ORCA – Online Research @ Cardiff

Scalable and High-Quality Neural Implicit Representation for 3D Reconstruction

Leyuan Yang, Bailin Deng, *Member, IEEE*, and Juyong Zhang†, *Member, IEEE*

Abstract—Various SDF-based neural implicit surface reconstruction methods have been proposed recently, and have demonstrated remarkable modeling capabilities. However, due to the global nature and limited representation ability of a single network, existing methods still suffer from many drawbacks, such as limited accuracy and scale of the reconstruction. In this paper, we propose a versatile, scalable and high-quality neural implicit representation to address these issues. We integrate a divide-and-conquer approach into the neural SDF-based reconstruction. Specifically, we model the object or scene as a fusion of multiple independent local neural SDFs with overlapping regions. The construction of our representation involves three key steps: (1) constructing the distribution and overlap relationship of the local radiance fields based on object structure or data distribution, (2) relative pose registration for adjacent local SDFs, and (3) SDF blending. Thanks to the independent representation of each local region, our approach can not only achieve high-fidelity surface reconstruction, but also enable scalable scene reconstruction. Extensive experimental results demonstrate the effectiveness and practicality of our proposed method.

Index Terms—Signed Distance Function, Scalability, Registration, Optimization, Blending.

---

1 INTRODUCTION

RECONSTRUCTING 3D surfaces from multi-view images is a fundamental problem in computer vision and computer graphics. The recovered dense geometric surfaces provide spatial structure information useful for a variety of downstream applications, such as robotics navigation, animation, physical simulation, virtual and augmented reality, and more.

Traditionally, multi-view stereo (MVS) algorithms [1], [2], [3] have been one of the preferred methods in the realm of 3D reconstruction. However, these classical algorithms typically demand substantial computational resources and time, and their capability to precisely reconstruct geometric surfaces is constrained. Additionally, a critical limitation of these approaches is their inability to effectively address ambiguous areas within the scene. Such areas encompass repeated texture patterns, extensive regions characterized by uniform colors and low-texture details, semi-transparent elements like fog, or regions with abrupt color transitions. These challenges will further lead to inaccuracies in the reconstruction process, resulting in missing surfaces or significant noises.

Recently, neural radiance fields (NeRF) [4] have been proposed to utilize multi-layer perceptrons (MLP) for representing 3D scenes and achieve impressive results for novel view synthesis. Since the introduction of NeRF, various neural implicit surface reconstruction methods have been developed, and it has become a promising alternative paradigm for reconstruction. These implicit methods utilize a neural network to represent a function encoding the surface, such as the signed distance function (SDF) to the surface [5], [6].

Thanks to its capability to model complex shapes without being limited by resolution, this new approach has greatly improved the performance of multi-view 3D reconstruction [7], [8].

Nevertheless, existing neural implicit surface reconstruction methods still face several challenges. A significant restriction arises from the limited expressive capacity of a single MLP, which may be insufficient for accurate reconstruction of high-frequency details. In addition, due to an MLP's global nature, optimizing the geometry in one part of a scene may unintentionally alter distant parts and lead to undesirable results. Furthermore, reconstructing an entire scene with a single network requires global data including global camera poses for the whole scene, which will become increasingly impractical as the data volume increases. For a very large scene, collecting all the data at once is challenging, and pose estimation algorithms [9], [10], [11] require substantial time and computational resources and may even fail to provide accurate global camera poses.

These challenges significantly hinder SDF-based reconstruction. To address these issues, we propose a novel neural implicit representation that applies the divide-and-conquer strategy to the SDF-based realm. Unlike previous works that solely rely on global information for block reconstruction of the color radiance field [12], [13], we aim to achieve this strategy from the stage of data collection and processing, thinking that this way is essential to realize the goal of scalable large-scale scene reconstruction. Consequently, automatically and conveniently determining the relative positions between different local radiation fields has become an urgent problem to address. Another distinction lies in our focus on solving the discontinuous jump problem arising from splicing in SDFs, rather than solely addressing blending problem in color spaces.

Specifically, unlike previous methods that only use a single radiance field, we design to model an object or

---

- L. Yang and J. Zhang are with the School of Mathematical Sciences, University of Science and Technology of China.
- B. Deng is with the School of School of Computer Science and Informatics, Cardiff University.

†Corresponding author. Email: `juyong@ustc.edu.cn`.

scene as a fusion of multiple neural SDFs with overlapping regions. The distribution of loacl neural radiance fields and the overlapping relationship are constructed based on the structure of the object or the layout of the data, and then each local SDF is reconstructed separately. Afterward, we utilize the overlapping to register the camera poses between adjacent radiance fields, which aligns all the local SDFs into a global coordinate system. Finally, the local neural SDFs with common regions are blended using a softmax-based weighting scheme to avoid discontinuity in the overlap area, producing a global SDF that represents the whole scene. Our proposed representation offers several significant advantages compared with existing methods: leftmargin=*

- First, using separate MLPs to represent different local regions can narrow down the area expressed by each MLP, enabling us to fully utilize its representational power for high-fidelity reconstruction of the details in the whole scene.
- Moreover, the use of multiple independent local neural SDFs offers flexibility that benefits the reconstruction and downstream applications: firstly, each local region can recover camera poses in its own coordinate system, avoiding the failure to obtain global poses due to the enormous volume of data; secondly, at the application level, our representation allows extraction, editing, or other usage of individual or a subset of local SDFs without affecting other components.
- Finally, our representation makes the reconstruction process scalable: larger scenes can be easily handled by introducing more additional and independent local SDFs to the existing radiation field, while maintaining the reconstruction quality of the local regions. This provides a practical solution that can reconstruct urban-scale scenes.

Extensive experiments demonstrate that our proposed representation achieves high-quality and scalable 3D reconstruction, compared to existing methods using a single neural network. By reconstruction with eight local SDFs, we improve the accuracy of Lego's Chamfer distance by up to 45.6%, capturing a higher level of intricacy and detail. Additionally, by simply introducing more independent local neural SDFs, we achieve scalable and high-quality reconstruction with high-fidelity texture map covering an area of up to 1200m×800m. Moreover, with the independence of local SDFs, we can implement object-level editing within the scene.

2 RELATED WORK

Neural Implicit Surface Reconstruction. Compared with explicit reconstruction, such as voxels [2], [14] and triangular mesh [15], [16], implicit reconstruction employs an implicit function to represent the surface. Consequently, it is continuous, enabling the extraction of the surface at any resolution. IDR [17] reconstructs surfaces with neural networks by representing the geometry as the zero level set of SDF. VolSDF [5] and NeuS [6] use a weight function that involves SDF during the rendering process to make colors and geometry closer, resulting in improved quality compared to

traditional MVS methods. Furthermore, some works [18], [19] introduce several priors to make the reconstructed surface more refined. Building on the impressive work of Instant-NGP [20], NeuS2 [7] introduces hash-encoding to further significantly improve the reconstruction speed. To address the limitations of the hash table on the regularization term, Eikonal loss [21], Neuralangelo [8] proposes numerical gradient to further refine surface reconstruction. Our representation is universal for neural SDF reconstruction, which means that these methods can serve as models for any local SDF within our method.

Large-scale Scene Reconstruction. Scene reconstruction is a long-standing problem in computer vision and graphics, and many early optimization-based methods [22], [23], [24], [25], [26], [27], [28] have been proposed to solve this problem. While these methods have shown impressive reconstruction performance, they often exhibit artifacts or holes in the reconstructed scenes. Since the introduction of NeRF, several works, such as pretraining networks to learn scene priors [29], [30], [31], [32], [33], [34] and exploring grid representations to accelerate training [20], [35], [36], [37], have enabled neural radiance fields to reconstruct large scenes. Block-NeRF [12] extend NeRF to city-level scenes, dividing the scene into multiple blocks for independent reconstruction and then merging them together. Mega-NeRF [13] propose to use a 3D distance-based method to cluster training pixels into parts that can be trained separately. Grid-guided NeRF [38] introduce a two-branch model that combines NeRF-based methods with feature grid-based methods. Switch-NeRF [39] design a gating network to dispatch 3D points to different NeRF sub-networks. In this work, we mainly focus on geometric surface reconstruction and develop a scalable 3D reconstruction method.

NeRF Registration. To enable future NeRF research to be applicable in complex, city-level scenarios, the registration problem of overlapping NeRFs must be addressed. NeRF2NeRF [40] is the first work to attempt registering multiple NeRFs, calculating the initial transformation based on manually annotated key points and then refining it using Metropolis-Hastings sampling. BARF [41] and its variants [42], [43] contribute to the registration of camera poses by learning the 3D scene representation from NeRFs. Zero NeRF [44] leverages NeRF to register two sets of images with correspondence, although it still require a global registration method for initialization. These methods suffer from various issues, such as relying on global camera poses or needing for human interaction, which is impractical for large-scale scene with extensive datasets. Our registration method achieves automatic registration based on similarity in a small overlapping area, which provides greater convenience and applicability for large-scale reconstruction.

Based on the above relevant excellent works, and in order to solve mentioned limitations in the existing neural implicit SDF-based methods, we propose a scalable and high-quality representation for 3D geometric surface reconstruction.

3 METHOD

In the following, we first briefly review the basics of NeRF and neural implicit surfaces in Section 3.1. After-
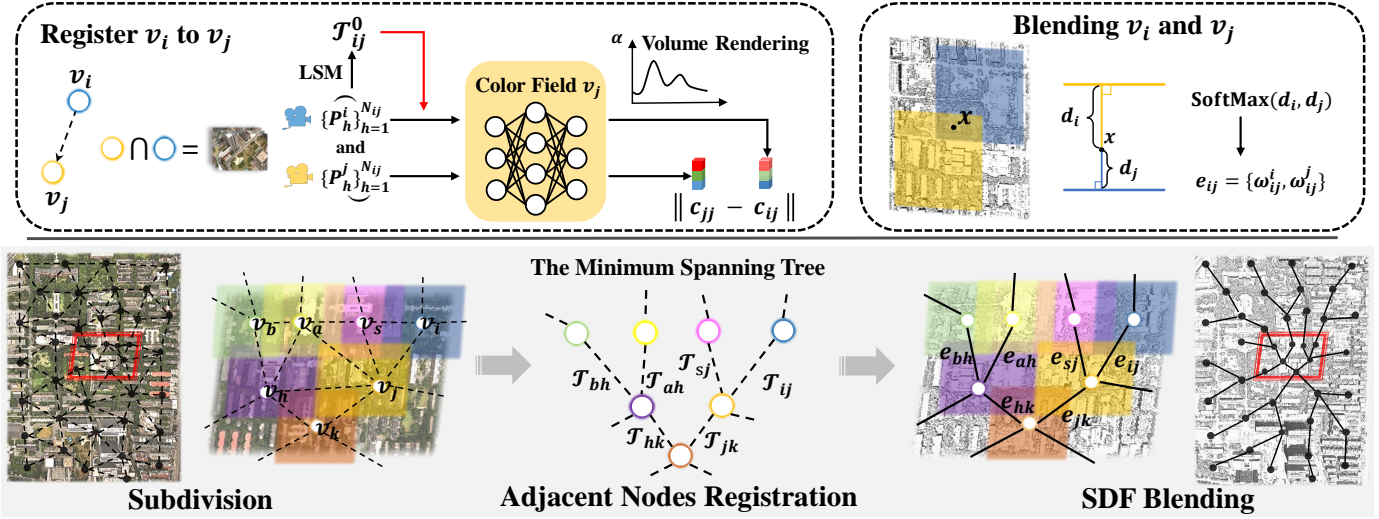
Fig. 1. **Pipeline.** At the bottom, we present the main construction processes of our method: Constructing the distribution and overlap relationship of the local radiance fields based on object structure or data distribution, Adjacent Nodes Registration, and SDF Blending. At the top, we take adjacent nodes v_i and v_j as an example to explain in detail how to use volume rendering for registration and then blending by the softmax-based weighting.

wards, we present the details of our scalable and high-quality neural implicit representation, including representation (Section 3.2), registration (Section 3.3), and blending (Section 3.4). Finally, Section 3.5 shows two additional applications based on our method.

3.1 Preliminary

Neural volume rendering. Given a set of multi-view posed images, NeRF represents a 3D scene as volume density and color field by volume rendering. Considering a ray $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$ emanated from a camera position $\boldsymbol{o} \in \mathbb{R}^3$ in the direction of $\boldsymbol{d} \in \mathbb{R}^3$, the volume rendering scheme essentially involves the integration of radiance from sampled points along the ray. Each 3D sampled point is associated with two spatial factors: volume density σ and color radiance $\boldsymbol{c}$, which are predicted using two coordinate-based MLPs. The rendered color of the given pixel is calculated as the integral of the transparency $T(t) = \exp\left(-\int_{t_n}^t \sigma(\boldsymbol{r}(u))du\right)$, density $\sigma(t)$, and radiance $\boldsymbol{c}(t)$ along the ray from the near bound t_n to the far bound t_f:

$$\hat{C} = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))\boldsymbol{c}(t)dt. \tag{1}$$

Practically, this integral is approximated using numerical quadrature.

SDF-based neural implicit surface. A surface $\mathcal{S}$ can be implicitly represented by the zero level set of its signed distance function $f(\boldsymbol{x}) : \mathbb{R}^3 \mapsto \mathbb{R}$, i.e., $\mathcal{S} = \{\boldsymbol{x} \in \mathbb{R}^3 | f(\boldsymbol{x}) = 0\}$. In the context of neural SDFs, NeuS propose converting the volume density in NeRF to SDF representations using a logistic function to allow optimization with neural volume rendering:

$$\sigma(\boldsymbol{x}) = \phi_s(f(\boldsymbol{x})),$$

where $\phi_s(x) = se^{-sx}/(1 + e^{-sx})^2$ is the logistic density distribution which is the derivative of the sigmoid function

$\Phi_s(x) = (1 + e^{-sx})^{-1}$ with slope s. Therefore, an unbiased opaque density function for a ray is:

$$\sigma(t) = \max\left(\frac{-\frac{d\Phi_s}{dt}(f(\boldsymbol{r}(t)))}{\Phi_s(f(\boldsymbol{r}(t)))}, 0\right). \tag{2}$$

3.2 Representation

Unlike existing single radiance field representations, we represent an object or scene as a fusion of multiple radiance fields with overlaps. To this end, we need to first determine the spatial range of each local neural SDF, and subsequently perform local reconstruction for each SDF while ensuring consistency between the SDFs in their overlapping regions. If we consider each local SDF as a node in a graph, then the overlapping relationship between local SDFs can be interpreted as forming an edge between the corresponding graph nodes. Thus, the representation we propose can be seen as an SDF graph, and we will present it from a graph perspective.

Specifically, we represent a single object or scene as an SDF graph $G = (\mathcal{V}, \mathcal{E})$ consisting of a set $\mathcal{V} = \{v_1, ..., v_n\}$ of n nodes and a set $\mathcal{E} = \{e_1, ..., e_m\}$ of m edges. Each node v_k represents an SDF associated with a local region of the scene, which is reconstructed from a subset of the input images. For any two nodes v_i, v_j whose image subsets have non-empty intersection, there will be an overlap between their local regions, and we use an edge connecting v_i and v_j to indicate their overlap relation. In our approach, the local SDF at each node is reconstructed separately, whereas the edges are utilized to register the camera poses from adjacent nodes and align their SDFs, producing a global SDF for the whole scene (see Section 3.3). To ensure such a global registration can be carried out successfully, the graph G must consist of a single connected component.

Depending on the structure of the objects and the layout of data, there are different strategies for partitioning the scene to construct a SDF graph. For instance, if the data is collected based on the components in the scene, such as

different parts of a human body shape or different buildings within a campus, then each component can be treated as a node. On the other hand, in some applications, the scene can be divided into grids with overlapping cells. For example, oblique photography datasets can be partitioned based on latitude and longitude. For such cases, each grid cell can be associated with one node.

After the nodes are determined, we use the image subset associated with each node to derive an SDF for the local region, by utilizing any existing neural implicit surface reconstruction method [5], [6], [7], [8], [18], [19]. Thanks to the independence of nodes, we can customize the reconstruction settings for each node to meet specific requirements. For instance, camera poses for each node can be independently estimated. Furthermore, we can select different SDF reconstruction methods for different nodes based on the characteristics of their associated regions. For example, in the case of large scenes, certain locations might demand tailored losses or specialized training strategies to improve reconstruction quality. This level of flexibility is not available in existing methods that rely on a single MLP for the whole scene.

3.3 Adjacent Nodes Registration

Since we only use a subset of the input images to reconstruct the SDF for each node, the resulting SDFs may not share a common world coordinate system across all nodes. Therefore, to derive a global SDF describing the whole scene, we perform registration to align the SDFs with overlapping regions.

Notably, since an edge in our graph indicates the overlapping relation, a straightforward solution is to perform registration for every edge. However, such a dense registration can be time-consuming and redundant. Instead, we first extract a minimum spanning tree (MST) from the full graph, and then only perform registration on the MST edges.

Each registration determines a transformation between the coordinate systems for the SDFs on the two nodes connected by an edge. Then, using the coordinate system of one node as the global coordinate system for the whole scene, we traverse the MST starting from this node, and apply the transformations on the edges to propagate the global coordinate system to all nodes. Afterwards, all local neural SDFs are represented using the global coordinate system. Next, we describe the details of our registration method.

Suppose we perform registration on an edge connecting two nodes v_i and v_j. We denote the input images and camera poses associated with a node as $\{I_1^k, ..., I_{N_k}^k\}$ and $\{P_1^k, ..., P_{N_k}^k\}$, respectively $(k = i, j)$. In addition, let $\{I_1^{ij}, ..., I_{N_{ij}}^{ij}\}$ be the common images between the two nodes, and $\{P_1^k, ..., P_{N_{ij}}^k\}$ $(k = i, j)$ be their corresponding camera poses in the respective coordinate system of each node. For each image I_h^{ij}, we use a binary mask M_h^{ij} to indicate the area that appears in the reconstruction results of both nodes. Each camera pose P is represented as $P = [R|T]$, where $R \in SO(3)$ is a rotation matrix and $T \in \mathbb{R}^3$ is a translation vector, is a world-to-camera (w2c)
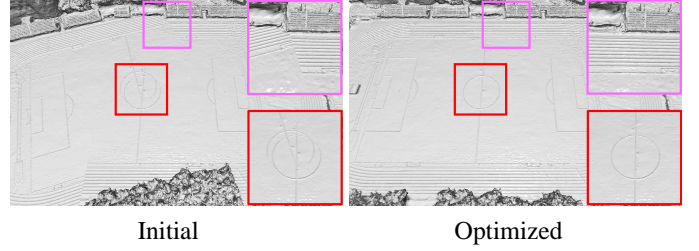

Initial   Optimized

Fig. 2. **Mesh Visualization for Optimization of Registration.** The left image shows misalignment caused by the initial registration, while the right displays the optimized result.

matrix, which denotes the transformation from the world coordinate system into the camera coordinate system:

$$\boldsymbol{x}_c = P\boldsymbol{x}_w, \tag{3}$$

where $\boldsymbol{x}_w \in \mathbb{R}^3$ is a point in the world coordinate system, and $\boldsymbol{x}_c \in \mathbb{R}^3$ is the corresponding point in the camera coordinate system.

Initialization for registration. The purpose of registration is to transform from one world coordinate system to another world coordinate system, corresponding to the training scenes of their respective nodes. We derive the initial registration primarily based on the fact : for any point $\boldsymbol{x}_w$ within the overlapping area of two adjacent nodes, its corresponding point in the camera coordinate system $\boldsymbol{x}_c$ is uniquely determined.

We denote that point $\boldsymbol{x}_w$ is $\boldsymbol{x}_w^k$ in node v_k $(k = i, j)$ in their respective local world coordinate systems. Hence, the corresponding points in the camera coordinate system are obtained after applying the respective w2c matrices for transformation among their common images:

$$P_h^i \cdot \boldsymbol{x}_w^i = P_h^j \cdot \boldsymbol{x}_w^j. \tag{4}$$

Additionally, we reevaluate the relationship between corresponding points from a registration perspective. Specifically, we seek the registration transformation $\mathcal{T}_{ij}^0$ for these two adjacent nodes to satisfy:

$$\boldsymbol{x}_w^i = \mathcal{T}_{ij}^0 \left(\boldsymbol{x}_w^j \right). \tag{5}$$

Therefore, in the local world coordinate system of node v_i, through the corresponding camera transformation, there will be:

$$P_h^i \cdot \boldsymbol{x}_w^i = P_h^i \cdot \mathcal{T}_{ij}^0 \left(\boldsymbol{x}_w^j \right). \tag{6}$$

These are two different conversion methods, but the points in the camera coordinate system obtained are uniquely determined. Thus, by observing Eq. (4) and Eq. (6), we attribute the registration transformation to the conversion of the two local world coordinate systems to the camera coordinate system. Then we should expect the following condition to be satisfied approximately:

$$P_h^i \cdot \mathcal{T}_{ij}^0 = P_h^j, \quad h = 1, ..., N_{ij}. \tag{7}$$

Notice that the transformation here becomes right multiplication. More importantly, in this way, the registration transformation can be added to the calculation graph of the optimization network through the step of generating the

camera origin and direction, and then can be optimized by volume rendering.

We rearrange these N_{ij} equations to get the following system of equations:

$$\begin{pmatrix} R_1^i & T_1^i \\ R_2^i & T_2^i \\ \vdots & \vdots \\ R_{N_{12}}^i & T_{N_{12}}^i \end{pmatrix} \cdot \mathcal{T}_{ij}^0 = \begin{pmatrix} R_1^j & T_1^j \\ R_2^j & T_2^j \\ \vdots & \vdots \\ R_{N_{12}}^j & T_{N_{12}}^j \end{pmatrix}. \quad (8)$$

This leads to an overdetermined system of equations, which can be solved in a least-squares manner. Therefore, we obtain the initial registration transformation expressed in matrix form:

$$\mathcal{T}_{ij}^0 = \begin{pmatrix} R_{ij}^0 & T_{ij}^0 \\ 0 & s_{ij}^0 \end{pmatrix}. \quad (9)$$

$R_{ij}^0 \in SO(3)$ is a rotation matrix, $T_{ij}^0 \in \mathbb{R}^3$ is a translation vector, and $s_{ij}^0 \in \mathbb{R}$ is a scaling factor. Note that we introduce a scaling factor here, because the SFM system [9] cannot distinguish whether the scales of the world coordinate systems of the two nodes are consistent. Therefore, the transformation between the two world coordinate systems involves not only a rigid transformation, but also a scale expansion. On the other hand, it should be noted that the transformation in Eq. (5) will involve the operation of homogeneous coordinates.

Optimization for registration. It should be noted that the SFM system is not completely accurate. Additionally, the least-squares solver further accumulates and amplifies errors. These factors lead to the inaccuracy of the initial registration matrix, causing misalignment between two nodes after initial registration, as shown in Figure 2 and Figure 12. Hence, we further need to optimize the registration matrix to improve the alignment.

It has been demonstrated in [45], [46] that volume rendering can be used to optimize the intrinsic and extrinsic camera parameters. Our approach to optimizing the registration matrix follows the same principle. However, unlike these works, we use the rendering results within the registered node as supervision rather than ground-truth images.

For the registration of two adjacent nodes, compared to directly optimizing the transformation matrix, we choose to optimize a variation $\Delta \mathcal{T}_{ij}$ with respect to the initial transformation $\mathcal{T}_{ij}^0$. Then the resulting registration is computed as:

$$\mathcal{T}_{ij} = \mathcal{T}_{ij}^0 + \Delta \mathcal{T}_{ij}. \quad (10)$$

Assume the neural radiance fields of both nodes, C_i and C_j, have been well trained and fixed. We then use these two poses, the transformed camera pose $P_h^i \cdot \mathcal{T}_{ij}$ and the corresponding pose P_h^j in v_j, for volume rendering in C^j and obtain the following colors:

$$\begin{aligned} c_h^{ij} &= C_j(P_h^i \cdot \mathcal{T}_{ij}), \\ c_h^{jj} &= C_j(P_h^j). \end{aligned} \quad (11)$$

Theoretically, the colors within the common region indicated by the mask M_h^{ij} should be identical. Therefore, we employ a color loss to optimize the registration matrix:

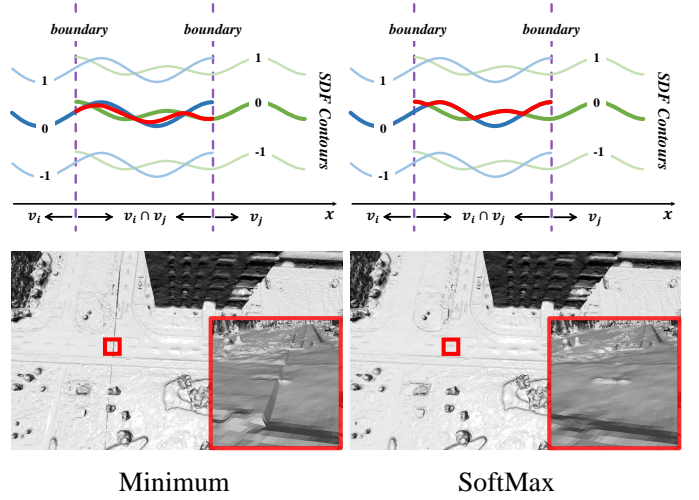$$\Delta \mathcal{T}_{ij} = \arg\min \sum_h \|(c_h^{ij} - c_h^{jj}) \cdot M_h^{ij}\|. \quad (12)$$

Fig. 3. **Blending.** The top images depict simplified SDF contours within a vertical y-axis section. The left image shows a visible seam caused by directly taking the minimum value, while the right displays the smoothed result achieved through Softmax weighting.

Note that we use the rendering results rather than the ground-truth images as supervision. This is because various factors may hinder accurate learning in certain regions during the training of neural radiance fields. Thus, using the ground truth as supervision may lead to incorrect optimization directions.

In detail, we convert the rotation matrix into an Euler angle representation $E_{ij}^0 = (\phi_{ij}^0, \theta_{ij}^0, \psi_{ij}^0) \in \mathbb{R}^3$. Therefore, the variation $\Delta \mathcal{T}_{ij}$ is represented using the change in Euler angles ΔE_{ij}, the change in translation ΔT_{ij}, and the change in scaling factor Δs_{ij}. Then the resulting registration is derived from the updated Euler angle, translation, and scaling factor computed as:

$$\begin{aligned} E_{ij} &= E_{ij}^0 + \Delta E_{ij}, \\ T_{ij} &= T_{ij}^0 + \Delta T_{ij}, \\ s_{ij} &= s_{ij}^0 + \Delta s_{ij}. \end{aligned} \quad (13)$$

In our implementation, we utilize the Adam optimizer [47] to solve the optimization problem. The initial learning rate is set to $lr_0 = 5 \times 10^{-5}$, and the decay function for the learning rate follows the formula $lr_i = lr_0 \times 0.8^{\frac{i}{100}}$, where i represents the iteration index. We conduct a total of 5K iterations, with 2048 rays per iteration.

3.4 SDF Blending

After we transform the local SDFs via registration, all SDFs are defined in a common global coordinate system. By construction, these SDFs have overlaps in some areas. However, as reconstruction errors are inevitable, the SDFs may not fully coincide in their overlapping regions. As a result, we need to combine these SDFs in their overlapping regions to derive a consistent global SDF for the whole scene.

Since each SDF represents a part of the scene, one simple solution is to model this as a Boolean union operation that joins the overlapping SDFs. For two objects represented by two SDFs f_i and f_j respectively, it is well known that the

object resulting from the union operation is represented by the following SDF [48], [49], [50]:

$$f_{ij}(\boldsymbol{x}) = \min(f_i(\boldsymbol{x}), f_j(\boldsymbol{x})). \tag{14}$$

However, as illustrated in Figure 3, blending the SDFs in this direct manner may result in a noticeable anomalous seam across the boundary of the common region. This occurs due to the abrupt transition between the two local SDFs at the boundary of the overlapping regions, leading to a discontinuity of the blending SDF. To avoid such issues, we instead blend the two SDFs in their overlapping regions:

$$f_{ij}(\boldsymbol{x}) = \omega_{ij}^i f_i(\boldsymbol{x}) + \omega_{ij}^j f_j(\boldsymbol{x}). \tag{15}$$

Here ω_{ij}^i and ω_{ij}^j are softmax based weights:

$$\omega_{ij}^k(\boldsymbol{x}) = \frac{e^{-\beta d_k(\boldsymbol{x})}}{e^{-\beta d_i(\boldsymbol{x})} + e^{-\beta d_j(\boldsymbol{x})}}, \quad k = i, j, \tag{16}$$

where β is a user-specfied parameter (set to $\beta = 10$ in our experiments), and d_k represents the distance from the point $\boldsymbol{x}$ to the boundary of node v_k. This scheme increases the relative weight for one SDF as the point $\boldsymbol{x}$ gets closer to its node boundary, thus smoothing blending the two SDFs and avoiding the undesirable seams (see Figure 3).

After the blending, we obtain a global SDF function for the whole scene, ready for downstream processing such as surface extraction with Marching Cubes [51].

3.5 Applications

Our representation offers several benefits compared with existing neural implicit surface reconstruction methods. First, this divide-and-conquer approach utilizes an MLP to reconstruct the SDF in a localized region rather than the whole scene, enhancing its capability to capture high-fidelity details and achieve high-quality surface reconstruction. Second, the graph structure makes the reconstruction process scalable, as we can easily handle larger-scale scenes by adding more nodes while maintaining the computational cost per node and the high-quality of each node. Section 4.2, 4.3 and 4.4 demonstrate some examples of high-quality and scalable reconstruction using our method.

In addition, the independence of nodes offers flexibility for various downstream applications. In the following, we illustrate the versatility of our method through two additional and practical applications—texture generation and scene editing. Detailed examples of these applications are provided in Section 4.5.

Texturing. Our method can not only extract surface geometry but also compute textures for the surface mesh. To do so, we first extract the corresponding mesh for each node separately and perform parameterization to obtain its texture domain. Then, we initialize the textures separately using the color radiance field within that node. Since the textures are derived from surface rendering and the neural implicit reconstruction relies on volume rendering, there are inherent differences between the two representations, thus the initial texture may contain noises. To address this, we employ surface rendering to optimize the texture map of each node. Specially, for node v_k, utilizing multi-view images $I_l^k, l = 1, ..., N_k$, within this node as supervision, we

obtain the rendering images $\hat{I}_l^k$ through rasterization and optimize the texture map using a photometric loss:

$$\mathcal{L}_{color} = \|\hat{I}_l^k - I_l^k\|. \tag{17}$$

Finally, we merge the meshes and textures [52] from multiple nodes to obtain a large scene's mesh with high-fidelity textures.

Editing. Another important application of 3D surface reconstruction is the editing of objects within the scene. Through our method, if we consider a single object in the scene as a node, then the independence between nodes provides the flexibility to edit its position in the scene. Specifically, we can adjust the node's registration transformation relative to the global coordinate system, such as adding a certain translation or rotation, and then use our SDF blending to smooth the new boundaries resulting from this modification. In this way, we can manipulate the position and orientation of the objects and move them within the scene.

4 EXPERIMENTS

To demonstrate the effectiveness and practicality of our method, we conduct a variety of experiments, including high-quality reconstruction of different categories of objects or scenes, scalable reconstruction of large scenes, and ablation studies. At the same time, we also show its applications in texturing and shape editing.

4.1 Experimental Configuration and Details

Datasets. We evaluate our method's capability in high-precision reconstruction using several datasets: leftmargin=*

- The NeRF Synthetic Datasets [4] contains rendered RGB images created through Blender, and we use the "Lego" object. The Lego dataset contains 100 images with a resolution of 800×800, as well as corresponding masks and ground-truth camera poses.
- The BlendedMVS datasets [53] is a large-scale MVS dataset for generalized multi-view stereo networks. Among them, the "Jade" object contains 58 multi-view images with a resolution of 768×576, masks, and camera poses. In addition, we chose several large scenes at a higher resolution of images, 2048×1536, for experiments.
- The Actors-HQ Dataset [54] is a high-fidelity dataset of clothed humans in motion. We use one subject for evaluation, extracting the first frame from each video to create a multi-view image dataset. Thus, one of the human body datasets comprises 160 images, each with a resolution of 4090×2992 or 2992×4090, accompanied by corresponding camera poses and masks.
- We also created a large-scale, named *Sub-Campus* in the following, which primarily consists of four adjacent buildings in a campus. We used a DJI M300 RTK drone equipped with a Zenmuse P1 camera to capture a total of 2273 images using oblique photography. These images were then classified based on the different buildings photographed, resulting in four sub-datasets. The corresponding number of
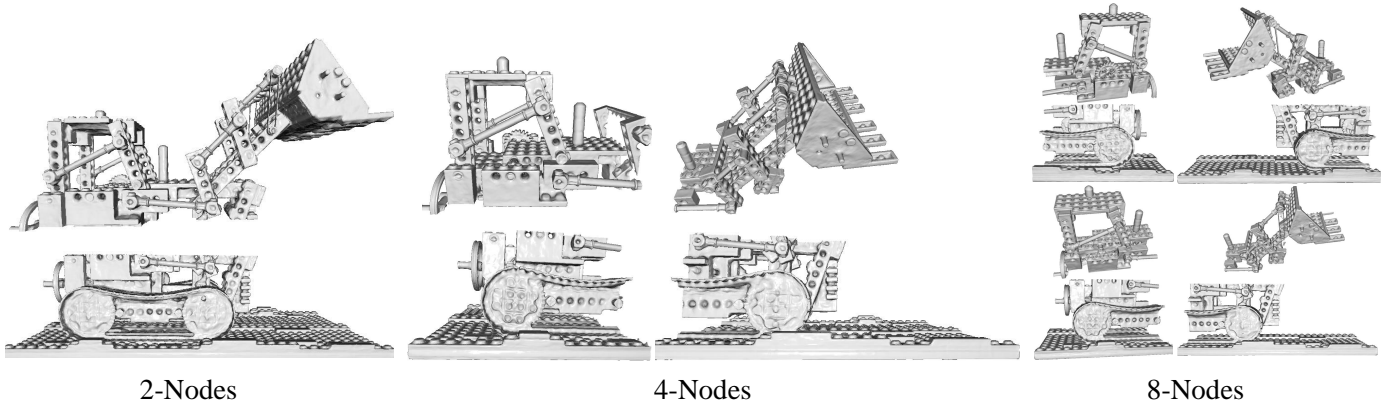
Fig. 4. **Lego Nodes Division.** From left to right are the node divisions of 2-nodes, 4-nodes and 8-nodes.
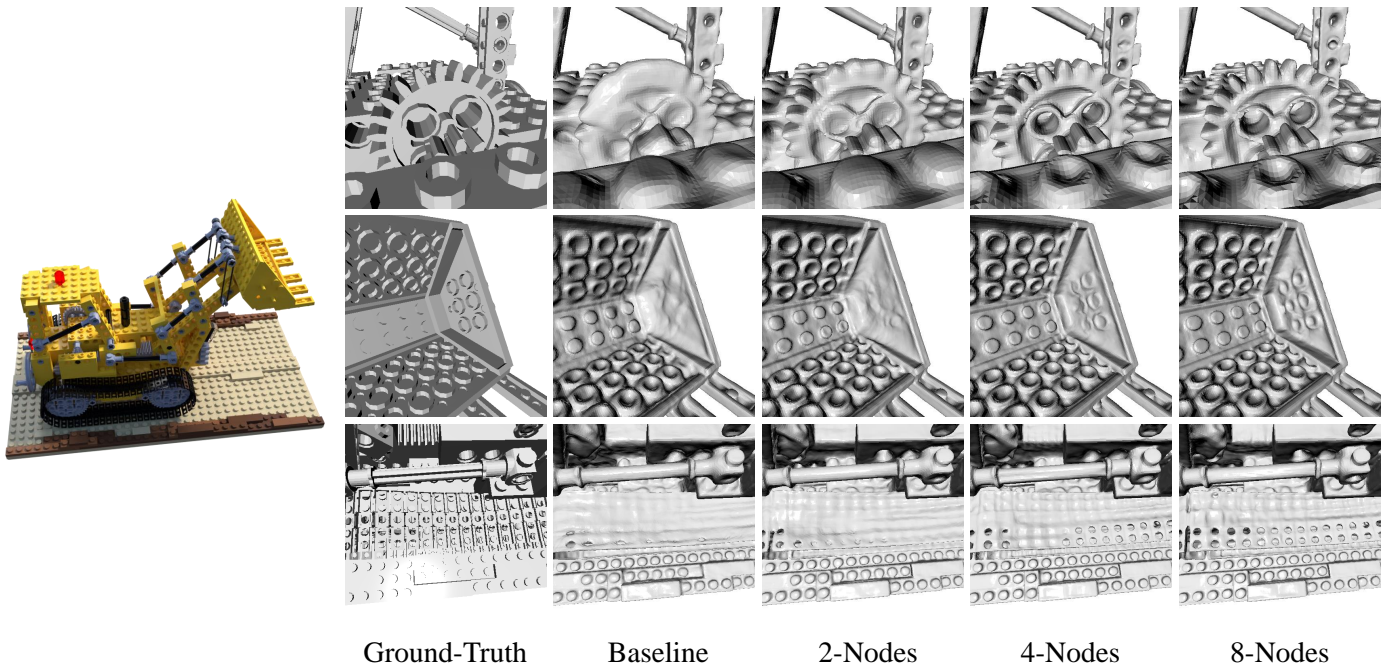

Fig. 5. **Visualization of High-Quality Reconstruction Results for Lego.** The right side is the comparison, demonstrating that our representation progressively reconstructs finer details.

images for each sub-dataset is presented in Table 5. The original resolution of the images is $8192{\times}5460$, and we performed down-sampling to reduce the image resolution to 1024×682 for reconstruction.

In addition, for the evaluation of scalable large scene reconstruction, we constructed a larger dataset called *Campus* using the same equipment as the Sub-Campus dataset. This dataset covers a larger campus area of about $1200{\times}800$ m^2 and comprises 5973 images obtained through oblique photography. Specially, this dataset was collected progressively, beginning from a corner of the scene and gradually expanding to cover the entire area block by block, the specific process as shown in Figure 10. Then, based on the division made during collection, we partitioned the entire dataset into 25 sub-datasets. Similar to the previous Sub-Campus dataset, the captured images were down-sampled from $8192{\times}5460$ to $1024{\times}682$ resolution.

Evaluation metrics. We calculate the Chamfer distance [55]

TABLE 1
Quantitative Evaluation on the Lego Dataset. CD represents the Chamfer distance from the ground-truth mesh to the reconstructed mesh. The last column represents the mean of the SDF absolute values corresponding to points uniformly sampled on the ground-truth surface. The optimal result is indicated in bold.

	CD↓ (10^{-4})	F-score↑ (threshold=$5e^{-4}$)	SDF↓ (10^{-3})
Baseline	5.209	0.8445	8.779
2-nodes	4.743	0.8621	7.348
4-nodes	3.877	0.8826	7.054
8-nodes	**2.836**	**0.8972**	**6.749**

and the F-score [56] if the ground-truth mesh is available, such as the Lego dataset. For the BlendedMVS dataset, we utilize the provided textured mesh reconstructed using the MVS method [53] as the ground truth. For the Actors-HQ dataset, we utilize the provided mesh of the first frame as the ground truth, which comes from the frame-

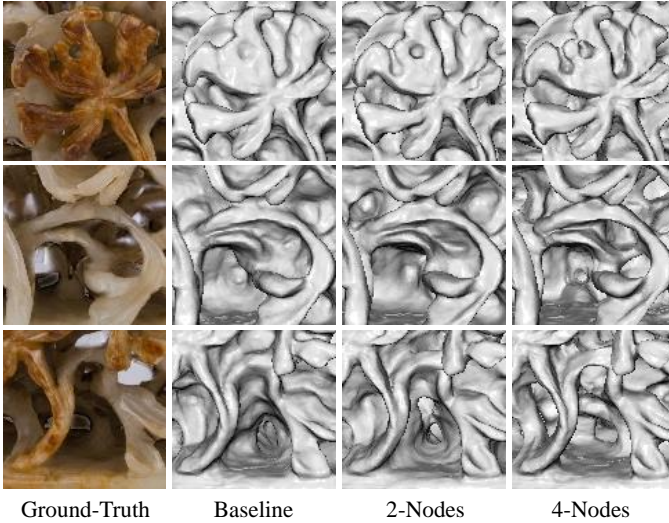Ground-Truth   Baseline   2-Nodes   4-Nodes

Fig. 6. **Visualization of High-Quality Reconstruction of Jade.** It can be seen that as the number of nodes increases, the carvings and hollowing out of the jade are gradually reconstructed and refined.

TABLE 2
Quantitative Evaluation on the Jade Dataset. The optimal result is indicated in bold.

	CD↓ (10^{-4})	F-score↑ (threshold=$5e^{-4}$)	SDF↓ (10^{-3})
Baseline	2.507	0.7801	8.527
2-nodes	2.431	0.7847	7.548
4-nodes	**0.958**	**0.7873**	**4.136**

by-frame mesh reconstructed by applying Humanrf [54] at 2x resolution. Additionally, to mitigate the impact of errors introduced by Marching Cubes during mesh extraction, we uniformly sample a specific number of points on the ground-truth mesh and then calculate the average absolute SDF values of these points within the reconstructed SDF. For the large scene datasets Sub-Campus and Campus where no ground-truth mesh is available, we qualitatively visualize and magnify the reconstruction details for evaluation. Furthermore, we report rendering quality metrics—including Peak Signal to Noise Ratio (PSNR), Structural Similarity (SSIM) [57], and Learned Perceptual Image Patch Similarity (LPIPS) [58]—both before and after the registration optimization to illustrate the effectiveness of our approach.

Implementation details. We run our experiments on a workstation with an Intel Xeon E5-2690 v3, an NVIDIA GeForce RTX 3090, and 128GB of RAM. We use the open-source implementation of NeuS [6][1] as well as the open-source project SDFStudio [59][2] for reconstructing SDF at a node and for the whole scene. Camera poses are estimated using COLMAP [9], [10]. Our initial registration is computed using NumPy and takes no more than 1 second. The registration optimization is performed using PyTorch with $5k$ iterations and 2048 rays per iteration, and takes about 15 minutes on average. More implementation details can be found in the supplementary materials.

1. https://github.com/Totoro97/NeuS
2. https://github.com/autonomousvision/sdfstudio

4.2 Validation of Effectiveness

We aim to first verify the effectiveness of the proposed graph-based SDF representation from two aspects:

- Firstly, we demonstrate the inherent limitations in the expressive capacity of a single MLP. In other words, as the complexity of objects or scenes increases, a single MLP may fail to reconstruct refined geometry effectively.
- Secondly, we validate the feasibility and effectiveness of node partitioning. Specifically, by narrowing down the spatial range expressed by a single MLP, we can improve the quality and accuracy of reconstruction.

Baseline. To comprehensively validate these assertions, we employ NeuS [6] as our baseline method, representing the most fundamental method within the neural SDF domain. This choice serves two primary purposes: Firstly, NeuS is the foundational representation for subsequent improvements in neural implicit surface reconstruction, and its validation is crucial for affirming the general feasibility of future enhancement efforts. Secondly, NeuS does not rely on any additional prior information or regularization loss during training, thus ensuring that extraneous factors do not influence our experimental verification results. Even though there are some improved SDF models that can achieve better performance than NeuS on the same scene, these single-MLP models still run into the same issues eventually if the scene becomes large and complex enough. Therefore, this issue is universal for single-MLP approaches regardless of the model used. In the following experiments, we utilize the Lego and Jade datasets as reconstruction targets, representing two different datasets of synthetic and real scenes respectively. Therefore, the baseline is reconstructing the entire object using only a single MLP with NeuS.

Our method. To apply our method, we partition the bounding box along axis directions into overlap cuboids, resulting in a graph with 2, 4, and 8 nodes, respectively, as shown in Figure 4. For the situation where Lego is divided into upper and lower nodes, the bounding box range on the z-axis is $[-0.35, 1.01]$, and we directly divide it into $[-0.35, 0.25]$ and $[0.0, 1.01]$. This uneven division accounts for the fact that most parts of the scene are located at the lower part of the bounding along the z-axis. The overlapping areas of the two nodes account for 41.67%, 24.75%, and 18.38% of the lower node range, upper node range, and global range, respectively. For the case involving four nodes, we build upon the aforementioned division into two nodes and further partition the bounding box range along the y-axis, $[-1.15, 1.15]$, into $[-1.15, 0.4]$ (representing the car rear area) and $[0.0, 1.15]$ (representing the front bucket area). The overlapping area constitutes 25.81%, 34.78%, and 17.39% of the global range along the tail, front, and y-axis, respectively. Similar to the division along the z-axis, the uneven division here accounts for the distribution of the scene within the bounding box. For the scenario with eight nodes, we build upon the previous division into four nodes and further partition the bounding box range along the x-axis ($[-0.64, 0.64]$) into $[-0.64, 0.10]$ and $[-0.10, 0.64]$. The overlapping area constitutes 27.02% and 15.63% of one
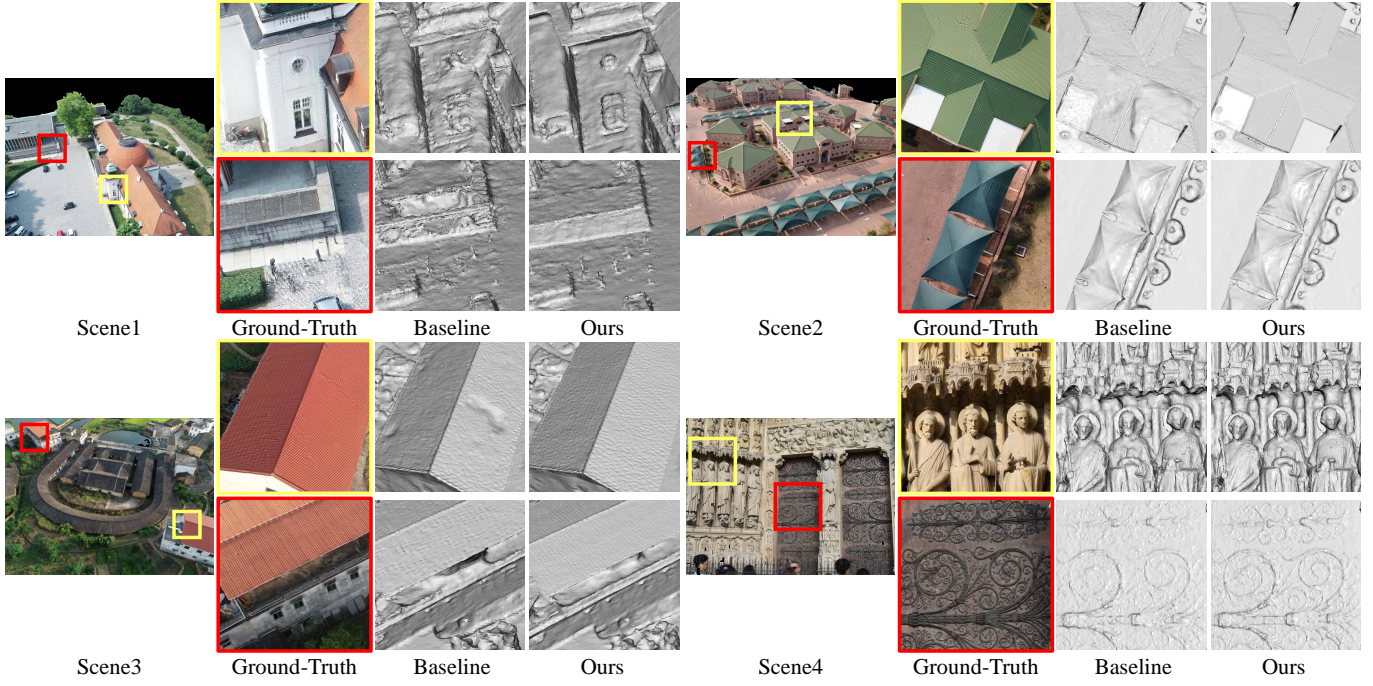
Fig. 7. **Visualization of High-Quality Reconstruction of BlendedMVS Datasets.** By magnifying and comparing local details, our method achieves higher-quality reconstruction and recovers more high-fidelity details.

TABLE 3
Quantitative Evaluation on the BlendedMVS Datasets. For F-score, the distance threshold is $5e^{-4}$. The optimal result is indicated in bold.

		CD↓ (10^{-6})	F-score↑	SDF↓ (10^{-4})
Scene1	Baseline	3.728	0.9797	7.167
	4-nodes	**2.424**	**0.9894**	**4.459**
Scene2	Baseline	5.923	0.9993	8.421
	4-nodes	**4.383**	**0.9999**	**6.372**
Scene3	Baseline	7.565	0.9974	6.107
	4-nodes	**5.847**	**0.9981**	**4.893**
Scene4	Baseline	8.389	0.7120	9.897
	4-nodes	**6.017**	**0.7670**	**6.320**

TABLE 4
Quantitative Evaluation on the Human Body. The optimal result is indicated in bold.

	CD↓ (10^{-6})	F-score↑ (threshold=$5e^{-4}$)	SDF↓ (10^{-4})
Baseline	3.588	0.9226	8.491
4-nodes	**3.403**	**0.9998**	**8.450**

side and the global range of the x-axis, respectively. For the Jade dataset, we directly divide its bounding box into overlapping cuboids of the same size along the axes, and the overlapping areas account for 20% of the global range. For a fair comparison, we use the same reconstruction model and configuration for the baseline and for each node. Moreover, the reconstructed surfaces are extracted via Marching Cubes using the same settings.

Table 1 and 2 show quantitative results from our method and the baseline that uses a single MLP for reconstruction. Figures 5 and 6 show qualitative visualization of the results.
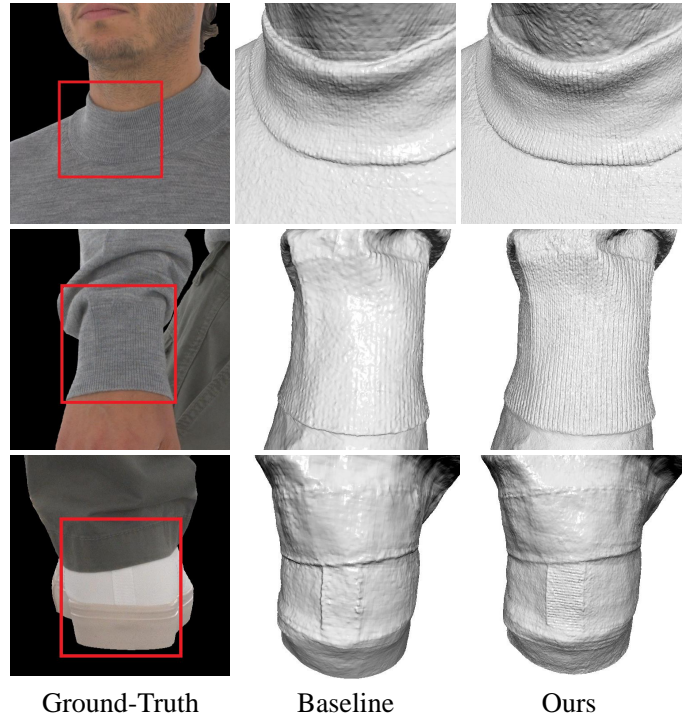

Ground-Truth   Baseline   Ours

Fig. 8. **Visualization of High-Quality Reconstruction of Human Body.** By the magnified comparison of local areas, our method recovers more details on clothes and shoes than the overall reconstruction.

It can be seen that the reconstruction quality improves as the number of nodes increases. Notably, whether it is the concave and convex structures of the knobs in Lego, the outlines of the gears, and the holes in the tires, or the carvings and hollowing out of the jade, they are gradually
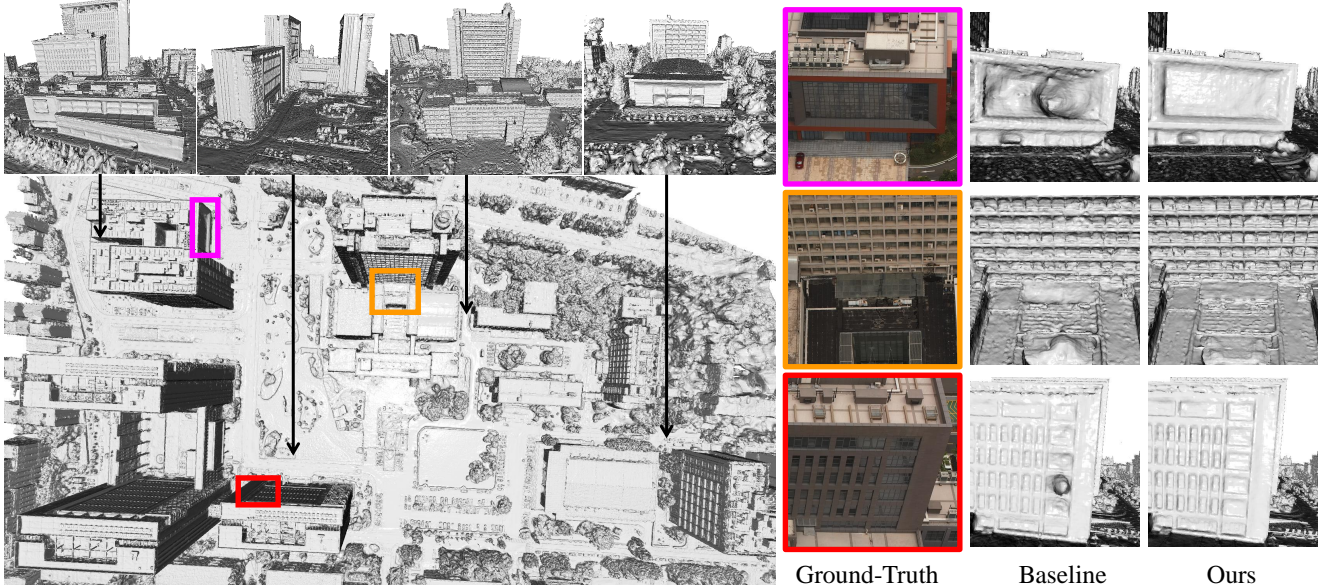
Fig. 9. **Visualization of High-Quality Reconstruction of Sub-Campus with 4 Nodes.** On the left side, we present an overview of the object-based scene Sub-Campus reconstructed using our method, featuring the four objects utilized for node partitioning. A detailed comparison of local results on the right reveals a significant advantage in terms of reconstruction quality for our method.

reconstructed and delineated with increasing detail. Therefore, both quantitative and qualitative results demonstrate that the expressive capability of a single MLP is limited. In comparison, as the number of nodes increases, our method gradually improves the quality of the results and captures the fine details in local regions.

4.3 High-Quality Reconstruction

In the previous section, we verified two fundamental facts and demonstrated the feasibility of our method through designed experiments. In this section, in addition to the high-precision reconstruction of Lego and Jade, we conduct further experiments to demonstrate that our method can achieve high-quality reconstruction. To illustrate the effectiveness of our approach, we compare it with a baseline that reconstructs the whole scene using a single MLP. Similarly, for a fair comparison, we use the same reconstruction model for the baseline and for our nodes, including, of course, utilizing identical settings for Marching Cubes.

Human body. For the scene of a clothed human body, we divide it into four nodes corresponding to the head, the torso, and two legs, respectively. We use the Bakedangelo model from SDFStudio for reconstruction. This model combines BakedSDF [60] with Neuralangelo [8] to achieve high-quality reconstruction, providing a strong baseline of single-MLP reconstruction for our comparison.

Figure 8 qualitatively visualizes the results from the baseline and our method. Our method captures a higher level of detail, particularly the fine stripe patterns on the clothing and the shoes. The numerical results in Table 4 further verify that our approach achieves high-quality reconstruction.

BlendedMVS. We select multiple large-scale scenes from the BlendedMVS datasets to further evaluate the advantage of our method in high-quality reconstruction. Due to the

TABLE 5
The Information of Nodes in Sub-Campus. We display the number of images for each node, the number of shared images between adjacent node, and the COLMAP running time for each node and baseline (reconstruction as a whole), respectively.

	Node v_1	v_2	v_3	v_4	Baseline
Total Images	893	561	782	522	2273
Shared Images with v_1	-	140	82	87	-
COLMAP Runtime	1h47m	58m	1h33m	44m	14h25m
	Total : 5h8m				

relatively narrow range of the z-coordinates in the scene, we conduct partition only along the x-axis and y-axis, dividing the scene into four nodes with overlapping areas. We still utilize the more robust baseline, Bakedangelo, for both the single-MLP reconstruction and each node.

Figure 7 presents the visual comparison results. It can be seen that, despite the basline is the sota method, our method still achieves higher-quality reconstruction in large-scale scenes, particularly in capturing high-fidelity details such as the flatness of building facades, the integrity of steps and edges, the smoothness of rooftops, and the three-dimensionality of reliefs. Moreover, the numerical comparison results in Table 3 further ddemonstrate the substantial enhancement in reconstruction accuracy achieved by our method.

Sub-Campus. We partition the scene into four nodes containing different subsets of buildings. We use COLMAP for independent estimation of camera poses for each node, as well as camera pose estimation for the baseline. Table 5 displays the number of images for each node, the number of shared images between adjacent nodes, and the COLMAP running time. The SDF reconstruction is then performed using the Bakedangelo model.
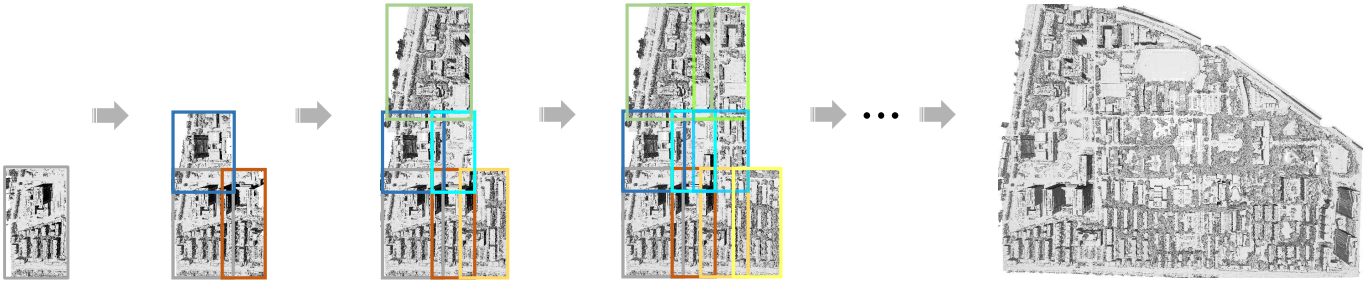
Fig. 10. **Process of data collection and Scalable Scene Reconstruction of Campus Dataset.** The image above illustrates the process of reconstructing the scalable large scene, Campus, based on the direction and extent of data collection.

The Reconstructed Complete Mesh   Rendering with Texture

Fig. 11. **Visualization Results of Scalable Scene Reconstruction on Campus Dataset.** A comprehensive view of meshes and textures is presented above. Below is a partial close-up comparison. The top row displays the ground-truth image, the middle row showcases the rendered image using our texture, and the bottom row exhibits the mesh image.

From Table 5, it is worth noting that the COLMAP processing takes 14 hours and 25 minutes for the baseline, but only 5 hours and 8 minutes for our approach. This shows our significant advantage in efficiency when dealing with a large amount of images. Figure 9 shows some qualitative visualization of the results. Our method correctly captures the building facade shapes, whereas the baseline produces erroneous results in some areas due to limited representational power. Additionally, our result shows clearer windows outlines with improved accuracy. These observations verify the capability of our method in handling large-scale scenes.

4.4 Scalable Large Scene Reconstruction

To verify the scalability of our approach, we further test our method on our Campus dataset, which contains 5973 oblique photography images covering an area of about 1200m×800m. At such a large scale, a single-MLP approach becomes impractical in both data processing and SDF-based reconstruction.

According to the flight path employed during the data collection process, we designate the local area collected at the beginning as the starting node. From this node, we gradually expand the collection and nodes to encompass the entire scene. The distribution of nodes and the expansion

TABLE 6
Rendering Metrics Before and After Registration Optimization. Assume that node v_i is registered to v_j, for shared images between the two nodes, "**target**" represents the rendering metrics achieved after training the neural radiance fields of v_j; "**initial**" indicates the rendering metrics in v_j after transforming the camera pose in v_i using the initial registration; "**final**" denotes the rendering metrics obtained after applying the optimized registration. We expect the optimized "**final**" to reach the "**target**" metrics, which means that the registration is aligned. Each set of three columns constitutes a distinct experiment, illustrating the registration of one node to another.

|	Edge 1			Edge 2			Edge 3			Edge 4			Edge 5		
	target	initial	final	target	initial	final	target	initial	final	target	initial	final	target	initial	final
PSNR↑	25.94	21.22	25.35	27.04	22.89	26.98	27.95	21.46	27.59	24.52	19.28	24.21	27.82	20.87	27.20
SSIM↑	0.803	0.518	0.781	0.845	0.647	0.836	0.856	0.605	0.856	0.839	0.485	0.824	0.860	0.433	0.830
LPIPS↓	0.096	0.116	0.096	0.078	0.095	0.078	0.069	0.091	0.069	0.075	0.109	0.075	0.064	0.101	0.064
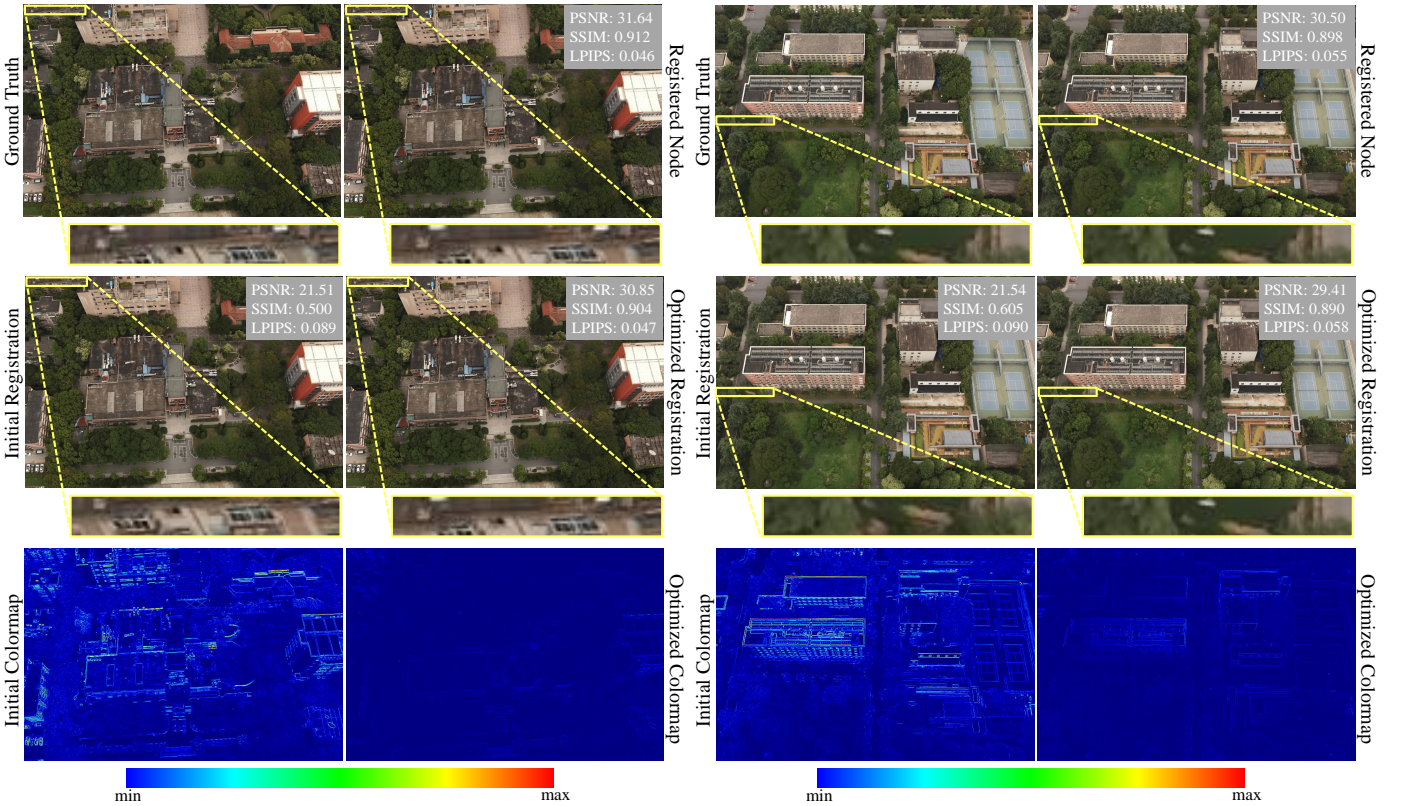
Fig. 12. **Rendering Evaluation for Optimization of Registration.** We present rendering visualizations and metrics for the registered node, initial registration, optimized registration, and a colormap indicating errors relative to supervision (registered node). The partial magnification and rendering metrics reveal significant deviations in the initial registration, as clearly depicted in the figure as an upward bias relative to the reference. Through our optimization method, we effectively rectify these errors.

strategy are depicted in Figure 10. Therefore, based on the collection path and data distribution, we partition the whole scene into a total of 25 nodes, each containing an average of 540 images. For node pairs with overlapping regions, the approximate count of shared images in their intersection is around 100. Notably, this partition process is automated and does not require manual intervention. At each node, the camera poses are independently estimated using COLMAP, and the SDF is reconstructed using the Bakedangelo model.

We use the coordinate system of the starting node as the global coordinate system and propagate it along the minimum spanning tree of nodes. Consequently, the registration process involves 24 edges. Due to space limitations, Table 6 only presents the rendering metrics of five edges before and after registration optimization. It can be seen that

the rendering metrics of the initial registration (the "initial" column) are significantly poorer compared to the rendering metrics of the registered node (the "target" column), indicating a considerably larger error in the initial registration. By applying our method, we note a substantial improvement in the rendering metrics after optimization (the "final" column), which essentially approaches the metrics of the registered node. To more clearly demonstrate the necessity of registration optimization, we illustrate the changes in the mesh of two adjacent nodes before and after registration optimization in Figure 2. The initial registration pose exhibits significant abnormal deviation and misalignment in the relative positions of the meshes, and after optimization, the two meshes achieve perfect alignment. Furthermore, we present the rendering-based visual comparison before
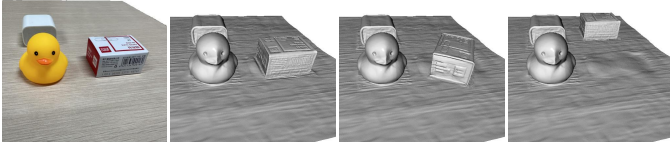
Fig. 13. **Editing Results.** From left to right: RGB images, original mesh, edited mesh examples 1 and 2.

and after registration optimization in Figure 12. The error color map shows a significant misalignment after the initial registration, which is substantially reduced after the registration optimization. Overall, the comparison of all these numerical and visual results demonstrates the necessity and effectiveness of our registration optimization method.

After registration optimization, we conduct blending for all the SDFs of the 25 nodes according to the method described in Section 3.4. Subsequently, we employed Marching Cubes to extract the mesh of the entire scene. The complete mesh of the reconstructed scene and enlarged details in localized areas are depicted in Figure 11. It can be seen that our method faithfully reconstructs the geometry as well as the texture of the whole scene. This demonstrates the effectiveness of our method in handling urban-scale scenes that are intractable for existing single-MLP reconstruction methods.

4.5 Additional Application Results

In addition to the experiments and results shown above, we will show two applications proposed in Section 3.5. Firstly, we have generated texture maps for the previously reconstructed scalable large scene, as illustrated in Figure 11. We utilized Xatlas[3] for parameterization and then employed the surface rendering "SoftPhongShader" model with "AmbientLights" provided by PyTorch3D [61] for rendering. More details regarding rasterization settings and shader parameter configuration can be found in the appendix. The results demonstrate a high level of fidelity, which further substantiates the practical value of our method. Secondly, using the NeuS model, we apply rotation and translation to the registration transformation of a node within the scene to achieve the effect of editing, as illustrated in Figure 13. This once again illustrates the versatility and practicality of our method.

4.6 Ablation Study

We conduct an ablation study from three aspects: (1) examining the impact of registration optimization, (2) comparing the results with and without SDF blending, and (3) exploring the influence of network parameter count on model performance.

Registration optimization. We first focus on the impact of registration optimization on the accuracy of the reconstruction. We present a visual comparison before and after registration optimization at the mesh level in Figure 2, a rendering-level visual comparison in Figure 12, and more numerical comparison of additional rendering metrics in
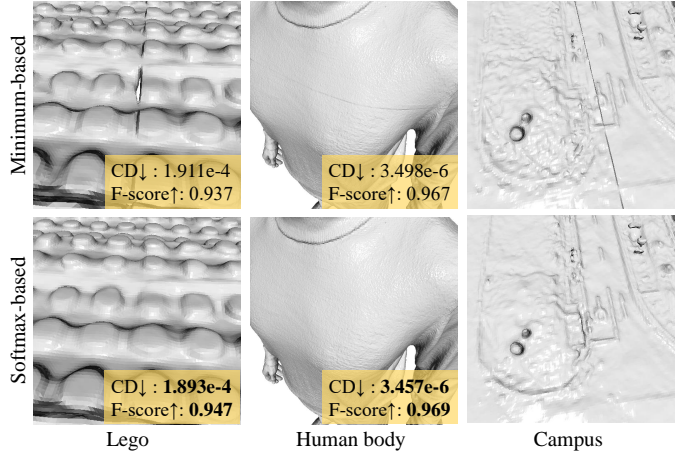
3. https://github.com/mworchel/xatlas-python

Fig. 14. **Ablation Study of SDF Blending.** Quantitatively evaluate the precision of the mesh after minimum-based and softmax-based SDF blending in the common area respectively. Clearly, our proposed softmax-based blending not only eliminates the seam problem but also maintains the accuracy of the reconstruction.

TABLE 7
Ablation Study on Parameter Count. For the F-score, the distance threshold is $5e^{-4}$. "1/4 layers" and "1/4 hidden" represents reducing the baseline network to one-fourth of the original size and then applying our method. "×4 iterations" means increasing the number of iterations. "×4 layers" and "×4 hidden" indicate quadrupling the baseline network and correspondingly the number of iterations. The optimal results are indicated in bold.

	CD↓ (10^{-4})	F-score↑	SDF↓ (10^{-3})
1/4 layers	4.953	0.8524	8.632
1/4 hidden	5.777	0.8115	8.936
Baseline	5.209	0.8445	8.779
×4 iterations	5.048	0.8547	8.252
×4 layers	5.135	0.8595	8.452
×4 hidden	5.207	0.8551	8.564
Ours 4-nodes	**3.877**	**0.8826**	**7.054**

Table 6. These results demonstrate significant errors in the initial registration, highlighting the necessity and effectiveness of the registration optimization method we proposed.

SDF blending. The comparison in Figure 3 shows that the minimum-based blending, according to the SDF definition, results in visible seams, whereas our proposed softmax-based blending method effectively eliminates this issue. The presence of seams highlights the necessity for SDF blending. At the same time, we quantitatively evaluate the reconstruction quality after blending in the common area, as shown in Figure 14. It is evident that softmax-based blending not only effectively resolves the seam problem, but also addresses the issue of poor reconstruction quality at the boundaries of each node, (the boundary of a node being a relatively internal position compared to adjacent nodes). Consequently, the reconstruction quality is not compromised due to SDF blending.

Parameter count. The expressiveness of the implicit representation is directly related to the parameter count, which depends on the network's layer depth and the number of neurons per layer. To ensure a fair comparison and further
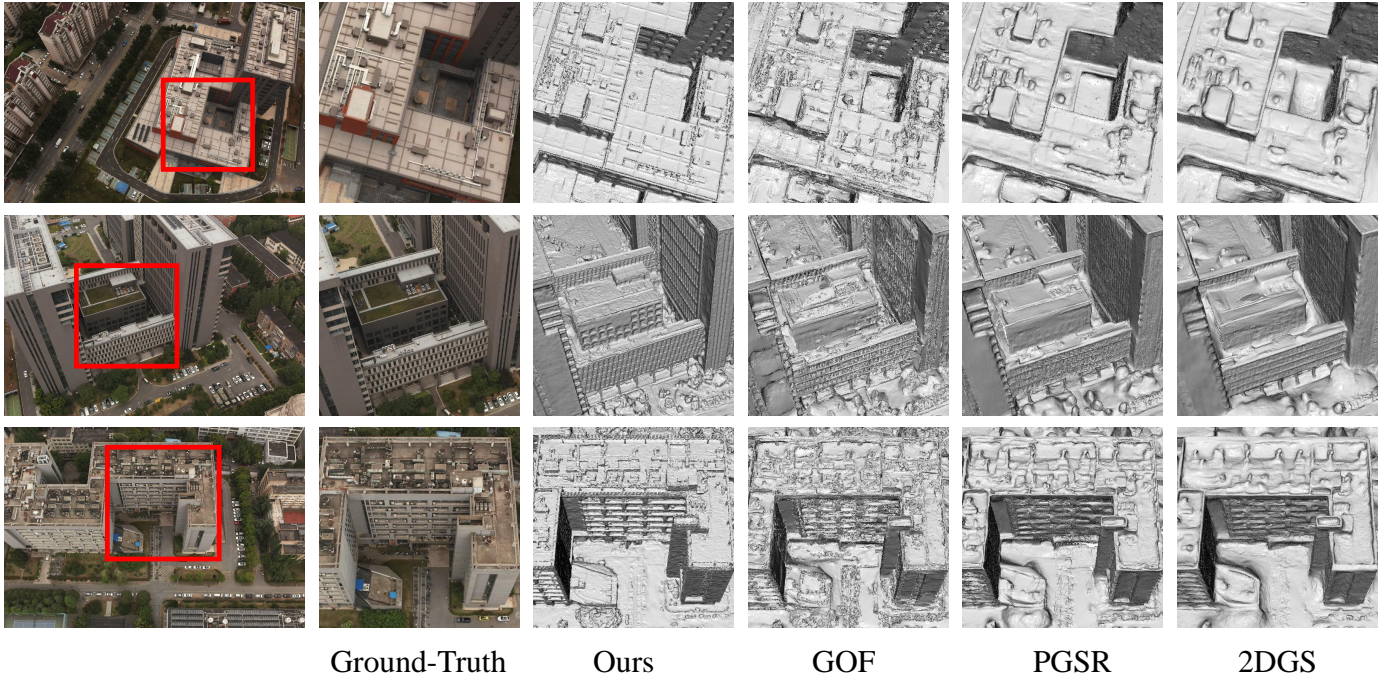
| Ground-Truth | Ours | GOF | PGSR | 2DGS |

Fig. 15. **Visualization Results of Sub-Campus Dataset Compared with Other Representation.** We compare our method with some novel and state-of-the-art geometric surface reconstruction methods based on 3DGS [62], including Gaussian Opacity Fields (GOF) [63], PGSR [64], and 2D Gaussian Splatting (2DGS) [65]. It can be observed that our method has better reconstruction ability, and even the current novel 3DGS-based surface reconstruction methods also face limitations in expressive capability when dealing with large-scale scene.

demonstrate the effectiveness of our method, we conduct an ablation study on the number of network parameters. We utilize the high-precision Lego reconstruction based on 4 nodes in Section 4.2 for the comparison. This study is divided into two aspects. First, to verify that simply increasing the number of parameters and training iterations for a single network cannot significantly improve the precision, we quadrupled the parameter count of the baseline single network and correspondingly quadrupled the number of iterations. Second, to show that our method can improve the reconstruction accuracy while keeping the number of parameters fixed, we reduced the baseline model to one-fourth of its original size and then combined t with our method. Notably, both increasing and decreasing parameters involve adjusting the number of network layers and the number of neurons in each hidden layer.

Table 7 presents the performance metrics of different models. The term "Baseline" refers to reconstruction using the original single network model. "×4 iterations" indicates that the number of training iterations is quadrupled compared to the baseline, while "×4 layers" and "×4 hidden" respectively refer to the cases where the number of network layers and the number of neurons per layer in the baseline are each quadrupled, and the corresponding number of iterations are also quadrupled. It can be observed that simply increasing the number of parameters and training iterations for a single network yields only a modest accuracy improvement; however, the gains are minimal relative to the substantial computational and time costs incurred. This highlights that, with equivalent parameter counts, our proposed method is both effective and efficient for high-precision reconstruction. It is important to note that reducing the number of neurons in each layer of the baseline network

to $1/4$ of the original number and using four such networks for reconstruction, as shown in the row "$1/4$ hidden", leads to poorer reconstruction performance compared with the original "Baseline" row. This is because reducing the size of the hidden layer diminishes the modelling ability of the network output features. On the other hand, reducing the number of layers to $1/4$ of the original baseline network and then reconstructing with four such networks through our method, as shown in the row "$1/4$ layers", results in better performance than the original network, indirectly validating the effectiveness of our divide-and-conquer strategy.

4.7 Comparison with Other Representations

Recently, 3D Gauss Splatting (3DGS) [62] has become a highly promising representation for 3D scenes due to its excellent capability for color field expression. To further demonstrate the high-precision reconstruction capabilities of our method, we compare our approach with some state-of-the-art geometric surface reconstruction methods based on 3DGS, including Gaussian Opacity Fields (GOF) [63], PGSR [64], and 2D Gaussian Splatting (2DGS) [65]. Specifically, we perform comparative tests on the Sub-Campus dataset, using parameter settings from the original papers of these methods. Notably, these 3DGS-based methods still reconstruct the entire scene using a single model. Therefore, to ensure a fair comparison, we increase the training iterations for each model to 200K, allowing full convergence and optimal training.

Figure 15 presents the qualitative visualization comparison results. We can see that for large-scale scenes with sufficiently high complexity, these 3DGS-based methods still face limitations in expressive capability. For example, they

cannot accurately reconstruct the flat surfaces of buildings or the refined contours of windows. In contrast, our method achieves more accurate and detailed surface reconstruction. This further emphasizes the effectiveness effectiveness and high accuracy of our divide-and-conquer strategy.

5 CONCLUSION AND DISCUSSION

We introduced Neural SDF-Graph, a neural implicit surface reconstruction framework for scalable and high-quality reconstruction, editing, and more. Leveraging the locality of SDF, we represent individual components of a scene as graph nodes, where the SDF for each node is first reconstructed separately and then aligned via registration to derive a global SDF for the whole scene. This enables independent and flexible operations at different nodes, significantly enhancing the scalability and accuracy of SDF-based reconstruction.

Our approach can be improved from multiple aspects. First, our registration optimization is currently performed with an off-the-shelf solver. It is possible to design a specialized solver to boost its efficiency. Additionally, existing neural implicit reconstruction methods assume watertight surfaces, which may not be true for each node's shape. This can lead to irregular iso-surface shapes around the node boundary. Although our SDF blending can mitigate this issue and produce a smooth global SDF, it would be interesting to develop a reconstruction method for non-watertight surfaces to eliminate this problem. Finally, although we only apply our divide-and-conquer approach to the reconstruction of 3D scenes represented with SDFs in this paper, our idea is also applicable to other 3D representations. In the future, we plan to investigate its extension to explicit 3D representations such as 3DGS.

ACKNOWLEDGMENTS

REFERENCES

[1] Y. Furukawa and J. Ponce, "Accurate, dense, and robust multi-view stereopsis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 8, pp. 1362–1376, 2010. 1

[2] S. Seitz and C. Dyer, "Photorealistic scene reconstruction by voxel coloring," in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 1997, pp. 1067–1073. 1, 2

[3] E. Tola, C. Strecha, and P. Fua, "Efficient large-scale multi-view stereo for ultra high-resolution image sets," *Mach. Vision Appl.*, vol. 23, no. 5, p. 903–920, sep 2012. [Online]. Available: https://doi.org/10.1007/s00138-011-0346-8 1

[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020. 1, 6, 18

[5] L. Yariv, J. Gu, Y. Kasten, and Y. Lipman, "Volume rendering of neural implicit surfaces," in *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 1, 2, 4

[6] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," in *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, vol. 34, 2021, pp. 27 171–27 183. 1, 2, 4, 8

[7] Y. Wang, Q. Han, M. Habermann, K. Daniilidis, C. Theobalt, and L. Liu, "Neus2: Fast learning of neural implicit surfaces for multi-view reconstruction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. 1, 2, 4

[8] Z. Li, T. Müller, A. Evans, R. H. Taylor, M. Unberath, M.-Y. Liu, and C.-H. Lin, "Neuralangelo: High-fidelity neural surface reconstruction," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 4, 10, 18

[9] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1, 5, 8

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

[11] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *CVPR*, 2019. 1

[12] M. Tancik, V. Casser, X. Yan, S. Pradhan, B. Mildenhall, P. P. Srinivasan, J. T. Barron, and H. Kretzschmar, "Block-nerf: Scalable large scene neural view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 8248–8258. 1, 2

[13] H. Turki, D. Ramanan, and M. Satyanarayanan, "Mega-nerf: Scalable construction of large-scale nerfs for virtual fly-throughs," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 12 922–12 931. 1, 2

[14] A. Broadhurst, T. Drummond, and R. Cipolla, "A probabilistic framework for space carving," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 1, 2001, pp. 388–393 vol.1. 2

[15] J.-D. Boissonnat and B. Geiger, "Three-dimensional reconstruction of complex shapes based on the Delaunay triangulation," in *Biomedical Image Processing and Biomedical Visualization*, R. S. Acharya and D. B. Goldgof, Eds., vol. 1905, International Society for Optics and Photonics. SPIE, 1993, pp. 964 – 975. [Online]. Available: https://doi.org/10.1117/12.148710 2

[16] Y. Sahillioğlu and Y. Yemez, "Coarse-to-fine surface reconstruction from silhouettes and range data using mesh deformation," *Computer Vision and Image Understanding*, vol. 114, no. 3, pp. 334–348, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314209001969 2

[17] L. Yariv, Y. Kasten, D. Moran, M. Galun, M. Atzmon, B. Ronen, and Y. Lipman, "Multiview neural surface reconstruction by disentangling geometry and appearance," *Advances in Neural Information Processing Systems*, vol. 33, 2020. 2

[18] Z. Yu, S. Peng, M. Niemeyer, T. Sattler, and A. Geiger, "Monosdf: Exploring monocular geometric cues for neural implicit surface reconstruction," *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 4

[19] Q. Fu, Q. Xu, Y.-S. Ong, and W. Tao, "Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction," *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. 2, 4

[20] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: https://doi.org/10.1145/3528223.3530127 2

[21] R. Jensen, A. Dahl, G. Vogiatzis, E. Tola, and H. Aanæs, "Large scale multi-view stereopsis evaluation," in *2014 IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 406–413. 2, 18

[22] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski, "Building rome in a day," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 72–79. 2

[23] C. Früh and A. Zakhor, "An automated method for large-scale, ground-based city model acquisition," *International Journal of Computer Vision*, vol. 60, pp. 5–24, 10 2004. 2

[24] R. Raguram, C. Wu, J. M. Frahm, and S. Lazebnik, "Modeling and recognition of landmark image collections using iconic scene graphs," *International Journal of Computer Vision*, vol. 95, no. 3, pp. 213–239, 2011. 2

[25] M. Pollefeys, D. Nistér, J. M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrell,

C. Salmi, S. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles, "Detailed real-time urban 3d reconstruction from video," *Int. J. Comput. Vision*, vol. 78, no. 2–3, p. 143–167, jul 2008. [Online]. Available: https://doi.org/10.1007/s11263-007-0086-4 2

[26] J. L. Schönberger and J.-M. Frahm, "Structure-from-motion revisited," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4104–4113. 2

[27] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3d," vol. 25, no. 3, 2006. [Online]. Available: https://doi.org/10.1145/1141911.1141964 2

[28] S. Zhu, R. Zhang, L. Zhou, T. Shen, T. Fang, P. Tan, and L. Quan, "Very large-scale global sfm by distributed motion averaging," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4568–4577. 2

[29] A. Bergman, P. Kellnhofer, and G. Wetzstein, "Fast training of neural lumigraph representations using meta learning," in *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, Eds., vol. 34. Curran Associates, Inc., 2021, pp. 172–186. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2021/file/01931a6925d3de09e5f87419d9d55055-Paper.pdf 2

[30] M. Tancik, B. Mildenhall, T. Wang, D. Schmidt, P. P. Srinivasan, J. T. Barron, and R. Ng, "Learned initializations for optimizing coordinate-based neural representations," *CoRR*, vol. abs/2012.02189, 2020. [Online]. Available: https://arxiv.org/abs/2012.02189 2

[31] E. Ramon, G. Triginer, J. Escur, A. Pumarola, J. G. Giraldez, X. Giró-i-Nieto, and F. Moreno-Noguer, "H3d-net: Few-shot high-fidelity 3d head reconstruction," *CoRR*, vol. abs/2107.12512, 2021. [Online]. Available: https://arxiv.org/abs/2107.12512 2

[32] Q. Wang, Z. Wang, K. Genova, P. P. Srinivasan, H. Zhou, J. T. Barron, R. Martin-Brualla, N. Snavely, and T. A. Funkhouser, "Ibrnet: Learning multi-view image-based rendering," *CoRR*, vol. abs/2102.13090, 2021. [Online]. Available: https://arxiv.org/abs/2102.13090 2

[33] E. Dupont, H. Kim, S. M. A. Eslami, D. J. Rezende, and D. Rosenbaum, "From data to funca: Your data point is a function and you should treat it like one," *CoRR*, vol. abs/2201.12204, 2022. [Online]. Available: https://arxiv.org/abs/2201.12204 2

[34] X. Zhang, S. Bi, K. Sunkavalli, H. Su, and Z. Xu, "Nerfusion: Fusing radiance fields for large-scale scene reconstruction," *CVPR*, 2022. 2

[35] C. Sun, M. Sun, and H. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," *CoRR*, vol. abs/2111.11215, 2021. [Online]. Available: https://arxiv.org/abs/2111.11215 2

[36] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 5449–5459. 2

[37] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *CVPR*, 2022. 2

[38] L. Xu, Y. Xiangli, S. Peng, X. Pan, N. Zhao, C. Theobalt, B. Dai, and D. Lin, "Grid-guided neural radiance fields for large urban scenes," 2023. 2

[39] Z. Mi and D. Xu, "Switch-nerf: Learning scene decomposition with mixture of experts for large-scale neural radiance fields," in *International Conference on Learning Representations (ICLR)*, 2023. [Online]. Available: https://openreview.net/forum?id=PQ2zoIZqvm 2

[40] L. Goli, D. Rebain, S. Sabour, A. Garg, and A. Tagliasacchi, "nerf2nerf: Pairwise registration of neural radiance fields," 2022. 2

[41] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, "Barf: Bundle-adjusting neural radiance fields," 2021. 2

[42] Y. Chen, X. Chen, X. Wang, Q. Zhang, Y. Guo, Y. Shan, and F. Wang, "Local-to-global registration for bundle-adjusting neural radiance fields," 2023. 2

[43] H. Kim, M. Song, D. Lee, and P. Kim, "Visual-inertial odometry priors for bundle-adjusting neural radiance fields," in *2022 22nd International Conference on Control, Automation and Systems (ICCAS)*, 2022, pp. 1131–1136. 2

[44] C. Peat, O. Batchelor, R. Green, and J. Atlas, "Zero nerf: Registration with zero overlap," 2022. 2

[45] L. Yen-Chen, P. Florence, J. T. Barron, A. Rodriguez, P. Isola, and T.-Y. Lin, "iNeRF: Inverting neural radiance fields for pose estima-

tion," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021. 5

[46] Y. Lin, T. Müller, J. Tremblay, B. Wen, S. Tyree, A. Evans, P. A. Vela, and S. Birchfield, "Parallel inversion of neural radiance fields for robust pose estimation," 2023. 5

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017. 5

[48] S. F. Frisken and R. N. Perry, "Designing with distance fields," *ACM SIGGRAPH 2006 Courses*, pp. 60–66, 2006. 6

[49] Q. Wu, X. Liu, Y. Chen, K. Li, C. Zheng, J. Cai, and J. Zheng, "Object-compositional neural implicit surfaces," in *European Conference on Computer Vision*. Springer, 2022, pp. 197–213. 6

[50] Q. Wu, K. Wang, K. Li, J. Zheng, and J. Cai, "Objectsdf++: Improved object-compositional neural implicit surfaces," 2023. 6

[51] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," ser. SIGGRAPH '87. New York, NY, USA: Association for Computing Machinery, 1987, p. 163–169. [Online]. Available: https://doi.org/10.1145/37401.37422 6

[52] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia, "MeshLab: an Open-Source Mesh Processing Tool," in *Eurographics Italian Chapter Conference*, V. Scarano, R. D. Chiara, and U. Erra, Eds. The Eurographics Association, 2008. 6

[53] Y. Yao, Z. Luo, S. Li, J. Zhang, Y. Ren, L. Zhou, T. Fang, and L. Quan, "Blendedmvs: A large-scale dataset for generalized multi-view stereo networks," *CoRR*, vol. abs/1911.10127, 2019. [Online]. Available: http://arxiv.org/abs/1911.10127 6, 7

[54] M. Işık, M. Rünz, M. Georgopoulos, T. Khakhulin, J. Starck, L. Agapito, and M. Nießner, "Humanrf: High-fidelity neural radiance fields for humans in motion," *ACM Transactions on Graphics (TOG)*, vol. 42, no. 4, pp. 1–12, 2023. [Online]. Available: https://doi.org/10.1145/3592415 6, 8

[55] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *CoRR*, vol. abs/1612.00593, 2016. [Online]. Available: http://arxiv.org/abs/1612.00593 7

[56] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics*, vol. 36, no. 4, 2017. 7

[57] Z. Wang, E. P. Simoncelli, and A. C. Bovik, "Multiscale structural similarity for image quality assessment," in *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, vol. 2. IEEE, 2003, pp. 1398–1402. 8

[58] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595. 8

[59] Z. Yu, A. Chen, B. Antic, S. Peng, A. Bhattacharyya, M. Niemeyer, S. Tang, T. Sattler, and A. Geiger, "Sdfstudio: A unified framework for surface reconstruction," 2022. [Online]. Available: https://github.com/autonomousvision/sdfstudio 8

[60] L. Yariv, P. Hedman, C. Reiser, D. Verbin, P. P. Srinivasan, R. Szeliski, J. T. Barron, and B. Mildenhall, "Bakedsdf: Meshing neural sdfs for real-time view synthesis," *arXiv*, 2023. 10

[61] N. Ravi, J. Reizenstein, D. Novotny, T. Gordon, W.-Y. Lo, J. Johnson, and G. Gkioxari, "Accelerating 3d deep learning with pytorch3d," *arXiv:2007.08501*, 2020. 13

[62] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/ 14

[63] Z. Yu, T. Sattler, and A. Geiger, "Gaussian opacity fields: Efficient adaptive surface reconstruction in unbounded scenes," *ACM Transactions on Graphics*, 2024. 14

[64] D. Chen, H. Li, W. Ye, Y. Wang, W. Xie, S. Zhai, N. Wang, H. Liu, H. Bao, and G. Zhang, "Pgsr: Planar-based gaussian splatting for efficient and high-fidelity surface reconstruction," *arXiv preprint arXiv:2406.06521*, 2024. 14

[65] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, "2d gaussian splatting for geometrically accurate radiance fields," in *SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery, 2024. 14

[66] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-nerf 360: Unbounded anti-aliased neural radiance fields," *CVPR*, 2022. 18

[67] ——, "Zip-nerf: Anti-aliased grid-based neural radiance fields," *ICCV*, 2023. 18

Leyuan Yang is currently working as a master student in the School of Mathematical Science, University of Science and Technology of China. His research interests include computer vision, 3D vision and large-scale scene reconstruction.
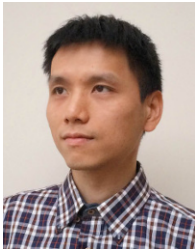
Bailin Deng is a Senior Lecturer in the School of Computer Science and Informatics at Cardiff University. He received the BEng degree in computer software (2005) and the MSc degree in computer science (2008) from Tsinghua University (China), and the PhD degree in technical mathematics (2011) from Vienna University of Technology (Austria). His research interests include geometry processing, numerical optimization, computational design, and digital fabrication. He is a member of the IEEE, and an associate editor of IEEE Computer Graphics and Applications.

Juyong Zhang is a professor in the School of Mathematical Sciences at University of Science and Technology of China. He received the BS degree from the University of Science and Technology of China in 2006, and the PhD degree from Nanyang Technological University, Singapore. His research interests include computer graphics, computer vision, and numerical optimization. He is an associate editor of IEEE Transactions on Multimedia.

APPENDIX A
EXPERIMENTAL CONFIGURATION AND DETAILS

We will provide details on the configurations of our experiments, including network structure, parameter settings.

A.1 SDF Method and Configuration

We first present the network structure and some basic parameter settings for the two selected SDF-based methods, NeuS and Bakedangelo. For additional details, please refer to the original project homepage and paper.

A.1.1 NeuS

Loss. NeuS utilizes only photometric loss [4], Eikonal loss [21] and mask loss. We set the loss weights as $\lambda_{color} = 1.0$, $\lambda_{eikonal} = 0.1$, and $\lambda_{mask} = 0.1$.

Network architecture. We retain the original network architecture of NeuS, consisting of two MLPs to encode SDF and color respectively. The signed distance function f is modeled by an MLP that consists of 8 hidden layers with hidden size of 256. The activation function for all hidden layers is Softplus with $\beta = 100$, and a skip connection is used to connect the input with the output of the fourth layer. The function c for color prediction is modeled by an MLP with 4 hidden layers with size of 256, which takes not only the spatial location p as inputs but also the view direction d, the normal vector of SDF $\nabla f(p)$, and a 256-dimensional feature vector from the SDF MLP. For additional information on the network structure and related details of NeuS, please consult the original paper.

Training details. The neural networks of NeuS are trained using the Adam optimizer. The learning rate is first linearly warmed up from 0 to 5×10^{-4} in the first 5K iterations, and then controlled by the cosine decay schedule to the minimum learning rate of 2.5×10^{-5}. We use 2048 rays in each iteration. The number of iterations is determined based on various experimental objectives and datasets, so we will elaborate on the specific experimental details below.

A.1.2 Bakedangelo

Loss. In addition to the fundamental photometric loss, Eikonal loss and mask loss, Bakedangelo incorporates curvature loss [8] and interlevel loss [66], [67]. We set the loss weights as $\lambda_{color} = 1.0$, $\lambda_{eikonal} = 0.01$, $\lambda_{mask} = 0.1$, $\lambda_{curvature} = 0.0005$, and $\lambda_{interlevel} = 1.0$.

Network architecture. We also retain the original network structure of Bakedangelo. The signed distance function f is modeled by an MLP that consists of only 1 hidden layers with a size of 256, and the function of color prediction c is modeled by an MLP with 4 hidden layers with a size of 256. Simultaneously, the model leverages a proposal MLP [66] for guiding the sampling process. Building upon Neuralangelo [8], the Bakedangelo model progressively activates the Hash encoding level.

Training details. The neural networks are trained using the Adam optimizer. The bias for initializing the SDF field is $bias = 0.1$. The learning rate for the SDF network is first linearly warmed up from 0 to 0.001 in the first 5K iterations,

and then controlled by the exponential decay schedule with milestones set at 600K and 800K iterations. The learning rate for the Proposal MLP is 0.01. The number of Hash encoding layers is $level\_init = 8$. In each iteration we use 1024 rays. We basically maintained the initial settings of this model in SDFStudio. Similarly, the number of iterations relies on various experimental goals and datasets, so a detailed explanation will be provided below. For other detailed parameter settings, please refer to the original project.

A.2 Implementation Details

In this part, we elaborate on the experimental details for our reconstruction.

Lego. We utilize the NeuS model for reconstruction. Considering the total number of rays is $100 \times 800 \times 800 = 6.4 \times 10^7$, the number of rays in each iteration is 2048, and we assume an average training time is 10 per ray, that is, $10 \approx 6.4 \times 10^7 \times \#iterations/2048$, so the total number of iterations is set to 310K.

Jade. We still use the NeuS model for reconstruction. We directly apply original experimental configuration of NeuS for the BlendedMVS datasets, that is, 512 rays per iteration, and the total number of iterations is 300K.

Human body. We employ the Bakedangelo model. The iteration count for the baseline is set to 3000K, while the iteration count for each node is set to 750K.

BlendedMVS. We employ the Bakedangelo model for reconstruction. The iteration count for the baseline is set to 400K, while the iteration count for each node is set to 100K.

Sub-Campus. We employ the Bakedangelo method for reconstruction in both the baseline (reconstructing as a whole) and each node. The iteration count for the baseline is set to 4000K, while the iteration count for each node is set to 1000K.

Campus. We still employ the Bakedangelo method, and the number of training iterations for each node is 1000K.

A.3 Surface Rendering Configuration

For the experiments in Section 3.5 to generate textures based on our method, we used the "SoftPhongShader" surface rendering model provided by PyTorch3D. The rasterization configuration parameters are: the distance used to expand the face bounding boxes for rasterization $blur\_radius = 0.0$, number of faces to keep track of per pixel $faces\_per\_pixel = 1$, size of bins to use for coarse-to-fine rasterization $bin\_size = 125$, whether to apply perspective correction when computing barycentric coordinates for pixels $perspective\_correct = True$, whether to correct a location outside the face to a position on the edge of the face $clip\_barycentric\_coords = False$, whether to only rasterize mesh faces which are visible to the camera $cull\_backfaces = False$. As for the lighting, we select the ambient light model "AmbientLights", and the parameter is $ambient\_color = (0.95, 0.95, 0.95)$. Additionally, for "SoftPhongShader", the parameter for controling the width of the sigmoid function used to calculate the 2D distance based probability is $blend\_params = BlendParams(sigma = 10^{-5})$.