FroDO: From Detections to 3D Objects

Martin Rünz[1,2,*], Kejie Li[1,3,*], Meng Tang[1], Lingni Ma[1], Chen Kong[1], Tanner Schmidt[1], Ian Reid[3], Lourdes Agapito[2], Julian Straub[1], Steven Lovegrove[1], and Richard Newcombe[1]

[1]Facebook Reality Labs
[2]Department of Computer Science, University College London
[3]School of Computer Science, The University of Adelaide

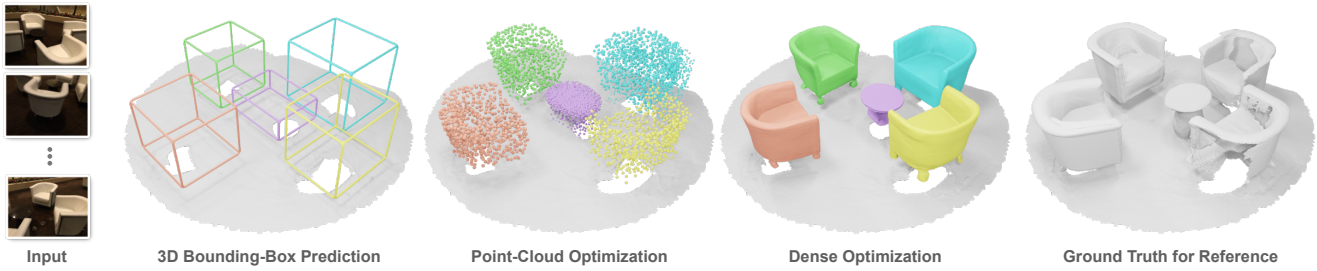

Figure 1: Given a localized input RGB sequence, FroDO dectects objects and infers their pose and a progressively fine grained and expressive object shape representation. Results on a real-world sequence from ScanNet [7].

Abstract

Object-oriented maps are important for scene understanding since they jointly capture geometry and semantics, allow individual instantiation and meaningful reasoning about objects. We introduce FroDO, a method for accurate 3D reconstruction of object instances from RGB video that infers object location, pose and shape in a coarse-to-fine manner. Key to FroDO is to embed object shapes in a novel learnt space that allows seamless switching between sparse point cloud and dense DeepSDF decoding. Given an input sequence of localized RGB frames, FroDO first aggregates 2D detections to instantiate a category-aware 3D bounding box per object. A shape code is regressed using an encoder network before optimizing shape and pose further under the learnt shape priors using sparse and dense shape representations. The optimization uses multi-view geometric, photometric and silhouette losses. We evaluate on real-world datasets, including Pix3D, Redwood-OS, and ScanNet, for single-view, multi-view, and multi-object reconstruction.

1. Introduction

Localizing and reconstructing 3D objects from RGB video is a fundamental problem in computer vision. Traditional geometry-based multi-view reconstruction [40, 41] can deal with large scenes given rich textures and large baselines but it is prone to failure in texture-less regions or when the photo-consistency assumption does not hold. Besides, these methods only provide geometry information but no semantics. An even more challenging question is how to fill in unobserved regions of the scene. Recently, learning based 3D reconstruction methods [2, 6, 9, 12, 28, 50] have emerged and achieved promising results. However, data-driven approaches rely heavily on synthetic renderings and do not generalize well to natural images. On the other hand, we have seen impressive progress in 2D recognition tasks such as detection and segmentation [15, 21, 26].

In this paper, we propose a system for object-centric reconstruction that leverages the best properties of 2D recognition, learning-based object reconstruction and multi-view optimization with deep shape priors. As illustrated in Fig. 2 FroDO takes a sequence of localized RGB images as input, and progressively outputs 2D and 3D bound-

* The first two authors contributed equally.

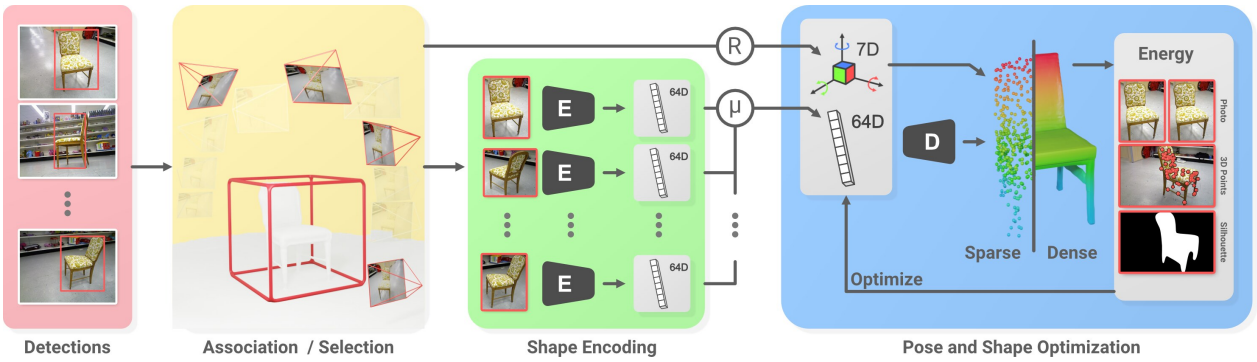

Figure 2: Given a sequence of calibrated, and localized RGB images, FroDO detects objects and infers their shape code and per-frame poses in a coarse-to-fine manner. We demonstrate FroDO on challenging sequences from real-world datasets that contain a single object (Redwood-OS) or multiple objects (ScanNet).

ing boxes, 7-DoF pose, a sparse point cloud and a dense mesh for 3D objects in a coarse-to-fine manner. FroDO demonstrates deep prior-based 3D reconstruction of real world multi-class and multi-object scenes from real-world RGB video. Related approaches are limited to single view [9,13,14,28,50], or multi-view but single objects [24], are purely geometry-based [40,41], or require depth and object scans [17,18].

Choosing the best shape representation remains a key open problem in 3D reconstruction. Signed Distance Functions (SDF) have emerged as a powerful representation for learning-based reconstruction [28, 32] but are not as compact or efficient as point clouds. One of our key contributions is a new joint embedding where shape codes can be decoded to both a sparse point cloud and a dense SDF. Our joint shape embedding enables seamless switching between both representations and can be used as a shape prior for shape optimization, enabling faster inference. As Fig. 2 illustrates, FroDO takes a calibrated, and localized image sequence as input and proceeds in four distinct steps: *2D detection*, *data association*, *single-view shape code inference* and *multi-view shape code optimization*. First, per-frame 2D bounding box detections are inferred using an off-the-shelf method [15]. Secondly, bounding boxes are associated over multiple frames and lifted into 3D. Next, a 64D code is predicted for each detection of the same object instance, using a novel encoder network. Per-image shape codes of the same instance are fused into a single code. Finally, shape code and pose are further refined by minimizing terms based on geometric, photometric and silhouette cues using our joint embedding as a shape prior. The final outputs of our system are dense object meshes placed in the correct position and orientation in the scene.

The contributions of our paper are as follows: *(i)* FroDO takes as input RGB sequences of real world multi-object scenes and infers an object-based map, leveraging 2D recognition, learning-based 3D reconstruction and multi-view optimization with shape priors. *(ii)* We introduce a novel deep joint shape embedding that allows simultaneous decoding to sparse point cloud and continuous SDF representations, and enables faster shape optimization. *(iii)* We introduce a new coarse-to-fine multi-view optimization approach that combines photometric and silhouette consistency costs with our deep shape prior. *(iv)* FroDO outperforms state of the art 3D reconstruction methods on real-world datasets — Pix3D [44] for single-object single-view and Redwood-OS [4] for single-object multi-view. We demonstrate multi-class and multi-object reconstruction on challenging sequences from the ScanNet dataset [7].

2. Related Work

At its core our proposed system infers dense object shape reconstructions from RGB frames, so it relates to multiple areas in 3D scene reconstruction and understanding.

Single-view learning-based shape prediction In recent years, 3D object shape and pose estimation from images has moved from being purely geometric towards learning techniques, which typically depend on synthetic rendering of ShapeNet [3] or realistic 2d-3d datasets like Pix3d [44]. These approaches can be categorized based on the shape representation utilized, for example occupancy grids [6,50], point clouds [9], meshes [47], or implicit functions [28]. Gkioxari *et al.* [12] jointly train detection and reconstruction by augmenting Mask RCNN with an extra head that outputs volume and mesh.

Our coarse-to-fine reconstruction pipeline includes a single-image encoder decoder network that predicts a latent shape code, point cloud, and SDF for each detected instance. Our single-view reconstruction network leverages a novel joint embedding that simultaneously outputs point cloud and SDF (Fig. 3). Our quantitative evaluation shows that our approach provides better single view reconstruction than competing methods.

Multi-view category-specific shape estimation Structure-from-Motion (SfM) and simultaneous localization and mapping (SLAM) are useful to reconstruct 3D structure from image collections or videos. However, traditional methods are prone to failure when there is a large gap between viewpoints, generally have difficulty with filling featureless areas, and cannot reconstruct occluded surfaces. Deep learning approaches like 3D-R2N2 [6], LSM [19], and Pix2Vox [51] have been proposed for 3D object shape reconstruction. These can infer object shape from either single or multiple observations using RNN or voxel based fusion. However, these fusion techniques are slow and data association is assumed.

3D reconstruction with shape priors These methods are the most closely related to our approach since they also use RGB video as input and optimize object shape and pose using 3D or image-based reprojection losses such as photometric and/or silhouette terms while assuming, often category-specific, learnt compact latent shape spaces. Some examples of the low dimensional latent spaces used are PCA [23, 48], GPLVM [8, 35, 37] or a learnt neural network [24]. In similar spirit we optimize a shape code for each object, using both 2D and 3D alignment losses, but we propose a new shape embedding that jointly encodes point cloud and DeepSDF representations and show that our coarse-to-fine optimization leads to more accurate results. These optimizable codes have also been used to infer the overall shape of entire scenes [1, 42] without lifting the representation to the level of objects. Concurrent work [25] proposes to optimize DeepSDF embeddings via sphere tracing, closely related to FroDO's dense optimization stage. We chose to formulate the energy via a proxy mesh, which scales better when many views are used.

Object-aware SLAM Although our system is not sequential or real-time, it shares common ground with recent object-oriented SLAM methods. Visual SLAM has recently evolved from purely geometric mapping (point, surface or volumetric based) to object-level representations which encode the scene as a collection of reconstructed object instances. SLAM++ [39] demonstrated one of the first RGB-D object-based mapping systems where a set of previously known object instances were detected and mapped using an object pose graph. Other instance-based object-aware SLAM systems have either aligned objects from a pre-trained database to volumetric maps [45] or models learnt during an exploration step to a surfel representation [43]. In contrast, others have focused on online object discovery and modeling [5] to deal with unknown object instances, dropping the need for known models and pre-trained detectors. Recent RGB-D object-aware SLAM methods leverage the power of state of the art 2D instance semantic segmentation masks [15] to obtain object-level scene graphs and per-object reconstructions [27] even in the case of dynamic scenes [38, 52]. Object oriented SLAM has also been extended to the case of monocular RGB-only [11, 16, 31, 33, 34] or visual inertial inputs [10]. Pillai and Leonard [34] aggregate multiview detections to perform SLAM-aware object recognition and semi-dense reconstruction, while [31] fit per-object 3D quadric surfaces. CubeSLAM [53] proposes a multi-step object reconstruction pipeline where initial cuboid proposals, generated from single view detections, are further refined through multi-view bundle-adjustment.

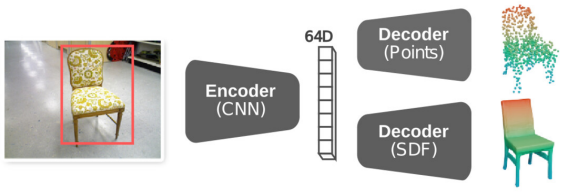

Figure 3: Our new joint shape embedding leverages the advantages of sparse point-based (efficiency) and dense surface (expressiveness) object shape representations.

3. Method Overview

FroDO infers an object-based map of a scene, in a coarse-to-fine manner, given a localized set of RGB images. We assume camera poses and a sparse point cloud have been estimated using standard SLAM or SfM methods such as ORB-SLAM [29] or COLMAP [40, 41]. We represent the object-based map as a set of object poses $\{T_{wo}^k\}$ with associated 3D bounding boxes $\{bb_3^k\}$ and shape codes $\{z^k\}$. $\mathbf{T}_{ba}$ denotes a transformation from coordinate system a to b. Our new joint shape embedding is described in Sec. 4.

The steps in our pipeline are illustrated in Fig. 2: First (Sec. 5.1) objects are detected in input images using any off-the-shelf detector [15], correspondences are established between detections of the same object instance in different images and 2D bounding boxes are lifted into 3D, which enables occlusion reasoning for view selection. Second, a 64D shape code is predicted for each visible cropped detection of the same object, using a novel convolutional neural network (Sec. 5.2). Codes are later fused into a unique object shape code (Sec. 5.2). Finally, object poses and shape code are incrementally refined by minimizing energy terms based on geometric and multiview photometric consistency cues using our joint shape embedding as a prior (Sec. 5.3).

4. Joint Shape Code Embedding

We propose a new joint latent shape-code space to represent and instantiate complete object shapes in a compact way. This novel embedding is also used as a shape prior to efficiently optimize object shapes from multi-view ob-

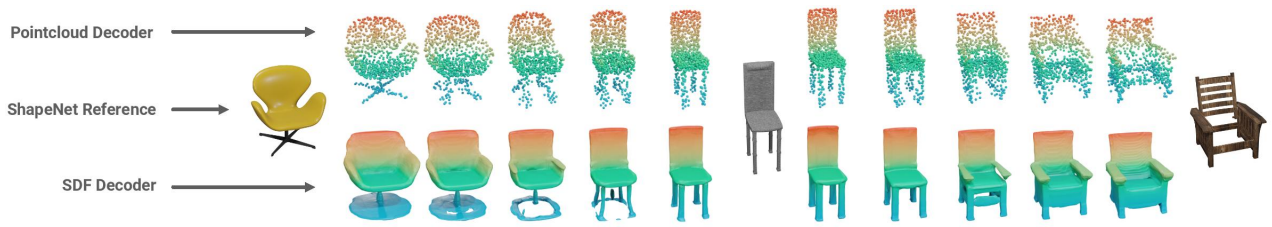

Figure 4: Joint latent shape space interpolation between 3 ShapeNet instances with ground-truth codes. Pointcloud and SDF decodings of intermediate codes are coherent.

servations. We parametrize object shapes with a latent code $\mathbf{z} \in \mathbb{R}^{64}$, which can be jointly decoded by two generative models $\mathbf{X} = G_s(\mathbf{z})$ and $\phi = G_d(\mathbf{z})$ into an explicit sparse 3D pointcloud $\mathbf{X}$ and an implicit signed distance function ϕ. While the pointcloud decoder generates 2048 samples on the object surface, the SDF decoder represents the surface densely via its zero-level set. The decoders are trained simultaneously using a supervised reconstruction loss against ground-truth shapes on both representations:

$$L = \lambda_1 D_{\mathrm{C}}(G_s(\mathbf{z}), \mathbf{X}_{gt}) + \lambda_2 L_\phi + \frac{1}{\sigma^2}\|\mathbf{z}\|^2, \quad (1)$$

$$L_\phi = |clamp(G_d(\mathbf{z}), \delta) - clamp(d_{gt}, \delta)|, \quad (2)$$

$$D_{\mathrm{C}}(\mathbf{A}, \mathbf{B}) = \frac{1}{|\mathbf{A}|}\sum_{\mathbf{x}\in\mathbf{A}} \min_{\mathbf{y}\in\mathbf{B}} \|\mathbf{x}-\mathbf{y}\|_2^2 + \frac{1}{|\mathbf{B}|}\sum_{\mathbf{y}\in\mathbf{B}} \min_{\mathbf{x}\in\mathbf{A}} \|\mathbf{x}-\mathbf{y}\|_2^2 \quad (3)$$

where D_{C} evaluates a symmetric Chamfer distance, and L_ϕ is a clipped L_1 loss between predicted $G_d(\mathbf{z})$ and ground-truth d_{gt} signed distance values with a threshold $\delta = 0.1$. We use 3D models from the CAD model repository ShapeNet [3] as ground truth shapes. While the original DeepSDF architecture [32] is employed for the SDF decoder, a variant of PSGN [9] is used as the pointcloud decoder. Its architecture is described in detail in the supplementary material. Joint embeddings decoded to both representations are illustrated in Fig. 4. The trained decoders allow us to leverage learnt object shape distributions, and act as effective priors for optimization based 3D reconstruction. In contrast to related prior-based shape optimization approaches [24, 25] where the shape embedding is specialized to a specific representation, our embedding offers the advantages of both sparse and dense representations at different stages of the optimization. Although DeepSDF can represent smooth and dense object surfaces, it is slow to evaluate as each point needs a full forward pass through the decoder. In contrast, the pointcloud representation is two orders of magnitude faster but fails to capture shape details. Our strategy is therefore to infer an initial shape using the point-based decoder before switching to the DeepSDF decoder for further refinement (Sec. 5.3). While inspired by [30] to use multiple shape representations, our embedding offers two advantages. First, the same latent code is used by both decoders, which avoids the need for a latent code consistency loss [30]. Secondly, training a shape encoder for each representation is not required.

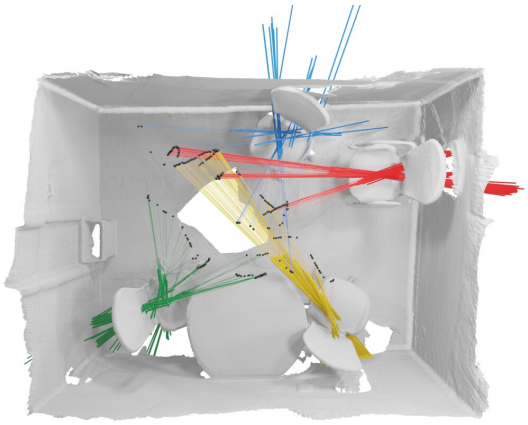

Figure 5: Data association: 3D line-segment clustering to predict b-box correspondences. Colors denote instance IDs.

5. From Detections to 3D Objects

5.1. Object Detection and Data Association

We use a standard instance segmentation network [15] to detect object bounding boxes bb_i^2 and object masks $\boldsymbol{M}$ in the input RGB video. To enable multi-view fusion and data aggregation for object shape inference, we predict correspondences between multiple detections of the same 3D object instance. Since the 3D ray through the center of a 2D bounding box points in the direction of the object center, the set of rays from all corresponding detections should approximately intersect. Knowledge of the object class sets reasonable bounds on the object scale to further restrict the expected object center location in 3D to a line segment as indicated by the thicker line segments in Fig. 5.

Object instance data association can then be cast as a

clustering problem in which the goal is to identify an unknown number of line segment sets that approximately intersect in a single point. We adopt an efficient iterative non-parametric clustering approach similar to DP-means [22] where the observations are line segments and the cluster centers are 3D points. Further details of the clustering algorithm are given in the supplementary material.

After clustering, each object instance k is associated with a set of 2D image detections I_k and a 3D bounding box bb_k^3, computed from the associated bounding box detections as described in [31]. By comparing the projection of the 3D object bounding box and the 2D detection box, we reject detections that have low IoU, an indication of occlusions or truncations. The filtered set of image detections I'_k is used in all following steps. Examples of the filtered detections are shown in supplementary material.

5.2. Single-view Shape Code Inference and Fusion

As illustrated in the *shape encoding* section of Fig. 2, a 64D object shape code is predicted for each filtered detection. We train a new encoder network that takes as input a single image crop and regresses its associated shape code $\mathbf{z_i} \in \mathbb{R}^{64}$ in the joint latent space described in Sec. 4.

The network is trained in a fully supervised way. However, due to the lack of 3D shape annotations for real world image datasets, we train the image encoder using synthetic ShapeNet [3] renderings. Specifically, we generate training data by rendering ShapeNet CAD models with random viewpoints, materials, environment mapping, and background. We also perturb bounding boxes of rendered objects and feed perturbed crops to the encoder during training. We chose a standard ResNet architecture, modifying its output to the size of the embedding vector. During training, we minimize the Huber loss between predicted and target embeddings, which we know for all CAD models. For the experiment on ScanNet in Sec. 6.3, we fine-tune the encoder network with supervised data from Pix3D [44].

Multi-view Shape Code Fusion For each object instance k we fuse all single-view shape codes $\{\mathbf{z_i} | i \in I'_k\}$ into a unique code z_k^0. We propose two fusion approaches and evaluate them in Table 4: *(i)* Average – we average shape codes to form a mean code z_k^{mean}; *(ii)* Majority voting – We find the 4 nearest neighbors of each predicted code $\mathbf{z_i}$ among the models in the training set. The most frequent of these is chosen as z_k^{vote}. Unlike the average code, z_k^{vote} guarantees valid shapes from the object database.

5.3. Multi-view Optimization with Shape Priors

For each object instance k, all images with non-occluded detections are used as input to an energy optimization approach to estimate object pose T_{wo}^k and shape code z_k in two steps. First, we optimize the energy over a sparse set of surface points, using the point decoder $G_s(\mathbf{z})$ as a shape prior. This step is fast and efficient due to the sparse nature of the representation as well as the light weight of the point-cloud decoder. Second, we further refine pose and shape minimizing the same energy over dense surface points, now using the DeepSDF decoder $G_d(\mathbf{z})$ as the prior. This slower process is more accurate since the loss is evaluated over all surface points, and not sparse samples.

Energy. Our energy is a combination of losses on the 2D silhouette E_s, photometric consistency E_p and geometry E_g with a shape code regularizer E_r:

$$E = \lambda_s \cdot E_{\text{s}} + \lambda_p \cdot E_{\text{p}} + \lambda_g \cdot E_{\text{g}} + \lambda_r \cdot E_{\text{r}}\,, \qquad (4)$$

where $\lambda_{s,p,g,r}$ weigh the contributions of individual terms. The regularization term $E_{\text{r}} = \frac{1}{\sigma^2}\|\mathbf{z}\|_2^2$ encourages shape codes to take values in valid regions of the embedding, analogously to the regularizer in Eq. 1. Note that the same energy terms are used for sparse and dense optimization – the main differences being the number of points over which the loss is evaluated, and the decoder $\mathcal{G}(\mathbf{z})$ used as shape prior.

Initialization. The 64D shape code is initialized to the fused shape code (Sec. 5.2), while the pose $\mathbf{T}_{wo}$ is initialized from the 3D bounding box bb_k^3 (Sec. 5.1): translation is set to the vector joining the origin of the world coordinate frame with the 3D bounding box centroid, scale to the 3D bounding box height and rotation is initialised using exhaustive search for the best rotation about the gravity direction – under the assumption that objects are supported by a ground-plane perpendicular to gravity.

Sparse Optimization. Throughout the sparse optimization, the energy E is defined over the sparse set of 2048 surface points $\mathbf{X}$, decoded with the point-based decoder $G_s(\mathbf{z})$. The energy E is minimized using the Adam optimizer [20] with autodiff. We now define the energy terms.

• The photometric loss E_{p} encourages the colour of 3D points to be consistent across views. In the sparse case, we evaluate E_{p} by projecting points in $\mathbf{X}$ to N nearby frames via known camera poses $\mathbf{T}_{cw}$ and comparing colors in reference $\mathcal{I}^R$ and source $\mathcal{I}_i^S$ images under a Huber norm $\|.\|_h$:

$$E_{\text{p}}(\mathbf{X}, \mathcal{I}^R, \mathcal{I}_1^S, ..., \mathcal{I}_N^S) = \frac{1}{N \cdot |\mathbf{X}|} \sum_{i=1}^{N} \sum_{\mathbf{x} \in \mathbf{X}} \|r(\mathcal{I}^R, \mathcal{I}_i^S)\|_h$$
$$r(\mathcal{I}^R, \mathcal{I}^S) = \mathcal{I}^R(\pi(\mathbf{T}_{cw}^R \mathbf{x})) - \mathcal{I}^S(\pi(\mathbf{T}_{cw}^S \mathbf{x})) \qquad (5)$$

where $\pi(\mathbf{x})$ projects 3D point $\mathbf{x}$ into the image.

• The silhouette loss E_{s} penalizes discrepancies between the 2D silhouette obtained via projection of the current 3D object shape estimate and the mask predicted with MaskR-CNN [15]. In practice, we penalize points that project outside the predicted mask using the 2D Chamfer distance:

$$E_s(\mathbf{z}_k, \mathbf{T}_{wo}^k) = D_{\text{C}}(\mathbf{M}, \pi(\mathbf{T}_{cw} \mathbf{T}_{wo}^k \mathcal{G}(\mathbf{z}))) \qquad (6)$$

where $\mathbf{M}$ is the set of 2D samples on the predicted mask and D_C is the symmetric Chamfer distance defined in Eq. 3.

• The geometric loss E_{g} minimizes the 3D Chamfer distance between 3D SLAM (or SfM) points and points on the current object shape estimate:

$$E_{\mathrm{g}}(\mathbf{z_k}, \mathbf{T}_{wo}^{k}) = D_{\mathrm{C}}(\mathbf{X}_{slam}, \mathbf{T}_{wo}^{k}\mathcal{G}(\mathbf{z})), \tag{7}$$

Dense Optimization. The shape code and pose estimated with the sparse optimization can be further refined with a dense optimization over all surface points and using the DeepSDF decoder $G_d(\mathbf{z})$. Since $G_d(\mathbf{z})$ uses an implicit representation of the object surface, we compute a proxy mesh at each iteration, and formulate the energy over its vertices. This strategy proved faster than sphere tracing [25], while achieving on-par accuracy, see Table 3. Relevant Jacobians are derived analytically and are given in the supplementary material together with further implementation details. We now describe the dense energy terms.

• The photometric and geometric losses $E_{\mathrm{p}}, E_{\mathrm{g}}$ are equivalent to those used in the sparse optimization (see Eq. 5, 7). However, they are evaluated densely and the photometric optimization makes use of a Lucas-Kanade style warp.

• The silhouette loss E_{s} takes a different form to the sparse case. We follow traditional level set approaches, comparing the projections of object estimates with observed foreground and background probabilities $P_{f,b}$:

$$E_s = \int_{\Omega} H(\phi) P_f(x) + (1 - H(\phi))\, P_b(x) d\Omega, \tag{8}$$

where ϕ is a 3D or 2D shape-kernel, and H a mapping to a 2D foreground probability field, resembling an object mask of the current state. Empirically, we found that 3D shape-kernels [36] provide higher quality reconstructions when compared with a 2D formulation [37] because more regions contribute to gradients. While H is a Heaviside function in the presence of 2D level-sets, we interpret signed distance samples of the DeepSDF volume as logits and compute a per-pixel foreground probability by accumulating samples along rays, similar to Prisacariu *et al.* [37]:

$$H = 1 - \exp \prod_{\mathbf{x} \text{ on ray}} (1 - \mathrm{sig}(\zeta \cdot \phi(\mathbf{x}))) \;, \tag{9}$$

where ζ is a smoothing coefficient, and $1 - \mathrm{sig}(\zeta \cdot \phi(\mathbf{x}))$ the background probability at a sampling location $\mathbf{x}$. A step-size of $\frac{r}{50}$ is chosen, where r is the depth range of the object-space unit-cube.

6. Experimental Evaluation

Our focus is to evaluate the performance of FroDO on real-world datasets wherever possible. We evaluate *quantitatively* in two scenarios: *(i)* single-view, single object on Pix3D [44]; and *(ii)* multi-view, single object on the Redwood-OS [4] dataset. In addition, we evaluate our full approach *qualitatively* on challenging sequences from the real-world ScanNet dataset [7] that contain multiple object instances. In all cases we use MaskRCNN [15] to predict object detections and masks. We run Orb-SLAM [29] to estimate trajectories and keypoints on Redwood-OS but use the provided camera poses and no keypoints on ScanNet.

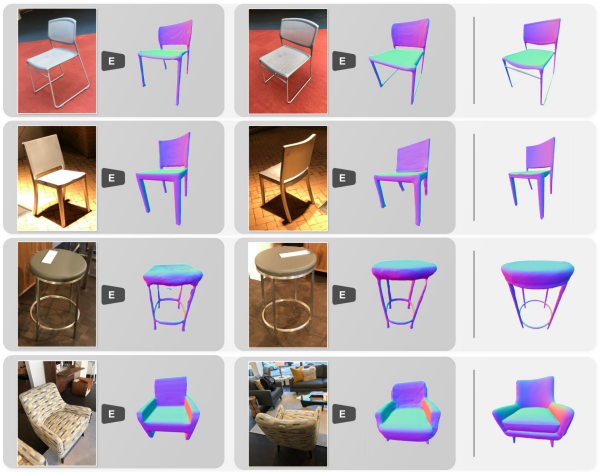

Figure 6: Examples of single view reconstruction on Pix3D dataset [44]. Ground truth on the right for reference.

	IoU ↑	EMD ↓	CD ↓
3D-R2N2 [6]	0.136	0.211	0.239
PSGN [9]	N/A	0.216	0.200
3D-VAE-GAN [50]	0.171	0.176	0.182
DRC [46]	0.265	0.144	0.160
MarrNet [49]	0.231	0.136	0.144
AtlasNet [14]	N/A	0.128	0.125
Sun et al. [44]	0.282	0.118	0.119
Ours (DeepSDF Embedding)	**0.302**	**0.112**	**0.103**
Ours (Joint Embedding)	**0.325**	**0.104**	**0.099**

Table 1: Results on Pix3D [44]. Our method gives the highest Intersection over Union and lowest Earth Mover's and Chamfer Distances.

6.1. Single-View Object Reconstruction

First we evaluate the performance of our single-view shape code prediction network (Sec. 5.2) on the real world dataset Pix3D [44]. Table 1 shows a comparison with competing approaches on the *chair* category. The evaluation protocol described in [44] was used to compare IoU, Earth Mover Distance (EMD) and Chamfer Distance (CD) errors (results of competing methods are from [44]). Our proposed encoder network outperforms related work in all metrics. Table 1 also shows an improvement in performance when our new joint shape embedding is used (Ours Joint Embedding) instead of DeepSDF [32] (Ours DeepSDF Embedding). Figure 6 shows example reconstructions.

Optim. Method	Energy Terms	CD (cm.)
Sparse	$E_s + E_r$	8.97
Sparse	$E_s + E_p + E_g + E_r$	8.59
Sparse + Dense	$E_s + E_r$	7.41
Sparse + Dense	$E_s + E_p + E_g + E_r$	7.38

Table 2: Ablation study of estimates after sparse and dense optimization stages on the Redwood-OS dataset. We compare the effect of different energy terms in Eq. (4).

	PMO (o)	PMO (r)	DIST (r)	Ours (r)
Cars	0.661	1.187	0.919	1.202
Planes	1.129	6.124	1.595	1.382

Table 3: Non-symmetric Chamfer distance (completion) on first 50 instances of the synthetic PMO [24] test set. While (o) indicates the original PMO method with its own initialization, (r) indicates random initialization.

6.2. Multi-View Single Object Reconstruction

We quantitatively evaluate our complete multi-view pipeline on the *chair* category of the real-world Redwood-OS dataset [4] which contains single object scenes. We perform two experiments: an ablation study to motivate the choice of terms in the energy function (Table 2) and a comparison of the performance of the different steps of our pipeline with related methods (Table 4). Table 3 includes a comparison of our dense photometric optimization with the two closest related approaches [24, 25] on a commonly-used synthetic dataset [24].

Ablation study. Table 2 shows an ablation study on different energy terms in our sparse and dense optimizations (Eq. 4). The combination of geometric and photometric cues with a regularizer on the latent space achieves best results. The supplementary material includes further experiments on the effect of filtering object detections (Sec. 5.1) and the efficiency gains of using our joint embedding.

Synthetic dataset. Table 3 shows a direct comparison of the performance on the synthetic PMO test set [24] of our dense optimization when only the photometric loss E_p is used in our energy, with the two closest related methods: PMO [24] and DIST [25]. Notably, both DIST and our approach achieve comparable results to PMO from only random initializations. When PMO is also initialized randomly the results degrade substantially.

Redwood-OS dataset. Table 4 shows a comparison with Pix2Vox [51], a purely deep learning approach, and with PMO [24], both of which are state-of-the-art. For reference, we also compare with COLMAP [40, 41] a traditional SFM approach. Since COLMAP reconstructs the full scene without segmenting objects, we only select points within the ground-truth 3D bounding box for evaluation. We report errors using: Chamfer distance (CD), accuracy (ACC (5cm)), completion (COMP (5cm)) and F1 score – all four commonly used when evaluating on Redwood-OS. Chamfer distance (CD) measures the symmetric error, while shape accuracy captures the 3D error as the distance between predicted points to their closest point in the ground truth shape and vice-versa in the case of shape completion. Both shape accuracy and completion are measured in percentage of points with an error below 5cm. Following [24], we use an average of 35 input frames sampled from the RGB sequences, though for completeness we show results with 350 views. Fig. 7 shows example reconstructions.

We outperform Xie *et al.* [51] by a significant margin which could point to the lack of generalization of purely learning based approaches. We also outperform PMO [24], a shape prior based optimization approach like ours, but which lacks our proposed coarse-to-fine shape upgrade. COLMAP fails to reconstruct full 3D shapes when the number of input images or the baseline of viewpoints is limited as it cannot leverage pre-learnt object priors. Although, as expected, the performance of COLMAP increases drastically with the number of input images, it requires hundreds of views to perform comparably to our approach.

6.3. Multi-Object Reconstruction

We demonstrate qualitative results of our full approach on the ScanNet dataset [7] on challenging real world scenes with multiple object instances in Fig. 1 and Fig. 8. MaskR-CNN [15] was used to predict 2D b-boxes and masks. The association of object detections to 3D object instances becomes an additional challenge when dealing with multi-object scenarios. Our results show that our ray clustering approach successfully associates detected bounding boxes across frames and our coarse-to-fine optimization scheme provides high quality object poses and reconstructions.

7. Conclusions and Discussion

We introduced FroDO, a novel object-oriented 3D reconstruction framework that takes localized monocular RGB images as input and infers the location, pose and accurate shape of the objects in the scene. Key to FroDO is the use of a new deep learnt shape encoding throughout the different shape estimation steps. We demonstrated FroDO on challenging sequences from real-world datasets in single-view, multi-view and multi-object settings. An exciting open challenge would be to extend FroDO to the case of dynamic scenes with independently moving objects.

Acknowledgement The Univ. of Adelaide authors' work has been supported by the Australian Research Council through the Centre of Excellence for Robotic Vision CE140100016 and Laureate Fellowship FL130100102, and UCL authors' work has been supported by the SecondHands project, funded from the EU Horizon 2020 Research and Innovation programme under GA No 643950.

Method	Few observations (average 35 views)				Over-complete observations (average 350 views)			
	CD (cm)	ACC (5cm)	COMP (5cm)	F1 score	CD (cm)	ACC (5cm)	COMP (5cm)	F1 score
COLMAP [40, 41]	10.58	**84.16**	54.28	65.99	**6.05**	**91.41**	**94.59**	**92.97**
Pix2Vox [51]	12.12	55.27	64.74	59.63	11.87	55.88	66.09	60.56
PMO [24]	12.13	53.08	69.42	60.16	11.93	54.80	69.54	61.30
FroDO Code Fusion (Vote)	12.19	60.74	60.55	60.64	11.97	61.37	58.20	59.74
FroDO Code Fusion (Aver.)	10.74	61.31	72.11	66.27	10.57	61.06	72.14	66.14
FroDO Optim. Sparse	8.69	70.58	79.10	74.60	8.59	71.69	81.63	76.34
FroDO Optim. Dense	**7.38**	73.70	**80.85**	**76.64**	7.37	74.78	81.08	77.32

Table 4: Quantitative evaluation on 86 sequences of Redwood-OS. We compare state of the art competitors Pix2Vox [51] and PMO [24] with the results at different stages of our multi-view pipeline (code fusion → sparse optimization → dense optimization). Average code outperforms majority voting. FroDo outperforms all methods when 35 input images are used.

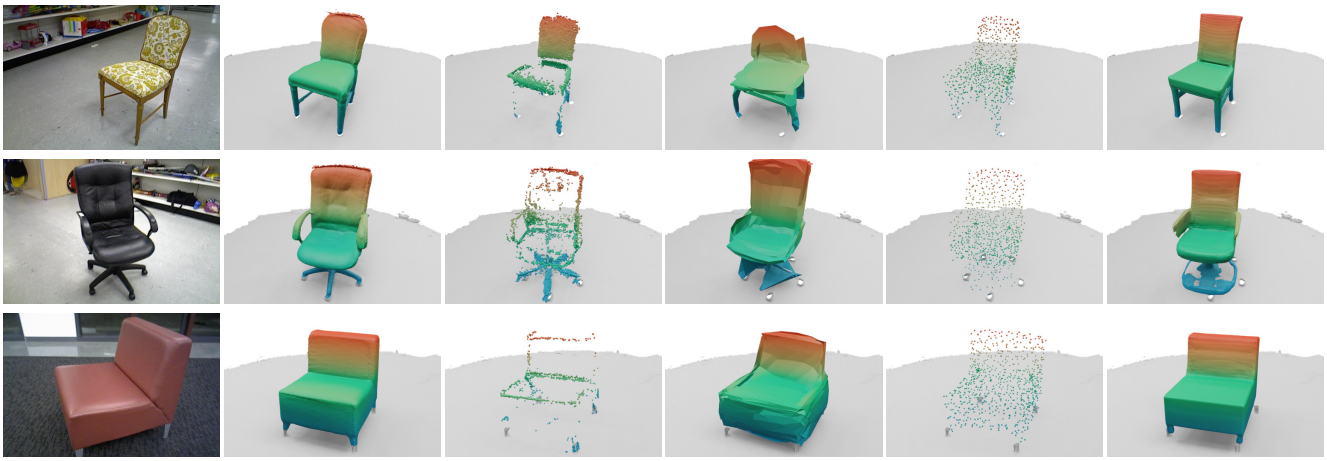

Figure 7: Example 3D reconstructions achieved with different approaches on three sample sequences from Redwood-OS. In all cases 35 input views were used.

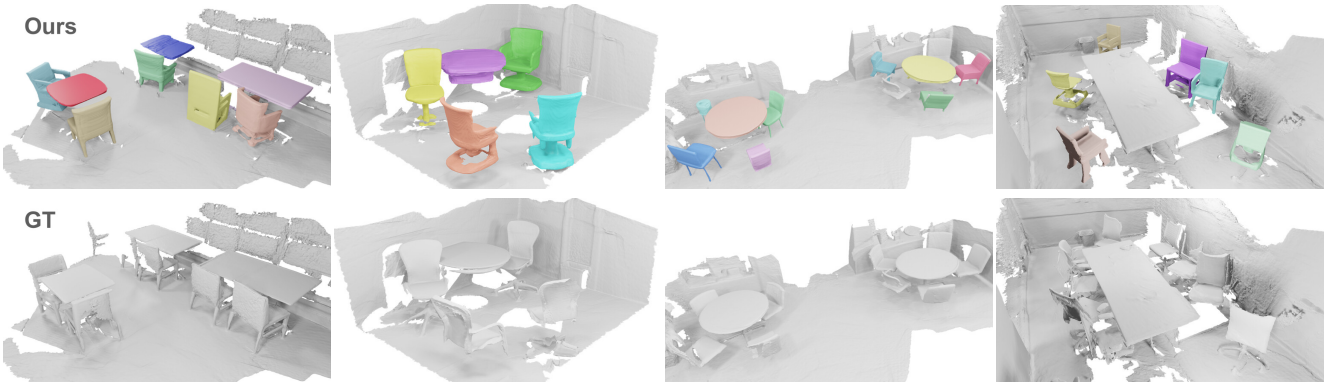

Figure 8: Qualitative results on four ScanNet RGB input sequences. We reconstruct multiple instances of the chair and table classes. While outputs are satisfactory for the first three scenes, the last one highlights failures due to heavy occlusions and partial observations. Top row: Object instances reconstructed by FroDO are shown in colour while grey shows the ground truth background (not reconstructed by our method) for reference. Bottom row: full ground truth scan for comparison.

References

[1] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison. Codeslam—learning a compact, optimisable representation for dense visual slam. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2560–2568, 2018. 3

[2] R. Chabra, J. Straub, C. Sweeney, R. Newcombe, and H. Fuchs. Stereodrnet: Dilated residual stereonet. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11786–11795, 2019. 1

[3] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2, 4, 5

[4] S. Choi, Q.-Y. Zhou, S. Miller, and V. Koltun. A large dataset of object scans. *arXiv:1602.02481*, 2016. 2, 6, 7

[5] S. Choudhary, A. J. B. Trevor, H. I. Christensen, and F. Dellaert. Slam with object discovery, modeling and mapping. *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1018–1025, 2014. 3

[6] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016. 1, 2, 3, 6

[7] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5828–5839, 2017. 1, 2, 6, 7

[8] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid. Dense reconstruction using 3d object shape priors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1288–1295, 2013. 3

[9] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 1, 2, 4, 6

[10] X. Fei and S. Soatto. Visual-inertial object detection and mapping. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 301–317, 2018. 3

[11] D. Gálvez-López, M. Salas, J. D. Tardós, and J. M. M. Montiel. Real-time monocular object slam. *Robotics Auton. Syst.*, 75:435–449, 2015. 3

[12] G. Gkioxari, J. Malik, and J. Johnson. Mesh r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 9785–9795, 2019. 1, 2

[13] A. Grabner, P. M. Roth, and V. Lepetit. 3d pose estimation and 3d model retrieval for objects in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3022–3031, 2018. 2

[14] 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. 2, 6

[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017. 1, 2, 3, 4, 5, 6, 7

[16] M. Hosseinzadeh, K. Li, Y. Latif, and I. Reid. Real-time monocular object-model aware sparse slam. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7123–7129. IEEE, 2019. 3

[17] J. Hou, A. Dai, and M. Nießner. 3d-sic: 3d semantic instance completion for rgb-d scans. *arXiv preprint arXiv:1904.12012*, 2019. 2

[18] J. Hou, A. Dai, and M. Nießner. 3d-sis: 3d semantic instance segmentation of rgb-d scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4421–4430, 2019. 2

[19] A. Kar, C. Häne, and J. Malik. Learning a multi-view stereo machine. In *Advances in neural information processing systems*, pages 365–376, 2017. 3

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 5

[21] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár. Panoptic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9404–9413, 2019. 1

[22] B. Kulis and M. I. Jordan. Revisiting k-means: New algorithms via bayesian nonparametrics. *arXiv preprint arXiv:1111.0352*, 2011. 5

[23] K. Li, R. Garg, M. Cai, and I. Reid. Optimizable object reconstruction from a single view. *arXiv preprint arXiv:1811.11921*, 2018. 3

[24] C.-H. Lin, O. Wang, B. C. Russell, E. Shechtman, V. G. Kim, M. Fisher, and S. Lucey. Photometric mesh optimization for video-aligned 3d object reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 969–978, 2019. 2, 3, 4, 7, 8

[25] S. Liu, Y. Zhang, S. Peng, B. Shi, M. Pollefeys, and Z. Cui. Dist: Rendering deep implicit signed distance function with differentiable sphere tracing. *arXiv preprint arXiv:1911.13225*, 2019. 3, 4, 6, 7

[26] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015. 1

[27] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *2018 International Conference on 3D Vision (3DV)*, pages 32–41. IEEE, 2018. 3

[28] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[29] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *TRO*, 31(5):1147–1163, 2015. 3, 6

[30] S. Muralikrishnan, V. G. Kim, M. Fisher, and S. Chaudhuri. Shape unicode: A unified shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4

[31] L. Nicholson, M. Milford, and N. Sünderhauf. Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam. *IEEE Robotics and Automation Letters*, 4(1):1–8, 2018. 3, 5

[32] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 4, 6

[33] P. Parkhiya, R. Khawad, J. K. Murthy, B. Bhowmick, and K. M. Krishna. Constructing category-specific models for monocular object-slam. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018. 3

[34] S. Pillai and J. Leonard. Monocular slam supported object recognition. *arXiv preprint arXiv:1506.01732*, 2015. 3

[35] V. A. Prisacariu and I. Reid. Shared shape spaces. In *2011 International Conference on Computer Vision*, pages 2587–2594. IEEE, 2011. 3

[36] V. A. Prisacariu and I. D. Reid. Pwp3d: Real-time segmentation and tracking of 3d objects. *International journal of computer vision*, 98(3):335–354, 2012. 6

[37] V. A. Prisacariu, A. V. Segal, and I. Reid. Simultaneous monocular 2d segmentation, 3d pose recovery and 3d reconstruction. In *Asian conference on computer vision*, pages 593–606. Springer, 2012. 3, 6

[38] M. Runz, M. Buffier, and L. Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 10–20. IEEE, 2018. 3

[39] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013. 3

[40] J. L. Schönberger and J.-M. Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 2, 3, 7, 8

[41] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016. 1, 2, 3, 7, 8

[42] V. Sitzmann, M. Zollhöfer, and G. Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *arXiv preprint arXiv:1906.01618*, 2019. 3

[43] J. Stückler and S. Behnke. Model learning and real-time tracking using multi-resolution surfel maps. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012. 3

[44] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 5, 6

[45] K. Tateno, F. Tombari, and N. Navab. When 2.5d is not enough: Simultaneous reconstruction, segmentation and recognition on dense slam. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2295–2302, 2016. 3

[46] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik. Multi-view supervision for single-view reconstruction via differentiable ray consistency. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2626–2634, 2017. 6

[47] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang. Pixel2mesh: Generating 3d mesh models from single rgb images. In *ECCV*, 2018. 2

[48] R. Wang, N. Yang, J. Stueckler, and D. Cremers. Directshape: Photometric alignment of shape priors for visual vehicle pose and shape estimation. *arXiv preprint arXiv:1904.10097*, 2019. 3

[49] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5 d sketches. In *Advances in neural information processing systems*, pages 540–550, 2017. 6

[50] J. Wu, C. Zhang, T. Xue, B. Freeman, and J. Tenenbaum. Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in neural information processing systems*, pages 82–90, 2016. 1, 2, 6

[51] H. Xie, H. Yao, X. Sun, S. Zhou, S. Zhang, and X. Tong. Pix2vox: Context-aware 3d reconstruction from single and multi-view images. *arXiv preprint arXiv:1901.11153*, 2019. 3, 7, 8

[52] B. Xu, W. Li, D. Tzoumanikas, M. Bloesch, A. Davison, and S. Leutenegger. Mid-fusion: Octree-based object-level multi-instance dynamic slam. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5231–5237. IEEE, 2019. 3

[53] S. Yang and S. A. Scherer. Cubeslam: Monocular 3-d object slam. *IEEE Transactions on Robotics*, 35:925–938, 2019. 3

Supplemental Material for FroDO: From Detections to 3D Objects

Figure 1: Examples of 2D detections filtering based on the projection of 3D bounding box. The red and green boxes are 2D detections and projections of 3D bounding boxes respectively. From left to right: reject due to size, reject due to occlusion, reject due to overlap, accept.

1. Detection Pruning

After estimating the bounding box of an object, good views are selected by employing a pruning scheme. This scheme uses three criteria to reject frames based on (1) bounding box size, (2) occlusions and (3) overlap with other objects. (1) is implemented via a threshold on the width and height of the bounding box (BB), and (2,3) are implemented using a threshold on the intersection over union (IoU) of the bounding boxes. Figure 1 illustrates these strategies. The selection helps to identify better initial detections to extract good shape codes and later on leads to an optimization with clean energy terms. An ablation study on the detection pruning on Redwood-OS dataset is demonstrated in Table 1.

2. DP-mean based Line Segment Clustering

Dirichlet Process (DP)-mean clustering is similar to K-mean clustering as it also runs in an Expectation-Maximization manner. A key difference is that the total number of cluster is unknown at first. A new cluster is generated when the distance between a line segment and any existing clusters is larger than a cluster penalty threshold, which we set to 0.4. Our clustering algorithm runs as follows: Given a new object detection, which is represented by a line segment, we calculate the distance between this new line segment and existing clusters. If the distance is larger than the cluster penalty threshold, this observation becomes a new cluster, the cluster center is initialized at the median point of the line segment. Algorithm 1 details each step.

Algorithm 1: DP-mean clustering for 3D line segments

Input: $\boldsymbol{r_1}, \boldsymbol{r_2}, ..., \boldsymbol{r_n}$: n rays
λ : cluster penalty threshold
Output: $c_1, c_2, ..., c_m$: m object clusters
1. init. $\boldsymbol{\mu_1} = mid(\boldsymbol{r_1})$, $mid()$ is to compute the middle point of a ray;
2. **while** *not converged* **do**
  for *each* $\boldsymbol{r_i}$ **do**
    $d_{ij} = \min_{\mu_1 ... k} dist(\boldsymbol{\mu_k}, \boldsymbol{r_i})$;
    if $d_{ij} \geq \lambda$ **then**
      set k = k + 1;
      $\boldsymbol{\mu_k} = mid(\boldsymbol{r_i})$;
    else
      $z_i = j$;
    end
  end
  k clusters are generated, where $c_k = \{\boldsymbol{r_i} | z_i = k\}$;
  for *each* c_i **do**
    $\boldsymbol{\mu_i} = lstsq(c_k)$;
  end
end

3. Dense Shape Code Optimization

In order to implement the dense optimization efficiently, certain measures were taken to achieve an effective opti-

Method	Vote	Average
CD [cm] w/o occlusion filter	12.17	11.73
CD [cm] w/ occlusion filter	11.97	10.57

Table 1: Ablation study on using the inferred 3D bounding box to filter occluded views.

mization procedure. The following two sections explain decisions for the formulation of analytical partial derivatives and for the implementation these.

3.1. Analytical Partial Derivatives

As the DeepSDF decoder yields signed distance values, in contrast to the pointcloud decoder which outputs object coordinates, neither E_g nor E_p are explicitly defined. Therefore, dense object coordinates are extracted by sampling the zero-crossing of the distance field and Jacobians are derived analytically. While most of the derivatives of E_p and E_g wrt. code and pose follow the chain rule, the relevant term $\frac{\partial \mathbf{x}}{\partial \mathbf{z}} = \frac{\partial \mathbf{x}}{\partial G_d}\frac{\partial G_d}{\partial \mathbf{z}}$ needs more attention.

Intuitively, the change of a surface coordinate $\mathbf{x}$ with respect to a change in code $\mathbf{z}$ is ambiguous due to potential changes in topology. However, as shown by [1] it is possible to derive a first-order approximation as follows. Given a location $\mathbf{x}(\mathbf{z})$ on the surface as a function of code $\mathbf{z}$, the corresponding distance value remains constant at the zero-crossing, implying:

$$\frac{\partial G_d(\mathbf{x}(\mathbf{z}), \mathbf{z})}{\partial \mathbf{z}} = 0 \tag{1}$$

Using the multivariable chain rule, and solving for $\frac{\partial \mathbf{x}}{\partial G_d}$, yields:

$$\frac{\partial \mathbf{x}}{\partial G_d} = -\frac{\partial G_d}{\partial \mathbf{x}}^{-1}\frac{\partial G_d}{\partial \mathbf{z}} \tag{2}$$

Here, the pseudo-inverse of $\frac{\partial G_d}{\partial \mathbf{x}}$ is orthogonal to the tangent plane and hence points in the direction of the surface normal. Since the SDF decoder G_d is differentiable, $\frac{\partial \mathbf{x}}{\partial \mathbf{z}}$ can be calculated easily using this first-order approximation. Figure 2 visualizes $\frac{\partial G_d}{\partial \mathbf{z}}$ for 9 different shape code components and demonstrates that many parts of an object are affected, when a single component changes.

3.2. Implementation Details

Multiple techniques are employed in order to speed up the dense optimization stage. Since executing the network is the most expensive step in the optimization, the number of forward and backward passes are kept to a minimum. At the beginning of each iteration the DeepSDF volume is sampled with an adaptive resolution. First, on a dense grid with a distance of δ between samples along a every dimension and afterwards at a higher resolution for voxels close

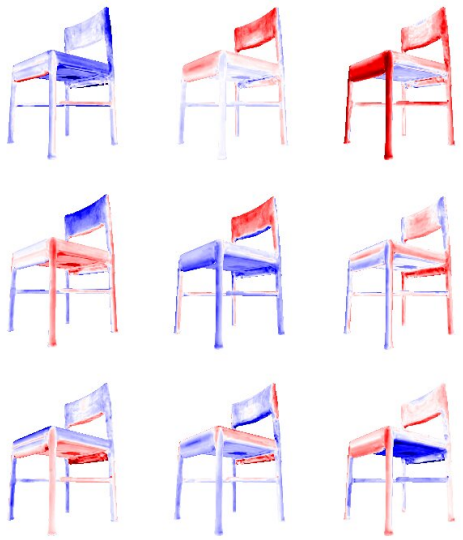

Figure 2: Visualization of $\frac{\partial G_d}{\partial \mathbf{z}}$ for 9 different code components, demonstrating shape changes when single components change. Red (positive values) indicates an intrusion, while blue (negative values) indicates an extrusion.

the object boundary, i.e. when $-\delta < G_d < \delta$, where δ is the DeepSDF truncation factor as described by Park *et al.* [7]. A mesh corresponding to the current code estimate is then obtained by running marching cubes [2] on the samples and used as a proxy for the current state. Each vertex contains $\frac{\partial G_d}{\partial \mathbf{x}}$ and $\frac{\partial G_d}{\partial \mathbf{z}}$ and as a result data required to minimize the energies is generated by rendering the mesh to observed detections and is independent of sampling. Rasterized mesh data is directly copied from OpenGL to CUDA for the optimization.

Further, a pyramid scheme is implemented which aids convergence when starting from a bad initialization, improves run-time and reduces the memory footprint of the optimization. For every detection with width w and height h, only pyramid levels l with a bounded shape ($r_{min}^2 < \frac{w \cdot h}{4^l} < r_{max}^2$) are considered. In our experiments, r_{min} and r_{max} are always 40 and 400 respectively.

4. Network Architecture and Training

4.1. Decoder for Joint embedding

The shared joint embedding is 64 dimensional. The network is split into two independent branches for point cloud and DeepSDF respectively. The point cloud branch is composed of four fully connected layers each with 512, 1024, 2048, 2048×3 as output dimensions. We use ReLU as a nonlinear activation function following each fully connected layer except the last one. Readers can refer to [7]

Method	sec. /iteration	# of iteration
PMO [5]	12.59	100
Optim. Dense	4.96	100
Optim. Sparse	0.07	200

Table 2: Speed comparison between PMO [5] and our method on the same sequence of 60 frames.

for a detailed discussion of the DeepSDF architecture that is employed in the dense branch. The joint autodecoder is trained for 2000 epochs with a batch size of 64. There are 16384 SDF samples for each shape. The learning rate for network parameters and latent vectors are $5 \cdot 10^{-3}$, and 10^{-3} respectively, decayed by 0.5 for every 500 epochs.

4.2. Encoder Network

After we train the decoder network as described in Sec. 4.1, we obtain a set of embeddings $\mathbf{z} \in \mathcal{R}^{64}$ for the corresponding CAD models. In the second stage, we train an encoder network that maps an image to the latent code. We tailor ResNet50 to output a vector of dimension 64 and initialize the network with pretrained models. We train the network for 50 epochs to minimize a Huber loss with a polynomial decaying learning rate of 10^{-3}. The network for the embedding is trained in a way similar to Li *et al.* [4]. However, our deep learning based shape embedding is very different from the non-parametric embedding used in [4].

5. CAD retrieval on Pix3D dataset

In Fig. 3 we show more examples of CAD model retrieval on Pix3D dataset [8]. We find the nearest CAD model based on Euclidean distance of our shape codes.

6. Dual-Representation Efficiency Gain

One motivation for using a shared code space for a point-based and SDF-based representation is that the efficiency of a sparse optimization can be leveraged, while exploiting richer information when subsequently applying fewer dense iterations. A comparison of optimization run-times shows that the pointcloud-based representation is approximately two orders of magnitude faster than the mesh optimization used by PMO [5] and DeepSDF [7] as shown in Table 2.

7. Redwood Dataset Augmentation

While Pix3d [8] is a real-world dataset available for single-view object shape reconstruction, multi-view datasets with ground truth 3D data are scarce. Previous multi-view learning based methods primarily evaluate their methods on synthetic data [5]. We post-process the Redwood-OS dataset [3] and will release it to facilitate benchmarking on real-world data. It contains RGBD sequences and dense full scene reconstructions. To isolate object shapes for evaluation, we manually select sequences that have full 3D object shapes and manually segment them from the full scene mesh. We run RGBD ORB-SLAM2 [6] to estimate camera poses.

8. Qualitative results on Redwood-OS dataset

Additional qualitative results on the Redwood-OS dataset are shown in Figure 4. As in the main paper, groundtruth, sparse COLMAP, dense COLMAP, our sparse reconstructions and our dense reconstructions are compared.

References

[1] M. Atzmon, N. Haim, L. Yariv, O. Israelov, H. Maron, and Y. Lipman. Controlling neural level sets. In *Advances in Neural Information Processing Systems*, pages 2032–2041, 2019. 2

[2] E. Chernyaev. Marching cubes 33: Construction of topologically correct isosurfaces. Technical report, 1995. 2

[3] S. Choi, Q.-Y. Zhou, S. Miller, and V. Koltun. A large dataset of object scans. *arXiv:1602.02481*, 2016. 3

[4] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. J. Guibas. Joint embeddings of shapes and images via cnn image purification. *ACM transactions on graphics (TOG)*, 34(6):234, 2015. 3

[5] C.-H. Lin, O. Wang, B. C. Russell, E. Shechtman, V. G. Kim, M. Fisher, and S. Lucey. Photometric mesh optimization for video-aligned 3d object reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 969–978, 2019. 3

[6] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017. 3

[7] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2, 3

[8] X. Sun, J. Wu, X. Zhang, Z. Zhang, C. Zhang, T. Xue, J. B. Tenenbaum, and W. T. Freeman. Pix3d: Dataset and methods for single-image 3d shape modeling. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 3

Figure 3: From left to right: input image and the nearest CAD models in latent space sorted by distances.

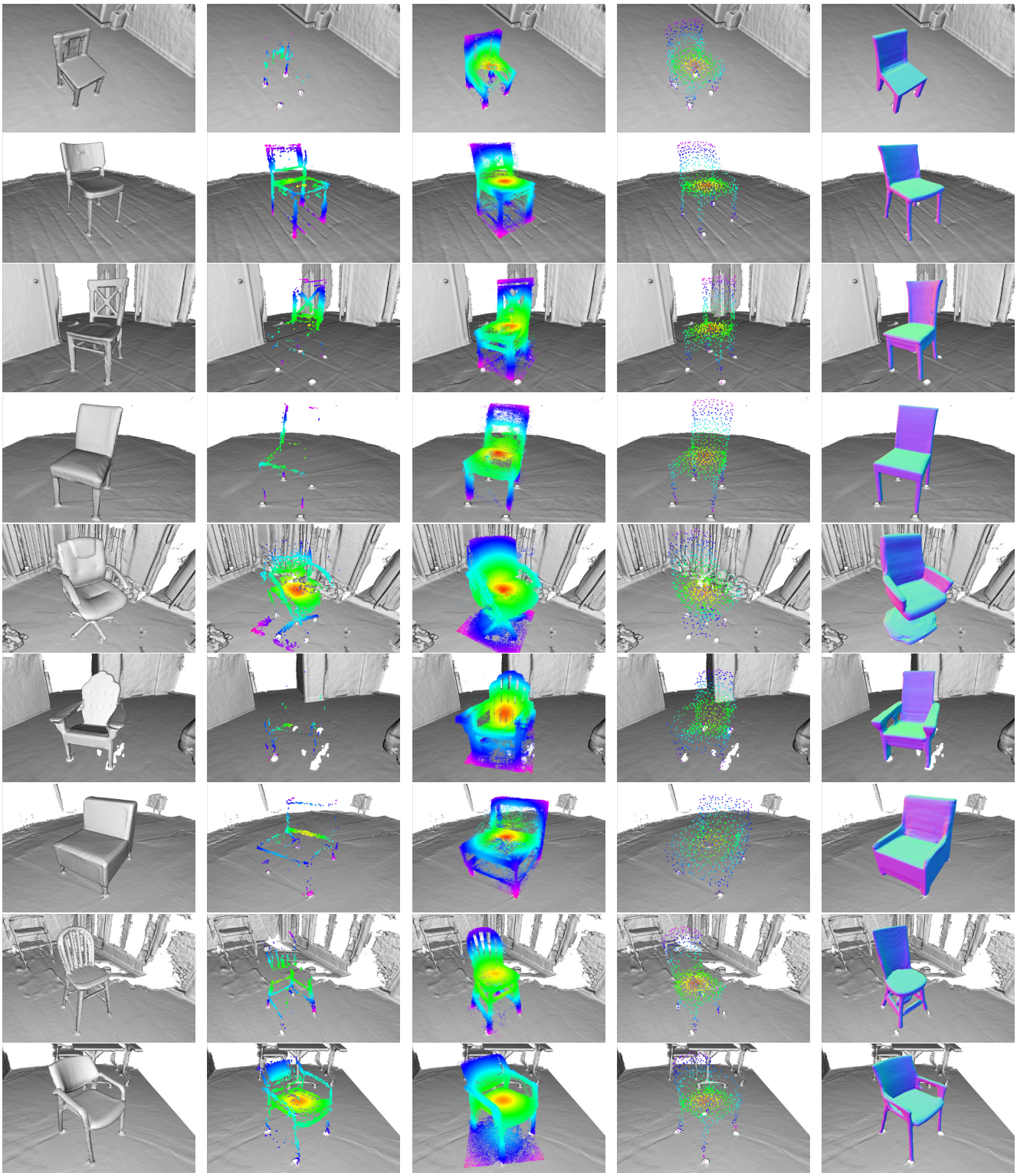

Figure 4: Comparison of different approaches to object shape reconstruction on some examples from Redwood-OS dataset.