3D Object Proposals using Stereo Imagery for Accurate Object Class Detection

Xiaozhi Chen*, Kaustav Kundu*, Yukun Zhu, Huimin Ma, Sanja Fidler and Raquel Urtasun

Abstract—The goal of this paper is to perform 3D object detection in the context of autonomous driving. Our method aims at generating a set of high-quality 3D object proposals by exploiting stereo imagery. We formulate the problem as minimizing an energy function that encodes object size priors, placement of objects on the ground plane as well as several depth informed features that reason about free space, point cloud densities and distance to the ground. We then exploit a CNN on top of these proposals to perform object detection. In particular, we employ a convolutional neural net (CNN) that exploits context and depth information to jointly regress to 3D bounding box coordinates and object pose. Our experiments show significant performance gains over existing RGB and RGB-D object proposal methods on the challenging KITTI benchmark. When combined with the CNN, our approach outperforms all existing results in object detection and orientation estimation tasks for all three KITTI object classes. Furthermore, we experiment also with the setting where LIDAR information is available, and show that using both LIDAR and stereo leads to the best result.

Index Terms—object proposals, 3D object detection, convolutional neural networks, autonomous driving, stereo, LIDAR.

1 Introduction

Autonomous driving is receiving a lot of attention from both industry and the research community. Most self-driving cars build their perception systems on expensive sensors, such as LIDAR, radar and high-precision GPS. Cameras are an appealing alternative as they provide richer sensing at a much lower cost. This paper aims at high-performance 2D and 3D object detection in the context of autonomous driving by exploiting stereo imagery.

With impressive advances in deep learning in the past few years, recent efforts in object detection exploit object proposals to facilitate classifiers with powerful, hierarchical visual representation [1], [2]. Compared with traditional sliding window based methods [3], the pipeline of generating object proposals that are combined with convolutional neural networks has lead to more than 20% absolute performance gains [4], [5] on the PASCAL VOC dataset [6].

Object proposal methods aim at generating a moderate number of candidate regions that cover most of the ground truth objects in the image. One typical approach is to perform region grouping based on superpixels using a variety of similarity measures [7], [8]. Low-level cues such as color contrast, saliency [9], gradient [10] and contour information [11] have also been exploited in order to select promising object boxes from densely sampled windows. There has also been some recent work on learning to generate a diverse set of region candidates with ensembles of binary segmentation models [12], parametric energies [13] or CNN-based cascaded classifiers [14].

The object proposal methods have proven effective on the PASCAL VOC benchmark. However, they have very low achievable recall on the autonomous driving benchmark KITTI [15], which presents the bottleneck for the state-of-the-art object detector R-CNN [4], [16] on this benchmark. On one hand, the PASCAL VOC dataset uses a loose overlap criteria for localization measure, i.e., a predicted box is considered to be correct if its overlap with the ground-truth box exceeds 50%. For self-driving cars, however, object detection requires a stricter overlap criteria to enable correct estimates of the distance of vehicles from the ego-car. Moreover, objects in KITTI images are typically small and many of them are heavily occluded or truncated. These challenging conditions limit the performance of most existing bottom-up proposals that rely on intensity and texture for superpixel merging and window scoring.

In this paper, we propose a novel 3D object detection approach that exploits stereo imagery and contextual information specific to the domain of autonomous driving. We propose a 3D object proposal method that goes beyond 2D bounding boxes and is capable of generating high-quality 3D bounding box proposals. We make use of the 3D information estimated from a stereo camera pair by placing 3D candidate boxes on the ground plane and scoring them via 3D point cloud features. In particular, our scoring function encodes several depth informed features such as point densities inside a candidate box, free space, visibility, as well as object size priors and height above the ground plane. The inference process is very efficient as all the features can be computed in constant time via 3D integral images. Learning can be done using structured SVM [17] to obtain class-specific weights for these features. We also present a 3D object detection neural network that takes 3D object proposals as input and predict accurate 3D bounding boxes. The neural net exploits contextual information and uses a multi-task loss to jointly regress to bounding box coordinates and object orientation.

We evaluate our approach on the challenging KITTI detection benchmark [15]. Extensive experiments show that: 1) The proposed 3D object proposals achieve significantly

- * *Denotes equal contribution.*
- *X. Chen and H. Ma are with the Department of Electronic Engineering, Tsinghua University, China.*
- *K. Kundu, Y. Zhu, S. Fidler and R. Urtasun are with the Department of Computer Science, University of Toronto, Canada.*

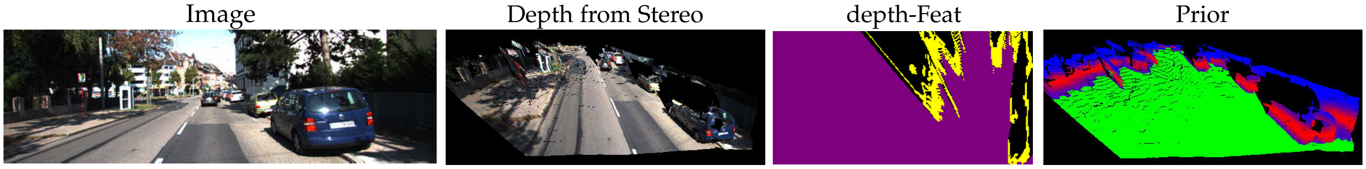


Fig. 1: **Features in our model** (from left to right): left camera image, stereo 3D reconstruction, depth-based features and our prior. In the third image, occupancy is marked with yellow (P in Eq. (1)) and purple denotes free space (F in Eq. (2)). In the prior, the ground plane is green and blue to red indicates increasing prior value of object height.

higher recall than the state-of-the-art across all overlap thresholds under various occlusion and truncation levels. In particular, compared with the state-of-the-art RGB-D method MCG-D [18], we obtain 25% higher recall with 2K proposals. 2) Our 3D object detection network combined with 3D object proposals outperforms all published results on object detection and orientation estimation for *Car*, *Cyclist* and *Pedestrian*. 3) Our approach is capable of producing accurate 3D bounding box detections, which allows us to locate objects in 3D and infer the distance and pose of objects from the ego-car. 4) We also apply our approach to LIDAR point clouds with more precise, but sparser, depth estimation. When combining stereo and LIDAR data, we obtain the highest 3D object detection accuracy.

A preliminary version of this work was presented in [19]. In this manuscript, we make extensions in the following aspects: 1) A more detailed description of the inference process of proposal generation. 2) The 3D object proposal model is extended with a class-independent variant. 3) The detection neural network is extended to a two-stream network to leverage both appearance and depth features. 4) We further apply our model to point clouds obtained via LIDAR, and provide comparison of the stereo, LIDAR and the hybrid settings. 5) We extensively evaluate the 3D bounding box recall and 3D object detection performance. 6) Our manuscript includes ablation studies of network design, depth features, as well as ground plane estimation.

2 Related Work

Our work is closely related to object proposal generation and 3D object detection. We briefly review the literature with a focus on the domain of autonomous driving.

Object Proposal Generation. Object proposal generation has become an important technique in object detection. It speeds up region searching and enables object detectors to leverage the great power of deep neural networks [1], [2]. Numerous works have been proposed for different modalities, i.e., RGB [7], [8], [10], [11], [13], [20], RGB-D [18], [21], [22], [23], and video [24], [25].

In RGB, one typical paradigm is to generate candidate segments by grouping superpixels or multiple figure-ground segmentations with diverse seeds. Grouping-based methods [7], [8], [26] build on multiple oversegmentations and merge superpixels based on complementary cues such as color, texture and shape. Geodesic proposals [27] learn to place diverse seeds and identify promising regions by computing geodesic distance transforms. CPMC [20] solves a sequence of binary parametric min-cut problems with different seeds and unary terms. The resulting regions are then ranked using Gestalt-like features and diversified using maximum marginal relevance measures. This approach is widely used in recognition tasks [5], [28], [29]. Some recent approaches also follow this pipeline by learning an ensemble of local and global CRFs [12] or minimizing parametric energies that encode mid-level cues such as symmetry and closure [13]. Another paradigm generates bounding box proposals by scoring exhaustively sampled windows. In [9], a large pool of windows are scored with a diverse set of features such as color contrast, edges, location and size. BING [10] scores windows using simple gradient features which serve as an object closure measure and can be computed extremely fast. BING++ [30] further improves its localization quality using edge and superpixel based box refinement [31]. EdgeBoxes [11] design an effective scoring function by computing the number of contours that exist in or straddle the bounding box. [14] computes integral image features from inverse cascading layers of CNN for candidate box scoring and refinement. A detailed comparison of existing proposal methods has been carried out in [32]. While most of these approaches achieve more than 90% recall with 2K proposals on the PASCAL VOC benchmark [6], they have significant lower recall on the KITTI dataset.

In RGB-D, [21], [22] extend CPMC [20] with depth cues and fit 3D cubes around candidate regions to generate cuboid object proposals. [18] extends MCG [8] with RGB-D contours as well as depth features to generate 2.5D proposals. They obtain significantly better performance compared with purely RGB approaches. In [23], candidate objects are proposed from 3D meshes by oversegmentation and several intrinsic shape measures. Our work is also relevant to Sliding Shapes [33], which densely evaluates 3D windows with exemplar-SVM classifiers in 3D point clouds. However, they train exemplar classifiers on CAD models with hundreds of rendered views and complex shape features, resulting in very inefficient training and inference. In our work, we advance over past work by exploiting the physical sizes of objects, the ground plane, as well as depth features and contextual information in 3D.

3D Object Detection. In the domain of autonomous driving, accurate 3D localization and pose estimation of objects beyond 2D boxes are desired. In [34], the Deformable Part-based Model [3] is extended to 3D by adding viewpoint information and 3D part geometry. The potentials are parameterized in 3D object coordinates instead of the image plane. Zia et al. [35] initialize a set of candidate objects using a variant of poselets detectors and model part-level occlusion and configuration with 3D deformable wireframes. [36] trains an ensemble of subcategory models by clustering object instances with appearance and geometry features. In [37], a top-down bounding box re-localization scheme is proposed to refine Selective Search proposals with Regionlets features. [38] combines cartographic map

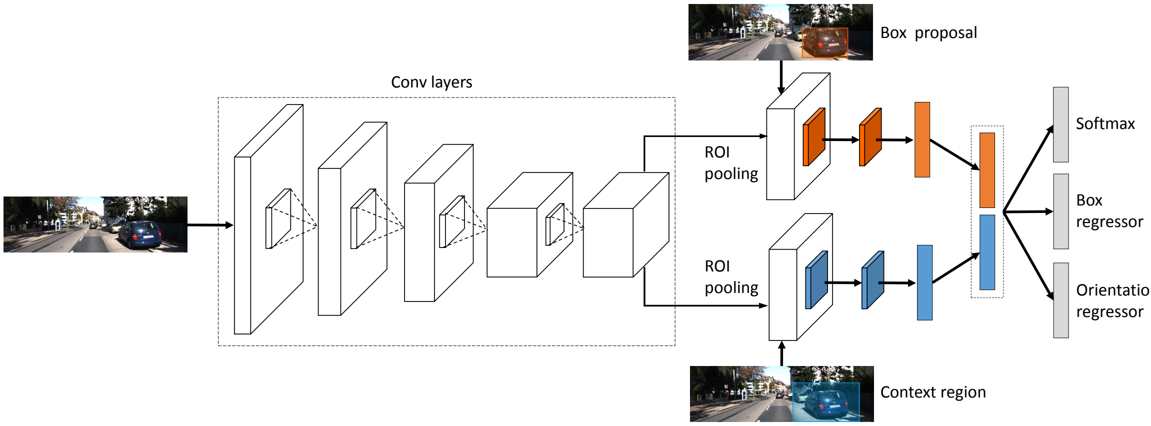


Fig. 2: **The single-stream network for 3D object detection:** Input can be an RGB image or a 6-channel RGB-HHA image.

priors and DPM detectors into a holistic model to re-reason object locations. [39] uses And-Or models to learn car-to-car context and occlusion patterns. [40] learns AdaBoost classifier with dense local features within subcategories. The recently proposed 3DVP [41] employs ACF detectors [42] and learns occlusion patterns with 3D voxels.

With the shift of low-level features to multi-layer visual representation, most of recent approaches exploit CNNs for object detection also in the context of autonomous driving. In [43], R-CNN is applied on pedestrian detection with proposals generated by SquaresChnFtrs detector, achieving moderate performance. [44] learns part detectors with convolutional features to handle occlusion in pedestrian detection. [45] designs a complexity-aware cascade pedestrian detector with convolutional features. Parallel to our work, Faster R-CNN [46] improves upon their prior R-CNN [4] pipeline by integrating proposal generation and R-CNN into an end-to-end trainable network. However, these methods only produce 2D detections, whereas our work aims at 3D object detection in order to infer both, accurate object pose as well as the distance from the ego-car.

3 3D Object Proposals

Our approach aims at generating a diverse set of 3D object proposals in the context of autonomous driving. 3D reasoning is crucial in this domain as it eases problems such as occlusion and large scale variation. The input to our method is a stereo image pair. We compute depth using the method by Yamaguchi et al. [47], yielding a point cloud $\mathbf{x}$. We place object proposals in 3D space in the form of 3D bounding boxes. Note that only depth information (no appearance) is used in our proposal generation process. Next we describe our parameterization and the framework.

3.1 Proposal Generation as Energy Minimization

We use a 3D bounding box to represent each object proposal $\mathbf{y}$, which is parametrized by a tuple, (x, y, z, θ, c, t), where (x, y, z) is the 3D box center and θ denotes the azimuth angle. Here, $c \in C$ is the object class and $t \in \{1, \ldots, T_c\}$ indexes a set of 3D box templates, which are learnt from training data to represent the typical physical size of each class c (details in Sec. 3.3.1). We discretize the 3D space into voxels for candidate box sampling and thus each box $\mathbf{y}$ is represented in discretized form (details in Sec. 3.2).

We generate proposals by minimizing an energy function which encodes several depth-informed potentials. We encode the fact that the object should live in a space occupied with high density by the point cloud. Furthermore, the box $\mathbf{y}$ should have minimal overlap with the free space in the scene. We also encode the height prior of objects, and the fact that the point cloud in the box's immediate vicinity should have lower prior values of object height than the box. The energy function is formulated as:

$$E(\mathbf{x},\mathbf{y}) = \mathbf{w}_{c,pcd}^{\top}\phi_{pcd}(\mathbf{x},\mathbf{y}) + \mathbf{w}_{c,fs}^{\top}\phi_{fs}(\mathbf{x},\mathbf{y}) + \mathbf{w}_{c,ht}^{\top}\phi_{ht}(\mathbf{x},\mathbf{y}) + \mathbf{w}_{c,ht-contr}^{\top}\phi_{ht-contr}(\mathbf{x},\mathbf{y}).$$

The weights of the energy terms are learnt via structured SVM [48] (details in Sec. 3.3). Note that the above formulation encodes dependency of weights on the object class, thus weights are learnt specific to each class. However, we can also learn a single set of weights for all classes (details in Sec. 3.3.3). We next explain each potential in more detail.

Point Cloud Density: This potential encodes the point cloud density within the box:

$$\phi_{pcd}(\mathbf{x},\mathbf{y}) = \frac{\sum_{v \in \Omega(\mathbf{y})} P(v)}{|\Omega(\mathbf{y})|} \tag{1}$$

where $P(v) \in \{0, 1\}$ indicates whether voxel v contains point cloud points or not, and $\Omega(\mathbf{y})$ denotes the set of voxels within box $\mathbf{y}$. The feature P is visualized in Fig. 1. This potential is simply computed as the fraction of occupied voxels within the box. By using integral accumulators (integral images in 3D), the potential can be computed efficiently in constant time.

Free Space: Free space is defined as the space that lies on the rays between the point cloud and the camera. This potential encodes the fact that the box should not contain a significant amount of free space (since it is occupied by the object). We define F as a binary valued grid, where $F(v) = 1$ means that the ray from the camera to voxel v is not intercepted by any occupied voxel, i.e., voxel v belongs to the free space. The potential is defined as follows:

$$\phi_{fs}(\mathbf{x},\mathbf{y}) = \frac{\sum_{v \in \Omega(\mathbf{y})} (1 - F(v))}{|\Omega(\mathbf{y})|} \tag{2}$$

It encourages less free space within the box, and can be efficiently computed using integral accumulators.

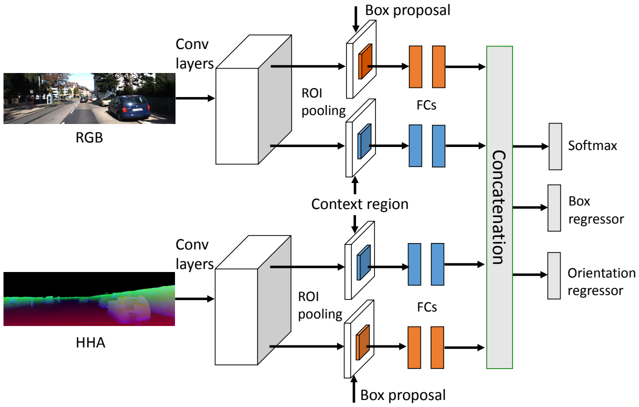


Fig. 3: **Two-stream network for 3D object detection:** The convnet learns from RGB (top) and HHA [18] (bottom) images as input, and concatenates features from *fc7* layers for multi-task prediction. The model is trained end-to-end.

Height Prior: This potential encourages the height of the point cloud within the box w.r.t. the road plane to be close to the mean height $\mu_{c,ht}$ of the object class c. We encode it as follows:

$$\phi_{ht}(\mathbf{x}, \mathbf{y}) = \frac{1}{|\Omega(\mathbf{y})|} \sum_{v \in \Omega(\mathbf{y})} H_c(v) \tag{3}$$

with

$$H_c(v) = \begin{cases} \exp\left[-\frac{1}{2}\left(\frac{d_v - \mu_{c,ht}}{\sigma_{c,ht}}\right)^2\right], & \text{if } P(v) = 1 \\ 0, & \text{o.w.} \end{cases} \tag{4}$$

Here, d_v is the distance between the center of the voxel v and the road plane, along the direction of the gravity vector. By assuming a Gaussian distribution of the data, we compute $\mu_{c,ht}, \sigma_{c,ht}$ as the MLE estimates of mean height and standard deviation. The feature is shown in Fig. 1. It can be efficiently computed via integral accumulators.

Height Contrast: This potential encodes the fact that the point cloud surrounding the box should have lower values of the height prior relative to the box. We first compute a surrounding region $\mathbf{y}^+$ of box $\mathbf{y}$ by extending $\mathbf{y}$ by 0.6m[1] in the direction of each face. We formulate the contrast of height priors between box $\mathbf{y}$ and surrounding box $\mathbf{y}^+$ as:

$$\phi_{ht-contr}(\mathbf{x}, \mathbf{y}) = \frac{\phi_{ht}(\mathbf{x}, \mathbf{y})}{\phi_{ht}(\mathbf{x}, \mathbf{y}^+) - \phi_{ht}(\mathbf{x}, \mathbf{y})} \tag{5}$$

3.2 Inference

We compute the point cloud $\mathbf{x}$ from a stereo image pair using the approach by Yamaguchi et al. [47]. Then we discretize the 3D space and estimate the road plane for 3D candidate box sampling. We perform exhaustive scoring of each candidate using our energy function, and use non-maximal suppression (NMS) to obtain top K diverse 3D proposals. In particular, we use a greedy algorithm, where at each iteration we select the next proposal that has the lowest energy and its IoU overlap with the previously selected proposals does not exceed a threshold δ. Specifically, the m^{th} proposal $\mathbf{y}^m$ is obtained by solving the following problem:

$$\begin{aligned} \mathbf{y}^m &= \operatorname*{argmin}_{\mathbf{y} \in \mathcal{Y}} E(\mathbf{x}, \mathbf{y}) \\ \text{s.t.} \quad & \text{IoU}(\mathbf{y}, \mathbf{y}^i) < \delta, \quad \forall i \in \{0, \ldots, m-1\}, \end{aligned} \tag{6}$$

Discretization and Accumulators: The point cloud is defined in a left-handed coordinate system, where the Y-axis goes in the direction of gravity and the positive Z-axis is along the camera's viewing direction. We discretize the 3D continuous space such that the each voxel has length of 0.2m in each dimension. We compute the point cloud occupancy, free space and height prior grids in this voxel space, as well as their 3D integral accumulators.

Ground Plane Estimation: We estimate the ground plane by classifying superpixels [47] using a very small neural network, and fitting a plane to the estimated ground pixels using RANSAC. We use the following features on the superpixels as input to the network: mean RGB values, average 2D and 3D position, pitch and roll angles relative to the camera of the plane fit to the superpixel, a flag as to whether the average 2D position was above the horizon line, and standard deviation of both the color values and 3D position. This results in a 22-dimensional feature vector. The neural network consists of only a single hidden layer which also has 22 units. We use *tanh* as the activation function and cross-entropy as the loss function. We train the network on the KITTI's road benchmark [15].

Bounding Boxes Sampling and Scoring: For 3D candidate box sampling, we use three size templates per class and two orientations $\theta \in \{0, 90\}$. As all the features can be efficiently computed via integral accumulators, it takes constant time to evaluate each configuration $\mathbf{y}$. Despite that, evaluating exhaustively in the entire space would be slow. We reduce the search space by skipping empty boxes that do not contain any points. With ground plane estimation, we further reduce the search space along the vertical dimension by only placing candidate boxes on the ground plane.

However, to alleviate the noise of stereo depth at large distances, we sample additional candidate boxes at distances larger than 20m from the camera. In particular, let y_{road} denote the height of the ground plane. We deviate this height along the vertical dimension to compute two additional planes that have heights $y = y_{road} \pm \sigma_{road}$. Here σ_{road} denotes the MLE estimate of the standard deviation of a Gaussian distribution modeling the distance of objects from the ground plane. We then sample additional boxes on these planes. With our sampling strategy, scoring all configurations can be done in a fraction of a second.

Note that the energy function is computed independently with respect to each candidate box. We rank all boxes according to the values of $E(\mathbf{x}, \mathbf{y})$, and perform greedy inference with non-maxima suppression (NMS). In practice, we perform NMS in 2D as it achieves similar recall as NMS in 3D while being much faster. The IoU threshold δ is set to 0.75. The entire feature computation and inference process takes 1.2s per image on average for 2K proposals.

[1] The value as well as other hyper parameters (e.g., voxel size) are determined based on the performance on the validation set.

3.3 Learning

We next explain how we obtain the 3D bounding box templates, and how we learn the weights in our model.

3.3.1 3D Bounding Box Templates

The size templates are obtained by clustering the ground truth 3D bounding boxes on the training set. In particular, we first compute a histogram for the object sizes, and choose a cluster of boxes that have IoU overlaps with the mode of the histogram above 0.6, then remove those boxes and iterate. The representative size templates are computed by averaging the box sizes in each cluster.

3.3.2 Learning the Weights in Our Model

We use structured SVM [48] to learn the model's weights $\{w_{c,pcd}, w_{c,fs}, w_{c,ht}, w_{c,ht-contr}\}$. Given N input-output training pairs, $\{\mathbf{x}^{(i)}, \mathbf{y}^{(i)}\}_{i=1,\cdots,N}$, we obtain the parameters by solving the following optimization problem:

$$
\begin{aligned}
&\min_{\mathbf{w}\in\mathbb{R}^D} \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{N}\sum_{i=1}^{N}\xi_i \\
\text{s.t.:}\quad & \mathbf{w}^T(\phi(\mathbf{x}^{(i)}, \mathbf{y}) - \phi(\mathbf{x}^{(i)}, \mathbf{y}^{(i)})) \\
& \geq \Delta(\mathbf{y}^{(i)}, \mathbf{y}) - \xi_i, \forall \mathbf{y} \setminus \mathbf{y}^{(i)}
\end{aligned}
\tag{7}
$$

We use the parallel cutting plane implementation of [49] to solve this minimization problem. As the task loss $\Delta(\mathbf{y}^{(i)}, \mathbf{y})$, we use the strict 3D Intersection-over-Union (IoU) to encourage accurate placement of the 3D proposals. In particular, 3D IoU is computed as the volume of intersection of two 3D bounding boxes divided by the volume of their union.

3.3.3 Class-Independent 3D Proposals

The method described above learns separate weights for each category, thus generating class-dependent object proposals. However, the approach can be easily modified to generate class-independent proposals by learning only a single scoring model for all categories. In particular, we learn object templates for all classes jointly rather than for each specific class. Therefore, the weights in this energy are class-independent (we have only a single set of weights). We compare these two approaches in the experiments.

4 3D Object Detection Networks

In this section, we describe how we score the top-ranked 3D object proposals via convolutional networks. We design a network architecture for two tasks: joint 2D object detection and orientation estimation, and 3D object detection.

4.1 Joint 2D Object Detection and Pose Estimation

The architecture of our network for joint 2D object detection and orientation estimation is shown in Fig. 2. The network is built upon Fast R-CNN [16], which share the convolutional features across all proposals and use a ROI pooling layer to compute proposal-specific features. We extend this basic network by adding a context branch after the last convolutional layer (i.e., *conv5*), and an orientation regression loss to jointly learn object location and orientation. Specifically, the first branch encodes features from the original candidate regions while the second branch is specific to context regions, which are computed by enlarging the candidate boxes by a factor of 1.5, following the segDeepM approach [5]. Both branches consist of a ROI pooling layer and two fully connected layers. ROIs are obtained by projecting the 3D proposals onto the image plane and then onto the *conv5* feature maps. We concatenate the features from fc_7 layers and feed them to the prediction layers.

We predict the class labels, bounding box coordinate offsets, and object orientation jointly using a multi-task loss. We define the category loss as cross entropy, the orientation loss and bounding box offset loss as a smooth ℓ_1 loss. We parameterize the bounding box coordinates as in [4]. Each loss is weighted equally and only the category label loss is employed for the background boxes.

4.2 3D Object Detection

For 3D object detection, we want to output full 3D bounding boxes for objects. We use the same network as in Fig. 2, except that 2D bounding box regressors are replaced by 3D bounding box regressors. Similarly to 2D box regression, we parametrize the centers of 3D boxes with size normalization for scale-invariant translation, and the 3D box sizes with log-space shift. In particular, we denote a 3D box proposal as $P = (P_x, P_y, P_z, P^s_x, P^s_y, P^s_z)$, and its corresponding ground truth 3D box as $G = (G_x, G_y, G_z, G^s_x, G^s_y, G^s_z)$, which specify the box center and the box size in each dimension. The regression targets for the box center $T_c(P)$ and the box size $T^s_c(P)$ are parametrized as follows:

$$
T_c(P) = \frac{G_c - P_c}{P^s_c}, \quad T^s_c(P) = log\frac{G^s_c}{P^s_c}, \quad \forall c \in \{x, y, z\} \tag{8}
$$

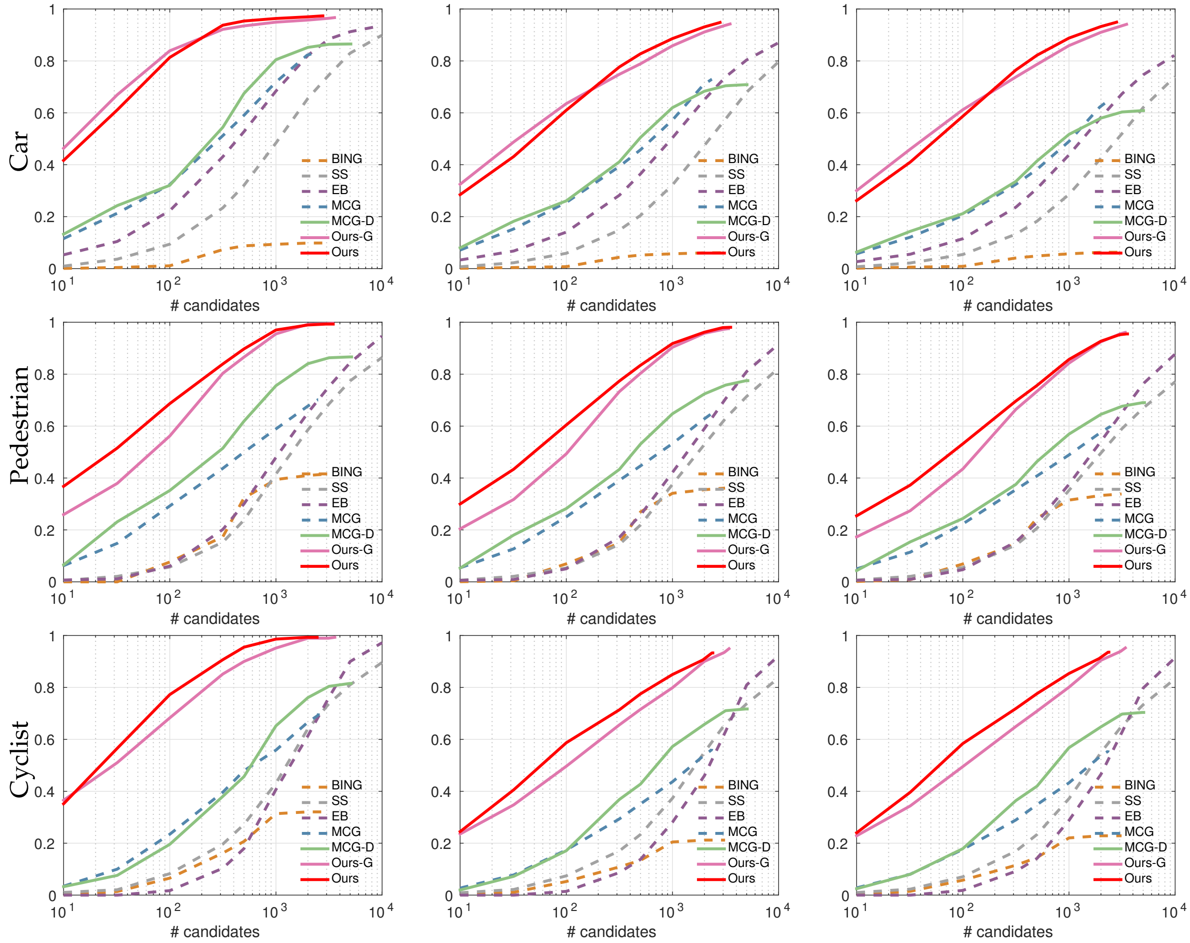
Given the 3D box coordinates and the estimated orientation, we then compute the azimuth angle θ of the box.

4.3 CNN Scoring with Depth Features

Despite only using appearance features (i.e. RGB image), the basic network described above already performs very well in practice. To take advantage of depth information in CNN scoring process, we further compute a depth image encoded with HHA features [18]. HHA has three channels which represent the disparity map, height above the ground, and the angle of the normal at each pixel with respect to the gravity direction. We explore two approaches to learn feature representation with both RGB and depth images as input. The first approach is a single-stream network, which directly combines RGB channels and HHA channels to form a 6-channel image, and feed it to the network. This architecture is exactly the same as the basic model in Fig. 2, except that its input is a 6-channel image. The second approach is a two-stream network which learns features from RGB and HHA images respectively, as shown in Fig. 3. Note that the two-stream network has almost double the parameters of the single-stream model, and thus requires more GPU memory in training.

4.4 Implementation Details

In our object detection experiments, we use class-specific weights for proposal generation. For network training, we choose training samples based on the IoU overlap threshold of the 2D bounding box proposals and ground truth boxes.



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

Fig. 4: **2D bounding box Recall vs number of Candidates**. **"Ours-G"**: class-independent proposals. **"Ours"**: class-dependent proposals. We use an overlap threshold of 0.7 for *Car*, and 0.5 for *Pedestrian* and *Cyclist*, following the KITTI evaluation protocol [15]. From left to right are for *Easy*, *Moderate*, and *Hard* evaluation regimes, respectively.

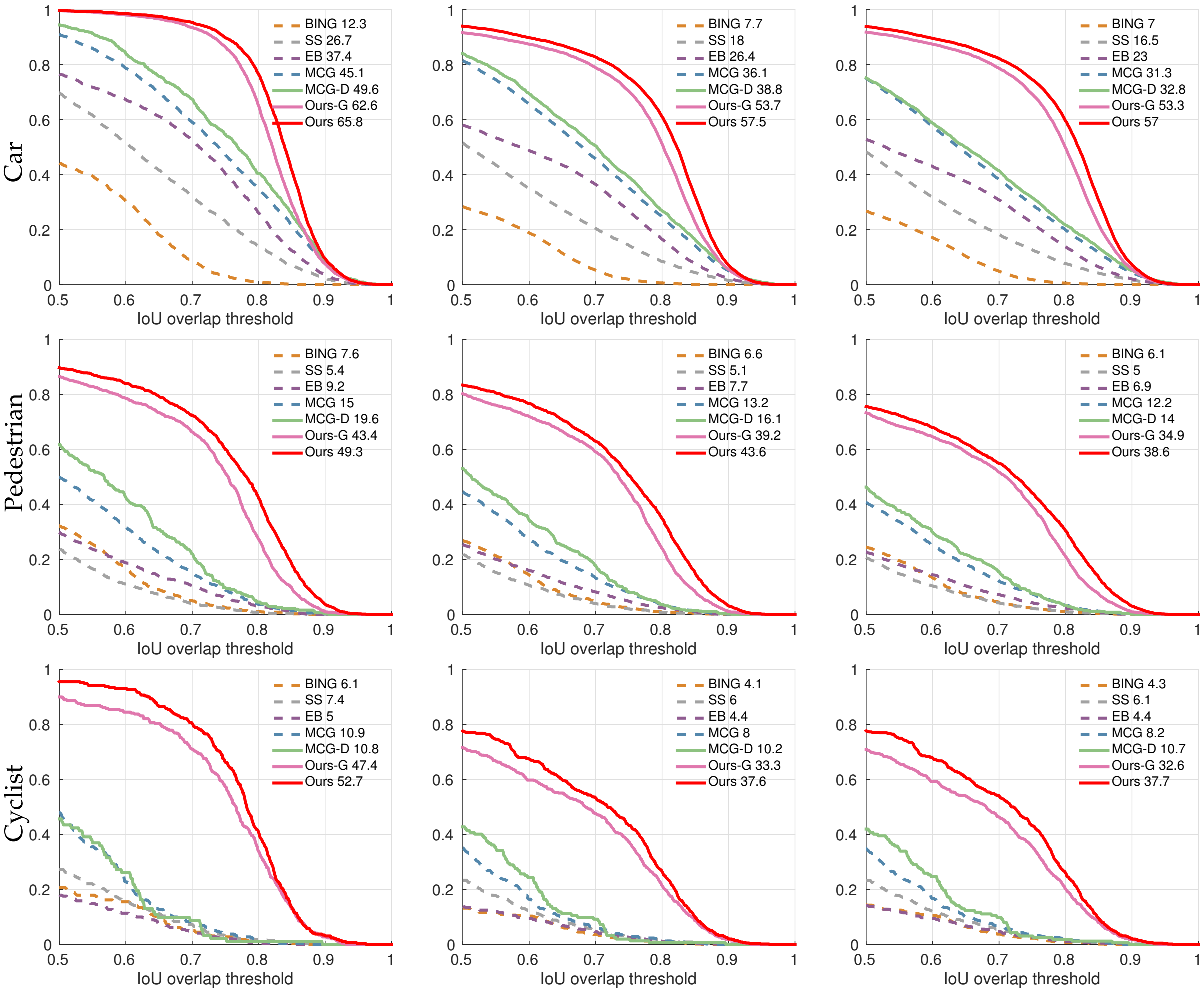
Since KITTI uses different overlap criteria for *Car* and *Pedestrian*/*Cyclist*, we set the threshold for *Car* to 0.7 and 0.5 for the *Pedestrian*/*Cyclist* classes. By default, we use the VGG-16 network [2] trained on ImageNet to initialize our networks. We initialize the context branch by copying the weights of the fully-connected layers from the pre-trained model. For the two-stream RGB-HHA model, which requires more GPU memory, we use the 7-layer VGG_CNN_M_1024 network [50]. The weights for the HHA channels/branch are copied from the corresponding RGB channels/branch. For the one-stream model, we fine-tune all the layers starting from *conv1*. For the two-stream model, all layers are fine-tuned for the HHA branch, and only layers above *conv2* are fine-tuned for the RGB branch. We use bilinear interpolation to upscale the input image by a factor of 3.5, which is crucial to achieve very good performance since objects in KITTI imagery are typically small. We use a single scale for input images in both training and testing. We run SGD and set the initial learning rate to 0.001. After 30K iterations we reduce it to 0.0001 and run another 10K iterations. Training proposals are sampled in a image-centric manner with a batch size of 1 for images and 128 for proposals. At test time, the network takes around 2s to evaluate one image with 2K proposals on a Titan X GPU.

5 Experimental Evaluation

We evaluate our approach on the challenging KITTI detection benchmark [15], which has 7,481 training and 7,518 test images. The benchmark contains three object classes: *Car*, *Pedestrian*, and *Cyclist*. Evaluation is done for each class in three regimes: *Easy*, *Moderate* and *Hard*, which contain objects of different occlusion and truncation levels. We split the 7,481 training images into a *training* set (3,712 images) and a *validation* set (3,769 images). We ensure that the training and validation set do not contain images from the same video sequences, and evaluate the performance of our proposals on the validation set.

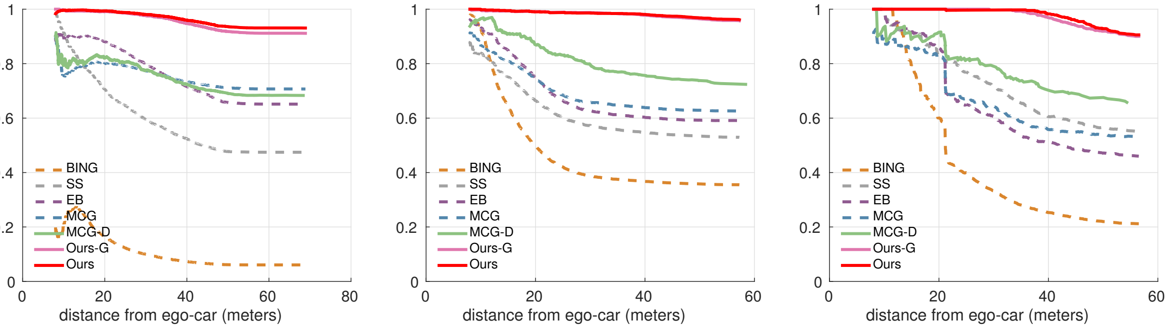
Metrics: To evaluate proposals, we use the oracle recall as the metric, following [7], [32]. A ground truth object is said to be recalled if at least one proposal overlaps with it with IoU above a certain threshold. We set the IoU threshold to 70% for *Car*, and 50% for *Pedestrian* and *Cyclist*, following the standard KITTI's setup. The oracle recall is then computed as the percentage of recalled ground truth objects. We also report average recall (AR) [32], which has been shown to be highly correlated with the object detection performance.

We also evaluate the whole pipeline of our 3D object detection model on KITTI's two tasks: 2D object detection, and joint 2D object detection and orientation estimation.



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

Fig. 5: **2D bounding box Recall vs IoU for 500 proposals**. The number next to the label indicates the average recall (AR).



Car Pedestrian Cyclist

Fig. 6: **2D bounding box Recall vs Distance with 2000 proposals on *moderate* data**. We use overlap threshold of 0.7 for *Car*, and 0.5 for *Pedestrian*, *Cyclist*.

Following the standard KITTI setup, we use the Average Precision (AP_{2D}) metric for 2D object detection task, and Average Orientation Similarity (AOS) [15] for joint 2D object detection and orientation estimation task.

For the task of 3D object detection, we evaluate the performance using two metrics: Average Precision (AP_{3D}) using 3D bounding box overlap measure, and Average Localization Precision (ALP). Similar to the setting in [33], we use 25% overlap criteria for the 3D bounding box overlap measure. Average Localization Precision is computed similarly to AP, except that the bounding box overlap is replaced by 3D localization precision. We consider a predicted 3D location to be correct if its distance to the ground truth 3D location is smaller than certain threshold. Note that this 3D localization precision measure is used when computing both precision and recall.

Baselines: We compare our proposal method with several top-performing approaches on the validation set: MCG-D [18], MCG [8], Selective Search (SS) [7], BING [10], and EdgeBoxes (EB) [11].

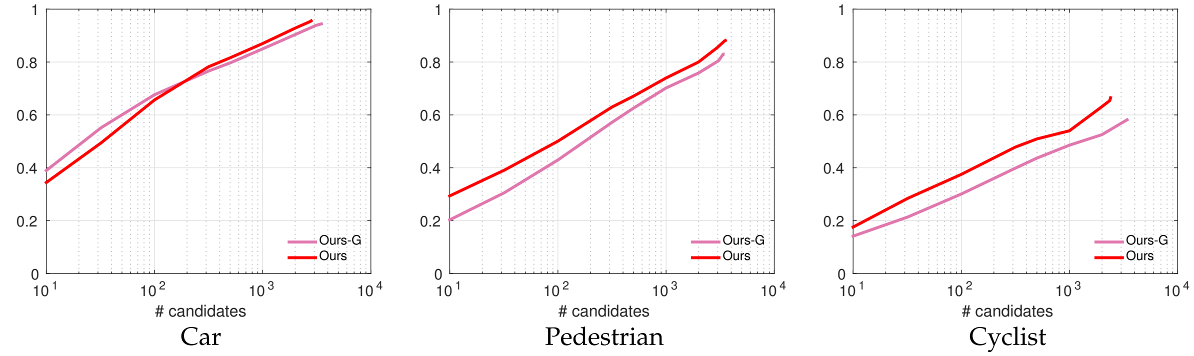


Fig. 7: **3D bounding box Recall vs #Candidates on *moderate* data**. 3D IoU threshold is set to 0.25.

TABLE 1: Running time of different proposal methods.

Method	Time (sec.)
BING [10]	0.01
Selective Search (SS) [7]	15
EdgeBoxes (EB) [11]	1.5
MCG [8]	100
MCG-D [18]	160
Ours	1.2

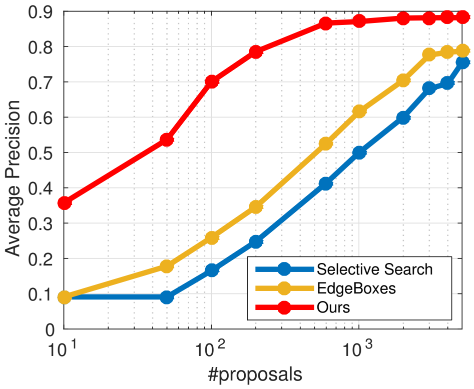


Fig. 8: AP_{2D} vs #proposals on *Car* for the *Moderate* setting.

5.1 Proposal Recall

We evaluate recall of the two variants of our approach: class-dependent and class-independent proposals. We denote the class-independent variant as 'Ours-G'.

2D Bounding Box Recall: Fig. 4 shows recall as a function of the number of candidates. We can see that in general our class-specific proposals perform slightly better than the class-independent variant. This suggests the advantage of exploiting size priors tailored to each class. By using 1000 proposals, our approach achieves almost 90% recall for *Car* in the *Moderate* and *Hard* regimes, while for *Easy* we need only 200 proposals to reach the same recall. In contrast, other methods saturate or require orders of magnitude more proposals to reach 90% recall. For *Pedestrian* and *Cyclist* our approach achieves similar improvements over the baselines. Note that while our approach uses depth-based features, MCG-D combines depth and appearance features, and all other methods use only appearance features. This suggests the importance of 3D reasoning in the domain of autonomous driving. In Fig. 5, we show recall as a function of the IoU overlap for 500 proposals. We obtain significantly higher recall over the baselines across all IoU overlap levels, particularly for *Cyclist*.

Recall vs Distance: We also plot recall as a function of the object's distance from the ego-car in Fig. 6. We can see that our approach remains at a very high recall even at large distances (>40m), while the performance of all other proposal methods drops significantly with the increasing distance. This shows the advantage of our approach in the autonomous driving scenario.

3D Bounding Box Recall: Since the unique benefit of our approach is the output of 3D bounding boxes, we also evaluate recall in 3D, i.e., using 3D bounding box overlap. We set IoU overlap threshold computed in 3D between the ground-truth 3D bounding boxes and ours to 0.25. As shown in Fig. 7, when using 2000 proposals, our approach achieves around 90%, 80% and 60% 3D recall for *Car*, *Pedestrian* and *Cyclist*, respectively.

Running Time: The running time of different proposal methods are shown in Table 1. Our approach is fairly efficient and takes 1.2s on a single core. The only faster proposal method is BING, which however achieves a much lower recall than our approach (Fig. 4).

Qualitative Results: Figs. 11 and 12 show qualitative results for cars and pedestrians in KITTI images. We show the input RGB image, top 100 proposals, the 3D ground truth boxes, as well as the best proposals that have the highest IoU with ground truth boxes (chosen from 2K proposals). We can see that our proposals are able to locate objects precisely even for the distant and occluded objects.

5.2 2D Object Detection and Orientation Estimation

Performance on KITTI Test: We test our approach on KITTI's Test for two tasks: 2D object detection, and joint 2D object detection and orientation estimation. We choose our best model based on the validation set, and submit our results to the benchmark. We choose the class specific variant of our method for generating proposals. For the detection network, we use the single-stream model with RGB image as input. As shown in Table 2 and Table 3, our approach significantly outperforms all published methods, including both image-based and LIDAR-based methods. In terms of 2D object detection, our approach outperforms Faster R-CNN [46], which is the state-of-the-art model on ImageNet, MS COCO, and PASCAL datasets. We achieve

TABLE 2: Average Precision (AP_{2D}) (in %) on the test set of the KITTI Object Detection Benchmark. †: LIDAR-based methods; ‡: using both LIDAR and image data.

Method	Cars			Pedestrians			Cyclists		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
mBoW† [51]	36.02	23.76	18.44	44.28	31.37	30.62	28.00	21.62	20.93
CSoR† [52]	34.79	26.13	22.69	-	-	-	-	-	-
Vote3D† [53]	56.80	47.99	42.57	44.48	35.74	33.72	41.43	31.24	28.60
VeloFCN† [54]	71.06	53.59	46.92	-	-	-	-	-	-
MV-RGBD-RF‡ [55]	76.40	69.92	57.47	73.30	56.59	49.63	52.97	42.6	37.42
SubCat [36]	84.14	75.46	59.71	54.67	42.34	37.95	-	-	-
DA-DPM [56]	-	-	-	56.36	45.51	41.08	-	-	-
Fusion-DPM [57]	-	-	-	59.51	46.67	42.05	-	-	-
R-CNN [43]	-	-	-	61.61	50.13	44.79	-	-	-
pAUCEnsT [58]	-	-	-	65.26	54.49	48.60	51.62	38.03	33.38
FilteredICF [59]	-	-	-	67.65	56.75	51.12	-	-	-
DeepParts [44]	-	-	-	70.49	58.67	52.78	-	-	-
CompACT-Deep [45]	-	-	-	70.69	58.74	52.71	-	-	-
3DVP [41]	87.46	75.77	65.38	-	-	-	-	-	-
AOG [39]	84.80	75.94	60.70	-	-	-	-	-	-
Regionlets [37]	84.75	76.45	59.70	73.14	61.15	55.21	70.41	58.72	51.83
spLBP [40]	87.19	77.40	60.60	-	-	-	-	-	-
Faster R-CNN [46]	86.71	81.84	71.12	78.86	65.90	61.18	72.26	63.35	55.90
Ours	**93.04**	**88.64**	**79.10**	**81.78**	**67.47**	**64.70**	**78.39**	**68.94**	**61.37**

TABLE 3: AOS scores (in %) on the test set of KITTI's Object Detection and Orientation Estimation Benchmark. †: LIDAR-based methods.

Method	Cars			Pedestrians			Cyclists		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
CSoR† [52]	33.97	25.38	21.95	-	-	-	-	-	-
VeloFCN† [54]	70.58	52.84	46.14	-	-	-	-	-	-
LSVM-MDPM-sv [3], [60]	67.27	55.77	43.59	43.58	35.49	32.42	27.54	22.07	21.45
DPM-VOC+VP [34]	72.28	61.84	46.54	53.55	39.83	35.73	30.52	23.17	21.58
OC-DPM [61]	73.50	64.42	52.40	-	-	-	-	-	-
SubCat [36]	83.41	74.42	58.83	44.32	34.18	30.76	-	-	-
3DVP [41]	86.92	74.59	64.11	-	-	-	-	-	-
Ours	**91.44**	**86.10**	**76.52**	**72.94**	**59.80**	**57.03**	**70.13**	**58.68**	**52.35**

7.98%, 3.52% and 5.47% improvement in AP_{2D} for *Car*, *Pedestrian*, and *Cyclist*, in the *Hard* regime.

For the task of joint 2D object detection and orientation estimation, our approach also outperforms all methods by a large margin. In particular, our approach obtains $\sim$12% higher AOS scores than 3DVP [41] on *Car* in the *Moderate* and *Hard* regimes. For *Pedestrian* and *Cyclist*, the improvements are even more significant as our results exceed the second best method by more than 20%.

Comparison with Baselines We also apply our detection network on two state-of-the-art bottom-up proposals, Selective Search (SS) [7] and EdgeBoxes (EB) [11]. We conduct the experiments on the validation set. 2K proposals per image are used for all methods. As shown in Table 4, our approach outperforms Selective Search and EdgeBoxes by around 20% in terms of 2D detection AP and orientation estimation score AOS. We also report AP_{2D} as a function of the proposal budget for *Car* on *Moderate* data, in Fig. 8. When using only 100 proposals, our approach already achieves 70% AP, while EdgeBoxes and Selective Search obtain only 25.9% and 16.5%, respectively. Furthermore, EdgeBoxes reaches its best performance (78.7%) with 5000 proposals, while our approach needs only 200 proposals to achieve a similar AP.

5.3 3D Object Detection Performance

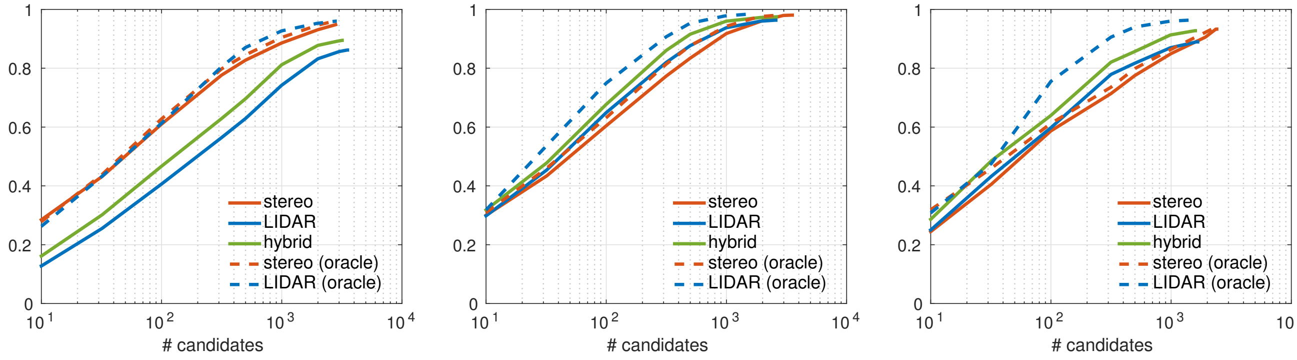
As the KITTI object detection benchmark does not evaluate 3D object detection on the (held-out) test set, we evaluate 3D object detection performance on the validation set. We use 1m and 2m as the threshold when evaluating Average Localization Precision (ALP). We report the results of our basic model with VGG-16 network for *Car* in Table 6. Results of RGB-HHA models are presented in Table 8. We obtain the highest AP_{3D} and ALP with a two-stream RGB-HHA model using proposals generated with hybrid data, i.e., combining stereo and LIDAR. In particular, we achieve 81.21% AP_{3D} and 75.44%/88.83% ALP within threshold of 1m/2m in the *Moderate* regime for 3D Car detection.

5.4 Stereo vs LIDAR

We also apply our proposal method to LIDAR data. Compared with stereo, LIDAR point cloud has more precise depth estimation while being very sparse. Since road plane

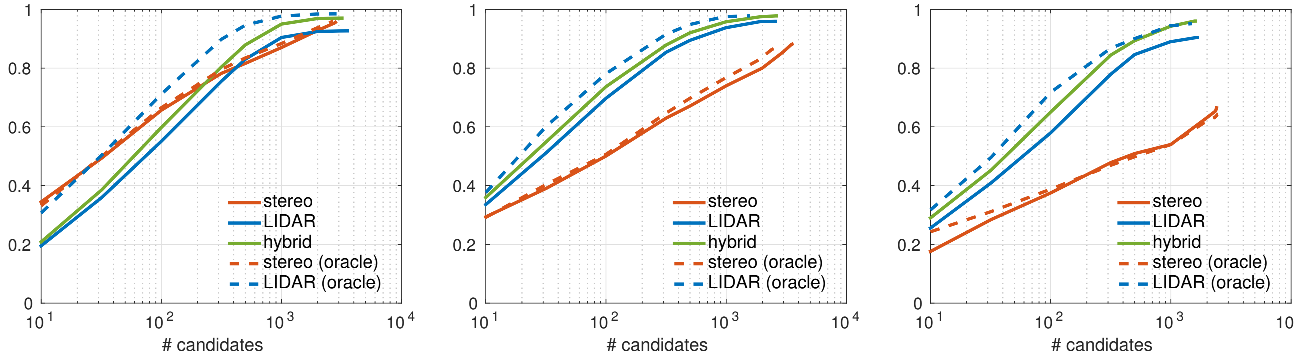
TABLE 4: **Object detection (top)** and **orientation estimation (bottom) results on KITTI's validation set**. Here, ort: orientation regression loss; ctx: contextual information; cls: class-specific weights in proposal generation. All methods use 2K proposals per image. VGG-16 network is used.

Metric	Method	ort	ctx	cls	Cars			Pedestrians			Cyclists		
					Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
AP$_{2D}$	SS [7]		-		75.91	60.00	50.98	54.06	47.55	40.56	56.26	39.16	38.83
	EB [11]				86.81	70.47	61.16	57.79	49.99	42.19	55.01	37.87	35.80
	Ours			✓	92.18	87.26	78.58	72.56	69.08	61.34	**90.69**	62.82	58.26
		✓		✓	92.67	87.52	78.78	72.42	**69.42**	61.55	85.92	62.54	57.71
		✓	✓		92.76	87.30	78.61	**73.76**	66.26	**63.15**	85.91	62.82	57.05
		✓	✓	✓	**93.08**	**88.07**	**79.39**	71.40	64.46	60.39	83.82	**63.47**	**60.93**
AOS	SS [7]		-		73.91	58.06	49.14	44.55	39.05	33.15	39.82	28.20	28.40
	EB [11]				83.91	67.89	58.34	46.80	40.22	33.81	43.97	30.36	28.50
	Ours			✓	39.52	38.24	34.01	34.15	33.08	29.27	63.88	43.85	40.36
		✓		✓	91.46	**85.80**	76.73	**62.25**	**59.15**	**52.24**	**77.60**	**55.75**	51.23
		✓	✓		91.22	85.12	75.74	61.62	55.01	52.14	74.28	53.96	49.05
		✓	✓	✓	**91.58**	**85.80**	**76.80**	61.57	54.79	51.12	73.94	55.59	**53.00**



Car Pedestrian Cyclist

Fig. 9: **Stereo vs LIDAR: 2D bounding box Recall vs number of Candidates on *Moderate* data**. We use an overlap threshold of 0.7 for *Car*, and 0.5 for *Pedestrian* and *Cyclist*. By hybrid we mean the approach that uses both stereo and LIDAR for road plane estimation, and LIDAR for feature extraction.



Car Pedestrian Cyclist

Fig. 10: **Stereo vs LIDAR: 3D bbox Recall vs number of Candidates on *Moderate* data**. IoU threshold is set to 0.25.

estimation is less accurate when using only sparse LIDAR point cloud, we also experiment with a *hybrid* approach, which uses dense stereo point cloud to compute superpixel features for road region classification, and LIDAR point cloud to fit a ground plane and to compute energy potentials for inference in our model.

Proposal Recall: As shown in Fig. 10, using LIDAR point cloud significantly boosts the 3D bounding box recall, especially for small objects, i.e., pedestrians, cyclists and distant objects. We obtain the highest 3D recall with the hybrid approach, which combines stereo and LIDAR for road estimation. In terms of 2D bounding box recall shown in Fig. 9, stereo works slightly better.

Detection Performance: We report 2D/3D detection performance in Table 5 and Table 6 for comparison of stereo and LIDAR approaches. In terms of 2D detection and orientation estimation, we obtain the best performance with stereo. For 3D detection, we achieve significantly higher 3D AP and ALP using LIDAR data under the *Moderate* and *Hard* regimes. However, stereo still works better in the *Easy*

TABLE 5: **Stereo vs LIDAR on 2D Object Detection and Orientation Estimation:** AP_{2D} and AOS for *Car* on validation set of KITTI. VGG-16 network is used.

Data	AP_{2D}			AOS		
	Easy	Moderate	Hard	Easy	Moderate	Hard
stereo (oracle)	92.73	88.30	79.48	90.98	86.08	76.90
LIDAR (oracle)	**93.21**	**88.77**	**79.70**	**91.67**	**86.61**	**77.22**
stereo	**93.08**	**88.07**	**79.39**	**91.58**	**85.80**	**76.80**
LIDAR	87.78	79.51	70.74	85.90	77.24	68.23
hybrid	92.17	86.52	78.37	90.62	84.44	75.91

TABLE 6: **Stereo vs LIDAR on 3D Object Detection:** AP_{3D} and ALP for *Car* on KITTI validation. VGG-16 network is used.

Data	AP_{3D}			ALP (< 1m)			ALP (< 2m)		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
stereo (oracle)	**87.33**	70.46	63.82	**75.98**	58.01	52.37	**94.12**	78.64	73.95
LIDAR (oracle)	84.63	**82.04**	**74.92**	75.32	**74.54**	**68.50**	91.88	**89.29**	**84.06**
stereo	**88.45**	69.52	62.65	**77.90**	58.09	52.17	**94.39**	77.57	72.69
LIDAR	80.73	73.56	66.83	72.26	67.77	62.42	87.98	81.07	76.52
hybrid	86.47	**80.56**	**73.71**	77.19	**73.85**	**67.99**	93.00	**87.52**	**82.51**

TABLE 7: **Comparison of different architectures on 2D object detection and orientation estimation:** AP_{2D} and AOS for *Car* on validation set of KITTI. 7-layer VGG_CNN_M_1024 [50] network is used.

Data	Approach	AP_{2D}			AOS		
		Easy	Moderate	Hard	Easy	Moderate	Hard
stereo	RGB	92.56	87.27	78.38	90.24	83.98	74.69
	RGB-HHA, one-stream	90.81	87.41	78.82	**90.80**	**84.28**	**75.39**
	RGB-HHA, two-stream	**93.03**	**87.97**	**78.98**	90.34	84.27	74.90
LIDAR	RGB	87.12	78.64	69.85	84.30	75.08	66.13
	RGB-HHA, one-stream	87.42	78.98	70.11	84.88	75.51	66.53
	RGB-HHA, two-stream	**88.04**	**79.39**	**70.48**	**85.05**	**75.70**	**66.63**
hybrid	RGB	90.99	84.40	76.02	87.93	80.47	71.76
	RGB-HHA, one-stream	90.81	84.40	**77.12**	88.45	**80.98**	**73.20**
	RGB-HHA, two-stream	**92.69**	**84.78**	76.43	**89.23**	80.53	71.73

TABLE 8: **Comparison of different architectures on 3D object detection:** AP_{3D} and ALP for *Car* on validation set of KITTI. 7-layer VGG_CNN_M_1024 [50] network is used.

Data	Approach	AP_{3D}			ALP (< 1m)			ALP (< 2m)		
		Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
stereo	RGB	77.50	56.97	50.84	64.89	47.34	42.20	89.02	69.27	64.74
	RGB-HHA, one-stream	89.22	68.19	**62.24**	81.00	58.53	**52.76**	95.23	76.57	72.50
	RGB-HHA, two-stream	**90.43**	**68.90**	62.22	**82.25**	**58.99**	52.71	**95.74**	**77.66**	**73.03**
LIDAR	RGB	76.51	68.77	62.10	66.46	62.14	57.53	86.22	78.67	74.44
	RGB-HHA, one-stream	**84.83**	73.92	67.55	76.61	68.52	63.25	**92.16**	82.56	77.95
	RGB-HHA, two-stream	84.02	**74.11**	**67.62**	**77.40**	**69.62**	**63.92**	91.32	**82.72**	**78.13**
hybrid	RGB	81.83	75.86	68.66	71.12	68.46	62.89	90.59	85.10	79.86
	RGB-HHA, one-stream	87.86	79.61	72.86	79.54	74.06	68.58	94.88	87.94	83.16
	RGB-HHA, two-stream	**89.49**	**81.21**	**74.32**	**82.16**	**75.44**	**69.27**	**95.46**	**88.83**	**83.75**

regime. This demonstrates the advantage of LIDAR point cloud in detecting small, occluded, and distant objects owing to its precise depth information, while dense point cloud from stereo works better for objects at shorter distances. When using the hybrid approach, we get the highest 3D accuracy in the *Moderate* and *Hard* setting.

5.5 Ablation Studies

Network Design: We study the effect of orientation regression and the contextual branch in our detection network. As shown in Table 4, by jointly doing bounding box and orientation regression we get a large boost in performance in terms of the AOS, while the 2D detection AP remains very similar. By adding the contextual branch, we achieve the highest AP_{2D} and AOS on *Car*. We also observe different effect on *Pedestrian* and *Cyclist* in some cases, which is probably due to the fact that the contextual branch nearly doubles the size of the model and makes it more difficult to train with very few pedestrian and cyclist instances.

RGB-D Networks: We study the effect of depth features in CNN scoring model on 2D detection, orientation estimation, and 3D detection. Comparisons of different approaches are shown in Table 7 and Table 8. As the RGB-HHA models require more GPU memory, we use 7-layer VGG_CNN_M_1024 network [50] for this experiment.

Overall, RGB-HHA models achieve improvement over the RGB model for both 2D and 3D detection. The two-stream RGB-HHA model also gains in performance over the one-stream RGB-HHA model in most cases. The improvement is significant for 3D detection while marginal (∼0.5%) for 2D detection. In particular, the two-stream RGB-HHA model outperforms the RGB model by about 10% and 5% for the stereo and LIDAR/hybrid approaches respectively for 3D detection. Note that for 3D car detection, we obtain the highest accuracy using a 7-layer two-stream RGB-HHA model with hybrid data, which even outperforms the 16-layer RGB model (see Table 6). This suggests the importance of depth information for the 3D object detection task.

Ground Plane: To study the effect of ground plane estimation on detection performance, we allow access to "oracle" ground planes. We approximate the oracle ground plane by fitting a plane to the footprints of ground truth 3D bounding boxes. We show results for both, the stereo and LIDAR approaches on *Car* detection. Proposal recall plots are shown in Fig. 9 and Fig. 10. 2D/3D detection results are shown in Table 5 and Table 6. Giving access to oracle ground plane significantly boosts the 2D and 3D box recall of the LIDAR approach. Similarly, AP_{2D}/AOS improve by about 9%, AP_{3D}/ALP by about 8% for the LIDAR approach, while the improvement on stereo is marginal. This suggests that exploiting a better approach for ground plane estimation using sparse LIDAR point clouds would further improve the performance of the LIDAR approach.

6 Conclusion

We presented a novel approach to 3D object detection in the context of autonomous driving. In contrast to most existing work, we take advantage of stereo imagery and generate a set of 3D object proposals that are then run through a convolutional neural network to obtain high-quality 3D object detections. We generate 3D proposals by minimizing an energy function that encodes object size priors, ground plane context, and some depth informed features. For CNN scoring, we exploit appearance as well as depth and context information in our neural net and jointly predict 3D bounding box coordinates and object pose.

Our approach significantly outperforms existing state-of-the-art object proposal methods on the KITTI benchmark. Particularly, our approach achieves 25% higher recall than the state-of-the-art RGB-D method MCG-D [18] for 2K proposals. We have evaluated our full 3D detection pipeline on the tasks of 2D object detection, joint 2D object detection and orientation estimation, as well as 3D object detection on KITTI. Our method significantly outperforms all previous published object detection methods for all three object classes on the challenging KITTI benchmark [15].

Acknowledgments

The work was partially supported by NSFC 61171113, NBRPC 2016YFB0100900, NSERC, ONR-N00014-14-1-0232 and Toyota Motor Corporation. We would like to thank NVIDIA for supporting our research by donating GPUs.

References

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

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

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

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

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

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

[7] K. Van de Sande, J. Uijlings, T. Gevers, and A. Smeulders, "Segmentation as selective search for object recognition," in *ICCV*, 2011.

[8] P. Arbelaez, J. Pont-Tusetand, J. Barron, F. Marques, and J. Malik, "Multiscale combinatorial grouping," in *CVPR*, 2014.

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

[10] M. Cheng, Z. Zhang, M. Lin, and P. Torr, "BING: Binarized normed gradients for objectness estimation at 300fps," in *CVPR*, 2014.

[11] L. Zitnick and P. Dollár, "Edge boxes: Locating object proposals from edges," in *ECCV*, 2014.

[12] P. Kr ahenb uhl and V. Koltun, "Learning to propose objects," in *CVPR*, 2015.

[13] T. Lee, S. Fidler, and S. Dickinson, "A learning framework for generating region proposals with mid-level cues," in *ICCV*, 2015.

[14] A. Ghodrati, A. Diba, M. Pedersoli, T. Tuytelaars, and L. V. Gool, "Deepproposal: Hunting objects by cascading deep convolutional layers," in *ICCV*, 2015.

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

[16] R. Girshick, "Fast R-CNN," in *ICCV*, 2015.

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

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

[19] 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.

[20] J. Carreira and C. Sminchisescu, "Cpmc: Automatic object segmentation using constrained parametric min-cuts," *PAMI*, vol. 34, no. 7, pp. 1312–1328, 2012.

[21] 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.

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

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

[24] D. Oneata, J. Revaud, J. Verbeek, and C. Schmid, "Spatio-temporal object detection proposals," in *ECCV*, 2014.

[25] K. Fragkiadaki, P. Arbelaez, P. Felsen, and J. Malik, "Learning to segment moving objects in videos," in *CVPR*, 2015.

[26] S. Manen, M. Guillaumin, and L. Van Gool, "Prime object proposals with randomized prim's algorithm," in *ICCV*, 2013.

[27] P. Kr ahenb uhl and V. Koltun, "Geodesic object proposals," in *ECCV*, 2014.

[28] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu, "Semantic segmentation with second-order pooling," in *ECCV*, 2012.

[29] S. Fidler, R. Mottaghi, A. Yuille, and R. Urtasun, "Bottom-up segmentation for top-down detection," in *CVPR*, 2013.

[30] Z. Zhang, Y. Liu, T. Bolukbasi, M.-M. Cheng, and V. Saligrama, "Bing++: A fast high quality object proposal generator at 100fps," *arXiv:1511.04511*, 2015.

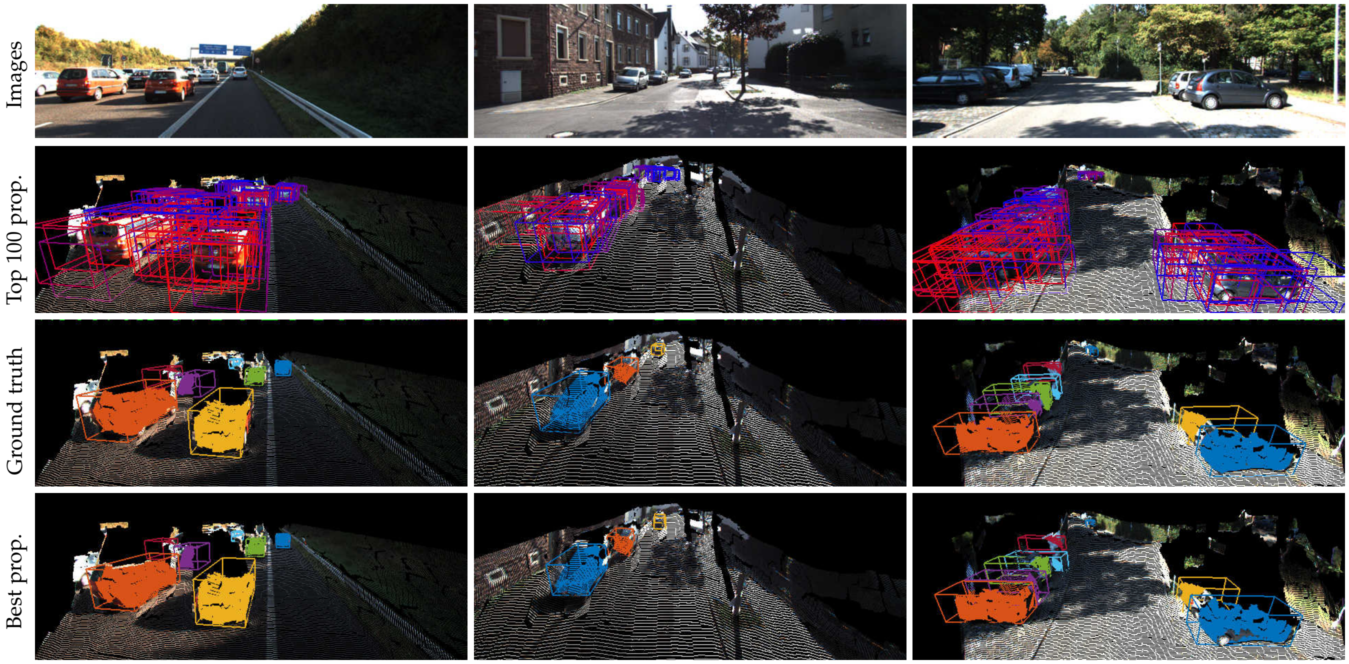


Fig. 11: Qualitative results for the *Car* class. We show the original image, 100 top scoring proposals, ground-truth 3D boxes, and our best set of proposals that cover the ground-truth.

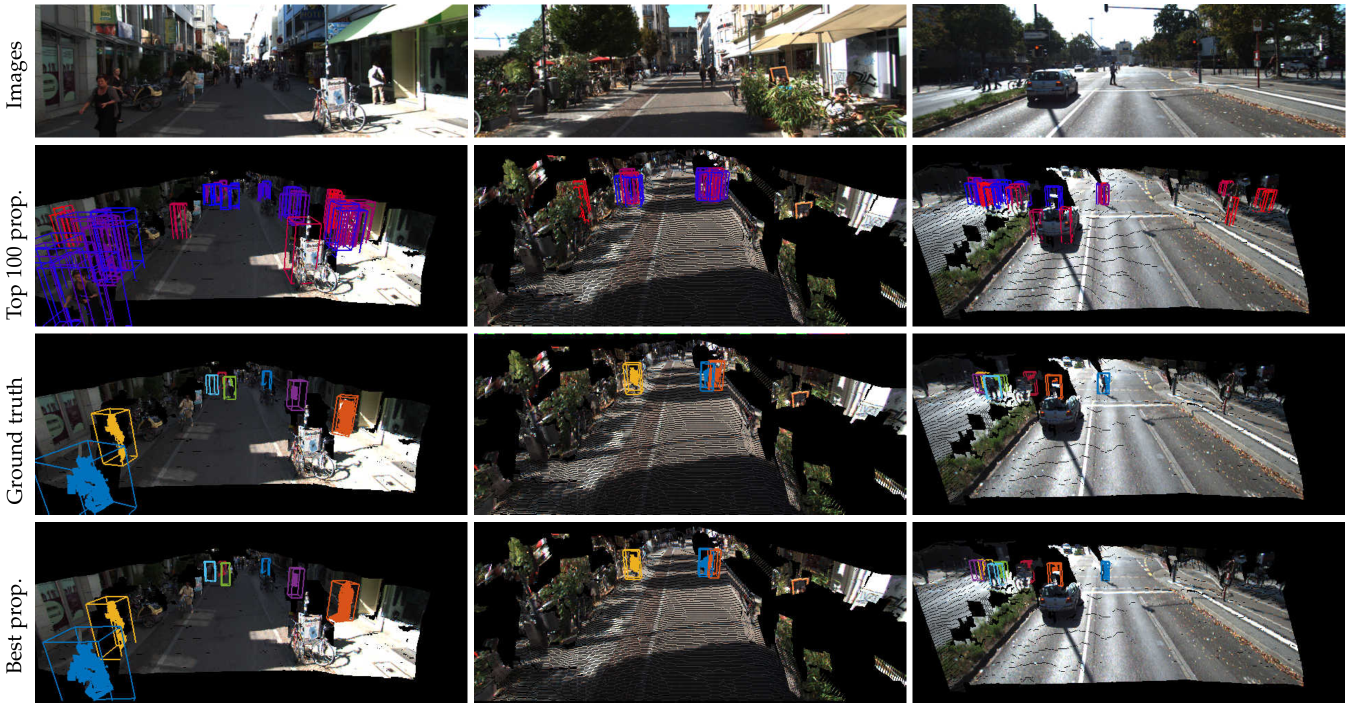


Fig. 12: Qualitative examples for the *Pedestrian* class.

[31] X. Chen, H. Ma, X. Wang, and Z. Zhao, "Improving object proposals with multi-thresholding straddling expansion," in *CVPR*, 2015.

[32] J. Hosang, R. Benenson, P. Dollár, and B. Schiele, "What makes for effective detection proposals?" *PAMI*, 2015.

[33] S. Song and J. Xiao, "Sliding shapes for 3d object detection in depth images," in *ECCV*, 2014.

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

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

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

[37] C. Long, X. Wang, G. Hua, M. Yang, and Y. Lin, "Accurate object detection with location relaxation and regionlets relocalization," in *ACCV*, 2014.

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

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

[40] Q. Hu, S. Paisitkriangkrai, C. Shen, A. van den Hengel, and F. Porikli, "Fast detection of multiple objects in traffic scenes with a common detection framework," *T-ITS*, 2015.

[41] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Data-driven 3d voxel patterns for object category recognition," in *CVPR*, 2015.

[42] P. Dollár, R. Appel, S. Belongie, and P. Perona, "Fast feature pyramids for object detection," *PAMI*, 2014.

[43] J. Hosang, M. Omran, R. Benenson, and B. Schiele, "Taking a deeper look at pedestrians," *CVPR*, 2015.

[44] Y. Tian, P. Luo, X. Wang, and X. Tang, "Deep learning strong parts for pedestrian detection," in *ICCV*, 2015.

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

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

[47] K. Yamaguchi, D. McAllester, and R. Urtasun, "Efficient joint segmentation, occlusion labeling, stereo and flow estimation," in *ECCV*, 2014.

[48] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun, "Support Vector Learning for Interdependent and Structured Output Spaces," in *ICML*, 2004.

[49] 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.

[50] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *BMVC*, 2014.

[51] J. Behley, V. Steinhage, and A. B. Cremers, "Laser-based Segment Classification Using a Mixture of Bag-of-Words," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2013.

[52] L. Plotkin, "Pydriver: Entwicklung eines frameworks fr rumliche detektion und klassifikation von objekten in fahrzeugumgebung," Bachelor's Thesis, Karlsruhe Institute of Technology, 2015.

[53] D. Z. Wang and I. Posner, "Voting for voting in online point cloud object detection," in *Proc. of Robotics: Science and Systems*, 2015.

[54] B. Li, T. Zhang, and T. Xia, "Vehicle detection from 3d lidar using fully convolutional network," in *Proc. of RSS*, 2016.

[55] A. Gonzalez, G. Villalonga, J. Xu, D. Vazquez, J. Amores, and A. Lopez, "Multiview random forest of local experts combining rgb and lidar data for pedestrian detection," in *IV*, 2015.

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

[57] C. Premebida, J. Carreira, J. Batista, and U. Nunes, "Pedestrian detection combining rgb and dense lidar data," in *IROS*, 2014.

[58] S. Paisitkriangi, C. Shen, and A. van den Hengel, "Pedestrian detection with spatially pooled features and structured ensemble learning," in *arXiv:1409.5209*, 2014.

[59] S. Zhang, R. Benenson, and B. Schiele, "Filtered channel features for pedestrian detection," in *CVPR*, 2015.

[60] A. Geiger, C. Wojek, and R. Urtasun, "Joint 3d estimation of objects and scene layout," in *NIPS*, 2011.

[61] B. Pepik, M. Stark, P. Gehler, and B. Schiele, "Occlusion patterns for object class detection," in *CVPR*, 2013.

Xiaozhi Chen received the B.S. degree in Electronic Engineering from Tsinghua University, Beijing, China in 2012, where he is currently pursuing the Ph.D degree. His research interests include computer vision and machine learning.

Kaustav Kundu received the B.Tech. (Hons.) degree in Computer Science and Engineering from IIIT Hyderabad, India in 2012. He is currently pursuing his Ph.D. degree in Computer Science from University of Toronto. His research interests include computer vision and machine learning.

Yukun Zhu received the BEng degree in electronic engineering from Shanghai Jiao Tong University in 2011, dual MSc degree in information & communication engineering from Shanghai Jiao Tong University and electric & computer engineering from Georgia Institute of Technology in 2014, the MSc degree in computer science from University of Toronto in 2016. He is currently working at Google. His research interests include computer vision and machine learning.

Huimin Ma received the M.S. and Ph.D. degrees in Mechanical Electronic Engineering from Beijing Institute of Technology, Beijing, China in 1998 and 2001 respetively. She is an associate professor in the Department of Electronic Engineering of Tsinghua University, and the director of 3D Image Simulation Lab. She worked as an visiting scholar in University of Pittsburgh in 2011. She is also the executive director and the vice secretary general of China Society of Image and Graphics. Her research and teaching interests include 3D object recognition and tracking, system modeling and simulation, psychological base of visual cognition.

Sanja Fidler is an Assistant Professor at the Department of Computer Science, University of Toronto. Previously she was a Research Assistant Professor at TTI-Chicago. She completed her PhD in computer science at University of Ljubljana in 2010, and was a postdoctoral fellow at University of Toronto in 2011-2012. In 2010 she visited UC Berkeley. She has served as a Program Chair of the 3DV conference, and as Area Chair of CVPR, ICCV, ACCV, EMNLP, ICLR, and NIPS. She received the NVIDIA Pioneer of AI award. Her main research interests lie in the intersection of language and vision, as well as 3D scene understanding.

Raquel Urtasun Raquel Urtasun is an Associate Professor in the Department of Computer Science at the University of Toronto and a Canada Research Chair in Machine Learning and Computer Vision. Prior to this, she was an Assistant Professor at the Toyota Technological Institute at Chicago (TTIC), an academic computer science institute affiliated with the University of Chicago. She received her Ph.D. degree from the Computer Science department at Ecole Polytechnique Federal de Lausanne (EPFL) in 2006 and did her postdoc at MIT and UC Berkeley. Her research interests include machine learning, computer vision, robotics and remote sensing. Her lab was selected as an NVIDIA NVAIL lab. She is a recipient of an NSERC EWR Steacie Award, an NVIDIA Pioneers of AI Award, a Ministry of Education and Innovation Early Researcher Award, three Google Faculty Research Awards, an Amazon Faculty Research Award, a Connaught New Researcher Award and a Best Paper Runner up Prize awarded at the Conference on Computer Vision and Pattern Recognition (CVPR). She is also Program Chair of CVPR 2018, an Editor of the International Journal in Computer Vision (IJCV).