2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

FSD: Fast Self-Supervised Single RGB-D to Categorical 3D Objects

Mayank Lunayach[1] Sergey Zakharov[2] Dian Chen[2] Rares Ambrus[2] Zsolt Kira[1] Muhammad Zubair Irshad[1]
[1]Georgia Institute of Technology [2]Toyota Research Institute
{lunayach, mirshad7, zkira}@gatech.edu, {firstname.lastname}@tri.global

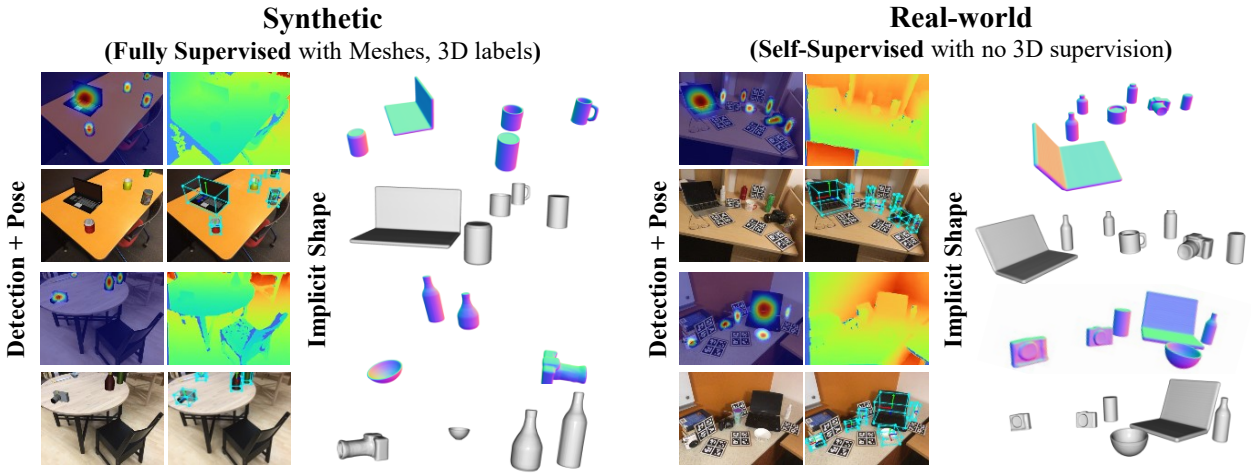

Fig. 1: **Overview:** We present **FSD**, a fast self-supervised categorical 6D pose and size estimation and shape reconstruction framework. Our method is a fully feed-forward approach that doesn't require any real-world 3D labels such as meshes or 6D pose annotations and it does not necessitate inference time optimization. Our results demonstrate effective sim-to-real transfer on the NOCS [46] real-world test-set.

***Abstract*— In this work, we address the challenging task of 3D object recognition without the reliance on real-world 3D labeled data. Our goal is to predict the 3D shape, size, and 6D pose of objects within a single RGB-D image, operating at the category level and eliminating the need for CAD models during inference. While existing self-supervised methods have made strides in this field, they often suffer from inefficiencies arising from non-end-to-end processing, reliance on separate models for different object categories, and slow surface extraction during the training of implicit reconstruction models; thus hindering both the speed and real-world applicability of the 3D recognition process. Our proposed method leverages a multi-stage training pipeline, designed to efficiently transfer synthetic performance to the real-world domain. This approach is achieved through a combination of 2D and 3D supervised losses during the synthetic domain training, followed by the incorporation of 2D supervised and 3D self-supervised losses on real-world data in two additional learning stages. By adopting this comprehensive strategy, our method successfully overcomes the aforementioned limitations and outperforms existing self-supervised 6D pose and size estimation baselines on the NOCS test-set with a 16.4% absolute improvement in mAP for 6D pose estimation while running in near real-time at 5 Hz. Project page: fsd6d.github.io**

I. INTRODUCTION

The task of 3D perception from monocular RGB-D input has garnered significant attention within the domains of computer vision [7, 10, 12, 52] and robotics [5, 14, 18, 24]. This compelling problem holds immense relevance, with wide-ranging implications in critical areas such as autonomous navigation [15, 16] and robotic manipulation [4, 14, 18, 24]. At its core, this challenge revolves around the extraction of intricate 3D object representations from a single RGB-D perspective of the environment. This entails the precise estimation of a multitude of object attributes, encompassing 3D shape, 6D pose (comprising orientation and position), size, and visual appearance of individual object instances. Due to its inherent complexity, the problem is ill-posed, and predicting 3D information from 2D images can be ambiguous.

Various data-driven (supervised learning) approaches have been proposed for this challenging problem [9, 17, 23, 29, 40]. However, annotating 3D data can be quite expensive. Furthermore, when it comes to making robots capable of transitioning seamlessly from simulated environments to the real world, labeling real-world data may not be always feasible. In this context, the appeal of 2D supervision becomes evident as a cost-effective and widely accessible alternative. 2D supervision comes at minimal cost, and pseudo labels generated by recent methods like Segment Anything [22] and MiDaS [35] for in-the-wild data are of very high quality.

However, the absence of methods to generate robust 3D real labels necessitates the exploration of innovative training paradigms, particularly those grounded in self-supervised learning, to address this constraint. Recent developments have introduced self-supervised approaches that operate

Feature	NOCS [46]	ShAPO [17]	RePoNet [6]	CenterSnap [14]	SSC- 6D [34]	**FSD (Ours)**
Without real-world 3D labels	✗	✗	✓	✗	✓	✓
3D shape	✗	✓	✗	✓	✓	✓
One model for all categories	✗	✓	✗	✓	✗	✓
Simultaneous detection and 6D Pose	✗	✓	✗	✓	✗	✓
Without post-optimization	✓	✗	✓	✓	✓	✓

TABLE I: **Comparison with other 6D pose estimation and shape reconstruction methods:** ✓ indicates a method has the feature, and ✗ indicates that it doesn't.

without the dependency on 3D labels [6, 34]. While these proposed methods have exhibited promise, they may not be feasible for practical applications. For example, object detection and 3D prediction happen in separate stages [6, 34], resulting in sub-optimal inference times. Additionally, the linear growth of model size with the number of object categories poses scalability challenges. Consequently, the applicability of such approaches in real-world settings may be limited. Notably, [6] needs additional unlabelled real data to enable effective transfer to real-world scenarios.

To account for the discussed limitations, we propose a novel self-supervised end-to-end method to infer the 3D shape, size, and 6D poses of multiple objects. We employ a multi-stage training strategy to transfer synthetic performance to real-world domains. Firstly, the model is pre-trained on synthetic data (CAMERA [46]) using both 2D and 3D supervision (which comes for free). This is followed by joint training on synthetic and real data (Real 275 [46]). During this phase, only 2D labels from the real data are employed. In the final stage, the model undergoes fine-tuning exclusively on the real data, once again relying solely on 2D labels from the real dataset. To account for no direct 3D supervision from real data, we employ 3D self-supervision using chamfer loss. Specifically, we get a pseudo ground truth for point clouds by back-projecting the input depth maps into 3D space. These estimated point clouds serve as a loose supervision mechanism for guiding the predicted point clouds. As a result, our approach consistently yields superior quantitative and qualitative results when compared to robust baseline methods.

Our proposed method is most similar to [17] and [34] but with key differences. Unlike our method, [17] relies on 3D labeled real-world data and uses post-optimization after training (resulting in less efficient processing times). Conversely, in contrast to [34], our method is end-to-end (detection and 3D prediction happen in one forward pass) and utilizes a single universal model for all categories. Key distinctions from existing methods are concisely summarised in Table I. Our contributions are summarized as follows:

- A novel **multi-stage training pipeline** for **fast and efficient** shape reconstruction and 6D pose and size estimation **without requiring real-world 3D labels**.
- Our approach achieves state-of-the-art results and outperforms existing baselines for self-supervised 6D pose estimation, showing **over 16% absolute improvement** in mAP for 6D pose at $10°10\,\mathrm{cm}$ on the NOCS [46] test set.
- A **faster-batched shape extraction** and an **end-to-end feed-forward approach** making fine-tuning pipeline and model inference **orders of magnitude faster** then competing baselines.

II. RELATED WORK

Implicit shape representations: Scalar field approximators have emerged as a prominent direction to model 3D shapes. Notable works include Occ-Net [28], IM-Net [3] and DeepSDF [30]. These methods output either an SDF (Signed Distance Field) or an occupancy estimate for every 3D coordinate. NGLOD [41], ROAD [50], and MeshSDF [37] have been proposed for efficient octree representation and differentiable mesh representation, respectively. Our method utilizes implicit fields along with a fast octree-based sampling (Section III-C) to decode shapes which is used for self-supervised loss during training.

6D pose and size estimation: Work streams using pose regression [14, 19, 44, 47], template matching [20, 39, 42] and establishing correspondences [8, 13, 32, 38, 46] have been proposed. Most works, however, focus only on pose estimation and not the simultaneous 3D shape prediction. We deal with the task of end-to-end 6D pose and size estimation without relying on any test-time optimization or real-world 3D labels.

Self-supervised methods for 6D pose estimation: DSC-PoseNet [48] and Self6d [45] have introduced instance-level 6D pose estimation methods that heavily rely on the rendering of CAD models of the target objects. However, such approach may face scalability challenges when applied to real-world scenarios. In contrast, [31] proposed a pose estimation technique that utilizes multiple reference images rather than 3D scans, which may also be not practical when applied to real-world data at scale. In contrast, both [11] and [49] carried out sim2real adaption after training on synthetic data. For a more robust generalization, SSC-6D [34], RePoNet [6], and CPS++[27] train on both real and synthetic data in combination.

III. METHOD

We propose FSD, a novel learning-based object-centric scene understanding method. From an RGB-D observation, it estimates 6D pose, 3D shape, and size of all the seen object instances in an image without requiring real-world 3D labels to train its method. It performs object detection, localization, and 3D reconstruction. All three steps happen end-to-end without any post-processing. The network design is inspired by [17] and consists of two major components: a) A detection

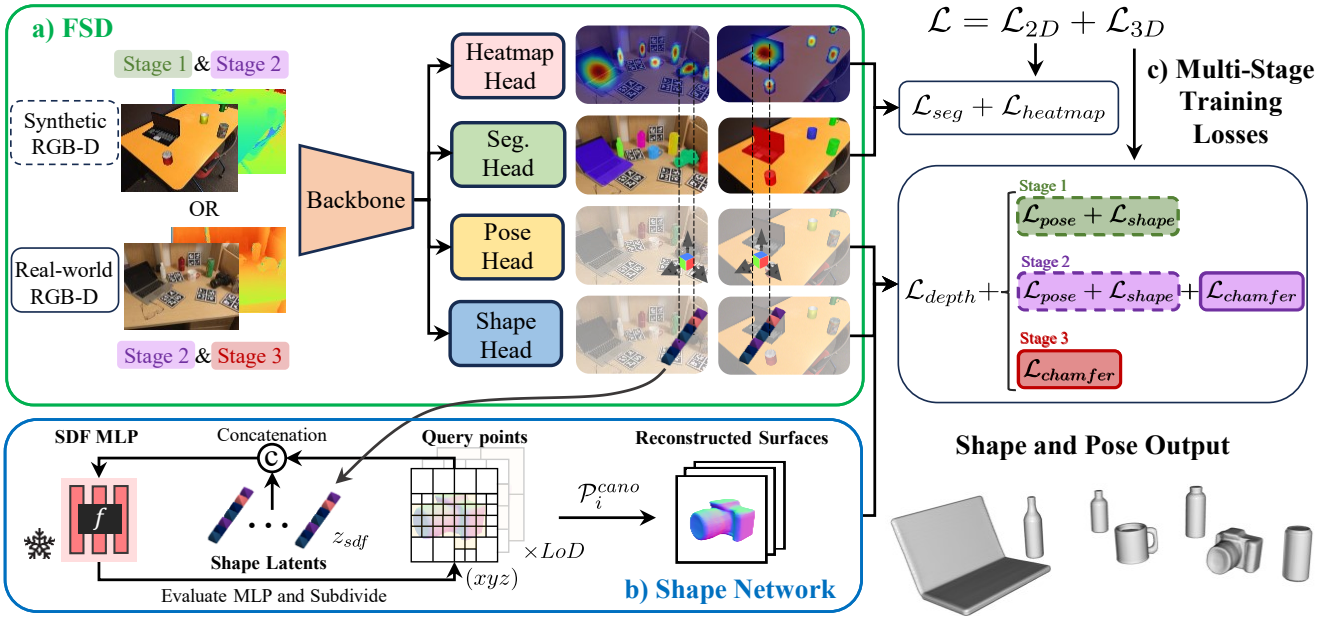

Fig. 2: **Method:** a) Forward pass of the proposed model across different training stages. From a single-view RGB-D observation, the model predicts a segmentation mask, depth map, object heatmap, pose map, and shape map. b) Batchified recursive point sampling is illustrated where the batch of concatenated 3D points and latent vectors are evaluated for SDF using a frozen shape auto-decoder. c) Losses for different training stages. Losses are color-coded based on the training stages and have solid out lines for real data and dotted lines for synthetic data.

module for detecting the 2D locations of the objects; b) A 3D prediction module for predicting the 3D shape, 6D pose, and size of the objects. For our synthetic 3D priors, we use DeepSDF [30] which is a learned continuous Signed Distance Function (SDF) representing shapes of different categories. We first train it offline on ShapeNet [1] to get a shape decoder. Then the decoder is frozen and is used to recover the implicit object shape using predicted latent codes and query points contained in a unit cube.

Concretely, given a single-view RGB-D observation containing an image $I \in \mathbb{R}^{h \times w \times 3}$ and depth map $D \in \mathbb{R}^{h \times w}$, our method, **FSD**, infers the 6D pose $\tilde{\mathcal{P}} \in SE(3)$, 1D scale $\tilde{\mathcal{T}} \in \mathbb{R}^1$, and 3D shape (as SDF) for each detected object. The framework is depicted in Fig. 2. First, an FPN-based backbone network extracts and fuses features from the RGB-D input (following [17]), which are processed by task-specific heads to get dense predictions of heatmap, shape, pose, and category score. Next, we describe our network's forward pass in Sections III-A and Section III-B, our fast batched surface extraction in III-C and lastly our novel multi-stage 3D training strategy in III-E along with self-supervised loss in III-D to aid the effective transfer of fully supervised synthetic domain learning to real-world domain without requiring any real 3D labels.

A. Segmentation and Heatmap head

The segmentation head and heatmap head predict the categories and 2D centers for objects in the scene. For n categories, the segmentation head predicts an $(n+1)$-channel logit map (an extra channel for the background class) $\hat{M} \in \mathbb{R}^{\frac{h}{R} \times \frac{w}{R} \times 1}$, which is supervised with ground truth instance masks. The object 2D centers are predicted as heatmaps $\hat{H} \in [0,1]^{\frac{h}{R} \times \frac{w}{R} \times 1}$ by the heatmap head, where each local maxima in the heatmap $\hat{H}$ becomes the detected point $(\hat{x}_i, \hat{y}_i)$. L2 loss $\mathcal{L}_{heatmap} = \sum_{xy} \left(\hat{H} - H\right)^2$ is applied to supervise the head where ground truth heatmaps H are constructed from 2D coordinates with Gaussian kernels. R is the downsampling factor of the backbone.

B. Pose, Shape and Depth head

In parallel, the pose and shape heads predict a pose map $z_{pose} \in \mathbb{R}^{\frac{h}{R} \times \frac{w}{R} \times 13}$ and a shape map $z_{shape} \in \mathbb{R}^{\frac{h}{R} \times \frac{w}{R} \times D}$. $D = 64$ is the dimension of the shape latent embedding. For each instance, shape and pose embeddings are queried using the predicted object centers from the heatmap $\hat{H}$ as follows:

$$z_{sdf_i} = z_{shape}[x_i, y_i, :]; \quad sRT_i = z_{pose}[x_i, y_i, :] \qquad (1)$$

where (x_i, y_i) is one of N predicted object centers in $\hat{H}$, and N is the number of instances detected by the model in the given input. These two heads are supervised during training stage 1 and 2 (see Section III-E) using synthetic data, where L1 losses of pose and shape embedding are calculated between the ground truth and estimated values, weighted by the estimated heatmap $\hat{H}$.

Latent code z_{sdf_i} for each instance is fed to a SDF (Signed Distance Function) based auto-decoder to get the predicted point cloud in the canonical space $\mathcal{P}_i^{cano}$ as done in [30].

Shape estimation using SDFs is computationally expensive. Thus, we employ an octree-based point sampling

method, similar to [17]. We implement a novel batched differentiable shape extraction and discuss it in Section III-C. The shape decoder is pre-trained and frozen as shown in the section b of Fig. 2. The point cloud in the camera frame is estimated as $\mathcal{P}_i^{cam} = (sRT_i)\,\mathcal{P}_i^{cano}$. Additionally, we supervise a depth head with an auxiliary depth loss. Since the real-world depth maps are noisy with artifacts, we introduce the same noise in synthetic data and try to recover the clean depth. This enhances the performance transfer from synthetic to real.

C. Differentiable Surface Extraction

When provided with query points q_i and their corresponding signed distance values s_i, we require a differentiable approach to access the implicit surface. Merely selecting query points based on distance values does not allow us to compute derivatives with respect to the latent vector z_{sdf_i}. However, by computing signed distance functions (SDFs) with respect to their locations, we can efficiently calculate the normal vector at each surface point during a reverse pass $n_i = \frac{\partial f(q_i;\mathbf{z})}{\partial q_i}$ similar to [17, 51].

Normals point toward the nearest surface, and signed distance values provide precise distance information, enabling us to map the query location to a 3D surface position denoted as p_i:

$$p_i = q_i - \frac{\partial f(q_i; z_{sdf_i})}{\partial q_i} f(q_i; z_{sdf_i}). \tag{2}$$

a) Batchified Recursive Point Sampling: To efficiently extract surface points for predicted objects associated with their respective latent vectors, we employ a recursive point extraction method inspired by [17]. We start by defining a coarse voxel grid and computing SDF values for each point using our trained SDF network. Voxels with SDF values exceeding the voxel grid size for the current resolution level are discarded, while the remaining voxels undergo subdivision, generating eight new voxels in each iteration. This process continues until reaching the desired resolution level, initiated at the level of detail (LoD) 1 and progressing to LoD 6. Once completed, we extract the point coordinates, along with their corresponding SDF values and normals, projecting them onto the object surface using the previously described iso-surface projection procedure.

To efficiently recover shapes for multiple objects concurrently, we implement a batchified version of the extraction algorithm. Since different objects may have distinct shapes, we traverse a single octree structure encompassing all predicted objects. This involves initializing a coarse grid for each predicted latent vector and collectively traversing them while monitoring boundaries that separate points belonging to different objects. Upon reaching the final LoD, we extract point clouds for each object based on established boundaries.

D. Chamfer loss

To provide a 3D learning signal for the real data, we compute the Chamfer loss between the estimated point cloud from scale, pose, and SDF and the ground-truth point cloud. For each instance, similar to [34], a point cloud of the visible points $\mathcal{P}_i^{real}$ is lifted from the input depth map D using the camera intrinsics K; correspondingly, the estimated point cloud is calculated from $\mathcal{P}_i^{cam} = (sRT_i)\,\mathcal{P}_i^{cano}$. Typical chamfer loss calculates distance between all pairs of points the two point clouds. However, the depth-lifted point clouds are noisy and prone to outliers. Therefore, to make it robust to noises, we use a thresholded version of chamfer loss and only calculate the loss if two points are less than ϵ units apart. Specifically,

$$\mathcal{L}_{chamfer} = \frac{1}{N_p} \sum_{p_j \in \mathcal{P}_i^{real}} \max\left(0, \epsilon - \min_{p_k \in \mathcal{P}_i^{cam}} \|p_j - p_k\|_2\right) + \frac{1}{N_p} \sum_{p_j \in \mathcal{P}_i^{cam}} \max\left(0, \epsilon - \min_{p_k \in \mathcal{P}_i^{real}} \|p_j - p_k\|_2\right)$$

where $\epsilon > 0$ and N_p is the total number of points satisfying $\|p_j - p_k\|_2 < \epsilon,\ \forall\ p_j \in \mathcal{P}_i^{real}$ and $\forall\ p_k \in \mathcal{P}_i^{cam}$. To stabilize the convergence, $\mathcal{P}_i^{cano}$ is detached from the gradient computation graph.

E. 3D learning strategy

In this section, we describe our approach to address the challenge of training a 3D reasoning model with the absence of real 3D labels. Our strategy consists of three distinct stages: pre-training on synthetic data followed by mixed training with a combination of synthetic and real data, and lastly fine-tuning on real data. We employ this sequence of training steps to effectively enable 3D reasoning and generalization on real-world data.

1) Stage 1: Pre-training on Synthetic Data: In the initial stage, we perform pre-training using synthetic data from the CAMERA dataset [46]. Synthetic data inherently provides 3D labels, allowing us to leverage this information to learn 3D priors. This pre-training stage serves as a foundational step in our approach. Formally, the objective looks like the following: $\mathcal{L}_{pretrain} = \mathcal{L}_{seg} + \mathcal{L}_{depth} + \mathcal{L}_{heatmap} + \mathcal{L}_{pose} + \mathcal{L}_{shape}$

2) Stage 2: Mixed Training: Directly transitioning from synthetic pre-training to fine-tuning on real data can lead to two undesirable effects: 1) the forgetting of 3D priors due to the absence of explicit 3D supervision in real data, and 2) overfitting on the real training data without meaningful 3D learning. To address these challenges, we introduce a mixed training phase following pre-training. During mixed training, each batch of inputs comprises a combination of real and synthetic data. While synthetic data points come with 3D labels, real data points do not. This design ensures that the model retains the 3D priors acquired during pre-training while adapting to the nuances of real-world data. We conceptualize this intermediate mixed-training step as a form of soft exposure to real-world data.

For our experiments, we keep the ratio of synthetic to real samples as 5. Specifically, for a sample b in a mixed batch B, total loss looks like this,

$$\mathcal{L}_{mixed} = \mathcal{L}_{seg} + \mathcal{L}_{depth} + \mathcal{L}_{heatmap} + \mathbb{1}(b \in syn)(\mathcal{L}_{pose} + \mathcal{L}_{shape}) + \mathbb{1}(b \in real)\mathcal{L}_{chamfer}$$

Method	Type	**IOU25** ↑	**IOU50** ↑	**5°5 cm** ↑	**5°10 cm** ↑	**10°5 cm** ↑	**10°10 cm** ↑
NOCS [46]		**84.8**	78.0	10.0	9.8	25.2	25.8
Metric Scale [25]		81.6	68.1	5.3	5.5	24.7	26.5
ShapePrior [43]	Supervised	81.2	77.3	21.4	21.4	54.1	54.1
CASS [2]		84.2	77.7	23.5	23.8	58.0	58.3
CenterSnap [14]		83.5	**80.2**	**27.2**	**29.2**	**58.8**	**64.4**
CPS++ [27]		-	17.7	-	-	≤ 22.3	-
SSC-6D [34]	Self-supervised	**83.2**	73.0	19.6	-	54.5	56.2
FSD (Ours)		80.9	**77.4**	**28.1**	**34.4**	**61.5**	**72.6**

TABLE II: **Results on REAL275:** Quantitative comparison of our proposed method with competing baselines. Supervised denotes utilizing both camera and real-world 3D supervision during training whereas self-supervised methods do not utilize any real-world 3D annotations. All methods train on both CAMERA25 and Real275 datasets.

where $\mathbb{1}(b \in syn)$ denotes if the sample b is synthetic and $\mathbb{1}(b \in real)$ denotes if the sample b is real. To better facilitate learning from two different data distributions, batch normalization with fixed mean and variance (learnt during pre-training) is used.

3) Stage 3: Fine-tuning on Real Data: In the final stage, we focus on maximizing the learning from relevant real data. Fine-tuning is carried out exclusively on the real data, allowing the model to refine its understanding of real-world 3D structures. The following loss objective is used: $\mathcal{L}_{finetune} = \mathcal{L}_{seg} + \mathcal{L}_{depth} + \mathcal{L}_{heatmap} + \mathcal{L}_{chamfer}$

All loss terms are weighted using loss weights λs that are omitted above for clarity.

IV. EXPERIMENTS

A. Datasets

We leverage both real and synthetic datasets. **CAMERA** [46] has 275K synthetic images with 3D annotations. The training dataset includes 1085 object models spanning across 6 categories: *bottle*, *bowl*, *camera*, *laptop*, and *mug*. The evaluation dataset has 184 object models spanning across the same set of categories. **REAL** [46] train set has 7 scenes spanned across 4300 images, with test-set having 6 scenes spanning across 2750 images.

B. Implementation details

Weights of $\mathcal{L}_{seg}$, $\mathcal{L}_{depth}$, $\mathcal{L}_{heatmap}$, $\mathcal{L}_{pose}$, $\mathcal{L}_{shape}$, and $\mathcal{L}_{chamfer}$ are 1, 1, 100, 0.1, 0.1 and 10 respectively, determined empirically. The distance threshold (ϵ) for $\mathcal{L}_{chamfer}$ is set at 0.2. Batch size of 32 is used and multi-gpu training on 7 NVIDIA A100s is performed. Adam [21] optimizer with 0.9 momentum is used with the learning rate of 6×10^{-4} decaying with polynomial decay (exponent=0.9) across epochs until convergence. Color jitter augmentation is used for both synthetic and real data. Flip augmentation is only used for real data because it is non-trivial to flip 3D labels for synthetic.

Predicting rotations in an unconstrained setting such as ours can be challenging. From a network perspective, they are arbitrary 9 real numbers ($SE3$ representation), and the constructed rotation metrics may not be orthogonal, which may introduce unwanted shear transformations. To control this effect, [26] introduced an SVD-based technique to ensure the orthogonalization of the predicted rotation metrics. We use this orthogonalization during our training. For training the SDF-based shape decoder, an MLP with 8 layers and a hidden size of 512 is used and is trained offline for 2000 epochs. PyTorch [33] and PyTorch3D [36] is used for implementation.

C. Metrics

Following [6, 14, 17, 34], the performance of 3D object detection and 6D pose estimation is evaluated independently. For 3D object detection, we report the average precision for IoU 25 and IOU 50 thresholds. For 6D pose estimation, average precision for which error is less than $x°$ for rotation and y cm for translation is reported where $(x, y) \in \{5, 10\}$.

D. Baselines

We compare against both fully-supervised and self-supervised baselines. For a fair comparison, we compare against methods that do not perform post-processing or post-optimization [17] or use any additional unlabelled data for a better synthetic to real world transfer [6].

1) Fully-supervised baselines:

(a) **NOCS**[46]: This architecture extends the Mask-RCNN framework to predict the NOCS map and utilizes a similarity transform with depth information for pose and size prediction.
(b) **Shape Prior**[43]: This method infers a 2D bounding box for each object and predicts shape deformations.
(c) **CASS**[2]: It employs a two-stage approach. First, it detects 2D bounding boxes, and then it regresses the pose and size of the objects.
(d) **Metric-Scale**[25]: An extension of NOCS, this method predicts the object center and metric shape separately.
(e) **CenterSnap**[14]: It is an end-to-end method that does object detection and 3D prediction in one forward pass.

2) Self-supervised baselines:

(a) **CPS++** [27]: This approach proposed coarse 3D alignment of 3D centroids before doing fine alignment using chamfer loss for self-supervision.
(b) **SSC-6D** [34]: It does 3D prediction on center cropped images having the individual detected objects. It uses a two-stage pipeline (warm up on the synthetic data followed by mixed training) and depth lifted point clouds for self-supervision.

Method	IOU25	IOU50	5°10 cm	10°10 cm
PT	27.3	26.1	6.0	34.7
PT + MT	70.9	67.3	13.8	51.5
PT + FT	**82.1**	**78.4**	16.9	57.2
PT + FT + FT	81.4	75.7	14.2	54.1
PT + MT + FT (Ours)	80.9	77.4	**34.4**	**72.6**

TABLE III: **Ablation:** Effect of training strategy (results on NOCS Real 275). PT, MT and FT stand for Pre-training, Mixed-training and Fine-tuning respectively.

Method	**Inference time**
SSC-6D [34]	1.99s
Ours	**0.20s**

TABLE IV: Quantitative inference-time comparison per image on A100 GPU

Method	**Shape time**
DeepSDF [30]	0.15s
Octree-SDF [17]	0.11s
Batched Octree (Ours)	**0.01s**

TABLE V: Quantitative shape extraction time with LoD 6 for rows 2, 3 and resolution 64 for [30]

Method	IOU50	10°10 cm
Zero-shot	26.1	34.7
2D supervision	4.2	4.1
Ours	**77.4**	**72.6**

TABLE VI: **Ablation:** Effect of loss objectives used during training stages (results on NOCS Real 275)

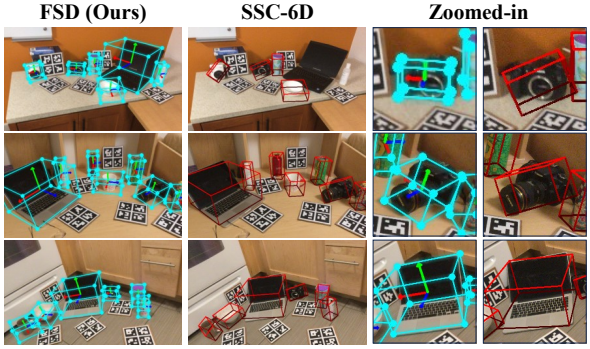

Fig. 3: **Qualitative comparison:** Our method vs. SSC-6D [34] on NOCS Real275 test-set.

E. Comparison with the baselines

1) 3D object detection and 6D pose estimation: Results are tabulated in the Table II. Against **fully-supervised baselines**, we observe superior performance (absolute gains as high as 8.2%) in the pose estimation performance across all the mAP metrics and very similar 3D detection performance (IoU). This is a significant result as all the supervised baselines used 3D real labels, unlike our method. This hints that our method was able to learn useful 3D inductive biases without being directly trained for it. Self-supervised methods are known to generalize better. Even though supervised methods have access to 3D labels, chamfer loss against noisy pointclouds helps our method in learning a more generalizable representation. Against **self-supervised baselines**, significant performance gains are also observed. There's an improvement in IoU50 by absolute 4.4% and consistent improvement in pose-estimation performance (absolute gains as high as 16.4%).

2) Inference speeds: We compare inference speed of our model in time per image with the baseline in the Table IV. Our method being single-shot and end-to-end makes it faster. It is also worth noting that SSC-6D [34] uses one model per category, thus allowing for more parameters for representation learning. Specifically, its space complexity is $\mathcal{O}(n_{categories})$ (where $n_{categories}$ is the number of categories in the dataset) whereas ours is $\mathcal{O}(1)$.

F. Qualitative comparison

In Fig. 3, we compare the output of our proposed model with SSC-6D [34]. Given the RGB-D input, the model learns to estimate accurate 6D poses (visualized bounding boxes with orientations). Our model achieves near-perfect pose (Fig. 3) and shape estimations (Fig. 1) without being trained on 3D real data and having not seen the test-time instances during training (category-level estimation).

G. Ablation

We ablate various design choices for our proposed method and answer the following questions.

1) Why not directly fine-tune the pre-trained model?: Directly fine-tuning the pre-trained model may seem attractive but in doing so, model may forget the 3D priors it learned during the pre-training step. Mixed training allows the model to smoothly transition from 3D labeled data points to 3D unlabeled data points while learning about the real data. This has been demonstrated in Table III where training for three stages and mixed training is superior than other combinations.

2) How important is chamfer loss?: A case may be made to fine-tune only using the 2D losses on the real data. Doing so may not help the model in learning 3D attributes of real data. Only using 2D losses (i.e. no chamfer loss) for real data in our training strategy makes it forget the 3D learnings. Zero-shot performance is directly evaluating the pre-trained model. Results are summarized in the Table VI.

3) How fast is batched shape extraction?: We introduce a novel batched shape extraction by batch querying Signed Distance Functions using Octree based sampling. This is summarised in the Table V. Faster shape extraction directly leads to faster training speeds and thus making it suitable for training on large-scale in-the-wild datasets.

V. CONCLUSION

In this paper, we introduce FSD, a novel method for estimating the 6D pose, size, and shape of objects in a scene. Our method 1) is end-to-end (object detection, localization, and 3D predictions happen in one forward pass), 2) uses one universal model for all the categories and does not need a separate model for different categories, 3) does not use 3D real labels 4) is faster to train compared to baselines. We propose a novel multi-stage training strategy to maximize learning from unlabelled real data. To our knowledge, ours is the first work to attempt all these goals together. Our method also achieved a significant increase in the pose estimation while outperforming strong self-supervised pose estimation baselines. We hope the utility and performance of our proposed method will motivate similar works in the future.

References

[1] A. X. Chang *et al.*, "ShapeNet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[2] D. Chen, J. Li, Z. Wang, and K. Xu, "Learning canonical shape space for category-level 6d object pose and size estimation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 973–11 982.

[3] Z. Chen and H. Zhang, "Learning implicit fields for generative shape modeling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5939–5948.

[4] C. G. Cifuentes, J. Issac, M. Wüthrich, S. Schaal, and J. Bohg, "Probabilistic articulated real-time tracking for robot manipulation," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 577–584, 2016.

[5] C. Ferrari and J. F. Canny, "Planning Optimal Grasps.," in *ICRA*, vol. 3, 1992, p. 6.

[6] Y. Fu and X. Wang, "Category-level 6d object pose estimation in the wild: A semi-supervised learning approach and a new dataset," *ArXiv*, vol. abs/2206.15436, 2022.

[7] G. Gkioxari, J. Malik, and J. Johnson, "Mesh R-CNN," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 9785–9795.

[8] W. Goodwin, S. Vaze, I. Havoutis, and I. Posner, "Zero-shot category-level object pose estimation," *arXiv preprint*, 2022.

[9] T. Groueix, M. Fisher, V. G. Kim, B. Russell, and M. Aubry, "AtlasNet: A Papier-Mâché Approach to Learning 3D Surface Generation," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[10] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *ICCV*, 2017.

[11] Y. He, H. Fan, H. Huang, Q. Chen, and J. Sun, "Towards self-supervised category-level object pose and size estimation," *ArXiv*, vol. abs/2203.02884, 2022.

[12] N. Heppert *et al.*, "Carto: Category and joint agnostic reconstruction of articulated objects," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2023, pp. 21 201–21 210.

[13] T. Hodan, D. Barath, and J. Matas, "Epos: Estimating 6d pose of objects with symmetries," in *CVPR*, 2020.

[14] M. Z. Irshad, T. Kollar, M. Laskey, K. Stone, and Z. Kira, "Centersnap: Single-shot multi-object 3d shape reconstruction and categorical 6d pose and size estimation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[15] M. Z. Irshad, C.-Y. Ma, and Z. Kira, "Hierarchical cross-modal agent for robotics vision-and-language navigation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 13 238–13 246.

[16] M. Z. Irshad, N. C. Mithun, Z. Seymour, H.-P. Chiu, S. Samarasekera, and R. Kumar, "Sasra: Semantically-aware spatio-temporal reasoning agent for vision-and-language navigation in continuous environments," 2022.

[17] M. Z. Irshad, S. Zakharov, R. Ambrus, T. Kollar, Z. Kira, and A. Gaidon, "Shapo: Implicit representations for multi-object shape, appearance, and pose optimization," in *European Conference on Computer Vision*, 2022.

[18] Z. Jiang, Y. Zhu, M. Svetlik, K. Fang, and Y. Zhu, "Synergies between affordance and geometry: 6-Dof grasp detection via implicit representations," *Robotics: science and systems*, 2021.

[19] 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 *ICCV*, 2017.

[20] 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*, Springer, 2016, pp. 205–220.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.

[22] A. Kirillov *et al.*, "Segment anything," *arXiv:2304.02643*, 2023.

[23] W. Kuo, A. Angelova, T.-Y. Lin, and A. Dai, "Mask2CAD: 3D shape prediction by learning to segment and retrieve," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, Springer, 2020, pp. 260–277.

[24] M. Laskey, B. Thananjeyan, K. Stone, T. Kollar, and M. Tjersland, "SimNet: Enabling robust unknown object manipulation from pure synthetic data via stereo," in *5th Annual Conference on Robot Learning*, 2021.

[25] T. Lee, B.-U. Lee, M. Kim, and I. S. Kweon, "Category-level metric scale object shape and pose estimation," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8575–8582, 2021.

[26] J. Levinson *et al.*, "An analysis of svd for deep rotation estimation," *ArXiv*, vol. abs/2006.14616, 2020.

[27] F. Manhardt *et al.*, "Cps++: Improving class-level 6d pose and shape estimation from monocular images with self-supervised learning," *arXiv: Computer Vision and Pattern Recognition*, 2020.

[28] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *CVPR*, 2019.

[29] M. Niemeyer, L. Mescheder, M. Oechsle, and A. Geiger, "Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision," in *CVPR*, 2020.

[30] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed

distance functions for shape representation," in *CVPR*, 2019.

[31] K. Park, A. Mousavian, Y. Xiang, and D. Fox, "Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10 707–10 716, 2019.

[32] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," in *ICCV*, 2019.

[33] A. Paszke *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., 2019, pp. 8024–8035.

[34] W. Peng, J. Yan, H. Wen, and Y. Sun, "Self-supervised category-level 6d object pose estimation with deep implicit shape representation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, 2022, pp. 2082–2090.

[35] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 3, 2022.

[36] N. Ravi *et al.*, "Accelerating 3d deep learning with pytorch3d," *arXiv:2007.08501*, 2020.

[37] E. Remelli *et al.*, "Meshsdf: Differentiable iso-surface extraction," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 22 468–22 478.

[38] I. Shugurov, S. Zakharov, and S. Ilic, "Dpodv2: Dense correspondence-based 6 dof pose estimation," *TPAMI*, 2021.

[39] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3d orientation learning for 6d object detection from rgb images," in *ECCV*, 2018.

[40] M. Sundermeyer, Z.-C. Marton, M. Durner, and R. Triebel, "Augmented autoencoders: Implicit 3D orientation learning for 6d object detection," *International Journal of Computer Vision*, vol. 128, no. 3, pp. 714–729, 2020.

[41] T. Takikawa *et al.*, "Neural geometric level of detail: Real-time rendering with implicit 3d shapes," in *CVPR*, 2021.

[42] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim, "Latent-class hough forests for 3d object detection and pose estimation," in *European Conference on Computer Vision*, Springer, 2014, pp. 462–477.

[43] M. Tian, M. H. Ang, and G. H. Lee, "Shape prior deformation for categorical 6d object pose and size estimation," in *European Conference on Computer Vision*, Springer, 2020, pp. 530–546.

[44] C. Wang *et al.*, "Densefusion: 6d object pose estimation by iterative dense fusion," in *CVPR*, 2019.

[45] G. Wang, F. Manhardt, J. Shao, X. Ji, N. Navab, and F. Tombari, "Self6d: Self-supervised monocular 6d object pose estimation," *ArXiv*, vol. abs/2004.06468, 2020.

[46] 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 *CVPR*, 2019.

[47] 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.

[48] Z. Yang, X. Yu, and Y. Yang, "Dsc-posenet: Learning 6dof object pose estimation via dual-scale consistency," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3906–3915, 2021.

[49] Y. You, R. Shi, W. Wang, and C. Lu, "Cppf: Towards robust category-level 9d pose estimation in the wild," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6856–6865, 2022.

[50] S. Zakharov, R. Ambrus, K. Liu, and A. Gaidon, "Road: Learning an implicit recursive octree auto-decoder to efficiently encode 3d shapes," in *Conference on Robot Learning (CoRL)*, 2022.

[51] S. Zakharov, W. Kehl, A. Bhargava, and A. Gaidon, "Autolabeling 3d objects with differentiable rendering of sdf shape priors," in *CVPR*, 2020.

[52] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.