Modeling Occlusion by Discriminative AND-OR Structures

Bo Li[†], Wenze Hu[‡], Tianfu Wu[‡*] and Song-Chun Zhu[‡]
[†]Beijing Lab of Intelligent Information Technology, Beijing Institute of Technology
[‡]Department of Statistics, University of California, Los Angeles
boli86@bit.edu.cn, {wzhu, tfwu, sczhu}@stat.ucla.edu

Abstract

Occlusion presents a challenge for detecting objects in real world applications. To address this issue, this paper models object occlusion with an AND-OR structure which (i) represents occlusion at semantic part level, and (ii) captures the regularities of different occlusion configurations (i.e., the different combinations of object part visibilities). This paper focuses on car detection on street. Since annotating part occlusion on real images is time-consuming and error-prone, we propose to learn the the AND-OR structure automatically using synthetic images of CAD models placed at different relative positions. The model parameters are learned from real images under the latent structural SVM (LSSVM) framework. In inference, an efficient dynamic programming (DP) algorithm is utilized. In experiments, we test our method on both car detection and car view estimation. Experimental results show that (i) Our CAD simulation strategy is capable of generating occlusion patterns for real scenarios, (ii) The proposed AND-OR structure model is effective for modeling occlusions, which outperforms the deformable part-based model (DPM) [6, 10] in car detection on both our self-collected street parking dataset and the Pascal VOC 2007 car dataset [4], (iii) The learned model is on-par with the state-of-the-art methods on car view estimation tested on two public datasets.

1. Introduction

Handling occlusion is a challenging problem in object detection since occlusion increases the intra-class variations of an object category significantly. Taking the car-to-car occlusion as an example illustrated in the left of Fig.1, the occlusion configurations (i.e., the combinations of visible parts) are all different for each car. This poses a difficult problem on learning an object model, as the model needs to encode a large number of occlusion configurations.

Modern object models such as DPM [6] and its 2D and 3D extensions [10, 34, 24] are fairly successful in detecting objects. Our objective is to extend these models so that their parts can be re-configured to represent objects with different occlusion configurations. Inspired by the expressive power of AND-OR graph [35], we propose a simple AND-OR structure (which is a directed and acyclic graph) to model the occlusion configurations effectively.

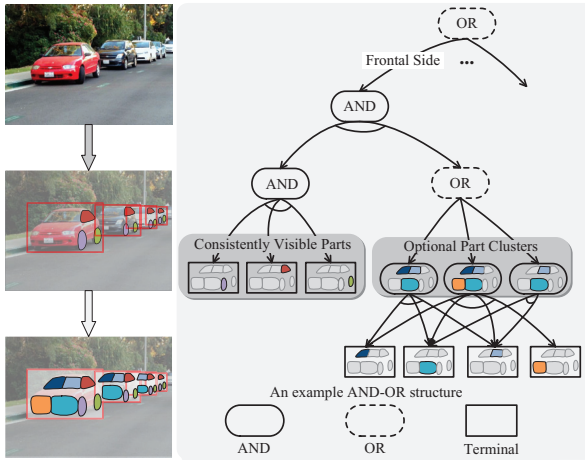


Figure 1. Illustration of our discriminative AND-OR structure for frontal-side view cars. It organizes object parts into consistently visible parts and optional part clusters, and then represents an object with the consistently visible parts (i.e., AND) and one of the optional part clusters (i.e., OR).

Fig.1 illustrates the AND-OR structure on frontal-side view cars. In this structure, a valid occlusion configuration can be generated by composing (AND) the consistently visible parts together with one of (OR) the optional part clusters. This structural representation is more compact than plainly remembering individual configuration, yet it effectively constrains the space of occlusion configurations as no other unobserved configurations are allowed.

Because manually labelling views, parts and part occlusion on real images are time-consuming and error-prone, we propose to learn the AND-OR structure using a large set of occlusion configurations generated by car CAD models and a graphics rendering engine. By directly incorporating the appearance and deformation formulations from the DPM [6] model, the parameters of this AND-OR structure

*T. Wu is the corresponding author.

can be discriminatively trained with real images under the latent structure SVM (LSSVM) framework [32]. Because the parts are shared by multiple occlusion configurations, we can use images with different occlusion configurations collectively to train them, which would be more efficient and robust than training them individually. As the proposed AND-OR structure is a directed and acyclic graph (DAG), the efficient DP algorithm can be used in inference.

In experiments, we test our method on both car detection and car view estimation. We use 5 different datasets including (1) our synthetic dataset generated from car CAD models, (2) our street parking dataset collected for evaluating the performance of handling occlusion (which will be released with this paper), (3) Pascal VOC 2007 [4] car dataset for evaluating detection performance in general situation, (4) Pascal VOC 2006 car dataset [5] and (5) 3D Car dataset [26] for testing view estimation. The experimental results show that (i) The proposed AND-OR structure is capable of modeling occlusions effectively, which outperforms DPM [6] with the latest implementation [10] significantly on our street parking dataset and remains comparable on the PASCAL VOC 2007 car dataset (which does not have many occlusions between cars as pointed out by [15]). (ii) The learned model is on-par with the state-of-the-art methods on car view estimation.

2. Related Work and Our Contributions

Performance of generic object models [2, 6, 34, 14] degrades once the objects are partially occluded. For example, many part-based deformable models compute object detection score as the summation over that of its parts [6, 31]. As a consequence, if some parts are occluded, scores of these part detectors would be very low, thus drags down the overall object score and might cause false negatives.

To extend these models for handling object occlusion, one important issue is to estimate the visibilities of object parts. Towards this problem, various occlusion models estimate the visibilities of parts from image appearance, using assumptions that the visibility of a part is independent from other parts [30, 29, 13], is consistent with neighbouring parts [7, 9], or is consistent with its parent or child parts describing object appearance at different scales [3]. Another essential problem is to organize part configurations. Recently, [9, 13] explored two different ways to deal with this problem. In particular, [13] modelled different part configurations by the local mixtures. [9] used a more flexible grammar model to infer both the occluder and visible parts of a occluded person.

We utilize the fact that parts are not occluded at random, and model the overall structures or regularities in the visibility of all object parts. While the idea of modelling occlusion structure is similar to [16], our paper is aimed at modelling a strong occlusion prior for an entire object class, instead of a generic prior suitable for regular objects on a table. For our application, the latter one will be less informative as it averages out the contribution of the object shape information to the occlusion structure.

The AND-OR structure used in this paper is inspired by the AND-OR graph [35], where most of the object models are generative with sparse image features. Our model keeps the AND-OR structure for its compositional power, but use semi-dense features [2] and discriminative learning to achieve high performance in object detection and view estimation. In the literature, [17] employed an AND-OR Tree (AOT) for 3D car modelling, but not on occlusion.

The proposed approach uses CAD models to synthesize the occlusion patterns under different object distances and views. While several recent methods use the synthesized images to learn part geometry [24] or detailed part appearance [20], our method only uses them to learn the configuration structure of occlusion, where the appearance terms are still trained using real images.

Finally, as this paper mainly concentrates on occlusion modeling, we directly incorporate the ingredients of DPM [6] for modeling object part appearance and deformation.

The main contributions of this paper include:

i) We propose a generic AND-OR structure to capture the structure of various occlusion configurations by hierarchically composing a small number of object parts.

ii) We propose to simulate different occlusions using weakly semantic parts of CAD models instead of manually annotating them on real images.

iii) We introduce a street parking car dataset emphasizing detection of cars with occlusions, as a benchmark for our method and future methods.

3. The Discriminative AND-OR model

3.1. The AND-OR Structure and its Parameters

Fig.2 gives an overview of the learned full AND-OR structure for the car category. The structure is a DAG which encodes the patterns of visible part combinations observed in occluded car images. Specifically, the structure is composed of the following nodes:

i). A set of OR nodes $\mathcal{V}_{\rm OR}$ representing the options of selecting one of their child nodes. There are two types of OR nodes: a) the root OR node, which switches between object models at different views. b) the occlusion OR nodes, which connect to different optional part clusters that are visible or occluded together in different occlusion configurations. Each OR node $O \in \mathcal{V}_{\rm OR}$ has a branching variable denoted by $\omega(O)$, indicating which one of its child nodes is selected, and $\omega(O)$ will be inferred on-the-fly in detection.

ii). A set of AND nodes $\mathcal{V}_{\rm AND}$ denoting compositions of object parts. According to their semantics, these AND nodes are also organized into two groups: a) Object level AND nodes, which collect all the parts in an occlusion configuration. In the following, we treat the coarse overall appearance of an object (i.e. the 'root') as a generalized part of

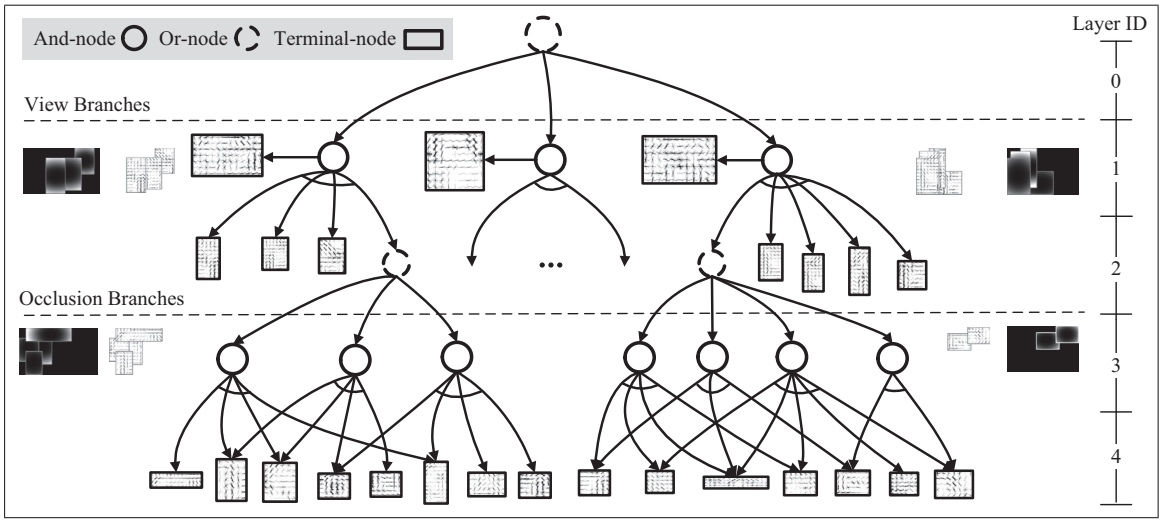


Figure 2. The learned AND-OR structure for cars. For clarity, we show a portion of the whole model. From the top to bottom, the root node represents an object category (car) OR-node (dashed circle), which has a set of child AND-nodes (solid circle) representing different view points. Each viewpoint AND-node consists of a small number of consistently visible parts (as terminal nodes plotted by rectangles) and a occlusion OR-node. The occlusion OR-node represents optional parts subject to different occlusion patterns.

the object. b) Part cluster AND nodes. Each of these AND nodes collects a subset of object parts that will become visible or invisible together in an occlusion configuration.

iii). A set of terminal nodes $\mathcal{V}_T$. These terminal nodes encode the image appearance of the whole object as well as its parts at different scales. Each terminal node t has its own location, which will also be inferred during detection. The inferred locations of the nodes represent positions of the object and its visible parts. Each t also has a scale shift parameter δs_t, denoting the difference of the feature scale from that of its parent AND node. Currently, we fix the δs to be 0 and 1 for generalized and object part node respectively, which means they are always in neighbouring octaves in the image pyramid.

These nodes define the structure of our AND-OR model, and the parameters of this model include: 1.) A vector of bias terms Θ^{bias}, each as a bias for an AND node. This term balances the templates for different occlusion configurations such that their scores are comparable. 2.) For terminal nodes, there are parameters Θ^{app}, Θ^{geo} for modelling their image appearance and geometry of a part. For simplicity, we pack these parameters into $\Theta = (\Theta^{app}, \Theta^{geo}, \Theta^{bias})$.

Therefore, our AND-OR Structure model is defined as a 4-tuple:

$$AOT = (\mathcal{V}_{\text{AND}}, \mathcal{V}_{\text{OR}}, \mathcal{V}_T, \Theta) \tag{1}$$

Our model could be transformed to a big mixtures-of-DPM by removing part sharing and moving all OR nodes to the root, which has more complexity and less robustness, as there are no shared nodes (i.e. the consistently visible parts) and shared training images in each view.

3.2. Scoring Functions of the AND-OR structure

Given an image $\mathbf{I}$ and its corresponding image feature pyramid H such as HOG [2], the score $S(\cdot,\cdot)$ of a node in the lattice Λ of the pyramid H is defined as follows:

i) For a terminal node $t \in \mathcal{V}_T$ w.r.t. its placed parent AND node A at u:

$$\begin{aligned}\mathrm{S}(t|A,u) = \max_{v\in\Lambda}(&\langle\theta_t^{app}, \Phi^{app}(H,t,v)\rangle - \\ &\langle\theta_{t|A}^{geo}, \Phi_{t|A}^{geo}(v,u)\rangle)\end{aligned} \tag{2}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product, θ_t^{app} and $\theta_{t|A}^{geo}$ are the parameters of node t, $\Phi^{app}(H,t,v)$ retrieves the appearance features cropped from pyramid H at location v with the scale shifted according to δs_t. For the geometry, we adopt the same quadratic penalty as in DPM and thus $\Phi_{t|A}^{geo}(v,u) = [\mathrm{d}x^2, \mathrm{d}y^2, \mathrm{d}x, \mathrm{d}y]$, where $\mathrm{d}x$ and $\mathrm{d}y$ are the displacements of v from u.

ii) For an AND node A placed at u:

$$S(A,u) = S(O,u) + \sum_{t\in T(A)} S(t|A,u) + \theta_A^b \tag{3}$$

where O is the child OR node of A (if has), the function $T(A)$ retrieves all the child terminal nodes of A, and θ_A^b is the corresponding bias.

iii) For an OR node O placed at u:

$$\mathrm{S}(O,u) = \max_{A\in ch(O)} S(A,u) \tag{4}$$

where A denotes a child AND node of O, and the function $ch(O)$ retrieves all the child AND nodes of O. Correspondingly, the branching variable is: $\omega(O) = \mathrm{argmax}_{A\in ch(O)}\, S(O,u)$.

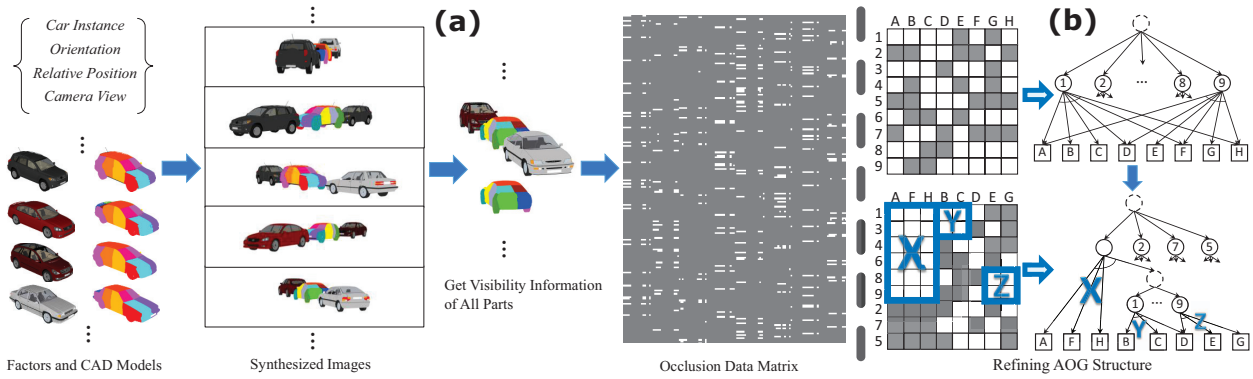


Figure 3. **(a)** To generate each occlusion configuration, we randomly choose values for the factors listed on the top-left corner, and generate images using CAD models such as shown in the bottom-left. When generating images (shown in the second column), we use the color coded model for the car instances in the center, and regular CAD models for the rest. The third column shows how to get the visible annotation of a part (see text for details). Each of these images then contribute one row to the occlusion data matrix shown on the right, where each column denotes if a part is visible (white) or not (gray). **(b)** An illustration of graph compression algorithm. Given the data matrix $\mathcal{D}$ and the initial AOT which plainly remembering each occlusion configuration as a row in $\mathcal{D}$, the algorithm iteratively pursues big blocks of 1s (e.g. X, Y, Z). Here the biggest block X corresponding to a subset of shared parts of the refined AOT (see the 1st subtree).

4. AND-OR Structure Learning

We propose to learn the AND-OR structure 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 labelling views and parts are time consuming and error-prone. Thus, we learn the AND-OR structure on 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. Taking the number of views as the only parameter, this structure learning process can be divided to 3 steps: (i) generating occlusion configurations, (ii) constructing data matrix for an initial AND-OR Tree (AOT), and (iii) refining the initial AOT structure.

4.1. Generating occlusion configurations

We choose to put 3 cars in generating each occlusion 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 3×3 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. Sample images generated using a pool of 40 car CAD models[1] for the scene of street parking are shown in Fig.3(a).

We assume that the occlusion configurations are affected by following factors: *car type* t, *orientation* ρ , *relative position* r and *camera view* Π. To generate a configuration, we randomly choose corresponding values for these factors, where for each car with type i, $\rho_i \in \{\text{frontal,rear}\}$, $r_i = r_i^{(0)} + \mathrm{d}r$, where $r_i^{(0)}$ is the nominated position for the i-th car on the 3×3 grid, and $\mathrm{d}r = (\mathrm{d}x, \mathrm{d}y)$ is the relative distance (along x axis and y axis) between sampled position and nominated position of the i-th car. We assume the camera view is in the range of $\text{azimuth} \in [0, 2\pi]$ and $\text{elevation} \in [0, \pi/4]$, and discretize the view space into B view bins uniformly along the azimuth angle. By changing values of these parameters, we can generate many different occlusion images for further processing.

As is shown in Fig.3(a), we manually segment a car into 17 parts (considering symmetry and simplicity), which are coded with different colors. We generate two images for each of the factor value combination, one as shown on the top of the third column in Fig.3(a), and the other using only the color coded car in the center, which is shown immediately below. In this way, we can detect if a part is occluded by simply comparing the areas of that part on the two images. We assume a part is occluded if 60 percent of that part is not visible.

4.2. Constructing initial AOT

With the part-level visibility information, we could get two vectors for each occlusion configuration. The first one is a (17 parts$\times B$ views) dimension binary valued vector v for the visibilities of object parts, and the second one is a real valued ((1 root $+17$ parts) $\times B$ 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 $\mathcal{D}$, where the first few rows of this matrix for $B = 8$ is shown in the last column of Fig.3(a). Note that we have partitioned the view space into B views, so for each row, the visible parts always concentrate in a

[1] from www.doschdesign.com and Google 3d warehouse

segment of the vector representing that view.

To get an initial AOT, we assume that each row in $\mathcal{D}$ 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 AOT are shown on the top of Fig.3(b). Here each row of $\mathcal{D}$ represents an occlusion configuration, and each column represents a part. The part is either visible (white), or occluded (gray). For better viewing, we just show the sub-trees for the $1st$ and $9th$ rows.

4.3. Refining AOT Structure

The initial AOT can be very large and redundant, since it has many duplicated occlusion configurations (i.e. duplicated rows in $\mathcal{D}$) and a combinatorial number of part compositions. In the following, we will pursue a compact AND-OR structure from the initial AOT. The problem can be formulated as:

$$\min \sum_{i}^{N} \mid v_i - v_i(AOT) \mid_2^2 + \lambda \mid AOT \mid \qquad (5)$$

where v_i is the i-th row of the data matrix $\mathcal{D}$, $v(AOT)$ returns its most approximate occlusion configuration generated by the AOT, $|AOT|$ is the number of nodes and edges in the structure, and λ 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.5 using a modified graph compression algorithm similar to [27]. As illustrated in Fig.3(b), the algorithm starts from the large initial AOT described above, and iteratively combines branches as long as the introduced loss is smaller than the decrements in complexity term $\lambda|AOT|$. 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.3(b). As there are consistently visible parts for each view, the algorithm will quickly converge to the structure similar to Fig.2 (except the generalized parts in layer 1, and they can be added afterwards).

With the refined AND-OR structure, 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, which will be introduced later.

5. Parameters Learning and Model Inference

5.1. Discriminative Learning by LSSVM

We propose to learn the parameters of our model using the LSSVM [34, 32]. The latent variables in our model and the way to initialize them are listed as follows:

The view and occlusion configuration of each object bounding box, which is related to the branching variable V of the root OR node in layer 0 and the branching variable $\omega(O)$ for OR nodes in layer 2 (see Fig.2). In practice, there are two steps for initialization: (i) after the AND-OR structure is learned, we train a temporary model just on synthetic data. (ii) then we use the temporary model to "infer" the view and occlusion configuration of each training positive on real data. This method is convenient and effective, since these information are hard to annotate on real data and synthetic data provides good gradient cues.

The location and bounding box for each visible part under corresponding occlusion configuration. In training, we initialize them using the corresponding attributes in the learned AND-OR structure, which is described in the end of Section 4.3.

All the latent variables will be re-estimated during training by solving argmax instead of $\max$ in Eqn.(4)(2). We use the concave-convex procedure (CCCP) [33] to solve our problem, where the details can be found in [33, 32, 34].

For the task of view estimation, to cope with the problem that objects are correctly localized but by wrong view branches, we extend the loss function L in [1] as:

$$L(y_n, y, z) = \begin{cases} 0, & y_n = y = -1 \\ 1 - \frac{A(y_n) \cap A(y)}{A(y_n) \cup A(y)}, & (y_n = y = +1) \wedge (V_n = V) \\ 1, & otherwise \end{cases} \qquad (6)$$

where $A(\cdot)$ is the area of a window, z is the latent variable in our model, y_n and V_n are the groundtruth label and view annotation for the n-th training positive, y and V are the accordingly predicted values.

5.2. Inference by Dynamic Programming (DP)

Since our AND-OR structure is a DAG, the inference can be efficiently accomplished by DP. The algorithm consists of a bottom-up pass and a top-down pass. The bottom-up pass places our model at all positions and scales of the image to compute appearance and geometry score maps for every terminal node, as well as these for the AND and OR nodes. This could be efficiently done by using the generalized distance transform [6]. Then on the score map of root OR node G, we find the positions u that $\mathrm{S}(G, u)$ are greater than a threshold and execute the top-down pass to retrieve the latent variables. To eliminate multiple detections of the same object instance, we use the non maximum suppression (NMS) to get the final detection.

6. Experiments

6.1. A Street Parking Car Dataset

There are several datasets featuring a large amount of car images [18, 26, 23, 4, 5], 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 [8] contains occluded cars parked along the streets, but it is not a good dataset for our task either, as 1) the car views are rather fixed as the video sequences are captured from a car driving on the road, and 2) the evaluation protocol only counts cars with no or mild occlusion ($< 20\%$) in the testing set. To evaluate our model on occlusion handling, we developed a large scale car dataset emphasizing street parking cars with a large amount of occlusions and diverse viewpoint changes (see Fig.7(b)). The dataset is composed of 881 images, most of which are collected by searching the internet and capturing cars on the streets around our campus, besides, we also collect and annotate some street scene images from [25, 4]. Fig.4 shows the bounding box overlap distribution and average number of cars per image on our dataset. These two statistics can be viewed as the summary of car occlusion distribution. For image annotation, we adopt the weak annotation strategy, and just label the bounding boxes of cars in each image. We split the dataset into training and testing sets containing 440 and 441 images, respectively. This dataset will be released to the public.

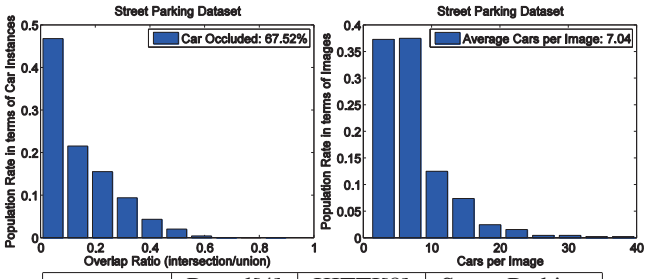
6.2. Detection

To evaluate our model on car detection task, we train and test our model on three datasets. In all cases, we use the DPM as the baseline model for comparison, as this is a very competitive model with source code publicly available. In practice, we use an experimental method to select the number of components for DPM[2] and the number of views for our model. Specifically, we train DPM with $(2, 4, \ldots, 24)$ components, and our model with $(6, 8, 10)$ views on each of the following datasets. We use the best models of each method for comparison.

Synthetic Dataset: In the first experiment, we test the effectiveness of our AND-OR structure in representing different part occlusion configurations. For this purpose, we generate a synthetic dataset using 5040 3 cars synthetic images as our training data, and a mixture of 3000 3 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 [22] and apply Gaussian blur to reduce the boundary effects. Samples of both training and testing data can be seen in Fig.7(a). 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 left of Fig.5, our model outperforms the DPM by 7.2% in AP. This experiment validates that our AND-OR model builds a flexible structure that can be used to explicitly model occlusion.

Street Parking Dataset: We further compare our model with the DPM on our self-collected dataset. On this dataset,

[2]We use source code from [10], and fix the number of latent parts to 8.



	Pascal[4]	KITTI[8]	Street Parking
Avg. cars	1.75	≈ 3	6.67

Figure 4. Top: we show the distribution of overlap ratio and cars per image on our dataset. Bottom: we compare our dataset with [4, 8] by the average number of cars within an image.

we directly use the AND-OR structure learned in previous experiment (but retrain the model parameters using the real images), and the best DPM has 20 components. We show the results of both models in the middle of Fig.5. From the figure, we can see that our model also outperforms the DPM by 5.8% in AP, demonstrating that the AND-OR structure learned from synthetic data could be applied to the real scenarios. This also suggests that the occlusion configurations in the synthetic data matches the real occlusion cases. Compared with the previous experiment, our model shows smaller performance increase from the DPM method. We attribute this to the more cluttered background.

Fig.7(b) shows some examples of car detection results by our model. The red rectangle shows the successful cases, the blue bounding boxes show the missing detections (we omit the cars smaller than 1000 pixels), and the green bounding box shows the false alarms. From these examples, it can be seen our model is able to detect the cars with small, moderate occlusions as well as a considerable amount of cars with severe occlusions. The failure cases are mainly caused by severe occlusions (greater than 60% of the car area is occluded), other occluders (e.g., trucks, trees, etc.), and too large or too small bounding boxes (i.e., bounding boxes includes more than one car or only a part of one car).

Pascal VOC 2007 Car Dataset: On the popular Pascal VOC 2007 dataset [4], car-to-car occlusions are much less frequent. This dataset is considered as a challenging dataset, but as is analyzed in [15], there will be very slight performance gain even if all the occlusions can be handled successfully. So we mainly use this dataset to show that our model can also be used in the general car detection task.

To approximate the occlusion configurations observed on this dataset, we generate synthetic images with car-to-car occlusion as well as with only car self-occlusions. For the car-to-car occlusions, we use the full 3×3 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

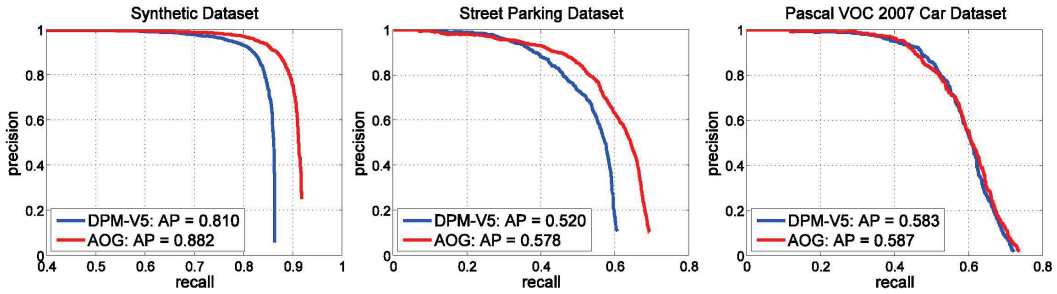


Figure 5. Car detection performance comparisons in terms of Precision-Recall curves on synthetic dataset, street parking dataset and VOC 2007 car dataset [4].

structure has 6 views with 10 occlusion branches, 5 layers and 109 nodes.

As is shown on the right of Fig.5, we can see that the performance of our AND-OR structure model is comparable with DPM. In detailed analysis, our model gets only a little more recall than DPM, which meets the analysis in [15]. This experiment demonstrates that our AND-OR structure method does not lose performance in general dataset.

6.3. View Estimation

To verify the capability of our model on view estimation, we report the mean precision in pose estimation (MPPE) on both Pascal VOC 2006 car dataset [5] and 3D Car dataset [26] following the protocol in [12] and [26], respectively. Since for view estimation, these two popular datasets both emphasize visible cars, we model our AND-OR structure using images only with self-occlusion. Table 1 shows a comparison of our model with state-of-the-art methods on these two datasets. We can see, our model is comparable or better than recently proposed models.

Pascal VOC 2006 Car Dataset[5]					
	DPM	[21]	[12]	[28]	ours
MPPE	0.69	0.73	**0.86**	0.57	0.73

3D Car Dataset [26]							
	DPM	[21]	[19]	[11]	$[24]^1$	$[24]^2$	ours
AP	99.6	96	76.7	99.2	**99.9**	99.7	**99.9**
MPPE	86.3	89	70	85.3	**97.9**	96.3	94

Table 1. View Estimation on Pascal VOC 2006 Car Dataset [5] and 3D Car Dataset [26]. $[24]^1$ and $[24]^2$ refer to DPM-VOC+VP and DPM-3D-Constraints, respectively.

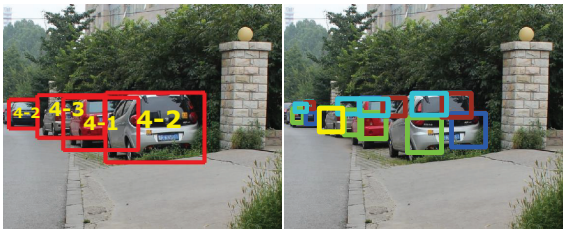


Figure 6. For each predicted window, our algorithm is also capable of estimating the view and occlusion configuration (left), and localizing object parts (right). Parts are colored as in CAD models in Fig.3(a). See text for details.

7. Discussion

This paper presents a discriminative AND-OR structure to model occlusions. The AND-OR structure is a DAG with each subtree consisting of consistently visible parts and optional part clusters. The structure of our model is learned with the help of CAD simulation, where its parameters can be trained using LSSVM. Experiments show that our CAD simulation strategy is effective and our model is better than the state-of-the-art model [6, 10] in terms of car detection and view estimation. In future work, we would like to further speed up the inference algorithm, and apply this model to other categories such as pedestrians.

Beyond Bounding Box Localization. As our model uses weakly semantic parts from synthetic training images, the trained model can also be used to estimate the view and visible parts, even though such supervision is not provided in the training set. By simply tracking down the hidden variables V and O that supports the object bounding box, Fig.6 shows the estimated views and visible part locations of a sample testing image. Here, string x-y means the car is in view x with occlusion configuration y, corresponding to the x-th AND node in layer 1 and y-th AND node within the corresponding subtree in layer 3 in our AND-OR structure. The small bounding boxes with different colors on the right image shows the locations of different semantic parts, where the parts are colored in the same way as in Fig.3(a). As a joint hierarchical representation, we believe our model could be applied to other vision tasks such as modeling activities involving object and human interactions.

Acknowledgment. B. Li is supported by 973 Program of China 2012CB316300, W. Hu, T. Wu and S.C. Zhu are supported by NSF IIS 1018751 and DARPA MSEE project FA 8650-11-1-7149.

References

[1] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008.

[2] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[3] G. Duan, H. Ai, and S. Lao. A structural filter approach to human detection. In *ECCV*, 2010.

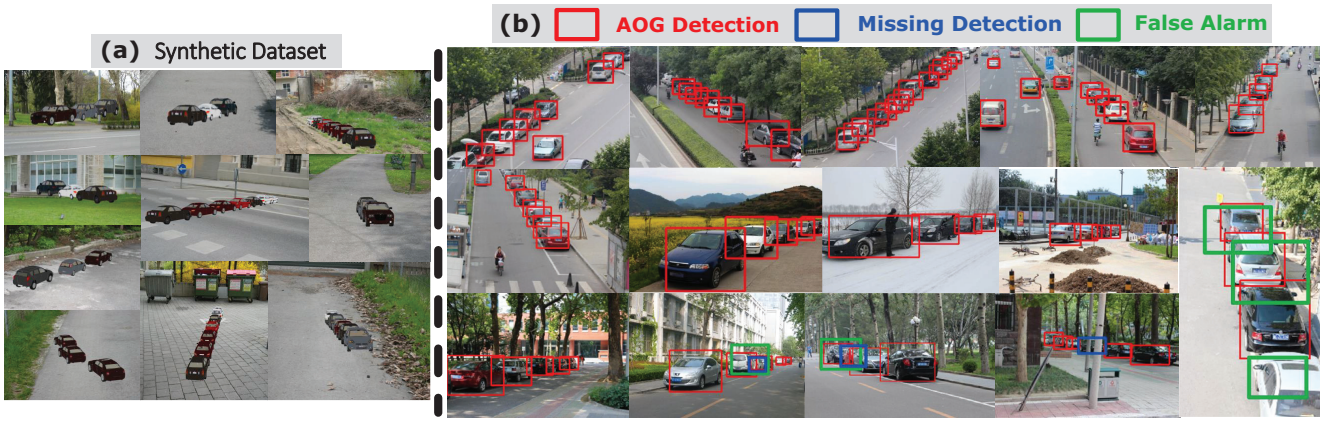


Figure 7. **(a)** Example images of the synthetic dataset. **(b)** Examples of successful and failure cases by our model on street parking dataset. Best viewed in color and zooming in.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[5] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool. The PASCAL Visual Object Classes Challenge 2006 (VOC2006) Results. http://www.pascal-network.org/challenges/VOC/voc2006/results.pdf.

[6] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *PAMI*, 2010.

[7] T. Gao, B. Packer, and D. Koller. A segmentation-aware object detection model with occlusion handling. In *CVPR*, 2011.

[8] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012.

[9] R. Girshick, P. Felzenszwalb, and D. McAllester. Object detection with grammar models. In *NIPS*, 2011.

[10] R. B. Girshick, P. F. Felzenszwalb, and D. McAllester. Discriminatively trained deformable part models, release 5. http://people.cs.uchicago.edu/ rbg/latent-release5/.

[11] D. Glasner, M. Galun, S. Alpert, R. Basri, and G. Shakhnarovich. Viewpoint-aware object detection and pose estimation. In *ICCV*, 2011.

[12] C. Gu and X. Ren. Discriminative Mixture-of-Templates for Viewpoint Classification. In *ECCV*, 2010.

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

[14] G. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 2006.

[15] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. In *ECCV*, 2012.

[16] E. Hsiao and M. Hebert. Occlusion reasoning for object detection under arbitrary viewpoint. In *CVPR*, 2012.

[17] W. Hu. Learning 3d object templates by hierarchical quantization of geometry and appearance spaces. In *CVPR*, 2012.

[18] B. Leibe and B. Schiele. Analyzing appearance and contour based methods for object categorization. In *CVPR*, 2003.

[19] J. Liebelt and C. Schmid. Multi-view object class detection with a 3D geometric model. In *CVPR*, 2010.

[20] J. Liebelt, C. Schmid, and K. Schertler. Viewpoint-independent object class detection using 3d feature maps. In *CVPR*, 2008.

[21] R. J. Lopez-Sastre, T. T., and S. Savarese. Deformable part models revisited: A performance evaluation for object category pose estimation. In *ICCV-WS CORP*, 2011.

[22] A. Opelt and A. Pinz. Object Localization with Boosting and Weak Supervision for Generic Object Recognition. In *SCIA*, 2005.

[23] M. Ozuysal, V. Lepetit, and P. Fua. Pose estimation for category specific multiview object localization. In *CVPR*, 2009.

[24] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3d geometry to deformable part models. In *CVPR*, 2012.

[25] B. C. Russell, A. B. Torralba, K. P. Murphy, and W. T. Freeman. LabelMe: A Database and Web-Based Tool for Image Annotation. *IJCV*, 2008.

[26] S. Savarese and L. Fei-Fei. 3d generic object categorization, localization and pose estimation. In *ICCV*, 2007.

[27] Z. Si and S.-C. Zhu. Learning and-or templates for object recognition and detection. *PAMI*, 2013.

[28] M. Sun, H. Su, S. Savarese, and F. fei Li. A multi-view probabilistic model for 3D object classes. In *CVPR*, 2009.

[29] X. Wang, T. Han, and S. Yan. An hog-lbp human detector with partial occlusion handling. In *ICCV*, 2009.

[30] B. Wu and R. Nevatia. Detection and tracking of multiple, partially occluded humans by bayesian combination of edgelet based part detectors. *IJCV*, 2007.

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

[32] C. Yu and T. Joachims. Learning structural svms with latent variables. In *ICML*, 2009.

[33] A. L. Yuille and A. Rangarajan. The Concave-Convex Procedure (CCCP). In *NIPS*, 2001.

[34] L. Zhu, Y. Chen, A. Yuille, and W. Freeman. Latent hierarchical structural learning for object detection. In *CVPR*, 2010.

[35] S.-C. Zhu and D. Mumford. A stochastic grammar of images. *Found. Trends. Comput. Graph. Vis.*, 2006.