowly decreasing spatial resolution and increasing increasing feature

channel count. The latter half, upconvolves in the reverse fashion while concatenating features from the encoder. We train our network end-to-end using the L_1 objective, or if $\hat{\mathbf{H}}$ denotes our prediction and $\mathbf{H}$ the ground-truth layout, $\mathrm{L}_H = ||\mathbf{H} - \hat{\mathbf{H}}||_1$.

6.2.2 Object Predictions

We represent the shape of an object as a 32^3 voxel occupancy grid $\mathbf{V}$ and the pose as anisotropic scaling $\mathbf{c}$, followed by a rotation represented by a quaternion $\mathbf{q}$, and finally a translation $\mathbf{t}$. Without any other constraints, this representation is underdetermined: one can apply many types of changes to the shape and undo them in the pose. Therefore, we represent the shape in a canonical (*i.e.* front-facing and upright) coordinate frame which is normalized and centered so the object dimensions vary between $[-0.5\mathrm{m}, 0.5\mathrm{m}]$. This in turn specifies the pose.

Architecture and features. First, we describe how the system maps an image and bounding box to this representation; the following subsection explains how this is done for a set of regions. Given a feature vector, linear layers map directly to $\mathbf{t}$, $\mathbf{q}$, and $\mathbf{c}$. Since $\mathbf{V}$ is high dimensional and structured, we first map to a shape code $\mathbf{s}$ which is then reshaped and upconvolved to $\mathbf{V}$.

We use three sources to construct our feature vector. The primary one is the fine module, which maps the image at its original resolution to convolutional feature maps, followed by ROI pooling to obtain features for the window [46]. As additional information, we also include fixed-length features from: (1) a coarse module that maps the entire image at a lower resolution through convolutional layers then vectorizes it; and (2) bounding box module mapping the bounding box location through fully connected layers. The three sources are concatenated. In the experiment section, we report experiments without the contextual features.

Shape loss. The shape of the object is a discrete volumetric grid $\mathbf{V}$, which we decode from a fixed-length shape code $\mathbf{s}$ using the shape decoder. Our final objective is a per-voxel cross-entropy loss between the prediction $\hat{\mathbf{V}}$ and ground-truth $\mathbf{V}$, or

$$\mathrm{L}_V = \frac{1}{N} \sum_n \mathbf{V}_n \log \hat{\mathbf{V}}_n + (1 - \mathbf{V}_n) \log(1 - \hat{\mathbf{V}}_n). \tag{6.1}$$

In practice, this objective is difficult to optimize, so we bootstrap the network following [45]. We learn an autoencoder on voxels whose bottleneck and decoder match our network in size. In the first stage, the network learns to mimic the autoencoder: given a window of the object, we minimize the L_2 distance between the autoencoder's bottleneck representation and the predicted shape code. The voxel decoder is then initialized with the autoencoder's decoder and the network is optimized jointly to minimize L_V.

Rotation Prediction. We parameterize rotation with a unit-normalized quaternion. We found that framing the problem as classification as in [133, 142] handled the multi-modality of the problem better. We cluster the quaternions in the training set into 24 bins and predict a probability distribution $\mathbf{k}_d$ over them. Assuming k denotes the ground-truth bin, we minimize the negative log-likelihood,

$$\mathrm{L}_q^c = -\log(\mathbf{k}_d^k). \tag{6.2}$$

The final prediction is then the most likely bin. We evaluate the impact of this choice in the experiment section and compare it with a standard squared Euclidean loss.
Scale and translation prediction. Finally, anisotropic scaling and translation are formulated as regression tasks, and we minimize the squared Euclidean loss (in log-space for scaling):

$$\mathrm{L}_t = ||\mathbf{t} - \hat{\mathbf{t}}||_2^2; \quad \text{and} \quad \mathrm{L}_c = ||\log(\mathbf{c}) - \log(\hat{\mathbf{c}})||_2^2. \tag{6.3}$$

6.2.3 Training to Predict A Full Scene

We now describe how to put these components together to predict the representation for a full scene: so far, we have described how to predict with the boxes given as opposed to the case where we do not know the boxes a-priori. At training time, we assume that we have a dataset of annotated images in which we have the box as well as corresponding 3D structure information (i.e., pose, shape, etc.).
Proposals. To handle boxes, we use an external bounding-box proposal source and predict, from the same features as those used for object prediction, a foreground probability f representing the probability that a proposal corresponds to a foreground object and optimize this with a cross-entropy loss. If $\mathcal{B}^+$ and $\mathcal{B}^-$ represent foreground and background proposals, our final objective is

$$\sum_{b \in \mathcal{B}^+} (\mathrm{L}_V + \mathrm{L}_q + \mathrm{L}_t + \mathrm{L}_r - \ln(f)) + \sum_{b \in \mathcal{B}^-} \ln(1-f),$$

which discriminates between foreground and background proposals and predicts the 3D structure corresponding to foreground proposals.

For proposals, we use 1K edge boxes [173] proposals per image. We assign proposals to ground truth objects based on modal 2D bounding box IoU. We treat proposals with more than 0.7 IoU as foreground-boxes ($\mathcal{B}^+$) and those with less than 0.3 IoU with any ground truth object as background boxes ($\mathcal{B}^-$).
Training Details. We initialize the coarse and fine convolution modules with Renset-18 [61] pretrained on ILSVRC [114] and all other modules randomly. We train the object network for 8 epochs with the autoencoder mimicking loss and then 1 additional epoch with the volumetric loss.

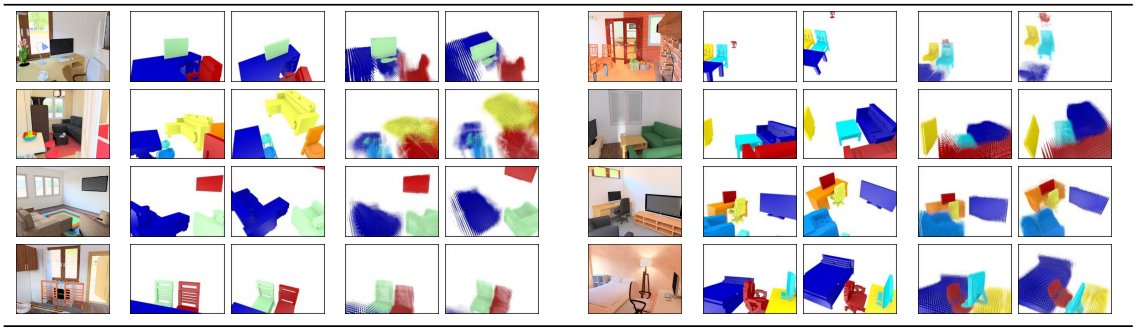

Figure 6.3: **Predicted 3D representation using ground-truth boxes**. Left: Input RGB image. Middle (2^{nd} and 3^{rd} column): Two views of ground-truth 3D configuration of the objects in the scene. The first view corresponds to the camera view and the second to a slight rotation towards the top. Right (4^{th} and 5^{th} column): The same two views of our predicted 3D structure. We visualize the predicted object shape by representing each voxel as a cube with size proportional to its occupancy probability and then transform it according to the predicted scaling, rotation and translation. The colors associate the corresponding ground-truth and predicted objects.

6.3 Experiments

We now describe the experiments done to validate our proposed representation and the described approach for predicting it. We first introduce the datasets that we use, one synthetic and one real, and the metrics used to evaluate our rich representation. Since our approach is the first to predict this representation, we begin by analyzing a number of design decisions by evaluating shape prediction in isolation. We then analyze the extent to which we can predict the representation and identify current performance bottlenecks. Having demonstrated that we can infer our representation, we compare the representation itself with alternate ones both qualitatively and quantitatively. Finally, we show some results on natural images.

6.3.1 Datasets

We use two datasets. The first is SUNCG, introduced by Song *et al.* [130]. The dataset consists of 3D models of houses created by users on an online modeling platform and has a diverse set of scenes with numerous objects and realistic clutter and therefore provides a challenging setup to test our approach. We use the physically-based renderings provided by Zhang *et al.* [166] for our experiments and randomly partition the houses into a 70%-10%-20% train, validation and test split. Overall, we obtain over 400,000 rendered training images and for each image we associate the

	Shape		Rotation		Translation		Scale	
method	%(IoU> 0.25)	IoU	%($\Delta_q < 30$)	Err	%($\Delta_t < 1$m)	Err	%($\Delta_c < 0.5$)	Err
Base	59.5%	0.31	75.2%	5.44	90.7%	0.38	85.5%	0.15
Base - context	54.4%	0.27	69.3%	7.69	85.4%	0.47	82.6%	0.19
Regression	58.4%	0.31	48.1%	31.87	88.4%	0.38	86.1%	0.14
Base + decoder ft	70.7%	0.41	74.6%	5.28	87.3%	0.42	85.1%	0.15

Table 6.1: **Performance of predictions on SUNCG with ground-truth boxes:** We report the performance of the base network (quaternion classification, use of context) and its variants. We measure the median error (or IoU) across the 3D code parameters and also report the fraction of data with performance above/error below certain thresholds. See text for details on evaluation metrics.

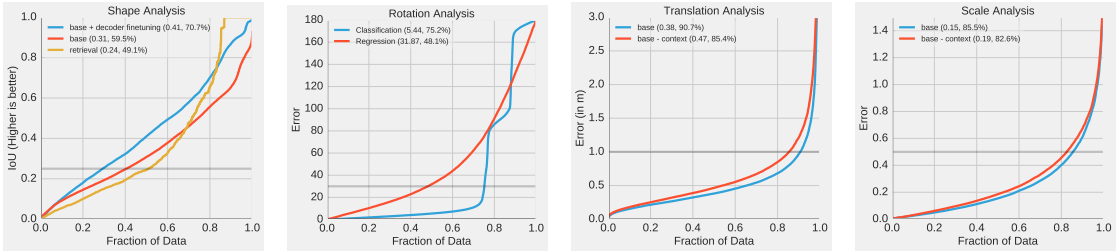

Figure 6.4: Analysis of the prediction performance across shape, rotation, translation and scale prediction. To compare alternative approaches, we plot the error (or IoU for shape) against the fraction of data up to the threshold. The plot legends also report the median value as well as fraction of data with error lower (or IoU higher) than the threshold depicted by the gray line.

visible objects with their corresponding 3D code by parsing the available house model. However, the objects present in the images are often too diverse *e.g.* fruit-baskets, ceiling lights, doors, candlesticks *etc.*. and detecting and reconstructing these is extremely challenging so we restrict the set of ground-truth boxes to only correspond to a small but diverse set of indoor object classes – bed, chair, desk, sofa, table, television. The second is NYU [127], which we use to verify qualitatively that our model is able to generalize, without additional training, to natural images.

6.3.2 Metrics

Our method and representation subsumes a number of different past works and thus there is no standard way of evaluating it. We therefore break down

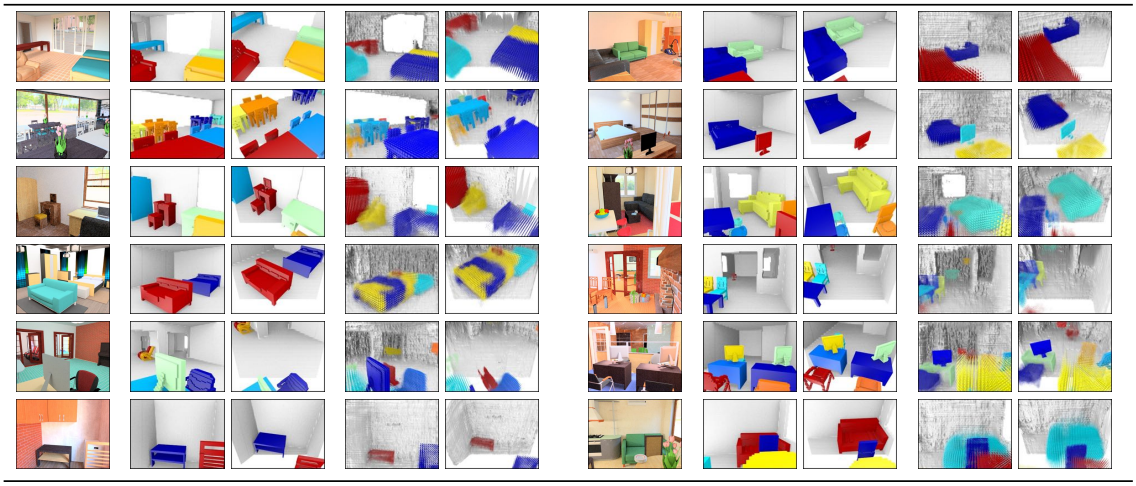

Figure 6.5: **Predicted 3D representation from an unannotated RGB image**. Left: Input image. Middle (2^{nd} and 3^{rd} column): Two views of ground-truth 3D configuration of the objects in the scene. Right (4^{th} and 5^{th} column): The corresponding two views of our predicted 3D structure. The colors only indicate a grouping of the predicted points and the coloring is uncorrelated between the prediction and the ground-truth. We observe that we can infer the 3D representations despite clutter, occlusions *etc.*.

the components of our approach and use the standard evaluation metrics for each component (*i.e.* $\mathbf{V}$, $\mathbf{q}$, $\mathbf{t}$, and $\mathbf{c}$). For each component, we define an error Δ that measures the discrepancy between the predicted value and the ground truth as well as a threshold δ that defines a true positive in the detection setting. For evaluating shape prediction in isolation, we aggregate results by taking the median over Δ and fraction of instances with distance below (or for IoU, overlap above) δ.

Shape ($\mathbf{V}$)*:* We use the standard [24] protocol and set Δ_V to measuring intersection over union (IoU) and use as threshold $\delta_V = 0.25$.

Rotation ($\mathbf{q}$)*:* We compute the geodesic distance between two rotations, or $\Delta_q(\mathbf{R}_1, \mathbf{R}_2) = (2)^{-1/2}||\log(\mathbf{R}_1^T\mathbf{R}_2)||_F$. We set $\delta_q = \frac{\pi}{6}$ following [142].

Scale ($\mathbf{c}$)*:* We define distance as the average logarithmic difference in scaling factors, or $\Delta(\mathbf{c}_1, \mathbf{c}_2) = \frac{1}{3}\sum_{i=1}^{3} |\log_2(\mathbf{c}_1^i) - \log_2(\mathbf{c}_2^i)|$. We threshold at $\delta_c = 0.5$, corresponding to being within a factor of $\sqrt{2}$.

Translation ($\mathbf{t}$)*:* We use the standard Euclidean distance $\Delta_t(\mathbf{t}_1, \mathbf{t}_2) = ||\mathbf{t}_1 - \mathbf{t}_2||$ and threshold at $\delta_t = 1$m.

2D Bounding Box ($\mathbf{b}$)*:* In the detection setting, we also consider the 2D bounding box ($\mathbf{b}$) and define Δ_b and δ_B as standard 2D IoU with the standard threshold of 0.5.

Detection Metrics. In the detection setting, we combine these metrics and define a true positive as one within/above the threshold for all of the five metrics. We use this to define average precision $\text{AP}(\delta_b, \delta_V, \delta_r, \delta_t, \delta_c)$. To better understand performance limitations, we consider variants where we relax one of these predicates (indicated by a $\cdot$).

6.3.3 Analyzing 3D Object Prediction

This is the first work that attempts to predict this representation from images and so many design decisions along the way were not obvious. We therefore study the 3D prediction model in isolation to analyze the impact of these approaches. This avoids mixing detection and shape prediction errors, which helps remove confounding factors; it is also the setting in which all other voxel prediction approaches have been evaluated historically.

Qualitative Results. We first show some predictions of the method using ground-truth boxes in Figure 6.3. Our approach is able to obtain a good interpretation of the image in terms of a scene and set of objects.

Comparisons. We report comparisons to test the importance of various components. We begin with a base model (*Base*) from which we add and remove components. This base model is trained using the losses and features described previously, but the decoder set to the autoencoder's decoder.

We first experiment with shape prediction. We add decoder fine-tuning to get (*Base + Decoder Finetuning*), which tests the effect of fine-tuning the decoder. We also try a retrieval setup (*Retrieval*); rather than use the decoder, we retrieve the nearest shape in the shape embedding space.

We then evaluate our features and losses. We try (*No Context*) in which we use *only* the ROI-pooled features; this tests whether context, in the form of bounding box coordinates and full image features, is necessary. Since our classification approach to rotation prediction may seem non-standard, we try an antipodal regression loss for estimating $\mathbf{q}$. We normalize the prediction $\hat{\mathbf{q}}$ and minimize $\min(||\mathbf{q} - \hat{\mathbf{q}}||, ||\mathbf{q} + \hat{\mathbf{q}}||)$.

Quantitative Results. We plot cumulative errors over fractions of the data in Figure 6.4 and report some summary statistics in Table 6.1. For the task of shape inference, we observe that fine-tuning the decoder improves performance. The alternate method of retrieval using a shape embedding yields some accurate retrievals, but is less robust to uncertainties, and incurs large errors for many instances. We note that as SUNCG dataset has a common set of 3D objects across all scenes, this retrieval performance can be further improved by explicitly learning to predict a model index. However, this approach would still suffer from large errors in case of

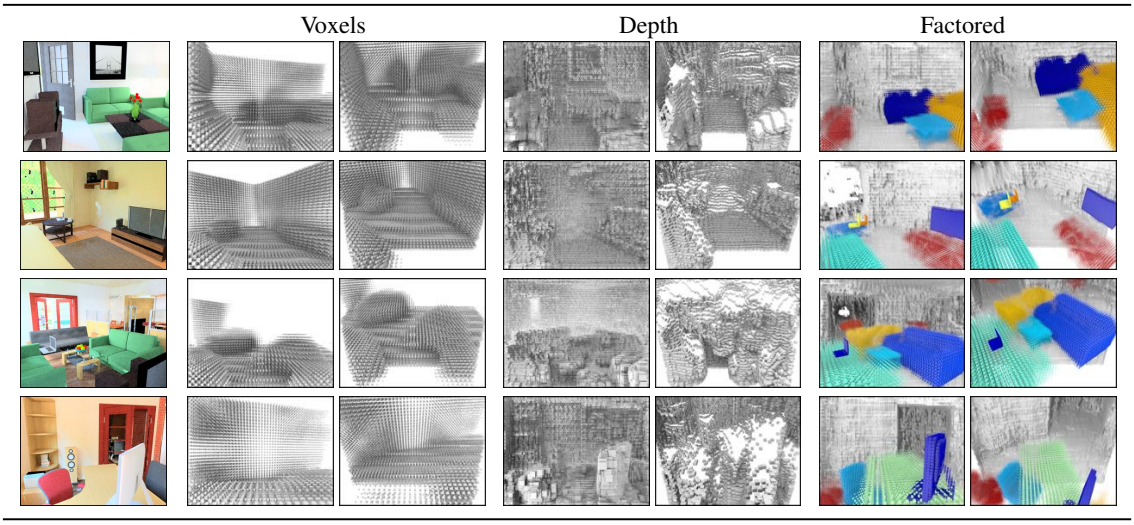

Figure 6.6: A visualization of the proposed (*Factored*) representation in comparison to (*Voxels*) a single voxel grid and (*Depth*) a depthmap. For each input image shown on the left, we show the various inferred representations from two views each: a) camera view (left), and b) a novel view (right).

incorrect retrieval, and not be generally applicable.

We observe that classification outperforms regression for predicting rotation. We hypothesize that this is because classification handles multi-modality (*e.g.* whether a chair is front- or back-facing) better. Additionally, classification has systematically different failure modes than regression: as compared to a nearly uniform set of errors from regression, the model trained with classification tends to be either very accurate or off by 90° or 180° degrees, corresponding to natural ambiguities. Finally, having context features is consistently important for each error metric, in particular for inferring absolute translation and scale, which are hard to infer from a cropped bounding box.

6.3.4 Placing Objects in Scenes

Having analyzed the factors of performance for 3D object prediction with known 2D bounding boxes, we now analyze performance on the full problem including detection.

We report in Figure 6.7 some variants of average precisions on the SUNCG test set for our approach. We obtain an average precision of 40.3% for the full 3D prediction task $AP(0.5, \frac{\pi}{6}, 1, 0.5, 0.25)$. This is particularly promising on our challenging task of making full 3D predictions in cluttered images of scenes from a single RGB image.

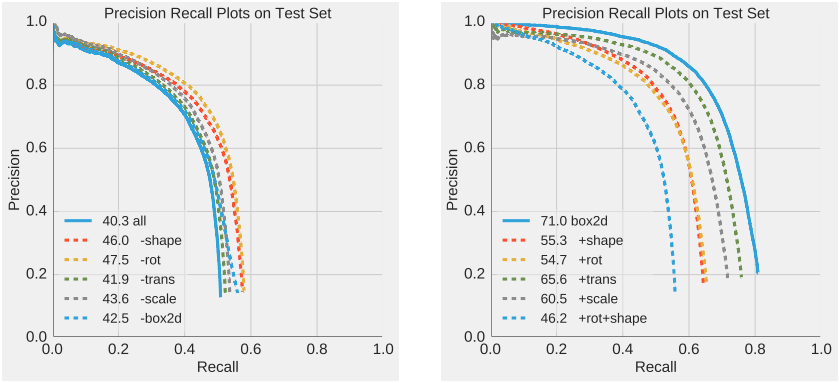

Figure 6.7: **Detection Performance on SUNCG Test Set**: We plot PR curves for our method on the SUNCG test set under different settings. Left: PR curve for full 3D prediction (denoted by 'all') and its variants when relaxing one condition at a time. Right: 2D bounding box PR curve (denoted by 'box2d') and its variants when adding one additional constraint at a time. The average precision for each setting is indicated in the legend.

We also report variants of the AP when relaxing one constraint at a time Figure 6.7 (left).

Figure 6.5 visualizes the output of our detector on some validation images from the SUNCG dataset. We show the input RGB image, ground truth and predicted objects from the current view and an additional view (obtained by rotating the camera up about a point in the scene). We observe some interesting error modes, *e.g.* duplicate detections in 3D space despite the underlying boxes not being classified as duplicates via 2D non-max suppression.

6.3.5 Comparing Scene Representations

We have proposed a new way of representing the 3D structure of scenes, and so one might ask how it compares to the alternatives in use, per-pixel depth or a single voxel grid. As has been argued throughout the chapter, our representation is qualitatively different and captures aspects that are missing in the others: as shown in Figure 6.6), voxel grids and depthmaps present an undifferentiated array of surfaces and volumes whereas ours represents a world of objects. However, we also quantitatively evaluate this. Each representation (depth, voxels, factored) is trained on different tasks. We study how well each representation solves the tasks being solved by the other representations. While each representation will perform the best at the specific task that it was trained for, a versatile representation will

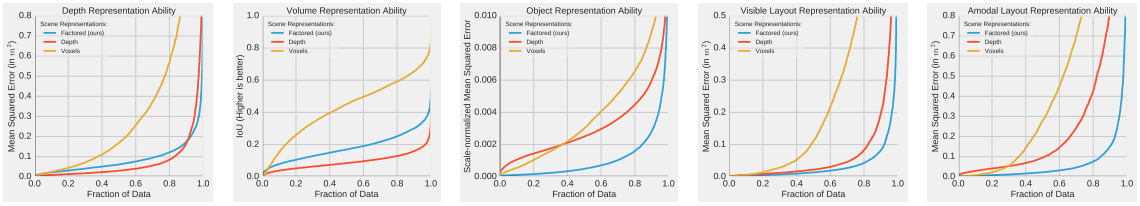

Figure 6.8: Analysis of the ability of various representations to capture different aspects of the whole scene. We compare our proposed factored representation against voxel or depth-based alternatives and evaluate their ability to capture the following aspects of the 3D scene (from left to right): a) Visible depth, b) Volumetric occupancy, c) Individual objects, d) Visible depth for scene surfaces (floor, walls *etc.*.), and e) Amodal depth for scene surfaces. See text for a detailed discussion.

also work reasonably well on the others' tasks. Our experiments below show that our factored representation is empirically more versatile than these two other popular 3D representations.

Other representations.

Per-pixel depth representations estimate the depth (or equivalently disparity *i.e.* inverse depth) for each pixel in the image. We train this representation the same way we train our layout prediction module as described in Section 6.2.1 on the SUNCG dataset, except instead of predicting the amodal disparity for scene surfaces we make predictions for all pixels in the image as would be observed from a depth sensor.

Full scene voxel representations use occupancy of voxels to represent the scene. We use $64 \times 32 \times 64$ voxels each of size $8cm \times 8cm \times 8cm$ to represent the scene. These voxels are expressed in the camera coordinate frame.

Quantitative results. We show quantitative results in Figure 6.8. We plot the cumulative distribution for various performance metrics (described below).

Explaining Visible Depth: We obtain a point cloud for the scene from each of these representations (for depth, we backproject points in space using the camera matrix, for voxels we use the point at the center of the voxel). We measure the average distance of points in the predicted point cloud to points in the ground truth point cloud obtained by back projecting the ground truth depth image.

Explaining Scene Voxels: We obtain voxel occupancy from each of the representations (depth is converted to voxel occupancy by checking if any back-projected point lies within a voxel). We measure intersection over union for the output voxel occupancy with the ground truth voxel occupancy.

Our factored representation involves two tasks: reasoning about the objects and the scene surfaces.

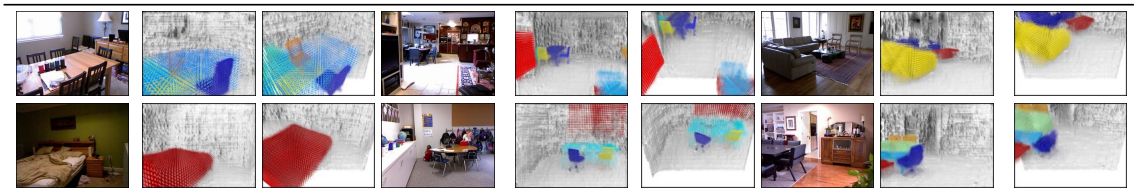

Figure 6.9: **Results on NYU dataset**. We show the results of our model trained using synthetically rendered data on real, unannotated images from the NYU dataset. Left: Input RGB image. Middle and Right: Two views of predicted 3D representation.

Explaining Objects: To measure this we align the ground truth object point clouds (obtained by sampling points at center of occupied voxels) to point clouds obtained from the three representations (in the same manner as above). We use iterative closest point to align and report the final fitness value (mean squared distance normalized with respect to the object size). We compute this at instance level for the six categories that we study.
Explaining Layout: We measure how well depth corresponding to the scene layout surfaces is explained and consider two cases: modal scene surfaces (Figure 6.8(d)) and amodal scene surfaces (Figure 6.8(e). This metric is similar to the visible depth evaluation described above except it appropriately adjusts the ground truth to focus only on layout (i.e., walls/floors/ceiling).

We observe that indeed each representation excels at the task they were specifically trained for. However our representation consistently shows much better generalization to the other tasks compared to other the other representations. Additionally, even though the visible layout can, in principle, be equally well described using a depth image, our representation works better at predicting visible layout as compared to the full image depth prediction baseline, showing the merit of factored modeling of scene composition.

6.3.6 Results on NYU

We also tested our models trained on the SUNCG dataset on images from the NYU dataset (Note we only use the RGB image to obtain these results). Figure 6.9 visualizes the output of our models on NYU Test set images. We obtain these visualizations by running our model with 2D bounding box proposals from [4]. Despite being trained on synthetic data, we are able to obtain a good interpretation of the scene.

6.4 Discussion

We argue that the representation that one should infer to understand the structure of a 3D scene should be factored in terms of a small number of components: a scene layout, and individual objects, each in turn explained in terms of its shape and pose. We presented a learning based system capable of inferring such a 3D representation from a single image. However, a number of challenges remain. In particular, we do not reason about the physics and support relationships of the predicted scenes. Additionally, we rely on synthetically rendered data with associated ground-truth for training which limits the performance on real data. However, we hope that parallel efforts in the vision community on more realistic renderings [103], leveraging weaker supervision [168], or scaling up real datasets [27] will help bridge this gap.

Chapter 7

Conclusion

In this thesis, we addressed the task of single-view 3D reconstruction. In particular, in Part I we developed learning based techniques that leveraged the notion of geometric consistency to enable 3D inference without requiring explicit 3D supervision, and then in Part II proposed frameworks to infer compositional representations.

As an initial effort to learn 3D structure via geometric consistency, Chapter 2 presented a method to learn deformable 3D models, and leverage these for inference. In Chapter 3, we proposed a differentiable ray consistency formulation that enabled us to learn a more expressive CNN based prediction model, while relying on multi-view training data. In Chapter 4, we then introduced a CNN based framework to learn prediction of textured 3D meshes using an annotated image collection. These highlight the feasibility of our proposed learning methodology – that it is possible to learn to predict 3D by enforcing consistency with the 2D supervisory data. Towards inferring compositional representations, we presented in Chapter 5 an unsupervised approach to discover the consistent compositional structure across instances of a category. In Chapter 6, we then presented an approach to similarly represent a 3D scene in terms of a small set of factors, and demonstrated the benefits of this representation.

While these are encouraging steps towards the goal of scalable, robust and accurate 3D reconstruction, a number of challenges still remain. We discuss some of these below and highlight interesting future directions.

Enforcing Geometric Consistency during Inference. In Part I, we relied on geometric consistency to provide supervisory signal during training. However, during inference, our learned CNN based prediction models are used to simply perform feedforward prediction to yield the output reconstruction. Therefore, the predicted 3D reconstructions as not constrained (or even encouraged) to be photometrically or geometrically consistent with the input image. Some recent approaches [155] propose

mechanisms to rectify this in certain scenarios, but incorporating this consistency across different representations and 2D observations remains a challenge.

Physically Consistent Scene Representation. We proposed in Part II a mechanism to infer compositional 3D representations. While this approach allowed us to predict the independent entities that comprised the scene, we did not ensure that the resulting composed structure is physically plausible. As an example, two inferred chairs might be intersecting each other, or a laptop floating above a table. It would be an interesting direction to incorporate such physical constraints in the inferred structure, perhaps by additionally modeling relationships across entities.

Learning Multi-view Reconstruction. The goal of of this thesis was to reconstruct objects and scenes given a single input image. However, there are oten scenarios where a small number of additional views might be available for inference. Such a setup is a natural generalization of the single-view reconstruction task, and would be ideal to explore approaches that integrate learning based inference with classical geometric reconstruction methods. Some recent methods [77] propose novel solutions to utilize the information across multiple views in a geometrically motivated manner, but require known camera poses for inference, and the more general setting remains a largely unaddressed problem.

Learning 3D Reconstruction by and for Interaction. A motivation for several techniques and setups in the thesis was to enable learning 3D reconstruction in an ecologically plausible manner. While the proposed solutions allowed us to learn using previously challenging supervision setups, we primarily relied on supervisory data collected via internet images. It would be interesting to apply these ideas to settings where the supervisory data comes from an actual robotic agent actively exploring and interacting with the world. Further, the ability to infer and reason with these structured 3D representations might also be beneficial for such agents in their goals *e.g.* learning to grasp [99].

Bibliography

[1] https://www.ebay.com/.

[2] E. H. Adelson and A. P. Pentland. The perception of shading and reflectance. *Perception as Bayesian inference*, 1996.

[3] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. SCAPE: Shape Completion and Animation of PEople. *SIGGRAPH*, 2005.

[4] P. Arbeláez, J. Pont-Tuset, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*, 2014.

[5] N. Aspert, D. Santa-Cruz, and T. Ebrahimi. Mesh: Measuring errors between surfaces using the hausdorff distance. In *ICME*, 2002.

[6] M. Aubry, D. Maturana, A. A. Efros, B. C. Russell, and J. Sivic. Seeing 3D chairs: exemplar part-based 2D-3D alignment using a large dataset of CAD models. In *CVPR*, 2014.

[7] A. Bansal, B. Russell, and A. Gupta. Marr revisited: 2D-3D alignment via surface normal prediction. In *CVPR*, 2016.

[8] J. T. Barron and J. Malik. Color constancy, intrinsic images, and shape estimation. *ECCV*, 2012.

[9] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. *PAMI*, 2015.

[10] H. Barrow and J. Tenenbaum. Interpreting line drawings as three-dimensional surfaces. *Artificial Intelligence*, 1981.

[11] A. Bartoli, V. Gay-Bellile, U. Castellani, J. Peyras, S. Olsen, and P. Sayd. Coarse-to-fine low-rank structure-from-motion. In *CVPR*, 2008.

[12] I. Biederman. Recognition-by-components: a theory of human image understanding. *Psychological review*, 1987.

[13] T. O. Binford. Visual perception by computer. In *IEEE Conference on Systems and Control*, 1971.

[14] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, 1999.

[15] V. Blanz and T. Vetter. Face recognition based on fitting a 3d morphable model. *TPAMI*, 2003.

[16] L. Bourdev, S. Maji, T. Brox, and J. Malik. Detecting people using mutually consistent poselet activations. In *ECCV*, 2010.

[17] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. In *CVPR*, 2000.

[18] A. Broadhurst, T. W. Drummond, and R. Cipolla. A probabilistic framework for space carving. In *ICCV*, 2001.

[19] T. J. Cashman and A. W. Fitzgibbon. What shape are dolphins? building 3d morphable models from 2d images. *TPAMI*, 2013.

[20] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012 [cs.GR], 2015.

[21] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *TPAMI*, 2017.

[22] Y. Chen, T.-K. Kim, and R. Cipolla. Inferring 3d shapes and deformations from single views. In *ECCV*, 2010.

[23] B. Cheung, J. A. Livezey, A. K. Bansal, and B. A. Olshausen. Discovering hidden factors of variation in deep networks. *arXiv preprint arXiv:1412.6583*, 2014.

[24] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, 2016.

[25] T. F. Cootes and C. J. Taylor. Active shape modelsâĂŤâĂŸsmart snakesâĂŹ. In *BMVC*, 1992.

[26] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[27] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *CVPR*, 2017.

[28] J. De Bonet and P. Viola. Roxels: Responsibility weighted 3d volume reconstruction. In *ICCV*, 1999.

[29] A. Dürer. *Four Books on Human Proportion*. Formschneyder, 1528.

[30] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015.

[31] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014.

[32] S. A. Eslami, N. Heess, T. Weber, Y. Tassa, D. Szepesvari, G. E. Hinton, et al. Attend, infer, repeat: Fast scene understanding with generative models. In *NIPS*, 2016.

[33] C. H. Esteban and F. Schmitt. Silhouette and stereo fusion for 3d object modeling. *CVIU*, 2004.

[34] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 2010.

[35] A. Faktor and M. Irani. Co-segmentation by composition. In *ICCV*, 2013.

[36] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, 2017.

[37] S. Fidler, S. Dickinson, and R. Urtasun. 3D object detection and viewpoint estimation with a deformable 3D cuboid model. In *NIPS*, 2012.

[38] N. Fish, M. Averkiou, O. van Kaick, O. Sorkine-Hornung, D. Cohen-Or, and N. J. Mitra. Meta-representation of shape families. *Transactions on Graphics (SIGGRAPH)*, 2014.

[39] D. F. Fouhey, A. Gupta, and M. Hebert. Data-driven 3D primitives for single image understanding. In *ICCV*, 2013.

[40] D. F. Fouhey, W. Hussain, A. Gupta, and M. Hebert. Single image 3D without a single 3D image. In *ICCV*, 2015.

[41] R. Garg and I. Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *ECCV*, 2016.

[42] R. Garg, A. Roussos, and L. Agapito. Dense variational reconstruction of non-rigid surfaces from monocular video. In *CVPR*, 2013.

[43] P. Gargallo, P. Sturm, and S. Pujades. An occupancy–depth generative model of multi-view images. In *ACCV*, 2007.

[44] J. J. Gibson. The ecological approach to visual perception. 1979.

[45] R. Girdhar, D. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *ECCV*, 2016.

[46] R. Girshick. Fast R-CNN. In *ICCV*, 2015.

[47] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[48] C. Godard, O. Mac Aodha, and G. J. Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, 2017.

[49] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

[50] K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

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

[52] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3D models to RGB-D images of cluttered scenes. In *CVPR*, 2015.

[53] S. Gupta, J. Hoffman, and J. Malik. Cross modal distillation for supervision transfer. In *CVPR*, June 2016.

[54] A. Guzman. Computer recognition of three-dimensional objects in a visual scene. *PhD thesis, MIT AI-Lab.*, 1968.

[55] J. Gwak, C. B. Choy, A. Garg, M. Chandraker, and S. Savarese. Weakly supervised 3d reconstruction with adversarial constraint. In *3DV*, 2017.

[56] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. In *3DV*, 2017.

[57] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *ICCV*, 2011.

[58] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *ECCV*. Springer, 2014.

[59] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. In *CVPR*, 2015.

[60] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *ICCV*, 2017.

[61] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[62] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, 2009.

[63] M. Hejrati and D. Ramanan. Analyzing 3d objects in cluttered images. In *NIPS*, 2012.

[64] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[65] D. Hoiem, A. A. Efros, and M. Hebert. Automatic photo pop-up. *ACM Transactions on Graphics (TOG)*, 2005.

[66] D. Hoiem, A. A. Efros, and M. Hebert. Geometric context from a single image. In *CVPR*, 2005.

[67] B. Horn. Shape from shading: A method for obtaining the shape of a smooth opaque object from one view. PhD thesis, Massachusetts Inst. of Technology, 1970.

[68] J. Huang and K. Murphy. Efficient inference in occlusion-aware generative models of images. *arXiv preprint arXiv:1511.06362*, 2015.

[69] Q. Huang, V. Koltun, and L. Guibas. Joint shape segmentation with linear programming. In *ACM Transactions on Graphics (TOG)*. ACM, 2011.

[70] Q. Huang, H. Wang, and V. Koltun. Single-view reconstruction via joint analysis of image and shape collections. *ACM Transactions on Graphics (TOG)*, 2015.

[71] J. F. Hughes and J. D. Foley. *Computer graphics: principles and practice*. Pearson Education, 2014.

[72] P. Isola and C. Liu. Scene collaging: Analysis and synthesis of natural images with semantic layers. In *ICCV*, 2013.

[73] H. Izadinia, Q. Shan, and S. M. Seitz. IM2CAD. In *CVPR*, 2017.

[74] E. Kalogerakis, S. Chaudhuri, D. Koller, and V. Koltun. A probabilistic model for component-based shape synthesis. *ACM Transactions on Graphics (TOG)*, 31(4):55, 2012.

[75] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018.

[76] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. *arXiv preprint arXiv:1803.07549*, 2018.

[77] A. Kar, C. Häne, and J. Malik. Learning a multi-view stereo machine. In *NIPS*, 2017.

[78] A. Kar, S. Tulsiani, J. Carreira, and J. Malik. Category-specific object reconstruction from a single image. In *CVPR*, 2015.

[79] K. Karsch, Z. Liao, J. Rock, J. T. Barron, and D. Hoiem. Boundary cues for 3d object shape recovery. In *CVPR*, 2013.

[80] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *CVPR*, 2018.

[81] S. Khamis, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon. Learning an efficient model of hand shape variation from depth images. In *CVPR*, 2015.

[82] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[83] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[84] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[85] T. D. Kulkarni, W. F. Whitney, P. Kohli, and J. Tenenbaum. Deep convolutional inverse graphics network. In *Advances in Neural Information Processing Systems*, pages 2539–2547, 2015.

[86] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. In *ECCV*, 2014.

[87] A. Kurenkov, J. Ji, A. Garg, V. Mehta, J. Gwak, C. Choy, and S. Savarese. Deformnet: Free-form deformation network for 3d shape reconstruction from a single image. *arXiv preprint arXiv:1708.04672*, 2017.

[88] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. *IJCV*, 2000.

[89] S. Laine, T. Karras, T. Aila, A. Herva, S. Saito, R. Yu, H. Li, and J. Lehtinen. Production-level facial performance capture using deep convolutional neural networks. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, page 10. ACM, 2017.

[90] A. Laurentini. The visual hull concept for silhouette-based image understanding. *TPAMI*, 1994.

[91] D. C. Lee, A. Gupta, M. Hebert, and T. Kanade. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *NIPS*, 2010.

[92] K. Li, B. Hariharan, and J. Malik. Iterative instance segmentation. In *CVPR*, 2016.

[93] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. J. Guibas. Joint embeddings of shapes and images via cnn image purification. *TOG*, 2015.

[94] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. In *ICCV*, 2013.

[95] S. Liu and D. B. Cooper. Ray markov random fields for image-based 3d modeling: model and efficient inference. In *CVPR*, 2010.

[96] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 2015.

[97] D. G. Lowe. Three-dimensional object recognition from single two-dimensional images. *Artificial intelligence*, 1987.

[98] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *JMLR*, 2008.

[99] J. Mahler, F. T. Pokorny, B. Hou, M. Roderick, M. Laskey, M. Aubry, K. Kohlhoff, T. Kröger, J. Kuffner, and K. Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *ICRA*, 2016.

[100] D. Marr. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* Henry Holt and Co., Inc., 1982.

[101] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan. Image-based visual hulls. In *SIGGRAPH*, 2000.

[102] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016.

[103] J. McCormac, A. Handa, S. Leutenegger, and A. J.Davison. Scenenet RGB-D: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation? In *ICCV*, 2017.

[104] C. Nandakumar, A. Torralba, and J. Malik. How little do we need for 3-d shape perception? *Perception-London*, 2011.

[105] R. Nevatia and T. O. Binford. Description and recognition of curved objects. *AI*, 1977.

[106] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis. 6-dof object pose from semantic keypoints. In *ICRA*, 2017.

[107] B. Pepik, M. Stark, P. Gehler, T. Ritschel, and B. Schiele. 3d object class detection in the wild. In *Workshop on 3D from a Single Image (3DSI) (in conjunction with CVPR'15)*, 2015.

[108] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental mathematics*, 2(1):15–36, 1993.

[109] M. Prasad, A. Fitzgibbon, A. Zisserman, and L. Van Gool. Finding nemo: Deformable object class modelling using curve matching. In *CVPR*, 2010.

[110] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[111] D. J. Rezende, S. A. Eslami, S. Mohamed, P. Battaglia, M. Jaderberg, and N. Heess. Unsupervised learning of 3d structure from images. In *NIPS*, 2016.

[112] L. G. Roberts. *Machine Perception of Three-Dimensional Solids*. PhD thesis, MIT, 1963.

[113] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *CVPR*, 2013.

[114] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[115] B. Russell, A. Efros, J. Sivic, B. Freeman, and A. Zisserman. Segmenting scenes by matching image composites. In *NIPS*, 2009.

[116] B. C. Russell, A. A. Efros, J. Sivic, W. T. Freeman, and A. Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006.

[117] Y. Sahillioğlu and Y. Yemez. A surface deformation framework for 3d shape recovery. In *International Workshop on Multimedia Content Representation, Classification and Security*, 2006.

[118] S. Saito, L. Wei, L. Hu, K. Nagano, and H. Li. Photorealistic facial texture inference using deep neural networks. In *CVPR*, 2017.

[119] S. Satkin, M. Rashid, J. Lin, and M. Hebert. 3dnn: 3d nearest neighbor. *IJCV*, 2014.

[120] N. Savinov, C. Hane, L. Ladicky, and M. Pollefeys. Semantic 3d reconstruction with continuous regularization and ray potentials using a visibility consistency constraint. In *CVPR*, 2016.

[121] N. Savinov, C. Häne, M. Pollefeys, et al. Discrete optimization of ray potentials for semantic 3d reconstruction. In *CVPR*, 2015.

[122] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3D scene structure from a single still image. *TPAMI*, 2009.

[123] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box In the Box: Joint 3D Layout and Object Reasoning from Single Images. In *ICCV*, 2013.

[124] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 2016.

[125] A. Shrivastava and A. Gupta. Building part-based object detectors via 3D geometry. In *ICCV*, 2013.

[126] O. Sidi, O. van Kaick, Y. Kleiman, H. Zhang, and D. Cohen-Or. Unsupervised co-segmentation of a set of shapes via descriptor-space spectral clustering. *ACM Trans. on Graphics (Proc. SIGGRAPH Asia)*, 2011.

[127] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from RGBD images. In *ECCV*, 2012.

[128] P. Sinha and E. Adelson. Recovering reflectance and illumination in a world of painted polyhedra. In *ICCV*, 1993.

[129] H. O. Song, Y. Xiang, S. Jegelka, and S. Savarese. Deep metric learning via lifted structured feature embedding. In *CVPR*, 2016.

[130] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. In *CVPR*, 2017.

[131] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*. ACM, 2004.

[132] H. Su, Q. Huang, N. J. Mitra, Y. Li, and L. Guibas. Estimating image depth using shape collections. *ACM Transactions on Graphics (TOG)*, 33, 2014.

[133] H. Su, C. R. Qi, Y. Li, and L. J. Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *ICCV*, 2015.

[134] M. Sung, V. G. Kim, R. Angst, and L. Guibas. Data-driven structural priors for shape completion. *ACM Transactions on Graphics (TOG)*, 2015.

[135] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *ICCV*, 2017.

[136] J. Taylor, R. Stebbing, V. Ramakrishna, C. Keskin, J. Shotton, S. Izadi, A. Hertzmann, and A. Fitzgibbon. User-specific hand modeling from monocular depth sequences. In *CVPR*, 2014.

[137] D. Thompson. *On Growth and Form*. Cambridge Univ. Press, 1917.

[138] L. Torresani, A. Hertzmann, and C. Bregler. Non-rigid structure-from-motion: Estimating shape and motion with hierarchical priors. *TPAMI*, 2008.

[139] S. Tulsiani, A. A. Efros, and J. Malik. Multi-view consistency as supervisory signal for learning shape and pose prediction. In *CVPR*, 2018.

[140] S. Tulsiani, S. Gupta, D. Fouhey, A. A. Efros, and J. Malik. Factoring shape, pose, and layout from the 2d image of a 3d scene. In *CVPR*, 2018.

[141] S. Tulsiani, A. Kar, J. Carreira, and J. Malik. Learning category-specific deformable 3d models for object reconstruction. *TPAMI*, 2017.

[142] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *CVPR*, 2015.

[143] S. Tulsiani, H. Su, L. J. Guibas, A. A. Efros, and J. Malik. Learning shape abstractions by assembling volumetric primitives. In *CVPR*, 2017.

[144] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *CVPR*, 2017.

[145] N. R. Twarog, M. F. Tappen, and E. H. Adelson. Playing with puffball: simple scale-invariant inflation for use in vision and graphics. In *ACM Symposium on Applied Perception*, 2012.

[146] A. O. Ulusoy, A. Geiger, and M. J. Black. Towards probabilistic volumetric reconstruction using ray potentials. In *3DV*, 2015.

[147] A. van den Hengel, C. Russell, A. Dick, J. Bastian, D. Pooley, L. Fleming, and L. Agapito. Part-based modelling of compound scenes from images. In *CVPR*, 2015.

[148] S. Vicente, J. Carreira, L. Agapito, and J. Batista. Reconstructing pascal voc. In *CVPR*, 2014.

[149] S. Vicente and L. de Agapito. Balloon shapes: Reconstructing and deforming objects with volume from images. In *3DV*, 2013.

[150] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

[151] X. Wang, D. Fouhey, and A. Gupta. Designing deep networks for surface normal estimation. In *CVPR*, 2015.

[152] Y. Wang, S. Asafi, O. van Kaick, H. Zhang, D. Cohen-Or, and B. Chen. Active co-analysis of a set of shapes. *ACM Transactions on Graphics (TOG)*, 2012.

[153] R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 1992.

[154] O. J. Woodford and G. Vogiatzis. A generative model for online depth fusion. In *ECCV*, 2012.

[155] J. Wu, Y. Wang, T. Xue, X. Sun, W. T. Freeman, and J. B. Tenenbaum. MarrNet: 3D Shape Reconstruction via 2.5D Sketches. In *NIPS*, 2017.

[156] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman. Single image 3d interpreter network. In *ECCV*, 2016.

[157] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In *NIPS*, 2016.

[158] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *WACV*, 2014.

[159] J. Xiao, B. Russell, and A. Torralba. Localizing 3D cuboids in single-view images. In *Advances in neural information processing systems*, pages 746–754, 2012.

[160] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee. Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision. In *NIPS*, 2016.

[161] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011.

[162] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

[163] M. E. Yumer and L. B. Kara. Co-abstraction of shape collections. *ACM Transactions on Graphics (TOG)*, 2012.

[164] M. E. Yumer and L. B. Kara. Co-constrained handles for deformation in shape collections. *ACM Transactions on Graphics (TOG)*, 2014.

[165] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang. The unreasonable effectiveness of deep networks as a perceptual metric. In *CVPR*, 2018.

[166] Y. Zhang, S. Song, E. Yumer, M. Savva, J.-Y. Lee, H. Jin, and T. Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. *CVPR*, 2017.

[167] Y. Zheng, D. Cohen-Or, M. Averkiou, and N. J. Mitra. Recurring part arrangements in shape collections. *Computer Graphics Forum (Eurographics 2014)*, 2014.

[168] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.

[169] T. Zhou, S. Tulsiani, W. Sun, J. Malik, and A. A. Efros. View synthesis by appearance flow. In *ECCV*, 2016.

[170] R. Zhu, H. Kiani, C. Wang, and S. Lucey. Rethinking reprojection: Closing the loop for pose-aware shape reconstruction from a single image. In *ICCV*, 2017.

[171] S. Zhu, L. Zhang, and B. Smith. Model evolution: An incremental approach to non-rigid structure from motion. In *CVPR*, 2010.

[172] M. Z. Zia, M. Stark, B. Schiele, and K. Schindler. Detailed 3D representations for object recognition and modeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(11):2608–2623, 2013.

[173] C. L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014.

[174] S. Zuffi, A. Kanazawa, D. Jacobs, and M. J. Black. 3d menagerie: Modeling the 3d shape and pose of animals. In *CVPR*, 2017.