

High-Fidelity Lightweight Mesh Reconstruction from Point Clouds

—Supplementary Material—

Chen Zhang Wentao Wang Ximeng Li Xinyao Liao Wanjuan Su Wenbing Tao*

National Key Laboratory of Science and Technology on Multi-spectral Information Processing,
Huazhong University of Science and Technology, China

{zhangchen_, wentaowang, ximengli, xinyao_liao, suwanjuan, wenbingtao}@hust.edu.cn

1 Implementation Details

2 More Experiments and Results

2.1 Scalability on Large-Scale Scenes

2.2 Edge Retention on CAD Models

2.3 Application on Human Mesh

2.4 Comparison with Mesh Simplification

3 More Ablation and Analysis

3.1 Ablation of Hybrid Features in SDF Network

3.2 Ablation of Neighborhood Label Constraint

3.3 Robustness to Varying Point Density

3.4 Robustness to Random Vertex Initialization

3.5 Limitations

1. Implementation Details

Experimental Device All of our experiments are conducted on an NVIDIA GeForce RTX 3090 GPU in an Intel Core i7-13700KF CPU system.

Delaunay Meshing. We use the Computational Geometry Algorithms Library (CGAL) [3] to construct the Delaunay triangulation \mathcal{D} for the generated vertices \mathcal{V} , where \mathcal{D} is the set of finite tetrahedrons and infinite tetrahedrons. The finite tetrahedrons in \mathcal{D} form the convex hull of \mathcal{V} , with each having four vertices from \mathcal{V} . At the boundary of the convex hull, there are infinite tetrahedrons sharing an infinite vertex. Such a setting ensures that each tetrahedron has four neighbor tetrahedrons sharing common triangular faces. By classifying the tetrahedrons, any triangular faces both inside and on the boundary of the convex hull can be extracted. In our multi-label voting strategy, we sample multiple reference points within each tetrahedron for classification. Using the SDF network, we predict the SDF values $\{f_\theta(r)\}$ for these reference points $\{r\}$. If $f_\theta(r) > S_{value}$, r is considered outside the implicit surface; otherwise, it is inside. S_{value} is the level set that defines the implicit surface and is

*Corresponding author.

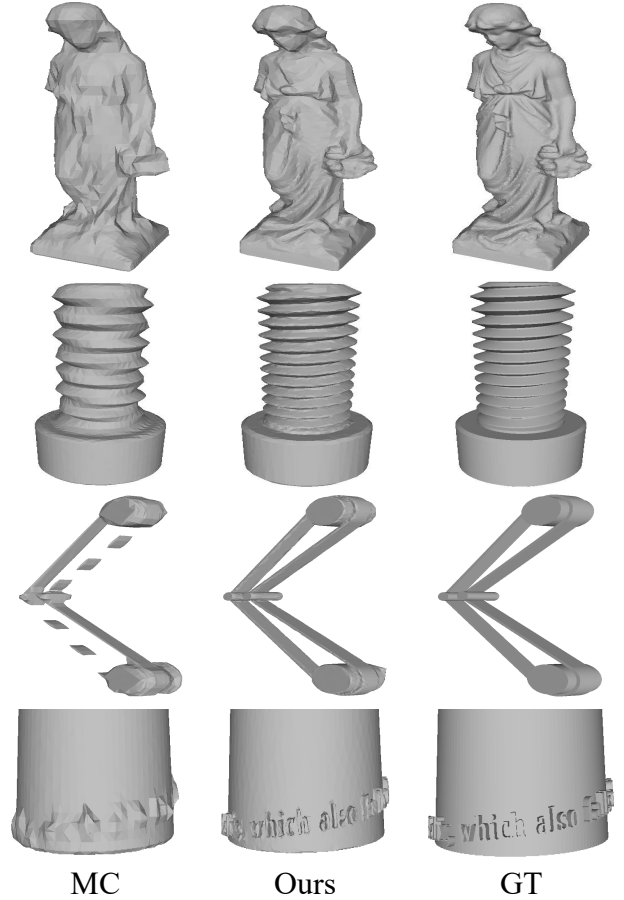


Figure 1. The comparison of our adaptive meshing method with 32-resolution MC using the same element count. The input SDFs for both methods are the same and our method preserves details and sharpness well.

typically set to 0. We then compare the occupancy counts of the two classes, and the class with the higher count determines the tetrahedron’s label.

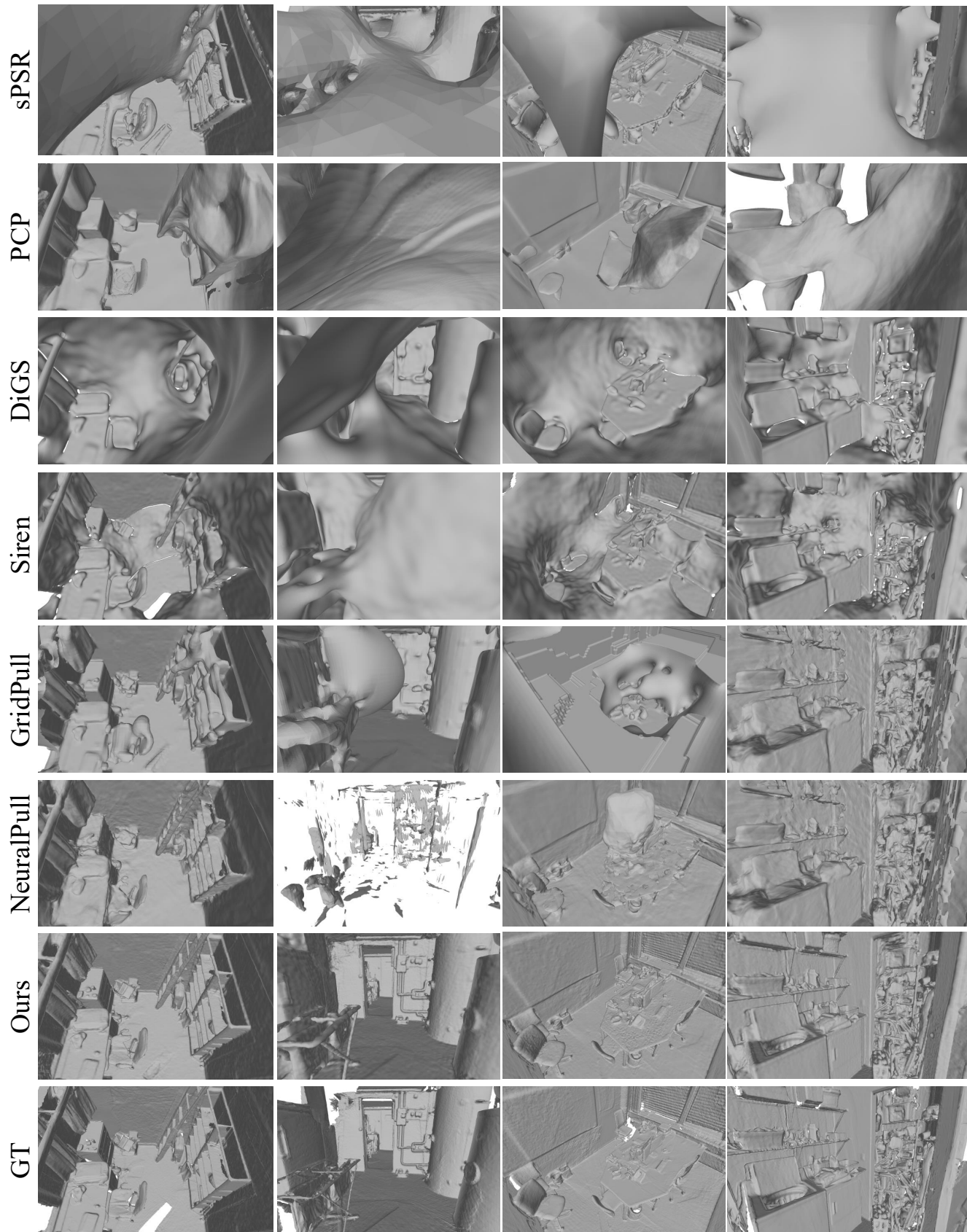


Figure 2. Visual results of various implicit reconstruction methods on ScanNet. All methods use 512-resolution MC to extract meshes, allowing for a comprehensive comparison of the accuracy of modeled implicit representations.

2. More Experiments and Results

2.1. Scalability on Large-Scale Scenes

Lightweight reconstruction in large-scale scenes poses significant challenges, requiring reconstruction methods to ensure both scalability and detail preservation. The extensive data involved renders many explicit reconstruction (including point cloud downsampling and point set triangulation) and neural meshing methods unfeasible due to memory limitations. To evaluate the scalability of our lightweight reconstruction on large-scale scenes, we conduct experiments on the ScanNet dataset [6]. Each scene is sampled with 1 million points, characterized by complex details and missing regions, which pose significant challenges for reconstruction. We compare recent neural implicit reconstruction methods, including Siren [14], NeuralPull [11], DiGS [1], GridPull [4], and PCP [12], along with the classic screened Poisson (sPSR) [8]. Given that sPSR requires point normals, these are estimated using Open3D [17], with the number of nearest neighbors set to 150.

We first evaluate the implicit representations modeled by different implicit reconstruction methods. Marching Cubes (MC) [10] with a grid resolution of 512 is used to extract dense meshes for all methods. The visual results, shown in Figure 2, highlight the significant reconstruction advantages of our method. For these complex large-scale scenes, other implicit methods frequently produce numerous artifacts that severely degrade visual quality. In contrast, our method not only avoids such artifacts but also effectively preserves fine-grained details, demonstrating the effectiveness of the proposed hybrid features in enhancing SDF representation. The quantitative evaluation results are shown in Table 1, and our method achieves the best results on all metrics. It is important to note that the scenes in ScanNet often contain incomplete areas, which can lead to lower chamber distance (CD) values for methods that struggle to effectively fill these holes. As shown in Figure 2, our method is able to produce more complete surfaces, resulting in CD values that are only slightly different from those of Siren. However, our method demonstrates a significant advantage in the F-score (F1), which more comprehensively accounts for both reconstruction accuracy and completeness. In addition, our method shows a substantial improvement in curvature error (CE), indicating its effectiveness in preserving the fine-grained details of the scenes, which can also be proven in Figure 2.

Next, we further validate the performance of our adaptive meshing (AM) on large-scale scenes by comparing it with Marching Cubes (MC) at a grid resolution of 128. To ensure a fair comparison, the number of vertices generated by our adaptive meshing is matched to that produced by MC at 128 resolution. As shown in Table 1, our adaptive meshing achieves more accurate results compared to MC. Re-

Method	CD ↓ (10^{-4})	NC ↑	F1 ↑	CE ↓ (10^{-3})	V (10^4)	F (10^4)
sPSR [8]	29.393	0.819	0.686	5.904	95.3	191.2
PCP [12]	11.207	0.841	0.671	3.937	188.3	376.2
DiGS [1]	4.279	0.868	0.808	2.843	397.3	794.7
Siren [14]	1.886	0.872	0.744	9.815	152.2	304.4
GridPull [4]	6.424	0.859	0.786	4.098	101.4	202.8
NeuralPull [11]	14.614	0.804	0.662	4.785	103.1	204.5
Ours / MC512	1.808	0.912	0.891	1.676	154.7	309.5
Ours / MC128	2.064	0.886	0.872	2.064	8.8	17.6
Ours / AM	1.816	0.896	0.884	1.854	8.8	17.6

Table 1. Quantitative comparison on the ScanNet dataset. Above the double horizontal line is the evaluation of dense meshes extracted by 512-resolution MC, used to compare the modeled implicit representations. Below is the evaluation of lightweight meshes. We maintain the mesh element count of our adaptive meshing (AM) consistent with that of 128-resolution MC for a fair comparison.

markably, even with only 5% of the dense mesh elements, our method still achieves comparable performance, significantly surpassing the dense meshes produced by other implicit reconstruction methods. These results demonstrate the capability of our approach to achieve high-quality lightweight mesh reconstruction for large-scale scenes. The visual comparisons between our adaptive meshing and 128-resolution MC are presented in Figure 3. Obviously, our adaptive meshing preserves more details to the greatest extent possible.

2.2. Edge Retention on CAD Models

We further validate the edge retention capability of our lightweight reconstruction on complex CAD models. Following NeuralPull [11], we conduct experiments on a subset of ABC dataset [9], which contains scans of one hundred models. Consistent with previous experiments, we first use 512-resolution MC to extract dense meshes for evaluating the accuracy of the modeled SDFs. Subsequently, we evaluate the resulting lightweight meshes generated by our adaptive meshing and MC with a grid resolution of 32, ensuring an equivalent number of elements (approximately 0.3% of those in a dense mesh). Figure 5 presents the visualization results of our dense and lightweight meshes. Even using only 0.3% of the mesh elements, our lightweight mesh achieves similar results to the dense mesh. A more detailed comparison between our adaptive mesh (AM) and MC is shown in Figure 4. Compared with MC, our adaptive mesh effectively perceives sharp edges and extracts more realistic lightweight meshes. The quantitative results are shown in Table 2, and our adaptive meshing (AM) holds a significant advantage over MC.

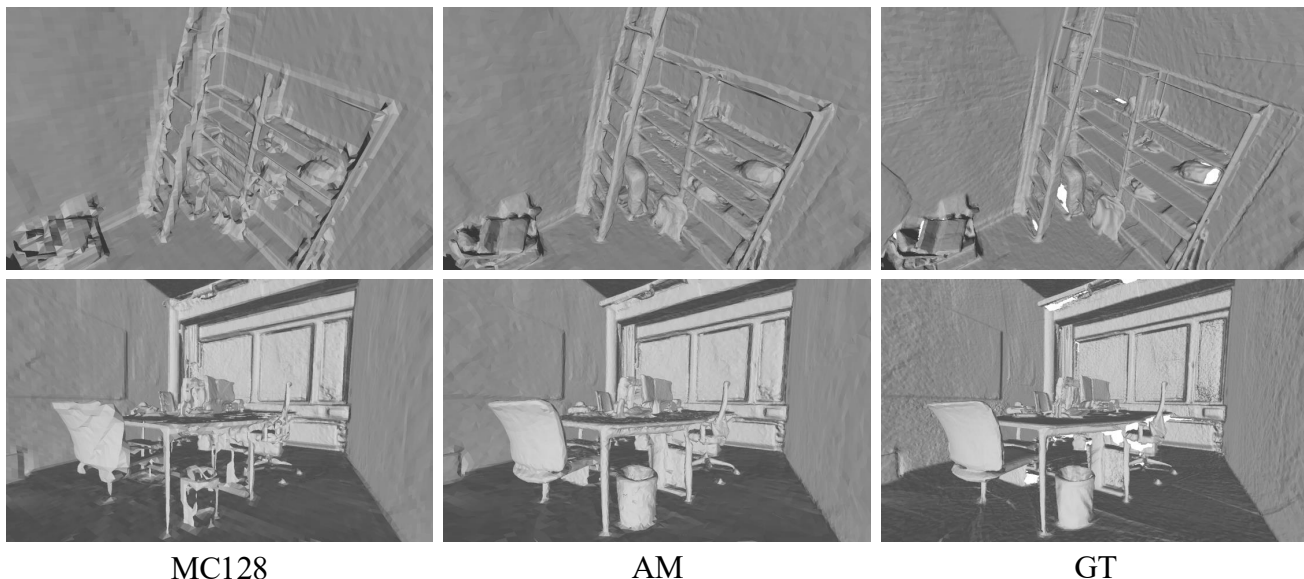


Figure 3. Visual results of lightweight meshes produces by our adaptive meshing (AM) and 128-resolution MC on ScanNet. Both results have the same number of vertices. The lightweight meshes generated by our adaptive meshing effectively preserve both model integrity and fine-grained details.

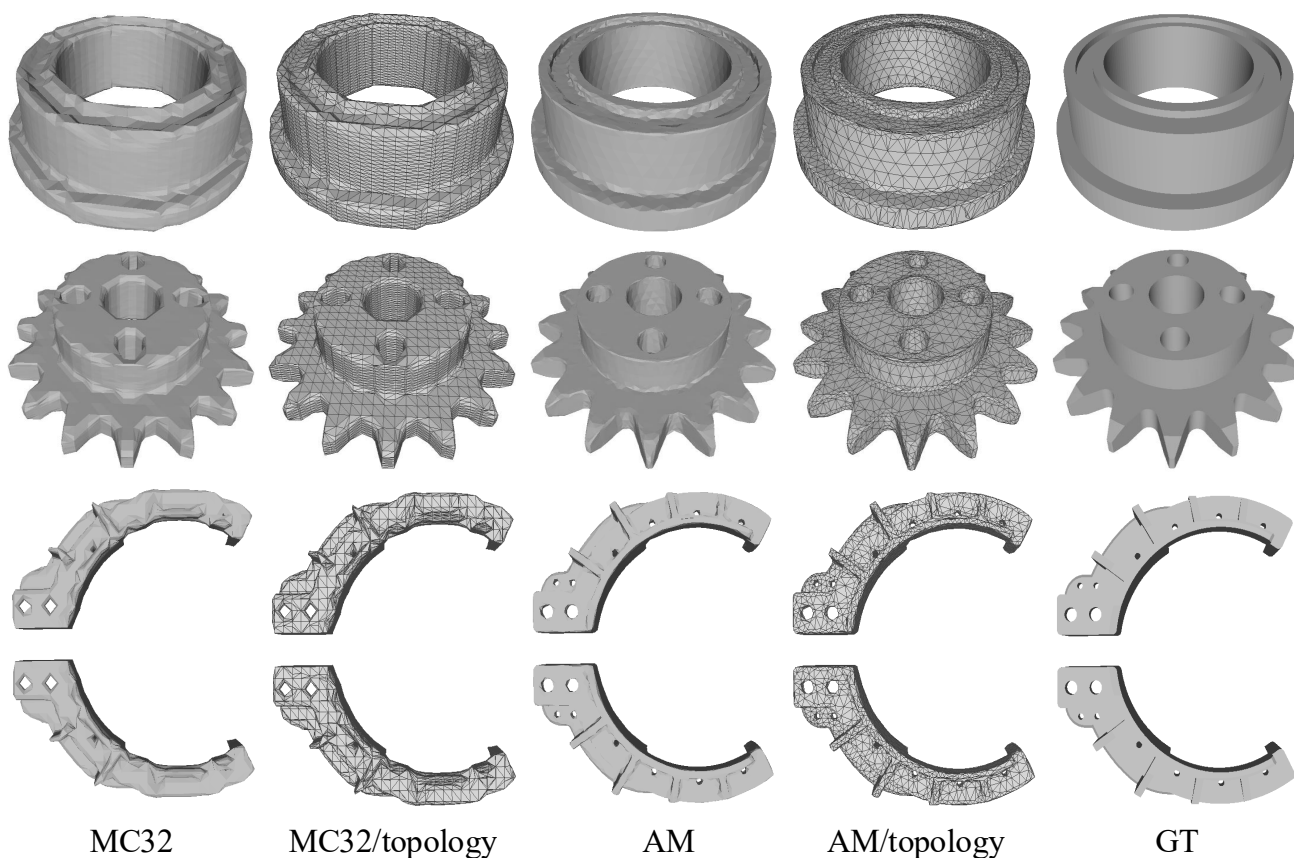


Figure 4. Visual results of lightweight meshes produced by 32-resolution MC and our adaptive meshing (AM) method on the ABC dataset. Both methods produce the same number of mesh elements. In contrast, our adaptive meshing efficiently perceives sharp edges and allocates vertices densely to preserve sharpness and details.

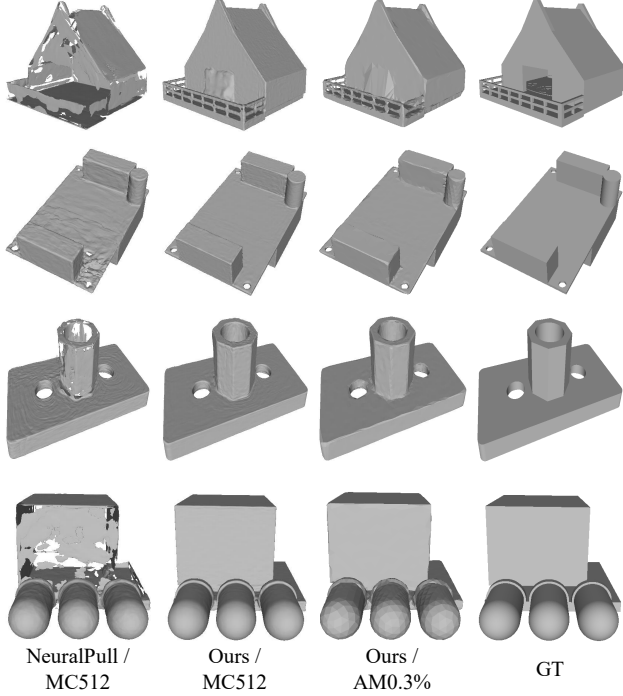


Figure 5. Visual comparison between the lightweight mesh produced by our method and the dense mesh on the ABC dataset. ‘MC512’ denotes dense meshes extracted using Marching Cubes at a resolution of 512, while ‘AM’ represents lightweight meshes generated by our adaptive meshing algorithm, containing only 0.3% of the elements in the dense meshes.

Method	CD ↓ (10^{-3})	NC ↑	F1 ↑	CE ↓ (10^{-2})	V (10^3)	F (10^3)
NeuralPull [11]	0.805	0.928	0.716	0.285	1072.4	2144.2
Ours / MC512	0.257	0.944	0.758	0.257	1248.2	2496.2
Ours / MC32	0.408	0.911	0.661	0.316	4.2	8.4
Ours / AM	0.352	0.934	0.742	0.281	4.2	8.4

Table 2. Quantitative comparison on the ABC dataset. Above the double horizontal line is the evaluation of high-resolution meshes extracted by 512-resolution MC, used to compare the modeled SDF. Below is the evaluation of lightweight meshes. We maintain the mesh element count of our adaptive meshing (AM) consistent with that of MC at a resolution of 32 for a fair comparison.

2.3. Application on Human Mesh

Reconstructing human data is a popular experiment currently. We use the Dfaust dataset [2] to construct experiments, sampling each shape with $20k$ points. Some recent neural implicit reconstruction methods are compared, including NeuralPull [11] and DiGS [1]. We first evaluate the accuracy of the modeled implicit representation. All methods use MC at a grid resolution of 512 to extract the surface. As can be seen from Figure 6.a, our method is able to recover better human details, such as fingers and toes, re-

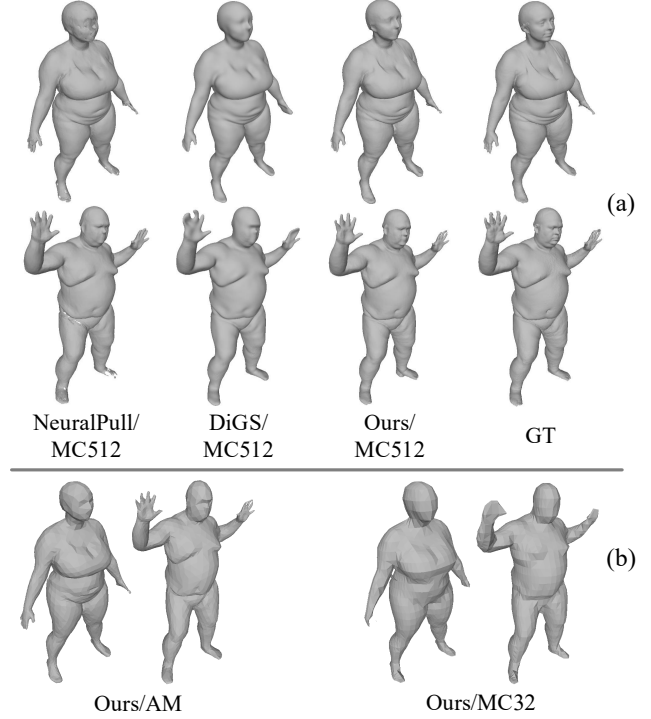


Figure 6. Visual results on Dfaust. (a) High-resolution meshes extracted by MC with a grid resolution of 512 using different neural implicit reconstruction methods. (b) Lightweight meshes extracted using MC with a grid resolution of 32 and our adaptive meshing (AM) algorithm. Both results have the same number of mesh elements.

Method	CD ↓ (10^{-5})	NC ↑	F1 ↑	CE ↓ (10^{-3})	V (10^3)	F (10^3)
NeuralPull [11]	1.408	0.977	0.975	1.884	876.8	1754.4
DiGS [1]	1.042	0.982	0.972	0.195	1815.5	3631.1
Ours / MC512	1.039	0.984	0.976	0.179	644.0	1287.9
Ours / MC32	2.685	0.949	0.873	0.432	2.2	4.5
Ours / AM	1.039	0.974	0.976	0.193	2.2	4.5

Table 3. Quantitative comparison on the Dfaust dataset. Above the double horizontal line is the evaluation of high-resolution meshes extracted by 512-resolution MC, used to compare the modeled SDF. Below is the evaluation of lightweight meshes. We maintain the vertex element count of our adaptive meshing (AM) consistent with that of 32-resolution MC.

ducing the generation of artifacts. The quantitative results are shown in Table 3, and our method achieves the best results on all metrics.

Next, we further test the performance of our adaptive meshing (AM) algorithm. The mesh element count of our AM results is set to match that of 32-resolution MC. Table 3 shows the quantitative results. Surprisingly, although our lightweight meshes contain only 0.34% of the vertex ele-

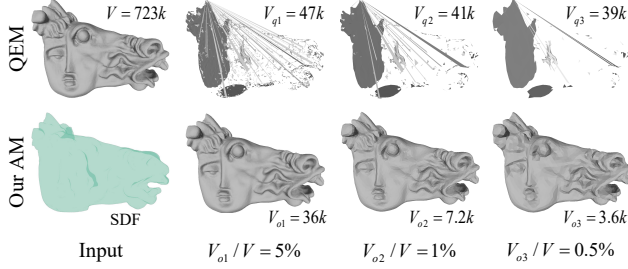


Figure 7. Visual comparison between our adaptive meshing (AM) algorithm and the classic mesh simplification algorithm, QEM. We extract a lightweight mesh directly from the implicit surface, while QEM simplifies the high-resolution mesh extracted by MC. Since QEM only allows setting the output triangular face count, we match it with ours for comparison. V denotes the vertex count. QEM faces the problem of robustness.

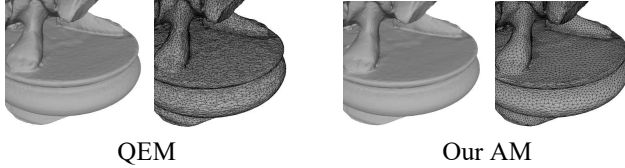


Figure 8. Mesh construction comparison of our adaptive meshing (AM) algorithm and QEM. Our method produces triangles with more uniform angles.

ments compared to the high-resolution meshes, they achieve results similar to those of the high-resolution meshes and significantly outperform the 32-resolution MC. The visual results of lightweight meshes are shown in Figure 6.b. Compared to MC, our method is able to recover details from the implicit representation with few mesh elements.

2.4. Comparison with Mesh Simplification

Mesh simplification is also an effective way to produce lightweight meshes. Unlike our adaptive meshing method, which takes implicit representations as input, it requires high-resolution meshes as input. Therefore, from the perspective of the entire reconstruction system, three steps are needed to obtain lightweight meshes: first, fitting an implicit representation from point clouds; second, extracting a high-resolution mesh using MC; and finally, performing mesh simplification. However, this three-stage manner typically results in a loss of fidelity relative to the implicit surface, causing accumulated errors. Additionally, mesh simplification methods often face robustness challenges due to their strict requirements on the quality of input meshes. Ensuring watertightness of output meshes is often not feasible, and distorted outcomes may arise, as shown in Figure 7. Nevertheless, mesh simplification remains a strong competitor to our method in some cases. To further compare the performance of the two methods in generating lightweight

Method	CD ↓ (10^{-5})	NC ↑	F1 ↑	CE ↓ (10^{-3})	CT ↑ W/M/NS	V (10^4)	F (10^4)
MC (res512)	0.558	0.967	0.958	0.106	1.0	86.8	182.6
QEM (15%)	0.594	0.967	0.947	0.128	0.0/1.0/1.0	13.1	25.1
AM (15%)	0.558	0.969	0.959	0.113	1.0	12.6	25.1
QEM (10%)	0.598	0.965	0.946	0.152	0.0/1.0/1.0	9.1	17.2
AM (10%)	0.559	0.968	0.959	0.125	1.0	8.6	17.2
QEM (5%)	0.618	0.964	0.941	0.179	0.0/1.0/1.0	4.8	8.9
AM (5%)	0.560	0.966	0.959	0.154	1.0	4.4	8.9

Table 4. Quantitative comparison of our adaptive meshing (AM) and QEM on Stanford. QEM takes a dense mesh extracted from MC at a resolution of 512 as input and reduces the number of triangles to 15%, 10%, and 5% of the original. For a fair comparison, our adaptive meshing generates the same number of elements.

meshes, we conduct experiments on the Stanford dataset [5]. Our adaptive meshing method utilizes our learned SDF as input, whereas the classical mesh simplification algorithm, QEM [7], uses high-resolution meshes extracted from 512-resolution MC as input. Both methods are set to produce lightweight meshes, with 15%, 10%, and 5% of the elements of the high-resolution meshes, respectively. The quantitative results are presented in Table 4. Satisfactorily, our adaptive meshing method produces competitive results compared to QEM, while ensuring correct topology (CT) and avoiding the creation of non-watertight meshes. This demonstrates the potential superiority of directly extracting lightweight meshes from implicit surfaces, avoiding the error accumulation inherent in multi-stage processes. The outstanding performance also demonstrates the robustness and accuracy of our proposed adaptive meshing method. As an additional advantage, our method tends to produce triangles with uniform angles, which are often preferred in finite element analysis. In contrast, QEM sacrifices uniform triangle angles to preserve sharpness. The mesh construction visualization is shown in Figure 8.

3. More Ablation and Analysis

3.1. Ablation of Hybrid Features in SDF Network

In the ablation studies about SDF learning within the manuscript, we demonstrate the effectiveness of each module in our introduced hybrid features. The results in Table 5 of the manuscript show that while grid features capture details well, they tend to cause overfitting and artifacts, leading to significant degradation in the Chamfer Distance (CD). To better illustrate this situation, we present the visual results of the ablation studies in Figure 9 of this supplementary material. It can be seen that the hybrid representation of grid and tri-plane features combines the complementary advantages of both, enhancing detail expression while avoiding the risk of overfitting.

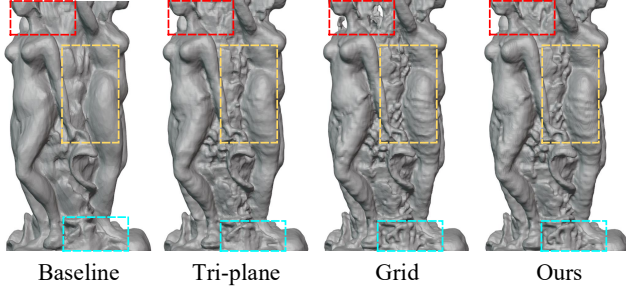


Figure 9. Visual results of our ablation studies on SDF Network. ‘Grid’ and ‘Tri-plane’ represent the results of using only single-type features, while our final method uses a hybrid of both features. ‘Grid’ preserves more details than ‘Tri-plane,’ as highlighted in the yellow and light green boxes, but it also introduces overfitting and artifacts, as indicated in the red box. In contrast, the hybrid features we adopt can avoid this problem.

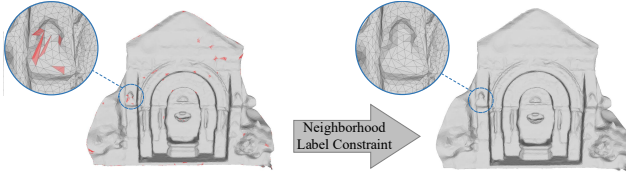


Figure 10. Visual results of our neighborhood label constraint. The red triangles represent the triangular faces with non-manifold edges, which are caused by the misclassification of some narrow tetrahedrons. These triangulation artifacts are eliminated to produce a manifold mesh.

3.2. Ablation of Neighborhood Label Constraint

In our Delaunay meshing algorithm, a neighbor label constraint strategy is proposed to ensure the manifold properties of the mesh. Due to page limitations, Figure 3 in our manuscript only shows a 2D example. To more clearly demonstrate its corrective effect on mesh construction, we present detailed 3D results in Figure 10 of this supplementary material. It can be seen that our strategy can significantly reduce the generation of non-manifold edges and triangular artifacts.

3.3. Robustness to Varying Point Density

To clearly demonstrate the adaptability of our method to different point densities, we sample various numbers of points for experiments on the Stanford dataset [5], including 200k, 100k, and 20k. Since the input point cloud is solely for our SDF modeling, we use Marching Cubes (MC) [10] with a grid resolution of 512 to extract the surface. The quantitative evaluation results are shown in Table 5. We also include reconstruction results from NeuralPull [11] with 200k points for comparison. As point density decreases significantly, our method continues to provide robust reconstruction. In fact, the results of our method with 20k points out-

Method	CD ↓ (10^{-5})	NC ↑	F1 ↑	CE ↓ (10^{-3})
NeuralPull-200k [11]	0.606	0.963	0.950	0.329
ours-200k	0.558	0.967	0.958	0.106
ours-100k	0.564	0.968	0.958	0.156
ours-20k	0.605	0.957	0.955	0.277

Table 5. Quantitative comparison with various point number.

Method	FPS	Random Sampling	CD ↓ (10^{-5})	F1 ↑	CE ↓ (10^{-3})
Model-I		✓	0.7551	0.9282	0.1277
Ours	✓		0.7550	0.9292	0.1256

Table 6. Quantitative results of different point selector implementations. Our adaptive meshing method shows robustness to random initialization of vertices.

perform NeuralPull’s results with 200k points in terms of accuracy metrics. This further highlights the superiority of our method in reconstructing sparse data.

3.4. Robustness to Random Vertex Initialization

In our vertex generator, we use farthest point sampling (FPS) as the point selector implementation to generate initial vertices from surface queries. This is mainly because FPS produces a uniformly distributed point set, preserving the overall structure of the implicit surface. To validate the robustness of our method, we employ random sampling as the point selector implementation to assess the impact of different vertex initialization. Experiments are constructed on the Thing10K dataset [16] and the quantitative results are shown in Table 6. It can be seen that applying random sampling only produces a slight performance degradation, demonstrating the robustness of our method to different vertex initialization methods.

3.5. Limitations

As an optimization-based method, its primary advantage lies in not requiring ground truth data for training, enabling direct reconstruction of given unoriented point clouds. However, excessive time consumption remains a common challenge for optimization-based approaches. Our lightweight reconstruction achieves robust results within twenty minutes, comparable to NeuralPull [11] and DiGS [1], and significantly outperforms NeuralIMLS [15] and PINC [13] which require several hours. However, achieving more efficient reconstruction is still demanding. It would be a good direction to combine more prior information in future work.

References

- [1] Yizhak Ben-Shabat, Chamin Hewa Koneputugodage, and Stephen Gould. Digs: Divergence guided shape implicit neural representation for unoriented point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19323–19332, 2022. [3](#), [5](#), [7](#)
- [2] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering human bodies in motion. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. [5](#)
- [3] Jean-Daniel Boissonnat, Olivier Devillers, Monique Teilaud, and Mariette Yvinec. Triangulations in cgal. In *Proceedings of the sixteenth annual symposium on Computational geometry*, pages 11–18, 2000. [1](#)
- [4] Chao Chen, Yu-Shen Liu, and Zhizhong Han. Gridpull: Towards scalability in learning implicit representations from 3d point clouds. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 18322–18334, 2023. [3](#)
- [5] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. [6](#), [7](#)
- [6] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias 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. [3](#)
- [7] Michael Garland and Paul S Heckbert. Simplifying surfaces with color and texture using quadric error metrics. In *Proceedings Visualization'98 (Cat. No. 98CB36276)*, pages 263–269. IEEE, 1998. [6](#)
- [8] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Transactions on Graphics (ToG)*, 32(3):1–13, 2013. [3](#)
- [9] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. [3](#)
- [10] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *Seminal graphics: pioneering efforts that shaped the field*, pages 347–353. 1998. [3](#), [7](#)
- [11] Baorui Ma, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker. Neural-pull: Learning signed distance function from point clouds by learning to pull space onto surface. In *International Conference on Machine Learning*, pages 7246–7257. PMLR, 2021. [3](#), [5](#), [7](#)
- [12] Baorui Ma, Yu-Shen Liu, Matthias Zwicker, and Zhizhong Han. Surface reconstruction from point clouds by learning predictive context priors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6326–6337, 2022. [3](#)
- [13] Yesom Park, Taekyung Lee, Jooyoung Hahn, and Myungjoo Kang. p-poisson surface reconstruction in curl-free flow from point clouds. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, pages 60077–60098, 2023. [7](#)
- [14] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020. [3](#)
- [15] Zixiong Wang, Pengfei Wang, Pengshuai Wang, Qiujie Dong, Junjie Gao, Shuangmin Chen, Shiqing Xin, Changhe Tu, and Wenping Wang. Neural-impls: Self-supervised implicit moving least-squares network for surface reconstruction. *IEEE Transactions on Visualization & Computer Graphics*, (01):1–16, 2023. [7](#)
- [16] Qingnan Zhou and Alec Jacobson. Thingi10k: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016. [7](#)
- [17] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018. [3](#)