Monocular 3D Object Detection for Autonomous Driving

Xiaozhi Chen[1], Kaustav Kundu[2], Ziyu Zhang[2], Huimin Ma[1], Sanja Fidler[2], Raquel Urtasun[2]
[1]Department of Electronic Engineering, Tsinghua University
[2]Department of Computer Science, University of Toronto
{chenxz12@mails., mhmpub@}tsinghua.edu.cn, {kkundu, zzhang, fidler, urtasun}@cs.toronto.edu

Abstract

The goal of this paper is to perform 3D object detection from a single monocular image in the domain of autonomous driving. Our method first aims to generate a set of candidate class-specific object proposals, which are then run through a standard CNN pipeline to obtain high-quality object detections. The focus of this paper is on proposal generation. In particular, we propose an energy minimization approach that places object candidates in 3D using the fact that objects should be on the ground-plane. We then score each candidate box projected to the image plane via several intuitive potentials encoding semantic segmentation, contextual information, size and location priors and typical object shape. Our experimental evaluation demonstrates that our object proposal generation approach significantly outperforms all monocular approaches, and achieves the best detection performance on the challenging KITTI benchmark, among published monocular competitors.

1. Introduction

In recent years, autonomous driving has been a focus of attention of both industry as well as the research community. Most initial efforts rely on expensive LIDAR systems, such as the Velodyne, and hand-annotated maps of the environment. In contrast, recent efforts try to replace the LIDAR with cheap on-board cameras, which are readily available in most modern cars. This is an exciting time for the vision community, as this application domain provides us with many interesting challenges.

The focus of this paper is on high-performance 2D and 3D object detection from monocular imagery in the context of autonomous driving. Most of the recent object detection pipelines [19, 20] typically proceed by generating a diverse set of object proposals that have a high recall and are relatively fast to compute [45, 2]. By doing this, computationally more intense classifiers such as CNNs [28, 42] can be devoted to a smaller subset of promising image regions, avoiding computation on a large set of futile candidates. Our paper follows this line of work.

Different types of object proposal methods have been developed in the past few years. A common approach is to over-segment the image into superpixels and group these using several similarity measures [45, 2]. Approaches that efficiently explore an exhaustive set of windows using simple “objectness” features [1, 11], or contour information [55] have also been proposed. The most recent line of work aims to learn how to propose promising object candidates using either ensembles of binary segmentation models [27], parametric energies [29] or window classifiers based on CNN features [18].

These proposal generation approaches have been shown to be very effective in the context of the PASCAL VOC challenge, which require a rather loose notion of localization, i.e., a detection is said to be correct if it overlaps more than 50% with the ground truth. In the context of autonomous driving, however, a much more strict overlap is required, in order to provide a more accurate estimate of the distance from the ego-car to the potential obstacles. As a consequence, popular approaches, such as R-CNN [20] fall significantly behind the competitors on autonomous driving benchmarks such as KITTI [16]. The current leader on KITTI is Chen et al. [10], which exploits stereo imagery to create accurate 3D proposals. However, most cars are currently equipped with a single camera, and thus monocular object detection is of crucial importance.

Inspired by this approach, this paper proposes a method that learns to generate class-specific 3D object proposals with very high recall by exploiting contextual models as well as semantics. These proposals are generated by exhaustively placing 3D bounding boxes on the ground-plane and scoring them via simple and efficiently computable image features. In particular, we use semantic and object instance segmentation, context, as well as shape features and location priors to score our boxes. We learn per-class weights for these features using S-SVM [24], adapting to each individual object class. The top object candidates are then scored with a CNN, resulting in the final set of detec-

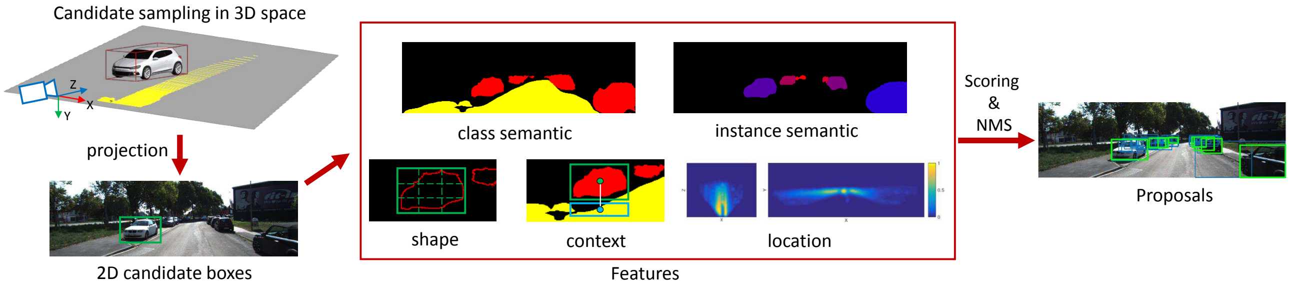


Figure 1: **Overview of our approach:** We sample candidate bounding boxes with typical physical sizes in the 3D space by assuming a prior on the ground-plane. We then project the boxes to the image plane, thus avoiding multi-scale search in the image. We score candidate boxes by exploiting multiple features: class semantic, instance semantic, contour, object shape, context, and location prior. A final set of object proposals is obtained after non-maximum suppression.

tions. Our experiments show that our approach is able to perform really well on KITTI, outperforming all published monocular object detectors and being almost on par with the leader [10], which exploits stereo imagery.

2. Related Work

Our work is related to methods for object proposal generation, as well as monocular 3D object detection. We will mainly focus our literature review on the domain of autonomous driving.

Significant progress in deep neural nets [28, 42] has brought increased interest in methods for object proposal generation since deep nets are typically computationally demanding, making sliding window challenging [20]. Most of the existing work on proposal generation uses RGB [45, 55, 9, 2, 11, 29], RGB-D [4, 21, 31, 25], or video [35]. In RGB, most methods combine superpixels into larger regions via several similarity functions using e.g. color and texture [45, 2]. These approaches prune the exhaustive set of windows down to about 2K proposals per image achieving almost perfect recall on PASCAL VOC [12]. [9] defines parametric affinities between pixels and finds the regions using parametric min-cut. The resulting regions are then scored via simple features, and the top-ranked proposals are used in recognition tasks [8, 15, 53]. Exhaustively sampled boxes are scored using several "objectness" features in [1]. BING proposals [11] score boxes based on an object closure measure as a proxy for "objectness". Edgeboxes [55] score an exhaustive set of windows based on contour information inside and on the boundary of each window.

The most related approaches to ours are recent methods that aim to learn how to propose objects. [29] learns parametric energies in order to propose multiple diverse regions. In [27], an ensemble of figure-ground segmentation models are learnt. Joint learning of the ensemble of local and global binary CRFs enables the individual predictors to specialize in different ways. [26] learned how to place promising object seeds and employ geodesic distance transform to obtain candidate regions. Parallel to our work, [18] introduced a method that generates object proposals by cascading the layers of the convolutional neural network. The method is efficient since it explores an exhaustive set of windows via integral images over the CNN responses. Our approach also exploits integral images to score the candidates, however, in our work we exploit domain priors to place 3D bounding boxes and score them with semantic features. We use pixel-level class scores from the output layer of the grid CNN, as well as contextual and shape features.

In RGB-D, [10] exploited stereo imagery to exhaustively scored 3D bounding boxes using a conditional random field with several depth-informed potentials. Our work also evaluates 3D bounding boxes, but uses semantic object and instance segmentation and 3D priors to place proposals on the ground plane. Our RGB potentials are partly inspired by [15, 53] which exploits efficiently computed segmentation potentials for 2D object detection.

Our work is also related to detection approaches for autonomous driving. [54] first detects a candidate set of objects via a poselet-like approach and then fits a deformable wireframe model within the box. [38] extends DPM [13] to 3D by linking parts across different viewpoints, while [14] extends DPM to reason about deformable 3D cuboids. [34] uses an ensemble of models derived from visual and geometrical clusters of object instances. Regionlets [32] proposes boxes via Selective Search and re-localizes them using a top-down approach. [46] introduced a holistic model that re-reasons about DPM object candidates via cartographic priors. Recently proposed 3DVP [47] learns occlusion patterns in order to significantly improve performance of occluded cars on KITTI.

3. Monocular 3D Object Detection

In this paper, we present an approach to object detection, which exploits segmentation, context as well as location priors to perform accurate 3D object detection. In particular, we first make use of the ground plane in order to propose objects that lie close to it. Since our input is a single monocular image, our ground-plane is assumed to be orthogonal to

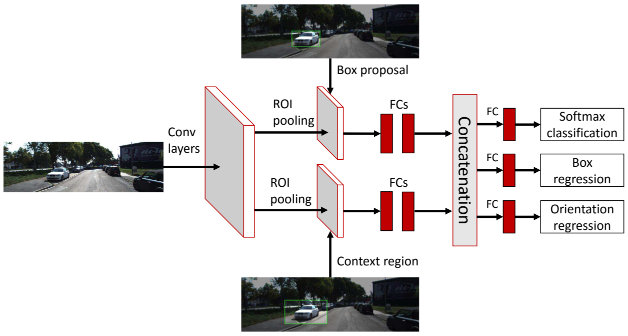


Figure 2: CNN architecture adopted from [10] used to score our proposals for object detection and orientation estimation.

the image plane and a distance down from the camera, the value of which we assume to be known from calibration. Since this ground-plane may not reflect perfect reality in each image, we do not force objects to lie on the ground, and only encourage them to be close. The 3D object candidates are then exhaustively scored in the image plane by utilizing class segmentation, instance level segmentation, shape, contextual features and location priors. We refer the reader to Fig. 1 for an illustration. The resulting 3D candidates are then sorted according to their score, and only the most promising ones (after non-maxima suppression) are further scored via a Convolutional Neural Net (CNN). This results in a fast and accurate approach to 3D detection.

3.1. Generating 3D Object Proposals

We represent each object with a 3D bounding box, $\mathbf{y} = (x, y, z, \theta, c, t)$, where (x, y, z) is the center of the 3D box, θ denotes the azimuth angle and $c \in C$ is the object class (*Cars*, *Pedestrians* and *Cyclists* on KITTI). We represent the size of the bounding box with a set of representative 3D templates t, which are learnt from the training data. We use 3 templates per class and two orientations $\theta \in \{0, 90\}$. We then define our scoring function by combining semantic cues (both class and instance level segmentation), location priors, context as well as shape:

$$\begin{aligned} E(\mathbf{x}, \mathbf{y}) = &\mathbf{w}_{c,sem}^{\top} \phi_{c,sem}(\mathbf{x}, \mathbf{y}) + \mathbf{w}_{c,inst}^{\top} \phi_{c,inst}(\mathbf{x}, \mathbf{y}) + \\ &\mathbf{w}_{c,cont}^{\top} \phi_{c,cont}(\mathbf{x}, \mathbf{y}) + \mathbf{w}_{c,loc}^{\top} \phi_{c,loc}(\mathbf{x}, \mathbf{y}) + \\ &\mathbf{w}_{c,shape}^{\top} \phi_{c,shape}(\mathbf{x}, \mathbf{y}) \end{aligned}$$

We next discuss each of these potentials in more detail.

Semantic segmentation: This potential takes as input a pixelwise semantic segmentation containing multiple semantic classes such as car, pedestrian, cyclist and road. We incorporate two types of features encoding semantic segmentation. The first feature encourages the presence of an object inside the bounding box by counting the percentage of pixels labeled as the relevant class:

$$\phi_{c,seg}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i \in \Omega(\mathbf{y})} S_c(i)}{|\Omega(\mathbf{y})|},$$

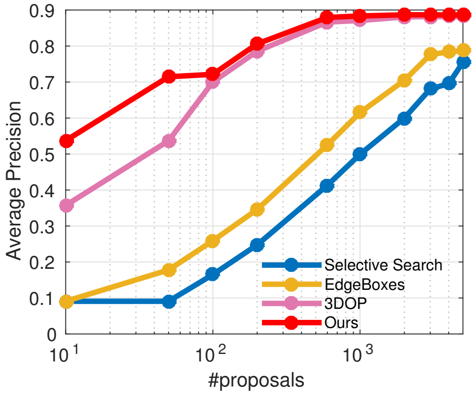


Figure 3: AP vs #proposals on *Car* for moderate setting.

with $\Omega(\mathbf{y})$ the set of pixels in the 2D box generated by projecting the 3D box $\mathbf{y}$ to the image plane, and S_c the segmentation mask for class c. The second feature computes the fraction of pixels that belong to classes other than the object class

$$\phi_{c,non-seg,c'}(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i \in \Omega(\mathbf{y})} S_{c'}(i)}{|\Omega(\mathbf{y})|},$$

This feature is two dimensional, as one dimension contains the road and the other aggregates all other classes (but the class of the proposal). Hence this potential tries to minimize the fraction of pixels inside the bounding box belonging to other classes. Note that these features can be computed very efficiently using as many integral images as classes. In this paper we use [41, 52, 3] to compute the semantic segmentation features. [41, 52] jointly learn the convolutional features as well as the pairwise Gaussian MRF potentials to smooth the output labeling. SegNet [3] performs semantic labeling via a fully convolutional encoder-decoder. In particular, we use the pre-trained model on PASCAL VOC + COCO from [52] for *Car* segmentation. To reduce discrepancies of surrogate classes, we use the pre-trained SegNet model from [3] for *Pedestrian* and *Cyclist* segmentation. Note that very few semantic annotations are available for KITTI and thus we did not fine-tune their models. Additionally, we exploited the annotations in the road benchmark of KITTI, and fine-tuned the network of [41] for road.

Shape: This feature captures the shape of the objects. Specifically, we first compute the contours in the output of the segmentation (instead of the original image). We then create two grids for the 2D candidate box, one containing only a single cell and one that has $K \times K$ cells. For each cell, we count the number of contour pixels inside it. Overall, this gives us a $(1 + K \times K)$ feature vector across all cells. This potential tries to place a bounding box tightly around the object, encouraging the spatial distribution of contours within its grid to match the expected shape of a specific class. These features can be computed very efficiently using an integral image (counting contour pixels).

Instance Segmentation: Similar to [15, 53], we exploit instance level segmentation features, which score the amount of segment inside the box and outside the box. However, we simply choose the best segment for each bounding box based on the IoU overlap, and not reason about the segment ID at inference time. This speeds up computation. This feature helps us to detect objects that are occluded as they form different instances. Note that these features can be very efficiently computed using as many integral images as instances that compose the segmentation. To compute instance-level segmentation we exploit the approach of [51, 50], which uses a CNN to create both instance-level pixel labeling as well as ordering in depth. We re-trained their model so that no overlap (not even in terms of sequences) exist between our training and validation. Note that this approach is only available for *Cars*.

Context: This feature encodes the presence of contextual labels, e.g. cars are on the road, and thus we can see road below them. We use a rectangle below the 2D projection of the 3D bounding box as the contextual region. We set its height to 1/3 of the height of the box, and use the same width, as in [33]. We then compute the semantic segmentation features in the contextual region. We refer the reader to Fig. 1. for an illustration.

Location: This feature encodes a location prior of objects in both birds-eye perspective as well as in the image plane. We learn the prior using kernel density estimation (KDE) with a fixed standard deviation of $4m$ for the 3D prior and 2 pixels for the image domain. The 3D prior is learned using the 3D ground-truth bounding boxes available in [16]. We visualize the prior in Fig. 1.

3.2. 3D Proposal Learning and Inference

We use exhaustive search as inference to create our candidate proposals. This can be done efficiently as all the features can be computed with integral images. In particular, it takes 1.8s in a single core, but inference can be trivially parallelized to be real time. We learn the weights of the model using structured SVM [44]. We use the parallel cutting plane implementation of [40]. We use 3D Intersection-over-Union (IoU) as our task loss.

3.3. CNN Scoring of Top Proposals

In this section, we describe how the top candidates (after non-maxima suppression) are further scored via a CNN. We employ the same network as in [10], which for completeness we briefly describe here. The network is built using the Fast R-CNN [19] implementation. It computes convolutional features from the whole image and splits it into two branches after the last convolutional layer, i.e., *conv5*. One branch encodes features from the proposal regions while another is specific to context regions, which are obtained by enlarging the proposal regions by a factor of 1.5, following [53]. Both branches are composed of a RoI pooling layer and two fully-connected layers. RoIs are obtained by projecting the proposals or context regions onto the *conv5* feature maps. We obtain the final feature vectors by concatenating the output features from the two branches. The network architecture is illustrated in Fig. 2.

We use a multi-task loss to jointly predict category labels, bounding box offsets, and object orientation. For background boxes, only the category label loss is employed. We weight each loss equally, and define the category loss as cross entropy, the orientation loss as a smooth ℓ_1 and the bounding box offset loss as a smooth ℓ_1 loss over the 4 coordinates that parameterized the 2D bounding box, as in [20].

3.4. Implementation Details

Sampling Strategy: We discretize the 3D space such that the voxel size is 0.2m along each dimension. To reduce the search space during inference in our proposal generation model, we place 3D candidate boxes on the ground plane. As we only use one monocular image as input, we cannot estimate an accurate road plane. Instead, as the camera location is known in KITTI, we use a fixed ground plane for all images with the normal of the plane facing up along camera's Y axis (assuming that the image plane is orthogonal to the ground plane), and the distance of the camera from the plane is $h_{cam} = 1.65$m. To be robust to ground plane errors (e.g., if the road has a slope), we also sample candidate boxes on additional planes obtained by deviating the default plane by several offsets. In particular, we fix the normal of the plane and set height to $h_{cam} = 1.65 + \delta$. We set $\delta \in \{0, \pm\sigma\}$ for *Car* and $\delta \in \{0, \pm\sigma \ \pm 2\sigma\}$ for *Pedestrian* and *Cyclist*, where σ is the MLE estimate of the standard deviation by assuming a Gaussian distribution of the distance from the objects to the default ground plane. We use more planes for *Pedestrian* and *Cyclist* as small objects are more sensitive to errors. We further reduce the number of sampled boxes by removing boxes inside which all pixels were labeled as road, and those with very low prior probability of 3D location. This results in around 14K candidate boxes per ground plane, template and per image. Our sampling strategy reduces it to 28%, thus speeding up inference significantly.

Network Setup: We use the VGG16 model from [42] trained on ImageNet to initialize our network. We initialize the two branches with the weights of the fully-connected layers of VGG16. To handle particularly small objects in KITTI images, we upscale the input image by a factor of 3.5 following [10], which was found to be crucial to achieve very good performance. We employ a single scale for the images during both training and testing. We use a batch size of $N = 1$ for images and a batch size of $R = 128$ for pro-

- Mono 3D ≃ 2D object Detection

물체 위치? → Region Proposal 추출 → 선택 → classification + regression

7개의 점 Bbox

Mono3D = ① ground plane 가정 + typical physical size → 3D object Bbox proposal 생성

지면

② 각 3D Bbox 좌표 → 2D image Projection

③ Project 된 각 proposal 에 대해 Score 매김.

- Class semantic = semantic segmentation
- instance semantic = instance segmentation
- Shape : 테두리
- Context : 조금 더 큰 Bbox로 주변 픽셀 정보의 맥락도 Scoring

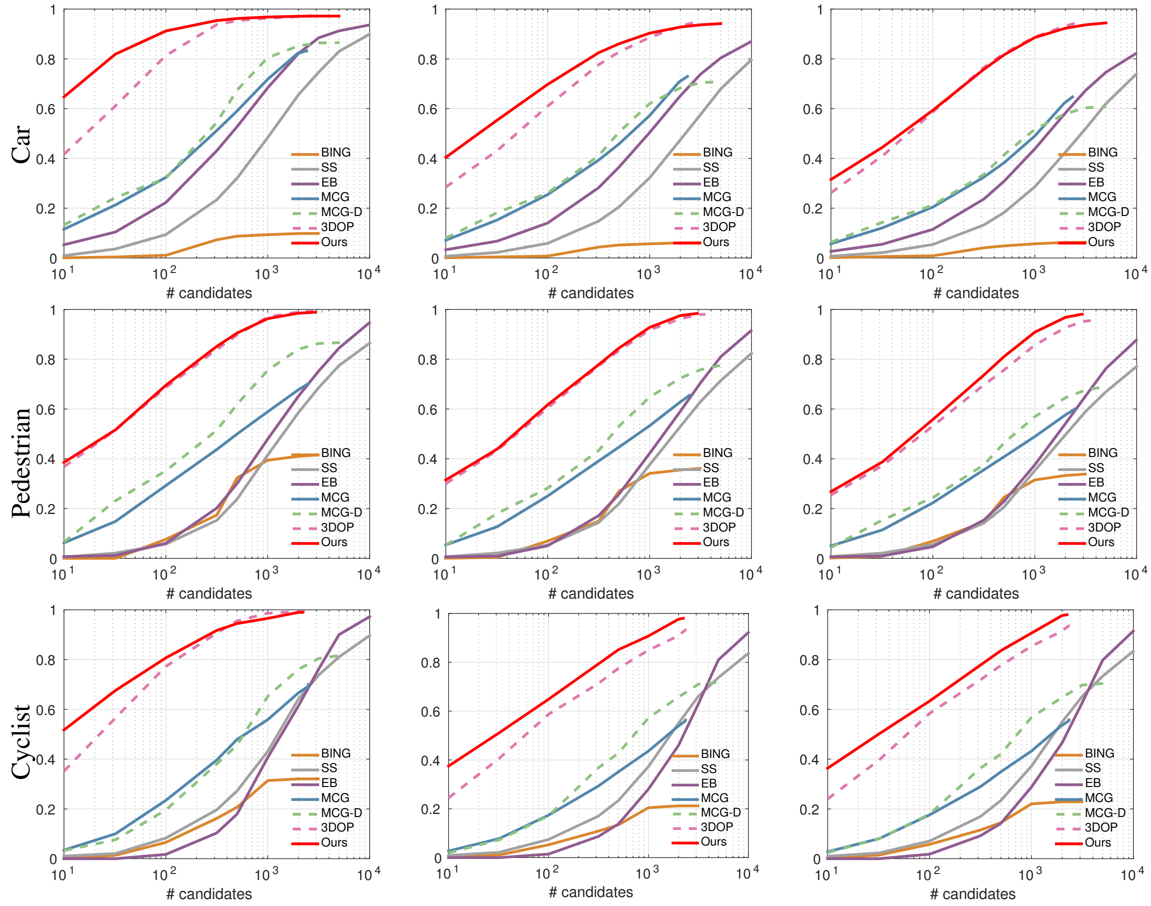
  ex) 차량의 아랫부분 픽셀 = 도로 픽셀
- location : 학습 데이터 상에서 차량이 어느 위치에 있는지 ⇒ 통계

④ NMS ⇒ 중복 proposal 제거 ⇒ 제일 높은 score Proposal 추출

⑤ 남은 Proposal 에 대해 Faster-RCNN Second stage 실행

Input → Conv layer $\xrightarrow{\text{RoI Pooling x2}}$ { Box proposal, Context region } = 동일 크기 feature map → linear layer (FC) →

- Softmax classification : Class
- Box regression : Bbox 미세 조정
- Orientation regression : yaw 미세 조정



(a) Easy (b) Moderate (c) Hard

Figure 4: **Proposal Recall vs #Candidates**. We use an overlap threshold of 0.7 for *Car*, and 0.5 for *Pedestrian* and *Cyclist*. Methods that use depth information are indicated in dashed lines. Note that the comparison to 3DOP [10] and MCG [2] is unfair as we use a monocular image and they use a stereo pair.

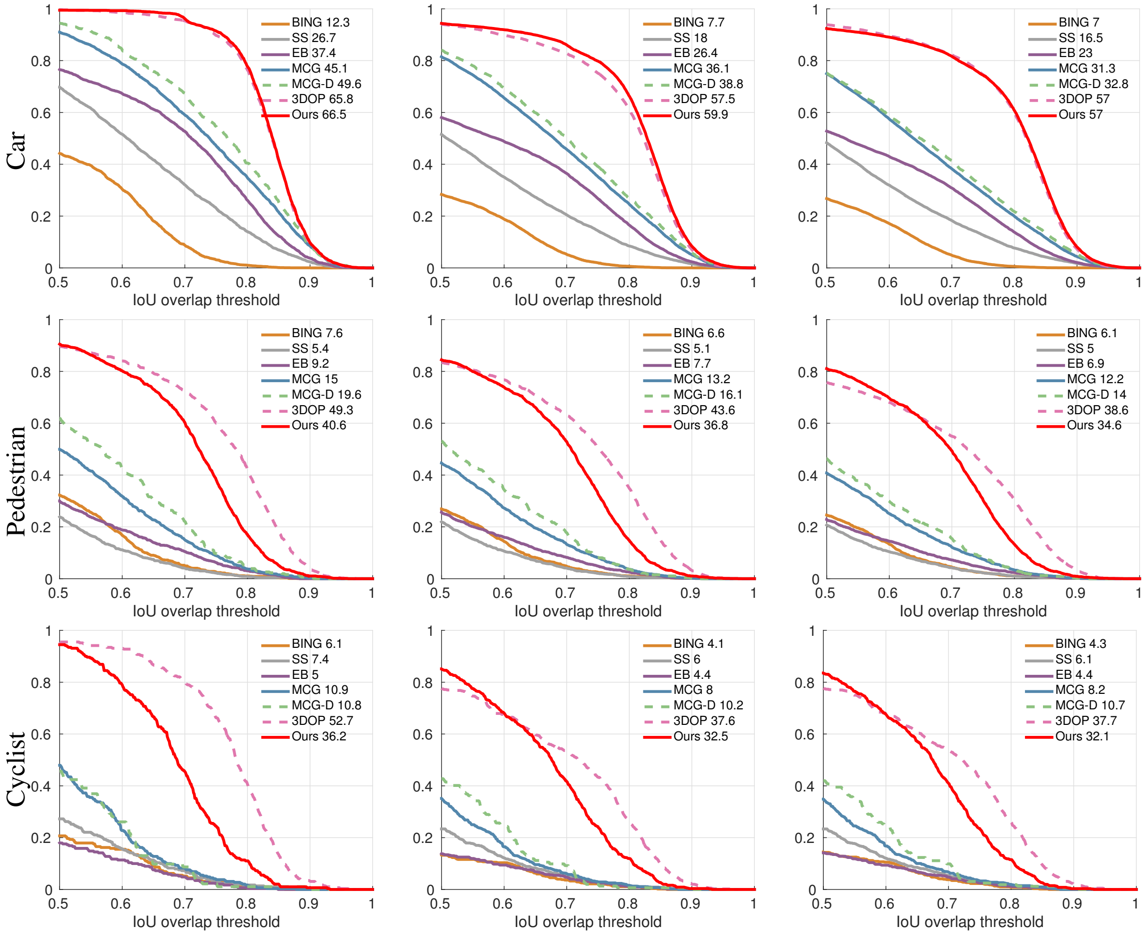
posals. We run SGD with an initial learning rate of 0.001 for 30K iterations and then reduce it to 0.0001 for another 10K iterations.

4. Experimental Evaluation

We evaluate our approach on the challenging KITTI dataset [16]. The KITTI object detection benchmark has three classes: *Car*, *Pedestrian*, and *Cyclist*, with 7,481 training and 7,518 test images. Detection for each class is evaluated in three regimes: *easy*, *moderate*, *hard*, which are defined according to the occlusion and truncation levels of objects. We use the train/val split provided by [10] to evaluate the performance of our class-dependent proposals. The split ensures that images from the same sequence do not exist in both training and validation sets. We then evaluate our full detection pipeline on the test set of KITTI. We refer the reader to the supplementary material for many additional results.

Metrics: We evaluate our class-dependent proposals using best achievable (oracle) recall following [22, 45]. Oracle recall computes the percentage of ground-truth objects covered by proposals with IoU overlap above a certain threshold. We set the threshold to 70% for *Car* and 50% for *Pedestrian* and *Cyclist*, following the KITTI setup. We also report average recall (AR), which has been shown to be highly correlated with detection performance. We also evaluate the whole pipeline of our 3D object detection model on KITTI's two tasks: object detection, and object detection and orientation estimation. Following the standard KITTI setup, we use the Average Precision (AP) metric for the object detection task, and Average Orientation Similarity (AOS) for object detection and orientation estimation task.

Baselines: We compare our proposal generation method to several top-performing approaches on the validation set: 3DOP [10], MCG-D [21], MCG [2], Selective Search (SS) [45], BING [11], and Edge Boxes (EB) [55]. Note that 3DOP and MCG-D exploit depth information, while



(a) Easy (b) Moderate (c) Hard

Figure 5: **Recall vs IoU using 500 proposals**. The number next to the labels indicates the average recall (AR). Note that 3DOP and MCG-D exploit stereo imagery, while the remaining methods as well as our approach use a single monocular image.

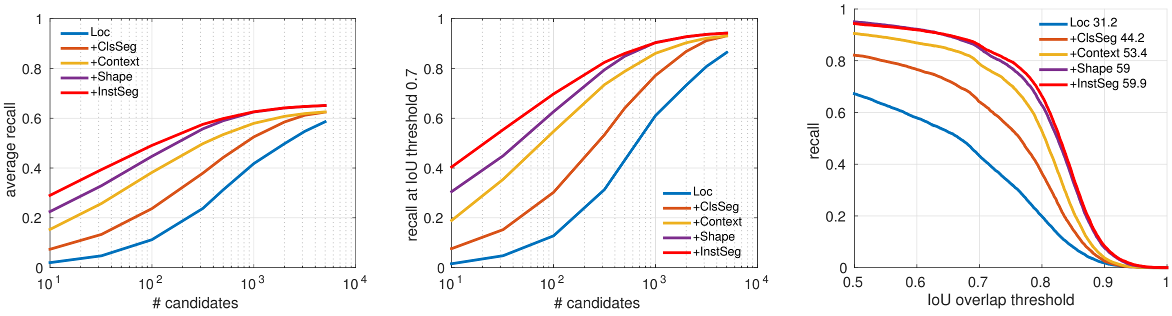


Figure 6: **Ablation study of features on *Car* proposals for moderate data:** From left to right: average recall (AR) vs #candidates, Recall vs #candidates at IoU threshold of 0.7, Recall vs IoU for 500 proposals. We start from the basic model (*Loc*), which only uses location prior feature, and then gradually add other types of features: class semantics, context, shape, and instance semantics.

the remaining methods as well as our approach only use a single RGB image. Note that all of the above approaches, but 3DOP, are class independent (trained to detect any foreground object), while we use class-specific weights as well as semantic segmentation in our features.

Proposal Recall: We evaluate the oracle recall for the generated proposals on the validation set. Fig. 4 shows recall as a function of the number of proposals. Our approach achieves significantly higher recall than all baselines when using less than 500 proposals on *Car* and *Pedestrian*. In

	Cars			Pedestrians			Cyclists		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
LSVM-MDPM-sv [17, 13]	68.02	56.48	44.18	47.74	39.36	35.95	35.04	27.50	26.21
SquaresICF [5]	-	-	-	57.33	44.42	40.08	-	-	-
ACF-SC [6]	69.11	58.66	45.95	51.53	44.49	40.38	-	-	-
MDPM-un-BB [13]	71.19	62.16	48.43	-	-	-	-	-	-
DPM-VOC+VP [38]	74.95	64.71	48.76	59.48	44.86	40.37	42.43	31.08	28.23
OC-DPM [37]	74.94	65.95	53.86	-	-	-	-	-	-
SubCat [34]	84.14	75.46	59.71	54.67	42.34	37.95	-	-	-
DA-DPM [48]	-	-	-	56.36	45.51	41.08	-	-	-
R-CNN [23]	-	-	-	61.61	50.13	44.79	-	-	-
pAUCEnsT [36]	-	-	-	65.26	54.49	48.60	51.62	38.03	33.38
FilteredICF [49]	-	-	-	67.65	56.75	51.12	-	-	-
DeepParts [43]	-	-	-	70.49	58.67	52.78	-	-	-
CompACT-Deep [7]	-	-	-	70.69	58.74	52.71	-	-	-
3DVP [47]	87.46	75.77	65.38	-	-	-	-	-	-
AOG [30]	84.80	75.94	60.70	-	-	-	-	-	-
Regionlets [32]	84.75	76.45	59.70	73.14	61.15	55.21	70.41	58.72	51.83
Faster R-CNN [39]	86.71	81.84	71.12	78.86	65.90	61.18	72.26	63.35	55.90
Ours	**92.33**	**88.66**	**78.96**	**80.35**	**66.68**	**63.44**	**76.04**	**66.36**	**58.87**

Table 1: Average Precision (AP) (in %) on the test set of the KITTI Object Detection Benchmark.

	Cars			Pedestrians			Cyclists		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
AOG [30]	33.79	30.77	24.75	-	-	-	-	-	-
LSVM-MDPM-sv [17, 13]	67.27	55.77	43.59	43.58	35.49	32.42	27.54	22.07	21.45
DPM-VOC+VP [38]	72.28	61.84	46.54	53.55	39.83	35.73	30.52	23.17	21.58
OC-DPM [37]	73.50	64.42	52.40	-	-	-	-	-	-
SubCat [34]	83.41	74.42	58.83	44.32	34.18	30.76	-	-	-
3DVP [47]	86.92	74.59	64.11	-	-	-	-	-	-
Ours	**91.01**	**86.62**	**76.84**	**71.15**	**58.15**	**54.94**	**65.56**	**54.97**	**48.77**

Table 2: AOS scores (in %) on the test set of KITTI's Object Detection and Orientation Estimation Benchmark.

Metric	Proposals	Type	Cars			Pedestrians			Cyclists		
			Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
AP	SS [45]	Monocular	75.91	60.00	50.98	54.06	47.55	40.56	56.26	39.16	38.83
	EB [55]	Monocular	86.81	70.47	61.16	57.79	49.99	42.19	55.01	37.87	35.80
	3DOP [10]	Stereo	93.08	88.07	79.39	71.40	64.46	60.39	83.82	63.47	60.93
	Ours	Monocular	**93.89**	**88.67**	**79.68**	**72.20**	**65.10**	**60.97**	**84.26**	**64.25**	**61.94**
AOS	SS [45]	Monocular	73.91	58.06	49.14	44.55	39.05	33.15	39.82	28.20	28.40
	EB [55]	Monocular	83.91	67.89	58.34	46.80	40.22	33.81	43.97	30.36	28.50
	3DOP [10]	Stereo	91.58	85.80	76.80	61.57	54.79	51.12	**73.94**	**55.59**	**53.00**
	Ours	Monocular	**91.90**	**86.28**	**77.09**	**62.20**	**55.77**	**51.78**	71.95	53.10	51.32

Table 3: Object detection and orientation estimation results on validation set of KITTI. We use 2000 proposals for all methods.

particular, our approach requires only 100 proposals for *Car* and 300 proposals for *Pedestrian* to achieve 90% recall in the *easy* regime. Note that the other 2D methods require orders of magnitude more proposals to reach the same recall. When using 2K proposals, we achieve recall on par with the best 3D approach, 3DOP [10], while being more than 20% higher than other baselines. Note that the comparison to 3DOP [10] and MCG [2] is unfair as we use a monocular image and they use depth information. We also show recall as a function of the overlap threshold for top 500 proposals in Fig. 5. Our approach outperforms the baselines except for 3DOP (which uses stereo) across all IoU thresholds. Compared with 3DOP, we get lower recall at high IoU thresholds on *Pedestrian* and *Cyclist*.

Ablation Study: We study the effects of different features on the object proposal recall in Fig. 6. It can be seen that adding each potential improves performance, particularly at the regime of fewer proposals. The instance semantic feature improves recall especially when using fewer proposals

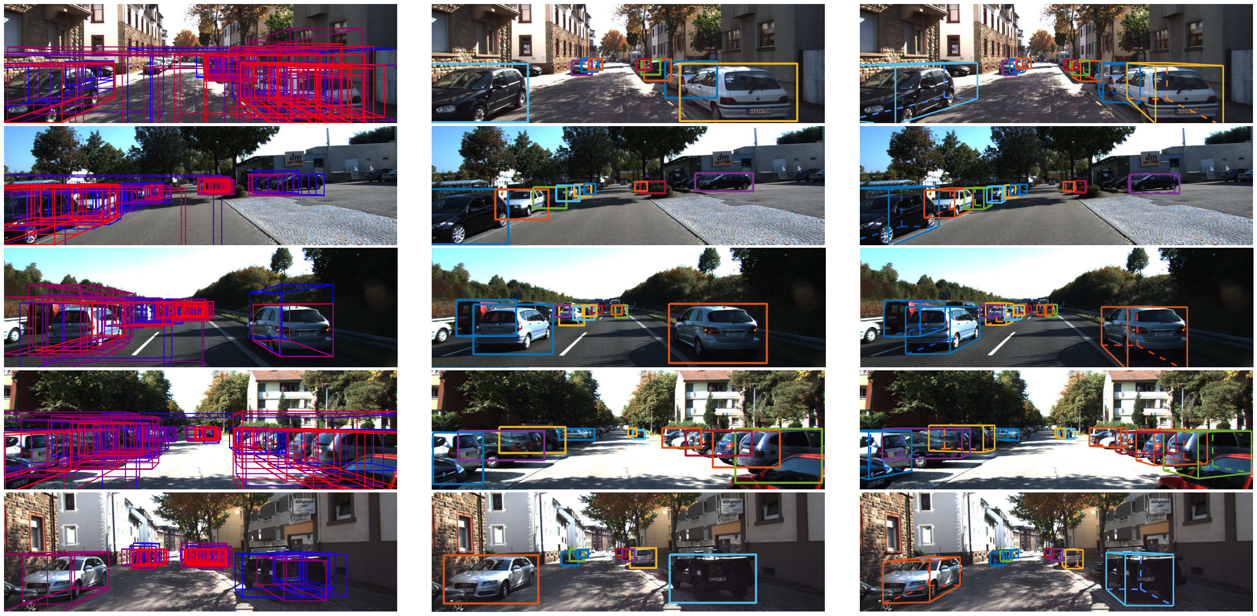


Figure 7: **Qualitative examples of car detections results:** (**left**) top 50 scoring proposals (color from blue to red indicates increasing score), (**middle**) 2D detections, (**right**) 3D detections.

(e.g., < 300). Without the instance feature, we still achieve 90% recall using 1000 proposals. By removing both instance and shape features, we would need twice the number of proposals (i.e., 2000) to reach 90% recall.

Object Detection and Orientation Estimation: We use the network described in Sec. 3.3 to score our proposals for object detection. We test our full detection pipeline on the KITTI test set. Results are reported and compared with state-of-the-art monocular methods in Table 1 and Table 2. Our approach significantly outperforms all published monocular methods. In terms of AP, we outperform the second best method Faster R-CNN [39] by a significant margin of 7.84%, 2.26%, and 2.97% for *Car*, *Pedestrian*, and *Cyclist*, respectively, in the hard regime. For orientation estimation, we achieve 12.73% AOS improvement over 3DVP [47] on *Car* in the hard regime.

Comparison with Baselines: As strong baselines, we also use our CNN scoring on top of three other proposals methods, 3DOP [10], EdgeBoxes (EB) [55], and Selective Search (SS) [45], where we re-train the network accordingly. Table 3 shows detection and orientation estimation results on KITTI validation. We can see that our approach outperforms Edge Boxes and Selective Search by around 20% in terms of AP and AOS, while being competitive with the best method, 3DOP. Note that this comparison is not fair as 3DOP uses stereo imagery, while we employ a single monocular image. Nevertheless it is interesting to see that we achieve similar performance. We also report AP as a function of the number of proposals for *Car* in the moderate setting, in Fig. 3. When using only 10 proposals per image, our approach already achieves AP of 53.7%, while 3DOP is 35.7%. With more than 100 proposals, our AP is almost the same as 3DOP. EdgeBoxes reaches its best performance (78.7%) with 5000 proposals, while we need only 200 proposals to achieve AP of 80.6%.

Qualitative Results: Examples of our 3D detection results are in Fig. 7. Notably, our approach produces highly accurate detections in 2D and 3D even for very small or occluded objects.

5. Conclusions

We have proposed an approach to monocular 3D object detection, which generates a set of candidate class-specific object proposals that are then run through a standard CNN pipeline to obtain high-quality object detections. Towards this goal, we have proposed an energy minimization approach that places object candidates in 3D using the fact that objects should be on the ground-plane, and then scores each candidate box via several intuitive potentials encoding semantic segmentation, contextual information, size and location priors and typical object shape. We have shown that our object proposal generation approach significantly outperforms all monocular approaches, and achieves the best detection performance on the challenging KITTI benchmark.

Acknowledgements. The work was partially supported by NSFC 61171113, NSERC and Toyota Motor Corporation.

References

[1] B. Alexe, T. Deselares, and V. Ferrari. Measuring the objectness of image windows. *PAMI*, 2012. 1, 2

[2] P. Arbelaez, J. Pont-Tusetand, J. Barron, F. Marques, and J. Malik. Multiscale combinatorial grouping. In *CVPR*. 2014. 1, 2, 5, 7

[3] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv:1511.00561*, 2015. 3

[4] D. Banica and C. Sminchisescu. Cpmc-3d-o2p: Semantic segmentation of rgb-d images using cpmc and second order pooling. In *CoRR abs/1312.7715*, 2013. 2

[5] R. Benenson, M. Mathias, T. Tuytelaars, and L. Van Gool. Seeking the strongest rigid detector. In *CVPR*, 2013. 7

[6] C. Cadena, A. Dick, and I. Reid. A fast, modular scene understanding system using context-aware object detection. In *ICRA*, 2015. 7

[7] Z. Cai, M. Saberian, and N. Vasconcelos. Learning complexity-aware cascades for deep pedestrian detection. In *ICCV*, 2015. 7

[8] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *ECCV*. 2012. 2

[9] J. Carreira and C. Sminchisescu. Cpmc: Automatic object segmentation using constrained parametric min-cuts. *PAMI*, 34(7):1312–1328, 2012. 2

[10] X. Chen, K. Kundu, Y. Zhu, A. Berneshawi, H. Ma, S. Fidler, and R. Urtasun. 3d object proposals for accurate object class detection. In *NIPS*, 2015. 1, 2, 3, 4, 5, 7, 8

[11] M. Cheng, Z. Zhang, M. Lin, and P. Torr. BING: Binarized normed gradients for objectness estimation at 300fps. In *CVPR*, 2014. 1, 2, 5

[12] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2010 (VOC2010) Results. 2

[13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 2010. 2, 7

[14] S. Fidler, S. Dickinson, and R. Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In *NIPS*, 2012. 2

[15] S. Fidler, R. Mottaghi, A. Yuille, and R. Urtasun. Bottom-up segmentation for top-down detection. In *CVPR*, 2013. 2, 4

[16] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 1, 4, 5

[17] A. Geiger, C. Wojek, and R. Urtasun. Joint 3d estimation of objects and scene layout. In *NIPS*, 2011. 7

[18] A. Ghodrati, A. Diba, M. Pedersoli, T. Tuytelaars, and L. V. Gool. Deepproposal: Hunting objects by cascading deep convolutional layers. In *arXiv:1510.04445*, 2015. 1, 2

[19] R. Girshick. Fast R-CNN. In *ICCV*, 2015. 1, 4

[20] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. *CVPR*, 2014. 1, 2, 4

[21] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *ECCV*. 2014. 2, 5

[22] J. Hosang, R. Benenson, P. Dollár, and B. Schiele. What makes for effective detection proposals? *arXiv:1502.05082*, 2015. 5

[23] J. Hosang, M. Omran, R. Benenson, and B. Schiele. Taking a deeper look at pedestrians. In *arXiv*, 2015. 7

[24] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *JLMR*, 2009. 1

[25] A. Karpathy, S. Miller, and L. Fei-Fei. Object discovery in 3d scenes via shape analysis. In *ICRA*, 2013. 2

[26] P. Kr ahenb uhl and V. Koltun. Geodesic object proposals. In *ECCV*, 2014. 2

[27] P. Kr ahenb uhl and V. Koltun. Learning to propose objects. In *CVPR*, 2015. 1, 2

[28] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 2

[29] T. Lee, S. Fidler, and S. Dickinson. A learning framework for generating region proposals with mid-level cues. In *ICCV*, 2015. 1, 2

[30] B. Li, T. Wu, and S. Zhu. Integrating context and occlusion for car detection by hierarchical and-or model. In *ECCV*, 2014. 7

[31] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3d object detection with rgbd cameras. In *ICCV*, 2013. 2

[32] C. Long, X. Wang, G. Hua, M. Yang, and Y. Lin. Accurate object detection with location relaxation and regionlets relocalization. In *ACCV*, 2014. 2, 7

[33] R. Mottaghi, X. Chen, X. Liu, S. Fidler, R. Urtasun, and A. Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014. 4

[34] E. Ohn-Bar and M. M. Trivedi. Learning to detect vehicles by clustering appearance patterns. *IEEE Transactions on Intelligent Transportation Systems*, 2015. 2, 7

[35] D. Oneata, J. Revaud, J. Verbeek, and C. Schmid. Spatio-temporal object detection proposals. In *ECCV*, 2014. 2

[36] S. Paisitkriangkrai, C. Shen, and A. van den Hengel. Pedestrian detection with spatially pooled features and structured ensemble learning. In *arXiv:1409.5209*, 2014. 7

[37] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Occlusion patterns for object class detection. In *CVPR*, 2013. 7

[38] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Multi-view and 3d deformable part models. *PAMI*, 2015. 2, 7

[39] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 7, 8

[40] A. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box in the box: Joint 3d layout and object reasoning from single images. In *ICCV*, 2013. 4

[41] A. G. Schwing and R. Urtasun. Fully connected deep structured networks. 2015. 3

[42] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *arXiv:1409.1556*, 2014. 1, 2, 4

[43] Y. Tian, P. Luo, X. Wang, and X. Tang. Deep learning strong parts for pedestrian detection. In *ICCV*, 2015. 7

[44] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support Vector Learning for Interdependent and Structured Output Spaces. In *ICML*, 2004. 4

[45] K. Van de Sande, J. Uijlings, T. Gevers, and A. Smeulders. Segmentation as selective search for object recognition. In *ICCV*, 2011. 1, 2, 5, 7, 8

[46] S. Wang, S. Fidler, and R. Urtasun. Holistic 3d scene understanding from a single geo-tagged image. In *CVPR*, 2015. 2

[47] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Data-driven 3d voxel patterns for object category recognition. In *CVPR*, 2015. 2, 7, 8

[48] J. Xu, S. Ramos, D. Vozquez, and A. Lopez. Hierarchical Adaptive Structural SVM for Domain Adaptation. In *arXiv:1408.5400*, 2014. 7

[49] S. Zhang, R. Benenson, and B. Schiele. Filtered channel features for pedestrian detection. In *arXiv:1501.05759*, 2015. 7

[50] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation with deep densely connected mrfs. In *CVPR*, 2016. 4

[51] Z. Zhang, A. G. Schwing, S. Fidler, and R. Urtasun. Monocular Object Instance Segmentation and Depth Ordering with CNNs. In *ICCV*, 2015. 4

[52] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. S. Torr. Conditional random fields as recurrent neural networks. In *ICCV*, 2015. 3

[53] Y. Zhu, R. Urtasun, R. Salakhutdinov, and S. Fidler. SegDeepM: Exploiting segmentation and context in deep neural networks for object detection. In *CVPR*, 2015. 2, 4

[54] M. Zia, M. Stark, and K. Schindler. Towards scene understanding with detailed 3d object representations. *IJCV*, 2015. 2

[55] L. Zitnick and P. Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*. 2014. 1, 2, 5, 7, 8