Learning Canonical Shape Space for Category-Level 6D Object Pose and Size Estimation

Dengsheng Chen[1,*] Jun Li[1,*] Zheng Wang[3] Kai Xu[1,2,†]
[1]National University of Defense Technology [2]SpeedBot Robotics Ltd. [3]Taobao.com

Abstract

We present a novel approach to category-level 6D object pose and size estimation. To tackle intra-class shape variations, we learn canonical shape space (CASS), a unified representation for a large variety of instances of a certain object category. In particular, CASS is modeled as the latent space of a deep generative model of canonical 3D shapes with normalized pose. We train a variational auto-encoder (VAE) for generating 3D point clouds in the canonical space from an RGBD image. The VAE is trained in a cross-category fashion, exploiting the publicly available large 3D shape repositories. Since the 3D point cloud is generated in normalized pose (with actual size), the encoder of the VAE learns view-factorized RGBD embedding. It maps an RGBD image in arbitrary view into a pose-independent 3D shape representation. Object pose is then estimated via contrasting it with a pose-dependent feature of the input RGBD extracted with a separate deep neural networks. We integrate the learning of CASS and pose and size estimation into an end-to-end trainable network, achieving the state-of-the-art performance.

1. Introduction

6D object pose estimation based on a single-view RGB(D) image is an essential building block for several real-world applications ranging from robotic navigation and manipulation to augmented reality. Most existing works have so far been addressing *instance-level* 6D pose estimation where each target object has a corresponding CAD model with exact shape and size [17]. Thereby, the problem is largely reduced to finding sparse or dense correspondence between the target object and the stock 3D model. Pose hypotheses can then be generated and verified based on the correspondences. Although enjoying high pose accuracy, the requirement of exact CAD models by these techniques hinders their practical use in many application scenarios.

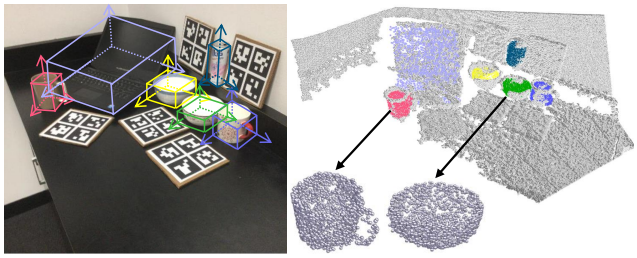

Figure 1: We present a method for category-level 6D object pose and size estimation (left) via learning canonical shape space. The input RGBD image is embedded into the shape space, resulting in a view-factorized RGBD embedding. Object pose is then estimated via contrasting it with a pose-dependent feature of the input RGBD. A side-product of our method is the full-shape reconstruction of the input single-view RGBD, which supports not only size calculation but also precise robotic grasping. In the figure, the reconstructed 3D point clouds are placed into the scene point cloud (unprojected from the input depth map). Two of them are highlighted in the middle.

Recently, *category-level* 6D object pose estimation starts to gain attention [21,29]. In this problem, the target object of a shape category is unseen before and no CAD model is available, although some other instances of the same category may have been seen. Therefore, the major challenge is how to deal with *intra-class variation* [20]. In general, household objects could exhibit significant variations in color, texture, shape and size even within the same category. Without an exactly same CAD model, correspondence-based approach would find difficulty given the considerable intra-class shape variation.

To resolve this, a *unified representation* for a variety of instances of an object category is needed, to which the target object in observation could be "matched". The recently proposed Normalized Object Coordinate Space (NOCS) [29] is a nice example of such unified representation. Based on this representation, category-level object poses can be estimated with high accuracy through mapping each pixel of the in-

*Joint first authors
†Corresponding author: kevin.kai.xu@gmail.com

put image to a point in NOCS. Finding *dense* mappings between NOCS and an *unseen* object, however, is an ill-posed problem under significant shape variation. Therefore, a generalizable mapping function accommodating large amount of unknown shape variants can be difficult to learn.

In this work, we propose to learn a *canonical shape space (CASS)* as our unified representation. CASS is modeled by the latent space of a deep generative model of canonical 3D shapes with normalized pose and actual metric size. In particular, we train a variational auto-encoder (VAE) for generating 3D point clouds in the canonical space from an RGBD image. The VAE is trained in a cross-category fashion, exploiting the publicly available large 3D shape repositories. Since the 3D point cloud is generated with normalized pose and metric size, the encoder of the VAE learns *view-factorized* RGBD embedding. It maps an RGBD image in arbitrary view into a *pose-independent* 3D shape representation. Object pose can then be estimated via contrasting it with a *pose-dependent* feature of the input RGBD extracted with a separate deep neural networks (Figure 1). This circumvents the difficulty in estimating dense correspondence between two representations as in other methods [17, 29].

We integrate the learning of CASS and pose and size estimation into an end-to-end trainable network which involves several key designs. *First*, through learning the canonical shape space with plentiful shape variants, we obtain a unified representation encompassing adequate shape variations. *Second*, to overcome the lack of real-world training images with 3D point clouds, we enhance the encoder of the VAE to take both RGBD images and 3D shapes as input. This allows us to train the VAE through exploiting off-the-shelf 3D shape repositories. *Third*, in realizing pose estimation, we opt for feature contrasting over dense correspondence, leading to better generality to unseen instances. Meanwhile, to match the distributions of pose-dependent and pose-independent features so that pose estimation can be easily trained, we propose a few crucial designs, e.g., network weight sharing and training batch mixing. *Last*, our VAE model is able to reconstruct a 3D point cloud of the target object with metric size, which reduces the learning difficulty by decoupling the estimation of pose and size.

Through evaluating on public category-level datasets, we show that our method archives the state-of-the-art pose accuracy and comparably high size accuracy. Our work makes the following contributions:

- We propose a novel correspondence-free approach to category-level object pose and size estimation based on learned canonical shape space.
- We design an end-to-end trainable deep neural network for jointly learning the canonical shape space and estimating object pose and size.
- We devise several key designs to ease the network training such as distribution matching between pose-dependent and pose-independent features.

2. Related Work

Instance-level approaches. Many works on instance-level 6D pose estimation adopt template-based methods [11, 15]. In these methods, a set of RGB(D) templates rendered from CAD models in various poses are matched against the input image in a sliding-window fashion, based on hand-designed or learned feature descriptors. The final pose is retrieved from the best matched template or estimated by 3D model registration. Another group of works pursue to match the target object to the corresponding 3D model. Depending on the input modality, the core task is to find 2D-to-3D [8, 12], 2.5D(depth)-to-3D [5, 6] or 3D-to-3D correspondence [1]. Several other works opt to learn a 6D object pose regressor directly from the feature descriptions [22, 25]. Brachmann et al. [3] learn to regress an object coordinate representation which can then be used in pose estimation.

Learning effective feature representation using convolutional neural networks (CNNs) for robust matching has become a main focus of the recent literature [2, 14, 24, 30, 33]. Another line of works utilize CNNs to detect feature points or corner points [19, 26]. PVNet [17] is a unique approach of feature point detection using CNNs: A vector field is estimated for the input RGB image based on which the feature points are voted. Some other works choose to learn an end-to-end deep model that can directly regress 6D object pose from the raw RGB(D) input [7, 16, 31]. SSD-6D [13] combines single-shot object detection in RGB images with pose hypothesis regression and verification. Similar approach has also been used for multi-view active pose estimation [23]. Wang et al. [28] propose DenseFusion to learn pixel-wise feature extraction and pose estimation. The network can also predict a confidence for each pose hypothesis for final pose selection.

Xiao et al. [32] train a CNN that takes as input both an image and a CAD model, and outputs object pose with respect to the 3D model. This model can generalize to the target objects which are unseen during training. However, they still require the target CAD model during inference. Thus, we classify it as an instance-level work.

Category-level approaches. There has been a large body of works on category-level object detection and 2D/3D/4D pose estimation. However, methods designed for estimating *6D poses* is still scarce [20]. Sahin et al. [21] introduce a part-based random forest approach for this task. In their method, parts extracted from CAD model instances of some category are represented with skeletons, which are

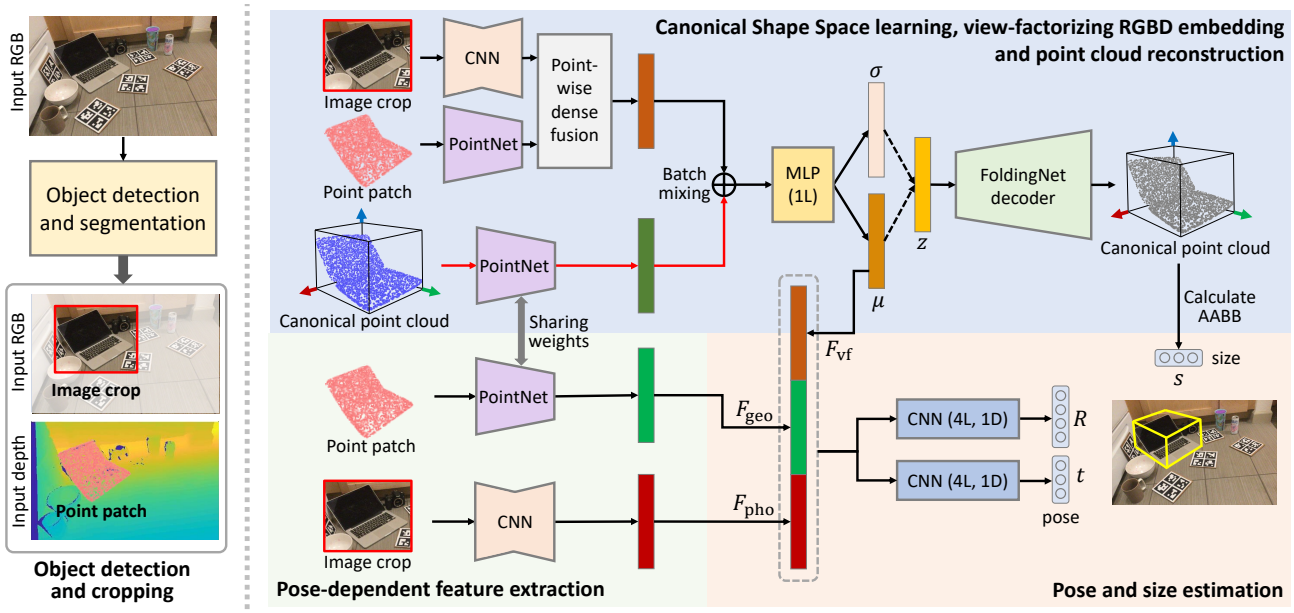

Figure 2: An overview of our network architecture. A pre-processing (left) is devised to produce image crop and point patch of the object of interest which are fed into the main network (right). The main network is composed of three modules: 1) CASS learning, pose-factorizing embedding and point cloud reconstruction (background shaded in light blue), 2) pose-dependent feature extraction (light green), and 3) pose estimation (light red). The network branch indicated with red arrows is used only in training.

fed into a random forest for hypothesizing 6D poses. Due to the reliance on purely geometric features, this method mainly deals with depth input. Wang et al. [29] introduce Normalized Object Coordinate Space (NOCS) as a shared canonical representation of object instances within a category. They train a region-based neural network to directly infer the pixel-wise correspondence between an RGB image to NOCS. Together with instance mask and depth map, 6D object pose is estimated using shape matching. Recently, Wang et al. [27] realized category-level 6D pose tracking based on keypoint matching.

CASS vs. NOCS. Although both CASS and NOCS can be regarded as a unified shape space spanning intra-class variations, there are several substantial differences. *First*, NOCS is explicitly defined through consistently aligning all object instances of a category in a normalized 3D space. Our CASS is a shape embedding space implicitly learned with a generative model. *Second*, when conducting object pose estimation, NOCS is used as the target of pixel-wise correspondence, based on which 6D pose is computed geometrically. In contrast, CASS is treated as a normalized, holistic shape representation from which pose is estimated in an end-to-end and *correspondence-free* manner. *Third*, different from NOCS where the coordinates are regressed only for visible area, our network learns to reconstruct a complete 3D shape in CASS which is a global shape understanding beneficial to pose estimation.

3. Model

Our model is an end-to-end trainable network integrating the learning of both shape space and pose estimation. We first provide an overview of the network architecture and then elaborate the various network modules. Training details such as loss functions, parameter setting and training protocol will then follow.

Architecture overview. Figure 2 shows an overview of our network architecture. The input to the network is a calibrated RGBD image. The one output is the 6DoF pose of the object of interest, represented by a rigid transformation $[R|t]$ with $R \in SO(3)$ and $t \in \mathbb{R}^3$. The other output is a reconstructed 3D point cloud of the object in normalized pose but with metric size. To handle multiple objects in a cluttered scene, we employ an off-the-shelf object detector to detect and segment the individual object instances. For each detected object, we crop the RGB image with the bounding box of the segmentation mask and segment it out of the point cloud (converted from the depth image) using the mask, resulting in an image crop and a point patch for the object, respectively. Both the image crop and the point patch are sent to the main part of our network.

Our core network is composed of three modules responsible for 1) Canonical Shape Space learning, view-factorizing RGBD embedding and point cloud reconstruction, 2) pose-dependent feature extraction and 3) pose estimation, respec-

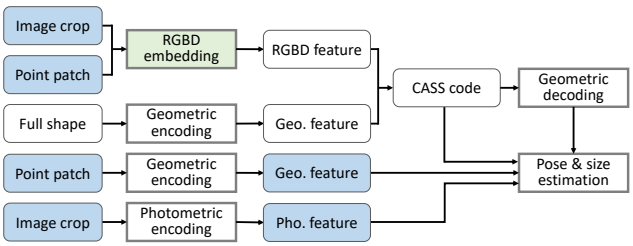

Figure 3: Illustrating the flow of pose information in our network. Data and features are depicted with rounded boxes and networks with rectangles. Data/features shaded in light blue contain pose information; no shading means pose-normalized. Networks are shaded in light green if they are pose factorizing and no shading otherwise.

tively. The three components are tightly coupled and jointly trained using both synthetic and real-world data. Next, we elaborate the design of the three modules.

3.1. Canonical Shape Space and View Factorization

Our goal is to learn a shape space spanning as many shape variants of a category as possible, where all shapes are pose-normalized but with actual metric size. Moreover, to work with RGBD inputs, we also need a function to map an RGBG image to the point in that space representing the corresponding full shape in normalized pose and metric size. Such a mapping factorizes the view in the RGBD image so that the RGBD feature embedding is view-factorized.

Learning Canonical Shape Space. We model the space of canonical shapes with the latent space of a deep generative model of pose-normalized shapes. In achieving so, we leverage the publicly available 3D shape repositories such as ShapeNet [4]. The 3D models in ShapeNet are consistently oriented and properly scaled within each category. We sample each model into a point cloud of $M = 500$ points. The point sampled 3D shapes, X_{3D}, are used to train a variational auto-encoder (VAE). The encoder employs the geometric embedding network for 3D point clouds proposed in [28], which is a variant of PointNet [18]. Based on the learned feature, the decoder warps a point cloud of 3D ellipsoid to match the shape of the input point cloud. We turn the auto-encoder into a VAE by adding a sampling layer between the encoder and decoder. The learned posterior distribution $z \sim p(z|X_{\text{3D}})$ models the space of canonical shapes.

Learning view-factorizing RGBD embedding. Having learned the CASS, our next task is to project an RGBD image in arbitrary view to the space so that the projector functions as view factorization. Such cross-modality data projection task could be finished with the help of data correspondence between the two modalities [9], where metric loss is used to optimize the projector. Let us refer to this solution as *correspondence-based projection*. This approach can be adapted to VAE straightforwardly where a projector is trained to map data in one modality to the latent space learned for another modality based on cross-modality data correspondence. However, we found that this method leads to suboptimal point cloud reconstruction and pose estimation due to 1) possibly incorrect correspondences and 2) the compromise between the metric loss and other losses.

To address these issues, we opt for a *joint embedding* approach. Specifically, we learn a VAE which has two encoders mapping both RGBD images and 3D point clouds to a shared latent space. Whilst the 3D encoder adopts PointNet, the RGBD encoder employs the dense fusion architecture proposed in [28] (we use the global feature for the whole image instead of the pixel-wise features). Our crucial design is that the two encoders, albeit having different network architectures, are trained with *mixed training batch* and *shared training gradients*. The latter means that gradients computed for either modality are back-propagated to tune both encoders. Through such mixed training, the learned shared latent space spans the joint feature space of the both modalities.

Compared to correspondence-based approaches, our joint embedding has the following advantages: *Firstly*, our model can be trained in a correspondence-free or unpaired fashion. This means the two modalities do not have to share object instances: It is unnecessary for the object in an RGBD image to have a corresponding 3D model in the training shape set. *Secondly*, our model introduces no extra loss function other than the basic ones of conventional VAEs. *Thirdly* and most importantly, the mixed training of the two encoders help to match the feature distributions of the two data modalities (see Figure 4), leading to better model generality and domain transferability.

In summary, the learning of the CASS and the RGBD feature embedding (denoted by F_{vf}) optimizes the following loss functions:

$$L_{\text{CASS}} = L_{\text{recon}}(X_{\text{3D}}, X^{\text{R}}_{\text{3D}}) + L_{\text{recon}}(X^{\text{R}}_{\text{rgbd}}, X^{*}_{\text{3D}}) + L_{\text{KL}}, \tag{1}$$

where L_{recon} and L_{KL} are the reconstruction loss and KL divergence loss, respectively. X_{3D} and X^{R}_{3D} are the input and reconstructed 3D point clouds, and $X^{\text{R}}_{\text{rgbd}}$ and X^{*}_{3D} the 3D point cloud reconstructed from the input RGBD and the corresponding ground-truth, respectively. We use Chamfer distance to measure reconstruction loss. During test, the 3D point cloud encoder (corresponding to the network branch with red arrows in Figure 2) is discarded and only the RGBD encoder is used for feature extraction.

The RGBD encoder factorizes image view, resulting in pose-independent RGBD feature (CASS code). However,

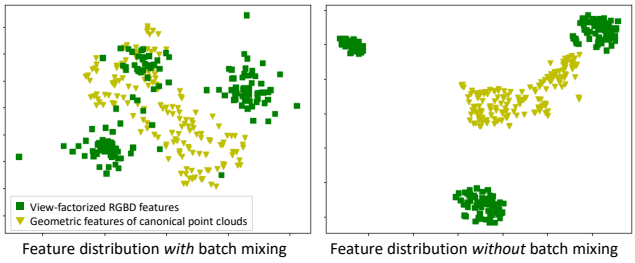

Figure 4: Comparing the t-SNE plot of view-factorized RGBD features and the geometric features of canonical point clouds for with and without batch mixing. Batch mixing helps to match the distributions of the two features.

the 3D encoder does *not* factorize object pose or size. This is because both the input and output of the 3D encoder are pose-normalized and metrically sized. It simply maps a pose-normalized shape to the canonical shape space, without processing on its pose or size. Figure 3 gives an illustrative summary of the view/pose-factorization ability for all network modules, and the pose-dependency of all involved data and features. Figure 5 (top row) shows the t-SNE visualization of the two feature embeddings. In the plot of view-factorized RGBD features, objects are clustered by category with different poses mixed together, indicating the factorization of view. The plot of geometric features of canonical point clouds (no pose) also demonstrates category-based clustering effect.

3.2. Pose-Dependent Feature Extraction

To facilitate pose estimation from an input RGBD image, we also extract pose-dependent features for the RGBD image. We devise two networks for extracting photometric and geometric features separately, based on the RGB and the depth images, respectively. In our network, these features are used for pose estimation through comparing against pose-dependent features, they are expected to encode the information of pose-color and pose-geometry correlations, respectively. Figure 5 (bottom row) shows the t-SNE plots of the two features, which both exhibit pose-induced subspace clustering effect.

Photometric feature extraction. Given the image crop containing the object of interest, we train a fully convolutional network that processes the color information into a color feature F_{pho}. Similar to [28], the image embedding network is an auto-encoder architecture that maps an image of size $H \times W \times 3$ to a pixel-wise feature map $H \times W \times N$. Each pixel has a N-dimensional vector. We then perform an average pooling over all pixel-wise features, obtaining a N-dimensional feature for the full image.

Geometric feature extraction. Given the corresponding point patch, we utilize point-based CNNs to extract an N-dim geometric feature F_{geo}. Here, a key design is that this point-based feature extractor can share the same network of the PointNet-based geometric feature encoder trained for CASS learning. As mentioned above, the geometry encoder is *not* pose-factorizing. Consequently, it can be used to extract pose-dependent geometric features. Consequently, we have a Siamese network of PointNet-based encoders, one for pose-*independent* CASS embedding and the other for pose-*dependent* geometric feature extraction (see Figure 2). Having these two tasks share network weights reduces the amount of parameters to be learned. Furthermore, it helps to match the distributions of the CASS codes and the geometric features. This makes them more comparable in facilitating feature-comparison-based pose estimation.

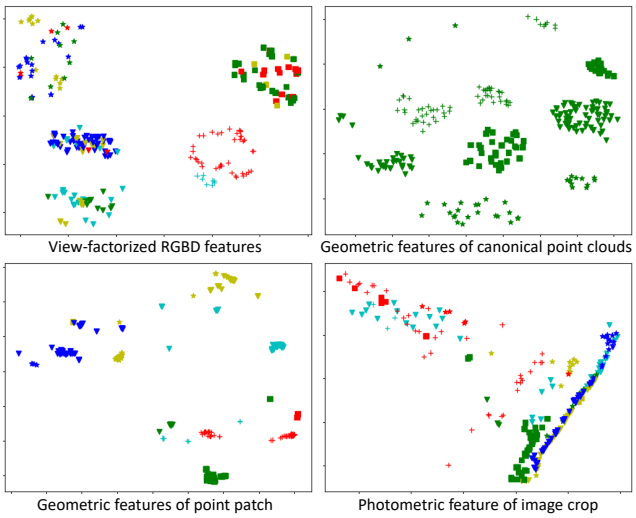

Figure 5: Comparing t-SNE visualization of the various features involved in our network. Different symbols indicate different object categories while distinct colors correspond to different poses.

3.3. Pose and Size Estimation

We concatenate F_{vf}, F_{pho} and F_{geo} into a feature vector of $3N$ length and then feed it into a CNN with 1D convolutions. The output contains a rotation represented by a quaternion q and a 3D translation vector t. The loss function for pose prediction is defined as the discrepancy between the object point clouds transformed by the ground-truth pose and by the predicted one:

$$L_{\text{pose}} = \frac{1}{M} \sum_{i} \|(Rx_i + t) - (R^* x_i + t^*)\|, \qquad (2)$$

where x_i is the i-th point of the $M = 500$ sampled points for the object. $[R^*|t^*]$ and $[R|t]$ are the ground-truth and predicted poses, respectively. To handle the alignment ambiguity of symmetric objects, we relax the point-wise

matching loss to Chamfer distance, similar to [28]. Object size is calculated as the dimension of the axis-aligned bounding box (AABB) of the reconstructed 3D point cloud.

3.4. Training Details

Network settings. The input to our method is a 640×480 RGBD image. With the RGB image, we perform object detection and segmentation. Any off-the-shelf method can be used. For example, we utilize Mask-RCNN [10] for the CAMERA dataset. The image crops do not need to be resize as they are fed into pixel-wise CNNs. All point patches and 3D models are re-sampled into 500 points for PointNet feature encoding. The dimensionality of CASS code and all other features is $N = 1024$. The DenseFusion and FoldingNet modules involved in the various network components use the same network configuration as the original works. The configuration of all other network modules such as CNNs and MLPs are given in Figure 2 (e.g., "4L" means four layers and "1D" means 1D convolutional layers). For each convolutional layer in the various modules, we add a Batch Normalization layer followed by an ReLU nonlinearity. See more details in the supplemental material.

Training protocol. We adopt a three-stage training. The *first* stage trains the VAE for CASS learning and view-factorizing RGBD embedding (the part shaded in light blue in Figure 2) for 80K iterations. The size of mixed batch is 8 which randomly mixes the training data of RGBD encoding and 3D encoding. In the *second* stage, we fix the VAE and jointly train pose-dependent feature extraction (the light green part) and pose estimation (the light red part) for 80K iterations. The *third* stage then jointly fine-tunes all parts for 40K iterations. All training batch has the size of 8. We use an initial learning rate of 0.0001 and the ADAM optimizer ($\beta_1 = 0.9$ and $\beta_2 = 0.999$) with a 1×10^{-6} weight decay. In each stage, we decrease the learning rate by a factor of 10 for every 40K iterations.

4. Results and evaluations

In this section, we aim to answer the following questions with both qualitative and quantitative evaluations. 1) Whether are the various network modules and design choices necessary? 2) How does our method perform in terms of pose accuracy and when does it outperform the state-of-the-arts? 3) How capable is our network in terms of single-view shape reconstruction?

4.1. Datasets

We use the datasets from NOCS [29] which contains six categories: *bottle*, *bowl*, *camera*, *can*, *laptop*, and *mug*.

Table 1: Quantitative comparison with NOCS [29] (we use its best-performing variant, i.e., 32-bin NOCS map classification).

Method	mAP				
	IoU_{25}	IoU_{50}	5° 5cm	10° 5cm	10° 10cm
NOCS	**84.4**	**79.3**	16.1	43.7	43.1
Ours	84.2	77.7	**23.5**	**58.0**	**58.3**

The dataset has two parts: a *real-world dataset* with 4.3K RGBD images from 7 scene videos (3 instances per category) and a *synthetic dataset* with 275K rendered images generated with 1085 model instances from ShapeNetCore [4] under random views. We evaluate our method on the NOCS-REAL275 dataset, which contains 2.75K real scene images with 3 unseen instances per category. In learning the CASS, we also utilized the 3D models from the ShapeNetCore dataset.

4.2. Evaluation Metrics

We follow the evaluation metrics in NOCS [29] which jointly measure the object detection and pose estimation:

- **IoU25 & IoU50**: the average precision of object instances for which the 3D overlap between the two bounding boxes is larger than 25% or 50% under predicted and ground truth poses respectively;
- **5°5cm, 10°5cm & 10°10cm**: the average precision of object instances for which the the error is less than n° for rotation and m cm for translation. We choose 5°5cm, 10°5cm, 10°10cm similar to [29].

Additionally, we employ the Chamfer Distance (CD) and Earth Mover's Distance (EMD) to evaluate shape reconstruction from single-view RGBD images.

4.3. Evaluation on NOCS-REAL275 Dataset

Category-level pose and size estimation. In Table 1, we compare our method against NOCS [29], which is the state-of-the-art method for category-level 6D object pose and size estimation. In their method, the network is trained to find a normalized coordinate for each pixel and then solve for the pose and size with the help of depth map. On the contrary, our method directly regresses the 6D pose by comparing pose-independent and pose-dependent features. We report the results of NOCS with 32 pose classification bins, which is its best-performing variant. Like NOCS, our results were *not* post-processed, e.g., by ICP refinement, although that

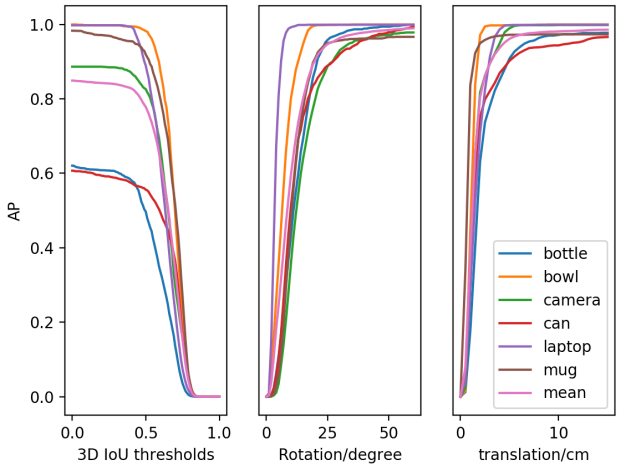

Figure 6: Result on NOCS REAL275 test dataset, average precision (AP) vs. different thresholds on 3D IoU, rotation error, and translation error.

is potentially facilitated by our point cloud reconstruction. The results show that our method outperforms NOCS in all metrics except the IoU metrics. The slightly lower IoU values are caused by our less accurate size calculation based on point cloud reconstruction. Direct regression of 6D pose is a hard problem. The success of our method is mainly attributed to the strong view-factorized (pose-independent) feature learning with the help of CASS learning and its RGBD embedding. Figure 6 shows more detailed analysis with category-wise plots of the various evaluation metrics.

Shape reconstruction. Table 2 reports a quantitative evaluation of 3D point cloud reconstruction from RGBD input, based on the test set of NOCS-REAL275. From the table, batch mixing leads to much higher reconstruction accuracy in terms of both Chamfer Distance (CD) and Earth Mover's Distance (EMD) metrics. This is because batch mixing ensures the distributions matching between the RGBD embedding and canonical point cloud embedding. This leads to a more accurate RGBD projection (embedding) into the latent shape space.

4.4. Ablation Studies

To experimentally justify the various design choices of our method, we make the following ablations (or their combinations) to our model:

- **w/o CASS**. Train the pose & size estimation network without the CASS code as an input.
- **w/o Distribution Matching (DM)**. Replace the Siamese network with two independent modules, without matching the distribution of the CASS codes and the geometric features.

Table 2: Evaluation of point cloud reconstruction accuracy with CD ($\times 10^{-3}$) and EMD metrics. The results show that method with batch mixing achieves uniformly higher reconstruction accuracy.

	w/o Batch Mixing		w/ Batch Mixing	
	CD	EMD	CD	EMD
bottle	1.71	0.24	0.75	0.04
bow	0.93	0.07	0.38	0.04
camera	5.26	0.22	0.77	0.05
can	1.79	0.20	0.42	0.04
laptop	1.94	0.10	3.73	0.09
mug	2.40	0.11	0.32	0.03
overall	2.33	0.16	1.06	0.05

Table 3: Ablation study of our model. The results show that our full method works the best for most criteria.

Method	mAP				
	IoU_{25}	IoU_{50}	5° 5cm	10° 5cm	10° 10cm
w/o CASS	83.8	76.2	4.2	29.5	30.0
w/o BM	83.6	77.3	4.7	31.8	32.7
w/o DM	84.0	**79.0**	8.4	39.5	40.2
w/o VAE	83.7	77.0	17.0	42.1	43.6
Full	**84.2**	77.7	**23.5**	**58.0**	**58.3**

- **w/o Batch Mixing (BM)**. Remove batch mixing and use L_2 distance as an additional loss to train the projection from an RGBD image in arbitrary view to the canonical shape space.
- **w/o VAE**. Replacing the VAE with AE.

From the results reported in Table 3, we can see that CASS learning is the most important component for our method. Without CASS learning, the accuracy drops the most especially on the $xx°yy$-cm metrics. Next to CASS learning, batch mixing is also very important factors. VAE is beneficial to model generalization to unseen objects since it helps learning a well-spanned CASS space with the normal distribution prior. However, it does lead to blurred 3D reconstruction at the same time, which may sacrifices size accuracy (see the IoU comparison in Table 1). Nevertheless, all factors together contributes the high-precision ($5°5$cm) estimation of pose and size.

4.5. Qualitative results

Visual results of pose and size estimation. Figure 7 shows some visual comparisons between our method and NOCS. According to the estimate pose and scale, we draw orien-

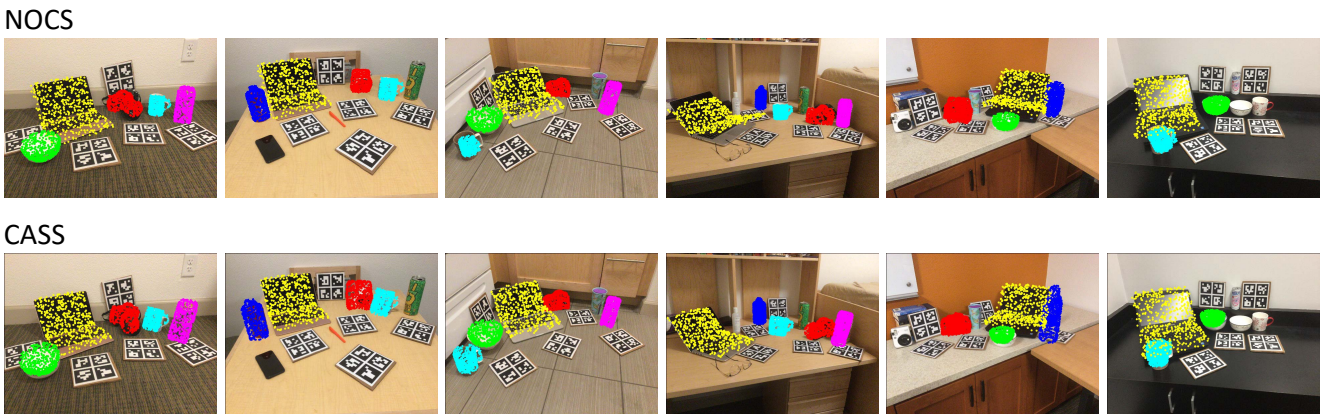

Figure 7: Qualitative comparison with NOCS [29] for pose and size estimation (depicted with reconstructed point clouds).

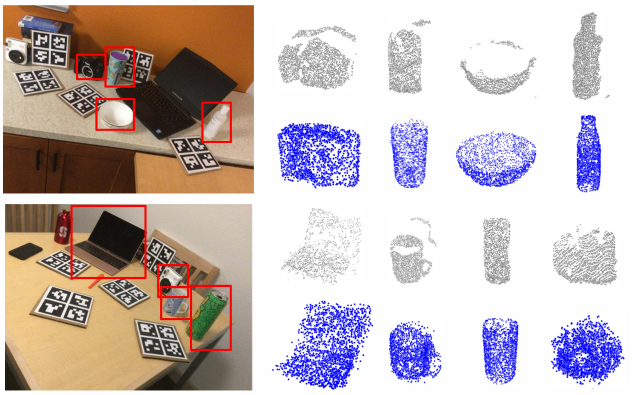

Figure 8: 3D Reconstruction from single-view RGBD. The point clouds in grey color are unprojected from depth maps. The blue point clouds are reconstructed by our network.

tated bounding box for each detected instance overlaid on top of the input RGB images. As can be observed, our method achieves better accuracy especially for size estimation under object occlusion and background distraction.

Visual results of shape reconstruction. Figure 8 shows visual results of shape reconstruction. Our method is able to reconstruct full 3D shapes in point cloud from single-view RGBD images, contrasting them with the point clouds unprojected from depth maps.

5. Conclusion

We have presented a novel correspondence-free approach to category-level object pose and size estimation. This is achieved by learning a shape space of 3D models in normalized pose and metric size based on deep generative model. The input RGBD image is embedded into the shape space, extracting pose-independent features. Pose estimation is realized by comparing the pose-independent and pose-dependent features. Evaluation shows that our method arrives at the state-of-the-art performance.

Limitations and future work. Our current method has a few limitations on which we aim to improve as future work. *First*, our method cannot handle well very complex shapes due to the difficulty in reconstructing shapes with complicated geometry (e.g. high genus). In this aspect, our method can be enhanced by learning a more powerful shape reconstruction with, e.g., volumetric 3D representation. *Second*, our current method does not close the loop in terms of utilizing the reconstructed shape geometry to guide/supervise the training of pose estimation. This may lead to a unsupervised or self-taught approach which we plan to investigate in a future work. *Third*, our method still cannot achieve very high precision, as reflected by the relatively lower accuracy for the $5°5$cm metric. This may be an inherent limitation for a correspondence-free or sparse approach. Note, however, our method did not use ICP to refine the pose or size. *Last*, we plan to extend our current framework to online object pose tracking similar to [27].

Acknowledgement

We thank the anonymous reviewers for the valuable suggestions. We are grateful to Chen Wang, one of the authors of DenseFusion, for the help and discussion. This work was supported in part by the National Key Research and Development Program of China (No. 2018AAA0102200), the NSFC (61572507, 61532003, 61622212, 61902419), the NUDT Research Grants (No.ZK19-30) and the Natural Science Foundation of Hunan Province for Distinguished Young Scientists (2017JJ1002).

References

[1] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X Chang, and Matthias Nießner. Scan2cad: Learning cad model alignment in rgb-d scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2614–2623, 2019.

[2] Vassileios Balntas, Andreas Doumanoglou, Caner Sahin, Juil Sock, Rigas Kouskouridas, and Tae-Kyun Kim. Pose guided rgbd feature learning for 3d object pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3856–3864, 2017.

[3] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *European conference on computer vision*, pages 536–551. Springer, 2014.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[5] Changhyun Choi and Henrik I Christensen. 3d pose estimation of daily objects using an rgb-d camera. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3342–3349. IEEE, 2012.

[6] Changhyun Choi, Yuichi Taguchi, Oncel Tuzel, Ming-Yu Liu, and Srikumar Ramalingam. Voting-based pose estimation for robotic assembly using a 3d sensor. In *2012 IEEE International Conference on Robotics and Automation*, pages 1724–1731. IEEE, 2012.

[7] Thanh-Toan Do, Ming Cai, Trung Pham, and Ian Reid. Deep-6dpose: Recovering 6d object pose from a single rgb image. *arXiv preprint arXiv:1802.10367*, 2018.

[8] Georgios Georgakis, Srikrishna Karanam, Ziyan Wu, and Jana Kosecka. Matching rgb images to cad models for object pose estimation. *arXiv preprint arXiv:1811.07249*, 2018.

[9] Rohit Girdhar, David F Fouhey, Mikel Rodriguez, and Abhinav Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016.

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[11] Tomáš Hodaň, Xenophon Zabulis, Manolis Lourakis, Štěpán Obdržálek, and Jiří Matas. Detection and fine 3d pose estimation of texture-less objects in rgb-d images. In *Proc. IROS*, pages 4421–4428. IEEE, 2015.

[12] Hamid Izadinia, Qi Shan, and Steven M Seitz. Im2cad. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5134–5143, 2017.

[13] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1521–1529, 2017.

[14] Wadim Kehl, Fausto Milletari, Federico Tombari, Slobodan Ilic, and Nassir Navab. Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation. In *European Conference on Computer Vision*, pages 205–220. Springer, 2016.

[15] Yoshinori Konishi, Kosuke Hattori, and Manabu Hashimoto. Real-time 6d object pose estimation on cpu. *arXiv preprint arXiv:1811.08588*, 2018.

[16] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

[17] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise voting network for 6dof pose estimation. In *Proc. CVPR*, pages 4561–4570, 2019.

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

[19] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Proc. ICCV*, pages 3828–3836, 2017.

[20] Caner Sahin, Guillermo Garcia-Hernando, Juil Sock, and Tae-Kyun Kim. Instance-and category-level 6d object pose estimation. *arXiv preprint arXiv:1903.04229*, 2019.

[21] Caner Sahin and Tae-Kyun Kim. Category-level 6d object pose recovery in depth images. In *Proceedings of the ECCV Workshop*, 2018.

[22] Caner Sahin, Rigas Kouskouridas, and Tae-Kyun Kim. Iterative hough forest with histogram of control points for 6 dof object registration from depth images. In *Proc. IROS*, pages 4113–4118. IEEE, 2016.

[23] Juil Sock, S Hamidreza Kasaei, Luis Seabra Lopes, and Tae-Kyun Kim. Multi-view 6d object pose estimation and camera motion planning using rgbd images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2228–2235, 2017.

[24] Juil Sock, Kwang In Kim, Caner Sahin, and Tae-Kyun Kim. Multi-task deep networks for depth-based 6d object pose and joint registration in crowd scenarios. *arXiv preprint arXiv:1806.03891*, 2018.

[25] Alykhan Tejani, Danhang Tang, Rigas Kouskouridas, and Tae-Kyun Kim. Latent-class hough forests for 3d object detection and pose estimation. In *Proc. ECCV*, pages 462–477. Springer, 2014.

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

[27] Chen Wang, Roberto Martín-Martín, Danfei Xu, Jun Lv, Cewu Lu, Li Fei-Fei, Silvio Savarese, and Yuke Zhu. 6-pack: Category-level 6d pose tracker with anchor-based keypoints. *arXiv preprint arXiv:1910.10750*, 2019.

[28] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *Proc. CVPR*, pages 3343–3352, 2019.

[29] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proc. CVPR*, pages 2642–2651, 2019.

[30] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3109–3118, 2015.

[31] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[32] Yang Xiao, Xuchong Qiu, Pierre-Alain Langlois, Mathieu Aubry, and Renaud Marlet. Pose from shape: Deep pose estimation for arbitrary 3d objects. *arXiv preprint arXiv:1906.05105*, 2019.

[33] Haoruo Zhang and Qixin Cao. Holistic and local patch framework for 6d object pose estimation in rgb-d images. *Computer Vision and Image Understanding*, 180:59–73, 2019.