

UCLA

UCLA Electronic Theses and Dissertations

Title

Integrating 3D and 2D Representations for View Invariant Object Recognition

Permalink

<https://escholarship.org/uc/item/7sv1c5w3>

Author

HU, WENZE

Publication Date

2012

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

**Integrating 3D and 2D Representations for View
Invariant Object Recognition**

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Statistics

by

Wenze Hu

2012

© Copyright by

Wenze Hu

2012

ABSTRACT OF THE DISSERTATION

Integrating 3D and 2D Representations for View Invariant Object Recognition

by

Wenze Hu

Doctor of Philosophy in Statistics

University of California, Los Angeles, 2012

Professor Song-Chun Zhu, Chair

This thesis presents representations and corresponding algorithms which learn models to recognize objects in the full continuous view space. Particularly, we propose to integrate the 3D object-centered representations with 2D viewer-centered representations, which fills in the representation gap between the sparse and simple 3D shapes and their view variant appearances observed as image pixels. Towards this goal, this thesis studies the following models and corresponding algorithms:

1. A mixed model and a pursuit algorithm that integrates 3D object primitives and 2D image primitives according to their information contributions measured as information gains. This proposed measure is consistently used in subsequent models, and also provides a numerical answer to the debates over object-centered representation and viewer-centered representation.
2. A 2D compositional image model and a sum-max data structure which groups the 2D image primitives to represent middle level image structures, such as line segments, curves and corners. This middle level image model can be used to find sparse representations of natural images, and connects the low level 2D image representations to 3D object representations.

3. A 3D hierarchical compositional object model and an AND-OR tree structure which represents a huge number of possible 3D object templates using a limited number of nodes. This AND-OR tree hierarchically quantizes the infinite and continuous space of object geometry and appearance, and decomposes the 3D object representation into 3D panels, whose appearance on images are further decomposed into active curves and the 2D primitives. Though with multiple hierarchies, learning and inference can be done efficiently by dynamic programming, which is essentially composed of layers of sum and max operations.

The dissertation of Wenze Hu is approved.

Hongquan Xu

Luminita Aura VESE

Ying Nian Wu

Song-Chun Zhu, Committee Chair

University of California, Los Angeles

2012

To my parents, my wife and my daughter.

TABLE OF CONTENTS

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | Motivations and Objective | 1 |
| 1.2 | Background I: Structural 3D Object Representations | 3 |
| 1.2.1 | Marr’s Theory | 3 |
| 1.2.2 | Recognition by Components | 4 |
| 1.3 | Background II: Image Representations by Sparse Coding | 7 |
| 1.3.1 | Olshausen-Field Model | 7 |
| 1.3.2 | Active Basis Model | 8 |
| 1.4 | Organization | 10 |
| 2 | A Flat Probabilistic Model Mixing 3D and 2D Primitives | 11 |
| 2.1 | Introduction | 11 |
| 2.2 | Related Work | 13 |
| 2.3 | Dictionaries of 3D and 2D Primitives | 15 |
| 2.3.1 | 3D Object Primitives | 15 |
| 2.3.2 | 2D Image Primitives | 18 |
| 2.4 | Probabilistic Image Model | 19 |
| 2.5 | Learning Algorithm | 21 |
| 2.5.1 | Evaluate the Information Gain of a Primitive | 23 |
| 2.5.2 | Shared Primitive Pursuit | 26 |
| 2.6 | Inference Algorithm | 27 |
| 2.7 | Experiments | 30 |

| | | |
|----------|--|-----------|
| 2.7.1 | Dataset, Image Pre-processing and Parameters | 30 |
| 2.7.2 | Learning Mixed Templates | 32 |
| 2.7.3 | Learning 3D Templates for Car Recognition and Viewpoint In- ference | 34 |
| 2.8 | Discussion | 37 |
| 3 | Composing Image Primitives to Active Curves | 38 |
| 3.1 | Introduction | 38 |
| 3.2 | Related Work | 41 |
| 3.3 | Active Curve and Corner Templates | 42 |
| 3.3.1 | Active Curves | 42 |
| 3.3.2 | Active Corners | 44 |
| 3.4 | Image Model and Template Matching Score | 45 |
| 3.5 | Scoring All Templates by Sum-Max Maps | 46 |
| 3.6 | Selection of Corner and Curve Templates for Single Image | 50 |
| 3.7 | Active Curves and Corners as Features for Object Images | 51 |
| 3.8 | Experiments | 51 |
| 3.8.1 | Implementation and Parameters | 51 |
| 3.8.2 | Sketch Natural Images | 52 |
| 3.8.3 | Learning from Object Images | 53 |
| 3.9 | Discussion | 64 |
| 4 | Integrating 3D and 2D Representations by AND-OR Tree | 66 |
| 4.1 | Introduction | 67 |
| 4.1.1 | Motivation and objective | 67 |

| | | |
|-------|---|----|
| 4.1.2 | Overview of the Proposed Method | 68 |
| 4.1.3 | Related Literature | 70 |
| 4.1.4 | Contributions | 73 |
| 4.2 | AoT for Space Quantization | 74 |
| 4.2.1 | G-AoT for Part Geometry | 74 |
| 4.2.2 | A-AoT for Part Appearance | 77 |
| 4.2.3 | Instantiation of AoT | 82 |
| 4.2.4 | Parse Trees as Samples of the Quantized Space | 83 |
| 4.3 | Probabilistic Image Model | 84 |
| 4.3.1 | Probability Density Decomposition | 84 |
| 4.3.2 | Image Modeling by Density Substitution | 85 |
| 4.4 | Template Learning by Dynamic Programming | 86 |
| 4.4.1 | Information Gain as the Objective for AND-OR Search | 87 |
| 4.4.2 | Information Gain as the Objective for AND-OR Search | 87 |
| 4.4.3 | AND-OR Search Algorithm | 88 |
| 4.5 | Inference Scheme | 91 |
| 4.5.1 | Converting 3D Template to 2D Templates | 91 |
| 4.5.2 | Feature Weight Adjustment | 92 |
| 4.5.3 | Hypothesis Verification by Color Histogram | 92 |
| 4.6 | Experiments | 93 |
| 4.6.1 | Image Dataset and Parameters | 93 |
| 4.6.2 | Scales of AoT as a Function of Volume Size | 95 |
| 4.6.3 | Representation Power of AoT and Octree | 96 |

| | | |
|----------|--|------------|
| 4.6.4 | Learning Object Templates | 98 |
| 4.6.5 | Object Recognition Experiments | 98 |
| 4.7 | Discussion | 101 |
| 5 | Conclusions and Future Work | 104 |
| 5.1 | Thesis Summary | 104 |
| 5.2 | Future Work | 105 |

LIST OF FIGURES

| | | |
|------|--|----|
| 1.1 | Samples of geons and how they are generated | 4 |
| 1.2 | An overview of the 3D approach proposed by Dickinson | 6 |
| 2.1 | An illustration of a template mixing 3D and 2D primitives. | 12 |
| 2.2 | An overview of the mixed template representation | 16 |
| 2.3 | An overview of the image generating model | 19 |
| 2.4 | Primitive information gain as a function of μ_r | 24 |
| 2.5 | An illustration of 3D primitive pursuit | 26 |
| 2.6 | The conversion of Ω from a set to a 2 dimensional lookup table | 29 |
| 2.7 | A snapshot of the user interface for image view annotation. | 31 |
| 2.8 | Learned object templates for 10 object categories and their pursuit indexes | 33 |
| 2.9 | Results of pose estimation task | 35 |
| 2.10 | The performance and sample results of detection task | 36 |
| 3.1 | An illustration of proposed visual codeword hierarchy | 40 |
| 3.2 | An illustration of active curve templates | 43 |
| 3.3 | An illustration of active corner templates | 46 |
| 3.4 | How indices of Active Curves are changed | 49 |
| 3.5 | An example of the sum-max data structure | 54 |
| 3.6 | More results on the single image representation | 55 |
| 3.7 | Representing images of objects | 56 |
| 3.8 | Learned templates by adaBoost, using active basis and active curves as features. For each case, | 57 |

| | | |
|------|---|-----|
| 3.9 | AUC curves of boosted classifiers on horse images and leaves images . . . | 58 |
| 3.10 | A comparison of learned object templates using active basis and active curves | 59 |
| 3.11 | Learned object templates and their deformations on training images . . . | 60 |
| 3.12 | Sample detection results on ETHZ dataset | 62 |
| 3.13 | The object detection performance on the ETHZ dataset. | 63 |
| 4.1 | An overview of the proposed 3D object representation. | 69 |
| 4.2 | An overview of the AND-OR Tree for the representation space | 70 |
| 4.3 | Volume quantization and decompositions | 75 |
| 4.4 | An Example of the Geometry AND-OR Tree | 76 |
| 4.5 | An example of the Appearance AND-OR Trees | 79 |
| 4.6 | An example of the deformation for shape templates | 80 |
| 4.7 | View distribution and sample images of our new dataset | 94 |
| 4.8 | Comparison of representation power between AoT and octree | 97 |
| 4.9 | Projection of 3D template to 2D templates, their decomposition, and corresponding detection score maps. | 99 |
| 4.10 | Object detection performance on our proposed dataset | 100 |
| 4.11 | Pose estimation error on our newly collected dataset. | 101 |
| 4.12 | Object recognition performance on the 3D car dataset [Savarese and Fei-Fei, 2007] | 102 |
| 4.13 | Results on post estimation task | 102 |
| 4.14 | Sample object detection and post estimation results | 103 |

LIST OF TABLES

| | | |
|-----|---|----|
| 3.1 | Parameters for all the experiments on single images | 52 |
| 4.1 | List of visual concepts and their details | 78 |
| 4.2 | The scale of AoT and the number of possible 3D deformable templates as bounding volume size grows. | 95 |

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Professor Song-Chun Zhu, for giving me the opportunity to conduct fascinating research in computer vision and applied statistics, as well as his invaluable support and visionary guidance during my study on this particular research direction.

My gratitude also goes to my other past or current thesis committee members: Professors Stefano Soatto, Luminita Vese, Yingnian Wu, Hongquan Xu and Alan L. Yuille. I thank professor Ying Nian Wu, as he is always happy and open to discuss any issues related to my research, give me detailed suggestions and share his experience of research. His minimalism style in pursuing simple yet effective models deeply affected my thinking and practice on the subject of my research. I thank other members, for their insightful discussions, advice and help in various aspects.

I feel very fortunate to have worked with my excellent fellow group members at UCLA and collaborators at Beijing Institute of Technology: (in alphabetical order) Haifeng Gong, Zhi Han, Yi Hong, Jungseock Joo, Bo Li, Amy Morrow, Seyoung Park, Maria Pavlovskaja, Mingtao Pei, Arash G. Rad, Brandon Rothrock, Kent Shi, Yao Shi, Zhangzhang Si, Qiongchen Wang, Shuo Wang, Ping Wei, Tianfu Wu, Dan Xie, Xingyao Ye, Zhenyu Yao, Mingtian Zhao, Yibiao Zhao, Jianguan Zhang and Bo Zheng. I have benefited greatly from their peer interactions and friendships during the past four years.

I am also very fortunate to have the help from members in image annotation group in Lotus Hill Institute. They helped me a lot in collecting and labeling data.

I would also like to thank my past roommates: Liang Lin, Lin Nie, Alan Lee and Craig Lap-Fai Yu. As graduate students in vision and highly related fields, my discussions with them give me new knowledge and perspectives about the direction I am studying.

Last, but not least, I would like to dedicate this thesis to my parents, my wife Celia Xiaoyu Xu, for their love, patience, and understanding. During the last month, it is my wife and parents in law who took care of our one month old Ruby, their offering allowed me to spend most of the time on this thesis.

VITA

- 2006 B.S., Optical Information Science and Technology, Beijing Jiaotong University
- 2008 M.S., Optical Engineering, Beijing Institute of Technology
- 2009 C.Phil., Statistics, University of California, Los Angeles
- 2010 M.S., Statistics, University of California, Los Angeles
- 2009-2012 Graduate Student Researcher, Department of Statistics, University of California, Los Angeles

PUBLICATIONS

W. Hu, H. Gong, S.-C. Zhu, and Y. Wang. An Integrated Background Model for Video Surveillance Based on Primal Sketch and 3D Scene Geometry. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008

W. Hu and S.-C. Zhu. Learning a Probabilistic Model Mixing 3D and 2D Primitives for View Invariant Object Recognition. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010

W. Hu, Y. N. Wu and S.-C. Zhu. Image Representation by Active Curves. In *Proceedings of International Conference on Computer Vision (ICCV)*, 2011

J. Zhang, **W. Hu**, B. Yao, Y. Wang, and S.-C. Zhu. Inferring Social Roles in Long Timespan Video Sequence. In *Proceedings of International Workshop on Video Event Categorization, Tagging and Retrieval for Real World Applications*, 2011

W. Hu. Learning 3D Object Templates by Hierarchical Quantization of Geometry and Appearance Spaces. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012

CHAPTER 1

Introduction

1.1 Motivations and Objective

Through retinal images, we can easily perceive and categorize objects regardless of their views. Starting from 1970s, the vision research community proposed various theories and frameworks, trying to explain the mechanism of view invariant object recognition in human vision and empower the computer similar capability.

With a limited amount of images for validation, researchers at the early stage emphasis on developing coherent frameworks and finding principled approaches that offer invariance to viewpoint and small object structure change, as well as to other factors such as articulation and occlusion. Many of these early systems employ 3D object representations, as intuitively the effects of these factors can be synthesized with limited operations in 3D object space. Despite powerful modeling paradigms, these early systems lacked the low- and middle-level processing that would recover their assumed intermediate representations, such as boundaries and junctions from real images of real objects.

With the increasing interest in getting high performance on large amount of real images, researchers in current era migrate away from 3D models, and take their object categorization models more closer to images, using easily extractable image appearance representations and recent advances from machine learning and optimization. This trend popularized many image descriptors such as SIFT [Lowe, 1999], SURF [Bay

et al., 2008] and HoG[Dalal and Triggs, 2005] etc., which trim the dimension of image patch sets to the level that optimization methods can process, and offer some level of invariance to factors such as image noise, contrast change, and local displacement of certain image features. However, this trend also makes the result classifiers view specific, because the readily available tools from machine learning do not model this vision specific effect.

On image datasets with limited views, researchers propose to solve this problem by combining multiple view-specific classifiers following the theory of aspect graph. However, a quick review of the aspect graph literature in 90s immediately reveals its problem: there are simply too many aspects if we extend this approach to model the whole view sphere.

In this thesis, we want to go back to the 3D representations, and ground them with image pixels using 2D image appearance models, so that the 3D model can be learned and applied to recognize objects in real images invariant to view change. We hope that by having 3D object representations, we can further take advantage of the early research and better solve other vision problems such as occlusion. Towards this goal, we have studied the following models:

1. A flat model that integrates 3D object primitives and 2D image primitives, which combines and selects primitives according to their information gains. This model and the corresponding algorithm provide a numerical answer to the debates over object-centered representation and viewer-centered representation.
2. A 2D compositional model (active curves) that groups the 2D image primitives (active basis) to represent middle level visual concepts, such as line segments, curves and corners. These middle level concepts connect the low level 2D image representations to 3D hierarchical compositional object representations.
3. A 3D hierarchical compositional object model and an AND-OR tree structure

from which the model can be learned. This AND-OR tree hierarchically quantizes the infinite and continuous space of object geometry and appearance, and fills in the representation gap between the stylized 3D shape components and their 2D image observations as pixel intensities.

Before presenting the details of these models and algorithms, we want to first introduce the following models and methods which inspire this thesis.

1.2 Background I: Structural 3D Object Representations

Many early theories of object recognition propose to represent objects in 3D space. In these theories, objects are composed of parts and represented by these parts and their spatial relationships. Usually, the used parts are from a small set of stylized 3D shapes. The idea was first proposed by Binford [Binford, 1971], and are further developed by lots of researchers including Marr [Marr and Nishihara, 1978], Biederman [Biederman, 1987], and Dickinson [Dickinson et al., 1992] from both human vision and the computer vision perspective.

1.2.1 Marr's Theory

In the seminal work by David Marr, the final stage of the proposed vision paradigm is forming a 3D shape and a 3D model. To represent these shapes, Marr proposed volumetric features which are called primitives. The complexity of these primitives varies from simple units such as small cubes, to complex shapes of object parts, such as limbs of animals.

Marr further argues that these primitives should be organized in modules, representing the hierarchical relations of the entire 3D object to its parts. For example, a human body is consisted of several parts such as limbs, a torso, and a head. Each limb is then

decomposed into two components: lower and upper limb and are connected by a joint. The lower arm part has in turn is composed of two parts and so on (see Marr's figure 5.3 in [Marr, 1982]).

To recognize these models, Marr proposes that the 3D shape of objects is formed using certain depth cues and a 2-1/2D sketch representation, which in turn uses a primal sketch representation derived from retinal image. In this process, Marr emphasizes a transition from image based viewer-centered representation to an object-centered representation, where the later one is independent of the view.

1.2.2 Recognition by Components

1.2.2.1 Biederman's Theory

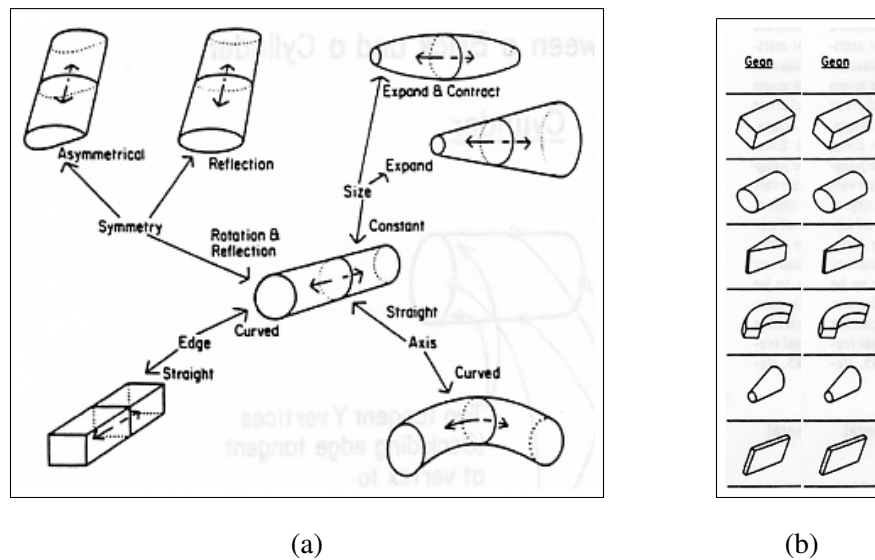


Figure 1.1: (a) An illustration of how geons can be germinated by varying attributes of a cross section. (b) Some examples of proposed geons. Both figures are adapted from [Biederman, 1987] with permission.

Biederman [Biederman, 1987] proposes that a 3D object of arbitrary shape can be represented as placements of several simple parts called as "geons". The geons

resemble cubes, cylinders, cones and spheres. In his recognition by components (RBS) theory, objects are differentiated from each other by the specific types of geons used and how they are arranged in 3D space.

By considering all possible compositions of basic geometric features such as junctions, and curvatures of part contours, Biederman derived (though a bit arbitrarily) that as few as thirty-six geons would be sufficient to model the universe of 3D shapes. This dramatically reduced the set of infinitely many generalized shapes that were described by Binford.

Quite different from Marr, Biederman emphasized that the recognition of 3D shape depends on the recognition of its component geons and their 3D spatial relations. This argument is quite natural as many objects have parts. It also makes the recognition scales up easily, as we only need to recognize a fixed set of primitive shapes instead of a much larger set of object shapes. However, as to recognition of these geons from images, Biederman simply connects it to the figure-ground organization, and assumes the components of geons, the curves, and groups of curves as well as junctions could be recognized easily.

1.2.2.2 Developments by Pentland and Dickinson

Pentland [Pentland, 1986] proposed a set of fifty-six superquadrics as the primitives for 3D object representation. Different from Biederman, these shapes are all characterized by a small set of parameters. By changing parameters, it was able to produce fifty-six shape families. Thus this set is able to describe more closely to real 3D shapes, and categories object shapes according to their shape families.

Dickinson further developed this theory on the modeling and recognition process. While still keeping the idea of recognition by parts and parametric shape families, he further proposed an object modeling/recognition framework that used 2D views to

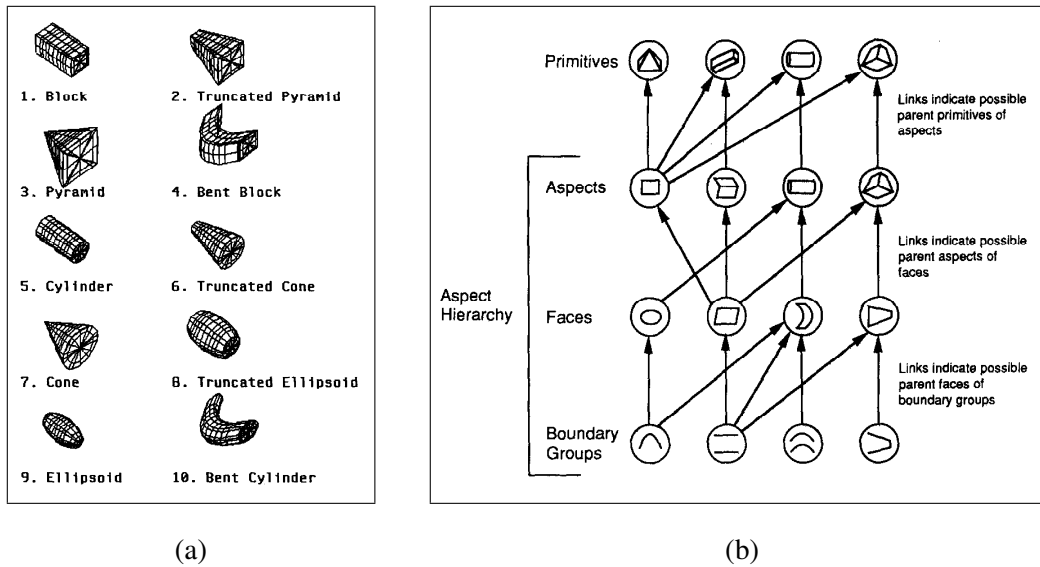


Figure 1.2: An overview of the 3D approach proposed by Dickinson [Dickinson et al., 1991]. (a) The proposed 10 qualitative volumetric primitives. (b) The proposed aspect hierarchy. Both figures are adapted from [Dickinson et al., 1991]. ©[1991] IEEE.

model a finite set of 3D parts. This approach connects to but are also significantly different traditional aspect graph approaches, which model entire objects and where the number of aspects grows with object complexity and with the number of parts. In his new approach, the database of views is fixed, and its size is only related to the size of the 3D shape dictionary.

Moreover, the views of parts are organized into an aspect hierarchy, consisting of regions, component regions and contours for the component regions. This hierarchy produces a series of equivalent representations of volumetric parts in different views, which indicates a perceptual organization process that could assemble local and potentially ambiguous evidence from images.

To summarize, all these theories and frameworks propose representing objects in 3D using a small set of simple and regularized shapes. While constructing interesting roadmaps to achieving recognition, the details of how these shapes can be recovered

from the image are all rather obscure.

1.3 Background II: Image Representations by Sparse Coding

Besides reconstructing 3D shapes, vision researchers also want to find simple models that reconstruct input images. This leads to the sparse coding models that generate images using linear addition of a small set of wavelets or image primitives. In this section, we will briefly review these models as they laid the foundation for the image appearance part of our 3D object model.

1.3.1 Olshausen-Field Model

Let $\{\mathbf{I}_m, m = 1, \dots, M\}$ be a set of training image patches (e.g. 12×12), Olshausen-Field model seeks to represent these images by

$$\mathbf{I}_m = \sum_{i=1}^N c_{m,i} B_i + U_m, \quad (1.1)$$

where $(B_i, i = 1, \dots, N)$ is a dictionary of basis elements of the same dimensionality as \mathbf{I}_m , $c_{m,i}$ are the coefficients, and U_m is the unexplained residual image. N is often assumed to be greater than the dimensionality of \mathbf{I}_m (e.g. $N = 2 \times 12 \times 12$), so the dictionary is said to be overcomplete. On the other hand, the number of coefficients $(c_{m,i}, i = 1, \dots, N)$ that are non-zero or significantly different from zero is assumed to be very small for each image \mathbf{I}_m . The dictionary of $(B_i, i = 1, \dots, N)$ can be learned automatically from the training images $\{\mathbf{I}_m\}$ by imposing a sparsity constraint.

One may also assume that the dictionary of the basis elements are translated, rotated and dilated version of one another, as in Olshausen et al. [Olshausen et al., 2001], so that each B_i can be written as $B_{x,s,\alpha}$, where x is the location (a two-dimensional vector), s is the scale, and α is the orientation. We call such a dictionary self-similar, and we call (x, s, α) the geometry attribute of $B_{x,s,\alpha}$.

From now on, we assume that the dictionary of wavelets is self-similar, and $(B_{x,s,\alpha}, \forall(x, s, \alpha))$ is already available. It can either be learned or designed. In the following, we assume that $B_{x,s,\alpha}$ is a Gabor wavelet, and we also assume that $B_{x,s,\alpha}$ is normalized to have unit ℓ_2 norm so that $|B_{x,y,\alpha}|^2 = 1$. $B_{x,s,\alpha}$ may also be a pair of Gabor sine and cosine wavelets, so that for each Gabor wavelet B , $B = (B_0, B_1)$. The corresponding coefficient $c = (c_0, c_1)$, and $cB = c_0B_0 + c_1B_1$. For projection $\langle \mathbf{I}, B \rangle = (\langle \mathbf{I}, B_0 \rangle, \langle \mathbf{I}, B_1 \rangle)$, and $|\langle \mathbf{I}, B \rangle|^2 = \langle \mathbf{I}, B_0 \rangle^2 + \langle \mathbf{I}, B_1 \rangle^2$.

Given the dictionary $(B_{x,s,\alpha}, \forall(x, s, \alpha))$, the image patch \mathbf{I}_m which could be much larger than 12×12 can be encoded by

$$\mathbf{I}_m = \sum_{i=1}^n c_{m,i} B_{x_{m,i}, s_{m,i}, \alpha_{m,i}} + U_m, \quad (1.2)$$

where $n \ll N$ is a small number, and $(x_{m,i}, s_{m,i}, \alpha_{m,i}, i = 1, \dots, n)$ are the geometry attributes of the selected wavelet elements whose coefficients $(c_{m,i})$ are non-zero.

1.3.2 Active Basis Model

The active basis model is proposed by Wu et al. [Wu et al., 2010] for modeling deformable compositional patterns of the selected wavelet elements.

Suppose we have a set of training image patches $\{\mathbf{I}_m, m = 1, \dots, M\}$. This time they are defined on the same object bounding box, and the objects in these images come from the same category. They appear at the same location, scale and orientation, and in the same pose within the bounding box. The active basis model is of the following form

$$\mathbf{I}_m = \sum_{i=1}^n c_{m,i} B_{x_i + \delta x_{m,i}, s, \alpha_i + \delta \alpha_{m,i}} + U_m, \quad (1.3)$$

where $\mathbf{B} = (B_{x_i, s, \alpha_i}, i = 1, \dots, n)$ form the original template. Here the scale s is assumed to be fixed and given. $\mathbf{B}_m = (B_{x_i + \delta x_{m,i}, s, \alpha_i + \delta \alpha_{m,i}}, i = 1, \dots, n)$ is the deformed template for encoding \mathbf{I}_m , where $(\delta x_{m,i}, \delta \alpha_{m,i})$ are the perturbations from the nom-

inal location and orientation respectively, in order to account for deformation. Both $\delta x_{m,i}$ and $\delta \alpha_{m,i}$ are assumed to follow independent uniform distributions within limited ranges (default values: $\delta x_{m,i} \in [-3, 3]$ pixels, and $\delta \alpha_{m,i} \in [-\pi/16, \pi/16]$).

Sparsification procedures such as matching pursuit [Mallat and Zhang, 1993] usually select wavelet elements with little correlations. For computational and modeling simplicity, the active basis model enforces sparsity directly by assuming that the wavelet elements in each deformed template $\mathbf{B}_m = (B_{x_i+\delta x_{m,i},s,\alpha_i+\delta \alpha_{m,i}}, i = 1, \dots, n)$ are orthogonal to each other, so that the coefficient $c_{m,i} = \langle \mathbf{I}_m, B_{x_i+\delta x_{m,i},s,\alpha_i+\delta \alpha_{m,i}} \rangle$, and the coefficients can be denoted as $C_m = (c_{m,i}, i = 1, \dots, n)$. In practice, the active basis model allows small overlaps between the elements of \mathbf{B}_m .

For statistical modeling, the active basis model assumes that the distribution of \mathbf{I}_m given the deformed template $\mathbf{B}_m = (B_{x_i+\delta x_{m,i},s,\alpha_i+\delta \alpha_{m,i}}, i = 1, \dots, n)$, i.e., $p(\mathbf{I}_m | \mathbf{B}_m)$, is obtained by modifying the distribution of natural images $q(\mathbf{I}_m)$ in such a way that it only change the distribution of $C_m = (c_{m,i}, i = 1, \dots, n)$ from $q(C_m)$ to $p(C_m)$, while leaving the conditional distribution of U_m given C_m unchanged. Here $p(C_m)$ and $q(C_m)$ are the distributions of C_m under $p(\mathbf{I}_m | \mathbf{B}_m)$ and $q(\mathbf{I}_m)$ respectively. Specifically, $p(\mathbf{I}_m | \mathbf{B}_m) = q(\mathbf{I}_m)p(C_m)/q(C_m)$. Such a density substitution scheme was first used in projection pursuit density estimation [Friedman, 1987].

For computational simplicity, the model further assumes that $(c_{m,i}, i = 1, \dots, n)$ are independent given \mathbf{B}_m , under both p and q , so that

$$p(\mathbf{I}_m | \mathbf{B}_m) = q(\mathbf{I}_m) \prod_{i=1}^n p_i(c_{m,i})/q(c_{m,i}), \quad (1.4)$$

where $q(c)$ is assumed to be the same for $i = 1, \dots, n$ because $q(\mathbf{I}_m)$ is stationary. $q(c)$ can be pooled from natural images in the form of a heavy-tailed histogram of Gabor filter responses.

For parametric modeling, the following exponential family model is employed:

$$p_i(c) = \frac{1}{Z(\lambda_i)} \exp\{\lambda_i h(|c|^2)\} q(c), \quad (1.5)$$

where $h(r)$ is a monotone function of the response $r = |c|^2$ (sum of squares of Gabor cosine and sine responses) that saturates for large r . Specifically, $h(r) = \xi[2/(1 + e^{-2r/\xi}) - 1]$. $h(r)$ behaves like $h(r) \approx r$ for small r , but $h(r) \rightarrow \xi$ (default value: $\xi = 6$) as $r \rightarrow \infty$. $Z(\lambda)$ is the normalizing constant.

1.4 Organization

The rest chapters of this thesis are organized as follows: In chapter 2, we describe a mixed model in reminiscent of active basis mode, through which we introduce the 2D primitives and 3D primitives, which are building blocks of models in following chapters. This chapter also proposes to use information gain as the evaluation criterion for the effectiveness of model elements, which are used throughout this thesis. In Chapter 3, we extend the 2D primitives, and formulate models and algorithms for inferring middle level image concepts, such as curves and corners, which serve as an intermediate level of representation connecting the image pixels and the 3D hierarchical compositional templates introduced in Chapter 4. Other than the object representation, Chapter 4 also introduces an AND-OR structure that quantize the space of the object representation, which also give a new view of the approach used in computing the curves and corners. Chapter 5 summarizes this paper, and discusses about directions that we can further extend this thesis.

CHAPTER 2

A Flat Probabilistic Model Mixing 3D and 2D Primitives

This chapter presents a probabilistic model and a mixed template learning algorithm for view invariant object recognition. The template is composed of 3D and 2D primitives which are stick like elements defined in 3D and 2D spaces with appearance on images as Gabor filters. The primitives are allowed to translate within a local range to account for the object instance variation. Both 3D and 2D primitives have parameters describing their visible ranges, which models the effect of occlusion and view specific primitives respectively. We present an algorithm which sequentially selects primitives to build a mixed template. The selection order of primitives is decided by information gains, which can be estimated together with the visible range parameters efficiently. In experiments, we evaluate performance of the learned 3D templates on car recognition and pose estimation. We also show that the algorithm can learn intuitive mixed templates on various object categories, which suggests that information gain could be used as a numerical answer to the debate over viewer-centered representation and object-centered representation.

2.1 Introduction

This chapter presents a flat model mixing different types of primitives together with a learning algorithm for view invariant object recognition.

As is illustrated in Fig.2.1, we use a template containing both 3D and 2D primitives to represent the object images from different views. The 3D and 2D primitives are stick

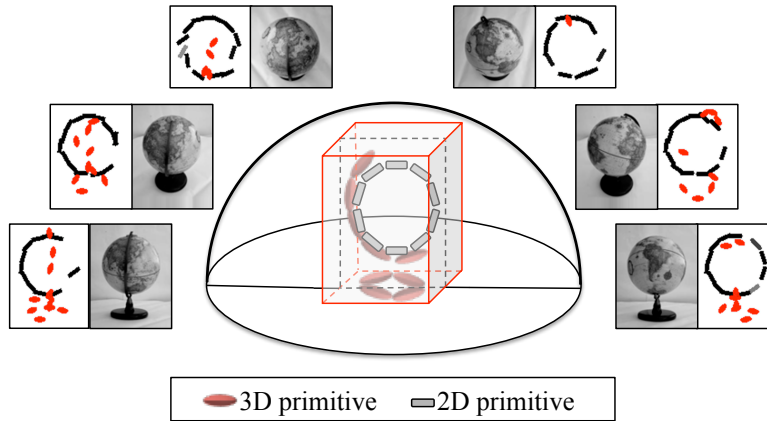


Figure 2.1: An illustration of the learned template for images of a desktop globe (Better viewed in color). The template has both 3D primitives and 2D primitives, which are denoted in red and gray respectively. Note that not all primitives are visible across all views. Our algorithm automatically selects 3D primitives to describe the appearance of base and handle of the globe, which vary across views, and 2D primitives to describe the occlusion boundary of the globe, which is a circular pattern across all views.

like elements defined in 3D and 2D spaces. To model object image variation, primitives are allowed to translate and rotate within a small location and orientation range. We use a visible range parameter for both 3D and 2D primitives to model the effect of 3D primitive occlusion and 2D primitives only visible in part of the view space. On images, the appearances of these primitives are Gabor filters. Due to projection, positions and orientations of Gabor filters for 3D primitives change across different views.

Starting from a reference image model, we build up our generative probabilistic model from view variant images of a same object category. For each primitive, we use different model families to describe its visible and invisible state separately. When visible, we model the filter response distribution of the primitive using the exponential model. When it is not, we assume the response follows the distribution of Gabor filtering responses on natural images.

We learn a sparse model where only a small set of primitives in a primitive pool

is selected using a pursuit algorithm. The algorithm sequentially selects independent primitives using information gain as pursuit index. The key issue in evaluating the information gain of a primitive is to estimate its visible range parameter. By carefully analyzing the information gain term in our problem, we are able to avoid enumerating all the possible visible ranges and apply a very efficient procedure to estimate the optimal visible range and information gain of a primitive simultaneously.

Selected primitives form intuitive templates as these shown in Fig.2.1, which can be used in tasks such as object detection and pose estimation. Moreover, experiment results on various object categories show the learned templates may transit from the fully 3D to fully 2D, depending on structural complexity of the modeled object category. These experiments suggest that there exists a representation spectrum for various object categories, which also indicates that the information gain could be used as a numerical answer to the debate over viewer-centered representation and object-centered representation.

This work is inspired by the active basis model [Wu et al., 2010] and recent development of learning object template mixing texture and structure [Si et al., 2009]. We further extend the key ideas in them into learning view variant object images for view invariant object modeling. We further extend the key ideas in them into learning view variant object images for view invariant object modeling.

2.2 Related Work

Proposals on how to represent visual knowledge that is invariant of views can be seen in literature from as early as 1974 [Minsky, 1974]. Through decades of developments, it gradually evolves to two general alternatives: viewer-centered representation such as aspect graphs [Koenderink and Doorn, 1979; Minsky, 1974] and object-centered representation whose early developments include those [Biederman and Gerhardstein,

1993; Marr, 1982] reviewed in Section 1.2.

To model view variant object images, an intuitive idea is to learn both the shape and appearance in the 3D space, so that it forms an object-centered representation. Learning 3D shape and appearance is successfully implemented in [Kushal and Ponce, 2006; Brown and Lowe, 2005] for multi-view images of a single object, with 3D SIFT [Lowe, 1999] points as primitives. But for images from an object category, previous work either learns the appearance model [Hoiem et al., 2007; Liebelt et al., 2008] directly by taking the 3D shape as granted, or learns shape and appearance one after another [Yan et al., 2007].

Researchers also acknowledge the simplicity of modeling 2D image patterns and the stability of these patterns on fixed views, so they propose to link 2D features [Thomas et al., 2006] or find shared features [Torralba et al., 2007] across views to efficiently build robust models. Researchers further propose to build up image patches composed of multiple feature points as an intermediate level of object category representation, then add links and transformations between these patches [Kushal et al., 2007; Savarese and Fei-Fei, 2007; Sun et al., 2009] or construct hierarchies based on them [Su et al., 2009] to build better models.

Over the two methods, some researchers have been engaged in a debate well documented by [Biederman and Gerhardstein, 1995; Tarr and Bülthoff, 1995; Hayward and Tarr, 1997], but do not get a final conclusion. From our point of view, the answer to which representation is better varies for different objects, and depends on the efficiency of the representation in explaining object images patterns. Moreover, elements (primitives) in these representations may be combined to achieve more efficient explanations. The desktop globe case shown in Fig.2.1 serves as such an example. For the purpose of recognition, a circular pattern template build by a few 2D primitives would be equivalent to a large amount of 3D primitives forming the 2D ball shape of the globe. Therefore for the globe part, directly using the 2D representation to model a 2D circle

would be more efficient than modeling a 3D shape. Similarly, a 3D template of the globe’s base and handle would be much simpler and more efficient than 2D templates, because these patterns change significantly across views.

Based on these observations, we propose to mix the two representations, and use information gain as a numerical criterion to select primitives from both to get better representations. With such a criterion, our model can automatically transit between the two types of representations.

The main contributions of this chapter include:

1. We propose 3D object primitives and 2D image primitives that can be used to model images of an object category from different views.
2. We propose an efficient algorithm that can compute primitive information gain and estimate its parameters simultaneously.
3. By using information gain as a criterion to select both 3D and 2D primitives regardless of their types, the criterion and the proposed model learning algorithm provide a numerical answer to the debate over object-centered and viewer-centered representation.

2.3 Dictionaries of 3D and 2D Primitives

We propose dictionaries of 3D and 2D primitives, which can be used separately or combined to represent object images of different views.

2.3.1 3D Object Primitives

To represent the 3D structure of an object category, we propose a type of stick like 3D elements as 3D object primitives, which are shown in Fig.2.2. For a 3D primitive,

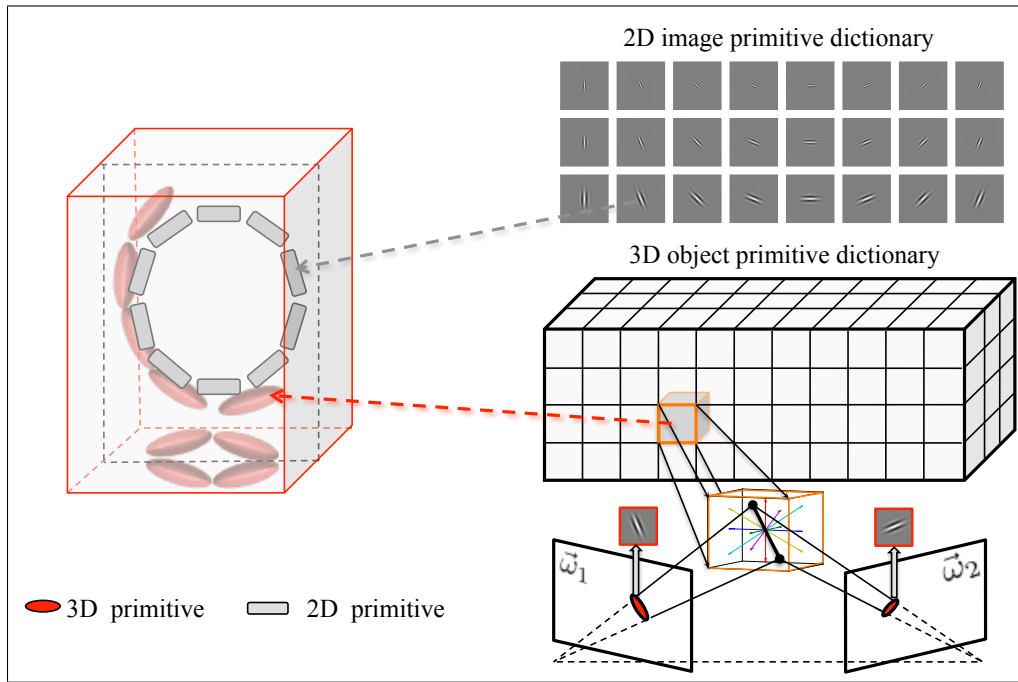


Figure 2.2: An illustration of 3D and 2D primitives and how they are used to compose a mixed representation. The 3D primitives are shown on the lower-right part. A 3D primitive can be viewed as a stick with selected orientation and rotation in 3D space, and its appearance is a view dependent Gabor filter placed at its projected position and orientation. A 2D primitive is a Gabor filter located at a selected 2D position and orientation.

we define its geometry parameter in the 3D space, that is its position (X, Y, Z) and orientation Θ . The length of 3D primitives is defined as unit length.

For any given view ω , the appearance of such a 3D primitive is a Gabor filter, whose position and orientation is defined by the projection of the primitive onto that view. Hence the appearance of a 3D primitive will be different for different views. In this paper, we use orthographic projection as our projection model, for which the projection matrix can be expressed as:

$$P = \begin{bmatrix} s & 0 & u \\ 0 & s & v \\ 0 & 0 & 0 \end{bmatrix} \left[\begin{array}{c|c} R_{3 \times 3} & \begin{matrix} -t_u \\ -t_v \\ 1 \end{matrix} \end{array} \right] \quad (2.1)$$

where s is the scale of object image, (u, v) denote the center of the camera optical axis on the image, R is a 3×3 rotation matrix parameterized by pan α , tilt β and roll γ angle of the camera, and (t_u, t_v) denote the object center offset relative to the center of the image.

Given the projection model, a view ω can be defined as a vector $(s, \alpha, \beta, \gamma, t_u, t_v)$, and thus the view space of this corresponding projection model can be defined as the product space of the ranges of these parameters.

When projecting 3D primitives onto object images, some of them may be occluded. So we use a parameter Ω to denote the set of views in which the corresponding primitive is visible. For example,

$$\Omega = \{\omega_m, m = 1, 2, \dots, M\}, \quad (2.2)$$

where M is the number of views.

For each projected Gabor filter, we allow it to locally rotate and translate in a small range, so that it can be adapted to small appearance changes of different object instances.

The 3D primitives are instantiated inside an assumed object bounding volume, by enumerating their parameters. To avoid excessively enumerating them, we segment the target bounding volume into a set of non-overlapping cubes, and only instantiate primitives on the centers of each cube. For each point in the sampled centers \mathcal{C} , we place primitives at evenly spaced orientations \mathcal{O} , where these orientations are derived by a sphere equal partitioning method [Leopardi, 2006]. Thus the 3D primitive dictionary Δ^{3D} can be written as:

$$\Delta^{3D} = \{B_{X,Y,Z,\Theta}^{3D} \mid (X, Y, Z) \in \mathcal{C}, \Theta \in \mathcal{O}\} \quad (2.3)$$

The primitive response on an image is defined as the Gabor filter response in the same way as in the active basis model [Wu et al., 2010], and is thus not elaborated.

2.3.2 2D Image Primitives

The 2D primitives used in this thesis are inspired by the active basis model [Wu et al., 2010]. The key difference between our 2D primitives and the active basis is that we further add the visible range parameter Ω . This Ω has the same form as the one of 3D primitives, and is used to model the effect that 2D primitives may only appear in part of the view space.

As object images are from different views and thus cannot be aligned by assuming a shared bounding box, we assume there is a virtual shared image lattice Λ and construct the dictionary by enumerating primitives at each position and each of a set of discrete orientations in this lattice. So,

$$\Delta^{2D} = \{B_{u,v,\theta}^{2D} \mid (u, v) \in \Lambda, \theta \in \mathcal{O}\}, \quad (2.4)$$

where the \mathcal{O} is a set of K equally spaced orientations $\mathcal{O} = \{0, \pi/K, \dots, \pi\}$.

For each image, we scale and translate primitive positions according to the scale and offset in its view label ω , so that images from similar views can still be aligned.

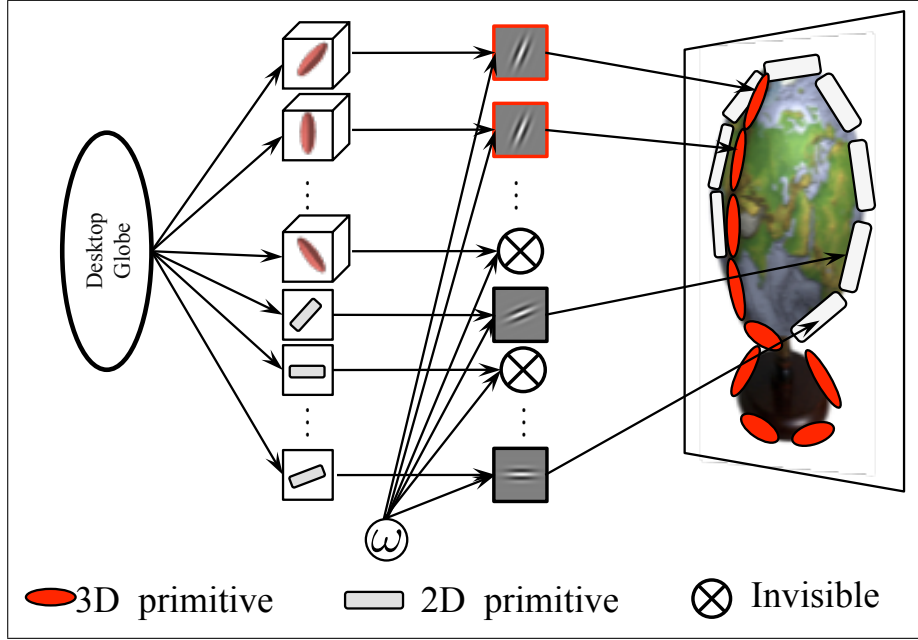


Figure 2.3: An overview of the image generating model. When generating object images at a particular view ω , we project the learned 3D primitives and transform the 2D primitives to generate Gabor filters at corresponding image positions and orientations. If a primitive is not visible in a particular view, the corresponding Gabor filter will not be generated.

2.4 Probabilistic Image Model

As is shown in Fig.2.3, given N primitives $\mathbf{B} = \{B_i, i = 1, \dots, N\}$ as a template of object images at different views, we assume a specific image \mathbf{I} at view ω can be generated as a linear superposition of these primitives:

$$\mathbf{I} = \sum_{i=1}^N c_i B_i + U, \quad (2.5)$$

where c_i is the coefficient associated with i -th primitive B_i , and U is the unexplained part of the image. Note that coefficients c_i for invisible primitives are zero.

Consider the image \mathbf{I} in Eqn. (2.5) as a random variable and following the derivation

of active basis model [Wu et al., 2010], the target distribution can be expressed as:

$$p(\mathbf{I}, \omega \mid \mathbf{B}) = q(\mathbf{I}) \prod_{i=1}^N \frac{p[h(r_i) \mid \omega]}{q[h(r_i)]} \quad (2.6)$$

where $p[h(r_i) \mid \omega]$ refers to the distribution of the transformed response of B_i on target images, $h(\cdot)$ is a function that performs a sigmoid transform, and $q(\mathbf{I})$ is the reference image distribution. In the above derivation, we assume the primitive responses $\{r_i, i = 1, \dots, N\}$ are independent of each other, both $p(\omega)$ and $q(\omega)$ follows a uniform distribution. As $q[h(r_i)]$ is the distribution of primitive response on reference images, it is assumed to be independent of ω and Gabor filter position.

Similar to active basis model, we use the natural image distribution as the reference image model q , thus the $q[h(r_i)]$ should be the same for all primitives.

Whether a primitive is visible or not significantly affects the distribution of its observed primitive response. Hence, when modeling $p[h(r_i) \mid \omega]$, we model the case of primitive is visible and invisible separately.

When the primitive is visible, according to maximum entropy principle [Jaynes, 1957; Pietra et al., 1997], the response can be modeled as

$$p[h(r_i); \lambda_i] = \frac{1}{Z} \exp[\lambda_i h(r_i)] q[h(r_i)], \quad (2.7)$$

where Z is the partition function. When it is not, the observed response is assumed to follow reference distribution, thus

$$p[h(r_i)] = q[h(r_i)]. \quad (2.8)$$

Taking the visible range parameter Ω_i into account, the conditional probability of $p[h(r_i) \mid \omega]$ can be expressed as:

$$p[h(r_i); \lambda_i, \Omega_i \mid \omega] = \begin{cases} \frac{1}{Z_i} \exp[\lambda_i h(r_i)] q(r_i) & \omega \in \Omega_i \\ q[h(r_i)] & \omega \notin \Omega_i \end{cases} \quad (2.9)$$

To ensure that Eqn. (2.9) fits the target distribution, λ_i should be estimated subject to the constraint that given M' visible responses $\{r_{im}\}_{m=1}^{M'}$,

$$\frac{1}{M'} \sum_{m=1}^{M'} h(r_{im}) = \int h(r_i) \frac{1}{Z_i} \exp[\lambda_i j(r_i)] q[h(r_i)] dh(r_i) \quad (2.10)$$

2.5 Learning Algorithm

Given M view labeled training images $\{(\mathbf{I}_m, \omega_m), m = 1, \dots, M\}$ from the target distribution $p(\mathbf{I}, \omega \mid \mathbf{B})$ and a set of N primitives $\mathbf{B} = \{B_i, i = 1, \dots, N\}$, the likelihood model in Eqn. (2.6) can be optimized by maximizing the information gain of each primitive. This is because:

1. maximizing the log-likelihood of $p(\mathbf{I}, \omega \mid \mathbf{B})$ is equivalent to maximizing the information gain of \mathbf{B} from reference model q to target model p :

$$\text{IG}(\mathbf{B}) = \iint p(\mathbf{I}, \omega \mid \mathbf{B}) \log \frac{p(\mathbf{I}, \omega \mid \mathbf{B})}{q(\mathbf{I}, \omega)} d\mathbf{I} d\omega \quad (2.11)$$

$$\approx \sum_{m=1}^M \log \frac{p(\mathbf{I}_m, \omega_m \mid \mathbf{B})}{q(\mathbf{I}_m, \omega_m)} \quad (2.12)$$

$$= \sum_{m=1}^M \log p(\mathbf{I}_m, \omega_m \mid \mathbf{B}) - \sum_{m=1}^M \log q(\mathbf{I}_m, \omega_m) \quad (2.13)$$

where $\sum_{m=1}^M \log q(\mathbf{I}_m, \omega_m)$ is a constant and $\sum_{m=1}^M \log p(\mathbf{I}_m, \omega_m \mid \mathbf{B})$ is the log-likelihood.

2. The information gain of the template \mathbf{B} is equal to the sum of that of individual

primitives B_i as:

$$\text{IG}(\mathbf{B}) = \iint p(\mathbf{I}, \omega | \mathbf{B}) \log \frac{p(\mathbf{I}, \omega | \mathbf{B})}{q(\mathbf{I}, \omega)} d\mathbf{I}d\omega \quad (2.14)$$

$$= \iint p(\mathbf{I}, \omega | \mathbf{B}) \log \frac{p(\mathbf{I} | \omega, \mathbf{B})}{q(\mathbf{I} | \omega)} d\mathbf{I}d\omega \quad (2.15)$$

$$= \iint p(\mathbf{I}, \omega | \mathbf{B}) \sum_{i=1}^N \log \frac{p[h(r_i) | \omega]}{q[h(r_i) | \omega]} d\mathbf{I}d\omega \quad (2.16)$$

$$= \sum_{i=1}^N \iint p(\mathbf{I}, \omega | B_i) \log \frac{p[h(r_i), \omega]}{q[h(r_i), \omega]} d\mathbf{I}d\omega \quad (2.17)$$

$$= \sum_{i=1}^N \iint p(\mathbf{I}, \omega | B_i) \log \frac{p(\mathbf{I}, \omega | B_i)}{q(\mathbf{I}, \omega | B_i)} d\mathbf{I}d\omega \quad (2.18)$$

$$= \sum_{i=1}^N \text{IG}(B_i) \quad (2.19)$$

$$(2.20)$$

Thus the problem of maximizing the overall information gain can be decomposed into the problem of maximizing that of each primitive. Note that r_i is actually a function of image \mathbf{I} as $r_i = \|\langle \mathbf{I}, B_i \rangle\|^2$.

In the following, we will first show how to estimate the maximum information gain together with the parameters λ and Ω effectively. Then we show how to solve the problem of finding the best set of primitives \mathbf{B} using a shared pursuit algorithm.

2.5.1 Evaluate the Information Gain of a Primitive

Using Eqn.(2.16) to (2.19), the information gain of a primitive B_i by updating its distribution from p to q is:

$$\text{IG}(B_i) = \iint p(\mathbf{I}, \omega | B_i) \log \frac{q[h(r_i), \omega]}{p[h(r_i), \omega]} d\mathbf{I} d\omega \quad (2.21)$$

$$\approx \sum_{m=1}^M \log \frac{p[h(r_{im}) | \omega_m]}{q(r_{im})} \quad (2.22)$$

$$= \sum_{\omega_m \in \Omega_i} [\lambda_i h(r_{im}) - \log Z_i] + \sum_{\omega_m \notin \Omega_i} 0 \quad (2.23)$$

$$= \sum_{\omega_m \in \Omega_i} [\lambda_i h(r_{im}) - \log Z_i] \quad (2.24)$$

If Ω_i is given, the parameter λ_i should be fitted such that

$$\hat{\mu}_r \approx \frac{1}{M'} \sum_{m'=1}^{M'} h(r_{im'}) = E_p [h(r_i); \lambda_i], \quad (2.25)$$

where $\hat{\mu}_r$ is the average response of B_i on samples in visible range Ω_i , and M' is the number of these samples. To reduce computation, one can compute a histogram of $E_p [h(r); \lambda]$ indexed by λ before learning starts, so the best λ can be estimated by searching the histogram and interpolating between the nearest two bins.

In the above derivation, we assume Ω_i is given, but our objective is to maximize the information gain on both λ_i and Ω_i . Because Ω is defined as a set of views, given M training images, a naive way to estimate these parameters is to enumerate all the 2^M effectively different Ω_i , find the corresponding maximum information gain for each one by fitting λ , and then find the global optimal. This method scales badly as the complexity grows exponentially with M .

By analyzing the structure of the problem, we can find an algorithm linear in M that also guarantees the global optimal for this problem. After rearranging the elements in Eqn.(2.24), maximizing Eqn.(2.24) is equivalent to:

$$\max |\Omega_i| (\lambda_i \cdot \hat{\mu}_r - \log Z_i) \quad (2.26)$$

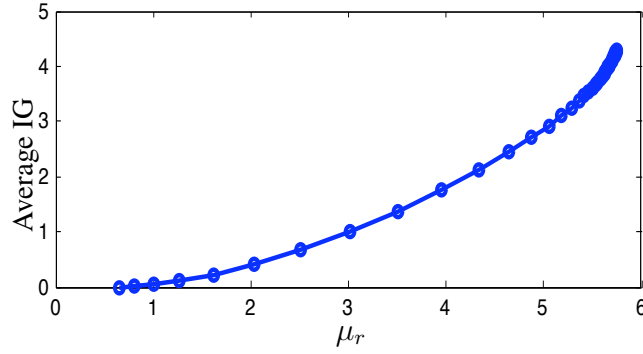


Figure 2.4: The relationship of average information gain against μ_r . This curve shows that the average information gain is an increasing function of μ_r . The reason that why μ_r is in range $[0, 6]$ is explained in Section 2.7.1.

where $|\Omega_i|$ is the number of training views in Ω_i . Eqn.(2.26) indicates that according to the number of views in Ω_i , we can put the 2^M possible visible ranges into M groups. Benefits of grouping them are led by the following observations:

1. For visible ranges in each group, $|\Omega_i|$ is fixed, so maximizing Eqn (2.26) is reduced to maximizing the average information gain $(\lambda_i \cdot \hat{\mu}_r - \log Z_i)$.
2. For log linear model, it has been proved that the average information gain is an increasing function of $\hat{\mu}_r$.

The second observation is proved in [Wu et al., 2010], and here we just show the corresponding curve in Fig.2.4 as a demonstration.

According to these two observations, in each of the groups of visible ranges, the optimal Ω_i must be the one that leads to largest $\hat{\mu}_r$, which should be the mean of the first $|\Omega_i|$ largest of totally M responses.

Therefore, we can avoid enumerating 2^M combinations and instead sort the primitive responses, construct the optimal Ω_i in each group by assembling views from the images with first largest $|\Omega_i|$ responses, and compare corresponding fitted information gain to find the global optimal. Int this way, we can reduce the 2^M computations to M .

Further investigation suggests that the information gain must be a single peak function of $|\Omega_i|$, because as $|\Omega_i|$ goes larger, the average information gain will be smaller as $\hat{\mu}_r$ will be smaller. Thus, we can further reduce the computations by computing the groups in descending or ascending order on the number of views, and stop once we find that the information gain stops growing.

The detailed procedure of this evaluation algorithm is shown in Algorithm 2.1. Note that the step 6 is done in the similar way to computing λ .

Algorithm 2.1: Evaluate information gain of a primitive

Input: $\{h(r_{im}), \omega_m\}_{m=1}^M$

Output: Estimated parameter $\hat{\lambda}_i, \log Z_i, \hat{\Omega}_i$, Information gain IG_i

- 1 . $IG^{\max} \leftarrow 0, t^{\max} \leftarrow 0;$
 - 2 $\{h(r'_{im}), \omega'_m\}_{m=1}^M \leftarrow$ descending sort on $h(r_{im});$
 - 3 **for** $t \leftarrow 1$ to M **do**
 - 4 $\mu^t \leftarrow \text{mean}[h(r'_{i1}), \dots, h(r'_{it})];$
 - 5 $\lambda^t \leftarrow$ interpolate table $E_p[h(r); \lambda], s.t. \mu^t \approx E_p[h(r); \lambda^t];$
 - 6 $\log Z^t \leftarrow$ search table $\log Z(\lambda)$ to get $\log Z(\lambda^t);$
 - 7 $IG^t \leftarrow t \cdot (\lambda^t \mu^t - \log Z^t);$
 - 8 **if** $IG^{\max} < IG^t$ **then**
 - 9 $\hat{\lambda}_i \leftarrow \lambda^t, IG^{\max} \leftarrow IG^t, t^{\max} \leftarrow t;$
 - 10 **else**
 - 11 Break;
 - 12 $\log Z_i \leftarrow \log Z(\hat{\lambda}_i), \hat{\Omega}_i \leftarrow \{\omega'_k\}_{k=1}^{t^{\max}}, IG_i \leftarrow IG^{\max};$
 - 13 **return** $\hat{\lambda}_i, \log Z_i, \hat{\Omega}_i, IG_i$
-

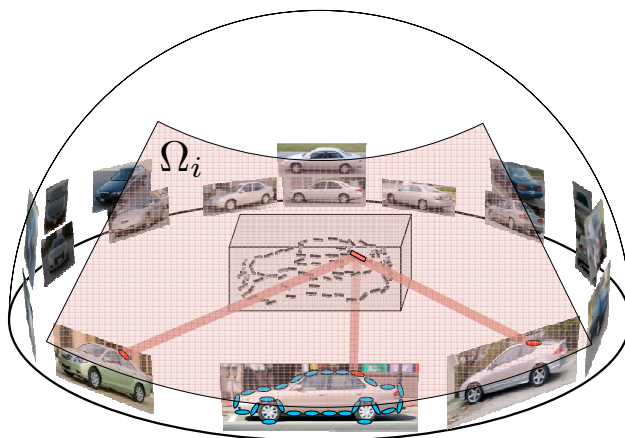


Figure 2.5: An illustration of how 3D primitives are learned from and projected to images in different views. The learning step can be interpreted as trying all possible locations and orientations of 3D primitives and keep the independent ones with large projected responses. In testing stage, the learned 3D model is projected to each possible view and test if the projected template matches testing image.

2.5.2 Shared Primitive Pursuit

Section 2.5.1 solves the problem of finding the maximum information gain of each primitive. In this section, we present a shared pursuit algorithm that learns a set of primitives \mathbf{B} which maximize the overall likelihood of our target model in Eqn. (2.6).

The learning algorithm first computes the responses of all Gabor filters on all training images. Specifically, we compute responses of these filters centered at each pixel of each input image, and at $K = 17$ equally spaced orientations. Following [Wu et al., 2010], we call these filter responses together as SUM1 maps. To account for primitive deformations, we compute for each pixel and orientation the maximum Gabor filter response in its neighboring locations and orientations, and save these maximized responses in MAX1 maps.

We then compute the responses of each primitive on training images, by referencing the projected Gabor responses on MAX1 maps. After that, we can use Algorithm 2.1

to compute the information gain for each primitive and then select the primitive with maximum information gain. At the same time, parameters associated with this primitive are also stored.

To enforce the independence assumption, we further inhibit primitives with significant correlations with the selected one. For each image, we trace the referenced Gabor filter according to the type of the primitive and the image view ω . We then set the responses of correlated Gabor filters to be 0 and update the corresponding MAX1 maps on potentially affected areas by re-calculating the local maximum response. Correlations of different Gabor filters are functions of their relative locations and orientations, thus can be computed before learning. A Gabor filter is considered correlated with the referenced one if the absolute value of their correlation is greater than a threshold.

After the first primitive is selected, we repeat the process starting from updating the information gain of each primitive so as to sequentially pursue a collection of primitives from the enumerated primitive pool. The procedure of this algorithm is summarized in Algorithm 2.2.

Note that during primitive pursuit, information gain of a primitive is non-increasing, hence we only need to compute IG for all primitives in the first iteration, and then delete primitives lower than a threshold, since they will never be selected.

Also note that in both Section 2.4 and 2.5, there is no assumption on the type of primitive. Thus depending on the primitive dictionary, the above model and learning algorithm can be used to learn templates composed of either purely 3D or 2D primitives, or composed of both.

2.6 Inference Algorithm

In our formulation, Ω is defined as a collection of single views. To ensure that our model can also be used on views not appeared in training set, we further convert this Ω

Algorithm 2.2: Primitive pursuit

Input: M training images $\{I_m, \omega_m\}$

Output: N selected primitives with parameter

$$\{(X_i, Y_i, Z_i, \Theta_i), \lambda_i, \log Z_i, \Omega_i\}_{i=1}^N$$

- 1 Compute SUM1 maps and MAX1 maps.;
 - 2 Enumerate all the possible primitives.;
 - 3 Project each primitive on to the MAX1 maps.;
 - 4 **for** $i \leftarrow 1$ to N **do**
 - 5 Retrieve primitive responses from MAX1 maps.;
 - 6 **for** *each primitive* **do**
 - 7 compute IG_i , using algorithm 2.1.
 - 8 Find the maximum IG , denote it as IG_{\max} ;
 - 9 Retrieve and save parameters associated with IG_{\max} to $\lambda_i, \log Z_i$ and Ω_i ;
 - 10 $(X_i, Y_i, Z_i, \Theta_i) \leftarrow (X_{\max}, Y_{\max}, Z_{\max}, \Theta_{\max})$;
 - 11 Local inhibition.;
-

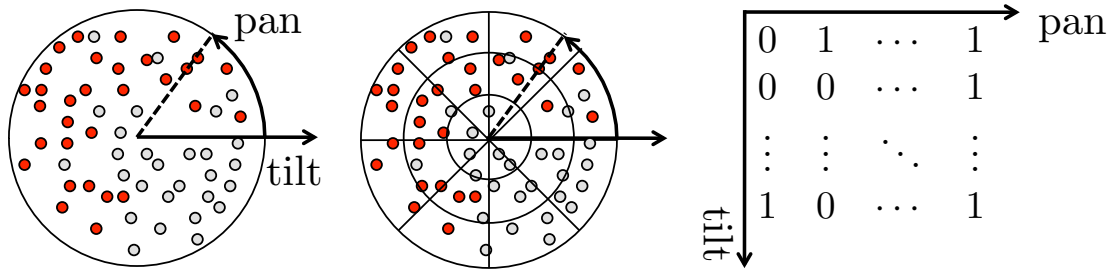


Figure 2.6: Convert Ω from a set to a 2 dimensional lookup table (Better viewed in color). **Left:** red points represent the views that are in Ω , gray ones are the rest. **Middle:** we segment the view sphere into several view bins. **Right:** Each bin corresponds to an entry of a 2D lookup table, where its value is set by the majority of the training views in that bin.

into a binary lookup table, where each entry denotes the visibility state of a primitive in a specific and fixed view range. The conversion steps are shown in Fig.2.6. Considering the fact that in orthogonal projection, image center offset, scale, and roll angle do not affect occlusion states of primitives, we only need to set up a 2 dimensional lookup table along the pan and tilt angle. The entry value is set to 1 if a majority ($> 50\%$) of training images in that bin is included in Ω .

The inference problem can be formulated as a hypothesis testing problem, where the testing score is defined as

$$\begin{aligned}
 \text{score} &= \log \frac{p(\mathbf{I} \mid \omega, \mathbf{B})}{q(\mathbf{I})} \\
 &= \sum_{i=1}^N (\lambda_i r_i - \log Z_i) \cdot \mathbf{1}(\omega \in \Omega_i).
 \end{aligned} \tag{2.27}$$

As Ω_i has already been converted to a lookup table, the function $\mathbf{1}(\omega \in \Omega_i)$ is to simply index the lookup table.

To perform object recognition, the above score is computed at a large amount of hypothesized views. The highly scored hypothesized views are then reported through a non-maximum suppression procedure.

2.7 Experiments

2.7.1 Dataset, Image Pre-processing and Parameters

To show that our algorithm can learn intuitive templates for various categories, we use ETH80 dataset [Leibe and Schiele, 2003], which contains 8 categories with 10 object instances for each category. We further use soda cans as an object category, and the desktop globe images used in Fig.2.1. The last two sets of images are taken by ourselves by capturing objects on a turning table. View labels for images in ETH80 dataset are already provided, and the images collected by ourselves are shoot in the same angles as in ETH80 dataset.

To show the effect of our algorithm on uncontrolled images with relatively complex background, a multi-view car dataset [Savarese and Fei-Fei, 2007] is used. Since view labels are not provided in this dataset, we label them by imposing a 3D CAD car model onto images, and save the corresponding virtual camera parameters as views when the model matches the images. For this purpose, we wrote a Plug-In to extend the freely available CAD software Google SketchUp, where a snapshot of the image annotation interface is shown in Fig.2.7

Preprocessing steps used on the Gabor filter responses in SUM1 maps are:

1. We normalize the responses by dividing them with the variance of all Gabor filter responses on that image.
2. For each image, we perform local normalization, which divides the globally normalized response with an average of that in a local window for each filter. We use a local window spanning all the K orientations and 32×32 pixels centered at the target Gabor filter.
3. We further regulate responses using sigmoid function $h(r)$, which is:

$$r' = \text{sigmoid}(r) = \xi[2/(1 + e^{-2r/\xi})], \quad (2.28)$$

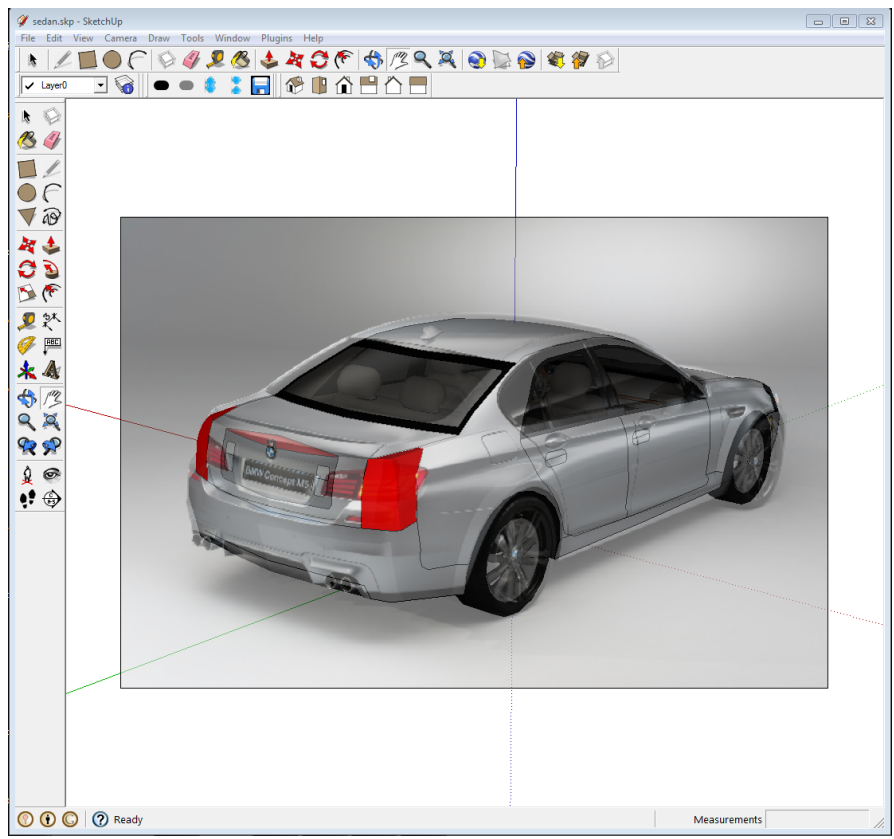


Figure 2.7: A snapshot of the user interface for image view annotation.

and ξ is set to 6.

Theoretical underpinnings of these preprocessing steps can be found in [Wu et al., 2010].

In experiments on ETH80 dataset, the location and orientation change for computing MAX1 maps is ± 4 pixels and ± 1 orientations. In experiments on dataset [Savarese and Fei-Fei, 2007], these are ± 3 pixels and ± 1 orientations. Using a desktop computer with Intel Core 2 Duo 2.4G processor, the learning stage usually costs about 30 minutes with a training set of 420 images and about 3 million proposed primitives.

2.7.2 Learning Mixed Templates

To demonstrate the effect of our mixed template learning algorithm, we show the learned templates for 10 categories in Fig.2.8. The categories are sorted by the proportion of 2D primitives in the template, and are shown in descending order.

From the figure, we can see that our method automatically finds suitable representations for different object categories, which spans a spectrum from nearly pure 2D to pure 3D. For object categories with stable 2D shapes, the model automatically selects 2D primitives to form the rough shape of that category. For specific details, like the top of the tomatoes, and handle of cups and desktop globe, the algorithm will select 3D primitives, because these details are view specific and only appear on part of the view sphere.

For categories with complex shapes, coding the general shape for each viewing angle will be less efficient than coding the general 3D shape using 3D primitives. So, the algorithm automatically transits to select 3D primitives, which forms an object-centered representation. With these experiment results, one can see that our model can learn templates of different mixing proportions, according to the shape and appearance of different object categories.

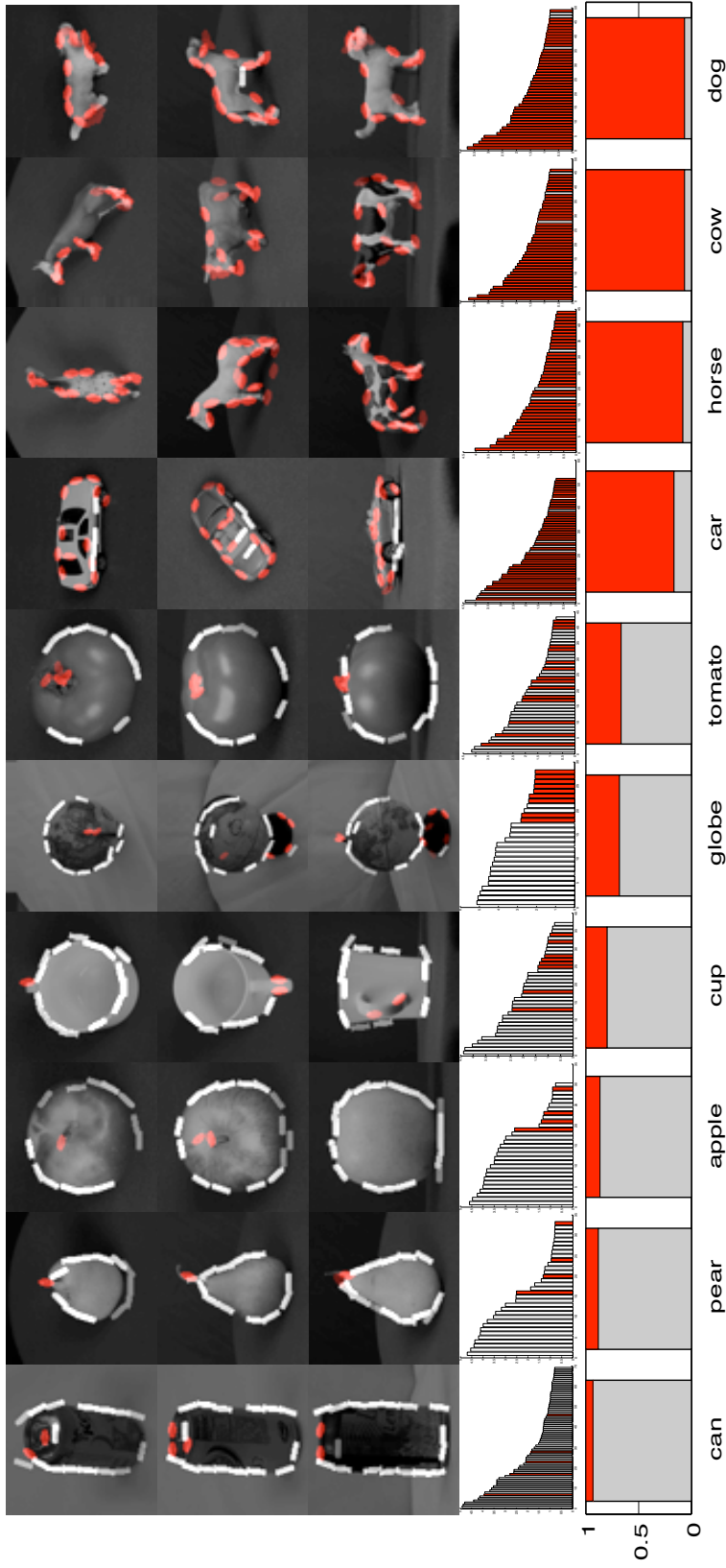


Figure 2.8: Learned object templates for 10 object categories and their pursuit indexes (Better viewed in color). **Row.1-Row.3:** Learned templates for different object images. The gray bars represent 2D primitives and red ellipses are for 3D primitives, the intensity of those symbols show the corresponding score of each primitive. **Row.4:** The primitive pursuit index of each object category. The horizontal axis are the pursuit order of each primitive, and vertical axis is the information gain of each primitive. Again, red bars are for 3D primitives and white bars are for 2D primitives. **Row.5:** The ratio on the number of 3D primitives (red) and 2D primitives (gray) for each category.

Row 4 and 5 of Fig.2.8 shows the selection order of the 3D (red) and 2D (gray) primitives, their information gains, and their proportion of information contribution in each template. By sorting these categories according to this proportion, Fig.2.8 clearly shows that the learned representations reside in different positions on the spectrum, and their positions are related to the complexity of the object shape.

2.7.3 Learning 3D Templates for Car Recognition and Viewpoint Inference

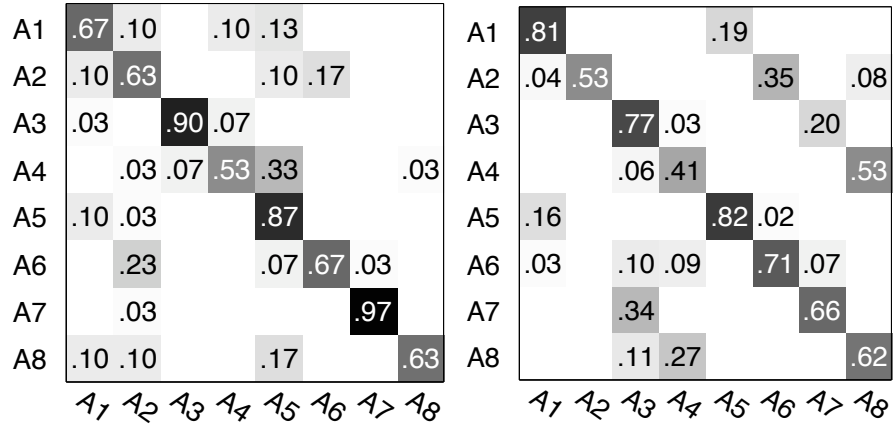
We further use our learning algorithm to learn 3D car templates on the multi-view car dataset [Savarese and Fei-Fei, 2007], as Fig.2.8 suggests that car templates is almost composed of purely 3D primitives. We use the first 5 object instance in the dataset as training images, and the rest 5 as testing.

We first perform the experiment of object pose inference. In this chapter, this task is defined as finding the most probable view category given that the object image is in a bounding box. For this part, we took the images in the given bounding box as input, and search around views ω to find the one with maximum score. The estimated views are then converted to view categories as the inputs to compute the confusion matrix. Specifically, we compute the test scores by enumerating all views at 2° for pan, tilt, and roll angles, and assume the object offsets are in the center of the bounding box.

The result is shown in Fig.2.9 with both confusion matrix and some of template matching results. ¹

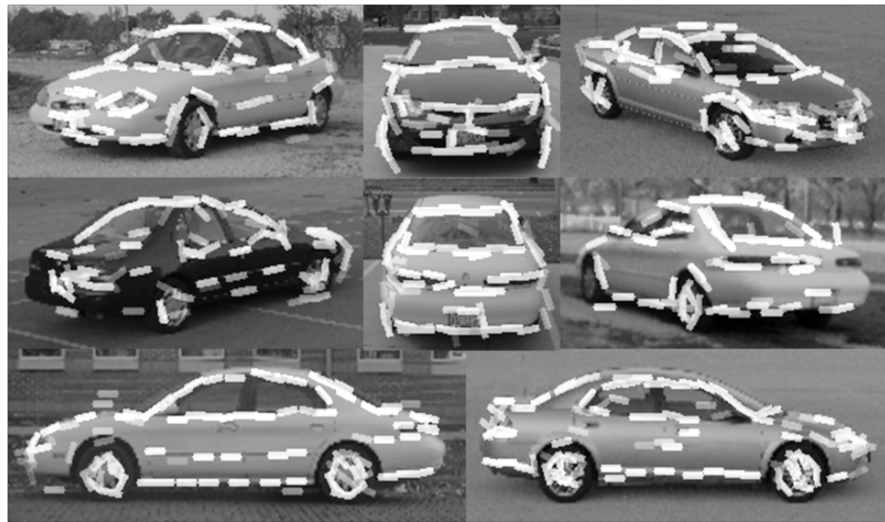
Since the output of our inference algorithm is directly view angles instead of view category. Our result should be more useful in pose related tasks such as license plate searching. Also, by projecting the templates, this algorithm roughly traces the boundary of object images, which can be used as a good initial state for tasks like image

¹It should be noted that this experiment is not completely the same as in [Su et al., 2009], since 1.) We test not only on correctly recognized images, but all testing images. 2.) We are also testing images at scale 3 of the dataset.



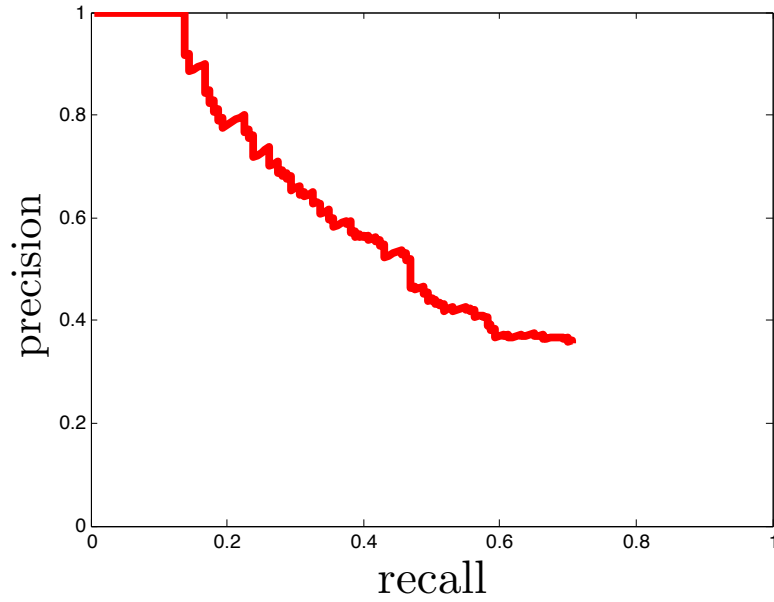
(a)

(b)



(c)

Figure 2.9: Result of pose estimation task. (a) Confusion matrix of our method. (b) Confusion matrix from [Su et al., 2009], which we use as a reference. (c) Projected templates on testing images of different views.



(a)



(b)

Figure 2.10: (a) Performance of detection task in terms of precision-recall curves. (b) Some sample detection results, each from one of the eight poses in the dataset.

segmentation.

We also show object detection results on this dataset. The detection task is done by computing the testing scores at enumerated views at step width 10° in pan, 5° in tilt and 2° in roll, with offsets at all positions in the image. Highest testing scores are then reported together with their views after non-maximum suppression. Following the protocol of VOC detection task, we evaluate the performance of our algorithm in the term of precision-recall (PR) curve. The PR curve together with some detection results are shown in Fig2.10.

2.8 Discussion

This chapter presents a learning algorithm modeling view variant object images for object recognition in arbitrary views. The model can automatically transit between and mix object-centered representation and viewer-centered representation, which together with the information gain criterion provides a numerical answer to the debate over the two types of representation. It also provides a foundation for developing compositional 3D and 3D models

CHAPTER 3

Composing Image Primitives to Active Curves

In this chapter, we study a problem facilitates the study of view invariant recognition, which is to find sparse representations of 2D images. We propose to use deformable templates of simple geometric structures that are commonly observed in images of objects as well as natural scenes. These deformable templates include active curve templates and active corner templates. An active curve template is a composition of active Gabor elements placed with equal spacing on an arc segment, where each Gabor filter is allowed to locally shift its location and orientation, so that the active curve template can be deformed to fit the observed image. An active corner template is a composition of two active curve templates that share a common end point, and the active curve templates are allowed to vary their overall lengths and curvatures. This chapter then presents a hierarchical computational architecture of sum-max maps that pursues a sparse representation of an image by selecting a small number of active curve and corner templates from a dictionary of all such templates. Experiments show that the proposed method is capable of finding sparse representations of natural images. It is also shown that object templates can be learned by selecting and composing active curve and corner templates.

3.1 Introduction

Finding sparse representations of images is one of the most fundamental problems in both computer and biological vision, as a sparse representation gives a meaningful

interpretation of the image. By varying the attributes of the sparse coding elements, we can generalize from one image to another, or construct and learn statistical models that capture the regularities and variabilities of images from various images in an object or scene category.

This chapter proposes a sparse representation of images using active curve templates and active corner templates. In an active curve template, the prototype template is a composition or grouping of a small number of Gabor filter elements placed with equal spacing on a straight line segment or a circular arc segment of constant curvature. So the prototype template is parameterized and very simple. However, these filter elements are allowed to locally shift their locations and orientations so that the prototype curve template can be deformed to different instances in order to fit observed images. Thus an active curve template is also flexible enough to account for deviations from the simple prototype.

Two active curve templates that share a common endpoint can be further composed into an active corner template, where the two constituent active curve templates are allowed to vary their overall geometric attributes such as lengths and curvatures, so that a prototype corner template corresponds to many variants. See Fig.3.1 for an illustration.

This chapter then proposes a layered computational architecture of sum-max maps that pursues a sparse representation of an image by selecting a small number of active corner templates and active curve templates from a dictionary of all such templates. The computation has a bottom-up pass and a top-down pass. The bottom-up pass tests all the possible templates exhaustively but efficiently by parallel recursive computations that alternate between local sum pooling and local max pooling. The sum pooling scores the active curve and corner templates by combining the scores of their constituent components, while the max pooling optimizes the variations of the constituent components to infer deformations. After the bottom-up pass, the top-down pass selects the active corner templates and active curve templates, and deforms them into their optimal vari-

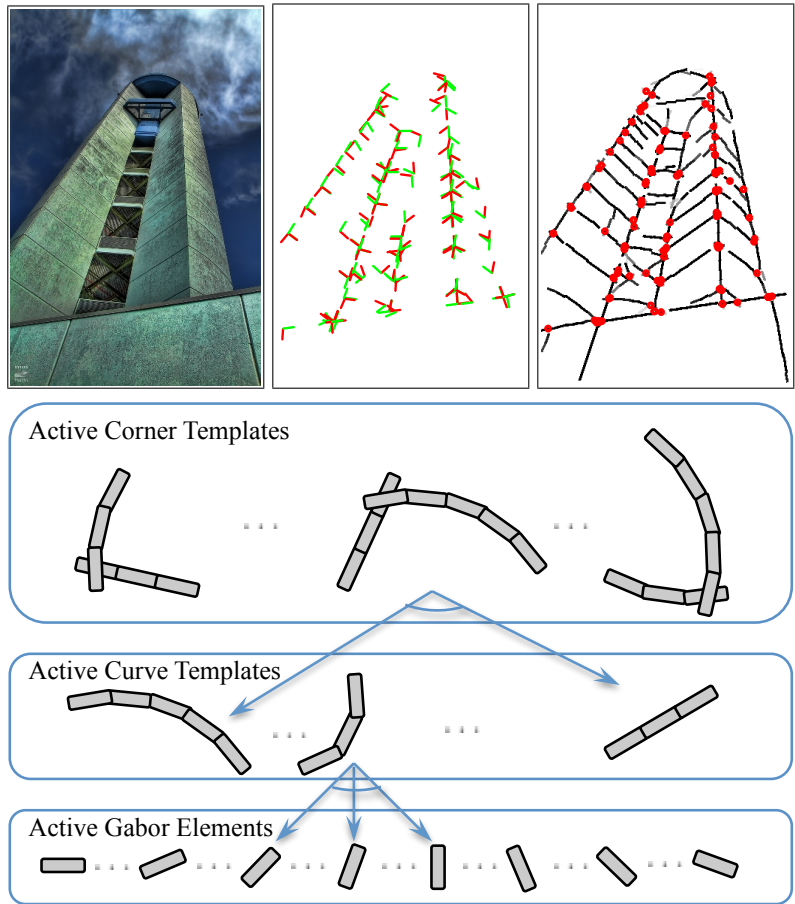


Figure 3.1: (Better viewed in color) An image is represented by a small number of active corner templates and active curve templates. **Top Row, Left:** Original image. **Middle:** Selected active corner templates, where a corner is illustrated by a red arm and a green arm. For clarity of illustration, the red and green arms do not cover the whole extents of the two curves of a corner template. **Right:** Sketches of the image by deforming the active curve templates. **Bottom Three Rows:** Compositional relation among active Gabor elements, active curve templates and active corner templates.

ants to represent the observed image, by retrieving the argmax elements in the local max pooling.

3.2 Related Work

The proposed representation can be considered a concrete implementation of the idea of primal sketch [Marr, 1982; Guo et al., 2003]. It is inspired by two theories on the primary visual cortex or V1. One is the sparse coding theory of Olshausen and Field [Olshausen and Field, 1996] for the simple cells in V1. The other is the local max pooling proposed by Riesenhuber and Poggio [Riesenhuber and Poggio, 1999] as a function of the complex cells in V1.

As is presented in Section 1.3.1, in the sparse coding theory of Olshausen and Field, an image is represented by a linear superposition of a small number of wavelet elements selected from a large dictionary. The active curve templates and corner templates in our work are based on the representation of Olshausen and Field, where each template is a composition or grouping of a small number of Gabor filters. Because of such grouping, the representation based on such templates can be considered a further sparse coding of the selected Gabor filters similar to the wavelets used in the Olshausen-Field representation.

The proposed architecture of sum-max maps is a variation of the cortex-like structure of Riesenhuber and Poggio [Riesenhuber and Poggio, 1999]. The difference between the proposed sum-max architecture and the structure of Riesenhuber and Poggio is that we test and select explicit geometric templates of curves and corners, and there is a top-down pass that deforms the templates by retrieving the argmax elements in the local max pooling, in order to represent and sketch the observed image. This procedure is also similar to the inference algorithm in hierarchical object models such as [Epshtein et al., 2008].

The proposed representation follows the general principle of hierarchical compositionality and re-usable parts [Bienenstock et al., 1997]. In terms of detecting large structures by grouping smaller ones, the proposed sum-max architecture tests all possible templates exhaustively yet efficiently, instead of resorting to greedy schemes that make early decisions, which may not be correct. Specifically, no edge detection is performed before the active curve and active corner templates are detected. Decisions on edges are made after the templates are detected and deformed to match the observed data.

The proposed representation is different from feature detectors such as edge detectors [Canny, 1986], corner detectors [Harris and Stephens, 1988; Shi and Tomasi, 1994], or invariant features such as SIFT [Lowe, 1999]. The active curve and corner templates used in our representation are more sparse and usually cover larger ranges than these feature detectors.

The method of meaningful alignment [Desolneux et al., 2000] also seeks to find geometric structures in images. The line segments tested by meaningful alignment are not deformable, and the computation there does not employ recursive schemes such as those in sum-max maps.

3.3 Active Curve and Corner Templates

3.3.1 Active Curves

Following the active basis model [Wu et al., 2010], an image \mathbf{I} is represented by

$$\mathbf{I} = \sum_{i=1}^N c_i B_{\mathbf{x}_i + \delta \mathbf{x}_i, \theta_i + \delta \theta_i} + \mathbf{U}, \quad (3.1)$$

where each B is a Gabor filter with position \mathbf{x} and orientation θ . Each Gabor filter element has a pair of sine and cosine components, so the coefficient c_i also has a pair of components correspondingly. \mathbf{U} is the unexplained residual image. In this chapter, we

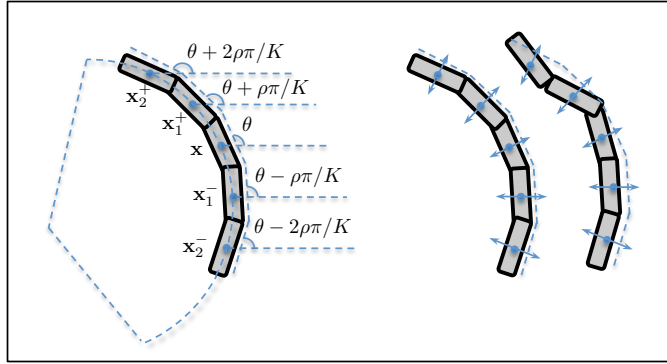


Figure 3.2: An illustration of active curve templates. **Left:** an example of active curve template $A_{\mathbf{x},\theta,\rho,2}$. **Right:** the active curve template and one of its possible variants. Blue arrows indicate the location deformation range of the Gabor elements.

fix the scale of Gabor filters, so this attribute is not shown in the subscript. We allow each Gabor filter element i to shift from its nominal position \mathbf{x}_i and orientation θ_i by $\delta\mathbf{x}_i$ and $\delta\theta_i$ respectively. The nominal positions and orientations are either designed as in this chapter or learned as in the active basis model, and the perturbations $\delta\mathbf{x}_i$ and $\delta\theta_i$ deform the template to fit the image \mathbf{I} . These perturbations can be inferred by the local max pooling of the Gabor filter responses.

As is shown in Fig.3.2, an active curve template A is a composition or grouping of Gabor filters placed with equal spacing on a straight line segment or a circular arc segment of a constant curvature. By design, the Gabor filters are placed so that the distance between two consecutive elements is 0.9 times their length, so these filters are slightly overlapping. As an approximation, we still treat them as being orthogonal to each other, so that the coefficients c_i are simply the projections of the image \mathbf{I} on the Gabor filter elements. As Gabor filters resemble the edge and ridge patches, they are symbolically illustrated by bars.

We use the center position \mathbf{x} , orientation θ , curvature ρ and length l to index each

active curve template, which is a composition of Gabor filter elements:

$$A_{\mathbf{x},\theta,\rho,l} = \{B_{\mathbf{x}_0,\theta_0}, B_{\mathbf{x}_i^+,\theta_i^+}, B_{\mathbf{x}_i^-, \theta_i^-}; i = 1, \dots, l-1\}, \quad (3.2)$$

where $\mathbf{x}_0 = \mathbf{x}$, $\theta_0 = \theta$, \mathbf{x}_i^\pm refer to the position of the i -th Gabor element, and \pm denotes the two sides relative to the arc center.

We quantize the orientations of Gabor filters into K values equally spaced by $\alpha = \pi/K$. The curvature is then quantized such that the orientation difference between neighboring elements is a multiple of α . We use this integer to index the curvature, thus an active curve template of curvature ρ satisfies: $\theta_i^+ = \theta_{i-1}^+ + \rho \cdot \alpha$, and $\theta_i^- = \theta_{i-1}^- - \rho \cdot \alpha$. The length is quantized by the length of Gabor filters. A curve of length l has $2l + 1$ Gabor filter elements.

The active curve template reduces to a line segment if $\rho = 0$, and it further reduces to a single Gabor element if $l = 0$. We use arcs as prototype shapes because they are more general than straight line segments, yet simpler to parameterize than more general curves such as those can be represented by splines.

3.3.2 Active Corners

An active corner template is a composition of two active curve templates. As is shown in Fig.3.3, an active corner template C is defined in terms of two active curve templates that share a common endpoint \mathbf{x}' :

$$C = \{A_{\mathbf{x}',\theta'_1,\rho_1,l_1}, A_{\mathbf{x}',\theta'_2,\rho_2,l_2}\}, \quad (3.3)$$

where we use superscript $'$ to indicate that the curve is indexed by its end point instead of its center. We require that the curvature ρ should be below a threshold b_1 , the length l_1 and l_2 should be larger than a threshold b_2 , and the angle $\alpha = \theta'_2 - \theta'_1$ should be confined to a proper range, e.g. $\alpha \in [\pi/2 - b_3, \pi/2 + b_3]$, where b_1 , b_2 and b_3 are parameters. Through these constraints, only long curves with low curvatures can be composed into corners.

3.4 Image Model and Template Matching Score

Following the active basis model, we assume that the probability distribution of the image, $p(\mathbf{I})$, is specified by tilting a background distribution $q(\mathbf{I})$, which may be considered as the distribution of natural images. Theoretical underpinnings of this model can be found in [Wu et al., 2010].

Given the N Gabor elements of a template, the log likelihood ratio of foreground image versus the background image is:

$$\log \frac{p(\mathbf{I})}{q(\mathbf{I})} = \sum_{i=1}^N [\lambda h(r_i) - \log Z(\lambda)], \quad (3.4)$$

where $h(r_i)$ is a monotone increasing transformation of Gabor filter response $r_i = |c_i|^2$, and $h(r_i)$ saturates for large r_i . λ is a parameter, and $Z(\lambda)$ is the normalizing constant for the corresponding exponential distribution. This is essentially a linear scoring scheme. If r_i is small, it will lead to a negative score. We want to detect those curve templates whose constituent elements contribute large positive scores. Different values of λ correspond to different expectations $E[h(r_i)]$ on the foreground, and $E[h(r_i)]$ is a monotone increasing function of λ . We treat all the selected Gabor elements with equal importance, so we use the same λ for all $h(r_i)$. This λ is kept as a parameter in our model, and the corresponding $Z(\lambda)$ is computed by pooling Gabor responses from natural images.

With such a model, we can simply define the template matching score of an active curve template as the log likelihood ratio score:

$$\lambda \left\{ h(r_0) + \sum_{i=1}^{l-1} [h(r_i^+) + h(r_i^-)] \right\} - (2l + 1) \log Z. \quad (3.5)$$

The template matching score of a corner template is defined by summing the scores of the two constituent curve templates.

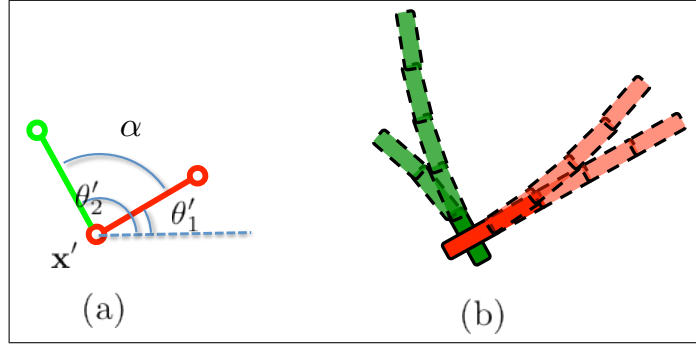


Figure 3.3: (a) A schematic plot of an active corner at position x' and two arm orientations θ'_1, θ'_2 . (b) By definition, an active corner template corresponds to many possible instances of corners. For example, we can choose any pair of red and green arms in (b) to compose them into an instance of the active corner in (a).

3.5 Scoring All Templates by Sum-Max Maps

According to the definition of the active curve templates in Eqn.(3.2), an active curve template of length l can be composed by an active curve of length $l - 1$, plus two Gabor filters (i.e., active curves of length 0) at the two ends:

$$A_{x,\theta,\rho,l} = \{A_{x,\theta,\rho,l-1}, A_{x_l^+, \theta+l\cdot\alpha, \rho, 0}, A_{x_l^-, \theta-l\cdot\alpha, \rho, 0}\}. \quad (3.6)$$

This immediately implies a recursive algorithm to compute the template matching score Eqn.(3.5) of long active curve templates by first computing the scores of shorter ones. Scores of length zero active curves are obtained by local maximum pooling of the Gabor filter responses, because the Gabor elements are allowed to shift their locations and orientations. The computation can be highly parallel, since scores of active curves at different positions, orientations and curvatures can be computed independently. Based on these two features, the following sum-max maps are proposed to compute the scores of all active curve and corner templates.

1. **S1 maps.** S1 maps store the responses of Gabor filters at all positions and orien-

tations. It is computed by convolving the Gabor filter with the image, followed by local normalization and saturation transformation on the responses. The local normalization step divides the response of a Gabor filter element by the local average of the responses of all the Gabor elements within a window centered at this element.

2. **M1** maps. To account for the deformations of active curve templates, each element on **M1** maps is computed as the maximum of **S1** map values within its local deformation range:

$$\mathbf{M1}(\mathbf{x}, \theta) = \max_{\delta\mathbf{x}, \delta\theta} \mathbf{S1}(\mathbf{x} + \delta\mathbf{x}, \theta + \delta\theta), \quad (3.7)$$

where $\delta\mathbf{x} = d \cdot (\cos \theta, \sin \theta)$, $d \in [-b_4, b_4]$ and $\delta\theta \in [-b_5, b_5]$.

3. **S2** maps. The **S2** maps store the scores of all the active curve templates. According to curve length, **S2** maps can be further divided into sub-layers. Scores at different sub-layers are computed as follows:

$$\mathbf{S2}(\mathbf{x}, \theta, \rho, 0) = \lambda \mathbf{M1}(\mathbf{x}, \theta) - \log Z, \quad (3.8)$$

$$\begin{aligned} \mathbf{S2}(\mathbf{x}, \theta, \rho, l) &= \mathbf{S2}(\mathbf{x}, \theta, \rho, l - 1) \\ &+ \mathbf{S2}(\mathbf{x}_{i-1}^+, \theta + l \cdot \alpha, \rho, 0) \\ &+ \mathbf{S2}(\mathbf{x}_{i-1}^-, \theta - l \cdot \alpha, \rho, 0), \end{aligned} \quad (3.9)$$

where in Eqn. (3.9), we assume $l > 0$.

The template matching score of active corner templates can be expressed as

$$\mathbf{S2}(\mathbf{x}', \theta'_1, \rho_1, l_1) + \mathbf{S2}(\mathbf{x}', \theta'_2, \rho_2, l_2). \quad (3.10)$$

According to Eqn.(3.3), for a given point \mathbf{x}' and a given supporting curve orientation θ'_1 , there are $2b_1 \cdot (|l| - b_2) \cdot |\delta|$ different corners. If we assume that there

can be at most one corner instance among these, we shall choose it by solving the following problem:

$$\max : \quad \mathbf{S2}(\mathbf{x}', \theta'_1, \rho_1, l_1) + \mathbf{S2}(\mathbf{x}', \theta'_2, \rho_2, l_2),$$

subject to the constraints on the ranges of ρ and l for each curve, and δ spanned by the two curves.

Since there is no constraint on the relation between (ρ_1, l_1) and (ρ_2, l_2) , this problem can be solved by first solving smaller problems. The corresponding computation steps can be summarized as the sequentially computing of the following maps:

4. **M2** maps

$$\mathbf{M2}(\mathbf{x}', \theta') = \max_{\rho, l} \mathbf{S2}(\mathbf{x}', \theta', \rho, l), \quad (3.11)$$

where $-b_1 < \rho < b_1$ and $l > b_2$.

5. **S3** maps

$$\mathbf{S3}(\mathbf{x}', \theta'_1, \theta'_2) = \mathbf{M2}(\mathbf{x}', \theta'_1) + \mathbf{M2}(\mathbf{x}', \theta'_2), \quad (3.12)$$

where **S3** maps are the scores for all the possible active corner templates. In computing **S3** maps, we should only compute over (θ'_1, θ'_2) pairs so that the spanned angle α is within the defined range.

In the local maximization steps 2 and 4, the optimal deformations that achieve the maxima can be obtained by argmax operations.

The sum-max maps score all the active curve and corner templates without early decisions such as edge detection. The max operations make the scores invariant to shape deformations.

When converting index of the **S2** maps from arc center to two ends, the index entries will override because each arc now has two indices (shown in Fig. 3.4). We resolve this

Algorithm 3.1: Matching Pursuit of Active Curve Templates

Input: S1 maps after inhibition, threshold T .

Output: Selected active curve templates $\mathbf{A} = \{A_1, \dots, A_N\}$

```

1 repeat
2   Compute M1 and S2 maps.;
3    $s^* \leftarrow \max \mathbf{S2}(\mathbf{x}, \theta, \rho, l)$ ;
4    $A^* \leftarrow \arg \max \mathbf{S2}(\mathbf{x}, \theta, \rho, l)$ ;
5   Decompose  $A^*$  into a set of Gabor elements  $\mathbf{B}$ ;
6   foreach  $B_{\mathbf{x}, \theta} \in \mathbf{B}$  do
7      $(\mathbf{x}^*, \theta^*) \leftarrow \arg \max \mathbf{M1}(\mathbf{x}, \theta)$ ;
8     foreach  $B_{\mathbf{x}', \theta'}$  correlated with  $B_{\mathbf{x}^*, \theta^*}$  do
9        $\mathbf{S1}(\mathbf{x}', \theta') \leftarrow 0$ ;
10      Update M1 map entries in the neighborhood of  $B_{\mathbf{x}^*, \theta^*}$ ;
11    $\mathbf{A} \leftarrow \mathbf{A} \cup \{A^*\}$ ;
12 until  $s^* < T$ ;

```

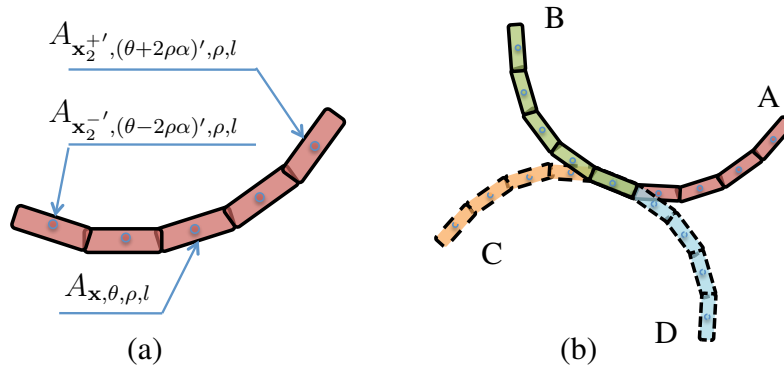


Figure 3.4: (a) Three indices of an active corner, by center and two ends. (b) When changing center point index to end point index, one index will correspond to two curves (such as A,B or C,D). We solve this overriding issue by extending the orientation index from $[0, K - 1]$ to $[0, 2K - 1]$, and setting the orientation of one curve to $\theta + K$.

overriding by doubling the number of orientations from K to $2K$, which corresponds to changing the orientation range from $[0, \pi]$ to $[0, 2\pi]$.

3.6 Selection of Corner and Curve Templates for Single Image

As can be seen from Fig.3.5, most responses on **S3** maps are very low, because corners are rare events and highly location sensitive. So, after the **M3** maps are computed, we can simply apply a threshold and use non-maximum suppression to select active corner templates for image representation. Specific curves and their Gabor elements that belong to these selected corners can then be obtained by a top-down process that retrieves the optimal deformations from the argmax maps of the local max operations.

To select active curve templates that are not overlapping with the selected active corner templates, we need to inhibit the scores of those curves that overlap with the selected corners. To do this, we trace back the active curve templates that belong to the selected active corner templates, and further trace back the Gabor elements of these curves. For each Gabor filter of the selected and deformed corner, we let it inhibit its nearby Gabor filters whose correlations with the current one exceeds a certain threshold, by setting the responses of those Gabor filters to zero. In practice, the correlations can be computed before the algorithm starts, so that inhibition can be done very efficiently.

After inhibition, we can select salient active curve templates by sequentially selecting the current best curve with the highest score, and let the selected curve inhibit the overlapping ones. This is essentially a matching pursuit [Mallat and Zhang, 1993] process, where in each step, we select a group of Gabor elements packed into an active curve template. The pursuit algorithm stops once the selected score is lower than a threshold T . The process is summarized in Algorithm 3.1.

3.7 Active Curves and Corners as Features for Object Images

We can also select and compose curve templates and active corner templates into an object template by learning from multiple images, either generatively or discriminatively. To accommodate geometric variations of active curve templates across multiple object images, we allow the active curves and active corners to deform at their own group level, by changing their parameters in a small deformation range. Following the **S2** and **S3** maps, the score maps of group deformable templates for active curves can be computed as:

$$\mathbf{M2}^+(\mathbf{x}, \theta, \rho, l) = \max_{\delta\mathbf{x}, \delta\theta, \delta\rho} \mathbf{S2}(\mathbf{x} + \delta\mathbf{x}, \theta + \delta\theta, \rho + \delta\rho, l) \quad (3.13)$$

Similarly, for active corner templates:

$$\mathbf{M3}^+(\mathbf{x}', \theta'_1, \theta'_2) = \max_{\delta\mathbf{x}} \mathbf{S3}(\mathbf{x}' + \delta\mathbf{x}, \theta'_1, \theta'_2) \quad (3.14)$$

As the two arms in an active corner template are already allowed to have orientation deformation, only corner position changes are needed to deform the active corner template. Note that in the original **M2** maps are still computed as they are needed for the **M3**⁺ maps.

3.8 Experiments

3.8.1 Implementation and Parameters

The recursive and parallel algorithm is mainly composed of the sum and max operations, which can be computed very efficiently. However, if we keep all the maps and argmax maps, the memory consumption can be excessive. In practice, since only a small number of Gabor elements will be backtracked in top-down process, we choose

| Parameter | Notation | Value |
|--------------------------------|-----------|--|
| Gabor Filter Size | s | 15×15 pixel |
| Number of Orientations | K | 25 |
| Position Change | b_4 | ± 1 pixel |
| Orientation Change | b_5 | ± 1 orientation |
| Curve Curvature Range | ρ | $\rho \in [-4, 4]$ |
| Curve Length Range | l | $l \in [0, 3]$ |
| Corner Arm curvature Threshold | b_1 | 2 |
| Corner Arm Length Threshold | b_2 | 0 |
| Corner Angle Range | b_3 | $\pi/6, \delta \in [\pi/3, 2\pi/3]$ |
| Weight | λ | 2.5 |
| Arc Score Threshold | T | $3\{\lambda \cdot E[h(r); \lambda] - \log Z\}$ |
| Corner Score Threshold | b_6 | $2.3T$ |

Table 3.1: Parameters for all the experiments on single images

not to save the argmax maps and instead re-do the local maximum computing for selected corners or curves to retrieve their argmax deformations. For an image of size 341×512 and parameters in Table.3.1, the algorithm would consume about 1.2 GB of memory. Although we listed 12 parameters in the table, most of them are not essential, and only for properly discretizing the space of possible curves and corners. Only λ , T , and b_6 are of importance. We calibrate the last three parameters according to the image in Fig.3.1, and fix these parameters for all the experiments on single images below and those in the project page.

3.8.2 Sketch Natural Images

Fig.3.5 shows the intermediate results for the example image in Fig.3.1, where each row shows maps from different layers, and each column corresponds to curves or corners in different orientations. As the size of active curve and corner templates become larger, the high scores on the corresponding maps become more sparse and clustered. Maps at S3 level become very sparse, and high scores only appear around the true corner points

on the image. As evidences are accumulated over larger areas, it becomes safer to make decisions at higher levels. Since the selected templates are much larger than a single Gabor filter, the resulting representation is more sparse than filter representation.

More results are shown in Fig.3.6 and in our project page¹. For each image, we also show the result from edge link [Kovesi, 2012] as a reference. We would like to stress that our representation is more sparse than the representation based on edge points, yet still captures the main structures in the images.

3.8.3 Learning from Object Images

In this section, we explore the possibility of using active curves and active corners as features for object modeling. For simplicity in implementation, we assume object images are all from the same view, and use very simple learning and inference algorithms, only to evaluate the eligibility of using the proposed visual concepts as an intermediate level representation of objects.

3.8.3.1 Experiments on Single Object Images

On single images with uncluttered background, the proposed representation can be effortlessly adapted to represent objects. Fig. 3.7 shows that on a single horse image, the output active corners and their composing active arcs are already able to represent the specific shape of that object image. Parameters in this example are also the same as in Table. 3.1, except we changed the size of the Gabor elements because the input image size is different.

¹<http://www.stat.ucla.edu/~wzhu/IRAC>

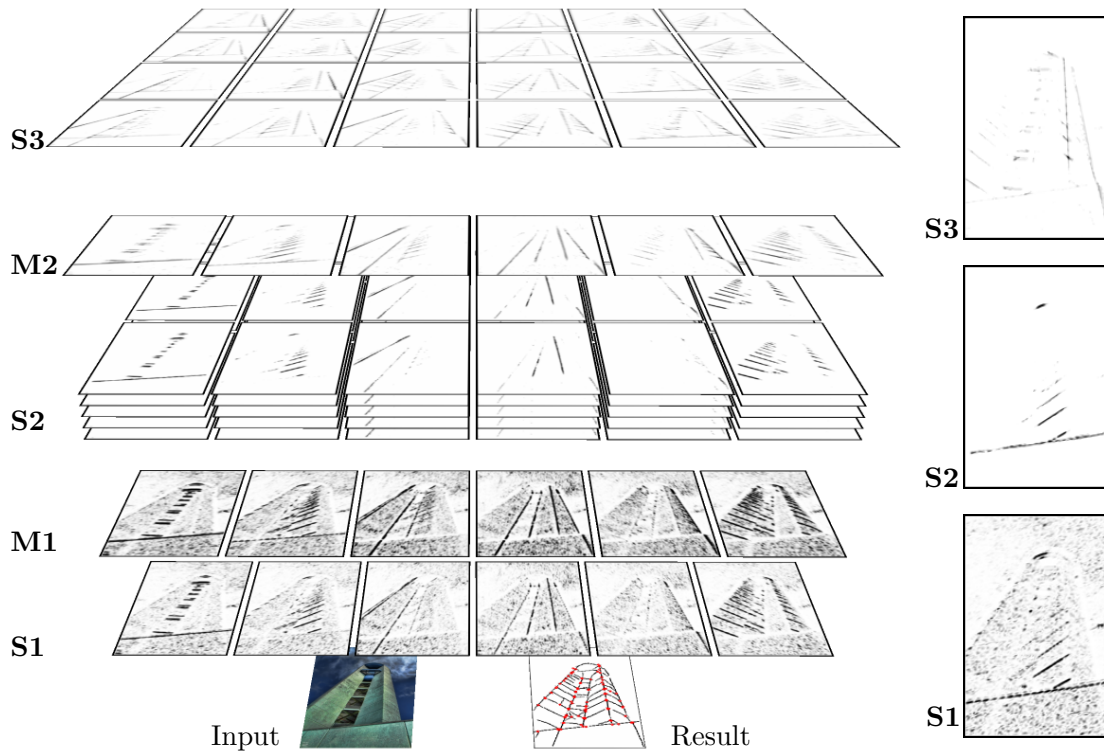


Figure 3.5: The sum-max data structure, using the image in Fig.3.1 as an example. Maps for different orientations are laid out horizontally, and maps for different layers are laid out vertically. Sub-layers in the S2 maps correspond to active curves of different lengths, and depth direction in the S2 maps corresponds to curves of different curvatures. Depth direction in the S3 maps corresponds to the orientation of the second arm. Maps on the right are typical maps from each sum layer.

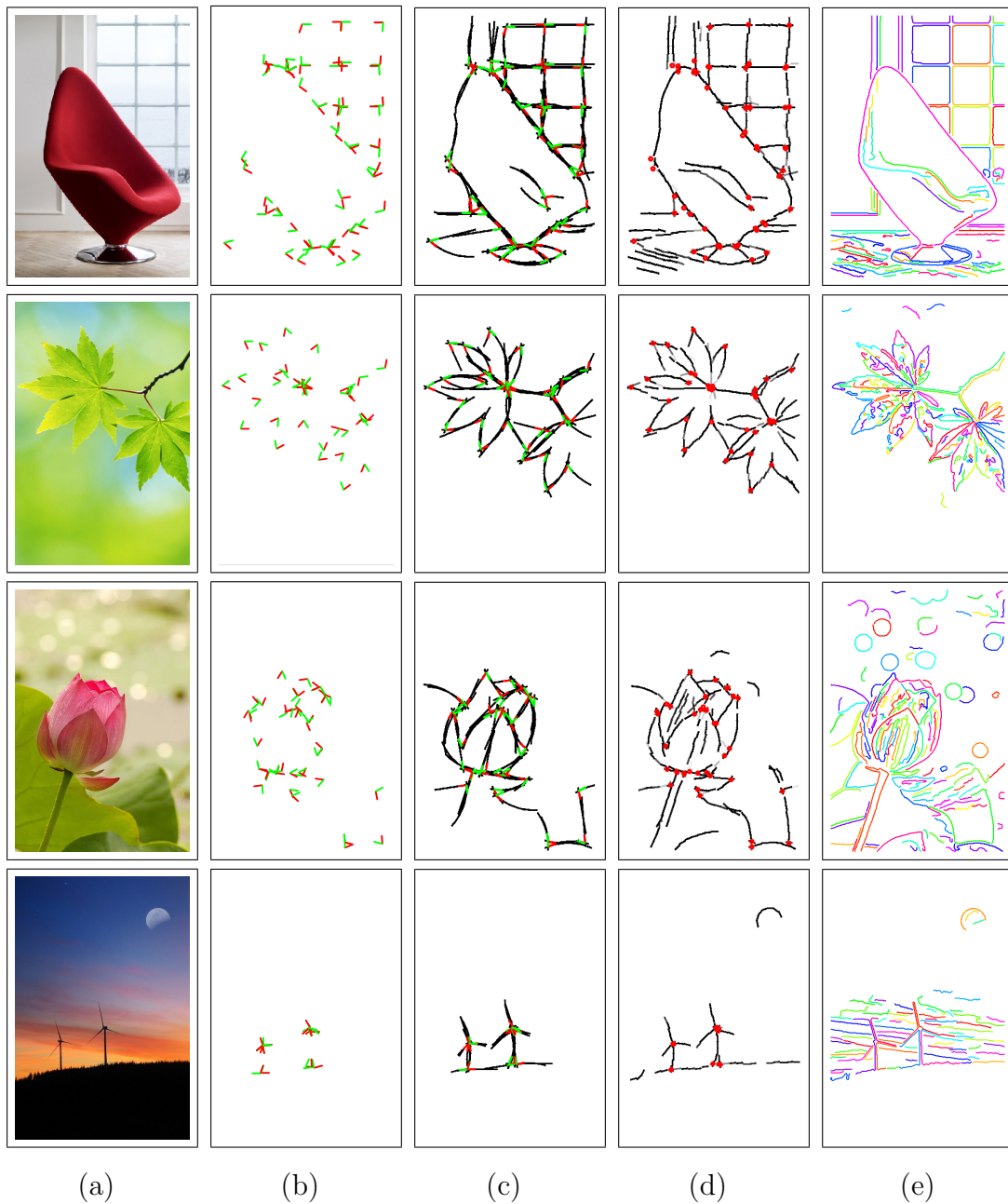


Figure 3.6: More results on the single image representation. (a) Input image. (b) Selected corners. (c) Selected corners with supporting curves. (d) Deformed curves. (e) Output from edgeline code.

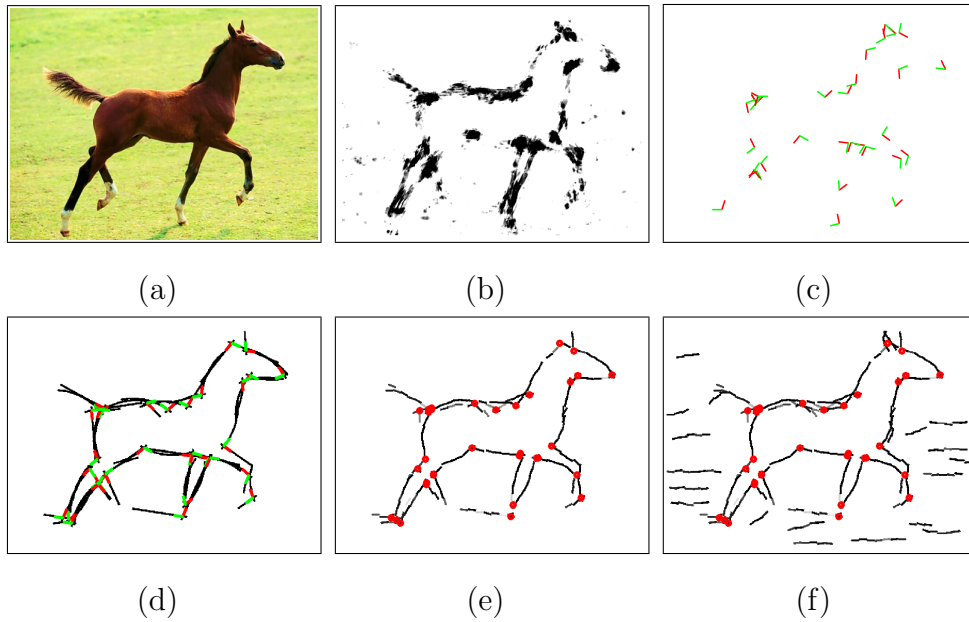
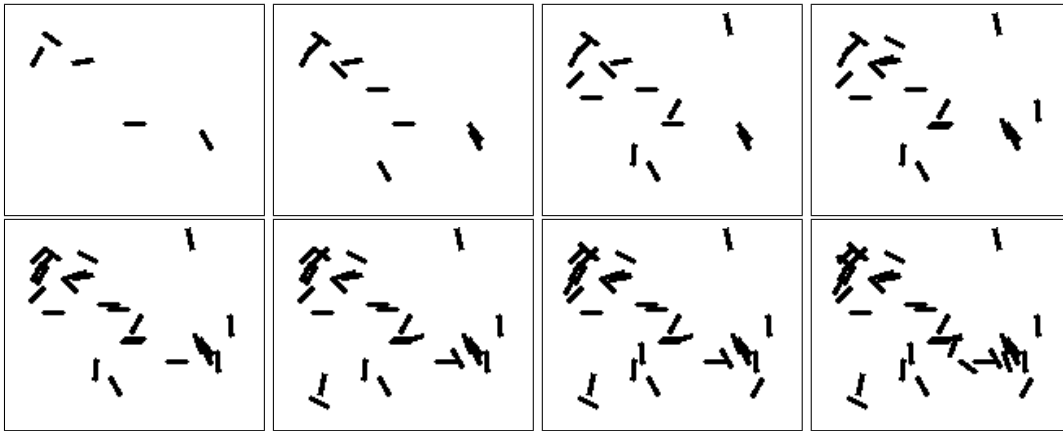
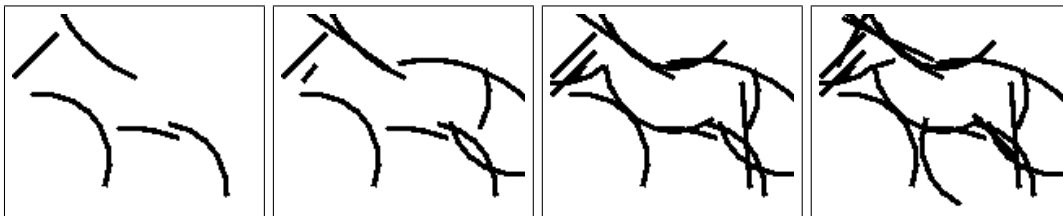


Figure 3.7: Representing images of objects. (a) Original Image. (b) Corner saliency map, defined as $\max_{\theta'_1, \theta'_2} \mathbf{S3}(\mathbf{x}', \theta'_1, \theta'_2)$. (c) Selected corner prototypes. Angles spanned by the corners are from red arm to green arm. The arms only cover the small portions of the arc primitives of the corners. (d) Prototypes of the selected corners and their composing active curves. (e) Deformed active curves and their composing active corners. (f) Deformed corners and arcs.



(a) Templates using 5, 10, 15, \dots , 40 active basis elements.



(b) Templates using 5, 10, 15 and 20 active curves.

Figure 3.8: Learned templates by adaBoost, using active basis and active curves as features. For each case,

3.8.3.2 Discriminative Learning on Multiple Images By AdaBoost

To illustrate the use of the active curves and active corners as features for object category modeling, we collect a dataset and compare the testing errors of adaBoost classifiers [Freund and Schapire, 1996; Viola and Jones, 2001] based on different sets of features. We collected 280 horse images, 186 leaf images as positive examples, and 559 images as negative examples. The leaf images are from Vision lab of Caltech.

Taking the horse category as an example, we show the learned templates using only active basis ($\mathbf{M1}$ maps) and only active curves ($\mathbf{M2}^+$ maps) in Fig.3.8. Specifically, we show the active basis templates using 5 to 40 basis at every 5 basis, and the active curve templates using 5 to 20 curves at every 5 curves.

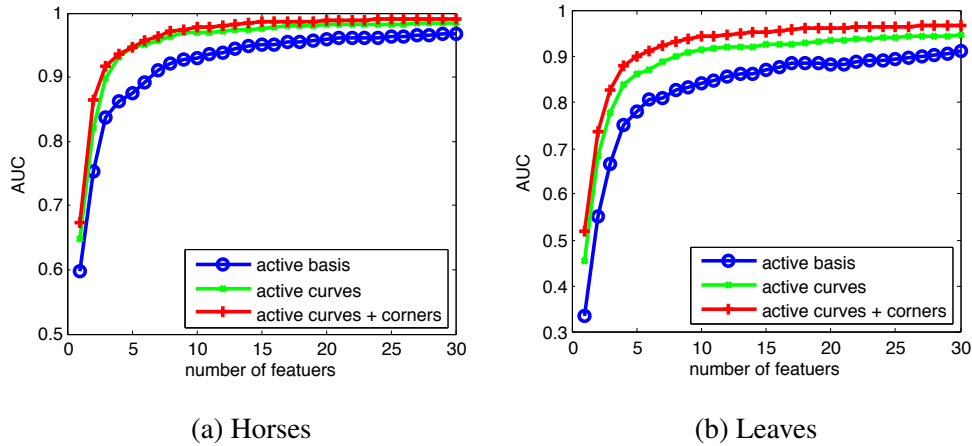


Figure 3.9: AUC curves of boosted classifiers on horse images and leaves images, using active Gabor features, active curve templates, and both active corner and curve templates.

From the figure, we can see that though using the discriminative learning method, the learned active curve templates still form the shape of a horse, while in the active basis case, the template becomes much less meaningful. As the active curves structures are much larger than active basis, the possibility of their frequent occurrence on a shared location and orientation in the negative image set is much lower than that of active basis. As a consequence, classifiers using active curves on the aligned object boundary will become strong classifiers, thus adaBoost will mostly select these classifiers, leading to a meaningful template many describing the object. On the contrary, the weak classifiers using active basis may model the accidental alignment of some active basis on negative images, thus the learned template become noisy and less meaningful.

Using 20% of the images in each category as training examples, we further run adaBoost to build three discriminative models, by replacing the Haar features with the feature pools of $\mathbf{M1}$ maps, $\mathbf{M2}^+$ maps, and both $\mathbf{M2}^+$ and $\mathbf{M3}^+$ maps, corresponding to use only active basis, with active curves and further with active corners.

For each feature combination, we compute the AUC (Area Under the ROC Curve)

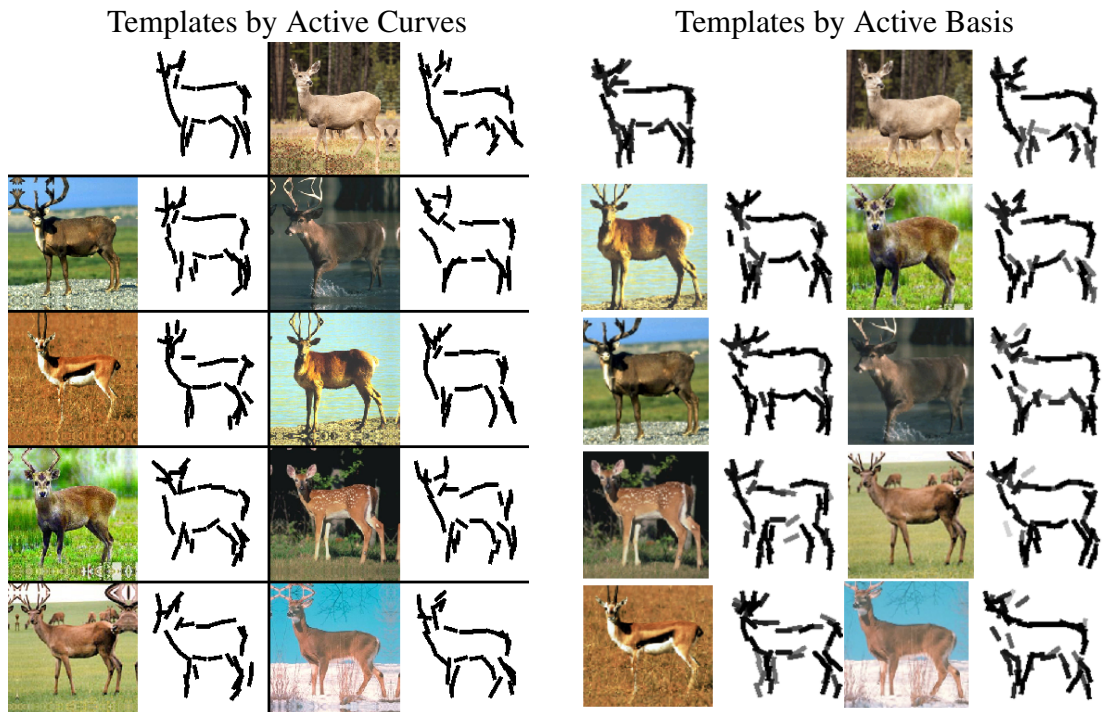


Figure 3.10: A comparison of the deer templates generatively learned using active curves and active basis.

scores of the classification results on the rest 80% of the image dataset. For each category, we repeat the above procedure five times by randomly splitting the data at each time, and show the averaged AUC in Fig.3.9. From the results, we can see that to achieve the same classification performance (AUC), the number of active corners or curves used is always less than that of active Gabor filters.

3.8.3.3 Generative Learning Using Shared Matching Pursuit

Object templates can also be learned by assuming generative models. For simplicity, we directly use the active basis formulation, and learn linear additive image models composed of active curves instead of active basis. Then we can use the shared matching pursuit algorithm from active basis to learn the template, by just extending the inhibition step to inhibition of Gabor elements correlated with each Gabor element composing the

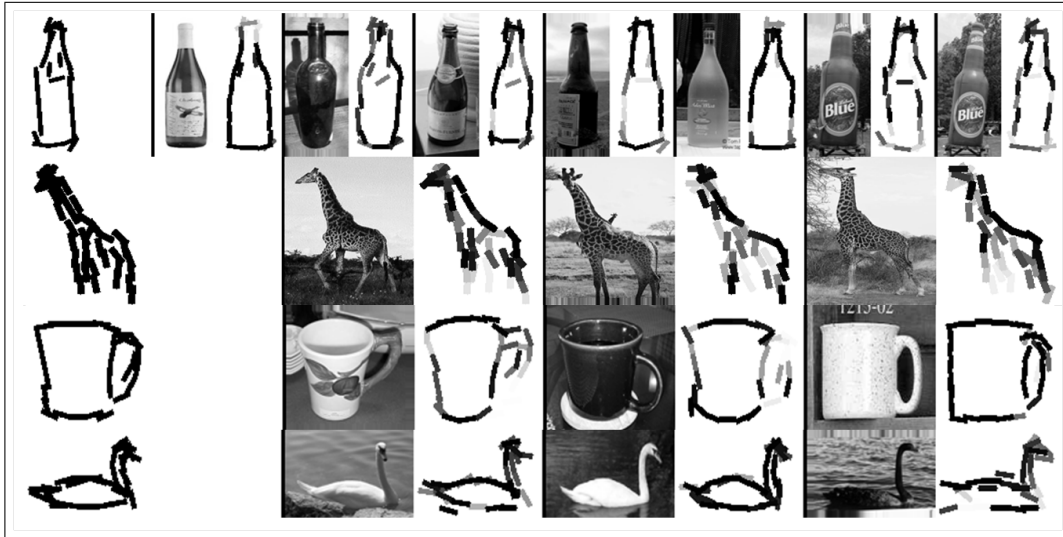


Figure 3.11: Learned object templates and their deformations on training images. The learned template for each category is shown in the first column. The rest of the templates are deformed versions of images on their left. Grayscale of the strokes in the deformed templates indicates the strengths of Gabor responses, where darker means stronger.

selected active curve in each iteration.

Figure.3.10 shows the generatively learned templates composed of active curves, together with these composed of active basis. From the two sets of templates we can see that the shapes learned by active curves are less fragmented, as the added hierarchy of active curves replaces large deformation of individual elements with an intermediate level of deformation of active curves plus small deformation of individual elements.

3.8.3.4 Object Detection on ETHZ object dataset

We further run experiments on the publicly available ETHZ object dataset [Ferrari et al., 2010], where templates learned using the first half of images in each category are shown in Fig.3.11. As object bounding boxes in training images are in different aspect ratios,

given the training images, we compute the geometric mean of those aspect ratios, and resize the cropped images in bounding boxes to the geometric mean.

Since the parameter λ on each of the Gabor filters are prefixed before learning, we want to adjust these weights before the learned templates are used for object detection. To keep using the generative framework, we use the ℓ_2 regularized logistic regression for adjustment, which optimize:

$$\min_{\lambda_1, \dots, \lambda_N} \frac{1}{2} \sum_{n=1}^N \lambda_n^2 + C \sum_{m=1}^M \log \left\{ 1 + \exp \left[-y_m \sum_{n=1}^N \lambda_n h(r_{mn}) \right] \right\}, \quad (3.15)$$

where $y_m \in \{+1, -1\}$ is the label of the m -th example, and r_{mn} is the response of n -th Gabor element in m -th image. In Eqn.(3.15), we assume there are N Gabor element in the template composed of active curves, and totally M training examples (including both positive and negative).

We use the LIBLINEAR implementation of this logistic regression, which is publicly available on the web page for [Fan et al., 2008]. In training the weights, the positive examples are extracted by back tracing the Gabor element responses from images used in learning the templates, and negative examples are back traced responses in randomly cropped windows from positive images, regardless of the overlap between the window and the object bounding box.

We use the sliding window method to perform object detection. Given a testing image, we first construct the image pyramid, where the image scale change between consecutive layers is 1.1x. For each layer, we compute the $\mathbf{M2}^+$ maps for active curves whose length, orientation and angle combination matches those of the active curves used in the learned templates.

For object aspect ratio variation, we choose to change the template aspect ratio instead of resizing images and re-computing active curve score maps. This is implemented by changing the horizontal axis coordinate of each active curve to $\{0.9, 1.0, 1.1\}$ times the original value, while fixing the vertical axis unchanged. The orientation and

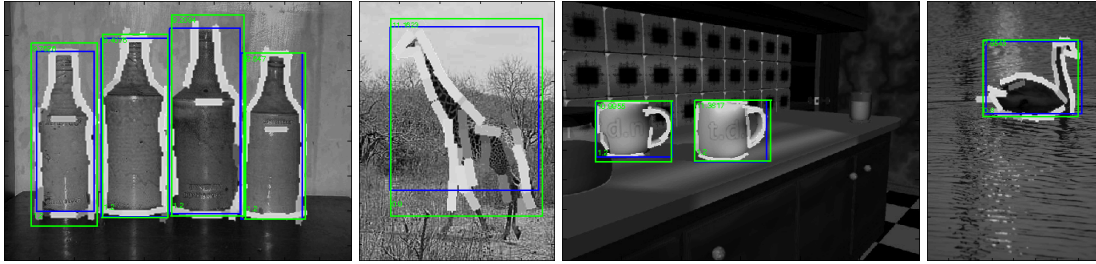


Figure 3.12: Sample detection results on ETHZ dataset. The blue bounding boxes are for the ground truths, and green ones are reported by our detection algorithm. For each reported window, we also retrieve its the deformed template, which is also imposed onto original images.

curvature of these curves are kept unchanged. For each aspect ratio we run separate sliding windows using the same weights learned from the logistic regression.

After getting the score for each potential object window in the image pyramid, we report object detection windows using non-maximum suppression, under the criterion that reported windows should not have intersection over union area ratios less than 50%. For each reported window, we further go back to retrieve the deformations of active curves and their composing Gabor filters, which could sketch the detected object. Examples of detection results are shown in Fig.3.12.

We perform the detection task with 5 repetitions as in the experiments of k-adjacent segments [Ferrari et al., 2010]. For each repetition, we randomly select half of the images in the category as training data, and the rest together with all images from all other categories as testing data. We use the code from [Ferrari et al., 2010] to compute the detection-rate v.s. FPPI (false positives per image) curves for each repetition and their averaged curves. The result averaged curves are shown in Fig.3.13. From the results, we can see that without complicated shape modeling, the simple additive model using our active curves can already achieve comparable performance as [Ferrari et al., 2010].

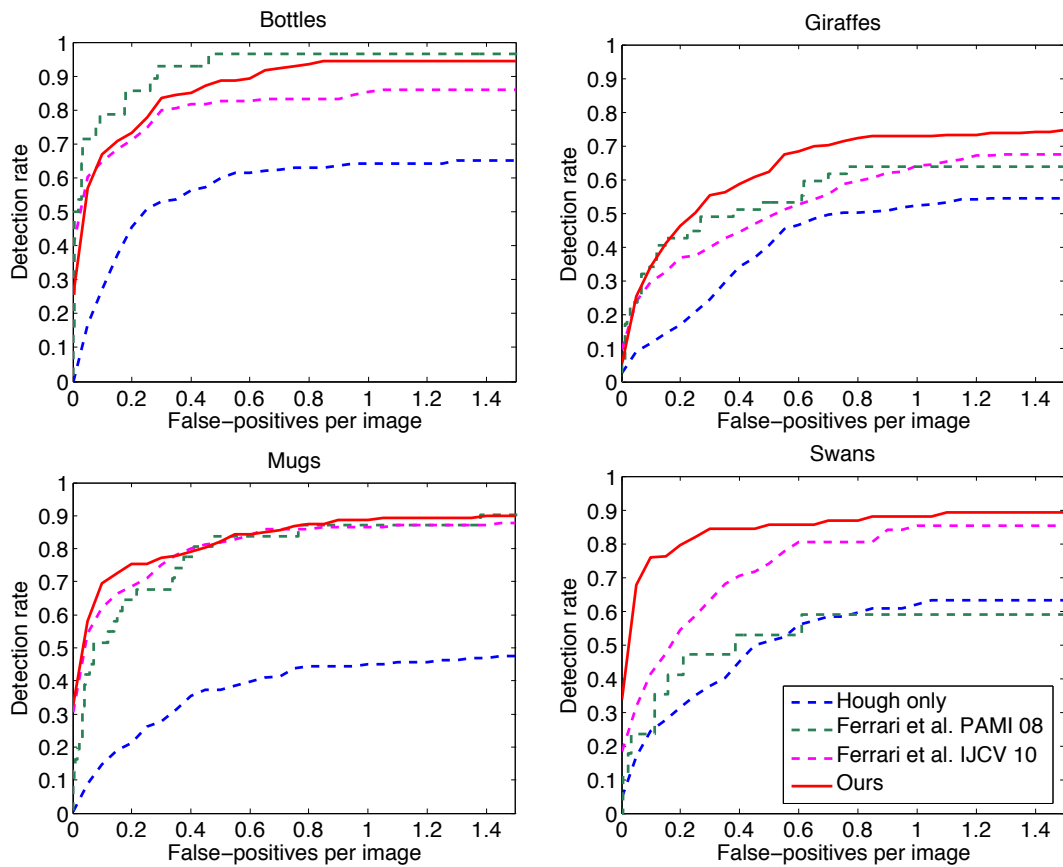


Figure 3.13: Detection results on the ETHZ dataset. We use the same evaluation procedure and criterion as [Ferrari et al., 2010]. Blue, green and purple curves are from [Ferrari et al., 2010]. Note that only one repetition is used in producing the green curve.

3.9 Discussion

We propose a sparse image representation based on deformable templates of simple geometric structures such as curves and corners. We also propose a computational architecture that exhaustively tests all the possible deformable templates in the bottom-up pass and then retrieves the deformed templates in the top-down pass. Such templates can be pursued from single images. They can also be composed into object templates by learning from multiple images either generatively or discriminatively. The template matching scores of curve and corner templates can also be used as features for object modeling.

Besides corners and angles, it is possible to compose the curve templates into templates of more complex geometric shapes, such as triangles, rectangles and ellipsoids. And there might be equivalent compositions for these more complex shapes, such as comping triangles by three corners or tree line segments. While these problems are certainly interesting for further investigation, on natural images, the complex structures are much less frequent than curves and corners, which makes exhaustively computing them less useful.

In our current scheme of template matching, we only include the responses of Gabor filter elements on the line or arc segments. We should also pool the responses of those elements that are away from the line or arc segments, in the form of local averages or local max, which are weighted negatively in order to model the flatness of the regions around the line or arc segments.

Parameterizing the geometric structures such as curves and corners provides us the opportunity of quantizing the spaces of these structures. The practical benefit of this quantization is that instead of assuming infinitely many structures, we can only assume a subset of them, which could be typical examples representing the whole structure family, yet small enough so that exhaustive computation over all of them is affordable.

Moreover, by assuming these structures are composed of shared substructures, the computation could be more efficient as these for substructures can be reused. The idea of combining quantization and composition motives the next chapter, which construct a hierarchical compositional representation that grows from image pixels in 2D images all the way to 3D object bounding volumes.

CHAPTER 4

Integrating 3D and 2D Representations by AND-OR Tree

This chapter proposes an AND-OR tree structure (AoT) and an efficient algorithm that learns 3D deformable object templates from view labeled object images. The proposed AoT is composed of a Geometry AoT (G-AoT) and Appearance AoTs (A-AoTs). The G-AoT hierarchically partitions the target object volume into sub-volumes, where panels associated with sub-volumes define the geometry of the template. The A-AoT defines hierarchies of stylized visual concepts on these panels and corresponding images at different views, as well as their equivalence classes as deformations. The AoT quantizes the continuous and infinite space of 3D object geometry and appearance into a discrete and finite space, which also fills in the gap between stylized 3D object representations and their observations as image pixels. With prefixed AoT, the model structure learning problem is also transformed into a structure search problem, where an efficient dynamic programming algorithm called AND-OR search can be used. Experiments show that the proposed approach can learn meaningful 3D car template, and perform object detection and pose estimation tasks. Experiments on a public dataset also show that the proposed algorithm performs better on object detection and comparable on pose estimation tasks with recent approaches [Liebelt and Schmid, 2010].

4.1 Introduction

4.1.1 Motivation and objective

The idea of using 3D models with stylized shape primitives for object recognition has long been suggested by researchers in both computer vision and psychology [Biederman, 1987; Dickinson et al., 1992; Marr, 1982]. However, little progress has been made after its emergence in the early 90's, because: 1) There is a representation gap between the hidden concept of stylized 3D shape primitives and their observed appearance as raw image pixels, due to deformation, projection and lighting etc.; 2) The number of possible 3D primitive compositions is huge, which makes designing efficient algorithms a difficult problem.

In this chapter, we take car images as an example, and propose a method that learns a 3D deformable template from view labeled object images. We propose to address the above issues using an AND-OR Tree structure, which:

- Quantizes the continuous and infinite space of object geometry and appearance into a discrete and finite space.
- Hierarchically decomposes this space, so that computational complexity can be reduced by reusing sub-compositions.
- Poses the model structure learning problem as a structure search problem in a prefixed search space.

Using the AND-OR Tree (AoT), we are able to learn 3D object templates composed of stylized planar shapes, such as the one shown in Fig.4.1(a), as well as project and deform the 3D template to match observed images, such as the examples shown in Fig.4.1(b)-(d).

4.1.2 Overview of the Proposed Method

An overview of the proposed AoT structure is shown in Fig.4.2. We first decompose the space of our model into geometry and appearance spaces, which are further quantized by AND-OR Tree for Geometry (G-AoT) and AND-OR Tree for Appearance (A-AoT) respectively. More detailed illustrations of these AoTs are shown in Fig.4.4 and Fig.4.5 respectively.

In this chapter, geometry refers to the 3D placement of our appearance components defined in corresponding A-AoTs. The G-AoT quantizes and recursively decomposes the bounding volume of target object category into smaller volumes. During the decomposition, each volume can be either split into two smaller volumes or directly connected to panels on the front surface or inscribing the volume, which serve as root nodes for A-AoTs.

As is shown in Fig.4.5, each A-AoT defines a regularized and deformable shape template representing the appearance of a certain area of images decided by the view and the geometry of corresponding panels. Given the geometry, these A-AoTs are derived by quantizing the parameters of simple 2D shapes, such as circles, rectangles, etc. Starting from the root node of an A-AoT, the corresponding shape is decomposed into parameterized line segments, which are realized by specialized active curves [Hu et al., 2011] after projection, and further decomposed into deformable Gabor filters [Daugman, 1985] as active basis [Wu et al., 2010].

The above hierarchy of decompositions constitutes the AND nodes in the proposed AoT, whereas OR nodes connect the volumes to their alternative decompositions in G-AoT, and enable deformations for shapes, line segments etc. along the hierarchy of A-AoT.

Connecting the two types of trees, the G-AoT and A-AoTs construct a coherent AoT which quantizes the model space spanning from 3D object bounding volume to

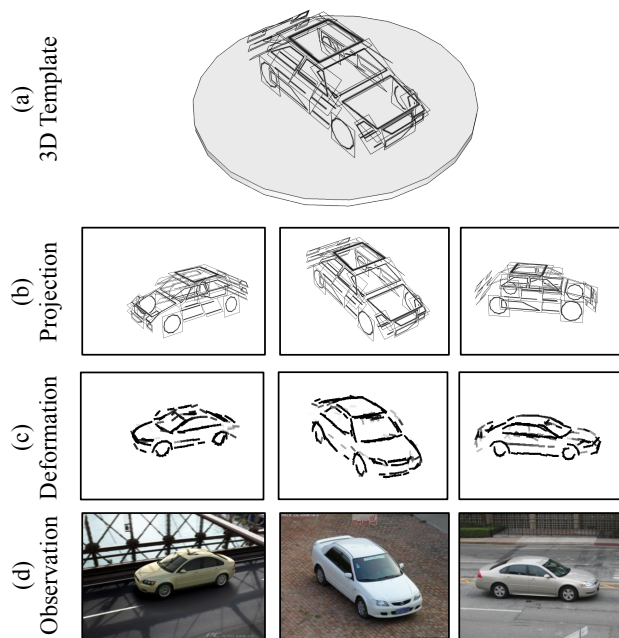


Figure 4.1: Overall view of the proposed object representation. (a) A learned 3D car template, which is composed of planar part templates. (b) At each specific view, the learned 3D template is projected to derive a 2D template. (c) 2D templates are then deformed to match observations, which are images shown in (d).

image pixels generated by Gabor filters.

In learning stage, we want to find the best volume decomposition together with their shape templates which best explain the observed training images. With the proposed AoT, this learning problem is posed as a structure search problem that finds the best composition among the possible compositions defined by the AoT. Using the information gain as a quantitative measure, this search problem can be solved efficiently by dynamic programming.

Given a testing image, we generate 2D image templates by projecting the 3D object template to a set of hypothesized views, and match these 2D templates with the image using dynamic programming. In this way, the 3D template can be used to detect objects from arbitrary views in the continuous view sphere, including those might not seen in

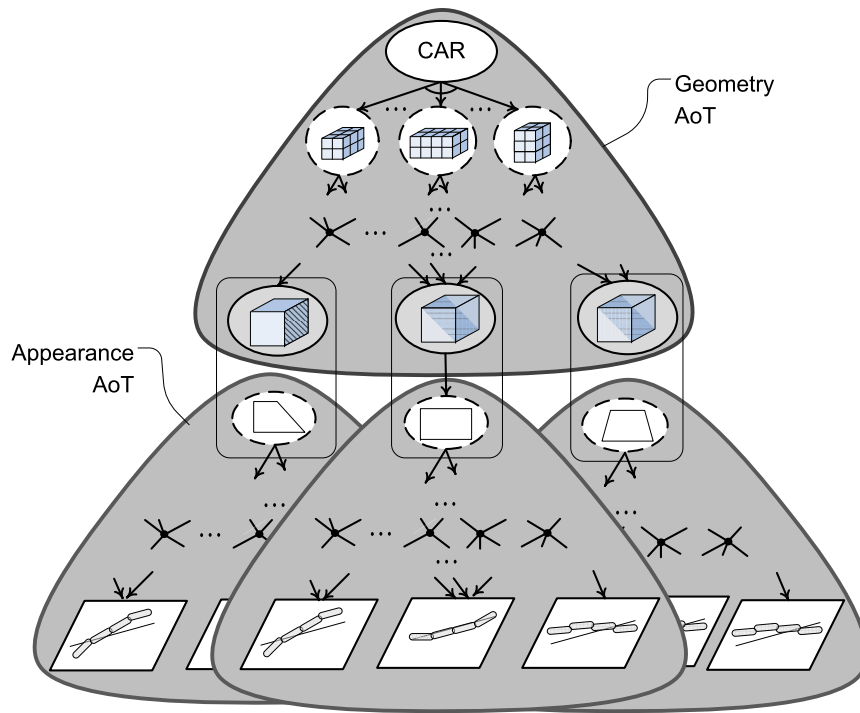


Figure 4.2: An overview of the AND-OR Tree for the representation space

training examples.

4.1.3 Related Literature

Most models in the 3D object recognition literature can be categorized into two classes:

Object-centered models. Early work proposes to represent 3D objects as a composition of stylized volumetric primitives, such as Geons [Biederman, 1987] by Biederman et al., 3D primitives [Dickinson et al., 1992] by Dickinson et al, and the generalized cylinder representation by Marr [Marr, 1982]. These representations are usually illustrative. In fact, many of these primitives are still used as building components in modern CAD design softwares. However, from object recognition perspective, it is difficult to learn and recognize those 3D primitives because of the intrinsic ambiguity of inferring 3D shapes from single views and the gap between these hidden concepts and observed

image pixels. Though using planar shape templates on planes associated with volumes, our proposed method resolves view ambiguity by pooling over partial evidence across views, and also fills in the representation gap by allowing a hierarchy of decomposition and deformations that fits the regularized shape to its image observations.

Recently, many papers ([Yan et al., 2007; Arie-Nachimson and Basri, 2009; Kushal et al., 2007; Liebelt et al., 2008; Hsiao et al., 2010]) propose recognizing objects using point based 3D models, with appearance as a set of SIFT [Lowe, 1999] descriptors, or its quantized version [Hsiao et al., 2010]. The SIFT descriptor is good at raising 3D hypothesis by creating point correspondences. However, by only extracting SIFT features, other salient image information is neglected, such as object boundaries. Our proposed model is complementary to these models, since it mainly relies on boundary information from images.

Viewer-centered models. Viewer-centered models date back to the aspect-graph representation by Koenderink and Doorn [Koenderink and Doorn, 1976, 1979]. These models are becoming popular, as they ([Thomas et al., 2006; Su et al., 2009; Payet and Todorovic, 2011]) do not need to construct a view consistent 3D model, and can easily use many developments ([Lazebnik et al., 2006; Dance et al., 2004]) on the widely studied problem of object recognition on single or a few views. Right because the view consistency prior is not explicitly used, information in individual images is not fully shared across views, which suggests more data are required to learn a robust model for each view.

In terms of model hierarchy, most of the models above are flat. For models with parts or feature groups, they are either prefixed by creating a grid on object images [Liebelt and Schmid, 2010; Payet and Todorovic, 2011], or step-wise clustered without optimizing the global objective of their model [Su et al., 2009].

Recently, there are models assuming a 3D geometry model with 2D view specific appearances, among them Liebelt and Schmid [Liebelt and Schmid, 2010] construct

view specific spatial pyramid models for both the object and its parts using real training images, and associate them to 3D space by rendered images of CAD models at the same views; Pepik et.al. [Pepik et al., 2012] extends the deformable part based model [Felzenszwalb et al., 2010] by heuristically initializing part positions and sizes in different views together in the 3D space, so that part view consistency can be employed. While achieving high detection performance, both models use quasi-densely sampled appearance features, resulting in models with high model complexity. While achieving comparable performance in object recognition tasks, our model uses around 3000 feature dimensions, which is about the length of 100 HoG [Dalal and Triggs, 2005] cells in deformable part based model, or 3 cells in spatial pyramids using 1024 codewords as in [Liebelt and Schmid, 2010]. Our learned sparse templates are also very sparse and meaningful, which provides clear interpretations of input data about what are the important features or factors. Besides, as a 3D model, the model complexity does not scale up as the number of modeled views increases.

The proposed learning framework uses AND-OR tree structures, which resembles the And-Or Graph by Zhu and Mumford [Zhu and Mumford, 2006], yet instances in our AND-OR tree do not necessarily represent an object interpretation or parse graph, and the embedded grammar in our tree is context free, which is just a special case of the Grammars defining the And-Or Graph.

The space quantization approach similar to our AoT can be seen as the quad-trees in some image coding methods [Huo and Chen, 2005; Taubman and Marcellin, 2002], where image lattice is recursively decomposed into equal sized sub-lattices. Instead of only allowing this dyadic decomposition, our AoT allows multiple possible decompositions for each volume, thus embeds more expression power than the quad-tree in image lattice or octree in the 3D case. Moreover, the OR nodes in our A-AoT allows explicitly defining equivalence classes of visual concepts, which are shapes subject to geometric transformations. In this way, the appearance of shapes can be categorized

explicitly.

The bottom levels of our AoT are based on the active curves model [Hu et al., 2011], which composes the deformable Gabor filters in active basis model [Wu et al., 2010] into middle level image structures such as line segments and efficiently computes them using dynamic programming. Our statistical model is also consistent with these models, and are further extended on to modeling 3D templates.

4.1.4 Contributions

The contributions of this chapter include:

- We propose an AND-OR Tree structure which quantizes the space of our model and fills in the gap in representation between the stylized shapes and their observations as image pixels.
- We propose to transform the model structure learning problem into a structure search problem in the AND-OR Tree, which can be efficiently solved by dynamic programming.
- We present a new 3D car dataset with labeled image views. Compared with existing 3D car datasets, our dataset features much more widespread views, which provides a new benchmark to test various 3D object category modeling methods.

Using the newly introduced dataset, we show that the proposed approach can learn meaningful 3D car templates with less than 100 shapes, draw boundaries of the object instances on different views, and detect objects and accurately estimate their poses. On the publicly available 3D car dataset [Savarese and Fei-Fei, 2007], we show that our model performs better than [Liebelt and Schmid, 2010] in term of object detection and comparable with it for image view estimation.

The rest of the chapter is organized as follows. Section 4.2 introduces the AND-OR tree design. Section 4.3 explains our information gain criterion and the probabilistic image model. Section 4.4 and 4.5 presents in detail the bottom-up top-down learning algorithm and procedures used in inference algorithm. Our proposed datasets and experiment results are presented in Section 4.6.

4.2 AoT for Space Quantization

As is shown in Fig.4.2, the AoT is composed of three types of nodes: AND nodes, OR nodes and leaf nodes, and their relations as directed edges. We will introduce these nodes and their relations in G-AoT and A-AoT separately.

4.2.1 G-AoT for Part Geometry

AND nodes. As is shown in Fig.4.2, the G-AoT roots from an AND node representing the 3D bounding volume of target car category we want to model. This volume is then decomposed into 12 Volumes of Interests (VoIs), each representing a bounding volume of the components extracted from a 3D CAD model representing the typical shape of the car category. The relative sizes and locations of these components are shown in Fig.4.3(a).

Other than root, AND nodes in the G-AoT represent the volumes of VoIs or their sub-volumes, where examples are shown in Fig.4.4. These nodes are indexed by three sets of parameters, and are denoted as $V_{X,D,C}^A$, where X is the 3D position of the innermost vertex of the volume, D is the dimension and C is a vector parallel to one of the three coordinate axis, denoting the relative offset and the surface normal of a plane. The volume is decomposed into 2 volumes by the plane, which can be expressed as

$$V_{X,D,C}^A \rightarrow V_{X,D_1}^O V_{X+C,D_2}^O, \quad (4.1)$$

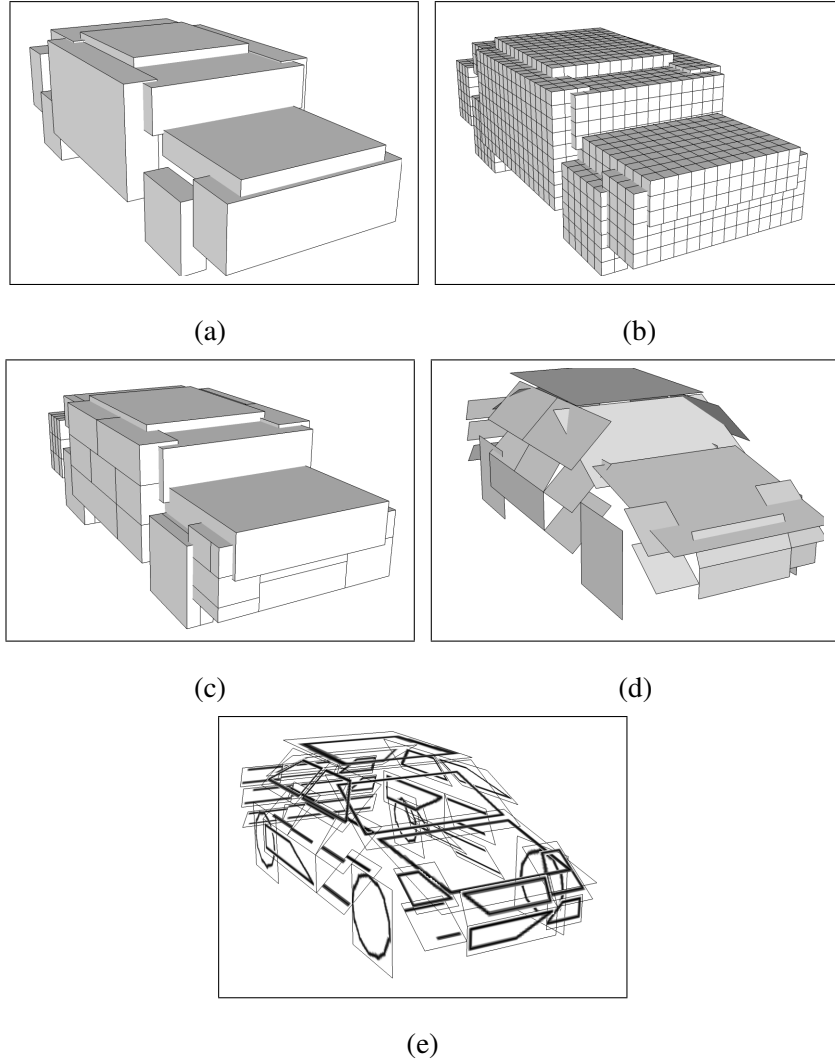


Figure 4.3: **(a)**Initial VoIs. **(b)** VoI partitioning. The size of each VoI is different from that in (a) because of volume size rounding. **(c)** Selected volumes that compose the VoIs **(d)** Selected panels in selected VoIs. **(e)** Selected shape templates which compose the 3D object template.

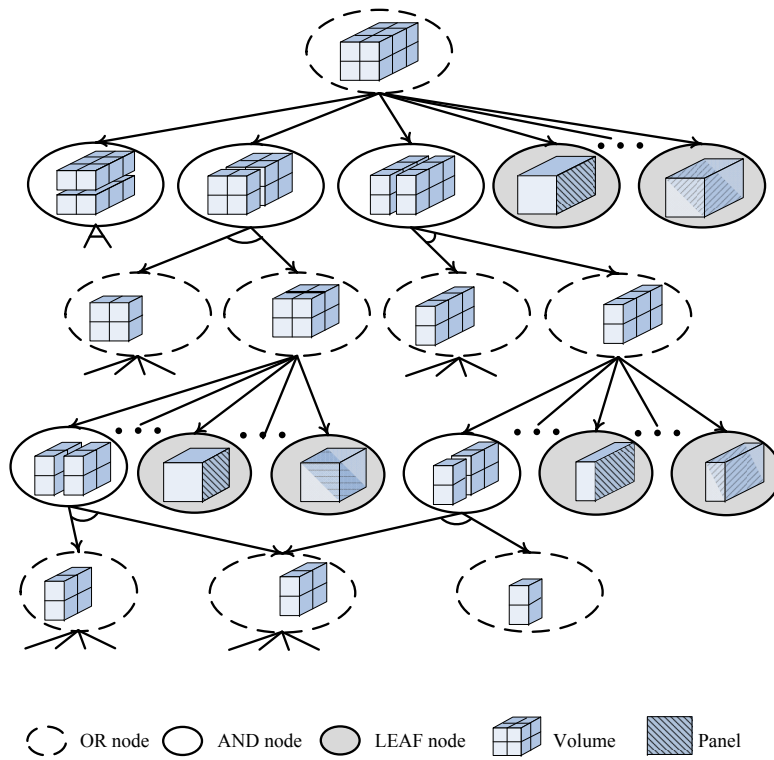


Figure 4.4: A Geometry AND-OR tree, where AND nodes represent combinations of two sub-volumes occupying larger sub-volumes, OR nodes connect to multiple AND nodes representing possible combinations for the same sub-volume, and leaf nodes represent panels inscribing their parent volumes.

where $D_1 + D_2 = D$. As a valid decomposition, C should be in range $0 \prec C \prec D$, where \prec denotes the element-wise inequality.

OR nodes. $V_{X,D}^O$ in Eqn.(4.1) denotes the OR nodes in decompositions, which indicates that there are several alternative decompositions of current volume:

$$V_{X,D}^O \rightarrow V_{X,D,C}^A, 0 \prec C \prec D \quad (4.2)$$

Leaf nodes. Each OR node is also connected to a set of leaf nodes, denoting the options of stop decomposing the current volume and instead put panels on it. The parameters of panels (positions, sizes, and orientations) define the placements of the root nodes of A-AoTs into 3D space. We count panels with the same parameters but connecting to different A-AoTs as different ones, and instantiate two types of panels for each OR node: those on the frontal surface of the volume, and those connecting the top-inner edge and the bottom-outer edge of the volume, which are illustrated in Fig.4.4. The reference point of the positions of inner and outer is the center of the object volume.

4.2.2 A-AoT for Part Appearance

An A-AoT models the appearance of image areas defined by its corresponding panel and image view angles. The OR nodes define hierarchies of stylized visual concepts or visual words, where each one is connected to a set of AND nodes which define its equivalence class. The equivalence class of a visual concept is the concept and its neighbors in its parameter space, and these concepts can be treated as its deformations. The AND node further connects it their children OR nodes, which are stylized constituent concepts. In the following, we define these concepts as layers of visual dictionaries Δ , where the AND nodes are those in even number indexed layers, OR nodes are in those in odd numbered ones, and leaf nodes are in $\Delta^{(0)}$.

$\Delta^{(5)}$: Words in $\Delta^{(5)}$ are defined on the panels in 3D space. The appearance of these


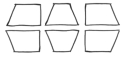




| Template type | Appearance | Parameters | Deformation Range | Instantization |
|----------------------------|---|---|---|---|
| Circles |  | center u, v radius r # of segments n | $\partial u = \{-.1w, 0, .1w\}$ $\partial v = \{-.1h, 0, .1h\}$ | $u = w/2, v = h/2$ $d = \min(w, h)$ $r \in \{.9d, 1d, 1.1d\}$ |
| Trapezoids (rectangles) |  | Parameters of parallel lines $\{L_1, L_2\}$ | above deformation range with in plane rotation: $\{-\pi/\rho, 0, \pi/\rho\}$ | $v_{L_1} = 1/8h, v_{L_2} = 7/8h$ fixed long line length $\iota = 0.9w$ shot line length $\iota \in [.5w, .9w]$ L_3 and L_4 by connecting L_1 and L_2 |
| Parallel Lines |  | same as above | same as above | same as above with no L_3, L_4 |
| Line Segments |  | $L = \{\text{center } (u, v),$ orientation $\theta,$ length $\iota\}$ | deformation realized in Active Curves | $u = w/2, v = h/2$ $\theta = 0$ $\iota = 0.9w$ |
| Active Curves |  | $l = \{\text{center } (x, y),$ orientation $o,$ length $l\}$ | $\partial x = \{-1, 0, 1\}$ pixels $\partial y = \{-1, 0, 1\}$ pixels $\partial o = \{-\pi/\rho, 0, \pi/\rho\}$ | $(x, y) \in \text{image lattice } \Lambda$ $o \in \mathcal{O}$ $l \in \{1, 2, \dots, 5\}$ |
| Active Basis |  | $B = \{\text{center } x, y$ orientation o scale $s\}$ | same as above | $(x, y) \in \Lambda$ $o \in \mathcal{O}$ filter size set to 17×17 |

Table 4.1: List of visual concepts used in our representation, their parameters, deformation range and instantiation range. w and h in column 4 and 5 denote the width and height of the panel respectively.

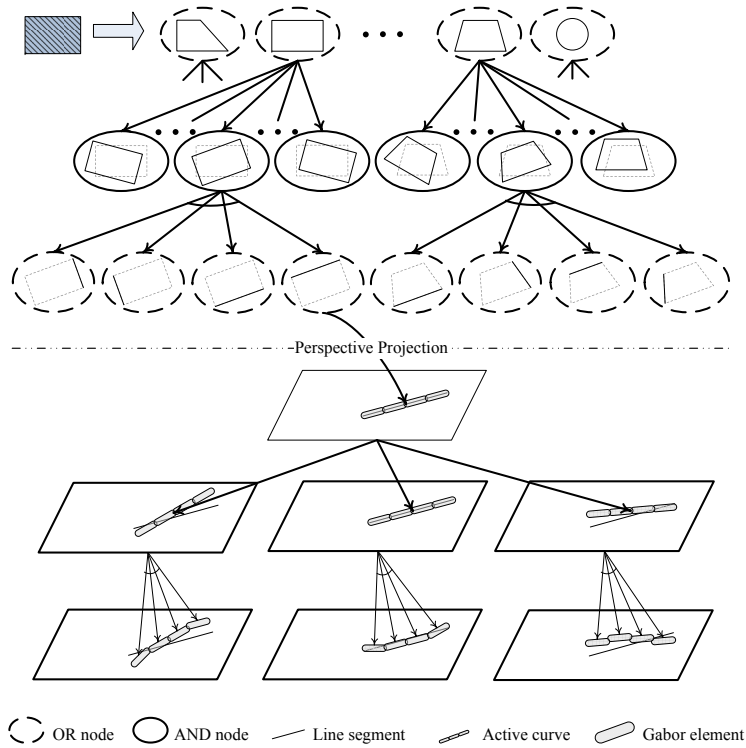


Figure 4.5: In Appearance AND-OR Trees, each panel represents geometry of a shape template, and are connected to AND-OR Tree for part appearance. Here AND corresponds to composition and OR corresponds to deformation. It extends the AoT for Geometry to image spaces since its leaf nodes are Gabor filters.

words are templates of regularized shapes, which we call shape templates S . As is shown in Table.4.1, these shapes include circle, trapezoids with rectangles as a special case, parallel lines and line segments, whereas the trapezoids and parallel lines further contain 6 sub-types separately.

$\Delta^{(4)}$: In 3D space, we allow each S to translate along the two axis of the panel sides and rotate in the panel plane, each at three levels specified in column 4 of Table.4.1. By combining these transforms, we can derive a set of 27 neighboring shape templates S for each stylized one, which constructs its equivalence class. As an example, a rotated and translated trapezoid is shown in Fig.4.6. Combining the equivalence classes for all

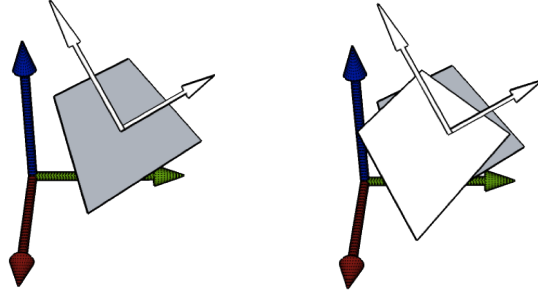


Figure 4.6: An example of the 3D deformation for shape templates. We allow the template to rotate round panel center and translate along the axis direction.

stylized shapes,

$$\Delta^{(4)} = \{S | S \in \partial S_0, S_0 \in \Delta^{(5)}\} \quad (4.3)$$

where ∂S_0 denotes the equivalent class of stylized S_0 .

$\Delta^{(3)}$: $\Delta^{(3)}$ is defined both on panels and on image planes as $\Delta_{3D}^{(3)}$ and $\Delta_{2D}^{(3)}$, which are the line segments L as elements composing shape templates in $\Delta^{(4)}$, and their realizations on images as a subset of active curves A .

For the convenience of parameterizing trapezoid shapes S , line segments L are parameterized in coordinates of panel plane, with the origin at the center of the panel, and two axis along the side direction of panels. In this coordinate system, a line segment is parameterized by (u, v, θ, ι) , which are center position, line orientation and length respectively. Trapezoid shape templates can then be easily decomposed and parameterized by those of constituent parallel line segments. For example, as a special case of trapezoid templates, a rectangular shape template with width and height as (w, h) can be decomposed into:

$$S \rightarrow L_{0, -\frac{h}{2}, 0, w} L_{0, \frac{h}{2}, 0, w} L_{-\frac{w}{2}, 0, \frac{\pi}{2}, h} L_{\frac{w}{2}, 0, \frac{\pi}{2}, h}, \quad (4.4)$$

and indexed by $(0, -\frac{h}{2}, 0, w, 0, \frac{h}{2}, 0, w)$.

Thus in 3D space:

$$\Delta_{3D}^{(3)} = \{L | L \in {}^*S, S \in \Delta^{(4)}\}, \quad (4.5)$$

where *S denotes the set of line segments by applying the rules of decomposition.

On image plane, these line segments are realized by a subset of active curves indexed by their parameters as $A_{x,y,o,l}$, where (x, y, o, l) are for the center position, orientation and length. Thus,

$$\Delta_{2D}^{(3)} = \left\{ A \mid A = \mathcal{P}(L, \omega), L \in \Delta_{3D}^{(3)}, \omega \in \Omega \right\} \quad (4.6)$$

where ω denotes a view in the set of views Ω , and \mathcal{P} denotes the projection function that projects L to A .

$\Delta^{(2)}$: Elements in $\Delta^{(2)}$ are equivalent active curves of those in $\Delta_{2D}^{(3)}$. For an active curve $A_{x,y,o,l}$, we allow it to translate in the range $\partial x, \partial y$, and rotate in a small orientation range ∂o , thus the equivalence classe $\partial A_{x,y,o,l}$ can be defined as

$$\partial A_{x,y,o,l} = \left\{ A_{x',y',o',l} \left| \begin{array}{ll} x' = x + \delta x \cos o', & \delta x \in \partial x \\ y' = y + \delta y \sin o', & \delta y \in \partial y \\ o' = o + \delta o, & \delta o \in \partial o \end{array} \right. \right\}, \quad (4.7)$$

and

$$\Delta^{(2)} = \{ \partial A_{x,y,o,l} \mid A_{x,y,o,l} \in \Delta_{2D}^{(3)} \} \quad (4.8)$$

$\Delta^{(1)}$: As in [Hu et al., 2011], an active curve can be decomposed into a series of weakly overlapping active basis B , which are placed along the curve, and are parameterized by position (x, y) and orientation o . As a special case, the active curves used in this chapter are active basis or equivalently the 2D primitives placed along straight line segments, and can be decomposed as:

$$A_{x,y,o,l} \rightarrow B_{x,y,o} B_{x_1^+, y_1^+, o} B_{x_1^-, y_1^-, o} \cdots B_{x_l^-, y_l^-, o} \quad (4.9)$$

In Eqn.(4.9), $x_i^\pm = x_{i-1}^\pm \pm 0.9b \cos o$ and $y_i^\pm = y_{i-1}^\pm \pm 0.9b \sin o$, and b is the length of active basis elements B in pixels. Note that the length of the A is measured by the number of used active basis, which can be converted to pixel units using the length of B .

Therefore $\Delta^{(1)}$ is the dictionary of active basis:

$$\Delta^{(1)} = \{B | B \in {}^*A, A \in \Delta^{(2)}\} \quad (4.10)$$

$\Delta^{(0)}$: Visual elements composing the leaf nodes of A-AoT are Gabor filters at specific locations (x, y) and orientations o , these filters are translated and rotated versions of Gabor function $G(x, y) \approx \exp\{-[(x/\sigma_x)^2 + y/(\sigma_y)^2]\}e^{ix}$ with $\sigma_x = 5$, $\sigma_y = 10$, which are further normalized to have zero mean and unit ℓ_2 norm.

Similar to the active curves, the active basis B in $\Delta^{(1)}$ are defined as Gabor filters deformable in ranges ∂x , ∂y and ∂o :

$$\partial B_{x,y,o} = \left\{ B_{x',y',o'} \left| \begin{array}{l} x' = x + \delta x \cos o', \quad \delta x \in \partial x \\ y' = y + \delta y \sin o', \quad \delta y \in \partial y \\ o' = \theta + \delta o, \quad \delta o \in \partial o \end{array} \right. \right\} \quad (4.11)$$

4.2.3 Instantiation of AoT

We instantiate the G-AoT on a set of permissible volumes. For each VoI, we round its size to multiples of a unit size, and put an equally unit size spaced grid Σ onto it, such as shown in Fig.4.3(b). When generating the AND and OR nodes using rules in Eqn.(4.1) and (4.2), we start from the rounded VoIs, and restrict C such that generated sub-volumes must have all vertices on the grid Σ .

Recursively applying the rules generates layers of AND, OR nodes and their relations as edges. The recursion stops when the volume reaches a size limit. The full G-AoT can then be instantiated by further adding leaf nodes and corresponding edges.

The leaf nodes of G-AoT are instantiated together with the root nodes of A-AoTs.

For each of the two panel types of an OR node, we instantiate a number of shape template nodes according to the panel size and the specification in column 5 of Table.4.1.

Using these shape templates S as root nodes, the A-AoTs can be recursively generated using definitions of dictionaries in Section 4.2.2. Although we use a discrete view set to define Δ_{2D}^3 in Eqn.4.6, we do not need to enumerate these views and can still use continuous view space in practice, as the view ω of image is assumed to be given in the training stage, and is hypothesized before projecting L in the testing process.

Even given the ω , the number of nodes in $\Delta^{(2)}$ to $\Delta^{(0)}$ would still be numerous. We choose to densely instantiate these nodes over the image lattice Λ at a set of ρ discrete orientations $O = \{\pi/\rho, \pi/\rho, \dots, \pi\}$ and lengths. In this way, we can avoid storing parameters for these nodes as they are encoded as the order where nodes are stored in matrix. When referenced by parent nodes, they can be easily accessed using their parameters.

4.2.4 Parse Trees as Samples of the Quantized Space

The proposed AoT quantizes the infinite continuous space we want to model into a structured discrete space, where each point corresponds to a view ω and a parse tree pt .

A parse tree pt is a sub-tree of the AoT. Given a view ω , a parse tree can be generated starting from the root node of AoT, by recursively tracing child OR nodes for AND nodes and keeping only one child node (either AND node or leaf node) at traced OR nodes. Therefore, a parse tree represents a possible partition of the object bounding volume, regularized shape templates in these composing volumes and a specific deformation for each of the shape templates for view ω . As an illustration, the partition, panels, and the regularized shape templates for a learned parse tree are shown Fig.4.3 (c) to (e).

4.3 Probabilistic Image Model

In this section, we will first define that how the conditional probability of image \mathbf{I} given view ω can be decomposed along the nodes in a parse tree gt . We then introduce how we model the joint probability $p(\mathbf{I}, \omega)$, and its corresponding information gain.

4.3.1 Probability Density Decomposition

Through the parse tree, the target object volume is first decomposed by layers of AND and OR nodes in G-AoT. In an OR node, the conditional probability of $p(\mathbf{I}_{\Lambda_{V^O}} | \omega, V^O)$ is equal to that of the traced child node:

$$p(\mathbf{I}_{\Lambda_{V^O}} | \omega, V^O) = p[\mathbf{I}_{\Lambda_V} | \omega, V \in ch(V^O, pt)], \quad (4.12)$$

where $\mathbf{I}_{\Lambda_{V^O}}$ represents the part of image explained by V^O , and function $ch(V^O, pt)$ retrieves the traced immediate child node of V^O in the parse tree pt , which could be either an AND node or a leaf node in G-AoT.

In an AND node, its conditional probability is decomposed into the product of that of its immediate two child nodes:

$$p(\mathbf{I}_{\Lambda_{V^A}} | \omega, V^A) = \prod_{V^O \in ch(V^A, pt)} p(\mathbf{I}_{\Lambda_{V^O}} | \omega, V^O). \quad (4.13)$$

In Eqn.(4.13), we assume the areas of the image covered by the child nodes do not overlap. This assumption will not hold if the two children OR nodes are its decomposition along the depth direction. In this case, we enforce this assumption by allowing only one volume to be active.

The leaf nodes of G-AoT defines a set of shape templates $\{S_i\}$. To model the effect of object self-occlusion, we assume each S_i has a visible range, and can only be projected onto image when it is visible. In practice, we assume a shape template S_i is visible if the inner product between the panel surface normal and view direction is smaller than -0.6.

If the S_i is visible, it can be further decomposed along the parse tree as:

$$p(\mathbf{I}_{\Lambda S_i} | \omega, S_i) = \prod_j p(\mathbf{I}_{\Lambda L_j} | \omega, L_j) \quad (4.14)$$

$$= \prod_{i,j} p(\mathbf{I}_{\Lambda A_{i,j}} | A_j) = \prod_{i,j,k} p(\mathbf{I}_{\Lambda B_{i,j,k}} | B_{i,j,k}) \quad (4.15)$$

where A_i is the corresponding active curve by projecting L_i in view ω . As we assume the probability for OR nodes is equal to that of their immediate child AND nodes or leaf nodes in pt , the node types are not differentiated in Eqn.(4.14) to (4.15).

Cascading decompositions above, the conditional probability of pt can be factorized into that of its constituting active basis elements in the leaf nodes:

$$\begin{aligned} p(\mathbf{I} | \omega, pt) &= p(\mathbf{I}_{\bar{\Lambda}_{pt}}, \mathbf{I}_{\Lambda_{pt}} | \omega, pt) \\ &= p(\mathbf{I}_{\bar{\Lambda}_{pt}} | \omega, \mathbf{I}_{\Lambda_{pt}}) p(\mathbf{I}_{\Lambda_{pt}} | \omega, pt) \\ &= p(\mathbf{I}_{\bar{\Lambda}_{pt}} | \omega, \mathbf{I}_{\Lambda_{pt}}) \prod_{ijk} p(\mathbf{I}_{\Lambda B_{ijk}} | B_{ijk}) \end{aligned} \quad (4.16)$$

where $\bar{\Lambda}_{pt}$ refers to pixels not covered by pt , i indexes over visible shape templates under view ω . The terms of invisible shape templates are absorbed to $(\mathbf{I}_{\bar{\Lambda}_{pt}} | \omega, \mathbf{I}_{\Lambda_{pt}})$ as they do not explain any image pixels.

4.3.2 Image Modeling by Density Substitution

We model our target distribution $p(\mathbf{I}, \omega)$ starting from a reference distribution $q(\mathbf{I}, \omega)$ using density substitution. This method was first proposed in [Friedman, 1987], and was used in image modeling by active basis model [Wu et al., 2010]. As the explained portion of image $\mathbf{I}_{\Lambda_{pt}}$ is modeled exactly as a set of active basis, the details and proof of density substitution used here can be derived in the same way as [Wu et al., 2010], and are thus not repeated.

Using density substitution, we get:

$$\begin{aligned}
p(\mathbf{I}, \omega | pt) &= q(\mathbf{I}, \omega) \frac{\prod_{ijk} p(\mathbf{I}_{\Lambda_{B_{ijk}}} | B_{ijk})}{\prod_{ijk} q(\mathbf{I}_{\Lambda_{B_{ijk}}})} \\
&= q(\mathbf{I}, \omega) \prod_{ijk} \frac{p[h(r_{ijk})]}{q[h(r_{ijk})]}.
\end{aligned} \tag{4.17}$$

In Eqn.(4.17), r is the response of a Gabor filter, defined as the squared inner-product of the base function B in $\Delta^{(0)}$ with image \mathbf{I} : $r = \| \langle \mathbf{I}, B \rangle \|^2$. h is the sigmoid transform that saturates the large Gabor filter response to τ : $h(x) = \tau [2/(1 + e^{-2x/\tau}) - 1]$.

For modeling $p[h(r)]$, we assume the following exponential model:

$$p[h(r)] = \frac{1}{Z} q[h(r)] \exp[\lambda h(r)] \tag{4.18}$$

where λ is the parameter and Z is the corresponding normalizing constant. The above model can be justified by maximum entropy principle [Pietra et al., 1997]. The $q[h(r)]$ can be represented as a histogram pooled over a set of natural images.

In the original active basis model, the parameter λ reflects the importance of the corresponding active basis element in the learned template, and is estimated so that the expectation $E_p[h(r)]$ matches the corresponding mean observed from training images. As we assume that all the basis elements in our template are equally important, the λ is simply set to a constant corresponding to a high expected response. Z is then estimated using the same method as in active basis.

4.4 Template Learning by Dynamic Programming

We assume images of a visual object category should share the same geometry and stylized part appearances. Thus we want to model an image category by a parse tree set pt in which all pt share the same set of stylized shape template $\{S_i\}$. In the full AoT,

this \mathbf{pt} corresponds to a parse tree in G-AoT, and visually, they form a deformable 3D object template.

4.4.1 Information Gain as the Objective for AND-OR Search

We assume images of a visual object category should share the same geometry and stylized part appearances. Thus we want to model an image category by a parse tree set \mathbf{pt} in which all pt share the same set of stylized shape template $\{S_i\}$. In the full AoT, this \mathbf{pt} corresponds to a parse tree in G-AoT, and visually, they form a deformable 3D object template.

4.4.2 Information Gain as the Objective for AND-OR Search

Given M view labeled training images $\{\mathbf{I}_m, \omega_m\}_{m=1}^M$, these parse trees \mathbf{pt} can be learned by maximizing the corresponding image likelihood, or equivalently maximizing the information gain of corresponding model $p(\mathbf{I}, \omega | \mathbf{pt})$ over reference model $q(\mathbf{I}, \omega)$:

$$\text{IG}(\mathbf{pt}) = \iint p(\mathbf{I}, \omega | \mathbf{pt}) \log \frac{p(\mathbf{I}, \omega | \mathbf{pt})}{q(\mathbf{I}, \omega)} d\mathbf{I}d\omega \quad (4.19)$$

$$\approx \sum_{m=1}^M \log \frac{p(\mathbf{I}_m, \omega_m | \mathbf{pt})}{q(\mathbf{I}_m, \omega_m)} \quad (4.20)$$

Along the G-AoT, the best \mathbf{pt} can be learned recursively, as we can get recursions among AND and OR node using Eqn.(4.12) and (4.13):

$$\max \text{IG}(V^A) = \max \sum_{V^O \in \text{ch}(V^A, \text{AoT})} \text{IG}(V^O) \quad (4.21)$$

$$= \sum_{V^O \in \text{ch}(V^A, \text{AoT})} \max \text{IG}(V^O) \quad (4.22)$$

$$= \sum_{V^O \in \text{ch}(V^A, \text{AoT})} \max_{V^{A'} \in \text{ch}(V^O, \text{AoT})} \max \text{IG}(V^{A'}) \quad (4.23)$$

and

$$\max IG(V^O) = \max_{V^A \in ch(V^O, AoT)} \max IG(V^A) \quad (4.24)$$

$$= \max_{V^A \in ch(V^O, AoT)} \max_{V^{O'} \in ch(V^A, AoT)} \sum IG(V^{O'}) \quad (4.25)$$

$$= \max_{V^A \in ch(V^O, AoT)} \sum_{V^{O'} \in ch(V^A, AoT)} \max IG(V^{O'}) \quad (4.26)$$

where IG refers to the information gain.

The recursion above will ultimately reach leaf nodes of G-AoT, and for the stylized shape template S_i in the leaf node,

$$\max IG(S_i) \approx \max \sum_m \log \frac{p(\mathbf{I}_m, \omega_m | S_i)}{q(\mathbf{I}_m, \omega_m)} \quad (4.27)$$

$$\approx \sum_m \max \log \frac{p(\mathbf{I}_m, \omega_m | S_i)}{q(\mathbf{I}_m, \omega_m)} \quad (4.28)$$

$$= \sum_m \max_{i,j,k} \sum_{i,j,k} \lambda h(r_{mijk}) - \log Z \quad (4.29)$$

So the recursion could continue in A-AoTs, which maximize the log-likelihood ratio of each image \mathbf{I}_m given the visual concepts and their equivalence classes in the hierarchy.

The recursion above actually reduces to dynamic programming, as it always decomposes a problem into a few similar sub-problems, and results of sub-problems are reused because each node in AoT would have multiple parents. In the following, we call this dynamic programming on the AND-OR Tree as AND-OR search.

4.4.3 AND-OR Search Algorithm

Converting the recursions into iterations, the AND-OR Search algorithm is composed of one bottom-up pass and one top-down pass, which are listed in Algorithm 4.1 and 4.1 respectively. The bottom-up pass is composed of layers of sum and max operations at AND and OR nodes respectively, which computes the maximum information gain of

Algorithm 4.1: The Bottom-Up Pass of AND-OR Search Algorithm

Input: $h[i]$, $t[i]$, $ch[i]$ as height, type, children index of node i .

Output: The learned template T

```
1 for  $m = 1$  to  $M$  do
2   | Compute S1, M1, S2, M2 maps for image  $I_m$ 
3 end
4 Node layer  $l \leftarrow 4$ 
5 foreach node  $i$  in layer  $l$  do
6   | if  $t[i] = \text{"ROOT"}$  then                                Root node
7     | return GET-TEMPLATE ( $i$ )
8   | else if  $l=4$  then                                       Deformed shape template
9     |  $IG_{im} \leftarrow \sum_{L_j \in S_i} M2(I_m, A_j)$ 
10  | else                                                         ther AND nodes
11  |  $IG_i \leftarrow \sum_{j \in ch(i)} IG_j$ 
12  | end
13 end
14 foreach node  $i$  in layer  $l + 1$  do
15  | if  $l=4$  then                                             Shape template
16  |  $IG_i \leftarrow \sum_{j \in ch(i)} \max_m IG_{jm}$ 
17  | else                                                         Other OR nodes
18  |  $IG_i \leftarrow \max_{j \in ch(i)} IG_j$ 
19  | end
20 end
21  $l \leftarrow l + 2$ , go to line 5
```

Algorithm 4.2: The Top-Down Pass of AND-OR Search Algorithm

```
1 Function GET-TEMPLATE ( $i$ )
2    $T \leftarrow \emptyset$ ;
3   if  $t[i] = \text{"OR"}$  then
4      $j = \arg \max_{k \in ch(i)} IG_k$ ;
5      $T \leftarrow \text{GET-TEMPLATE}(j)$ ;
6   else if  $t[i] = \text{"AND"}$  then
7     foreach  $j \in ch(i)$  do
8        $T \leftarrow \{T, \text{GET-TEMPLATE}(j)\}$ ;
9   else Leaf node of G-AoT
10     $T \leftarrow \text{node } i$ ;
11  return  $T$ ;
```

all the possible pt or the 3D object templates. The top-down pass is a series of arg-max operations that retrieve the optimal composition at each activated OR node.

The bottom-up pass starts from computing the log-likelihood ratios of active curves and active basis in $\Delta^{(0)}$ to $\Delta^{(3)}$, which are saved in the form of score maps for retrieval by L in S . Details of these sum-max operations can be found in Chapter 3 as the steps of computing **S1**, **M1**, **S2**, **M2** maps, thus are not repeated here.

In implementing the AND-OR search algorithm, we need to decide the visiting order of these nodes so that child nodes are processed before their parents. For G-AoT, we simply assign the height of AND and leaf nodes to be $2v$ and that of OR nodes to be $2v + 1$, where v is the size of the volume. The height of each node in A-AoT is assigned to the layer index of the vocabulary they belong to. As operations within each layer of the tree can be computed independently, they can be done in parallel, which makes the algorithm more efficient.

For the special case of AND composition in depth order, we simply use max instead of sum over the two information gains in bottom-up pass, and only retrieve the maximum node in the top-down pass.

For specific VoIs, we further learn alternative part template combinations, in order to model large structural variations in various datasets. We use the hard EM clustering framework to alternatively impute image cluster labels and learn shape template combinations for each image cluster iteratively. In the following experiments, we learn two clusters for the head, tail and each of the four wheel VoIs.

4.5 Inference Scheme

4.5.1 Converting 3D Template to 2D Templates

Given a specific view, the 3D deformable template can be projected to a 2D deformable template, with 3D in-plane deformation of each shape template realized by 27 different image templates. After projection, the sliding window method can be employed to perform object detection in that view. In each window, the inference problem is still solved by dynamic programming algorithm, which essentially finds the maximum likelihood ratio on a small AoT with root connecting to all the activated shape templates. Alternative shape template combinations are also treated as deformations: we simply project both cases to a specific view, and the one with highest likelihood ratio prevails.

As is shown in Fig.4.9, we enumerate discrete views in the view sphere, and use the approach above to perform object detection in each view. To generate discrete views, we fix the internal camera parameters by assuming a general focal length, and discretize the external parameter space of pan, tilt, and camera distance to the world origin. For simplicity, roll angle of the camera is set to zero.

4.5.2 Feature Weight Adjustment

The proposed learning method builds a sparse object model, which in the view of discriminative learning, provides features for object recognition. However, to achieve high recognition performance on image datasets, the reference distribution $q[h(r)]$ should be re-calibrated to compensate the error introduced by the position and orientation invariant assumption. This leads to the adjusted weights on the scores of active curves. To this aim, we lump the scores of all line segments on the learned template into a feature vector, and use linear SVM to re-train the weights of these features.

We collect negative training examples for the SVM reweighting in the way similar to the hard negative mining steps in deformable part based model [Felzenszwalb et al., 2010]. Specifically, we use the equally weighted original model to run sliding windows on positive examples, crop high score windows whose intersection over union with true positive window less than 50 percent, and extract line segment scores as negative examples.

We use LIBSVM [Chang and Lin, 2011] as the implementation of this adjustment procedure.

4.5.3 Hypothesis Verification by Color Histogram

In experiments, we found that templates only using sketches tend to generate false positives on textured areas, such as images of fences or brick walls. We use color information to further suppress these false positives, by re-testing on high score windows using both sketch and color information.

To this aim, we evenly sample patches in each part template, concatenate their color histograms into the feature vector, and also feed it into the linear SVM. We allocate 8 bins on each of the 3 color channels, so that for each part template another 24 dimensions are added to the feature vector. Note that the color and sketch features are

concatenated into one vector and their weights are trained together.

After computing the score maps for the sketch-only templates at enumerated views, we sequentially select the top 200 highest score windows as highly suspicious windows in each view, and update the score of these windows using both sketch and color histograms. The object detection windows are then reported using non-maximum suppression through these highly suspicious windows by the criterion that reported windows should not overlap more than 50 percent with each other.

4.6 Experiments

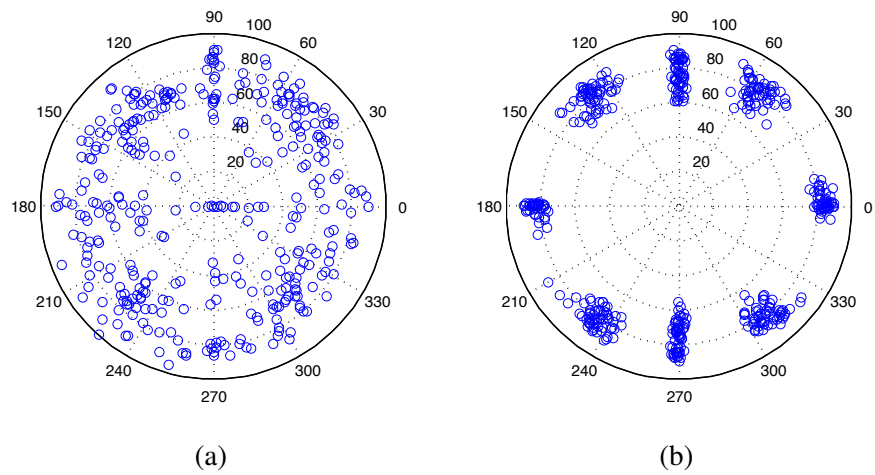
4.6.1 Image Dataset and Parameters

There are some widely used datasets emphasizing 3D object recognition [Leibe and Schiele, 2003; Savarese and Fei-Fei, 2007], but most of them only provide images from a few specified views or limited ranges of views. This could potentially trivialize the problem of 3D object recognition, as models may simply memorize the object appearance in these views, thus essentially cast the problem into a multi-class object recognition problem.

We introduce a new dataset¹ of car images, featuring a large variety of views, which are collected uncontrolled both from Internet and at the intersections and parking lots on UCLA campus, such as the ones in Fig.4.1. For each image, we label object view by extending the annotation software provided on the project page of [Hu and Zhu, 2010]. We also labeled views for all images of the car dataset in [Savarese and Fei-Fei, 2007], and show both of them in Fig.4.7.

In the following experiments, we randomly use 160 of the 360 images dataset as training data, and the rest as testing data.

¹Available at: <http://www.stat.ucla.edu/~wzhu/CVPR12>



(c)

Figure 4.7: **Top:** View distribution of our dataset (a) and the 3D car dataset (b) in [Savarese and Fei-Fei, 2007]. The angular direction represents pan angle and radius direction represents tilt angle. **Bottom:** Sample images of our dataset.

For parameters in G-AoT, we set the minimum volume size to $2 \times 2 \times 1$ in unit size, where 1 is along the depth direction, and unit size is set to 150 mm.

For parameters in A-AoT, we set the Gabor filter size to $b = 17$ in pixels, and basis response saturation threshold τ to 6, which is the same as in [Wu et al., 2010]. The range of λ in [Wu et al., 2010] is in the range $[0, 5]$, with 5 corresponding to the highest possible expected response, and we set it to 2.0 which corresponds to about 90% of the highest. Our experiments show that the learned template does not change much when λ is in range $[2.0, 3.0]$.

4.6.2 Scales of AoT as a Function of Volume Size

| Volume Size | V^A | V^O | leaf | # of pt |
|--------------|------------|---------|------------|--------------------------|
| (5, 5, 5) | 1,170 | 270 | 10,206 | $2,7969 \times 10^6$ |
| (8, 8, 8) | 99,684 | 11,340 | 394,065 | 7.2096×10^{25} |
| (10, 10, 10) | 628,540 | 55,440 | 1,967,328 | 2.2061×10^{48} |
| (15, 15, 15) | 14,517,360 | 851,760 | 33,043,920 | 6.6772×10^{153} |

Table 4.2: The scale of AoT and the number of possible 3D deformable templates as bounding volume size grows.

In the first experiment, we want to show how the size of AoT and the number of possible **pt** grows as the volume size grows. For this purpose, we set the entire object volume as one VoI, and grow G-AoT using the rules in Section 4.2.1 and parameters specified above. For each generated tree, we count the number of AND, OR and leaf nodes in G-AoT. We also compute the number of possible 3D deformable object templates(# of **pt**) for each AoT. This can be computed using the bottom-up pass of the AND-OR search, with IG replaced by the number of sub-compositions, and sum-max operations replaced by product-sum operations for AND and OR nodes respectively.

The results are shown in Table.4.2. It can be seen that the number of possible 3D templates is much larger than the number of nodes combined at all volume sizes, and the number of 3D templates grows much faster than that of the nodes. This shows the efficiency of AND-OR structure in encoding compositions.

4.6.3 Representation Power of AoT and Octree

To test the expression power of our AoT with the octree in 3D space, we compare the distortion of learned 3D templates from AoT and octree at different granularities and for different vehicle types. Since most of the 3D car image datasets only have images of sedans, we choose to directly use 3D CAD models as both training data and the ground truth for evaluating the goodness of the learned model.

We collected 5 3D CAD models for each of the following vehicle categories: sedan, SUV, truck and minivan from Google 3D warehouse. For a given tree structure, we learn the best 3D panel composition that represents the VoIs using the AND-OR search specified above, with the information gain of each template replaced by the total area of facets within a certain distance threshold from the corresponding panel. For simplicity, we define the distance between the panel and the a facet as the average distance between the panel and facet vertices, and the distance threshold is set to 1 inch.

We set the volume size limit to $1 \times 1 \times 1$ of unit size, generate AoTs and octrees with different unit sizes, and compute the maximum within threshold areas for each of the learned panel composition. We compare the results from AoTs and octrees by the curve of the proportions of within threshold areas v.s. unit sizes, which are shown in Fig.4.8. It is clear to see that in all cases, the AoT explains more area than the octree, which justifies our qualitative analysis in literature review.

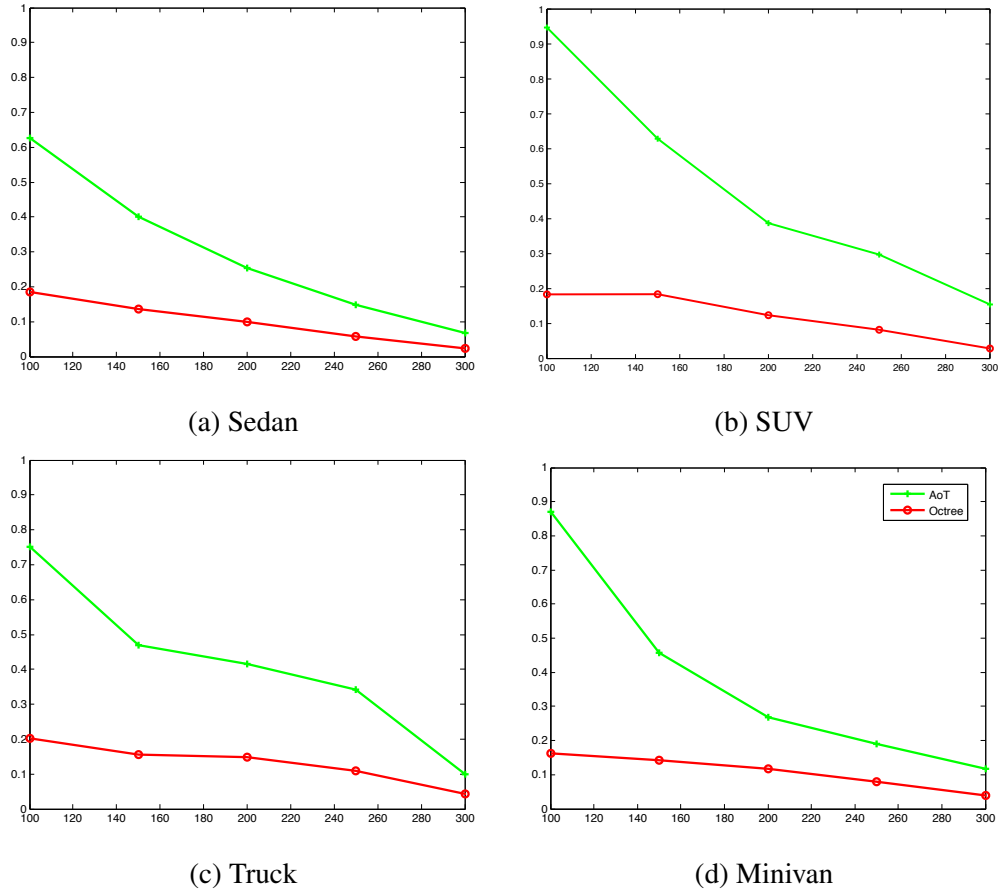


Figure 4.8: Comparison of representation power between AND-OR tree and commonly used Octree. The horizontal axis represents the side length of minimum volumes in millimeters, and the vertical axis represents the percentage of CAD model surfaces inside VOI and represented by the computed solution. The AND-OR tree consistently outperforms Octree across all the 4 vehicle categories at all granularity.

4.6.4 Learning Object Templates

Fig.4.1 shows the learned template for car images. From the template, we can clearly interpret some shape templates as wheels, windows. Even some detailed parts such as headlights and grills can also be recognized. This is benefited by that fact that the appearance of the proposed shape templates is composed of large and regularized shapes. Plus, the combination of these individual shape templates forms a car shape, which demonstrates that 3D templates represented by AND-OR Tree include meaningful ones, and they can be searched through by the proposed algorithm. Deformed templates also demonstrate that the proposed hierarchies of deformations can adapt the regularized shapes to its variants observed in images.

4.6.5 Object Recognition Experiments

4.6.5.1 Dataset 1: our own dataset

On our newly collected dataset, we run the inference steps in Section 4.5 to perform object detection. We search pan angle at 15° interval in $[0^\circ, 360^\circ)$, tilt angle at 5° interval from $[5^\circ, 90^\circ)$, and 8 camera distances for each pan and tilt angle combination.

Fig.4.9 shows an example of the projected 2D templates and their score maps at the detected and its neighboring views. It can be seen that while all three views have high scores at the object location, the peak at the correct view has significantly higher score than the others. As we further decompose the templates into VoIs and individual shape templates, we find that the meaningful parts, such as the windshield and wheels, are also of very high saliency in their score maps.

We show the object detection performance by precision recall curves as shown in Fig.4.10, where windows with intersection over union area ratio greater than 0.75 are considered positive detection. Specifically, we show the performance of our model using and without using the color features mentioned above. From the curves, we can

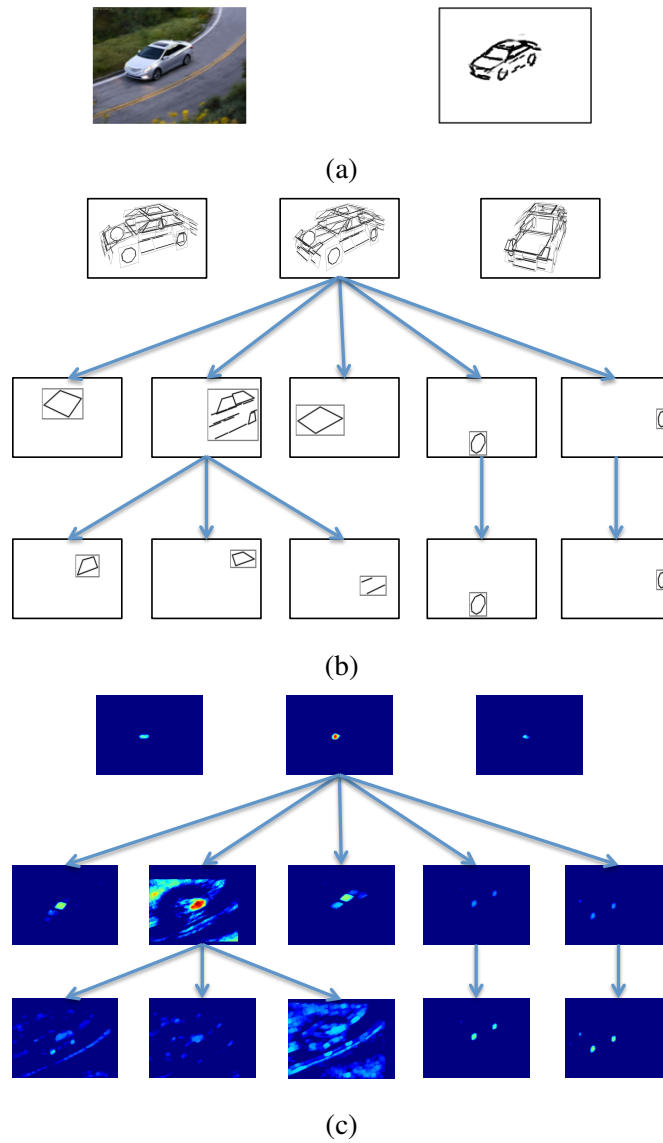


Figure 4.9: Projection of 3D Template to 2D Templates, their decomposition, and corresponding detection score maps. **Left:** In inference process, we first project the learned 3D templates into a few view specific 2D templates. For each 2D template, it can be decomposed into VoI templates, and further into the shape templates. **Right:** Score maps for different 2D templates and its elements, each corresponds to the node on the left. Note that score maps are normalized such that intensities are only comparable on score maps of the same row.

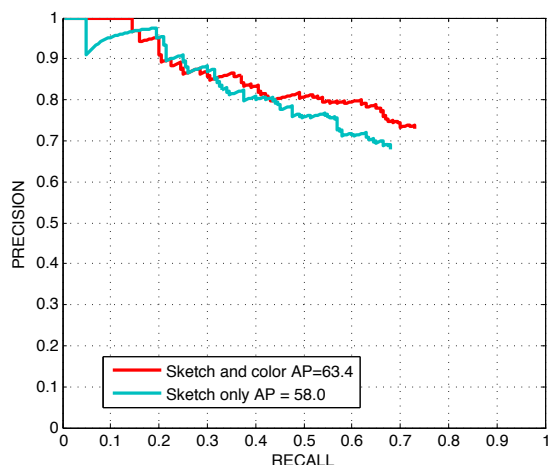


Figure 4.10: Object detection performance on our proposed dataset

see that adding color feature help the object recognition performance.

For correctly detected instances, we also plot the histogram of view estimation errors on pan angles, which are shown on Fig.4.11. From the plot, we can see that majority of the instances are detected at the correct angles. We notice that a few of estimates are totally flipped from head to tail, this suggests we should model more details of the head and tails at higher resolutions, as the general shape of cars at flipping views are similar.

4.6.5.2 Dataset 2: the 3D car dataset in [Savarese and Fei-Fei, 2007]

We also tried our method on the 3D car dataset in [Savarese and Fei-Fei, 2007]. We use the learned model from the experiment above, and retrain feature weights using training images in this dataset. Object detection performance is evaluated as precision recall curves, which are shown in Fig.4.12, together with the rest of curves from [Liebelt and Schmid, 2010].

We also show the performance of the pose estimation task using the confusion matrix, together with that from [Liebelt and Schmid, 2010] in Fig.4.13. From the results,

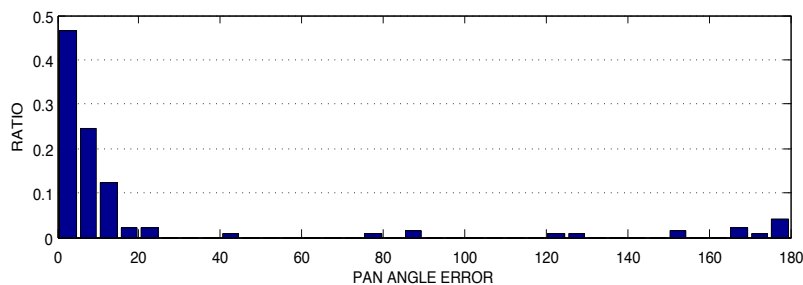


Figure 4.11: Pose estimation error on our newly collected dataset.

we can see that our model achieves higher performance in terms of object detection and comparable performance in pose estimation. Note that according to the evaluation criteria in [Sun et al., 2009], only correctly detected samples in the testing set are accounted into the confusion matrix. Our model achieved higher detection rate, so more images are accounted into the confusion matrix.

By comparing the confusion matrices, we also find that accuracies for different poses are not as uniform as that in [Liebelt and Schmid, 2010]. We believe this is because: 1.) our feature weights are shared across views and 2.) in the re-weighting step, the linear-SVM only optimizes class labeling errors, regardless of the pose estimation performance. With our continuous view formulation, we believe a structured-SVM that optimize both class label and view should eliminate this problem. This is worth investigation in subsequent studies.

4.7 Discussion

In this chapter, we propose a 3D object representation using part templates of geometric shapes, and a method for learning 3D object templates from images by quantizing spaces. Experiments show that the proposed method can learn meaningful 3D car templates from view labeled images, and give comparable performance in object detection and pose estimation.

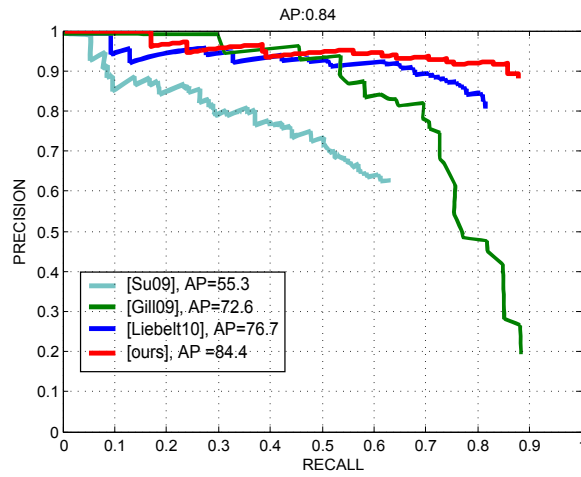


Figure 4.12: Object recognition performance on the 3D car dataset [Savarese and Fei-Fei, 2007]. All curves except the red one are from [Liebelt and Schmid, 2010].

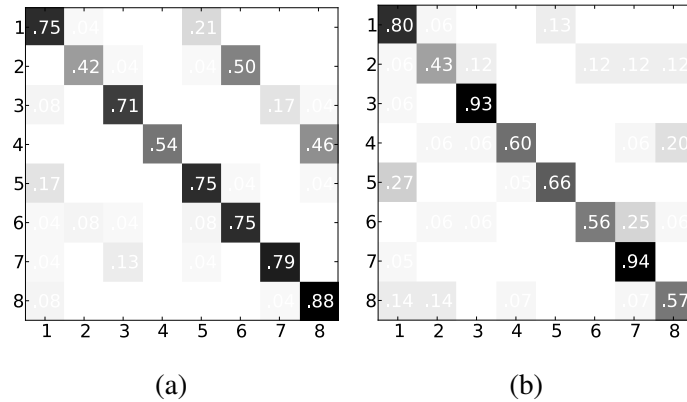


Figure 4.13: Confusion matrix for pose estimation in dataset [Savarese and Fei-Fei, 2007]. Left: results from [Liebelt and Schmid, 2010], AP = 0.70. Right: ours, AP = 0.69.

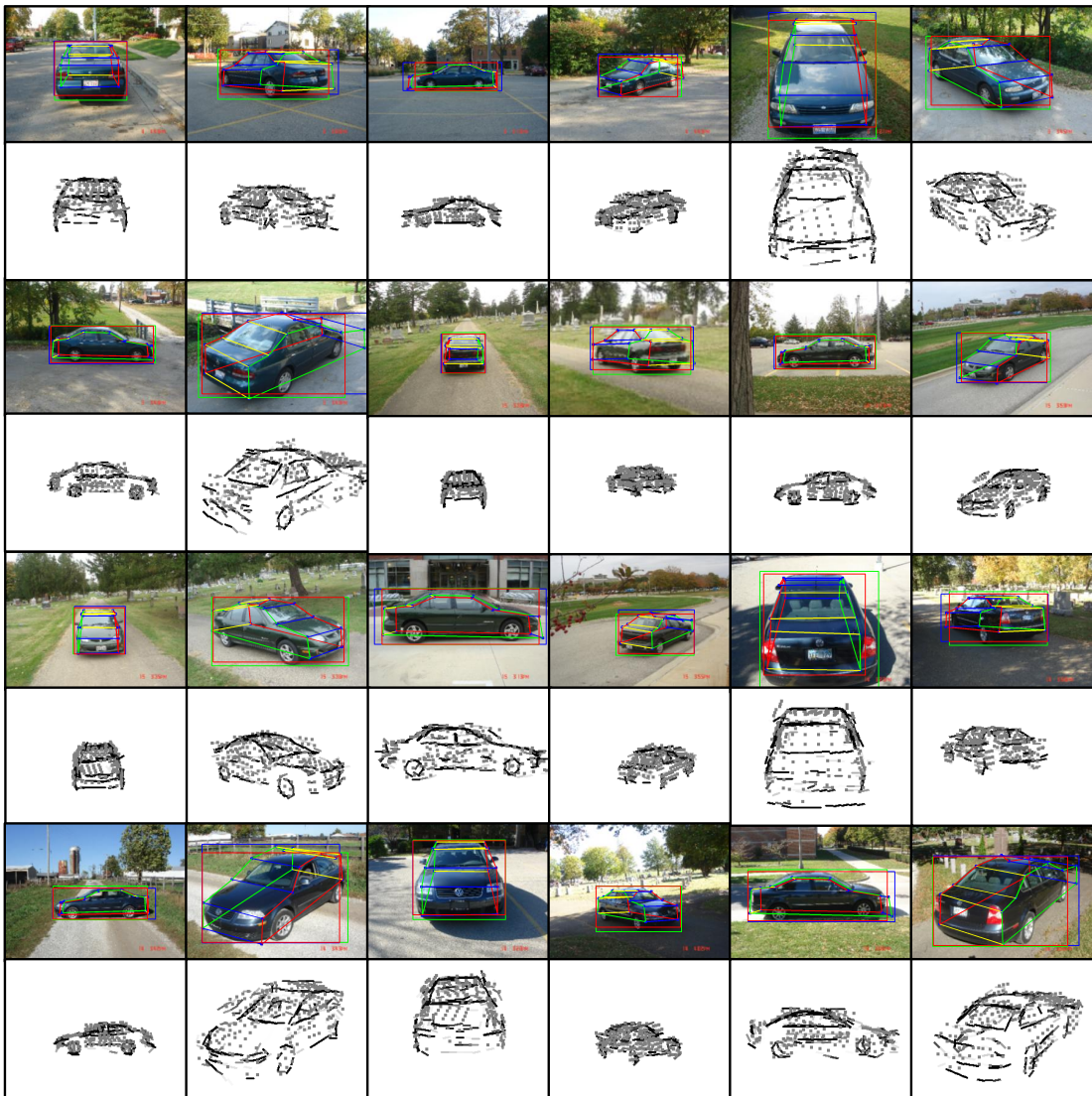


Figure 4.14: Sample experiment results. Green rectangle: the ground truth object bounding box. Blue rectangle: reported bounding box. Red rectangle: their intersection. 3D wire frame shows estimated object pose. Dots in templates show positions of sampled color patches.

CHAPTER 5

Conclusions and Future Work

5.1 Thesis Summary

Previous chapters of this thesis explore models that integrate 3D and 2D representations for view invariant object representation.

This thesis first presents the 3D object primitives and 2D image primitives as the atomic elements of the compositional models introduced in Chapter 3 and Chapter 4, which alone can already be used to learn meaningful templates from object images with relatively clean background. We also propose to use information gain as the criterion to evaluate the information contribution of these elements to their composing models, and introduce a pursuit algorithm that learns flat models mixing different proportions of 3D and 2D primitives for different object images, depending on the structural complexity of the modeled object category.

We further propose to compose the 2D image primitives to middle level visual concepts, such as line segments, curves and corners, which are called active curve templates and active corner templates. Compared to the 2D primitives, or equivalently active basis or Gabor wavelets when view are not considered, introducing middle level concepts would lead to even more sparse image representations, as these concepts can be considered as further coding of the sparsely selected 2D image primitives. By proposing these middle level concepts, we also implicitly introduce the ideas of shape parameterization and parameter space quantization. Together with the designed composition

hierarchy, an efficient algorithm based on dynamic programming can be used to exhaustively compute scores for all the possible active curve and corner templates in the quantified space.

Finally, we formally introduce the space quantization approach through the AND-OR Tree (AoT) structure. Based on the hierarchy from 2D primitives to active curves, the AoT further composes active curves into regularized shapes as appearance of 3D panels, and compose panels to 3D object templates. By enumerating compositions (AND) and their equivalents (OR), the AoT quantifies a huge joint space of 3D object templates using a limited number of nodes, where the space spans from 3D object volume all the way to observed image pixels. Through this quantization, we are also able to integrate the compositional 3D object representations and 2D image representations in a coherent formulation, which leads to consistent learning and inference algorithms mainly composed of simple sum and max operations.

5.2 Future Work

Based on the current research presented in this thesis, there are still many opportunities not yet be investigated. In the following, we provide a short outline of future lines of work which build on the results of the present thesis.

1. Introducing more shapes and alternative paths of shape compositions. The objects around us are not solely composed of circles, rectangles and trapezoids, and those shapes can be decomposed to entities other than line segments, where at least the corners serve as good candidates. Bringing more shapes and more ways of compositions will definitely increase the representation power of the proposed visual hierarchy, while the proposed learning and inference algorithms can still be used without significant modification.
2. Adding other visual cues into the proposed representation hierarchy. Though im-

portant, the shape is just one of many visual cues [Livingstone and Hubel, 1987] which contributes to the robustness of the human vision system. Correspondingly, it would certainly help if our proposed model would also process other cues such as color, texture or even motion. With the current proposed framework, it is not difficult to add features or primitives capturing these information, but it would dramatically increase the computational complexity of the algorithm, as the resulting image representation would not be as sparse as only using shape cues.

3. Modeling more object categories and their interconnections. In this thesis, the frequently used term "car" largely refers to the sedan, whereas other car types are often neglected. It would be certainly interesting to extend the current model to more types of cars, in both theory and practical perspectives. As these car types are closely related, we can further extend the current model to share 3D parts or even part clusters. In training, the sharing increases the number of effective training images for each part, and in testing, it will boost the speed of object detection. Compared with models such as [Liebelt and Schmid, 2010] and [Felzenszwalb et al., 2010] which model cars regardless of their types, the proposed modeling approach would describe the image distribution of car images in finer scale, and provide car type information that is useful in scenarios such as traffic surveys.
4. A more efficient object recognition scheme. In terms of methodologies, this thesis mainly proposes approaches which efficiently learn models for view invariant recognition, and the methods for recognition is largely using dense sliding window method. Though enough for the purpose of validating learned models, the method is inefficient in that it tests too many hypotheses. It is definitely worth exploring algorithms that would dynamically scheduling bottom-up and top-down process [Wu, 2011] along the hierarchy, which would significantly increase the speed of recognition.

BIBLIOGRAPHY

- Arie-Nachimson, M. and R. Basri (2009). Constructing implicit 3d shape models for pose estimation. In *Proceedings of IEEE Int'l Conf. Computer Vision*, pp. 1341–1348.
- Bay, H., A. Ess, T. Tuytelaars, and L. Van Gool (2008). Speeded-up robust features (surf). *Computer Vision and Image Understanding* 110(3), 346–359.
- Biederman, I. (1987). Recognition-by-components: A theory of human image understanding. *Psychological Review* 94, 115–117.
- Biederman, I. and P. C. Gerhardstein (1993). Recognizing depth-rotated objects: Evidence and conditions for three-dimensional viewpoint invariance. *J. Exp. Psychol. Hum. Percept. Perform.* 19, 1162–82.
- Biederman, I. and P. C. Gerhardstein (1995). Viewpoint-dependent mechanisms in visual object recognition: Reply to tarr and bülhoff (1995). *J. Exp. Psychol. Hum. Percept. Perform.* 21(6), 1506–1514.
- Bienenstock, E., S. Geman, and D. Potter (1997). Compositionality, mdl priors, and object recognition. In *Advances in Neural Information Processing Systems*.
- Binford, T. (1971). Visual perception by computer. In *Proceedings of IEEE Conf. Systems and Control*.
- Brown, M. and D. G. Lowe (2005). Unsupervised 3d object recognition and reconstruction in unordered datasets. In *Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling*, pp. 56–63.
- Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* 8(6), 679–698.

- Chang, C.-C. and C.-J. Lin (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2, 27:1–27:27. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- Dalal, N. and B. Triggs (2005). Histograms of oriented gradients for human detection. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 886–893.
- Dance, C. R., J. Willamowski, L. Fan, C. Bray, and G. Csurka (2004). Visual categorization with bags of keypoints. In *ECCV International Workshop on Statistical Learning in Computer Vision*.
- Daugman, J. G. (1985). Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters. *J. Opt. Soc. Am. A* 2(7), 1160–1169.
- Desolneux, A., L. Moisan, and J.-M. Morel (2000). Meaningful alignments. *Int. J. Comput. Vision* 40(1), 7–23.
- Dickinson, S., A. Pentland, and A. Rosenfeld (1991, jun). From volumes to views: an approach to 3-d object recognition. In *Workshop on Directions in Automated CAD-Based Vision*, pp. 85 –96.
- Dickinson, S. J., A. P. Pentland, and A. Rosenfeld (1992). From volumes to views: An approach to 3-d object recognition. *CVGIP: Image Understanding* 55(2), 130 – 154.
- Epshtein, B., I. Lifshitz, and S. Ullman (2008). Image interpretation by a single bottom-up top-down cycle. *Proceedings of the National Academy of Sciences* 105(38), 14298–14303.
- Fan, R., K. Chang, C. Hsieh, X. Wang, and C. Lin (2008). Liblinear: A library for large linear classification. *The Journal of Machine Learning Research* 9, 1871–1874.

- Felzenszwalb, P. F., R. B. Girshick, D. McAllester, and D. Ramanan (2010). Object detection with discriminatively trained part based models. *IEEE Trans. Pattern Anal. Mach. Intell.* 32(9), 1627–1645.
- Ferrari, V., F. Jurie, and C. Schmid (2010). From images to shape models for object detection. *Int. J. Comput. Vision* 87(3), 284–303.
- Freund, Y. and R. E. Schapire (1996). Experiments with a new boosting algorithm. In *Proceedings of Int'l Conf. Machine Learning*.
- Friedman, J. H. (1987). Exploratory projection pursuit. *Journal of the American Statistical Association* 82(397), pp. 249–266.
- Guo, C., S.-C. Zhu, and Y. N. Wu (2003). Towards a mathematical theory of primal sketch and sketchability. In *Proceedings of IEEE Int'l Conf. Computer Vision*, pp. 1228–1235.
- Harris, C. and M. Stephens (1988). A combined corner and edge detector. In *Proceedings of Fourth Alvey Vision Conference*.
- Hayward, W. G. and M. J. Tarr (1997). Testing conditions for viewpoint invariance in object recognition. *J. Exp. Psychol. Hum. Percept. Perform.* 23(5), 1511–1521.
- Hoiem, D., C. Rother, and J. Winn (2007). 3d layout crf for multi-view object class recognition and segmentation. In *Proceedings of IEEE Int'l Conf. Computer Vision*.
- Hsiao, E., A. Collet Romea, and M. Hebert (2010). Making specific features less discriminative to improve point-based 3d object recognition. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 2653–2660.
- Hu, W., Y. N. Wu, and S.-C. Zhu (2011). Image representation by active curves. In *Proceedings of IEEE Int'l Conf. Computer Vision*.

- Hu, W. and S.-C. Zhu (2010). Learning a probabilistic model mixing 3d and 2d primitives for view invariant object recognition. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 2273–2280.
- Huo, X. and J. Chen (2005). JBEAM: multiscale curve coding via beamlets. *IEEE Transactions on Image Processing* 14, 1665–1677.
- Jaynes, E. T. (1957). Information theory and statistical mechanics. *Phys. Rev.* 106(4), 620–630.
- Koenderink, J. and A. Doorn (1976). The singularities of the visual mapping. *Biological cybernetics* 24(1), 51–59.
- Koenderink, J. and A. Doorn (1979). The internal representation of solid shape with respect to vision. *Biological cybernetics* 32(4), 211–216.
- Kovesi, P. D. (2012). MATLAB and Octave functions for computer vision and image processing. <<http://www.csse.uwa.edu.au/~pk/research/matlabfns/>>.
- Kushal, A. and J. Ponce (2006). Modeling 3d objects from stereo views and recognizing them in photographs. In *Proceedings of the 9th European conference on Computer Vision - Volume Part II, ECCV'06, Berlin, Heidelberg*, pp. 563–574. Springer-Verlag.
- Kushal, A., C. Schmid, and J. Ponce (2007). Flexible object models for category-level 3d object recognition. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 2273–2280.
- Lazebnik, S., C. Schmid, and J. Ponce (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition, Volume 2*, pp. 2169 – 2178.
- Leibe, B. and B. Schiele (2003). Analyzing appearance and contour based methods

- for object categorization. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, Volume 2, pp. 409–415.
- Leopardi, P. (2006). A partition of the unit sphere into regions of equal area and small diameter. *Electronic Transactions on Numerical Analysis* 25, 309–327.
- Liebelt, J. and C. Schmid (2010). Multi-view object class detection with a 3D geometric model. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 1688–1695.
- Liebelt, J., C. Schmid, and K. Schertler (2008). Viewpoint independent object class detection using 3d feature maps. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*.
- Livingstone, M. and D. Hubel (1987). Psychophysical evidence for separate channels for the perception of form, color, movement, and depth. *The Journal of Neuroscience* 11(1), 3416–3468.
- Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Proceedings of IEEE Int'l Conf. Computer Vision*, pp. 1150–1157.
- Mallat, S. and Z. Zhang (1993). Matching pursuits with time-frequency dictionaries. *IEEE Trans. on Sig. Proc.* 41(12), 3397–415.
- Marr, D. (1982). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. San Francisco, CA, USA: W.H. Freeman.
- Marr, D. and H. K. Nishihara (1978). Representation and recognition of the spatial organization of three-dimensional shapes. In *Proceedings of the Royal Society of London.*, Volume 200 of B, pp. 269–294.
- Minsky, M. (1974). A framework for representing knowledge. Technical report, Massachusetts Institute of Technology.

- Olshausen, A. B., P. Sallee, and M. S. Lewicki (2001). Learning sparse image codes using a wavelet pyramid architecture. In *Advances in Neural Information Processing Systems*, Volume 13, pp. 887–893.
- Olshausen, B. A. and D. J. Field (1996, 06). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature* 381(6583), 607–609.
- Payet, N. and S. Todorovic (2011). From contours to 3d object detection and pose estimation. In *Proceedings of IEEE Int'l Conf. Computer Vision*.
- Pentland, A. P. (1986). Perceptual organization and the representation of natural form. *Artificial Intelligence* 28(3), 293 – 331.
- Pepik, B., M. Stark, P. Gehler, and B. Schiele (2012). Teaching 3d geometry to deformable part models. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 3362–3369.
- Pietra, S. D., V. D. Pietra, and J. Lafferty (1997). Inducing features of random fields. *IEEE Trans. Pattern Anal. Mach. Intell.* 19(4), 380–393.
- Riesenhuber, M. and T. Poggio (1999). Hierarchical models of object recognition in cortex. *Nature Neuroscience* 2, 1019–1025.
- Savarese, S. and L. Fei-Fei (2007). 3d generic object categorization, localization and pose estimation. In *Proceedings of IEEE Int'l Conf. Computer Vision*.
- Shi, J. and C. Tomasi (1994). Good features to track. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 593–600.
- Si, Z., H. Gong, Y. N. Wu, and S.-C. Zhu (2009). Learning mixed image templates for object recognition. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 272–279.

- Su, H., M. Sun, F.-F. Li, and S. Savarese (2009). Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *Proceedings of IEEE Int'l Conf. Computer Vision*, pp. 213–220.
- Sun, M., H. Su, S. Savarese, and F.-F. Li (2009). A multi-view probabilistic model for 3d object classes. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*.
- Tarr, M. J. and H. H. Bühlhoff (1995). Is human object recognition better described by geon structural descriptions or by multiple views? comment on biederman and gerhardstein (1993). *J. Exp. Psychol. Hum. Percept. Perform.* 21(6), 1494–1505.
- Taubman, D. S. and M. W. Marcellin (2002). *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Springer.
- Thomas, A., V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool (2006). Towards multi-view object class detection. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, pp. 1589–1596.
- Torralba, A., K. P. Murphy, and W. T. Freeman (2007). Sharing visual features for multiclass and multiview object detection. *IEEE Trans. Pattern Anal. Mach. Intell.* 29(5), 854–869.
- Viola, P. A. and M. J. Jones (2001). Rapid object detection using a boosted cascade of simple features. In *Proceedings of IEEE CS Conf. Computer Vision and Pattern Recognition*, Volume 1, pp. 511–518.
- Wu, T. (2011). *Integration and Goal-Guided Scheduling of Bottom-Up and Top-Down Computing Processes in Hierarchical Models*. Ph. D. thesis, University of California, Los Angeles.

- Wu, Y. N., Z. Si, H. Gong, and S.-C. Zhu (2010). Learning active basis model for object detection and recognition. *International Journal of Computer Vision* 90(2), 198–235.
- Yan, P., S. M. Khan, and M. Shah (2007). 3d model based object class detection in an arbitrary view. In *Proceedings of IEEE Int'l Conf. Computer Vision*.
- Zhu, S.-C. and D. Mumford (2006). A stochastic grammar of images. *Found. Trends. Comput. Graph. Vis.* 2(4), 259–362.