Learning And-Or Models to Represent Context and Occlusion for Car Detection and Viewpoint Estimation

Tianfu Wu*, Bo Li* and Song-Chun Zhu

Abstract—This paper presents a method for learning And-Or models to represent context and occlusion for car detection and viewpoint estimation. The learned And-Or model represents car-to-car context and occlusion configurations at three levels: (i) spatially-aligned cars, (ii) single car under different occlusion configurations, and (iii) a small number of parts. The And-Or model embeds a grammar for representing large structural and appearance variations in a reconfigurable hierarchy. The learning process consists of two stages in a weakly supervised way (i.e., only bounding boxes of single cars are annotated). Firstly, the structure of the And-Or model is learned with three components: (a) mining multi-car contextual patterns based on layouts of annotated single car bounding boxes, (b) mining occlusion configurations between single cars, and (c) learning different combinations of part visibility based on car 3D CAD simulation. The And-Or model is organized in a directed and acyclic graph which can be inferred by Dynamic Programming. Secondly, the model parameters (for appearance, deformation and bias) are jointly trained using Weak-Label Structural SVM. In experiments, we test our model on four car detection datasets — the KITTI dataset [1], the PASCAL VOC2007 car dataset [2], and two self-collected car datasets, namely the Street-Parking car dataset and the Parking-Lot car dataset, and three datasets for car viewpoint estimation — the PASCAL VOC2006 car dataset [2], the 3D car dataset [3], and the PASCAL3D+ car dataset [4]. Compared with state-of-the-art variants of deformable part-based models and other methods, our model achieves significant improvement consistently on the four detection datasets, and comparable performance on car viewpoint estimation.

Index Terms—And-Or Graph, Hierarchical Model, Context, Occlusion Modeling, Car Detection, Viewpoint Estimation.

1 INTRODUCTION

1.1 Motivation and Objective

Car is one of the most frequently seen object category in every day scenes. Car detection and viewpoint estimation by a computer vision system has broad applications such as autonomous driving and parking management. Fig. 1 shows a few scenarios and complexities in car detection in four datasets. Car detection and viewpoint estimation are challenging problems due to the large structural and appearance variations, especially ubiquitous occlusions which further increase the intra-class variations significantly. In this paper, we are interested in learning a unified model which can detect cars in the four scenarios above and estimate car viewpoints. We aim to address two main challenges.

The first challenge is to handle occlusion. Occlusion is one of the most critical aspect of object representation, since (i) we do not know ahead of time what portion of an object (e.g., car) will be visible in a test image, (ii) we also do not know that information exactly in weakly-annotated training data (i.e., only bounding boxes of single cars are given, as considered in this paper), and (iii) object occlusions in testing data could be significantly different from those in training data. Handling occlusion entails models capable of capturing the underlying regularities of occlusions (i.e., different occlusion configurations). It enforces part-based representation in general because (i) dealing with occlusion entails searching through, or marginalizing, all possible occlusion configurations on-the-fly, and (ii) estimating viewpoints under occlusion entails information from visible parts.

The second challenge is to exploit contextual information. Car-to-car occlusion configurations (see examples in Fig.1 (b), (c) and (d)) naturally lead to car-to-car contextual patterns (e.g., different multi-car configurations such as 2, 3...
Car CAD Models and Parts

Car Type
Orientation
Relative Position
Camera View

Fig. 2. Illustration of the statistical regularities of car occlusions and multi-car context by 3D CAD simulation. We represent car-to-car occlusion at semantic part level (left) and generate a large number of synthetic occlusion configurations (middle) w.r.t. four factors (car type, orientation, relative position and camera view). We represent the regularities of different combinations of part visibilities, occlusion configurations by a hierarchical And-Or model. This model also represents multi-car layouts (right) based on the geometric configurations of single cars.

or 4 cars). The contextual patterns can be utilized in detection and viewpoint estimation due to the layout regularities and should be integrated with occlusion configurations.

To represent both occlusion and context, a model needs to take into account structural and appearance variations at multi-car, single car and car part levels jointly. We are interested in grammar models [8], [9] embedded in a hierarchical graph structure which can express a large number of configurations (occlusion configurations and multi-car configurations) in a compositional and reconfigurable manner.

1.2 Method Overview

1.2.1 Data Preparation with Simulation Study

Because manually annotating car views, parts and part occlusions on real images are time-consuming and error-prone, one innovation in this paper is that we utilize a large set of occlusion configurations and multi-car configurations generated by car CAD models and a graphics rendering engine (we used the publicly available SketchUp SDK). In the CAD simulation, we assume that the occlusion configurations and multi-car contextual patterns are subject to the four factors: car type, orientation, relative position and camera view. We use 17 semantic car parts as shown in different colors in the left of Fig. 2. According to the four factors, we then generate a large number of examples by placing 3 cars in a $3 \times 3$ grid (see the middle of Fig. 2). For the cars in the center, we obtain their assignments of part visibility which are used to learn the occlusion configurations. Similarly, we learn different multi-car contextual patterns based on the geometric configurations (see some examples in the right of Fig. 2). Note that the semantic part annotations in the synthetic examples are used to learn the structure of occlusion configurations and the parts are treated as latent variables in weakly-annotated training data of real images. We do not evaluate the performance of part localization and instead evaluate the viewpoint estimation based on the inferred part configurations.

1.2.2 The And-Or Model

There are three types of nodes: an And-node represents decomposition (e.g., a car is composed of a small number of parts), an Or-node represents alternative ways of decomposition accounting for structural variations (e.g., different part configurations of a single car due to occlusions), and a Terminal-node captures appearance variations to ground a car or a part to image data.

Fig. 3 illustrates the learned And-Or model. The hierarchy consists of layers of multi-car contextual patterns (top) and layers of occlusion configurations (bottom). The overall structure of our And-Or model is as follows:

i) The root Or-node represents different multi-car configurations which capture both viewpoints and car-to-car contextual patterns. Each multi-car contextual pattern is then represented by an And-node (e.g., car pairs and car triples shown in the figure). The contextual information reflects the layout regularities of a small number $N$ of cars in real scenarios (such as cars in a parking lot).

ii) A multi-car And-node is decomposed into a small number $N$ of single cars. Each single car is represented by an Or-node (e.g., the 1st car and the 2nd car), since we have different combinations of viewpoints and occlusion configurations (e.g., the car in the back of a car-pair can have different occluding situations due to the layouts).

iii) Each occlusion configuration is represented by an And-node which is further decomposed into parts. Parts are learned using car CAD simulation and are organized into consistently visible parts and optional part clusters (see the example in the right-bottom of Fig. 3). Then, a single car can be represented by the consistently visible parts (i.e., And) and one of the optional part clusters (i.e., Or). The green dashed bounding boxes show some examples corresponding to different occlusion configurations (i.e., visible parts) within the same viewpoint.

1.2.3 Weakly-supervised Learning of the And-Or Model

Using weakly-annotated real image training data, we learn the And-Or model with two stages:

i) Learning the structure of the hierarchical And-Or model. Both the multi-car contextual patterns and occlusion configurations of single cars are learned automatically based on the annotated single car bounding boxes in training data together with the synthetic examples generated from car CAD simulation. The multi-car contextual patterns are mined based on the clustering using the geometric layout
Fig. 3. Illustration of our And-Or model for car detection. It represents multi-car contextual patterns and occlusion configurations jointly by modeling spatially-aligned multi-cars together and composing visible parts explicitly for single cars. (Best viewed in color)

features. The occlusion configurations are learned based on the clustering using the part visibility data matrix (each row of which represents the assignment of part visibility of a generated example). The learned structure is a directed and acyclic graph since we have both single-car-sharing and part-sharing, which facilitates Dynamic Programming (DP) in inference.

ii) Learning the parameters for appearance, deformation and bias. Given the learned structure of the And-Or model, we jointly train the parameters under the structural SVM framework and adopt the Weak-Label Structural SVM (WLSSVM) [10], [11] in implementation.

The proposed And-Or model is flexible and reconfigurable to account for the large variations of both multi-car context and occlusion configurations in complex situations. It represents occlusions at semantic part level and captures the regularities of different combinations of part visibilities (i.e., occlusion configurations). By reconfigurable, it means that we learn appearance templates and deformation models for single cars and parts, and the composed appearance templates for a multi-car contextual pattern is inferred on-the-fly in detection according to the selections of their child single car Or-nodes. So, our model can express a large number of multi-car contextual patterns with different compatible occlusion configurations of single cars. Reconfigurability is one of the most desired property in hierarchical models, which plays the main role in boosting the performance in our experiments, and also distinguishes the proposed method to other models such as the visual phrase model [12] and different object pair models [13], [14], [15], [16].

In experiments, we evaluate the detection performance of our model on four car datasets: the KITTI dataset [1], the PASCAL VOC2007 car dataset [2] and two self-collected datasets – the Street-Parking dataset [6] and the Parking Lot dataset [7]. Our model outperforms different state-of-the-art variants of DPM [17] (including the latest implementation [18]) on all the four datasets, as well as other state-of-the-art models [6], [16], [19], [20] on the KITTI and the Street-Parking datasets. We evaluate viewpoint estimation performance on three car datasets: the PASCAL VOC2006 car dataset [2], the 3D car dataset [3], and the PASCAL3D+ car dataset [4]. Our model achieves comparable performance with the state-of-the-art methods (significantly better than the method using deep learning features [21]). The code and data are available on the author’s homepage .

Paper Organization. The remaining of this paper is organized as follows. Sec.2 overviews the related work and summarizes our contributions. Sec.3 presents the And-Or model and defines its scoring functions. Sec.4 presents the method of mining multi-car contextual patterns and occlusion configurations of single cars in weakly-labeled training data. Sec.5 discusses the learning of model parameters using

WLSSVM, as well as details of the DP inference algorithm. Sec. 6 presents the experimental results and comparisons of the proposed model on the four car detection datasets and the three viewpoint estimation datasets. Sec. 7 concludes this paper with discussions.

2 Related Work and Our Contributions

Over the last decade, object detection has made much progress in various vision tasks such as face detection [22], pedestrian detection [23], and generic object detection [2], [17]. In this section we focus on occlusion and context modeling in object detection, and briefly classify the recent literature into three streams. For a full review of contemporary approaches, we refer the reader to recent survey articles [17], [24].

- **Single Object Models and Occlusion Modeling.** Hierarchical models are widely used in recent literature of object detection and most existing approaches are devoted to learning a single object model. Many work share the similar spirit to the deformable part-based model [17] (which is a two-layer structure) by exploring deeper hierarchy and global part configurations [10], [24], [28], with strong manually-annotated parts [29] or available 3D CAD models [30], or by keeping human in-the-loop [31]. To address the occlusion problem, various occlusion models estimate the visibilities of parts from image appearance, using assumptions that the visibility of a part is (a) independent from other parts [32], [33], [34], [35], [36], (b) consistent with neighbouring parts [10], [37], or (c) consistent with its parent or child parts describing object appearance at different scales [38]. Another essential problem is to organize part configurations. Recently, [6], [10], [34] explored different ways to deal with this problem. In particular, [34] modelled different part configurations by the local part mixtures. [10] used a more flexible grammar model to infer both the occluder and visible parts of a occluded person. [6] regularized parts into consistently visible parts and optional part clusters, which is more efficient to represent occlusion configurations. In the recent work [39], [40], [41], [42], [43] also propose to enumerate possible occlusion configurations and model each occlusion configuration as a specific component. Though those models are successful in some heavily occluded cases, they do not represent contextual information, and usually learn another separate context model using the detection scores as input features. Recently, an And-Or quantization method is proposed to learn And-Or tree models [24] for generic object detection in PASCAL VOC [2] and learn car 3D And-Or models [44] respectively, which could be useful in occlusion modeling.

- **Object-Pair and Visual Phrase Models.** To account for the strong co-occurrence, object-pair [13], [14], [15], [16] and visual phrase [12] methods model occlusions and interactions using a X-to-X or X-to-Y composite template that spans both one object (i.e., “X” such as a person or a car) and another interacting object (i.e., “Y” or “X” such as the other car in a car-pair in parking lots or a bicycle on which a person is riding). Although these models can handle occlusion better than single object models in occluded situations, the object-pair or visual phrase just model occlusion implicitly, and they are often manually designed and fixed (i.e., not reconfigurable in inference), and as investigated in the KITTI dataset [16], their performance are worse than original DPM in complex scenarios.

- **Context Models.** Many context models have been exploited in object detection showing performance improvement [45], [46], [47], [48], [49]. Hoiem et al. [47] explored an influential work in scene context, Desai et al. [46] improved object detectors by incorporating the multi-class context on the challenging pascal dataset [2] in a max-margin framework. In [48], Tu and Bai integrate the detector responses with background pixels to determine the foreground pixels. In [49], Chen, et. al. propose a multi-order context representation to take advantage of the co-occurrence of different objects. Recently, [50] also explore using geographic contextual information to help car detection, and [51] explore the role of 3D panoramic context in object detection. Although these works verify that context is crucial in object detection, most of them model objects and context separately. In other words, they don’t model context and object representation in a unified framework.

This paper is extended from our two previous conference papers [6], [7] in the following aspects: (i) A unified representation is learned for integrating occlusion and context; (ii) More details on the learning algorithm and the detection algorithm are discussed; (iii) More analyses and comparisons on the experimental results are presented with better performance achieved.

In comparison, this paper makes three contributions to the literature of car detection.

- **It proposes an And-Or model to represent multi-car context and occlusion configurations.** The proposed model is multi-scale and reconfigurable to account for large structure, viewpoint and occlusion variations.

- **It presents a simple, yet effective, approach to mine context and occlusion configurations from weakly-labeled training data.**

- **It introduces two car datasets for evaluating occlusion and multi-car context, and outperforms state-of-the-art car detection methods in four challenging datasets.**

3 Representation

3.1 The And-Or Model and Scoring Functions

In this section, we introduce the notations to define the And-Or model and its scoring functions.

An And-Or model is defined by a 3-tuple

$$G = (\mathcal{V}, E, \Theta),$$

where $\mathcal{V} = \mathcal{V}_{\text{And}} \cup \mathcal{V}_{\text{Or}} \cup \mathcal{V}_{\text{Tr}}$, represents the set of nodes consisting of three subsets of And-nodes $\mathcal{V}_{\text{And}}$, Or-nodes $\mathcal{V}_{\text{Or}}$, and Terminal-nodes $\mathcal{V}_{\text{Tr}}$ respectively; $E$ the set of edges connecting all the nodes into a directed and acyclic graph (DAG); And,

$$\Theta = (\Theta^{app}, \Theta^{def}, \Theta^{bias}),$$

the set of parameters (for appearance, deformation and bias respectively, to be defined later). Fig. 4 shows the learned car And-Or model which has five layers.

A Parse Tree is an instantiation of the And-Or model by selecting the best child (according to the scoring functions to be defined) for each encountered Or-node where traversing
the And-Or model using Breadth-First search (BFS). See an example of parse tree shown by the green arrows in Fig. 4.

Appearance Features. We adopt the Histogram of Oriented Gradients (HOG) feature [17], [52] to describe car appearance. Let I be an image defined on a image lattice. Denote by \( H \) the HOG feature pyramid computed for I using \( \lambda \) levels per octave, and by \( \Lambda \) the lattice of the whole pyramid. Let \( p = (l, x, y) \in \Lambda \) specify a position \((x, y)\) in the \( l\)-th level of the pyramid \( H \). Denote by \( \Phi_{app}(H, p_l) \) the extracted HOG feature for a Terminal-node \( t \) placing at position \( p_l \) in the pyramid.

Deformation Features. We allow local deformation when composing the child nodes into a parent node. In our model, car parts are placed at twice the spatial resolution w.r.t. single cars, while single cars and composite multi-cars are placed at the same spatial resolution. We penalize the displacements between the anchor locations of child nodes (w.r.t. the placed parent node) and their actual deformed locations. Denote by \( \delta = [dx, dy] \) the displacement. The deformation feature is defined by,

\[
\Phi_{def}(\delta) = [dx^2, dx dy, dy^2].
\]

A Terminal-node \( t \in V_T \) grounds a single car or a car part to image data (see Layer 3 and 4 in Fig.4). Given a parent node \( A \), the model for \( t \) is defined by a 4-tuple \((\theta_{t}^{app}, s_t, a_{t|A}, \theta_{t|A}^{def})\) where \( \theta_{t}^{app} \subset \Theta^{app} \) is the appearance template, \( s_t \in \{0, 1\} \) the scale factor for placing node \( t \) w.r.t. its parent node, \( a_{t|A} \) a two-dimensional vector specifying an anchor position relative to the position of parent node \( A \), and \( \theta_{t|A}^{def} \subset \Theta^{def} \) the deformation parameters. Given the position \( p_A = (l_A, x_A, y_A) \) of the parent node \( A \), the scoring function of a Terminal-node \( t \) is defined by,

\[
\text{score}(t|A, p_A) = \max_{\delta \in \Delta} \left( \langle \theta_{t}^{app}, \Phi_{app}(H, p_l) \rangle - \langle \theta_{t|A}^{def}, \Phi_{def}(\delta) \rangle \right),
\]

where \( \Delta \) is the space of deformation (i.e., the lattice of the corresponding level in the feature pyramid), \( p_l = (l, x_t, y_t) \) with \( l = l_A - s_t \lambda \) and \( (x_t, y_t) = 2^v(x_A, y_A) + a_{t|A} + \delta \) where \( s_t = 0 \) means the object and parts are placed at the same resolution and \( s_t = 1 \) means parts are placed at twice the resolution of the object templates, and \( \langle \cdot, \cdot \rangle \) denotes the inner product.

An And-node \( A \in V_{And} \) represents a decomposition of a large entity (e.g., a multi-car layout at Layer 1 or a single car at Layer 3 in Fig.4) into its constituents (e.g., 2 or 3 single cars or a small number of car parts). Denote by \( ch(v) \) the set of child nodes of a node \( v \in V_{And} \cup V_{Or} \). The scoring function of an And-node \( A \) is defined by,

\[
\text{score}(A, p_A) = \sum_{v \in ch(A)} \text{score}(v|A, p_A) + b_A
\]

where \( b_A \in \Theta^{bias} \) is the bias term. Each single car And-node (at Layer 3) can be treated as the And-Or structure proposed in [6] or the DPM [17]. So, our model is flexible to integrate state-of-the-art single object models. For multi-car And-nodes (at Layer 1), their child nodes are Or-nodes and the scoring function \( \text{score}(v|A, p_A) \) is defined below.

An Or-node \( O \in V_{Or} \) represents different structure variations (e.g., the root node at Layer 0 and the \( i\)-th car node at Layer 2 in Fig.4). For the root Or-node \( O \), when placing at the position \( p \in \Lambda \), the scoring function is defined by,

\[
\text{score}(O, p) = \max_{v \in ch(O)} \text{score}(v, p),
\]

where \( ch(O) \subset V_{And} \). For the \( i\)-th car Or-node \( O \), given a parent multi-car And-node \( A \) placed at \( p_A \), the scoring function is then defined by,

\[
\text{score}(O|A, p_A) = \max_{v \in ch(O)} \max_{\delta \in \Delta} \left( \text{score}(v, p_{A|v}) - \langle \theta_{O|A}^{def}, \Phi_{def}(\delta) \rangle \right)
\]
where \( p_v = (l_v, x_v, y_v) \) with \( l_v = l_A \) and \( (x_v, y_v) = (x_A, y_A) + \delta \). The best child of an Or-node is computed by taking \( \arg \max \) of Eqn.(4) and Eqn.(5).

### 3.2 The DP Algorithm in Detection

In detection, we place the And-Or model at all positions \( p \in \Lambda \) and retrieve the parse trees for all positions at which the scores are greater than the detection threshold. Thank to the directed and acyclic structure of our And-Or model, we can utilize the efficient DP algorithm which consists of two stages:

**In the bottom-up pass:** Following the depth-first-search (DFS) order of nodes in the And-Or model, the bottom-up pass computes appearance score maps and deformed score maps for the whole feature pyramid \( \mathcal{H} \) for all Terminal-nodes, And-nodes and Or-nodes. The deformed score maps can be computed efficiently by the generalized distance transform [53] algorithm as done in [17].

**In the top-down pass,** we first find all detection candidates for the root Or-node \( O \), i.e., the positions

\[
P = \{ p; \text{score}(O, p) \geq \tau \text{ and } p \in \Lambda \}.
\]

Then, following the breadth-first-search (BFS) order of nodes, we can retrieve the parse tree at each \( p \).

**Post-processing.** To generate the final detection results of single cars for evaluation, we apply multi-car guided non-maximum suppression (NMS) to deal with occlusion:

i) Some of the single cars in a multi-car detection candidate are highly overlapped due to occlusion, so if we directly use conventional NMS, we will miss the detection of the occluded cars. We enforce that all the single car bounding boxes in a multi-car prediction will not be suppressed by each other. The similar idea is also used in [14].

ii) Overlapped multi-car detection candidates might report multiple predictions for the same car. For example, if a car is shared by a 2-car detection candidate and a 3-car detection candidate, it will be reported twice. We will keep the one with higher score only.

### 4 Learning And-Or Structures

In this section, we present the methods of learning the structures of And-Or model by mining context and occlusion configurations in the positive training dataset.

#### 4.1 Generating Multi-car Positive Samples

Denote by \( D^+ = \{ (I_1, B_1^+), \ldots, (I_n, B_n^+) \} \) the positive training dataset where \( B_i = \{ B_i^j \} \) is the set of \( k_i \) annotated single car bounding boxes in image \( I_i \) (where \( (x, y) \) is the top-left corner and \( (w, h) \) the width and height).

The set of multi-car positive samples by,

\[
D_{N\text{-car}}^+ = \{ (I_i, B_i^j) : k_i \geq N, j \in [1, k_i], j \neq N, B_i^j \subseteq \mathbb{B}, i \in [1, n] \}, \tag{6}
\]

where \( N\text{-car} \) represents a multi-car consisting of \( N \) single cars. we have,

i) \( D_{1\text{-car}}^+ \) consists of all the single car bounding boxes which do not overlap the other ones in the same image. For \( N \geq 2 \), \( D_{N\text{-car}}^+ \) is generated iteratively.

ii) To generate \( D_{2\text{-car}}^+ \) (see Fig.5 (a)), for each positive image \((I_i, B_i)\), with \( k_i \geq 2 \), we enumerate all valid 2-car configurations starting from \( B_i^1 \subseteq \mathbb{B} \): (i) select the current \( B_i^j \) as the first car \((1 \leq j \leq k_i)\), (ii) obtain all the surrounding car bounding boxes \( N_{B_i^j} \) which overlap \( B_i^j \), and (iii) select the second car \( B_i^{j'} \in N_{B_i^j} \) which has the largest overlap if \( N_{B_i^j} \neq \emptyset \) and \((I_i, B_i^j) \notin D_{2\text{-car}}^+ \) (where \( J = \{ j, k \} \)).

iii) To generate \( D_{N\text{-car}}^+ \) \((N > 2\), see Fig.5 (b)), for each positive image with \( k_i \geq N \) and \( \exists (I_i, B_i^K) \in D_{(N-1)\text{-car}}^+ \), (i) select the current \( B_i^K \) as the seed, (ii) obtain the neighbors \( N_{B_i^K} \) each of which overlap at least one bounding box in \( B_i^K \), (iii) select the bounding box \( B_i^j \in N_{B_i^K} \) which has the largest overlap and add \((I_i, B_i^j) \) to \( D_{N\text{-car}}^+ \) (where \( J = K \cup \{ j \} \)) if valid.

#### 4.2 Mining Multi-car Contextual Patterns

Considering \( N \geq 2 \), we use the relative positions of single cars to describe the layout of a multi-car sample \((I_i, B_i^j) \in D_{N\text{-car}}^+ \). Denote by \((cx, cy)\) the center of a car bounding box. Assume \( J = \{ 1, \cdots, N \} \). Let \( w_j \) and \( h_j \) be the width and height of the union bounding box of \( B_i^j \). With the center of the first car being the centroid, we define the layout feature by,

\[
\left[ \frac{cx^2 - cx_1^2}{w_j}, \frac{cy^2 - cy_1^2}{h_j}, \ldots, \frac{cx^N - cx_1^1}{w_j}, \frac{cy^N - cy_1^1}{h_j} \right]. \tag{7}
\]

We cluster these layout features over \( D_{N\text{-car}}^+ \) to get \( T \) clusters using \( k\)-means. The obtained clusters are used to specify the And-nodes at Layer 1 in Fig.4. The number of cluster \( T \) is specified empirically for different training datasets in our experiments.

In Fig. 6 (top), we visualize the clustering results for \( D_{2\text{-car}}^+ \) on the KITTI [1] and self-collected Parking Lot datasets. Each set of color points represents a specific 2-car context pattern. In the KITTI dataset, we can observe there are some specific car-to-car “peak" modes in the dataset (similar to the analyses in [16]), while the context patterns are more diverse in the Parking Lot dataset.

#### 4.3 Mining Occlusion Configurations

Above, we present the method of specifying Layer 0 – 2 in Fig.4. In this section we present the method of learning occlusion configurations for single cars (i.e., Layer 3 and 4 in Fig.4). We learn the occlusion configurations automatically from a large number of occlusion configurations. Because of the ambiguity of views and the relatively large number of parts (17 in our case), manually labeling views and parts are time consuming and error-prone. Thus, we utilize car 3D CAD data for the ease of getting these annotations. Note that the synthetic data is only used to learn the occlusion structure, while the appearance and geometry parameters are still learned from real data.
4.3.1 Generating Occlusion Configurations

We choose to put 3 cars in generating each occlusion configuration image, as this is a basic unit that can be used to further compose general car-to-car occlusions. Specifically, we choose the center and 2 other randomly selected positions on a $3 \times 3$ grid, and put cars around these grid points to simulate occlusion. For each set of the position triplet, we randomly choose values for a few factors controlling the occlusion, and then extract an occlusion configuration from the generated images. See some examples in Fig. 2.

We assume the occlusion configurations are controlled by the four factors: \textit{car type} $t$, \textit{orientation} $\rho$, \textit{relative position} $r$ and \textit{camera view} $\Pi$. To generate a configuration, we randomly choose values for these factors, where for each car with type $i$, $\rho_i \in \{\text{frontal, rear}\}$, $r_i = r_i^{(0)} + \Delta r$, where $r_i^{(0)}$ is the nominated position for the $i$-th car on the $3 \times 3$ grid, and $\Delta r = (dx, dy)$ is the relative distance (along $x$ axis and $y$ axis) between sampled position and nominated position of the $i$-th car. Assume the camera view is in the range of azimuth $\in [0, 2\pi]$ and elevation $\in [0, \pi/4]$, we discretize the view space into $B$ view bins uniformly along the azimuth angle. In the synthesized configurations, we assume a part is occluded if 60% of that part is not visible.

4.3.2 Constructing the Initial And-Or model of Single Cars

With the part-level visibility information, we could get two vectors for each occlusion configuration. The first one is a $(17 \text{ parts} \times B \text{ views})$ dimension binary valued vector $v$ for the visibilities of object parts, and the second one is a real valued ($(1 \text{ root} + 17 \text{ parts}) \times B \text{ views} \times 4$) dimension vector $b$ for the bounding boxes of the object and its parts. In both vectors, entries corresponding to invisible parts are set to 0.
Denoting $M$ as the dimension of the vector $v$, and by stacking $v$ for $N$ occlusion configurations, we can get an $N \times M$ occlusion matrix $D$, where the first few rows of this matrix for $B = 8$ is shown in the last column of Fig.7. Note that we have partitioned the view space into $B$ views, so for each row, the visible parts always concentrate in a segment of the vector representing that view.

To get an initial And-Or model, we assume that each row in $D$ corresponds to a small subtree of the root OR node. In particular, each subtree consists of an And-node as root and a set of terminal nodes as its children. An example of the data matrix and corresponding initial And-Or model is shown in the middle of Fig.7. Each row of $D$ represents an occlusion configuration, and each column represents a part. The part is either visible (white), or occluded (gray).

### 4.3.3 Refining the And-Or Structure

The initial And-Or model can be large and redundant, since it has many duplicated occlusion configurations (i.e. duplicated rows in $D$) and a combinatorial number of part compositions. In the following, we will pursue a compact And-Or structure from the initial one. The problem can be formulated as:

$$\min \sum_{i}^{N} \left| v_{i} - v_{i}(G) \right|_{2}^{2} + \lambda \left| G \right|$$  \quad (8)

where $v_{i}$ is the $i$-th row of the data matrix $D$, $v(G)$ returns its most approximate occlusion configuration generated by the And-Or graph (AOG), $|G|$ is the number of nodes and edges in the structure, and $\lambda$ is the trade-off parameter balancing the model precision and complexity. In each view, we assume the number of occlusion branches is not greater than $K(=4)$.

We solve Eqn.8 using a modified graph compression algorithm similar to [54]. As illustrated in the right of Fig.7, the algorithm starts from the large initial AOG described above, and iteratively combines branches as long as the introduced loss is smaller than the decrements in complexity term $\lambda|G|$. This process is equivalent to iteratively finding large blocks of 1s on the corresponding data matrix through row and column permutations, where an example is shown on the bottom of Fig.7. As there are consistently visible parts for each view, the algorithm will quickly converge to the structure shown in Fig.4.

With the refined And-Or model, we could get occlusion configurations (i.e., the consistently visible parts and optional occluded parts) in each view. Besides, the bounding box sizes and nominal positions of each terminal node w.r.t. its parent And-node can also be estimated by geometric means of corresponding values in the vector $b$. These information will be used to initialize the latent variables of our model in learning the parameters.

### 5 Learning Parameters

The parse tree $pt$ for each multi-car positive sample is hidden in the training. The parameters $\Theta = (\Theta^{app}, \Theta^{bias}, \Theta^{bias})$ are learned iteratively. We initialize the parse tree for each multi-car positive sample as stated in Sec.4. During learning, we run the DP inference to assign the optimal parse trees for them. We adopt the WLSSVM method [10] in learning. The objective function to be minimized is defined by,

$$\mathcal{E}(\Theta) = \frac{1}{2}||\Theta||^{2} + C \sum_{i=1}^{M} L'(\Theta, x_{i}, y_{i})$$  \quad (9)

where $x_{i} \in D_{N-car}^{+}$ represents a training sample ($N \geq 1$) and $y_{i}$ is the $N$ bounding box(es). $L'(\Theta, x, y)$ is the surrogate loss function,

$$L'(\Theta, x, y) = \max_{pt \in \Omega_{D}} \left[ \text{score}(x, pt; \Theta) + L_{\text{margin}}(y, box(pt)) \right] - \max_{pt \in \Omega_{G}} \left[ \text{score}(x, pt; \Theta) - L_{\text{output}}(y, box(pt)) \right]$$  \quad (10)

where $\Omega_{G}$ is the space of all parse trees derived from the And-Or model $G$, $\text{score}(x, pt; \Theta)$ computes the score of a
parse tree as stated in Sec.3, and box(pt) the predicted bounding box(es) base on the parse tree. As pointed out in [10], the loss \( \ell_{\text{margin}}(y, \text{box}(pt)) \) encourages high-loss outputs to “pop out of the first term in the RHS, so that their scores get pushed down. The loss \( \ell_{\text{output}}(y, \text{box}(pt)) \) suppresses high-loss outputs in the second term in the right hand side, so the score of a low-loss prediction gets pulled up. More details are referred to [10], [11]. The loss function is defined by,

\[
\ell_{t,r}(y, \text{box}(pt)) = \begin{cases} 
\ell & \text{if } y = \bot \text{ and } pt \neq \bot \\
0 & \text{if } y = \bot \text{ and } pt = \bot \\
\ell & \text{if } y \neq \bot \text{ and } \exists B \in y \text{ with } ov(B, B') < \tau, \forall B' \in \text{box}(pt) \\
0 & \text{if } y \neq \bot \text{ and } \exists B, B' \in \text{box}(pt) \\
\end{cases}
\]

where \( \bot \) represents background output and \( ov(\cdot, \cdot) \) is the intersection-union ratio of two bounding boxes. Following the PASCAL VOC protocol we have \( \ell_{\text{margin}} = \ell_{1,0.5} \) and \( \ell_{\text{output}} = \ell_{\infty,0.7} \). In practice, we modify the implementation in [18] for our loss formulation.

6 EXPERIMENTS

In this section, we evaluate our models on four car detection datasets and three car viewpoint estimation dataset and present detail analyses on different aspects of our models. We first introduce two self-collected car datasets of street-parking cars and parking-lot cars respectively (Sec. 6.1), and then evaluate the detection performance of our models on four datasets (Sec. 6.2): the two self-collected datasets, the KITTI car dataset [1] and the PASCAL VOC2007 car dataset [2]. We further analyze the performance of our model w.r.t. different aspects of our models (Sec. 6.3). The performance of car viewpoint estimation is discussed in Sec. 6.4.

And-Or Model Specifications. We test our model using two types of specifications to be consistent with our two previous conference papers, one is called And-Or Structure [6] for occlusion modeling based on CAD simulation without multi-car context components, and the other called Hierarchical And-Or Model [7] for occlusion and context. We also compare two methods of part selection in hierarchical And-Or model, one is based on the greedy parts as done in DPM [17], denoted by AOG+Greedy, and the other based on the proposed CAD simulation, denoted by AOG+CAD.

Training and Testing Time. In all experiments, we utilize a parallel computing technique to train our model. It takes about 9 hours to train an And-Or Structure model and 16 hours to train a hierarchical And-Or Model due to inferring the assignments of part latent variables on positive training examples and mining hard negatives. For detection, it takes about 2 and 3 seconds to process an image with size of 640 x 480 pixels for a And-Or structure and a hierarchical And-Or model, respectively.

6.1 Datasets

To test our model on occlusion and context modeling, we collected two car datasets (which are released with this paper). The first one is Street-Parking car dataset and the second one is Parking Lot car dataset 4.


Street Parking Car Dataset. There are several datasets featuring a large amount of car images [2], [3], [55], [56], but they are not suitable to evaluating occlusion handling, as the proportion of (moderately or heavily) occluded cars is marginal. The recently proposed KITTI dataset [1] contains occluded cars parked along the streets, but it is not a good dataset for our task either for the following reasons: (i) the car views are rather fixed as the video sequences are captured from a car driving on the road, and (ii) the evaluation protocol only counts cars with no or mild occlusion (< 20%) in the testing set. To evaluate our model on occlusion handling, we collected a car dataset emphasizing street parking cars with a large amount of occlusions and diverse viewpoint changes (see the last two rows of Fig.10). The dataset consists of 927 images. Fig. 8 shows the bounding box overlap distribution and average number of cars per image on the dataset. These two statistics indicate car occlusion distribution. For the simplicity of annotation, we adopt the weak annotation strategy, that is to only label the bounding boxes of single cars in each image. We split the dataset into training and testing sets containing 463 and 464 images, or 3, 260 and 3, 267 cars, respectively.

Parking Lot Dataset. In terms of the car-to-car context and occlusion, although the KITTI dataset [1] provides challenging features, the camera viewpoints are relatively restricted due to the camera platform (e.g., no birdeye’s view), and the number of cars in each image is small. Our Street Parking Car Dataset provides more viewpoints, however, the context and occlusion configurations are somewhat restricted (most cars just compose the head-to-head occlusions). To thoroughly evaluate our models in terms of context and occlusion, we collected the parking lot car dataset, which provides more features on occlusion variation and the large number of cars in each image (see the 4-th and 5-th rows of Fig. 10). It contains 65 training images and 63 testing images. Although the number of images is small, the number of cars is noticeably large, with 3, 346 cars (including left-right mirrored ones) for training and 2, 015 cars for testing.

6.2 Detection

We test our hierarchical And-Or Model on four challenging datasets. For simplicity, we just use greedy part selection for occlusion modelling.
6.2.1 Results on the KITTI Dataset

The KITTI dataset [1] contains 7,481 training images and 7,518 testing images, which are captured from an autonomous driving platform. We follow the provided benchmark protocol for evaluation. Since the authors of [1] have not released the test annotations, we test our model in the following two settings.

Training and Testing by Splitting the Trainset. We randomly split the KITTI trainset into the training and testing subsets equally.

Baseline Methods. Since DPM [17] is a very competitive model with source code publicly available, we compare our model with the latest version of DPM (i.e., voc-release5) [18]. The number of components are set to 16 as the baseline methods trained in [1], other parameters are set as default.

Parameter Settings. We consider multi-car contextual patterns with the number of cars $N = 1, 2$. We set the number of context patterns and occlusion configurations to be 10 and 16 respectively in Sec.4. As a result, the learned hierarchical And-Or model has 10 2-car configurations in layer 1, and 16 single car branches in layer 3 (see Fig. 4).

Detection Results. The left figure of Fig. 9 shows the precision-recall curves of DPM and our model. Our model outperforms DPM by 9.1% in terms of average precision (AP). The performance gain comes from both precision and recall, which shows the importance of context and occlusion modeling.

Testing on the KITTI Benchmark. We test the trained models above (i.e., using half training set) on the KITTI testset. The detection results and performance comparison are shown in Table 1. This benchmark has three subsets (Easy, Moderate, Hard) w.r.t the difficulty of object size, occlusion and truncation. Our model outperforms all the other methods tested on this benchmark. Specifically, our model outperforms OC-DPM [16] on all the three subsets by 5.32%, 1.08%, and 1.74%. We also compare with the baseline DPM trained by ourselves using the voc-release5 code [18], the performance gain of our model mainly comes from the Moderate and Hard car subsets, with 11.01% and 12.46% in terms of AP respectively. For other DPM based methods trained by the benchmark authors, our model outperforms the best one - MDPM-un-BB by 9.07%, 4.87% and 7.17% respectively.

Note that our model is trained using half of the KITTI trainset, while other methods in the benchmark use more training data (e.g., 1/6 cross validation). The performance improvement by our model is significant. As mentioned in [16], because of the large number of cars in KITTI dataset, even a small amount (1.6%) of AP increasing is still considered significant.

The first 3 rows of Fig. 10 show the qualitative results of our model. The red bounding boxes show the successful detection, the blue ones the missing detection, and the green ones the false alarms. In experiments, our model is robust to detect cars with heavy car-to-car occlusion and clutter. The failure cases are mainly due to extreme occlusion, too small car size, car deformation and/or inaccurate (or multiple) bounding box localization.

6.2.2 Results on the Parking Lot Dataset

Evaluation Protocol. We follow the PASCAL VOC evaluation protocol [2] with the overlap of intersection over union being greater than or equal to 60% (instead of original 50%). In practice, we set this threshold to make a compromise between localization accuracy and detection difficulty. The detected cars with bounding box height smaller than 25 pixels do not count as false positives as done in [1]. We compare with the latest version of DPM implementation [18] and set the number of contextual patterns and occlusion configurations to be 10 and 18 respectively.

Detection Results. In the right of Fig. 9 we compare the performance of our model with DPM. Our model obtains 55.2% in AP, which outperforms the latest version of DPM by 10.9%. The fourth and fifth rows of Fig. 10 show the qualitative results of our model. Our model is capable of detecting cars with different occlusion and viewpoints.

6.2.3 Results on the Street Parking Dataset

To compare with the benchmark methods, we follow the evaluation protocol provided in [6].

Results of our model and other benchmark methods are shown in Table 2, our hierarchical And-Or model outperforms DPM [18] and our previous And-Or Structure [6] by 10.1% and 4.3% respectively. We believe this is because our model accounts for context and occlusion jointly, and the flexible structure provides more expressive power of representing occlusions. The last two rows in Fig. 10 show some qualitative examples. Our model is capable of detecting occluded street-parking cars, meanwhile it also has a few inaccurate detection results and misses some cars that are too small or uncommon in the trainset.
6.3 Diagnosing the Performance of our Model

In this section, we evaluate various aspects to diagnose the effects of each individual component in our model.

6.3.1 The Effect of Occlusion Modeling

Our And-Or Structure model is based on CAD simulation. Thus in the first analysis, we test the effectiveness of the learned And-Or structure in representing different part occlusion configurations. For this purpose, we generate a synthetic dataset using 5,040 cars synthetic images as our training data, and a mixture of 3,000 cars and 7 cars (we generate the 7 cars in a 1 × 7 grid) synthetic images as our testing data. For each generated image, we add the background from the category None of the TU Graz-02 dataset [57] and apply Gaussian blur to reduce the boundary effects. Samples of both training and testing data can be seen on the left and middle of Fig.11. On this dataset, the best DPM has 16 components and the best And-Or structure has 8 views with 19 occlusion branches, 5 layers and 111 nodes. As can be seen on the right of Fig.11, our model outperforms the DPM by 7.2% in AP.

6.3.2 The Effect of CAD Simulation in Real Scenarios

To verify the effectiveness of our And-Or Structure model in terms of occlusion modeling, we compare it with state-of-the-art DPM [17]. Both of these two models are based on part-level occlusion modeling. For And-Or Structure, the semantic visible parts are learned based on the guide of CAD simulation. For DPM, it assumes that all latent parts are visible, and can cope with a general case of occlusion. The second and third column of Table 2 shows the performance of these two models on Street Parking Dataset. We can see the semantic visible parts generated by CAD simulation can be generalized to real datasets. By adding context, we are interested in whether it affects the effectiveness of occlusion modeling. To compare AOG+Greedy and AOG+CAD fairly, they have the same number of context patterns and occlusion configurations, 8 and 16 respectively. As shown in the fourth and fifth column of Table 2, AOG+CAD is better than AOG+Greedy, which further shows the advantage of...
modelling occlusion using semantic visible parts.

Fig. 12 shows the inferred part bounding boxes by AOG+Greedy and AOG+CAD. We can observe that the semantic parts in AOG+CAD are meaningful, although they may be not accurate enough in some examples.

6.3.3 The Effect of Multi-car Context Modeling

State-of-the-art models are mainly based on single car modelling. To evaluate the effectiveness of context, we compare our hierarchical And-Or model with other non-context models in Table 1. We can see the our model outperforms all of other models in different occlusion settings. Specifically, our model outperforms DPM by a large margin (above 10% in AP) on the “Moderate” and “Hard” KITTI test data, which reflects context is very important to object detection especially in heavily occluded car-to-car cases.

On the Street Parking Dataset, we observe the same results. As can be seen in Table 2, both AOG+Greedy and AOG+CAD outperform DPM and And-Or Structure by a large margin. Here AOG+Greedy and AOG+CAD jointly model context and occlusions, while DPM and And-Or Structure model occlusions only.

6.3.4 Performance on General Occlusion Settings

Our model is generalizable in terms of context and occlusion modelling, it could cope with both occlusion and non-occlusion situations. To verify our model on less occluded settings, we use the PASCAL VOC 2007 Car dataset as a testbed. As analyzed by Hoiem, et. al. in [5], cars in PASCAL VOC dataset do not have much occlusion and car-to-car context.

Firstly, we verify our And-Or Structure is capable to deal with cars on PASCAL VOC 2007 as well as state-of-the-art DPM method [18]. To approximate the occlusion configurations observed on this dataset, we generate synthetic images with car-to-car occlusions as well as with only car self-occlusions. For the car-to-car occlusions, we use the full $3 \times 3$ grid instead of the special case in the street parking dataset. Correspondingly, the learned And-Or structure contains branches for self-occlusions as well as those for car-to-car occlusions. On this dataset, the DPM has 6 components and the And-Or structure has 6 views with 10 occlusion branches, 5 layers and 109 nodes.

As is shown in the third column of Table 3, the performance of our And-Or structure model is comparable with DPM. Our model achieves slightly better recall than DPM, which meets the analysis in [5]. This experiment demonstrates that our And-Or structure method does not lose performance in general dataset.

Secondly, we verify our hierarchical And-Or model is capable to detect cars on PASCAL VOC 2007 as other single object based models. We compare with the latest version of DPM [18]. The APs are 60.6% (our model) and 58.2% (DFM) respectively (Table 3).

6.4 View Estimation

With the help of CAD simulation, our And-Or Structure model could output the viewpoints of detected cars. To verify the capability of this model on view estimation. We perform 2 experiments.

First, we report the mean precision in pose estimation (MPPE) on both Pascal VOC 2006 car dataset [2] and 3D Car dataset [3] following the protocol in [59] and [3], respectively. Since for view estimation, the two datasets emphasize visible cars, we model our And-Or structure using images only with self-occlusion. Table 4 shows a comparison of our model with state-of-the-art methods on these two datasets.
Our model is comparable to or better than recently proposed models.

Second, we compare our model with state-of-the-art models on the recently proposed PASCAL3D+ Dataset [4]. This dataset augments 12 rigid categories of the PASCAL VOC 2012 [2] with 3D annotations, and is a novel and challenging dataset for 3D object detection and pose estimation. In our case, we just test on the car category, the evaluation protocol follows the dataset benchmark.

### 7 Conclusion

In this paper, we propose an And-Or model to represent context and occlusion for car detection and viewpoint estimation. The model structure is learned by mining multi-car contextual patterns and occlusion configurations at three levels: a) multi-car layouts, b) single car and c) car parts. Our model is organized in a directed and acyclic graph structure so the efficient DP algorithm can be used in inference. The model parameters are learned by WLSSVM [10]. Experimental results show that our model is effective in modeling context and occlusion information in complex situations, and achieves better performance over state-of-the-art car detection methods and comparable performance on viewpoint estimation.

There are two main limitations in our current implementation. The first one is that we exploited the multi-car contextual patterns using 2-car composite only. In the scenarios similar to street parking cars and parking lot cars, we could explore multi-car context with more than 2 spatially-aligned cars, as well as 3D scene parsing context [63]. The second one is that we utilized the HOG features for appearance only. Based on the recent progress on feature learning by convolutional neural network (CNN) [64], [65], we can also substitute the HOG by the CNN features. Both aspects are addressed in our on-going work and may potentially improve the performance.

Meanwhile, we are applying the proposed method to other object categories and studying different ways of mining contextual patterns and occlusion configurations (e.g., integrating with the And-Or quantization methods for 2D object modeling [24] and 3D car modeling [44]).

### Acknowledgments

B. Li is supported by China 973 Program under Grant no. 2012CB316300. T.F. Wu and S.C. Zhu are supported by DARPA MSEE project FA 8650-11-1-7149, MURI grant ONR N00014-10-1-0933, and NSF IIS1018751. We thank Dr. Wenzhe Hu for helpful discussions.

### References


Tianfu Wu received a Ph.D. degree in Statistics from University of California, Los Angeles (UCLA) in 2011. He is currently a research assistant professor in the center for vision, cognition, learning and art (VCLA) at UCLA. His research interests include: (i) Statistical learning of large scale hierarchical and compositional models (e.g., And-Or graphs) from images and videos. (ii) Statistical inference by learning near-optimal cost-sensitive decision policies. (iii) Statistical theory of performance guaranteed learning algorithm and inference procedure.

Bo Li received a B.S. degree from Beijing Institute of Technology in 2010. He is currently a Ph.D. student in School of Computer Science and Technology, Beijing Institute of Technology, and a visiting student at Center for Vision, Cognition, Learning and Art. He has published over 160 papers in computer vision, statistical modeling and learning, cognition, and visual arts. He received a number of honors, including the J.K. Aggarwal prize from the Int'l Association of Pattern Recognition in 2008 for “contributions to a unified foundation for visual pattern conceptualization, modeling, learning, and inference”, the David Marr Prize in 2003 with Z. Tu et al. for image parsing, twice Marr Prize honorary nominations in 1999 for texture modeling and in 2007 for object modeling with Z. Si and Y.N. Wu. He received the Sloan Fellowship in 2001, a US NSF Career Award in 2001, and an US ONR Young Investigator Award in 2001. He received the Helmholtz Test-of-time award in ICCV 2013, and he is a Fellow of IEEE since 2011.

Song-Chun Zhu received a Ph.D. degree from Harvard University in 1996. He is currently a professor of Statistics and Computer Science at UCLA, and the director of the Center for Vision, Cognition, Learning and Art. He has published over 160 papers in computer vision, statistical modeling and learning, cognition, and visual arts. He received a number of honors, including the J.K. Aggarwal prize from the Int'l Association of Pattern Recognition in 2008 for “contributions to a unified foundation for visual pattern conceptualization, modeling, learning, and inference”, the David Marr Prize in 2003 with Z. Tu et al. for image parsing, twice Marr Prize honorary nominations in 1999 for texture modeling and in 2007 for object modeling with Z. Si and Y.N. Wu. He received the Sloan Fellowship in 2001, a US NSF Career Award in 2001, and an US ONR Young Investigator Award in 2001. He received the Helmholtz Test-of-time award in ICCV 2013, and he is a Fellow of IEEE since 2011.