ROI-10D: Monocular Lifting of 2D Detection to 6D Pose and Metric Shape

Fabian Manhardt*
TU Munich
fabian.manhardt@tum.de

Wadim Kehl*
Toyota Research Institute
wadim.kehl@tri.global

Adrien Gaidon
Toyota Research Institute
adrien.gaidon@tri.global

Abstract

We present a deep learning method for end-to-end monocular 3D object detection and metric shape retrieval. We propose a novel loss formulation by lifting 2D detection, orientation, and scale estimation into 3D space. Instead of optimizing these quantities separately, the 3D instantiation allows to properly measure the metric misalignment of boxes. We experimentally show that our 10D lifting of sparse 2D Regions of Interests (RoIs) achieves great results both for 6D pose and recovery of the textured metric geometry of instances. This further enables 3D synthetic data augmentation via inpainting recovered meshes directly onto the 2D scenes. We evaluate on KITTI3D against other strong monocular methods and demonstrate that our approach doubles the AP on the 3D pose metrics on the official test set, defining the new state of the art.

1. Introduction

How much can one understand a scene from a single color image? Using large annotated datasets and deep neural networks, the Computer Vision community has steadily pushed the envelope of what was thought possible, not just for semantic understanding but also in terms of 3D properties of scenes and objects. In particular, Deep learning methods on monocular imagery have proven competitive with multi-sensor approaches for important ill-posed inverse problems like 3D object detection ([3, 31, 20, 34, 24], 6D pose tracking [30, 40], depth prediction [9, 11, 13, 42, 33], or shape recovery [18, 23]. These improvements have been mainly accomplished by incorporating strong implicit or explicit priors that regularize the underconstrained output space towards geometrically-coherent solutions. Furthermore, these models benefit directly from being end-to-end trainable in general. This leads to increased accuracy, since networks are discriminatively tuned towards the target objective instead of intermediate outputs followed by non-trainable post-processing heuristics. The main challenge,

---

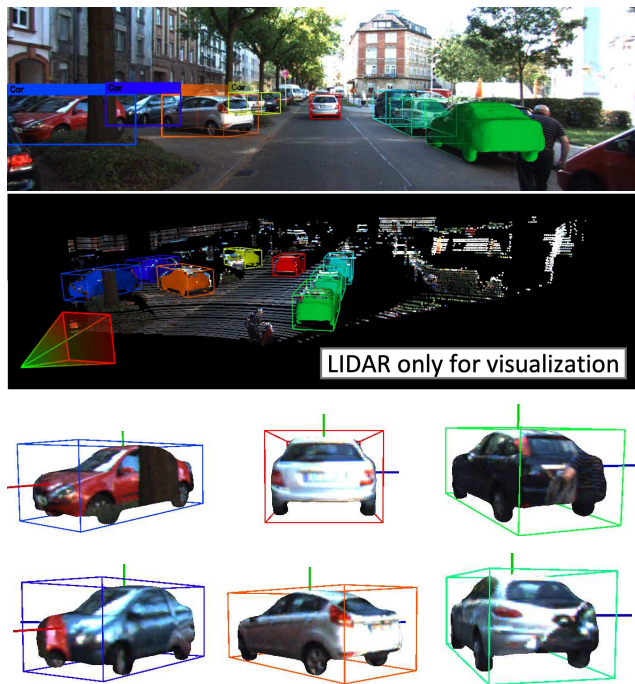* Equal contribution. This work was part of an internship stay at TRI.

Figure 1. Top (from left to right): our 2D detections, 3D boxes, and meshed shapes inferred from a single monocular image in one forward pass. Middle: our predictions on top of a LIDAR point cloud, demonstrating metric accuracy. Bottom: example well-localized, metrically-accurate, textured meshes predicted by our network.

though, is to design a model and differentiable loss function that lend itself to well-behaved minimization.

In this work we introduce a new end-to-end method for metrically accurate monocular 3D object detection, *i.e.* the task of predicting the location and extent of objects in 3D using a single RGB image as input. Our key idea is to regress oriented 3D bounding boxes by lifting predicted 2D Regions of Interest (RoIs) using a monocular depth network. Our main contributions are:

- an end-to-end multi-scale deep network for monocular 3D object detection, including a differentiable 2D to 3D RoI lifting map that internally regresses all required components for 3D box instantiation;

- a loss function that aligns those 3D boxes in metric space, directly minimizing their error with respect to ground truth 3D boxes;

- an extension of our model to predict metric textured meshes, enabling further 3D reasoning, including 3D-coherent synthetic data augmentation.

We call our method "ROI-10D", as it lifts 2D regions of interests to 3D for prediction of 6 degrees of freedom pose (rotation and translation), 3 DoF spatial extents, and 1 or more DoF shape. Experiments on the KITTI3D [12] benchmarks show that our approach enables accurate predictions from a single RGB image. Furthermore, we show that our monocular 3D poses are competitive or better than the state of the art.

2. Related Work

Since the amount of work on object detection has expanded significantly over the last years, we will narrow our focus to recent advances among RGB-based methods for 3D object detection. 3DOP from Chen *et al.* [4] use KITTI [12] stereo data and additional scene priors to create 3D object proposals followed by a CNN-based scoring. In their follow-up work Mono3D [3], the authors replace the stereo-based priors by exploiting various monocular counterparts such as shape, segmentation, location, and spatial context. Mousavian *et al.* [31] propose to couple single-shot 2D detection with an additional binning of azimuth orientations plus offset regression. Similarly, SSD-6D from Kehl *et al.* [20] introduces a structured discretization of the full rotational space for single-shot 6D pose estimation. The work from Xu *et al.* [41] incorporates a monocular depth module to further boost the accuracy of inferred poses on KITTI.

Instead of discretizing $SO(3)$, [34, 37] formulate the 6D estimation problem as a regression of the 2D projections of the 3D bounding box. These methods assume the scale of the objects to be known and can therefore use a perspective-n-point (PnP) variant to recover poses from 2D-3D correspondences. Grabner *et al.* [14] present a mixed approach where they regress 2D control points and absolute scale to recover pose and, subsequently, the object category. In addition, Rad *et al.* [34] empirically show the superiority of this proxy loss over standard regression of the 6 degrees of freedom. In contrast, [40, 24, 30] directly encode the 6D pose. In particular, Xiang *et al.* [40] first regress the rotation as Euler angles and the 3D translation as the backprojected 2D centroid. Thereafter, they transform the 3D mesh into the camera frame and measure the average distance of the model points [16] towards the ground truth. Similarly, [24] also minimizes the average distance of model points for 6D pose refinement. Manhardt *et al.* [30] also conduct 6D pose refinement but regress a 4D update quaternion to describe the 3D rotation. Their proxy loss samples and transforms

3D contour points to maximize projective alignment.

Notably, all these direct encoding methods require knowledge of the precise 3D model. However, when working at a category-level the 3D models are usually not available, and these approaches are not designed to handle intra-class 3D shape variations (for instance between different types of cars). We therefore propose a more robust way of lifting to 3D that only requires bounding boxes. Thereby, the extents of these bounding boxes can also be of variable size. Similar to us, [7] use RoIs to lift 2D detections, but their pipeline is not trained end-to-end and reliant on RGB-D input for 3D box instantiation.

In terms of monocular shape recovery, 3D-RCNN from Kundu *et al.* [23] uses an RPN to estimate the orientation and shape of cars on KITTI with a render-and-compare loss. Kanazawa *et al.* [18] predict instance shape, texture, and camera pose using a differentiable mesh renderer [19]. While these methods show very impressive results as part of their synthesis error minimization, they recover shapes only up to scale. Furthermore, our approach does not require differentiable rendering or approximations thereof.

3. Monocular lifting to 10D for pose and shape

In this section we describe our method of detecting objects in 2D space and consequently, computing their 6D pose and metric shape from a single monocular image. First, we give an overview of our network architecture. Second, we explain how we lift the loss computation to 3D in order to improve pose accuracy. Third, we describe our learned metric shape space and its use for 3D reconstruction from estimated shape parameters. Finally, we describe how our shape estimation enables 3D-coherent data augmentation to improve detection.

3.1. End-to-end Monocular Architecture

Our architecture (Figure 2) follows a two-stage approach, similar to Faster R-CNN [36], where we first produce 2D region proposals and then run subsequent predictions for each. For the first stage we employ a RetinaNet [26] that uses a ResNet-34 backbone with FPN structure [25] and focal loss weighting. For each detected and precise 2D object proposal, we then use the RoIAlign operation [15] to extract localized features for each region.

In contrast to the aforementioned related works, we do not directly regress 3D information independently for each proposal from these localized features. Predicting this information from monocular data, in particular absolute translation, is ill-posed due to scale and reprojection ambiguities, which the lack of context exacerbates. In contrast, networks that aim to predict global depth information over the whole scene can overcome these ambiguities by leveraging geometric constraints as supervision [11]. Consequently, we use a parallel stream based on the state-of-the-art Su-
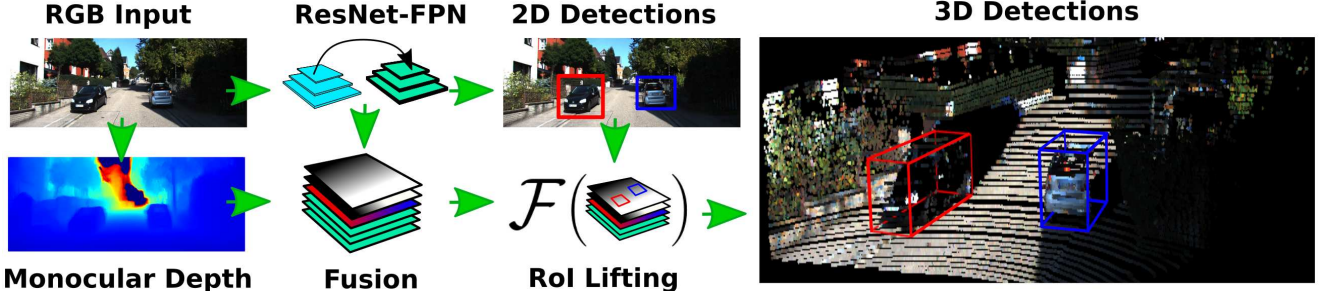
Figure 2. We process our input image with a ResNet-FPN architecture for 2D detection and a monocular depth prediction network. We use the predicted Regions of Interest (RoI) to extract fused feature maps from the ResNet-FPN and depth network via a RoIAlign operation before regressing 3D bounding boxes, a process we call RoI lifting.

perDepth network [33], which predicts per-pixel depth from the same monocular image.

We use these predicted depth maps to support distance reasoning in the subsequent 3D lifting part of our network. Besides the aforementioned localized feature maps from our 2D RPN, we also want to include the corresponding regions in the predicted depth map. For better localization accuracy, we furthermore decided to include a 2D coordinates map [27]. We thus propagate all the information to our fusion module, which consists of two convolutional layers with Group Normalization [39] for each input modality. Finally, we concatenate all features, use RoIAlign and run into separate branches for the regression of 3D rotation, translation, absolute (metric) extents, and object shape, as described in the following sections.

3.2. From Monocular 2D Instance to 6D Pose

Formally, our approach towards the problem is to define a fully-differentiable lifting mapping $\mathcal{F} : \mathbb{R}^4 \to \mathbb{R}^{8 \times 3}$ from a 2D RoI $\mathcal{X}$ to a 3D box $\mathcal{B} := \{B_1, ..., B_8\}$ of eight ordered 3D points. We chose to encode the rotation as a 4D quaternion and the translation as the projective 2D object centroid (similar to [31, 20, 40]) together with the associated depth. In addition, we describe the 3D extents as the deviation from the mean extents over the whole data set.

Given RoI $\mathcal{X}$, our lifting $\mathcal{F}(\mathcal{X})$ runs RoIAlign at that position, followed by separate prediction heads to recover rotation q, RoI-relative 2D centroid (x, y), depth z and metric extents (w, h, l). From this we build the 8 corners B_i:

$$B_i := q \cdot \begin{pmatrix} \pm w/2 \\ \pm h/2 \\ \pm l/2 \end{pmatrix} \cdot q^{-1} + K^{-1} \begin{pmatrix} x \cdot z \\ y \cdot z \\ z \end{pmatrix} \quad (1)$$

with K^{-1} being the inverse camera intrinsics. We build the points B_i in a defined order to preserve absolute orientation. We depict the instantiation in Figure 3.

Our formulation is reminiscent of 3D anchoring (as MV3D [5], AVOD [22]). However, our 2D instantiation of those 3D anchors is sparse and works over the whole im-
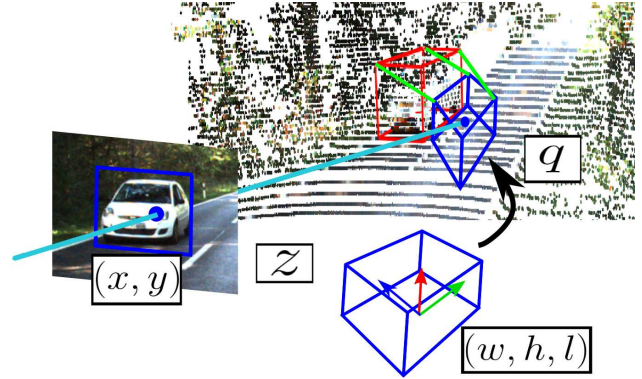

Figure 3. Our lifting $\mathcal{F}$ regresses all components to estimate a 3D box $\mathcal{B}$ (blue). From here, our loss minimizes the pointwise distances towards the ground truth $\mathcal{B}^*$ (red). We visualize three of the eight correspondences in green.

age plane. While such 3D anchors explicitly provide the object's 3D location, our additional degree of freedom also requires the estimation of the depth.

Lifting Pose Error Estimation to 3D When estimating the pose from monocular data only, little deviations in pixel-space can induce big errors in 3D. Additionally, penalizing each term individually can lead to volatile optimization and is prone to suboptimal local minima. We propose to lift the problem to 3D and employ a proxy loss describing the full 6D pose. Consequently, we do not force to optimize all terms equally at the same time, but let the network decide its focus during training. Given a ground truth 3D box $\mathcal{B}^* := \{B_1^*, ..., B_8^*\}$ and its associated 2D detection $\mathcal{X}$ in the image, we run our lifting map to retrieve the 3D prediction $\mathcal{F}(\mathcal{X}) = \mathcal{B}$. The loss itself is the mean over the eight corner distances in metric space:

$$\mathcal{L}(\mathcal{F}(\mathcal{X}), \mathcal{B}^*) = \frac{1}{8} \sum_{i \in \{1..8\}} ||\mathcal{F}(\mathcal{X})_i - \mathcal{B}^*_i||. \quad (2)$$

We depict some of the 3D-3D correspondences that the loss is aligning as green lines in Figure 3.
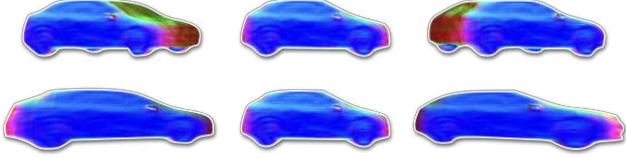
Figure 4. Comparison between egocentric (top) and allocentric (bottom) poses. While egocentric poses undergo viewpoint changes towards the camera when translated, allocentric poses always exhibit the same view, independent of the object's location.

When deriving the loss, the chain rule leads to

$$\left[\frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla q}, \frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla(x,y)}, \frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla z}, \frac{\nabla\mathcal{F}(\mathcal{X})}{\nabla(w,l,h)}\right] \frac{\nabla\mathcal{L}(\cdot)}{\nabla\mathcal{F}(\mathcal{X})} \mathcal{L}(\cdot) \tag{3}$$

and shows clearly the individual impact that each lifting component contributes towards 3D alignment. Similar to work that employ projective or geometric constraints [29, 30], we observe that we require a warm-up period to bring regression into proper numerical regimes. We therefore train with separate terms until we reach a stable 3D box instantiation and switch then to our lifting loss.

We also want to stress that our parametrization allows for general 6D pose regression. Although the object annotations in KITTI3D exhibit only changing azimuths, many driving scenarios and most robotic use cases require solving for all 6 degrees of freedom.

Allocentric Regression and Egocentric Lifting Multiple works [31, 23] emphasize the importance of estimating the allocentric pose for monocular data, especially for larger fields of view. The difference is depicted in Figure 4 where the relative object translation with respect to the camera changes the observed viewpoint. Accordingly, we follow the same principle since RoIs lose the global context. Therefore, rotations q are considered allocentric during regression inside $\mathcal{F}$ and then corrected with the inferred translation to build the egocentric 3D boxes.

3.3. Object Shape Learning & Retrieval

In this section we explain how we extend our end-to-end monocular 3D object detection method to additionally predict meshes and how to use them for data augmentation.

Learning of a Smooth Shape Space Given a set of 50 commercially available CAD models of cars, we created projective truncated signed distances fields (TSDF) ϕ_i of size $128 \times 128 \times 256$. We initially used PCA to learn a low-dimensional shape, similar to [23]. During experimentation we found the shape space to be quickly discontinuous away from the mean, inducing degenerated meshes. Using PCA to generate proper shapes requires to evaluate each dimension according to its standard deviation. To avoid this
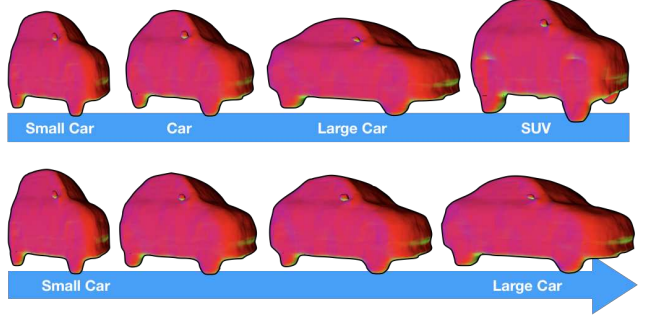

Figure 5. Top: Median of each category in the learned shape space. Bottom: Smooth interpolation on the latent hypersphere between two categories.

tedious process, we instead trained a 3D convolutional autoencoder, consisting of encoder E as well as decoder $\mathcal{D}$, and enforced different constraints on the output TSDF. In particular, we employed 4 convolutional layers with filter sizes of 1,8,16,32 for both E and $\mathcal{D}$. In addition, we used a fully-connected layer of 6 to represent the latent space. During training we further map all latent representations on the unit hypersphere to ensure smoothness within the embedding. Furthermore, we penalize jumps in the output level set via total variation, which regularizes towards smoother surfaces. The final loss is the sum of all these components:

$$\mathcal{L}_{tsdf}(E, \mathcal{D}, \phi) = |\mathcal{D}(E(\phi)) - \phi| + |(||E(\phi)|| - 1)| + |\nabla\mathcal{D}(E(\phi))| \tag{4}$$

We additionally classified each CAD model as either 'Small Car', 'Car', 'Large Car' or 'SUV'. Afterwards, we computed the median shape over each class, and all cars together, using the Weiszfeld algorithm [38], as illustrated in Fig. 5 (top). Below, we show our ability to smoothly interpolate between the median shapes in the embedding. We observed that we could safely traverse all intermediate points on the embedding without degenerate shapes and found a six-dimensionsal latent space to be a good compromise between smoothness and detail.

Ground truth shape annotation. To avoid gradient approximation through central differences as [23], we labeled the KITTI3D car instances offline. Running greedy search initialized from every median, we seek for the minimal projective discrepancy in LIDAR and segmentation from [15].

For the shape branch of our 3D lifter, we measure the similarity between predicted shape s and ground truth shape s^* as the angle between the two points on the hypersphere.

$$\mathcal{L}_{shape}(s, s^*) = \arccos\left(2\langle s, s^*\rangle^2 - 1\right) \tag{5}$$

During inference we predict the low-dimensional latent vector and feed it to the decoder to obtain its TSDF representation. We can also compute the 3D mesh from the TSDF employing marching cubes [28].

Simple mesh texturing. Since our method computes absolute scale and 6D pose, we conduct projective texturing of the retrieved 3D mesh. To this end, we project each vertex that faces towards the camera onto the image plane and assign the corresponding pixel value. Afterwards, we mirror the colors along the symmetry axis for completion.

3.4. Synthetic 3D data augmentation

Since annotating monocular data with metrically accurate 3D annotations is usually costly and difficult, many recent works leverage synthetic data [10, 8, 2, 1] to train their methods [20, 17, 35]. Nevertheless, this often comes with a significant drop in performance due to the domain gap. This is especially true for KITTI3D, since it is a very small dataset with only around 7k images (or 3.5k images for train and val respectively with the split from [3]). This can easily result in severe overfitting to the training data distribution.

An interesting solution to this domain gap, proposed by Alhaija et al. [1], consists in extending the dataset by inpainting 3D synthetic renderings of objects onto real-world image backgrounds. Inspired by this Augmented Reality type of approach, we propose to utilize our previously extracted meshes in order to produce realistic renderings. This allows for increased realism and diversity, in contrast to using a small set of fixed CAD models as in [1]. Furthermore, we do not use strong manual or map priors to place the synthetic objects in the scene. Instead, we employ the allocentric pose to move the object in 3D without changing the viewpoint. We apply some rotational perturbations in 3D to generate new unseen poses and decrease overfitting. Fig. 6 illustrates one synthetically generated training sample. While the red bounding boxes show the original ground truth annotations, the green bounding boxes depict the synthetically added cars and their sampled 6D pose.

3.5. Implementation details

The method was implemented in PyTorch [32] and we employed AWS p3.16xlarge instances for training. We used SGD with momentum, a batch size of 8 and a learning rate of 0.001 with linear warm-up. We ran a total of 200k iterations and decayed the learning rate after 120k and 180k steps by 0.1. We employed both scale-jittering and horizontal flipping to augment the dataset. For the synthetic car augmentations, we extracted in total 140 meshes from the training sequences, which we textured using the corresponding ground truth poses. We then augmented each input sample with up to 3 different cars by shooting rays in random directions and sampling a 3D translation along the ray. Additionally, we employed the original allocentric rotation to avoid textural artifacts, however, perturbed the rotations up to 10 degrees in order to always produce new unseen 6D poses. Our shape space is six-dimensional although smaller dimensionality can lead to well-behaving
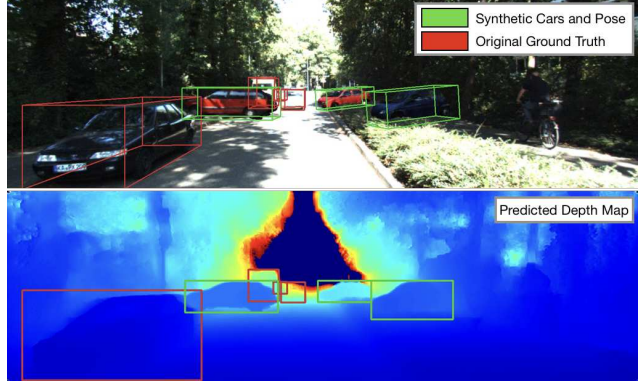

Figure 6. Synthetically generated training sample. Top: Green bounding boxes show original ground truth cars and poses. In contrast, red boxes illustrate the rendered meshes from a sampled 6D pose. Bottom: Augmented depth map from SuperDepth [33]. Notice that we utilized the annotated meshes, which we colored using the ground truth pose and our projective texturing.

spaces, too. We show qualitative results in the supplement. During testing, we resize the shorter side of the image to 600 and run 2D detection. We filter the detections with 2D-NMS at 0.65 before RoI-lifting. The resulting 3D boxes are then processed by a very strict Bird's Eye View-NMS at 0.05 that prevents physical intersections.

4. Evaluation

In this section, we describe our evaluation protocol, compare to the state of the art for RGB-based approaches, and provide an ablative analysis discussing the merits of our individual contributions.

4.1. Evaluation Protocol

We use the standard KITTI3D benchmark [12] and its official evaluation metrics. We evaluate our method on three different difficulties: easy, moderate, hard. Furthermore, as suggested we also set the IoU threshold to 0.7 for both 2D and 3D. For the pose, we compute the average precision (AP) in the Bird's eye view, which measures the overlap of the 3D bounding boxes projected on the ground plane. We also compute the AP for the full 3D bounding box.

4.2. Comparison to Related Work

We compare ourselves on the train/validation split from [3] and on the official test set against state-of-the-art RGB-based methods on KITTI3D, namely (stereo-based) 3DOP [4], Mono3D [3], and Xu *et al.* [41] which also uses a depth module for better reasoning. Note that, although slightly lower in 2D AP, our model using synthetic data provides the best pose accuracy among our trained networks and we chose this model to compete against the others. As can be seen in Table 1 and 2, our method performs worse in 2D

Method	Type	Bird Eye View AP [val / test]			3D Detection AP [val / test]		
		Easy	Moderate	Hard	Easy	Moderate	Hard
Mono3D [3]	Mono	5.22 / –	5.19 / –	4.13 / –	2.53 / –	2.31 / –	2.31 / –
3DOP [4]	Stereo	12.63 / –	9.49 / –	7.59 / –	6.55 / –	5.07 / –	4.10 / –
Xu *et al.* [41]	Mono	**22.03** / 13.73	**13.63** / 9.62	**11.60** / 8.22	**10.53** / 7.08	5.69 / 5.18	5.39 / 4.68
ROI-10D	Mono	10.74 / –	7.46 / –	7.06 / –	7.79 / –	5.16 / –	3.95 / –
ROI-10D (Syn.)	Mono	14.50 / **16.77**	9.91 / **12.40**	8.73 / **11.39**	9.61 / **12.30**	**6.63** / 10.30	**6.29** / 9.39

Table 1. 3D detection performance on KITTI3D validation [3] and official KITTI3D test set. We report our AP for Bird's eye view and 3D IoU at the official IoU threshold of 0.7 for each metric. Note that we only evaluated the synthetic ROI-10D version on the online test set.

Method	2D Detection AP [val /test]		
	Easy	Moderate	Hard
Mono3D [3]	**93.89** / 92.33	**88.67** / **88.66**	**79.68** / 78.96
3DOP [4]	93.08 / **93.04**	88.07 / 88.64	79.39 / **79.10**
Xu *et al.* [41]	– / 90.43	– / 87.33	– / 76.78
ROI-10D	89.04 / –	88.39 / –	78.77 / –
ROI-10D (Syn.)	85.32 / 75.33	77.32 / 69.64	69.70 /61.18

Table 2. 2D AP performance on KITTI3D validation [3] and official test set at official IoU threshold of 0.7.

due to our strict 3D-NMS, but we are by far the strongest in the Bird's Eye View and the 3D AP. This underlines the important aspect of of proper data analysis to counteract overfitting. On the official test set, we get around twice the 3D AP of our closest monocular competitor. It is noteworthy that [41] trained their depth module on both KITTI3D and Cityscapes [6] for better generalization whereas the SuperDepth model we use has been pre-trained on KITTI data only. Interestingly, they have a strong drop in numbers when moving from the validation set onto the test set (e.g. from 22.03% to 13.73% or 10.53% to 7.08%), which suggests aggressive tuning towards the validation set with known ground truth. We want to mention that the evaluation protocol forces the 3D AP and Bird's eye view AP to be bounded from above by the 2D detection AP since missed detections in 2D always reflect negatively on the pose metrics. This strengthens our case further since our pose metric numbers would be even higher if we were to correct them with a 2D AP normalization.

4.3. Ablative Analysis

In the ablative analysis we want to first investigate how our new loss specifically minimizes the alignment problem. Additionally, we will identify where and why certain poses in KITTI3D are so much more difficult to estimate right. Finally, we analyze our method in respect to different inputs and how well our loss affects the quality of the poses.

Lifting Loss   We run a controlled experiment where, isolating one instance with ground truth RoI $\mathcal{X}$ and 3D box B^*, we solely optimize the lifting module $\mathcal{F}$ with randomly initialized parameters. The step-wise improvement in alignment between $\mathcal{F}(\mathcal{X})$ and B^* is depicted in Figure 7 and we refer to the supplementary material for the full animations. Independent of initialization, we can observe that our loss always converges smoothly towards the global optimum. We also show the magnitude of each Jacobian component from Eq. 3 and can see that the loss focuses strongly on depth while steadily increasing importance towards rotation and 2D centroid position. Since our scale regression recovers deviation from the average car size, it was mostly neglected during optimization since the original error in extents was minimal. Without manually enforcing any principal direction during optimization or scaling the magnitudes, the loss steers the impact of each component quite well.

Pose Recall vs. Training Data   To better understand our strengths and weaknesses, in Fig. 8 we show our recall for different bins for depth and rotation on the train/val split from [3]. We accept a detection if the Bird's Eye View IoU is larger than 0.5. Note that we followed the KITTI convention, such that an angle of 0 degrees corresponds to an object facing to the right. Since the dataset is rather small for deep learning methods, we also plot the training data distribution to understand if there is a correlation between sample frequency and pose quality.

For translation we did not discover any connection between the number of occurrences in the training data and the pose results. Nevertheless, closer objects are in general significantly better localized in 3D than objects further away. This can be explained by the fact that the network strongly relies on the predicted depth map to estimate the distance. However, the uncertainty of our monocular depth estimation also grows with distance. Very interestingly, utilizing our synthetic data generation improves the results across all bins. This confirms that, since the variety of scenes is limited, the network learns biases quickly and risks over-fitting without our proposed augmentation.

Our synthetic approach also clearly leads to better rotation estimates. In contrast to translation, we can find a strong correlation between the training data distribution and pose quality. While our method achieves good results on frequent viewpoints, the recall naturally drops when objects are seen from an underrepresented angle.

Method	2D Detection AP [0.7]			Bird's Eye View AP [0.5 / 0.7]			3D Detection AP [0.5 / 0.7]		
	Easy	Moderate	Hard	Easy	Moderate	Hard	Easy	Moderate	Hard
No Weighting	88.95	87.54	78.68	40.17 / 11.85	27.85 / 7.32	24.49 / 7.22	33.95 / 7.47	22.53 / 4.83	21.78 / 3.76
Multi-Task Weighting [21]	88.20	83.81	74.87	36.22 / 10.00	26.82 / 6.60	23.02 / 5.84	31.40 / 6.70	21.04 / 4.64	17.32 / 3.63
ROI-10D (w/o depth)	78.57	73.44	63.69	36.21 / 14.04	24.90 / 3.69	21.03 / 3.56	29.38/ **10.12**	19.80 / 1.76	18.04 / 1.30
ROI-10D	**89.04**	**88.39**	**78.77**	42.65 / 10.74	29.80 / 7.46	25.03 / 7.06	36.25 / 7.79	23.00 / 5.16	**22.06** / 3.95
ROI-10D (Syn.)	85.32	77.32	69.70	**46.85 / 14.50**	**34.05 / 9.91**	**30.46 / 8.73**	**37.59** / 9.61	**25.14 / 6.63**	21.83 / **6.29**

Table 3. Different weighting strategies and input modalities on the train/validation split from [3]. We report our AP referring to 2D detection, the bird's eye view challenge and 3D IoU. Besides the official IoU threshold of 0.7, we also report for a softer threshold of 0.5.
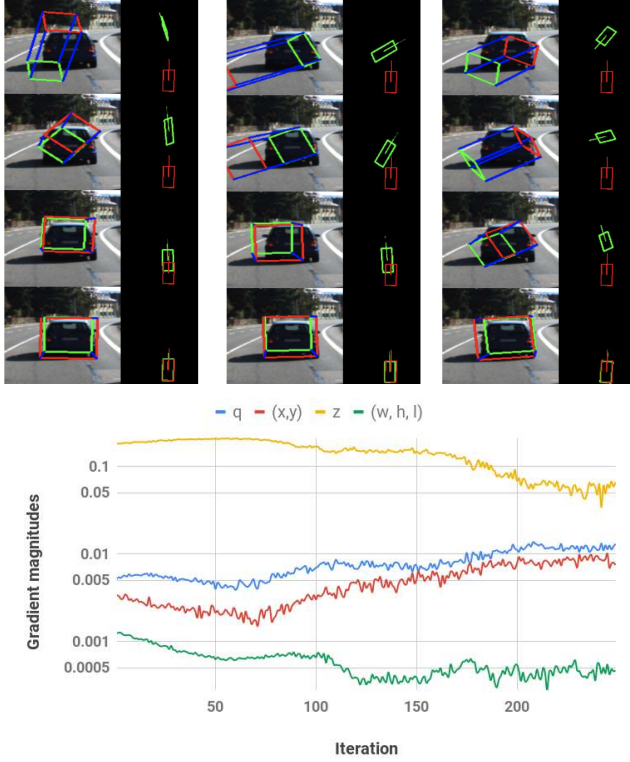

Figure 7. Controlled lifting loss experiment with given 2D RoI $\mathcal{X}$ over multiple runs with different seeding. Top: Visualizing $\mathcal{F}(\mathcal{X})$ during optimization in camera and bird's eye view. Bottom: Gradient magnitudes of each lifting component, averaged over all runs. We refer to the supplement for the full animations.
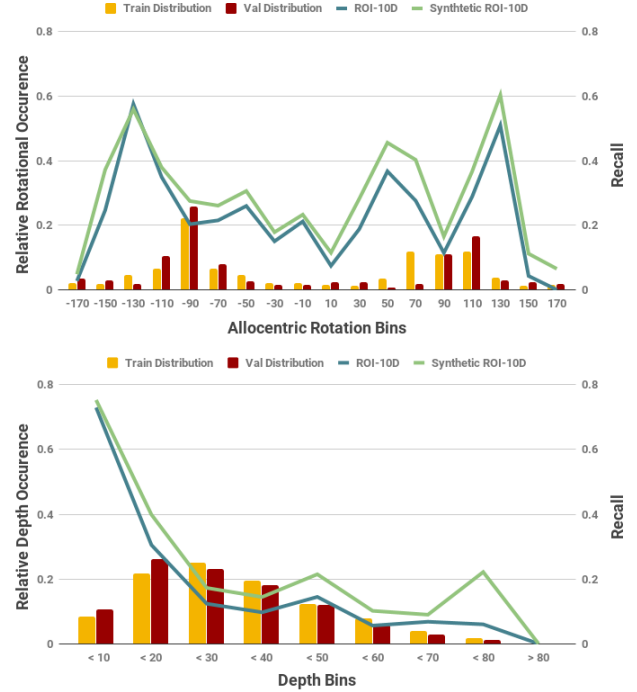

Figure 8. Recall of orientation and depth against the ground truth split distributions. Evidently, there exists a strong correlation between model performance and sample distribution. Synthetically augmenting underrepresented bins leads to overall better results.

Loss and input data We trained networks with different loss and data configurations on the train/validation [3] split to incrementally highlight our contributions. In the first two rows of Table 3 we ran training with separate regression terms instead of our lifting loss. While the first row shows the results with uniform weighting of all terms of $\mathcal{F}$ (similar to the approach of Xu *et al.* [41]), the second row shows training with the adaptive multi-task weighting from Kendall *et al.* [21]. Interestingly, we were not able to see an improvement with the adaptive weighting. We believe it comes from the fact that each term's magnitude is not at all comparable: while the (x, y) centroid moves in RoI-normalized image coordinates, the depth z is metric, the ex-

tents (w, h, l) are multiples of standard deviation from the mean extent, and the rotation q moves on a 4D unit sphere. Any uninformed weighting about the actual 3D instantiation has no means to properly assess the relative importance apart from numerical magnitude, thus comparing apples to oranges. Our formulation (row 4) avoids these problems and is either equal or better across all metrics.

Table 3 also presents results of a trained variant without monocular depth (row 3) and results for our method using depth without (rows 4) and with (row 5) synthetic augmentation. The results without depth cues are clearly worse, but we nonetheless get respectable numbers for the Bird's eye view and 3D AP. Unfortunately, our aggressive 3D-NMS discarded some correct solutions because of wrongly-regressed overlapping z-values, reducing our 2D AP significantly. Our synthetic data training shows strong im-
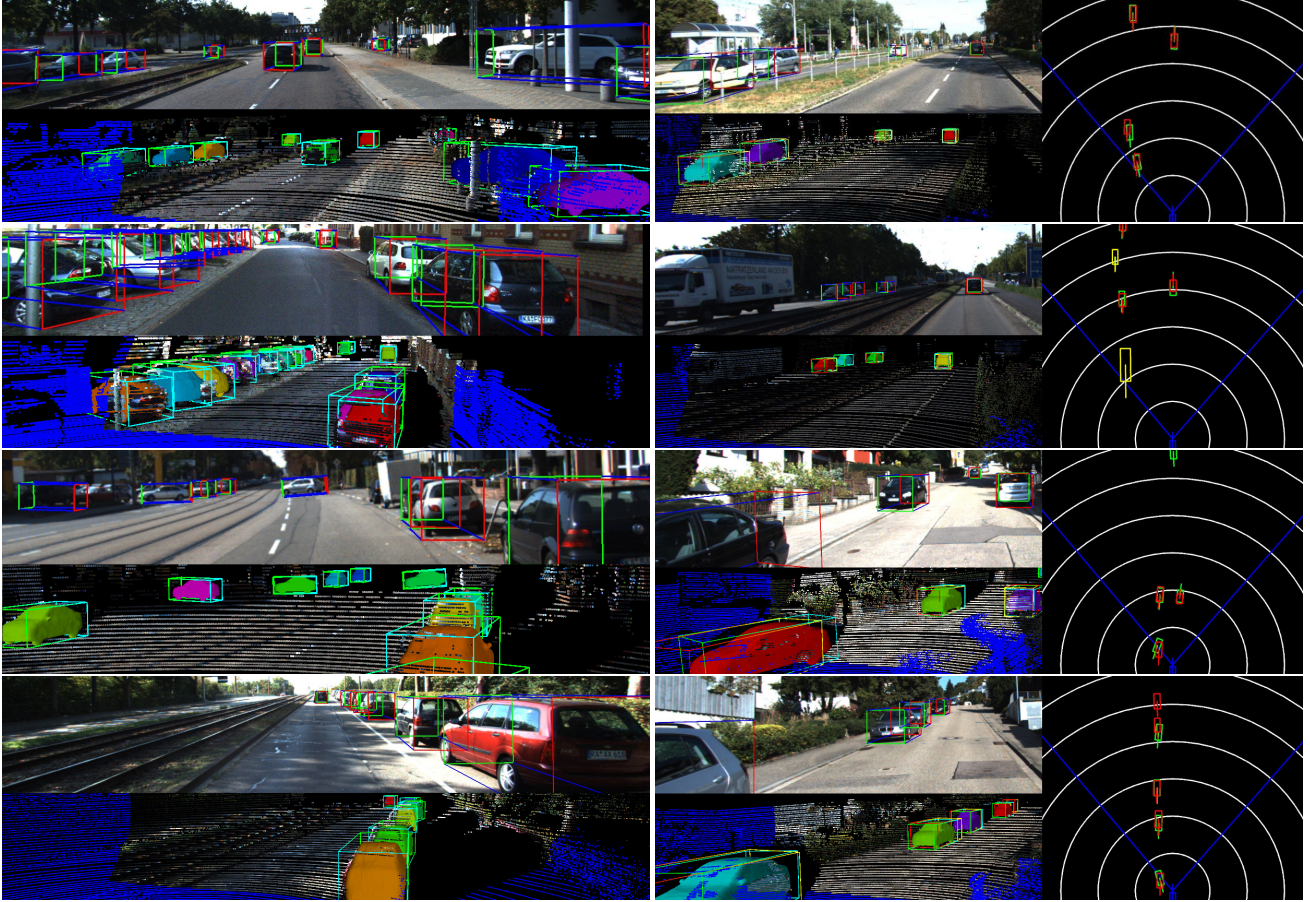
Figure 9. Qualitative results on the test (left) and validation (right) set. Noteworthy, we only trained on the train split to ensure that we never saw any of these images. For the validation samples, we additionally depict the ground truth poses in red. To get a proper estimate of the accuracy of the poses, we also plot the Bird's eye view (right) where we show clearly that we can recover accurate poses and proper metric shapes for unseen data, even at a distance.

provement on the pose metrics since we reduced the rotational data sample imbalance. By inspecting the drop in 2D AP, we realized that we designed our augmentations to be occlusion-free to avoid unrealistic intersections with the environment. In turn, this led to a weaker representation of strongly-occluded instances and to another introduced bias. We also show some qualitative results in Figure 9.

5. Conclusion

We proposed a monocular deep network that can lift 2D detections in 3D for metrically accurate pose estimation and shape recovery, optimizing directly a novel 3D loss formulation. We found that maximizing 3D alignment end-to-end for 6D pose estimation leads to very good results since we optimize for exactly the quantity we seek. We provided some insightful analysis on pose distributions in KITTI3D and how to leverage this information with recovered meshes for synthetic data augmentation. We found this reflection to be very helpful and quite important to improve the pose re-

call. Non-maximum-suppression in 2D and 3D is, however, a major influence on the final results and should to be addressed in future work, too.

References

[1] Hassan Abu Alhaija, Siva Karthik Mustikovela, Lars Mescheder, Andreas Geiger, and Carsten Rother. Augmented reality meets computer vision: Efficient data generation for urban driving scenes. *International Journal of Computer Vision*, 126(9):961–972, 2018.

[2] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *CVPR*, pages 95–104, 2017.

[3] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3d object detection for autonomous driving. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[4] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3d object proposals for accurate object class detection. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS, pages 424–432, Cambridge, MA, USA, 2015. MIT Press.

[5] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *IEEE CVPR*, 2017.

[6] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] Zhuo Deng and Longin Jan Latecki. Amodal detection of 3d objects: Inferring 3d bounding boxes from 2d ones in rgb-depth images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[8] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[9] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014.

[10] A Gaidon, Q Wang, Y Cabon, and E Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016.

[11] Ravi Garg, Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European Conference on Computer Vision*, pages 740–756. Springer, 2016.

[12] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[13] Clément Godard, Oisin Mac Aodha, and Gabriel J Brostow. Unsupervised monocular depth estimation with left-right consistency. In *CVPR*, volume 2, page 7, 2017.

[14] Alexander Grabner, Peter M. Roth, and Vincent Lepetit. 3D Pose Estimation and 3D Model Retrieval for Objects in the Wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017.

[16] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, , and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*, 2012.

[17] Stefan Hinterstoisser, Vincent Lepetit, Paul Wohlhart, and Kurt Konolige. On pre-trained image features and synthetic images for deep learning. *CoRR*, abs/1710.10710, 2017.

[18] Angjoo Kanazawa, Shubham Tulsiani, Alexei A. Efros, and Jitendra Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018.

[19] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3d mesh renderer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3907–3916, 2018.

[20] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[21] Alex Kendall, Yarin Gal, and Roberto Cipolla. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[22] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven Waslander. Joint 3d proposal generation and object detection from view aggregation. *IROS*, 2018.

[23] Abhijit Kundu, Yin Li, and James M. Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[24] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[25] Tsung-Yi Lin, Piotr Dollár, Ross B Girshick, Kaiming He, Bharath Hariharan, and Serge J Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017.

[26] Tsung-Yi Lin, Priyal Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[27] Rosanne Liu, Joel Lehman, Piero Molino, Felipe Petroski Such, Eric Frank, Alex Sergeev, and Jason Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *CoRR*, abs/1807.03247, 2018.

[28] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 163–169, New York, NY, USA, 1987. ACM.

[29] Reza Mahjourian, Martin Wicke, and Anelia Angelova. Unsupervised learning of depth and egomotion from monocular video using 3d geometric constraints. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[30] Fabian Manhardt, Wadim Kehl, Nassir Navab, and Federico Tombari. Deep model-based 6d pose refinement in rgb. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[31] Arsalan Mousavian, Dragomir Anguelov, John Flynn, and Jana Kosecka. 3d bounding box estimation using deep learning and geometry. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5632–5640, 2017.

[32] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Al-

ban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[33] Sudeep Pillai, Rares Ambrus, and Adrien Gaidon. Superdepth: Self-supervised, super-resolved monocular depth estimation, 2018.

[34] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[35] Mahdi Rad, Markus Oberweger, and Vincent Lepetit. Feature mapping for learning fast and accurate 3d pose inference from synthetic images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Neural Information Processing Systems (NIPS)*, 2015.

[37] Bugra Tekin, Sudipta N. Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[38] Endre Weiszfeld. Sur le point pour lequel la somme des distances de n points donnés est minimum. *Tohoku Mathematical Journal, First Series*, 43:355–386, 1937.

[39] Yuxin Wu and Kaiming He. Group normalization. In *CVPR*, 2018.

[40] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *Robotics: Science and Systems (RSS)*, 2018.

[41] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3d object detection from monocular images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[42] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, volume 2, page 7, 2017.