Multi-View Reprojection Architecture for Orientation Estimation

Hee Min Choi, Hyoa Kang and Yoonsuk Hyun
Samsung Electronics, Korea
{chm.choi, hyoa.kang, yoonsuk.hyun}@samsung.com

Abstract

In autonomous driving scenarios, pose estimation of surrounding vehicles and other objects is required in decision making and planning. This paper proposes Multi-View Reprojection Architecture, a flexible, highly accurate architecture to adopt any 2D detection network and extend it to regress the orientation of object's 3D bounding box and its dimensions. In contrast to previous techniques, our network incorporates geometric constraints on 3D box imposed by 2D detection box. In particular, we regress dimensions, orientation and reprojected boxes in multi-view obtained from a novel 3D reconstruction layer using perspective geometry. In the 3D reconstruction layer, we use an iterative refinement strategy to accurately recover 3D boxes even when 2D boxes are truncated. The proposed architecture is shown to outperform state-of-the-art methods on the challenging KITTI car orientation benchmark and obtain top results on 3D detection benchmark while running in real time, making it suitable for autonomous vehicles.

1. Introduction

Over the last years, deep learning approaches have dramatically improved the performance of vision-based 2D object detection system. However, object detection as 2D bounding box in image is not sufficient for autonomous driving cars to perform planning and decision making. For a robust perception system in self-driving vehicle, pose estimation of surrounding objects is essential. In this paper, we are interested in heading angle estimation of surrounding vehicles from monocular images in the context of self-driving cars. This is a relevant research field because currently most cars are equipped with front-facing mono camera. In addition, vehicles have rigid bodies with well-known geometry, so we are able to recover 3D vehicle information from monocular images.

In this paper, we propose a flexible, highly accurate method that estimates the orientation and dimensions of an object's 3D bounding box from a 2D detection result and corresponding image crop. One of the main contributions of our approach is in the design of Multi-View Reprojection Architecture called MVRA and the associated training objective functions for the problem. In particular, the proposed architecture regresses to 3D dimensions, orientation and novel reprojected 2D boxes in image and bird-eye view map. This is in contrast to previous techniques that attempt to simply estimate the 3D box parameters separately.

We leverage the success of mature 2D object detector and extend it by training a convolutional neural network (CNN) to regress the orientation of the object's 3D bounding box and its spatial dimensions. In particular, the proposed architecture encodes geometric constraints on 3D box imposed by 2D detection window. The idea is to add a 3D reconstruction layer in which we recover the object's 3D bounding box using estimated dimensions and orientation and the constraints that the perspective projection of a 3D box should fit tightly into the 2D box in image. Indeed, the idea of 3D reconstruction is recently introduced in [15] and used in the post-processing stage to estimate 3D box. Inspired by their success, we improve this reconstruction idea and bring it into network itself to better regress orientation and 3D template (3D dimensions). In contrast to [15], the projections in image and bird-eye view map obtained from the recovered 3D box are included in the training target as well as the 3D object parameters. In order to regress to the reprojected boxes, we used novel multi-view reprojection loss. Although conceptually straightforward, our method is proved to be effective. Our experimental results reveal that the proposed 3D reconstruction layer improves orientation regression performance. We also introduce a novel iterative orientation refinement method for handling truncated 2D boxes in the 3D reconstruction layer and demonstrate that our method is effective.

Another contribution is in the introduction of novel box augmentation strategy. We use the ground truth 3D information and reconstruction idea to automatically produce input 2D boxes for accurate orientation estimation robust to viewpoint changes and occlusions.

We evaluate our method on the KITTI dataset. On the KITTI dataset [5], we performed a comparison of our results to the results of other methods based on various in-

put sources such as monocular and stereo imagery, LiDAR and sensor fusion. The proposed architecture is shown to outperform all published state-of-art methods on the challenging KITTI car orientation estimation benchmark while running in real time. We also performed a thorough analysis of our 3D reconstruction layer and multi-view reprojection loss on the KITTI validation dataset.

Moreover, we note that the 3D box information itself from our 3D construction layer can be effectively used. Indeed, our method obtains top results on KITTI 3D object detection benchmark, being the third best among all monocular methods that do not use network for either depth estimation or pseudo-LiDAR point cloud.

The remainder of this paper is organized as follows. In the next section, related work is reviewed. Section 3 explains our proposed method in detail. Experimental results demonstrating effectiveness of our approach for KITTI cars are illustrated in Section 4. In the final section, we summarize our contributions.

2. Related work

We highlight some of recent related works on monocular 3D object detection. Monocular 3D object detection method can be divided into 3 categories by types of features used therein. After presenting (1) RGB-only works, we review works utilizing (2) synthetically generated features or (3) 3D shape information.

RGB data only. Mono3D [4] draws 3D box samples in the physical world assuming flat ground plane constraint. The sampled boxes are scored by semantic segmentation, high level contextual, shape and category specific features. SubCNN [31] clusters the set of possible poses into viewpoint-dependent sub-categories. The subcategories are obtained by clustering 3D voxel patterns introduced in [30].

Recently, there are some works on single-stage monocular 3D object detector. SSD-6D [8] extends the popular SSD [12] paradigm to provide 6D pose of 3D objects by discretizations of the full rotational space. In [25], a new CNN architecture inspired by SSD that predicts the 2D image locations of the projected vertices of the object's 3D box is proposed. The 6D pose is recovered by PnP algorithm. SS3D [7] framework consists of a CNN, which outputs a redundant representation of each object in image with corresponding uncertainty measures, and a 3D box fitting optimizer.

Several approaches are based on a two-stage architecture. MonoDIS [23] designs a CNN architecture which disentangles dependencies of different parameters by isolating and handling parameter groups individually at a loss level. In Shift R-CNN [16], an adapted Faster R-CNN [20] network regresses 2D box and 3D object properties, a mathematical system of equation is solved using least squares of the inverse 2D to 3D mapping problem, and the final result is refined by another network. GS3D [11] modifies Faster R-CNN framework by adding a new branch of orientation prediction to obtain a coarse cuboid for each object. In contrast to other methods that only use the feature extracted from the 2D box for box refinement, GS3D exploits features from visible surfaces of the projected cuboid on 2D image to obtain the final 3D pose.

The work that most related to ours is the one utilizing 2D detector's outputs and regressing 3D model information. Deep3DBox [15] uses a CNN to regress the 3D box dimensions and orientation. In contrast to our work, they simply regress 3D object properties separately. Using the estimated dimensions and orientation, full 3D pose is recovered by exploiting constraints from projective geometry. The key idea is that the perspective projection of a 3D box should fit tightly to at least one side of its corresponding 2D box detection.

Synthetically generated features. OFTNet [21] introduces an orthographic feature transform which maps image-based features into an orthographic birds-eye-view, implemented efficiently using integral-image representation. There are several approaches that take advantage of depth information. ROI-10D [14] processes input image for 2D detection and monocular depth prediction networks and uses the predicted regions of interest (RoIs) to extract fused feature maps from both networks for 3D box regression. MonoGRNet [19] consists of four sub-networks for 2D detection, instance depth estimation, 3D local estimation, and local corner regression. In MonoGRNet, the network first estimates depth and 2D projection of the 3D box center to seek for the global 3D location, and then predicts corner coordinates in local context.

Other works are based on generation of a pseudo-LiDAR point cloud from image input. MultiFusion [32] estimates the disparity and infers 3D point cloud, and then fused features are extracted from RGB image, disparity information and the point cloud. In Pseudo-LiDAR [27] and AM3D [13], generated pseudo-LiDAR point cloud is processed in LiDAR-based 3D detection algorithm like PointNet [18]. Mono3DPLiDAR [29] mitigates local misalignment and long tail issues of pseudo point cloud by using bounding box consistency constraint and instance mask. MonoPSR [9] leverages 3D proposals and shape reconstruction. The 3D proposals are generated from 2D detection box using the fundamental relations of a pinhole camera model, and simultaneously a point cloud is predicted to learn local scale and shape information.

3D shape information. 3D-RCNN [10] proposes an inverse-graphics CNN framework for instance-level 3D understanding. The CNN learns to map image regions to the full 3D shape and pose of all object instances. In [33], authors reason jointly about the 3D shape of multiple ob-

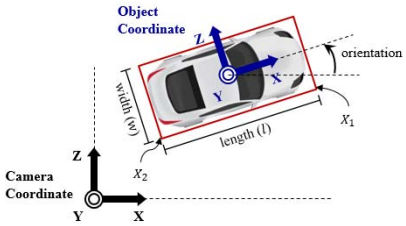

Figure 1. Coordinate system of 3D bounding box with respect to camera coordinate system. Here, orientation corresponds to azimuth angle.

jects. DeepMANTA [3] uses 3D CAD models and designs a many-task CNN architecture that optimizes region proposal, orientation, 2D box regression, part localization, part visibility, and 3D template prediction simultaneously. In Mono3D++ [6], a morphable wireframe model for generating a fine-scaled representation of vehicle 3D shape and pose is used, and its network is trained to optimize projection consistency between generated 3D hypotheses and corresponding 2D pseudo-measurements.

3. MVRA approach

In this section, we describe our proposed approaches for accurate orientation estimation from monocular images. Our network architecture has two parts. First, 2D detection crop is passed through the CNN architecture that outputs orientation of object's 3D bounding box and its dimensions. The CNN network architecture is described in Section 3.3. The second part is a 3D reconstruction layer, which incorporates 3D geometry imposed by 2D bounding box into network in order to better regress the dimensions and orientation. The 3D reconstruction layer is elaborated in Section 3.1. We also introduce a novel iterative orientation refinement method in the 3D reconstruction layer for taking care of truncated boxes in Section 3.2.

3.1. 3D reconstruction layer and multi-view representation

The 3D reconstruction layer takes as input dimensions, orientation and 2D box coordinates and outputs 3D box. Indeed, we further estimate the 3D location of the object's bottom center in camera coordinate in the reconstruction layer. This reconstruction is based on the representation of projected 3D box as a function of the spatial dimensions, orientation and original 2D box as in [15]. From well-known projective geometry, the relation between a 3D point $\mathbf{X} = [X, Y, Z, 1]^T$ in object's (homogeneous) coordinate and its 2D projection $\mathbf{x} = [x, y, 1]^T$ in (homogeneous) image coordinate is

$$\mathbf{x} = K \begin{bmatrix} \mathbf{R} & \mathbf{T} \end{bmatrix} \mathbf{X}, \tag{1}$$

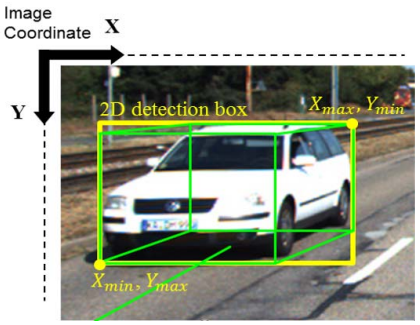

Figure 2. Point-to-side correspondence. The projected 3D points that are active constraints in each of the 2D box sides are shown with a circle.

where $(\mathbf{R}, \mathbf{T})$ is the the 3D pose of the object in camera coordinate and K is the camera intrinsics matrix. On the other hand, assuming the origin of the object's coordinate frame is located at the 3D box bottom center, the 3D box corners are described by its spatial dimensions $\mathbf{D} = [h, w, l]^T$: $\mathbf{X}_0 = [l/2, 0, w/2]^T$, $\mathbf{X}_1 = [l/2, 0, -w/2]^T$, $\mathbf{X}_2 = [-l/2, 0, -w/2]^T$, $\mathbf{X}_3 = [-l/2, 0, w/2]^T$ and $\mathbf{X}_{i+4} = \mathbf{X}_i + [0, 0, -h]^T$ for $i = 0, 1, 2, 3$ (see Figure 1). Here, we assume that 2D detector is trained to produce 2D bounding boxes of the projected 3D box. The point-to-side correspondence constraint, enforcing each side of 2D bounding box to be touched by the projection of at least one of the 3D vertices, results in 4 linear equations from (1) corresponding to 2D side parameters, $(x_{min}, y_{min}, x_{max}, y_{max})$, with 3 unknown translation parameters in $\mathbf{T} = [t_x, t_y, t_z]^T$. For example, if 3D box corners (X_2, X_5, X_1, X_1), corresponds to $(x_{min}, y_{min}, x_{max}, y_{max})$, then the closed form expression of the resulting linear system of equations is given by

$$\begin{bmatrix} f_x & 0 & c_x - x_{min} \\ 0 & f_y & c_y - y_{min} \\ f_x & 0 & c_x - x_{max} \\ 0 & f_y & c_y - y_{max} \end{bmatrix} \mathbf{T} = \begin{bmatrix} \mathbf{a_2^{(3)}} x_{min} - \mathbf{a_2^{(1)}} \\ \mathbf{a_5^{(3)}} y_{min} - \mathbf{a_5^{(2)}} \\ \mathbf{a_1^{(3)}} x_{max} - \mathbf{a_1^{(1)}} \\ \mathbf{a_1^{(3)}} y_{max} - \mathbf{a_1^{(2)}} \end{bmatrix} \tag{2}$$

where (f_x, f_y) and (c_x, c_y) are focal lengths and principal points in the camera intrinsic matrix K, and $\mathbf{a_i^{(j)}}$ is jth element of vector $\mathbf{a_i}$. Here, given rotation matrix $\mathbf{R}(\theta, \phi, \alpha)$, each $\mathbf{a_i}$ is calculated by

$$\mathbf{a_i} = K\mathbf{R}\mathbf{X_i}. \tag{3}$$

Therefore, we can recover 3D box (see, e.g., Figure 2) by solving the over-constrainted system of equations (2) for the translation $\mathbf{T}$. In order to obtain $\mathbf{T}$, we calculate the 2D intersection over union (IoU) scores of the initial 2D box constraint and reprojected 3D box for all possible configurations of the point-to-side constraint and choose the best least squares solution minimizing the IoU. As commonly done in vehicle-oriented applications, we assume object pitch ϕ

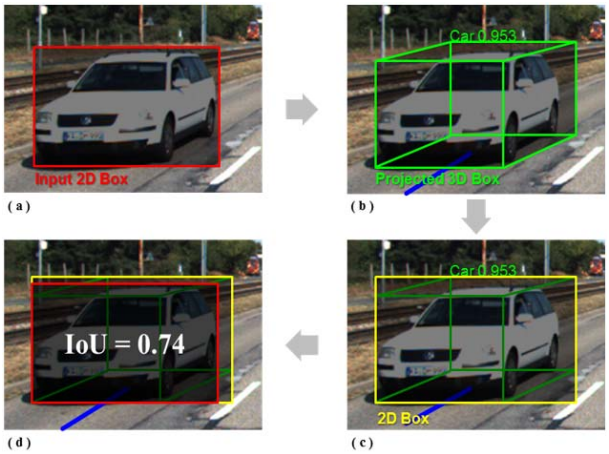

Figure 3. Reprojected box in image. (a) illustrates the input 2D box in red. (b) describes the projected 3D box in green obtained by point-to-side correspondence constraint. (c) tightly draw 2D box in yellow fitting projected 3D box. (d) visualizes intersection over union between yellow and red boxes.

and roll α angles are zeros, and thus the number of possible configurations is 64. (See [15] and [16] for details.) From this reconstructed 3D box, we obtain projected 2D boxes in bird-eye view map and image. Examples of the projected boxes in image and bird-eye view map are illustrated in Figure 3(c) and Figure 4(b, d), respectively.

3.2. Iterative orientation refinement method

We note that truncated initial 2D box may result in incorrect 3D reconstruction (see e.g., Figure 5). This difficult situation often happens in driving scenarios when some parts of vehicles are out of camera field of view. To ensure safe driving, we must deal with this issue properly. In this section, we will illustrate a novel iterative method of improving the reconstruction performance for this case.

As commonly done, we regress to the local orientation, which is the observation angle, instead of regressing to the global orientation (see e.g., [11], [15] and [16]). Figure 6 shows a relationship between local orientation and global orientation. This is due to the well-known fact that the local orientation is directly related to the appearance of the object in image. Assuming the center of crop goes through the actual center of 3D box, we are able to estimate the global orientation easily by summing the ray angle from the camera origin to the center of image crop and predicted local orientation. That is, $\theta = \theta_l + \theta_{ray}$ where θ_{ray} is an angle between two vectors of $((x_{center} - c_x)/f_x, 1)$ and camera x-axis. However, truncated detection box in image may lead to incorrect estimation of ray angle and global orientation, and in turn leads to incorrect 3D box reconstruction as described in Figure 5.

We now propose a novel iterative method for accurately estimating ray angle when one of 2D box parameters is out of the image plane and is clipped inside the image. Each iteration consists of 4 main steps: (1) Discretize the global orientation in a pre-specified range (e.g., $[0, \pi]$) with some resolution (e.g., $\pi/8$). (2) With these candidate global orientations and estimated dimensions, reconstruct 3D boxes using the method described in Section 3.1. Here, we do not include the clipped parameter in the point-to-side constraints equation (2) and compute T for each candidate global orientation value. (3) We then pick the best T that minimizes the reprojection error with respect to the initial 2D box. (4) Finally, calculate the ray angle directly from the selected T and recover the global orientation by combining it with the local orientation estimate. Repeat the same procedures described above with new global orientation candidates until convergence. The new candidates are selected with a finer resolution (e.g., $\pi/32$) and centered at the global orientation estimate in the final step of the previous iteration. In our experiment, we repeat the four-step procedures twice and are able to get sufficiently accurate 3D reconstructions. A qualitative illustration of our iterative method is in Figure 5. We also evaluated the global orientation accuracy to demonstrate the effectiveness of the proposed iterative method in Section 4.3.

3.3. Network architecture

We now present our multi-view reprojection architecture for regressing the orientation and object spatial dimensions. The network is constructed based on Deep3DBox [15] network with improvement. In particular, we encode geometric constraints imposed by 2D box in the network by adding the 3D reconstruction layer. The proposed CNN architecture is shown in Figure 7. There are three branches: two for orientation regression head and one branch for dimension regression head. The objective function for training consists of three losses: one for angle, another for 3D template, and the other for projected boxes. All branches follow after the shared convolutional feature layers, and the total loss is the weighted sum of the three loss functions:

$$L_{total} = \alpha_1 L_{multibin} + \alpha_2 L_{dim} + \alpha_3 L_{MVR}. \quad (4)$$

Here, the first and second terms are the MultiBin loss and 3D dimensions loss of [15]. As mentioned in Section 3.2, the regression target for orientation estimation is the local orientation. Again, at inference time, global orientation is determined by considering the ray angle. Also, if the detected boxes are located near the boundary of image, we can apply the iterative method described in Section 3.2 to obtain accurate global orientation. MultiBin loss consists of classification loss and cosine loss:

$$L_{multibin} = L_{conf} + \alpha_4 L_{cos}, \quad (5)$$

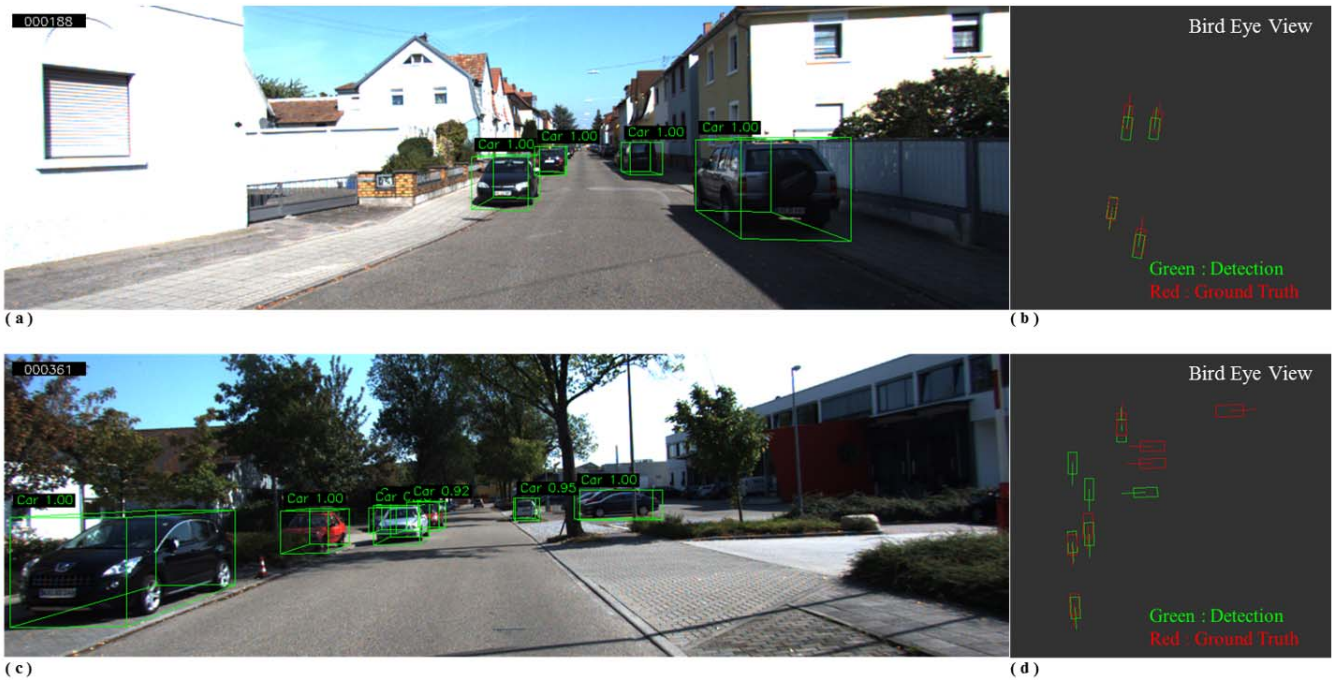

Figure 4. Reprojected box in bird-eye view map. (a) and (c) describe the projected 3D box in green obtained by point-to-side correspondence constraint. (b) and (d) describe the projected ground truth box in red and reprojected box in green.

where L_{conf} is the softmax loss and L_{cos} is the cosine loss. (We call localization loss of [15] cosine loss.) The first term estimates the probability that the local orientation lies inside each bin, and the second estimates corresponding residual offset from the center ray of the bin. As suggested in [15], we use bin size equal to 2. The 3D object height, width and length are also regressed using L_2 loss. As usual, the regression target for each dimension is the residual relative to the mean parameter value computed over the training dataset:

$$L_{dim} = \frac{1}{n}\sum(D^* - \bar{D} - \delta)^2, \tag{6}$$

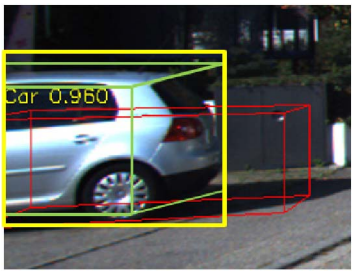

Figure 5. Truncated detection box example. Yellow line is the 2D detection box, and $x_{\min}$ is clipped to zero. Green line illustrates the reconstructed 3D box using the iterative method. Red line is the reconstructed 3D box obtained from all 2D box parameters. If we use the clipped value of $x_{\min} = 0$, the ray vector is $((0.5x_{\max} - c_x)/f_x, 1)$.

where D^* are the ground truth dimensions of the box, $\bar{D}$ are the mean dimensions of objects of a certain category.

In order to incorporate geometric properties into network, we regress to reprojected boxes obtained in 3D reconstruction layer. In particular, the regression target is the reprojected boxes in image and bird-eye view map. In contrast to other methods, we use novel multi-view reprojection (MVR) loss for these box regression defined by

$$L_{MVR} = L_{persp} + \alpha_5 L_{bev}. \tag{7}$$

We calculate perspective IoU loss with respect to the original 2D box for the reprojected box in image (see Figure 3):

$$L_{persp} = 1 - \text{IoU}(\text{true2Dbox}, \text{reprojected2Dbox}). \tag{8}$$

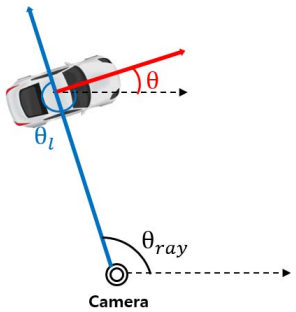

Figure 6. Relationship between global orientation and local orientation. Global orientation of the car θ is equal to $\theta_l + \theta_{ray}$.

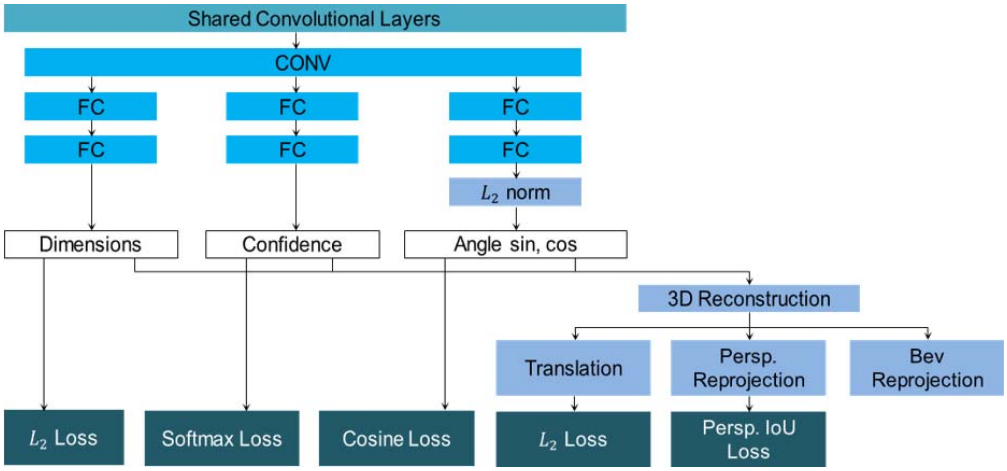

Figure 7. Proposed multi-view reprojection architecture.

The loss for the bird-eye view representation is reflected in L_2 loss for translation T of the recovered 3D box:

$$L_{bev} = \frac{1}{n}\sum(T^* - T)^2, \tag{9}$$

where T^* are ground truth 3D box bottom center locations. Here, the (x, z)-terms in T are associated with the center location of the reprojected box on the bird-eye view map, and the y-term is associated with the height of the box from ground. The L_{bev} adjusts the center of the box on the bird-eye view map while preventing from a fluctuation of the box's height location from ground. We note that (w, l)-terms of (6) and (x, z)-terms of (9) are associated with the regression of the *axis-aligned* bird-eye view representation. In particular, it amounts to a 2D box regression after rotating the estimated bird-eye view 2D box such that its yaw angle matches with the ground truth. During training, we also monitored the IoU score of the reprojected box on the bird-eye view map for checking the entire boundary of the box. The design choices of the proposed MVR loss are described in Section 4.3.

4. Experiments

4.1. Implementation details

Dataset. We performed our experiment on the KITTI dataset dedicated to autonomous driving [5]. This dataset consists of 7481 training set and 7518 test set. The calibration matrices are given. We evaluated the proposed approach on two different training/test splits. We used all available training images to report results on the official KITTI test set, and the results are illustrated in Section 4.2. Since the ground truth annotations for the test set are not available, we use a train/validation split in [3, 15, 30, 31] to compare our approach to other methods for tasks which are not evaluated on the KITTI benchmark. Evaluation on the validation set is described in Section 4.3.

Network training. We trained a Faster R-CNN [20] variant network, to produce 2D boxes and then estimated the dimensions and orientation. For regressing the angle and dimensions, we use pre-trained VGG16 [24] features up to conv5 layers followed by 1x1 convolutional layers for dimension reduction to half and add 3 branches as shown in Figure 7. In our parameter estimation module, the fully connected (FC) layers have 256 dimensions for dimension regression, and the other two branches have 128 dimensional FC layers. During training, each 2D bounding box crop is resized to 112x112, and the network is trained with stochastic gradient descent using a batch size of 8 and a fixed learning rate of 0.0001. The training is run for 200 epochs and the best model is chosen.

Data augmentation strategy. In order to make the network more robust to 2D detection, viewpoint changes and occlusions, we used a novel box augmentation technique by applying Section 3.1's point-to-side correspondence constraints to the ground truth information. To be specific, we use ground truth 2D box parameters, dimensions and orientation to obtain all possible 64 reprojected 2D boxes in image and pick the ones with sufficiently small reprojection errors; e.g., $1 - \text{IoU} \leq 0.03$. Of course, we adjust the ground truth local orientation of each generated box to account for the movement of the center ray of the crop. Figure 8 illustrates an example of our box augmentation method. In addition to the box augmentation, brightness augmentation and mirroring of images are applied.

4.2. KITTI test set evaluation

Evaluation metric. The official metric of KITTI 2D and 3D detection benchmark is the mean Average Precision (mAP) with overlapping criteria of 0.7 for vehicle. Also,

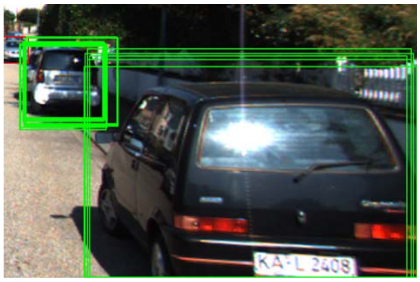

Figure 8. Box augmentation example. Ground truth 2D box, dimensions and global orientation are used to obtain all possible 64 reprojected 2D boxes in image and pick the ones with reprojection IoU with respect to the ground truth box is greater than 0.97.

the difficulty (easy, moderate and hard) is measured by the minimal pixel of 2D box's height and truncation and occlusion levels. To measure the orientation performance, the Average Orientation Similarity (AOS), which multiplies the AP of the 2D detector with the average cosine distance similarity, is used (see [5] for precise definition).

KITTI orientation accuracy. We are the first among all submissions for car examples on the KITTI leaderboard [1]. Our results of 2D detection and orientation estimation on the test set are summarized in Table 1. We can see that the proposed architecture outperforms all the methods on the orientation estimation for cars in the moderate setting. Although DeepMANTA [3] outperforms our method on the other settings, DeepMANTA requires complex preprocessing and heavily relies on templates of 3D models corresponding to several types of vehicles, making it hard to generalize to classes where such template does not exist. Moreover, compared to DeepStereoOP [17], which uses stereo information, or F-ConvNet [28], HRI-VoxelFPN [26] and MMLab-PointRCNN [22], which even use 3D information from LiDAR, our method is shown to be more effective. In terms of runtime, our approach is fastest among monocular state-of-the-arts methods. The runtime including 2D detector is 0.18 seconds per frame using Titan X and cuDNN v6 with Intel Xeon CPU E5-2620 v3 @ 2.40GHz, and the inference time of our proposed architecture except for the 2D detector's time is 0.02 seconds per frame, which is shown to be real time.

KITTI 3D detection accuracy. The 3D box from our 3D reconstruction layer is also evaluated on the KITTI 3D detection metric. Again, we use car class only. As we can see in Table 2, our method obtains top results on KITTI 3D obejct detection benchmark among monocular methods. AM3D [13], Mono3D PLiDAR [29], and MonoGRNet [19] outperform our methods. However, their inference engines are complicated because they rely on network for either depth estimation or pseudo-LiDAR point cloud. Our method is the third best among all monocular methods that do not use the synthetically generated features.

4.3. Analysis on KITTI validation set

Multi-view reprojection loss analysis. We performed an ablation study of the proposed multi-view reprojection loss on the KITTI cars in the moderate setting. (For comparison, we reproduce the baseline, which is [15].) Table 3 shows the effect of adding each loss on the KITTI validation set performance. Adding L_{persp} loss improves AOS by 0.41. We can see that AOS increases 0.59, and the best performance is achieved when we use the proposed multi-view reprojection loss. This verifies importance and effectiveness of incorporating 3D properties into the network architecture and our proposed training objective for better orientation estimation.

3D template analysis. We also evaluate 3D template prediction as our network training is based on 3D box reconstruction and our network outputs dimensions of the 3D box. We use the metric suggested by DeepMANTA [3] in order to measure the prediction performance. In particular, the 3D template prediction is evaluated by comparing the three predicted dimensions (w, h, l) to the ground truth 3D box dimensions (w_{gt}, h_{gt}, l_{gt}) provided by KITTI dataset. Given the correct 2D detection, the predicted value (w, h, l) is considered correct if $|(w_{gt} - w)/w_{gt}| < 0.2$, $|(h_{gt} - h)/h_{gt}| < 0.2$ and $|(l_{gt} - l)/l_{gt}| < 0.2$. Table 4 shows the performance comparison to other methods. In order to measure the performance of Deep3DBox [15], we use the estimated 3D boxes on the validation results available online at http://bit.ly/2oaiBgi. As shown in Table 4, our approach outperforms other methods in moderate and hard settings. Admittedly, DeepMANTA outperforms our method in easy setting. However our method does not rely on 3D CAD models in contrary to DeepMANTA.

Iterative orientation refinement analysis. We demonstrate the effect of iterative method for accurately estimating ray angle in case where detection boxes are truncated. We compare the proposed iterative method with baseline where the global orientation is obtained simply from the sum of local orientation and ray angle without adjustment. Here, we measure global orientation accuracy to evaluate the effectiveness of our method. We define a metric called Average Global Orientation Similarity (AGOS) where orientation in AOS is replaced with global orientation. Table 5 presents better performances for the proposed iterative method over baseline in all settings.

5. Conclusion

In summary, the main contributions of our paper include: 1) design of a flexible, highly accurate network architecture for orientation estimation incorporating geometric constraints and associated training strategy of the network. In contrast to the previous state-of-the-art methods, our ap-

Rank	Method	Sensor	Time(s)	Easy		Moderate		Hard	
				AOS(%)	AP(%)	AOS(%)	AP(%)	AOS(%)	AP(%)
1	**MVRA+I-FRCNN+**	mono	0.18	90.60	90.78	**89.93**	90.36	79.78	80.48
2	DeepMANTA [3]	mono	0.70	**97.19**	97.25	89.86	90.03	**80.39**	80.62
4	F-ConvNet [28]	mono, LiDAR	0.47	90.41	90.44	89.60	89.79	80.39	80.66
10	HRI-VoxelFPN [26]	LiDAR	0.02	90.43	90.66	89.27	89.89	80.31	80.97
11	MMLab-PointRCNN [22]	LiDAR	0.10	90.73	90.74	89.22	89.32	85.53	85.73
24	Deep3DBox [15]	mono	1.50	90.39	90.47	88.56	88.86	77.17	77.60
28	SubCNN [31]	mono	2.00	90.61	90.75	88.43	88.86	78.63	79.24
36	Shift R-CNN [16]	mono	0.25	90.27	90.56	87.91	88.90	78.72	79.86
37	MonoPSR [9]	mono	0.20	89.88	90.18	87.83	88.84	70.48	71.44
47	DeepStereoOP [17]	stereo	3.40	89.01	90.34	86.57	88.75	77.13	79.39
52	Mono3D [4]	mono	4.20	89.00	90.27	85.83	87.86	76.00	78.09

Table 1. Results for 2D vehicle detection (AP) and orientation (AOS) on the KITTI test set from leaderboard [1]

Method	Time(s)	Car AP(%)		
		Easy	Moderate	Hard
AM3D [13]	0.40	21.48	16.08	15.26
M3D-RPN [2]	0.16	20.65	15.70	13.32
MonoDIS [23]	0.10	11.81	15.12	12.71
Mono3D PLiDAR [29]	0.10	17.12	13.44	12.38
MonoGRNet [19]	0.04	11.29	12.90	11.34
MVRA+I-FRCNN+	0.18	12.92	11.01	10.45
MonoPSR [9]	0.20	12.57	10.85	9.06
ROI-10D [14]	0.20	12.30	10.30	9.39
SS3D [7]	0.048	11.74	9.58	7.7
GS3D [11]	2.00	7.69	6.29	6.16
Shift R-CNN [16]	0.25	8.13	5.22	4.78
OFT-Net [21]	0.50	3.28	2.50	2.27

Table 2. Results for 3D monocular vehicle detection (AP) on the KITTI test set from leaderboard [1]

Option	L_{multibin}	L_{dim}	L_{persp}	L_{bev}	AOS(%)
Baseline	✓	✓			95.47
Baseline+L_{persp}	✓	✓	✓		95.88
Ours	✓	✓	✓	✓	96.06

Table 3. Ablation study of adding losses from baseline on KITTI validation set

Template(%)	Easy	Moderate	Hard
DeepMANTA [3]	**94.04**	86.62	78.72
Deep3DBox [15]	86.45	86.55	75.28
Ours	88.97	**88.36**	**82.78**

Table 4. 3D template prediction evaluation on KITTI validation set

AGOS(%)	Easy	Moderate	Hard
Baseline	98.19	95.95	79.72
Ours	**98.23**	**96.07**	**79.86**

Table 5. Global orientation accuracy evaluation on KITTI validation set

proach is real time and does not require complex inference engines or 3D shape datasets. 2) A novel box augmentation technique based on projective geometry is proposed. 3) An experimental evaluation demonstrating the superiority of our approach for KITTI car orientation estimation and 3D detection is illustrated. 4) A thorough analysis of our proposed methods is provided.

References

[1] KITTI Object Detection Benchmark Leaderboard. http://www.cvlibs.net/datasets/kitti/eval_object.php. (Accessed on 13 August 2019). 7, 8

[2] Garrick Brazil and Xiaoming Liu. M3D-RPN: Monocular 3D region proposal network for object detection. In *Proceedings of the IEEE Conference on Computer Vision*. IEEE, 2019. 8

[3] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teulière, and Thierry Chateau. Deep MANTA: A

coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2040–2049, 2017. 3, 6, 7, 8

[4] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3D object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016. 2, 8

[5] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012. 1, 6, 7

[6] Tong He and Stefano Soatto. Mono3D++: Monocular 3D vehicle detection with two-scale 3D hypotheses and task priors. *CoRR*, abs/1901.03446, 2019. 3

[7] Eskil Jörgensen, Christopher Zach, and Fredrik Kahl. Monocular 3D object detection and box fitting trained end-to-end using intersection-over-union loss. *CoRR*, abs/1906.08070, 2019. 2, 8

[8] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navad. SSD-6D: Making RGB-based 3D detection and 6D pose estimation great agin. In *Proceedings of the IEEE Conference on Computer Vision*, pages 1530–1538, 2017. 2

[9] Jason Ku, Alex D. Pon, and Steven Lake Waslander. Monocular 3D object detection leveraging accurate proposals and shape reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11867–11876, 2019. 2, 8

[10] Abhijit Kundu, Yin Li, and James M. Rehg. 3D-RCNN: Instance-level 3D object reconstruction via render-and-compare. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3559–3568, 2018. 2

[11] Buyu Li, Wanli Ouyang, Lu Sheng, Xingyu Zeng, and Xiaogang Wang. GS3D: An efficient 3D object detection framework for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1019–1028, 2019. 2, 4, 8

[12] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In *Proceedings of the European Conference on Computer Vision*, pages 21–37, 2016. 2

[13] Xinzhu Ma, Zhihui Wang, Haojie Li, Wanli Ouyang, and Pengbo Zhang. Accurate monocular 3D object detection via color-embedded 3D reconstruction for autonomous driving. *CoRR*, abs/1903.11444, 2019. 2, 7, 8

[14] Fabian Manhardt, Wadim Kehl, and Adrien Gaidon. ROI-10D: Monocular lifting of 2D detection to 6D pose and metric shape. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2069–2078, 2019. 2, 8

[15] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Košecká. 3D bounding box estimation using deep learning and geometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5632–5640, 2017. 1, 2, 3, 4, 5, 6, 7, 8

[16] Andretti Naiden, Vlad Paunescu, Gyeongmo Kim, Byeong-Moon Jeon, and Marius Leordeanu. Shift R-CNN: Deep monocular 3D object detection with closed-form geometric constraints. *CoRR*, abs/1905.09970, 2019. 2, 4, 8

[17] Cuong Cao Pham and Jae Wook Jeon. Robust object proposals re-ranking for object detection in autonomous driving using convolutional neural networks. *Signal Processing: Image Communication*, 53:110–122, 2017. 7, 8

[18] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017. 2

[19] Zengyi Qin, Jinglu Wang, and Yan Lu. MonoGRNet: A geometric reasoning network for monocular 3D object localization. In *Proceedings of the thirty-third AAAI Conference on Artificial Intelligence*, pages 8852–8858, 2019. 2, 7, 8

[20] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. 2, 6

[21] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3D object detection. *CoRR*, abs/1811.08188, 2018. 2, 8

[22] Shaoshuai Shi, Xiaogang Wang, and Hongsheng Li. PointRCNN: 3D object proposal generation and detection from point cloud. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–779, 2019. 7, 8

[23] Andrea Simonelli, Samuel Rota Bulò, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3D object detection. *CoRR*, abs/1905.12365, 2019. 2, 8

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014. 6

[25] Burga Tekin, N. Sudipta Sinha, and Pascal Fua. Real-time seamless single shot 6D object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 292–301, 2018. 2

[26] Bei Wang, Jianping An, and Jiayan Cao. Voxel-FPN: Multi-scale voxel feature aggregation in 3D object detection from point clouds. *arXiv:1907.05286v2*, 2019. 7, 8

[27] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q. Weinberger. Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3D object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8445–8453, 2019. 2

[28] Zhixin Wang and Kui Jia. Frustum ConvNet: sliding frustums to aggregate local point-wise features for amodal 3D object detection. *arXiv:1903.01864v2*, 2019. 7, 8

[29] Xinshuo Weng and Kris Kitani. Monocular 3D object detection with pseudo-LiDAR point cloud. *CoRR*, abs/1903.09847, 2019. 2, 7, 8

[30] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Data-driven 3D voxel patterns for object category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1903–1911, 2015. 2, 6

[31] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Subcategory-aware convolutional neural networks for object proposals and detection. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, pages 924–933, 2017. 2, 6, 8

[32] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3D object detection from monocular images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2345–2353, 2018. 2

[33] Muhammad Zeeshan Zia, Michael Stark, and Konrad Schindler. Are cars just 3D boxes? Jointly estimating the 3D shape of multiple objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3678–3685, 2014. 2