Voxel-based 3D Detection and Reconstruction of Multiple Objects from a Single Image

Anonymous Author(s)
Affiliation
Address
email

Abstract

Inferring 3D locations and shapes of multiple objects from a single 2D image is a long-standing objective of computer vision. Most of the existing works either predict one of these 3D properties or focus on solving both for a single object. One fundamental challenge lies in how to learn an effective representation of the image that is well-suited for 3D detection and reconstruction. In this work, we propose to learn a regular grid of 3D voxel features from the input image which is aligned with 3D scene space via a 3D feature lifting operator. Based on the 3D voxel features, our novel CenterNet-3D detection head formulates the 3D detection as keypoint detection in the 3D space. Moreover, we devise an efficient coarse-to-fine reconstruction module, including coarse-level voxelization and a novel local PCA-SDF shape representation, which enables fine detail reconstruction and two orders of magnitude faster inference than prior methods. With complementary supervision from both 3D detection and reconstruction, one enables the 3D voxel features to be geometry and context preserving, benefiting both tasks. The effectiveness of our approach is demonstrated through 3D detection and reconstruction on single-object and multiple-object scenarios. Code, dataset, and models will be released to the public upon publication.

1 Introduction

As a fundamental computer vision task, instance-level 3D scene understanding from a single image has drawn substantial attention from researchers due to its importance in applications such as robotics [1, 2], AR/VR [3] and autonomous driving [4–6]. The important 3D properties includes 3D bounding box (pose, size, location) and 3D shape of object instances. In this work, we aim to design a framework to infer all the above information of multiple objects from a single image.

In recent years, various monocular methods are proposed to predict either 3D boxes [7–10] or 3D shapes [11–15]. However, only a few studies [16–21] consider both 3D detection and reconstruction for total 3D scene understanding. The complexity of real-world scenarios and diverse category variations make it challenging to fully reconstruct the scene context (both semantics and geometry) at the instance level from a single image. Moreover, those methods primarily assign 3D semantic labels to pixels. Yet, such a 2D representation with depth ambiguity is insufficient for 3D geometry and context reasoning. It is thus crucial to develop an effective representation of the image that is relevant to 3D geometry and spatial information to perform accurate 3D detection and reconstruction.

In light of this, attempts like grid-based representation have been made for tasks such as rendering [22], detection [23, 24], or reconstruction [25]. OFT [23] proposes to sample and transform image features into a BEV grid representation, which enables to reason holistically about the 3D scene configuration. CaDDN [24] extends the BEV grid representation with a categorical depth prior, leading to higher 3D detection accuracies. DeepVoxels [22] and UCLID-Net [25] build voxel features by back-projecting 2D features to 3D space for respective rendering or single object reconstruction purposes. Inspired by

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

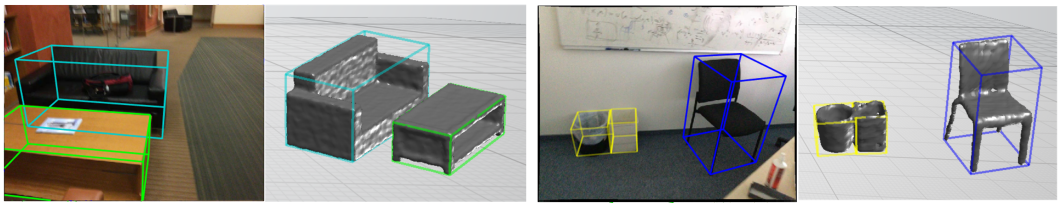


Figure 1: Given a singe image as input, our approach jointly predicts 3D object bounding boxes and surfaces.

this line of works, we propose a novel voxel-based 3D detection and reconstruction framework for predicting 3D bounding boxes and surfaces of multiple objects from a single image (see Fig. 1).

Specifically, we first divide a 3D scene space into a regular grid of voxels. For each voxel, we assign 3D features by sampling from the image plane via a 2D-to-3D feature lifting operator and the known camera projection matrix. As multiple voxels can be projected to the same position, this leads to similar features along the camera ray and increased difficulty for downstream tasks. To remedy this, we use a positional encoding strategy to make our voxel features position-aware and more discriminative. Based on the intermediate voxel features, we carefully devise our detection and reconstruction modules. For detection, we introduce a novel CenterNet-3D detector head. Instead of formulating the 3D detection as 2D keypoint detection problem as conventional CenterNet-based methods [26, 27], each object is directly represented by its 3D keypoint. Predicting a class-specific 3D heatmap can show probabilities of 3D object centers in the pre-defined voxel space, leading to improved 3D center accuracy. For reconstruction, we propose a multi-level shape representation with two components: coarse-level occupancy representation and fine-level local PCA-SDF representation. The coarse-level voxel grid represents the whole 3D scene with continuous occupancy values. At a fine level, we represent the boundary of the coarse-level voxels with a PCA-based signed distance function (SDF) by assuming that the local shapes of different boundary voxels are similar within an object instance and across different objects. In summary contributions of this work include:

◇ We propose a novel voxel-based 3D detection and reconstruction framework, which infers the 3D locations and 3D surfaces for multiple object instances with only a 2D image as input.

◇ We present a novel CenterNet-3D detector, where each object is represented by its center point in a partitioned 3D grid space. CenterNet-3D avoids estimating depth directly from image features, leading to increased detection performance.

◇ We propose a novel local PCA-SDF shape representation, which provides finer reconstruction and orders of magnitude faster inference than SOTA local implicit function methods like DeepLS [28].

◇ We demonstrate the superiority of our method in multi-object 3D reconstruction and detection, as well as 3D shape representation. We assemble a 3D detection and reconstruction benchmark with $20,000+$ real images, annotated with 3D models and bounding boxes of 19 object categories.

2 Related Work

3D Scene Understanding and Single Object Reconstruction. Tremendous efforts have been devoted to instance-level 3D scene understanding [7, 9, 16, 26, 29–35] over the last decade. However, most of these approaches estimate object orientation [32, 33, 36] or 3D bounding boxes [7–9, 37–43]. Since describing objects with boxes only offers a coarse information of 3D objects in images, the usage of 3D models as shape priors can complement and enrich 3D scene understanding. Yet, scene understanding at the instance level remains challenging due to the large number of objects with various categories. With the substantial growth in the number of publicly available 3D models, datasets such as ShapeNet [44] have allowed neural networks to train on the 3D shape reconstruction task from single images [45–50]. However, most of these methods estimate 3D shapes in the object-centric coordinate system, which differs from the shape prediction of multiple instances at scene-level 3D reconstruction – the focus of our work.

Multi-Object 3D Reconstruction A common characteristic amongst aforementioned single object 3D reconstruction approaches is that they usually treat objects as isolated geometries without considering the scene context, such as object locations, and instance-to-instance interactions. Recently, there is progress in multi-object 3D reconstruction. 3D-RCNN [16] exploits the idea of using inverse graphics to map image regions to the 3D shape and pose of object instances. The shape is represented

by a simple linear subspace which limits its application for objects with large intra-class variability. Mesh R-CNN [19] augments Mask R-CNN [51] with a mesh predictions branch that estimates a 3D mesh for each detected in an image. Total3DUnderstanding [17] presents a framework that predicts room layout, 3D object bounding boxes, and meshes for all objects in an image based on the known 2D bounding boxes. However, these three methods first detect objects in the 2D image, and then *independently* produce their 3D shapes with single-object reconstruction modules.

Recently, CoReNet [20] performs multi-object reconstructions in a fixed 128^3 voxel grid while missing 3D position information in the world space. Points2Objects [21] combines a 3D object detector and shape retrieval to detect and reconstruct 3D objects from an image. However, it suffers from two limitations: 1) Its CenterNet-based 3D detector reasons 3D boxes directly on the 2D image domain, which is inherently challenging due to the lack of reliable depth cue. 2) Retrieval-based methods depend on the size and diversity of the pre-defined CAD model pool. Moreover, [20, 21] train on synthetic renderings, which limits applicability to real-world scenarios. Instead of relying on 2D feature for 3D detection or reconstruction, we propose to learn a geometry and context preserving voxel feature representation, which is well-suited for 3D detection and reconstruction. Moreover, we validate our method on real-world images from 19 object categories.

Local Shape Priors Many neural architectures are proposed to model 3D objects via geometric representations, *e.g.*, point clouds [52], meshes [14, 53], voxels [11, 54], or implicit functions [13, 55, 56]. Recently, neural implicit functions have demonstrated their effectiveness by encoding geometry in latent vectors and network weights, which parameterize surfaces through level-sets. Instead of an object-level representation, some follow-up works learn patch-level or primitive-level representations of surfaces, *e.g.*, PatchNet [57], CvxNet [58], BSP-Net [59]. To further leverage local geometric priors, another line of works learns implicit geometry on sparse regular [60, 61] or 3D voxel grids [28, 62]. A latent code of each voxel is responsible for representing implicit geometry in a small neighborhood, enabling fine detailed reconstruction. However, these methods often suffer from inefficient inference as each point needs a forward pass through of the implicit function network. Instead, we build a local PCA-SDF shape representation, which represents each local shape as a linear combination of implicit volumetric prototypes, leading to finer details and orders of magnitude faster inference than prior works. Similar eigenanalysis of SDF has been applied for either global shape representation [63] or geometry compression [64, 65]. However, none of them develop their algorithms from our perspective of local shape priors, which is motivated from the assumption that local shapes at the voxel level share similarities. They also do not support finer or scalable reconstruction.

3 Methodology

We illustrate the overall architecture of our method in Fig. 2, which consists of three key modules: i) 3D voxel feature learning; ii) CenterNet-3D detector; and iii) Coarse-to-fine 3D reconstruction.

3.1 3D Voxel Feature Learning

Our network learns to produce a compact 3D voxel feature representation of the image with *complementary* supervision from both 3D detection and reconstruction, which enables rich 3D context and geometric information and allows the two tasks to benefit each other. We first define a 3D grid $\mathbf{V} \in \mathbb{R}^{X \times Y \times Z}$ by partitioning of the scene space into voxels $\mathbf{V}_i$ with a voxel size of r.

Feature Extraction. We utilize a convolutional feature extractor to generate a hierarchy of multi-scale 2D feature maps. Specifically, the input to the feature extraction network is a RGB image $\mathbf{I} \in \mathbb{R}^{W_I \times H_I \times 3}$. The convolutions are followed by down scaling the input, creating growing receptive fields and resulting in D-channel multi-scale 2D feature maps $\mathbf{F} \in \mathbb{R}^{W_F \times H_F \times D}$.

Lifting 2D Features to 3D. The lifting layer back-projects 2D feature maps $\mathbf{F}$ into 3D voxel space, resulting in *initial* 3D voxel features $\mathbf{G}$. Formally, the objective of the lifting operator is to populate the 3D voxel features $\mathbf{G}(x, y, z)$ with the projected 2D features $\mathbf{F}\{(u, v)\}$, where $\{\cdot\}$ denotes bilinear interpolation on the 2D feature maps. We assume a full perspective camera model. Any voxel center (x, y, z) can be projected to image plane via a camera projection matrix $\mathbf{P} \in \mathbb{R}^{3\times4}$: $[u \cdot d, v \cdot d, d]^T = \mathbf{P}[x, y, z, 1]^T$. Here u and v are the 2D position of the projection and d is its depth from the camera. The resulting voxel features $\mathbf{G} \in \mathbb{R}^{X \times Y \times Z \times D}$ provide a representation of the scene which is free from the effects of perspective projection.

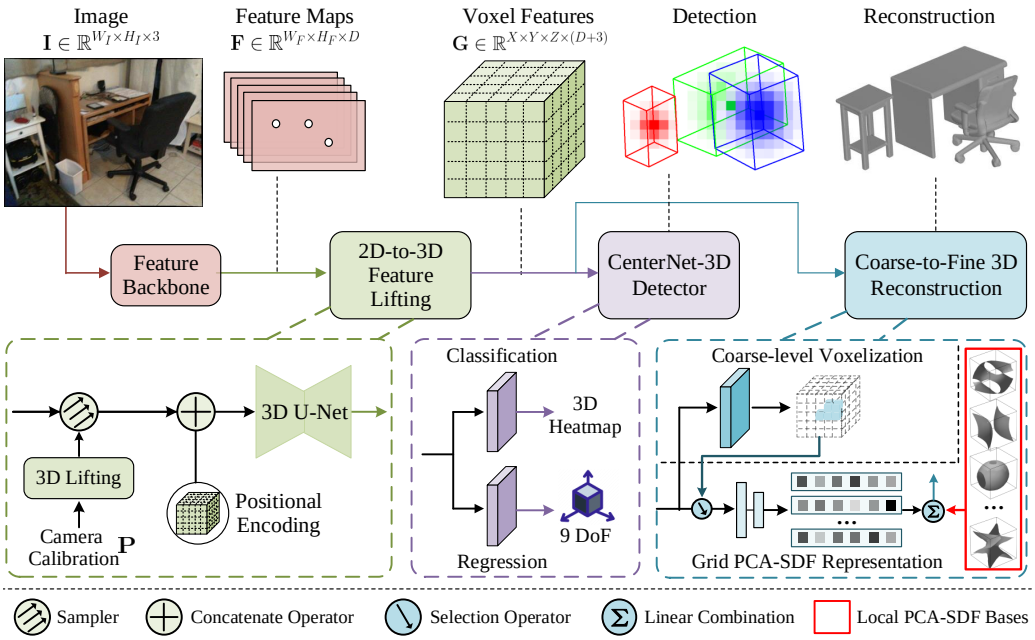


Figure 2: **Overview of our approach.** The proposed joint framework is composed of four modules: 2D feature extraction, 2D-to-3D feature lifting, CenterNet-3D detector, and coarse-to-fine 3D reconstruction. 2D feature maps are first generated from input image $\mathbf{I}$, which are back-projected into voxel features $\mathbf{G}$ using a known camera projection matrix $\mathbf{P}$. The voxel features are used for our novel 3D object detection and reconstruction.

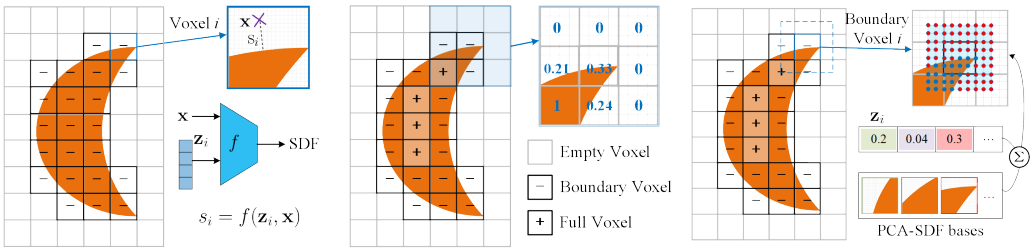
3D Voxel Features Aggregation. The lifting mechanism we use is similar to the one in [22, 23, 25], which has a major weakness that all voxels along a camera ray might receive the same 2D feature. This feature smearing issue increases the difficulties of 3D detection and reconstruction. To mitigate this issue, we first propose to employ the *positional encoding* (PE) [66] strategy that adds channels embedding 3D voxel center position to the voxel features: $\mathbb{R}^{X \times Y \times Z \times D} \xrightarrow{PE} \mathbb{R}^{X \times Y \times Z \times (D+3)}$, which helps the voxel features to be more discriminative and position-embedded. We further utilize a 3D convolutional hourglass (U-Net) network [67], which are composed of a series of down- and upsampling convolutions with skip connections, to integrate both local and global information. The final voxel features is thus $\mathbf{G} : \mathbb{R}^{X \times Y \times Z \times (D+3)} \xrightarrow{U-Net} \mathbb{R}^{X \times Y \times Z \times (D+3)}$. This voxel feature serves as a cornerstone for the downstream tasks of 3D detection and reconstruction.

3.2 Monocular CenterNet-3D Detector

Conventional CenterNet-based [26] monocular 3D detection methods such as Points2Objects [21] formulates 3D detection as a projected 2*D keypoint* detection problem. In contrast, we propose a novel CenterNet-3D detection head, where each object is directly represented by its 3*D keypoint*. Then 3D properties such as object size and orientation can be intuitively inferred from the 3D voxel features at the center location by a regression branch. Compared to [21], CenterNet-3D avoids estimating depth values of 3D boxes directly from 2D features, leading to improved detection accuracy.

3D Keypoint Branch. The 3D keypoint branch takes the voxel features $\mathbf{G}$ as input and predicts a 3*D heatmap* $\mathcal{Y} \in \mathbb{R}^{X \times Y \times Z \times C}$ (see Fig. 2), where C is the number of object categories. The values of each voxel in $\mathcal{Y}$ indicates how likely the 3D centroid of a certain object category exists at the voxel center. By computing the local maxima and filtering via a threshold, we obtain a preliminary estimation of the 3D centroids, denoted as $\tilde{\mathbf{c}}_{3d} = [x_c, y_c, z_c]^T$. To remedy the discretization error of voxels, the regression branch additionally predicts a local offset to $\tilde{\mathbf{c}}_{3d}$, which is discussed next.

Regression Branch. The regression branch predicts the essential properties to construct a 3D bounding box for each voxel of the 3D heatmap. The 3D box information is encoded as a 12-tuple $\tau = [\delta_{x_c}, \delta_{y_c}, \delta_{z_c}, \delta_h, \delta_w, \delta_l, \sin\alpha, \cos\alpha, \sin\beta, \cos\beta, \sin\gamma, \cos\gamma]$. Here $\Delta\mathbf{c}_{3d} = [\delta_{x_c}, \delta_{y_c}, \delta_{z_c}]^T$ denotes the 3D center offset compensating voxel discretization, $[\delta_h, \delta_w, \delta_l]$ the residual of object



(a) (b) (c)

Figure 3: 2D examples of (a) DeepLS [28], (b) our continuous coarse-level voxelization, and (c) our local PCA-SDF shape representation. The SDF function f in DeepLS outputs a scalar value conditional on the local latent code $\mathbf{z}_i$ and local coordinate $\mathbf{x}$. However, its inference is computationally expensive since it requires forward pass through f for every $\mathbf{x}$. Our shape representation consists of coarse-level voxelization and fine-level local PCA-SDF. The coarse-level voxels with continuous occupancy values can be categorized into three types: empty, full and boundary voxels. To further represent fine-level surfaces, we propose a novel PCA-SDF model, representing any boundary voxel as a linear combination of regular SDF function bases, which enables a more efficient and accurate representation than DeepLS.

size $[h, w, l]$, and α, β, γ the rotation Euler angles about the axes of the object coordinate system. Here the network estimates the vectorial representation of those three angles. The output size of the regression branch is thus $X \times Y \times Z \times 12$. Given the outputs of keypoint and regression branches, the 3D bounding box can be restored as 8 corners:

$$\mathcal{B} = R(\alpha, \beta, \gamma) \begin{bmatrix} \pm l/2 \\ \pm h/2 \\ \pm w/2 \end{bmatrix} + \mathbf{c}_{3d}, \quad \mathbf{c}_{3d} = \tilde{\mathbf{c}}_{3d} + \Delta \mathbf{c}_{3d}, \tag{1}$$

Here, $R(\alpha, \beta, \gamma) \in \mathbb{R}^{3\times3}$ is the rotation matrix. $[l, h, w]^T = [\bar{l} \cdot e^{\delta_l}, \bar{h} \cdot e^{\delta_h}, \bar{w} \cdot e^{\delta_w}]^T$, where $[\bar{l}, \bar{h}, \bar{w}]^T$ is a pre-calculated category-wise average box size.

3.3 Coarse-to-Fine 3D Reconstruction

Our reconstruction module is based on a coarse-to-fine shape representation, which consists of two components: coarse-level voxlization and fine-level local PCA-SDF.

Coarse-Level Voxelization. Based on the extracted 3D voxel features $\mathbf{G}$, we first estimate a coarse-level voxelization $\tilde{\mathcal{V}}$ by a specific branch. The coarse-level voxelization holistically represents the whole 3D surface with continuous occupancy values. As shown in Fig. 3(b), the voxels can be divided into three types: full occupancy, empty, and boundary. For only the boundary type, we further reconstruct a fine-level local shape via the local PCA-SDF.

Local PCA-SDF Shape Representation. Recent works such as DeepLS [28] represent 3D surfaces by a set of independent latent codes on a regular grid (see Fig. 3(a)). Each latent code $\mathbf{z}_i$, concatenated with any point location $\mathbf{x}$, can be decoded into a SDF value s_i by the learned implicit network f: $s_i = f(\mathbf{z}_i, \mathbf{x})$. However, this shape representation has two limitations. i) Inference is inefficient (in the order of seconds) since every point of one test voxel (*i.e.,* 256^3 points) is required to be sent to f for SDF calculation, making it unsuitable for real-time applications. ii) Our key observation is that boundary voxels, either within an object instance or across different categories, share similar local shapes, *e.g.*, voxels across the table surface all have planar shapes. However, DeepLS treats voxels as independent training samples, without fully leveraging such local shape priors in training. To address these issues, we propose a novel local PCA-SDF shape representation, which represents each voxel shape as a linear combination of a set of implicit volumetric prototypes, leading to significantly finer reconstruction and $100\times$ faster inference than DeepLS (see Tab. 4).

Formally, as shown in Fig. 3(c), for each boundary voxel $\mathbf{V}$, we define a regular lattice $\mathbf{q} \in \mathbb{R}^{k\times k\times k\times 3}$ and compute their SDFs $\mathbf{s} \in \mathbb{R}^{k\times k\times k\times 1}$ toward the surface. By collecting SDFs of N_S boundary voxels from the training surfaces, we apply Principal Component Analysis (PCA) to find l_B ($l_B << k^3$) local shape bases, $\mathcal{S}_B \in \mathbb{R}^{l_b\times k\times k\times k}$. As such, given the learned $\mathcal{S}_B$ and the latent code $\mathbf{z}_i$, any local boundary shape $\mathcal{S}_i$ of the underlying surface can be implicitly represented by $\mathcal{S}_i = \mathbf{z}_i \mathcal{S}_B$. The latent code $\mathbf{z}_i$ for $\mathcal{S}_i$ can be generated by the corresponding voxel feature $\mathbf{G}_i$ by $\mathbf{z}_i = \text{MLPs}(\mathbf{G}_i)$. MLPs is a mapping network, implemented with two fully-connected layers. By combining the contributions of all the boundary voxels, we can infer a global iso-surface from the SDF field. Similar

Table 1: **Multiple-object reconstruction comparison.** We report per-class and mean IoU over all classes, and class-agnostic global IoU on 128^3 voxel grid.

Method	ShapeNet-triplets								ShapeNet-pairs	
	bottle	bowl	chair	mug	sofa	table	mean	global	mean	global
CoReNet [20]	61.8	36.2	30.1	48.0	52.9	34.8	43.9	49.8	43.1	52.7
Points2Objects [21]	63.5	30.2	18.9	41.5	44.5	19.8	36.4	44.7	–	–
Proposed	**67.2**	**41.6**	**32.0**	**51.3**	**54.1**	**39.5**	**47.6**	**55.2**	**48.2**	**56.7**

to DeepLS [28], we apply a 1.5 times receptive field strategy to avoid the inconsistent surface estimates at the voxel boundaries (Fig. 3(c)). *It is worth mentioning that our local PCA-SDF allows reconstruction at resolutions higher than the one used during training by simply applying trilinear interpolation on the learned local shape bases $\mathcal{S}_B$.*

3.4 Loss Functions and Implementation Details

The training data of one example consists of RGB image $\mathbf{I}$, ground-truth 3D bounding boxes $\mathcal{B}^*$ of objects, coarse-level voxlization $\mathcal{V}^*$, and a set of regular SDF pairs $\{(\text{idx}_j, \mathbf{s}_j)\}_{j=1}^K$ sampled from the surface boundary. Here, each set of SDFs $\mathbf{s}_j \in \mathbb{R}^{k\times k\times k}$, idx_j is the idx_j-th voxel of the holistic grid. During training, we jointly optimize the parameters of 2D feature extraction network f, 3D U-Net, detection and reconstruction modules by minimizing three losses: 3D keypoint classification loss $\mathcal{L}_{cls}$, regression loss $\mathcal{L}_{reg}$, and 3D reconstruction loss $\mathcal{L}_{recon}$, *i.e.,* $\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{reg} + \mathcal{L}_{recon}$.

Loss Functions. We generate the target heatmaps $\mathcal{Y}^*$ by splatting the ground truth 3D center points using a Gaussian kernel (please refer to **Supp** for details). If two Gaussians of the same class overlap, we take the element-wise maximum. The 3D keypoint branch is trained with a penalty-reduced focal loss [26] in a point-wise manner on the 3D heatmap.

$$\mathcal{L}_{cls} = \frac{-1}{N} \sum_{xyzc} \begin{cases} (1-\mathcal{Y}_{xyzc})^{\mu} \log(\mathcal{Y}_{xyzc}) & \text{if} \quad \mathcal{Y}_{xyzc} = 1 \\ (1-\mathcal{Y}^*_{xyzc})^{\sigma} (\mathcal{Y}_{xyzc})^{\mu} \log(1-\mathcal{Y}_{xyzc}) & \text{otherwise} \end{cases} \tag{2}$$

where N is the number of objects per image, $\mu = 2$ and $\sigma = 4$ are hyper-parameters of the focal loss.

We define the 3D bounding box regression loss as the L_1 distance between the predicted transform $\mathcal{B}$ and the ground-truth $\mathcal{B}^*$, $\mathcal{L}_{reg} = \frac{1}{N}||\mathcal{B} - \mathcal{B}^*||_1$. The reconstruction loss consists of coarse-level voxelization regression loss and fine-level SDF regression loss.

$$\mathcal{L}_{recon} = ||\mathcal{V} - \mathcal{V}^*||_1 + \sum_j^K ||\mathbf{z}_j \mathcal{S}_B - \mathbf{s}_j||_1, \quad \mathbf{z}_j = \text{MLPs}(\mathbf{G}_{\text{idx}_j}). \tag{3}$$

Implementation Detail. We use a ResNet-34 network without as our 2D feature extractor. We extract features immediately before the final three downsampling layers, resulting in a set of feature maps at scales of $1/8$, $1/16$, $1/32$ of the input resolution. We then resize them to the original size of the input images via bilinear interpolation. Convolutional layers with 1×1 kernels are used to map these feature maps to a common channel size of 64. The set of feature maps is concatenated as $\mathbf{F}$ (with $D = 64 \times 3$ channels) before passing to 3D lifting layers. We train our models with a batch size of 4 on a GTX 1080Ti GPU for 60 epochs. The learning rate is set at 5×10^{-4} and drops at 20 and 40 epochs by a factor of 10. For more details, please refer to Sec. 4 or **Supp**.

4 Experiments

4.1 Multiple-Object Detection and Reconstruction on ShapeNet-pairs and -triplets

Datasets. Following the experimental setting of CoReNet [20] and Points2Objects [21], we evaluate multi-object detection and reconstruction on *ShapeNet-pairs* and *ShapeNet-triplets* datasets [20]. These datasets contain 256×256 photorealistic renderings of either pairs or triplets of ShapeNet [44] objects placed on a ground plane with random scale, rotation, and camera viewpoint.

Experimental Settings. Since the 3D surfaces of these two datasets have been normalized in $[0, 1]$, we use a voxel grid of size $X \times Y \times Z$ = $30 \times 30 \times 30$ in range of $[-0.1, 1.1]$ ($r = 0.04$), which is sufficient to enclose all objects in the datasets. We randomly select 200 surfaces from the training set to generate $N_S \approx 200,000$ boundary voxels. In this experiment, we set k=9, l_B=32, C=6 (ShapeNet-triplets) or C=13 (ShapeNet-pairs). We compare with SOTA methods for multi-object

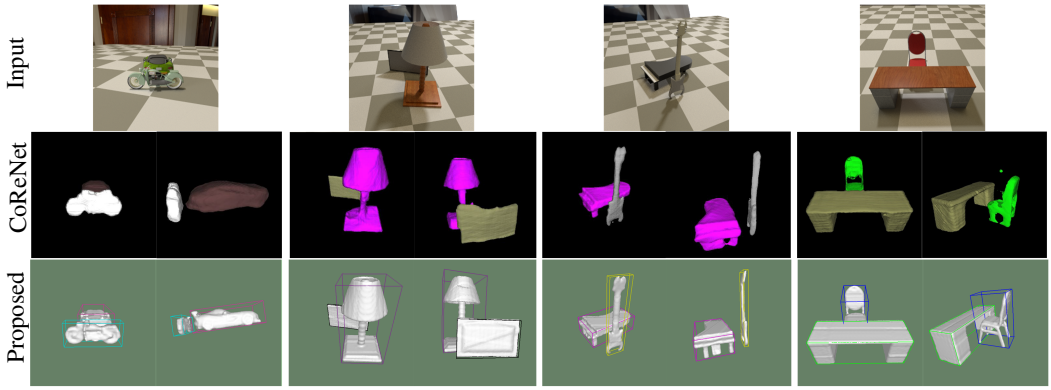


Figure 4: **Qualitative results on ShapeNet-triplets dataset.** We compare to CoReNet [20] in two different viewpoints. Our model more accurately reconstructs details and hallucinates occluded parts.

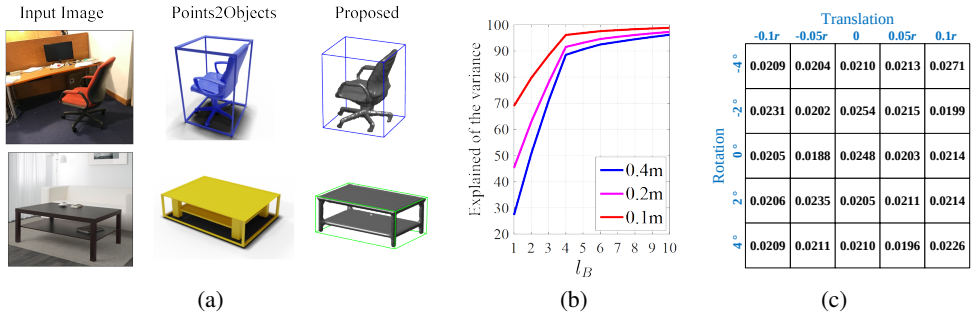


Figure 5: (a) Qualitative comparison on Pix3D. Our reconstructions closely match objects' genuine shape, *e.g.,* the table legs and chair arms. (b) Explained variation of our PCA-SDF representation with three voxel sizes r. (c) Reconstruction errors (Chamfer Distance-$L2$) of PCA-SDF w.r.t. translated and rotated 3D shapes.

reconstruction: CoReNet [20] and Points2Objects [21]. As CoReNet doesn't perform 3D detection, we only compare with Points2Objects on detection. Following [21], we use mean average precision (mAP) as the detection metric with 3D box IoU threshold 0.25 and 0.5. Following [20,21], the metric for reconstruction is intersection-over-union (IoU) on a 128^3 voxel grid.

Results. We first report the 3D detection results. Our method achieves a higher detection accuracy than Points2Objects [21]: **53.3**% *vs.* 48.6% (threshold@0.5) and **82.1**% *vs.* 77.2% (threshold@0.25), which demonstrates that voxel features perform better than image-based features for monocular 3D detection. As compared in Tab. 1, our method significantly outperforms two SOTA baselines on both datasets, On ShapeNet-triplets, our method achieves 5.4% global IoUs gains while 3.7% mean IoUs gains, which indicates that our model performs well on reconstructing the overall shapes of objects. Qualitative results of detection and reconstruction are shown in Fig. 4.

4.2 Single Object Reconstruction on Pix3D

We further compare to CoReNet [20] and Points2Objects [21] on real image database, Pix3D [68] in the same protocol (splits S_1 and S_2) as in [19]. In this experiment, we train our model with the same experimental setting and the same pre-computed PCA-SDF bases as in Sec. 4.1. On average IoU over all 9 object classes, we achieve **42.6**% $vs.$ 34.1% (CoReNet) $vs.$ 33.3% (Points2Objects) on S_1 and **30.6**% $vs.$ 26.3% (CoReNet) $vs.$ 23.6% (Points2Objects) on S_2. The results demonstrate that our approach improves over baselines on real images. Qualitative results are shown in Fig. 5(a).

4.3 Multiple-Object Detection and Reconstruction on ScanNet-MDR

Dataset. Since there is no benchmark providing both 3D CAD models and 3D bounding boxes for multiple objects within a single real image. We thus assemble a dataset with 20, 550 real images from the ScanNet [69], termed ScanNet Monocular Detection and Reconstruction (ScanNet-MDR) dataset. For each object in an image, its CAD model is produced by [70]. We then generate the corresponding 3D bounding box label and camera calibration matrix. Unlike the ShapeNet-pairs or ShapeNet-triplets datasets, all the 3D objects in ScanNet-MDR are at absolute scale. Additionally, this dataset contains greater diversity including 19 object categories (trash, bag, basket, bathtub, bed,

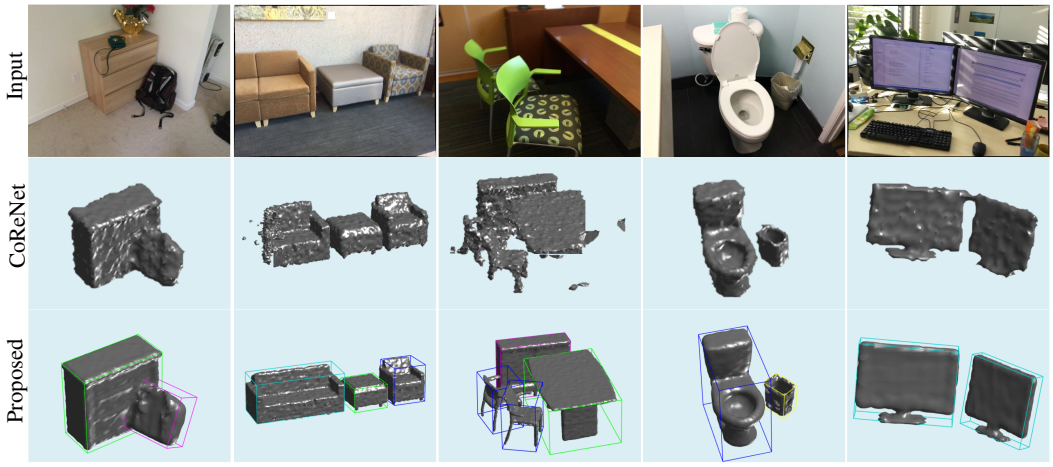


Figure 6: **Qualitative results on real images from ScanNet-MDR.** Our reconstructions closely match the objects than CoReNet [20]. Moreover, our method performs better for reconstruction for the truncated objects.

Table 2: Comparisons of 3D object detection and reconstruction on ScanNet-MDR dataset. [Key: ①=CenterNet, ②=CenterNet-3D], ①=DeepLS, ②=Local PCA-SDF]

Method	Detection		Recon.		Evaluation	
	①	②	①	②	mAP (@0.15)	IoU
CoReNet [20]			–	–	–	35.2
Proposed-1				✓	–	37.6
Proposed-2		✓			24.9	–
Proposed-3	✓			✓	23.2	38.2
Proposed-4		✓	✓		23.9	36.3
Proposed		✓		✓	**26.7**	**39.8**

Table 3: Effect of voxel size r and latent code size l_B in detection and reconstruction on ScanNet-MDR dataset. (mAP/IoU)

r \ l_B	16	32	64	Time (ms)
$0.4m$	22.6/ 32.7	22.9/ 33.1	23.4/ 33.9	46.8
$0.2m$	25.7/ 36.4	25.4/ 37.2	25.5/ 37.9	57.2
$0.1m$	26.5/ 39.2	**26.7**/ **39.8**	26.2/ 38.9	80.1

bench, bookshelf, cabinet, chair, display, file, lamp, microwave, piano, printer, sofa, stove, table and washer). We split the data into $15,990$ for training and $4,560$ for testing.

Experimental Settings. In this experiment, we use a voxel grid of size $6.4 \times 6.4 \times 6.4m$ (X=Y=Z=64, $r = 0.1m$). The PCA-SDF is pre-computed with $N_S \approx 560,000$ boundary voxels from 200 training surfaces. We set $k = 13$, $C = 19$ and $l_B = 32$. For comparison, we train CoReNet [20] using the released code on our training data. We use mAP with 3D box IoU threshold of 0.15 as detection metric [17], and global 3D IoU on a 128^3 voxel grid as reconstruction metric.

Results and Ablation Studies. We report detection and reconstruction results on the testing set. As shown in Tab. 2, our method significantly improves over CoReNet [20] on 3D reconstruction and advances the ablated versions on both detection and reconstruction. Qualitative results are shown in Fig. 6 and **Supp**. CoReNet, as an image-to-voxel reconstruction network without special design for feature smearing issue, cannot handle truncated objects in the input image.

Joint Framework vs. Separate Modules. Moreover, Tab. 2 shows the ablation results of our models without detection (Proposed-1) or reconstruction (Proposed-2) modules, where one can conclude that joint framework in this work performs better than solving either task exclusively.

CenterNet-3D and PCA-SDF. To further validate the effectiveness of the proposed CenterNet-3D detector over the conventional CenterNet, we train a model (Proposed-3) by combining the reconstruction module with the conventional CenterNet, which formulates 3D detection as 2D keypoint detection problem directly from the pixel-based image features. As compared in Tab. 2, our model outperforms Proposed-3 in both detection and reconstruction. To compare PCA-SDF with DeepLS [28] in the joint detection and reconstruction framework, we train a model (Proposed-4) by using DeepLS representation as our fine-level reconstruction module. Tab. 2 shows that both detection and reconstruction performances are worse than ours.

Effect on Voxel Size r Latent Code Size l_B. The validity of local shape pair, expressed by PCA-SDF in our work, depends on the voxel size. For instance, we show the percentage of explained variation

Table 4: Comparison of reconstructing 3D shapes from ShapeNet test set, evaluated by Chamfer Distance-$L2$ (multiplied by 10^3). PCA-SDF achieves higher accuracy and efficiency than DeepLS even with fewer number of decoder and representation parameters. [Key: **Best**, **Second Best**]

Method	chair	plane	table	lamp	sofa	mean	Unseen	Decoder Para.	Represent. Para.	Inference Time
DeepSDF [56]	0.204	0.143	0.553	0.832	0.132	0.372	-	1.8M	**0.3**K	11.0872s
DeepLS [28] (l_B=125)	0.030	**0.018**	0.032	0.078	0.044	0.040	-	0.05M	4096K	3.1851s
PCA-SDF (l_B=32)	0.023	0.031	0.034	0.041	0.045	0.035	0.111	**0.02**M	1049K	**0.0126**s
PCA-SDF (l_B=64)	**0.014**	0.025	**0.025**	**0.036**	**0.040**	**0.028**	0.059	0.05M	2097K	0.0129s
PCA-SDF (l_B=125)	**0.011**	**0.023**	**0.020**	**0.033**	**0.037**	**0.025**	**0.028**	0.09M	4096K	0.0132s

for three voxel sizes in Fig. 5(b). As larger voxel sizes are used, the first few bases could explain less variation, due to the diminished local shape similarity among larger boundary voxels, *i.e.*, weakened local shape prior. This is also validated by the ablation of voxel size r and latent code size l_B in Tab. 3, where larger voxel sizes lead to lower detection and reconstruction accuracies. On the other hand, a larger latent code size results in better representation power (Fig. 5(b)) but not necessarily reconstruction, since it imposes a more challenging task for the network to predict a higher-dim code.

Computation Time. Tab. 3 validates our inference time with different voxel sizes on a GTX 1080Ti GPU. As l_B does not affect the runtime much, we show the average across three l_B. Even with a small voxel size $r = 0.1m$, our method is more efficient than CoReNet [20]: **80.1** *vs.* $260.3ms$.

4.4 3D Shape Representation Power of PCA-SDF

While PCA-SDF has demonstrated its advantage in 2D to 3D reconstruction, this is rooted from its ability in representing 3D shapes. To quantify its 3D shape representation power, we design the following experiment and compare with DeepSDF [56] and DeepLS [28], without involving 2D image inputs. Following the setting of [28], we utilize $1,000$ ShapeNet shapes (200 each from 5 categories) to compute our PCA-SDF bases. Each 3D shape is split by a $32\times32\times32$ grids ($r = \frac{1}{32}$). During training, both DeepSDF and DeepLS optimize the latent codes and decoder weights through backpropagation, for best representing the training shapes. In inference, decoder weights are fixed, and optimal latent codes are estimated given a testing shape. In contrast, we compute the latent codes whose multiplication with PCA-SDF bases can best approximate the ground-truth SDF.

We evaluate 3D shape reconstruction accuracy on various categories. As shown in Tab. 4, PCA-SDF has lower reconstruction error than DeepLS, even with a smaller number of representation parameters. Moreover, our inference is $100\times$ more efficient (infer at 256^3 resolution) which meets the real-time requirement for downstream tasks.

To study whether the PCA-based representation is sensitive to tiny geometry perturbation, we apply minimal translation ($\pm0.1r$, $\pm0.05r$) and rotation ($\pm4^\circ$, $\pm2^\circ$) to testing surfaces of the 5 categories and evaluate surface reconstruction error on these data, while no data augmentation was applied to the training data of shape bases computing. As shown in Fig. 5(c), the error is stable in a very small range, which illustrates that PCA-SDF is robust to translation and rotation variations.

Generalization to Unseen Category. In order to investigate the generalization of PCA-SDF, we design an experiment to compute PCA-SDF from a single category (200 shapes), and reconstruct 3D shapes from the other four unseen categories. We repeat the training/testing 5 times across 5 categories. As reported in Tab. 4, PCA-SDF trained on a single category achieves a comparable performance (0.028 *vs.* 0.025) with the one trained on 5 categories when $l_B = 125$, which indicates that local 3D shapes at the voxel level are indeed similar to each other, even across different categories.

5 Conclusion

We present a voxel-based 3D detection and reconstruction framework for predicting 3D bounding boxes and shapes of multiple objects from a single image. Specifically, we first learn a regular grid of 3D voxel features for the input images. Based on the voxel features, we devise a novel CenterNet-3D detection head to detect and regress 3D bounding boxes in the 3D space. With a coarse-level voxelization and a fine-level local PCA-SDF representation, our reconstruction module provides high efficient and accurate reconstructions. The comprehensive experimental results show the superiority of the proposed method in 3D detection and reconstruction, as well as shape representation power. Same as CoReNet and Points2Objects, one limitation of our approach is that it requires camera calibration matrix as input which might limit its application to real images. Therefore, one future direction is to invest the necessity of this requirement and/or integrate with auto-calibration methods.

References

[1] Joris Guerry, Alexandre Boulch, Bertrand Le Saux, Julien Moras, Aurélien Plyer, and David Filliat. Snapnet-r: Consistent 3D multi-view semantic labeling for robotics. In *ICCV*, 2017.

[2] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. CNN-SLAM: Real-time dense monocular slam with learned depth prediction. In *CVPR*, 2017.

[3] Lei Han, Tian Zheng, Yinheng Zhu, Lan Xu, and Lu Fang. Live semantic 3D perception for immersive augmented reality. *TVCG*, 2020.

[4] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3D scene flow estimation in autonomous driving scenarios? In *ICCV*, 2017.

[5] Yiping Chen, Jingkang Wang, Jonathan Li, Cewu Lu, Zhipeng Luo, Han Xue, and Cheng Wang. LiDAR-video driving dataset: Learning driving policies effectively. In *CVPR*, 2018.

[6] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *CVPR*, 2012.

[7] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. Monocular 3D object detection for autonomous driving. In *CVPR*, 2016.

[8] Siyuan Huang, Siyuan Qi, Yinxue Xiao, Yixin Zhu, Ying Nian Wu, and Song-Chun Zhu. Cooperative holistic scene understanding: Unifying 3D object, layout, and camera pose estimation. In *NeurIPS*, 2018.

[9] Garrick Brazil and Xiaoming Liu. M3D-RPN: Monocular 3D region proposal network for object detection. In *ICCV*, 2019.

[10] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. Monopair: Monocular 3D object detection using pairwise spatial relationships. In *CVPR*, 2020.

[11] Jiajun Wu, Yifan Wang, Tianfan Xue, Xingyuan Sun, Bill Freeman, and Josh Tenenbaum. Marrnet: 3D shape reconstruction via 2.5D sketches. In *NeurIPS*, 2017.

[12] Jun-Yan Zhu, Zhoutong Zhang, Chengkai Zhang, Jiajun Wu, Antonio Torralba, Josh Tenenbaum, and Bill Freeman. Visual object networks: Image generation with disentangled 3D representations. In *NeurIPS*, 2018.

[13] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *CVPR*, 2019.

[14] Thibault Groueix, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry. Atlasnet: A papier-mâché approach to learning 3D surface generation. In *CVPR*, 2018.

[15] Chao Wen, Yinda Zhang, Zhuwen Li, and Yanwei Fu. Pixel2Mesh++: multi-view 3D mesh generation via deformation. In *ICCV*, 2019.

[16] Abhijit Kundu, Yin Li, and James M Rehg. 3D-RCNN: Instance-level 3D object reconstruction via render-and-compare. In *CVPR*, 2018.

[17] Yinyu Nie, Xiaoguang Han, Shihui Guo, Yujian Zheng, Jian Chang, and Jian Jun Zhang. Total3DUnderstanding: Joint layout, object pose and mesh reconstruction for indoor scenes from a single image. In *CVPR*, 2020.

[18] Martin Runz, Kejie Li, Meng Tang, Lingni Ma, Chen Kong, Tanner Schmidt, Ian Reid, Lourdes Agapito, Julian Straub, Steven Lovegrove, et al. FroDO: From detections to 3D objects. In *CVPR*, 2020.

[19] Justin Johnson Georgia Gkioxari, Jitendra Malik. Mesh R-CNN. In *ICCV*, 2019.

[20] Stefan Popov, Pablo Bauszat, and Vittorio Ferrari. CoReNet: Coherent 3D scene reconstruction from a single RGB image. In *ECCV*, 2020.

[21] Francis Engelmann, Konstantinos Rematas, Bastian Leibe, and Vittorio Ferrari. From points to multi-object 3D reconstruction. In *CVPR*, 2021.

[22] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhofer. DeepVoxels: Learning persistent 3D feature embeddings. In *CVPR*, 2019.

[23] Thomas Roddick, Alex Kendall, and Roberto Cipolla. Orthographic feature transform for monocular 3D object detection. In *BMVC*, 2018.

[24] Cody Reading, Ali Harakeh, Julia Chae, and Steven L Waslander. Categorical depth distribution network for monocular 3D object detection. In *CVPR*, 2021.

[25] Benoit Guillard, Edoardo Remelli, and Pascal Fua. UCLID-Net: Single view reconstruction in object space. In *NeurIPS*, 2020.

[26] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *arXiv preprint arXiv:1904.07850*, 2019.

[27] Francis Engelmann, Jörg Stückler, and Bastian Leibe. Joint object pose estimation and shape reconstruction in urban street scenes using 3D shape priors. In *GCPR*, 2016.

[28] Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local SDF priors for detailed 3D reconstruction. In *ECCV*, 2020.

[29] Muhammad Zeeshan Zia, Michael Stark, and Konrad Schindler. Are cars just 3D boxes?-jointly estimating the 3d shape of multiple objects. In *CVPR*, 2014.

[30] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teuliere, and Thierry Chateau. Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image. In *CVPR*, 2017.

[31] Chi Li, M Zeeshan Zia, Quoc-Huy Tran, Xiang Yu, Gregory D Hager, and Manmohan Chandraker. Deep supervision with shape concepts for occlusion-aware 3D object parsing. In *CVPR*, 2017.

[32] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for CNN: Viewpoint estimation in images using cnns trained with rendered 3D model views. In *ICCV*, 2015.

[33] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *CVPR*, 2015.

[34] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Data-driven 3D voxel patterns for object category recognition. In *CVPR*, 2015.

[35] M Zeeshan Zia, Michael Stark, and Konrad Schindler. Towards scene understanding with detailed 3D object representations. *IJCV*, 112(2):188–203, 2015.

[36] Patrick Poirson, Phil Ammirato, Cheng-Yang Fu, Wei Liu, Jana Kosecka, and Alexander C Berg. Fast single shot detection and pose estimation. In *3DV*, 2016.

[37] Yousef Atoum, Joseph Roth, Michael Bliss, Wende Zhang, and Xiaoming Liu. Monocular video-based trailer coupler detection using multiplexer convolutional neural network. In *ICCV*, 2017.

[38] Xiaozhi Chen, Kaustav Kundu, Yukun Zhu, Andrew G Berneshawi, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 3D object proposals for accurate object class detection. In *NeurIPS*, 2015.

[39] Bin Xu and Zhenzhong Chen. Multi-level fusion based 3D object detection from monocular images. In *CVPR*, 2018.

[40] Andrea Simonelli, Samuel Rota Bulo, Lorenzo Porzi, Manuel López-Antequera, and Peter Kontschieder. Disentangling monocular 3D object detection. In *ICCV*, 2019.

[41] Lijie Liu, Jiwen Lu, Chunjing Xu, Qi Tian, and Jie Zhou. Deep fitting degree scoring network for monocular 3D object detection. In *CVPR*, 2019.

[42] Wongun Choi, Yu-Wei Chao, Caroline Pantofaru, and Silvio Savarese. Understanding indoor scenes using 3D geometric phrases. In *CVPR*, 2013.

[43] Siyuan Huang, Siyuan Qi, Yixin Zhu, Yinxue Xiao, Yuanlu Xu, and Song-Chun Zhu. Holistic 3D scene parsing and reconstruction from a single RGB image. In *ECCV*, 2018.

[44] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[45] Zi-Hang Jiang, Qianyi Wu, Keyu Chen, and Juyong Zhang. Disentangled representation learning for 3D face shape. In *CVPR*, 2019.

[46] Anurag Ranjan, Timo Bolkart, Soubhik Sanyal, and Michael J Black. Generating 3D faces using convolutional mesh autoencoders. In *ECCV*, 2018.

[47] Feng Liu, Luan Tran, and Xiaoming Liu. 3D face modeling from diverse raw scan data. In *ICCV*, 2019.

[48] Timur Bagautdinov, Chenglei Wu, Jason Saragih, Pascal Fua, and Yaser Sheikh. Modeling facial geometry using compositional VAEs. In *CVPR*, 2018.

[49] Angela Dai, Charles Ruizhongtai Qi, and Matthias Nießner. Shape completion using 3D-encoder-predictor CNNs and shape synthesis. In *CVPR*, 2017.

[50] David Stutz and Andreas Geiger. Learning 3D shape completion from laser scan data with weak supervision. In *CVPR*, 2018.

[51] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

[52] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, 2017.

[53] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2mesh: Generating 3D mesh models from single RGB images. In *ECCV*, 2018.

[54] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In *ECCV*, 2016.

[55] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3D reconstruction in function space. In *CVPR*, 2019.

[56] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019.

[57] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *ICCV*, 2019.

[58] Boyang Deng, Kyle Genova, Soroosh Yazdani, Sofien Bouaziz, Geoffrey Hinton, and Andrea Tagliasacchi. CvxNet: Learnable convex decomposition. In *CVPR*, 2020.

[59] Zhiqin Chen, Andrea Tagliasacchi, and Hao Zhang. BSP-Net: Generating compact meshes via binary space partitioning. In *CVPR*, 2020.

[60] Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, Thomas Funkhouser, et al. Local implicit grid representations for 3D scenes. In *CVPR*, 2020.

[61] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural geometric level of detail: Real-time rendering with implicit 3D shapes. In *CVPR*, 2021.

[62] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. In *NeurIPS*, 2020.

[63] Mateusz Michalkiewicz, Eugene Belilovsky, Mahsa Baktashmotlagh, and Anders Eriksson. A simple and scalable shape representation for 3d reconstruction. In *BMVC*, 2020.

[64] Daniel Ricao Canelhas, Erik Schaffernicht, Todor Stoyanov, Achim J Lilienthal, and Andrew J Davison. Compressed voxel-based mapping using unsupervised learning. *Robotics*, 2017.

[65] Danhang Tang, Mingsong Dou, Peter Lincoln, Philip Davidson, Kaiwen Guo, Jonathan Taylor, Sean Fanello, Cem Keskin, Adarsh Kowdle, Sofien Bouaziz, et al. Real-time compression and streaming of 4D performances. *TOG*, 2018.

[66] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[67] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In *MICCAI*, 2016.

[68] Xingyuan Sun, Jiajun Wu, Xiuming Zhang, Zhoutong Zhang, Chengkai Zhang, Tianfan Xue, Joshua B Tenenbaum, and William T Freeman. Pix3D: Dataset and methods for single-image 3D shape modeling. In *CVPR*, 2018.

[69] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3D reconstructions of indoor scenes. In *CVPR*, 2017.

[70] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X Chang, and Matthias Nießner. Scan2CAD: Learning CAD model alignment in RGB-D scans. In *CVPR*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Please see Section 5.
   (c) Did you discuss any potential negative societal impacts of your work? [No] The applications of this work could be autonomous driving, AR/VR, depth perception and etc. There are no obvious negative societal impacts of our work.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [No] Code, dataset, and models will be released to the public upon publication.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please see the paragraph of implementation detail, Section 4 and supplementary.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A] We will release source code and trained models for reproducibility upon publication.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please refer to the paragraph of implementation detail.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] We use ShapeNet, ShapeNet-pairs, ShapeNet-triplets, Pix3D and ScanNet datasets for evaluation and we properly cite them all.
   (b) Did you mention the license of the assets? [No] All the data we use in this work is publicly available and is for non commercial research purposes.
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes] All the data we use in this work is publicly available.
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No] The data we use in this work do not contain any personally identifiable information or offensive content.
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]