A JOINT FRAMEWORK FOR OBJECT RECOGNITION: CATEGORIZATION, INSTANCE RECOGNITION AND POSE ESTIMATION

BY TAREK EL-GAALY

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Computer Science

Written under the direction of

Ahmed Elgammal

and approved by

_____

_____

_____

_____

New Brunswick, New Jersey

January, 2016

ABSTRACT OF THE DISSERTATION

A JOINT FRAMEWORK FOR OBJECT RECOGNITION: CATEGORIZATION, INSTANCE RECOGNITION AND POSE ESTIMATION

by TAREK EL-GAALY

Dissertation Director: Ahmed Elgammal

Visual object recognition is a challenging problem with a wide range of real-life applications. The difficulty of this problem is due to variation in shape and appearance among objects within the same category, as well as varying viewing conditions, such as viewpoint, scale, illumination, occlusion and articulation of multi-part deformable objects. In addition, beyond the visual spectrum, depth and range sensors suffer from noise that inhibits object recognition. Under visual object recognition lie three subproblems that are each challenging: category recognition, instance recognition and pose estimation. Impressive work has been done in the last decade on developing systems for generic object recognition. Previous research has covered many recognition-related issues, however, the problem of multi-view recognition remains among the most fundamental challenges in computer vision. In this dissertation we focus on discovering low-dimensional latent representations that enable efficient joint multi-view object recognition over multiple modalities. These discovered latent representations allow us to work in lower dimensional latent spaces that capture the factors needed for object recognition from multi-view images and over multiple modalities; from images to depthmaps and 3D point clouds. Each of the models we present in this dissertation explore a different representation space of latent

factors. The first model builds multiple kernel induced spaces to fuse information between different modalities and performs object pose estimation in a regression framework. The second model performs manifold analysis to solve categorization and pose estimation simultaneously. It does this by factorizing the space of topological mappings between a unified conceptual manifold and feature spaces. We present two variations of this; an unsupervised learning model and a supervised learning model. The third approach analyzes the representational spaces of the layers of Convolutional Neural Networks and builds on the findings by proposing a network that jointly solves category and pose. The fourth approach explores solving pose-invariant categorization of multi-part objects by shape information, in the form of 3D point clouds. We build a representation that inherently encodes pose and allows objects to be represented by multiple levels of object-part decompositions for more robust object recognition. In each approach we support our hypotheses by extensive experimentation.

Acknowledgements

I would like to thank my advisor Ahmed Elgammal for the support throughout my Ph.D. struggles. I would like to thank the committee members who over-saw my dissertation. I would like to thank my family, friends and lab-mates for helping me through this stage of my life. It was an exceedingly valuable and joyous experience and it would not have been so if it was not for all of you.

Dedication

To my family and to myself...nothing is impossible...

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introductory Remarks

The layout of this dissertation takes the form of a *dissertation-by-publication*. We refer to the problem of simultaneously solving for object category and pose as *joint object recognition*. In all the work of this dissertation no color information from the images was used for categorization and pose estimation, unless otherwise noted. The object-view manifold refers to the low-dimensional intrinsic manifold that capture the multiple views of an object. In this dissertation we commonly refer to it as just the manifold.

Each chapter in this dissertation digs deep into analyzing and discovering representation spaces that capture the latent factors needed to efficiently solve the problem of multi-view joint object recognition over multiple modalities, from images to depthmaps and point clouds. Chapter 3 solves object pose regression by using multiple kernel induced spaces to fuse visual and depth features of multi-view images. We perform two variations: multi-kernel learning and multi-kernel regression. Chapters 4 and 5 focus on a framework to jointly solve categorization and pose estimation using multi-view images. This work uses a manifold analysis approach to factorize the space of topological mappings between a unified conceptual manifold and the input feature space. In its simplest form, the conceptual manifold takes the form of a unit circle in the case of a viewing circle around an object, e.g. when the camera is fixed and an object is turning on a turn-table. The space of topological mappings is factorized into two factors: the first we call *style* which captures the deformation in appearance and shape of objects (and can then be used to represent object instances) and the second is the object pose. A generalization of the unit circle to a full viewing sphere of more degrees of freedom is described in Chapter 4. We present two branches of this manifold analysis approach: unsupervised linear decomposition (Chapter 4) and supervised nonlinear decomposition (Chapter 5). These approaches are also shown to fuse both visual and depth modes for more accurate recognition. Chapter 6 looks into using a more recent Machine Learning paradigm, known as Deep Learning, to solve both categorization and pose estimation simultaneously. The representation spaces of layers of a Convolutional Neural Network are analyzed from the perspective of encoding information pertaining to object category and pose. From the findings of the analysis we propose a new network that solves for both category and pose. The fourth and final chapter explores building

representations from 3D point clouds for categorization that are inherently pose-invariant. This final work uses a probabilistic approach to decompose 3D point clouds of multi-part objects into parts and thus paves the way for 3D Recognition-by-Parts robust to pose variation and articulation.

1.2 Problem Statement

Visual object recognition is a challenging problem with ample real-life applications. The difficulty of this problem is due to variations in shape and appearance among objects within the same category, as well as varying viewing conditions, such as viewpoint, scale, illumination, occlusion in cluttered environments and articulation in multi-part objects. In addition, beyond the visual spectrum, depth sensors suffer from noise that inhibit the ability to perform robust object recognition.

Under this perceptual problem of visual recognition lie three subproblems that are each quite challenging: category recognition, instance recognition, and pose estimation. Impressive work have been done in the last decade on developing computer vision systems for generic object recognition. Research has spanned a wide spectrum of recognition-related issues, however, the problem of multi-view recognition remains one of the most fundamental challenges to the progress of computer vision.

It is essential that approaches discover and use lower dimensional latent representations to solve this challenging problem of joint multi-view object recognition. Latent representations here dictate the appearance and shape of objects over different categories, object instances and poses. In addition latent representations should be able to capture and enable fusion over multiple modalities. Models need to be able to discover these lower dimensional latent representation spaces and utilize them for more efficiently solving joint object recognition.

The problems of object classification from multi-view setting (multi-view recognition) and pose estimation are closely linked and directly impacted by the way shape is represented. Inspired by Marr's 3D object-centric doctrine [92], traditional 3D pose estimation algorithms often solved the recognition, detection, and pose estimation problems simultaneously (*e.g.*[51, 79, 87, 126]), through 3D object representations, or through invariants. However, such models

were limited in their ability to capture large within-class variability, and were mainly focused on recognizing instances of objects. In the last two decades the field has shifted to study 2D representations based on local features and parts, which encode the geometry loosely (*e.g.* pictorial structure like methods [44, 43]), or does not encode the geometry at all (*e.g.* bag of words methods [144, 129].) Encoding the geometry and the constraints imposed by objects' 3D structure are essential for pose estimation. Most research on generic object recognition bundle all viewpoints of a category into one representation; or learn view-specific classifiers from limited viewpoints, *e.g.* frontal cars, side-view cars, rear cars, etc. Recently, there has been an increasing interest in object categorization in the multi-view setting, as well as recovering object pose in 3D, *e.g.* [21, 75, 119, 86, 130, 132, 103, 95]. However, the representations used in these approaches are mainly category-specific representations, which do not support scaling up to a large number of categories.

More recently Deep Learning has emerged as an end-to-end Machine Learning paradigm that by-passes explicit hand-engineered features, and learns feature hierarchies within layered models that best achieve a certain task. Despite the success of these methods in object recognition, not much success has been achieved in pose estimation. Deep Learning models capable of pose estimation have been built for estimating human pose in images but this is a slightly different problem to recovering the pose of rigid objects. The work by [140] explored using CNNs for recognizing human pose. Other notable works in human pose estimation are [84, 105]. Recently [49] proposed joint optimization on human pose and activity. Object pose estimation is fundamentally different from human posture estimation. In human pose estimation, there is no problem getting millions of image of people at different postures. The human activities are correlated with human poses, while in the object-pose domain the category is independent of pose. That makes the joint learning of category and pose more challenging than joint learning of human activity and pose.

In addition to multi-view image data, a proliferation of depth sensors in the field of Computer Vision and Robotics provide depthmaps and 3D point clouds (*e.g.* Xbox Kinect [148]). These sensor modes provide shape of objects and are a very important additional cue to augment vision for object recognition. In particular shape is important to disambiguate and recognize multi-part objects (*e.g.* animals, furniture and vehicles). An example of this is an airplane that

can be decomposed into parts: fuselage, wings and tail. Many real-world objects possess *parts* and it is the structure of these parts that dictate object pose and the category of the object. We refer to this as *Recognition-by-Parts* (similar to Recognition-by-Components in [11]). Many research questions are raised here, for example: How can we efficiently represent shape to account for both the category of multi-part objects and at the same time encode their pose? Humans are able to seamlessly achieve this task. Are there ways of mimicking this human ability? Object parts suffer from ambiguity in the sense that there are multiple levels of granularity at which objects can be divided up into their constituent parts. How can we model these multiple levels of granularity? How can we use these levels to achieve more robust object recognition?

1.3 Objectives

The first objective of this dissertation is to solve the three sub-problems of object recognition simultaneously and robustly in a general framework that is not class-specific. To reiterate, the three sub-problems are: object categorization, instance recognition (can be seen as a more fine-grained object categorization) and pose estimation.

The second objective is to perform the first objective over multiple modalities of perception: images, depthmaps and 3D point clouds, and to fuse between the modes where possible (*e.g.* fuse registered grayscale images and depthmaps from the Kinect sensor [148]).

An implicit objective of this dissertation is to analyze and discover latent representation spaces that allow efficient solving of joint object recognition across the multiple modalities. Each of the presented computational models explores latent representation across the different modalities and thus enables them to solve the task of joint object recognition.

1.4 Contributions

We investigate the use of multi-modal fusion for object pose estimation. In Chapter 3, we address the challenging problem of pose recognition using simultaneous color and depth information. For this purpose, we extend a state-of-the-art regression framework by using a multi-kernel approach to incorporate depth information to perform more effective pose recognition on table-top objects. We do extensive experiments on a large publicly available dataset to

validate our approach. We show significant performance improvements (more than 20%) over published results.

The premise of the work in Chapter 4 is that multi-view images of the same object lie on intrinsic low-dimensional manifolds in descriptor spaces (e.g. visual/depth descriptor spaces). These object manifolds share the same topology despite being geometrically different. Each object manifold can be represented as a deformed version of a unified manifold. The object manifolds can thus be parameterized by its homeomorphic mapping/reconstruction from the unified manifold. In Chapter 4 we describe a novel framework to jointly solve the three challenging recognition sub-problems, by explicitly modeling the deformations of object manifolds and factorizing it in a view-invariant space for recognition. We perform extensive experiments on several challenging datasets and achieve state-of-the-art results.

Thirdly, in Chapter 5, we further explore using manifold analysis of the common topology of multi-view images of objects. We utilize this common topology between object manifolds by learning a low-dimensional latent space which non-linearly maps between a common unified manifold and the object manifold in the input space. Using a supervised embedding approach, the latent space is computed and used to jointly infer the category and pose of objects. We empirically validate our model by using multiple inference approaches and testing on multiple challenging datasets. We compare our results with the state-of-the-art and present our increased category recognition and pose estimation accuracy.

Chapter 6 is focused on studying the view-manifold structure in the feature spaces implied by the different layers of Convolutional Neural Networks (CNN). There are several questions that this work aims to answer: Does the learned CNN representation achieve viewpoint invariance? How does it achieve viewpoint invariance? Is it achieved by collapsing the view manifolds, or separating them while preserving them? At which layer is view invariance achieved? How can the structure of the view manifold at each layer of a deep convolutional neural network be quantified experimentally? How does fine-tuning of a pre-trained CNN on a multi-view dataset affect the representation at each layer of the network? In order to answer these questions we propose a methodology to quantify the deformation and degeneracy of view manifolds in CNN layers. We apply this methodology and report interesting results in this work that answer the aforementioned questions.

Chapter 7 presents a probabilistic approach to shape decomposition that creates a skeleton-based shape representation of a 3D object while simultaneously decomposing it into constituent parts. Our approach probabilistically combines two prominent threads from the shape literature: skeleton-based (medial axis) representations of shape, and part-based representations of shape, in which shapes are combinations of primitive parts. Our approach recasts skeleton-based shape representation as a mixture estimation problem, allowing us to apply probabilistic estimation techniques to the problem of 3D shape decomposition, extending earlier work on the 2D case. The estimated 3D shape decompositions approximate human shape decomposition judgments. We present a tractable implementation of the framework, which begins by over-segmenting objects at concavities, and then probabilistically merges them to create a distribution over possible decompositions. This results in a hierarchy of decompositions at different structural scales, again closely matching known properties of human shape representation. The probabilistic estimation procedures that arise naturally in the model allow effective prediction of missing parts. We present results on shapes from a standard database illustrating the effectiveness of the approach.

1.5 Acknowledgements

The research work in Chapter 3 was conducted in collaboration with Marwan Torki and Maneesh Singh and resulted in the following conference publication: T. El-Gaaly, M. Torki, A. Elgammal, M. Singh, RGBD Object Pose Recognition using Local-Global Multi-Kernel Regression, International Conference on Pattern Recognition (ICPR) 2012.

The research in Chapter 4 was conducted in collaboration with Haopeng Zhang and Zhiguo Jiang. This research resulted in the following conference and journal publication: H. Zhang, T. El-Gaaly, A. Elgammal and Z. Jiang, Joint Object and Pose Recognition using Homeomorphic Manifold Analysis, Association for the Advancement of Artificial Intelligence (AAAI) 2013 and H. Zhang, T. El-Gaaly, A. Elgammal and Z. Jiang, Factorization on View-Object Manifold for Joint Object Recognition and Pose Estimation, ElSevier Journal of Computer Vision and Image Understanding (CVIU) 2015: Special Issue on Shape Representation Meet Visual Recognition.

The research presented in Chapter 5 was conducted in collaboration with Amr Bakry and Mohamed Elhoseiny. The work is to be published in IEEE Winter Conference on Applications of Computer Vision (WACV) 2016.

The research work in Chapter 6 was conducted in collaboration with Mohamed Elhoseiny and Amr Bakry and currently submitted for publication.

The research described in Chapter 7 was conducted in collaboration with Vicky Froyen, Manish Singh and Jacob Feldman and resulted in the following publication: T. El-Gaaly et. al., A Bayesian Approach to Perceptual 3D Object-Part Decomposition using Skeleton-based Representations, Association for the Advancement of Artificial Intelligence (AAAI) 2015.

Chapter 2

Background and Related Work

We divide up this section into three parts:

- A review of previous approaches to solving object recognition, particularly the approaches concerned with joint categorization, instance recognition and pose estimation.
- Focused review on work done in Deep Learning to solve joint object recognition.
- A review of 3D representations for object recognition that also inherently capture pose information in order to be robust to viewpoint changes and articulation.

2.1 Joint Object Recognition

2.1.1 Recognition and Pose Estimation

Traditional 3D pose estimation algorithms often solve the recognition and pose estimation problems simultaneously using 3D object model-bases, hypothesis and test principles, or through the use of invariants, e.g. [51, 79, 87, 126]. Such models are incapable of dealing with large within-class variability and have been mainly focused on recognizing instances previously seen in the model-base. This limitation led to the development, over the last decade, of very successful categorization methods mainly based on local features and parts. Such methods loosely encode the geometry, e.g. methods like pictorial structure [44]; or does not encode the geometry at all, e.g. bag of words [144, 129].

There is a growing recent interest in developing representations that captures 3D geometric constraints in a flexible way to handle the categorization problem. The work of Savarese and Fei-Fei [119, 120] was pioneering in that direction. In [119, 120] a part-based model was proposed where canonical parts are learned across different views, and a graph representation is used to model the object canonical parts. Successful recent work have proposed learning category-specific detection models that is able to estimate object pose (e.g. [95, 103, 121, 104]). This has an adverse side-effect of not being scalable to a large number of categories and dealing with high within-class variation. Typically papers on this area focus on evaluating the detection and pose estimation performance and do not evaluate the categorization performance. In contrast to category-specific representations, we focus on developing a common representation for

recognition and pose estimation. This achieved through learning a view-invariant representation using a proposed three-phase process that can use images and videos in a realistic learning scenario.

Almost all the work on pose estimation and multi-view recognition from local features is based on formulating the problem as a classification problem where view-based classifiers and/or viewpoint classifiers are trained. These *classification*-based approaches solve pose estimation problem in a discrete way simultaneously or not with recognition problem. They use several discrete (4, 8, 16 or more) view-based/pose-based classifiers, and take the classification results as the estimated poses. For example, in [53], 93 support vector machine (SVM) classifiers were trained. It is obvious that only discrete poses can be obtained by these classification-based methods, and the accuracy depends on the number of classifiers. On the other hand, there are also works formulate the problem of pose estimation as a regression problem by learning the regression function within a specific category, such as car or head, e.g. [4, 139, 40, 33, 102]. These *regression*-based approaches solve pose estimation in a continuous way, and can provide continuous pose prediction. A previous comparable study in [53] shows that the regression method (i.e. support vector regression, SVR) performs well in either horizontal or vertical head pose variations comparing to SVM classifiers. More recent regression-based approaches [139, 33] also report better pose estimation results than classification-based methods on some challenging datasets. Generally, pose estimation is essentially a continuous problem, since the pose varies continuously in real world. Thus, continuously estimating the poses is more conformable to the essence of the problem.

In the domain of multi-modal data, recent work by [78] uses synchronized multi-modal photometric and depth information (i.e. RGB-D) to achieve significant performance in object recognition. They build an object-pose tree model from RGBD images and perform hierarchical inference. Although performance of category and instance recognition is significant, object pose recognition performance is less so. The reason is the same: a classification strategy for pose recognition results in coarse pose estimates and does not fully utilize the information present in the continuous distribution of descriptor spaces. In the work by [40, 39], random regression forests were used for real time head pose estimation from depth images, and such a continuous pose estimation method can get 3D orientation errors less than 10° respectively.

2.1.2 Modeling Visual Manifolds for Recognition

Learning image manifolds has been shown to be useful in recognition, for example for learning appearance manifolds from different views [97], learning activity and pose manifolds for activity recognition and tracking [34, 141], etc. The seminal work of Murase and Nayar [97] showed how linear dimensionality reduction using PCA [66] can be used to establish a representation of an object's view and illumination manifolds. Using such representation, recognition of a query instance can be achieved by searching for the closest manifold. However, such a model is mainly a projection of the data to a low-dimensional space and does not provide a way to untangle the visual manifold. The pioneering work of Tenenbaum and Freeman [136] formulated the separation of style and content using a bilinear model framework. In that work, a bilinear model was used to decompose face appearance into two factors: head pose and different people as style and content interchangeably. They presented a computational framework for model fitting using SVD. A bilinear model is a special case of a more general multilinear model. In [142], multilinear tensor analysis was used to decompose face images into orthogonal factors controlling the appearance of the face including geometry (people), expressions, head pose, and illumination using High Order Singular Value Decomposition (HOSVD) [25]. N-mode analysis of higher-order tensors was originally proposed and developed in [67, 91] and others. A fundamental limitation with bilinear and multilinear models is that they need an aligned product space of data (all objects $\times$ all views $\times$ all illumination etc.).

The proposed framework utilizes bilinear and multilinear analysis. However, we use such type of analysis in a different way that avoids their inherent limitation. The content manifold, which is the view manifold in our case, is explicitly represented using an embedded representation, capitalizing in the knowledge of its dimensionality and topology. Given such representation, the style parameters are factorized in the space of nonlinear mapping functions between a representation of the content manifold and the observations. The main advantage of this approach is that, unlike bilinear and multilinear models that mainly discretize the content space, the content in our case is treated as a continuous domain, and therefore aligning of data is not needed.

The introduction of nonlinear dimensionality reduction techniques such as Local Linear

Embedding (LLE) [115], Isometric Feature Mapping (Isomap) [135], and others [8, 17, 81, 143], provide tools to represent complex manifolds in low-dimensional embedding spaces, in ways that aim at preserving the manifold geometry. However, in practice, away from toy examples, it is hardly the case that various orthogonal perceptual aspects can be shown to correspond to certain directions or clusters in the embedding space. In the context of generic object recognition, direct dimensionality reduction of visual features was not shown to provide an effective solution; to the contrary, the state of the art is dominated by approaches that rely on extremely high-dimensional feature spaces to achieve class linear separability, and the use of discriminative classifier, typically SVM, in these spaces. By learning the visual manifold, we are not advocating a direct dimensionality reduction solution that just preserves the manifold geometry locally and/or globally. We are arguing for a solution that is able to factorize and untangle the complex visual manifold to achieve multi-view recognition.

A comprehensive review of recent work on object recognition can be seen in [120]. We will highlight the most relevant research in this section.

Successful work has been done in estimating the object pose of a single object [23, 95, 121]. This model, referred to as single-instance 3D model, has the limitation of being category-specific and does not scale to a large number of categories and deal with high intra-class variation. This stipulates that independent models are learnt for different object. This has an adverse side-effect of not being scalable to a large number of categories and dealing with high intra-class variation. Work in this area focus on evaluating the pose estimation performance and do not evaluate how the model can be used for categorization.

Recently, category recognition and pose estimation have been solved simultaneously (*e.g.* [119, 78]). In [119], multiple-view object modeling is addressed by linking diagnostic parts of the objects from different (discrete) viewing points. This model is extendable to different instances of previously trained objects. It still belongs to the category of limited-pose (discrete-pose) object recognition since it uses a classification approach to deal with pose estimation. This may not be extendable to general continuous pose estimation. Very few works formulate the problem of pose estimation as a regression problem over a continuous space. In [78], an object pose tree is built for doing multi-level inference (object category, instance and pose recognition). This work involves a classification strategy for pose recognition which results in coarse

pose estimates and does not fully utilize the information present in the continuous distribution of descriptor spaces. Work presented in [119] and [139] explicitly model the continuous pose variations of objects but the scalability of these models is limited. A more recent work [7] proposes a feedforward approach to solve the two problems jointly by balancing between continuous and discrete modeling of pose in order to increase performance and scalability. In these models, the nonlinearity in the category representations is not modeled, which is mandatory for many applications.

Solving object recognition and pose estimation using a manifold-based representation is not novel. The pioneering work of Murase and Nayar explored this idea in [97]. However, the recognition in [97] is for object instances and not for generic categories. The model used in [97] is a linear model based on PCA, while our model is nonlinear in both the pose and the category. Manifold based representations has also been recently used in [95], however for object-specific view estimation.

2.2 Deep Learning Approaches for Joint Object Recognition

LeCun *et al.* has widely used CNNs for various vision tasks [123, 70, 64, 110, 82]. The success of CNNs can be partially attributed to these efforts, in addition to training techniques that have been adopted. Krizhevsky *et al.* [74] used a CNN in the ImageNet Challenge 2012 and achieved state-of-the-art accuracy. Since then, there have been many variations in CNN architectures and learning techniques within different application contexts. In this section we mainly emphasize related works that focused on bringing an understanding of the representation learned at the different layers of CNNs and related architectures. This understanding will enable us to leverage CNNs to perform joint object recognition.

Yobinski *et al.* [150] studied how CNN layers transition from general to specific. An important finding in this study is that learning can be transferred, and by using fine-tuning, performance is boosted on novel data. Other transfer learning examples include [112, 29, 2]. Zeiler *et al.* [151] investigated the properties of CNN layers for the purpose of capturing object information. This study is built on the premise that there is no coherent understanding of why

CNNs work well or how we can improve them. Interesting visualizations were used to explore the functions of layers and the intrinsics of categorization. The study stated that CNN output layers are invariant to translation and scale but not to rotations. The study in [19] evaluated different deep architectures and compares between them. The effect of the output-layer dimensionality was explored.

Due to the surge of work in deep architectures over the last few years, there has amassed a large number of research studies. Despite this, using CNNs for regression and capturing pose information is still a relatively unexplored area. This further motivates the goals of some of the work in this dissertation in exploring the capability of CNNs to capture distributed pose representations and utilize them for pose estimation within deep architectures.

We focus on the most relevant work, in particular, CNN studies that focus on understanding the functions and impact of CNN layers and CNNs that solve for pose information.

The work by [140] explored using CNNs for recognizing human pose. Other notable works in human pose estimation are [84, 105]. Recently [49] proposed joint optimization on human pose and activity. Object pose estimation is fundamentally different from human posture estimation. In human pose estimation, there is no problem getting millions of image of people at different postures. The human activities are correlated with human poses, while in the object-pose domain the category is independent of pose. That makes the joint learning of category and pose more challenging than joint learning of human activity and pose.

2.3 Medial Axis Representations for Object Recognition

Throughout history, man has been trying to understand the nature of shape, representations of it and our inherent visual and cognitive perception of it. It hasn't been until modern times that scientists have attempted to scientifically establish what it means for humans to perceive shape. In the latter half of the last century, scientists have focused on trying to understand and mimic human perception of both 2D and 3D shapes.

Understanding the visual representation of shape is a difficult problem due to the infinite dimensions of object geometry. Starting in the mid 1900s, much effort was channeled towards

discovering low-dimensional *meaningful* representations for shape. A *meaningful* representation requires that it should be intuitively capable of summarizing shape information and support tasks such as shape/object recognition and matching. These are key tasks in Computer Vision, Computer Graphics and Robotics applications.

Many researchers explored the description of shape by the holistic structure of object parts and described each *primitive* part geometrically, *e.g.* geons [11] and geometric cylinders [93, 92]. Others developed feature descriptors to describe surface localities of shapes, *e.g.* spin-images [65] and 3D shape context [76]. A very large body of work has focused on *skeletons* - more generally referred to as *Medial Representations*. The topic of Medial Representations is caught between multiple disciplines: Computer Science, Robotics, Psychology, Neuroscience and Perceptual Science. In this chapter we will focus on Medial Representations and their importance in understanding shape.

2.3.1 The Rise of Medial Representations

In 1967, Harry Blum published his work on representing 2D shapes by 1D skeletons. He called this representation the Medial Axis or Medial Axis Transform (MAT) [13]. He based this work on the insight that shape contours are intuitively captured by a representation that extracts local symmetries of the contour boundary. Evidence supporting medial representations can be found in psychophysics and neuroscience [28]. Children have no problem at all in recognizing shapes represented by line-drawings without prior experience [28].

The Medial Axis has been found to be similar to many other approaches and has several analogous definitions. For this reason Medial Axis is also sometimes referred to as topological skeletons, symmetry axis/sets, grassfire propagation and central sets. All these names refer to similar medial representations of shape that capture both the interior, local symmetry and boundary of shapes by reducing them to 1D skeleton segments, that evidently summarize the ”parts” of an object at various levels of abstraction. The representation is formed by computing the loci of centers of *maximal discs* contained within the shape. Blum also extended this work to representing object parts and 3D shapes by representing 3D objects using surface skeletons (*i.e. sheets*), seams and junctions joining the two and then building a graph of the object parts. This was done in a later paper, published in 1973 [14]. The work by Blum has sparked numerous

medial and graph representations of shapes over the years.

The recent book by Sidiqqi et. al. [127] goes through much of the literature since Blum's seminal paper. Sidiqqi et. al. divides shape understanding approaches into the following categories: landmark representations, boundary representations, displacement by voxel representations and medial representations. Landmark representations are similar to work done in the Computer Vision community where each object is represented by a collection of distinctive visual/geometric features. Boundary representations represent objects/shapes by their boundaries in the form of meshes, functional decomposition...etc. These can be thought of as *exo-skeleton* representations but do not have the same simplicity as medial representations and do not intuitively represent the symmetries in shapes. Displacement by voxel representations represent shape by the displacement from an atlas of the space, much like *morphing* into the shape. This representation also has trouble describing shapes in terms of symmetry and in simple terms. By far the most popular shape representation is the medial representation stemming from Blum's work. There have been numerous models and extensions to this. Examples of these are Shock graphs [90], Reeb graphs [10], Flux graphs [28], Object Cores [18], Multiscale Medial Axis [106], Bone graphs [89] and a Bayesian generative form of medial representation [41, 47]. All these representations are similar in the essential step of reducing the dimension of the shape representation to a series of curve segments making up the skeleton of the shape. The final representation in these methods usually takes the form of a graph built from the skeleton curve segments. For example, the popular Shock Graphs describes a skeleton as a Directed Acylic Graph (DAG) using a scheme of combining adjacent equal types of *shocks* (*i.e.* necks, protrusions, bends, seeds (junctions)), where *shocks* are different types of skeleton segments.

Problems with *ligatures*, noise, missing shape information due to occlusion and ambiguity in the final representation haunt many of these models because they inherit the basic limitations of Blum's approach [28, 122]. Shock Graphs are affected by noise in the object boundary, occlusions, the choice of root node and they do not maintain planar order of nodes [122]. Improvements have been made by either improvements in building the skeleton, smoothing over the object's shape prior to fitting the skeleton or smoothing the skeletal representation after fitting. A discussion of these methods appears in [96].

There is no doubt that medial representations are essential in understanding shape and have

been widely used across disciplines. In addition to psychological and perceptual evidence to support the medial representation, we have compiled the main advantages of this elegant and intuitive representations as follows:

- Medial representations offer an intuitive representation of the structure of shape (including articulated objects) and as a result represent the topology of shapes in a natural way.
- They describe internal local symmetry as well as global shape structure. Symmetry is a key feature of shape that humans perceive [145] and skeletons are naturally very adept at representing this feature.
- Skeletons or medial representations allow part decomposition of shapes and hence object/shape matching through the use of graph matching. Graphs built from skeletons represent the global structure of objects' shape and are thus useful for object recognition/matching [122].
- Invariance to rotation (to some degree - experiments on view-based robustness can be found in [28, 122]), translation, scale and articulation.
- They offer compact powerful representations in reduced dimension (1D curves or 2D sheets in some cases for 3D shapes).

2.3.2 3D Medial-Axis: An Inherently Pose-Invariant Representation

Much of the research in 2D shape understanding was also done theoretically on 3D shapes at the same time because these two problems come hand in hand and the reality of the world lies in 3D. Blum's 1973 work also explores representations of 3D shapes. Other examples of work exploring 3D shape understanding can be found in [60, 32, 61, 93, 92, 107, 28] and a good integrative study of the subject can be found in [145], although many of these approaches focus on decomposition of shape into primitive parts and not explicitly medial representations.

In the 3D realm, there has been a lot of work in the Computer Graphics community on geometrically fitting skeletons to 3D object data in the form of meshes or unorganized point clouds. The aim in the computer graphics community is to use these models to perform shape modeling and analysis. Two recent methods are [133] and [59]. Both methods compute skeletons on

incomplete point clouds. The former computes a ROtational Symmetry Axis (ROSA) which captures local symmetry in the shape of objects. The latter improves upon this by creating a L1-median skeleton which locally adapts to the shape of object. There are many more previous approaches that fit skeletons to 3D object models and these are discussed further in [59]. A very interesting approach in [100] approaches the problem from another direction. They first decompose the object into parts using segmentation techniques and critical-point/inflection-point computation based on surface concavities and then extract the skeleton from the segmented parts by joining the parts and refining the skeleton to represent the topology of the shape in accordance with *Morse Theory*. This theory is the basis for Reeb Graphs 1D skeletons and analyzing the topology of shapes using critical points. Inflection and critical points is discussed in detail by perceptual scientists and psychologists in [145]. The authors of [100] claim that this method follows the process of human perception of shape. Despite this it fails when the shape is partially occluded.

Many approaches extract 1D skeletons, not sheets or skeletal surfaces. An approach that extracts 2D medial representations from 3D is [96]. This approach, named the Discrete Scale Axis, improves upon Blum's original medial axis transform for 3D objects by following the medial axis transform with topology-preserving angle filtering. The approach argues that 2D shape medial representations are not directly extendable to 3D shapes [96, 127].

The problem with 2D medial representations from a computer vision perspective (*i.e* for object recognition/matching) is that the skeletal representation is still in a relatively high dimension. In [96] an illustrative example shows the body of a dolphin where it is represented by a relatively vertical sheet. This is unnecessary for this tubular shape. A sufficient representation can be a curve going through the center of the dolphin's body with some definition of the cross-sectional variation along the main axis as was done in Biederman's definitions of 3D object parts (geons) [11]. It is our view that only flat objects, such as rectangular walls, flattened ellipsoids or tabletops should have sheet representations in 2D. A 1D curve/line going through these objects no longer makes intuitive sense and complicates the representation for object recognition/matching purposes. All other objects or their primitive parts can be represented in their tubular/cylinderical form by a 1D curve.

Some of the approaches that have achieved good object segmentation into parts have been

tested on the Princeton Benchmark for 3D Mesh Segmentation [20] have also represented their shapes using skeletons, for example [124, 147]. Object segmentation, object-part decomposition and medial representations have a lot in common because they all complement each other in decomposing objects according to their shape. The decomposition itself (*i.e.* the shape structure) as well as the individual shape representations of the parts can then be used to perform shape matching.

In the field of Computer Vision and Robotics, the research community has been focusing intensely on shape representation in 2D images over the past decades. An example of shape matching in binary 2D images can be seen in [122] where distance metrics are defined between the graph representations. More recently [28] describes approaches to match Flux Graphs between 2D image projections of 3D objects under viewpoint variation. Recently with the emergence of 3D sensors (*e.g.* Xbox Kinect [148] and laser-scanners) and more available 3D data, shape representation for the foundational task of object recognition/matching has appeared. A recent large dataset of objects scanned by a Kinect sensor has set a object recognition benchmark for Robotics and Computer Vision systems [77]. Many, if not all, representations of 3D objects in this dataset have focused on 2D representations of the 3D, in other words, using feature descriptors such as HOG [24] or depth kernel descriptors [16] computed directly on the depth image. This is shape representation in a limited way and the actual full 3D shape information is lost. It is important to note that the Kinect sensors produce 2.5D information and not full 3D. 2.5D is another way of saying depth/range map which is represented as a grayscale depth image. The approach in [68] analyzes shape representations in order to detect and recognize objects using point clouds from depth sensors. It seems that a robust 3D representation of noisy realistic 2.5D and 3D data that are able to cope with the demanding task of object recognition are still lacking.

There are only two works that we are aware of that have explored representing medial representations probabilistically [47, 154]. These probabilistic approaches are important to be able to generate missing information and make predictions of shape boundaries based on the medial representation. This is very similar to the way humans perceive shape and can interpolate missing shape information. These approaches are confined to 2D shapes and no exploration of representing full 3D shapes stochastically have yet taken place.

Representing objects by shape alone is counter-intuitive in the sense that humans use many other cues, including visual cues for recognition and matching. A stable model of *object* representation should be built to combine shape and visual information.

Chapter 3

Multi-Modal Pose Estimation

The advent of inexpensive depth augmented color (RGBD) sensors (e.g. Microsoft Xbox Kinect [148]) has brought about a large advancement in the 3D perceptual capability of vision systems and mobile robots. Challenging vision problems like object category, instance and pose recognition have all benefited from this recent technological advancement. In this work we address the challenging problem of pose recognition using simultaneous color and depth information. For this purpose, we extend a state-of-the-art regression framework by using a multi-kernel approach to incorporate depth information to perform more effective pose recognition on table-top objects. We do extensive experiments on a large publicly available dataset to validate our approach. We show significant performance improvements (more than 20%) over published results.

3.1 Introduction

The introduction of RGBD sensors has had an immediate impact on mobile robotics where robots are being fitted with Kinect sensors to enhance their perceptive capability. A challenging problem for robots is to automatically identify objects (category and instance) as well as their poses. We feel that in this space, while the problems of object category and instance recognition have received a lot of attention, pose recognition has not been addressed adequately. Pose recognition is an important problem that can help the robot answer important questions, for example, *what is the arrangement and orientation of various objects or people?, Where is the handle of the mug?*, *etc.* . Addressing these questions is important for a variety of tasks like scene understanding, activity recognition, object manipulation in mobile robotics as well as for the wider areas of computer vision.

3D object pose recognition is a rich field of research [94, 52, 119, 15, 88]. However, the cheap availability of scene depth information synchronized with photometric (RGB/grayscale) is only a recent phenomenon. As a result, approaches that use both modalities of information are rather sparse. It is expected that this new modality has the potential to provide considerable boost in the accuracy of pose recognition systems.

The most relevant work in the area of pose (along with category and instance) recognition using synchronized multimodal photometric and depth data (*i.e.* RGBD) is by Lai et al.

[78, 77]. In [78], the authors show significant performance improvements for simultaneous recognition of object category, instance and pose recognition. They use machine learning to build an object-pose tree model from RGBD images and perform hierarchical inference on it. Although the performance improvements on category and instance recognition are impressive, object pose recognition performance is modest. The main reasons for this is that they use a classification strategy for pose recognition (resulting in coarse pose estimates) and do not fully utilize the information present in the spatial distribution of features on object surfaces.

In our previous work [139], we have addressed these two issues. Since the pose space is continuous, we developed a *regression* framework for estimating object pose (using only color images). Moreover, along with the appearance properties, our framework explicitly represents and models spatial distributions of salient features on the object surfaces to get a (continuous) pose estimate. This framework was shown to achieve state-of-the-art results on pose estimation from 2D images and significantly improved estimation on a variety of difficult datasets.

This work makes the following contributions: firstly, we use a multi-kernel learning (MKL) approach to extend our framework for pose estimation to use multimodal RGBD data and show the benefits of doing this. This confirms that 3D shape information captured in the form of depth-maps provides valuable additional information about object pose as well as the fact that the proposed framework is able to exploit this information. Using depth is very useful for objects which lack visual features and/or suffer from partial occlusion (e.g. second pitcher in fig. 3.3) which severely limits pose estimation using visual cues alone. Furthermore, depth features also have the advantage of being illumination-invariant.

Secondly, we do extensive experiments on a large, publicly available RGBD dataset [77] specially suited for this purpose. The dataset consists of RGBD data for 300 objects. The objects were put on a turntable and the Kinect sensor was used to gather data from 3 different sensor heights and 250 different object poses at each sensor height[2]. We evaluate our algorithm on this dataset and demonstrate that we achieve significant performance improvements over best published results.

In section 3.2, we describe the regularized kernel regression framework (from [139]) and

[2]Other RGBD datasets are available which are smaller and less suitable to validate pose-estimation performance – for a comprehensive description refer to [63].

how we extend it to use multiple kernels to incorporate depth information. In section 3.2.1 we describe the method of building the visual and depth kernels and lastly in section 3.2.2 we describe the various methods we have explored to learn using multiple kernels (MKL).

3.2 RGBD Multi-Kernel Regression

Let an image X^k be represented by its features – $X^k = \{x_i^k \in \mathbb{R}^2, f_i^k \in \mathbb{R}^{\mathbb{F}}, d_i^k \in \mathbb{R}^{\mathbb{D}}\}$ where $i = 1, ..., N_k$. Let x_i^k denote the i^{th} spatial location, and f_i^k and d_i^k the respective RGB and depth feature descriptors at that location. N_k is the number of local features (variant across images). Each image is also associated with a pose $v^k \in \mathbb{R}^{\mathbb{V}}$. The goal of regression is to learn a regularized mapping function $\hat{g}$ from input image features to the pose space from (paired) training data: (X^k, v^k). As shown in [139], the regression can be reduced to:

$$v = \hat{g}(X) = \sum_j b_j K(X, X^j) \tag{3.1}$$

$K(\cdot, \cdot)$, a positive definite (p.d.) kernel, measures the similarity between images. We extend this approach to use multiple kernels. We define two kernels – K for representing visual similarity and $G(\cdot, \cdot)$ for depth similarity. Multi-kernel regression is similarly reduced to:

$$v = \hat{g}(X) = \sum_j (b_j K(X, X^j) + c_j G(X, X^j)) \tag{3.2}$$

where G measures depth similarity between images and K measures similarity of visual local features within and between images.

We depict the proposed multi-kernel regression pipeline in fig. 3.1. It consists of two steps: feature embedding (section 3.2.1) and pose regression (section 3.2.2).

3.2.1 Feature Embedding

Our pose regression framework uses the feature embedding concept defined in prior work [139]. This section summarizes the key ideas of this work and then focuses on the extensions to use depth information.

The goal of feature embedding is to enforce regularity constraints: inter-image feature affinity, intra-image spatial affinity and prior manifold structure (topology, 1D...etc). At the same time, we want the embedded feature space to be independent of the variant number of features

in input images. In this work, we learn two feature embeddings: one for the RGB features and another for the depth features. This requires the specification of corresponding affinity matrices.

RGB feature embedding follows [139]: The spatial affinity within each image and feature affinity between images is computed using kernels resulting in two weight matrices. We use Geometric Blur (GB) as the RGB feature. A regularized objective function is then optimized (eqn 7, [139]) from training images to obtain a Laplacian Eigenmap embedding. Depth feature embedding is done in a slightly different (albeit simpler) way. Kinect depth data has many missing holes (see fig. 3.4) due to infrared absorption by some materials. We empirically established that a single global depth feature (per image) is more reliable than locally anchored multiple depth features. We use HOG applied to depth images (dHOG) as a single depth descriptor per image. We now obtain a Laplacian embedding for the depth features also.

Consequently, each training image is mapped to two embedded feature representations corresponding to the RGB and depth data. Embedding the features from an unseen test image is done by solving an out-of-sample problem which estimates the mapping between the input space and embedding space.

3.2.2 Pose Regression

Once the feature embedding is done, pose regression is performed to learn the model for predicting pose from embedded features (see fig. 3.1). We experimented with two approaches.

MKR: Multi-Kernal Regression

In the first approach, called Multi-Kernel Regression (MKR), we estimated coefficients b_j and c_j in eq. 3.2 from the training data using Euclidean error norm on the estimated pose. Note that the X in eq. 3.2 now corresponds to embedded features computed in the last section. The advantage of this approach is that since weights for both RGB and depth kernels are learned simultaneously for each training example, example-specific weights are learnt than can achieve a good tradeoff between the relative quality of information (or lack thereof) in depth or RGB data. Thus, objects with good discriminative visual features can be robust to noisy or non-discriminative depth features and vice versa. The resulting pose estimator can be represented

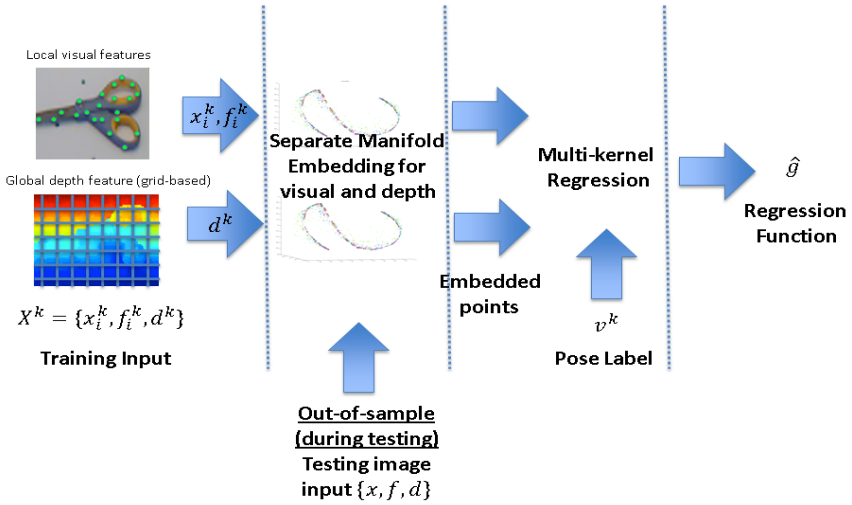


Figure 3.1: Pose Regression Using Feature Embedding. Model is learnt using training data. Estimation is done for test data by treating them as out-of-sample cases.

as follows:

$$v = \hat{g}(X) = \left[K|G\right] \begin{bmatrix} b \\ - \\ c \end{bmatrix} \tag{3.3}$$

K and G here are the p.d. kernels built over the RGB and depth feature embeddings independently.

MKL: Multi-Kernal Learning

The second approach we tried is Multi-Kernel Learning (MKL) [3] where effectively a new kernel is represented as a weighted sum of several kernels (as in eq. 3.4). This method is analogous to a system with multiple experts. Weights must be assigned to each kernel according to its discriminative power in the regression problem. For our scenario, this approach has an effect of learning an apriori relative bias for/against the RGB information vis-a-vis the depth

information.

$$K = \sum_{j}^{M} \eta_j K_j \tag{3.4}$$

M is the number of kernels and η are the scalar weights corresponding to each individual kernel. When combining RGB and depth we have two kernels representing each of these two modes. There are multiple ways of learning the discriminative power of each kernel. A common way is to use a correlation measure between individual kernel regressor and the ground truth values in the training data. This work uses two heuristics: F-measure and M-measure [109]. The former measures the kernel alignment using the Frobenius norm between the kernel regresser and the inner product of the pose labels. The higher this value is for a kernel, the more it contributes to the combined kernel. The latter measure is based on the mean square error when performing regression using each kernel on training data. The smaller the MSE per kernel is, the less contribution that kernel has to the combined kernel. These two measures give us the weights η_j corresponding to each kernel K^j. The weighted sum kernel is now used to learn the pose regression from (3.1). We call the two pose regressors MKL-F and MKL-M respectively.

3.3 Experiments

We evaluate the presented approach on a large, publicly available RGBD dataset [77] specially suited for pose recognition. The Kinect sensor was used to gather data from 3 different sensor heights and poses were densely captured per object using a turntable. We followed the same experimental setup and loss function as in [78] for a fair comparison with this state-of-the-art approach. Thus, images captured from elevation angles of 30° and 60° were used for training while those from 45° were used for testing.

Geometric Blur (GB) features were computed on RGB images to represent photometric information while HOG features [24] were computed on the depth images to represent depth information (dHOG). While 45 most-significant GB features (local) were used, we used a single global dHOG per image. The depth data has large areas of missing depth (see fig. 3.4) due to infrared absorption by some materials. A single global dHOG descriptor was found to be more

reliable than multiple local features because of its ability to capture the overall structure of the objects, including noisy areas of the depth image with missing depth (that happen to be consistent per the material of the object and thus pose-invariant). We also empirically analyzed the eigenvalues of the Laplacian Eigenmap embedding on the training data and found the best embedding dimensions to be 100 for GB (204-dimensional) and 75 for the 9x9-grid dHOG. We used Radial-Basis Function (RBF) kernels with Euclidean distances between feature vectors. These settings were used for all experiments.

The experimental results are presented in Table 3.1. The baseline approach [78] is presented in the last row. The first two rows represent our regression approach using only depth features (dHOG) and only RGB features (gsGB [139]), respectively. MKR represents our multi-kernel regression approach by concatenating the visual features kernel and global depth features kernel. MKL-F and MKL-M represent our regression approach when the kernel weights were computed using the F-measure and M-measure, respectively.

The dHOG approach shows worst performance, significantly lower than gsGB. This is due to two reasons: the depth feature is more noisy (large missing holes) than the RGB data. Secondly, for robustness, we use a single global depth feature while gsGB uses multiple locally anchored visual features. The relative spatial arrangement of these features is very informative about the object pose and is effectively exploited by our visual-only algorithm. It can be seen that the proposed MKR and MKL approaches are able to use information from both color and depth features to provide a statistically significant improvement over regression using only depth or only color features. Note that we see a larger jump in the median performance than in the mean performance. The reason for this is that we get a significant performance boost for most of the object classes. A few object categories do not perform as well which can be attributed to being objects that have uniform non-discriminative pose-invariant shape, such as ball and bowl categories. In fact, asking the question of which pose these objects are in is an ill-posed question. Note that we get a relative improvement of approximately 11% and 4% over our own color-only baseline implementation.

Lastly, we compare the performance of the proposed regression framework with the baseline algorithm in [78]. The proposed MKR and MKL approaches easily outperform the baseline. In fact, MKL-M achieves a relative performance improvement of approx. 21% and 32%

Method	Median Pose Accuracy	Avg Pose Accuracy	St. Dev.
dHOG (Depth)	51.25	50.62	5.16
gsGB (RGB) [139]	77.8	72.06	14.39
MKR (RGB+D)	**85.0**	**74.58**	13.69
MKL-F (RGB+D)	**86.3**	**75.50**	12.71
MKL-M (RGB+D)	**86.7**	**74.76**	14.21
[78]	71.40	56.80	-

Table 3.1: Pose Recognition Performance over all 51 classes of RGBD-dataset

in the median and average pose accuracy, respectively. The median performance using MKL-M is shown for a representative subset of objects in fig. 3.2.

3.4 Conclusion

In this work, we presented a novel MKL based regression framework that automatically combines the color and depth information present in the multi-modal RGBD image data for effective object pose estimation. We extensively evaluated the performance of the presented framework on a large dataset and showed that we are able to significantly outperform best published results in this area, thus validating the efficacy of the presented framework.

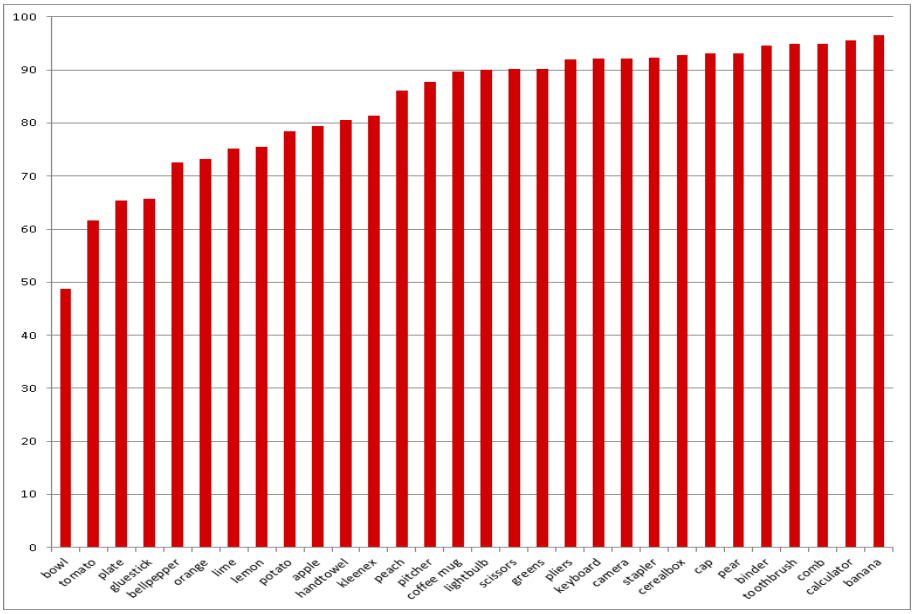


Figure 3.2: Median % accuracy for a subset of objects from RGBD-dataset

[1]Pose annotations indicate the yaw angle plotted on a unit circle anchored to the center of the image

Figure 3.3: Ground truth and estimated poses overlaid for sample images. The red line signifies the ground-truth pose, green represents the estimated pose using visual + depth and blue is the estimated pose using visual local features only. [1]

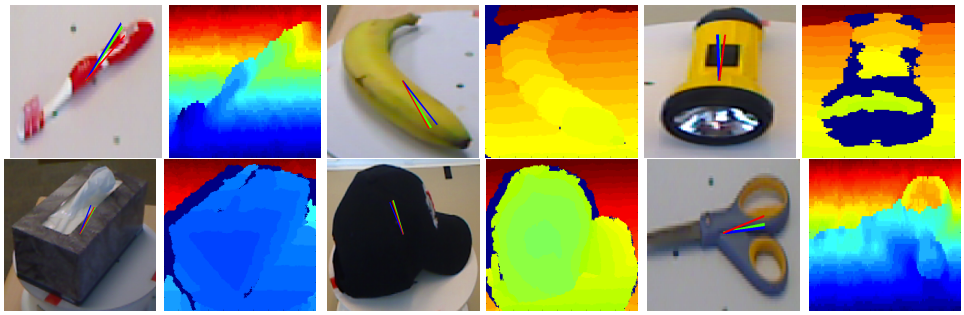


Figure 3.4: Pose estimates with RGB and depth images. Note missing depth on parts of the flashlight depth image (dark blue regions) [1]

Chapter 4

Unsupervised Joint Categorization and Pose Estimation using Manifold Approaches

Due to large variations in shape, appearance, and viewing conditions, object recognition is a key precursory challenge in the fields of object manipulation and robotic/AI visual reasoning in general. Recognizing object categories, particular instances of objects and viewpoints/poses of objects are three critical subproblems robots must solve in order to accurately grasp/manipulate objects and reason about their environments. Multi-view images of the same object lie on intrinsic low-dimensional manifolds in descriptor spaces (e.g. visual/depth descriptor spaces). These object manifolds share the same topology despite being geometrically different. Each object manifold can be represented as a deformed version of a unified manifold. The object manifolds can thus be parameterized by its homeomorphic mapping/reconstruction from the unified manifold. In this work, we develop a novel framework to jointly solve the three challenging recognition sub-problems, by explicitly modeling the deformations of object manifolds and factorizing it in a view-invariant space for recognition. We perform extensive experiments on several challenging datasets and achieve state-of-the-art results.

4.1 Introduction

Visual object recognition is a challenging problem with many real-life applications. The difficulty of the problem is due to variations in shape and appearance among objects within the same category, as well as the varying viewing conditions, such as viewpoint, scale, illumination, etc. Under this perceptual problem of visual recognition lie three subproblems that are each quite challenging: category recognition, instance recognition, and pose estimation. Impressive work has been done in the last decade on developing computer vision systems for generic object recognition. Research has spanned a wide spectrum of recognition-related issues, however, the problem of multi-view recognition remains one of the most fundamental challenges to the progress of the computer vision.

The problems of object classification from multi-view setting (multi-view recognition) and pose recovery are coined together, and directly impacted by the way shape is represented. Inspired by Marr's 3D object-centric doctrine [92], traditional 3D pose estimation algorithms often solved the recognition, detection, and pose estimation problems simultaneously (*e.g.*[51, 79, 87, 126]), through 3D object representations, or through invariants. However, such models

were limited in their ability to capture large within-class variability, and were mainly focused on recognizing instances of objects. In the last two decades the field has shifted to study 2D representations based on local features and parts, which encode the geometry loosely (*e.g.* pictorial structure like methods [44, 43]), or does not encode the geometry at all (*e.g.* bag of words methods [144, 129].) Encoding the geometry and the constraints imposed by objects' 3D structure are essential for pose estimation. Most research on generic object recognition bundle all viewpoints of a category into one representation; or learn view-specific classifiers from limited viewpoints, *e.g.* frontal cars, side-view cars, rear cars, etc. Recently, there has been an increasing interest in object categorization in the multi-view setting, as well as recovering object pose in 3D, *e.g.* [21, 75, 119, 86, 130, 132, 103, 95]. However, the representations used in these approaches are mainly category-specific representations, which do not support scaling up to a large number of categories.

The fundamental contribution of this work is the way we address the problem. We look at the problem of multi-view recognition and pose estimation as a style and content separation problem, however, in an unconventional and unintuitive way. The intuitive way is to model the category as the content and the viewpoint as a style variability. Instead, we model the viewpoint as the content and the category as a style variability. This unintuitive way of looking at the problem is justified from the point of view of learning the visual manifold of the data. The manifold of different views of a given object is intrinsically low in dimensionality, with known topology. Moreover, we can show that view manifolds of all objects are deformed version of each other. In contrast, the manifold of all object categories is hard to model given all within-class variability of objects and the enormous number of categories. Therefore, we propose to model the category as a "style" variable over the view manifold of objects. We show that this leads to models that can untangle the appearance and shape manifold of objects, and lead to multi-view recognition.

The formulation in this work is based on the concept of Homeomorphic Manifold Analysis (HMA) [37]. Given a set of topologically equivalent manifolds, HMA models the variation in their geometries in the space of functions that maps between a topologically-equivalent common representation and each of them. HMA is based on decomposing the style parameters in the space of nonlinear functions that map between a unified embedded representation of the

content manifold and style-dependent visual observations. In this work, we adapt a similar approach to the problem of object recognition, where we model the viewpoint as a continuous content manifold and separate object style variables as view-invariant descriptors for recognition. This results in a generative model of object appearance as a function of multiple latent variables, one describing the viewpoint and lies on a low-dimensional manifold, and the other describing the category/instance and lies on a low-dimensional subspace. A fundamental different in our proposed framework is the way 3D shape is encoded. An object's 3D shapes imposes deformation of its view manifold. Our framework, explicitly models the deformations of object manifolds and factorizes it in a view-invariant space for recognition. It should be notice that we ignore the problem of detection/localization in this work, and only focus on the problem of recognition and pose estimation assuming that bounding boxes or masks of the objects are given.

Pose recognition/estimation is fundamentally a six-degree-of-freedom (6DoF) problem [134], including 3DoF position $[x, y, z]$ and 3DoF orientation $[yaw, pitch, roll]$. However, in practical computer vision and robotic applications, pose estimation typically means solving for the some or all of the orientation degrees of freedom, while solving for the 3DoF position is usually called *localization*. In this work, we focused on the problem of estimating the 3DoF orientation of the object (or the 3DoF viewing orientation of the camera relatively), i.e. we assume that the camera is at a fixed distance to the object. We firstly considered the case of 1DoF orientation, i.e. a camera looking at an object on a turntable setting, which results in a one-dimensional view manifold, and then generalized to 2DoF and 3DoF orientation. Generalization to recover the full 6DoF of a camera is not obvious. Recovering the full 6DoF camera pose is possible for a given object instance. This can be achieved by traditional model-based method. However, this is a quite challenging task for the case of generic object categories.

There are various reasons why we only consider 3DoF viewing orientation and not full 6DoF. First, it is quite hard to have training data that cover the space of poses in that case; all the state-of-the-art dataset are limited to only a few views, or at most, multiple views of an object on a turn-table with a couple of different heights. Second, practically, we do not see objects in all possible poses, in many applications the poses are quite limited to a viewing circle or sphere. Even humans will have problems recognizing objects in unfamiliar poses. Third, for

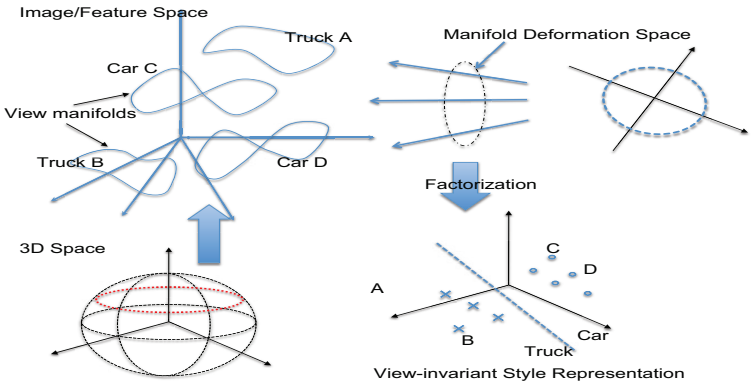


Figure 4.1: Framework for factorizing the view-object manifold. In the bottom left lies the space over which the views of the object are captured. Points in this space have corresponding feature points in the image/feature space which lie on entangled manifolds. A generative mapping can be found between points on a conceptual manifold space (*i.e.* a unit circle in the simple viewing circle setup) to the points in the feature space (see Figure 4.2 for two example object mappings). This mapping can then be factorized to find the latent factors over which we can solve for category and pose of objects.

most applications, it is not required to know the 6DoF pose, 1DoF pose is usually enough. Definitely for categorization 6DoF is not needed. In this work we show that we can learn from a viewing circle and generalize very well to a large range of views around it.

4.2 Framework

Intuition

The objective of our framework is to learn a manifold representation for multi-view objects that supports category, instance and viewpoint recognition. In order to achieve this, given a set of images captured from different viewpoints, we aim to learn a generative model that explicitly factorizes the following:

- Viewpoint variable (within-manifold parameterization): smooth parameterization of the viewpoint variations, invariant to the object's category.
- Object variable (across-manifold parameterization): parameterization at the level of each manifold that characterizes the object's instance/category, invariant to the viewpoint.

Consider collections of images containing instances of different object classes and different views of each instance. The shape and appearance of an object in a given image is a function

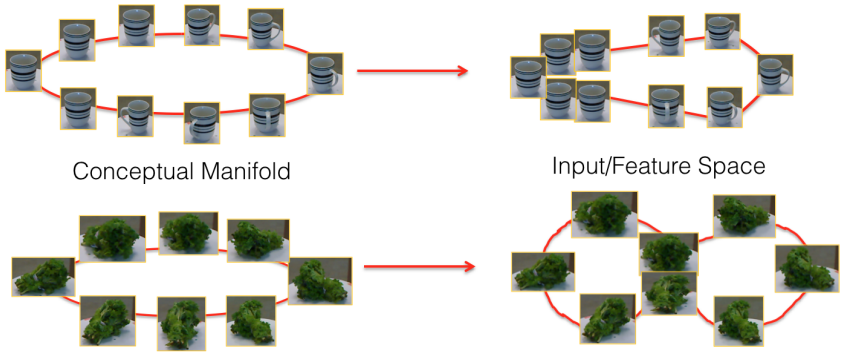


Figure 4.2: A motivational illustration of the mapping of a conceptual manifold to the input spaces of multiple view images of two different objects. Notice that the manifold can collapse where similar views lie in the input feature space.

of its category, style within category, viewpoint, besides other factors that might be nuisances for recognition. Our discussion do not assume any specific feature representation of the input, we just assume that the images are vectors in some *input space*. The visual manifold given all these variability collectively is impossible to model. Let us first simplify the problem. Let us assume that the object is detected in the training images (so there is no 2D translation or in-plane rotation manifold). Let us also assume we are dealing with rigid objects (to be relaxed), and ignore the illumination variations (assume using an illumination invariant feature representation). Basically, we are left with variations due to category, within category, and viewpoint, i.e., we are dealing with a combined *view-object manifold*.

The underlying principle is that multiple views of an object lie on an intrinsic low-dimensional manifolds in the input space (denoted as *view manifold*). The view manifolds of different objects are distributed in input space. To recover the category, instance and pose of a test image we need to know which manifold this image belongs to and the intrinsic coordinates of that image within the manifold. This basic view of object recognition and pose estimation is not new, and was used in the seminal work of [97]. PCA [66] was used to achieve linear dimensionality reduction of the visual data, and the manifolds of different object were represented as parameterized curves in the embedding space. However, dimensionality reduction techniques, whether linear or nonlinear, will just project the data (preserving the manifold's local and/or global geometry), and will not be able to achieve the desired untangled representation.

What is novel in our framework, is that we use the view manifold deformation as an invariant that can be used for categorization and modeling the within-class variations. Let us consider the case where different views are obtained from a viewing circle, e.g. camera viewing an object on a turntable. The view manifold of the object is a 1D closed manifold embedded in the input space. That simple closed curve deforms in the input space depending on the object shape and appearance (see Figure 4.2 for example mappings for two objects. Notice the deformation in the input feature space which depends on the shape and appearance of the objects). The view manifold can be degenerate, e.g. imaging a textureless sphere from different views results in the same image, i.e. the visual manifold in this case is degenerate to a single point. Therefore, capturing and parameterizing the deformation of a given object's view manifold tells us information about the object category and within category variation. If the views are obtained from a full or part of the view-sphere centered around the object, it is clear that the resulting visual manifold should be a deformed sphere as well (assuming the cameras are facing toward the object).

Let us denote the view manifold of an object instance s in the input space by $\mathcal{D}^s \subset \mathbb{R}^D$. D is the dimensionality of the input space. Assuming that all manifolds $\mathcal{D}^s$ are not degenerate (we will discuss this issue shortly), then they are all topologically equivalent, and homeomorphic to each other[1]. Moreover, suppose we can achieve a common view manifold representation across all objects, denoted by $\mathcal{M} \subset \mathbb{R}^e$, in an Euclidean embedding space of dimensionality e. All manifolds $\mathcal{D}^s$ are also homeomorphic to $\mathcal{M}$. In fact all these manifolds are homeomorphic to a unit circle in 2D for the case of a viewing circle, and a unit sphere in 3D for the case of full view sphere. *In general, the dimensionality of the view manifold of an object is bounded by the dimensionality of viewing manifold (degrees of freedom imposed by the camera-object relative pose).*

---

[1] A function $f : X \rightarrow Y$ between 2 topological spaces is called a homeomorphism if it is a bijection, continuous, and its inverse is continuous. In our case the existence of the inverse is assumed but not required for computation, i.e., we do not need the inverse for recovering pose. We mainly care about the mapping in a generative manner from $\mathcal{M}$ to $\mathcal{D}^s$.

Manifold Parameterization

We can achieve a parameterization of each manifold deformation by learning object-dependent regularized mapping functions $\gamma_s(\cdot) : \mathbb{R}^e \rightarrow \mathbb{R}^D$ that map from $\mathcal{M}$ to each $\mathcal{D}^s$. Given a Reproducing Kernel Hilbert Space (RKHS) of functions and its corresponding kernel $K(\cdot, \cdot)$, from the representer theorem [71, 108] it follows that such functions admit a representation of the form

$$\gamma_s(\boldsymbol{x}) = \boldsymbol{C}^s \cdot \psi(\boldsymbol{x}) \ , \tag{4.1}$$

where $\boldsymbol{C}^s$ is a $D \times N_\psi$ mapping coefficient matrix, and $\psi(\cdot) : \mathbb{R}^e \rightarrow \mathbb{R}^{N_\psi}$ is a nonlinear kernel map, as will be described in Section 4.3.

In the mapping (Eq. 4.1), the geometric deformation of manifold $\mathcal{D}^s$, from the common manifold $\mathcal{M}$, is encoded in the coefficient matrix $\boldsymbol{C}^s$. Therefore, the space of matrices $\{\boldsymbol{C}^s\}$ encodes the variability between different object manifolds, and can be used to parameterize such manifolds. We can parameterize the variability across different manifolds in a subspace in the space of coefficient matrices. This results in a generative model in the form

$$\gamma(\mathbf{x}, \mathbf{s}) = \mathcal{A} \times_2 \mathbf{s} \times_3 \psi(\mathbf{x}). \tag{4.2}$$

In this model $\mathbf{s} \in \mathbb{R}^{d_s}$ is a parameterization of manifold $\mathcal{D}^s$ that signifies the variation in category/instance of an object. $\mathbf{x}$ is a representation of the viewpoint that evolves around the common manifold $\mathcal{M}$. $\mathcal{A}$ is a third order tensor of dimensionality $D \times d_s \times N_\psi$, where $\times_i$ is the mode-i tensor product as defined in [25]. In this model, both the viewpoint and object latent representations, $\mathbf{x}$ and $\mathbf{s}$, are continuous.

There are several reasons why we learn the mapping in a generative manner from $\mathcal{M}$ to each object manifold (not the other way). First, this direction guarantees that the mapping is a function, even in the case of degenerate manifolds (or self intersections) in the input space. Second, mapping from a unified representation as $\mathcal{M}$ results in a common RKHS of functions. All the mappings will be linear combinations of the same finite set of basis functions. This facilitates factorizing the manifold geometry variations in the space of coefficients in Eq 4.2.

Given a test image $\mathbf{y}$ recovering the category, instance and pose reduces to an inference

problem where the goal is to find $\mathbf{s}^*$ and $\mathbf{x}^*$ that minimizes a reconstruction error, i.e.,

$$\arg\min_{\mathbf{s},\mathbf{x}} \|\mathbf{y} - \mathcal{A} \times_2 \mathbf{s} \times_3 \psi(\mathbf{x})\|^2. \tag{4.3}$$

Once $\mathbf{s}$ is recovered, an instance classifier and a category classifier can be used to classify $\mathbf{y}$.

Learning the model is explained in Section 4.3. Here we discuss and justify our choice of the common manifold embedded representation. Since we are dealing with 1D closed view manifolds, an intuitive common representation for these manifolds is a unit circle in $\mathbb{R}^2$. A unit circle has the same topology as all object view manifolds (assuming no degenerate manifolds), and hence, we can establish a homeomorphism between it and each manifold.

Dimensionality reductions (DR) approaches, whether linear (such as PCA [66] and PPCA [138]) or nonlinear (such as isometric feature mapping (Isomap) [135], Locally linear embedding (LLE) [115], Gaussian Process Latent Variable Models (GPLVM) [81]) have been widely used for embedding manifolds in low-dimensional Euclidean spaces. DR approaches find an optimal embedding (latent space representation) of a manifold by minimizing an objective function that preserves local (or global) manifold geometry. Such low-dimensional latent space is typically used for inferring object pose or body configuration. However, since each object has its own view manifold, it is expected that the embedding will be different for each object. On the other hand, using DR to embed data from multiple manifolds together will result in an embedding dominated by the inter-manifold distance and the resulting representation cannot be used as a common representation.

Embedding multiple manifolds using DR can be achieved using manifold alignment, e.g. [54]. If we embed aligned view manifolds for multiple objects where the views are captured from a viewing circle, we observe that the resulting embedding will converge to a circle. Similar results were shown in ([139]), where a view manifold is learned from local features from multiple instances with no prior alignment. This is expected since each object view manifold is a 1D closed curve in the input space, i.e. a deformed circle. Such deformation depends on object geometry and appearance. Hence it is expected that the latent representation of multiple aligned manifolds will converge to a circle. This observation empirically justifies the use of a unit circle as a general model of object view manifold in our case. Unlike DR where the goal is to find an optimal embedding that preserves the manifold geometry, in our case we only

need to preserve the topology while the geometry is represented in the mapping space. This facilitates parameterizing the space of manifolds. Therefore, the unit circle represents an *ideal* conceptual manifold representation, where each object manifold is a deformation of that ideal case. In some sense we can think of a unit circle as a prior model for all 1D view manifolds. If another degree of freedom is introduced which, for example, varies the pitch angle of the object on the turn-table then a sphere manifold would capture the conceptual geometry of the pose and be topologically-equivalent.

Dealing with degeneracy

Of course the visual manifold can be degenerate in some cases or it can be self intersecting, because of the projection from 3D to 2D and lack of visual features, e.g., images of a textureless sphere. In such cases the homeomorphic assumption does not hold. The key to tackle this challenge is in learning the mapping in a generative manner from $\mathcal{M}$ to $\mathcal{D}^s$, not in the other direction. By enforcing the known non-degenerate topology on $\mathcal{M}$, the mapping from $\mathcal{M}$ to $\mathcal{D}^s$ still exists, still is a function, and still captures the manifold deformation. In such cases the recovery of object pose might be ambiguous and ill-posed. In fact, such degenerate cases can be detected by rank-analysis of the mapping matrix $\boldsymbol{C}^s$.

4.3 Learning the Model

The input to the learning algorithm is images of different objects from different viewpoint, with viewpoint labels, and category label. For learning the representation, only the viewpoint labels are needed, while the category labels are used for learning classifiers on top of the learned representation, *i.e.* learning the representation is ”unsupervised” from the category perspective. Images of the same object from different views is dealt with as a set of points sampled from its view manifold. The number of sampled views do not necessarily be the same, nor they have to be aligned. We first describe constructing a common ”conceptual” view manifold representation $\mathcal{M}$ then we describe learning the model.

View manifold representation

Let the sets of input images be $Y^k = \{(\mathbf{y}_i^k \in \mathbb{R}^D, \mathbf{p}_i^k), i = 1, \cdots, N_k\}$ where D is the dimensionality of the input space (i.e. descriptor space) and p denotes the pose label. We construct a conceptual unified embedding space in $\mathbb{R}^e$, where e is the dimensionality of the conceptual embedding space. Each input image will have a corresponding embedding coordinate defined by construction using the pose labels. We denote the embedding coordinates by $X^k = \{\mathbf{x}_i^k \in \mathbb{R}^e, i = 1, \cdots, N_k\}$.

If we assume the input is captured from a viewing circle with yaw angles (viewpoints): $\Theta = \{\theta_i^k \in [0, 2\pi), i = 1, \cdots, N_k\}$, then the k-th image set is embedded on a unit circle such that $\mathbf{x}_i^k = [\cos\theta_i^k, \sin\theta_i^k] \in \mathbb{R}^2, i = 1, \cdots, N_k$. By such embedding, multi-view images with 1D pose variation are represented on a conceptual manifold (unit circle in 2D), i.e. a normalized 1-sphere. For the case of a full view sphere (2D pose variation represented by yaw and pitch angles), images are represented on a unit-sphere in 3D, i.e. a normalized 2-sphere. And for the case of 3D pose variation represented by yaw, pitch and roll angles, the conceptual manifold will be a normalized 3-sphere in 4D. Generally, assuming the pose angles of the input are $\mathbf{p}_i^k = \{(\theta_i^k, \beta_i^k, \zeta_i^k), i = 1, \cdots, N_k\}$ where θ, β and ζ indicate yaw angle, pitch angle and roll angle respectively, then the embedded coordinate of the i-th image $\mathbf{y}_i^k$ is defined as

$$\mathbf{x}_i^k = \begin{cases} \left[\cos\theta_i^k, \sin\theta_i^k\right]^T \in \mathbb{R}^2 \text{ (1D case)} \\ \begin{bmatrix} \cos\theta_i^k \cos\beta_i^k \\ \sin\theta_i^k \cos\beta_i^k \\ \sin\beta_i^k \end{bmatrix} \in \mathbb{R}^3 \text{ (2D case)} \\ \begin{bmatrix} \cos\theta_i^k \cos\beta_i^k \cos\zeta_i^k \\ \sin\theta_i^k \cos\beta_i^k \cos\zeta_i^k \\ \sin\beta_i^k \cos\zeta_i^k \\ \sin\zeta_i^k \end{bmatrix} \in \mathbb{R}^4 \text{ (3D case)} \end{cases} \tag{4.4}$$

Notice that by embedding on a conceptual manifold, we just preserve the topology of the manifold, not the metric input space. For clarity and without loss of generality, we only consider 1D case when describing the learning and inferring procedures in the following parts of this section and the next.

Homeomorphic Manifold Mapping

Given an input set Y^k and its embedding coordinates X^k on a unit circle, we learn a regularized nonlinear mapping function from the embedding to the input space, i.e. a function $\gamma_k(\cdot) : \mathbb{R}^e \rightarrow \mathbb{R}^D$ that maps from embedding space, with dimensionality e, into the input space with dimensionality D.

To learn such mappings, we learn individual functions $\gamma_k^l : \mathbb{R}^e \rightarrow \mathbb{R}$ for the l-th dimension in the feature space. Each of these functions minimizes a regularized loss functional in the form

$$\sum_{i}^{n^k} \left\| \mathbf{y}_{il}^k - \gamma_k^l(\mathbf{x}_i^k) \right\|^2 + \lambda \, \Omega[\gamma_k^l], \tag{4.5}$$

where $\|\cdot\|$ is the Euclidean norm, Ω is a regularization function that enforces the smoothness in the learned function, and λ is the regularizer that balances between fitting the training data and smoothing the learned function. From the representer theorem [71] we know that a nonlinear mapping function that minimizes a regularized risk criteria admits a representation in the form of linear combination of basis functions around arbitrary points $\mathbf{z}_j \in \mathbb{R}^e, j = 1, \cdots, M$ on the manifold (unit circle). In particular we use a semi-parametric form for the function $\gamma(\cdot)$. Therefore, for the l-th dimension of the input, the function γ_k^l is an RBF interpolant from $\mathbb{R}^e$ to $\mathbb{R}$. This takes the form

$$\gamma_k^l(\mathbf{x}) = p^l(\mathbf{x}) + \sum_{j=1}^{M} \omega_j^l \cdot \phi(|\mathbf{x} - \mathbf{z}_j|), \tag{4.6}$$

where $\phi(\cdot)$ is a real-valued basis function, ω_j are real coefficients and $|\cdot|$ is the 2^{nd} norm in the embedding space. p^l is a linear polynomial with coefficients c^l, i.e. $p^l(\mathbf{x}) = [1 \quad \mathbf{x}] \cdot c^l$. The polynomial part is needed for positive semi-definite kernels to span the null space in the corresponding RKHS. The polynomial part is essential for regularization with the choice of specific basis functions such as Thin-plate spline kernel [72]. The choice of the centers is arbitrary (not necessarily data points). Therefore, this is a form of Generalized Radial Basis Function (GRBF) [108]. Typical choices for the basis function include thin-plate spline, multiquadric, Gaussian[2], biharmonic and tri-harmonic splines. The whole mapping can be written in a matrix

[2] A Gaussian kernel does not need a polynomial part.

form

$$\gamma^k(\mathbf{x}) = \mathbf{C}^k \cdot \psi(\mathbf{x}), \tag{4.7}$$

where $\mathbf{C}^k$ is a $D \times (M + e + 1)$ dimensional matrix with the l-th row $[\omega_1^l, \cdots, \omega_M^l, c^{l^T}]$. The vector $\psi(x) = [\phi(|x - z_1|) \cdots \phi(|x - z_M|), 1, x^T]^T$ represents a nonlinear kernel map from the embedded conceptual representation to a kernel induced space. To ensure orthogonality and to make the problem well posed, the following condition constraints are imposed: $\Sigma_{i=1}^M \omega_i p_j(x_i) = 0, j = 1, \cdots, m$, where p_j are the linear basis of p. Therefore, the solution for $\mathbf{C}^k$ can be obtained by directly solving the linear system:

$$\begin{pmatrix} \mathbf{A} & \mathbf{P}_x \\ \mathbf{P}_t^T & \mathbf{0}_{(e+1)\times(e+1)} \end{pmatrix}_k \mathbf{C}^{k^T} = \begin{pmatrix} \mathbf{Y}_k \\ \mathbf{0}_{(e+1)\times d} \end{pmatrix}, \tag{4.8}$$

$\mathbf{A}$, $\mathbf{P}_x$ and $\mathbf{P}_t$ are defined for the $k - th$ set of object images as: $\mathbf{A}$ is a $N_k \times M$ matrix with $\mathbf{A}_{ij} = \phi(|x_i^k - z_j|), i = 1, \cdots, N_k, j = 1, \cdots, M, \mathbf{P}_x$ is a $N_k \times (e + 1)$ matrix with i-th row $[1, \mathbf{x}_i^{k^T}]$, $\mathbf{P}_t$ is $M \times (e + 1)$ matrix with i-th row $[1, \mathbf{z}_i^T]$. Y_k is a $N_k \times D$ matrix containing the input images for set of images k, i.e. $\mathbf{Y}_k = [\mathbf{y}_1^k, \cdots, \mathbf{y}_{N_k}^k]$. Solution for $\mathbf{C}^k$ is guaranteed under certain conditions on the basic functions used.

Decomposition

Each coefficient matrix $\mathbf{C}^k$ captures the deformation of the view manifold for object instance k. Given learned coefficients matrices $\mathbf{C}^1, \cdots, \mathbf{C}^K$ for each object instance, the category parameters can be factorized by finding a low-dimensional subspace that approximates the space of coefficient matrices. We call the category parameters/factors *style* factors as they represent the parametric description of each object view manifold.

Let the coefficients be arranged as a $D \times K \times (M + e + 1)$ tensor $\mathcal{C}$. The form of the decomposition we are looking for is:

$$\mathcal{C} = \mathcal{A} \times_2 \mathbf{S}, \tag{4.9}$$

where A is a $D \times d_s \times (M + e + 1)$ tensor containing category bases for the RBF coefficient space and $\mathbf{S} = [\mathbf{s}^1, \cdots, \mathbf{s}^K]$ is $d_s \times K$. The columns of $\mathbf{S}$ contain the instance/category parameterization. This decomposition can be achieved by arranging the mapping coefficients

as a $(D(M+e+1)) \times K$ matrix:

$$\mathbf{C} = \begin{pmatrix} \mathbf{c}_1^1 & \cdots & \mathbf{c}_1^K \\ \vdots & \ddots & \vdots \\ \mathbf{c}_{M+e+1}^1 & \cdots & \mathbf{c}_{M+e+1}^K, \end{pmatrix} \tag{4.10}$$

$[\mathbf{c}_1^k, \cdots, \mathbf{c}_{M+e+1}^k]$ are the columns of $\mathbf{C}^k$. Given $\mathbf{C}$, category vectors and content bases can be obtained by SVD as $\mathbf{C} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. The bases are the columns of $\mathbf{U}\mathbf{\Sigma}$ and the object instance/category vectors are the rows of $\mathbf{V}$. Usually, $(D(M+e+1)) \gg K$, so the dimensionality of instance/category vectors obtained by SVD will be K, i.e. $d_s = K$. The time complexity of SVD is $O(K^3)$ so here our approach scales cubically with the number of objects, and the space complexity is not much of a problem as SVD can be done on a large enough matrix containing tens of thousands of rows.

4.4 Inference of Category, Instance and Pose

Given a test image $\mathbf{y} \in \mathbb{R}^D$ represented in a descriptor space, we need to solve for both the viewpoint parameterization $\mathbf{x}^*$ and the object instance parameterization $\mathbf{s}^*$ that minimize Eq. 4.3. This is an inference problem and various inference algorithms can be used. Notice that, if the instance parameters $\mathbf{s}$ is known, Eq. 4.3 reduces to a nonlinear 1D search for viewpoint $\mathbf{x}$ on the unit circle that minimizes the error. This can be regarded as a solution for viewpoint estimation, if the object is known. On the other hand, if $\mathbf{x}$ is known, we can obtain a least-square closed-form approximate solution for $\mathbf{s}^*$. An EM-like iterative procedure was proposed in [35] for alternating between the two factors. If dense multiple views along a view circle of an object are available, we can solve for $\mathbf{C}^*$ in Eq. 4.7 and then obtain a closed-form least-square solution for the instance parameter $\mathbf{s}^*$ as

$$\mathbf{s}^* = \arg\min_{\mathbf{s}} ||\mathbf{C}^* - \mathcal{A} \times_2 \mathbf{s}||. \tag{4.11}$$

In the case where we need to solve for both $\mathbf{x}$ and $\mathbf{s}$, given a test image, we use a sampling methods similar to particle filters [6] to solve the inference problem (with K category/style samples $\mathbf{s}^1, \mathbf{s}^2, \cdots, \mathbf{s}^K$ in the category/style factor space and L viewpoint samples $\mathbf{x}^1, \mathbf{x}^2, \cdots, \mathbf{x}^L$ on the unit circle). We use the terms *particle* and *sample* interchangeably in our description of the approach.

To evaluate the performance of each particle we define the likelihood of a particle $(\mathbf{s}^k, \mathbf{x}^l)$ as

$$w_{kl} = exp\frac{-||\mathbf{y} - \mathcal{A} \times_2 \mathbf{s}^k \times_3 \psi(\mathbf{x}^l)||^2}{2\sigma^2}. \tag{4.12}$$

It should be noticed that such a likelihood depends on the reconstruction error to be minimized in Eq. 4.3. The less the reconstruction error is, the larger the likelihood will be.

We marginalize the likelihood to obtain the weights for $\mathbf{s}^k$ and $\mathbf{x}^l$ as

$$W_{s^k} = \frac{\sum_{l=1}^{L} w_{kl}}{\sum_{k=1}^{K}\sum_{l=1}^{L} w_{kl}}, W_{x^l} = \frac{\sum_{k=1}^{K} w_{kl}}{\sum_{k=1}^{K}\sum_{l=1}^{L} w_{kl}}. \tag{4.13}$$

Style samples are initialized as the K style vectors learned by our model (decomposed via SVD of matrix $\mathbf{C}$ in Eq. 4.10), and the L viewpoint samples are randomly selected on the unit circle.

In order to reduce the reconstruction error, we resample style and viewpoint particles according to W_s and W_x from Normal distributions, i.e. more samples are generated around samples with high weights in the previous iteration. To keep the reconstruction error decreasing, we keep the particle with the minimum error at each iteration. Algorithm 1 summarizes our sampling approach.

In the case of classification and instance recognition, once the parameters $\mathbf{s}^*$ are known, typical classifiers, such as k-nearest neighbor classifier, SVM classifier, etc., can be used to find the category or instance labels. Given $\mathbf{x}^*$ on the unit circle, the exact pose angles can be computed by the inverse trigonometric function as

$$\theta^* = \arctan(x_2^*/x_1^*), \tag{4.14}$$

where x_1^* and x_2^* are the first and second dimensions of $\mathbf{x}^*$ respectively. Similar solutions can be solved for 2D or 3D case.

Multimodal Fusion

For each individual channel (e.g. RGB and depth), a homeomorphic manifold generative model is built. Our model can be extended to include multiple modalities of information as long as there is smooth variation along the manifold as the viewpoint/pose changes.

We combine visual information (i.e. RGB) and depth information by using a combined objective function that encompasses the reconstruction error in each mapping. This is done by

Algorithm 1 Sampling approach for style and viewpoint inference.

Input:

Testing image or image feature, $\mathbf{y}$;
Core tensor in Eq. 4.2, $\mathcal{A}$;
Iteration number, $IterNo$;

Output:

Initialization:
Initialize particles $(\mathbf{s}^k, \mathbf{x}^l)$ where $k = 1, \cdots, K,\ l = 1, \cdots, L$;
Initialize weights of style samples, $W_{s^k} = 1/K$;
Initialize weights of viewpoint samples, $W_{x^l} = 1/L$;
Iteration:
for $i = 1; i < IterNo; i++$ **do**
    Compute the likelihood of particles $w_{kl} = exp\frac{-||\mathbf{y}-\mathcal{A}\times_2\mathbf{s}^k\times_3\psi(\mathbf{x}^l)||^2}{2\sigma^2}$;
    Update the weights of style samples $W_{s^k} = \frac{\sum_{l=1}^{L} w_{kl}}{\sum_{k=1}^{K}\sum_{l=1}^{L} w_{kl}}$;
    Update the weights of viewpoint samples $W_{x^l} = \frac{\sum_{k=1}^{K} w_{kl}}{\sum_{k=1}^{K}\sum_{l=1}^{L} w_{kl}}$;
    Keep the particle $(\mathbf{s}^*, \mathbf{x}^*) = \arg\max_{k=1,\cdots,K,\ l=1,\cdots,L}\{w_{kl}\}$;
    Resample $\mathbf{s}^k$ and $\mathbf{x}^l$ according to W_{s^k} and W_{x^l} respectively;
end for
return $(\mathbf{s}^*, \mathbf{x}^*)$;

training on each mode (RGB and depth) independently and then using a combined objective function (reconstruction error). The combined reconstruction error becomes:

$$\begin{aligned} E_{rgbd}(\mathbf{s}_{rgb}, \mathbf{s}_d, \mathbf{x}) = \lambda_{rgb}||\mathbf{y}_{rgb} - \mathcal{A}_{rgb} \times_2 \mathbf{s}_{rgb} \times_3 \psi(\mathbf{x})||^2 \\ +\lambda_d||\mathbf{y}_d - \mathcal{A}_d \times_2 \mathbf{s}_d \times_3 \psi(\mathbf{x})||^2 \end{aligned} \tag{4.15}$$

Notice that the two terms share the same viewpoint variable $\mathbf{x}$. λ_{rgb} and λ_d were selected empirically. Since visual data has less noise than depth (which commonly exhibits missing depth values, i.e. holes), we bias the visual reconstruction error term of Eq. 4.15. When resampling style and viewpoint samples in our approach (Algorithm 1), we calculate the likelihood of a particle $(\mathbf{s}^k_{rgb}, \mathbf{s}^k_d, \mathbf{x}^l)$ as

$$w_{kl} = exp\frac{-E_{rgbd}(\mathbf{s}^k_{rgb}, \mathbf{s}^k_d, \mathbf{x}^l)}{2\sigma^2}, \tag{4.16}$$

which is a little different from Eq. 4.12. The formulations of weights of style and viewpoint samples are the same as 4.13, where visual style sample $\mathbf{s}^k_{rgb}$ and depth style sample $\mathbf{s}^k_d$ share the same weight W_{s^k}. This means we use a combined style sample as $\mathbf{s}^k = (\mathbf{s}^k_{rgb}, \mathbf{s}^k_d)$ for inferring. When the optimal solution $(\mathbf{s}^*_{rgb}, \mathbf{s}^*_d, \mathbf{x}^*) = \arg\min_{\mathbf{s}_{rgb},\mathbf{s}_d,\mathbf{x}} E_{rgbd}(\mathbf{s}_{rgb}, \mathbf{s}_d, \mathbf{x})$ is obtained, a combined parameters $\mathbf{s}^* = [\lambda_{rgb}\mathbf{s}^*_{rgb}; \lambda_d\mathbf{s}^*_d]$ can be used for category and instance recognition.

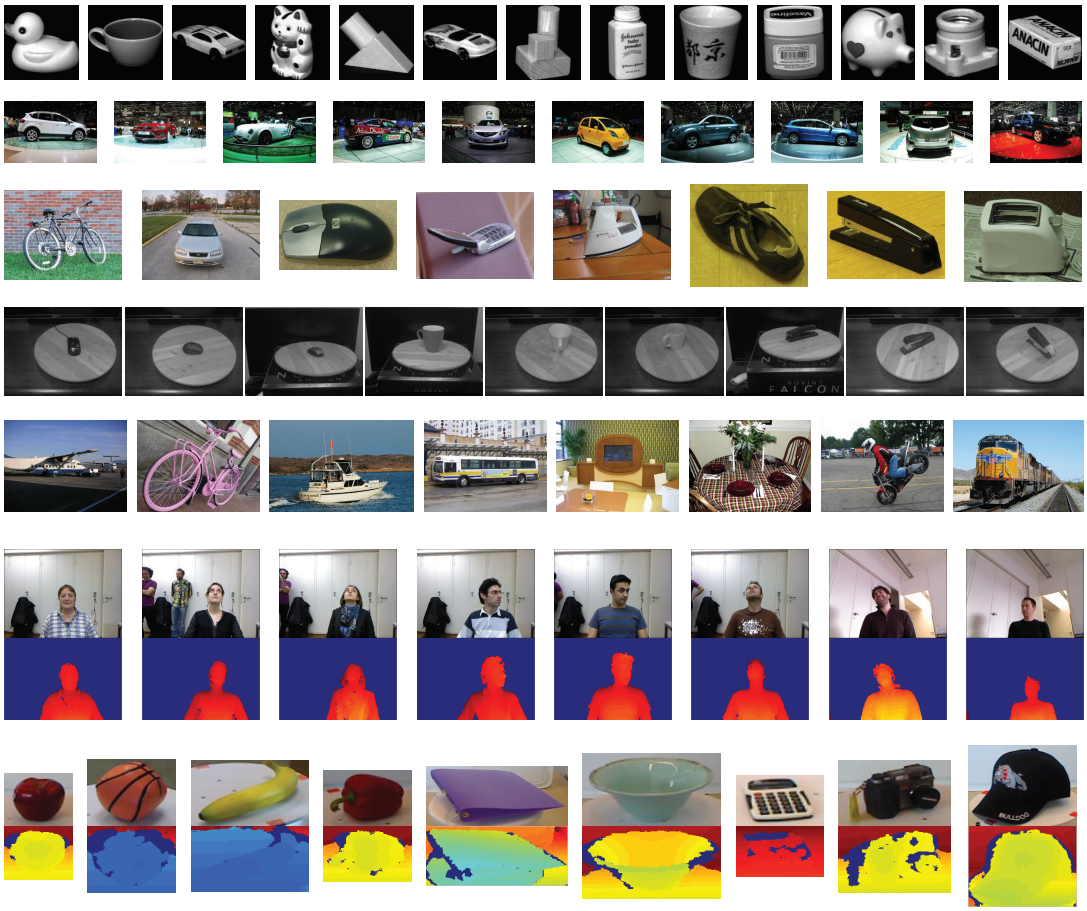


Figure 4.3: Sample images of different datasets. Rows from top to bottom: COIL-20 dataset [99], Multi-View Car dataset [101], 3D Object Category dataset [119], Table-top Object dataset [131], PASCAL3D+ dataset [149], Biwi Head Pose database [40], and RGB-D Object dataset [77].

4.5 Experiments and Results

Datasets

To validate our approach we experimented on several challenging datasets: COIL-20 dataset [99], Multi-View Car dataset [101], 3D Object Category dataset [119], Table-top Object dataset [131], PASCAL3D+ dataset [149], Biwi Head Pose database [40], and RGB-D Object dataset [77]. We give a brief introduction of these datasets in the following subsections. Fig. 4.3 shows sample images from each dataset.

COIL-20 dataset [99]: Columbia Object Image Library (COIL-20) dataset contains 20 objects, and each of them has 72 images captured every 5 degrees along a viewing circle. All

images consist of the smallest patch (of size 128×128) that contains the object, i.e. the background has been discarded. We used this datset for the task of arbitrary view image synthesis in order to illustrate the generative nature of our model.

Multi-View Car dataset [101]: The Multi-View Car dataset contains 20 sequences of cars captured as the cars rotate on a rotating platform at a motor show. The sequences capture full 360 degrees images around each car. Images have been captured at a constant distance from the cars. There is one image approximately every 3-4 degrees. Finely discretized viewpoint ground truth can be calculated by using the time of capture information from the images. This dataset is suitable for the validation of dense pose estimation.

3D Object Category dataset [119]: This dataset consists of 8 object categories (bike, shoe, car, iron, mouse, cellphone, stapler and toaster). For each object category, there are images of 10 individual object instances under 8 viewing angles, 3 heights and 3 scales, *i.e.* 24 different poses for each object. There are about 7000 images in total. Mask outlines for each object in the dataset are provided as well. The entire dataset can be used for multi-view object categorization and pose estimation. The car subset of 3D Object Category is typically used to evaluate the performance of sparse pose estimation.

Table-top Object dataset [131]: This dataset contains table-top object categories with both annotated 2D image and 3D point clouds. There are two subsets called Table-Top-Local and Table-Top-Pose. Table-Top-Local subset is specific to the task of object detection and localization. We only use Table-Top-Pose subset for pose estimation task. Table-Top-Pose contains 480 images of 10 object instances for each object categories (mice, mugs and staplers), where each object instance is captured under 16 different poses (8 angles and 2 heights). Data includes the images, object masks, annotated object categories, annotated object viewpoints and 3D point clouds of the scene.

PASCAL3D+ dataset [149]: PASCAL3D+ is a novel and challenging dataset for 3D object detection and pose estimation. It contains 12 rigid categories and more than 3000 object instances per category on average. The PASCAL3D+ images captured in real-world scenarios exhibit much more variability compared to the existing 3D datasets, and are suitable to test real-world pose estimation performance.

Biwi Head Pose database [40]: This database contains 24 sequences of 20 different people

(some recorded twice), captured with a Kinect sensor[3]. The subjects were recorded at roughly one meter distance to the sensor. The subjects move their heads around to try and span all possible yaw/pitch angles they could perform. There are over 15K images in the dataset. Each frame was annotated with the center of the head in 3D and the head rotation angles (respectively pith, yaw, and roll angles) by using the automatic system. For each frame, a depth image, the corresponding RGB image, and the annotation is provided. The head pose range covers about $\pm 75^{\circ}$ yaw, $\pm 60^{\circ}$ pitch, and $\pm 50^{\circ}$ roll. It is a good choice to use Biwi Head Pose database for 3D head pose estimation, as it provides fine ground truth of 3D rotation angles.

RGB-D Object dataset [77]: This dataset is large and consists of 300 common household table-top objects. The objects are organized into 51 categories. Images in the dataset were captured using a Kinect sensor that records synchronized and aligned visual and depth images. Each object was placed on a turntable and video sequences were captured for one whole rotation. There are 3 video sequences for each object each captured at three different heights (30°, 45°, and 60°) so that the object is viewed from different elevation angles (with respect to the horizon). The dataset provides ground truth pose information for all 300 objects. Included in the RGB-D Object dataset are 8 video sequences of common indoor environments (office workspaces, meeting rooms, and kitchen areas) annotated with objects that belong to the RGB-D Object dataset. The objects are visible from different viewpoints and distances, and may be partially or completely occluded. These scene sequences are part of the RGB-D Scenes dataset. As one of the largest and most challenging multi-modal multi-view datasets available, we used RGB-D Object dataset for joint category, instance and pose estimation on multi-modal data, and used it to test a near real-time system we built for category recognition of table-top objects.

Parameter Determination

As in Subsection 4.3, there is one key parameter in our model that significantly affects the performance. This is the number of mapping centers M. This parameter determines the density of arbitrary points $\mathbf{z}_j$ on the homeomorphic manifold when learning the nonlinear mapping function. If M is too small, the learnt mapping function may be not able to model the relationship

[3]http://www.xbox.com/en-us/kinect

between view manifolds and visual inputs well enough. On the other hand, the computation cost of learning the mapping function will increase in proportion to M. When M is larger than the number of training data points the learning problem becomes ill-posed. In addition to M, the image features are also important for our model. The images features are what represent the objects in the visual/input space. However our approach is orthogonal to the choice of the image representation, and any vectorized representation can be used.

To get proper parameters, we performed cross validation within the training data of each fold. For example, in the 50% split experiment of Subsection 4.5, we learnt our model on 9 out of the 10 car sequences in the training set and tested using the 1 left out. We performed 10 rounds of cross validation. Fig. 4.4 shows the performance of our model with different parameters: the dimensionality of HOG [24] features we used, and the number of mapping center (M). We used 35 mapping centers along a 2D unit circle to define the kernel map $\psi(\cdot)$ in Eq 4.1, and used HOG features calculated in 7×7 grids with 9 orientation bins to represent the inputs. The results in Table 4.2 were obtained using these parameters. Such cross validation is performed for each experiment in this section.

Arbitrary View Synthesis

Since our model is a generative model mapping from the manifold representation to visual inputs, we can perform arbitrary view synthesis if image intensities are used as visual inputs. We did arbitrary view synthesis experiments on COIL-20 dataset to show the generative nature of our model. We used 54 images to learn our generative model for each object in COIL-20 dataset, and tested the rest 18 images (every 4th), i.e. synthesized images from the viewpoints of the 18 testing images. We report mean squared error (MSE) to evaluate the synthesized images, which can be defined as following

$$MSE = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} [I_o(i,j) - I_s(i,j)]^2 \tag{4.17}$$

where $I_o(i,j)$ is the intensity of the pixel located at (i,j) in the testing image I_o of size $M \times N$, and I_s is the synthesized image from the same viewpoint of image I_o. For comparison, we also used typical manifold learning methods, including LLE [115], Isomap [135], and Laplacian Eigenmap (LE) [8], to learn a latent representing of the view manifold, and then learned a

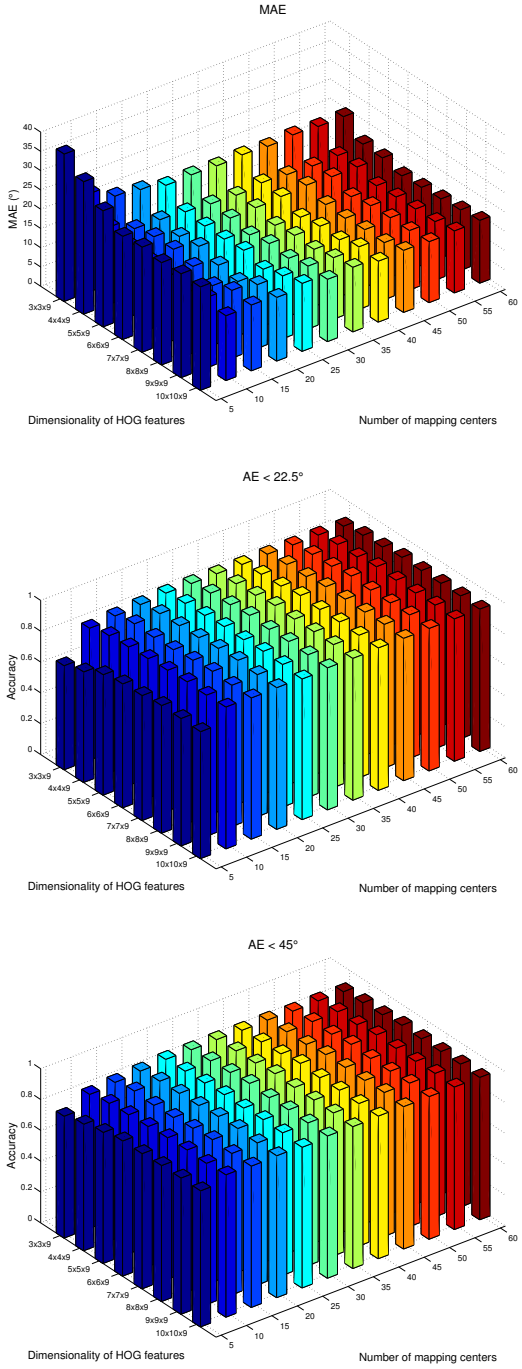


Figure 4.4: Cross validation results on Multi-View Car dataset for parameter determination. x and y axes are the number of mapping centers and the dimensionality of HOG features, z axis is the pose estimation performance. Titles are shown on the axes (zoom in to see the text). From top to bottom: MAE, $AE < 22.5°$, and $AE < 45°$. The dimensionality of HOG features is indicated as $n \times n \times 9$, meaning that HOG features are computed in $n \times n$ grids with 9 orientation bins.

similar generative map as Eq. 4.7. Notice that different from the conceptual manifold used in Subsection 4.3 where the embedded coordinates can be computed according to Eq 4.4 given the pose angles, the embedding coordinates of the unseen views (in the testing set) are obtained by linear interpolation between its neighbors in the training set with the assumption that the manifold learned by LLE, Isomap, or LE is locally linear. Results in Table 4.1 and Fig. 4.5 show that our model can correctly generate unseen view of a learned object, and our synthesis results are both quantitatively and qualitatively better than those obtained by typical manifold learning methods.

Table 4.1: Arbitrary view synthesis results on COIL-20 dataset

Method	Mean Squared Error
Isomap [135]	704
LLE [115]	950
LE [8]	7033
Ours	**361**

Dense Viewpoint Estimation

We experimented on the Multi-View Car dataset to evaluate our model for dense pose estimation. Following previous approaches [101, 139], there are two experimental setups: *50% split* and *leave-one-out*. For the former we take the first 10 cars for training and the rest for testing, resulting a 10-dimensional style space. For the latter we learn on 19 cars and test on the remaining 1, and the dimensionality of the style space is 19. Pixels within the bounding box (BBox) provided by the dataset were used as inputs.

For quantitative evaluation, we use the same evaluation criterion as [101, 139], *i.e.* Mean Absolute Error (MAE) between estimated and ground truth viewpoints. To compare with classification-based viewpoint estimation approaches (which use discrete bins) we also compute the percentages of test samples that satisfy $AE < 22.5^\circ$ and $AE < 45^\circ$ where the Absolute Error (AE) is $AE = |EstimatedAngle - GroundTruth|$). According to [139], the percentage accuracy in terms of $AE < 22.5^\circ$ and $AE < 45^\circ$ can achieve equivalent comparison with classification-based pose estimation approaches that use 16-bin and 8-bin viewpoint

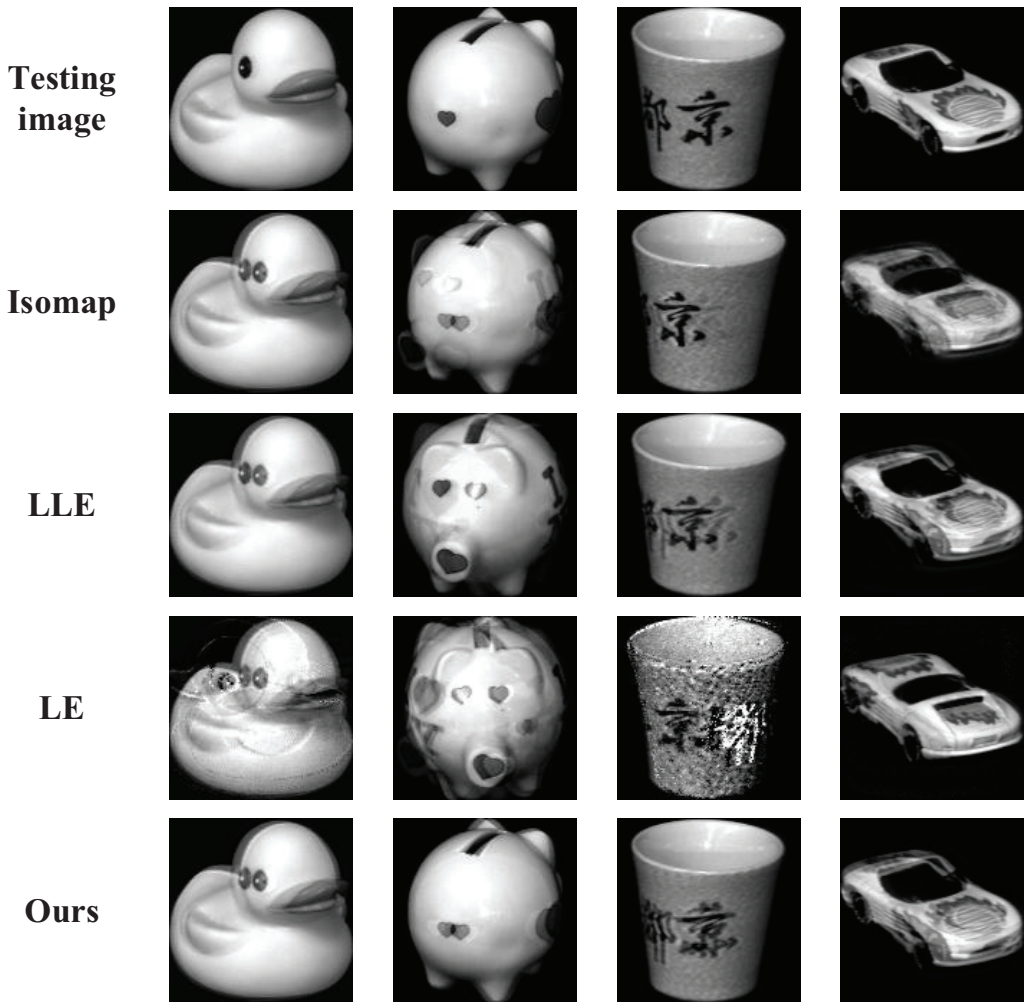


Figure 4.5: Synthesized images of unseen views. The first row shows image samples in testing set, and the rest four rows show synthesized images. Our results are visually better than other manifold learning methods, and are more robust as well.

Table 4.2: Results on Multi-View Car dataset

Method	MAE (°)	% of $AE < 22.5°$	% of $AE < 45°$
[101]	46.48	41.69	71.20
[139] - leave-one-out	35.87	63.73	76.84
[139] - 50% split	33.98	70.31	80.75
Ours - leave-one-out	**19.34**	**90.34**	**90.69**
Ours - 50% split	**24.00**	**87.77**	**88.48**

classifiers respectively.

We represented the input using HOG features. Table 4.2 shows the view estimation results in comparison to the state-of-the-art. Notice that results of [139] were achieved given bounding boxes of the cars while those of [101] were without bounding boxes, i.e. simultaneously performed localization. The quantitative evaluation clearly demonstrates the significant improvement we achieve.

Sparse Pose Estimation

To validate our approach for viewpoint estimation using sparse training samples on the viewing circle we did experiments on the 3D Object Category dataset and the Table-top Object dataset. Both datasets contain only 8 sparse views on the viewing circle. We used HOG features as input, calculated within the BBoxes (obtained from the mask outlines).

For 3D Object Category dataset, we used its car subset and bicycle subset, and followed the same setup as [119, 132]: 5 training sequences and 5 sequences for testing (160 training and 160 testing images). The Table-Top-Pose subset of the Table-top dataset was used for evaluating the viewpoint estimation of the following classes: staplers, mugs and computer mice. We followed the same setup as [131, 103]: the first 5 object instances are selected for training, and the remaining 5 for testing. Following the above setups, the dimensionality of the style space we used are both 5.

For comparison with [119, 132, 85, 103, 139, 131], we report our results in terms of $AE < 45°$ (equivalent to an 8-bin classifier). Results are shown in Table 4.3. Some of the state-of-the-art algorithms mentioned to jointly do detection and pose estimation (without BBox) and reported pose estimation only for successfully detected objects, while we do pose estimation for

all objects in the database given BBoxes. Therefore, the comparisons in Table 4.3 may be not completely fair. We indicate the setting for each approach and put results in the corresponding columns in Table 4.3. As shown in Table 4.3, our homeomorphic manifold analysis framework achieves 93.13% on the car subset of the 3D Objects dataset. This is far more than the state-of-the-art result of 85.38% in [103] and 77.5% in [139]. On the bicycle subset of the 3D Objects dataset our accuracy is 94.58%. This is more than 17% and 25% improvement over the results in [103] and [85], respectively. We also achieve the best average accuracy of 89.17% on the three classes of the Table-Top-Pose subset, improving about 26% and 43% over [103] and [131], respectively. These results show the ability of our framework to model the visual manifold, even with sparse views.

Table 4.3: Sparse pose estimation results and comparison with the state-of-the-arts

Dataset	Method	Pose estimation (without BBox)	Pose estimation (with BBox)
3D Object Category (car)	[119]	52.5%	-
3D Object Category (car)	[132]	66.63%	-
3D Object Category (car)	[85]	69.88%	-
3D Object Category (car)	[139]	-	77.5%
3D Object Category (car)	[103]	-	85.38%
3D Object Category (car)	Ours	-	93.13%
3D Object Category (bike)	[85]	75.5%	-
3D Object Category (bike)	[103]	-	80.75%
3D Object Category (bike)	Ours	-	94.58%
Table-Top-Pose	[131]	-	62.25%
Table-Top-Pose	[103]	-	70.75%
Table-Top-Pose	Ours	-	89.17%

Pose Estimation on PASCAL3D+ dataset

We performed pose estimation on PASCAL3D+ dataset [149]. Such a novel and challenging dataset is suitable to test pose estimation performance in real-world scenarios. We also used HOG features calculated within the BBoxes as input. We tested our model on 11 categories as the benchmark [149], including aeroplane, bicycle, boat, bus, car, chair, dining table, motorbike, sofa, train, and tv monitor, following the same experimental setting as [149]. Results in Table 4.4 show the power of our model for pose estimation. Noting that the benchmark

results of [149] were performed simultaneously with detection, the comparison in Table 4.4 is not completely fair.

Table 4.4: Pose performance (%) on PASCAL3D+ dataset. It can be seen that our approach outperforms VPDM [149]

Class	Ours (% of $AE < 45°$)	VDPM-8V	Ours (% of $AE < 22.5°$)	VDPM-16V
aeroplane	60.3	23.4	40.2	15.4
bicycle	60.7	36.5	40.3	18.4
boat	39.7	1.0	20.6	0.5
bus	73.0	35.5	68.7	46.9
car	55.4	23.5	46.4	18.1
chair	50.0	5.8	34.1	6.0
diningtable	45.2	3.6	37.5	2.2
motorbike	67.2	25.1	48.9	16.1
sofa	75.9	12.5	59.2	10.0
train	56.0	10.9	48.0	22.1
tvmonitor	80.1	27.4	55.1	16.3
average	59.0	18.7	44.2	15.6

3D Head Pose Estimation

To test our model for multi-view object pose estimation with 3D pose/viewpoint variation we performed experiments on the Biwi Head Pose database [40]. In our experiments, we only considered the problem of pose estimation and not head detection. We assumed that the faces were detected successfully, thus we just used the depth data within the bounding boxes (obtained from the provided masks) to compute HOG features. Head poses were represented on a 3-dimensional conceptual manifold in 4D Euclidean space, *i.e.* a normalized 3-sphere. For comparison, we ran a 5-fold and a 4-fold subject-independent cross validation on the entire dataset, resulting a 16-dimensional and a 15-dimensional style space respectively. This is the same experimental setup as [40] and [39]. We also reported the mean and standard deviation of the errors for each rotation angles. Results are shown in Table 4.5. It should be noticed that the pose results of [40] and [39] in Table 4.5 are computed only for correctly detected heads with 1.0% and 6.6% missed respectively. It can been seen that our model significantly outperforms [40] in 5-fold cross validation. In 4-fold cross validation, our mean errors are a little higher than [39] with respect to yaw and pitch, but our standard deviations are lower, which means

that our estimation results are more stable. These results show the ability of our model to solve continuous 3D pose estimation robustly.

Table 4.5: Summary of results on Biwi Head Pose database

Method	Validation	Yaw error	Pitch error	Roll error
Ours (Depth)	5-fold	$4.72 \pm 4.69°$	$3.84 \pm 3.90°$	$4.78 \pm 5.49°$
Ours (RGB)	5-fold	$8.09 \pm 7.90°$	$6.46 \pm 6.79°$	$6.00 \pm 6.49°$
Ours (RGB+D)	5-fold	$\mathbf{4.67 \pm 4.57°}$	$\mathbf{3.85 \pm 3.68°}$	$\mathbf{4.59 \pm 5.24°}$
Baseline [40] (Depth)	5-fold	$9.2 \pm 13.7°$	$8.5 \pm 10.1°$	$8.0 \pm 8.3°$
Ours (Depth)	4-fold	$4.84 \pm 4.78°$	$3.87 \pm 4.06°$	$4.79 \pm 5.61°$
Ours (RGB)	4-fold	$8.40 \pm 8.31°$	$6.60 \pm 6.87°$	$6.10 \pm 6.65°$
Ours (RGB+D)	4-fold	$4.81 \pm 4.77°$	$3.86 \pm 3.93°$	$\mathbf{4.73 \pm 5.49°}$
Baseline [39] (Depth)	4-fold	$3.8 \pm 6.5°$	$3.5 \pm 5.8°$	$5.4 \pm 6.0°$

Categorization and Pose Estimation

We used the entire 3D Objects Category dataset to evaluate the performance of our framework on both object categorization and viewpoint estimation. Similar to [119, 132], we tested our model on an 8-category classification task, and the farthest scale is not considered. We followed the same experimental setting as [119] by randomly selecting 8/10 object instances for learning and the remaining 2 instances for testing. Since there are totally 64 instances for training, the dimensionality of the style space used in this experiment is 64. Average recognition results of 45 rounds are shown in Table 4.6. We achieve an average recognition accuracy of 80.07% on 8 classes and an average pose estimation performance of 73.13%[4] on the entire test set which satisfies $AE < 45°$. We achieve markedly higher accuracy in recognition in 5 of the 8 classes than [119]. However our performance is not better than [104], which shows the room to improve the categorization capability of our model. In fact, a follow up paper [7] that uses our framework with a feed forward solution (without sampling) achieves much better results than [104].

[4]Notice that only pose results of correctly categorized images were taken for evaluation.

Table 4.6: Category recognition performance (%) on 3D Object Category dataset

Class	Ours	Baseline [119]	[104]
Bicycle	99.79	81.00	98.8
Car	99.03	70.00	99.8
Cellphone	66.74	76.00	62.4
Iron	75.78	77.00	96.0
Mouse	48.60	87.00	72.7
Shoe	81.70	62.00	96.9
Stapler	82.66	77.00	83.7
Toaster	86.24	75.00	97.8

Joint Object and Pose Recognition

We evaluated our model for joint object and pose recognition on multi-modal data by using the RGB-D Object dataset. Training and testing follows the exact same procedure as [78]. Training was performed using sequences at heights: 30° and 60°. Testing was performed using the 45° height sequence. We treated the images of each instance in the training set as one sequence, thus resulted a 300-dimensional style space. We used HOG features for both RGB channels and depth channel. We also experimented with an additional more recent depth descriptor called Viewpoint Feature Histogram (VFH) [117] computed on the 3D point cloud data.

Table 4.7 summarizes the results of our approach and compares to 2 state-of-the-art baselines. In the case of category and instance recognition (column 2 & 3), we achieve results on par with the state-of-the-art [78]. We find that $\approx 57\%$ of the categories exhibit better category recognition performance when using RGB+D, as opposed to using RGB only (set of these categories shown in Fig. 4.6-top). Fig. 4.6-bottom shows an illustration of sample instances in the object style latent space. Flatter objects lie more towards the lefthand side and rounder objects lie more towards the righthand side. Sample correct results for object and pose recognition are shown in Fig. 4.7.

Incorrectly classified objects were assigned pose accuracies of 0. Avg. and Med. Pose (C) are computed only on test images whose categories were correctly classified. Avg. and Med. Pose (I) were computed only using test images that had their instance correctly recognized. All the object pose estimations significantly out-performs the state-of-the-art [78, 33]. This verifies that the modeling of the underlying continuous pose distribution is very important in

pose recognition.

Lime and bowl categories were found to have better category recognition accuracy when using depth only instead of using either visual-only or visual and depth together. This can be explained by the complete lack of visual features on their surfaces. Some object instances were classified with higher accuracy using depth only also. There were 19 (out of 300) of these instances, including: lime, bowl, potato, apple and orange. These instances have textureless surfaces with no distinguishing visual features and so the depth information alone was able to utilize shape information to achieve higher accuracy.

In Table 4.7 we see that depth HOG (DHOG) performs quite well in all the pose estimation experiments except for where misclassified categories or instances were assigned 0 (column 3 & 4). DHOG appears to be a simple and effective descriptor to describe noisy depth images captured by the Kinect in the dataset. It achieves better accuracy than [78] in the pose estimation. Similar to [78], recursive median filters were applied to depth images to fill depth holes. This validates the modeling of the underlying continuous distribution which our homeomorphic manifold mapping takes advantage of. VFH is a feature adapted specifically to the task of viewpoint estimation from point cloud data. No prior point cloud smoothing was done to filter out depth holes and so its performance suffered.

Table 4.7: Summary of results on RGB-D Object dataset using RGB/D and RGB+D (%). Cat. and Inst. refer to category recognition and Instance recognition respectively

Methods	Cat.	Inst.	Avg Pose	Med Pose	Avg Pose (C)	Med Pose (C)	Avg Pose (I)	Med Pose (I)
Ours (RGB)	92.00	74.36	61.59	89.46	80.36	93.50	82.83	93.90
Linear SVM (RGB)	75.57	41.50	-	-	-	-	-	-
Ours (Depth - DHOG)	74.49	36.18	26.06	0.00	66.36	86.60	72.04	90.03
Linear SVM (Depth - DHOG)	65.30	18.50	-	-	-	-	-	-
Ours (Depth - VFH)	27.88	13.36	7.99	0.00	57.79	62.75	59.82	67.46
Ours (RGB+D)	93.10	74.79	**61.57**	**89.29**	**80.01**	**93.42**	**82.32**	**93.80**
Baseline (RGB+D) [78]	94.30	78.40	53.30	65.20	56.80	71.40	68.30	83.20
Linear SVM (RGB+D)	86.86	47.42	-	-	-	-	-	-
Baseline (RGB+D) [33]	-	-	-	-	74.76	86.70	-	-

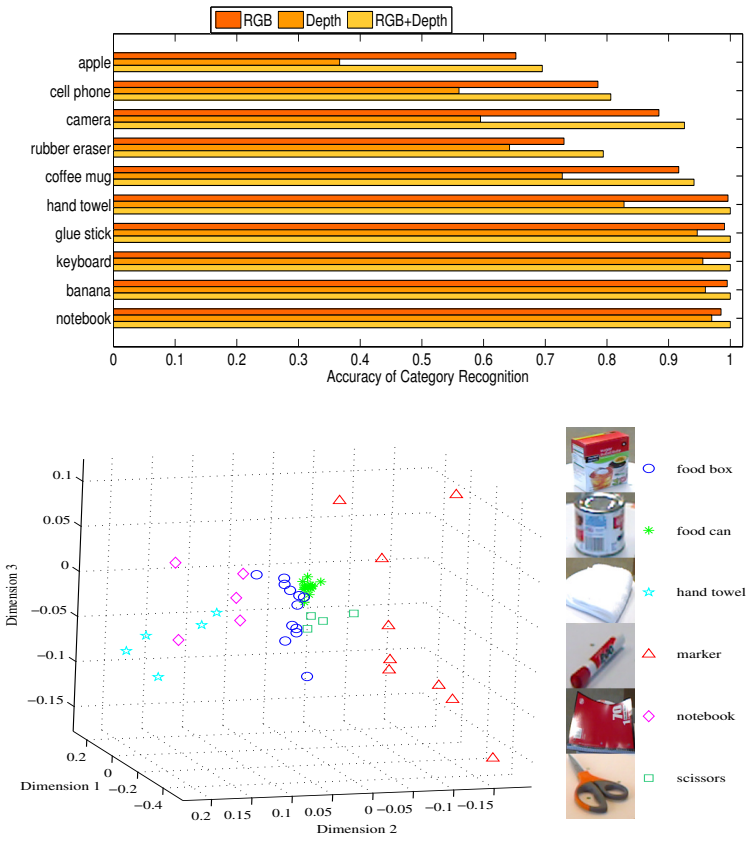


Figure 4.6: Top: Category recognition using different modes for a subset of categories in RGB-D Object dataset. Bottom: Sampled instances from 6 different categories in RGB-D Object dataset. Notice: flatter objects lie to the left and more rounded shapes to the right

Table-top Object Category Recognition System

We built a near real-time system for category recognition of table-top objects based on the homeomorphic manifold analysis framework described. Our system was trained on a subset of 10 different categories from the RGB-D Object dataset. The category recognition runtime per object in one frame is less than 2 seconds. Our MATLAB implementation was not optimized for real-time processing but despite this, the potential for real-time capability is evident. We only performed visual-only and depth-only training and testing of the system. We did not experiment with the combination of both modes as we wanted to optimize for speed as much as possible.

The system was tested on videos provided in the RGB-D Scenes dataset that contain cluttered scenes with occlusion, a much larger variation of viewpoint and varying scales (*e.g.*

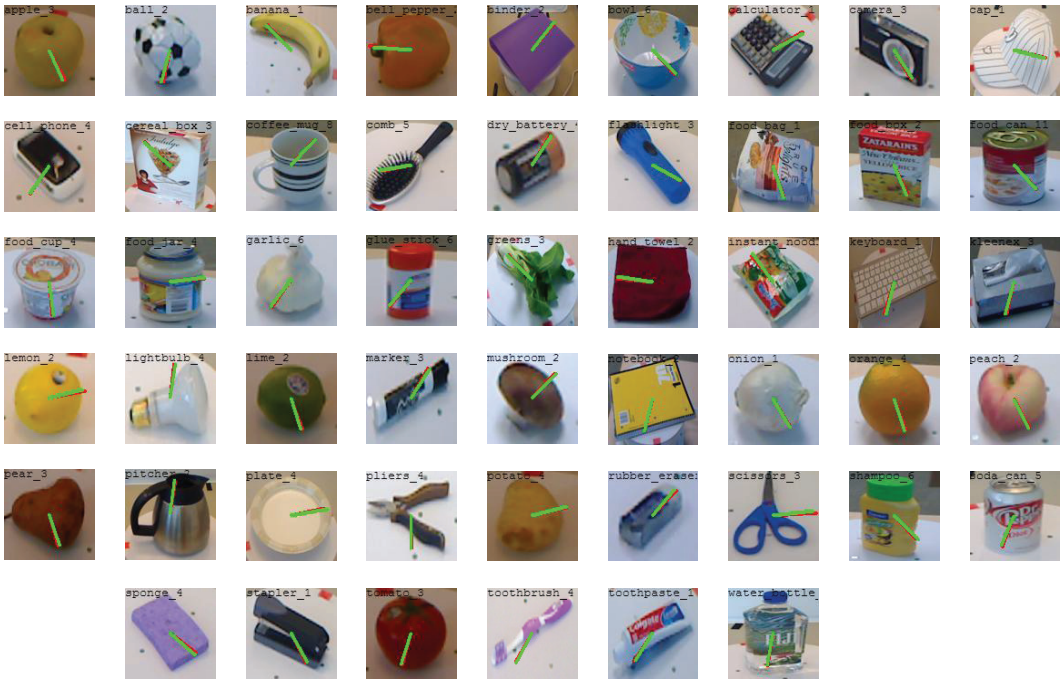


Figure 4.7: Sample correct results for object and pose recognition on RGB-D Object dataset. Black text: category name and instance number. Red line: estimated pose. Green line: ground truth pose.

kitchen and desk scenes). Our system achieved $>62\%$ category recognition accuracy using the depth mode only. An interesting observation was that depth-only recognition outperformed visual-only recognition in these cluttered scenes; intuitively due to the fact that background texture around objects introduces visual noise. In the depth mode, large depth discontinuities help to separate objects from background clutter and this aids recognition. We also tested our system on never-seen-before objects placed on table-tops without clutter. For this, we used the visual-only mode since there was no clutter in the scene. Fig. 4.8 shows results achieved by our system running on never-seen-before objects and objects from the videos provided in the RGB-D Scenes dataset.

Depth segmentation was performed on point clouds generated using the Kinect sensor in real-time using the Point Cloud Library [118]. This allows the table-top object to be segmented away from the table plane. The segmented objects are then found in the visual and depth images using the segmented object in the point cloud and then cropped to the size of the object. We then perform category recognition on these cropped images.

Figure 4.8: Near real-time system running on single table-top objects (first 2 rows) and the RGB-D Scenes dataset (last 3 rows, where green boxes indicate correct results while red boxes indicate incorrect results).

Computational Complexity

The computation complexity of SVD scales cubically with the number of objects in our case ($O(N^3)$). SVD can be done offline on large enough matrices containing tens of thousands of rows. The running time of our near real-time system for table-top object category recognition also shows that the computational complexity of the estimation phase is acceptable and has the potential for real-time.

4.6 Conclusion

In this work we have presented a unified framework that is based on homeomorphic mapping between a common manifold and object manifolds in order to jointly solve the 3 subproblems of object recognition: category, instance and pose recognition. Extensive experiments on several recent and large datasets validates the robustness and strength of our approach. We significantly outperform state-of-the-art in pose recognition. For object recognition we achieve accuracy on par and in some cases better than state-of-the-art. We have also shown the capability of our approach in estimating full 3D pose. We have also shown the potential for real-time application to AI and robotic visual reasoning by building a working near real-time system that performs table-top object detection and category recognition using the Kinect sensor.

Chapter 5

Supervised Joint Categorization and Pose Estimation using Manifold Approaches

Object recognition and pose estimation are two fundamental problems in the field of computer vision. Recognizing objects and their poses/viewpoints are critical components of ample vision and robotic systems. Multiple viewpoints of an object lie on an intrinsic low-dimensional manifold in the input space (*i.e.* descriptor space). Different objects captured from the same set of viewpoints have manifolds with a common topology. In this work we utilize this common topology between object manifolds by learning a low-dimensional latent space which non-linearly maps between a common unified manifold and the object manifold in the input space. Using a supervised embedding approach, the latent space is computed and used to jointly infer the category and pose of objects. We empirically validate our model by using multiple inference approaches and testing on multiple challenging datasets. We compare our results with the state-of-the-art and present our increased category recognition and pose estimation accuracy.

5.1 Introduction

Visual object recognition and pose estimation are two fundamental problems in the field of computer vision. Recognizing objects and their poses/viewpoints are critical components of vision and robotic systems. With the pervasiveness of robots today, they are required to not only visually recognize objects but also grasp and interact with them. For this reason simultaneously recognizing objects as well as their poses is of utmost importance. The difficulty of the problem lies in the large variation in appearance within object categories and between varying poses of the same objects. This is due to variations in, for example, illumination, texture and self-occlusions.

Recent research in the field of generic object categorization and pose estimation can be divided into four tracks, depending on how the models deal with different views and different categories. First, there are models that ignore estimating object pose and learn discriminative object models from all training data to categorize objects. The assumption here is that the representation and classifier become view-invariant. This is an assumption that is hard to meet in reality. Second, there are approaches that learn view-specific models for object category recognition. These approaches discretize the view space into a small number of canonical views (*e.g.* car front, car back, car rear) and learn classifiers for each of them. Thirdly, there

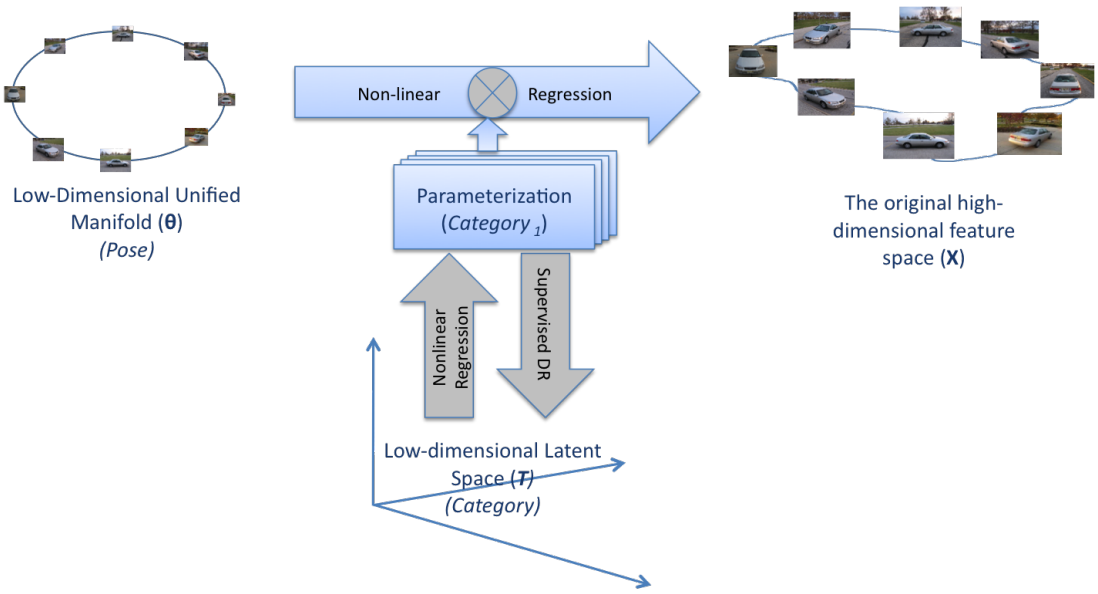


Figure 5.1: Illustration of our dual latent generative model: First, we parameterize the manifold using multiple-view images of a specific object using non-linear RBF mapping. The mapping is from the view-point space (Θ) to the image feature space ($\mathbf{X}$). The manifold paramerterizations are embedded into low-dimensional latent space ($\mathbf{T}$), using supervised technique (based on KPLS). Consequently, each point in the feature space is generated from a point in view-point space and a point in the $\mathbf{T}$-space.

are approaches that learn category-specific models, with the aim of estimating the viewpoint at a finer granularity, *e.g.* [139, 95, 121]. Finally, there are very few recent approaches that aim at learning a joint representation of object categories and poses [78, 152].

In this work, we tackle the problems of object category recognition and pose estimation simultaneously through a joint representation. The main intuition behind our representation follows the concept proposed in [152], where a common view manifold of different objects is used as a central representation and different categories are modeled as the style variability of that manifold. This unintuitive way of looking at the problem capitalizes on the fact that the view manifold of a given object is low dimensional and of known topology. For example, considering a viewing circle, the view manifold will be one dimension, and considering a viewing sphere, the view manifold is two dimensional. In this model all objects are assumed to share the same view manifold topology and that there is a homomorphism between these manifolds and the input space. In contrast, the manifold of all object categories can be of infinite dimensions and hard to model given all within-class variability of objects. Therefore the model separates

different categories by factorizing the deformation space of objects' view manifolds to reach a latent representation of categories, which is used in recognition.

There are clear limitations in the model proposed in [152]. The factorization of the manifold deformation space among different objects was done in an unsupervised way, without utilizing the available class labels. Instead we propose a supervised approach to achieve a discriminative factorization of the deformation space. As a result, we propose a generative model that is nonlinear in both the viewpoint and category latent variables. We also address scalability limitation of [152] by proposing a hybrid hierarchical model where a discriminative model is used to classify super-categories and then several generative models are used, one for each super-category in order to infer the pose and category.

There are three main contributions of this work. First, we propose a novel framework for view-invariant category recognition and pose estimation. This framework presents a nonlinear method for separating the manifold parameterization (referred to as *style*) and pose variations over the manifold (referred to as *content*) in sets of images. Our generative model is a purely non-linear approach which represents both spaces (category and pose) using nonlinear latent space embedding. This is in contrast to previous approaches that do not represent the non-linearities across object categories. For this reason our approach has the advantage of being more robust to within-category and pose variations. The second contribution is that our framework of style/content factorization moves the inference of the category and pose from the very high-dimensional feature space into two orthogonal low-dimensional spaces, one for category and the other for pose. Inference in a low-dimensional space guarantees increased accuracy and computational performance. Our framework uses supervised manifold embedding in a low-dimensional space and thus increases the point clustering, and in turn the classification accuracy. The third contribution is that we present the use of different distance metrics, different optimization techniques and compare the results of these configurations through extensive experimentation.

To validate our approach, we present theoretical derivations of the generative model and also test our model on three large public datasets. We compare our results with state-of-the-art approaches for these datasets.

5.2 Dual Latent Generative Model

We learn the model over sets of images. Each set contains images of the same object instance from different viewpoints. Let us denote the k-th set by $S^k = \{(\mathbf{x}_i^k, v_i^k, y^k), i = 1, \cdots, N_k\}$, where $\mathbf{x}_i^k \in \mathbb{R}^D$ is the feature representation of the i-th image with viewpoint $v_i^k \in \mathbb{R}^v$ and class label $y^k \in \{1, \cdots, C\}$. Our goal is to learn a generative model that isolates the viewpoint and the category variations into two separated latent variables: a viewpoint latent variable $\theta \in \mathbb{R}^{d_1}$ and a category latent variable $\mathbf{t} \in \mathbb{R}^{d_2}$

The image sequence of an object rotating on a turn-table lie on a low-dimensional view manifold ($\mathcal{M}^k$) in $\mathbb{R}^D$. We call these manifolds *instance manifolds*. Under the assumption that all the input sets are captured using the same degrees of freedom between the camera and the object, all these manifolds are topologically equivalent, however they have different geometry in $\mathbb{R}^D$. In other words, all these manifolds are geometrically deformed instances of each other. We can find a unified manifold that is topological equivalent to all these instance manifolds. The dimensionality of the view manifold of each instance depends on the degrees of freedom between the object and the camera. Assuming no degeneracy, a one viewing circle around the object implies a one-dimensional view manifold homeomorphic to a unit circle. A viewing sphere implies a two-dimensional view manifold homeomorphic to a unit sphere. All manifolds are internally parameterized by a latent variable θ lying on the unified manifold, representing the viewpoint.

As a result of the homeomorphism, each manifold geometry can be parameterized by its geometric deformation of the unified manifold. This parametrization space is view-invariant. The large dimensionality of the manifold parameterization space makes the inference hard and non-robust. Therefore, we learn low-dimensional representation $\mathbf{t}$ for the deformation space. We use supervised kernel-based partial least squares (K-PLS [114]) to discover this low-dimensional latent space. As result, the latent variable $\mathbf{t}$ is view-invariant category based representation, which holds the appearance and geometric characteristics of the instance.

As the number of classes increases, it is useful to have hierarchical model which group categories that share similar geometric characteristics together to form super-categories.

During test, the goal is to recognize the class label y^* and the viewpoint value v^* for a test

image $\mathbf{x}^*$. This requires inferring the latent pair $(\mathbf{t}^*, \theta^*)$ which represent the image $\mathbf{x}^*$. For doing that, we learn the following nonlinear generative models

$$\hat{\mathbf{x}}(\mathbf{t}, \theta) = \mathcal{A} \times_1 \phi(\mathbf{t}) \times_2 \psi(\theta) \tag{5.1}$$

Where ϕ is the nonlinear mapping of the latent space and ψ is the Radial-Basis function of the viewpoints θ.[1] This model is based on nonlinear regression over two factors; the pose θ and the latent category representer $\mathbf{t}$. Figure 5.1 shows graphical representation of the used generative model. This figure shows that for each point in the image space it corresponds to a point on the unified manifold ($\mathbf{\Theta}$) and a point in the low-dimensional parameterization space ($\mathbf{T}$). A new point in the image space can be computed using the tensor $\mathcal{A}$ in Eq. 5.1.

Given a test image, we solve for the optimal point in the latent space and the pose by solving the following optimization function.

$$(\mathbf{t}^*, \theta^*) = \arg\min_{(\mathbf{t},\theta)} \delta(\mathbf{x}_{test}, \hat{\mathbf{x}}(\mathbf{t}, \theta)) \tag{5.2}$$

where $\delta(\cdot)$ is a distance metric in the feature space. Once $\mathbf{t}^*$ and θ^* are recovered, we can obtain the class label and view point through two classification functions $y^* = f(\mathbf{t}^*)$ and $v^* = g(\theta^*)$. Figure 5.1 illustrates the overall framework, and the details are provided in Section 5.3 and 5.4.

5.3 Learning the Model

As shown in Figure 5.1, each manifold is initially parameterized using a nonlinear mapping from a unified manifold representation to the input space (Section 5.3.1). Then the manifold parameterization is embedded in a low-dimensional latent space using a supervised dimension reduction (DR) technique called K-PLS (Section 5.3.2). Finally, we learn the nonlinear mapping from latent space to the parameterization space to complete the generative model (Section 5.3.3). The details of each step is provided in this section.

---

[1] For convenience: we use bold small letters for vectors, bold capital for matrices, calligraphic capital for tensors and regular small for scalars. $\times_i$ denotes mode-i tensor vector multiplication as defined in [80].

5.3.1 Manifold Parameterization

The first step is to find a parameterization for each instance manifold. We adapt the approach in [152]. Let $\{\mathbf{x}_i^k \in \mathbb{R}^D, i = 1, \cdots, n_k\}$ be a set of points on instance manifold $\mathcal{M}^k$. Let $\{\mathbf{z}_i^k \in \mathbb{R}^e, i = 1, \cdots, n_k\}$ be the corresponding points on the unified manifold $\mathcal{U}$,

We learn a regularized mapping functions $\gamma^k(\cdot) : \mathbb{R}^e \rightarrow \mathbb{R}^D$, which maps from $\mathcal{U}$ to each instance manifold $\mathcal{M}^k$. The mapping can be written in the following matrix form.

$$\gamma^k(\mathbf{z}) = \mathbf{C}^k \psi(\mathbf{z}) \tag{5.3}$$

where $\mathbf{C}$ is a $D \times n$ matrix, the vector $\psi(\mathbf{z}) = [k(\mathbf{z}, \mathbf{w}_1), \cdots, k(\mathbf{z}, \mathbf{w}_N)]^\top$ represents a non-linear kernel map from the embedded representation to a kernel induced space, given a set of RBF centers $\{\mathbf{w}_1, \cdots, \mathbf{w}_N\}$. $k(\cdot, \cdot)$ is an RBF kernel. The solution of Eq 5.3 is shown in [108] to have a closed form solution:

$$\mathbf{C}^{k^\top} = (\mathbf{A}^{k^\top} \mathbf{A}^k + \lambda \mathbf{G})^{-1} \mathbf{A}^{k\top} \mathbf{X}^{k^\top}, \tag{5.4}$$

where $\mathbf{A}^k$ is a $n_k \times n$ matrix with $\mathbf{A}_{ij} = k(\mathbf{z}_i, \mathbf{w}_j), i = 1, \cdots, n_k, j = 1 \cdots n$ and $\mathbf{G}$ is a $n \times n$ matrix with $\mathbf{G}_{ij} = k(\mathbf{w}_i, \mathbf{w}_j); i, j = 1 \cdots n$. $\mathbf{X}^k$ is the $n_k \times D$ data matrix for manifold instance k. Solution for $\mathbf{C}$ is guaranteed under certain conditions on the basic functions used.

5.3.2 Supervised Manifold Embedding

The second phase of our framework is to learn *good*[2] low-dimensional embedding for the parameterizations. Consider Eq 5.3, if we ignore the superscript k, and if we have a test image $\mathbf{x}_{test}$, then the objective is to find the best $\mathbf{C}^*$ and $\mathbf{z}^*$ that minimizes the objective function: $\delta(\mathbf{x}_{test} - \gamma^k(\mathbf{C}^*, \psi(\mathbf{z})^*))$. However, the generated parameterization ($\mathbf{C}$'s from Eq 5.4) belong to a high-dimensional space. This makes solving the objective hard and not robust. Therefore, there is a need to find a suitable low-dimensional latent space for those parameterizations.

[2]Good in sense that balance between encoding the spatial relationship between points in the original space and separate points with different labels

Subspace analysis is used in [36] to obtain a latent representation of the manifold parameterization space. However, these approaches do not benefit from available class labels. Alternatively we propose a supervised way to achieve a low-dimensional latent manifold parameterization space, which benefits from the class labels. We use Partial Least Squares (PLS)[146, 114, 9].

While PCA tends to keep most of the variance of the input space, and LDA tends to maximize the interclass and minimize the intraclass distances, PLS is a supervised version of PCA. PLS compromises by generating orthogonal components (in the latent space) using the already existing correlations between observation variables (in the input space) and corresponding labeling, while keeping most of the variance of the points in the input space. Good interpretation for PLS and its relationship with PCA can be found in [83, 9]. PLS has been proven to be useful in situations where the dimensionality of the input space exceeds significantly the number of observations (sparse case) and/or multicollinearity[3] is high among the explanatory variables. PLS embeds the input points X into low-dimensional latent space $\mathbf{T}$, so that it satisfies: $\arg\min_T \|\mathbf{XW} - \mathbf{T}\|$ and $\arg\max_T cov(\mathbf{T}, \mathbf{y})$. $\mathbf{W}$ is the learned projection matrix and $\mathbf{y}$ is the set of labels.

K-PLS:

Using Kernel PLS (K-PLS)[114] is more convenient in the case of coefficient matrices ($\mathbf{C}$'s).

First, we need to define a kernel on the parameterization space: since $\mathbf{C}$ is $D \times N$ matrix, and $D \gg N$, then $\mathbf{C}$ represents N-dimensional subspace in $\mathbb{R}^D$. Therefore, we can use Cosine-Similarity kernel (CSK) or Grassmannian kernels [55]. In this work, we use CSK, for its efficiency, defined by

$$K(\mathbf{C}_i, \mathbf{C}_j) = \frac{tr(\mathbf{C}_i \mathbf{C}_j^\top)^2}{\|\mathbf{C}_i\|_F \|\mathbf{C}_j\|_F}, \tag{5.5}$$

where $\|\cdot\|_F$ is matrix Frobenius norm.

Given the instance manifold parameterization $\mathbf{C}^k$ and parameterization kernel and label y^k, we use K-PLS for embedding parameterizations space into low-dimensional latent space. The points in this latent space satisfy the two objectives of the PLS.

[3] Multicollinearity refers to a situation in which two or more explanatory variables in a multiple regression model are highly linearly related.

K-PLS maps the point $\mathbf{C}^k$ to latent points $\{\mathbf{t}^k \in \mathbb{R}^m$, for each $k = 1 \cdots N$, by

$$\mathbf{t} = K(\mathbf{C}, \cdot)\mathbf{W} \tag{5.6}$$

where $\mathbf{W}$ is a non-linear projection matrix. Details of the K-PLS algorithm can be found in [114]. Like PCA, the choice of latent dimensionality (m) is a crucial step, since small and large values of m lead to under-fitting and over-fitting of the training data, respectively.

5.3.3 Nonlinear Mapping from Latent Space to Parameterization Space

Almost all linear DR techniques, such as (linear) PLS and PCA, provide mapping from latent space to input space as well as from input space to latent space. In contrast, almost all nonlinear DR techniques do not provide mapping from latent to input space. K-PLS, as nonlinear DR, does not provide a closed form mapping from the latent space $\mathbb{T}$ to the parameterization space $\mathbb{C}$. However, we need this reverse mapping to complete the generative model in Eq 5.1 and illustrated in Figure 5.1. In this work, we learn non-linear Gaussian RBF mapping $\beta : \mathbb{T} \rightarrow \mathbb{C}$

$$\mathbf{C} = \beta(t) = \mathcal{A} \times_1 \phi(\mathbf{t}) \tag{5.7}$$

where $\mathbf{C}$ is an $n \times D$ matrix, $\mathbf{t}$ is a m-D vector and $\mathcal{A}$ is a $h \times n \times D$ tensor, and

$$\phi(\mathbf{t}) = K(\mathbf{t}, \cdot) = exp(\sigma \left\|\mathbf{t} - \mathbf{u}_i\right\|) \tag{5.8}$$

where $\mathbf{u}_i; i = 1 \cdots h$ are the centers of the RBF kernel. Those centers are the h-means of the training points in the latent space.

The tensor $\mathcal{A}$ is computed as follows. Let $\mathbf{c}$ (nD-dimensional vector) is the vectorization of $\mathbf{C}$ ($n \times D$ matrix).

$$\mathbf{c} = \mathbf{A}^\top \phi(\mathbf{t})$$

Then we can find the mapping $\mathbf{A}$ by the same way we computed $\mathbf{C}$ in Eq 5.4. Finally, $\mathcal{A}$ ($h \times n \times D$ tensor) is the reshaped version of $\mathbf{A}$ ($h \times nD$ matrix).

Substituting Eq 5.7 in Eq 5.3 results in the generative model: $\mathbf{x}_{predicted} = \mathcal{A} \times_1 \phi(\mathbf{t}) \times_2 \psi(\theta)$.

5.4 Inference

This section shows how to use the underlining generative model for inferring (v^*, y^*) that best fit a given test image ($\mathbf{x}_{test}$). This is done in two steps: first, find $(\mathbf{t}^*, \theta^*)$ that optimizes Eq 5.9, then analyze $(\mathbf{t}^*, \theta^*)$ to get (v^*, y^*). In this work, we use two distance metrics to measure the difference between the test image and the predicted image $\delta\{\mathbf{x}_{test}, \mathbf{x}_{predict}\}$: Euclidean distance ($\|\mathbf{x}_{test} - \mathbf{x}_{predict}\|$) and Normalized Cross-Correlation (NCC) ($1 - \frac{\mathbf{x}_{test}^\top \mathbf{x}_{predict}}{\|\mathbf{x}_{test}\|\|\mathbf{x}_{predict}\|}$). We adopt two optimization techniques for solving Eq 5.9: gradient-based method and sampling-based methods.

5.4.1 Optimization

In this section, we need to show how we solve the objective function:

$$(\mathbf{t}^*, \theta^*) = \arg\min_{(\mathbf{t},\theta)} \{\delta\{\mathbf{x}_{test}, \mathbf{x}_{predict}(\mathbf{t}, \theta)\}\} \tag{5.9}$$

Gradient-based Method

Our gradient-based optimization uses the second order BFGS quasi-Newton optimizer with cubic polynomial line search for optimal step size selection [5].

To use this algorithm, for each value of $\mathbf{t}$ and θ, we need to compute the distance: $\delta\{\mathbf{x}_{test}, \mathbf{x}_{predict}(\mathbf{t}, \theta)\}$, and its derivative. The derivation is shown in section 5.4.3.

Sampling-based method

In this method we use MCMC sampling [12], similar to the unsupervised approach in the last chapter, to solve Eq 5.9. We use simulated annealing [1] to enhance resampling of particles. The details of the algorithm are provided Algorithm 2.

5.4.2 Classification

At this point, we have low-dimensional category and pose representations: $\mathbf{t}^*$ and θ^*. We wish to infer the label for object category y^* and for pose v^*.

Algorithm 2 Sampling Algorithm

Initialize particles by the M-means of the available latent points.
repeat
  for Every particle $\mathbf{p}$ **do**
    Generate N new samples for every old particle $\mathbf{p}$: ▷ Use Gaussean as the proposal distribution for generating new samples.
    for $\nu \leftarrow 1 \rightarrow N$ **do**
      $\mathbf{p}_d^\nu \sim \mathcal{N}(\mathbf{p}_d, \sigma)\ \forall d = 1 \cdots m+1$ ▷ m is the latent space dimensionality, and $+1$ for θ. σ shrinks over iterations
      $E(\mathbf{p}^\nu) \leftarrow$ distance from Eq 5.9. ▷ Energy value of every particle $E(\mathbf{p}^\nu)$
      $A^\nu = E(\mathbf{p})/E(\mathbf{p}^\nu)$ ▷ Acceptance ratio
      $u \leftarrow \mathbf{Uniform}(0, 1)$
      if $A^\nu > u$ **then**
        Accept $\mathbf{p}^\nu$
      else
        Reject $\mathbf{p}^\nu$, and set $\mathbf{p}^\nu \leftarrow \mathbf{p}$
      end if
    end for
  end for
  Re-sample M particles from the set of new particles. Use $e^{-\gamma E(\mathbf{p})}$ as a the re-sampling weights.
until All selected particles are close by ▷ Convergence happens when the variance of the newly selected particles is small $var \leq \epsilon$

For pose, we use a *nearest neighbor* classification function g. If the actual pose label is discrete, then we use $v^* = k - NN(\theta^*)$ and if not, $v^* = \theta^*$.

For category, we use Regression for classification (RfC), *i.e.* we use the regression results of K-PLS [114]

$$y^* = {t^*}^\top T^\top \mathbf{y} \tag{5.10}$$

where T is the set of embedded training points in the K-PLS latent space (Eq 5.6), and $\mathbf{y}$ is the corresponding labels of the training points.

5.4.3 Gradients-based inference details

In this section, we show the gradients of the objective function Eq 5.9, for both distance metrics Euclidean and Normalized Cross-Correlation.

Euclidean distance:

$$J(\mathbf{t}, \theta) = \delta\{\mathbf{x}_{test}, \mathbf{x}_{predicted}\} = \|\mathbf{x}_{test} - \mathbf{x}_{predict}\|$$

Where $\mathbf{x}_{test}$ is the test image and $\mathbf{x}_{predict}$ is the predicted image based on $\mathbf{t}$ and θ defined in Eq 5.1. Then the derivatives will be

$$\frac{\partial J}{\partial t} = -2\frac{\partial \phi(t)}{\partial t}(\mathcal{A} \times_2 \psi(\theta))^\top(\mathbf{x}_{test} - \mathbf{x}_{predict}) \tag{5.11}$$

where $\phi(t)$ defined in Eq 5.8 and its Jacobian matrix is

$$\frac{\partial \phi(t)}{\partial t} = 2\sigma\left[\frac{(t-u_1)}{\|t-u_1\|}e^{\sigma\|t-u_1\|}, \frac{(t-u_2)}{\|t-u_2\|}e^{\sigma\|t-u_2\|}\cdots\frac{(t-u_h)}{\|t-u_h\|}e^{\sigma\|t-u_h\|}\right].$$

And

$$\frac{\partial J}{\partial \theta} = -2\frac{\partial \psi(\theta)}{\partial \theta}(\mathcal{A} \times_2 \phi(t))^\top(\mathbf{x}_{test} - \mathbf{x}_{predict}) \tag{5.12}$$

where

$$\frac{\partial \psi(\theta)}{\partial \theta} = \frac{\partial \psi(z)}{\partial \theta}\frac{\partial z}{\partial \theta}$$

, where $z = [cos(\theta)\, sin(\theta)]^\top$ and $\psi(z)$ defined (in Eq 5 in the paper) as

$$\psi((z)) = exp(\sigma\,\|\mathbf{z} - \mathbf{w}_i\|) \tag{5.13}$$

where $\mathbf{w}_i; i = 1 \cdots n$ represent fixed equi-spaced centers on the unit circle. and its Jacobian matrix

$$\frac{\partial \psi(z)}{\partial z} = 2\sigma\left[\frac{(z-w_1)}{\|z-w_1\|}e^{\sigma\|z-w_1\|}, \frac{(z-w_2)}{\|z-w_2\|}e^{\sigma\|z-w_2\|}\cdots\frac{(z-w_n)}{\|z-w_n\|}e^{\sigma\|z-w_n\|}\right]$$

, and

$$\frac{\partial z}{\partial \theta} = [-sin(\theta)\, cos(\theta)]^T$$

Normalized Cross-Correlation (NCC):

$$J(\mathbf{t}, \theta) = \delta\{\mathbf{x}_{test}, \mathbf{x}_{predicted}\} = 1 - \frac{\mathbf{x}_{test}^\top \mathbf{x}_{predict}}{\|\mathbf{x}_{test}\|\,\|\mathbf{x}_{predict}\|}$$

Where $\mathbf{x}_{test}$ is the test image and $\mathbf{x}_{predict}$ is the predicted image based on $\mathbf{t}$ and θ defined in Equation **1** in this thesis. For convenience, lets use *s* for test and *p* for predict. Then we have

$$\frac{\partial J}{\partial t} = -\frac{\partial \mathbf{x}_p}{\partial t}\left(\frac{\mathbf{x}_s}{\|\mathbf{x}_s\|\|\mathbf{x}_p\|} + \frac{J\,\mathbf{x}_p}{\|\mathbf{x}_p\|^2}\right) \tag{5.14}$$

where the Jacobian matrix $\frac{\partial \mathbf{x}_p}{\partial t}$ is defined as:

$$\frac{\partial \mathbf{x}_p}{\partial t} = \frac{\partial \phi(t)}{\partial t}(\mathcal{A} \times_2 \psi(\theta))^\top$$

where $\phi(t)$ defined in Equation **8** in this thesis, and its Jacobian matrix is

$$\frac{\partial \phi(t)}{\partial t} = 2\sigma \left[\frac{(t-u_1)}{\|t-u_1\|} e^{\sigma\|t-u_1\|}, \frac{(t-u_2)}{\|t-u_2\|} e^{\sigma\|t-u_2\|} \cdots \frac{(t-u_h)}{\|t-u_h\|} e^{\sigma\|t-u_h\|} \right].$$

And similarly

$$\frac{\partial J}{\partial \theta} = -\frac{\partial \mathbf{x}_p}{\partial \theta} \left(\frac{\mathbf{x}_s}{\|\mathbf{x}_s\|\|\mathbf{x}_p\|} + \frac{J\ \mathbf{x}_p}{\|\mathbf{x}_p\|^2} \right) \tag{5.15}$$

where the Jacobian $\frac{\partial \mathbf{x}_p}{\partial \theta}$ is defined as:

$$\frac{\partial \mathbf{x}_p}{\partial \theta} = \frac{\partial \psi(\theta)}{\partial \theta}(\mathcal{A} \times_1 \phi(t))^\top$$

where

$$\frac{\partial \psi(\theta)}{\partial \theta} = \frac{\partial \psi(z)}{\partial \theta} \frac{\partial z}{\partial \theta}$$

, where $z = [cos(\theta)\ sin(\theta)]^\top$ and $\psi(z)$ defined (in Eq 5 in this thesis) as

$$\psi((z)) = exp(\sigma\ \|\mathbf{z} - \mathbf{w}_i\|) \tag{5.16}$$

where $\mathbf{w}_i; i = 1 \cdots n$ represent fixed equi-spaced centers on the unit circle.

and its Jacobian matrix

$$\frac{\partial \psi(z)}{\partial z} = 2\sigma \left[\frac{(z-w_1)}{\|z-w_1\|} e^{\sigma\|z-w_1\|}, \frac{(z-w_2)}{\|z-w_2\|} e^{\sigma\|z-w_2\|} \cdots \frac{(z-w_n)}{\|z-w_n\|} e^{\sigma\|z-w_n\|} \right]$$

, and

$$\frac{\partial z}{\partial \theta} = [-sin(\theta)\ cos(\theta)]^T$$

5.4.4 Hierarchical Model

Dealing with large datasets containing fine-grained classification (*i.e.* large sets of object images with large visual and semantic similarity - *e.g.* tableware, fruits, *etc.* found in the RGBD dataset [77]) is a challenging problem. To solve this we extend our framework to a hierarchical model. The hierarchical model performs recursive spectral clustering [31] to identify super-categories of classes in the data (see fig. below). On each of these clusters, we use the same

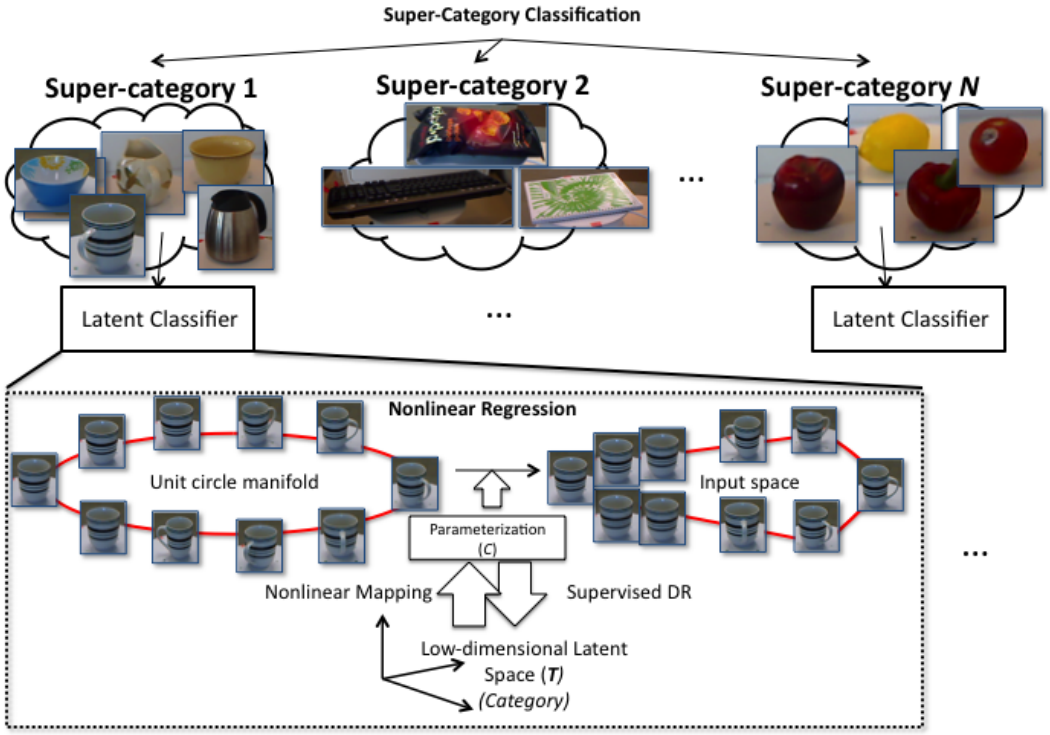


Figure 5.2: The overall hierarchical model. Recursive clustering is used to identify super-categories of similar objects. Our view-invariant latent generative model is then applied to each individual cluster to perform category recognition and pose estimation.

generative model described to learn the latent space and perform inference on the category label and pose. Each cluster has its own independent latent embedding. This enables more efficient classification due to the less number of object classes in each cluster.

To infer the category and pose of a test object, top-down inference is performed on the hierarchy. The super-category cluster is identified using a Support Vector Machine (SVM) on the raw training HoG descriptors. Once the cluster is identified the inference takes place over the latent space for that specific cluster.

5.5 Experiments and Results

We experiment on three datasets: 3DObjects [119], RGB-D [77] and EPFL [101]. 3DObjects consists of 10 classes of table-top objects as well as larger objects, such as, cars and bicycles. The objects are captured at multiple viewpoints. The RGB-D dataset is another multi-view

dataset but consists of a much larger set of 300 table-top objects. This is a challenging dataset due to its size and fine-grained categories. EPFL has 20 different car image sequences as the cars rotate on a platform.

We use HoG features [24] as our input image representation and we use a unit circle as the unified manifold, i.e, $\theta \in [0, 2\pi]$.

For inference (Section 5.4), we use gradient-based optimization with Euclidean distance and sampling-based optimization with NCC distance metric. The results here are reported based on these best configurations.

5.5.1 3DObjects

We show the experimental results of our work applied to 3D Objects dataset and compare our results to [119, 152]. The 3D Objects dataset has images of 10 objects categories. Every category has 10 different instances, differing in brand, color and shape. Every instance has images captured at 3 heights and 3 scales. Every image sequence has 8 poses covering all views: back, back-right, right, front-right, front, front-left, left, back-left.

We show our results for 2 different configurations: 1) 8 classes excluding the farthest scale. 2) All classes are included (included in supplementary material). Pose accuracy is reported even if the object is incorrectly classified. We use the first configuration for comparison purposes because this is the configuration in prior work ([119],[152]). For all experiment, we train over 7 instances (brands/shapes) and test on remaining 3 brands. The final results are the average of several folds.

Table 5.1,5.2 and 5.3 show both our sampling and gradient based methods compared against the baselines. We can see an improvement of about 38% in pose recognition (Table 5.2). Table 5.3 shows the comparison of the gradient optimization and sampling optimization in both category and pose recognition using different classification techniques (as described in Section 5.4.2). For pose, we computed the percentages of poses that are less than 22.5° and 45° away from groundtruth. The gradient optimization technique is more robust.

Table 5.1 shows the recognition rate of every category compared to the baselines. We used a 13-dimensional K-PLS latent space. Inference can be done efficiently on very low-dimensional latent spaces instead of the very high-dimensional feature spaces. From the table, it is clear

	Sampling	Gradient	Baseline[152]	Baseline[119]
Average	88.44%	**89.50%**	80.07%	75.65%
Bicycle	99.52%	99.54%	99.79%	81.00%
Car	96.45%	97.22%	99.03%	69.31%
Cellphone	98.10%	**99.54%**	66.74%	76.00%
Iron	88.10%	**90.28%**	75.78%	77.00%
Mouse	50.72%	54.46%	48.60%	86.14%
Shoe	87.56%	**89.35%**	81.70%	62.00%
Stapler	96.08%	**97.63%**	82.66%	77.00%
Toaster	**93.30%**	90.28%	86.24%	74.26%

Table 5.1: Recognition rate for each Category. We show improved accuracy for most categories

	Sampling	Gradient	Baseline[119]
Average	74.94%	79.19%	57.46%
Front	**75.24%**	74.27%	64.00%
Front-left	66.34%	**73.66%**	40.40%
Left	76.81%	**80.68%**	47.00%
Left-back	76.33%	**88.89%**	62.00%
Back	**80.00%**	78.05%	53.54%
Back-right	73.91%	**84.54%**	71.72%
Right	81.64%	**83.09%**	57.00%
Right-front	67.31%	**70.10%**	64.00%

Table 5.2: Recognition rate of each pose. These results are for the first experiment configuration on the 3DObject dataset.

	Gradient	Sampling
Category		
Linear-SVM	**89.50**	88.44
SVM	-	85.94
RfC	**88.65**	88.31
k-NN	**89.68**	88.63
Pose		
$<45.0^{\circ}$	**83.25**	80.13
$<22.5^{\circ}$	**79.19**	74.94

Table 5.3: Category recognition and pose estimation accuracy (%) on the 3DObjects dataset using our approach

that our model outperforms state-of-the-art results in most categories. We achieve an overall increase of up to about 10% in accuracy using sampling-based inference and 11.8% using gradient-based inference.

5.5.2 EPFL Cars

EPFL dataset has 20 different car sequences. Each sequence has around 120 image frames of the same car. The ground truth of this dataset provides the number of frames in every sequence, the frontal frame number, the number of frames that make up a whole viewing circle and the bounding boxes.

To compare with previous work ([101],[137] and [139]), we use the same experiment configuration, we train over the sequences of first ten cars (cars from $1 \rightarrow 10$) and test over the last ten cars (cars from $11 \rightarrow 20$). In this experiment test images are for car instances that are not present in the training data. Our framework finds the closest instance to the query car instance and estimates the pose.

Table 5.6 shows the results of this configuration compared with previous work. The result are: 90.81% have error less than 22.5 degrees compared to 41.69% reported in [101], 70.31% reported in [139] and 78.1% reported in [137]. Which means that we achieved 16.27% improvement. This shows the power of our framework for continuous pose estimation. We also

Method	MAE	AE<22.5°	AE<45°
[101] - 50% split	46.48	41.69	71.20
[137] - 50% split	47.40	78.10	79.70
[139] - 50% split	40.60	70.31	80.75
ours - 50% split	**22.34**	**90.81**	**90.81**
[139] - leave one out	35.87	63.73	76.84
ours - leave one out	**11.81**	**95.13**	**95.13**

Table 5.4: Comparison of our results with state-of-the-art baselines on the EPFL multi-view car dataset. For the baselines we report the best results of all configurations

Method	MAE	AE<22.5°	AE<45°
Train over $1 \rightarrow 10$	16.35	92.22	92.22
Train over $11 \rightarrow 20$	20.35	90.81	90.81
Train over odd numbers	18.34	91.64	91.64
Train over even numbers	16.35	92.22	92.22
Average	17.76	91.75	91.75

Table 5.5: Results for our framework on four different splits.

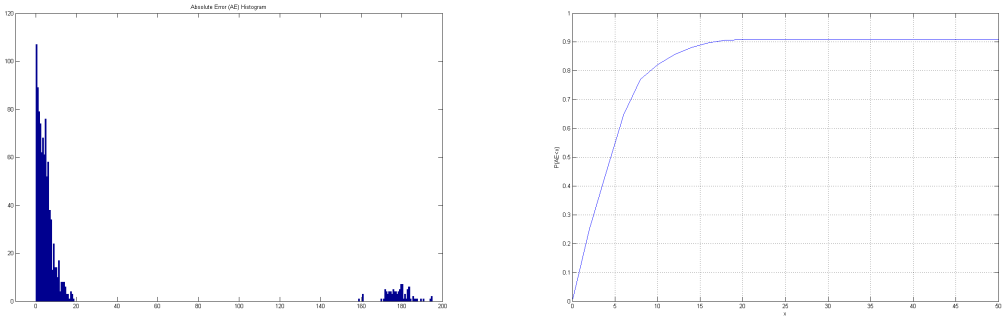


Figure 5.3: In EPFL, Statistics on 50% split (train on cars $1 \rightarrow 10$, test on cars $11 \rightarrow 20$): (left) histogram of Absolute Error (AE), (right) probability of $P(AE < x)$ for different values of x.

reduce the mean absolute error (MAE) by about 45%. Leave-one-out has significant improvement over 50% split, the reason behind this is that our algorithm has a wider space (more car instances) to search for the closest point to the query car. More discussion of the results and figures are in the supplementary material.

Figure 5.5 shows statistics for 50% split. Figure 5.5(**left**) shows histogram for Absolute Error (AE). We can see in this figure that except for the frames which has been has been estimated in reflection, all AE is less than $20°$. From this figure, we see that the 9.2% estimation error comes from reflection (estimating the pose by its opposite value ($estmated = actual + 180° + \epsilon$)). This error happen for cars which has high symmetry.

While Figure 5.5(**right**) shows $Pr\{AE < x\}$ for $x \in [0, 50]$ which give us more accurate overview. In this figure, we can see that saturation happen at $x = 20°$ and also we can see that $Pr\{AE < 10\} = 0.80$ which is the same accuracy that [139] achieved for $x = 45°$ which is significant improvement.

One 50% split may not draw the full picture of the framework performance. More accurate

Method	MAE	AE<22.5°	AE<45°
[101] - 50% split	46.48	41.69	71.20
[137] - 50% split	47.40	78.10	79.70
[139] - 50% split	40.60	70.31	80.75
ours - 50% split	**22.34**	**90.81**	**90.81**
[139] - leave one out	35.87	63.73	76.84
ours - leave one out	**11.81**	**95.13**	**95.13**

Table 5.6: Comparison of our results with state-of-the-art baselines on the EPFL multi-view car dataset. For the baselines we report the best results of all the configurations the authors used.

results are shown in Table 5.7. In these two items, we repeated the 50% split experiment, but for four different *fair* splits. One split is trained over cars $1 \rightarrow 10$ and test over the rest. Other split is trained over cars $11 \rightarrow 20$, the third one is trained over the odd numbered cars and the last one trains over the even numbered cars. Figure 5.4 shows the accuracy on different sequences of the dataset.

Pose regression (Instance independent)

To compare with previous work ([101],[137] and [139]), we use the same experiment configuration, we train over the sequences of first ten cars (cars from $1 \rightarrow 10$) and test over the last ten cars (cars from $11 \rightarrow 20$). In this experiment test images are for car instances that are not present in the training data. This is why we call it instance independent. Our framework finds the closest instance to the query car instance and estimates the pose. Table 5.6 shows the results of this configuration compared with previous work. The result are: 90.81% have error less than 22.5 degrees compared to 41.69% reported in [101], 70.31% reported in [139] and 78.1% reported in [137]. Which means that we achieved 16.27% improvement. This shows the power of our framework for continuous pose estimation. In the same time, we reduces the mean absolute error (MAE) by about 45%.

Leave one out has significant improvement over 50% split, the reason behind that is our algorithm has a wider space (more car instances) to search for the closest point to the query car.

Figure 5.5 shows statistics for 50% split. Figure 5.5(**a**) shows histogram for Absolute Error (AE). We can see in this figure that except for the frames which has been has been estimated in

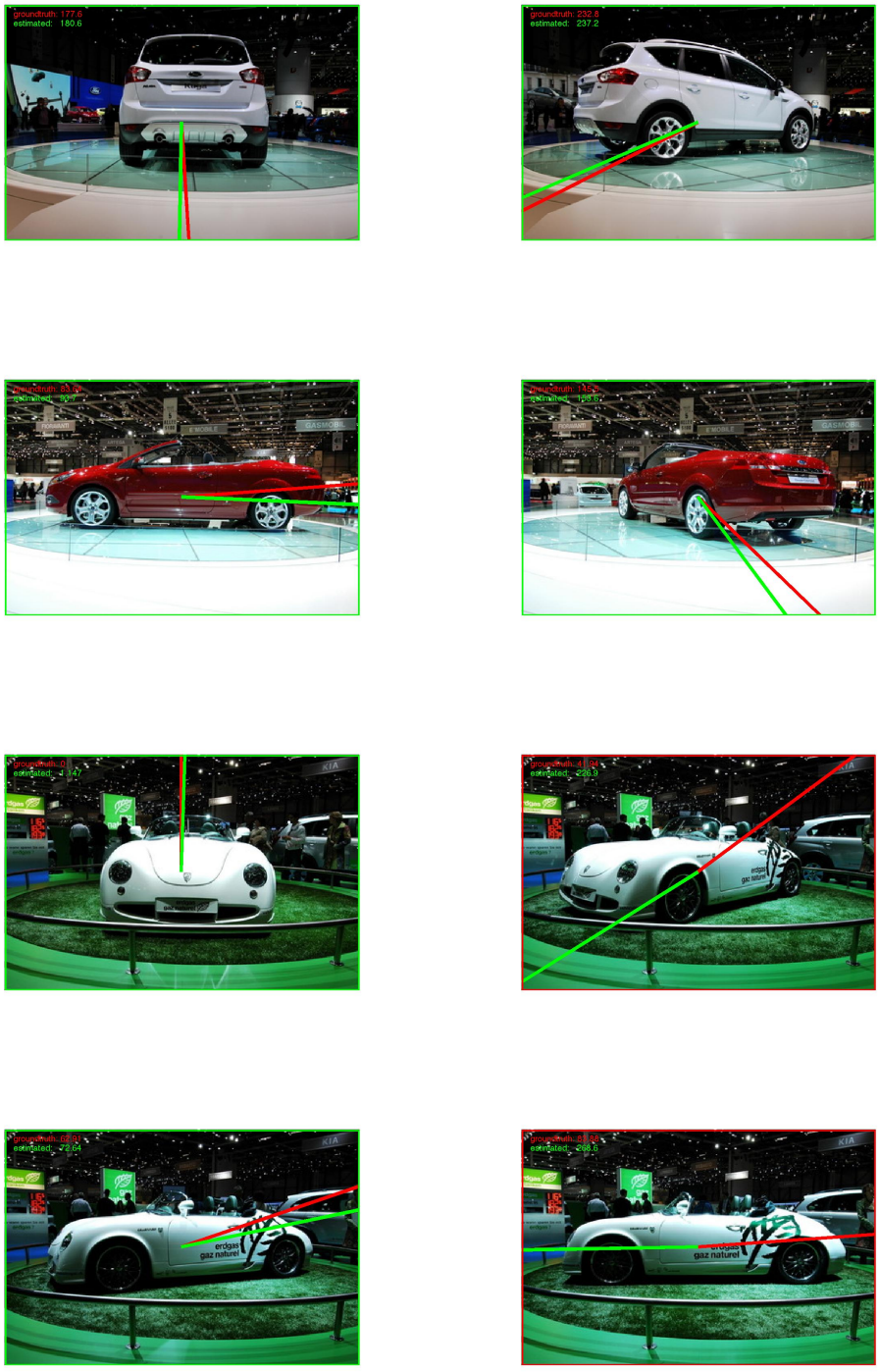
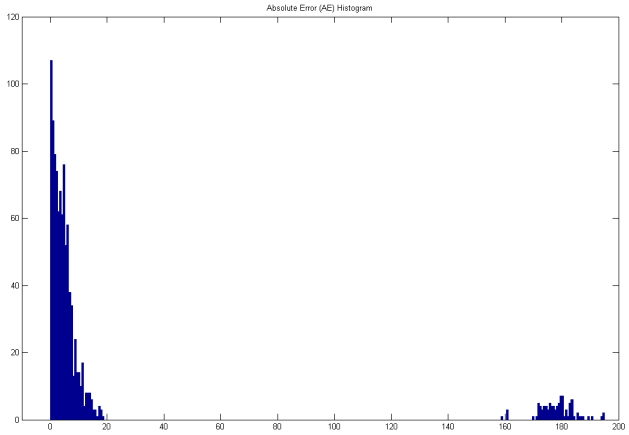
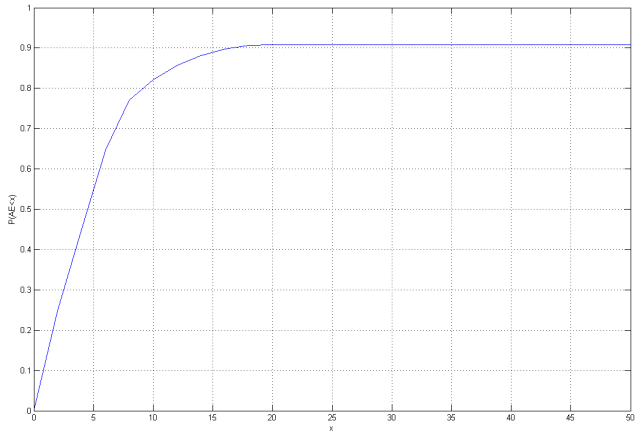


Figure 5.4: Snapshots of the video demo in the supplementary material. Color code: red: ground truth pose, green: estimated pose, frames highlighted green if error less than 22.5 degrees (red otherwise)



(a)



(b)

Figure 5.5: In EPFL, Statistics on 50% split (train on cars $1 \rightarrow 10$, test on cars $11 \rightarrow 20$): (a) histogram of Absolute Error (AE), (b) probability of $P(AE < x)$ for different values of x.

reflection, all AE is less than $20°$. From this figure, we see that the 9.2% estimation error comes from reflection (estimating the pose by its opposite value ($estmated = actual + 180° + \epsilon$)). This error happen for cars which has high symmetry.

While Figure 5.5(**b**) shows $Pr\{AE < x\}$ for $x \in [0, 50]$ which give us more accurate overview. In this figure, we can see that saturation happen at $x = 20°$ and also we can see that $Pr\{AE < 10\} = 0.80$ which is the same accuracy that [139] achieved for $x = 45°$ which is significant improvement.

One 50% split may not draw the full picture of the framework performance. More accurate

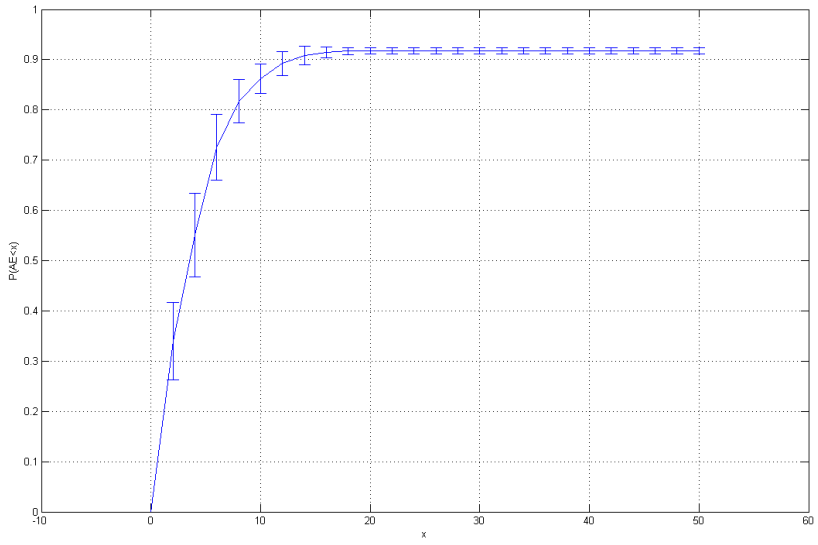


Figure 5.6: In EPFL, Statistics on four 50% splits. The probability is shown along with standard deviation across the splits

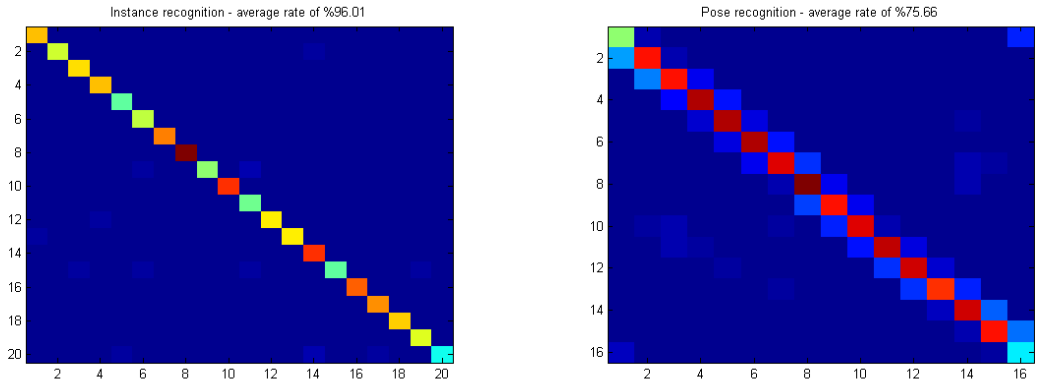
Method	MAE	AE<22.5°	AE<45°
Train over $1 \rightarrow 10$	16.35	92.22	92.22
Train over $11 \rightarrow 20$	20.35	90.81	90.81
Train over odd numbers	18.34	91.64	91.64
Train over even numbers	16.35	92.22	92.22
Average	17.76	91.75	91.75

Table 5.7: Results for our framework on four different splits.

results are shown in Table 5.7 and Figure 5.6. In these two items, we repeated the 50% split experiment, but for four different *fair* splits. One split is trained over cars $1 \rightarrow 10$ and test over the rest. Other split is trained over cars $11 \rightarrow 20$, the third one is trained over the odd numbered cars and the last one trains over the even numbered cars. Figure 5.6 shows $Pr\{AE < x\}$, along with standard deviation across all splits. Table 5.7 shows some statistics on this experiment.

Category and pose classification (Instance dependent)

For testing the category and pose in this dataset, we divide every sequence into four overlapping sequences. Each car has four sequences. We train over one sequence for every car and test over all images in the remaining three sequences. In this experiment all cars are presented in the



(**a**) (**b**)

Figure 5.7: (**a**) shows category confusion (Average of 96.01), (**b**) shows pose confusion (Average of 75.66).

training data (instance dependent). Figure 5.7(**a**) shows the instance (car instance) confusion matrix , and Figure 5.7(**b**) shows the pose classification confusion matrix. For pose estimation, to get the confusion, we classify every pose by the nearest neighbor based on 16 different poses covering the whole viewing circle. The final result of this experiment is 96.01% car instance recognition and 75.66% pose estimation accuracy. We also report that 97.38% of the results have error less than 22.5° and 97.79% have error less than 45°. The reason that makes the pose classification has lower accuracy that for specific image, the estimated pose could be very close to the actual pose but each value falls in different bin. Moreover, $P(AE < 22.5^\circ) = 97.79\%$, which clarify that the estimation error is very small.

5.5.3 RGB-D Dataset

In order to deal with the large number of objects in the RGB-D dataset, we used the hierarchical extension of our model. Referring to fig. 2, we can see that similarly shaped objects cluster together, *e.g.*, super-category 1 shows round objects, super-category 2 shows flat objects and super-category N shows spherical objects. The accuracy of the top level classification which classifies objects into the different semantic clusters of objects is about 95.6%.

The results are reported in table 5.8. The same accuracy metrics were used as the baseline approaches. We outperform the two baselines in pose estimation over all test images (whether or not they were classified correctly). This is shown in the Avg. Pose (All) column. For the

Method	Category	Avg. Pose (C)	Avg. Pose (All)
ours (visual only)	85.00	77.31	73.78
visual only baseline [152]	92.00	80.01	61.57
visual + depth baseline [78]	94.30	56.80	53.50

Table 5.8: Category recognition and pose estimation accuracy (%) on the RGBD dataset. We report the RGB-only accuracy of [152]. [78] only report their multi-modal RGB+D accuracy.

average pose accuracy over the correctly classified images (C), we achieve better accuracy than [78] but slightly less than [152]. For category recognition we achieve less accuracy than the two baselines. For [78] both visual and depth features are used for classification. Our approach only uses visual information. The hierarchical clustering approach aids the pose estimation for finding optimal poses for similar objects even though they are misclassified. For example, similar objects with handles such as mugs and pitchers have similar pose variations due to the presence of the handle.

In table 5.8, although the recognition accuracy is less than [152], the overall pose is significantly better. Even with wrong categorization, the method inferred a good representation in the category latent space (which can be a combination of multiple-categories), which enabled the correct pose estimation. The lower recognition accuracy can also be due to the classification scheme used after inference which can be improved.

5.6 Discussion

The proposed model is supervised and nonlinear, in contrast to linear-unsupervised previous approaches (including the approach seen earlier in this chapter). The use of supervision to achieve the category latent space is fundamental to retain the discriminative ability, while reducing the dimensionality for better inference in this space. This resulted in significantly better results in 3DObjects (recognition + pose) and EPFL-Cars (pose). While the use of supervision to achieve latent spaces has been studied before, this has not been addressed in the context of the space of manifold deformations as presented here. For instance, using LDA in the coefficient space gives worse results than unsupervised (in 3DObjects the recognition rate drops to 61%). Adding supervision is not trivial for high-dimensional parameterization spaces, and results in a nonlinear latent space, which mandates the use of a nonlinear model as we have presented. The nonlinearity in the model requires developing suitable inference methods. We

compare two inference methods.

Unfortunately there is no large multi-view dataset to really evaluate scalability. Category-specific models for pose estimation (*e.g.* [95, 121]) are not scalable, since one model per category is needed, and these models do not allow sharing knowledge among similar classes. In contrast, we aim for a model that solves simultaneously for category and pose, where there is one common representation for all categories, hence the scalability potentials. Interestingly, we achieved better pose estimation over many category-specific models, even though our model combines all objects.

Compared to [152], our approach is scalable in two fundamental ways. First, In [152], the category latent space was the subspace spanned by all training data, which is still high dimensional for inference. [152] used 300 dimensional space for the RGB-D dataset. We only use 20 dimensions and achieve comparable results. In addition, [152] used a k-NN classifier because of the use of the full subspace. This is not scalable and not robust, therefore, supervised DR is needed to achieve a low dimensional representation that retains the discriminative power. Second, we propose a hierarchical Model, which is essential for scalability.

Supervision and nonlinearity are coined together in our use of KPLS. Comparing our sampling results with [23] (also used sampling, although no algorithm was provided) shows the value of the unsupervised-linear/supervised-nonlinear (88.4% vs 80% Table 5.1).

5.7 Conclusion

We present a novel framework for recognizing object category and pose estimation. The framework uses a generative model that is based on homeomorphic manifold analysis, supervised manifold embedding into a latent space and nonlinear mapping. The advantage of our model is that inference is moved from the very high-dimensional feature space to two low-dimensional orthogonal pose and category spaces, which makes the inference more accurate and computationally easier. We also incorporate this model into a hierarchical structure to deal with large fine-grained classification problems. We show theoretical basis of our framework and compare our results with the state-of-the-art. We show that our framework both achieves higher performance than state-of-the-art in some configurations and is comparable in others.

Chapter 6

Joint Object Recognition using Convolutional Neural Networks

This work is focused on studying the view-manifold structure in the feature representation space implied by the different layers of Convolutional Neural Networks (CNN). There are several questions that this work aims to answer: Do the representations encoded in the layers of a trained CNN achieve viewpoint invariance? If so, how is viewpoint invariance achieved? Is it achieved by collapsing the view manifolds, or separating them while preserving and untangling them? At which layer is view invariance achieved? How can the structure of the object-view manifold at each layer of a deep convolutional neural network be quantified and analyzed experimentally? How does fine-tuning of a pre-trained CNN on a multi-view dataset affect the representation at each layer of the network? In order to answer these questions we propose a methodology to quantify the deformation and degeneracy of view manifolds in CNN layers. We apply this methodology and report interesting results in this work that answer the aforementioned questions.

We first analyze the layers of a state-of-the-art CNN and present our findings. Starting from section 6.6, based on the findings of our analysis, we investigate adapting Convolutional Neural Networks to the task of simultaneous categorization and pose estimation of objects. We further investigate and analyze the layers of various CNN models and extensively compare between them with the goal of discovering how the layers of distributed representations of CNNs represent object pose information and how this contradicts object category representations. We extensively experiment on two recent large and challenging multi-view datasets. Our models achieve better than state-of-the-art performance on both datasets.

6.1 Introduction

Impressive results have been achieved recently with the application of Convolutional Neural Networks (CNNs) in the tasks of object categorizations [74] and detection [123, 48]. Several studies recently investigated different properties of the learned representations at different layers of the network, *e.g.* [150, 151, 19]. One fundamental question is how CNN models achieve different invariances. It is well understood that consecutive convolution and pooling layers can achieve translation invariant. Training CNN networks with a large dataset of images, with arbitrary viewpoints and arbitrary illumination, while optimizing the categorization loss helps to

achieve viewpoint invariant and illumination invariant.

In this work we focus on studying the viewpoint invariant properties of CNNs. In many applications, it is desired to estimate the pose of the object, for example for robot manipulation and scene understanding. Estimating pose and object categorization are tasks that contradict each other; estimating pose requires a representation capable of capturing the viewpoint variance, while viewpoint invariance is desired for categorization. Ultimately, the vision system should achieve a representation that can factor out the viewpoint for categorization and preserve viewpoint for pose estimation.

The biological vision system is able to recognize and categorize objects under wide variability in visual stimuli, and at the same time is able to recognize object pose. It is clear that images of the same object under different variability, in particular different views, lie on a low-dimensional manifold in the high-dimensional visual space defined by the retinal array ($\sim$100 million photoreceptors and $\sim$1 million retinal ganglion cells). DiCarlo and Cox [27] hypothesized that the ability of our brain to recognize objects, invariant to different viewing conditions, such as viewpoint, and at the same time estimate the pose, is fundamentally based on untangling the visual manifold encoded in neural population in the early vision areas (retinal ganglion cells, LGN, V1). They suggested that this is achieved through a series of successive transformation (re-representation) along the ventral stream (V1,V2, V4, to IT) that leads to an untangled population at IT. Despite this, it is unknown how the ventral stream achieves this untangling. They argued that since IT population supports tasks other than recognition, such as pose estimation, the manifold representation is some how *"flattened"* and *"untangled"* in the IT layer. DiCarlo and Cox's hypothesis is illustrated in Figure 6.1. They stress that the feedforward cascade of neural re-representation is a way to untangle the visual manifold.

Inspired by recent impressive results of CNNs and by DiCarlo and Cox's hypothesis [27] on manifold untangling, this work focuses on studying the view-manifold structure in the feature spaces implied by the different layers of CNNs. There are several questions that this work aims to answer:

1. Does the learned CNN representations achieve viewpoint invariance? If so, how does it achieve viewpoint invariance? Is it by collapsing the view manifolds, or separating them

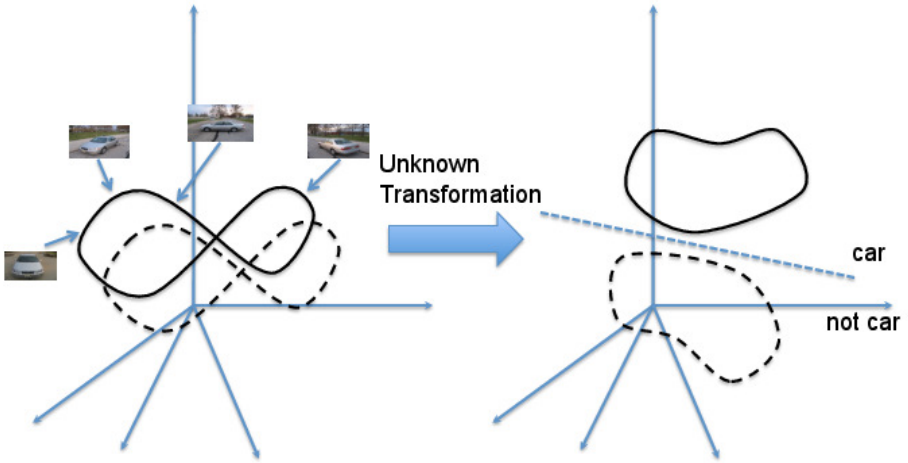


Figure 6.1: Illustration of DiCarlo and Cox model [27]: Left: tangled manifolds of different objects in early vision areas. Right: untangled (flattened) manifold representation in IT

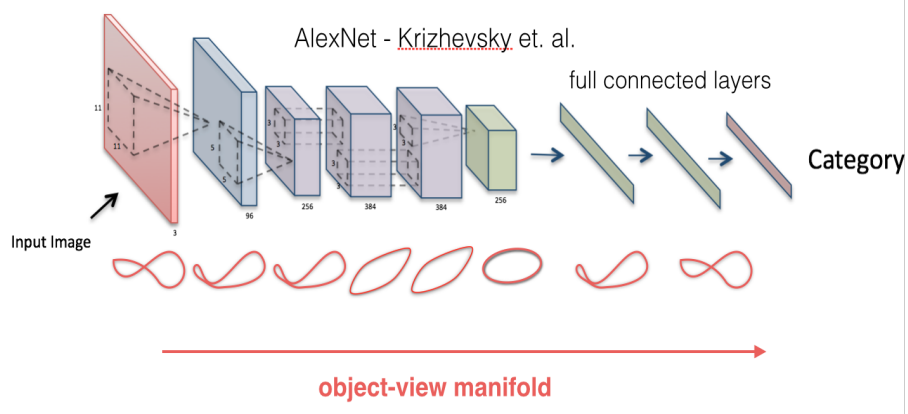


Figure 6.2: Illustration showing hypotheses of the object-view manifold within the layers of a CNN. We attempt to analyze how CNN models represent the object-view manifold within their layers?

while preserving them? At which layer is the view invariance achieved? See Figure for a hypothesized illustration of how CNN layers represent the object-view manifold 6.2.

2. How to experimentally quantify the structure of the viewpoint manifold at each layer of a deep convolutional neural network?
3. How does fine-tuning of a pre-trained CNN, optimized for categorization on a multi-view dataset, affect the representation at each layer of the network?

In order to answer these questions, we present a methodology that helps to get an insight about the structure of the viewpoint manifold of different objects as well as the combined

object-view manifold in the layers of CNN. We conducted a series of experiments to quantify the ability of different layers of a CNN to either preserve the view-manifold structure of data or achieve a view-invariant representation.

The contributions of this work are as follows: (1) We propose a methodology to quantify and get insight into the manifold structures in the learned representation at different layers of CNNs. (2) We use this methodology to analyze the viewpoint manifold of pre-trained CNNs. (3) We study the effect of transfer learning a pre-trained network with two different objectives (optimizing category loss vs. optimizing pose loss) on the representation. (4) We draw important conclusions about the structure of the viewpoint-object manifold and how it coincides with DiCarlo and Cox's hypothesis.

Section 6.2 defines the problem, experimental setup, and the basic CNN network that our experiments are based upon. Section 6.3 introduces our methodology of analysis. Sections 6.4 and 6.5 describe the findings on the pre-trained network and the fine-tuned networks respectively. After that, starting with section 6.6 we present our investigation into adapting the CNN model for simultaneous categorization and pose estimation.

6.2 Problem Definition and Experimental Setup

It is expected that multiple views of an object lie on intrinsically low-dimensional manifolds (*view manifold*[1]) in the input space. View manifolds of different instances and different objects are spread out in this input space, and therefore form jointly what we call the *object-view manifold*. The input space here denotes the $R^{N\times M}$ space induced by an input image of size $N \times M$, which is analogous to the retinal array in the biological system. For the case of a viewing circle(s), the view manifold of each object instance is expected to be a 1-dimensional closed curve in the input space. The recovery of the category and pose of a test image reduces to finding which of the manifolds this image belongs to, and what is the intrinsic coordinate of that image within that manifold. This view of the problem is shared among manifold-based approaches such as [97, 152, 7]

The ability of a vision system to recover the viewpoint is directly related to how the learned

[1] we use the terms *view manifold* and *viewpoint manifold* interchangeably

representation preserves the view manifold structure. If the transformation applied to the input space yields a representation that results in the collapsing of the view manifold, the system will no longer be able to discriminate between different views. Since each layer of a deep NN re-represents the input in a new feature space, the question would be how the re-representations deforms a manifold that already exists in the input space. A deep NN would satisfy the hypothesis of *"flattening"* or *"untangling"* by DiCarlo and Cox [27], if the representation in a given layer separates the view manifolds of different instances, without collapsing them, in a way to be able to put a separating hyperplanes between different categories.

Typically CNN layers exhibit general-to-specific feature encoding, from Gabor-like features and color blobs at low layers to category-specific features at higher layers [151]. We can hypothesize that for the purpose of pose estimation, lower layers should hold more useful representations that might preserve the view manifold and be better for pose estimation. But which of these layers would be more useful, and where does the view-manifold collapse to view-invariance.

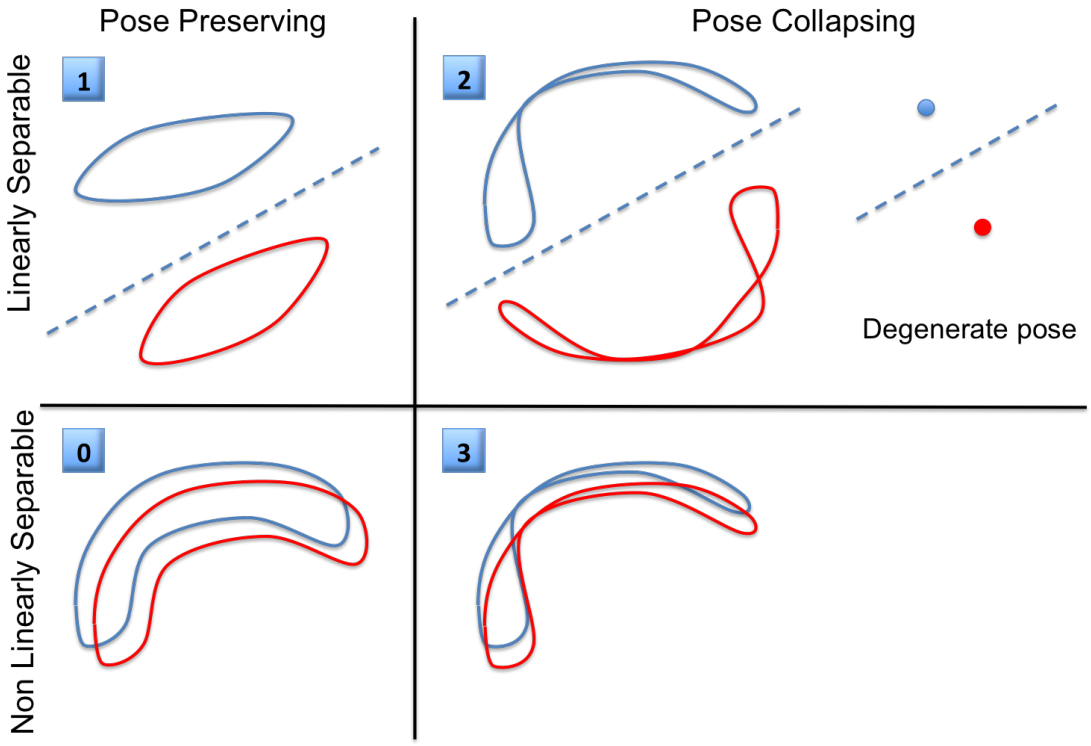


Figure 6.3: Sketches of four hypotheses about possible structures of the view manifolds of two objects in a given feature space

There are different hypotheses we can make about how the view manifolds of different objects are arranged in the feature space of a given layer. These hypotheses are shown in Figure 6.3. We arrange these hypotheses based on linear separability of the different objects' view manifolds and the preservation of the view manifolds. Case 0 is the non-degenerate case where the visual manifolds preserve the pose information but are tangled and there is no linear separation between them (this might resemble the input space, similar to left case in Figure 6.1). Case 1 is the ultimate case where the view manifolds of different objects are preserved by the transformation and are separable (similar to the right case in Figure 6.1). Case 2 is where the transformation in the network leads to separation of the object's view manifold at the expense of collapsing these manifolds to achieve view invariance. Collapsing of the manifolds can be to different degrees, to the point where each object's view manifold can be mapped to a single point. Case 3 is where the transformation results in more tangled manifolds (pose collapsing and non-separable). It is worth to notice that both cases 1 and 2 are view invariant representations. However, it is obvious that case 1 would be preferred since it also facilitates pose recovery. It is not obvious whether optimizing a network with a categorization loss result in case 1 or case 2. Getting an insight about which of these hypotheses are true in a given layer of a DNN is the goal of this work. In Section 6.3 we propose a methodology to get us to that insight.

6.2.1 Experimental Settings

To get an insight into the representations of the different layers and answer the questions posed in Section 6.1 we experiment on two datasets: I) RGB-D dataset [77], II) Pascal3D+ dataset [149]. We selected the RGB-D dataset since it is the largest available multiview dataset with the most dense viewpoint sampling. The dataset contains 300 instances of tabletop objects (51 categories). Objects are set on a turntable and captured by an Xbox Kinect sensor (Kinect 2010) at 3 heights (30°, 45° and 60° elevation angles). The dense view sampling along each height is essential for our study to guarantee good sampling of the view manifold. We ignore the depth channel and only used the RGB channels.

Pascal3D+ is very challenging because it consists of images "in the wild", in other words, images of object categories exhibiting high variability, captured in uncontrolled settings and

under many different poses. Pascal3D+ contains 12 categories of rigid objects selected from the PASCAL VOC 2012 dataset [38]. These objects are annotated with 3D pose information (*i.e.* azimuth, elevation and distance to camera). Pascal3D+ also adds 3D annotated images of these 12 categories from the ImageNet dataset [26]. The *bottle* category is omitted in state-of-the-art results. This leaves 11 categories to experiment with. There are about 11,500 and 7,000 training images in ImageNet and Pascal3D+ subsets, respectively. For testing, there are about 11,200 and 6,900 testing images for ImageNet and Pascal3D+, respectively. On average there are about 3,000 object instances per category in Pascal3D+, making it a challenging dataset for estimating object pose.

The two datasets provide different aspect of the analysis. While the RGB-D provides dense sampling of each instance's view manifold, Pascal3D+ dataset contains only very sparse sampling. Each instance is typically imaged from a single viewpoint, with multiple instances of the same category sampling the view manifold at arbitrary points. Therefore, in our analysis we use the RGB-D dataset to analyze each instance viewpoint manifold and the combined object-viewpoint manifolds, while the Pascal3D provides analysis of the viewpoint manifold at the category level.

Evaluation Split: For our study, we need to make sure that the objects we are dealing with have non-degenerate view manifolds. We observed that many of the objects in the RGB-D dataset are ill-posed, in the sense that the poses of the object are not distinct. This happens when the objects have no discriminating texture or shape to be able to identify the different poses (*e.g.* a texture-less ball, apple or orange on a turntable). This will cause view manifold degeneracy. Therefore we select 34 out of the 51 categories as objects that possess pose variations across the viewpoints, and thus are not ill-posed with respect to pose estimation.

We split the data into training, validation and testing. Since in this datasets, most categories have few instances, we left out two random object instances per category, one for validation and one for testing. In the case where a category has less than 5 instances, we form the validation set for that category by randomly sampling from the training set. Besides the instance split, we also left out all the middle height for testing. Therefore, the testing set is composed of unseen instances and unseen heights and this allows us to more accurately evaluate the capability of the CNN architectures in discriminating categories and estimating pose of tabletop objects.

6.2.2 Base Network: Model0

The base network we use is the Convolutional Neural Network described in Krizhevsky *et al.* [74] and winner of LSVRC-2012 ImageNet challenge [116]. The CNN was composed of 8 layers (including 1000 neuron output layer corresponding to 1000 classes). We call these layers in order: Conv1, Pool1, Conv2, Pool2, Conv3, Conv4, Conv5, Pool5, FC6, FC7, FC8 where Pool indicates Max-Pooling layers, Conv indicates layers performing convolution on the previous layer and FC indicates fully connected layer. The last fully connected layer (FC8) is fed to a 1000-way softmax which produces a distribution over the category labels of the dataset.

6.3 Methodology

The goal of our methodology is two-folds: (1) study the transformation that happens to the viewpoint manifold of a specific object instance at different layers, (2) study the structure of the combined object-view manifold at each layer to get an insight about how tangled or untangled the different objects' viewpoint manifolds are. Both these approaches will get us an insight to which of the hypotheses explained in Section 6.2 is correct at each layer, at least relatively by comparing layers. This section introduces our methodology, which consists of two sets of measurements to address the aforementioned two points. First, we introduce instance-specific measurements that quantify the viewpoint manifold in the different layers to help understand whether the layers preserve the manifold structure. *We performed extensive analysis on synthetic manifold data to validate the measures, which can be found in the appendix.* Second, we introduce empirical measurements that are designed to draw conclusions about the global object-viewpoint manifold (involving all instances).

6.3.1 Instance-Specific View Manifold Measurements

Let us denote the input data (images taken from a viewing circle and their pose labels) for of a specific object instance as $\{(x_i \in \mathbb{R}^D, \theta_i \in [0, 2\pi]), i = 1 \cdots N\}$, where D denotes the dimensionality of the input image to the network, and N is the number of the images, which are equally spaced around the viewing circle. These images form the view manifold of that object in the input space denoted by $\mathcal{M} = \{x_i\}_1^N$. Applying each image to the network will

result in a series of nonlinear transformations. Let us denote the transformation from the input to layer l by the function $f_l(x) : \mathbb{R}^D \rightarrow R^{d_l}$ where d_l is the dimensionality of the feature space of layer l. With an abuse of notation we also denote the transformation that happens to the manifold $\mathcal{M}$ at layer l by $\mathcal{M}^l = f_l(\mathcal{M}) = \{f_l(x_i)\}_1^N$. After centering the data by subtracting the mean, let $\mathbf{A}^l = [\acute{f}_l(x_i) \cdots \acute{f}_l(x_N)]$ be the centered feature matrix at layer l of dimension $d_l \times N$, which corresponds to the centered transformed images of the given object. We call $\mathbf{A}^l$ the sample matrix in layer l.

Since the dimensionality d_l of the feature space of each layer varies, we need to factor out the effect of the dimensionality. Since $N \ll l_d$ the transformed images on all the layers lie on subspaces of dimension N in each of the feature spaces. Therefore we can change the bases to describe the samples using N dimensional subspace *i.e.* , we define the $N \times N$ matrices $\hat{\mathbf{A}}^l = \mathbf{U}^\mathsf{T}\mathbf{A}$ where $\mathbf{U} \in \mathbb{R}^{d_l \times N}$ are the orthonormal bases spanning the column space of $\mathbf{A}^l$ (which we can get by SVD of $\mathbf{A}^l = \mathbf{USV}^\mathsf{T}$). This projection just rotates the samples at each layer.

The following measures will be applied to the N transformed images, representing the view manifold of each object instance individually. To obtain an overall measures for each layer we will average these measures over all the object instances.

1) Measure of spread - Nuclear Norm: There are several possible measures of the spread of the data in the sample matrix of each view manifold. We use the nuclear norm (also known as the trace norm [58]) defined as $||\mathbf{A}||_* = Tr(\sqrt{\mathbf{A}^\mathsf{T}\mathbf{A}}) = \sum_{i=1}^N \sigma_i$, *i.e.* it measure the sum of the singular values of $\mathbf{A}$.

2) Subspace dimensionality measure - Effective-p: As a measure of the effective dimensionality of the subspace where the view manifold lives, we define *Effective-p* as the minimum number of singular values (in decreasing order) that sum up to more that or equal to $p\%$ of the nuclear norm,

$$\text{Effective} - p = \sup\{n : \sum_{i=1}^{n} \sigma_i / \sum_{i=1}^{N} \sigma_i \leq p/100\} \tag{6.1}$$

A smaller number means that the view manifold samples live in a lower dimensional subspace.

3) Alignment Measure - KTA: Ideally the view manifold resulting of the view sitting is a one-dimensional closed curve in the feature space, which can be thought as a deformed circle [152].

This manifold can be degenerate in the ultimate case to a single point in case of a textureless object. The goal of this measurement is to quantify how the transformed manifold locally preserves the original manifold structure. To this end we compare the kernel matrix of the transformed manifold at layer l, denote by $\mathbf{K}_n^l$, with the kernel matrix of the an embedding of the ideal view manifold on unit circle, denote by $\mathbf{K}_n^\circ$, where n indicates the local neighborhood size used in constructing the kernel matrix. We construct the neighborhood based on the pose labels.

Given these two kernel matrices we can define several convergence measures. We use *Kernel Target Alignment (KTA)* which has been used in the literature for kernel learning [22]. It simply finds a scale invariant dependency between two normalized kernel matrices[2]. Therefore, we define the alignment of the transformed view manifold $\mathcal{M}^l$ at layer l with the ideal manifold as

$$KTA_n(\mathcal{M}^l) = \frac{< \mathbf{K}_n^l, \mathbf{K}_n^\circ >_F}{||\mathbf{K}_n^l, ||_F ||\mathbf{K}_n^\circ||_F} \tag{6.2}$$

4) KPLS-regression measures: Kernel Partial Least Squares (KPLS) [114] is a supervised regression method. KPLS iteratively extracts the set of principal components of the input kernel that are most correlated with the output [3]. We use KPLS to map from the transformed view manifold kernel $\mathbf{K}_n^l$ (input kernel) to the unit circle kernel $\mathbf{K}_n^\circ$ (output kernel). We enforce this mapping to use maximum of $d = 5$ principal components, where $d \ll N$ and N is the dimensionality of the kernels. Based on this mapping, we define *KPLS-Regression Error*, which measures the Normalized Cross Correlation between the output kernel and the mapped input kernel. This measures the mapping correctness from the transformed view manifold to the circle manifold kernel [4].

5) TPS-linearity measure: In this measure we learn a regularized Thin Plate Spline (TPS) nonlinear mapping [30] between the unit circle manifold and each $\mathcal{M}^l$. The reason for using TPS in particular is that the mapping has two parts, an affine part (linear polynomial) and a nonlinear

[2]We also experimented with HSIC [50], however HSIC is not scale invariant and is not designed to compare data in different feature spaces. Even after scaling the data, HSIC did not give any discriminative signal

[3]Unlike KPCA which extracts the principal components of the input kernel to maximize the variance of the output space, KPLS extracts the principal components of the input kernel that maximize the correlation with the output data.

[4]We also used another KPLS measure we call *KPLS-Norm Ratio*, which can be found in the appendix

part; details about the definition of the metric is in the appendix). Analysis of the two parts will tell us if the mapping is mostly linear or nonlinear. We use the *reciprocal-condition number (rcond)* of the sub coefficient matrices corresponding to the affine and the nonlinear part as a measure of the linearity of the transformation.

6.3.2 Global Object-Viewpoint Manifold Measures

To achieve an insight about the global arrangement of the different objects' view manifolds in a given feature space we suggest to use the following three empirical measurements:

6) L-SVM: For a test image x transformed to the l-th layer's feature space, $f_l(x)$, we compute the performance of a linear SVM classifier trained for categorization. Better performance of such a classifier directly implies more linear separability between different view manifolds of different categories.

7) Kernel Pose Regression: To evaluate whether the pose information is preserved in a local neighborhood of a point in a given feature space we evaluate the performance of kernel ridge regression for the task of pose estimation. Better performance implies better pose-preserving transformation, while poor performance indicates pose-collapsing transformation. The combination of L-SVM and kernel regression should be an indication to which of the hypotheses in Figure 6.3 is likely to be true.

8) Local Neighborhood Analysis: To evaluate the local manifold structure we also evaluate the performance of nearest neighbor classifiers for both category and pose estimation, with varying size of the neighborhood. This directly tell us whether the neighbors of a given point are from the same category and/or of similar poses. KNN for categorization cannot tell us about the linear separability of classes. However evaluating the pose estimation in a neighborhood of a datapoint gives us an insight about how the view manifolds are preserves, and even whether the view manifolds of different instances are aligned. To achieve this insight we use two different measurements:

1. KNN-Accuracy: the accuracy of KNN classifiers for category and pose estimation.
2. KNN-Gap: the drop in performance of each KNN classifier as the neighborhood size increases. In particular in our experiments we increase K from 1 to 9. Positive gap

indicates a drop (expected) and negative gap indicates improvement in performance.

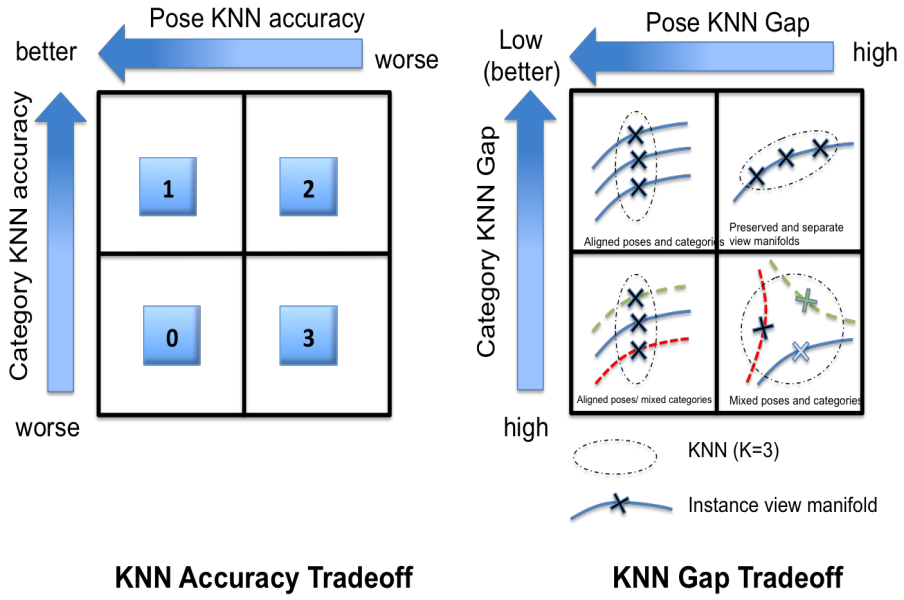


Figure 6.4: KNN Tradeoffs: Left) accuracy tradeoff between pose and category estimation using KNN. Right) Gap tradeoff (Gap is the drop in performance as K increases.)

The interaction between these two measures and how they tell us about the manifold structure is illustrated in Fig 6.4. The contrast between the accuracy of the KNN classifiers for pose and category directly implies which of the hypotheses in Figure 6.3 is likely. The analysis of KNN-Gap (assuming good 1-NN accuracy) gives further valuable information. As the KNN-gap reaches zero in both category and pose KNN classifiers, this implies that neighborhoods are from the same category and has the same pose, which indicates that the representation aligns the view manifolds of different instances of the same category. If the view manifolds of such instances are preserved and separated in the space, and the neighbors of a given point are from the same instance, this would imply small gap in the category KNN classifier and bigger gap in pose KNN classifier. Low gap in pose KNN vs high gap in category CNN implies the representation aligns view manifolds of instances of different categories. A high gap in both obviously implies the representation is tangling the manifolds such that a small neighborhood contains points from different categories and different poses. Notice that this implications are only valid when the 1-NN accuracy is high.

6.4 Analysis of the Pre-trained Network

6.4.1 Instance View Manifold Analysis

Figure 6.5 shows the application of the instance-specific view manifold measurements on the images of the RGBD dataset when applied to a pre-trained network (Model0 - no fine-tuning). This gives us an insight on the transformation that happens to the view manifold of each object instance at each layer of the network. Figure 6.5a shows that the nuclear norm of the transformed view manifolds in Model0 is almost monotonically decreasing as we go higher in the network, which indicates that the view manifolds is more spread in the lower layers. In fact at the output layer of Model0 the nuclear norm becomes too small, which indicates that the view manifold is collapsing to reach view invariant representation at this layer. Figure 6.5b ($p = 90\%$) shows that subspace dimension varies within a small range in the lower layers and it reduces dramatically in fully connected layers, which indicates that the network tries to achieve view invariance. The minimum is achieved at FC8 (even without fine tuning). Figure 6.5c shows the KTA applied to Model0, where we can notice that the alignment is almost similar across the lower layers, with Pool5 having the maximum alignment, and then starts to drop at the very high layers. which indicates that after Pool5, the FC layers try to achieve view invariant. Fig 6.5d shows that KPLS regression error on Model0 dramatically reduces from FC8 down to Pool5, where Pool5 has the least error. In general the lower layers have less error. This indicates that the lower layers preserve higher correlation with the ideal manifold structure. Fig 6.5e shows that the mapping is highly linear, which is expected because of the high dimensionality of the feature spaces. From Fig 6.5e we can clearly notice that the lower layers has more better-conditioned linear mapping[5].

From these measurements we can conclude: (1) The lower layers preserve the view manifolds. The manifolds start to collapse in the FC layers to achieve view invariance. Preserving the view manifold at the lower layers is intuitive because of the nature of the convolutional layers. (2) The manifold at Pool5 achieves the best alignment with the pose labels. This is a less intuitive result; why does the representation after successive convolutions and pooling

[5]Figures for the nonlinear part can be seen in the appendix, which shows very small values ($\approx 10^{-11}$) compared to the linear part

improves the view manifold alignment? even without seeing any dense view manifold in training, and even without any pose labels being involved in the loss. The hypotheses we have to justify that Pool5 has better alignment than the lower layers is that Pool5 has better translation invariant properties, which results in improvement of the view manifold alignment.

6.4.2 Global Object-View Manifold Analysis

The performance of Linear SVM and Kernel Regression is shown in Figure 6.6, using RGBD dataset (plots for Pascal3D are in the appendix). The figure clearly shows the conflict in the representation of the pre-trained network (categorization increases and pose estimation decreases). Linear separability of category is almost monotonically increasing up to FC6. Linear separability in FC7 and FC8 is worse, which is expected as they are task specific (no fine-tuning). Surprisingly Pool1 features perform very bad, despite being the most general features (typically they show Gabor like features and color blobs). In contrast, for pose estimation, the performance increases as we go lower in the network up to Conv4 and then slightly decreases. This confirms our hypothesis that lower layers offer better feature encoding for pose estimation. It seems that Pool5 provides feature encoding that offer the best compromise in performance, which indicates that it is the best in compromising between the linear separation of categories and the preservation of the view-manifold structure.

What is surprising is that the pose estimation results do not drop dramatically in FC6 and FC7. We can still estimate the pose (with accuracy around 63%) from the representation at these layers, even without any training on pose labels. This highly suggests that the network preserves the view manifold structure to some degree. For examples taking the accuracy as probability at layer FC6, we can vaguely conclude that 90% of the manifolds are linearly separable and 65% are pose preserved (we are somewhere between hypotheses 1 and 2 at this layer).

The aforementioned conclusion is also confirmed by the KNN analysis. Very high accuracy is achieved using KNN for category and pose. The category gap is reducing as we go up in the network up to FC7 (almost 0 gap at FC6 and FC7). In contrast the gap is large at all layers for pose estimation. This indicates separation of the instances' view manifolds where the individual manifolds are not collapsed (This is why as we increase the neighborhood, the

Approach	Split	Categorization %	Pose (AAAI metric)
Model0 (SVM/Kernel Regression) on FC6	1	86.71	64.39
Model0 (SVM/Kernel Regression) on conv4	1	58.64	67.39
HOG (SVM/Kernel Regression)	1	80.26	27.95
Model1	1	89.63	81.21%, 69.58 ($<$ 22.5), 81.09 ($<$ 45)

Table 6.1: RGBD Dataset Results for Model1, Model0, and HOG

category performance stays the same while pose estimation decreases smoothly - See the top-right block of Figure 6.4-right for illustration). The results above consistently imply that the higher layers of CNN (expect FC8 which is task specific), even without any fine-tuning on the dataset, and even without any pose label optimization achieve representations that separate and highly preserve the view manifold structure.

Tables 6.1 and 6.2 show the quantitative results of our models, compared to baselines, on the RGBD Dataset and Pascal3D+. Table 6.2 shows our quantitative results compared against two previous methods [153] and [149], using the two metrics $< 45^\circ$ and $< 22.5^\circ$. It is important to note that the comparison with [149] is unfair because they solve for detection and pose simultaneously while we do not solve for detection. We solve for categorization and pose.

6.5 Effect of Transfer Learning

In order to study the effect of fine-tuning the network (transfer learning to a new dataset) on the representation we trained the following model (denoted as Model1). This architecture consists of two parallel CNNs: one with category output nodes (Model1-Cat), and one with binned pose output nodes (Model1-Pose). We used 34 and 11 category nodes for RGBD and Pascal3D datasets respectively; while we used 16 pose nodes for both datasets). The parameters of both CNNs were initialized by Model0 parameters up to FC7. The parameters connecting FC7 to the output nodes are randomly initialized on both networks and they are fine-tuned by minimizing the categorization loss for Model1-Cat and the pose loss for Model1-pose. The purpose of these architectures is to study the effect of fine-tuning when the category and pose are independently

Approach	Categorization %	Pose (AAAI metric)
Model0 (SVM/Kernel Regression)		
FC6	73.64	49.72
FC7	76.38	48.24
FC8	71.13	45.41
Model1 (SVM/Kernel Regression)		
FC6	74.65	54.41
FC7	79.25	54.07
FC8	84.12	60.31
Model0 NN		
FC6	60.05	61.11
FC7	69.89	61.38
FC8	61.26	60.32
Model1 NN		
FC6	73.50	65.87
FC7	77.30	66.07
FC8	83.07	70.54
Model1 (direct prediction from CNN)		
output	84.00	71.60 (AAAI), 47.34(<22.5), 61.30 (<45)
State of the Art Methods		
[153]	-	44.20 (< 22.5), 59.00 (<45)
[149]	-	15.6% (<22.5), 18.7% (<45)

Table 6.2: Pascal3D Performance. Here we see our results compared to the two baselines from [153] and [149] using the two metrics $< 45^\circ$ and $< 22.5^\circ$. It is important to note that the comparison with [149] is slightly unfair because they solve for detection and pose simultaneously while we do not solve for detection. We solve for categorization and pose. Model1 here outperforms both [153] and [149] (despite the unfair comparison with the latter approach).

optimized.

We applied all the measures described in Sec 6.3 to understand how the view manifolds will be affected after such tuning. The questions are: To what degree optimizing on category should damage the ability of the network to encode view manifolds. On the other hand, how optimizing on pose should enhance that ability. Model1-Cat indicates the effect of optimizing on category, while Model1-Pose indicates the effect of optimizing on pose.

Fig 6.5 shows the five view manifold measures for the different layers of Model1(Cat/Pose), in comparison with Model0. In terms of data spread, from Fig 6.5a shows that the spread at FC8 has doubled after fine tuning on pose (Model1-Pose). Fig 6.5b shows the fine tuning on category (Model1-Cat) caused the view manifold subspace dimensionality to significantly reduce to 1, where it became totally view invariant. Optimizing on pose slightly enlarged the subspace dimensionality (*i.e.* become better) at FC8 and FC7. Fig 6.5c clearly shows the significant improvement achieved by fine tuning on pose, where the alignment of FC8 jumped to close to 0.9 from about 0.78, while fine tuning on category reduces the alignment of FC8 to close to 0.65. Similar behavior is also apparent in the KPLS ratio for FC8 and FC7 (sup-mat).

One very surprising result is that optimizing on pose makes the pose KTA alignment worse at the lower layers, while optimizing on category makes the pose alignment better compared to model0. In fact, although optimizing on pose significantly helps aligning FC8 with pose labels, Pool5 still achieves the best KTA alignment and the least regression reconstruction error. The regression reconstruction error in Fig 6.5d clearly shows significant improvement in the representation of FC8 and FC7 to preserve the view manifold. One surprising finding from these plots is that the representation of FC6 becomes worse after fine tuning for both pose and category. Fig 6.5e indicates that the deformation of the view manifold is reduced as a result of fine tuning on pose (larger rcond number), while it increases as a result of fine tuning on category.

On the global object-view manifold structure, we notice from Figures 6.6 some intuitive behavior at FC8. Basically optimizing on pose reduces the linear separability and increases the view manifold preservation (moves the representation towards hypothesis 0). In contrast, optimizing the category significantly improves the linear separability at FC9, however, interestingly, it only slightly reduces the pose estimation performance to be slightly less than 50%. Combining this conclusion with the observation from Fig 6.5b, that the view manifold subspace dimensionality reduces to 1, this implies that optimizing on category collapses the view manifolds to a line, but they are not totally degenerate. What is less obvious is the effect of fine tuning on the lower layers than FC8. Surprisingly, optimizing on pose did not affect the linear separability of FC7. Another very interesting observation is that optimizing on category actually improves the pose estimation slightly at the FC7, FC6, and Pool5; and did not reduce it at lower layers. This implies that fine tuning by optimizing on category only improved the internal view manifold preservation at the network, even without any pose labels. Analysis using KNN measures, as well as results of fine-tuning on Pascal3D dataset are available in the appendix.

6.6 Adapting CNNs to Categorization and Pose Estimation

Based on the findings presented so far, in the remainder of this chapter we present an in-depth analysis and discussion of the view-invariant properties of CNNs. We proposed a methodology

to analyze individual instance's view manifolds, as well as the global object-view manifold. We applied the methodology on a pre-trained CNN, as well as two fine-tuned CNNs, one optimized for category and one for pose. We performed the analysis based on two multi view datasets (RGBD and Pascal3D+). Results on both datasets give consistent conclusions.

Based on the proposed methodology and the datasets, we analyzed the layers of the pre-tained and fine-tuned CNNs. There are several findings from our analysis that are detailed throughout the work, some of them are intuitive and some are surprising. We find that a pre-trained network captures representations that highly preserve the manifold structure at most of the network layers, including the fully connected layers, except the final layer. Although the model is pre-trained on ImageNet, not a densely sampled multi-view dataset, still, the layers have the capacity to encode view manifold structure. It is clear from the analysis that, except of the last layer, the representation tries to achieve view invariance by separating individual instances' view manifolds while preserving them, instead of collapsing the view manifolds to degenerate representations. This is violated at the last layer which enforces view invariance.

Overall, our analysis using linear SVM, kernel regression, KNN, combined with the manifold analysis, makes us believe that CNN is a model that simulate the manifold flattening hypothesis of Cox and DiCarlo [27] even without training on multi-view dataset and without involving pose labels in the objective's loss.

Another interesting finding is that Pool 5 offers a feature space where the manifold structure is still preserved to the best degree. Pool 5 shows better representation for the view-manifold than early layers like Pool1. We hypothesize that this is because Pool5 has better translation and rotation invariant properties, which enhance the representation of the view manifold encoding.

We also showed the effect of fine-tuning the network on multi-view datasets, which can achieve very good pose estimation performance. In this work we only studied the effect of independent pose and category loss optimization. Optimizing on category achieves view invariance at the very last fully connected layers; interestingly it enhances the viewpoint preservation at earlier layers. We also find that fine-tuning mainly affects the higher layers and rarely affects the lower layers.

In the following sections, we further investigate how to use this analysis to build a CNN model that can be adapted to the task of simultaneous categorization and pose estimation of

objects. We investigate how each of these tasks affects the other, *i.e.* how category-specific information can help estimate the pose of an object, and how a balance between these contrasting tasks can be achieved. We analyze different CNN models and extensively compare between them to find an efficient balance between high accuracy object categorization and robust pose estimation. (4) We validate our work by extensive experiments on two recent large and challenging multi-view datasets and we achieve better than state-of-the-art performance on both.

The first question we pose in this part is: *how good are pre-trained representations of different CNN layers, without fine-tuning, for the task of pose estimation?* To answer this we took a state-of-the-art CNN trained on the ImageNet challenge (introduced in [74]) and tested it with dense multi-view images from the RGBD dataset [77] to see how well it represented object view-manifolds and hence was able to discriminate between object poses. This CNN is composed of 8 layers. We call these layers in order: Conv1, Pool1, Conv2, Pool2, Conv3, Conv4, Conv5, Pool5, FC6, FC7, FC8. Pool indicates Max-Pooling layers, Conv indicates layers performing convolution on the previous layer and FC indicates fully connected layer.

In order to quantitatively evaluate the representations of pose in this network, we trained pose regressors using Kernel Ridge-Regression and a SVM classifier for categorization (linear one-vs-all) on the features extracted from each of the layers of the CNN. Fig. 6.7-Left is the result of the regressor and classifier. It clearly shows the conflict in representation of the pre-trained network. For pose estimation, the performance increases around Pool5 and and then decreases in the following layers. This confirms that shallower layers offer better feature encoding for pose estimation, but only up to a point where before that the abstraction necessary to represent pose is not sufficient. It appears that Pool5 provides representations that offers the best balance in performance, indicating it is the best in compromising between the categorization and discriminating pose.

In Fig 6.7-Left we also report cross-evaluation of categorization using pose features and vice versa, we see that, FC8 (output) which is task specific, does not perform good pose estimation, while FC6/FC7 perform much better. It is interesting to observe the opposite is not true; when optimizing on pose, much of the category-specific information is still represented by the features of the CNN, as seen by the increase in performance of category recognition using

the pose-optimized features.

We further explored using other regressors on multiple views of a single object instance (GPR [111], WKNN [22], SVR [22], KTA [22]). We use a coffee mug instance that has enough visual and shape features to discriminate its poses. Fig 6.7-Right shows the MAE of the pose regression. The results also confirm that the pose representation improves as we approach Pool5. Since this is on one object instance, this indicates that Pool5 has the best representation of the object's view-manifold. We also found that the performance of features based on Pool5 are the closest to correlate with the performance when using HOG features on the objects' multi-view images. This further proves that Pool5 has the abstraction capability of representing pose efficiently.

6.7 Analyzed Models

We now discuss the various CNN models used for extensive analysis and evaluation. The CNN model used as a base network in our experiments is the CNN built by Krizhevsky et al. [74] which achieved state-of-the-art performance in the LSVRC-2012 Imagenet Challenge for object recognition [116]. We will refer to this model as the *Model0: base network*. This model is pre-trained on Imagenet. The last fully connected layer (FC8) is fed to a 1000-way softmax which produces a distribution over the category labels of the dataset. Dropout was employed during training and Rectified Linear Units (ReLU) were used for faster training. Stochastic gradient descent is used for back propagation.

It is important to note that this model is not fine-tuned on the datasets we experiment on. This is to show how the layers of a CNN trained on categorization of ImageNet lacks the ability to represent pose efficiently, but rather separates between (*i.e.* cluster) object categories where each *cluster* lacks pose information.

Throughout this study we vary the architecture of the base network and change the loss functions to study its behavior. All the models described are pre-trained on ImageNet and fine-tuned on each of the two large dataset we experimented on. The base network is only trained on ImageNet.

We investigate, in total, four different CNN models besides the base network (Base Network). We call these Parallel Model (PM), Cross-Product Model (CPM), Late Branching Model (LBM), Early Branching Model (EBM).

PM is a parallel version of the base network, in the sense that it is two parallel and independent versions of it, one for categorization and one for pose.

CPM involves jointly training over object category and pose, using a last layer with units for each category and pose combination, *i.e.* the cross-product of category and pose labels (depicted in Figure 6.8-a).

LBM and EBM models are also depicted in Figure 6.8. The LBM branches into two 1-layer subnetworks at the last layer of the overall architecture. EBM performs early branching into two subnetworks, each specialized in categorization and pose estimation, respectively. The output layer FC8 is not merged but instead the LBM and EBM networks optimize over two loss functions, one concerned with building view-invariance representations for category recognition and the other with category-invariance representation for pose estimation. Because of the branching, this causes two units to be active, one in each branch, at the same time. The only work that has done something similar to this in CNN models is the work by[49].

All losses are optimized by the multinomial logistic regression objective, similar to [74]. This loss is referred to as the *Softmax* loss. We denote softmax loss of label $l \in \{l^c, l^p\}$ and image x as $loss_i(x, l)$, where i indicates if this loss is over category or pose modes, l^p and l^c are the labels for pose and category, respectively.

In the following subsections we describe each model in detail.

6.7.1 Parallel Model (PM)

This model consists of two base networks running in parallel, each solving categorization and pose estimation independently. There is no sharing of information between the two networks. The goal of this model is to see how well the traditional CNN is capable of representing object-view manifolds and hence estimating object pose, independent of category-specific information. The category and pose losses minimized in this model are: $loss_c(x, l^c)$ and $loss_p(x, l^p)$, one for each of the tasks of categorization and pose estimation, respectively.

6.7.2 Cross-Product Model (CPM)

CPM explores a way to combine categorization and pose estimation by building a last layer capable of capturing both (see Fig6.8-a). We build a layer with the number of units equivalent to the number of combinations of category and pose, *i.e.* the cross-product of category and pose labels. The number of categories varies according to the dataset as we will see. The pose angles (in this case yaw or azimuth angle of an object) is discretized into angle bins across the viewing circle. This is the case with all our pose estimating models. The loss function for the CPM model is the softmax loss over the cross-product of category and pose labels: $loss(x, l^p \times l^c)$.

6.7.3 Late Branching Model (LBM)

We introduce a change in the architecture by splitting/branching the network into two last layers, each designed to be specific to the two tasks: categorization and pose estimation. Thus, this network has a shared representation for both category and pose information up until layer FC7 (see Fig6.8-b).

The goal is to learn category and pose information simultaneously from the representations encoded in the previous layers of the CNN. The question behind this model is whether or not one last layer would be enough to recover the pose information from the previous layers, in other words untangle the object view-manifold and give accurate pose estimates. In other words, one can think of this as testing the ability of the deep distributed representations of a CNN in holding both category-specific pose-invariant information as well as pose-variant information. LBM is trained using a linear combination of losses over category and pose: $\lambda_1 \cdot loss_c(x, l^c) + \lambda_2 \cdot loss_p(x, l^p)$ where λ_1, λ_2 are weights found empirically (see supplementary material).

6.7.4 Early Branching Model (EBM)

The question of moving the branching to an earlier layer in the network poses itself here: *Can the branching be moved earlier in the network to where the pose knowledge is still well preserved and in fact maximal across the layers?*

From our experiments (described later on) we observe that the objects' view-manifolds are

maximally represented at Pool5. Thus, this network has a shared representation for both category and pose information up until layer Pool5. At Pool5 it branches out into two subnetworks, that are jointly optimized using a combined loss function (same as for LBM): $\lambda_1 \cdot loss_c(x, l^c) + \lambda_2 \cdot loss_p(x, l^p)$. Similar to LBM, it is important to note that this network optimizes over two losses. This model achieves the most efficient balance between categorization and pose estimation and achieve state-of-the-art results on two large challenging datasets, as we shall see in Section 6.10.

6.8 Datasets

6.8.1 RGBD Dataset

One of the largest and challenging multi-view datasets available is the RGB-D dataset [77]. It consists of 300 tabletop object instances over 51 different categories. Images of objects are captured as it rotates on a turn-table, producing dense views of each object. The camera is positioned at three different heights with elevation angles: 30°, 45° and 60°.

In previous approaches the middle height (45°) is left out for testing [78, 152, 7, 33]. This means that object instances at test time have been seen before from other heights at training time. For this dataset it was important to experiment with an additional training-testing split of the data to give more meaningful results. We wanted to ensure that objects were never seen before at test time. We also wanted to make sure that the instances we are dealing with have non-degenerate view manifolds. We observed that many of the objects in the dataset are ill-posed, in the sense that the poses of the object are not distinct. This happens when the objects have no discriminating texture or shape to be able to identify its poses (*e.g.* a texture-less ball or orange). This will cause object-view manifold degeneracy. For this reason, we select 34 out of the 51 categories as objects that possess variation across the viewpoints, and thus are not ill-posed with respect to pose estimation. We split the data into training, validation and testing. In this datasets, most categories have few instances; therefore we left out two random instances per category, one for validation and one for testing. In the case where a category has less than 5 instances, we form the validation set for that category by randomly sampling one object instance from the training set. We also left out all the middle height for testing. Thus,

the testing set is composed of unseen instances and unseen heights and this allows us to more accurately evaluate the capability of CNNs in discriminating categories and poses of tabletop objects. We call this split, Split 1. In order to compare with state-of-the-art we also used the split used by previous approaches (we call this Split 2).

6.8.2 Pascal3D+ Dataset

We experiment on the recently released challenging dataset of multi-view images, called Pascal3D+ [149]. Pascal3D+ is very challenging because it consists of images *in the wild*, in other words, images of object categories exhibiting high variability, captured in uncontrolled settings, in cluttered scenes and under many different poses. Pascal3D+ contains 12 categories of rigid objects selected from the PASCAL VOC 2012 dataset [38]. These objects are annotated with pose information (azimuth, elevation and distance to camera). Pascal3D+ also adds pose annotated images of these 12 categories from the ImageNet dataset [116]. The *bottle* category is omitted in state-of-the-art results. To be consistent, we do the same. This leaves 11 categories to experiment with. There are about 11,500 and 7,000 training images in ImageNet and Pascal3D+ subsets, respectively. We take a small portion of these images for validation and use the rest for training. For testing, there are about 11,200 and 6,900 testing images for ImageNet and Pascal3D+, respectively. On average there are about 3,000 object instances per category in Pascal3D+ captured *in the wild*, making it a challenging dataset for estimating object pose.

6.9 CNN Layer Analysis

Similar to the analysis performed in Section **??**, we do this on all our described models. This gives insight into the ability of these models to represent pose and the intrinsic differences between them. We perform kernel Ridge-regression and SVM classification on each layer of the CNN models. The results of this analysis on the two muti-view datasets are shown in Fig. 6.9 and 6.10.

From Fig. 6.9 and 6.10, we can see that the base network monotonically decreases in pose accuracy after layer Pool5. Pool5 seems to again hold substantial pose information, before it is lost in the following layers. This is the premise behind the design of our EBM model. EBM is

able to efficiently untangle the object-view manifold and achieve good pose estimation on the branch specific to pose estimation.

From Fig. 6.9 and 6.10, it can be observed that the LBM is able to achieve a good boost in pose performance at its last layer. From 6.9-right, it can be observed that at the layers Conv4 and Pool5 EBM has slightly worse accuracy than LBM and PM on the RGBD dataset. This indicates that the optimization is putting emphasis on the category information just before branching to achieve better pose estimation at deeper layers.

We conduct $k - NearestNeighbor$ pose estimation on all the layers of the 4 models with varying neighborhood sizes (shown in Fig. 6.11). We observe that EBM is able to align object poses (*i.e.* align the instances' view manifolds) in layer FC7 while all the rest of the models align at the output layer. This can be concluded because the drop in KNN performance as k increases reduces to zero, which indicates the neighborhood contains images of the same pose. In addition, it can be observed that starting from Pool5, EBM has consistently more aligned object poses than the other models. This indicates the advantage of early branching, where pose representations are able to be preserved and enhanced within the CNN layers.

CPM does quite substantially worse than the other models on both datasets, in both categorization and pose estimation. This can be seen in Fig. 6.9-Left and 6.9-Right and to some extent in Fig. 6.10. The reason for this lies in the fact that CPM shares information to jointly optimize over category and pose. The drop is more evident in the task of categorization, indicating again that category information aids in estimating the pose, but not the other way round. The drop is more on the RGBD dataset because there are a lot more categories than Pascal3D+ and thus a lot more inter-class confusion. This is analogous to using category labels to separate between objects of different categories which may help bring similar posed objects of the same category together in the *latent* space encoded in the layers. On the other hand there is no clear untangling of the object-view manifold, where the pose information is stored, and thus this lack of pose information negatively impacts the categorization of objects.

6.10 Experiments

In this section we describe the experimental setup and present the quantitative results of our experiments as well as comparisons with state-of-the-art.

6.10.1 Training

All models are trained by back propagation with Stochastic gradient descent (batch size = 100 images). Refer to the supplementary material for details about the parameters, *e.g.* learning rate, decay *etc.* . At training time, we randomly sample 227x227 patches from the down-scaled 256x256 images. At test time the center 227x227 patches are taken.

All classification losses are optimized by the multinomial logistic regression objective. Similar to [74], we optimized it by maximizing the average of the log-probability of the correct label under the prediction distribution across training cases. For each of the category and pose losses, the gradient with respect to the CNN parameters is computed which is then fed into CNN training for back propagation.

The two metrics < 22.5 and < 45 used to evaluate the performance of pose estimation are the percentages of test samples that satisfy $AE < 22.5^{\circ}$ and $AE < 45^{\circ}$, respectively, where the Absolute Error (AE) is $AE = |EstimatedAngle - GroundTruth|$). The AAAI pose metric is defined as

$$\Delta(\theta_i, \theta_j) = min(|\theta_i - \theta_j|, 2\pi - |\theta_i - \theta_j|)/\pi \tag{6.3}$$

The base learning rate is assigned 0.5×10^{-3}. For fine-tuning, the learning rate of the randomly initialized parameters (*e.g.* FC8 parameters in PM) are assigned to be ten times higher than the learning rate of the parameters initialized from the pretrained CNN (*e.g.* . Conv1 to Pool5 in all the models). The decay of the learning rate γ is 0.1. While training our CNNs, we drop the learning rate by a factor of γ every 5000 iterations. The momentum and the weight decay were assigned to 0.9 and 0.0001 respectively. Training images are randomly shuffled before feeding the CNN for training.

6.10.2 Results

Table 6.3 shows the category recognition and pose estimation performance for the different models. Table 6.5 shows the performance of our models on Pascal3D+. We compare the accuracy achieved by our models with state-of-the-art results by [153] and results by [149]. It must be noted here that we are solving a slightly different problem than [149], where they solve detection and pose estimation, assuming correct detection. On the other hand [153] solve just pose estimation, assuming that the object categories are known. In our case we are jointly solving both category recognition and pose estimation, which can be considered a harder problem than that of [153] and [149]. Our pose estimation performance is better than both these previous approaches. For the sake of this comparison, we computed the pose performance using the metrics applied in [153]. These metrics are pose accuracy for images with pose errors $< 22.5^{\circ}$ and $< 45^{\circ}$.

Table 6.5 (rows 1-4) shows both our categorization and pose estimation results on Pascal3D dataset-Pascal12 part, compared with [153]. The table indicates 13.69% improvement of our method over [153] in $pose < 22.5^{\circ}$ metric and 4% improvement in $pose < 45^{\circ}$, which are significant results. We also show our performance when including ImageNet images in the training set and also the test set - see Table 6.5 (rows 5-8). The results show the benefit of ImageNet training images which boosts pose performance to 76.9% (from 57.89%) and 88.26% (from 63.0%) for $pose < 22.5^{\circ}$ and $pose < 45^{\circ}$, respectively. It is important to note that comparing to [149] is slightly unfair because [149] solve for detection and pose simultaneously, while we do not solve detection.

Since all CNN were trained with a multinomial loss, maximizing log-probability of the correct label under the prediction distribution across training cases. This indicates that the pose softmax output (FC8) layer is the pose probability distribution given the image. All the results that are presented in the paper were based on the prediction of $argmax_{pose}p(pose|x)$, where $pose$ is one of the 16 poses and x is the given image. We did an additional experiment where we predict the pose by computing the expected pose in the distribution of $p(pose|x)$; see Eq.

6.4.

$$E(p(pose|x)) = \sum_i p(pose_i|x) \times \phi(pose_i) \tag{6.4}$$

where $\phi(pose_i)$ is the center angle of the corresponding bin $pose_i$ (the pose of the i^{th} bin).

Using the pose prediction rule of eq. 6.4, the pose accuracy of EBM(800) increased from 78.83% to 79.30% for the $argmax$ prediction on split 2 on RGBD (reported in Table 6.4). This indicates that the expected pose of the distribution of the learnt $p(pose|x)$ is a more accurate measure. With this, we achieve over 1% increase in category recognition and just over 3% increase in pose estimation (79.30%) using the EBM(800) model. In Table 6.5, on the *in the wild* images of Pascal3D+ dataset, our EBM model achieves a very impressive increase of $\sim$8% and $\sim$5% using the two pose accuracy metrics, respectively. These measurements are likely to increase further when using the EBM(4096) as we see slight improvement of EBM(4096) over EBM(800) in Table 6.4. It is also possible that running k-NN on top of the layer features would improve performance further. In Table 6.4, [7] achieved 96.01% classification accuracy. This is achieved using both visual and depth channels. In our system, we used only the RGB (no depth). The approach in [7] achieved 94.84% with RGB only, which shows the advantage of our models for classification. We achieved 97.14% using the EBM model. We also achieved 99.0% classification accuracy using Nearest Neighbor classification on the Pool5 layer from EBM Model, which indicates that we learnt better convolutional filters.

6.11 Discussion

We offer multiple forms of analytical and quantitative results. Our analysis of the layers of the various CNN models are shown in Fig. 6.9, 6.10 and 6.11. We provide quantitative results over two challenging datasets and summarize them in Tables 6.3, 6.4 and 6.5.

Table 6.3: A summary of all the results of the CNN models. Split 2 is the traditional RGBD dataset split. Split 1 is the one we describe that better evaluates our experiments. Split 2 is the state-of-the-art training-testing split. C indicates category performance and P indicates pose accuracy (where it is measured using 3 different metrics consistent with state-of-the-art. P ($< 22.5^\circ$) indicates that the pose accuracy is measured for objects where the pose error was less than 22.5°

Model	Split	C	P ($< 22.5^\circ$)	P ($< 45^\circ$)	P (AAAI)
PM	1	89.63	69.58	81.09	81.21
CPM	1	80.68	63.46	75.45	77.35
LBM	1	**91.48**	68.25	79.31	79.94
EBM (4096)	1	89.94	**71.49**	**82.19**	**82.00**
EBM (800)	1	89.84	71.29	0.8229	81.91
EBM (400)	1	89.77	70.80	81.73	81.65
EBM (200)	1	90.11	67.70	79.43	79.77
EBM (100)	1	90.34	69.15	80.09	80.36
EBM (800)	2	**97.14**	**66.13**	**77.02**	**78.83**
EBM (4096)	2	97.07	65.82	76.51	78.66
SVM/Kernel Reg - Model 0 (best category - FC6)	1	86.71	-	-	64.39
SVM/Kernel Reg - Model 0 (best pose - Conv4)	1	58.64	-	-	67.39
SVM/Kernel Reg - HOG	1	80.26	-	-	27.95

Table 6.4: RGBD Dataset: Comparison with state-of-the-art approaches on category recognition and pose estimation.

Approach	Category %	Pose %
[78]	94.30	53.50
[7]	96.01	76.01
[152]	93.10	61.57
Ours	**97.14**	**79.30**

Table 6.5: Pascal3D dataset [149]: Comparison with state-of-the-art approaches on category recognition and pose estimation. The AAAI pose metric is the performance metric used in[153, 78, 152].

Train: Pascal12, Test: Pascal12				
Approach	Category	Pose % (error < 22.5)	Pose % (error < 45)	Pose AAAI metric
[149]	-	18.70	15.60	-
[153]	-	44.2	59.0	-
EBM (4096)	83.0	**51.80**	**64.27**	**73.53**
EBM (800)	**83.10**	51.37	64.20	73.26
LBM	82.69	48.38	60.11	70.88
CPM	76.35	49.39	61.90	71.80
PM	**84.0**	47.34	61.30	71.60
Train: Pascal12 + ImageNet, Test: Pascal12				
EBM (800)	83.79	51.89	60.74	75.39
Train: Pascal12 + ImageNet, Test: Pascal12 + ImageNet				
EBM (800)	92.83	67.26	75.11	83.27

We compare our models with two baselines in Table 6.3: Linear SVM and Kernel Ridge-Regression on HOG descriptors [24] and weights from the best selected layer of the base network. These baseline results were expected to be quite lower than our models' performance due to the lack of fine-tuning in the base model and due to the sharing of the network between the two tasks of category recognition and pose estimation.

Without fine-tuning the base network does not represent the object-view manifold well enough to estimate the pose efficiently. After fine-tuning on each of the respective datasets used in this work, we are able to achieve good category performance on both datasets using the PM model. The downside of this model is its inability to perform robust pose estimation on the more challenging natural sparser-views of Pascal3D+. This is evident in the results shown in Table 6.5, where PM drops by about 3-4% in pose accuracy.

In Tables 6.3 and 6.5, we see again that CPM does worse than the other models in both datasets. The drop is more evident in the task of categorization, *e.g.* , a drop of $\sim$7% and 2%-3% in category and pose accuracy on Pascal3D+, respectively, and similarly $\sim$10% and $\sim$6% on the RGBD dataset. This motivates the need for branching in the networks and branching at the particular layer that more robustly represents both category and pose.

Changing the weights for EBM only slightly affected performance; see supplementary material. The intuition behind this is that EBM splits into separate parameters starting from Pool5, which makes each of the pose and category subnetworks have independent parameters (FC6,FC7,FC8), in addition to shared parameters (Conv1 to Pool5).

LBM performs relatively well on RGBD but this is not the case on Pascal3D+. This can be attributed to the fact that RGBD has much more categories than Pascal3D+ and is composed of images of objects under controlled settings and not *in the wild* like Pascal3D+. The images in the RGBD dataset are captured at dense views as the object rotates on a turn-table. This is why the pose information is more prevalent in the last layers. This evident from the steep monotonically increasing curve of LBM in Fig. 6.9-Left while this is not the case in Pascal3D+ where the increase is more steady and in fact there is a decrease after layer FC6 (see Fig. 6.10-Right).

The reason why EBM performs better than PM even though its weights are randomly initialized is that PM's FC6 and FC7 layers in the category branch are initialized with category-specific weights. This adversely affects pose estimation here. Our hypothesis is that learning the convolutional filters jointly with categories help make them discriminative for both tasks. Therefore initializing FC6/FC7 by another network trained for a different purpose is not likely to help.

The slight effect of decreasing the size of the layers in the pose subnetwork of EBM can be observed from the results in Table 6.3 and 6.5.

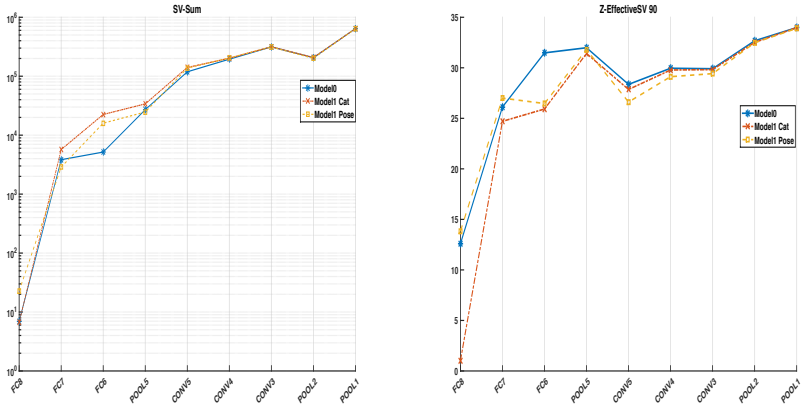
6.11.1 Computational Analysis

In Fig. 6.12 we show the convergence rates of the proposed models. EBM here is the larger EBM (4096) network. Despite having many more parameters than most of the other models (about $\sim$112 million parameters compared to 60 million in the base network), EBM converges substantially faster than all the other models. This shows the ability of this particular network to specialize faster in the two tasks. The shared first five layers are able to build up the object-view manifolds, preserve them and enhance them in the pose subnetwork of the model, while the other subnetwork specializes in pose-invariant category recognition.

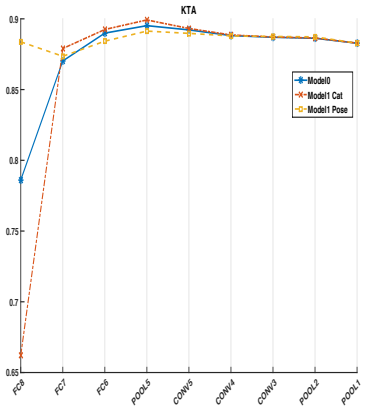
6.12 Conclusion

This work is the first exploration of using CNNs for object pose estimation. We present our analysis and comparison of CNN models with the goal of performing both efficient object categorization and pose estimation. Despite the dichotomy in categorization and pose estimation, we show how CNNs can be adapted, by novel means introduced in this work, to simultaneously solve both tasks. We make key observations about the intrinsics of CNNs in their ability to represent pose. We quantitatively analyze the models on two large challenging datasets with extensive experiments and achieve better than state-of-the-art accuracy on both datasets.

(a) $NuclearNorm$ (b) Effective 90% SV's



(c) KTA



(d) KPLS-Regression error (e) TPS-$RCond(CF-poly)$

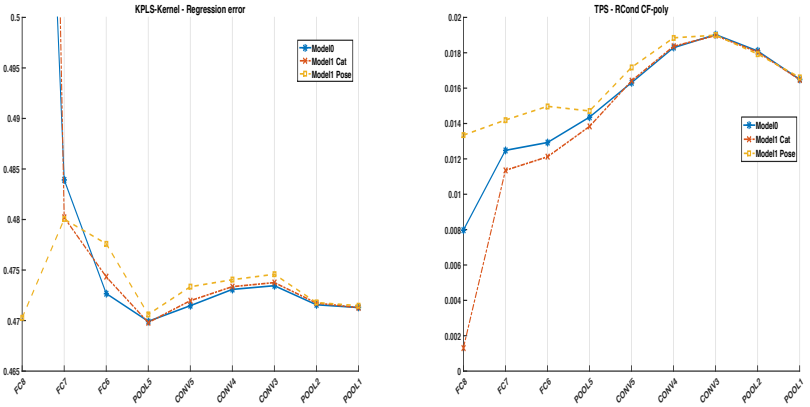


Figure 6.5: Measurement analysis for the view-manifold in RGBD dataset based on features extracted from different layers of several CNN models. Every figure shows single measurement. Multiple lines is for different CNN model. X-axis is labeled by the layers. Larger versions of the figures are available in the appendix.

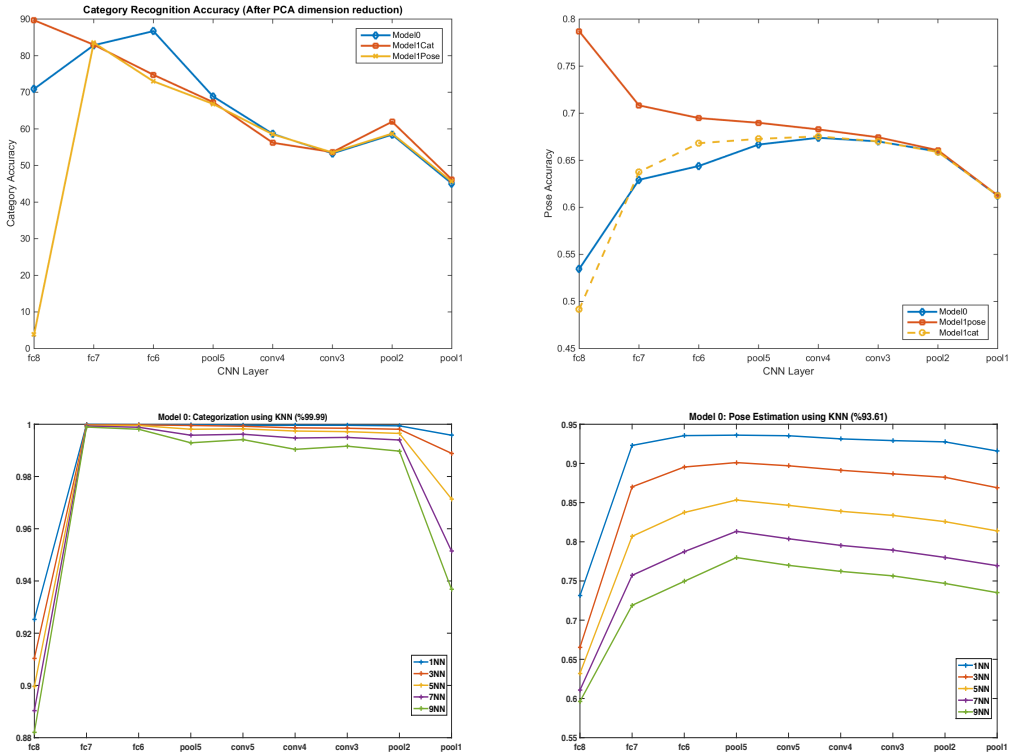


Figure 6.6: Top-Left: the performance of linear SVM category classification over the layers of different model. Top-Right: the performance of pose regression over the layers of different models. Bottom-Left: KNN categorization. Bottom-right: KNN pose estimation.

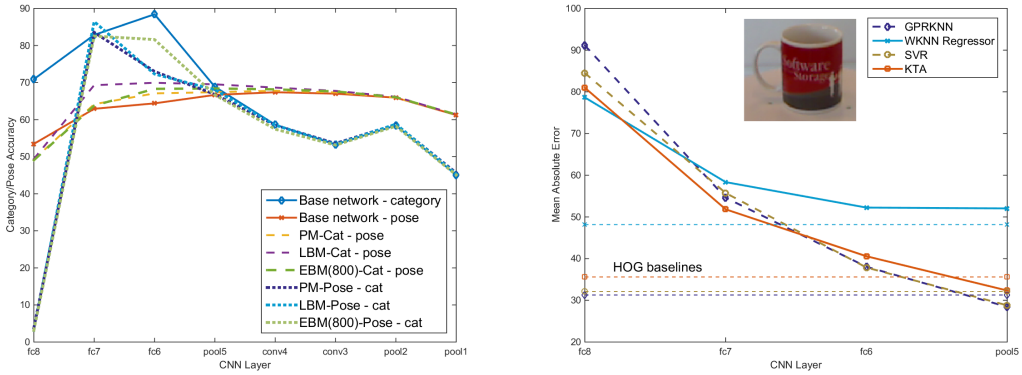


Figure 6.7: Left: Linear SVM categorization and Pose regression Performance based on feature encoding of different layers of a pre-trained CNN over all objects. The dotted lines in the graph are for cross-evaluation: for PM-Cat, LBM-Cat and EBM(800)-Cat represent the models' category representations evaluated on the task of pose estimation (to observe the effect of how category representations encode pose information). Also, PM-Pose, LBM-Pose and EBM(800)-Pose are evaluated on the task of category recognition to see how well pose representations in the network encodes object categories. This is to show the complete pose-invariant representations of the layers of the CNNs when learning category recognition. Right: Pose regression on a single object - this shows the Mean Absolute Error in degrees of various regression algorithms from FC8 (output layer) to Pool5. The horizontal lines represent the regression performance on HOG feature descriptors computed on the images

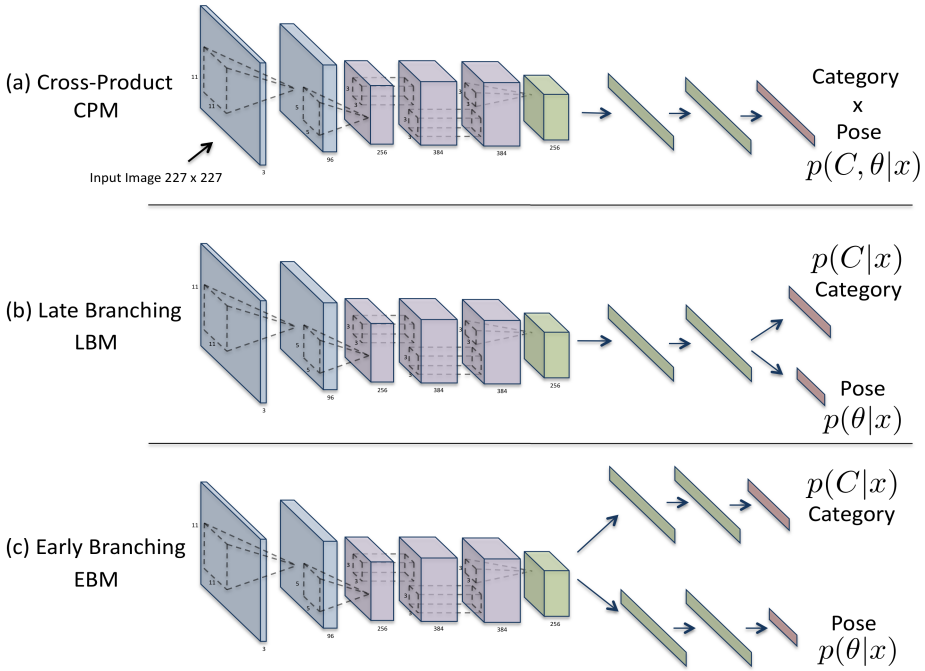


Figure 6.8: The proposed models showing the joint loss layer in CPM, late branching in LBM and early branching in EBM. Blue layers correspond to layers with convolution, pooling and normalization. Violet colored layers correspond to layers with just convolution. Green layers correspond to fully-connected layers.

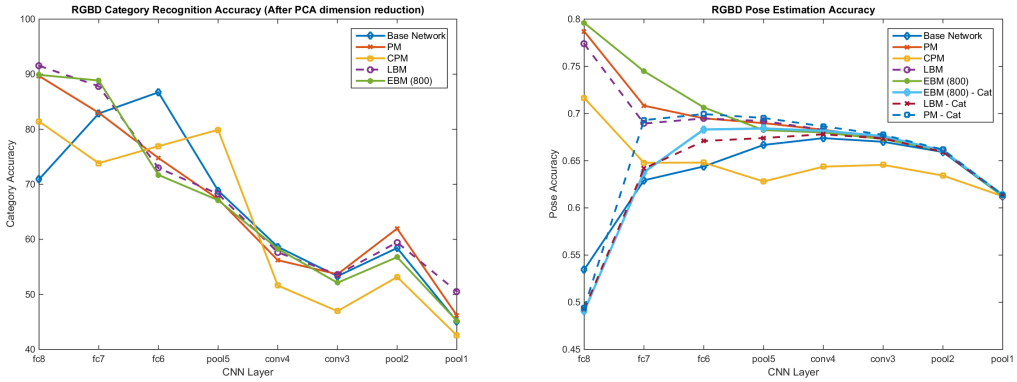


Figure 6.9: Analysis of layers trained on the RGBD dataset. Left: the performance of linear SVM category classification over the layers of different model. Right: the performance of pose regression over the layers of different models (including the category parts of some of the models - this shows the lack of pose information encoded within the object category representations)

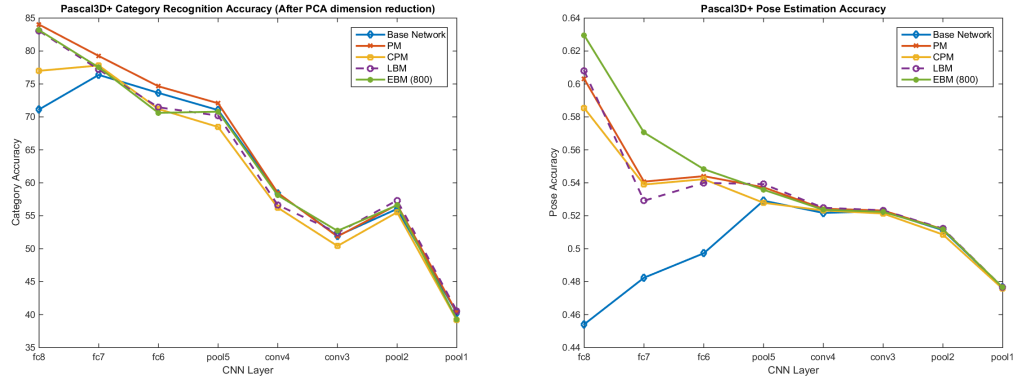


Figure 6.10: Analysis of layers trained on the Pascal3D+ dataset. Left: the performance of linear SVM category classification over the layers of different model. Right: the performance of pose regression over the layers of different models

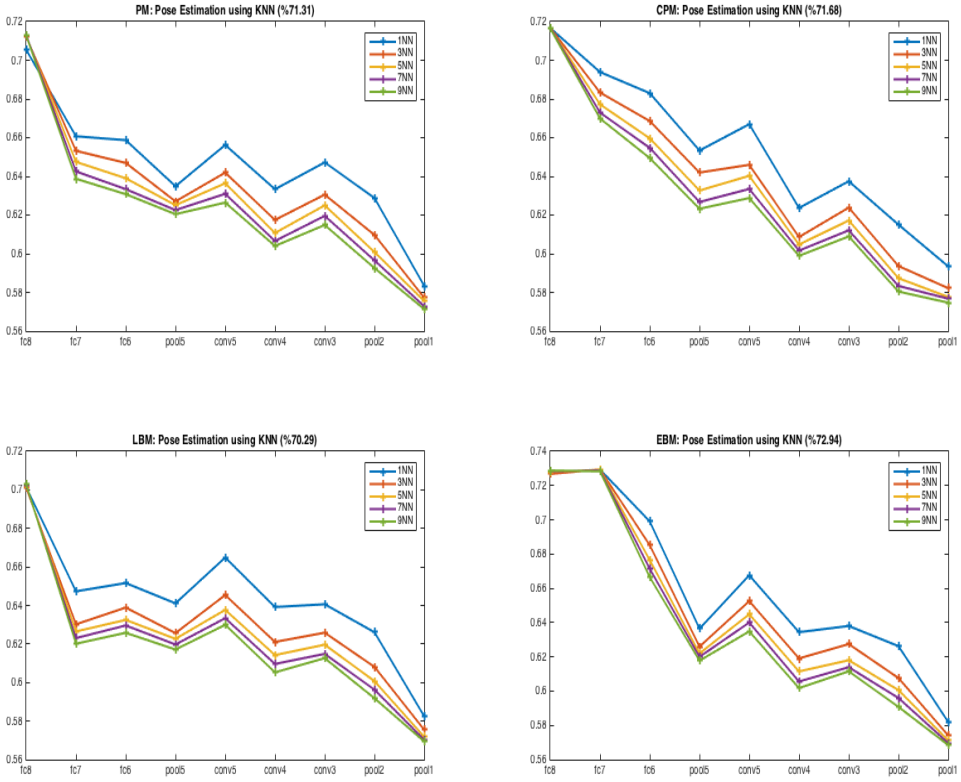


Figure 6.11: A comparison of the pose estimation at each layer of the CNN using k-NN with varying $k = \{1, 3, 5, 7, 9\}$ from top to bottom. It can be observed that EBM converges to the highest pose estimation accuracy one layer before any of the other models. This indicates that objects' view manifolds align with each other and similar poses become close to each other.

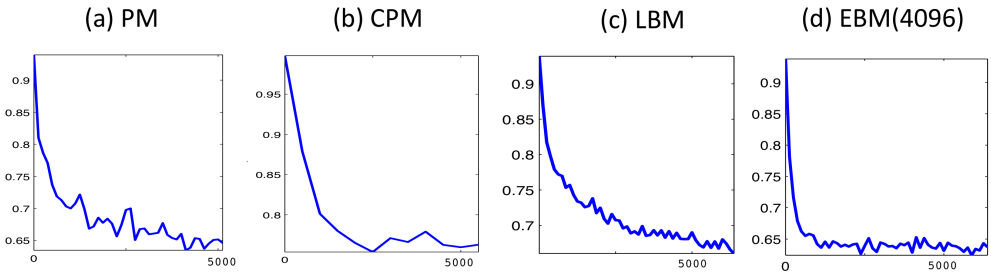


Figure 6.12: Comparison of convergence between the models. On the left is the category error and on the right is the pose error, on the validation set, respectively for each model (a) to (d). The error is computed per batch during each iteration. CPM shows the error for the joint category and pose. It can be seen that EBM (4096) converges much faster than the others which is another benefit of early branching. This is despite have a lot more parameters than the other models. This indicates that each of the subnetworks of EBM are able to specialize in both categorization and pose estimation faster. Each iteration is computed on one batch of 100 training samples.

Chapter 7

3D Joint Object Recognition: Recognition-by-Parts

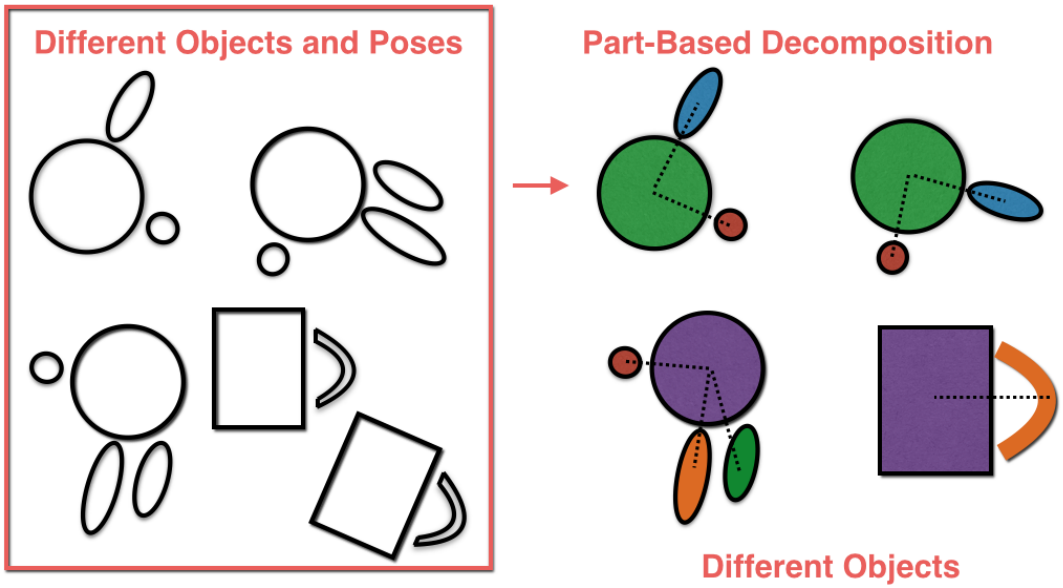


Figure 7.1: A motivational illustration of how part decomposition is key to categorizing multi-part objects and recognizing their poses.

We present a probabilistic approach to shape decomposition that creates a skeleton-based shape representation of a 3D object while simultaneously decomposing it into constituent parts. Our approach probabilistically combines two prominent threads from the shape literature: skeleton-based (medial axis) representations of shape, and part-based representations of shape, in which shapes are combinations of primitive parts. Our approach recasts skeleton-based shape representation as a mixture estimation problem, allowing us to apply probabilistic estimation techniques to the problem of 3D shape decomposition, extending earlier work on the 2D case. The estimated 3D shape decompositions approximate human shape decomposition judgments. We present a tractable implementation of the framework, which begins by over-segmenting objects at concavities, and then probabilistically merges them to create a distribution over possible decompositions. This results in a hierarchy of decompositions at different structural scales, again closely matching known properties of human shape representation. The probabilistic estimation procedures that arise naturally in the model allow effective prediction of missing parts. We present results on shapes from a standard database illustrating the effectiveness of the approach.

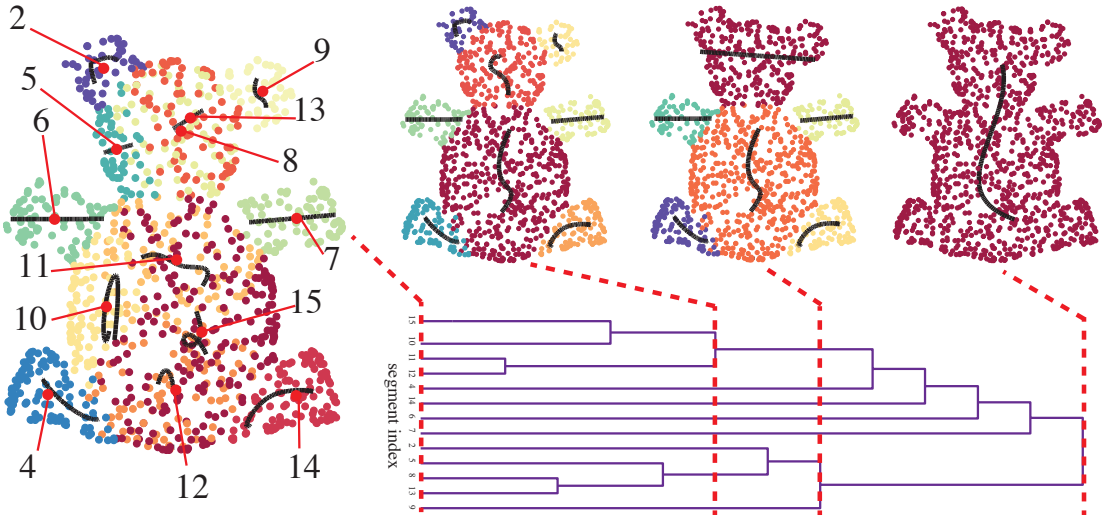


Figure 7.2: Dendogram showing 3D decompositions of a 3D model point cloud. We show 4 decompositions in the hierarchy, starting (left) with an over-segmented interpretation and ending (right) with an under-segmented interpretation in which the whole object is represented as a single central axis. Each interpretation shows a segmentation (color-code) and the most probable skeletal axes under the model.

7.1 Introduction

The goal of this work it to simultaneously decompose an object into its parts and compute a low-dimensional representation that can then be used in categorization and pose estimation of multi-part objects. For example, a coffee mug can be decomposed into the handle and the cylindrical vessel, and each of these primitive parts has a skeleton-based representation expressing its local symmetries (medial axes). The problem of interpreting 3D shape is fundamental in computer vision and robotics, especially with the increasing availability of 3D data. This is key to being able to robustly categorize multi-part objects and recognize object pose (see Figure 7.1 for a motivational illustration).

In 1967, Blum published his pioneering work on representing 2D shapes using the Medial Axis Transform (MAT) [13]. The intuition behind MAT is that shapes are intuitively captured by a lower-dimensional representation that extracts local symmetries of the shape's bounding contours, referred to as the the medial axis. MAT computes the axis from the loci of centers of maximal discs contained within the shape. Evidence supporting medial representations can be found in psychophysics and neuroscience, with recent work showing neural representations in areas V4 and MT of the visual cortex [62]. On another front, early influential work

has suggested that shapes can be recognized or classified based on their part decomposition [93, 57, 11]. Both these lines of thought have led to enormous research literatures, but computational techniques for effectively decomposing 3D shapes into component parts remain poorly developed. These problems remain central challenges today, especially with the emergence of a plethora of 3D data and 3D sensor applications.

We present a *probabilistic* model of object-part decomposition and skeletonization of 3D objects. The contributions of this work are as follows. First, we present a probabilistic estimation procedure for 3D skeleton-based representations of point clouds. Second, simultaneously, our method decomposes an object into intuitively distinct parts. Third, our model allows us to assign degrees of belief (probabilities) to distinct part decomposition interpretations. Fourth, the model provides *hierarchical* interpretations, providing not just a single interpretation but a family of possible interpretations at different levels of abstraction, along with a probability distribution over them. This accords with characteristics of human perceptual organization, and moreover allows the computational system to interpret objects at multiple scales. Finally, the probabilistic nature of the model allows it to identify both the *most probable* object decomposition hypothesis and a distribution over a set of alternative hypotheses.

7.2 Related Work

[127] [127] survey the many advances in shape representation since Blum. They divide shape understanding methods into several types, including landmark representations, boundary representations, displacement by voxel representations and medial representations. By far the most popular and intuitive shape representation is the medial representation. Numerous extension and improvements to the MAT have been proposed, including shock graphs [90], reeb graphs [10], flux graphs [113], object cores [18], multi-scale medial axes [106], Bone graphs [89], Bayesian estimation of the shape skeleton for 2D shapes [41, 46]. Problems with *ligatures*, noise, missing shape information due to occlusion and ambiguity in the final representation haunt many of these models because they inherit the basic limitations of Blum's approach [113, 122]. Improvements have been made by either smoothing over the object's shape before fitting the skeleton or smoothing the skeletal representation after fitting. A discussion of these

methods appears in [96] [96].

Most research on shape, including work nominally oriented towards the 2D case, takes the interpretation of 3D shape as the more fundamental problem [60, 32, 61, 93, 42]. Much of this work focuses on decomposition of shape into primitive parts that are not explicitly medial, and are applied to hand-crafted 3D objects rather than point-cloud data.

More recently, the Computer Graphics community has explored geometric methods for fitting skeletons to 3D shapes. The goal is to use these models to perform shape modelling and analysis for Computer Graphics applications. Two recent methods, [133] [133] and [59] [59], compute skeletons on incomplete and initially unorganized point clouds. The former computes a Rotational Symmetry Axis (ROSA) which captures local symmetry of the object shape. The latter approach improves upon this by creating an L1-median skeleton which locally adapts to the shape. There are many more previous approaches that fit skeletons to 3D object models, discussed further in [59] [59]. Although these approaches are able to find 3D skeletons, they do so using a large set of free parameters, and are not able to assign graded beliefs to different possible decomposition interpretations.

One study [100] approaches the problem from a cognitive perspective by first decomposing objects into parts using segmentation and then computing critical/inflection points based on surface concavities. Skeletons are then extracted from segmented parts by joining them and refining the skeleton to represent the topology of the shape in accordance with *Morse Theory* (the basis for Reeb Graphs 1D skeletons and analyzing the topology of shapes using critical points). The emphasis in this approach on critical points and inflections reflects the prominent role they play in human shape representation [145]. However, in part because of its reliance on critical points, the method tends to fail in the presence of occlusion, though the human system can accommodate missing data with relative ease.

One approach that extracts 2D medial representations from 3D is [96] [96]. This approach, named the Discrete Scale Axis, improves upon Blum's original MAT for 3D objects by following the medial axis transform with topology-preserving angle filtering. Similarly, [128] [128] proposed an algorithm for jointly computing the medial surfaces and object segmentations, resulting in a representation that yields favorable retrieval performance for rigid objects. The problem with 2D medial representations from a Computer Vision perspective (*i.e* for object

recognition/matching) is that 2D skeletal representation (*e.g.* sheets) is still relatively high dimensional. In [96] [96] an example of this is the body of the dolphin which is represented by a relatively vertical sheet. This seems unnecessary for tubular shapes. A sufficient representation can be a curve going through the center of the dolphin's body with some definition of the cross-sectional variation along the main axis as in Biederman's definitions of 3D object parts [11]. Despite this, some more rectangular objects require 2D representations, such as *sheets*.

Several approaches that focus on 3D mesh segmentation have achieved good performance on the Princeton Benchmark for 3D Mesh Segmentation [20] and some also represent shapes using skeletons [124, 147]. Most of these approaches depend on the number segments/parts k being set prior to decomposition, except Core Extraction and Shape Diameter Function. The former decomposes an object hierarchically with a focus on pose invariance using feature points. No notion of probability is defined to identify more probable or intuitive decompositions and this method does not use skeletons to achieve the decomposition. The latter approach is similar to our approach. The authors jointly decompose objects and recover the skeleton using a probabilistic membership function (*i.e.* membership of a vertex into one of the object clusters/parts). The method performs well on articulated objects, but has trouble with non-articulated objects. Many of the aforementioned methods suffer from the same limitations as the traditional MAT approach, which stem from their inherently deterministic nature. Only a small number of approaches aim to estimate medial representations probabilistically. One example of this is [46] [46] who assumes that the underlying skeleton is the generating source of the point-cloud. A probabilistic estimation procedure enables these approaches to handle noisy and missing information, assign beliefs to different decomposition interpretations, incorporate prior knowledge, and predict missing information. These approaches are, however, confined to 2D shapes, and probabilistic approaches have not yet been applied to the 3D case.

7.3 3D Perceptual Object-Part Decomposition

In this section we describe our 3D object-part decomposition approach. It is based on the idea that 3D object-part decomposition can be understood as a Bayesian mixture estimation

problem. We propose a tractable implementation of this framework, which begins by over-segmenting objects at shape concavities, and then probabilistically merging them into a hierarchical representation, creating a distribution over possible decompositions.

7.3.1 Initial Over-Segmentation

The smallest unit of data we consider in this work is a segment of 3D points extracted by Over-Segmentation. In order to find these segments we use Normalized Cuts [125], similar to [68] [68], that prefer segment cuts that lie along concavities. This mimics human perception as humans perceive part boundaries to be concave while the individual parts of the object tend to be convex [57, 68].

Our approach is set up such that Normalized Cuts computes the minimal cuts in the graph connecting the 3D points of the point cloud. The graph edges are assigned weights based on angles between neighboring points' surface normals as follows:

$$w_{ij} = \frac{e^{-\alpha(n_i \cdot (p_j - p_i))/2\sigma}}{d_{ij}} \tag{7.1}$$

where i is the subscript for the central point and j is for the neighbors of point p_i. n are the normals of the corresponding points and d_{ij} is the Euclidean distance between the points. When angles between n_i and $(p_j - p_i)$ are $> 90°$, this means that the surface is convex. The weights in equation 7.1 are larger for more convex surfaces. Additionally, we penalize concavities by using a large α value when the angle is $< 90°$ and using a small α for convex angles $> 90°$. This ensures that optimal cuts are along surface concavities.

The segments found in this way, shown in Fig. 7.3, serve as the basic building blocks in our hierarchical approach below and in addition we use the normalized cut values as an additional cue in determining most-probable merges in the hierarchy. The normalized cut values are equal to the normalized sum of weights on edges of optimal cuts. We scale these optimal cuts in the 0-1 range and combine them with the posterior probabilities of merging two segments. This is described further in the Hierarchical Grouping section.

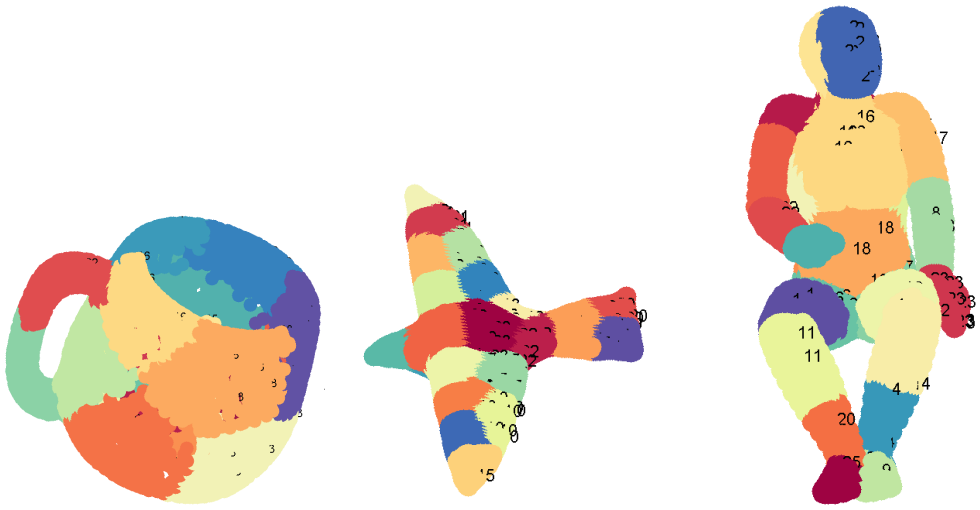


Figure 7.3: Over-Segmentation results for arbitrary large k values. Notice concavities on the surface of the objects constitute good cut boundaries.

7.3.2 Generative 3D Skeletal Representation

Our approach is based on a framework found in [46] [46] for perceptual grouping in general, and 2D part-decomposition specifically, called Bayesian Hierarchical Grouping (BHG). This framework has been shown to make qualitative and quantitative predictions of human behavior across several areas of perceptual grouping, including 2D part-decomposition. Here we apply the framework to the realm of 3D object decomposition with some novel extensions.

Mixture Model

Within BHG each possible part decomposition interpretation $\mathbf{c_j} \in \mathbf{C} = \{\mathbf{c_1} \ldots \mathbf{c_J}\}$ of a set of 3D points $D = \{x_1 \ldots x_N\}$ is associated with a posterior probability $p(\mathbf{C}|D)$, which according to Bayes rule is proportional to the product of a prior probability $p(\mathbf{C})$ and a likelihood $p(D|\mathbf{C})$. Here a decomposition hypothesis $\mathbf{c_j}$ is a vector of labels $\mathbf{c_j} = \{c_1 \ldots c_N\}$ assigning each set of 3D points x_n to a part m. By adopting the BHG framework, both the 3D part decomposition problem and 3D skeletonization can be viewed as a mixture estimation problem. We assume that the configuration of 3D points making up a 3D shape is generated by a mixture of M underlying skeletal axes:

$$p(x_n|\phi) = \sum_{m=1}^{M} p(x_n|\theta_m)p(c_n = m|\mathbf{p}) \tag{7.2}$$

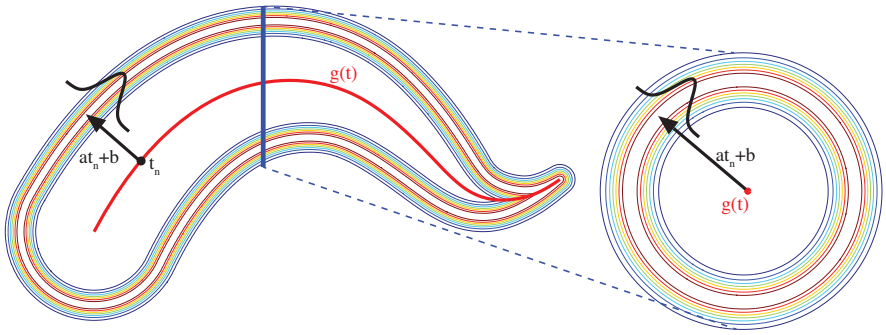


Figure 7.4: Two cuts showing our 3D generative function. Red line depicts the underlying B-spline axis. The gradient depicts the probability $p(x_n|\theta)$, where red has a higher probability.

where $c_n \in \mathbf{c} = \{c_1 \dots c_N\}$ are the assignments of data to an axis, $\mathbf{p}$ is a parameter vector of a multinomial distribution with $p(c_n = m|\mathbf{p}) = p_m$, θ_m are the parameters of the m-th axis, and $\phi = \{\theta_1, \dots, \theta_M, \mathbf{p}\}$. Since the problem of estimating mixture consists of estimating which axis m owns which point x_n and the parameters θ of the underlying axes, both the decomposition problem and the 3D skeletonization problem are solved in tandem.

Generative Function

The generative function $p(x_n|\theta_m)$ depicting how 3D points are generated from an underlying axis generalizes the skeletal generating function proposed in [41] [41] and [46] [46]. A flexible way of representing (and encoding) axes is by means of a 3D parametric B-spline curve $g_m(t)$ (Fig. 7.4), governed by a parameter vector $\mathbf{q_m}$. From this curve data points are generated from a univariate Gaussian distribution perpendicular to the curve,

$$p(x_n|\theta_m) = \mathcal{N}(d_n|a_m t_n + b_m, \sigma_m)$$

where $d_n = ||x_n - g_m(n)||$, called the *riblength*, is the distance between the datapoint x_n and its perpendicular projection on the curve $g_m(n)$, referred to as the *footpoint*. The mean of the normal distribution is defined as a linear function of the footpoint positions t_n, with slope a_m, intercept b_m and σ_m is variance over the riblengths (Fig. 7.4). Together parameter vector for the generative function becomes $\theta_m = \{a_m, b_m, \sigma_m, \mathbf{q_m}\}$. Note that the generative function constrains parts to obey a kind of local symmetry, and the linear function over riblengths allows us to capture conic parts.

In keeping with a full-fledged Bayesian approach a set of priors over the generative function's parameters θ_m is defined, summarized by $p(\theta|\beta)$. Two priors are introduced over the curve g_m itself: one over the squared arclengthF_{1m} of the curve g_m, $F_{1m} \sim \exp(\lambda_1)$ and one over the squared total curvature F_{2m} of the curve g_m, $F_{2m} \sim \exp(\lambda_1)$, both respectively governing the elastic and bending energy of the curve. Lastly, a conjugate prior to the normal distribution is adopted. The prior over the variance σ is $\sigma \sim \mathcal{X}^{-\in}(\sigma_0, \nu_0)$, where σ_0 depicts the mean variance of the riblength, and ν_0 how strongly we believe this. The slope a and intercept b both have a Gaussian prior distribution, $a \sim \mathcal{N}(\mu_0, \kappa_0^{-1})$ and $b \sim \mathcal{N}(\mu_0, \kappa_0^{-1})$. Thus the hyperparameter vector is defined as $\beta = \{\mu_0, \kappa_0, \sigma_0, \nu_0, \lambda_1, \lambda_2\}$

In order to compute the likelihood of a particular decomposition hypothesis $p(D|\mathbf{c_j})$, it is important to integrate over the axis parameters θ for each axis, resulting in the marginal likelihood $p(D_m|\beta) = \int p(D_m|\theta)p(\theta|\beta)d\theta$, where D_m are all the points x_n belonging to axis $c_n = m$. Unfortunately integrating over all possible curves (*i.e.* the parameter space of $\mathbf{q_m}$) is computationally expensive. The marginal is approximated by only integrating over the Gaussian components, which due to the conjugate formulation can be done analytically, and selecting $\mathbf{q_m}$ as to maximize,

$$p(D_m|\beta, \mathbf{q}) = \int \prod_{c_n=m} p(x_n|a, b, \sigma)p(\theta|\beta)dadbd\sigma. \tag{7.3}$$

This equation is maximized as a function of $\mathbf{q}$, by means of a 2-step algorithm (for details see [45], [45]). In step 1 the footpoints t_n for each point x_n is computed. Given these pairs $[x_n, t_n]$, in step 2, we maximize the above equation using *Simulated Annealing* [73]. Both steps are then repeated until convergence. In this way we not only find an at least locally optimal skeletal axis representation for a point cloud segment, but also (as incorporated in $p(D_m|\theta)$) we capture how "part-like" the segment is, given our prior assumptions.

Our approach should be formulated to be independent of any user-specified M components. To ensure independence, a Dirichlet process prior is used, as in [46] [46], over the decomposition hypotheses $p(\mathbf{c_j}|\alpha)$. In this way the above mixture model becomes a Dirichlet process mixture model which is a widely used non-parametric approach [98]. This results in the following definition of the above introduced posterior for a decomposition hypothesis:

$$p(\mathbf{c_j}|D, \alpha, \beta) \propto p(\mathbf{c_j}|\alpha) \prod_{m=1}^{M} p(D_m|\beta), \tag{7.4}$$

where M now denotes the number of axis specific to $\mathbf{c_j}$, and $p(\mathbf{c_j}|\alpha) = \int p(\mathbf{c_j}|\mathbf{p})p(\mathbf{p}|\alpha)d\mathbf{p}$ is a standard Dirichlet integral. Unfortunately the size of the hypothesis space $\mathbf{C}$ is exponential in the number of 3D points, N. The BHG framework approximates this large posterior, by implementing a version of the Bayesian Hierarchical Clustering algorithm by [56] [56]. Moreover, this approach results in an intuitive hierarchical representation of the configuration of 3D points.

7.3.3 Hierarchical Grouping

Bayesian hierarchical clustering [56] follows the same concept as traditional Agglomerative clustering algorithms, with the main difference that a Bayesian hypothesis test is used to decide which clusters to merge. Here we will briefly sketch the algorithm; for more details we refer to [56] [56]. In our implementation the algorithm will initiate with K trees T_k each containing the set of 3D points D_k belonging to the k-th initial segment as found by the over-segmentation stage (see above and Fig. 7.2). At each iteration the algorithm decides which pair of trees (T_i and T_j) are most likely to be merged using a Bayesian hypothesis test, resulting in a new tree $T_{i\cup j}$ containing the data $D_{i\cup j} = D_i \cup D_j$. The pairs of segment regions of the point cloud are considered for merging if they are adjacent to each other in Euclidean space. This is a simple way of drastically reducing the complexity of the computation, since not all pairs must be considered.

In order to compute the likelihood of a merge, two hypothesis are compared. The first hypothesis $\mathcal{H}_{merge}$, treats all data $D_{i\cup j}$ as generated from one underlying axis, and can thus readily be computed as $p(D_{i\cup j}|\mathcal{H}_{merge}) = p(D_{i\cup j}|\beta)$ (see Eq. 7.3). We also benefit from information from the over-segmentation phase which is in the form of the cut values between neighboring segments. We take the product $\log p(D_{i\cup j}|\beta) = Cut(D_i, D_j) \log p(D_{i\cup j}|\beta)$ as our new marginal likelihood (adding to Eq. 7.3). This affects the probability of merging by preferring merges with high cut-values (*i.e.* less concavity between the segments).

The second hypothesis $\mathcal{H}_{nomerge}$, treats all data $D_{i\cup j}$ as generated by more than one axis as constrained by the tree structure. Its probability is computed by taking the product over the subtrees $p(D_{i\cup j}|\mathcal{H}_{nomerge}) = p(D_i|T_i)p(D_j|T_j)$. Here $p(D_i|D_j)$ is computed recursively as

the tree is built:

$$p(D_{i\cup j}|T_{i\cup j}) = p(\mathcal{H}_{merge})p(D_{i\cup j}|\mathcal{H}_{merge}) + (1 - p(\mathcal{H}_{merge}))p(D_i|T_i)p(D_j|T_j)$$

where $p(\mathcal{H}_{merge})$ is based on the Dirichlet process prior and is also computed recursively as the tree is built. The probability of a pair D_i and D_j being merged is:

$$r_{i\cup j} = \frac{p(D_{i\cup j}|\mathcal{H}_{merge})p(\mathcal{H}_{merge})}{p(D_{i\cup j}|T_{i\cup j})}$$

The pair with the highest probability of the merge hypothesis is then merged. In this way the algorithm continues to greedily build the tree until all data is merged into one tree. Note that each level along the tree depicts a different decomposition hypothesis (see Fig. 7.2). The probability of each of these hypotheses is easily computed using Eq. 7.4.

7.4 Experimental Discussion

We experimented on the 3D Dataset from [20], which provides a large dataset of 3D meshes - 380 objects across 19 different categories. We subsample the 3D meshes and reduce the representation to unorganized 3D point clouds. Each point has its corresponding computed surface normal which is used in over-segmentation. We uniformly subsample the point clouds, substantially reducing the number of points.

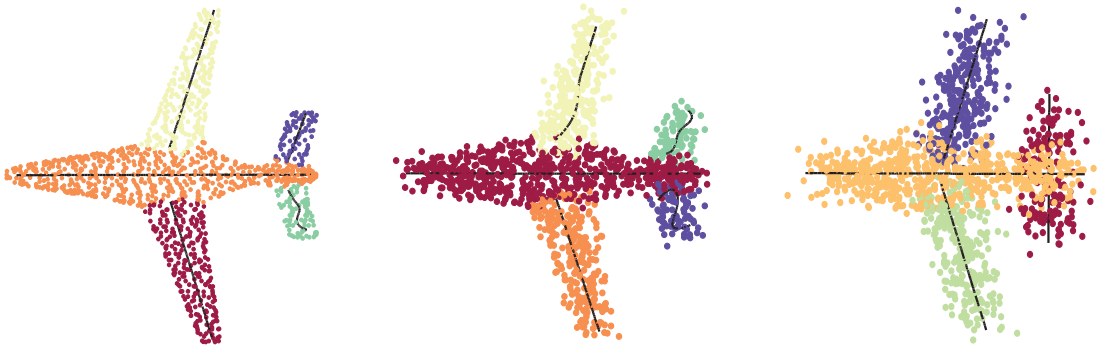


Figure 7.5: Added zero-mean Gaussian noise to one of the airplane object models. Despite noise of up 20% of the mean inter-point distance, the skeletonization and decomposition works well.

Figure 7.2 shows the hierarchical decomposition of a Teddy Bear. The two center decompositions shown correspond to local maxima of the posterior probability in the hierarchy. We find these to be among the most intuitive decompositions. Our maximum-a-posterior (MAP) decomposition is the second one from the right and corresponds exactly to 14 human annotated teddy bears. This particular object may be considered an "easy" example but more complex objects can be seen in Fig. 7.8 and some of their corresponding human segmented counterparts in Fig. 7.9.

We show that our object representations are somewhat robust to noise as shown in Fig. 7.5. We added Gaussian noise with zero-mean and standard deviation equal to 10-20% of the mean inter-point distance of neighboring points. As the noise increases we still obtain a relatively intuitive decomposition.

To validate the generative capability of our model on 3D data, we show how it performs in reverse by generating the input data. Fig. 7.6 and Fig. 7.7 show the original subsampled input point cloud with sampled points from our underlying generative model overlaid. Figure 7.7 is particularly interesting. We remove a hole in the surface of the Martini Glass model and are able to reconstruct it by filling it up with points sampled from the Gaussian distribution of our underlying generative model.

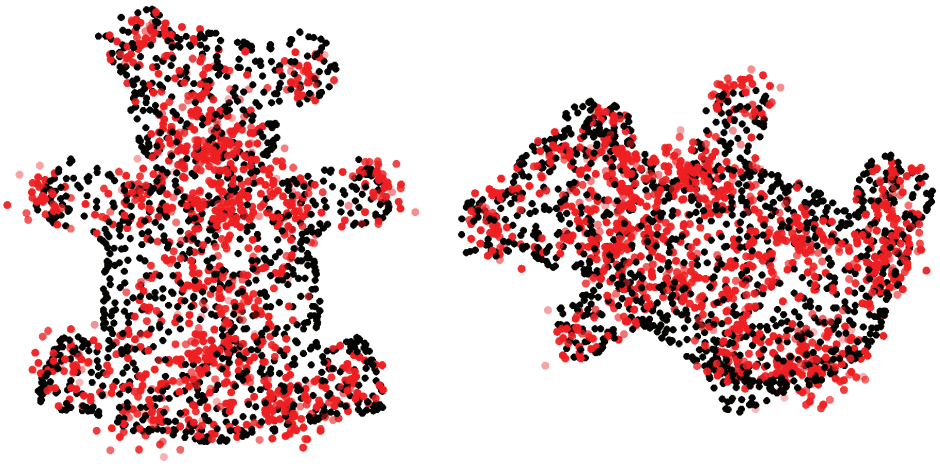


Figure 7.6: Black points are the subsampled input point cloud which is used to estimate the underlying generative model. Red points show sampled points from our underlying probabilistic model (the more opaque being more probable - best viewed using zoom).

In addition, we quantitatively compare our approach to two previous approaches Fig. 7.10.

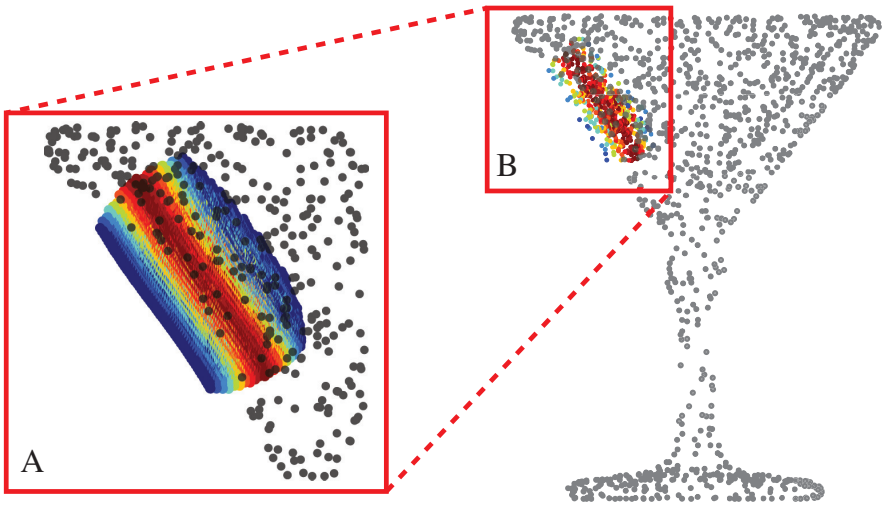


Figure 7.7: This figure shows the ability to predict missing data. Here a hole is cut into the side of object and it is filled by sampling from the underlying generative model. The heat map depicts the probability of that sample (with red being higher probability)

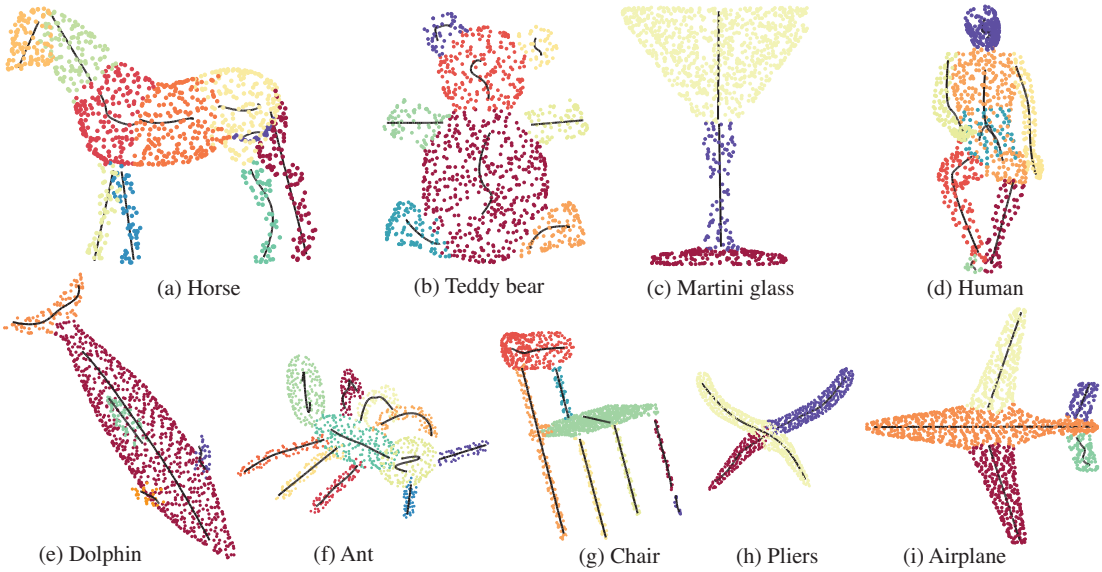


Figure 7.8: MAP object-part decomposition for a broad selection of 3D object models. The black curves in each colored segmented represents the estimated skeletal axis of that segment. Notice, for example, the ability to discriminate the 3 parts of the Martini glass. Notice also the ability to decompose both articulated and non-articulated objects. Some failure cases can be seen in (d). One or two problematic merges happen due to the non-symmetric torso. Despite this, the skeleton is not compromised.

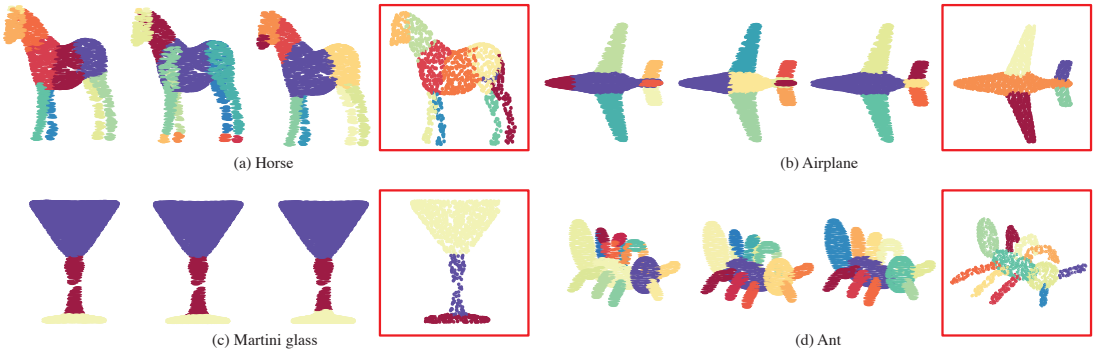


Figure 7.9: For comparison, these are a selection of 4 manually human segmented objects. Each object in the dataset has annotations corresponding to, on average, 11 human subject segmentations. MAP decompositions computed by our approach are shown in the red squares.

These two previous approaches are the only two run on the Princeton Benchmark for 3D Mesh Segmentation [20] that do not assume a prior number of parts. These two methods are the Shape Diameter Function [124] and the Core Extraction approach [69]. We compare with these methods over four standard metrics found in the [20]. These metrics are described in [20] and measure how much the predicted decompositions correspond to the ground-truth annotated segmentations of the object models in the dataset. We see that our approach labeled ”BHG” in the Fig. 7.10 achieves similar performance over the four segmentation metrics. The difference in our approach is that it can represent multiple granularities of decompositions for the same object and also represent the shape boundary of an object probabilistically, in order to be robust to noise. In addition to these benefits of our approach, Fig. 7.10 shows that our approach achieves similar performance to two state-of-the-art approaches.

7.5 Conclusion

We have presented a novel procedure for estimating skeletal representations from 3D point clouds and decomposing them into intuitive component parts. We produce an intuitive hierarchical tree of abstractions and a distribution over them. These multiple representations of an object closely match known properties of human shape representation. We validate this by an experimental study as well as the capability of dealing with noise and predicting missing data which can occur due to occlusion.

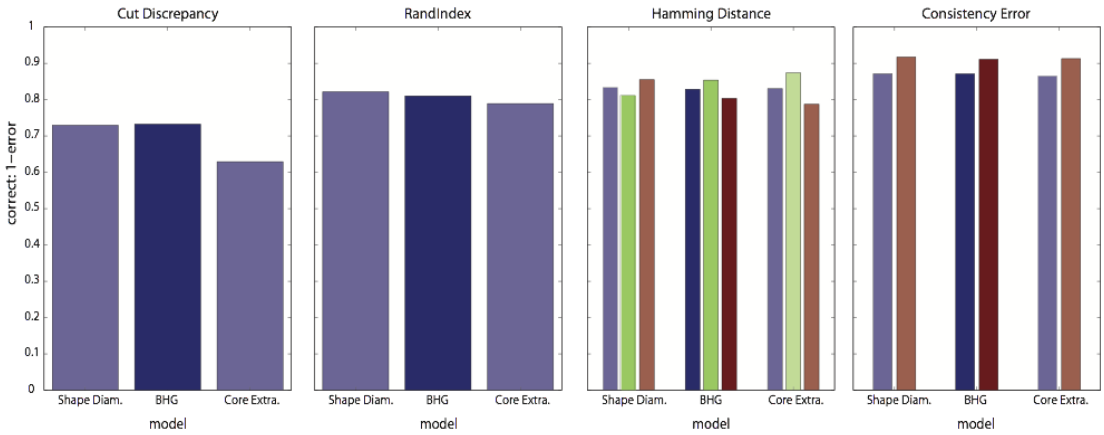


Figure 7.10: Quantitative comparison of our approach, labeled BHG, with two previous methods. The four metrics measures how much the predicted segmentation cuts correspond to the ground-truth segmentation cuts. Hamming Distance and Consistency Error have different variations, shown by the color bars. For Hamming Distance, the dark blue corresponds to the average of the missing rate and false alarm rate, the green is the missing rate and the dark red is the false alarm rate. For the Consistency Error - the dark blue bar corresponds to Global Consistency Error and the dark red corresponds to Local Consistency Error.

Chapter 8

Summary and Conclusion

In this dissertation we have explored machine learning approaches that use different representation spaces that capture the factors involved in solving the problem of simultaneous object categorization and pose estimation. We have presented a multi-kernel learning method for fusing different modalities for object pose estimation in low-dimensional embedding spaces. We have presented a general framework based on topological mapping for simultaneous object categorization and pose estimation across different modalities; with a nonlinear factorization extension to this also. With the advent of Deep Learning techniques, we have explored the representation space that Neural Networks populate and used this end-to-end learning paradigm to perform joint object categorization and pose estimation within a convolutional neural network. Lastly, we have presented an approach for performing probabilistic recognition-by-parts using a representation that inherently captures object pose information by its skeleton representation, making it robust to pose changes and articulation of multi-part objects. We have explored object recognition over multiple modalities in this dissertation - from 2D images to depthmaps and 3D point clouds. We support the hypotheses of the models by extensive experimentation and show how our models achieve state-of-the-art results in object recognition on challenging datasets.

References

[1] E. Aarts and J. Korst. *Simulated annealing and Boltzmann machines: a stochastic approach to combinatorial optimization and neural computing*. John Wiley & Sons, Inc. New York, NY, USA, 1989.

[2] P. Agrawal, R. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. In *Computer Vision–ECCV 2014*, pages 329–344. Springer, 2014.

[3] E. Alpaydin. *Introduction to Machine Learning*. The MIT Press, 2nd edition, 2010.

[4] S. Ando, Y. Kusachi, A. Suzuki, and K. Arakawa. Appearance based pose estimation of 3D object using support vector regression. In *IEEE International Conference on Image Processing*, pages I–341–4, 2005.

[5] Andrzej P. Ruszczynski. *Nonlinear Optimization, Volume 13*. Princeton University Press, 2006.

[6] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2):174–188, 2002.

[7] A. Bakry and A. Elgammal. Untangling object-view manifold for multiview recognition and pose estimation. *ECCV*, 2014.

[8] M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[9] K. Bennett and M. Embrechts. An optimization perspective on kernel partial least squares regression. *Nato Science Series sub series III*, 2003.

[10] S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Reeb graphs for shape analysis and applications. *Theoretical Computer Science*, 392:5 – 22, 2008. Computational Algebraic Geometry and Applications.

[11] I. Biederman. Recognition-by-components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.

[12] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.

[13] H. Blum. A transformation for extracting new descriptors of shape. In W. Wathen-Dunn, editor, *Proc. Models for the Perception of Speech and Visual Form*, pages 362–380, Cambridge, MA, November 1967. MIT Press.

[14] H. Blum. Biological shape and visual science (part i). *Journal of Theoretical Biology*, 38(2):205–287, Feb. 1973.

[15] L. Bo, X. Ren, and D. Fox. Kernel descriptors for visual recognition. In *NIPS 2010*, 2010.

[16] L. Bo, X. Ren, and D. Fox. Depth kernel descriptors for object recognition. *IROS*, 2011.

[17] M. Brand and K. Huang. A unifying theorem for spectral embedding and clustering. In *Proc. of the Ninth International Workshop on AI and Statistics*, 2003.

[18] C. A. Burbeck and S. M. Pizer. Object representation by cores: Identifying and representing primitive spatial regions. *Vision Research*, 35(13):1917 – 1930, 1995.

[19] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *CoRR*, abs/1405.3531, 2014.

[20] X. Chen, A. Golovinskiy, and T. Funkhouser. A benchmark for 3D mesh segmentation. In *SIGGRAPH*, 2009.

[21] H.-P. Chiu, L. Kaelbling, and T. Lozano-Perez. Virtual training for multi-view object class recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[22] N. Cristianini, J. Shawe-Taylor, and J. S. Kandola. Spectral kernel methods for clustering. *NIPS*, 2001.

[23] C. Cyr and B. Kimia. A similarity-based aspect-graph approach to 3D object recognition. *IJCV*, 2004.

[24] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *IEEE Conference on Computer Vision and Pattern Recognition 2005*, 2005.

[25] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[26] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition (CVPR)*, 2009.

[27] J. J. DiCarlo and D. D. Cox. Untangling invariant object recognition. *Trends in cognitive sciences*, 11(8):333–341, 2007.

[28] S. J. Dickinson and Z. Pizlo, editors. *Shape Perception in Human and Computer Vision*. Advances in Computer Vision and Pattern Recognition. Springer London, London, 2013.

[29] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. *arXiv preprint arXiv:1310.1531*, 2013.

[30] J. Duchon. Splines minimizing rotation-invariant semi-norms in sobolev spaces. In *Constructive theory of functions of several variables*, pages 85–100. Springer, 1977.

[31] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, 2nd Edition edition, 2009.

[32] S. Edelman. Computational theories of object recognition. *Trends in Cognitive Science*, 8(1):296–304, 1997.

[33] T. El-Gaaly, M. Torki, A. Elgammal, and M. Singh. RGBD object pose recognition using local-global multi-kernel regression. In *International Conference on Pattern Recognition*, pages 2468–2471, 2012.

[34] A. Elgammal and C.-S. Lee. Inferring 3d body pose from silhouettes using activity manifold learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 681–688, June 2004.

[35] A. Elgammal and C.-S. Lee. Separating style and content on a nonlinear manifold. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 478–485, 2004.

[36] A. Elgammal and C. S. Lee. Separating style and content on a nonlinear manifold. *Computer Vision and Pattern Recognition (CVPR)*, 2004.

[37] A. Elgammal and C.-S. Lee. Homeomorphic manifold analysis (HMA): Generalized separation of style and content on manifolds. *Image and Vision Computing*, 31(4):291–310, April 2013.

[38] M. Everingham, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision (IJCV)*, 2010.

[39] G. Fanelli, M. Dantone, J. Gall, A. Fossati, and L. Van Gool. Random forests for real time 3d face analysis. *International Journal of Computer Vision*, 101(3):437–458, February 2013.

[40] G. Fanelli, T. Weise, J. Gall, and L. Van Gool. Real time head pose estimation from consumer depth cameras. In R. Mester and M. Felsberg, editors, *Pattern Recognition*, volume 6835 of *Lecture Notes in Computer Science*, pages 101–110. Springer Berlin Heidelberg, 2011.

[41] J. Feldman and M. Singh. Bayesian estimation of the shape skeleton. *Proceedings of the National Academy of Sciences*, 103(47):18014–18019, 2006.

[42] J. Feldman, M. Singh, E. Briscoe, V. Froyen, S. Kim, and J. Wilder. An integrated bayesian approach to shape representation and perceptual organization. In S. Dickinson and Z. Pizlo, editors, *Shape perception in human and computer vision*, Advances in Computer vision and pattern recognition. Springer, 2013.

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

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

[45] S. Flöry. Fitting B-spline curves to point clouds in the presence of obstacles. Master's thesis, TU Wien, 2005.

[46] V. Froyen. *Bayesian Mixture Estimation For Perceptual Grouping*. PhD thesis, Rutgers, The State University of New Jersey, 2014.

[47] V. Froyen, J. Feldman, and M. Singh. A bayesian mixture model framework for understanding perceptual grouping. *Configural Processing Consortium (CPC)*, 2013.

[48] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CoRR*, abs/1311.2524, 2013.

[49] G. Gkioxari, B. Hariharan, R. B. Girshick, and J. Malik. R-cnns for pose estimation and action detection. *CoRR*, 2014.

[50] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with hilbert-schmidt norms. In *Algorithmic learning theory*, pages 63–77. Springer, 2005.

[51] W. Grimson and T. Lozano-Perez. Recognition and localization of overlapping parts from sparse data in two and three dimensions. In *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, volume 2, pages 61–66. IEEE, 1985.

[52] C. Gu and X. Ren. Discriminative mixture-of-templates for viewpoint classification. In *Proceedings of the 11th European Conference on Computer Vision: Part V*, ECCV'10, pages 408–421, Berlin, Heidelberg, 2010. Springer-Verlag.

[53] G. Guo, Y. Fu, C. Dyer, and T. Huang. Head pose estimation: Classification or regression? In *International Conference on Pattern Recognition*, pages 1–4, 2008.

[54] J. H. Ham, D. D. Lee, and L. K. Saul. Semisupervised alignment of manifolds. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 120–127, 2005.

[55] M. T. Harandi and C. Sanderson. Graph embedding discriminant analysis on Grassmannian manifolds for improved image set matching. *Computer Vision and Pattern Recognition (CVPR)*, 2011.

[56] K. Heller and Z. Ghahramani. Bayesian hierarchical clustering. In *ICML*, 2005.

[57] D. D. Hoffman and W. A. Richards. Parts of recognition. *Cognition*, 18(1):65–96, 1984.

[58] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 2nd Edition edition, 2012.

[59] H. Huang, S. Wu, D. Cohen-Or, M. Gong, H. Zhang, G. Li, and B.Chen. L1-medial skeleton of point cloud. In *SIGGRAPH*, 2013.

[60] J. E. Hummel and I. Biederman. Dynamic binding in a neural network for shape recognition. *Psychological Review*, 99(3):480–517, 1992.

[61] J. E. Hummel and B. J. Stankiewicz. Two roles for attention in shape perception: A structural description model of visual scrutiny. *VISUAL COGNITION*, 5:49–79, 1998.

[62] C.-C. Hung, E. T. Carlson, and C. E. Connor. Medial axis shape coding in macaque inferotemporal cortex. *Neuron*, 74(6):1099–1113, 2012.

[63] A. Janoch, S. Karayev, Y. Jia, J. Barron, M. Fritz, K. Saenko, and T. Darrell. A category-level 3-d object dataset: Putting the kinect to work. In *Workshop on Consumer Depth Cameras in the International Conference on Computer Vision*, pages 1168–1174, Nov 2011.

[64] K. Jarrett, K. Kavukcuoglu, and Y. Lecun. What is the best multi-stage architecture for object recognition?, 2009.

[65] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 433–449, 1999.

[66] I. T. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.

[67] A. Kapteyn, H. Neudecker, and T. Wansbeek. An approach to n-model component analysis. *Psychometrika*, 51(2):269–275, 1986.

[68] A. Karpathy, S. Miller, and L. Fei-Fei. Object discovery in 3d scenes via shape analysis. In *ICRA*, 2013.

[69] S. Katz, G. Leifman, and A. Tal. Mesh segmentation using feature point and core extraction. *The Visual Computer*, 21(8-10):649–658, 2005.

[70] K. Kavukcuoglu, P. Sermanet, Y.-L. Boureau, K. Gregor, M. Mathieu, and Y. LeCun. Learning convolutional feature hierarchies for visual recognition. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *NIPS*, pages 1090–1098. Curran Associates, Inc., 2010.

[71] G. Kimeldorf and G. Wahba. A correspondence between Bayesian estimation on stochastic processes and smoothing by splines. *The Annals of Mathematical Statistics*, 1970.

[72] G. Kimeldorf and G. Wahba. Some results on tchebycheffian spline functions. *Journal of Mathematical Analysis and Applications*, 33(1):82–95, 1971.

[73] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[74] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012.

[75] A. Kushal, C. Schmid, and J. Ponce. Flexible object models for category-level 3d object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, June 2007.

[76] M. Krtgen, M. Novotni, and R. Klein. 3d shape matching with 3d shape contexts. In *In The 7th Central European Seminar on Computer Graphics*, 2003.

[77] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view rgb-d object dataset. In *ICRA*, 2011.

[78] K. Lai, L. Bo, X. Ren, and D. Fox. A scalable tree-based approach for joint object and pose recognition. In *AAAI*, 2011.

[79] Y. Lamdan and H. Wolfson. Geometric hashing: A general and efficient model-based recognition scheme. In *Computer Vision., Second International Conference on*, pages 238–249, 1988.

[80] L. D. Lathauwer, B. de Moor, and J. Vandewalle. On the best rank-1 and rank-(r1, r2, ..., rn) approximation of higher-order tensors. *SIAM Journal On Matrix Analysis and Applications*, 2000.

[81] N. D. Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, pages 329–336. MIT Press, 2004.

[82] Y. LeCun, F. J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition (CVPR)*, pages 97–104, Washington, DC, USA, 2004. IEEE Computer Society.

[83] P. J. Lewi. Pattern recognition, reflections from a chemometric point of view. *Chemometrics and Intelligent Laboratory Systems*, 1995.

[84] S. LI, Z.-Q. Liu, and A. B. Chan. Heterogeneous multi-task learning for human pose estimation with deep convolutional neural network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2014.

[85] J. Liebelt and C. Schmid. Multi-view object class detection with a 3d geometric model. In *IEEE Conference on Computer Vision and Pattern Recognition 2010*, pages 1688–1695, 2010.

[86] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3d feature maps. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8, 2008.

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

[88] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[89] D. Macrini, S. J. Dickinson, D. J. Fleet, and K. Siddiqi. Bone graphs: Medial shape parsing and abstraction. *Computer Vision and Image Understanding*, 115(7):1044–1061, 2011.

[90] D. Macrini, A. Shokoufandeh, S. J. Dickinson, K. Siddiqi, and S. W. Zucker. View-based 3-d object recognition using shock graphs. In *ICPR (3)*, pages 24–, 2002.

[91] J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. John Wiley & Sons, 1988.

[92] D. Marr. *Vision. A Computational Investigation into the Human Representation and Processing of Visual Information*. The MIT Press, 2010.

[93] D. Marr and H. K. Nishihara. Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes. *Proceedings of the Royal Society of London Biological Sciences*, 200(1140):269–294, 1978.

[94] M. Martinez Torres, A. Collet Romea, and S. Srinivasa. Moped: A scalable and low latency object recognition and pose estimation system. In *Proceedings of ICRA 2010*, May 2010.

[95] L. Mei, J. Liu, A. Hero, and S. Savarese. Robust object pose estimation via statistical manifold modeling. *International Conference on Computer Vision (ICCV)*, 2011.

[96] B. Miklos, J. Giesen, and M. Pauly. Discrete scale axis representations for 3d geometry. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, pages 101:1–101:10, New York, NY, USA, 2010. ACM.

[97] H. Murase and S. Nayar. Visual learning and recognition of 3-D objects from appearance. *International Journal of Computer Vision (IJCV)*, 14(1):5–24, 1995.

[98] R. M. Neal. Markov chain sampling methods for dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265, 2000.

[99] S. A. Nene, S. K. Nayar, and H. Murase. Columbia object image library (COIL-20). Technical Report CUCS-005-96, Columbia University, 1996.

[100] X. Ning, E. Li, X. Zhang, and Y. Wang. Shape decomposition and understanding of point cloud objects based on perceptual information. In *Proceedings of the 9th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry*, VRCAI '10, pages 199–206, New York, NY, USA, 2010. ACM.

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

[102] L. Pan, R. Liu, and M. Xie. Mixture of related regressions for head pose estimation. In *IEEE International Conference on Image Processing*, pages 3647–3651, Sept 2013.

[103] N. Payet and S. Todorovic. From contours to 3D object detection and pose estimation. In *IEEE International Conference on Computer Vision*, 2011.

[104] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3d geometry to deformable part models. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3362–3369. IEEE, 2012.

[105] T. Pfister, K. Simonyan, J. Charles, and A. Zisserman. Deep convolutional neural networks for efficient pose estimation in gesture videos. In *Asian Conference on Computer Vision (ACCV)*, 2014.

[106] S. M. Pizer, C. A. Burbeck, J. M. Coggins, D. S. Fritsch, and B. S. Morse. Object shape before boundary shape: Scale-space medial axes. *Journal of Mathematical Imaging and Vision*, 4(3):303–313, 1994.

[107] Z. Pizlo. 3d shape: Its unique place in visual perception. *LLC*, 26(1):137–138, 2011.

[108] T. Poggio and F. . Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990.

[109] S. Qiu and T. Lane. A framework for multiple kernel support vector regression and its applications to sirna efficacy prediction. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 6(2), Apr. 2009.

[110] M. Ranzato, F. Huang, Y. Boureau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. Computer Vision and Pattern Recognition Conference (CVPR'07)*. IEEE Press, 2007.

[111] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[112] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2014 IEEE Conference on*, pages 512–519. IEEE, 2014.

[113] M. Rezanejad and K. Siddiqi. Flux graphs for 2D shape analysis. In S. Dickinson and Z. Pizlo, editors, *Shape perception in human and computer vision*, Advances in Computer vision and pattern recognition. Springer, 2013.

[114] R. Rosipal and L. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *The Journal of Machine Learning Research*, 2002.

[115] S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Sciene*, 290(5500):2323–2326, 2000.

[116] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. S. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

[117] R. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2155–2162, 2010.

[118] R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation, ICRA 2011, Shanghai, China, 9-13 May 2011*. IEEE, 2011.

[119] S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *International Conference on Computer Vision (ICCV)*, pages 1–8, Oct 2007.

[120] S. Savarese and L. Fei-Fei. View synthesis for recognizing unseen poses of object classes. In D. Forsyth, P. Torr, and A. Zisserman, editors, *ECCV*, volume 5304 of *Lecture Notes in Computer Science*, pages 602–615. Springer Berlin Heidelberg, 2008.

[121] J. Schels, J. Liebelt, and R. Lienhart. Learning an object class representation on a continuous viewsphere. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3170–3177, June 2012.

[122] T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing their shock graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 26(5):550–571, 2004.

[123] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, 2013.

[124] L. Shapira, A. Shamir, and D. Cohen-Or. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer*, 24(4):249–259, 2008.

[125] J. Shi and J. Malik. Normalized cuts and image segmentation. In *PAMI*, 2000.

[126] I. Shimshoni and J. Ponce. Finite-resolution aspect graphs of polyhedral objects. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(4):315–327, 1997.

[127] K. Siddiqi and S. Pizer. *Medial Representations: Mathematics, Algorithms and Applications*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[128] K. Siddiqi, J. Zhang, D. Macrini, A. Shokoufandeh, S. Bouix, and S. Dickinson. Retrieving articulated 3-d models using medial surfaces. *Machine vision and applications*, 19(4):261–275, 2008.

[129] J. Sivic, B. Russell, A. Efros, A. Zisserman, and W. Freeman. Discovering objects and their location in images. In *IEEE International Conference on Computer Vision*, volume 1, pages 370–377, 2005.

[130] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *IEEE 12th International Conference on Computer Vision*, pages 213–220, Sept 2009.

[131] M. Sun, G. Bradski, B.-X. Xu, and S. Savarese. Depth-encoded hough voting for joint object detection and shape recovery. In K. Daniilidis, P. Maragos, and N. Paragios, editors, *Computer Vision ECCV 2010*, volume 6315 of *Lecture Notes in Computer Science*, pages 658–671. Springer Berlin Heidelberg, 2010.

[132] M. Sun, H. Su, S. Savarese, and L. Fei Fei. A multi-view probabilistic model for 3D object classes. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1247–1254, 2009.

[133] A. Tagliasacchi, H. Zhang, and D. Cohen-Or. Curve skeleton extraction from incomplete point cloud. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, New York, NY, USA, 2009. ACM.

[134] A. Tamjidi, C. Ye, and S. Hong. 6-DOF pose estimation of a portable navigation aid for the visually impaired. In *IEEE International Symposium on Robotic and Sensors Environments*, pages 178–183, 2013.

[135] J. B. Tenenbaum. Mapping a manifold of perceptual observations. In M. Jordan, M. Kearns, and S. Solla, editors, *Advances in Neural Information Processing Systems 10*, pages 682–688. MIT Press, 1998.

[136] J. B. Tenenbaum and W. T. Freeman. Separating style and content with bilinear models. *Neural Computation*, 12(6):1247–1283, 2000.

[137] D. Teney and J. Piater. Continuous pose estimation in 2d images at instance and category levels. *2013 International Conference on Computer and Robot Vision*, 2013.

[138] M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[139] M. Torki and A. Elgammal. Regression from local features for viewpoint and pose estimation. In *IEEE International Conference on Computer Vision*, 2011.

[140] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. *CoRR*, abs/1312.4659, 2013.

[141] R. Urtasun, D. Fleet, and P. Fua. 3D people tracking with gaussian process dynamical models. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 238–245, June 2006.

[142] M. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: Tensorfaces. In A. Heyden, G. Sparr, M. Nielsen, and P. Johansen, editors, *Computer Vision ECCV 2002*, volume 2350 of *Lecture Notes in Computer Science*, pages 447–460. Springer Berlin Heidelberg, 2002.

[143] K. Weinberger and L. Saul. Unsupervised learning of image manifolds by semidefinite programming. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 988–995, June 2004.

[144] J. Willamowski, D. Arregui, G. Csurka, C. R. Dance, and L. Fan. Categorizing nine visual classes using local appearance descriptors. In *ICPR 2004 Workshop Learning for Adaptable Visual Systems Cambridge*, 2004.

[145] J. D. Winter and J. Wagemans. Segmentation of object outlines into parts: A large-scale integrative study. *Cognition*, 99(3):275–325, 2006.

[146] H. Wold. Soft Modeling by Latent Variables; the Nonlinear Iterative Partial Least Squares Approach. *Perspectives in Probability and Statistics. Papers in Honour of M. S. Bartlett*, 1975.

[147] S.-K. Wong, J.-A. Yang, T.-C. Ho, and J.-H. Chuang. A skeleton-based approach for consistent segmentation transfer. *Journal of information science and engineering*, 30(4):1053–1070, 2014.

[148] Xbox. Microsoft kinect. www.xbox.com/en-us/kinect.

[149] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2014.

[150] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *ArXiv e-prints*, Nov. 2014.

[151] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013.

[152] H. Zhang, T. El-Gaaly, A. Elgammal, and Z. Jiang. Joint object and pose recognition using Homeomorphic Manifold Analysis. In *Twenty-Seventh AAAI Conference on Artificial Intelligence (AAAI)*, pages 1012–1019, 2013.

[153] H. Zhang, T. El-Gaaly, A. Elgammal, and Z. Jiang. Factorization of view-object manifolds for joint object recognition and pose estimation. *Computer Vision and Image Understanding (CVIU)*, 2015.

[154] S.-C. Zhu. Stochastic jump-diffusion process for computing medial axes in markov random fields. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(11):1158–1169, Nov. 1999.