Glissando-Net: Deep sinGLe vIew category level poSe eStimation ANd 3D recOnstruction

Bo Sun, Hao Kang, Li Guan, Haoxiang Li, Philippos Mordohai, Senior Member, IEEE and Gang Hua, Fellow, IEEE

Abstract—We present a deep learning model, dubbed Glissando-Net, to simultaneously estimate the pose and reconstruct the 3D shape of objects at the category level from a single RGB image. Previous works predominantly focused on either estimating poses (often at the instance level), or reconstructing shapes, but not both. Glissando-Net is composed of two auto-encoders that are jointly trained, one for RGB images and the other for point clouds. We embrace two key design choices in Glissando-Net to achieve a more accurate prediction of the 3D shape and pose of the object given a single RGB image as input. First, we augment the feature maps of the point cloud encoder and decoder with transformed feature maps from the image decoder, enabling effective 2D-3D interaction in both training and prediction. Second, we predict both the 3D shape and pose of the object in the decoder stage. This way, we better utilize the information in the 3D point clouds presented only in the training stage to train the network for more accurate prediction. We jointly train the two encoder-decoders for RGB and point cloud data to learn how to pass latent features to the point cloud decoder during inference. In testing, the encoder of the 3D point cloud is discarded. The design of Glissando-Net is inspired by codeSLAM. Unlike codeSLAM, which targets 3D reconstruction of scenes, we focus on pose estimation and shape reconstruction of objects, and directly predict the object pose and a pose invariant 3D reconstruction without the need of the code optimization step. Extensive experiments, involving both ablation studies and comparison with competing methods, demonstrate the efficacy of our proposed method, and compare favorably with the state-of-the-art.

Index Terms—3D Shape Reconstruction, 3D Pose Estimation, Single View 3D Shape Estimation, Variational Autoencoder.

1 Introduction

Reconstruction of 3D geometry from images is one of the classical challenging problems in computer vision. Recently, researchers were able to recover the depth of the scene [1], 3D shape [2], or object pose [3] from a single RGB image. The monocular setting provides only partial observations and makes 3D understanding especially challenging. An effective method must rely on proper prior information, most likely learned from data.

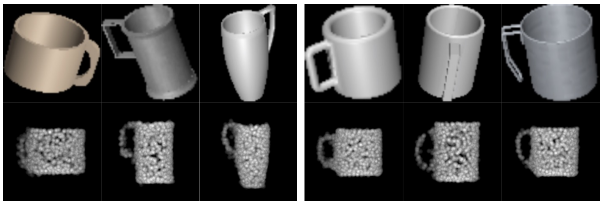


Fig. 1. Category-level 3D reconstruction: on the top are input images from NOCS dataset [4] and on the bottom are predicted 3D shapes; on the left are examples of training instances and on the right are testing instances. Our model predicts complete 3D shapes of instances unknown during training based on their RGB appearance and category shape prior learned from data.

- *B. Sun is with Adobe Inc, San Jose, California, 95110. E-mail: bosu@adobe.com The work was done when Bo Sun was at Stevens Institute of Technology and Wormpex.*
- *P. Mordohai is with Stevens Institute of Technology, Hoboken, New Jersey, 07030.*
- *H. Kang, L. Guan, H. Li and G. Hua are with Wormpex AI Research, Bellevue, Washington, 98004.*

Manuscript received April 19, 2005; revised August 26, 2015.

In this paper, from a single RGB image, we jointly estimate the 3D shape and 6D pose of an object. (Pose in this paper always refers to 6D pose.) Unlike existing works that require data from the exact instance when training the model [5], [6], our method works at the *category level*, observing only objects in the same category, but never the test object in training. It can predict the 3D shape of a novel object and simultaneously estimate its pose. This setting is under-explored on real data, due to its difficulty, and has broader practical applications. It also poses unique and interesting technical challenges since the model needs to capture both high-level semantic differences across categories and low-level geometric details between instances of the same category. We propose a deep CNN framework to overcome the challenge and name it Glissando-Net, short for deep sinGLe vIew category level poSe eStimation ANd 3D recOnstruction.

In training, our network takes RGB images and paired canonical point clouds to learn the point clouds (3D object shapes) and object poses. The RGB image is processed by a U-Net style RGB encoder-decoder network and the point cloud is processed by a PointNet++ encoder and multiple layers of fully-connected layers as the decoder. We carefully design a feature transform module to facilitate interactions between the RGB features and point cloud features. Our model predicts the canonical 3D point cloud and estimates the pose based on the output features of the point cloud decoder, leveraging the fused 2D and 3D information.

In testing, we take a single RGB image as input. The point cloud decoder leverages the RGB features concatenated through the feature transform module to generate object shapes and predict their poses. Our model consumes an RGB image with a single centered object. Given an image with multiple objects, we assume the availability of an object detector to localize and crop the

objects. As shown in Figure 1, without seeing the same instances in training, our model predicts distinguishable 3D shapes and well-aligned object poses. We experimentally compare our method with recent works on two real datasets and observe consistent improvements in both shape reconstruction and pose estimation.

Our contributions are three-fold: 1) a novel framework for category-level pose estimation and 3D reconstruction from a single RGB image that learns from 3D data during training but does not require them during testing; 2) a feature transform module that enhances 2D-3D interaction during learning; 3) better performance than the state-of-the-art in both pose estimation and 3D reconstruction on recent benchmarks.

2 Related Work

Object shape and pose recovery are crucial for many applications, such as 3D reconstruction, robotics, autonomous driving, augmented reality etc. Although this line of work originated from purely geometric methods, learning based approaches recently have shown superior capability to model shape priors. Given the scope of our work, we focus on only discussing the learning-based approaches in this section. What makes the problem challenging, is that the two sub-problems (i.e. the shape recovery problem and object pose recovery problem) are both ill-conditioned. Here, we only discuss learning-based approaches starting with those that are most closely related to ours.

To our knowledge, the only approach that shares all challenges we address, specifically (i) joint shape and pose estimation, (ii) from a single real RGB image, (iii) at the category level, is CPS [7]. CPS minimizes a loss comprising (i) the symmetric Chamfer distance between the ground truth point cloud and the estimated point cloud transformed according to the estimated pose, and (ii) a regularization term that penalized deviations from the learned class distributions in latent space. This loss measures shape alignment directly without separating shape and pose estimation. Recently, the same authors presented CPS++ [8], which undergoes fully supervised training on synthetic data, and is then finetuned on real RGB-D data using differentiable rendering techniques for self-supervision. Unlike our approach, CPS does not extract information from the 3D shapes during training and makes all predictions using the encoder. Moreover, CPS relies on a separate latent space per class, while objects from all classes are encoded in the same latent space by Glissando-Net.

A milestone in this line of research was presented by Tulsiani et al. [9] who use multiple views of the object to minimize a geometric loss during training. A single image is sufficient to infer pose and a low-resolution $32 \times 32 \times 32$ occupancy grid. Insafutdinov and Dosovitskiy [10] also train a network on multiple images, but they employ a point cloud representation and an ensemble of pose predictors to mitigate pose ambiguity. Both [9], [10] have been tested only on synthetic data. In addition to the Pix3D dataset, which we use in our experiments, Sun et al. [11] present a method for estimating voxelized shape and pose, adopting the 2.5-D sketch as an intermediate representation. Nie et al. [12] estimate room layout, camera pose, bounding boxes and meshes for the 3D objects from a single image benefiting from the synergies among all tasks. CoReNet [13] addresses the same problem by jointly estimating the class, pose and shape of all objects, instead of each object separately. Zhang et al. [14] also address joint inference of scene layout, object poses and meshes via an implicit representation that avoids intersecting surfaces and a graph convolutional network that refines the predictions based on context.

GSNet [15] estimates the 3D pose and detailed shape of cars from a single image leveraging context, but is limited to a single class of objects. All of these methods train a separate model for each category, making them susceptible to classification errors by the detector. Glissando-Net relies on a shared latent space across all categories.

Wang et al. [4] introduce the Normalized Object Coordinate Space (NOCS), which is a shared canonical representation for all instances in a category. The network infers the mapping from pixels in the RGB-D input to NOCS to estimate object pose and size. The representation was extended to yield X-NOCS [16], a system capable of reconstructing the visible and occluded surfaces of an object from one or more RGB images. While it can predict camera pose in the canonical NOCS space, it requires depth to estimate metric pose. Recently, Chen et al. [17] presented the Canonical Shape Space (CASS) that enables category-level object shape, pose and size estimation from an RGB-D image by predicting a pose-independent shape and using it to estimate pose and size. Both CASS and NOCS are unified object description spaces for shapes in a category. NOCS is explicitly defined by aligning the instances within a category in a normalized 3D space and estimates pose via geometric reasoning. CASS, on the other hand, is implicitly learned via a generative model and enables pose estimation by the network.

Our design is partially inspired by CodeSLAM and its follow-ups [1], [18], [19], which use a variational auto-encoder (VAE) to encode the inter-categorical and intra-categorical shape differences in the network. A key difference is that they treat the scene as generic, while we treat it as a collection of specific shapes learned from CAD models (in the form of point clouds).

We now turn our attention to category-level separate shape and pose estimation from RGB inputs. The predecessors to our work include approaches that detect 3D objects in RGB images [20], [21], [22] given CAD models from the same class for training. These methods, however, estimate approximate bounding boxes and are limited to objects on the ground plane, with three degrees of freedom. The IM2CAD system [23] creates a CAD model of a scene by aligning database CAD models with corresponding objects in the input image. Georgakis et al. [3] estimate 6D pose by learning RGB-to-CAD correspondences without requiring photo-realistic texture for the CAD models or explicit 3D pose annotations in the images. This is accomplished by learning how to select keypoints and enforcing viewpoint and modality invariance across RGB images and CAD model renderings. Several authors [24], [25] have trained networks on multiple images of known objects to regress the pose of unseen ones. A similar approach requiring only cluttered images for training has also been presented by Park et al. [26].

Assuming the object pose is known or to recover view-dependent shape, probabilistic and generative techniques have been used to impose constraints on 3D shape space. A variety of methods are capable of reconstructing 3D shape from a single image given the object pose, or at least an approximation in the form of a bounding box, by implicitly or explicitly learning class-specific generative models. Categorized by their shape representations, some use 3D point clouds [2], [27], like we do, while others use volumetric representations [28], [29], [30], [31], [32], [33], implicit functions [34], [35], meshes [36], [37], [38], [39], [40], [41], [42], or collections of parametric elements

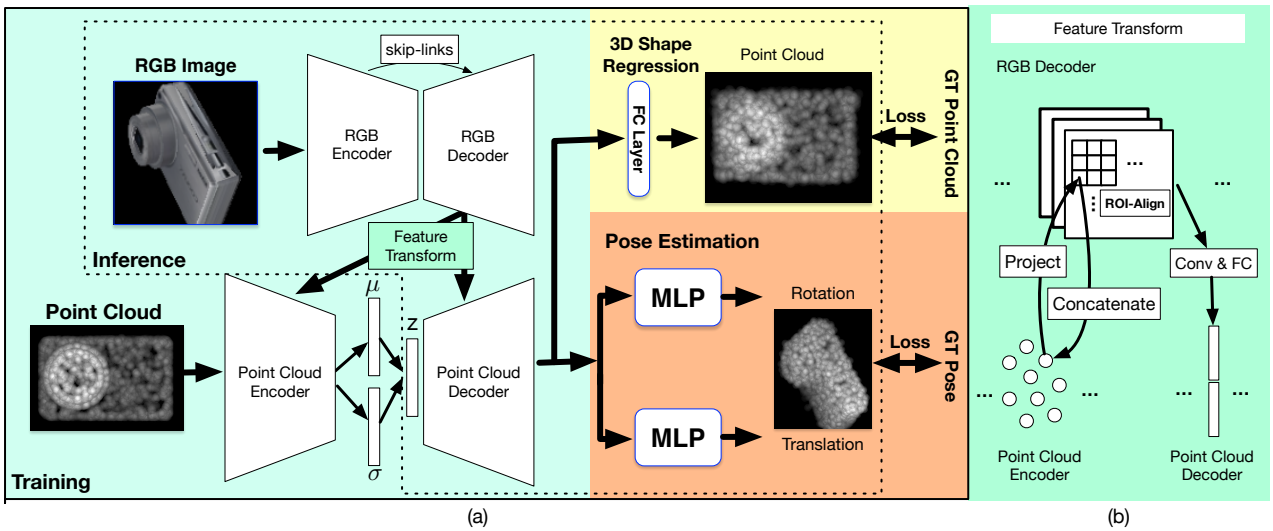


Fig. 2. Overview of Glissando-Net: the RGB image is processed by an encoder-decoder; the feature maps from the RGB decoder are then transformed and concatenated with features in the point cloud decoder, which takes in a latent code $\mathbf{z}$; the outputs from the point-cloud decoder are processed by a fully-connect layer to regress the 3D shape in canonical pose and two independent multilayer perceptrons (MLP) to estimate its pose; in training, the ground-truth canonical shape is an additional input to a point cloud encoder to learn the distribution of $\mathbf{z}$ and feature maps from the RGB decoder are transformed to augment point cloud features.

[43]. Other reconstruction approaches from RGB images use the 2.5-D sketch [44], silhouettes [29], [45], or a combination of depth maps, silhouettes, 3D voxels, and spherical maps as intermediate representation [46]. AtlasNet [43] can reconstruct a 3D shape, represented as a collection of parametric surface elements, from a single image. High-resolution reconstruction is possible since processing occurs in batches. Mesh R-CNN [36], which we compare with, begins by recognizing objects in a 2D image and infers a coarse 3D occupancy grid, which is converted to a finer-resolution mesh representing the 3D object. Mesh R-CNN, however, approximates the pose from the predicted 3D bounding boxes and does not output precise pose estimates that can be evaluated against ours. Ye et al. [47] present an approach along the same lines, where supervision is provided in the form of foreground masks returned by off-the-shelf object detectors. Shape reconstruction is carried out in two phases: category-level volumetric prediction and instance-level mesh specialization. Pose supervision is not required, and the accuracy of pose estimation is not assessed.

Our work is also relevant to object pose estimation. We refer readers to [48] for a survey and to [49] for a benchmark and a comprehensive evaluation of several methods. Several authors have developed methods relying on RGB-D inputs [50], [51], [52], [53], [54], [55], [56], [57], [58] to overcome clutter and occlusion. Clearly, pose estimation is harder based solely on RGB information, even when exact CAD models are used for training. Methods in this category can be further distinguished into those that learn to estimate 2D-3D correspondences and feed them to a perspective-n-point (PnP) solver [59], [60], [61], [62], [63], regress pose from image features [6], [51], perform retrieval and/or refinement [64], [65], or estimate the projections of the corners of the bounding box in the image [5], [66]. Li et al. [67] estimate pose at the category level by iterative rendering and refinement.

3 Method: Glissando-Net

Figure 2 shows an overview of the structure of our proposed Glissando-Net.

Training Stage (delineated by the solid lines in Fig. 2 (a)). The model learns to predict the pose and canonical shape in a supervised manner. The inputs to the network are (1) the object's RGB image, (2) 3D shape, in the form of a 3D point cloud, and (3) its ground truth pose with respect to the RGB view. The *canonical shape* (e.g. the bottom-left point cloud of a camera in Fig. 2 (a)) is defined as in [4]: by aligning all objects in the same category to the same pose according to some convention (e.g. axis-aligned and frontal) and normalized to the same scale. For 2D appearance learning, we use a U-Net-based encoder-decoder network [68] (top branch in Fig. 2 (a)). For 3D shape learning, we use a Variational Auto Encoder (VAE) network (bottom branch in Fig. 2 (a)) to encode the shape information from the canonical point cloud. We define the latent code as $\mathbf{z}$. The ground truth pose is used to regress the predicted pose, as well as to project the 3D points to the 2D image space, such that the geometrically correct features learned from the 2D appearance branch can be passed to the 3D shape inference branch.

Inference Stage (delineated by the dashed lines in Fig. 2 (a)). We predict an object's canonical 3D point cloud and its pose, given *only* one RGB image as input. Since no 3D shape or pose is given, we simply use all-zero-code for the $\mathbf{z}$ vector in the VAE, combined with the RGB information passed to the shape decoder part of the VAE to predict the final shape and pose. Such an ill-posed task is made possible because shape reconstruction is primarily guided by the RGB appearance passed from the top U-Net, while the bottom point cloud VAE "refines" the shape based on the learned shape-appearance priors in the training stage.

3.1 Image Feature Extraction

The RGB sub-network of Glissando-Net (top in Fig. 2 (a)) is a U-Net [68] comprising an encoder and a decoder. The encoder follows a typical convolutional architecture with a stack of five

groups of layers. Each group performs two 3×3 convolutions, each followed by a group normalization layer [69] and a ReLU. The first 3×3 convolution is used for downsampling with a stride of 2 and doubling the number of feature channels. We use group normalization instead of batch normalization, because the computation of group normalization is independent of batch sizes, and its accuracy is stable in a wide range of batch sizes. We also stack five group of layers in the decoder. Each group consists of a 2×2 deconvolution and two 3×3 convolutions, each followed by a group normalization layer and a ReLU. 2×2 deconvolution is used for upsampling, while the number of channels is halved. Layers in the decoder are skip-connected to the corresponding layers in the encoder.

3.2 Geometric Feature Extraction

The geometric sub-network of Glissando-Net (bottom network in Fig. 2 (a)) is a Variational Auto Encoder (VAE), which takes as input a point cloud sampled from a CAD model. All objects are aligned to the same canonical pose. An example is shown in the bottom-left of Fig. 2 (a): the frontal point cloud of a camera.

The backbone of the encoder is a PointNet++ [70] with five set abstraction layers. PointNet++ is a hierarchical neural network that applies PointNet recursively on a nested partitioning of the input 3D point set. It partitions the set of points into overlapping local regions according to L1 distance in the underlying space and it learns local features in increasing contextual scales. This PointNet++ based encoder transforms the geometric features from the canonical point cloud into a latent vector $\mathbf{z}$ following a normal distribution. After reparameterization, we have two fully connected layers to estimate its mean μ and variance σ^2. The latent code $\mathbf{z}$ is then sampled from a Gaussian distribution $N(\mu, \sigma^2)$. KL-divergence is used to constrain the code distribution close to a centered isotropic Gaussian distribution. The decoder part consists of five fully-connected layers (with 256, 512, 1024, 2048, and 4096 nodes respectively), which generate an abstract representation for the point cloud in different scales. Group normalization and ReLU are applied between layers.

3.3 Feature Transform Module

The top network (image encoder-decoder) extracts RGB features from the 2D image and the bottom network (point cloud VAE) extracts geometric features from the 3D point cloud. In order to fully utilize the information from both, we transform and concatenate the image decoder features with the point cloud encoder and the point cloud decoder.

As shown in Fig. 2 (b), to concatenate the image decoder features with those of the point cloud encoder, we project the 3D points to the 2D image according to the relative pose between the camera and the canonical 3D shape. We denote the sampled 3D points in the i^{th} scale (abstraction layer) in the point cloud encoder by M_i. Both our point cloud encoder and the image decoder have five scales allowing us to project M_i to the corresponding feature map of the image decoder. F_i denotes the feature map in the i^{th} scale of the image decoder. We use the ROI-align layer from mask-RCNN [71] to obtain the local features around the projected points. Then, we concatenate the features from the ROI-align layer with the corresponding 3D point features from the point cloud encoder $F_P(M_i)$:

$$F_{3D_i} = Conc(RoI_{align}(F_{5-i}(K \cdot (R \cdot M_i + T))), F_P(M_i)) \quad (1)$$

where the ground truth rotation and translation are denoted by R and T and the intrinsic matrix by K. This technique creates strong connections between each 3D point and the corresponding 2D pixel.

The point cloud encoder is only used during training, since there is no 3D shape input during inference. To concatenate image decoder features with those from the point cloud decoder, we transform the features from the entire image map at scale i (F_i) into a feature vector using a convolutional layer and a fully-connected layer. We denote the feature vector from the i^{th} scale of the point cloud decoder by F_{D_i}. Then, we concatenate F_i with the feature vector of the corresponding layer from the point cloud decoder ($F_{D_{5-i}}$).

With these two feature transformations, we facilitate interaction between 2D and 3D features. In training, the prediction from the point cloud decoder is partially conditioned on the 2D features, so that we can estimate pose and reconstruct 3D shape in testing without point cloud inputs.

3.4 Details of Network Structure

Figure 3 shows the details of the RGB and point cloud encoder-decoders, zooming in on parts of Fig. 2. $Conv2d(n_{in}, n_{out})$ denotes a 3×3 2D convolution layer with n_{in} input channels and n_{out} output channels. $DeConv2d(n_{in}, n_{out})$ denotes a 3×3 2D deconvolution layer with n_{in} input channels and n_{out} output channels. The red arrows denote the skip connections between the encoder layer and the decoder layer. Feature 1 - Feature 4 are the outputs from the RGB decoder layers. We feed these features to the corresponding layers of point cloud encoder-decoder. In the point cloud encoder, $SA\ \ Layer\ \ n\ \ points$ denotes a set abstraction layer with n sampled points from PointNet++ [70]. We project the sampled points in the SA layers back to the corresponding feature map of the RGB encoder. Then, we use RoI align to take the output features from the RGB encoder-decoder Feature 1 - Feature 4 and concatenate them with the output from the corresponding SA layers. To concatenate Feature 1 - Feature 4 with the output of the point cloud decoder layers, we use one 2D convolution layer and one fully connected layer to convert each of Feature 1 - Feature 4 to a one-dimensional feature.

3.5 3D Shape and Pose Regression

We use one fully-connected layer after the point cloud decoder to regress the 3D shape. We use the Earth Mover's Distance (EMD) or the Chamfer Distance as the shape reconstruction loss [2]. EMD is computationally more expensive since it solves an assignment problem, while Chamfer distance is faster to compute but leads to a slight loss of accuracy as shown in Section 4.

We design two small networks, each with five fully connected layers for pose estimation. These two networks are of the same structure up to the last layer: one outputs a 4D quaternion for the rotation, and the other outputs a 3D translation. We design a loss function named Per-Point L2 loss for pose regression. We, first, apply the estimated rotation $\hat{R}$ and translation $\hat{T}$ to input point cloud pc_{gt} to obtain pc_1, and the ground truth rotation R and translation T to input point cloud pc_{gt} to obtain pc_2. Then, we define the Per-Point L2 loss between these point clouds, pc_1 and pc_2, as:

$$Loss_{per\text{-}point} = \frac{1}{N} \sum_{x \in pc_{gt}} \left\| (\hat{R}x + \hat{T}) - (Rx + T) \right\|_2 \quad (2)$$

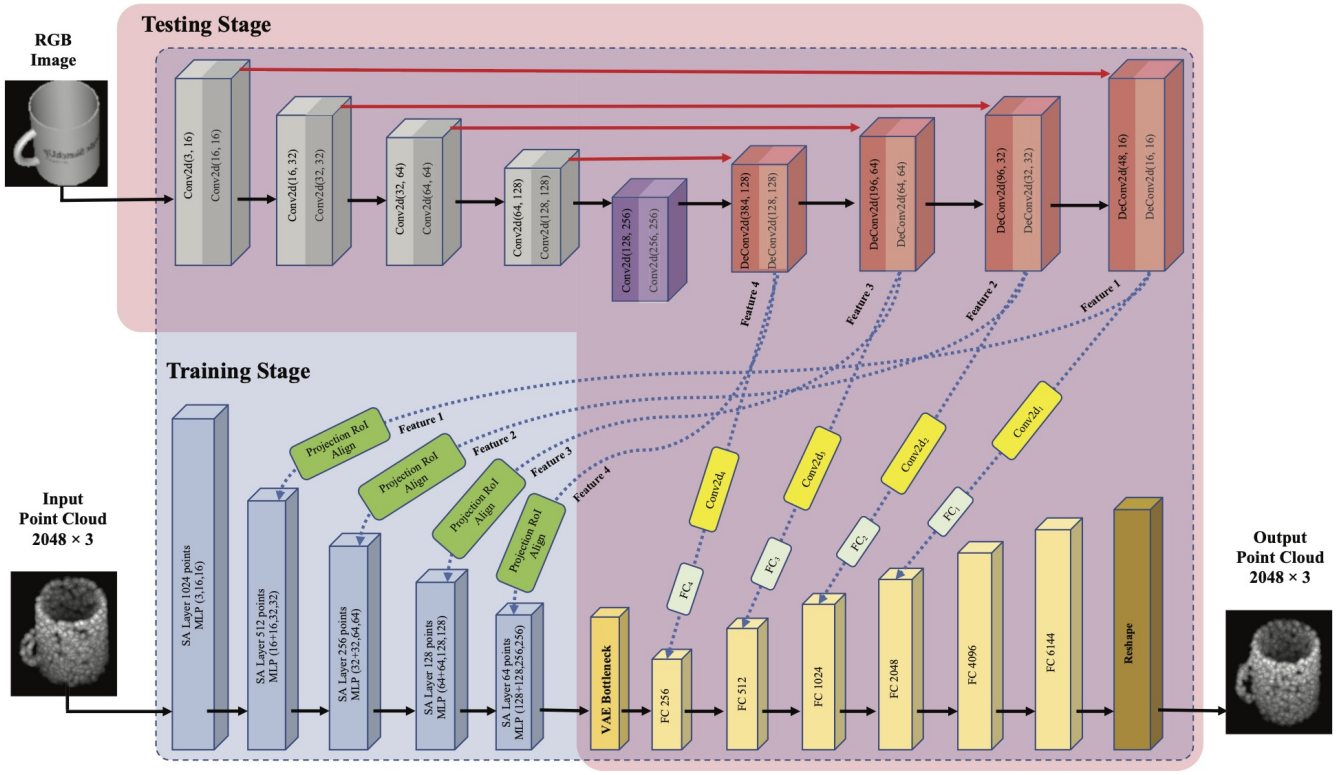


Fig. 3. Details of RGB and point cloud encoder-decoder from Glissando-Net.

where N is the number of points in pc_1 and pc_2.

Our total loss is the sum of three terms: shape reconstruction loss $Loss_{shape}$, KL Divergence loss $Loss_{kld}$ and pose loss $Loss_{per\text{-}point}$. We use weights w_{shape}, w_{kld} and w_{pose} to balance the terms.

$$Loss_{all} = w_{shape}Loss_{shape} + w_{kld}Loss_{kld} + w_{pose}Loss_{per\text{-}point} \quad (3)$$

4 Experimental Results

In this section, we present the datasets and metrics used in our experiments, experimental results, as well as several ablation studies.

4.1 Datasets, Experimental Setup and Evaluation Metrics.

We evaluate Glissando-Net on two recent datasets for category-level pose estimation and shape reconstruction. (1) The **NOCS dataset** [4] consists of six object categories: bottle, bowl, camera, can, laptop and mug, picked from ShapeNetCore [72]. NOCS includes both synthetic and real data with camera intrinsics and ground truth pose. The synthetic dataset consists of 275K training and 25K validation images rendered from 1085 individual object instances in ShapeNetCore. The real scene data consists of 5250 training images and 2750 testing images from 18 different real scenes. We crop the objects in each image using the ground truth mask and resize to 128 × 128. For synthetic scenes, we generate about 625K training crops and about 46K validation crops. For real scenes, we have about 26K training and 9K testing images. We sample point clouds with 2048 points from the CAD models for both data types. (2) The **Pix3D dataset** [11] has 395 3D models of furniture from 9 categories, with 10,069 precisely aligned RGB images, camera intrinsics and ground truth pose. We crop the object in each image using the ground truth mask and resize it to 256 × 192. To be consistent with [12], [36] which we compare with in the experiments, we sample point clouds with 10K points from the CAD models. We evaluate the 3D shape reconstruction on the Pix3D dataset following the train/test splits from Gkioxari et al. [36].

We implemented our method in PyTorch. We train our network, which has 67 million parameters, on all images and point clouds generated both from the synthetic and real training data of NOCS with a batch size of 128 for 180 epochs using the ADAM optimizer. We use an initial learning rate of 0.0001 and decay it by 0.1 after 100, 130, and 160 epochs. Training on 4 NVIDIA GTX 2080Ti GPU takes approximately 10 days. We test our model on the synthetic and real data from NOCS separately with inference taking 0.314 seconds per frame. For the Pix3D dataset, we train our network with a batch size of 32 for 160 epochs using the ADAM optimizer. We also use an initial learning rate of 0.0001, decayed after 110 and 150 epochs by 0.5 and 0.2. We set the weights w_{shape}, w_{kld} and w_{pose} in Eq. (3) to 1, 100 and 100, respectively.

3D Shape Evaluation We use Chamfer distance to evaluate 3D shape reconstruction.

Pose Estimation We use two metrics to evaluate pose estimation: the *Average Distance of Predicted Point Sets (APP)* [7] and the *10°& 10 cm metric* [4]. APP is extended from two metrics for 6D pose comparison known as the Average Distance of Distinguishable Model Points (ADD) and Average Distance of Indistinguishable Model Points (ADI) [73], [74]. ADD measures

TABLE 1
Shape Reconstruction Result on Synthetic and Real Data from NOCS. CASS requires RGB-D inputs. Glissando-Net (EMD)* represents Glissando-Net (EMD) without the 3D encoder.

Methods	Mean Chamfer Distance in mm ↓									
	Real Data							Synthetic Data		
	bottle	bowl	camera	can	laptop	cup	overall	cup	bottle	camera
CPS [7]	-	-	-	-	-	-	-	3.4	3.7	11.2
CASS [17]	0.75	0.38	**0.77**	0.42	3.73	0.32	1.06	-	-	-
Glissando-Net (Chamfer)	**0.42**	0.47	1.21	0.35	2.62	0.36	0.91	0.41	1.22	0.75
Glissando-Net (EMD)*	2.72	1.88	2.69	2.91	4.07	1.43	2.62	2.15	2.78	5.33
Glissando-Net (EMD)	0.49	**0.34**	1.09	**0.34**	**2.54**	**0.31**	**0.85**	**0.39**	**0.81**	**0.69**

TABLE 2
Pose Estimation Result on Synthetic and Real Data from NOCS. Glissando-Net (EMD)* represents Glissando-Net (EMD) without the 3D encoder.

Methods	APP (α = 0.2/0.5) ↑			10°& 10 cm ↑		
	cup	bottle	camera	cup	bottle	camera
Synthetic Data						
CPS [7]	21.6%/52.9%	18.8%/56.5%	14.8%/**43.7%**	28.8%	25.6%	12.5%
Glissando-Net (Chamfer)	46.12%/63.86%	34.02%/59.12%	15.95%/34.23%	34.55%	23.09%	8.85%
Glissando-Net (EMD)*	34.17%/53.23%	30.84%/51.15%	13.41%/28.47%	21.12%	19.25%	5.57%
Glissando-Net (EMD)	**56.73%/73.57%**	**50.70%/71.08%**	**16.05**%/32.94%	**44.65%**	**25.99%**	**12.62%**
Real Data						
CPS [7]	**36.8**%/64.6%	1.6%/14.1%	19.6%/**56.7**%	1.3%	9.5%	1.9%
Glissando-Net (Chamfer)	31.09%/74.39%	5.73%/19.19%	20.45%/44.79%	3.87%	8.62%	5.78%
Glissando-Net (EMD)*	18.78%/58.43%	3.67%/18.32%	6.67%/18.32%	1.08%	2.33%	1.33%
Glissando-Net (EMD)	34.48%/**78.99**%	**9.28**%/**22.80**%	**24.91%**/42.80%	**4.44%**	**10.16**%	**11.36%**

TABLE 3
Overall Pose Estimation Result on Synthetic and Real Data from NOCS. Glissando-Net (EMD)* represents Glissando-Net (EMD) without the 3D encoder.

Methods	Synthetic Data		Real Data	
	10°& 10 cm ↑	APP (α = 0.2/0.5) ↑	10°& 10 cm ↑	APP (α = 0.2/0.5) ↑
CPS [7]	31.7%	19.1%/49.6%	14.7%	30.8%/64.0%
CPS ++ [8]	27.4%	17.8%/45.2%	**22.3%**	**41.0%/73.6%**
Glissando-Net (Chamfer)	31.22%	34.25%/54.02%	16.27%	30.99%/61.98%
Glissando-Net (EMD)*	21.21%	30.55%/49.47%	2.97%	23.78%/37.82%
Glissando-Net (EMD)	**31.86%**	**47.11%/66.65%**	19.08%	34.06%/63.90%

whether the average point distance between the point cloud in the estimated and the ground truth pose is less than 10% of the object's diameter. ADI extends ADD for symmetries, measuring error as the mean distance to the closest model point. To circumvent the need for direct correspondences, Manhardt et al. [7] introduce APP which simply extends ADI to be computed bidirectionally:

$$APP = \begin{cases} 1, & \text{if } m_1 \leq \alpha \cdot d(pc) \wedge m_2 \leq \alpha \cdot d(\hat{pc}) \\ 0, & otherwise \end{cases} \tag{4}$$

pc is the point cloud with ground truth pose, $\hat{pc}$ is the point cloud with estimated pose. $d(\cdot)$ measures the diameter of pc. m_1 and m_2 are defined as:

$$\begin{aligned} m_1 &= \frac{1}{N} \sum_{x_1 \in pc} \min_{x_2 \in \hat{pc}} \left\| (Rx_1 + T) - (\hat{R}x_2 + \hat{T}) \right\|_2, \\ m_2 &= \frac{1}{N} \sum_{x_2 \in \hat{pc}} \min_{x_1 \in pc} \left\| (Rx_1 + T) - (\hat{R}x_2 + \hat{T}) \right\|_2 \end{aligned} \tag{5}$$

N is the number of points in pc. R and T are the ground truth rotation and translation, and $\hat{R}$ and $\hat{T}$ are the estimated rotation and translation. We use 20% and 50% as threshold values for α. We also use the 10°& 10 cm metric, which counts the number of samples for which rotation error is less than 10°and translation error is less than 10 cm.

4.2 NOCS Dataset Results

We compare the performance of Glissando-Net with CPS [7] and CASS [17]. To the best of our knowledge, CPS is the only method for 3D shape reconstruction and category level pose estimation on real data using an RGB image as input. We evaluate our method on the three categories which CPS reports, even though our model was trained on all categories, using both real and synthetic data. CASS requires RGB-D inputs.

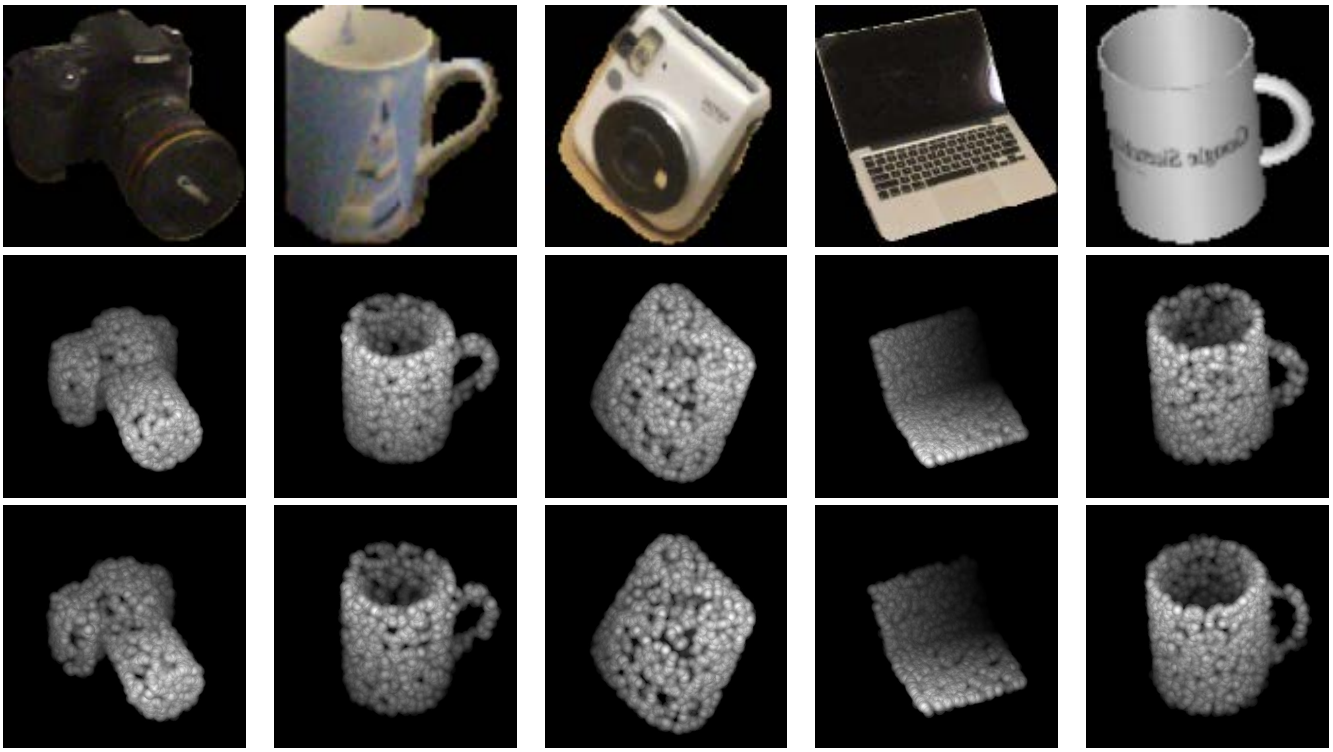
Glissando-Net without Point Cloud Encoder (Glissando-Net*): We remove the point cloud input and 3D encoder to create an additional baseline. We compare the performance of Glissando-Net with this baseline to evaluate the importance of the point cloud encoder.

Shape Reconstruction: Table 1 shows that Glissando-Net with EMD significantly outperforms CPS in terms of mean Chamfer distance across all three categories. (CPS does not report shape reconstruction results on real data.) Even without the benefit of depth information, Glissando-Net outperforms CASS on three categories of the real data and achieves better overall performance.

TABLE 4
Shape Reconstruction Results on Pix3D. The values are in units of 10^{-3}. (AtlasNet and TMN results are repeated from Total3D.)

Methods	bed	bookcase	chair	desk	sofa	table	tool	wardrobe	misc	mean
Instance-level Shape Reconstruction										
AtlasNet [43] (S_1)	9.03	6.91	8.37	8.59	6.24	19.46	6.95	4.78	40.05	12.26
TMN [39] (S_1)	7.78	5.93	6.86	7.08	4.25	17.42	4.13	4.09	23.68	9.03
Total3D [12] (S_1)	5.99	6.56	5.32	5.93	3.36	14.19	**3.12**	3.83	26.93	8.36
Glissando-Net (S_1)	**2.61**	**4.85**	**2.21**	**4.00**	**1.28**	**3.75**	13.39	**1.82**	**11.68**	**5.06**
Category-level Shape Reconstruction										
Mesh R-CNN [36] (S_2)	15.48	**14.14**	9.90	20.70	5.30	26.54	-	-	31.08	17.59
Glissando-Net (S_2)	**5.98**	14.22	**6.65**	**15.51**	**1.69**	**13.99**	40.96	5.78	**24.78**	**11.83**

Fig. 4. Qualitative results on NOCS dataset. Top: input image. Middle: ground-truth 3D shape with ground-truth rotation. Bottom: predicted 3D shape with predicted rotation.



Glissando-Net with Chamfer distance is not as good as with EMD, but still outperforms the baselines. The poor performance of Glissando-Net* without the 3D encoder shows that the 3D encoder contributes substantially to shape reconstruction.

Pose Estimation: The authors of CPS [7] implemented the NOCS network without the depth branch. We do not repeat these results here since they are not competitive. Table 2 shows pose estimation results. On the synthetic subset, Glissando-Net with EMD performs better than CPS on all metrics in all categories, except for APP with $\alpha = 0.5$ on the camera category. On the real subset, Glissando-Net with EMD performs better than CPS according to the 10°& 10 cm metric in all categories and also better in most shape estimation metrics. Table 3 shows the overall pose estimation results across all of 6 categories. Glissando-Net ranks first on synthetic data and performs better than CPS but worse than CPS++ on real data. (CPS++ has been trained on large amounts of additional data, but in a self-supervised manner.) Glissando-Net* without the 3D encoder performs worse than all other methods on both synthetic and real data. Figure 4 shows qualitative results of predicted 3D shapes and poses. Figures 7 and 8 contain additional results on the synthetic and real subsets of the NOCS dataset [4].

4.3 Pix3D Dataset Results

We also evaluate our method on the Pix3D Dataset using the two train/test splits from [36]: S_1 and S_2 which are instance-level and category-level, respectively. We compare with [12] and two baselines from [12] for instance-level shape reconstruction, and with Mesh R-CNN [36] for category-level shape reconstruction. Since Mesh R-CNN does not report Chamfer distance results, we ran the authors' code and computed Chamfer distances on the generated meshes. (We exclude the tool and wardrobe categories due to our inability to reproduce reasonable results.) Table 4 summarizes the results. For instance-level shape reconstruction, Glissando-Net achieves the best results on all categories, except for tool for which there are insufficient samples. For category-level shape reconstruction, Glissando-Net performs better than Mesh R-CNN. Compared with instance-level methods, Glissando-Net on S_2 performs a little worse than Total3D [12], but still better than

[43] and [39], on S_1. The mean error of Glissando-Net on S_2 is computed over 7 categories. The mean error over 9 categories is 14.39 which is still lower than the mean error of Mesh R-CNN on S_2 over 7 categories. Figures 6 and 10 show several qualitative shape reconstruction results from the Pix3D dataset [11].

4.4 Ablation Studies

We perform three ablation studies on a subset of the NOCS synthetic dataset. Specifically, for the first two ablation studies, we randomly sample 10K images and their point clouds from the synthetic training dataset for training, and 200 images from each of three categories from the testing dataset for validation. For the last ablation study, we train and test the model on synthetic data from the NOCS dataset.

Length of the latent code z in the VAE: We evaluate the effect of the latent code $\mathbf{z}$ with different vector lengths: 32, 64, 128, 256, 512, 1024. Table 5 shows the results of shape reconstruction and pose estimation with different sizes of the latent code. As expected, larger codes yield better results. When the code size is larger than 256, the performance does not change a lot. For computational efficiency, we set the code size to 256 for all other experiments in the paper.

TABLE 5
Ablation study on code size

Code Size	Shape Chamfer mm ↓	APP (α = 0.5) ↑	APP (α = 0.2) ↑	10° 10 cm ↑
32	0.98	52.50%	28.50%	6.67%
64	0.90	57.67%	30.83%	14.50%
128	0.87	57.33%	31.17%	16.83%
256	0.81	58.50%	35.50%	18.50%
512	0.81	58.83%	34.50%	17.91%
1024	0.80	58.67%	35.70%	18.84%

Effects of the Pose Sub-network: We experimented with different positions of the pose regression network: after the shape decoder or after the image encoder. We also try to train our network without the pose network. Table 6 shows the results. From the top two rows, we can see that shape reconstruction can help pose estimation. The results in the second and third columns indicate that jointly estimating pose can help shape reconstruction.

TABLE 6
Ablation study: with/without pose estimation

	Shape Chamfer mm ↓	APP (α = 0.5) ↑	APP (α = 0.2) ↑	10° 10 cm ↑
Pose after shape decoder	0.81	58.50%	35.50%	18.50%
Pose after image encoder	0.81	53.50%	30.60%	15.67%
Without pose	0.83	-	-	-

Effects of the Transform Modules: We performed an additional ablation study on our feature transforms. We made Glissando-Net$_{DisEn}$ by disconnecting the feature transform module between RGB decoder and point cloud encoder, and Glissando-Net$_{DisDe}$ by disconnecting the feature transform module between RGB decoder and point cloud decoder (see Fig. 2). We trained all variants on the NOCS dataset using the EMD distance in the loss. Glisando-Net* is the baseline without the 3D encoder as in Tables 1-3. As shown in Table 7, the second feature transform module is more critical than the first one and the 3D encoder. This table is an extension of Tables1 & 3 and contains shape and pose estimation results. (The averages are over all samples in the *six* NOCS categories, covering more synthetic data than Table 1.)

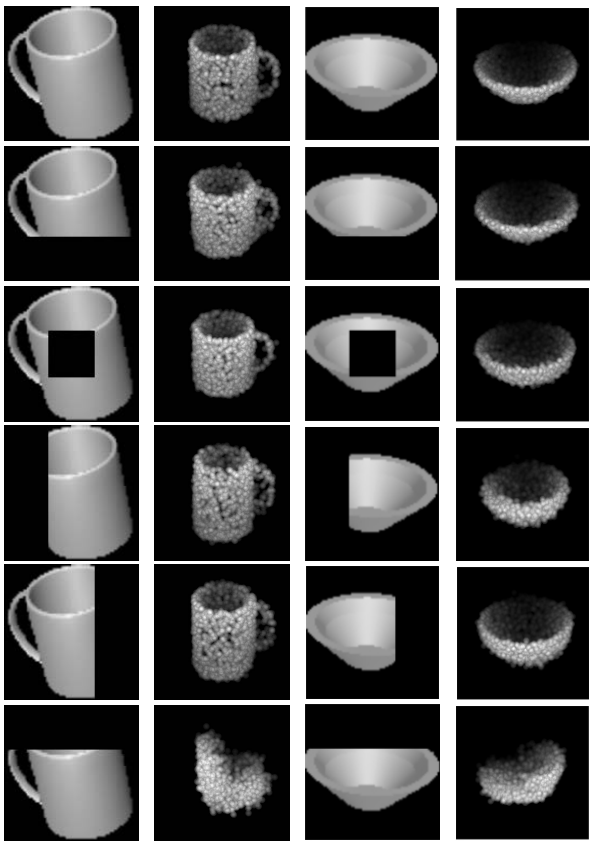


Fig. 5. Reconstructed 3D shape with occluded RGB input. Column 1 and 3, top to bottom: original RGB image and the ones occluded from bottom, center, left, right, and top. Column 2 and 4: Corresponding reconstructed shapes, shown in the same pose for visualization.

TABLE 7
Ablation study for feature transform module.

Methods	Synthetic Data		Real Data	
	Chamfer	10^o & 10 cm	Chamfer	10^o & 10 cm
Glissando-Net*	3.14	21.21%	2.85	2.97%
Glissando-Net	0.62	**31.86%**	0.71	19.08%
Glissando-Net $_{DisEn}$	0.94	24.98%	1.24	4.16%
Glissando-Net $_{DisDe}$	3.44	2.06%	6.16	1.60%

4.5 Results on Partially Occluded Objects

Glissando-Net can handle partially occluded objects thanks to the latent shape space in addition to the RGB appearance network. We randomly pick 2K images from the NOCS synthetic testing data and render five kinds of occlusion (top, bottom, left, right and center) on each image, using a black block with 1/3 width or height of the original image. We tested these 10K partially occluded images as well as the corresponding 2K occlusion-free original images using the network trained on the synthetic training data of NOCS *without occlusion*. Some of the 3D shape reconstruction results are shown in Fig. 5. Although the occluded shapes are

Fig. 6. Qualitative results on Pix3D: input (top row). ground-truth 3D shape (second row) and our predicted 3D shape (third row). The dataset does not provide pose estimation evaluation. Point clouds are shown in a pose chosen for visualization.

not as precise as their occlusion-free counterparts, the network still completes the shape for the unseen regions exceptionally. We believe that this is because our VAE successfully learns categorical shape space as a strong prior to compensate for the incomplete appearance information. We speculate that occluding the top part of a cup is more challenging since the top is harder to predict from the visible parts of the shape. The average Chamfer distance from ground truth is 0.56 mm on images without occlusion and 1.17 mm on images with occlusion.

We also present quantitative results on shape reconstruction from occluded RGB images. We report shape reconstruction results separately for the five directions of occlusion in Table 8. The quantitative results are consistent with the visualization results in Figure 5. When the top of the object is occluded, Glissando-Net cannot reconstruct its 3D shape correctly.

TABLE 8
Shape reconstruction results on occluded RGB inputs from the NOCS synthetic dataset. The table shows the mean Chamfer distance in mm for each occlusion direction and the average over all direction in the last column. As a reminder, the mean Chamfer distance without occlusion is 0.56 mm.

	bottom	center	left	right	top	overall
Chamfer Distance(mm) ↓	0.85	0.74	1.13	0.89	2.23	1.17

4.6 Limitations

Figure 9 shows examples on which our method fails. As mentioned in the conclusions, while Glissando-Net has proven to be robust to occlusion in general, it is less robust when the top of the object is occluded. We hypothesize that this is because the network cannot benefit from symmetry or closure in these cases. It seems easier to compensate for occlusion on the left, right or bottom. Occluding the middle of the object is not catastrophic since the network is able to complete the shape. The two leftmost examples in Figure 9 demonstrate that input images depicting very unusual shapes cause difficulties since Glissando-Net has not observed any similar examples in training.

5 Conclusions

We have presented a new deep learning model, Glissando-Net, for simultaneously estimating 6D pose and reconstructing 3D shape at the category level from a single RGB image. This is a research topic largely unexplored in the literature. The only work that attempts to achieve both goals from *real* RGB inputs, to the best of our knowledge, is CPS [7], [8]. Two key design differences enabled Glissando-Net to obtain more accurate prediction of both pose and shape.

First, CPS directly regresses object pose from the image. As a result the predicted pose cannot leverage any prior information conveyed by the point clouds that are only available during training. In contrast, Glissando-Net predicts both pose and shape in the decoder stage of the point cloud VAE, enabling pose prediction to better exploit priors learned from point clouds. Second, to more effectively utilize features from the RGB image, instead of just concatenating the codes, the feature maps extracted from the image encoder are also concatenated to the feature maps in the decoder of the point cloud VAE. This way, the image features are more thoroughly leveraged to achieve more accurate estimation. As validated by our experiments, these contributed to higher prediction accuracy of both object poses and 3D shapes by Glissando-Net, when compared with CPS. Our experiments also demonstrate that the proposed method is relatively robust to occlusion.

Acknowledgments

This research has been partially supported by the National Science Foundation under Awards IIS-1527294 and IIS-1637761.

References

[1] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "CodeSLAM—learning a compact, optimisable representation for dense visual SLAM," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2560–2568.

[2] H. Fan, H. Su, and L. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2463–2471.

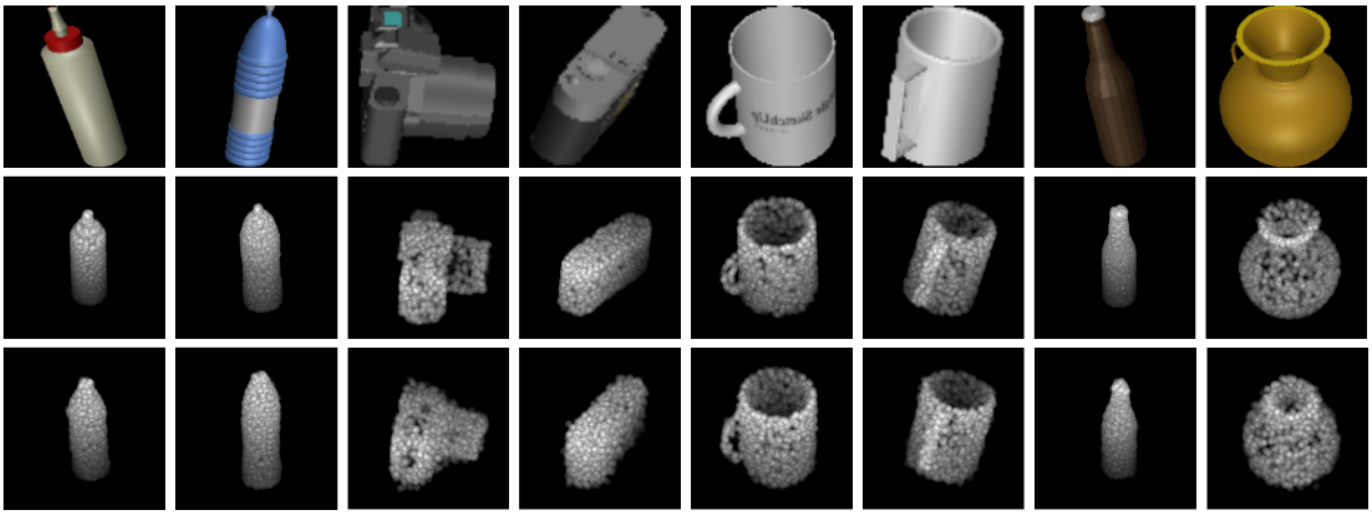


Fig. 7. Qualitative results on *synthetic* NOCS data. Top to bottom: input image, ground-truth 3D shape with ground-truth rotation, and our predicted 3D shape with predicted rotation.

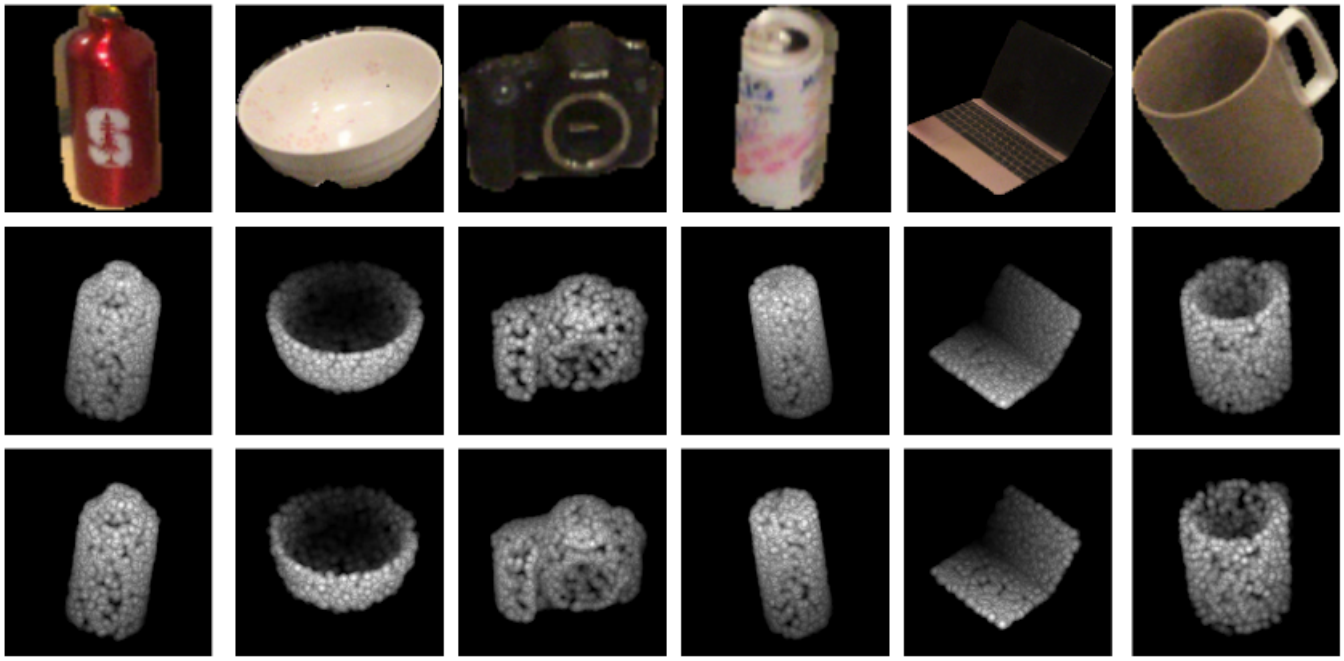


Fig. 8. Qualitative results on *real* NOCS data. Top to bottom: input image, ground-truth 3D shape with ground-truth rotation, and our predicted 3D shape with predicted rotation.

[3] G. Georgakis, S. Karanam, Z. Wu, and J. Kosecka, "Learning local RGB-to-CAD correspondences for object pose estimation," in *International Conference on Computer Vision*, 2019, pp. 8967–8976.

[4] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6d object pose and size estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2642–2651.

[5] 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 *International Conference on Computer Vision*, 2017, pp. 1521–1529.

[6] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *RSS*, 2018.

[7] F. Manhardt, M. Nickel, S. Meier, L. Minciullo, and N. Navab, "CPS: Class-level 6D Pose and Shape Estimation From Monocular Images," *arXiv preprint arXiv:2003.05848v2*, 2020.

[8] F. Manhardt, G. Wang, B. Busam, M. Nickel, S. Meier, L. Minciullo, X. Ji, and N. Navab, "CPS++: Improving Class-level 6D Pose and Shape Estimation From Monocular Images With Self-Supervised Learning," *arXiv preprint arXiv:2003.05848v3*, 2020.

[9] S. Tulsiani, A. Efros, and J. Malik, "Multi-view consistency as supervisory signal for learning shape and pose prediction," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, p. 2897–2905.

[10] E. Insafutdinov and A. Dosovitskiy, "Unsupervised learning of shape and pose with differentiable point clouds," in *Advances in Neural Information Processing Systems*, 2018, pp. 2802–2812.

[11] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman, "Pix3d: Dataset and methods for single-image 3d shape modeling," in *Proceedings of the IEEE Conference on Computer*

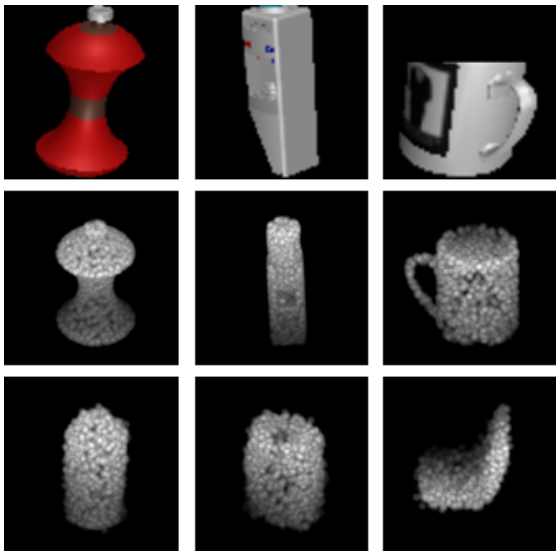


Fig. 9. Some examples where our algorithm fails. The two examples on the left are form the bottle category, while the example on the right is from the mug category with occlusion from the top. The left two examples are of very unusual shapes. The rightmost example is occluded at the top.

Vision and Pattern Recognition, 2018, pp. 2974–2983.

[12] Y. Nie, X. Han, S. Guo, Y. Zheng, J. Chang, and J. J. Zhang, "Total3DUnderstanding: Joint Layout, Object Pose and Mesh Reconstruction for Indoor Scenes from a Single Image," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[13] S. Popov, P. Bauszat, and V. Ferrari, "Corenet: Coherent 3d scene reconstruction from a single rgb image," in *European Conference on Computer Vision*, 2020.

[14] C. Zhang, Z. Cui, Y. Zhang, B. Zeng, M. Pollefeys, and S. Liu, "Holistic 3d scene understanding from a single image with implicit representation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8833–8842.

[15] L. Ke, S. Li, Y. Sun, Y.-W. Tai, and C.-K. Tang, "GSNet: Joint Vehicle Pose and Shape Reconstruction with Geometrical and Scene-aware Supervision," in *European Conference on Computer Vision*, 2020.

[16] S. Sridhar, D. Rempe, J. Valentin, B. Sofien, and L. J. Guibas, "Multiview aggregation for learning category-specific shape reconstruction," in *Advances in Neural Information Processing Systems*, 2019, pp. 2351–2362.

[17] D. Chen, J. Li, and K. Xu, "Learning canonical shape space for category-level 6D object pose and size estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[18] S. Zhi, M. Bloesch, S. Leutenegger, and A. J. Davison, "SceneCode: Monocular Dense Semantic Reconstruction using Learned Encoded Scene Representations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 776–11 785.

[19] M. Bloesch, T. Laidlow, R. Clark, S. Leutenegger, and A. J. Davison, "Learning meshes for dense visual slam," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 5855–5864.

[20] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun, "Monocular 3d object detection for autonomous driving," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2147–2156.

[21] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7074–7082.

[22] 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*, 2015, pp. 1903–1911.

[23] H. Izadinia, Q. Shan, and S. M. Seitz, "IM2CAD," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5134–5143.

[24] X. Chen, Z. Dong, J. Song, A. Geiger, and O. Hilliges, "Category level object pose estimation via neural analysis-by-synthesis," in *European Conference on Computer Vision*, 2020.

[25] K. Park, A. Mousavian, Y. Xiang, and D. Fox, "Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 710–10 719.

[26] K. Park, T. Patten, and M. Vincze, "Neural object learning for 6d pose estimation using a few cluttered images," in *European Conference on Computer Vision*, 2020.

[27] A. Kar, S. Tulsiani, J. Carreira, and J. Malik, "Category-specific object reconstruction from a single image," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[28] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, "3D-R2N2: A unified approach for single and multi-view 3D object reconstruction," in *European Conference on Computer Vision*, 2016, pp. 628–644.

[29] J. Gwak, C. B. Choy, M. Chandraker, A. Garg, and S. Savarese, "Weakly supervised 3d reconstruction with adversarial constraint," in *International Conference on 3D Vision (3DV)*, 2017.

[30] P. O. Pinheiro, N. Rostamzadeh, and S. Ahn, "Domain-adaptive single-view 3d reconstruction," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7638–7647.

[31] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik, "Multi-view supervision for single-view reconstruction via differentiable ray consistency," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[32] X. Yan, J. Yang, E. Yumer, Y. Guo, and H. Lee, "Perspective transformer nets: Learning single-view 3d object reconstruction without 3d supervision," in *Advances in Neural Information Processing Systems*, 2016, pp. 1696–1704.

[33] H. Xie, H. Yao, S. Zhang, S. Zhou, and W. Sun, "Pix2vox++: multi-scale context-aware 3d object reconstruction from single and multiple images," *International Journal of Computer Vision*, vol. 128, no. 12, pp. 2919–2935, 2020.

[34] Y. Jiang, D. Ji, Z. Han, and M. Zwicker, "SDFDiff: Differentiable rendering of signed distance fields for 3d shape optimization," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1251–1261.

[35] Y. Xu, T. Fan, Y. Yuan, and G. Singh, "Ladybird: Quasi-monte carlo sampling for deep implicit field based 3d reconstruction with symmetry," in *European Conference on Computer Vision*, 2020, pp. 248–263.

[36] G. Gkioxari, J. Malik, and J. Johnson, "Mesh R-CNN," in *International Conference on Computer Vision*, 2019.

[37] P. Henderson and V. Ferrari, "Learning single-image 3d reconstruction by generative modelling of shape, pose and shading," *International Journal of Computer Vision*, vol. 128, no. 4, p. 835–854, 2020.

[38] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik, "Learning category-specific mesh reconstruction from image collections," in *European Conference on Computer Vision*, 2018, pp. 371–386.

[39] J. Pan, X. Han, W. Chen, J. Tang, and K. Jia, "Deep mesh reconstruction from single rgb images via topology modification networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 9964–9973.

[40] M. Runz, K. Li, M. Tang, L. Ma, C. Kong, T. Schmidt, I. Reid, L. Agapito, J. Straub, S. Lovegrove *et al.*, "FroDO: From Detections to 3D Objects," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14 720–14 729.

[41] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, "Pixel2mesh: Generating 3d mesh models from single rgb images," in *European Conference on Computer Vision*, 2018, pp. 52–67.

[42] Y. Yao, N. Schertler, E. Rosales, H. Rhodin, L. Sigal, and A. Sheffer, "Front2Back: Single View 3D Shape Reconstruction via Front to Back Prediction," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 531–540.

[43] T. Groueix, M. Fisher, V. G. Kim, B. C. Russell, and M. Aubry, "A papier-mâché approach to learning 3d surface generation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[44] J. Wu, Y. Wang, T. Xue, X. Sun, W. T. Freeman, and J. B. Tenenbaum, "MarrNet: 3D Shape Reconstruction via 2.5D Sketches," in *Advances in Neural Information Processing Systems*, 2017.

[45] K. Navaneet, A. Mathew, S. Kashyap, W.-C. Hung, V. Jampani, and R. V. Babu, "From image collections to point clouds with self-supervised shape and pose networks," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1132–1140.

[46] X. Zhang, Z. Zhang, C. Zhang, J. B. Tenenbaum, W. T. Freeman, and J. Wu, "Learning to Reconstruct Shapes from Unseen Classes," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[47] Y. Ye, S. Tulsiani, and A. Gupta, "Shelf-supervised mesh prediction in the wild," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8843–8852.

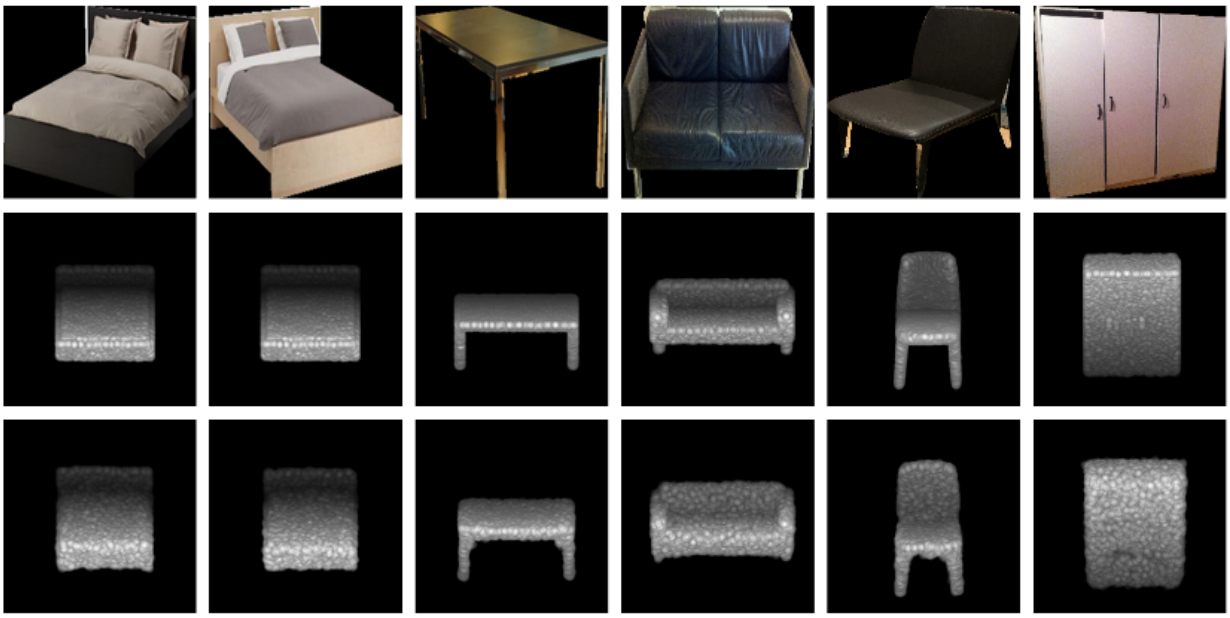


Fig. 10. Qualitative results on Pix3D. Top to bottom: input image, ground-truth 3D shape with ground-truth rotation, and our predicted 3D shape with predicted rotation. *Notice that many of the shapes are partially occluded in the images.*

[48] C. Sahin, G. Garcia-Hernando, J. Sock, and T.-K. Kim, "A review on object pose recovery: From 3d bounding box detectors to full 6d pose estimators," *Image and Vision Computing*, p. 103898, 2020.

[49] T. Hodaň, F. Michel, E. Brachmann, W. Kehl, A. GlentBuch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother, "BOP: Benchmark for 6D Object Pose Estimation," in *European Conference on Computer Vision*, 2018.

[50] A. Avetisyan, M. Dahnert, A. Dai, M. Savva, A. X. Chang, and M. Nießner, "Scan2CAD: Learning CAD model alignment in RGB-D scans," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2614–2623.

[51] W. Chen, X. Jia, H. J. Chang, J. Duan, and A. Leonardis, "G2l-Net: Global to local network for real-time 6D pose estimation with embedding vector features," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 4233–4242.

[52] C. Choi and H. I. Christensen, "3D pose estimation of daily objects using an RGB-D camera," in *IROS*, 2012, pp. 3342–3349.

[53] A. Avetisyan, A. Dai, and M. Nießner, "End-to-End CAD Model Retrieval and 9DoF Alignment in 3D Scans," *arXiv preprint arXiv:1906.04201*, 2019.

[54] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek, and J. Matas, "Detection and fine 3d pose estimation of texture-less objects in rgb-d images," in *IROS*, 2015, pp. 4421–4428.

[55] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab, "Deep learning of local rgb-d patches for 3d object detection and 6d pose estimation," in *European Conference on Computer Vision*, 2016, pp. 205–220.

[56] A. Tejani, R. Kouskouridas, A. Doumanoglou, D. Tang, and T. Kim, "Latent-class hough forests for 6 dof object pose estimation." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 1, pp. 119–132, 2018.

[57] C. Wang, R. Martín-Martín, D. Xu, J. Lv, C. Lu, L. Fei-Fei, S. Savarese, and Y. Zhu, "6-pack: Category-level 6d pose tracker with anchor-based keypoints," in *ICRA*, 2020, pp. 10 059–10 066.

[58] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3343–3352.

[59] 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 *IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3364–3372.

[60] A. Grabner, P. M. Roth, and V. Lepetit, "GP2C: Geometric Projection Parameter Consensus for Joint 3D Pose and Focal Length Estimation in the Wild," in *International Conference on Computer Vision*, 2019, pp. 2222–2231.

[61] Z. Li, G. Wang, and X. Ji, "CDPN: Coordinates-Based Disentangled Pose Network for Real-Time RGB-Based 6-DoF Object Pose Estimation," in *The IEEE International Conference on Computer Vision (ICCV)*, 2019.

[62] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 7668–7677.

[63] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4561–4570.

[64] C. Song, J. Song, and Q. Huang, "Hybridpose: 6d object pose estimation under hybrid representations," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 431–440.

[65] M. Sundermeyer, M. Durner, E. Y. Puang, Z.-C. Marton, and R. Triebel, "Multi-path learning for object pose estimation across domains," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[66] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 292–301.

[67] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox, "DeepIM: Deep iterative matching for 6d pose estimation," in *European Conference on Computer Vision*, 2018, pp. 683–698.

[68] O. Ronneberger, P.Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, ser. LNCS, vol. 9351, 2015, pp. 234–241.

[69] Y. Wu and K. He, "Group normalization," in *European Conference on Computer Vision*, September 2018.

[70] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[71] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *International Conference on Computer Vision*, 2017, pp. 2961–2969.

[72] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "ShapeNet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[73] T. Hodaň, J. Matas, and Š. Obdržálek, "On evaluation of 6d object pose estimation," in *European Conference on Computer Vision*, 2016, pp. 606–619.

[74] 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 *International Conference on Computer Vision*, 2011, pp. 858–865.

Bo Sun is a researcher at Adobe Inc. He earned his PhD from the Stevens Institute of Technology and earned his Bachelor and Master degree from Nankai University, China. His research interests span 3D reconstruction from images, robotics and panoptic segmentation.

Hao Kang is a researcher at Wormpex AI Research since the year of 2019. He received his PhD from Purdue University. He earned his Master degree from Stevens Institute of Technology and Bachelor degree from Beijing Jiaotong University, China. His research interests include computer graphics, computer vision, and human-computer interaction.

Li Guan received his Ph.D. in Computer Science at the University of North Carolina at Chapel Hill in 2009. He is currently a Principal Research Manager at Wormpex AI Research LLC, leading projects to develop 3D solutions for store environment modeling to improve the efficiency of store operations. He is the author of over 20 refereed papers in 3D computer vision papers and more than 17 patents. He is in the organizing committee of the OmniCV (360/ominidirectional computer vision) workshop affiliated with the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2020-2022.

Haoxiang Li received the BS degree in computer science from the University of Science and Technology of China, in 2010. He received the PhD degree from the Department of Computer Science at Stevens Institute of Technology, in 2016. He is currently a Principal Research Manager with Wormpex AI Research. Before that, he was a Researcher with AiBee and Adobe Research. His research interests include human face processing and analysis, computer vision, and robotics. He was an area chair for IEEE Winter Conference on Applications of Computer Vision (WACV) 2020 and 2022.

Philippos Mordohai is a professor of Computer Science at Stevens Institute of Technology. He earned his PhD from the University of Southern California and held postdoctoral appointments at the University of North Carolina and the University of Pennsylvania. His research interests span 3D reconstruction from images and video, range data analysis, perceptual organization and active vision. He serves as an associate editor for the IEEE Transactions on Pattern Analysis and Machine Intelligence, Computer Vision and Image Understanding, and Image and Vision Computing. He was a program co-chair of the International Conference on 3D Vision (3DV), 2019.

Gang Hua (M'03-SM'11-F'19) was enrolled in the Special Class for the Gifted Young of Xi'an Jiaotong University (XJTU), Xi'an, China, in 1994 and received the B.S. degree in Automatic Control Engineering from XJTU in 1999. He received the M.S. degree in Control Science and Engineering in 2002 from XJTU, and the Ph.D. degree in Electrical Engineering and Computer Science at Northwestern University, Evanston, Illinois, USA, in 2006. He is currently the Vice President and Chief Scientist of Wormpex AI Research. Before that, he served in various roles at Microsoft (2015-18) as the Science/Technical Adviser to the CVP of the Computer Vision Group, Director of Computer Vision Science Team in Redmond and Taipei ATL, and Principal Researcher/Research Manager at Microsoft Research. He was an Associate Professor at Stevens Institute of Technology (2011-15). During 2014-15, he took an on leave and worked on the Amazon-Go project. He was an Visiting Researcher (2011-14) and a Research Staff Member (2010-11) at IBM Research T. J. Watson Center, a Senior Researcher (2009-10) at Nokia Research Center Hollywood, and a Scientist (2006-09) at Microsoft Live labs Research. He is an associate editor of TIP, TCSVT, CVIU, IEEE Multimedia, TVCJ and MVA. He also served as the Lead Guest Editor on two special issues in TPAMI and IJCV, respectively. He will be a general chair for ICCV'2025. He is a program chair of CVPR'2019&2022. He is an area chair of CVPR'2015&2017&2021, ICCV'2011&2017, ECCV'2020 ICIP'2012&2013&2016, ICASSP'2012&2013, and ACM MM 2011&2012&2015&2017. He is the author of more than 160 peer reviewed publications in prestigious international journals and conferences. He holds 20 US patents and has 20 more US patents pending. He is the recipient of the 2015 IAPR Young Biometrics Investigator Award for his contribution on Unconstrained Face Recognition from Images and Videos, and a recipient of the 2013 Google Research Faculty Award. He is an IEEE Fellow, an IAPR Fellow, and an ACM Distinguished Scientist.