AGS-Mesh: Adaptive Gaussian Splatting and Meshing with Geometric Priors for Indoor Room Reconstruction Using Smartphones

Supplementary Material

In this supplementary material, we provide additional details regarding our AGS-Mesh optimization and the proposed Adaptive TSDF and IsoOctree meshing strategy in Appendix A. We also give further details about the meshing strategies in Appendix B. Lastly, we present qualitative renders for mesh reconstruction and novel-view synthesis in Appendix D.

A. Implementation Details

A.1. AGS-Mesh Optimization

We implement our regularization terms on top of the open-source implementations from 2DGS [19] and DN-Splatter [43]. We enable our DNR and ANR optimization terms at training iterations $T_d = 7$k (*cf*. Eq. (6)) and $T_n = 15$k (*cf*. Eq. (7)), respectively. We enable the filtered geometry prior after a certain number of steps to allow the Gaussians to be fully supervised during the initial phase and to relax the training process in later stages. We set the angle thresholds τ_d and τ_N used for filtering inconsistent depths and normals (refer to *cf*. Eq. (4) and *cf*. Eq. (8)) to $10°$. We immediately apply depth supervision at the beginning of training and enable normal regularization only after 7k iterations. The total number of training iterations is 30k. In the final optimization loss, we set λ_d to 0.2 and λ_n to 0.1. We use the estimated normals from Omnidata [14] as pre-trained normals, as they have shown to improve 3D reconstruction performance in our experiments.

A.2. Adaptive TSDF and IsoOctree Details

Our proposed meshing strategy consists of constructing an isofunctional inspired by TSDF approaches that is then meshed using an octree-based Marching Cubes algorithm IsoOctree.

The meshing stage takes in depth and normal renders from the Gaussian scene and camera poses. Input depths are first filtered based on a threshold that determines nearby depth similarity, if nearby depth values differ by a margin, they are filtered out. This effectively removes object edges from the depth maps and allows using linear interpolation on the remaining valid pixels. The motivation is that depth maps on the object edges are typically inaccurate and may represent a random intermediate depth value between the foreground object and the background. The normal maps are also filtered using the same mask.

We define the isofunction as

$$f(x) = \sum_j w_j(d_j(x) - \tilde{d}_j(x)), \tag{11}$$

where $d_j(x)$ is the value of the depth map j at the projection of point x and $\tilde{d}_j(x) = (x - p_j) \cdot c_z$ is the actual depth of x. Here p_j, c_z are the center and principal axis of camera j, respectively. The sum is taken over the values where the depth map is valid and the TSDF value $d_j(x) - \hat{d}_j(x)$ exceeds a lower truncation distance $-\tau \cdot d_j(x)$, which depends on the projected depth. We use $\tau = 0.05$ as the relative truncation distance. The weight in the formula is computed using a two-pass approach where we first compute a *maximum weight normal* $n(x) = n_k$, $k = \arg\max_j w'_j(x)$ where

$$w'_j = \frac{(d_j(x) - \tilde{d}_j(x)) \cdot (-r_j(x) \cdot n_j(x))}{d_j(x)^2} \tag{12}$$

and, on the second pass, compare the normal map value n_j and camera ray direction $r_j(x) = \frac{x - p_j}{|x - p_j|}$ to n' when computing the final weight w_j. The factor $d_j(x)^2$ in the denominator effectively down-weights observations with a larger distance to the camera, where the uncertainty of the depth map is also assumed to be the largest.

The isofunction defined above is then meshed using an IsoOctree [24] approach. We utilize the backprojected point cloud constructed from rendered depth maps as a *point cloud hint*. The point cloud hint serves as a subdivision criteria for IsoOctree. A uniform grid is first initialized based on an AABB enclosing the point cloud hint. If a voxel contains points above a user threshold (set to 50), the voxel is subdivided into an octant. This creates a three-dimensional octree subdivision structure that contains finer levels of detail at deeper octree depths. We set the maximum octree depth to 10.

B. Mesh Extraction Methods

In this section, we provide further details on the meshing strategies shown in Table 5 and Table 1.

TSDF. The Truncated Signed Distance Function (TSDF) method refers to the ScalableTSDFVolume [61] implementation from Open3D [62]. The method accepts depth, RGB, and camera poses as input, identifies points of interest, and calculates a TSDF from input values to extract a mesh using Marching Cubes [28]. We set the depth truncation distance to 10, the voxel size to 0.01, and the SDF truncation distance to 0.03 for all TSDF marked baselines in Tab. 5 and Tab. 1.

Table 5. **Mesh reconstruction evaluation on ScanNet++**. The mesh metrics are averaged over the "b20a261fdf" and "8b5caf3398" scenes. The best results from each category are marked with **bold**. Time represents training time.

	Methods		Sensor Depth	Meshing Algorithm	Accuracy ↓	Completion ↓	Chamfer-L_1 ↓	Normal Consistency ↑	F-score ↑	Time (min)
	Volumetric Fusion [9]		✓	TSDF	.0335	.0429	.0382	.7372	.8526	0.17
Implicit	Nerfacto [41]	NeRF	–	Poisson	.1305	.1484	.1394	.7153	.4698	8.0
Implicit	Depth-Nerfacto [41]		✓	Poisson	.0731	.1647	.1189	.6848	.5018	8.1
Implicit	MonoSDF [59]	SDF	✓	Marching-Cubes	**.0303**	**.0573**	**.0438**	**.8881**	**.8577**	47.5
Explicit	3DGS [26]	3DGS	–	TSDF	.1795	.1716	.1756	.6578	.1719	14.5
Explicit	SuGaR [16]		–	Poisson + IBR	**.0940**	.1011	**.0975**	**.7241**	**.4367**	70
Explicit	GOF [60]		–	Tetrahedral	.1398	**.0976**	.1187	.6998	.3239	142
Explicit	Splatfacto [41]	Splatfacto	–	Poisson	.1934	.1503	.1719	.6741	.1790	8.9
Explicit	DN-Splatter [43]		✓	Poisson	**.0940**	.0395	.0667	.8316	.7658	36.9
Explicit	DN-Splatter [43]		✓	TSDF	.1069	.0251	.0660	**.8539**	.8296	36.9
Explicit	Splatfacto [41] + Ours		✓	TSDF	.1060	**.0251**	**.0655**	.8506	**.8314**	36.9
Explicit	2DGS [19]	2DGS	–	TSDF	.1272	.0798	.1035	.7799	.4196	33.5
Explicit	2DGS [19] + Ours		✓	TSDF	**.0264**	.0305	.0285	.9097	**.9030**	40.4
Explicit	2DGS [19] + Ours		✓	SDF + IsoOctree (Ours)	.0269	**.0282**	**.0276**	**.9139**	.9028	40.4

Poisson. Poisson refers to the screened variant of Poisson Reconstruction [22] used to extract a mesh from an oriented point cloud. Optimized depth and normal maps are back-projected into world coordinates to obtain oriented points. Poisson surface reconstruction is sensitive to perturbations in the oriented point cloud; therefore, noise and multi-view inconsistencies in depth maps and backprojection can lead to poor surface generation.

Poisson + IBR. Poisson + IBR (Image Based Rendering) refers to the optimization strategy proposed in SuGaR [16]. A coarse mesh is first obtained from the Gaussian scene at 7k iterations by Poisson reconstruction from a point cloud sampled from a level set determined by the density of the Gaussian scene. The coarse mesh is then further optimized with differentiable image-based rendering (using PyTorch3D functionality) for 15k iterations to produce a refined mesh. Mesh metrics are evaluated on this refined mesh.

Tetrahedral. GOF [60] proposed generating a 3D bounding box for each Gaussian, then establishing tetrahedral grids within these 3D bounding boxes. Marching Tetrahedra [40] is applied to extract triangle meshes from the tetrahedral grid, using a binary search algorithm to precisely identify the level set.

SDF + IsoOctree. The SDF + IsoOctree method, proposed in our paper, utilizes a depth-aware truncated TSDF calculation combined with the IsoOctree meshing method. The approach can reduce the number of mesh vertices, for example, the size of the mesh extracted with TSDF is 192MB and the mesh extracted with SDF + IsoOctree is 30MB for the "vr_room" from MuSHRoom dataset.

C. Explanations of Benchmark selection

We choose Splatfacto as the representative of 3DGS-based baselines as it is an advanced version of 3DGS and well-suited for indoor room reconstruction. Additionally, we implement our method on 2DGS to demonstrate its effectiveness. Although methods such as [10, 47, 48] achieve high-quality object reconstructions, they face significant challenges in indoor room reconstruction due to their high computational requirements [10] and suboptimal feature extraction performance [47, 48].

D. More Experiments

D.1. Quantitative 3D Reconstruction Evaluation on ScanNet++

We show mesh comparison quantitative results on ScanNet++ in Table 5. Our method provides an overall improvement when added to baselines.

D.2. Visualizations of DNC and ANR

We visualize the output depth and normal maps produced by the DNC and ANR filtering terms in Fig. 8. We observe that the DNC and ANR terms successfully filter our unreliable edges and outlier depth and normal estimates, preventing them from misleading the Gaussian training process.

D.3. Qualitative Comparision of 3D Reconstruction

Similarly, we show additional qualitative comparisons of 3D mesh reconstruction quality on the ScanNet++ dataset in Fig. 7. Our method presents a notable improvement in smoothing flat surfaces on the extracted mesh.

D.4. Qualitative Comparision of Novel View Synthesis

Lastly, we compare the quality of novel view synthesis with our method along with error visualizations in Fig. 9. We compare 2DGS with and without our DNC and ANR regularization terms with highlighted details and l_2 differences. We demonstrate that regularization with more accurate geometric priors not only helps mesh reconstruction, but also aids in novel view rendering, especially for removing floaters.

E. Limitations and future work

Our method targets 3D reconstruction using RGB sequences with sensor depth. In future work, the method could be

extended to only use RGB images. The IsoOctree meshing technique we propose focuses on reducing the number of vertices and faces in the mesh while smoothing the surface. However, it does not consistently enhance the overall quality of 3D reconstructions.

F. Acknowledgments

We acknowledge funding from the Academy of Finland (grant No. 362409, 353139, 327911 and 353138) and support from the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Allice Wallenberg Foundation. MT was funded by the Finnish Center for Artificial Intelligence (FCAI). We also acknowledge CSC – IT Center for Science, Finland, for computational resources.

Figure 7. Qualitative mesh comparison for the ”8b5caf3398” scene from ScanNet++ dataset.

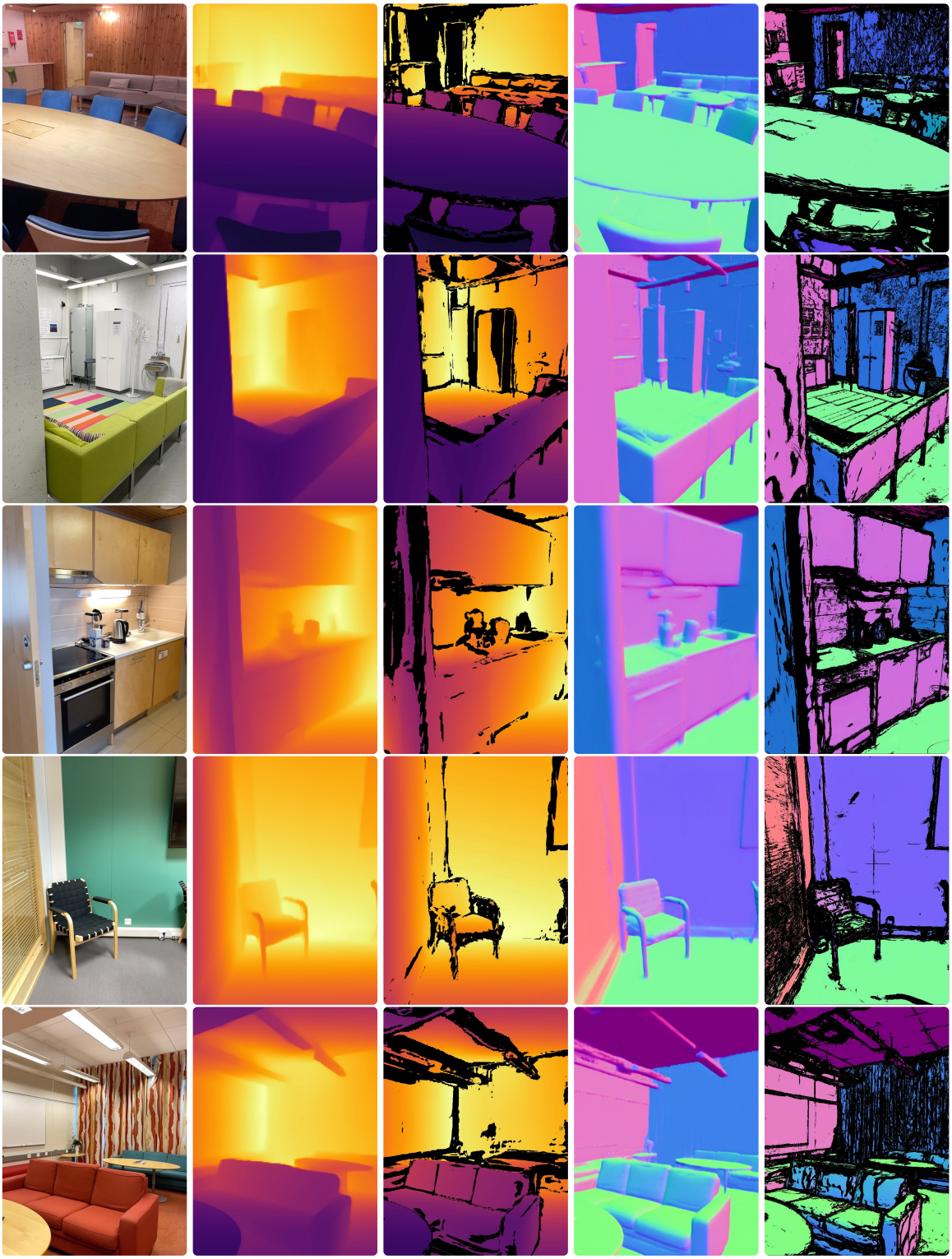

Figure 8. Qualitative visuals of our Depth Normal Consistency (DNR) and Adaptive Normal Regularization (ANR) terms. We visualize sensor depth and normals obtained from a pretrained network [14] after our filtering strategies. Our approach effectively filters out unreliable depth and normal values, especially in areas near boundaries, edges, and distant regions, leading to a more robust optimization process with more reliable prior regularization.

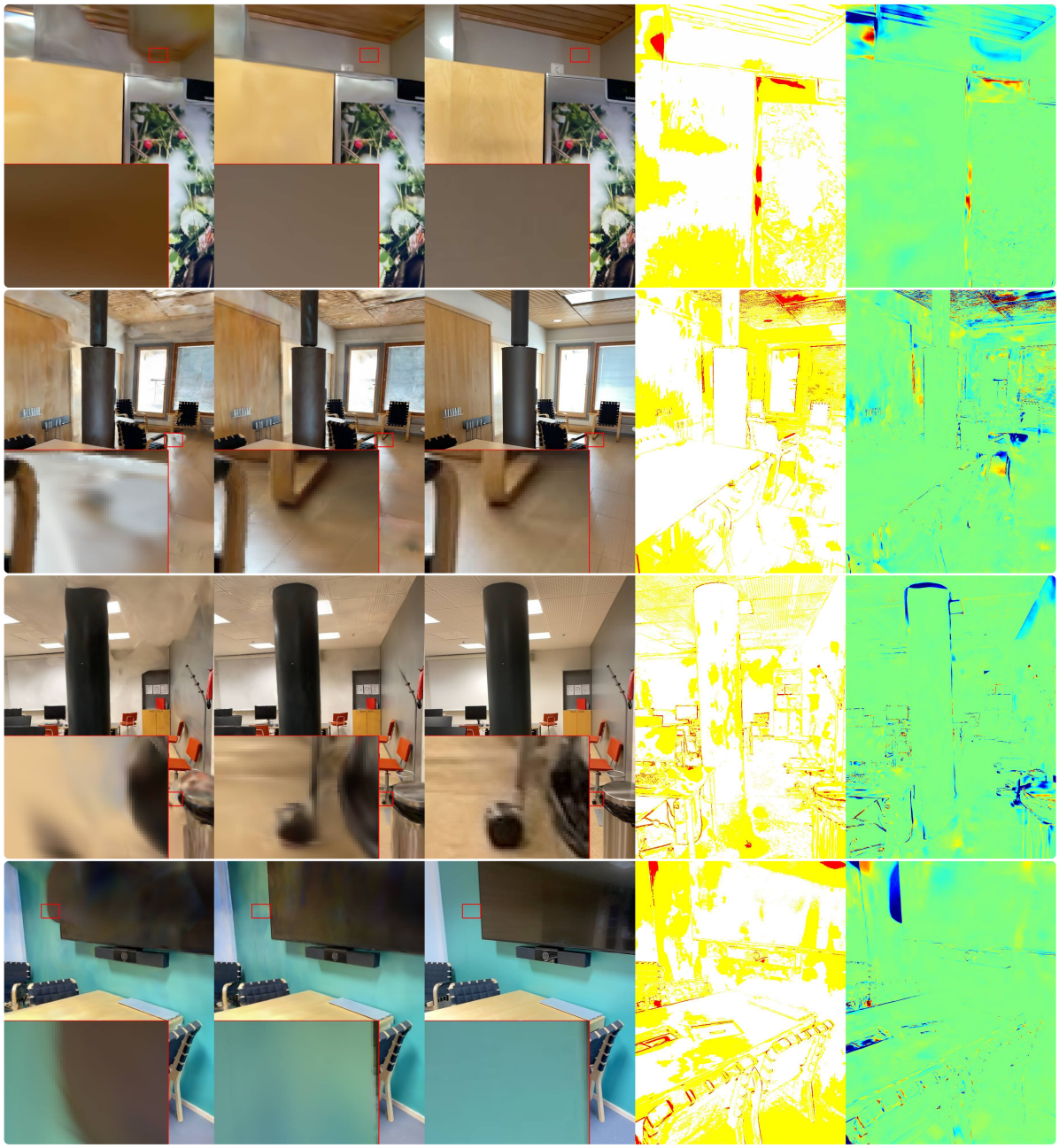

Figure 9. Novel view synthesis comparisons on the MuSHRoom dataset. From left to right: 2DGS [19] baseline, 2DGS with our proposed DNC and ANR optimization strategies, reference evaluation image, l_2 error contributions, 2DGS + Ours (red: 30%, yellow: 60%, white: 10%); l_2 error differences 2DGS + Ours vs 2DGS (red: higher error, blue: lower error).