Normalized Object Coordinate Space for Category-Level 6D Object Pose and Size Estimation

He Wang[1] Srinath Sridhar[1] Jingwei Huang[1] Julien Valentin[2] Shuran Song[3] Leonidas J. Guibas[1]
[1]Stanford University [2]Google Inc. [3]Princeton University

Abstract

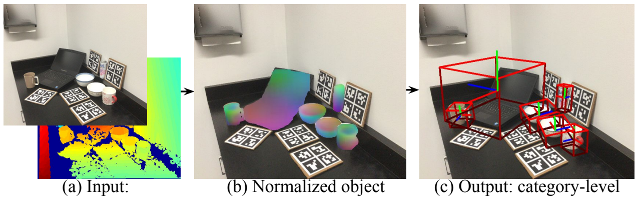
The goal of this paper is to estimate the 6D pose and dimensions of unseen object instances in an RGB-D image. Contrary to "instance-level" 6D pose estimation tasks, our problem assumes that no exact object CAD models are available during either training or testing time.

To handle different and unseen object instances in a given category, we introduce a ***Normalized Object Coordinate Space (NOCS)****—a shared canonical representation for all possible object instances within a category. Our region-based neural network is then trained to directly infer the correspondence from observed pixels to this shared object representation (NOCS) along with other object information such as class label and instance mask. These predictions can be combined with the depth map to jointly estimate the metric 6D pose and dimensions of multiple objects in a cluttered scene. To train our network, we present a new context-aware technique to generate large amounts of fully annotated mixed reality data. To further improve our model and evaluate its performance on real data, we also provide a fully annotated real-world dataset with large environment and instance variation.*

Extensive experiments demonstrate that the proposed method is able to robustly estimate the pose and size of unseen object instances in real environments while also achieving state-of-the-art performance on standard 6D pose estimation benchmarks.

1. Introduction

Detecting objects, and estimating their 3D position, orientation and size is an important requirement in virtual and augmented reality (AR), robotics, and 3D scene understanding. These applications require operation in new environments that may contain previously unseen object instances. Past work has explored the *instance-level 6D pose estimation* problem [37, 46, 27, 51, 6, 28] where exact CAD models and their sizes are available beforehand. Unfortunately, these techniques cannot be used in general settings where the vast majority of the objects have never been seen before and have no known CAD models. On the other hand, *category-level 3D object detection* methods [43, 36, 9, 34, 49, 12] can estimate object class labels and 3D bounding boxes without requiring exact CAD models. However, the estimated 3D bounding boxes are viewpoint-dependent and do not encode the precise orientation of objects. Thus, both these classes of methods fall short of the requirements of applications that need the 6D pose and 3 non-uniform scale parameters (encoding dimensions) of unseen objects in new environments.



(a) Input: single RGB-D image

(b) Normalized object coordinate space estimation

(c) Output: category-level 6D pose and size

Figure 1. We present a method for category-level 6D pose and size estimation of multiple unseen objects in an RGB-D image. A novel normalized object coordinate space (NOCS) representation (color-coded in (b)) allows us to consistently define 6D pose at the category-level. We obtain the full metric 6D pose (axes in (c)) and the dimensions (red bounding boxes in (c)) for unseen objects.

In this paper, we aim to bridge the gap between these two families of approaches by presenting, to our knowledge, the first method for **category-level 6D pose and size estimation** of multiple objects. Estimating the 6D pose and size of previously unseen objects is a challenging problem. Since we cannot use CAD models for unseen objects, the first challenge is to find a representation that allows definition of 6D pose and size for different objects in a particular category. The second challenge is the unavailability of large-scale datasets for training and testing. Datasets such as SUN RGB-D [41] or NYU v2 [40] lack annotations for precise 6D pose and size, or do not contain table-scale object categories—these are exactly the types of objects that arise in table-top or desktop manipulation tasks for which knowing the 6D pose and size would be useful.

To address the representation challenge, we formulate the problem as finding correspondences between object pixels to normalized coordinates in a shared object description space (see Section 3). We define a shared space called the **Normalized Object Coordinate Space** (NOCS) in which all objects are contained within a common normalized space, and all instances within a category are consistently oriented. This enables 6D pose and size estimation, even for unseen object instances. At the core of our method is a convolutional neural network (CNN) that jointly estimates the object class, instance mask, and a *NOCS map* of multiple objects from a single RGB image. Intuitively, the NOCS map captures the normalized shape of the visible parts of the object by predicting dense correspondences between object pixels and the NOCS. Our CNN estimates the NOCS map by formulating it either as a pixel regression or classification problem. The NOCS map is then used with the depth map to estimate the full metric 6D pose and size of the objects using a pose fitting method.

To address the data challenge, we introduce a spatially context-aware mixed reality method to automatically generate large amounts of data (275K training, 25K testing) composed of realistic-looking synthetic objects from ShapeNetCore [8] composited with real tabletop scenes. This approach allows the automatic generation of realistic data with object clutter and full ground truth annotations for class label, instance mask, NOCS map, 6D pose, and size. We also present **a real-world dataset for training and testing** with 18 different scenes and ground truth 6D pose and size annotations for 6 object categories, and in total 42 unique instances. To our knowledge, ours is the largest and most comprehensive training and testing datasets for 6D pose and size, and 3D object detection tasks.

Our method uses input from a commodity RGB-D sensor and is designed to handle both symmetric and asymmetric objects, making it suitable for many applications. Figure 1 shows examples of our method operating on a tabletop scene with multiple objects unseen during training. In summary, the main contributions of this work are:

- Normalized Object Coordinate Space (NOCS), a unified shared space that allows different but related objects to have a common reference frame enabling 6D pose and size estimation of unseen objects.
- A CNN that jointly predicts class label, instance mask, and NOCS map of multiple unseen objects in RGB images. We use the NOCS map together with the depth map in a pose fitting algorithm to estimate the full metric 6D pose and dimensions of objects.
- **Datasets**: A spatially context-aware mixed reality technique to composite synthetic objects within real images allowing us to generate a large annotated dataset to train our CNN. We also present fully annotated real-world datasets for training and testing.

2. Related Work

In this section, we focus on reviewing related work on category-level 3D object detection, instance-level 6D pose estimation, category-level 4 DoF pose estimation from RGB-D images, and different data generation strategies.

Category-Level 3D Object Detection: One of the challenges in predicting the 6D pose and size of objects is localizing them in the scene and finding their physical sizes, which can be formulated as a 3D detection problem [54, 22, 21, 31, 14]. Notable attempts include [43, 55] who take 3D volumetric data as input to directly detect objects in 3D. Another line of work [36, 20, 10, 29] proposes to first produce 2D object proposals in 2D image and then project the proposal into 3D space to further refine the final 3D bounding box location. The techniques described above reach impressive 3D detection rates, but unfortunately solely focus on finding the bounding volume of objects and do not predict the 6D pose of the objects.

Instance-Level 6 DoF Pose Estimation: Given its practical importance, there is a large body of work focusing on instance-level 6D pose estimation. Here, the task is to infer the 3D location and 3D rotation of objects (no scale), assuming exact 3D CAD models and size of these objects are available during training. The state of the art can be broadly categorized as template matching or object coordinates regression techniques. Template matching techniques align 3D CAD models to observed 3D point clouds with algorithms such as iterative closest point [4, 53], or use hand crafted local descriptors to further guide the alignment process [26, 11]. This family of techniques often suffer from inter- and intra-object occlusions, which is typical when we have only partial scans of objects. The second category of approaches based on object coordinates regression aim to regress the object surface position corresponding to each object pixel. Such techniques have been successfully employed for body pose estimation [45, 18], camera relocalization [39, 48] and 6D object pose estimation [5].

Both the above approaches need *exact 3D models* of the objects during training and test time. Besides the practical limitation in storing all 3D CAD models or learned object coordinate regressors in memory at test time, capturing high-fidelity and complete 3D models of a very large array of objects is a challenging task. Although our approach is inspired by object coordinate regression techniques, it also significantly differs from the above approaches since we no longer require complete and high-fidelity 3D CAD models of objects at test time.

Category-Level 4 DoF Pose Estimation: There has been some work on category-level pose estimation [20, 42, 19, 35, 7], however they all make simplifying assumptions. First, these algorithms constrain the rotation prediction to be only along the gravity direction (only four degrees of freedom). Second, they focus on a few big room-scale object categories (*e.g.*, chairs, sofa, beds or cars) and do not take object symmetry into account [20, 42, 19]. On the contrary, we estimate the pose of a variety of hand-scale objects, which are often much more challenging than the bigger room-scale objects due to larger pose variation. Our method also predicts full 6D pose and size without assuming the objects gravity direction. Finally, our method runs at interactive frame rates (0.5 s per frame), which is significantly faster than alternative approaches (∼70 s per frame for [20], 25 mins per frame for [42]).

Training Data Generation: A major challenge with training CNNs is the lack of training data with sufficient category, instance, pose, clutter, and lighting variation. There have been several efforts aimed at constructing real-world datasets containing object labels (*e.g.*, [40, 41, 50]). Unfortunately, these datasets tend to be relatively small, mostly due to the high cost (time and money) associated with ground truth annotation. This limitation is a motivator for other works (*e.g.*, [35, 44, 51]) which generate data that is exclusively synthetic allowing the generation of large amounts of perfectly annotated training data at a smaller cost. For the sake of simplicity, all these datasets ignore a combination of factors (material, sensor noise, and lighting) which creates a de-facto domain gap between the synthetic and real data distributions. To reduce this gap, [13] have generated datasets that mix real and synthetic data by rendering virtual objects on real backgrounds. While the backgrounds are realistic, the rendered objects are flying midair and out of context [13], which prevent algorithms from making use of important contextual cues.

We introduce a new mixed reality method to automatically generate large amounts of data composed of synthetic renderings of objects and real backgrounds in a context-aware manner. By leveraging depth, and realistic lighting and shading our data is more realistic than previous work.

This is supported by experiments that show that our context-aware mixed reality training data enables the model to generalize better to real-word test data. We also present a real-world dataset to further improve learning and for evaluation.

3. Background and Overview

Category-Level 6D Object Pose and Size Estimation: We focus on the problem of estimating the 3 rotation, 3 translation, and 3 scale parameters (dimensions) of object instances. The solution to this problem can be visualized as a tight oriented bounding box around an object (see Figure 1). Although not previously observed, these objects come from known object categories (*e.g.*, *camera*) for which training samples have been observed during training. This task is particularly challenging since we cannot use CAD models at test time and 6D pose is not well-defined for unseen objects.

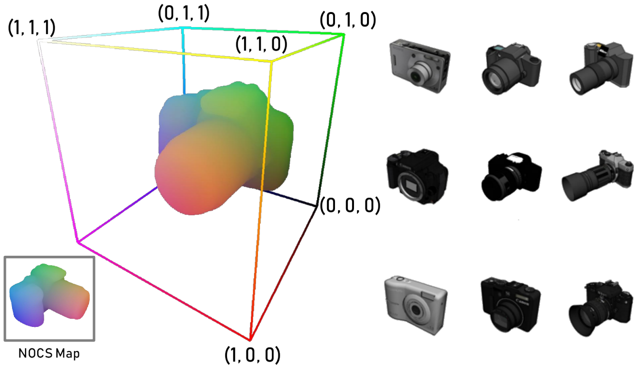


Figure 2. The Normalized Object Coordinate Space (NOCS) is a 3D space contained within a unit cube. For a given object category, we use canonically oriented instances and normalize them to lie within the NOCS. Each (x, y, z) position in the NOCS is visualized as an RGB color tuple. We train our network on the perspective projection of the NOCS on the RGB image, the NOCS map (bottom left inset). At test time, the network regresses the NOCS map which is then used together with the depth map for 6D pose and size estimation.

To overcome this, we propose a new representation that defines a shared object space enabling the definition of 6D pose and size for unseen objects.

Normalized Object Coordinate Space (NOCS): The NOCS is defined as a 3D space contained within a unit cube *i.e.*, $\{x, y, z\} \in [0, 1]$. Given a shape collection of known object CAD models for each category, we normalize their size by uniformly scaling the object such that the diagonal of its tight bounding box has a length of 1 and is centered within the NOCS space (see Figure 2).

Furthermore, we align the object center and orientation consistently across the same category. We use models from ShapeNetCore [8] which are already canonicalized for scale, position, and orientation.

Figure 2 shows examples of canonicalized shapes in the *camera* category. Our representation allows each vertex of a shape to be represented as a tuple (x, y, z) within the NOCS (color coded in Figure 2).

Our CNN predicts the 2D perspective projection of the color-coded NOCS coordinates, *i.e.*, a **NOCS map** (bottom left in Figure 2). There are multiple ways to interpret a NOCS map: (1) as a **shape reconstruction** in NOCS of the observed parts of the object, or (2) as **dense pixel–NOCS correspondences**. Our CNN learns to generalize shape prediction for unseen objects, or alternatively learns to predict object pixel–NOCS correspondences when trained on a large shape collection. This representation is more robust than other approaches (*e.g.*, bounding boxes) since we can operate even when the object is only partially visible.

Method Overview: Figure 3 illustrates our approach which uses an RGB image and a depth map as input. The CNN estimates the class label, instance mask, and the

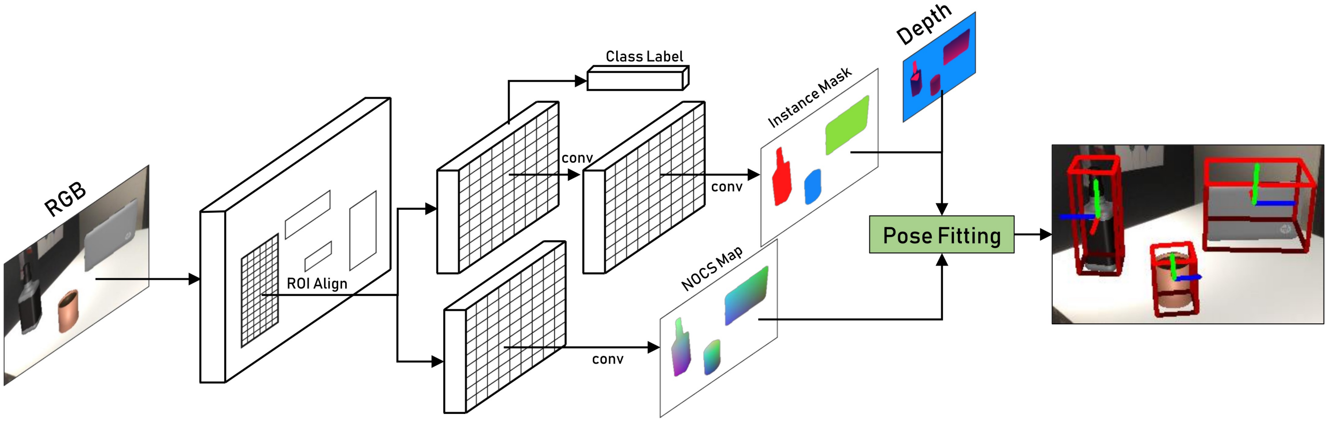


Figure 3. The inputs to our method are the RGB and depth images of a scene with multiple objects. Our CNN predicts the class label, instance mask, and NOCS map (color-coded) for each object in the RGB image. We then use the NOCS maps for each object together with the depth image to obtain the full metric 6D pose and size (axes and tight red bounding boxes), even if the object was never seen before.

NOCS map from only the RGB image. We do not use the depth map in the CNN because we would like to exploit existing RGB datasets like COCO, which do not contain depth, to improve performance.

The NOCS map encodes the shape and size of the objects in a normalized space. We can therefore use the depth map at a later stage to lift this normalized space, and to predict the full metric 6D object pose and size using robust outlier removal and alignment techniques.

Our CNN is built upon the Mask R-CNN framework [23] with improvements to jointly predict NOCS maps in addition to class labels, and instance masks. Section 5 contains more details on our improvements and new loss functions that can handle symmetric objects. During training, we use ground truth images rendered with a new Context-Aware MixEd ReAlity (CAMERA) approach (see Section 4). This large dataset allows us to generalize to new instances from new categories at testing time. To further bridge the domain gap we also use a smaller real-world dataset.

4. Datasets

A major challenge in category-level 3D detection, and 6D pose and size estimation is the unavailability of ground truth data. While there have been several attempts like NYU v2 [40] and SUNRGB-D [41], they have important limitations. First, they do not provide 6D pose of objects and focus on just 3D bounding boxes. Second, applications such as augmented reality and robotics benefit from hand-scale objects in tabletop settings which are missing from current datasets which focus on on larger objects such as as chairs and tables. Finally, these datasets do not contain annotations for the type of ground truth we need (*i.e.*, NOCS maps) and contain limited number of examples.

4.1. Context-Aware Mixed Reality Approach

To facilitate the generation of large amounts of training data with ground truth for hand-scale objects, we propose a new **Context-Aware MixEd ReAlity (CAMERA)** approach which addresses the limitations of previous approaches and makes data generation less time consuming and significantly more cost-effective. It combines real background images with synthetically rendered foreground objects in a *context-aware* manner *i.e.*, the synthetic objects are rendered and composited into real scenes with plausible physical locations, lighting, and scale (see Figure 4). This *mixed reality* approach allows us to generate significantly larger amounts of training data than previously available.

Real Scenes: We use real RGB-D images of 31 widely vaying indoor scenes as background (Figure 4 middle). Our focus is on tabletop scenes since the majority of indoor human-centric spaces consist of tabletop surfaces with hand-scale objects. In total, we collected 553 images for the 31 scenes, 4 of which were set aside for validation.

Synthetic Objects: To render realistic looking objects in the above real scenes, we picked hand-scale objects from ShapeNetCore [8], manually removing any that did not look real or had topology problems. In total, we picked 6 object categories—*bottle, bowl, camera, can, laptop*, and *mug*. We also created a *distractor* category consisting of object instances from categories not listed above such as *monitor, phone*, and *guitar*. This improves robustness when making predictions for our primary categories even if other objects are present in the scene. Our curated version of ShapeNetCore consists of 1085 individual object instances of which we set aside 184 instances for validation.

Context-Aware Compositing: To improve realism, we composite virtual objects in a context-aware manner *i.e.*, we place then where they would naturally occur (*e.g.*, on supporting surfaces) with plausible lighting. We use a plane detection algorithm [15] to obtain pixel-level plane segmentation in real images. Subsequently, we sample random locations and orientations on the segmented plane where synthetic objects could be placed. We then place several virtual light sources to mimic real indoor lighting conditions. Finally, we combine the rendered and real images to produce

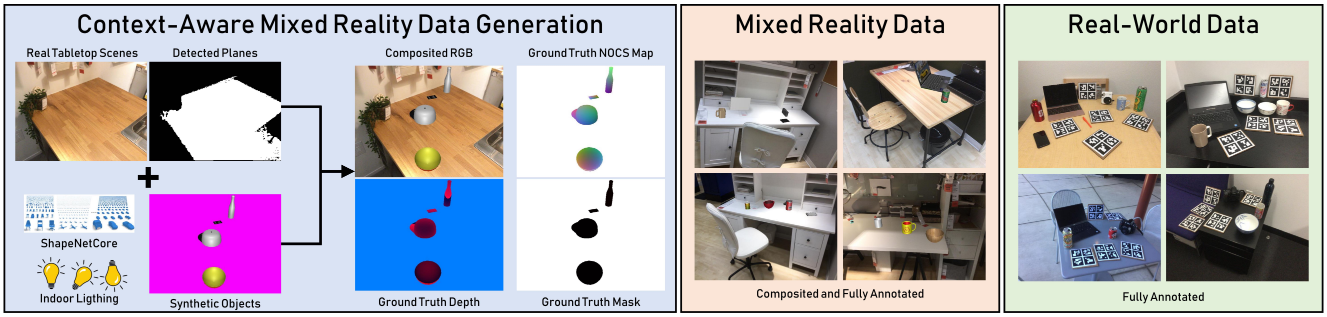


Figure 4. We use a Context-Aware MixEd ReAlity (CAMERA) approach to generate data by combining real images of tabletop scenes, detect planar surfaces, and render synthetic objects onto the planar surfaces (left). Since the objects are synthetic, we obtain accurate ground truth for class label, instance mask, NOCS map, and 6D pose and size. Our approach is fast, cost-effective, and results in realistic and plausible images (middle). We also gather a real-world dataset for training, testing, and validation (right).

a realistic composite with perfect ground truth NOCS maps, masks, and class labels.

In total, we render **300K composited images**, of which 25K are set aside for validation. To our knowledge, this the largest dataset for category-level 6D pose and size estimation. Our mixed reality compositing technique was implemented using the Unity game engine [2] with custom plugins for plane detection and point sampling (all of which will be publicly released). The images generated using our method look plausible and realistic resulting in improved generalization compared to using non-context aware data.

4.2. Real-World Data

To further improve and validate our algorithm's real-world performance under challenging clutter and lighting conditions, we curate two real-world datasets: (1) a real-world *training dataset* that supplements the mixed reality data we generated earlier, (2) a real-world *testing dataset* to evaluate the performance of 6D pose and size estimation. We developed a semi-automatic method to annotate ground truth object pose and size. Figure 4 shows examples of our real-world data.

We captured **8K RGB-D frames** (4300 for training, 950 for validation and 2750 for testing)

of 18 different real scenes (7 for training, 5 for validation, and 6 for testing) using a Structure Sensor [1]. For each of the training and testing subsets, we used 6 categories and 3 unique instances per category. For the validation set we use 6 categories with 1 unique instance per category. We place more than 5 object instances in each scene to simulate real-world clutter. For each instance, we obtained a clean and accurate 3D mesh using an RGB-D reconstruction algorithm that we developed for this purpose. In total, our combined datasets contain **18 different real scenes**, **42 unique object instances** spanning 6 categories making it one of the most comprehensive datasets for category-level 6D pose and size estimation.

5. Method

Figure 3 shows our method for 6D pose and size estimation of multiple previously unseen objects from an RGB-D image. We use a CNN to predict class labels, masks, and NOCS maps of objects. Subsequently, we use the NOCS map and the depth map to estimate the full metric 6D pose and size of objects.

5.1. NOCS Map Prediction CNN

The goal of our CNN is to estimate class labels, instance masks, and NOCS maps of objects based purely on RGB images. We build upon the region-based Mask R-CNN framework [23] since it has demonstrated state-of-the-art performance on 2D object detection and instance segmentation tasks, is modular and flexible, fast, and can easily be augmented to predict NOCS maps as described below.

5.1.1 NOCS Map Head

Mask R-CNN builds upon the Faster R-CNN architecture [38] and consists of two modules—a module to propose regions potentially containing objects, and a detector to detect and classify objects within regions. Additionally, it also predicts the instance masks of objects within the regions.

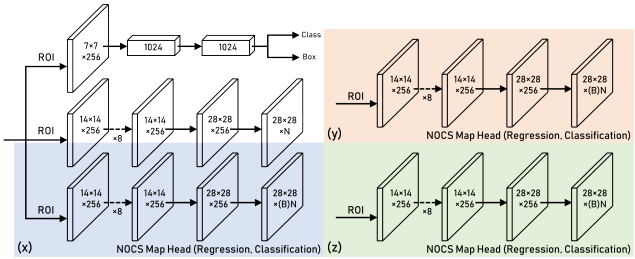


Figure 5. NOCS map head architecture. We add three additional heads to the Mask R-CNN architecture to predict the x, y, z coordinates of the NOCS map (colored boxes). These heads can either be used for direct pixel regression or classification (best). We use ReLU activation and 3×3 convolutions.

Our main contribution is the addition of 3 head architectures to Mask R-CNN for predicting the x, y, z components of the NOCS maps (see Figure 5). For each proposed region of interest (ROI), the output of a head is of size 28×28×N, where N is the number of categories and each category containing the x (or y, z) coordinates for all detected objects in that category. Similar to the mask head, we use the object category prior to look up the corresponding prediction channel during testing. During training, only the NOCS map component from the ground truth object category is used in the loss function. We use a ResNet50 [25] backbone together with Feature Pyramid Network (FPN).

Regression vs. Classification: To predict the NOCS map, we can either regress each pixel value or treat it as a classification problem by discretizing the pixel values (denoted by (B) in Figure 5). Direct regression is presumably a harder task with the potential to introduce instability during training. Similarly, pixel classification with large number of classes (*e.g.*, $B = 128, 256$) could introduce more parameters making training even more challenging than direct regression. Our experiments revealed that pixel classification with $B = 32$ performed better than direct regression.

Loss Function: The class, box, and mask heads of our network use the same loss functions as described in [23]. For the NOCS map heads, we use two loss functions: a standard softmax loss function for classification, and the following soft L^1 loss function for regression which makes learning more robust.

$$L(\mathbf{y}, \mathbf{y}^*) = \frac{1}{n} \begin{cases} 5\left(\mathbf{y} - \mathbf{y}^*\right)^2, & |\mathbf{y} - \mathbf{y}^*| \leq 0.1 \\ |\mathbf{y} - \mathbf{y}^*| - 0.05, & |\mathbf{y} - \mathbf{y}^*| > 0.1 \end{cases}, \qquad \forall \mathbf{y} \in N, \mathbf{y}^* \in N_p,$$

where $\mathbf{y} \in \mathcal{R}^3$ is the ground truth NOCS map pixel value, $\mathbf{y}^*$ is the predicted NOCS map pixel value, n is the number of mask pixels inside the ROI, I and I_p are the ground truth and predicted NOCS maps.

Object Symmetry: Many common household objects (*e.g.*, bottle) exihibit symmetry about an axis. Our NOCS representation does not take symmetries into account which resulted in large errors for some object classes.

To mitigate this issue, we introduce a variant of our loss function that takes symmetries into account. For each category in our training data, we define an axis of symmetry. Pre-defined rotations about this axis result in NOC maps that produce identical loss function values. For instance, a cuboid with a square top has a vertical symmetry axis. Rotation by angles, $\theta = \{0°, 90°, 180°, 270°\}$ on this axis leads to NOC maps that are identical and therefore have the same loss.

For non-symmetric objects, $\theta = 0°$ is unique. We found that a $|\theta| \leq 6$ is enough to handle most symmetric categories. We generate ground truth NOCS maps, $\{\tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_{|\theta|}\}$, that are rotated $|\theta|$ times along the symmetry axis. We then define our symmetric loss function, L_s as $L_s = \min_{i=1,\ldots,|\theta|} L\left(\tilde{\mathbf{y}}_i, \mathbf{y}^*\right)$, where $\mathbf{y}^*$ denotes the predicted NOCS map pixel (x, y, z).

Training Protocol: We initialize the ResNet50 backbone, RPN and FPN with the weights trained on 2D instance segmentation task on the COCO dataset[33]. For all heads, we use the initialization technique proposed in [24]. We use a batch size of 2, initial learning rate of 0.001, and an SGD optimizer with a momentum of 0.9 and a 1×10^{-4} weight decay. In the first stage of training, we freeze the ResNet50 weights and only train the layers in the heads, the RPN and FPN for 10K iterations. In the second stage, we freeze ResNet50 layers below level 4 and train for 3K iterations. In the final stage, we freeze ResNet50 layers below level 3 for another 70K iterations. When switching to each stage, we decrease the learning rate by a factor of 10.

5.2. 6D Pose and Size Estimation

Our goal is to estimate the full metric 6D pose and dimensions of detected objects by using the NOCS map and input depth map. To this end, we use the RGB-D camera intrinsics and extrinsics to align the depth image to color image. We then apply the predicted object mask to obtain a 3D point cloud P_m of the detected object. We also use the NOCS map to obtain a 3D representation of P_n. We then estimate the scales, rotation, and translation that transforms the P_n to P_m. We use the Umeyama algorithm [47] for this 7 dimensional rigid transformation estimation problem, and RANSAC [16] for outlier removal. See C for qualitative results.

6. Experiments and Results

Metrics: We report results on both 3D object detection, and 6D pose estimation metrics. To evaluate 3D detection and object dimension estimation, we use the intersection over union (IoU) metric with a threshold of 50% [17]. For 6D pose estimation, we report the average precision of object instances for which the error is less than m cm for translation and $n°$ for rotation similar to [39, 30]. We decouple object detection from 6D pose evaluation since it gives a clearer picture of performance. We set a detection threshold of 10% bounding box overlap between prediction and ground truth to ensure that most objects are included in the evaluation. For symmetric object categories (*bottle, bowl*, and *can*), we allow the predicted 3D bounding box to freely rotate around the object's vertical axis with no penalty. We treat *mug* as a special category and make it symmetric when the handle is not visible since it is hard to judge its pose in such cases, even for humans. We use [52] to detect handle visibility for CAMERA data and manually annotate for real data.

Baselines: Since we know of no other methods for category-level 6D pose and size estimation, we built our own baseline to help compare performance. It consists of the Mask R-CNN network trained on the same data but without the NOCS map heads. We use the predicted instance mask to obtain a 3D point cloud of the object from the depth map. We align the masked point cloud to one randomly chosen model from the corresponding category. For the instance-level 6D pose estimation, we present results that can readily be compared with [51].

Evaluation Data: All our experiments use one or both of these evaluation datasets: (1) the CAMERA validation dataset (CAMERA25), and (2) a 2.75K real dataset (REAL275) with ground truth annotations. Since real data is limited, this allows us to investigate performance without entangling pose estimation and domain generalization.

6.1. Category-Level 6D Pose and Size Estimation

Test on CAMERA25: We report category-level results for our method with the CNN trained only on the 275K CAMERA training set (CAMERA*). We test performance on CAMERA25 which is composed of objects and backgrounds completely unseen during training. We achieve a mean average precision (mAP) of **83.9%** for 3D IoU at 50% and an mAP of **40.9%** for the (5°, 5 cm) metric. (5°, 5 cm) is a strict metric for estimating 6D pose even for known instances [51, 6, 37]. Figure 6 shows the average precision at different 3D IoU, rotation, and translation errors for all 6 object categories.

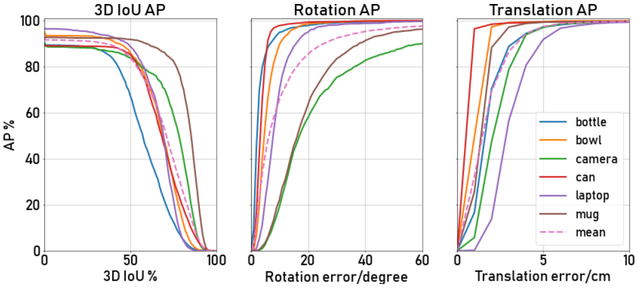


Figure 6. 3D detection and 6D pose estimation results on CAMERA25 when our network is trained on CAMERA*.

Test on REAL275: We then train our network on a combination of CAMERA*, the real-world dataset (REAL*), with weak supervision from COCO [33], and evaluate it on the real-world test set. Since COCO does not have ground truth NOCS maps, we do not use NOCS loss during training. We use 20K COCO images that contain instances in our categories. To balance between these datasets, for each minibatch we select images from the three data sources, with a probability of 60% for CAMERA*, 20% for COCO, and 20% for REAL*. This network is the best performing model which we use to produce all visual results (Figure 8).

In the real test set, we achieved an mAP of **76.4%** for 3D IoU at 50%, an mAP **10.2%** for the (5°, 5 cm) metric, and an mAP of **23.1%** for (10°, 5 cm) metric. In comparison, the baseline algorithm (Mask RCNN + ICP alignment) achieves an mAP of **43.8%** for 3D IoU at 50%,and an mAP of **0.8%** for both (5°, 5 cm) and (10°, 5 cm) metric, which is significantly lower than our algorithm's performance. Figures 7 shows more detailed analysis and comparison. This experiment demonstrates that by learning to predict the dense NOCS map, our algorithm is able to provide additional detailed information about the object's shape, parts and visibility, which are all critical for correct estimation the object's 6D pose and sizes.

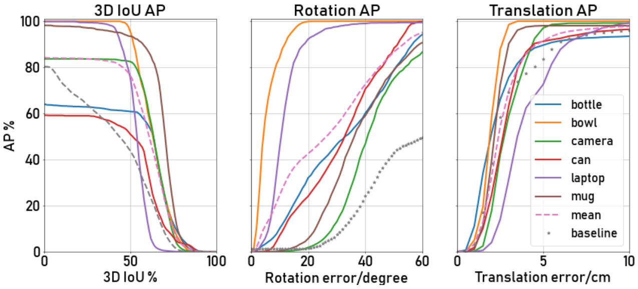


Figure 7. Result on REAL275 test set, average precision (AP) vs. different thresholds on 3D IoU, rotation error, and translation error.

6.2. Ablation Studies

CAMERA Approach: To evaluate our CAMERA data generation approach, we conducted an experiment with our network trained on different training data combinations. For this experiment, we set the network architecture to regress the NOCS maps. Table 1 shows the performance of our network on the REAL275 test set.

We also created a variant of CAMERA* where the images are composited in a **non-context aware** manner (denoted by **B** in Table 1). As shown in the table, using only CAMERA* results in poor performance due to domain gap. We see progressive improvements on adding COCO and REAL*. Training only on REAL*, or REAL* and COCO tend to overfit to the training data due to small dataset size. Training on CAMERA* with COCO and REAL* lead to the best results. Furthermore, we see that non-context aware data results in worse performance than context-aware data indicating that our CAMERA approach is useful.

Classification vs. Regression: On both CAMERA25 and REAL275, we observe that pixel classification is consistently better than regression. The network with 32 bins is better on pose estimation while the one with 128 bins is slightly better on detections (see Table 2).

Symmetry Loss: The symmetry loss is critical for many everyday object categories. To study the effect of symmetry loss, we conduct ablation experiments on the regression network on both CAMERA25 and REAL275 set. Table 2 shows that the pose accuracy degrades significantly, particularly for 6D pose, if the symmetry loss is not used.

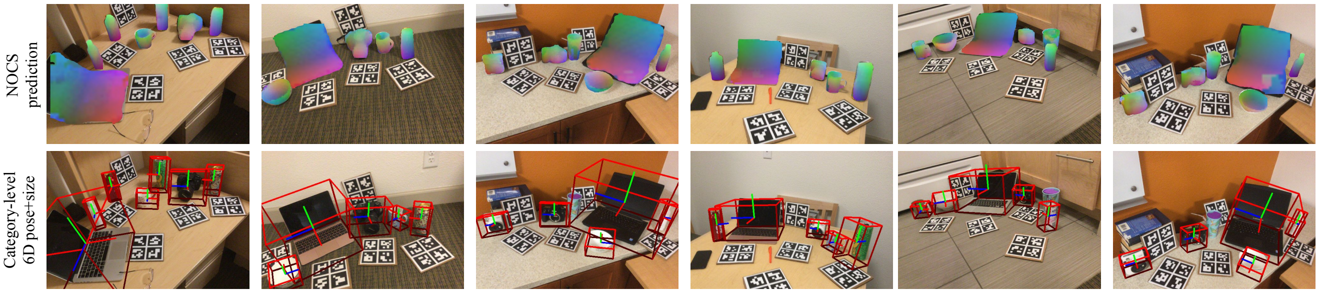


Figure 8. Qualitative result on REAL275 test set. Top row shows the predicted NOCS maps color coded. Bottom row shows the quality of 6D pose (axis) and size estimation (red tight bounding box).

Data			mAP				
CAMERA*	COCO	REAL*	$3D_{25}$	$3D_{50}$	5 ° 5 cm	10° 5 cm	10° 10cm
C			51.7	36.7	3.4	20.4	21.7
C	✓		57.6	41.0	3.3	17.0	17.1
		✓	61.9	47.5	6.5	18.5	18.6
	✓	✓	71.0	53.0	7.6	16.3	16.6
C		✓	79.2	69.7	6.9	20.0	21.2
C	✓	✓	**79.6**	**72.4**	**8.1**	**23.4**	**23.7**
B			42.6	36.5	0.7	14.1	14.2
B	✓	✓	79.1	71.7	7.9	19.3	19.4

Table 1. Validating CAMERA approach. **C** represents the unmodified CAMERA* data while **B** denotes a non-context aware version of CAMERA*. We report AP for 5 different metrics, where $3D_{25}$ and $3D_{25}$ represent 3D IoU at 25% and 50%, respectively.

Data	Network	mAP				
		$3D_{25}$	$3D_{50}$	5 ° 5 cm	10° 5 cm	10° 10cm
CAMERA25	Reg.	89.3	80.9	29.2	53.7	54.5
	Reg. w/o Sym.	86.6	79.9	14.7	38.5	40.0
	32 bins	91.1	83.9	**40.9**	**64.6**	**65.1**
	128 bins	**91.4**	**85.3**	38.8	61.7	62.2
REAL275	Reg.	79.6	72.4	8.1	23.4	23.1
	Reg. w/o Sym.	82.7	73.8	1.3	9.1	9.3
	32 bins	84.8	78.0	**10.0**	25.2	25.8
	128 bins	**84.9**	**80.5**	9.5	**26.7**	**26.7**

Table 2. Network architectures and losses. Reg. represents regression network trained with soft L^1 loss; 32 bins and 128 bins represent classification networks with the corresponding numbers of bins, respectively.

6.3. Instance-level 6D Pose Estimation

We also evaluate our method on instance-level 6D pose estimation task on OccludedLINEMOD [26] and compare with PoseCNN [51]. OccludedLINEMOD is a challenging dataset where 9 annotated objects are highly occluded in a cluttered scene.

The OccludedLINEMOD dataset has 9 object instances and provides a CAD model for each instance. It has 1214 images with annotated ground truth 6D pose. We follow the protocols from [46] [27] and randomly select 15% of the dataset as training images. Then we generate 15000 synthetic images using the technique described in Section 4.

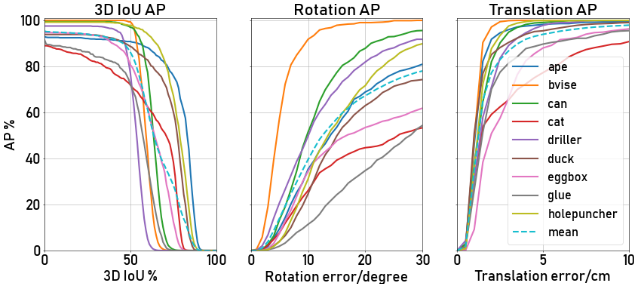


Figure 9. Result on OcculudedLINEMOD. Here we show the average precision (AP) vs. different thresholds on 3D IoU, rotation error, and translation error.

Using 32-bin classification network, we achieve a detection rate of 94.7%, an mAP of **88.4%** for 3D IoU at 50%, an mAP **13.9%** for the (5°, 5 cm) metric, and an mAP of **33.5%** for (10°, 5 cm) metric. This is substantially higher than PoseCNN[51] which only achieves an mAP of **1.7%** without iterative pose refinement (reported in [30]). Figure 9 provide more detailed analysis. This experiment demonstrates that while our approach designed for category-level pose estimation, it can also achieve state-of-the-art performance on standard 6D pose estimation benchmarks.

With 2D projection metric, which measures the average pixel distance between ground truth and estimated object poses, we achieve **30.2%** mAP on 2D projection at 5 pixel. Our method significantly outperforms PoseCNN[51] by a large margin, which reported 17.2% mAP on 2D projection at 5 pixel in [30].

The comparison between our method and other RGB-only methods with 2D projection metric is shown in Fig. 10. We only show the plots with 2D Projection metric because these are the only results reported in [37] and [46].

6.4. Limitations and Future Work

To our knowledge, ours is the first approach to solve the category-level 6D pose and size estimation problem. There are still many open issues that need to be addressed. First, in our approach, the pose estimation is conditioned on the region proposals and category prediction which could be incorrect and negatively affect the results. Second, our approach rely on the depth image to lift NOCS prediction to real-world coordinates. Future work should investigate estimating 6D pose and size directly from RGB images.

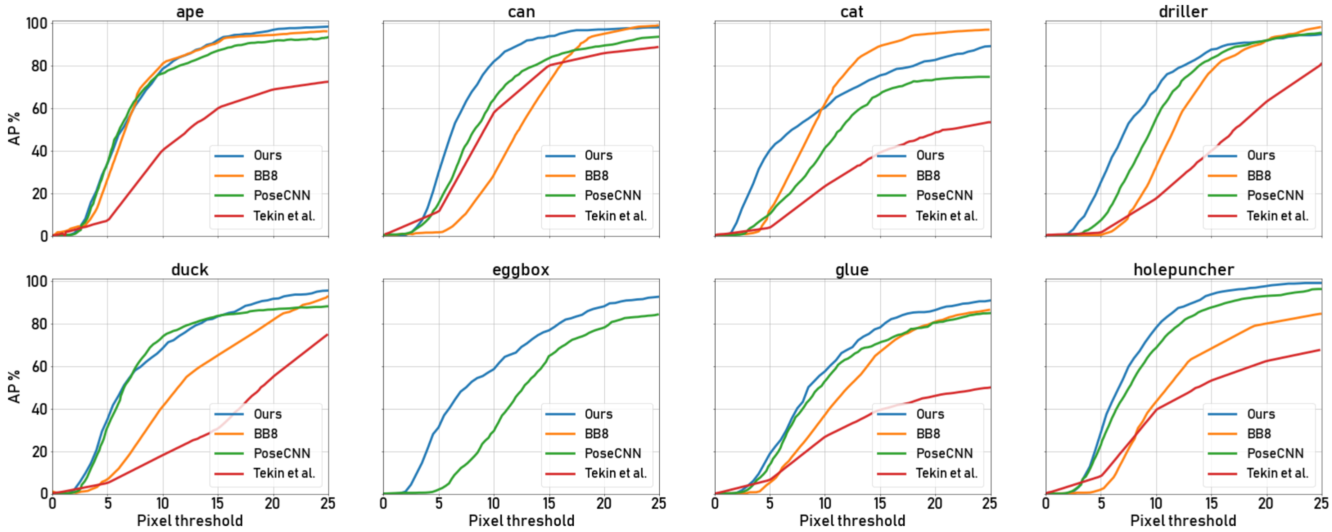


Figure 10. Comparison with state-of-the-art RGB or RGB-D based methods on the OccludedLINEMOD dataset[6].

7. Conclusion

We presented a method for category-level 6D pose and size estimation of previously unseen object instances. We presented a new normalized object coordinate space (NOCS) that allows us to define a shared space with consistent object scaling and orientation. We propose a CNN that predicts NOCS maps that can be used with the depth map to estimate the full metric 6D pose and size of unseen objects using a pose fitting method. Our approach has important applications in areas like augmented reality, robotics, and 3D scene understanding.

Acknowledgements

This research was supported by a grant from Toyota-Stanford Center for AI Research, NSF grant IIS-1763268, a gift from Google, and a Vannevar Bush Faculty Fellowship. We thank Xin Wang, Shengjun Qin, Anastasia Dubrovina, Davis Rempe, Li Yi, and Vignesh Ganapathi-Subramanian for helping with the object scans.

Appendix

A. Implementation and Computation Times

Our network is implemented on Python 3, Keras and Tensorflow. The code is based on MatterPort's Mask RCNN implementation[3]. The network uses Feature Pyramid Network (FPN)[32] and a ResNet50 backbone[25].

Our network takes images with a resolution of 640×360 as input. We achieve an interactive rate of around 4 fps on an Intel Xeon Gold 5122 CPU @ 3.60GHz desktop with a NVIDIA TITAN Xp. Our implementation takes an average time of 210 ms for neural network inference and 34 ms for pose alignment using Umeyama algorithm.

B. Scanned Real Instances

Our real dataset contains 6 object categories and 42 real scanned unique instances. For each category, we collect 7 instances with 4 for training and validation and the rest 3 for test. Figure 11 show a subset of our instances where one can see a large intra-category shape variance in the dataset. The first row are instances used in training. The second and third rows are held-out instances for testing.

C. Result Visualization

Here we provide more visual results of the 6D pose and size estimation. Due to sufficient training data, our method achieves very promising performance on CAMERA25 validation set as shown in Figure12. On REAL275 test set, we still observe decent performance even though the amount of real training data is small. We observe several failure modes on real data, including missing detection, wrong classification, and inconsistency in predicted coordinate maps.

References

[1] Structure sensor. https://structure.io/. 5

[2] Unity game engine. https://unity3d.com. 5

[3] W. Abdulla. Mask r-cnn for object detection and instance segmentation on keras and tensorflow. https://github.com/matterport/Mask_RCNN, 2017. 9

[4] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. In *PAMI*, 1992. 2

[5] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer, 2014. 2

[6] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold, et al. Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3364–3372, 2016. 1, 7, 9

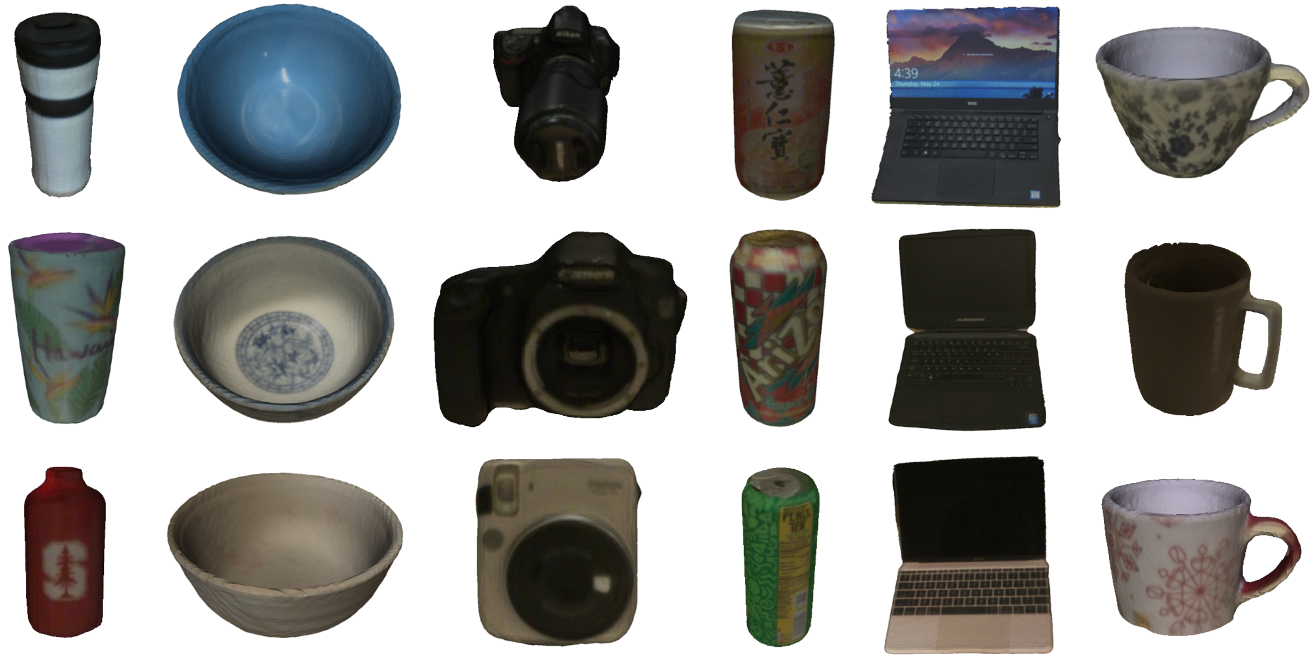


Figure 11. A subset of our 42 instances in our real dataset. From the left to the right column, they are: bottle, bowl, camera, can, laptop and mug. The first row shows instances that belong to training set. The second and third rows show instances that belong to test set.

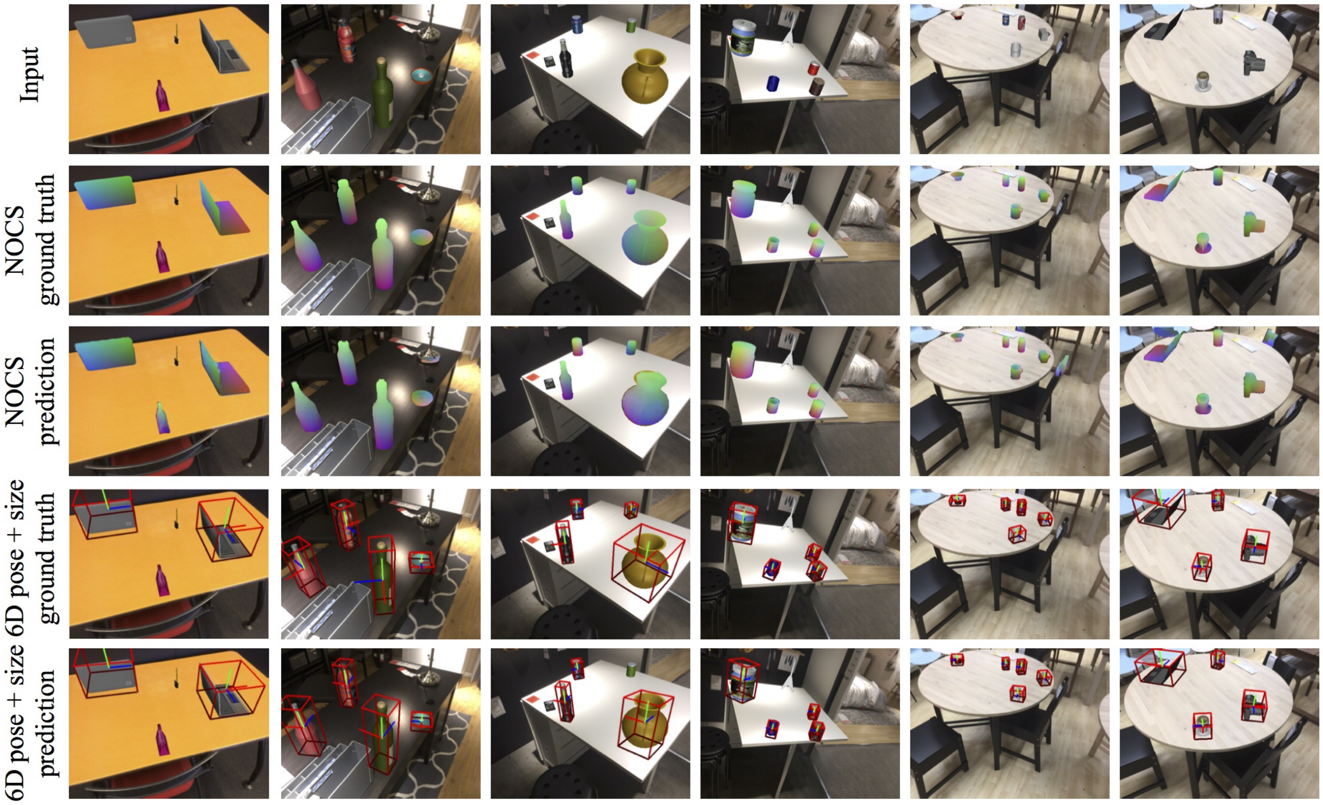


Figure 12. Qualitative result on CAMERA25 validationset set. The top row shows the input rgb image. The second and the third rows show the color-coded ground truth and predicted NOCS maps. The fourth and the fifth rows show the ground truth and predicted 6D pose (axis) and size estimation (red tight bounding box).

[7] M. Braun, Q. Rao, Y. Wang, and F. Flohr. Pose-rcnn: Joint object detection and pose estimation using 3d object proposals. In *Intelligent Transportation Systems (ITSC), 2016 IEEE 19th International Conference on*, pages 1546–1551. IEEE,

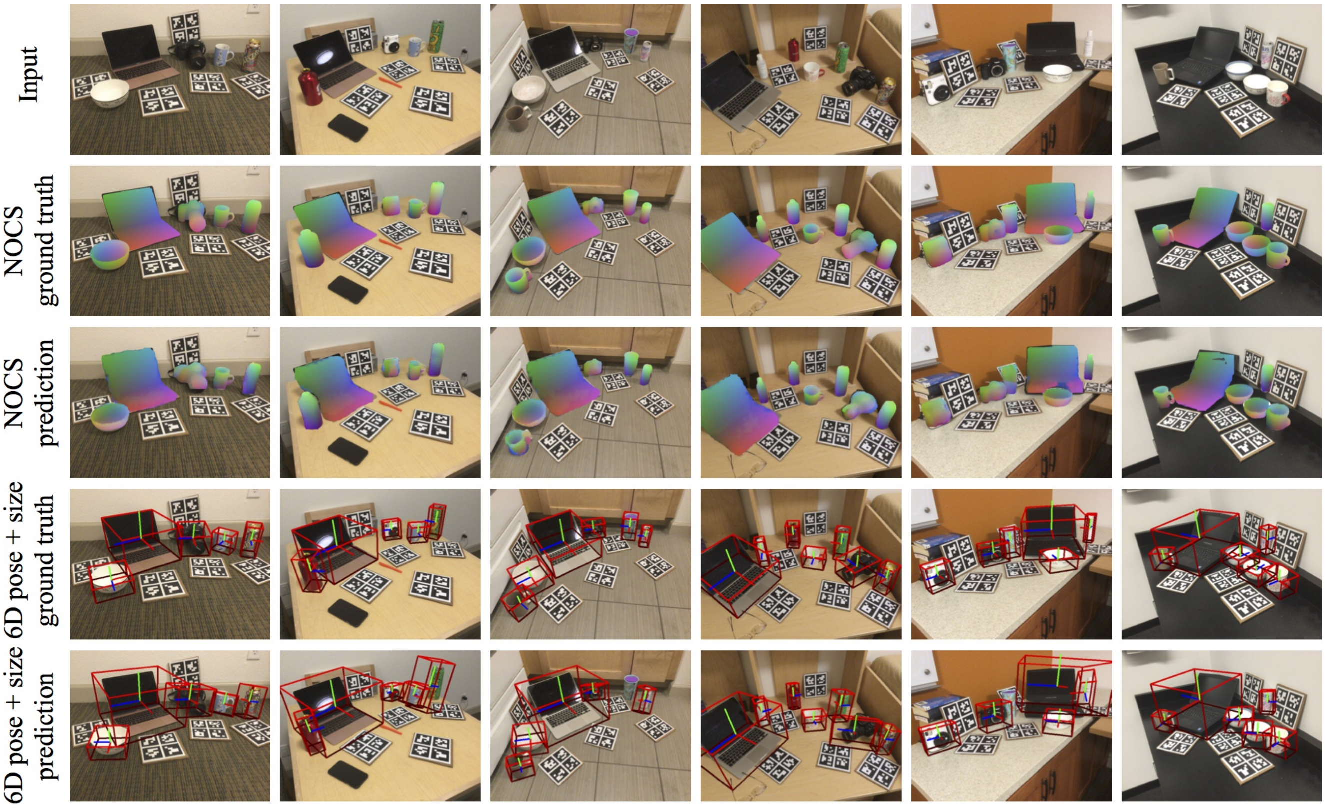


Figure 13. Qualitative result on REAL275 test set.

2016. 2

[8] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2, 3, 4

[9] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2147–2156, 2016. 1

[10] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia. Multi-view 3d object detection network for autonomous driving. In *IEEE CVPR*, volume 1, page 3, 2017. 2

[11] A. Collet, M. Martinez, and S. S. Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *IJRR*, 30(10):1284–1306, 2011. 2

[12] Z. Deng and L. J. 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)*, volume 2, page 2, 2017. 1

[13] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015. 3

[14] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1355–1361. IEEE, 2017. 2

[15] C. Feng, Y. Taguchi, and V. R. Kamat. Fast plane extraction in organized point clouds using agglomerative hierarchical clustering. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 6218–6225. IEEE, 2014. 4

[16] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *Readings in computer vision*, pages 726–740. Elsevier, 1987. 6

[17] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013. 6

[18] R. A. Güler, N. Neverova, and I. Kokkinos. Densepose: Dense human pose estimation in the wild. *arXiv preprint arXiv:1802.00434*, 2018. 2

[19] R. Guo. *Scene understanding with complete scenes and structured representations*. University of Illinois at Urbana-Champaign, 2014. 2, 3

[20] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3d models to rgb-d images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4731–4740, 2015. 2, 3

[21] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from RGB-D images. In *CVPR*, 2013. 2

[22] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *ECCV*, 2014. 2

[23] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2980–2988. IEEE, 2017. 4, 5, 6

[24] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 6

[25] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 6, 9

[26] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *ICCV*, 2011. 2, 8

[27] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1521–1529, 2017. 1, 8

[28] R. Kouskouridas, A. Tejani, A. Doumanoglou, D. Tang, and T.-K. Kim. Latent-class hough forests for 6 dof object pose estimation. *arXiv preprint arXiv:1602.01464*, 2016. 1

[29] J. Lahoud and B. Ghanem. 2d-driven 3d object detection in rgb-d images. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 4632–4640. IEEE, 2017. 2

[30] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. Deepim: Deep iterative matching for 6d pose estimation. *arXiv preprint arXiv:1804.00175*, 2018. 6, 8

[31] D. Lin, S. Fidler, and R. Urtasun. Holistic scene understanding for 3D object detection with RGBD cameras. In *ICCV*, 2013. 2

[32] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *CVPR*, volume 1, page 4, 2017. 9

[33] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 6, 7

[34] A. Mousavian, D. Anguelov, J. Flynn, and J. Košecká. 3d bounding box estimation using deep learning and geometry. In *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, pages 5632–5640. IEEE, 2017. 1

[35] J. Papon and M. Schoeler. Semantic pose using deep networks trained on synthetic rgb-d. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, pages 774–782. IEEE, 2015. 2, 3

[36] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas. Frustum pointnets for 3d object detection from rgb-d data. *arXiv preprint arXiv:1711.08488*, 2017. 1, 2

[37] M. Rad and V. Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *International Conference on Computer Vision*, 2017. 1, 7, 8

[38] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015. 5

[39] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2930–2937. IEEE, 2013. 2, 6

[40] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 1, 3, 4

[41] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR*, volume 5, page 6, 2015. 1, 3, 4

[42] S. Song and J. Xiao. Sliding Shapes for 3D object detection in depth images. In *ECCV*, 2014. 2, 3

[43] S. Song and J. Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 808–816, 2016. 1, 2

[44] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1746–1754, 2017. 3

[45] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 103–110. IEEE, 2012. 2

[46] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6d object pose prediction, 2017. 1, 8

[47] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on pattern analysis and machine intelligence*, 13(4):376–380, 1991. 6

[48] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4400–4408, 2015. 2

[49] Y. Xiang, W. Choi, Y. Lin, and S. Savarese. Data-driven 3d voxel patterns for object category recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1903–1911, 2015. 1

[50] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *Applications of Computer Vision (WACV), 2014 IEEE Winter Conference on*, pages 75–82. IEEE, 2014. 3

[51] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017. 1, 3, 7, 8

[52] L. Yi, V. G. Kim, D. Ceylan, I. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, L. Guibas, et al. A scalable active framework for region annotation in 3d shape collections. *ACM Transactions on Graphics (TOG)*, 35(6):210, 2016. 6

[53] A. Zeng, K.-T. Yu, S. Song, D. Suo, E. Walker, A. Rodriguez, and J. Xiao. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 1386–1383. IEEE, 2017. 2

[54] Y. Zhang, M. Bai, P. Kohli, S. Izadi, and J. Xiao. DeepContext: Context-encoding neural pathways for 3D holistic scene understanding. In *ICCV*, 2017. 2

[55] Y. Zhou and O. Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object detection. *arXiv preprint arXiv:1711.06396*, 2017. 2