Learning 3D Object Templates by Quantizing Geometry and Appearance Spaces

Wenze Hu and Song-Chun Zhu

Abstract—While 3D object-centered shape-based models are appealing in comparison with 2D viewer-centered appearance-based models for their lower model complexities and potentially better view generalizabilities, the learning and inference of 3D models has been much less studied in the recent literature due to two factors: i) the enormous complexities of 3D shapes in geometric space; and ii) the gap between 3D shapes and their appearances in images. This paper aims at tackling the two problems by studying an And-Or Tree (AoT) representation that consists of two parts: i) a geometry-AoT quantizing the geometry space, i.e. the possible compositions of 3D volumetric parts and 2D surfaces within the volumes; and ii) an appearance-AoT quantizing the appearance space, i.e. the appearance variations of those shapes in different views. In this AoT, an And-node decomposes an entity into constituent parts, and an Or-node represents alternative ways of decompositions. Thus it can express a combinatorial number of geometry and appearance configurations through small dictionaries of 3D shape primitives and 2D image primitives. In the quantized space, the problem of learning a 3D object template is transformed to a structure search problem which can be efficiently solved in a dynamic programming algorithm by maximizing the information gain. We focus on learning 3D car templates from the AoT, and collect a new car dataset featuring more diverse views. The learned car templates integrate both the shape-based model and the appearance-based model to combine the benefits of both. In experiments, we show three aspects: 1) the AoT is more efficient than the frequently used octree method in space representation; 2) the learned 3D car template matches the state-of-the art performances on car detection and pose estimation in a public multi-view car dataset; and 3) in our new dataset, the learned 3D template solves the joint task of simultaneous object detection, pose/view estimation, and part localization. It can generalize over unseen views and performs better than the version 5 of the DPM model in terms of object detection and semantic part localization.

Index Terms—Hierarchical Models, 3D Object Models, Structure Learning, And-Or Tree, Object Detection, Pose Estimation

1 Introduction

1.1 Motivation and objective

In the first three decades of research on object recognition from middle 1960s to middle 1990s, the predominant theory was to represent objects in 3D shapes for some obvious reasons [2], [6], [27]: i) 3D shapes are essential for human perception, grasp and manipulation; ii) complex objects can be decomposed into common 3D primitives (i.e. generalized cylinders) shared across categories; and iii) 3D shapes can be better generalized to novel views and poses. In addition, a parsimonious 3D model, from the perspective of learning, potentially needs less training examples as information can be pooled from different views. Despite these desirable properties, the paradigm shifted in the late 1990s to 2D view-based appearance models, as two factors defeated the computation of 3D representations: i) The enormous *appearance variations* create a gap between 3D shapes and their input images; ii) Real world objects have larger *geometric variations* than what the generalized cylinders can account for.

The appearance-based methods have made remarkable progresses in the past two decades, however, they are mainly focused on categorical classification and detection. When an image is classified correctly, the object may not be localized. When an object is detected, e.g. by the popular deformable part model [7], its parts may be located to wrong places. Thus the 3D shapes, their parts and poses are left unsolved. Recently there are increasing interests in revisiting the 3D representation paradigm [23], [29], [43] and to combine the two methods. The potential benefits of integrating shape-based and appearance-based models are substantial and motivate the work in this paper.

The objective of this paper is to learn 3D object templates from real images, and to apply the 3D model to solving the tasks of object detection, pose/view estimation, and part localization jointly. We focus on the category of cars (sedans) as cars are solid 3D objects and have been extensively studied in the literature. As the training images are from different car instances over different views, common geometric and appearance structures must be extracted and shared across views and instances to learn a coherent 3D template. Fig. 1 (a) displays a 3D car template which is projected to 2D templates at different views in (b), deformed and instantiated with appearance features in (c), and matched to images in (d). As the 3D template is hierarchical and compositional, a dynamic programming algorithm is used to find the best matching and thus compute the view, pose and parts simultaneously.

• *W. Hu and S.-C. Zhu are with the Department of Statistics, University of California, Los Angeles, Los Angeles, CA, 90095.*
E-mail: {wzhu,sczhu}@stat.ucla.edu

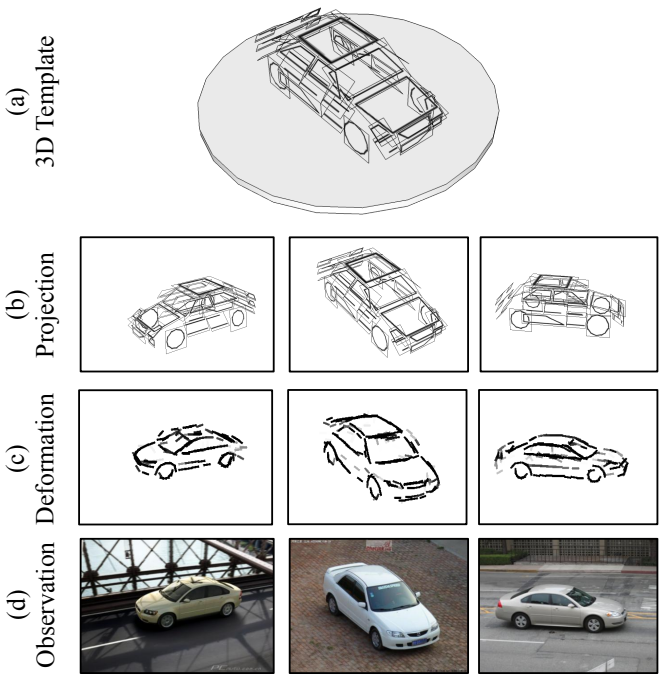


Fig. 1. 3D object recognition. (a) A learned 3D car template is composed of multiple 3D volumetric parts. Within each volume a 2D planar panel is used to fit the local surface. On each 2D panel, an appearance template is defined and is decomposed into primitive 2D shapes, line segments and sketches. (b) At each specific view, the learned 3D template is projected to derive a 2D template. (c) 2D templates are then deformed and instantiated with appearance features (Gabor filters) to match the images in (d).

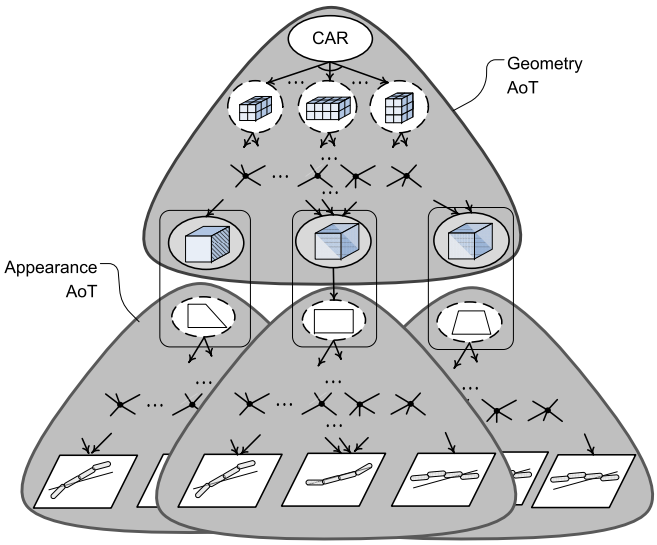


Fig. 2. Overview of the And-Or Tree Structure. The AoT is factorized as geometry-AoT and appearance-AoT to quantize the geometry and appearance spaces respectively

1.2 Overview of the Proposed Method

To tackle the two factors – large geometric complexities and appearance variations which defeated early 3D categorical modeling efforts, we propose an And-Or Tree (AoT) structure to quantize the geometry and appearance spaces in a principled way. The AoT is composed of consecutive layers of And and Or-nodes. An And-node represents a decomposition into parts in 2D or 3D, and an Or-node represents alternative ways of the decomposition. In the quantized spaces, the structure learning problem is transformed to a search task in a finite AoT and can be solved efficiently by dynamic programming.

As is shown in Fig. 2, we factorize the space of possible 3D templates into geometry and appearance spaces, and thus the AoT is divided into two aspects: i) a geometry G-AoT which is *object-centered* and *shape-based*; and ii) an appearance A-AoT which is *view-centered* and *appearance-based*.

1. Quantizing the geometry space by G-AoT. Consider a car bounded by a 3D finite volume with $M \times N \times K$ atomic cubes. At the top 2-levels of the G-AoT, we decompose the car into a number of 3D volumetric parts according to the best fit to CAD models. Then each volume can be terminated or split in a few ways (i.e. Or-nodes) along the x,y,z directions. Suppose a terminal volume has $m \times n \times k$ atomic cubes, there are a finite number of ways to place a 2D planar panel inside the volume surface. All the 2D panels in the selected (through Or-node switches) terminal volumes form a 3D surface to approximate the surface of the car. Fig. 1.(a) shows an example of the selected panels. As the G-AoT has many, technically over-complete, ways to divide the volumes and tilt the surface panels, it has enough flexibility to address the different car designs and their geometric variations. We will show the efficiency of the G-AoT in experiments in comparison with the octree decomposition.

2. Quantizing the appearance space by A-AoT. As is shown in the lower half of Fig. 2, each A-AoT defines a simple and deformable shape template on each 2D planar panel at the terminal node of the G-AoT in a certain range of views. These deformable appearance templates are simple shapes, such as circles, rectangles and trapezoids and so on. These primitive shapes can be parameterized and quantized into parametrized line segments. The line segments are instantiated by active curves [14] after their projections to 2D images, and further decomposed into deformable Gabor filters [5], as in the active basis models [38], to match with image pixels. Therefore the 3D shape-based model is grounded on images by an appearance-based model.

The AoT embodies a stochastic context free grammar. By selecting the children of Or-nodes like switches on the G-AoT, we can derive specific 3D templates, such as the one shown in Fig. 1(a). By switching along the selected A-AoTs, we can further derive the deformed 2D templates, such as the ones in Fig. 1(b) and (c). Therefore,

the AoT can generate a combinatorial number of configurations or templates, and thus has the expressive power to represent large geometric and appearance variations through small dictionaries of shape and appearance primitives.

The AoT not only tackles the model complexity, but also transforms the template learning problem to a structure search problem in the space of possible templates defined by the AoT. The learning algorithm finds the optimal 3D template which best explains the training images from different views in terms of maximizing the information gain criterion. The information gain is defined based on our probabilistic image model, in which the image likelihood is defined on individual Gabor responses. As the information gains can be factorized along the AoT hierarchy, this structure search problem can be efficiently solved by dynamic programming.

After the template is learned through dynamic programming, we further enrich the appearance model with texture features in the interior area of the selected 2D simple shapes and retrain all feature weights using SVM. In this way, the enriched model not only captures the rough outline of the objects, but also gains discriminative power through the shade and texture info of the object.

Given a testing image, we generate 2D templates by projecting the 3D template to a set of hypothesized views, and deform these 2D templates according to the A-AoTs to match the image. The matching process is another round of dynamic programming, as it tries to find the best deformation with maximum likelihood ratios among all the possible template deformations. In this way, the 3D template can be used to detect objects from arbitrary and even unseen views in the view sphere, estimate their poses by reporting the hypothesized view, and localize semantic parts by tracing the 2D positions of deformed Gabor filters of each semantic part.

1.3 Related Literature

Our work is related to the following three threads of research in the literature.

1, Object-centered and viewer-centered models for 3D object recognition. Early models for 3D object recognition can be categorized to object-centered models and viewer-centered models. The object-centered models were studied by Binford et al. [3] in the 1970s, and were further developed into models such as Geons [2] by Biederman et al. and 3D primitives [6] by Dickinson et al. In fact, many of these primitives are still used as building components in modern CAD softwares. The advantages and drawbacks of this representation have been discussed in Section 1.1. To solve the appearance gap, later work ([1], [12], [19], [24], [40]) proposed to use point-based 3D models, and replace explicit image appearance descriptions by image summary statistics such as SIFT [26] or its quantized version [12]. Though the SIFT descriptors are robust to global lighting changes and affine transforms, they do not generalize well across object instances and large view spans. Recent 3D object models try to use 3D panels with HOG descriptors as the representation, which gets good performance in terms of object detection and pose estimation. Different from our work, the panels are either pre-defined [30] or completely learned from CAD data [39].

Viewer-centered models can be dated back to the aspect-graph representation by Koenderink and Doorn [17], [18]. These models are popular [28], [33], [35] as there is no need to construct a view consistent 3D model, and they can utilize recent developments on appearance-based object classification ([4], [20]) for single or a few views. Because the view consistency prior is not explicitly used, information in individual images is not shared across views, which suggests that more training examples should be used to learn a robust model for each view.

2, Recent models combining the two types of representations. Some recent methods try to share the 3D geometry of object parts across views, while keeping their part appearance view specific. Among them Liebelt and Schmid [23] constructed view specific spatial pyramid models for both the object and its parts using training images, and associate them to 3D space using CAD images rendered at the same views. Pepik et.al. [29] extended the deformable part based model (DPM) [7] by initializing part positions and sizes in different views together in the 3D space, so that part geometry consistency can be achieved. Hejrati and Ramanan [11] use a 2D representation to detect the locations of key object parts, which are then fitted into a 3D model to estimate the 3D shape of the object as well as its pose. Though achieving high detection performance, these models use quasi-densely sampled appearance features, resulting in models with high model complexity. With comparable performance in object recognition tasks, our model uses much less features, as these appearance features are reused (shared) across views. Besides, as a 3D model, the model complexity of our model does not change as the number of views increases.

3, Learning hierarchical compositional models. Our model can be considered a 3D extension of the And-Or graph and the stochastic image grammar [42]. Similar problems of learning compositional models have been studied in single view object modeling problems. Most of the existing methods [8], [32], [41] learn the hierarchy layer-by-layer in a bottom-up greedy fashion, and do not necessarily optimize a consistent objective function in learning different layers. In particular, Si and Zhu [32] proposed an AoT learning algorithm for view-based object modeling, the key differences of the two methods are: a) The AoT model space in [32] has open structure and is continuous, while the AoT model space in this paper is discrete and finite. b) For the terminal nodes, Si and Zhu [32] uses sketch, color and texture features, while the current model only uses sketch part to learn the AoT. The texture part is later added to our representation as ”negative” features in a discriminative retrain-

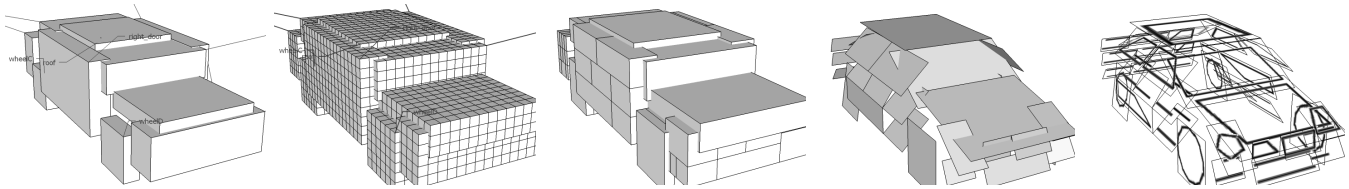


Fig. 3. Illustration of the learning process. (a) Initial volumes for rough parts from CAD models of car. (b) Each volume consists of basic cubic units. The size of each volume is slightly different from that in (a) because of volume size rounding. (c) The selected meaningful parts. (d) Selected 2D panels within each part volume. (e) The shape templates chosen for each 2D panel. The AoT search optimizes the selections from (b) to (c), (d) and (e).

ing step. This further simplifies the structure learning problem. c) The AoT structure learning problem in [32] is solved using a greedy pursuit approach. By quantizing the model space, our learning algorithm directly optimizes the information gain criterion and finds the global optimal efficiently using dynamic programming.

The space quantization approach similar to our AoT is the quad-trees [34] and beamlet [16] in image coding, where image lattice is recursively decomposed into equal sized sub-lattices. In contrast to such dyadic decomposition, the AoT allows multiple decompositions at each node, thus embeds more expressive power than the quad-tree in 2D or the octree in 3D case. The bottom levels of our AoT is based on the active curves model [14], which is composed of deformable Gabor filters in an active basis model [38].

1.4 Contributions

This paper makes the following contributions.

1) We propose a 3D And-Or Tree representation and learn 3D hierarchical and compositional templates for cars. It integrates 3D shape-based models with 2D appearance models and fills in the appearance gap between the shapes and images.
2) By quantizing the spaces of geometry and appearance variations with the 3D AoT, we transform the structure learning problem into a structure search problem and solve it by dynamic programming. This framework can be used in other object categories.
3) We introduce a new 3D car dataset with object views and parts annotated manually. Compared with existing 3D car datasets, our dataset features much more widespread views and part annotations, and provides a new benchmark to test various 3D object category modeling methods.
4) Using the new dataset, we show in experiments that the proposed method can learn meaningful 3D car template with less than 100 shapes, generate boundaries of object instances in different views, detect objects, estimate their poses and localize the semantic parts. Using the most recent version of the DPM model as a baseline [9], our model achieves slightly better performance in object detection and much better performance in object part localization. Since DPM does not localize semantic parts, we use positions of its parts to predict semantic part locations.

In comparison to a previous conference version [13], we reformulate the AoT using a clear layered definition, and add more discussion and details about the implementation. We significantly expand the experiments: i) we add experiments showing the capacity and efficiency of the AoT in space quantization in the context of reconstructing 3D CAD models; ii) we test the performance of the proposed 3D object template on view generalization; iii) we improve the object detection performance by incorporating texture features in the 2D surface panel; and iv) we add a quantitative evaluation of semantic object part localization against the latest release of the DPM model.

The rest of the paper is organized as follows: Section 2 introduces the And-Or tree design for space quantization. Section 3 presents the probabilistic image model and the information gain criterion. Section 4 and 5 presents the bottom-up and top-down learning and inference algorithms. Our proposed dataset and experiment results are presented in Section 6.

2 And-Or Tree Representation

In this section, we elaborate on how the G-AoT and A-AoT shown in Fig. 2 quantize the geometry and appearance variations respectively.

2.1 G-AoT for Geometry Space Quantization

The root of the G-AoT is an And-node representing the 3D bounding volume of the car category. This volume is decomposed into 12 Volumes of Interests (VoIs), each representing the bounding volume of the semantic parts. These parts are extracted from a 3D CAD model representing the typical shape of the cars and the i-th part is a rectangular volume with $m_i \times n_i \times k_i$ atomic cubes. The relative sizes and locations of these components are shown in Fig. 3(a).

An **Or-node** in the G-AoT corresponds to a volume, which can be either terminated as leaf-nodes or split into two And-nodes in multiple ways. For example, the top red dashed ellipse in Fig. 4(a) is an Or-node,

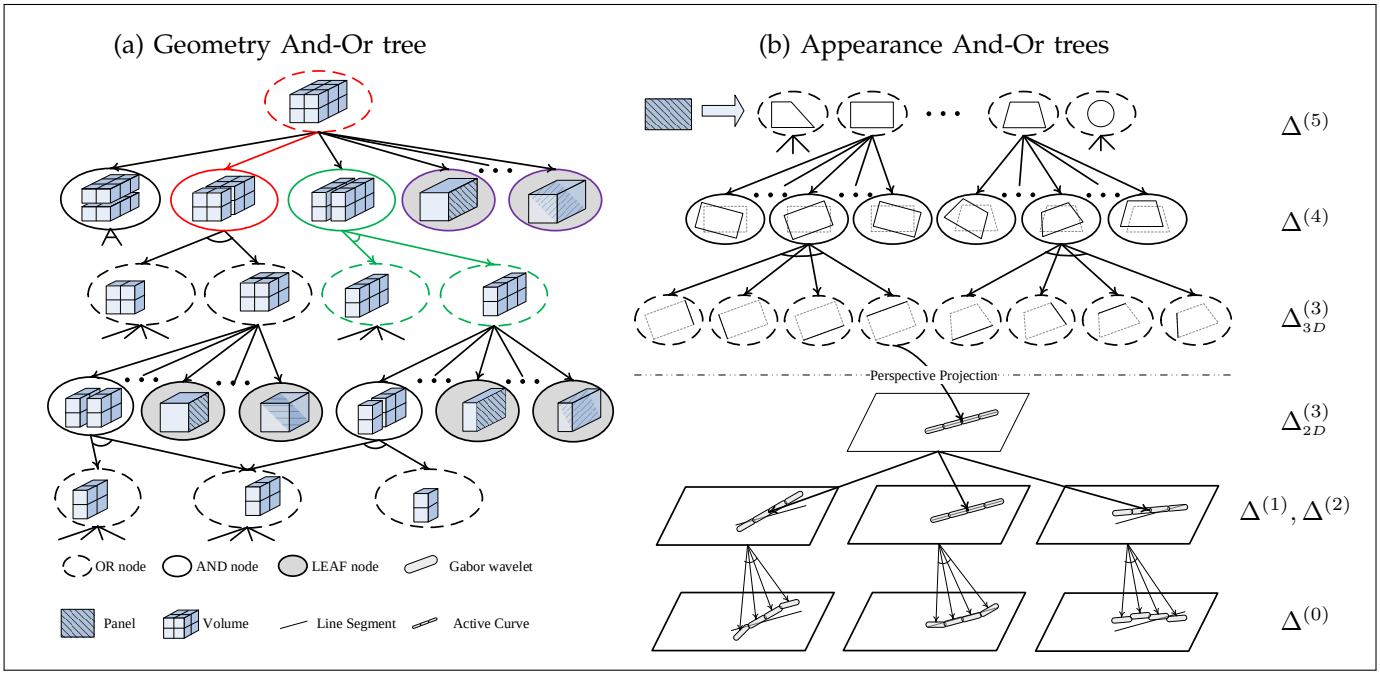


Fig. 4. (**a**): Geometry And-Or Tree (G-AoT), where And-nodes represent combinations of two sub-volumes occupying larger sub-volumes, Or-nodes connect to multiple And-nodes representing possible combinations for the same sub-volume, and leaf-nodes represent panels inscribing their parent volumes. (**b**): Each panel represents the geometry of a shape template, and is connected to an Appearance And-Or Tree (A-AoT). Here And means composition and Or is for deformation. It extends the G-AoT to image space since its leaf-nodes are Gabor filters.

and is denoted by $V^{\mathrm{O}}_{X,D}$ with $X = (0,0,0)$ being the 3D coordinate of the innermost vertex of the volume, $D = (3,2,2)$ is the size of the volume. It is split along one dimension and the split is denoted by

$$V^{\mathrm{O}}_{X,D} \rightarrow V^{\mathrm{A}}_{X,D,C}. \tag{1}$$

For example, the red subtree in Fig. 4(a) is: $V^{\mathrm{O}}_{(0,0,0),(3,2,2)} \rightarrow V^{\mathrm{A}}_{(0,0,0),(3,2,2),(2,0,0)}$ with $C = (2,0,0)$ indicating that the split is perpendicular to the first dimension at position 2.

An **And-node** in the G-AoT represents a volume split. For example the solid green node in Fig. 4(a) is denoted by $V^{\mathrm{A}}_{(0,0,0),(3,2,2),(0,1,0)}$. Its split is expressed as

$$V^{\mathrm{A}}_{X,D,C} \rightarrow V^{\mathrm{O}}_{X,D_1} V^{\mathrm{O}}_{X+C,D_2}, \tag{2}$$

where D_1 and D_2 are the sizes of the two volumes after splitting. So, the green sub-tree can be denoted as $V^{\mathrm{A}}_{(0,0,0),(3,2,2),(0,1,0)} \rightarrow V^{\mathrm{O}}_{(0,0,0),(3,1,2)} V^{\mathrm{O}}_{(0,1,0),(3,1,2)}$.

A **leaf-node** of the G-AoT stops the volume decomposition and places a planar panel inside the volume to fit the 2D surface of the object. We define two types of panels: those on the frontal surface of the volume, and diagonal of the volume. The panels are illustrated by the line-shaded parallelograms in Fig. 4(a). Though only a restricted set of panels is allowed, they still represent large variations in positions, sizes and orientations. Fig. 5 shows a cross-section view of these panels in a 3D VoI, where panels are defined on sub-volumes inside the VoI according to the rules above.

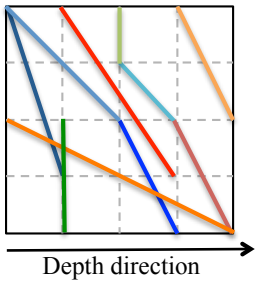


Fig. 5. The cross-section view of the alternative panels inside a VoI, which will be selected to fit the 3D surfaces.

2.2 A-AoT for Appearance Space Quantization

While the G-AoT is object-centered and shape-based, the A-AoT is view-centered and appearance-based. For each leaf-node in the G-AoT, the A-AoT defines the image appearance of its panel for a range of views where the panel is visible. In experiments, a panel is considered visible if the inner product between its surface normal and the inverse of camera view direction is larger than 0.6. Fig. 4.(b) shows a number of shape primitives – trapezoid, rectangle, circle etc assigned to a panel. Each of them is an Or-node in the A-AoT and branches to a number of quantized views as projected shape primitives on the image plane. Such nodes are invariant to local view changes.

These projected shape primitives are further decomposed into line segments which are split into deformable

sketches. The primitives are organized in layered dictionaries $\Delta^{(k)}$ in Fig. 4.(b), where even numbered layers are for the And-nodes, odd ones for Or-nodes, and $k = 0$ for leaf-nodes. Table 1 defines these quantized primitives, their parameters and value ranges, which we shall elaborate in the following.

$\mathbf{\Delta^{(5)}}$ includes three types of distinctive shapes defined on the planar panels in 3D space – i) circles; ii) trapezoids with rectangles as special cases; and iii) parallel lines. The trapezoid and parallel lines further contain 6 subtypes shown in Table 1. We call them the dictionary of *shape templates* $\mathbf{\Delta^{(5)}} = \{S\}$.

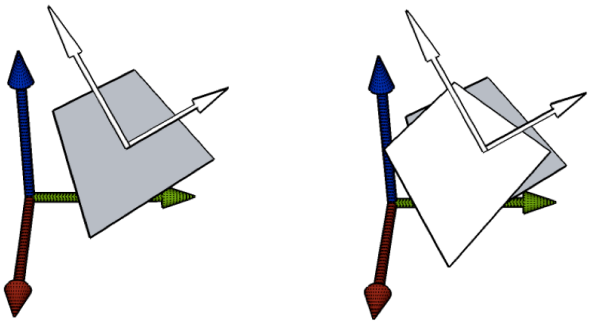


Fig. 6. An example of the 3D deformation for the shape templates. We allow the template to rotate round panel center and translate along the axis directions.

$\mathbf{\Delta^{(4)}}$ includes the *deformed shape templates*. We allow each S in $\Delta^{(5)}$ to translate along the two axis of the panel sides and rotate and scale in the panel plane (see Fig. 6), each at three discrete levels specified in column 5 of Table 1. Thus we derive a set of $3\times3\times3\times3 = 81$ deformed shape templates dS for each S in $\Delta^{(5)}$. Let ∂S denote the equivalent class of S subject to bounded deformations. Then $\Delta^{(4)}$ is the union of all these deformed templates

$$\Delta^{(4)} = \cup\, \partial S, \quad \text{for} \quad S \in \Delta^{(5)}. \tag{3}$$

$\mathbf{\Delta^{(3)}}$ is defined both on the panels in 3D $\Delta^{(3)}_{3D}$ and on image planes $\Delta^{(3)}_{2D}$. $\Delta^{(3)}_{3D}$ contains the line segments L composing the deformed shape templates in $\Delta^{(4)}$, and they are realized as a subset of active curves A on images. The latter are denoted by $\Delta^{(3)}_{2D}$.

For the convenience of parameterizing trapezoid shape templates S, the line segments L are parameterized in the coordinates of the panel, with the origin at the center of the panel, and two axes along the side directions of panels. In this coordinate system, a line segment is parameterized by (u, v, o, ι), which are center position $(u, v,)$ along the two coordinate axes, line orientation o and length ι respectively. Trapezoid shape templates can then be decomposed and parameterized by those of constituent parallel line segments. For example, as a special case of the trapezoid shape templates, a rectangular shape template $dS \in \Delta^{(4)}$ with width and height as (w, h) can be decomposed into:

$$dS \rightarrow L_{0,-h/2,0,w} L_{0,h/2,0,w} L_{-w/2,0,\pi/2,h} L_{w/2,0,\pi/2,h}, \tag{4}$$

and indexed by $(0, -h/2, 0, w, 0, h/2, 0, w)$.

Thus, in 3D space:

$$\Delta^{(3)}_{3D} = \{L \mid L \in {}^\star dS, \quad dS \in \Delta^{(4)}\}, \tag{5}$$

where ${}^\star dS$ denotes the set of line segments by applying the rules of decomposition.

On the 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 center position (x, y), orientation o and length l.

$$\Delta^{(3)}_{2D} = \Big\{A \mid A = \mathcal{P}(L, \omega),\ L \in \Delta^{(3)}_{3D}, \omega \in \Omega\Big\} \tag{6}$$

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

$\mathbf{\Delta^{(2)}}$ contains the deformed active curves in $\Delta^{(3)}_{2D}$. Each active curve $A_{x,y,o,l}$ is allowed to translate in the range $\partial x, \partial y$, and rotate in a small orientation range ∂o, which derives an equivalent class $\partial A_{x,y,o,l}$

$$\partial A_{x,y,o,l} = \left\{ A_{x',y',o',l} \;\middle|\; \begin{array}{ll} x' = x + \delta x \cos o', & \delta x \in \partial x \\ y' = y + \delta y \sin o', & \delta y \in \partial y \\ o' = o + \delta o, & \delta o \in \partial o \end{array} \right\},$$

and $\Delta^{(2)}$ is the union of all these classes,

$$\Delta^{(2)} = \cup\, \partial A_{x,y,o,l}, \quad \text{for} \in A_{x,y,o,l} \in \Delta^{(3)}_{2D}. \tag{7}$$

$\mathbf{\Delta^{(1)}}$ contains the edges decomposed from the active contours $A \in \Delta^{(2)}$. Following the terminology in [14], an active curve A is decomposed into a sequence of weakly overlapping basis elements B, which are placed along the curve and are parameterized by position (x, y) and orientation o. l is the length or the number of basis elements in A,

$$A_{x,y,o,l} \rightarrow B_{x,y,o} B_{x_1^+, y_1^+, o} B_{x_1^-, y_1^-, o} \cdots B_{x_l^-, y_l^-, o}. \tag{8}$$

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

Therefore $\Delta^{(1)}$ is a set of quantized elements decomposed from $\Delta^{(2)}$

$$\Delta^{(1)} = \{B \mid B \in {}^\star A,\ A \in \Delta^{(2)}\}. \tag{9}$$

${}^\star A$ denoted the decomposed set of basis from active curve A.

$\mathbf{\Delta^{(0)}}$ contains the deformed Gabor basis in $\Delta^{(1)}$, which are a set of Gabor functions. These deformed basis, as leaf-nodes of A-AoT, ground the templates onto image pixels. For each basis $B_{x,y,o} \in \Delta^{(1)}$ at specific locations (x, y) and orientations o, we allow translations and rotations in bounded ranges and derive a deformed set

$$\partial B_{x,y,o} = \left\{ B_{x',y',o'} \;\middle|\; \begin{array}{ll} x' = x + \delta x \cos o', & \delta x \in \partial x \\ y' = y + \delta y \sin o', & \delta y \in \partial y \\ o' = o + \delta o, & \delta o \in \partial o \end{array} \right\}$$

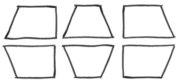
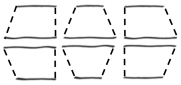
layer ID	Template type	Appearance	Parameters	Deformation Range	Instantiation
$\Delta^{(5)}$, $\Delta^{(4)}$	Circles		center u, v, radius r, # of segments n	$\partial u = \{-.1w, 0, .1w\}$, $\partial v = \{-.1h, 0, .1h\}$, scale: $\{.9\text{x}, 1\text{x}, 1.1\text{x}\}$	$u = w/2, v = h/2$, $d = \min(w, h)$, $r \in \{.9d, 1d, 1.1d\}$
	Trapezoids (rectangles)		Parameters of parallel lines $\{L_1, L_2\}$	above with in plane rotation $\{-\pi/\rho, 0, \pi/\rho\}$	$v_{L_1} = 1/8h, v_{L_2} = 7/8h$, long line length $\iota = 0.9w$, short 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 except no L_3, L_4
$\Delta^{(3)}$, $\Delta^{(2)}$	Line segments		$L = \{$center (u, v) , orientation θ length $l\}$	deformation realized in active curves	u = w/2, v= h/2, $\theta = 0$, $\iota = 0.9w$
	Active Curves		$l = \{$ center (x, y), orientation o, length l $\}$	$\partial x = \{-1, 0, 1\}$pixels, $\partial y = \{-1, 0, 1\}$ pixels, $\partial o = \{-\pi/\rho, 0, \pi/\rho\}$	$(x, y) \in$ image lattice Λ , $o \in O$, $l \in \{1, 2, \cdots, 5\}$
$\Delta^{(1)}$, $\Delta^{(0)}$	Active Basis		$B = \{$center x, y, orientation o, scale $s\}$	same as above	$(x, y) \in \Lambda$, $o \in O$, filter size set to 17×17

TABLE 1
List of visual concepts used in our representation, their parameters, deformation ranges and instantiating ranges. w and h in column 5 and 6 denote the width and height of the panel respectively.

Then $\Delta^{(0)}$ is the union of all these deformed basis functions,

$$\Delta^{(0)} = \cup\, \partial B_{x,y,o}, \quad \text{for} B_{x,y,o} \in \Delta^{(1)}. \tag{10}$$

Each Gabor basis element is the translated and rotated versions of the original 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 is further normalized to have zero mean and unit ℓ_2 norm.

Summary, dictionaries $\Delta^{(k)}, k = 5, 4, 3, 2, 1, 0$ represent the layered and quantized decomposition from the 3D panel (at the leaf-node of G-AoT) to the Gabor functions on pixels. $\Delta^{(5)}$ is decomposed to $\Delta^{(3)}$, and $\Delta^{(3)}$ is further decomposed to $\Delta^{(1)}$. The $\Delta^{(4)}$, $\Delta^{(2)}$ and $\Delta^{(0)}$ are the deformed versions of $\Delta^{(5)}$, $\Delta^{(3)}$ and $\Delta^{(1)}$ respectively.

2.3 AoT, Parse trees and Templates

The AoT specifies a large number of geometric and appearance configurations, most of which are invalid for the car category. The learning process will prune the AoT by removing the unused or less frequently used branches under the Or-nodes, and thus achieve at a parsimonious model – a hierarchical 3D car model. We flatten the model into a 3D template Tpt which is shown in Fig. 3.(e). This Tpt is general enough to account for the variations within the category.

Each realization of the AoT or the Tpt is a parse tree pt. The parse tree is derived by iteratively selecting a branch at each Or-node. At any specific view ω, the 3D object template is further instantiated to a deformable 2D template, by selecting Or-nodes at the A-AoT. Fig. 1(c) shows a few of such 2D templates.

It is also worth noting that as the pruned branches of the AoT are all from the G-AoT, the 3D template Tpt can also be viewed as a parse tree of the G-AoT.

3 Probabilistic Model on Images

In this section, we define a probability model $p(\mathbf{I}|\omega, pt)$ for any image $\mathbf{I}$ given view ω and a parse tree pt. The probability formulation is needed for the 3D template learning presented in the next section.

Let Λ be the domain (i.e. part of the image lattice) occupied by the object. For a parse tree pt, we have a set of n_S deformed shape templates $\{dS_i, i = 1, ..., n_S\} \subset \Delta^{(4)}$ visible for given view ω. So we can further divide Λ into the domains of the visible parts

$$\Lambda = \cup \Lambda_i,\ i = 1, ..., n_S. \tag{11}$$

Each part dS_i is further divided into n_i active curves $\{A_{ij},\ j = 1, ..., n_i\} \subset \Delta^{(2)}$ each occupying a sub-domain Λ_{ij}, thus we have

$$\Lambda_i = \Lambda_{i0}\ \cup\ [\cup \Lambda_{ij}],\ j = 1, ...n_i \tag{12}$$

where Λ_{i0} refers to the empty pixels inside Λ_i but not in the set $\{\Lambda_{ij},\ j = 1, ..., n_i\}$. They correspond to the flat or shading areas in the car. Then each active curve

A_{ij} is divided into n_{ij} Gabor basis functions $\{B_{ijk},\ k = 1, ..., n_{ij}\} \subset \Delta^{(0)}$ with their domains

$$\Lambda_{ij} = \cup \Lambda_{ijk},\ k = 1, ..., n_{ij} \tag{13}$$

Each Λ_{ijk} is an image patch and adjacent patches may overlap slightly.

Let $\overline{\Lambda}$ denote the pixels not covered by the Gabor basis functions. It includes the background and the empty areas $\{\Lambda_{i0}\}$ on the 2D panels. The image is then divided into two components:

$$\mathbf{I} = (\mathbf{I}_{\Lambda},\ \mathbf{I}_{\overline{\Lambda}}).$$

The likelihood for image $\mathbf{I}$, for a view ω and a parse tree pt, is factorized as follows, due to the context free assumptions in the AoT,

$$\begin{aligned} & p(\mathbf{I} \,|\, \omega, pt) = p(\mathbf{I}_{\overline{\Lambda}}, \mathbf{I}_{\Lambda} | \omega, pt) \\ = \ & p(\mathbf{I}_{\overline{\Lambda}}) p(\mathbf{I}_{\Lambda} \,|\, \omega, pt) \\ = \ & p(\mathbf{I}_{\overline{\Lambda}}) \prod_{i,j,k} p(\mathbf{I}_{\Lambda_{ijk}} \,|\, B_{ijk}) \end{aligned} \tag{14}$$

These image patches are conditionally independent after these positions and orientations are decided by the view ω and the deformed 2D template (flattened configuration from the parse tree pt).

Following the active basis template [38] and active curve model [15], we take a reference model $q(\mathbf{I})$ for generic natural images instead of a specific object template. Since the patches are decided by ω, pt, then $q(\mathbf{I}) = q(\mathbf{I}|\omega, pt)$. $q(\mathbf{I})$ is factorized into the product of the patch probabilities $q(\mathbf{I}_{\Lambda_{ijk}})$.

We compute the probability ratio

$$\frac{p(\mathbf{I} \,|\, \omega, pt)}{q(\mathbf{I} \,|\, \omega, pt)} = \frac{\prod_{ijk} p(\mathbf{I}_{\Lambda_{ijk}} \,|\, B_{ijk})}{\prod_{ijk} q(\mathbf{I}_{\Lambda_{ijk}})}. \tag{15}$$

Since $p(\mathbf{I}_{\overline{\Lambda}})$ uses the same background model as $q(\mathbf{I})$, the background probabilities are cancelled in the ratio.

It is worth noting that the empty patches in $\{\Lambda_{i0}, i = 1, ..., n_S\}$ contain useful information against the background. They have near zero Gabor responses and can be used as "negative features" to down-weight cluttered areas (such as trees) to overcome false positives. Furthermore, for the car category, these patches contain important shading information for the body parts. We ignore such patches in the stage of learning the car template, and we will extract features from these patches in the discriminative training stage for improving detection.

As each image patch $\mathbf{I}_{\Lambda_{ijk}}$ is still of high dimensionality, we project it to a one dimensional probability ratio along the response of basis functions B_{ijk}

$$r_{ijk} = \| < \mathbf{I}_{\Lambda_{ijk}}, B_{ijk} > \|^2,$$

and the latter is a one dimensional exponential distribution following the information projection principle [38].

$$\frac{p(\mathbf{I}_{\Lambda_{ijk}} \,|\, B_{ijk})}{q(\mathbf{I}_{\Lambda_{ijk}})} = \frac{p(r_{ijk})}{q(r_{ijk})} = \frac{1}{Z_{ijk}} \exp\{\lambda_{ijk} h(r_{ijk})\}. \tag{16}$$

The above model has four aspects.

- $q(r_{ijk})$ is a histogram of filter responses pooled over a set of natural images. It has high probabilities near zero and has a heavy tail.
- h is a sigmoid transform that saturates the large Gabor filter response to τ:

  $$h(x) = \tau \left[2/(1 + e^{-2x/\tau}) - 1 \right].$$

  It has high filter response when the patch coincides with an edge/bar feature in the image.
- λ_{ijk} reflects the importance of the corresponding active basis element in the learned template, and should be estimated so that the expectation $E_{\lambda}[h(r_{ijk})]$ matches the corresponding observed mean response from training images. If training images have a common edge or bar at patch Λ_{ijk}, then it is a salient feature against the background, and thus λ_{ijk} is higher. To simplify the learning approach below, we skip this estimation step and instead use a constant λ for all the basis functions in our template. This parameter will be adjusted later in the discriminative training step.
- Z_{ijk} can be computed using numerical integration to normalize the 1D probability $p(r_{ijk}) = \frac{1}{Z_{ijk}} q(r_{ijk}) \exp\{\lambda_{ijk} h(r_{ijk})\}$.

In summary, the log-probability ratio of a car against the background is

$$\log \frac{p(\mathbf{I} \,|\, \omega, pt)}{q(\mathbf{I} \,|\, \omega, pt)} = \sum_{ijk} \lambda_{ijk} h(r_{ijk}) - \log Z_{ijk}, \tag{17}$$

and the corresponding ratios conditioned on And and Or-nodes along the pt can be defined on its descendant terminal nodes respectively.

4 Learning the 3D Car Template

In this section, we present an algorithm for learning the car template Tpt from a set of training images which are annotated with views and assumed to be drawn from a distribution f.

$$\{(\mathbf{I}^{(m)}, \omega^{(m)}),\ m = 1, 2, ..., M\} \sim f(\mathbf{I}, \omega).$$

4.1 Maximum Information Gain

Learning the 3D template Tpt is performed in a generative framework. The objective is to learn the probability $p(\mathbf{I}, \omega|\mathrm{Tpt})$ to approximate an underlying true model $f(\mathbf{I}, \omega)$. It starts from the reference model q and maximizes the reduction of Kullback-Leibler divergence,

$$\mathrm{Tpt}^* = \arg\max KL(f||p) - KL(f||q) \tag{18}$$

$$= \arg\max KL(p||q) \tag{19}$$

$$\approx \arg\max \sum_{m=1}^{M} \log \frac{p(\mathbf{I}^{(m)} \,|\, \omega^{(m)}, \mathrm{Tpt})}{q(\mathbf{I}^{(m)} | \omega^{(m)}, \mathrm{Tpt})} \tag{20}$$

Eqn. (19) follows the Pythagorean theorem for the exponential family of models, and Eqn. (20) replaces the

expectation by sample mean and an assumption of uniform distribution on the view ω.

The target function in Eqn. (20) is called the "information gain" $\mathrm{IG}(\mathrm{Tpt})$ of Tpt. As the Tpt is a parse tree in G-AoT, this term is equal to the information gain of the root node of the AoT.

Thanks to the AoT structure and the context-free (or equivalently the conditional independence) assumption, this maximum information gain can be unfolded by the log-probability ratio in Eqn. (17) to four layers of summation over indices m, i, j, k, so that it can be computed recursively by a dynamic programming algorithm that finds the best IG of all Tpt as well as the global optimal templateTpt.

For each And-node A of the G-AoT, its information gain is the sum of its children nodes in $ch(A)$,

$$\mathrm{IG}(A) = \sum_{B \in ch(A)} \mathrm{IG}(B). \tag{21}$$

For each Or-node B of the G-AoT, its information gain is the maximum of its children nodes.

$$\mathrm{IG}(B) = \max_{A \in ch(B)} \mathrm{IG}(A). \tag{22}$$

Eqn. (21) and (22) constructs a recursion, which stops at leaf-nodes of G-AoT. The information gain of each shape template on the leaf-nodes can then be computed in a similar recursion on the A-AoT, which results in a summation over Gabor element responses as:

$$\mathrm{IG}(S_i) = \sum_{m} \sum_{jk} \lambda_{ijk} h(r_{ijk}^{(m)}) - \log Z_{ijk}. \tag{23}$$

Once the information gain is computed for every node in the AoT, we can search for the optimal template Tpt that has the highest information gain $\mathrm{IG}(\mathrm{Tpt})$ by back tracking the best child node of each Or-node in Eqn.(22).

Since the recursion alternates between layers of And and Or-nodes, we call such an algorithm the And-Or search. The recursion also keeps decomposing the problems into overlapping sub-problems, so the And-Or search algorithm is a case of the dynamic programming algorithm.

It is also worth noting an important fact. If our objective is to compute multiple templates for modeling structural variations, such as different types of vehicles (truck, bus, van, convertibles), the solution will be a pruned AoT which can derive multiple templates. This can be done recursively by an Inside-outside algorithm which guarantees a local optimal solution instead of a global one. This method was reported in scene modeling [36], [37].

4.2 And-Or Search Algorithm

Converting the recursions into iterations, the final And-Or Search algorithm consists of one bottom-up pass and one top-down pass. The bottom-up pass computes the information gains at the leaf-nodes and then uses the sum and max operations at And and Or-nodes recursively to compute optimal information gain of all nodes in the AoT. The top-down pass is a series of arg-max operations that retrieves the optimal template Tpt as well as its deformations on each training image.

The bottom-up pass starts from computing the log-likelihood ratios of active basis and active curves in $\Delta^{(0)}$ to $\Delta^{(3)}$. As these nodes in A-AoTs are numerous, we choose to generate these nodes online, only when their ancestor shape templates S are being visited. Correspondingly, we choose to densely pre-compute the likelihood ratios of these active basis and line segments in parallel and save them in the form of score maps, which can be used for fast retrieval by the projected L in S. Details of these sum-max operations can be found in [14] as the steps of computing $\mathbf{S1}, \mathbf{M1}, \mathbf{S2}, \mathbf{M2}$ maps.

The bottom-up pass then continues to compute the information gains of the shape templates, which consists of a series of sum for likelihood ratio of individual deformed shape templates on each image, max over deformations and then another sum over different images $\mathbf{I}_m$.

After the information gains for each shape template, a.k.a, leaf-node of G-AoT are computed in the form of Eqn.(23), the bottom-up pass goes to G-AoT and the information gains are computed according to Eqn.(21) and (22). In implementing the And-Or search in G-AoT, we need to decide the visiting order of these nodes so that children nodes are processed before their parents. We simply assign the height of the And and leaf-nodes to be $2v$ and that of Or-nodes to be $2v+1$, where v is the size of the volume. As operations within each layer of the tree can be computed independently, they can be done in parallel, which makes the algorithm more efficient.

After the information gain for the root node is computed, we simply trace back from the top of the tree, repeatedly compute arg-max on each selected Or-node, record its selected child node, and further goes down until individual Gabor filters on each image are selected. Along the hierarchy, the recorded volumes in leaf-nodes of G-AoT constructs a partition of the object volume, the shape templates S form a 3D car template, and the individual Gabor filters render deformed templates for different car images.

5 Inference Algorithm

We solve for a joint inference task: i) Detection by searching the domain Λ with maximum score; ii) View/pose-estimation by searching all quantized views and camera distance; iii) Part localization by finding the parts in the templates. This is implemented by a dynamic programming method illustrated in Fig. 7.

5.1 Projecting 3D Template to 2D Image Plane

On a testing image $\mathbf{I}$, we project the 3D template into 2D templates by specifying a set of views ω, and perform the

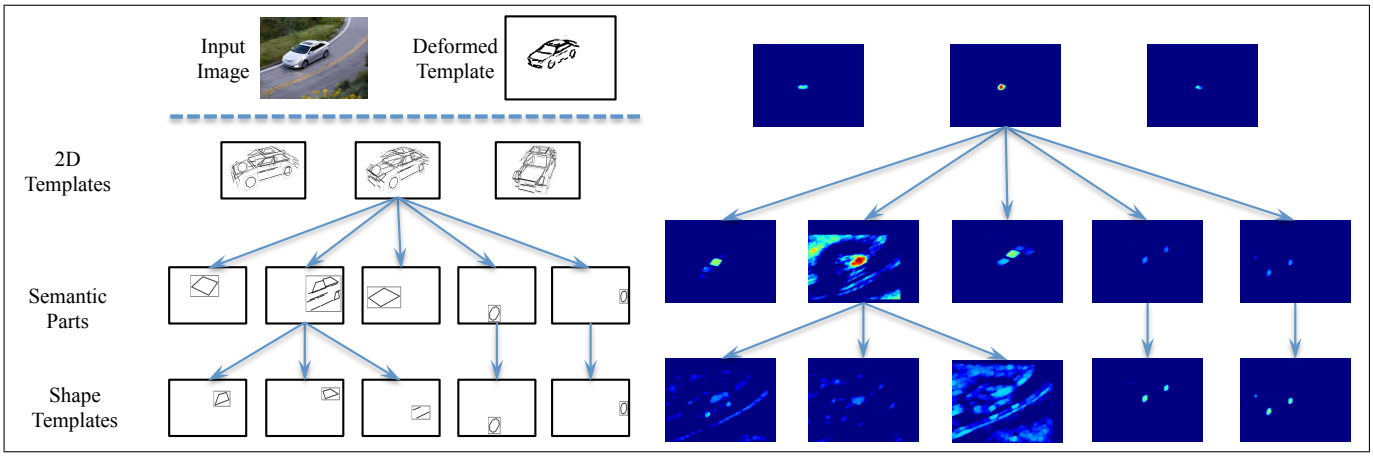


Fig. 7. The inference process. (Left) A few nodes in the template Tpt hierarchy. (Right) Score maps for each node on the left. The score maps are normalized such that intensities are only comparable on score maps in the same layer.

inference task by running sliding windows using these 2D templates.

The view vector $\omega = (\xi, \eta, d)$ includes three variables – pan, tilt and camera distance respectively. 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 origin of the world frame. For simplicity, roll angle of the camera is set to zero. More specifically, in experiments, we search pan angle at $15°$ interval in $[0°, 360°)$, tilt angle at $5°$ interval from $[5°, 90°)$, and 18 camera distances for each pan and tilt angle combination to account for image scaling and the perspective projection effects.

For a view ω, the 3D template Tpt is projected to a 2D deformable template in the way discussed in Section 2, with each 3D in-plane deformation of each shape template realized by a different set of the active curves. For each $\omega = (\xi, \eta, d)$, we assume the camera is looking at object center, and generate a 2D deformable template. We then run sliding window using this 2D deformable template over the testing image.

For each window, we start from a generic model $p_0 = q(\mathbf{I}|\omega, pt)$ as the current interpretation, and then maximizing the posterior probability which is equivalent to maximizing the log-likelihood ratio as we assume a uniform distribution on the discretized views and parse trees.

$$\begin{aligned}(\omega, pt)^* &= \arg\max p(\omega, pt|\mathbf{I}) \\ &= \arg\max \log \frac{p(\mathbf{I}\,|\,\omega, pt)}{q(\mathbf{I}\,|\,\omega, pt)}\end{aligned} \tag{24}$$

Following Eqn.(17), we derive the score function,

$$\begin{aligned}\mathrm{Score}(\omega, pt) &= \log \frac{p(\mathbf{I}\,|\,\omega, pt)}{q(\mathbf{I}\,|\,\omega, pt)} \\ &= \sum_{ijk} \lambda_{ijk} h(r_{ijk}) - \log Z_{ijk}.\end{aligned} \tag{25}$$

Similar to the optimization problem in learning, the best score can also be found using the dynamic programming. Fig. 7 illustrates a few nodes in the template Tpt on the left side, and their corresponding score maps on the right side.

Though all three views on the top row have high scores at the object location, the peak at the correct view has a significantly higher score than others. The meaningful shape templates, such as these for the windshield and wheels, are also highly salient in their score maps.

5.2 Discriminative Retraining

The model discussed so far is fully generative and parsimonious in the sense that it only extracts the sketches at sparse locations and the parameters λ_{ijk} are shared across views. This model is sufficient for learning the template Tpt. In the detection task, we further retrain the score function by adding new features and retrain the feature weights through some common discriminative steps so that its performance can be compared against the discriminatively trained models like DPM.

1, Adding negative features in empty areas. Recall that in Eqn.(12) each part has area Λ_{i0} unoccupied by the basis functions and have flat/smooth shading pattern that can be used to down-weight clutter as "negative features". We evenly sample points in Λ_{i0} on the 2D panel of Tpt, and extract Gabor responses at projected positions and orientations and concatenate them in a vector $\phi(\mathbf{I}_{\Lambda_{i0}})$. As their values are near zero, we do not use the sigmoid function $h()$ for simplicity. These features are added to the score function for linear SVM training.

So we rewrite the score function as,

$$\mathrm{Score}'(\omega, pt) = \sum_{ijk} \tilde{\lambda}_{ijk} h(r_{ijk}) + \sum_{i} < \lambda_{i0}, \phi(\mathbf{I}_{\Lambda_{i0}}) > .$$

Though it is more elegant to directly include these features into our 3D template learning formulation, we feel this is not necessary as it will significantly increase the computational complexity and the cluttered background is not a problem as the view labels are already given in training stage. Our choice will also reduce the complexity in inference step as computing the deformation of

these dense features will be slower than only extracting them after the the deformation is determined by the sketch information.

2, *Discriminative regression on parameters.* Our reference model $q()$ represent the background and is pooled from generic natural images. From a discriminative perspective, $q()$ governs the negative samples, and should be re-calibrated to compensate the error introduced by position and orientation invariant assumption in the active basis model. This leads to adjusted weights on the scores of active curves. To this aim, we use SVM to re-train the weights $\tilde{\lambda}_{ijk}$ and vectors λ_{i0} in the adjusted score function $\text{Score}'(\omega, pt)$.

3, *Hard negative training.* We collect negative training examples for the SVM reweighing in the way similar to the hard negative mining steps in deformable part based model [7]. We use the equally weighted original model to run sliding windows on positive examples, crop high score windows whose intersection over union with ground truth window are less than 50 percent, and extract both sketch features and negative features from these windows as negative training data. Note that we only perform one round of negative data collection, where in DPM, this step is repeated each time a new weighting of the model features are learned.

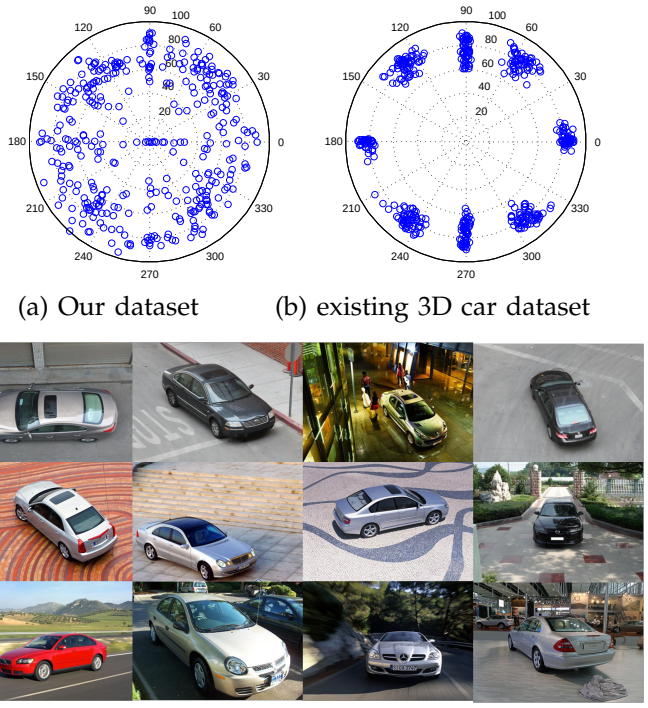
In implementation, we proceed in two steps for speed considerations. Step 1. We compute the score maps for the sketch-only (with the first term) at enumerated views, we select the top 100 highest score windows as seed windows in each view. In Step 2, we update the score of these windows using both sketch and the negative features. The object detection windows are then reported using non-maximum suppression through these seed windows by the criterion that reported windows should not overlap more than 50 percent with each other.

6 Experiments

In this section, we compare 3D car templates learned from different datasets in joint inference tasks: car detection, view and 3D pose estimation, and part localization. We also study the capacity and efficiency of the AoT representation and evaluate how the 3D car template can generalize to novel views.

6.1 Image Datasets and Parameter Settings

In the literature, there are a few widely used datasets emphasizing 3D object recognition ([21], [31]), but they only provide images from a few specified views or limited ranges of views. For example, the widely used 3D car dataset in [31] contains cars whose views are shown in Fig. 8(b). As the images are essentially clustered around 8 views, they cast the task to a multi-class object recognition problem and have saturated performance with average precision (detection) to be over 99%. Such datasets are not particularly suitable for evaluating 3D modeling and parsing.



(a) Our dataset (b) existing 3D car dataset

(c) Examples from our dataset

Fig. 8. (a) View distribution of our dataset; (b) View distribution of the 3D car dataset in [31]. The angular direction represents pan angle and radius direction represents tilt angle. (c) Sample images of our dataset.

We introduce a new dataset of car images. These images are collected in uncontrolled environments from the Internet and at the intersections and parking lots on UCLA campus. Fig. 8.(c) shows some examples. The images are taken from different distance and thus may have perspective projection (foreshortening) effects. For each image, we label the object views using an annotation tool (public code released by the authors [15]). Fig. 8(a) shows the view distribution of our datasets, which are evenly distributed over the viewing hemisphere. We also labeled the contours of objects and their semantic parts, so the dataset can be used to evaluate on other tasks, such as part localization and segmentation. More details of the image and annotations can be found in the dataset webpage [1].

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 17×17 in pixels, and basis response saturation threshold τ to 6, which is the same as in active basis model [38]. The range of λ in [38] is $[0, 5]$ with 5 corresponding to the highest possible expected response. We set it to 2.0 which corresponds to about 90% of the highest expected response. Experiment using different values of λ shows that the learned template does not change much when λ is in range $[2.0, 3.0]$. These parameters are fixed for all

1. http://www.stat.ucla.edu/~wzhu/3DAoT

the experiments.

6.2 The capacity of AoT

Volume Size	V^{A}	V^{O}	leaf	# of Tpt
$(5,5,5)$	1,170	270	10,206	2,7969 $\times 10^{6}$
$(8,8,8)$	99,684	11,340	394,065	7.2096 $\times 10^{25}$
$(10,10,10)$	628,540	55,440	1,967,328	2.2061 $\times 10^{48}$

TABLE 2
Capacity of the G-AoT.

In the first experiment, we study the capacity of the G-AoT in representing different 3D object shapes. For this purpose, we start with one volume at the root of the G-AoT and grow the G-AoT using the rules in Section 2.1 as well as the parameters specified above. The size of the volume is shown in the first column of Table 2. We count the number of And, Or and leaf-nodes in G-AoT, and compute the number of possible parse trees corresponding to possible 3D object templates Tpt inside the volume. 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 2. Due to the And-Or structure, the number of possible 3D templates are exponentially larger than the number of nodes in AoTs at every volume size. This demonstrates the expressive power of the AoT representation for quantizing a large space by using only a much smaller number of nodes.

6.3 Representation Efficiency of G-AoT

In the literature of image coding and compression, the representational efficiency of a coding scheme and dictionary is usually evaluated by the rate-distortion curve, which computes how fast the approximation error decreases as the coding length (i.e. number of coding elements) increases. One good example is the beamlet [16] which evaluates the efficiency of the beamlets dictionary (line segments quantized in a square) to approximate 2D curves. Following such methods, we evaluate the effectiveness of the G-AoT for quantizing 3D vehicle shapes.

We collected 20 3D CAD models for four vehicle categories: sedan, SUV, truck and mini-van from the SketchUp 3D warehouse, and test how well the G-AoT can approximate these 3D shapes in coarse-to-fine levels as we reduce the sizes of the atomic volume in the leaf-nodes of the G-AoT. We compare the G-AoT against the popular 3D octree as a baseline, which, in contrast to allowing multiple possible volume splits by Or-nodes, simply partitions the parent 3D cuboid into 8 equally sized children cubes. Within the leaf volume, we fit the 2D panel to the corresponding panel in the CAD model. The error is defined by the average distance between the panel and facet vertices.

We learn the best 3D panel composition using the And-Or search with the information gain of each template replaced by the total area of facets within a certain distance threshold ($\leq$ 1 inch) from the corresponding panel.

Fig. 9 plots the proportion of areas with error under 1 inch as the unit size increase (fine-to-coarse) from 100 millimeters to 400 millimeters. It is clear to see that in all cases, the AoT explains more area than the octree, which suggests the best template encoded by AoT is always closer to the true CAD models than that of the octree. Fig. 10 shows a fine-to-coarse approximation to a minivan shape by the G-AoT learning process. This illustrates the errors underlying the curves in Fig. 9.

6.4 Learning Object Templates

In the following experiments, we randomly pick 160 of the 360 images dataset as training examples, and the remaining 200 as testing examples. Fig. 1 and Fig. 3.(e) show the learned template for car images, which takes an i7 quad-core machine about 40 minutes. The initial volumes in Fig. 3.(a) are decomposed, through the learning process, into clearly interpretable parts and 2D panels for wheels, doors, windows, headlights and grills. The combination of these individual shape templates form a car shape, which demonstrates that the 3D templates represented by AoT include meaningful ones, and they can be searched through by the proposed algorithm. Deformed templates in Fig. 1 also demonstrate that the proposed hierarchies of deformations can adapt the shape templates to its variants observed in images.

6.5 Detection and View Estimation

We report the detection and view estimation results on two datasets.

On the 3D car dataset in [31], the detection is measured by the average precision of the bounding box and the pose estimation is measured by the MPPE (Mean Precision of Pose Estimation). Table 3 shows the results by a range of methods on this dataset. Our model achieves comparable performance on detection and pose estimation. As we mentioned before, the images in this dataset are clustered in 8 distinct views and can be resolved by 2D multi-class detection methods. This observation is confirmed by the fact that latest 2D view-based DPM model achieved saturated performance. So it is no longer suitable for 3D car benchmark.

On our newly collected dataset, we run the inference steps in Section 5 to perform object detection. We use the released version 5 of DPM model as a baseline for object detection on this dataset for two reasons: i) it represents state-of-the-art results for object detection on various categories including the 3D car dataset above; and ii) there is no publicly available code from other

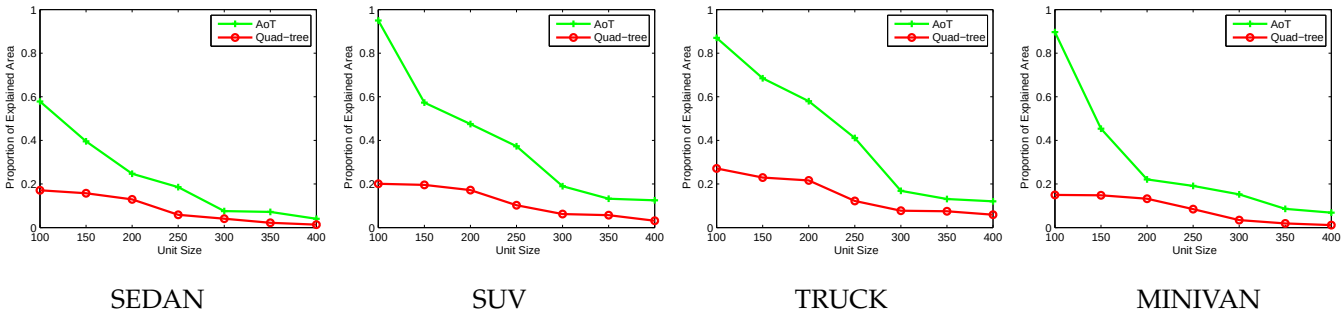


Fig. 9. Comparison of representation efficiency between the G-AoT and commonly used Octree. Horizontal axis represents the side length of minimum volumes in millimeters from fine-to-coarse, and the vertical axis represents the percentage of CAD model surfaces inside VOI and represented by the computed solution. The And-Or tree outperforms Octree consistently across all the four vehicle categories at all granularity.

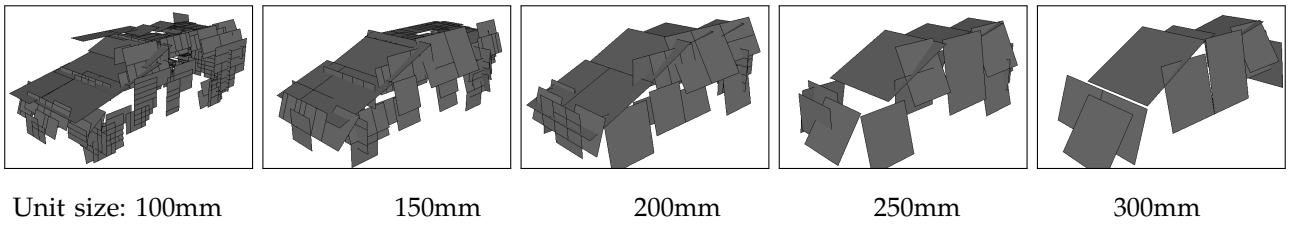


Fig. 10. Fine-to-coarse approximation to a minivan shape by the learned 3D template from G-AoT.

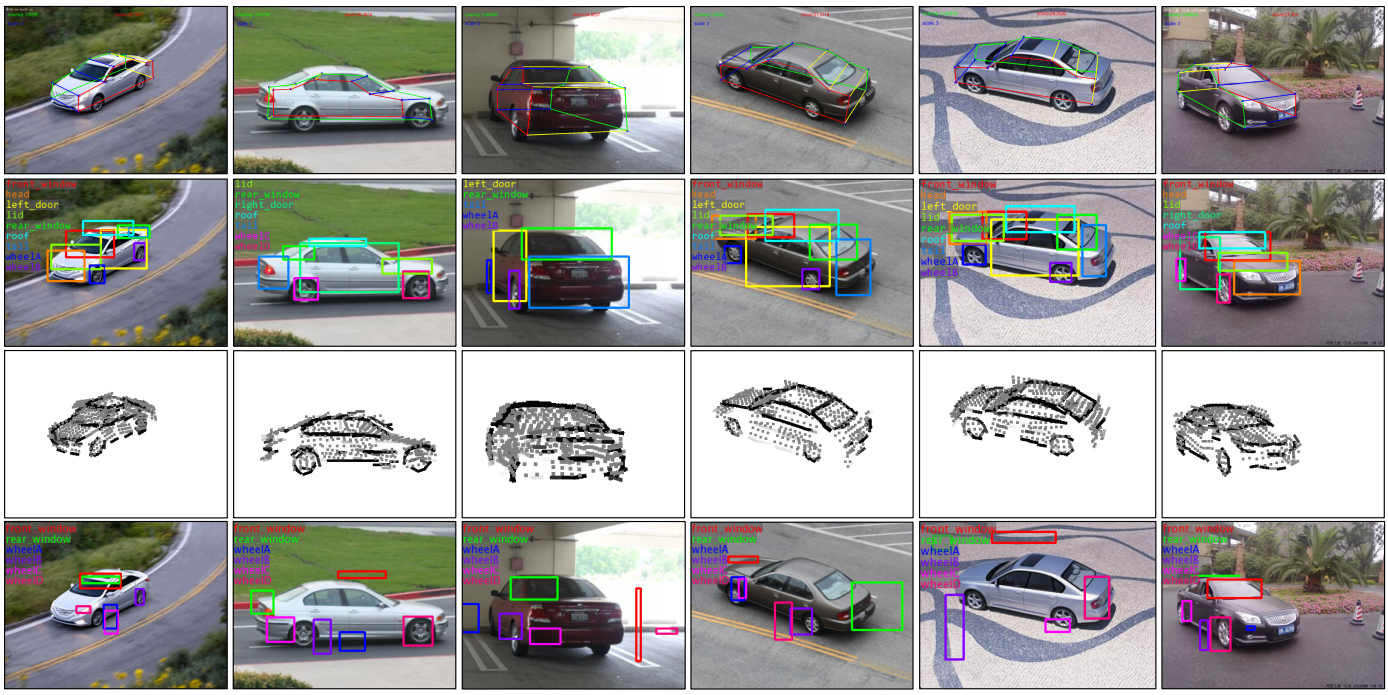


Fig. 11. Sample experiment results. Row 1: The 3D wire frame showing the detected cars and their estimated poses. Row 2: Semantic part localization results by our method. Row 3: The deformed templates for each detection. Dots in templates show the positions of sampled patches for negative features. Row 4: The semantic part localization results of the baseline method using the DPM model (release 5).

Method	DPM	[25]	[23]	[10]	[29][1]	[29][2]	ours
AP	99.6	96	76.7	99.2	99.9	99.7	99.4
MPPE	86.3	89	70	85.3	97.9	96.3	94.2

TABLE 3
Performance comparison of 2D car detection and pose estimation tasks in terms of AP and MPPE on the 3D Car dataset [31]. [29][1] and [29][2] refer to DPM-VOC+VP and DPM-3D-Constraints respectively.

Fig. 12. Some failure examples with templates imposed on testing images. (Left) Close camera position our of the computed range; (Middle) The template matches better on part of the object image area that resembles another view of the car; (Right) Background structure resembling car.

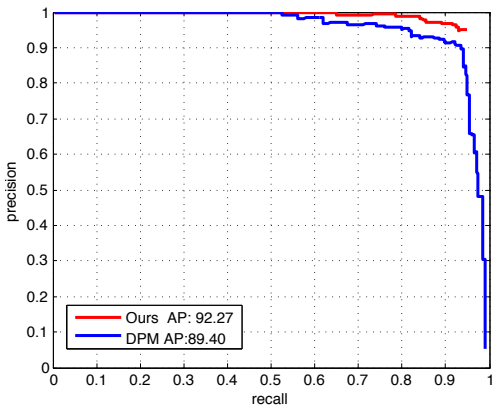


Fig. 13. Object detection performance on our dataset.

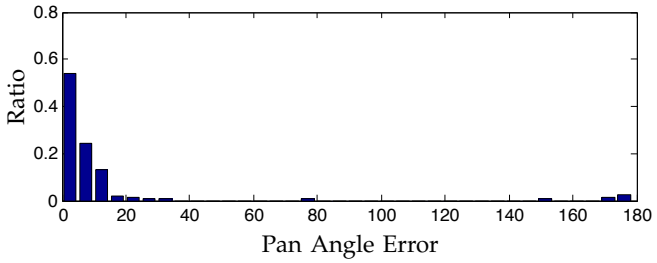


Fig. 14. Pose estimation error on our dataset. The horizontal axis is the error of Pan angle, and the vertical axis is the percentage of images falling in each error bin.

3D car recognition method, and therefore we cannot test their method on this dataset.

Fig. 7 illustrates the detection process by dynamic programming. Fig. 11 shows more examples of detection at various poses and scales. Our model can also be used to estimated view angles of the detected cars, which are more informative than most of the recent models which only report view category labels. For example, using the reported view angles, we can directly project a 3D wireframe onto the image in row 1.

Fig. 12 shows some failure results of our method. The reasons of these failures can be attributed to the following factors. i) The hypothesized camera focal length is too far away from the actual value and thus cannot model the severe perspective effects in very close views (left column); ii) View confusion – the template matches better on part of the object image and that part of image resembles another view of a car (middle). iii) structured background matches the template better (right).

To quantitatively evaluate the performance of our model on object detection, we compute the precision recall curves on our dataset. We use a common protocol used in the detection literature. That is, a detection is considered correct if the intersection of the object detection window and the human annotated window is larger than half of their union area. We compare against the version 5 of the DPM model, by converting our dataset annotation to the VOC format, and directly using code from [9] on the converted dataset. We train 5 DPM models with 1 to 5 mixture components respectively, and select to show the precision recall curve with the best average precision in Fig. 7. We report all the average precisions of the 5 DPM models in Table 4. The experiment shows our model performs slightly better than the DPM model in the detection task.

TABLE 4
Object detection performance of the DPM model with different number of components

# DPM components	1	2	3	4	5
Avearge precision	0.8534	0.8865	**0.8939**	0.8798	0.8825

For correctly detected instances, we also plot the histogram of view estimation errors on pan angles, which is shown on Figure 14. From the plot, we can see that majority of the instances are detected at the correct angles. We notice that a few of the estimates are totally flipped from head to tail, which 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. We did not compare with the DPM model on this task, as it does not have the view angle output.

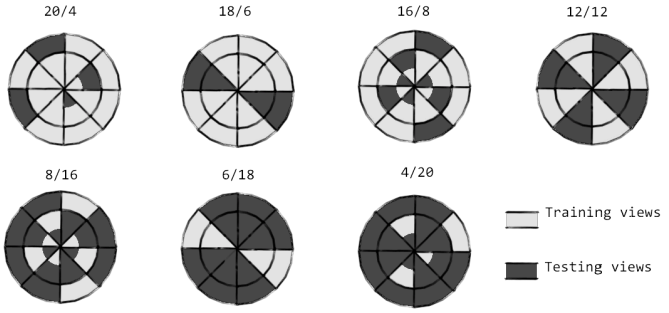
6.6 View Generalization

In early theory, one benefit of object-centered model over viewer-centered model is that the 3D model can

TABLE 6
Part localization performance in term of detection rate, where numbers in italic type are the best among the baseline models.

	Windshield	Rear Window	Frontal left wheel	Back left wheel	Back right wheel	Frontal right wheel
Ours	**0.8125**	**0.6000**	**0.5625**	**0.5077**	**0.5079**	**0.5152**
Baseline, 1 CPN	*0.4688*	*0.2375*	0.1194	0.0882	0.1884	0.0845
Baseline, 2 CPN	0.3333	0.125	0.0896	0.0588	0.0725	0.1111
Baseline, 3 CPN	0.4385	0.0875	0.1045	*0.4412*	*0.3333*	*0.1528*
Baseline, 4 CPN	0.2061	0.0741	0.1343	0.1324	0.0725	0.1389
Baseline, 5 CPN	0.3588	0.1852	*0.1642*	0.2941	0.1884	0.1389

TABLE 5
Results from view generalization experiment



# of train /test views	20/4	18/6	16/8	12/12	8/16	6/18	4/20
Ours	0.9102	**0.9513**	**0.9138**	**0.9289**	**0.8465**	0.6699	0.6152
DPM	**0.9133**	0.9011	0.8597	0.8772	0.8081	**0.6943**	**0.6436**

be easily generalized to novel views unseen in training. To show the view generalization ability of our model, we uniformly partition the viewing hemisphere into 24 view bins (8 along the pan angle multiplied by 3 along tilt angle). In each trial, we learn a 3D car template from images in a selected set of bins, and test on images in the remaining view bins. Table 5 shows the 7 trials with different partition of views for training and testing. For comparison, we train the DPM model with 3 components and compare the average precisions on the bottom of the same table.

The results can be interpreted in 3 groups. i) When the number of training views is large, the DPM and our model performs comparably, which is consistent with our results in detection experiment in section 6.5. ii) When the number of training and testing views are comparable, our model outperforms the DPM model. iii) When the training views are much fewer than testing views, the DPM outperforms our model (the margin is smaller than we had in the case ii). We find this is because our algorithm fails to learn a meaningful 3D model while the hard negative mining step in the DPM model makes it a better non-background classifier.

6.7 Part Localization

By retrieving the template deformations, our model also estimates the rough boundaries of semantic object parts. Locations and sizes of these object parts are useful for various applications: i) In scene parsing and event understanding, one often needs to describe the relationships between humans and vehicles, e.g. a person entering the car from the driver seat will be identified as the driver. ii) For fine-grained vehicle recognition, we can use them to re-identify a car instance in different camera views, or recognizing the make and model; and iii) The localizations of parts are needed for identifying the damages of parts for insurance agents.

As the baseline DPM model was not trained for car parts, a direct comparison against the DPM model is not meaningful. Similar to the way that DPM model refines object bounding boxes by its part locations, we extend the DPM model to predict the locations of semantic parts. The extension is based on linear regression, where the DPM part bounding boxes are predictors, and our semantic part bounding boxes are outcomes. We train one regression model for each DPM mixture component, and use the corresponding model to predict semantic parts when the detection is activated by the component. Fig. 11 shows the semantic part localization results from our model (row 2) and the baseline DPM method with 3 components (row 4) where the part windows are predicted from DPM parts. It can be seen that our results are much better.

It can also be seen in columns 4 and 6 of Fig. 11 that the baseline method predicted positions of the invisible wheels to the visible side of the car. This is because in the training stage the DPM model misclassified some cars into its flipped poses, and their bounding box data are then mixed together in fitting the regression model.

To quantitatively evaluate the part localization performance, we compute the detection rate of these semantic parts on the correctly detected cars, and show the numbers of our model against the baseline model using 1-5 components in Table 6. It can be seen that our model significantly outperforms the baseline method. In our method, the detection rates for the windshield and rear window are significantly higher than the wheels, we believe this is because these parts are closer to the object

center, so are less sensitive to quantization error of the search grid.

7 Discussion

In this paper, we present a 3D And-Or tree structure for quantizing the geometry and appearance space and for learning 3D car templates to solve a few vision tasks jointly. The learned template is a hierarchical and compositional model integrating the 3D shape-based representation in a G-AoT and appearance-based model in the A-AoT. In experiments we demonstrate that the proposed method can learn meaningful 3D car templates from view labeled images, and yield better performance in object detection, pose estimation, and semantic part localization than the most recent DPM model which is 2D view-based.

There are a few issues that are worth further study in future research.

1) The hierarchical representation can be augmented to represent object attributes, such as the make and model of the vehicle, for fine-grained object recognition and parsing.
2) We shall learn shared 3D parts across multi-categories, as it was done in the unsupervised learning of 2D AoT [8], [32] and extend the framework to representing articulated 3D objects.
3) In applications such as video surveillance, cars often appear with external occlusions. A preliminary study of the And-Or Tree representation for occlusion is presented in [22], and it is of our interest to understand how occlusion can be modeled together with other factors such car types mentioned above.

   The current method performs object detection by performing top-down projection of templates into individual views, which is much slower than the view-based method such as DPM. To reduce the number of projections, we shall further analyze our 3D model and partition the views in coarse-to-fine and replace the pure top-down inference be replaced by combining top-down and bottom-up inference.

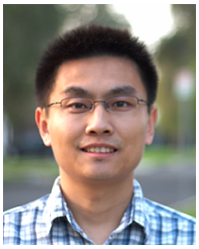
Acknowledgement

This project is supported by NSF IIS 1018751, ONR MURI N00014-10-1-0933, DARPA MSEE grant FA8650-11-1-7149 and NSF DMS 1007889. The author would also like to thank Dr. Yingnian Wu, Tianfu Wu, and Brandon Rothrock for insightful suggestions.

References

[1] M. Arie-Nachimson and R. Basri, "Constructing implicit 3d shape models for pose estimation," in *Int'l Conf. Comput. Vis. (ICCV)*, 2009.
[2] I. Biederman, "Recognition-by-components: A theory of human image understanding," *J. Psychol. Rev.*, vol. 94, pp. 115–117, 1987.
[3] T. O. Binford, "Visual perception by computer." in *Conf. Syst. Control*, 1971.
[4] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *ECCV*, 2004.
[5] J. G. Daugman, "Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters," *J. Opt. Soc. Am. A*, vol. 2, no. 7, pp. 1160–1169, Jul 1985.
[6] S. J. Dickinson, A. P. Pentland, and A. Rosenfeld, "From volumes to views: An approach to 3-d object recognition," *Comput. Vis. Image Und.*, vol. 55, no. 2, pp. 130 – 154, 1992.
[7] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, 2010.
[8] S. Fidler and A. Leonardis, "Towards scalable representations of object categories: Learning a hierarchy of parts," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2007.
[9] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester, "Discriminatively trained deformable part models, release 5," http://people.cs.uchicago.edu/ rbg/latent-release5/.
[10] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich, "Viewpoint-aware object detection and pose estimation," in *Int'l Conf. Comput. Vis. (ICCV)*, 2011.
[11] M. Hejrati and D. Ramanan, "Analyzing 3d objects in cluttered images," in *Neural Info. Proc. Systems (NIPS)*, 2012.
[12] E. Hsiao, A. Collet Romea, and M. Hebert, "Making specific features less discriminative to improve point-based 3d object recognition," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2010.
[13] W. Hu, "Learning 3d object templates by hierarchical quantization of geometry and appearance spaces," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2012.
[14] W. Hu, Y. N. Wu, and S.-C. Zhu, "Image representation by active curves," in *Int'l Conf. Comput. Vis. (ICCV)*, 2011.
[15] W. Hu and S.-C. Zhu, "Learning a probabilistic model mixing 3d and 2d primitives for view invariant object recognition," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2010.
[16] X. Huo and J. Chen, "JBEAM: multiscale curve coding via beamlets," *IEEE Trans. Image Process.*, vol. 14, pp. 1665–1677, 2005.
[17] J. Koenderink and A. Doorn, "The singularities of the visual mapping," *Biol. Cybern.*, vol. 24, no. 1, pp. 51–59, 1976.
[18] ——, "The internal representation of solid shape with respect to vision," *Biol. Cybern.*, vol. 32, no. 4, pp. 211–216, 1979.
[19] A. Kushal, C. Schmid, and J. Ponce, "Flexible object models for category-level 3d object recognition," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2007.
[20] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2006.
[21] B. Leibe and B. Schiele, "Analyzing appearance and contour based methods for object categorization," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2003.
[22] B. Li, W. Hu, T. Wu, and S.-C. Zhu, "Modeling occlusion by discriminative and-or structures," in *Int'l Conf. Comput. Vis. (ICCV)*, 2013.
[23] J. Liebelt and C. Schmid, "Multi-view object class detection with a 3D geometric model," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2010.
[24] J. Liebelt, C. Schmid, and K. Schertler, "Viewpoint independent object class detection using 3d feature maps," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2008.
[25] R. Lopez-Sastre, T. Tuytelaars, and S. Savarese, "Deformable part models revisited: A performance evaluation for object category pose estimation," in *ICCV-WS CORP*, 2011.
[26] D. G. Lowe, "Object recognition from local scale-invariant features," in *Int'l Conf. Comput. Vis. (ICCV)*, 1999.
[27] D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W.H. Freeman, 1982.
[28] N. Payet and S. Todorovic, "From contours to 3d object detection and pose estimation," in *Int'l Conf. Comput. Vis. (ICCV)*, 2011.
[29] B. Pepik, M. Stark, P. Gehler, and B. Schiele, "Teaching 3d geometry to deformable part models," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2012.
[30] R. U. Sanja Fidler, Sven Dickinson, "3d object detection and 3d object detection and viewpoint estimation with a deformable 3d cuboid model," in *Neural Info. Proc. Systems (NIPS)*, 2012.

[31] S. Savarese and L. Fei-Fei, "3d generic object categorization, localization and pose estimation." in *Int'l Conf. Comput. Vis. (ICCV)*, 2007.
[32] Z. Si and S.-C. Zhu, "Learning and-or templates for object recognition and detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 9, pp. 2189–2205, 2013.
[33] H. Su, M. Sun, F.-F. Li, and S. Savarese, "Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories." in *Int'l Conf. Comput. Vis. (ICCV)*, 2009.
[34] D. S. Taubman, M. W. Marcellin, and M. Rabbani, "JPEG2000: Image Compression Fundamentals, Standards and Practice," *J. Electron. Imaging*, vol. 11, pp. 286–287, 2002.
[35] A. Thomas, V. Ferrar, B. Leibe, T. Tuytelaars, B. Schiel, and L. Van Gool, "Towards multi-view object class detection," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2006.
[36] S. Wang, J. Joo, Y. Wang, and S.-C. Zhu, "Weakly supervised learning for attribute localization in outdoor scenes," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, 2013.
[37] S. Wang, Y. Wang, and S.-C. Zhu, "Hierarchical space tiling in scene modeling," in *Asia Conf. on Comput. Vis. (ACCV)*, 2012.
[38] Y. N. Wu, Z. Si, H. Gong, and S.-C. Zhu, "Learning active basis model for object detection and recognition," *Int. J. Comput. Vision*, vol. 90, no. 2, pp. 198–235, Nov. 2010.
[39] Y. Xiang and S. Savarese, "Estimating the aspect layout of object categories," in *IEEE Conf. Comput. Vis. Pattern. Recognit. (CVPR)*, June 2012.
[40] P. Yan, S. M. Khan, and M. Shah, "3d model based object class detection in an arbitrary view," in *Int'l Conf. Comput. Vis. (ICCV)*, 2007.
[41] L. L. Zhu, C. Lin, H. Huang, Y. Chen, and A. L. Yuille, "Unsupervised Structure Learning: Hierarchical Recursive Composition, Suspicious Coincidence and Competitive Exclusion," in *ECCV*, 2008.
[42] S.-C. Zhu and D. Mumford, "A stochastic grammar of images," *Found. Trends. Comput. Graph. Vis.*, vol. 2, no. 4, pp. 259–362, 2006.
[43] M. Zia, M. Stark, B. Schiele, and K. Schindler, "Revisiting 3d geometric models for accurate object shape and pose," in *ICCV-WS 3dRR*, 2011, pp. 569–576.



Wenze Hu received a PhD degree from Department of Statistics, University of California, Los Angeles in 2012, and was a postdoctoral scholar at the Center for Vision Cognition Learning and Art in UCLA. He joined Google in August, 2013. His research interest include computer vision and statistical modeling.

Song-Chun Zhu received a PhD degree from Harvard University. He is a professor of Statistics and Computer Science at University of California, Los Angeles, and director of the UCLA Center for Vision, Cognition, Learning and Art. He has published over 180 papers in computer vision, statistical modeling and learning, cognition, and visual arts. He received a number of honors, including the Aggarwal prize from the Int'l Association of Pattern Recognition in 2008 for contributions to a unified foundation to computer vision, the Marr Prize in 2003 with Z. Tu et al. for image parsing, the Marr Prize honorary nominations with Y. N. Wu et al. in 1999 for texture modeling and 2007 for object modeling, a Sloan Fellowship in 2001, a US NSF Career Award in 2001, and an US ONR Young Investigator Award in 2001. A Helmholtz Test-of-Time award in 2013. He is a Fellow of IEEE.