

MonoSelfRecon: Purely Self-Supervised Explicit Generalizable 3D Reconstruction of Indoor Scenes from Monocular RGB Views

Runfa Li¹ Upal Mahbub² Vasudev Bhaskaran² Truong Nguyen¹
¹UC San Diego ²Qualcomm

Abstract

Current monocular 3D scene reconstruction (3DR) works are either fully-supervised, or not generalizable, or implicit in 3D representation. We propose a novel framework - **MonoSelfRecon** that for the first time achieves explicit 3D mesh reconstruction for generalizable indoor scenes with monocular RGB views by purely self-supervision on voxel-SDF (signed distance function). MonoSelfRecon follows an Autoencoder-based architecture, decodes voxel-SDF and a generalizable Neural Radiance Field (NeRF), which is used to guide voxel-SDF in self-supervision. We propose novel self-supervised losses, which not only support pure self-supervision, but can be used together with supervised signals to further boost supervised training. Our experiments show that "MonoSelRecon" trained in pure self-supervision outperforms current best self-supervised indoor depth estimation models and is comparable to 3DR models trained in fully supervision with depth annotations. MonoSelfRecon is not restricted by specific model design, which can be used to any models with voxel-SDF for purely self-supervised manner.

1. Introduction

3D scene reconstruction is one of the most important problems in 3D computer vision, robotics, virtual reality, autonomous driving and so on. Indoor 3DR is even more challenging because of the large texture-less region and complicated relations between objects. As more complicated and accurate models emerge, the main concern shifts to three standards: 1. Self-supervised training without laborious large-amount of annotations in depth or SDF, 2. Generalizable to same type of scenes, and 3. Explicit 3D mesh representations.

Explicit monocular 3DR models originate from depth estimation [13, 19, 28, 33, 40, 42, 48], where depth maps can be directly supervised with pixel-wise depth ground truth. Depth maps can be also self-supervised by continuous RGB sequences from multiple camera views, with or without camera poses. Under constraints such as tempo-

Methods \ Standards	Self-Supervised	Generalizable	Explicit 3D Mesh
Self-supervised Depth	✓	✓	✗
Supervised Depth	✗	✓	✗
Supervised Voxel-SDF	✗	✓	✓
Implicit NeRF	✓	✗	✗
Explicit SDF-NeRF	✓	✗	✓
Ours	✓	✓	✓

Table 1. Monocular 3DR measured in three major standards.

ral smoothness, multiple key frames are selected from continuous depth map sequence, fused to global 3D volume and integrate “Truncated Signed Distance Function” value (TSDF) [28] for each voxel within the truncation distance. 3D surface mesh can be obtained from “marching cube” algorithm with estimated TSDF [24]. However, because of the depth inconsistency in multi-views, directly fusing depth estimation for TSDF may cause it either layered or too sparse. Additionally, using depth maps from multiple views for TSDF fusion typically causes large overlapping regions, which is a waste of computation in inference[36]. Later works [27, 35, 36] overcome the limitations by pre-defining 3D voxels and directly regressing SDF for each voxel. However, these works must be trained in supervision with laboriously large amount of fine-grained SDF annotations fusing from depth map ground truth.

Implicit monocular 3DR works start to dominate with the emergence of NeRF[25]. One advantage of implicit 3DR over explicit ones is its purely self-supervised training. With RGB sequence from different camera poses, NeRF reconstructs scenes in implicit representation by synthesizing images from new views. Recently, State-of-the-art (SOTA) works introduce implicit SDF function to NeRF, where the SDF-NeRF can estimate explicit SDF values w.r.t any specified 3D position in the scene [10, 30, 41, 46, 50, 54]. However, these 3DR works are still non-generalizable to other scenes. Although few works such as [16, 51] make NeRF generalizable, the generalization has not been extended to estimate explicit SDF values.

We summarize current 3DR methods based on the three standards in Table 1, where each type of the method is discussed in section 2. Different to all existing works, where the 3DR either requires large amount of annotated ground

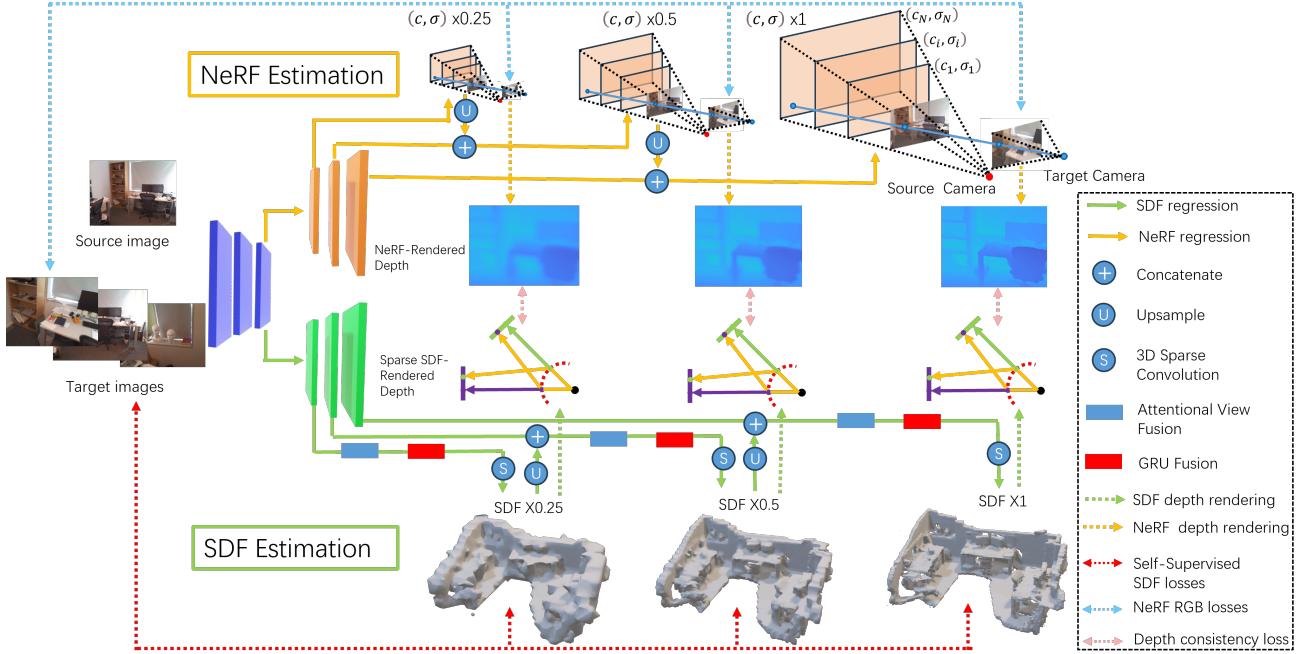


Figure 1. MonoSelfRecon Pipeline. We use monocular RGB sequence to estimate explicit 3D mesh of generalizable indoor scenes under purely self-supervision free of SDF or depth annotations. A coarse-to-fine architecture is used for both SDF and NeRF decoders, where our proposed self-supervised losses are used between SDF-inputRGB, NeRF-inputRGB, and SDF-NeRF.

truth, or non-generalizable, or not explicit in 3D representation, our work overcomes these three concerns altogether. Our key contributions are:

- We propose a novel framework ‘‘MonoSelfRecon’’ that first time achieves **explicit 3D mesh** reconstruction for **generalizable** indoor scenes with monocular RGB sequence by purely **self-supervised** training on voxel-SDF, which is different to all existing works.
- We propose novel self-supervised losses and conduct purely self-supervised training without SDF or depth ground truth. Experiments show that our approach outperforms the best self-supervised generalizable indoor 3DR (depth-based) works, and is comparable to fully-supervised works.
- Our proposed self-supervised losses can be used together with supervised losses to boost fully supervised training. The framework is not limited to specific models but free to extend to any models with voxel-SDF estimation. Consequently, it keeps the advantages of the original model, such as inference speed and memory efficiency.

2. Related Works

In this section, we discuss details of each method based on the three standards in Table 1.

Generalizable Explicit depth Reconstruction. Initially, 3D mesh reconstruction relies on depth estimation, the typical way is to perform TSDF fusion from depth and use marching cube to recover mesh from TSDF. Fully supervised training is the first choice when available depth map

ground truth is used. These models take as input single image, and explore different model designs [5, 19, 22, 38, 42], but all regress to 2D depth map and directly supervise with depth ground truth. However, since annotations are missing and unreliable in many cases, self-supervision has to be used without depth annotations. With camera poses, the depth can be estimated by binocular stereo matching, leveraging epipolar-geometric consistent matching between image pixels [42], or replacing pixels with extracted deep features to produce a matching cost volume [4, 23]. Without camera poses, it becomes a structure-from-motion (SFM) problem, the network can jointly estimate depth and pose with reconstruction error [9, 14, 15, 53, 56]. However, self-supervised depth has scale drifting and ambiguity problems, which need other priors to recover real scale. Alternatively, we directly regress voxel-SDF from network in real scale.

Generalizable Supervised Explicit 3D mesh Reconstruction. Directly regressing voxel-SDF is generalizable and straightforward, the advantages are real-scale, time and memory efficient. However the huge volume of 3D convolution [27] requires strong supervisory signals - voxel-SDF ground truth fused from depth ground truth, although later works [35, 36] shrink the volume by using 3D sparse convolution, SDF ground truth is still inevitable, which is not easy to obtain in many cases. In this work, we explore pure self-supervision to train voxel-SDF free from any SDF or depth annotations.

Unsupervised implicit 3D Reconstruction. NeRF [25] is the representative of implicit 3DR. Many works [1, 7, 21,

[26, 37, 47, 51, 55] improve the baseline NeRF from different aspects. MVSNeRF [1] speeds up NeRF in a cost volume built within a multi-view-system, while PointNeRF [47] speeds up by introducing point cloud from estimated depth maps to sample the ray-level key points only with the nearby point cloud. However, all these works require per-scene optimization and are limited to implicit RGB synthesis. PixelNeRF [51] improves volume rendering function by allowing rays from one view to refer to key points on rays from other views, which endows NeRF a limited generalization by using fewer views. Later works [16, 44] explored increasing generalization by adding an explicit image encoder on top of positional encoding that enables NeRF to generalize to similar but unseen scenes. However, these works are still implicit NeRF where the outputs are synthesized RGB image, unlike our desired explicit 3D mesh.

Non-Generalizable Unsupervised Explicit 3D mesh Reconstruction. The standard NeRF for implicit 2D view synthesis has been successfully extended to explicit 3D mesh representation [3, 10, 17, 29, 30, 41, 43, 46, 50, 54]. With specified 3D coordinates and pose as input, these models estimate SDF values wrt. ray-level key points. The core design is a density function converting per-point SDF estimation to opacity values for volumetric rendering [50], where the ray-level SDF values are queried to obtain rendered pixel intensity, which brings SDF to the reconstruction loss. Later works add more priors to supervise together with SDF-NeRF loss. [10] uses supervised pre-trained depth and semantic segmentation estimator, while [54] and [41] use supervised pre-trained depth estimator and surface normal detector. However, the generalization ability shown in implicit NeRF has not been successfully extended to explicit SDF-NeRF, which means once trained, the SDF-NeRF can only estimate 3D mesh of a specific scene. Our work aims to keep the explicit representation, self-supervised training, while enables generalization.

Motivation. The question is that If these per-scene optimization methods are fast enough, then there's no disadvantage to them. However, non-generalizable methods take a long time to converge per scene. Neuralangelo[17] takes over 4 hours(single 3090Ti) to converge on a simple lego toy and almost a day for a big room, which is comparable to the generalized methods (ours 36 hours training on single 3090Ti but once trained can be used on different indoor scenes). Non-generalizable methods can be used where a scene is accessed repeatedly and unchanged, such as a city block, long training is acceptable as once trained it can be used for a while unless the scene changes. However, generalizable methods are better when different scenes need to be reconstructed ASAP and no time for per-scene optimization for hours, such as a real estate agent visits 20 different buildings in a day to scan 3D mesh models for unseen rooms and send it to clients right away. One compromise is part opti-

mization but it is only suitable for unchanged scenes, whenever the scene is changed, they need to retrain the NeRF of corresponding part, while ours don't need retrain and can infer the new building in real time. We adapted NeuralRecon as backbone, our inference speed is 30 ms per frame (single 2080Ti).

3. Method

3.1. MonoSelfRecon Framework

Figure 1 shows our MonoSelfRecon framework. In both training and testing, we take the input monocular RGB sequence and camera poses, and reconstruct 3D mesh of the whole scene. We select key frames following the process in [12], where we consider a valid frame to be “greater than 15 degree in rotation or 0.3 meter in translation” to the previous frame. Every n consecutive key frames form a scene fragment. Fragments are fed as inputs, from which, the network extracts 2D features per key frame, creates 3D feature volume and jointly estimates SDF and NeRF using separate decoders at three pyramid scales. In SDF decoder, 2D features are fused to 3D voxel features and regress to voxel-SDF values at each level with 3D sparse convolution. Every time the network only estimates SDF corresponding to the fragment in a $[N, N, N]$ 3D voxel region, the Gated Recurrent Unit (GRU) module at each level updates SDF, fuses reconstruction from previous fragments, and completes the whole scene. During training, self-supervised losses are implemented between SDF-input, NeRF-input, and SDF-NeRF, with detailed discussion in 3.2. During testing, 3D mesh can be obtained from SDF through marching cube[24].

Attentional View Fusion. For each fragment, 3D voxel features can be simply obtained by projecting the 3D voxel to each 2D view in the fragment, searching for visible corresponding pixels, and averaging 2D features. However, since each view have different distance, angle, and occlusion to voxels, 2D features from different views should not contribute the same to 3D features. Inspired by recent works [35], we use an attentional view fusion module by adding a light transformer before averaging features. The transformer takes unordered sequence of 2D features and updates weighted features before average to 3D, which enables more flexibility to adjust the contribution of each frame to the fragment. Although simple, this module achieves significant improvement as shown in the ablation study in Table 4.

GRU. We adapt the GRU module from [36]. With camera poses, it is simple to concatenate fragments and replace the overlapping voxels of latter fragment to the previous one, but it ignores the effect of the latter views to the previous ones. Once fusing 3D voxel features from 2D and before regressing voxel-SDF at each level, the GRU module takes 3D voxel features of both previous and current fragments as input to update the current features. GRU fusion makes

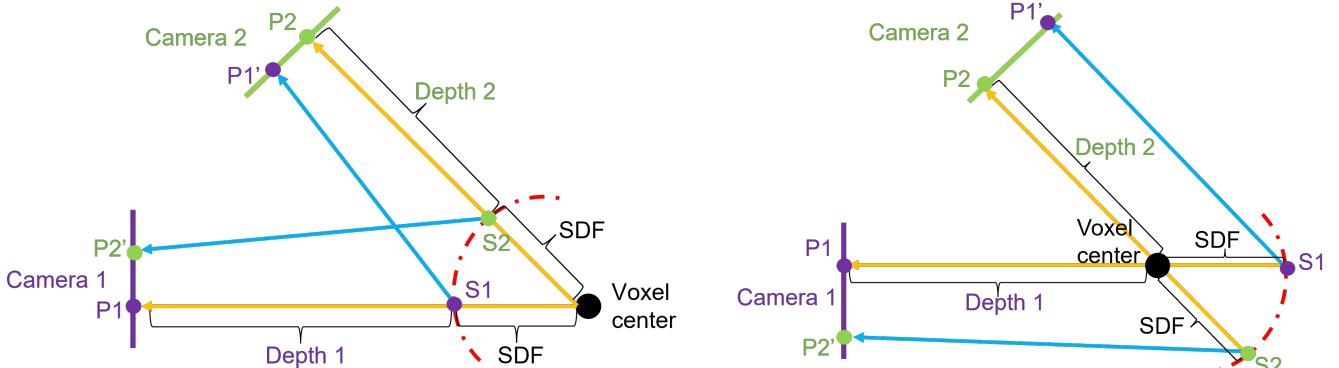


Figure 2. **Self-supervised SDF photometric loss between source and target views.** **Left:** Voxel center is inside of the surface, SDF is negative. Orange arrows show projecting rays from voxel centers to 2D pixels (P_1 , P_2) on each camera plane, blue arrows show reprojection of surface 3D points (S_1 , S_2) to 2D pixels (P_1' , P_2') in each camera plane. Surface points are estimated by SDF estimation. **Right:** Voxel center is outside of the surface, SDF is positive. **The loss is extended to all n views in a fragment.**

an obvious improvement as shown in the ablation study in Table 4.

NeRF. We adopt the generalizable MPI (MultiPlane-Images)-NeRF [8, 16, 57], introduce it to the framework and jointly train with SDF to boost SDF estimation. The core idea is to use an explicit encoder on top of standard positional encoding to enable implicit NeRF with generalization ability. The NeRF estimation also further boosts our SDF performance as shown in the ablation study in Table 4.

3.2. Self-supervised Losses

SDF Photometric Loss. Figure 2 shows a simplified version of the loss implementation between two camera views, where the corresponding 2D coordinates (P_1 and P_2) can be found by tracing rays (orange arrows) to camera planes. SDF is the distance between a point to its nearest surface, where the value is negative when the point is inside of the surface, and positive when it is outside of the surface. The model estimates SDF per voxel. The depth \hat{D}_{cam} can be estimated by Eq. 1, where V_{world} is 3D world coordinate of the pre-defined voxel center, $T_{world \rightarrow cam}$ is camera extrinsic, \hat{SDF} is the estimated voxel-SDF from the model. With depth and voxel center in the camera coordinate, the surface points S_1 and S_2 can be estimated by Eq. 2, where $r\vec{ay}$ is the unit vector at ray direction (orange arrows), and \hat{S}_{cam} is the 3D coordinate of a surface point in the camera view. Finally, the reconstructed pixels are obtained by reprojecting (blue arrows) surface points S_1 , S_2 to camera 2 and 1 as Eq. 3, where K is camera intrinsic, $T_{cam \rightarrow cam'}$ is camera pose from cam to cam' . P - P' is a pixel reconstruction pair with same photometric intensity, where a photometric consistency loss can be derived as Eq. 4. The exact pixel intensities $I_{cam}(P)$ and $I_{cam'}(P')$ are obtained with bilinear interpolation from projected 2D points lying between integer coordinates, and the loss is only traced to the points lying within the camera planes.

$$V_{cam} = T_{world \rightarrow cam} V_{world} \quad (1)$$

$$\hat{D}_{cam} = V_{cam} + \hat{SDF}$$

$$\hat{S}_{cam}(x, y, z) = V_{cam}(x, y, z) + |\hat{SDF}| r\vec{ay}(x, y, z) \quad (2)$$

$$P = \text{Interp}(KV_{cam}) \quad (3)$$

$$P' = \text{Interp}(K'T_{cam \rightarrow cam'} \hat{S}_{cam})$$

$$L_{sdf} = \sum_{P \in cam} \sum_{P' \in cam'} |(I_{cam}(P) - I_{cam'}(P'))| \quad (4)$$

In practice, we implement the loss across all views in the scene fragment and take the weighted average loss, where the weight is in direct proportion to the number of the candidate P - P' pair. If P' lies outside of the other camera’s plane, we ignore this P - P' pair. For SfM-based self-supervised depth works, they start from 2D pixel and ends up at 2D pixel to jointly regress depth and camera pose. However, we start from 3D voxel center and end up at 2D pixel to only regress the depth model while taking camera pose as prior, which is why SfM-based self-supervised depth estimation has scale ambiguity, while our SDF estimation is directly in real scale.

SDF Co-Planar Loss. Photometric constraints are insufficient for indoors scenes due to large non-textured regions and in-plane rotations. Thus, we take advantage of the special geometric constraints in indoor scenes. Most indoor scenes have large planes such as walls, floors, and ceilings, where textures within such planes are mostly similar. Inspired by [18, 20, 52, 53] that implement planar constraints in 2D depth maps, we extend it to 3D SDF. Specifically, we adopted ‘Felzenszwalb superpixel segmentation’ [6] to extract ‘super-pixels’, which covers piece-wise large group of regions that have low pixel intensity gradients, which are considered as a planar region. The algorithm uses greedy search to extract super-pixels and is free of learning. Based on the planar segmentation and the depth planar constraints from [15], we propose a voxel-SDF driven co-planar loss.

Our goal is to derive plane parameters under planar constraints, and learn the plane parameters in a self-supervised manner. Specifically, the plane segmentation extracts n super-pixels from a 2D image, with each super-pixel corresponding to a continuous plane. For the 2D projected voxel center point P (as shown in Figure 2), if it belongs to super-pixel SP_m in the 2D plane, then the surface 3D point S corresponding to P also belongs to the surface plane of class m in 3D space. Using the surface point S , the plane m can be defined as Eq. 5, where \hat{s}_0 is an estimated surface point in the plane, and A_m is the plane parameter.

$$A_m^T \hat{s}_0 = 1 \quad (5)$$

While using only one 3D point to simulate a plane is ill-posed, a large number of estimated 3D surface points are obtained by projecting voxel centers to different camera views. With n 2D projected points p_1, p_2, \dots, p_n belonging to super-pixel SP_m , there are n 3D surface points s_1, s_2, \dots, s_n belonging to 3D surface plane m . Eq. 5 is extended to Eq. 6, where $\hat{S}_n = [\hat{s}_1, \hat{s}_2, \dots, \hat{s}_n]$, and $Y_m = \vec{1} = [1, 1, \dots, 1]$.

$$\hat{S}_n A_m^T = Y_m \quad (6)$$

The plane parameter A_m is then estimated by least-square method as Eq. 7, where ϵ is a small scalar for stability, and I is an identity matrix.

$$A_m = (\hat{S}_n^T \hat{S}_n + \epsilon I)^{-1} \hat{S}_n^T Y_m \quad (7)$$

With the estimated plane parameter, the pseudo surface points can be retrieve by $\hat{S}_n' = (A_m^T \hat{S}_n)^{-1}$. The pseudo surface and estimated surface are expected to align together, and we implement such a co-planar geometric constraint as the self-supervised co-planar SDF loss as Eq. 8.

$$L_{plane} = \sum_M \sum_N |\hat{S}_n - \hat{S}_n'| \quad (8)$$

Depth Consistency Loss. We also propose depth consistency loss to further boost SDF from NeRF. Specifically, we estimate sparse Pseudo-SDF depth for target views from estimated SDF (as Figure 2), and render NeRF-depth for corresponding target views. Since Pseudo-SDF depth is in real scale, we first use it to recover NeRF-depth's scale, and enforce consistency between the two estimated depths.

$$L_{depth} = \sum_N \sum_{D \in cam} |\hat{D}_{sdf} - \hat{D}_{NeRF}| \quad (9)$$

where \hat{D}_{sdf} and \hat{D}_{NeRF} are Pseudo-SDF depth and the scale-recoverd NeRF-depth, respectively.

Total Loss. We implement standard NeRF losses for NeRF encoder, including RGB consistency with input images, SSIM, and smooth loss, and we jointly train everything end-to-end in pure self-supervision.

$$L_{NeRF} = L_{rgb} + L_{smooth} + (1 - SSIM) \quad (10)$$

The total loss is the weighted sum of all losses, where λ_s are the weights,

$$L_{total} = \lambda_{sdf} L_{sdf} + \lambda_{plane} L_{plane} + \lambda_{depth} L_{depth} + \lambda_{NeRF} L_{NeRF} \quad (11)$$

4. Experiments

Datasets. We conduct our experiments on ScanNetV2 [2] and 7Scenes datasets [34] for training and evaluation. ScanNet is a real dataset for indoor 3DR, which consists of 2.5M images captured in 1513 different indoor scenes. 7Scenes is another challenging indoor real dataset, consisting of 7 scenes, with rich contents of RGB, depth, and camera pose.

Metrics. We use the same evaluation metrics as Atlas [27] and NeuralRecon [36], and we evaluate both 3D geometric mesh metrics and 2D rendered depth metrics. As in [27], since there is no depth directly estimated from the model, we project 3D mesh to get 2D depth maps and use the rendered depth maps for 2D metrics evaluation.

Implementation Details. The weights for each loss “ λ_{sdf} , λ_{plane} , λ_{depth} , λ_{NeRF} ” are “1, 0.05, 1, 1”, respectively. For fair comparison with previous voxel-SDF works, we use the same voxel configurations where voxel size is set to 4cm, scene fragment resolution is set to $96 \times 96 \times 96$, SDF truncation distance is set to 30cm, max depth for ground truth SDF fusion is set to 3m. We train and test on all samples based on standard training/testing splits of ScanNet where entire framework is trained end-to-end in pure self-supervision from scratch with randomly initialized weights, except for the image encoder Resnet-18, which we choose to be light and efficient. We train on randomly separated scene fragments over all scenes for the first 20 epochs without GRU, which is to reinforce the inner-consistency within fragments and warmup for each part of the framework. We then train on each continuous scene fragments over all scenes together with GRU to reinforce the inter-consistency between fragment. We use 7Scenes only for few-shot transfer and testing. We use NeRF decoder only in training to boost SDF estimation, and we only keep SDF for testing and inference.

4.1. Evaluation Results

Baselines. We conduct evaluation and comparison with monocular 3DR in Table 1. Although non-generalizable methods[3, 17, 41] generate accurate results, they train with one scene and can be only tested on the same scene, which is under completely different training/testing setup to ours, so we do not directly compare with these works. For the generalizable supervised voxel-SDF methods [27, 36], we directly compare both 3D geometric mesh and 2D rendered depth. Since we get 3D mesh ground truth by fusing SDF with depth map ground truth, to make the comparison rigorous and fair, we get estimated 3D mesh by fusing SDF with

2D Depth	Train	AbsRel \downarrow	AbsDiff \downarrow	SqRel \downarrow	RMSE \downarrow	$\delta 1 \uparrow$	Comp \uparrow
MVDepthNet[42]	sup	0.098	0.191	0.061	0.293	89.6	0.928
GPMVS[11]	sup	0.130	0.239	0.339	0.472	90.6	0.928
DPSNet[5]	sup	0.087	0.158	0.035	0.232	92.5	0.928
COLMAP[32]	sup	0.137	0.264	0.138	0.502	83.4	0.871
Atlas[27]	sup	0.065	0.123	0.045	0.251	93.6	0.999
NeuralRecon[36]	sup	0.065	0.106	0.031	0.195	94.8	0.909
NeuralRecon[36]	weak	0.107	0.183	0.046	0.268	87.2	0.817
Distdepth[45]	self	0.360	0.485	0.308	0.583	48.0	0.889
Monodepth2[9]	self	0.205	-	0.129	0.453	67.9	-
P2Net[53]	self	0.253	0.336	0.171	0.429	65.1	0.888
Structdepth[15]	self	0.219	0.139	0.139	0.383	70.9	0.896
MonoNeRF[8]	self	0.169	-	0.089	0.375	76.0	-
Ours	self	0.143	0.231	0.090	0.333	79.2	0.872
Ours	weak	0.089	0.134	0.038	0.208	89.8	0.827
3D Mesh	Train	Comp \downarrow	Acc \downarrow	Recall \uparrow	Prec \uparrow	F-score\uparrow	
MVDepthNet[42]	sup	0.040	0.240	0.831	0.208	0.329	
GPMVS[11]	sup	0.031	0.879	0.871	0.188	0.304	
DPSNet[5]	sup	0.045	0.284	0.793	0.223	0.344	
COLMAP[32]	sup	0.069	0.135	0.634	0.505	0.558	
Atlas[27]	sup	0.062	0.128	0.732	0.382	0.499	
NeuralRecon[36]	sup	0.120	0.062	0.428	0.592	0.494	
NeuralRecon[36]	weak	0.262	0.089	0.157	0.308	0.205	
Distdepth[45]	self	0.262	0.371	0.159	0.104	0.121	
P2Net[53]	self	0.170	0.264	0.231	0.148	0.179	
Structdepth[15]	self	0.170	0.217	0.243	0.181	0.207	
Ours	self	0.212	0.185	0.262	0.263	0.260	
Ours	weak	0.197	0.075	0.293	0.469	0.358	

Table 2. **2D Depth and 3D Mesh metrics on ScanNet.** “sup” “self” “weak” denote supervised, self-supervised, and weakly supervised training, respectively. We use the same training/testing splits as [27, 36], where the results of [5, 11, 32, 42] are directly from [27, 36]’s papers, and we test [15, 45, 53] in this work. Results of [8, 9] are directly from [8] since it also uses same training/testing splits as [27, 36] and our method. Since MonoNeRF[8]’s code is not released, 3D mesh of [8, 9] from Table 2 are not available, so they are not compared in this table.

estimated depth from self-supervised/supervised depth estimation works. For supervised depth methods [5, 11, 42], we directly use their depth estimation for SDF fusion, for self-supervised depth [14, 15, 53, 56], and since there is scale ambiguity, we first use ground truth depth to recover real scale for estimated depth, and then fuse SDF. NYUV2 is the most popular dataset for indoor depth estimation. Since our SDF regression requires camera pose and NYUV2 does not offer it, we do not test on NYUV2. However, we choose the currently best three self-supervised models [15, 45, 53] from NYUV2 leaderboard for comparison. Although trained on NYUV2, these models claim zero-shot transfer for indoor scenes by testing a subset of ScanNet in their original papers. We test them with the full testing split for comparison with our work. MonoNeRF [8] is our strongest baseline, which also uses the same training/testing splits of ScanNet as in our method. It is the latest self-supervised work to jointly train NeRF with a SFM model, and use NeRF to boost SFM performance. Our self-supervised voxel-SDF regression is also based on SFM, but MonoNeRF is still depth estimation while our MonoSelfRecon estimates continuous 3D mesh for a whole scene.

ScanNet. For depth evaluation, we follow the same standards of previous works [27, 36] to render depth from 3D mesh. Table 2 shows both rendered 2D depth and 3D mesh comparison. Our MonoSelfRecon outperforms all SOTA self-supervised depth estimation from NYUV2 leaderboard, especially the latest strongest baseline MonoNeRF. The 3D mesh comparison shows consistent results to 2D rendered depth. Since MonoNeRF has not released their code, we only compare it for 2D depth and could not generate their 3D mesh for comparison.

Although our MonoSelfRecon outperforms all SOTA generalizable self-supervised 3DR and is comparable to depth-based purely-supervised 3DR, there is still a clear gap to purely-supervised voxel-SDF methods using fine-grained SDF annotations, so we conduct experiments on weakly-supervised training to further demonstrate our design. While all supervised voxel-SDF methods [27, 35, 36] use voxel-SDF annotations for a L1 loss, NeuralRecon also uses a coarse binary voxel-mask annotations as weakly supervised signal, where the mask tells if each voxel-SDF is within truncation distance. In our purely self-supervised model setup, we filter out invalid voxels beyond truncation distance at each level of the SDF decoder in both training and testing, and only pass valid voxels to the next level. While for our weakly-supervised training, inspired by NeuralRecon, besides using our self-supervised losses, we add a mask regressor at each level and supervise with coarse mask ground truth, which is only used for training, and the mask regressor estimates valid voxels in testing. As Table 2 shows, our model improves obviously with coarse mask signal, and is even better than some fully-supervised works. More importantly, we also conduct weakly-supervised training by removing SDF ground truth and only using mask for original NeuralRecon, and we train both works to same epochs. Comparing weakly-supervised results, we clearly outperform NeuralRecon in weak supervision, which validates our purely self-supervised design.

7Scenes. We also test our models on 7Scenes dataset. To show our model’s generalization, as previous works [36] we do not train from scratch on 7Scenes, but to do few-shot transfer learning from our pre-trained models on ScanNet. Table 3 shows comparisons on both 3D mesh and 2D depth. We conduct three few-shot learnings: 1. Self-supervised transfer from our self-supervised pre-trained model on ScanNet without any annotations, denoted as “Ours(self)” in the table; 2. Weakly-supervised transfer from our weakly-supervised pre-trained model on ScanNet with coarse voxel-mask annotations, denoted as “Ours(weak)” in the table; 3. Weakly-supervised transfer from the original NeuralRecon pre-trained in weak-supervision on ScanNet with coarse voxel-mask annotations, denoted as “Neurcon_weak”. Other results in the table are directly from [36].

The few-shot transfer is done in a few minutes with

2D Depth	Train	$\delta 1 \uparrow$	AbsRel \downarrow	SqRel \downarrow	RMSE \downarrow	Comp \uparrow
DeMoN[39]	sup	31.9	0.389	0.420	0.855	-
MVSNet[49]	sup	64.1	0.234	0.190	0.508	-
NeuralRGBD[19]	sup	69.3	0.176	0.112	0.440	-
MVDNet[42]	sup	71.8	0.193	0.235	0.459	-
DPSNet[5]	sup	70.9	0.199	0.142	0.438	-
DeepV2D[38]	sup	42.8	0.437	0.553	0.869	-
CNMNet[22]	sup	76.6	0.161	0.083	0.361	-
NeuralRecon[36]	sup	82.0	0.155	0.104	0.346	-
Neucon_weak[36]	weak	78.2	0.163	0.101	0.303	0.626
Ours(weak)	weak	68.0	0.209	0.148	0.474	0.634
Ours(self)	self	63.3	0.219	0.155	0.519	0.818
3D Mesh	Train	F-score \uparrow	Recall \uparrow	Prec \uparrow	Comp \downarrow	Acc \downarrow
DeepV2D[38]	sup	0.115	0.175	0.087	0.180	0.518
CNMNet[22]	sup	0.149	0.246	0.111	0.150	0.398
NeuralRecon[36]	sup	0.282	0.227	0.389	0.228	0.100
Neucon_weak[36]	weak	0.101	0.062	0.356	0.103	0.473
Ours(weak)	weak	0.135	0.085	0.369	0.417	0.152
Ours(self)	self	0.146	0.101	0.274	0.323	0.215

Table 3. **2D depth and 3D mesh metrics on 7Scenes.** “Ours(weak)” and “Ours(self)” denote ours pre-trained in weak-supervision and self-supervision on ScanNet. “Neucon_weak” denotes NeuralRecon pre-trained in weak-supervision on ScanNet.

few frames from 7Scenes training splits. Results show that our purely self-supervised transfer is comparable and even outperforms to fully-supervised methods like DeepV2D [38] and CNMNet [22]. More importantly, in 3D mesh results, “Ours(weak)” is better than “Neucon_weak” and “Ours(self)” is better than “Ours(weak)”, which indicates that results are better if less coarse mask annotations and more our self-supervised signals are used. The reason is that 7Scenes’ scenes are more localized and contain more details than ScanNet. While the voxel-mask annotation successfully reflects coarse representations in ScanNet, it fails to catch the scene details in 7Scenes. This is also proved from 2D depth results that although weakly supervision with mask gets better “ $\delta 1$ ”, the pure self-supervision has the highest “complete”, which is because the mask annotations are too coarse to guide training and lead to an extremely low recall. These results indicate that our self-supervised losses are more reliable than supervised signals when annotations are coarse, this is significantly useful when the model is generalized to a new domain with limited and coarse annotations.

4.2. Ablation Study

We conduct ablation study to show the importance of each part in our framework, where the results are all trained in self-supervision, and the difference is only in architecture. Table 4 shows our ablation study results of 3D mesh and depth on ScanNet. “naive” means no GRU/NeRF/Attention used in the framework and is purely trained with proposed self-supervised losses. GRU brings essential improvements to the framework, which shows the importance of consistency between each scene fragment, and not considered in depth-based methods. With MPI-NeRF, our result already outperforms the strongest base-

2D Depth	AbsRel \downarrow	AbsDiff \downarrow	SqRel \downarrow	RMSE \downarrow	$\delta 1 \uparrow$	Comp \uparrow
Ours(naive)	0.358	0.678	0.417	0.846	43.4	0.817
Ours(+GRU)	0.176	0.277	0.127	0.387	72.8	0.786
Ours(+GRU+NeRF)	0.163	0.253	0.103	0.348	76.3	0.806
Ours(+GRU+Attention)	0.143	0.231	0.090	0.333	79.2	0.872
3D Mesh	Comp \downarrow	Acc \downarrow	Recall \uparrow	Prec \uparrow	F-score \uparrow	
Ours(naive)	0.278	0.206	0.198	0.155	0.171	
Ours(GRU)	0.212	0.226	0.247	0.197	0.217	
Ours(GRU+NeRF)	0.230	0.204	0.248	0.225	0.233	
Ours(GRU+Attention)	0.212	0.185	0.262	0.263	0.260	

Table 4. **Ablation Study.** “naive” denotes no GRU/NeRF/Attention used in the framework and is purely trained with proposed self-supervised losses.

line MonoNeRF [8] which uses similar NeRF as Table 2 shown, and the Attentional Fusion boosts further improvements. We do not jointly train with NeRF and Attentional Fusion for memory efficiency, and the results is sufficient to show their contributions to the framework.

4.3. Visual Results

Figure 3 shows 3D mesh (upper part) and rendered 2D depth (lower part) visual comparisons, which are compatible with numerical results. Clearly, our result outperforms SOTA indoor self-supervised depth estimation including Distdepth [45], P2Net [53] and Structdepth [15], and is even better than supervised work NeuralRGBD [19]. For purely self-supervision, only Structdepth can reconstruct some details, but ours can show most of the scene/object details, like the bed, chairs and nightstand in the 1st mesh scene, and the stairs in the 2nd mesh scene. With coarse mask ground truth for weakly supervision, our mesh surface becomes much smoother and does not have much difference to the strongest supervised NeuralRecon trained with fine-grained SDF annotations. The 2D rendered depths at bottom are more straightforward to show details. In the 1st scene, our self-supervised estimation clearly shows the golden chair and the bed, while other self-supervised methods can barely show the contours of the bed. The 2nd scene is more challenging but we are still able to reconstruct fine details such as the farthest chair and the walls at back.

5. Conclusion

We propose a novel framework - **MonoSelfRecon** that for the first time achieves **explicit 3D mesh** reconstruction for **generalizable** indoor scenes with monocular RGB views by purely **self-supervision** on voxel-SDF. We propose novel self-supervised losses corresponding to our novel framework design, and test with thorough experiments on ScanNet and 7Scenes. Our method outperforms the best indoor self-supervised methods so far and is comparable to fully-supervised works. With few-shot learning, it is easily transferred to other domains. The framework is not limited to model design but extensive to any voxel-SDF models, which keeps the advantages of the original model.

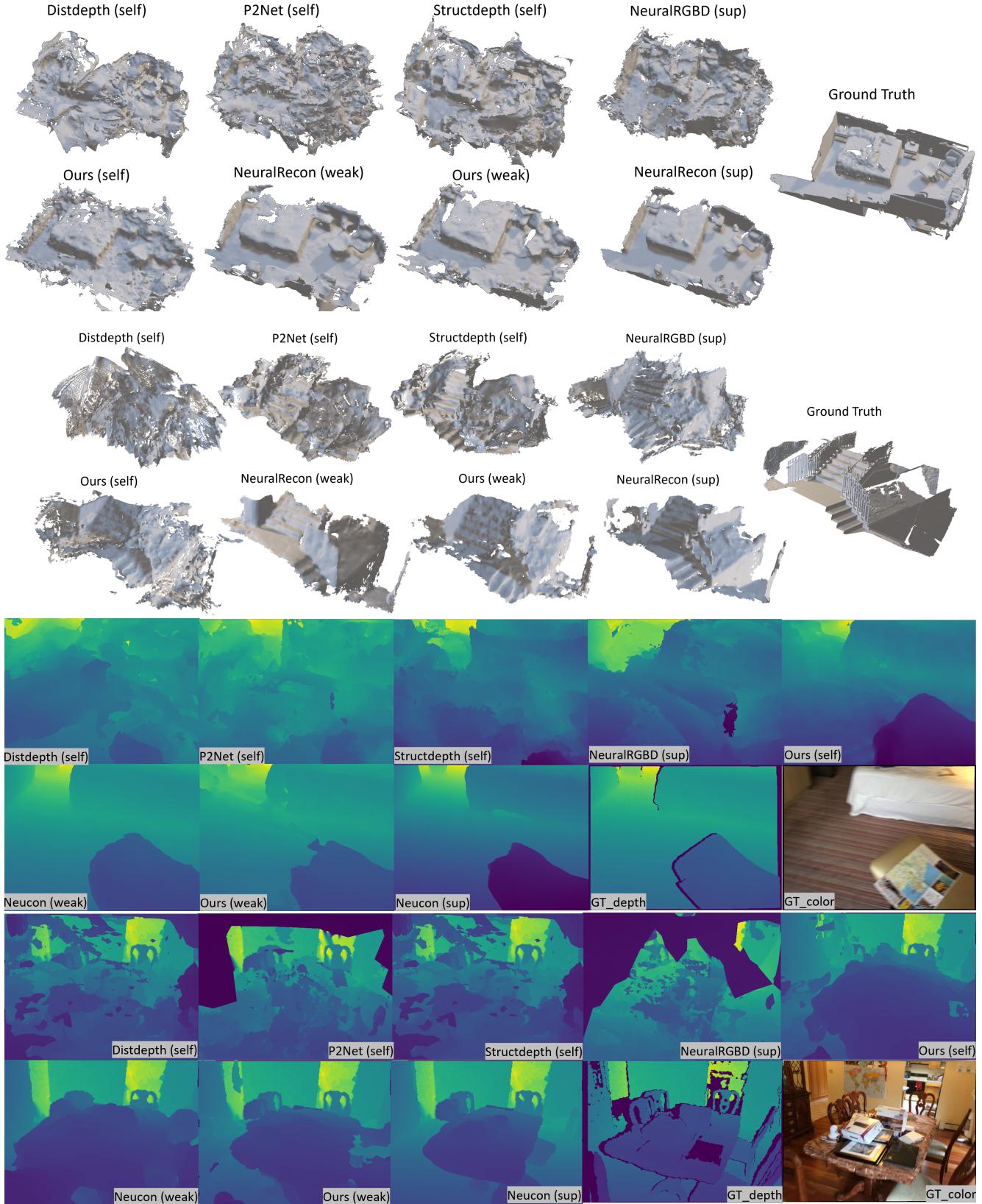


Figure 3. **Visual Results on ScanNet.** 3D meshes are shown at top, 2D rendered depth maps are shown at bottom. Our self-supervised results are clearly better than SOTA self-supervised methods on challenging cases and fuzzy RGB input, even better than supervised depth estimation for a few cases. With weak supervision, our result is clearly better than NeuralRecon with weak supervision, which demonstrates our self-supervised design.

References

- [1] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. MVSNeRF: Fast generalizable radiance field reconstruction from multi-view stereo. *arXiv preprint arXiv:2103.15595*, 2021. 2, 3
- [2] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 5
- [3] François Darmon, Bénédicte Bascle, Jean-Clément Devaux, Pascal Monasse, and Mathieu Aubry. Improving neural implicit surfaces geometry with patch warping. In *CVPR*, 2022. 3, 5
- [4] Arda Duzceker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15324–15333, 2021. 2
- [5] Bin Fan, Ke Wang, Yuchao Dai, and Mingyi He. RS-DPSNet: Deep plane sweep network for rolling shutter stereo images. *IEEE Signal Processing Letters*, 28:1550–1554, 2021. 2, 6, 7
- [6] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59:167–181, 2004. 4
- [7] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *CVPR*, 2022. 2
- [8] Yang Fu, Ishan Misra, and Xiaolong Wang. MonoNeRF: Learning generalizable NeRFs from monocular videos without camera poses. *arXiv preprint arXiv:2210.07181*, 2022. 4, 6, 7, 1, 2
- [9] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. 2019. 2, 6
- [10] Haoyu Guo, Sida Peng, Haotong Lin, Qianqian Wang, Guofeng Zhang, Hujun Bao, and Xiaowei Zhou. Neural 3D scene reconstruction with the manhattan-world assumption. In *CVPR*, 2022. 1, 3
- [11] Yuxin Hou, Juho Kannala, and Arno Solin. Multi-view stereo by temporal nonparametric fusion. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2651–2660, 2019. 6
- [12] Yuxin Hou, Juho Kannala, and Arno Solin. Multi-view stereo by temporal nonparametric fusion. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2651–2660, 2019. 3
- [13] Yuxin Hou, Juho Kannala, and Arno Solin. Multi-view stereo by temporal nonparametric fusion. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2651–2660, 2019. 1
- [14] Pan Ji, Runze Li, Bir Bhanu, and Yi Xu. MonoIndoor: Towards good practice of self-supervised monocular depth estimation for indoor environments. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12767–12776, 2021. 2, 6
- [15] Boying Li, Yuan Huang, Zeyu Liu, Danping Zou, and Wenxian Yu. StructDepth: Leveraging the structural regularities for self-supervised indoor depth estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 2, 4, 6, 7
- [16] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine: Towards continuous depth MPI with NeRF for novel view synthesis. In *ICCV*, 2021. 1, 3, 4, 2
- [17] Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin. Neuralangelo: High-fidelity neural surface reconstruction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3, 5
- [18] Chen Liu, Jimei Yang, Duygu Ceylan, Ersin Yumer, and Yasutaka Furukawa. PlaneNet: Piece-wise planar reconstruction from a single rgb image. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2579–2588, 2018. 4
- [19] Chao Liu, Jinwei Gu, Kihwan Kim, Srinivasa G. Narasimhan, and Jan Kautz. Neural RGB®D sensing: Depth and uncertainty from a video camera. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10978–10987, 2019. 1, 2, 7
- [20] Chen Liu, Kihwan Kim, Jinwei Gu, Yasutaka Furukawa, and Jan Kautz. PlaneRCNN: 3d plane detection and reconstruction from a single image. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4445–4454, 2019. 4
- [21] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 2
- [22] Xiaoxiao Long, Lingjie Liu, Christian Theobalt, and Wenping Wang. Occlusion-aware depth estimation with adaptive normal constraints. *ECCV*, 2020. 2, 7
- [23] Xiaoxiao Long, Lingjie Liu, Wei Li, Christian Theobalt, and Wenping Wang. Multi-view depth estimation using epipolar spatio-temporal networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8258–8267, 2021. 2
- [24] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, page 163–169, 1987. 1, 3
- [25] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1, 2
- [26] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, 2022. 3
- [27] Zak Murez, Tarrence van As, James Bartolozzi, Ayan Sinha, Vijay Badrinarayanan, and Andrew Rabinovich. Atlas: End-

- to-end 3d scene reconstruction from posed images. In *ECCV*, 2020. 1, 2, 5, 6
- [28] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, 2011. 1
- [29] Michael Oechsle, Songyou Peng, and Andreas Geiger. UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In *International Conference on Computer Vision (ICCV)*, 2021. 3
- [30] Joseph Ortiz, Alexander Clegg, Jing Dong, Edgar Sucar, David Novotny, Michael Zollhoefer, and Mustafa Mukadam. isdf: Real-time neural signed distance fields for robot perception. In *Robotics: Science and Systems*, 2022. 1, 3
- [31] Thomas Porter and Tom Duff. Compositing digital images. In *SIGGRAPH 1984*, 1984. 2
- [32] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 6
- [33] Thomas Schöps, Torsten Sattler, Christian Häne, and Marc Pollefeys. 3D modeling on the go: Interactive 3D reconstruction of large-scale scenes on mobile devices. In *2015 International Conference on 3D Vision*, pages 291–299, 2015. 1
- [34] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgbd images. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2013. 5
- [35] Noah Stier, Alexander Rich, Pradeep Sen, and Tobias Höllerer. VoRTX: Volumetric 3d reconstruction with transformers for voxelwise view selection and fusion. In *2021 International Conference on 3D Vision (3DV)*, pages 320–330. IEEE, 2021. 1, 2, 3, 6
- [36] Jiameng Sun, Yiming Xie, Linghao Chen, Xiaowei Zhou, and Hujun Bao. NeuralRecon: Real-time coherent 3D reconstruction from monocular video. *CVPR*, 2021. 1, 2, 3, 5, 6, 7
- [37] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8248–8258, 2022. 3
- [38] Zachary Teed and Jia Deng. DeepV2D: Video to depth with differentiable structure from motion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, 2020. 2, 7
- [39] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. DeMoN: Depth and motion network for learning monocular stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 7
- [40] Julien Valentin, Adarsh Kowdle, Jonathan T. Barron, Neal Wadhwa, Max Dzitsiuk, Michael Schoenberg, Vivek Verma, Ambrus Csaszar, Eric Turner, Ivan Dryanovski, Joao Afonso, Jose Pascoal, Konstantine Tsotsos, Mira Leung, Mirko Schmidt, Onur Guleryuz, Sameh Khamis, Vladimir Tankovich, Sean Fanello, Shahram Izadi, and Christoph Rhemann. Depth from motion for smartphone AR. *ACM Trans. Graph.*, 2018. 1
- [41] Jiepeng Wang, Peng Wang, Xiaoxiao Long, Christian Theobalt, Taku Komura, Lingjie Liu, and Wenping Wang. NeuRIS: Neural reconstruction of indoor scenes using normal priors. 2022. 1, 3, 5
- [42] K. Wang and S. Shen. Mvdepthnet: real-time multiview depth estimation neural network. In *International Conference on 3D Vision (3DV)*, 2018. 1, 2, 6, 7
- [43] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 3
- [44] Qianqian Wang, Zhicheng Wang, Kyle Genova, Pratul Srinivasan, Howard Zhou, Jonathan T. Barron, Ricardo Martin-Brualla, Noah Snavely, and Thomas Funkhouser. IBRNet: Learning multi-view image-based rendering. In *CVPR*, 2021. 3
- [45] Cho-Ying Wu, Jialiang Wang, Michael Hall, Ulrich Neumann, and ShuoChen Su. Toward practical monocular indoor depth estimation. In *CVPR*, 2022. 6, 7
- [46] Hongyi Xu, Thiendo Alldieck, and Cristian Sminchisescu. H-NeRF: Neural radiance fields for rendering and temporal reconstruction of humans in motion. In *Neural Information Processing Systems*, 2021. 1, 3
- [47] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-NeRF: Point-based neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5438–5448, 2022. 3
- [48] Zhenfei Yang, Fei Gao, and Shaojie Shen. Real-time monocular dense mapping on aerial robots using visual-inertial fusion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4552–4559, 2017. 1
- [49] Yao Yao, Zixin Luo, Shiwei Li, Tianwei Shen, Tian Fang, and Long Quan. Recurrent MVSNet for high-resolution multi-view stereo depth inference. *Computer Vision and Pattern Recognition (CVPR)*, 2019. 7
- [50] Lior Yariv, Jitao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 1, 3
- [51] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 1, 3
- [52] Zehao Yu, Jia Zheng, Dongze Lian, Zihan Zhou, and Shenghua Gao. Single-image piece-wise planar 3d reconstruction via associative embedding. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1029–1037, 2019. 4
- [53] Zehao Yu, Lei Jin, and Shenghua Gao. P²Net: Patch-match and plane-regularization for unsupervised indoor depth estimation. In *ECCV*, 2020. 2, 4, 6, 7

- [54] Zehao Yu, Songyou Peng, Michael Niemeyer, Torsten Sattler, and Andreas Geiger. MonoSDF: Exploring monocular geometric cues for neural implicit surface reconstruction. *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. [1](#), [3](#)
- [55] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. NeRFusion: Fusing radiance fields for large-scale scene reconstruction. *CVPR*, 2022. [3](#)
- [56] Junsheng Zhou, Yuwang Wang, Kaihuai Qin, and Wenjun Zeng. Moving indoor: Unsupervised video depth learning in challenging environments. In *Inter. Conf. on Computer Vision*. IEEE, IEEE, 2019. [2](#), [6](#)
- [57] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. [4](#), [2](#)

MonoSelfRecon: Purely Self-Supervised Explicit Generalizable 3D Reconstruction of Indoor Scenes from Monocular RGB Views

Supplementary Material

6. Relationship with MonoNeRF [8]

We discuss the relationship between our strongest baseline MonoNeRF[8] and our proposed method MonoSelfRecon as follows: 1) We share the same idea of SFM-based 3DR with monocular RGB sequence as input, and we both jointly train SFM and a generalizable NeRF, where the NeRF is used to boost SFM performance. 2) Although using SFM as the core of framework design, we regress to different 3D representations, where MonoNeRF regresses to view-based 2D depth map while we regress to 3D voxel-based SDF values. 3) While MonoNeRF also jointly estimates camera poses, their 2D view-based depth representation restricts the ability to incrementally complete a whole scene in 3D mesh representation. Fusing TSDF from direct depth estimation is time-consuming, and will cause layered or sparse mesh due to depth inconsistency between each frame. By comparison, our direct voxel-SDF regression enables us to incrementally add the previous mesh to complete the whole scene consistently in mesh representation.

The mesh representation is a stricter 3D representation over 2D depth map. Theoretically, the depth map can be perfectly rendered from 3D mesh but cannot in reverse, which is further validated by our experiments. Table 2 and 3 show that although all using ground truth for supervised training, the one that directly regresses SDF (NeuralRecon) has a clear advantage on 2D depth metrics over other supervised methods that regresses depth. The reason that although both our method and MonoNeRF are based on SFM while ours outperforms theirs can be also partly attributed to this different 3D representation. Our visual results also reflect this point in Table 3, where although there is no much difference of 2D depth, the difference of 3D mesh is clear. In other words, the depth representation is more visually straightforward than 3D mesh. Consequently, our 3D mesh regressing is a stricter 3D geometric representation than MonoNeRF’s 2D depth.

7. Evaluation Metrics

We follow the same evaluation metrics as [27, 36]. Details of the metrics are summarized in Table 5.

8. Model Details

8.1. Attentional View Fusion

We use a standard Vision Transformer (ViT) Encoder, where we keep the original high-level architecture of the

2D	3D
Abs Rel	$\frac{1}{n} \sum d - d^* / d^*$
Abs Diff	$\frac{1}{n} \sum d - d^* $
Sq Rel	$\frac{1}{n} \sum d - d^* ^2 / d^*$
RMSE	$\sqrt{\frac{1}{n} \sum d - d^* ^2}$
$\sigma < 1.25$	$\frac{1}{n} \sum \left(\max\left(\frac{d}{d^*}, \frac{d^*}{d}\right) < 1.25 \right)$
Comp	% valid predictions
RMSE log	$\sqrt{\frac{1}{n} \sum \log(d) - \log(d^*) ^2}$
Sc Inv	$\left(\frac{1}{n} \sum_i z_i^2 - \frac{1}{n^2} (\sum_i z_i)^2 \right)^{1/2}$
Acc	$\text{mean}_{p \in P} \left(\min_{p^* \in P^*} p - p^* \right)$
Comp	$\text{mean}_{p^* \in P^*} \left(\min_{p \in P} p - p^* \right)$
Prec	$\text{mean}_{p \in P} \left(\min_{p^* \in P^*} p - p^* < .05 \right)$
Recal	$\text{mean}_{p^* \in P^*} \left(\min_{p \in P} p - p^* < .05 \right)$
F-score	$\frac{2 \times \text{Prec} \times \text{Recal}}{\text{Prec} + \text{Recal}}$

Table 5. Evaluation Metrics.

ViT encoder to be: A norm layer, a multi-head attention layer, a norm layer, and a MLP (2 heads are used). Originally the ViT takes image patch/features as input, while we adopted the input to be the nearest 2D features from the projected 3D voxels, which is of size $[N_{view}, N_{points}, C]$, where N_{view} is the number of views in a scene fragment, N_{points} is the number of voxels in a fragment, C is the feature channel. The input also takes the voxel mask as input to filter out the pixels which are invisible to the voxels, and the transformer only takes the visible pixel features. We stack two ViT encoders to update the features, where the output is still of size $[N_{view}, N_{points}, C]$. Then we use a multi-view weighted feature pooling to fuse the updated features at the view channel to 3D features of size $[N_{points}, C]$, where the weight is the number of visible views in a fragment for each voxel. Such design enables more flexibility to adjust the contribution of each view to the 3D voxels.

8.2. GRU

We directly use the GRU module from [36], which is elaborately designed for sparse 3D convolution. The 3D voxel features are obtained by attentional view fusions and fed to the GRU module, where the current 3D fragment features are conditioned on the previous fragment. Using the current 3D global voxel features G_t^l and the previous hidden state H_{t-1}^l at layer l , the current hidden state H_t^l can be obtained, and the SDF value at each level is regressed from the hidden state H_t^l . More specifically,

$$\begin{aligned}
 z_t &= \sigma(\text{SparseConv}([H_{t-1}^l, G_t^l], W_z)) \\
 r_t &= \sigma(\text{SparseConv}([H_{t-1}^l, G_t^l], W_r)) \\
 \tilde{H}_t^l &= \tanh(\text{SparseConv}([r_t \odot H_{t-1}^l, G_t^l], W_h)) \\
 H_t^l &= (1 - z_t) \odot H_{t-1}^l + z_t \odot \tilde{H}_t^l
 \end{aligned} \tag{12}$$

where z_t is the update gate, r_t is the reset gate, σ is the sigmoid function and W_* is the weight for sparse convolution.

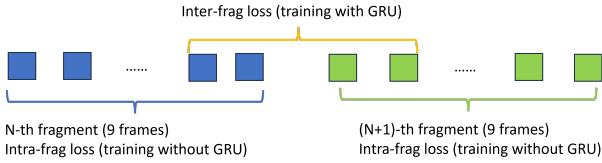


Figure 4. **Inter/Intra-fragment** losses illustration.

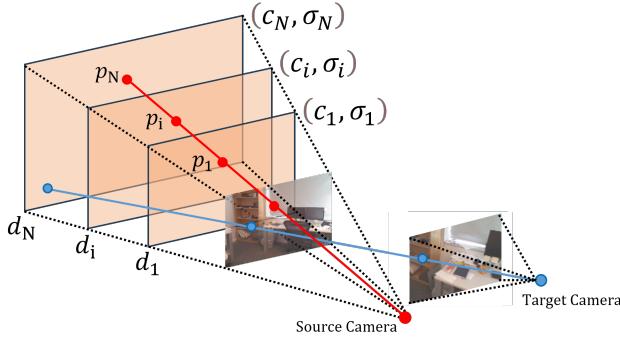


Figure 5. **Multi-Plane Image (MPI)** NeRF illustration.

We first train without GRU within each fragment to warmup the framework with our proposed self-supervised losses, where we call it **intra-fragment losses**. Because the GRU module is leveraged to enhance the consistency between fragments, the self-supervised learning strategy should be treated differently to intra-fragment losses. There is no need to change the training policy in purely supervised training because SDF ground truth is used, and there is no ground truth in our self-supervision except for the consistency clues between fragments. So we extend the **inter-fragment losses** to **intra-fragment losses**. While the model only takes input per fragment, backpropagating whole fragments brings memory challenges, so we only implement the inter-fragment losses on the frames around the boundary of fragments. Specifically, we use the last 4 and first 4 frames of the previous and current fragments to implement the inter-fragment loss.

8.3. NeRF

Since the SDF decoder is generalizable, the NeRF also must be generalizable to boost SDF decoder. For our work, we adopted MPI-NeRF[16, 57], which has been directly used by MonoNeRF[8] and proved to be generalizable. As Figure 5 shows, in Multi-Plane-Image (MPI) system, an image is represented by a set of parallel planes (orange planes) denoted as RGB- σ , specifically $(c_i, \sigma_i)_{i=1}^N$, where the i_{th} plane has d_i disparity (reverse of depth) to the camera. The shading points (red points) are selected as the intersection of the parallel planes and the rays shooting from pixels in the image, where c_i and σ_i are the RGB color and density

of each shading points at i_{th} plane. In a standard MPI system, the source view RGB image \hat{I}_s and depth map \hat{D} can be composed using the “over” operation [31] as

$$\begin{aligned}\hat{I}_s &= \sum_{i=1}^D (c_i \sigma_i \prod_{j=i+1}^D (1 - \sigma_j)) \\ \hat{D}_s &= \sum_{i=1}^D (d_i^{-1} \sigma_i \prod_{j=i+1}^D (1 - \sigma_j))\end{aligned}\quad (13)$$

To use MPI system in NeRF style, the composition operation above can be replaced by volumetric rendering [25] for both RGB and depth as

$$\begin{aligned}\hat{I}_s &= \sum_{i=1}^N T_i (1 - \exp(-\sigma_i \delta_i)) c_i \\ \hat{Z}_s &= \sum_{i=1}^N T_i (1 - \exp(-\sigma_i \delta_i)) z_i\end{aligned}\quad (14)$$

where z_i is the rendered depth (reverse of disparity) $z_i = 1/d_i$, and $\delta_i = \|p_{i+1} - p_i\|_2$ is the distance between the two neighbor shading points on a ray. Then we can extend volumetric rendering to target views. First, the corresponding pixels $[u_t, v_t]$ in the target view can be found by

$$\begin{bmatrix} u_s \\ v_s \\ 1 \end{bmatrix} \sim K_s (R - tn^T d_i) (K_t)^{-1} \begin{bmatrix} u_t \\ v_t \\ 1 \end{bmatrix} \quad (15)$$

Here, $[u_s, v_s]$ is the corresponding pixel locations in the source view, K_s and K_t are camera intrinsics of source and target views, R and t are rotation and translation from the target to source view, and n is the norm vector of the i_{th} plane. As the planes are parallel, the RGB c'_i and density σ'_i of shading points (blue points) on target rays (blue ray) are equal to those from source rays at the same disparity, as shown in Eq. 16,

$$\begin{aligned}c'_i(u_t, v_t) &= c_i(u_s, v_s) \\ \sigma'_i(u_t, v_t) &= \sigma_i(u_s, v_s)\end{aligned}\quad (16)$$

Once we have RGB and density for target views, we can perform volumetric rendering on target views as:

$$\begin{aligned}\hat{I}_t &= \sum_{i=1}^N T_i (1 - \exp(-\sigma'_i \delta_i)) c'_i \\ \hat{Z}_t &= \sum_{i=1}^N T_i (1 - \exp(-\sigma'_i \delta_i)) z'_i\end{aligned}\quad (17)$$

We use standard NeRF RGB loss, where \hat{I}_s and \hat{I}_t are self-supervised with their corresponding input images with a L1 loss. Since we directly use the reverse of disparity for depth, the depth value is scale-ambiguous. As mentioned

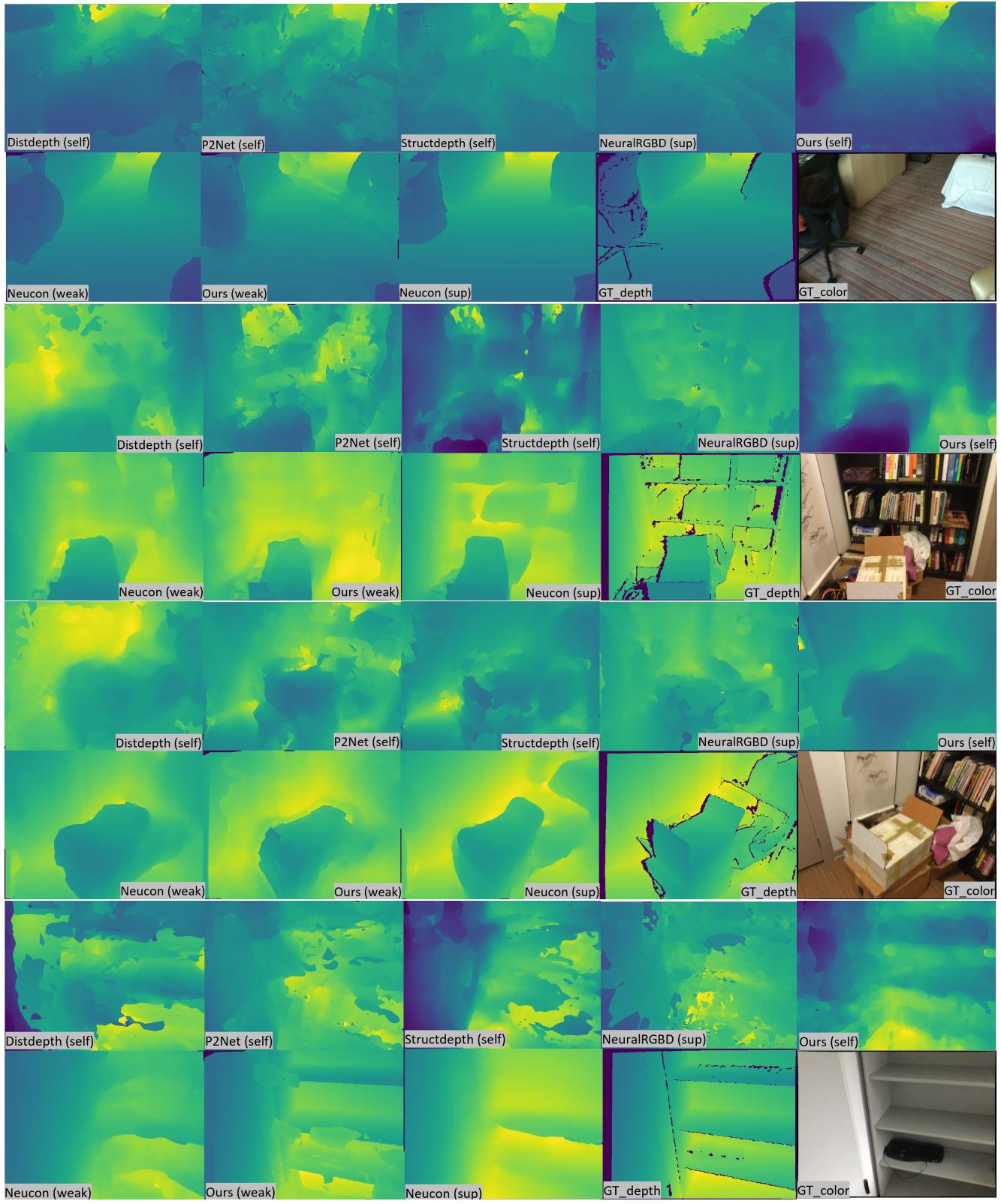
in the paper, since there is no depth ground truth for pure self-supervision, we use SDF-depth as pseudo-depth to first recover the real scale of \hat{Z}_s and \hat{Z}_t , then we impose a consistency loss between \hat{Z} and SDF-depth to boost SDF decoder.

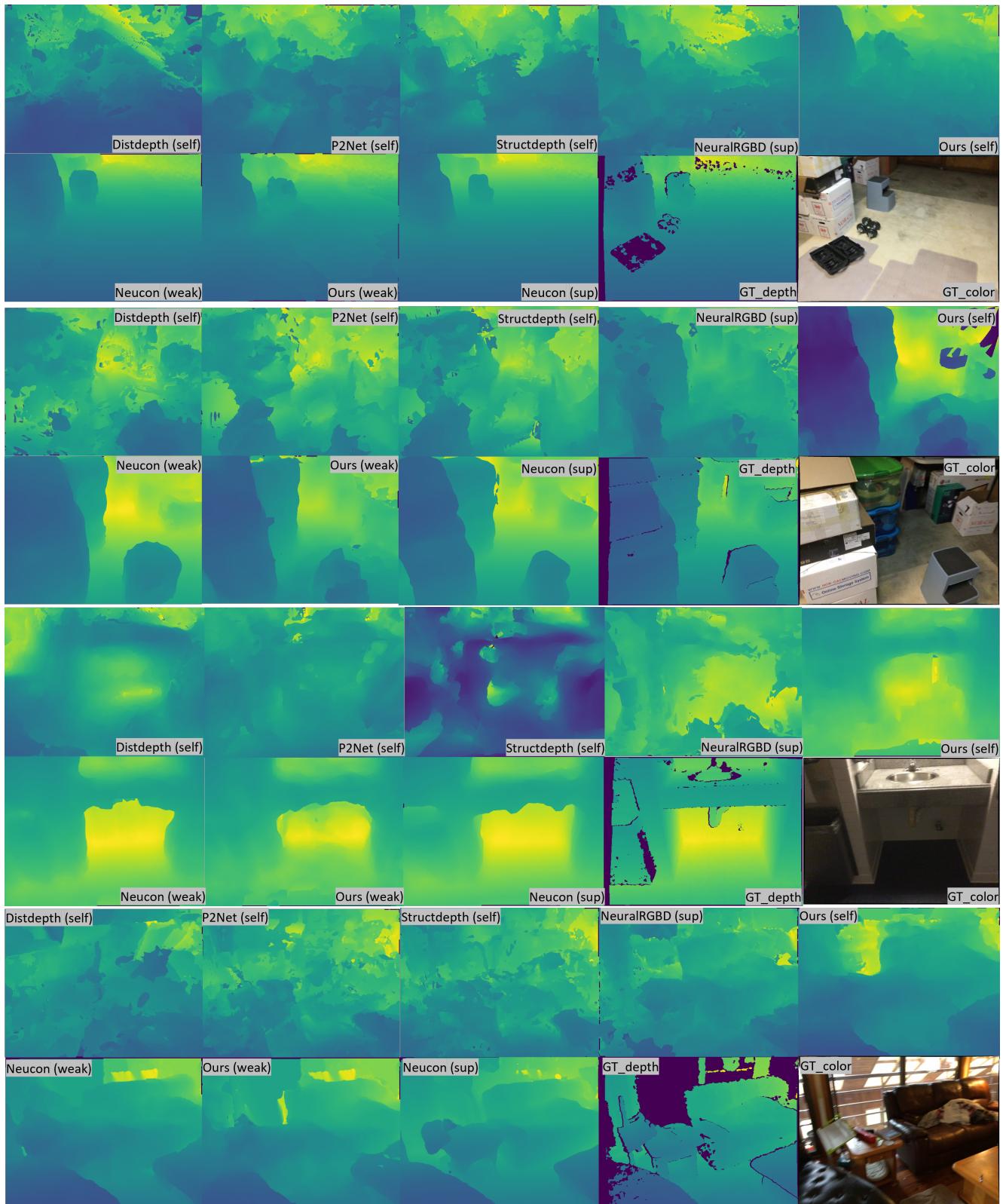
9. Visual Results

We show more visual results of 2D rendered depth and 3D mesh in Figure ?? . We also attach a PowerPoint file with visual results, where reviewers can rotate and zoom the 3D mesh to see the details,

10. Limitation

Although our work combines the advantages of “self-supervised” “generalizable” and “3D explicit mesh” altogether, there are still limitations. So far our MonoSelfRecon can be only used for indoor environments, because we pre-define the 3D scene fragment with a fixed voxel number. Unlike indoor 2D images where depth vary within few meters, the depth can vary significantly just within a single 2D image in outdoor. It is applicable to keep the voxel number while increasing the voxel size, but it will lead to very poor resolution within voxels, which misses most of the details. Moreover, since we regress SDF corresponding to the discrete $N \times N \times N$ voxels of scene fragment, we cannot directly estimate SDF of a continuous 3D space, unless by interpolation. By contrast, SDF-NeRF based methods estimate SDF values in continuous 3D space but it is not generalizable to another scene. Our future works will explore to make SDF-NeRF generalizable, so that the 3DR can be “self-supervised”, “generalizable”, “explicit”, “indoor/outdoor”, and “continous in 3D space”.





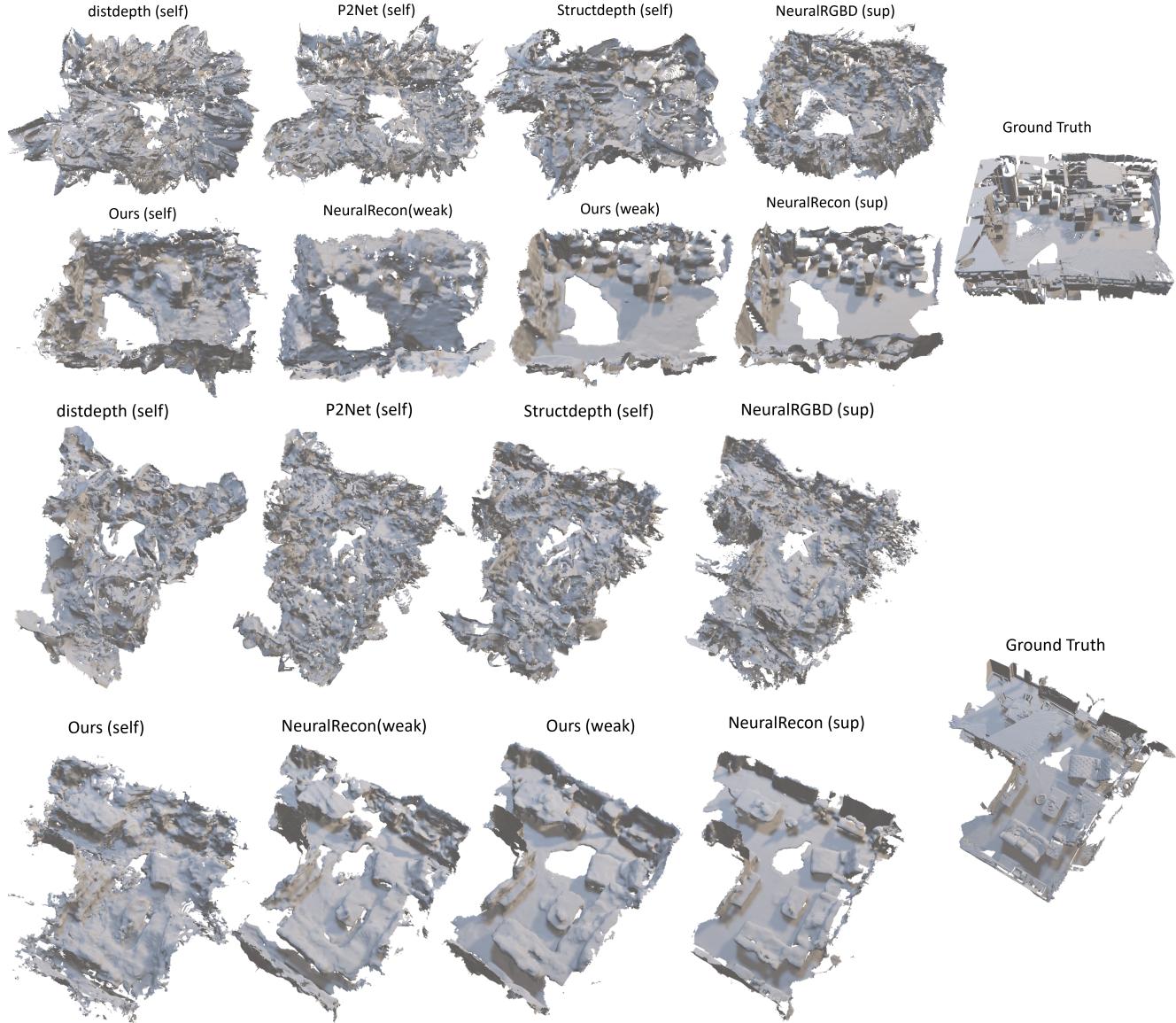


Figure 6. Visual Results