Monocular 3D Object Detection via Geometric Reasoning on Keypoints

Ivan Barabanau[1], Alexey Artemov[1], Evgeny Burnaev[1] and Vyacheslav Murashkin[2]
[1]*Skolkovo Institute of Science and Technology, Moscow, Russia*
[2]*Yandex.Taxi, Moscow, Russia*

Keywords: Deep Learning, Monocular Vision, 3D Object Detection.

Abstract: Monocular 3D object detection is well-known to be a challenging vision task due to the loss of depth information; attempts to recover depth using separate image-only approaches lead to unstable and noisy depth estimates, harming 3D detections. In this paper, we propose a novel keypoint-based approach for 3D object detection and localization from a single RGB image. We build our multi-branch model around 2D keypoint detection in images and complement it with a conceptually simple geometric reasoning method. Our network performs in an end-to-end manner, simultaneously and interdependently estimating 2D characteristics, such as 2D bounding boxes, keypoints, and orientation, along with full 3D pose in the scene. We fuse the outputs of distinct branches, applying a reprojection consistency loss during training. The experimental evaluation on the challenging KITTI dataset benchmark demonstrates that our network achieves state-of-the-art results among other monocular 3D detectors.

1 INTRODUCTION

The success of autonomous robotics systems, such as self-driving cars, largely relies on their ability to operate in complex dynamic environments; as an essential requirement, autonomous systems must reliably identify and localize non-stationary and interacting objects, *e.g*. vehicles, obstacles, or humans. In its simplest formulation, localization is understood as an ability to detect and frame objects of interest in 3D bounding boxes, providing their 3D locations in the surrounding space. Crucial to the decision-making process is the accuracy of depth estimates of the 3D detections.

Depth estimation could be approached from both hardware and algorithmic perspectives. On the sensors end, laser scanners such as LiDAR devices have been extensively used to acquire depth measurements sufficient for 3D detection in many cases (Yang et al., 2018; Chen et al., 2017; Lang et al., 2019; Zhou and Tuzel, 2018; Ku et al., 2018; Yan et al., 2018). However, point clouds produced by these expensive sensors are sparse, noisy and massively increase memory footprints with millions of 3D points acquired per second. In contrast, image-based 3D detection methods offer savings on CPU and memory consumption, use cheap onboard cameras, and work with a wealth of established detection architectures (*e.g*., (Liu et al., 2016; Redmon et al., 2016; Ren et al., 2015; Lin et al., 2017a; He et al., 2017; Lin et al., 2017b; Liu et al., 2018; Simonelli et al., 2019)), yet they require sophisticated algorithms for depth estimation, as raw depth cannot be accessed anymore.

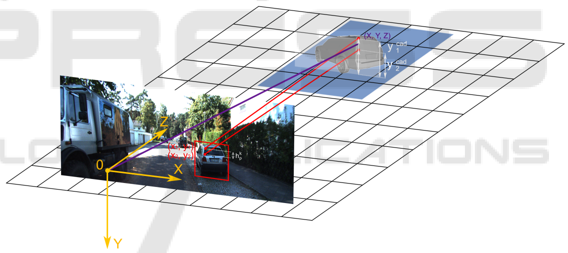

Figure 1: Geometric reasoning about instance depth (best viewed in zoom). We use camera intrinsic parameters, predict 2D keypoints and infer dimensions to "lift" the keypoints to 3D space.

Recent research on monocular 3D object detection relies on separate dense depth estimation models (Qin et al., 2019; Xu and Chen, 2018), but depth recovery from monocular images is naturally ill-posed, leading to unstable and noisy estimates. In addidion, in many practical instances, *e.g*., with sufficient target resolution or visibility, dense depth estimation might be redundant in context of 3D detection. Instead, one may focus on obtaining sparse but salient features, such as 2D keypoints, that are well-known visual cues often serving as geometric constrains in various vision tasks such as human pose estimation (Ramakrishna

Barabanau, I., Artemov, A., Burnaev, E. and Murashkin, V.
Monocular 3D Object Detection via Geometric Reasoning on Keypoints.
DOI: 10.5220/0009102506520659
In *Proceedings of the 15th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2020) - Volume 5: VISAPP*, pages 652-659
ISBN: 978-989-758-402-2; ISSN: 2184-4321
Copyright © 2022 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

et al., 2012; Martinez et al., 2017; Mehta et al., 2017a; Mehta et al., 2017b) and more general object interpretation (Hejrati and Ramanan, 2012; Wu et al., 2016).

Motivated by this observation, in this paper we propose a novel keypoint-based approach for 3D object detection and localization from a single RGB image. We build our model around 2D keypoint detection in images and complement it with a conceptually simple geometric reasoning framework, establishing correspondences between the detected 2D keypoints and their 3D counterparts defined on surfaces of 3D CAD models. The framework operates under the general assumptions, assuming the camera intrinsic parameters are given, and retrieves depth of closest keypoint instance to "lift" 2D keypoints to 3D space; the remaining 3D keypoints and the final 3D detection are assembled in a similar way. To enhance robustness of our 2D keypoint detection, we use a multi-task reprojection consistency loss; our model is ultimately end-to-end trainable. Additionally, our approach *does not* require images manually labelled with keypoints, but instead uses annotations automatically obtained from back-projecting data-aligned CAD models from 3D space to the image plane; we only mark 14 3D keypoints per each of 5 3D CAD models, totalling 70 manually labelled 3D keypoints, which makes our approach particularly labour-efficient in terms of annotation costs.

In summary, our contributions are as follows:

- We propose a novel deep learning-based framework for monocular 3D object detection, combining well-established region-based detectors and a geometric reasoning step over inferred keypoints.
- We create keypoint annotations for images in the KITTI 3D object detection benchmark dataset (Geiger et al., 2012; Geiger et al., 2013), using a collection of 3D CAD models, annotated with 3D keypoints. We note that the dataset is obtained without any human labour. We describe how to train our framework in an end-to-end fashion using the newly created annotations.

2 RELATED WORK

The most relevant to our work is research on monocular 3D object detection, that is well-known to be a challenging vision task. Deep3DBox (Mousavian et al., 2017) relies on a set of geometric constraints between 2D and predicted 3D bounding boxes and reduces 3D object localization problem to a linear system of equations, fitting 3D box projections into 2D detections. Their approach relies on a separate linear solver; in contrast, our model is end-to-end trainable and does not require external optimization. Mono3D (Chen et al., 2016) extensively samples 14K 3D bounding box proposals per image and evaluates each, exploiting semantic and image-based features. In contrast, our approach does not rely on an exhaustive sampling in 3D space, bypassing a significant computational overhead. OFT-Net (Roddick et al., 2018) introduces an orthographic feature transform which maps RGB image features into a birds-eye-view representation through a 3D scene grid, solving the perspective projection problem. However, back-projecting image features onto 3D grid results in a coarse feature assignment. Our approach detects 2D keypoints with sufficient precision, avoiding any additional discretization. MonoGRNet (Qin et al., 2019) directly deals with depth estimation from a single image, training an additional sub-network to predict the z-coordinate of each 3D bounding box. (Xu and Chen, 2018) exploit a similar approach, estimating disparity using a stand-alone pre-trained MonoDepth network (Godard et al., 2017). Both these methods rely on the non-trainable depth estimation networks, which introduce a computational overhead; in contrast to (Xu and Chen, 2018) and (Godard et al., 2017), our approach jointly estimates object 2D bounding-box and 3D pose in a fully trainable manner, not requiring a dense depth prediction and is similar in this respect to (Qin et al., 2019).

Perhaps, the most similar approach to ours is (Chabot et al., 2017), which utilizes 3D CAD models, along with predicting 2D keypoints. However, their network only models 2D geometric properties and aims at matching the predictions to one of the CAD shapes, while 3D pose estimation is postponed for the inference step. They additionally exploit extensive annotations of keypoints in their 3D models. In contrast, we only annotate 14 keypoints per each of the five 3D models and exploit them in a geometric reasoning module to bridge the gap between 2D and 3D worlds, which allows us to deal with 3D characteristics during training in an end-to-end manner.

3 3D OBJECT DETECTION FRAMEWORK

Given a single RGB image, our goal is to localize target objects in the 3D scene. Each target object is defined by its class and 3D bounding box, parameterized by 3D center coordinates $\mathbf{C} = (c_x, c_y, c_z)$ in a camera coordinate system, global orientation $\mathbf{R} = (\theta, \cdot, \cdot)$, and dimensions $\mathbf{D} = (w, h, l)$, standing for width, height and length, respectively.

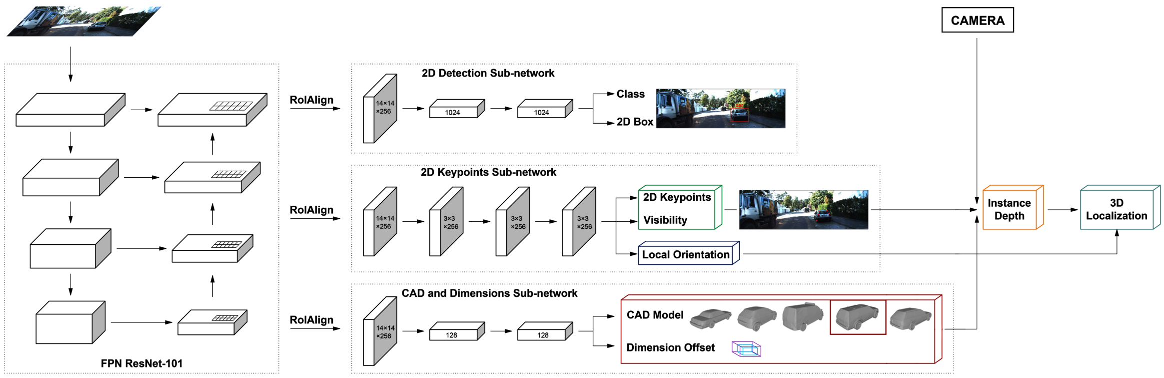

Figure 2: An overview of our monocular 3D detection architecture (text best viewed in zoom). We start with a universal backbone network (Mask R-CNN (He et al., 2017)) and complement it with three sub-networks: 2D object detection sub-network, 2D keypoints regression sub-network, and dimension regression sub-network. The network is trained end-to-end using a multi-task loss function.

2D Object Detection. For 2D detection, we follow the original Mask R-CNN architecture (He et al., 2017), which includes Feature Pyramid Network (FPN) (Lin et al., 2017a), Region Proposal Network (RPN) (Ren et al., 2015) and RoIAlign module (He et al., 2017). The RPN generates 2D anchor-boxes with a set of fixed aspect ratios and scales throughout the area of the provided feature maps, which are scored for the presence of the object of interest and adjusted. The spatial diversity of the proposed locations is processed by the RoIAlign block, converting each feature map, framed by the region of interest, into a fixed-size grid, preserving an accurate spatial location through bilinear interpolation. Followed by fully connected layers, the network splits into two feature sharing branches for the bounding box regression and object classification. During training, we utilize a sum of smooth L_1 and cross-entropy loss for each task respectively as our 2D detection loss $\mathcal{L}_{2\text{d}}$, as proposed by (Ren et al., 2015). Though we do not directly utilize the predicted 2D bounding boxes, we have experimentally observed the 2D detection sub-network to stabilize training.

2D Keypoint Detection. As described in greater detail in Section 4.1, we have created keypoint annotations for object instances encountered in the images of our training set (see Figure 3 and its caption for the details on our choice of the keypoints). Thus, using our 2D object detection sub-network, we learn to predict coordinates $k_{(x,y)} = (x_k, y_k)$ and a visibility state v_k for each of the manually-chosen 14 keypoints $\mathbf{K} = \{(x_k, y_k, v_k)_{k=1}^{14}\}$. Unlike suggested in (He et al., 2017; Tompson et al., 2014), we directly regress onto 2D coordinates of keypoints. The visibility state, determined by the occlusion and truncation of an instance, is a binary variable, and no difference between occluded, self-occluded and truncated states is made. Producing visibility estimates helps propagate information during training for visible keypoints only and acts as an auxiliary supervision for orientation sub-network. During training, similar to our 2D object detection sub-network, we minimize the multi-task loss combining smooth L_1 loss for coordinates regression and cross-entropy loss for visibility state classification, defined as:

$$
\begin{aligned}
\mathcal{L}_{\text{coord}} &= \sum_{k=1}^{K} \mathbb{1}_k \cdot L_1^{\text{smooth}}\left(k^{\text{gt}}_{(x,y)}, k^{\text{pred}}_{(x,y)}\right), \\
\mathcal{L}_{\text{vis}} &= -\sum_{k=1}^{K} v_k^{\text{gt}} \log v_k^{\text{pred}}, \\
\mathcal{L}_{\text{kp}} &= \mathcal{L}_{\text{coord}} + \mathcal{L}_{\text{vis}},
\end{aligned}
\tag{1}
$$

where $\mathbb{1}_k$ is the visibility indicator of k-th keypoint, while $k^{\text{gt}}_{(x,y)}$ and $k^{\text{pred}}_{(x,y)}$ denote ground-truth and predicted 2D coordinates, normalized and defined in a reference frame of a specific feature map post RoI-alignment. Similarly, v_k^{gt} is the ground truth visibility status, while v_k^{pred} is the estimated probability that keypoint k is visible.

3D Dimension Estimation and Geometric Classification. To each annotated 3D instance in the dataset, we have assigned a 3D CAD model out of a predefined set of 5 templates, obtaining 5 distinct geometric classes of instances. The assignment has been made based on the width, length and height ratios only. For each geometric class, we have computed mean dimensions (μ_w, μ_h, μ_l) over all assigned annotated 3D instances.

During training the 3D dimension estimation and geometric class selection sub-network, we utilize a multi-task loss combining cross-entropy loss (for the

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

geometric class selection) and a smooth L_1 loss for dimension regression. Instead of regressing the absolute dimensions, we predict the differences $\mathbf{D}_{\text{offset}} = (\Delta w, \Delta h, \Delta l) = (w - \mu_w, h - \mu_h, l - \mu_l)$ from the mean dimensions in the log-space:

$$\mathcal{L}_{\text{dim}} = \text{smooth}_{L_1}\left(\log(\mathbf{D}^{\text{gt}}_{\text{offset}} - \mathbf{D}^{\text{pred}}_{\text{offset}})\right), \quad (2)$$

where $\mathbf{D}^{\text{gt}}_{\text{offset}}$ and $\mathbf{D}^{\text{pred}}_{\text{offset}}$ represent the ground-truth and predicted offsets to the class mean values along each dimension, respectively.

Reasoning about Instance Depth. We define *instance depth* as the depth Z of a vertical plane passing through the two closest of visible keypoints, defined in the camera reference frame. To compute this depth value, we use predicted 2D keypoints (x_1, y_1) and (x_2, y_2), instance height h (in meters), and its geometric class. First, we select two keypoints (x_1, y_1) and (x_2, y_2) detected in the image and compute their y-difference $h_p = |y_1 - y_2|$. We then select the corresponding two keypoints $(x_1^{\text{cad}}, y_1^{\text{cad}})$ and $(x_2^{\text{cad}}, y_2^{\text{cad}})$ in 3D CAD model reference frame and compute their height ratio $r_{\text{cad}} = y_1^{\text{cad}}/y_2^{\text{cad}}$. Finally, the distance to the object Z is defined from the pinhole camera model:

$$Z = f \cdot \frac{r_{\text{cad}} \cdot h}{h_p}, \quad (3)$$

where f is a focal length of the camera, known for each frame. Figire 1 illustrates this computation. Depth coordinate Z allows to retrieve the remaining 3D location coordinates of one of the selected keypoints, using the back-projection mapping:

$$X = Z \cdot \frac{x_{\{1,2\}} - p_x}{f}, \quad Y = Z \cdot \frac{y_{\{1,2\}} - p_y}{f}, \quad (4)$$

giving predictions $k^{\text{pred}}_{(X,Y,Z)} = (X_k, Y_k, Z_k), k \in \{1, 2\}$, where (p_x, p_y) are the camera principal point coordinates in pixels and (x_1, y_1) and (x_2, y_2) are predicted 2D keypoints.

In this computation, we use the two closest of detected visible keypoints, which commonly results in selecting keypoints physically located at instance corners. We note that this step is crucial to the success of our method, as, if no keypoints are visible on the instance, we fail to recover its depth and precisely localize it (even though it may not be entirely occluded).

Orientation Estimation. Direct estimation of orientation $\mathbf{R}$ in a camera reference frame is not feasible, as the region proposal network propagates the context within the crops solely, cutting off the relation of the crop to the image plane. Inspired by (Mousavian et al., 2017), we represent the global orientation as a combination of two rotations with azimuths defined as:

$$\theta = \theta_{\text{loc}} + \theta_{\text{ray}}, \quad (5)$$

where θ_{loc} is the object's local orientation within the region of interest, and θ_{ray} is a ray direction from the camera to the object center, directly found from the 3D location coordinates. We estimate θ_{loc} using a modification of the MultiBin approach (Mousavian et al., 2017). Specifically, instead of splitting the objective into angle confidence and localization parts, we discretize the angle range from 0° to 360° degrees into 72 non-overlapping bins and compute the probability distribution over this set of angles by a softmax layer. We train the local orientation sub-network using cross-entropy as a loss function. To obtain the final prediction for θ_{loc}, we utilize the weighted mean of the bins medians ($WM(\theta_{\text{loc}})$), adopting the softmax output as the weights. Given 3D location coordinates (X, Z) of one of the keypoints and the weighted mean local orientation $WM(\theta_{\text{loc}})$, the global orientation is:

$$\theta = WM(\theta_{\text{loc}}) + \arctan\left(\frac{X}{Z}\right). \quad (6)$$

3D Object Detection. To obtain the center $\mathbf{C}$ of the final 3D bounding box, we use the global orientation $\mathbf{R}$ and the distance between the keypoint and the object center. For a particular CAD model, given the weight, height and length ratio between the selected keypoint and the object center $\mathbf{r}_{\text{cad}} = (x_1^{\text{cad}}/x_2^{\text{cad}}, y_1^{\text{cad}}/y_2^{\text{cad}}, z_1^{\text{cad}}/z_2^{\text{cad}})$ estimated object dimensions $\mathbf{D}$ and global orientation $\mathbf{R}$, the location $\mathbf{C}$ is predicted as

$$\mathbf{C} = (X, Y, Z) \pm \mathbf{R} \cdot \mathbf{D} \odot \mathbf{r}_{\text{cad}}, \quad (7)$$

where $\odot$ stands for an element-wise product. Depending on the selected keypoint position (left or right, etc.), a sign is chosen for each dimension.

Multi-head Reprojection Consistency Loss. One approach when training the system would be to optimize a sum of losses suffered by individual components:

$$\mathcal{L} = \mathcal{L}_{\text{2d}} + \mathcal{L}_{\text{kp}} + \mathcal{L}_{\text{dim}} + \mathcal{L}_{\text{ori}}. \quad (8)$$

However, while each sub-network is independent of its neighbors and unaware of other predictions, its geometric components are strongly interrelated. To provide consistency between the network branches, we introduce a loss function which integrates all the predictions. Predicted 3D keypoint coordinates $k^{\text{pred}}_{(X,Y,Z)}$ in a CAD model coordinate frame are scaled using $\mathbf{D}$, rotated using $\mathbf{R}$, translated using $\mathbf{C}$, and back-projected into the image plane via camera projection matrix to obtain 2D keypoint coordinates $k^{\text{pred+proj}}_{(x,y)} =$

Project$(k^{\text{pred}}_{(X,Y,Z)})$. A similar approach is applied to the eight corners $\mathbf{K}_c$ of the 3D bounding box obtained from 3D detection and orientation estimates, to ensure that they fit tightly into the ground truth 2D bounding box after back-projection. This leads us to adding the reprojection consistency loss $\mathcal{L}_{\text{repro}}$ to the final optimization:

$$\begin{aligned}\mathcal{L}_{\text{repro}} &= \sum_{k\in\mathbf{K}\cup\mathbf{K}_c} \text{smooth}_{L_1}\left(k^{\text{pred+proj}}_{(x,y)} - k^{\text{gt}}_{(x,y)}\right), \\ \mathcal{L} &= \mathcal{L}_{\text{2d}} + \mathcal{L}_{\text{kp}} + \mathcal{L}_{\text{dim}} + \mathcal{L}_{\text{ori}} + \mathcal{L}_{\text{repro}}.\end{aligned} \tag{9}$$

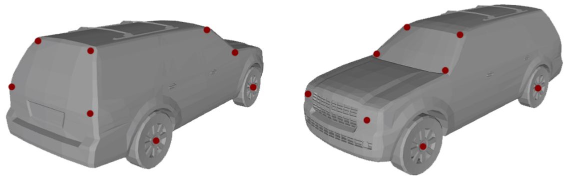

Figure 3: An example of our annotation of 3D keypoints. We label 4 centers of wheels, 8 corners of front and back windshields per each, and 2 centers of headlights. Per each 3D CAD model, we form the annotation so as to ensure that the two keypoints on each of the windshields form near-vertical lines.

4 EXPERIMENTS

4.1 Dataset Annotation

We train and evaluate our approach using the KITTI 3D object detection benchmark dataset, selecting only objects of class “car”. For the sake of comparison with state-of-the-art methods, we follow the setup presented in (Chen et al., 2016), which provides 3712 and 3769 images for training and validation, respectively, along with the camera calibration data. To extend KITTI dataset with assignment of geometric classes using CAD models and keypoints 2D coordinates, we employ the approach and data provided in (Xiang et al., 2015). Depending on the ratios between height, length and width, each car instance is associated with one of five 3D CAD model classes from a predefined set of CAD templates, presented on Figure 2. We manually annotated each CAD model with 14 keypoint locations, ending up with a total human labour of 70 annotations in the entire dataset. Figure 3 displays an example of the annotated keypoints, most of which are a common choice (Everingham et al., 2010) due to their interpretability, such as the car's edges, carcass, etc.; we also included corners of windshields to deal with the height of each instance. To obtain 2D coordinates of the keypoints, we back-projected CAD models from 3D space to the image plane using ground truth location, dimension and rotation values. Simultaneous projection of all 3D CAD models on a scene provides us with a depth ordering mask, allowing for defining the visibility state of each keypoint.

4.2 Experimental Setup

Network Architecture. We utilize Mask R-CNN with a Feature Pyramid Network (Lin et al., 2017a), based on a ResNet-101 (He et al., 2016) as our backbone network for the multi-level feature extraction. Instead of the higher resolution 14×14 and 28×28 feature maps in the original architecture, we stack the same amount of 3×3 kernels, followed by a fully connected layer to predict 2D normalized coordinates and visibility states for each of the 14 keypoints. From the same feature maps, we branch a fully-connected layer predicting local orientation in bins of 5° each, totaling 72 output units. The feature sharing between keypoints and local orientation was found crucial for network performance, as both characteristics imply similar geometric reasoning. In parallel to 2D detection and 2D keypoints estimation, we create a sub-network of a similar architecture for dimension regression and classification into geometric classes. The remaining components, including RPN, RoIAlign, bounding box regression and classification heads, are implemented following the original Mask R-CNN design. For instance depth retrieval we use only four pairs of keypoints: corners of the front and rear windows. Other keypoints are used for additional supervision in consistency loss calculation during training.

Model Optimization Details. We set hyperparameters following Mask R-CNN work (He et al., 2017). The RPN anchor set covers five scales, adjusting them to the values of 4, 8, 16, 32, 64, and three default aspect ratios. Each mini-batch consists of 2 images, producing 256 regions of interest, with a positive to negative samples ratio set 1:3, to achieve class sampling balance during training. Any geometric augmentations over the images are omitted, solely applying image padding to meet the network architecture requirements. ResNet-101 is initialized with the weights pre-trained on Imagenet (Deng et al., 2009), and frozen during further training steps. We first train the 2D detection and classification sub-network for 100K iterations, adopting Adam optimizer with a learning rate of 10^{-4} throughout the training, setting weight decay of 0.001 and momentum of 0.9. Then 2D keypoints and local orientation are trained for 50K iterations. Finally, enabling the multi-head consistency loss, the whole network is trained in an end-to-end fashion for 50K iterations. We combine

Table 1: **3D detection performance:** Average Precision of 3D bounding boxes on KITTI val set. The best score is in bold, the second best underlined. Ours (+loss) indicates a base network setup, trained with a consistency reprojection loss.

	IoU = 0.5			IoU = 0.7		
Method	Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D (Chen et al., 2016)	25.19	18.20	15.52	2.53	2.31	2.31
OFT-Net (Roddick et al., 2018)	-	-	-	4.07	3.27	3.29
MonoGRNet (Qin et al., 2019)	50.51	**36.97**	**30.82**	13.88	10.19	7.62
MF3D (Xu and Chen, 2018)	47.88	29.48	26.44	10.53	5.69	5.39
MonoDIS (Simonelli et al., 2019)	-	-	-	**18.05**	**14.98**	**13.42**
Ours	48.81	30.17	20.07	11.91	6.64	4.28
Ours (+loss)	**50.82**	31.28	20.21	13.96	7.37	4.54

losses from all of the network outputs, weighting them equally.

Evaluation Metrics. We evaluate the network under the conventional KITTI benchmark protocol, which enables comparison across approaches. Car category is the sole subject of our focus. By default, KITTI settings require evaluation in 3 regimes: easy, moderate and hard, depending on the instance difficulty of a potential detection. 3D bounding box detection performance implies 3D Average Precision (AP3D) evaluation, setting Intersection over Union (IoU) threshold to 0.5 and 0.7.

4.3 Experimental Results

3D Object Detection. We compare the performance with 5 monocular 3D object detection methods: Mono3D (Chen et al., 2016), OFT-Net (Roddick et al., 2018), MonoGRNet (Qin et al., 2019), MF3D (Xu and Chen, 2018), and MonoDIS (Simonelli et al., 2019), which reported their results on the same validation set for the *car* class. We borrow the presented average precision numbers from their published results. The results are reported in Table 1. The experiments show that our approach outperforms state-of-the-art methods on the easy subset by a small margin while remaining the second best on the moderate subset. This observation aligns with our intuition that visible salient features such as keypoints are crucial to the success of 3D pose estimation. For the moderate and hard images, 2D keypoints are challenging to robustly detect due to the high occlusion level or the low resolution of the instance. We also measure the effect of the reprojection consistency loss on our network performance, observing a positive effect of our loss function.

3D Bounding Box and Global Orientation Estimation. We follow the experiment presented in (Qin et al., 2019), evaluating the quality of the 3D bounding boxes sizes estimation, as well as the orientation in a camera coordinate system. We present our mean errors along with those of (Qin et al., 2019; Chen et al., 2016), borrowed from their work, in Table 2.

Table 2: **3D bounding box and orientation mean errors:** The best score is in bold, the second best underlined.

	Size (m)			
Method	Height	Width	Length	Ori. (rad)
Mono3D	0.172	0.103	0.504	0.558
MonoGRNet	**0.084**	**0.084**	0.412	0.31
Ours	0.115	0.107	0.516	0.215
Ours (+loss)	0.101	0.091	**0.403**	**0.191**

Though, the sizes of the 3D bounding boxes do not differ severely among the approaches, due to the estimating the offset from the median bounding box, the orientation estimation results differ significantly. Since we retrieve global orientation via geometric reasoning, learning local orientation from 2D image features, the network provides more accurate predictions, in contrast to obtaining orientation from the regressed 3D bounding box corners.

Qualitative Results. We provide a qualitative illustration of the network performance in Figure 4, displaying six road scenes with distinct levels of difficulty. In typical cases, our approach produces accurate 3D bounding boxes for all instances, along with the global orientation and 3D location. Remarkably, the truncated objects can also be successfully detected, given that only one pair of keypoints hits the image. Some hard cases, i.e. (e) and (f), primarily consist of objects that are distant, highly occluded or even invisible on the image. We believe such failure cases to be a common limitation of monocular image processing methods.

5 CONCLUSIONS

In this work, we presented a novel deep learning-based framework for monocular 3D object detection

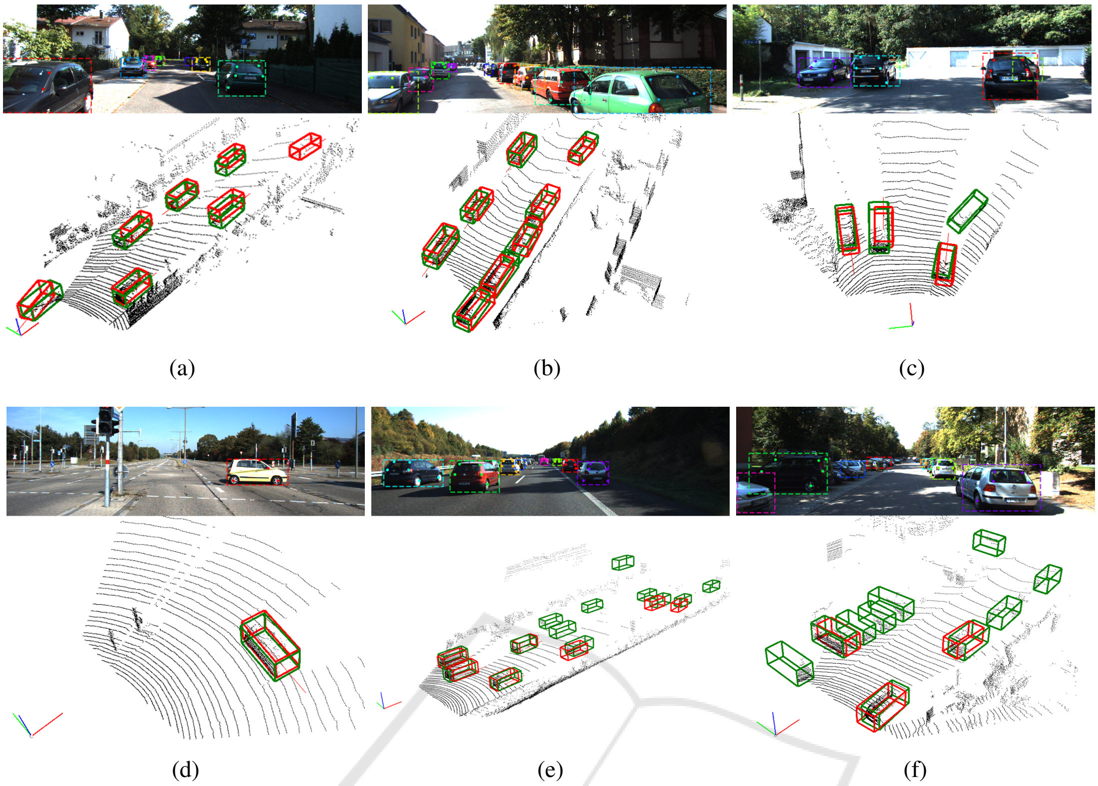

Figure 4: Qualitative results. The upper part of each sub-figure contains 2D detection inference, including 2D bounding and 2D locations of the visible keypoints. Each instance and its keypoints are displayed their distinctive color. The lower part visualizes the 3D point cloud, showing the camera location as the colored XYZ axes. Green and red colors stand for the ground truth and predicted 3D bounding boxes respectively. The scenes were selected to express diversity in complexity and cars positioning w.r.t. the camera.

SCIENCE AND TECHNOLOGY PUBLICATIONS

combining well-known detectors with geometric reasoning on keypoints. We proposed to estimate correspondences between the detected 2D keypoints and their 3D counterparts annotated on the surface of 3D CAD models to solve the object localization problem. Results of the experimental evaluation of our approach on the subsets of the KITTI 3D object detection benchmark demonstrate that it outperforms the competing state-of-the-art approaches when the target objects are clearly visible, leading us to hypothesize that dense depth estimation is redundant for 3D detection in some instances. We have demonstrated our multi-task reprojection consistency loss to significantly improve performance, in particular, the orientation of detections.

ACKNOWLEDGEMENT

E. Burnaev and A. Artemov were supported by the Russian Science Foundation Grant 19-41-04109.

REFERENCES

Chabot, F., Chaouch, M., Rabarisoa, J., Teulière, C., and Chateau, T. (2017). Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image. In *CVPR*, pages 2040–2049.

Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., and Urtasun, R. (2016). Monocular 3d object detection for autonomous driving. In *CVPR*, pages 2147–2156.

Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In *CVPR*, pages 1907–1915.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. Ieee.

Everingham, M., Van Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2010). The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *IJRR*, 32(11):1231–1237.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, pages 3354–3361. IEEE.

Godard, C., Mac Aodha, O., and Brostow, G. J. (2017). Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, pages 270–279.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *ICCV*, pages 2961–2969.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *CVPR*, pages 770–778.

Hejrati, M. and Ramanan, D. (2012). Analyzing 3d objects in cluttered images. In *NIPS*, pages 593–601.

Ku, J., Mozifian, M., Lee, J., Harakeh, A., and Waslander, S. L. (2018). Joint 3d proposal generation and object detection from view aggregation. In *IROS*, pages 1–8. IEEE.

Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., and Beijbom, O. (2019). Pointpillars: Fast encoders for object detection from point clouds. In *CVPR*, pages 12697–12705.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., and Belongie, S. (2017a). Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017b). Focal loss for dense object detection. In *ICCV*, pages 2980–2988.

Liu, S., Qi, L., Qin, H., Shi, J., and Jia, J. (2018). Path aggregation network for instance segmentation. In *CVPR*, pages 8759–8768.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. C. (2016). Ssd: Single shot multibox detector. In *ECCV*, pages 21–37. Springer.

Martinez, J., Hossain, R., Romero, J., and Little, J. J. (2017). A simple yet effective baseline for 3d human pose estimation. In *ICCV*, pages 2640–2649.

Mehta, D., Rhodin, H., Casas, D., Fua, P., Sotnychenko, O., Xu, W., and Theobalt, C. (2017a). Monocular 3d human pose estimation in the wild using improved cnn supervision. In *3DV*, pages 506–516. IEEE.

Mehta, D., Sridhar, S., Sotnychenko, O., Rhodin, H., Shafiei, M., Seidel, H.-P., Xu, W., Casas, D., and Theobalt, C. (2017b). Vnect: Real-time 3d human pose estimation with a single rgb camera. *ACM ToG*, 36(4):44.

Mousavian, A., Anguelov, D., Flynn, J., and Kosecka, J. (2017). 3d bounding box estimation using deep learning and geometry. In *CVPR*, pages 7074–7082.

Qin, Z., Wang, J., and Lu, Y. (2019). Monogrnet: A geometric reasoning network for monocular 3d object localization. In *AAAI*, volume 33, pages 8851–8858.

Ramakrishna, V., Kanade, T., and Sheikh, Y. (2012). Reconstructing 3d human pose from 2d image landmarks. In *ECCV*, pages 573–586. Springer.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*, pages 91–99.

Roddick, T., Kendall, A., and Cipolla, R. (2018). Orthographic feature transform for monocular 3d object detection. *arXiv preprint arXiv:1811.08188*.

Simonelli, A., Bulò, S. R. R., Porzi, L., López-Antequera, M., and Kontschieder, P. (2019). Disentangling monocular 3d object detection. *arXiv preprint arXiv:1905.12365*.

Tompson, J. J., Jain, A., LeCun, Y., and Bregler, C. (2014). Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, pages 1799–1807.

Wu, J., Xue, T., Lim, J. J., Tian, Y., Tenenbaum, J. B., Torralba, A., and Freeman, W. T. (2016). Single image 3d interpreter network. In *ECCV*, pages 365–382. Springer.

Xiang, Y., Choi, W., Lin, Y., and Savarese, S. (2015). Data-driven 3d voxel patterns for object category recognition. In *ICCV*.

Xu, B. and Chen, Z. (2018). Multi-level fusion based 3d object detection from monocular images. In *CVPR*, pages 2345–2353.

Yan, Y., Mao, Y., and Li, B. (2018). Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337.

Yang, B., Luo, W., and Urtasun, R. (2018). Pixor: Real-time 3d object detection from point clouds. In *CVPR*, pages 7652–7660.

Zhou, Y. and Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In *CVPR*, pages 4490–4499.

SCIENCE AND TECHNOLOGY PUBLICATIONS