# CPS: Class-level 6D Pose and Shape Estimation From Monocular Images

Fabian Manhardt[1]    Manuel Nickel[1]    Sven Meier[2]
Luca Minciullo[2]    Nassir Navab[1]

[1]Technical University of Munich    [2]Toyota Motor Europe

**Abstract.** Contemporary monocular 6D pose estimation methods can only cope with a handful of object instances. This naturally limits possible applications as, for instance, robots need to work with hundreds of different objects in a real environment. In this paper, we propose the first deep learning approach for class-wise monocular 6D pose estimation, coupled with metric shape retrieval. We propose a new loss formulation which directly optimizes over all parameters, *i.e.* 3D orientation, translation, scale and shape at the same time. Instead of decoupling each parameter, we transform the regressed shape, in the form of a point cloud, to 3D and directly measure its metric misalignment. We experimentally demonstrate that we can retrieve precise metric point clouds from a single image, which can also be further processed for *e.g.* subsequent rendering. Moreover, we show that our new 3D point cloud loss outperforms all baselines and gives overall good results despite the inherent ambiguity due to monocular data.

## 1  Introduction

The field of 2D object detection has made huge leaps forward with the advent of deep learning. Current 2D detectors can robustly detect more than a hundred different object classes in real-time [39,24,9]. Progress in the field of 6D pose estimation, however, is still limited due to the higher complexity of this task.

The difficulty of 6D pose estimation is also compounded with the much higher effort in annotating datasets. As a consequence, many approaches rely on synthetic training data. This, however, introduces an additional domain gap, which has a detrimental impact on performance [29]. Furthermore, many factors such as clutter and occlusion [35], ambiguities [26], and lighting changes [38] can significantly deteriorate performance.

Finally, the ability to estimate 6D poses of unseen objects is currently poor and often ignored in performance reports. Existing methods typically train separate networks for each object instance, which is slow and inflexible [13,45].

Recently, a few approaches extended 3D object detection to object classes. These typically aim for automotive contexts for detection and estimation of 3D bounding boxes for vehicles and pedestrians in outdoor environments [3,4]. In contrast to us, these methods usually focus on large objects, *e.g.* cars and pedestrians, leverage stereo [20] or lidar [16], and limit the degrees-of-freedom

| Monocular 6D Pose, Scale, and Shape Estimation on Unseen Instances | Recovered 3D Meshes in Absolute Metric Scale |

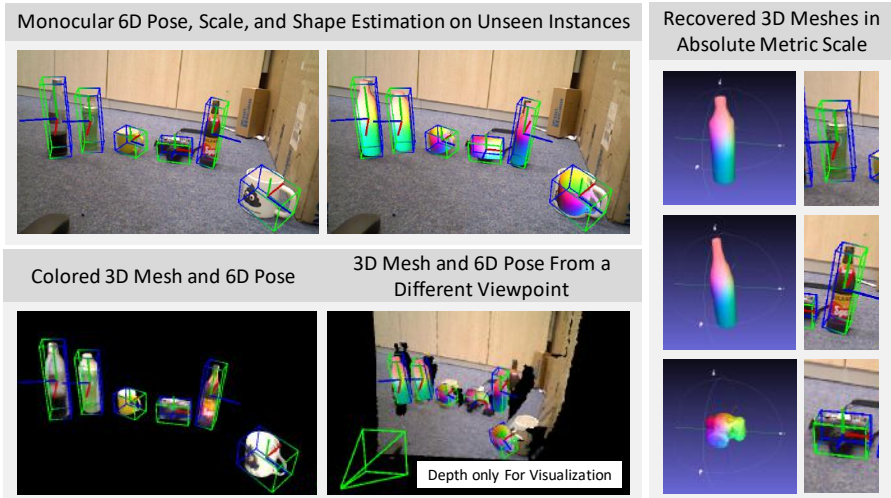| Colored 3D Mesh and 6D Pose | 3D Mesh and 6D Pose From a Different Viewpoint |

Depth only For Visualization

Fig. 1: From a monocular image, we detect (even unseen during training) objects from a particular class and estimate their 3D parameters, particularly, their 6D pose and 3D metric scale (top left). Moreover, we infer the corresponding 3D shape for each detection as shown in the right column and rendered into the scene on the top mid. To show that we are capable of inferring all parameters in correct 3D scale, we render our results from a different viewpoint (bottom mid).

for pose [49]. Assuming all objects rest on the ground plane, many applications in this field restrict the output to 3D bounding boxes, whose only degree of freedom is rotation around the Y-axis [3,4]. A handful approaches for category level 3D object detection in indoor environments have also been proposed [42,8].

In the field of robotics, where one of the applications is grasping and manipulation, estimating 3D bounding boxes is insufficient. This task also imposes different constraints: Objects are often physically and visually small, data is frequently available only in the form of color images and the ground-plane assumption is often unrealistic. Thus, 6D pose estimation is necessary. Due to the increase in degrees of freedom, this task is significantly more challenging.

As illustrated in Fig. 1, we propose a novel method for monocular class-level 6D pose estimation paired with metric object shape prediction, enabling new applications in augmented reality, and robotic manipulation. To ensure wide applicability of our method, we regress shape and pose parameters from a single RGB input image. This is particularly difficult due to the inherent ambiguities of 3D inference from 2D images. Inspired by [27,41], we introduce a novel 3D lifting module which directly aligns a predicted point cloud for each detection in 3D camera space. In contrast to other methods [27] which predict shape in a dedicated branch, trained independently from the pose estimation network, we back-propagate the final alignment through the whole network. Our method is thus trainable end-to-end and directly optimizes for the best alignment in 3D.

To the best of our knowledge, i) we are the first to introduce the task of monocular 6D pose paired with metric shape estimation. ii) We also introduce a new metric more appropriate for joint shape and pose estimation, which we call *Average Distance of Predicted Point Sets* and iii) propose a new loss which directly aligns the final outcome in 3D achieving superior results in pose over all metrics while giving accurate shapes.

## 2   Related Work

We first introduce essential recent works in monocular instance-level 6D object pose estimation. We then discuss the first approach to class-level 6D object pose estimation. Since most works for monocular class-level 3D object detection are found in the autonomous driving community, we also outline the most relevant works there. Finally, we take a look at recent trends in 3D shape recovery.

### 2.1   Monocular 6D Pose Estimation.

The related work for monocular pose estimation can be divided into *indirect* and *direct* methods. Indirect methods learn a pose embedding at an intermediate stage or aim at establishing 2D-3D correspondences. In contrast, direct methods compute the final 6D pose without using an intermediate representation.

As for indirect methods, inspired by [48,14], Sundermeyer *et al.* employ an augmented auto-encoder to learn an ambiguity-free pose embedding for retrieval [44]. The latter work was further extended by a single-encoder-multi-decoder architecture to increase scalability [43]. Rad *et al.* introduce a novel method to compute the 6D pose by establishing 2D-3D correspondences [37]. Thereby, they predict the 2D projections of the 3D bounding box corners and employ P$n$P to optimize for the 6D pose. Inspired by [37], many recent works similarly establish correspondences prior to applying P$n$P. The main differences reside in the sampling and regression of the keypoints as well as the increasing density of correspondences from sparse [37,12] to dense [52,21,33].

For direct methods, [13] treat pose estimation as a classification problem rather than regression. [28] learns to refine 6D poses using a projective proxy loss, while [51] enforces a point matching loss.

### 2.2   Class-level 6D Pose Estimation.

Wang *et al.* [46] recently proposed the first method for class-level object detection and 6D pose estimation. They predict 3D-3D correspondences such that the method can handle even unseen objects of the same class. Nonetheless, while we only use monocular data, [46] require depth data to compute the 6D pose and the object's scale. In addition, [46] only predicts 3D bounding boxes making robotic interaction infeasible.

## 2.3   Monocular Class-Level 3D Object Detection.

Chen *et al.* use multiple monocular cues such as shape, segmentation, location, and spatial context to instantiate 3D object proposals followed by a CNN-based scoring [3]. Kundu *et al.* [18] predict rotation and shape of cars employing a render-and-compare loss. In [27], the authors introduce a novel 3D lifting loss which measures the misalignment of the 3D bounding box corners. In addition, they also learn a shape space for *truncated sign distance functions* (TSDFs) using a 3D auto-encoder and train a sub-network to predict the latent representation for each detection. Nonetheless, these methods predict shape either only up to scale [18] or neglect it during optimization for pose and learn it at a later stage [27]. However, as shown in [3], shape can provide extra cues on the pose and should not be dissociated. Simonelli *et al.* [41] similarly measure the 3D bounding box misalignment, however, compute the error for each pose parameter separately to improve stability. Ku *et al.*propose to leverage instance-centric 3D proposal and local shape reconstruction [17]. In [25], the authors first conduct monocular depth prediction. Afterwards, they employ a PointNet architecture to obtain the objects poses and dimensions.

Interestingly, almost all these methods estimate the 3D rotation only up to the Y-axis and additionally assume all objects to be standing on the ground plane.

## 2.4   Recent Trends in Rigid 3D Shape Recovery.

Groeix *et al.* introduce AtlasNet, a network architecture built on top of Point-Net [36]. Sampling points from 2D uv-maps, the network can reconstruct 3D shapes of arbitrary resolution. Also based on PointNet, Yang *et al.* [50] proposed a light-weight end-to-end trainable encoder-decoder architecture, that learns to deform a 2D grid into the 3D object surface of a point cloud.

In [47], Wang *et al.* propose Pixel2Mesh, which directly regresses 3D meshes from RGB. The network iteratively refines the geometry of an input 3D ellipse, using features extracted from a single RGB image employing a graph convolutional neural network. Gkioxari *et al.* [6] introduce Mesh RCNN, an extension of Mask RCNN [9]. This network estimates a voxel representation for objects, which can be refined by a series of graph convolution-based refinement steps.

Finally, [30] and [32] both represent the 3D surface by the continuous decision boundary of a deep neural network classifier. Once the networks are trained, the object surface can be extracted from the learnt boundary.

## 3   Methodology

In this section, we introduce our method for detecting objects prior to computing 6D pose and metric shape, depicted as point cloud, from a single RGB image alone. We first present our proposed architecture, then describe how we learn an explainable shape space for each class using a PointNet auto-encoder and finally depict our novel loss for aligning the extracted 6D pose and shape in 3D space.
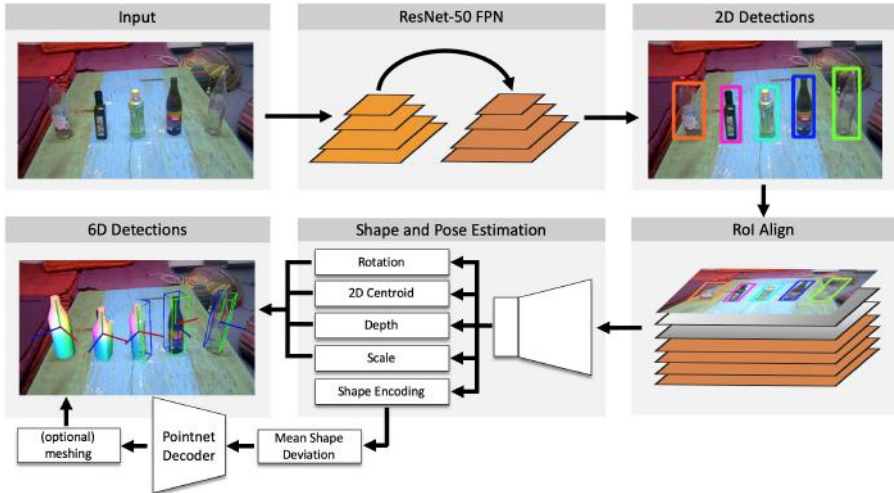
Fig. 2: **Schematic overview.** We feed the input image to a RetinaNet to infer 2D detections. We then collect all detections for each class and send them to the associated lifter module, which predicts the 6D pose together with the scale and the shape encoding. Finally, we retrieve point clouds from AtlasNet.

### 3.1   6D Pose and Metric Shape Estimation

Our method, as illustrated in Fig. 2, is based on a two-stage approach similar to Faster R-CNN [40]. We first predict 2D regions of interest using RetinaNet with Focal Loss [22]. The object proposals are then processed by our pose and shape estimator. We employ a ResNet-50 backbone with an FPN structure. For each detected object, we apply RoI align [9] to crop out $32 \times 32$ regions of interest. We also apply the RoI align operator on the input RGB image and the coordinate tensor [23]. Thus, the pose predictor is aware of the location of the crop and does not lose the global context. We concatenate both with the feature maps from the FPN to compute the feature map $x$ for the given RoI.

**From 2D Detection to 6D Pose and Metric Shape.** For each RoI, separate predictor networks branch off to infer: a 4D quaternion $q$ describing the *allocentric* 3D rotation in $SO(3)$, the 2D centroid $(x, y)$ as the projection of the 3D translation into the 2D image given camera matrix $K$, the distance $z$ of the detected object with respect to the camera, the metric size $(w, h, l)$ of the object, and the encoding $e$ of the shape.

The final pose is obtained by back-projecting the 2D centroid given the regressed depth and known camera matrix $K$ to compute the 3D translation. Then, we use the translation to compute the egocentric rotation from the allocentric rotation. Since we deal with cropped RoIs, the allocentric representation is favored as it is viewpoint invariant under 3D translation of the object [18,31].
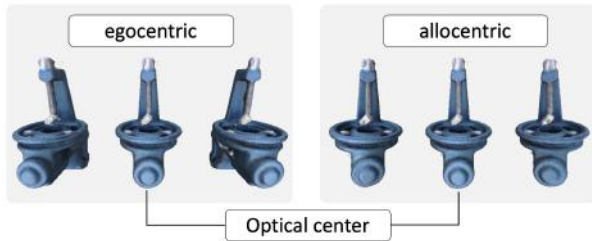
Fig. 3: **Egocentric *v.s.* allocentric rotation.** Under egocentric projection, a mere 3D translation of the object lateral to the image plane, leads to different object appearance. This is not the case under allocentric projection.

The difference is visualized in Fig. 3. Note that knowing the translation, one can easily convert from the allocentric to the egocentric representation and vice versa.

Since features from the FPN stage are forwarded to the second stage, we only require very small lifting networks for pose $(q, (x, y), z)$ and shape $(e, (w, h, l))$. Thus, we can easily afford using separate lifting modules for each object class. This further improves performance since poses and shapes from different classes do not interfere during optimization. In practice, for each lifter we first apply two 2D convolutions with batchnorm before diverging into separate branches for pose and shape. For each branch, we employ another two 2D convolutions with batchnorm followed by a fully-connected layer to predict the final parameters.

## 3.2   Learning of an Explainable Shape Space

A core novelty lies in the joint estimation of an object's shape alongside its 6D pose from a single RGB image. Since directly predicting the shape turns out to be unstable, we instead learn a low-dimensional latent space representation for each class. During inference, this enables the reconstruction of a 3D model by predicting only few shape parameters as opposed to a complete point cloud.

We employ AtlasNet [7] to learn a latent space representation of an object class $c$. The network is based on PointNet [36] and takes as input a complete point cloud which it encodes into a global shape descriptor. One can reconstruct a 3D shape by concatenating that descriptor with points sampled from a 2D uv-map and feeding the result to a decoder network. This approach decouples the number of predicted points from that in the original training shapes, thus enabling the reconstruction of shapes with arbitrary resolution. We train one AtlasNet network for each object class separately on a subset of point clouds $\mathcal{P}_c$ from ShapeNet [2], each one learning a class specific distribution of valid shapes in latent space. Using the AtlasNet encoder $\mathcal{E}_c$, we compute

$$\mathcal{S}_c := \{\, \mathcal{E}_c(p) \mid \forall p \in \mathcal{P}_c \,\}, \tag{1}$$
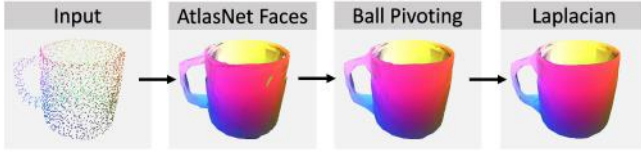
Fig. 4: **3D Point Cloud Meshing.** To mesh our prediction, we make use of the connectivity implied by AtlasNet and fill remaining holes with the ball-pivoting algorithm. Finally, we smooth the result using the Laplacian filter.

which is the set of all latent space representations of the training shapes. From $\mathcal{S}_c$, we then calculate a per-class mean latent shape $m_c$

$$m_c := \frac{1}{|\mathcal{S}_c|} \sum_{\forall s_c \in \mathcal{S}_c} s_c. \tag{2}$$

Let's now denote the shape prediction branch in Fig. 2 as $\mathcal{F}_{Shape}(x)$, a non-linear function that outputs a class-specific latent shape vector for the feature map $x$ from the given RoI. Then, instead of forcing $\mathcal{F}_{Shape}(x)$ to predict absolute shape vectors $e$, we let it infer a simple offset from $m_c$, such that $e = m_c + \mathcal{F}_{Shape}(x)$. Finally, the AtlasNet decoder network reconstructs a 3-dimensional point cloud, i.e. $p(x) = \mathcal{D}_c(m_c + \mathcal{F}_{Shape}(x)) = \mathcal{D}_c(e)$.

To encourage the latent shape predictions of $\mathcal{F}_{Shape}$ to stay inside of the learned shape distribution, we employ a special regularization loss. Assuming the training's per-class shape encodings span a convex shape space $Conv(S_c)$, we punish the network for any predicted $e \notin Conv(\mathcal{S}_c)$ and project them onto $\partial \mathcal{S}_c$, the boundary of $Conv(\mathcal{S}_c)$. In practice, we detect all $e \notin Conv(\mathcal{S}_c)$ as

$$I(e|\mathcal{S}_c) = \begin{cases} 0, & \text{if } \min_{\substack{\forall s_{c,i}, s_{c,j} \in \mathcal{S}_c \\ i \neq j}} ((e - s_{c,i})^T(e - s_{c,j})) \leq 0 \\ 1, & \text{otherwise.} \end{cases} \tag{3}$$

where $I(e|\mathcal{S}_c) = 0$ indicates that $e \in Conv(\mathcal{S}_c)$ and $I(e|\mathcal{S}_c) = 1$ otherwise.

We then project $e$ onto the line connecting the two closest points $(s_1, s_2) \in \mathcal{S}_c$. We retrieve $(s_1, s_2)$ by computing the euclidean distance for the regressed encodings with all elements of $\mathcal{S}_c$ and taking the two elements with the smallest distance. The error then amounts to the length of the vector rejection

$$\pi(e|s_1, s_2) = (s_1 - e) - \frac{(s_1 - e)^T(s_2 - e)}{(s_2 - e)^T(s_2 - e)}(s_2 - e). \tag{4}$$

The final loss for one sample $e$ then amounts to

$$\mathcal{L}_{reg}(e|\mathcal{S}_c, s_1, s_2) = I(e|\mathcal{S}_c) \cdot ||\pi(e|s_1, s_2)||_2. \tag{5}$$
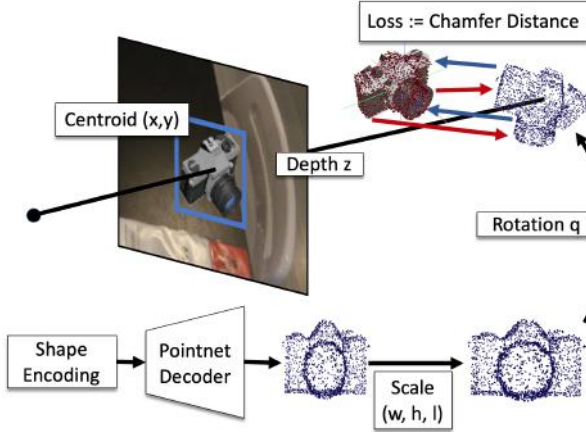
Fig. 5: **3D Point Cloud Loss.** Given the outputs from our network, we first retrieve the detected object's shape from the AtlasNet decoder. We then scale it to absolute size before transforming it into the scene with the predicted rotation and translation. We employ the chamfer distance between the ground truth point cloud and our prediction to enforce an optimal alignment in 3D.

**Meshing of 3D Point Clouds.** After estimating the shape as a point cloud, we can optionally also compute the associated mesh, *i.e.* the triangles of the model. Since AtlasNet samples 3D points uniformly from primitives, the triangles for each primitive can be easily inferred from the sampling. This facilitates meshing and allows a natural incorporation into our loss formulation, since we directly operate on point clouds. Unfortunately, the output mesh often exhibits holes. In order to fill these, we employ the ball-pivoting algorithm [1] and simply merge the output triangles. Finally, to reduce noise, we run one iteration of the Laplacian smoothing filter. The overall meshing process is also visualized in Fig. 4.

### 3.3   3D Point Cloud Alignment

Recent works [27,51,41] have shown that directly optimizing for the desired final target generally leads to superior results compared to enforcing separate loss terms for each regression target. Similarly, we propose a novel loss directly aligning our regressed 3D shape using the predicted 6D pose with the scene. Given the egocentric 3D rotation as 4D quaternion $q$ and 3D translation $t$, together with the shape encoding $e$, the decoder $\mathcal{D}_c$, and the scale $(w, h, l)$, we compute the shape of the detected object and transform it to the 3D camera space to obtain the point cloud

$$p_{3D} := q \cdot \left[ \begin{pmatrix} w \\ h \\ l \end{pmatrix} \cdot \mathcal{D}_c(e) \right] \cdot q^{-1} + K^{-1} \begin{pmatrix} x \cdot z \\ y \cdot z \\ z \end{pmatrix}, \tag{6}$$

with $K$ being the camera intrinsic matrix. We then simply measure the alignment against the ground truth point cloud $\bar{p}_{3D}$ using the chamfer distance with

$$\bar{p}_{3D} := \bar{R}\bar{p} + \bar{t}. \tag{7}$$

Thereby, $\bar{R}$ and $\bar{t}$ denote the ground truth 3D rotation and translation and $\bar{p}$ denotes the ground truth point cloud computed by uniformly sampling 2048 points from the CAD model. The loss for 3D alignment is calculated as

$$\mathcal{L}_{3D} := \frac{1}{|p_{3D}|} \sum_{v \in p_{3D}} \min_{\bar{v} \in \bar{p}_{3D}} ||v - \bar{v}||_2 + \frac{1}{|\bar{p}_{3D}|} \sum_{\bar{v} \in \bar{p}_{3D}} \min_{v \in p_{3D}} ||v - \bar{v}||_2. \tag{8}$$

When training from scratch, directly applying the chamfer distance turns out to be unstable due to local minima. Hence, we start with a warm-up training in which we compute the $L_1$-norm between each component and the ground truth using [15] to weight the different terms. Additionally, we disentangle $\mathcal{L}_{3D}$ for our predictions, similar to [41]. Therefore, we individually compute our 3D Point Cloud Loss for each pose parameter (*i.e.* $q$, $c$, $z$, $(w, h, l)$, $e$), while taking the ground truth for the remaining parameters. The final 3D loss is then calculated as the mean over all individual loss contributions.

The overall loss is the sum of the loss for 3D alignment and shape regularization

$$\mathcal{L} := \mathcal{L}_{3D} + \mathcal{L}_{reg}. \tag{9}$$

## 4    Evaluation

### 4.1    NOCS dataset.

For training and evaluation, we use the recently introduced NOCS dataset for class-level 6D pose estimation [46]. It consists of about 270k synthetic training images and 25k synthetic validation images. [46] employs a mixed-reality approach to render objects from ShapeNet [2] onto detected planes in real images. Additionally, they provide approximately 2.5k real test and 4.5k real training images. Overall, the dataset encompasses objects from 6 different classes.

### 4.2    Implementation details

We implemented our method in PyTorch [34] and trained all models with a batchsize of 6 for 200k iterations using ADAM optimizer and a learning rate of 0.0001. We decay the learning rate after 20k, 130k, and 170k iterations by 0.1, respectively. We train one network on three of the six object categories provided with the previously mentioned NOCS dataset [46], namely *Camera*, *Bottle* and *Cup*. The remaining classes *Bowl*, *Can* and *Laptop* are very limited in shape variation, especially since they are also normalized in scale. Even for *Laptop* the space of shapes only encompasses two pose states, *i.e. closed* and *open*. Since we aim to solve the problem of simultaneous estimation of 6D pose

and shape, we excluded them to stay focused on the central idea. We kindly refer to the supplementary material for examples illustrating the missing variations in those classes. Moreover, for the sake of completeness we additionally present the numbers for our method in the supplement.

Since annotating 6D pose, 3D scale, and 3D mesh is very difficult and time consuming, we decided to solely rely on synthetic data from NOCS for training the 3D parameters. Nonetheless, to bridge the domain gap, we also sample with a probability of 35% images from COCO or the real NOCS training set, however, only back-propagate the 2D loss for these samples. Further, when we evaluate on the real test set, we finetune the very last layers of our method for another 10k iterations on the real training data.

Optimally weighting multiple different loss terms is known to be complicated and a research topic on its own. It is particularly challenging when the loss terms originate from different domains such as rotation and translation. Consequently, we propose to instead directly measure the misalignment in 3D. Thus, to show the benefits of this reinterpretation, we also trained our network using an $L_1$ loss for each prediction. To this end, we set each weighting component $\lambda = 1$ (No Weighting). We also employ a more elaborate training strategy which involves learning the different weighting components [15] (Multi-Task Weighting).

### 4.3   Evaluation Protocol

**6D Pose Metrics.** Since we are the first to introduce the task of 6D pose estimation and metric shape retrieval in robotics, we want to introduce a new metric that measures both tasks jointly. Thus, we extend two of the most common metrics for 6D pose known as *Average Distance of Distinguishable Model Points* (ADD) and *Average Distance of Indistinguishable Model Points* (ADI) [11,10]. While ADD measures whether the average deviation $m$ of the transformed model points is less than 10% of the object's diameter
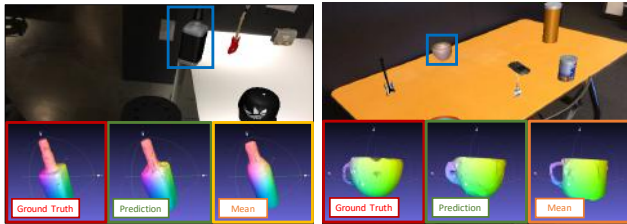
$$m = \operatorname*{avg}_{x \in \mathcal{M}} ||(Rx + t) - (\bar{R}x + \bar{t})||_2, \tag{10}$$

where $\mathcal{M}$ denotes the set of points for the given CAD model, ADI extends ADD for symmetries, measuring error as the mean distance to the *closest* model point

$$m = \operatorname*{avg}_{x_2 \in \mathcal{M}} \min_{x_1 \in \mathcal{M}} ||(Rx_1 + t) - (\bar{R}x_2 + \bar{t})||_2. \tag{11}$$

However, these metrics are not applicable in our case since point sets for ground truth $\bar{\mathcal{M}}$ and predicted shape $\mathcal{M}$ differ and even possess differences in scale. To circumvent the need for direct correspondences and be agnostic to scale discrepancies, we introduce the *Average Distance of Predicted Point Sets* (APP) which simply extends ADI to be computed bidirectionally

$$APP = \begin{cases} 1, & \text{if } m_1 \leq \alpha \cdot d(\mathcal{M}) \wedge m_2 \leq \alpha \cdot d(\bar{\mathcal{M}}) \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

| | Mean Chamfer Distance in mm | | |
|---|---|---|---|
| | Cup | Bottle | Camera |
| Predicted Shape | 3.4 | 3.7 | 11.2 |
| Mean Shape | 4.4 | 5.3 | 16.2 |

Table 1: Comparison of predicted and mean shapes to the corresponding ground truth shapes. For each class, the mean chamfer distance of the shapes estimated by our model to the ground truth is lower than that of the mean over all shapes.

with

$$m_1 = \operatorname*{avg}_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} ||(Rx_1 + t) - (\bar{R}x_2 + \bar{t})||_2 \quad m_2 = \operatorname*{avg}_{x_2 \in \mathcal{M}} \min_{x_1 \in \mathcal{M}} ||(Rx_1 + t) - (\bar{R}x_2 + \bar{t})||_2$$

and $d$ measuring the diameter of $\mathcal{M}$. We employ 20% and 50% as thresholds for $\alpha$. Since the related works do not incorporate shape prediction, we additionally compute the 3D IoU metric and $10°\&10cm$ metric to properly assess pose quality [46]. Similar to [41,46], we present all results computing the mean Average Precision (AP), measuring the area underneath the Precision-Recall curve.

**Implementation of Related Work.** We re-implemented two of the most relevant works from autonomous driving for monocular 3D object detection. We trained our network to optimize the alignment of the bounding box corners [27] and for the disentangled version from [41]. For a fair comparison with the only related work from 6D pose estimation, we implemented [46] on top of our backbone. To make it work with monocular data alone, we extended the method with an extra branch for depth prediction replacing the ground truth depth.

## 4.4   Shape Space Evaluation

In order to evaluate how well our model estimates the shape of detected objects, we crop at each ground truth RoI in the validation dataset and compute the chamfer distance from each predicted shape as well as the mean $m_c$ (see Eq 2) to the corresponding ground truth. Notice that the shapes in the validation dataset have never been seen before during training. Table 1 shows the mean chamfer distance for the *Cup*, *Bottle* and *Camera* class, respectively. The distance from predicted shape to ground truth is consistently lower, indicating that the model does indeed predict meaningful shapes for a specific image and object instance rather than trivially giving the mean. In fact, in 82.4% of the test cases for

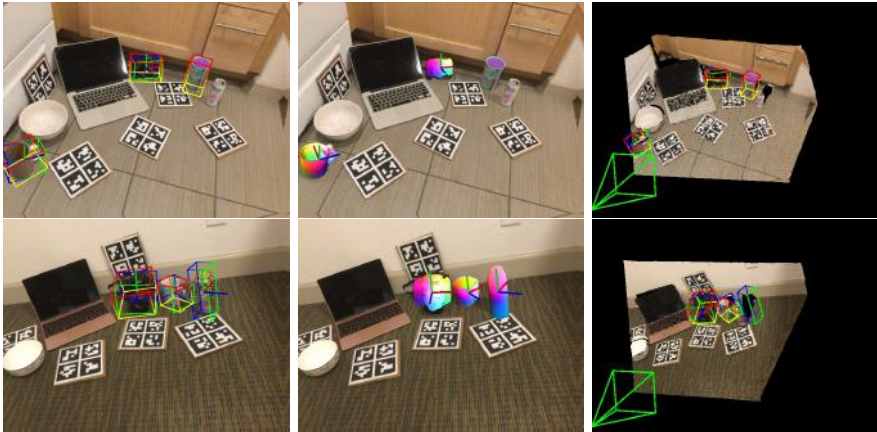| Method | 3D IOU @ (0.3 / 0.5) | | | 10° & 10cm | | | 3D APP @ (0.2 / 0.5) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Cup | Bottle | Camera | Cup | Bottle | Camera | Cup | Bottle | Camera |
| No Weighting | 18.1 / 6.1 | 22.8 / 7.9 | 11.0 / 1.5 | 11.1 | 19.1 | 8.5 | 17.2 / 49.2 | 16.1 / 48.7 | 10.5 / 35.4 |
| Multi-Task Weighting [15] | 21.8 / 7.1 | 18.7 / 6.0 | 13.5 / 3.3 | 27.8 | 20.8 | 9.6 | 19.6 / 50.1 | 14.2 / 47.8 | 13.4 / 35.2 |
| NOCS [46] w/o Depth | 10.1 / 1.6 | 11.1 / 1.1 | 5.7 / 0.5 | 0.1 | 5.5 | 0.1 | – | – | – |
| 3D Bbox [27] | 9.6 / 2.9 | 11.7 / 2.6 | 7.3 / 1.1 | 15.4 | 18.5 | 4.9 | | | |
| 3D Bbox Disentangled [41] | 13.4 / 3.7 | 20.3 / 6.1 | 10.8 / 1.5 | 1.1 | 18.4 | 1.8 | – | – | – |
| CPS w/o disentangling | 21.6 / 5.4 | 21.3 / 7.0 | 11.2 / 2.4 | 23.8 | 23.2 | 9.6 | 18.5 / 48.3 | 16.9 / 48.4 | 10.2 / 34.9 |
| CPS w/o $\mathcal{L}_{reg}$ | 9.0 / 3.1 | 21.1 / 6.7 | 8,6 / 2.3 | 23.7 | 25.0 | 7.6 | 8.6 / 26.8 | 14.9 / 50.9 | 7.2 / 30.2 |
| CPS | **22.2 / 7.3** | **23.4 / 8.2** | **16.6 / 3.6** | **28.8** | **25.6** | **12.5** | **21.6 / 52.9** | **18.8 / 56.5** | **14.8 / 43.7** |
| NOCS [46] w Depth | **92.7 / 92.5** | 87.6 / 67.0 | 87.9 / **80.8** | **57.8** | 84.9 | 45.8 | – | – | – |
| CPS w ICP (mean shape) | 89.0 / 72.2 | **88.6 / 83.8** | 89.9 / 51.9 | 29.6 | **93.2** | 37.3 | **89.2 / 89.4** | 86.4 / **89.8** | 94.6 / **95.6** |
| CPS w ICP | 89.1 / 81.6 | 87.9 / 83.1 | **94.1** / 70.8 | 55.5 | 93.0 | **52.7** | **89.2 / 89.4** | **86.6** / 89.6 | **95.5** / 95.6 |

Table 2: Top Left: Exemplary object pose estimations with rendered 3D bounding box, coordinate systems and shape mesh, overlaid on top of the respective input images. Top Right: BEV images emphasizing accurate depth estimation of multiple objects in the scene (Our results are visualized in *green* and ground truth in *red*.). Bottom: Ablation study and state-of-the-art methods evaluated on the synthetic validation dataset from [46]. We report AP scores for 3D IoU, rotation and translation as well as APP.

the *Cup* object class, our model produces more precise shapes compared to the mean. For the *Bottle* and *Camera* classes, this amounts to 69.1% and 71.4%, respectively. Qualitative examples of a bottle and a cup are given in the figures accompanying Table 1. The predicted bottle clearly exhibits the distinct corners and edges seen in the ground truth. In contrast, the mean bottle shows a very rounded out surface. Further, the predicted cup on the right shows the bowl-shape of the ground truth while the mean is much more cylindrical.

## 4.5   6D Pose, Scale, and Shape Assessment

**Synthetic Validation Dataset.** Table 2 shows evaluation results on the synthetic dataset published with [46]. One can denote easily, that among all methods, CPS clearly shows superior results with respect to all metrics.

Surprisingly, the two baselines (on the top) that simply weight the individual loss terms achieve overall strong results, yet, we still exceed them for all objects and metrics. As for the AP score on 3D IoU and a threshold of 0.3, we outperform [15] by 0.4% for *Cup*, 4.7% for *Bottle* and 3.1% for *Camera* with

| Method | 3D IOU @ (0.3 / 0.5) | | | 10°& 10cm | | | 3D APP @ (0.2 / 0.5) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Cup | Bottle | Camera | Cup | Bottle | Camera | Cup | Bottle | Camera |
| Multi-Task Weighting [15] | 16.1 / 2.1 | 0.3 / 0.0 | 2.8 / 0.2 | 0.7 | 0.2 | 0.0 | 19.9 / 55.0 | 0.0 / 2.1 | 3.6 / 32.4 |
| NOCS [46] w/o Depth | 20.7 / 2.9 | **7.9** / 0.8 | 1.1 / 0.0 | 0.8 | 1.2 | 0.0 | – | – | – |
| 3D Bbox Disentangled [41] | 29.0 / 6.4 | 6.5 / **1.0** | 21.4 / 2.5 | **1.6** | 7.0 | 0.7 | – | – | – |
| CPS | **34.4 / 11.9** | 3.1 / 0.4 | **21.8 / 3.8** | 1.3 | **9.5** | **1.9** | **36.8 / 64.6** | 1.6 / **14.1** | **19.6 / 56.7** |

Table 3: Top: Object pose estimations with rendered 3D bounding box, coordinate system and shape mesh, overlaid on top of the respective real input images. We show an alternative viewpoint on the right to better visualize the depth estimation. Bottom: State-of-the-art methods evaluated on the real test dataset from [46]. We report AP scores for 3D IoU, rotation and translation, and APP.

22.2%, 23.4% and 16.6%, respectively. Similar results can be also observed for $10°\&10cm$ metric. Regarding APP, we again significantly outperform both for all objects. Essentially, at a threshold of 0.5, we improve by 2.8% from 50.1% to 52.9% with respect to *Cup*, by 8.7% from 47.8% to 56.5% referring to *Bottle* and by 8.5% from 35.2% to 43.7% using the *Camera* object.

Other recently proposed methods [27,41] and [46] with depth prediction also produce inferior results. For most metrics we can even double their numbers, proving that our pointcloud alignment is unquestionably more effective than *e.g.* aligning 3D bounding box corners. In contrast to autonomous driving, objects in robotics often exhibit ambiguities such as symmetries (*e.g.* bottles and cups). Since [27,41] are minimizing the misalignment of 3D bounding box corners, their methods are very prone to ambiguities. As absolute pixel-wise depth estimation is very challenging, especially around the edges, our extended version of [46] is not able to properly fit the 6D pose and 3D scale.

When ablating the employed components, it becomes clear that the shape regularization term $\mathcal{L}_{reg}$ is crucial to prevent divergence. Especially complicated shapes such as *Camera* mostly benefit from penalizing encodings outside of their associated space. In terms of numbers, the relative performance difference is more than 35% for all metrics with respect to *Camera*. Disentangling the individual

loss components further stabilizes optimization, leading to a significant increase in performance across all objects.

To demonstrate the potential of CPS for real robotic applications and for a fair comparison with [46], we also evaluate our method after refining the poses with ICP, on the basis of our predicted shapes. The corresponding results are depicted in blue in Table 2. When employing depth, our numbers increase significantly. In particular, our AP score with respect to APP and 3D IoU approximately quadruples. In addition, for $10°\&10cm$ we can also double the results reported for CPS. Moreover, despite focusing on monocular data, we can even reach state-of-the-art performance for class-level 6D pose estimation using RGB-D data when comparing with [46]. While [46] exceeds us in terms of 3D IoU at a threshold of 0.5, with a mean AP of 80.1% compared to 78.5%, we can outperform [46] for the $10°\&10cm$ metric with 67.1% in comparison to 62.8%. Interestingly, we also run the same model with ICP, however, always use the mean shape instead of the predicted shape. While for rather simple objects as *Bottle* the difference is negligible, for more complex shapes such as *Cup* and *Camera* the discrepancy is vast. For more strict metrics, *e.g.* IoU@0.5 and $10°\&10cm$, the relative performance drops up to 47%. This clearly shows that our shapes are more accurate and can be leveraged for more reliable 6D pose estimation.

On top of Table 2, we show two qualitative examples. Notice how the shape matches nicely the perceived object in the scenes. Additionally, the 3D bounding box overlap is high. Inspired by [5], for better 3D understanding, we plotted the bird's-eye view visualization of the scene by conducting an orthogonal projection on the X-Z plane (2nd from the right). Similarly, we computed an orthogonal projection on the X-Y plane (right) as the ground plane assumption is invalid. Despite the ambiguity, our network is able to compute precise scales and poses as demonstrated in the almost perfect overlaps.

**Real Test Dataset.** We run evaluations on the real dataset of [46]. For this, we finetune the best performing networks by another 10k iterations in order to address the domain gap. The results are reported in Table 3. While the NOCS method performs best on the 3D IoU metric evaluated for the *Bottle* class, 3D Bbox Disentangled shows the top results for also the *Cup* class on the $10°\&10cm$ metric. Nonetheless, in all other cases our method clearly comes out as superior. Unfortunately, there is a significant drop for the *Camera* class with respect to the $10°\&10cm$ metric. Since COCO does not contain any data for *Camera*, the domain gap was most prominent for this object. We considered other datasets with camera annotations (*e.g.* Open Images [19]). However, these datasets contained either less data, no annotations, or less variations making them inapplicable. In relation to the other classes, the performance for *Bottle* for all methods is rather poor with respect to the 3D IoU metric. Due to the inherent ambiguity, estimating the Z-translation is the most difficult part but even a small offset can violate standard 3D IoU thresholds. This happens particularly easy for bottles due to their slim geometry. Despite monocular pose and shape estimation being the core focus of this work, it is worth mentioning that employing ICP we can

again strongly enhance the performance and report an average IoU of 79.5% and 59.0% with respect to a threshold of 0.3 and 0.5, respectively. The images accompanying Table 3 show some qualitative results without ICP, demonstrating the models' capabilities for real applications.

## 5   Conclusion

In this paper, we introduced the first method for monocular class-level 6D pose and shape estimation. We show that our point cloud alignment loss leads to superior results compared to all baselines. We further demonstrate that we can accurately estimate the underlying geometry of detected objects. One main limitation was the need to use synthetic data, introducing a significant domain gap when evaluating on real data. In the future, we want to investigate learning 6D pose and shape for classes from real data through self-supervised learning by making use of re-projection errors.

## References

1. Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., Taubin, G.: The ball-pivoting algorithm for surface reconstruction. IEEE transactions on visualization and computer graphics **5**(4), 349–359 (1999)
2. Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
3. Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Urtasun, R.: Monocular 3d object detection for autonomous driving. In: CVPR (2016)
4. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3d object detection network for autonomous driving. In: CVPR (2017)
5. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the kitti vision benchmark suite. In: CVPR (2012)
6. Gkioxari, G., Malik, J., Johnson, J.: Mesh r-cnn. arXiv preprint arXiv:1906.02739 (2019)
7. Groueix, T., Fisher, M., Kim, V.G., Russell, B.C., Aubry, M.: Atlasnet: A papier-m\^ ach\'e approach to learning 3d surface generation. arXiv preprint arXiv:1802.05384 (2018)
8. Gupta, S., Arbeláez, P., Girshick, R., Malik, J.: Aligning 3d models to rgb-d images of cluttered scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4731–4740 (2015)
9. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: ICCV (2017)
10. Hinterstoisser, S., Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., Lepetit, V.: Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In: ICCV (2011)
11. Hodan, T., Matas, J., Obdrzalek, S.: On Evaluation of 6D Object Pose Estimation. In: ECCVW (2016)
12. Hu, Y., Hugonot, J., Fua, P., Salzmann, M.: Segmentation-driven 6d object pose estimation. In: CVPR (2019)
13. Kehl, W., Manhardt, F., Ilic, S., Tombari, F., Navab, N.: SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again. In: ICCV (2017)

14. Kehl, W., Milletari, F., Tombari, F., Ilic, S., Navab, N.: Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In: ECCV (2016)
15. Kendall, A., Gal, Y., Cipolla, R.: Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In: CVPR (2018)
16. Ku, J., Mozifian, M., Lee, J., Harakeh, A., Waslander, S.L.: Joint 3d proposal generation and object detection from view aggregation. In: IROS (2018)
17. Ku, J., Pon, A.D., Waslander, S.L.: Monocular 3d object detection leveraging accurate proposals and shape reconstruction. In: CVPR (2019)
18. Kundu, A., Li, Y., Rehg, J.M.: 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In: CVPR (2018)
19. Kuznetsova, A., Rom, H., Alldrin, N., Uijlings, J., Krasin, I., Pont-Tuset, J., Kamali, S., Popov, S., Malloci, M., Duerig, T., Ferrari, V.: The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. arXiv:1811.00982 (2018)
20. Li, P., Chen, X., Shen, S.: Stereo r-cnn based 3d object detection for autonomous driving. In: CVPR (2019)
21. Li, Z., Wang, G., Ji, X.: Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 7678–7687 (2019)
22. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: ICCV (2017)
23. Liu, R., Lehman, J., Molino, P., Such, F.P., Frank, E., Sergeev, A., Yosinski, J.: An intriguing failing of convolutional neural networks and the coordconv solution. In: NeurIPS (2018)
24. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A.C.: Ssd: Single shot multibox detector. In: ECCV (2016)
25. Ma, X., Wang, Z., Li, H., Ouyang, W., Zhang, P.: Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In: ICCV (2019)
26. Manhardt, F., Arroyo, D., Rupprecht, C., Busam, B., Birdal, T., Navab, N., Tombari, F.: Explaining the ambiguity of object detection and 6d pose from visual data. In: ICCV (2019)
27. Manhardt, F., Kehl, W., Gaidon, A.: Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. In: CVPR (2019)
28. Manhardt, F., Kehl, W., Navab, N., Tombari, F.: Deep model-based 6d pose refinement in rgb. In: ECCV (2018)
29. Mercier, J.P., Mitash, C., Giguere, P., Boularias, A.: Learning object localization and 6d pose estimation from simulation and weakly labeled real images. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 3500–3506. IEEE (2019)
30. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019)
31. Mousavian, A., Anguelov, D., Flynn, J., Kosecka, J.: 3d bounding box estimation using deep learning and geometry. In: CVPR (2017)
32. Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: CVPR (2019)
33. Park, K., Patten, T., Vincze, M.: Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In: ICCV (2019)

34. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. In: NIPSW (2017)
35. Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H.: Pvnet: Pixel-wise voting network for 6dof pose estimation. In: CVPR (2019)
36. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: CVPR (2017)
37. Rad, M., Lepetit, V.: Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In: ICCV (2017)
38. Rad, M., Roth, P.M., Lepetit, V.: Alcn: Adaptive local contrast normalization for robust object detection and 3d pose estimation. In: BMVC (2017)
39. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR (2016)
40. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: NeurIPS (2015)
41. Simonelli, A., Rota Bulo, S., Porzi, L., Lopez-Antequera, M., Kontschieder, P.: Disentangling monocular 3d object detection. In: ICCV (2019)
42. Song, S., Xiao, J.: Deep sliding shapes for amodal 3d object detection in rgb-d images. In: CVPR (2016)
43. Sundermeyer, M., Durner, M., Puang, E.Y., Marton, Z.C., Triebel, R.: Multi-path learning for object pose estimation across domains. arXiv preprint arXiv:1908.00151 (2019)
44. Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R.: Implicit 3d orientation learning for 6d object detection from rgb images. In: ECCV (2018)
45. Tekin, B., Sinha, S.N., Fua, P.: Real-time seamless single shot 6d object pose prediction. In: CVPR (2018)
46. Wang, H., Sridhar, S., Huang, J., Valentin, J., Song, S., Guibas, L.J.: Normalized object coordinate space for category-level 6d object pose and size estimation. In: CVPR (2019)
47. Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 52–67 (2018)
48. Wohlhart, P., Lepetit, V.: Learning descriptors for object recognition and 3d pose estimation. In: CVPR (2015)
49. Xu, B., Chen, Z.: Multi-level fusion based 3d object detection from monocular images. In: CVPR (2018)
50. Yang, Y., Feng, C., Shen, Y., Tian, D.: Foldingnet: Point cloud auto-encoder via deep grid deformation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 206–215 (2018)
51. Yu, X., Tanner, S., Venkatraman, N., Dieter, F.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. In: RSS (2018)
52. Zakharov, S., Shugurov, I., Ilic, S.: Dpod: Dense 6d pose object detector in rgb images. In: ICCV (2019)