Computational Visual Media
https://doi.org/10.1007/s41095-023-0340-x

Research Article

Multi-scale hash encoding based neural geometry representation

Zhi Deng[1], **Haoyao Xiao**[1], **Yining Lang**[2], **Hao Feng** [2], **and Juyong Zhang**[1] (✉)

Abstract Recently, neural implicit function-based representation has attracted more and more attention, and has been widely used to represent surfaces using differentiable neural networks. However, surface reconstruction from point clouds or multi-view images using existing neural geometry representations still suffer from slow computation and poor accuracy. To alleviate these issues, we propose a multi-scale hash encoding-based neural geometry representation which effectively and efficiently represents the surface as a signed distance field. Our novel neural network structure carefully combines low-frequency Fourier position encoding with multi-scale hash encoding. The initialization of the geometry network and geometry features of the rendering module are accordingly redesigned. Our experiments demonstrate that the proposed representation is at least 10 times faster for reconstructing point clouds with millions of points. It also significantly improves speed and accuracy of multi-view reconstruction. Our code and models are available at `https://github.com/Dengzhi-USTC/Neural-Geometry-Reconstruction`.

Keywords neural geometry representation; hash encoding; point cloud reconstruction; multi-view reconstruction

1 Introduction

3D shape is fundamental in many problems in computer graphics, computer vision, and robotics, as our physical world lies in 3D space. Unlike images, which are usually represented as a regular matrix in the digital world, 3D geometry employs various representations according to the application. Conventional representations such as polygon meshes, point cloud, and voxel grids can directly model 3D objects, but require excessive storage to represent geometry at high precision. Parametric geometry representations describe 3D objects via a series of basis functions, but are limited by the expressive ability of a low-dimensional parameter space. Recently, MLP-based neural implicit representations have been demonstrated to be effective and compact. A coordinate-based MLP models 3D space as a continuous implicit function by mapping a given point to its corresponding scalar attribute, such as occupancy and (un)-signed distance. The geometric surface can be extracted from a specified level-set using the *marching cubes* method [3].

In this paper, we particularly consider inferring the signed distance field from an unorganized input point cloud, or calibrated multi-view images. We aim to learn a coordinate-based implicit function $\Phi(\theta, \boldsymbol{x})$ with learnable parameters $\theta \in \mathbb{R}^d$, which satisfies the eikonal equation:

$$\|\nabla_{\boldsymbol{x}}\Phi\| \equiv 1, \text{ such that } \{G_k(\Phi(\theta, \boldsymbol{x}_i)), \boldsymbol{x}_i \in \partial\Omega\}_{i,k} \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^3$ is a 3D point, Ω is a well-behaved open set with boundary $\partial\Omega$, and $G_k(\cdot)$ is a non-linear constraint for geometry representation Φ. MLP-based approaches [4, 5] first introduced the eikonal equation to constrain the neural implicit function to be a signed distance field given point cloud or multi-view image input. However, as discussed in Refs. [6, 7], simple coordinate-based MLPs with ReLU activation have limited representation ability due to the *spectral bias* of neural networks. In addition, this geometry representation typically results in slow

1 School of Mathematical Sciences, University of Science and Technology of China, Hefei 230026, China. E-mail: Z. Deng, zhideng@mail.ustc.edu.cn; H. Xiao, xhy1999512@mail.ustc.edu.cn; J. Zhang, juyong@ustc.edu.cn (✉).

2 Alibaba Artificial Intelligence Governance Laboratory, Alibaba Group, Hangzhou 310017, China. E-mail: Y. Lang, louis.lyn@alibaba-inc.com; H. Feng, yuanning.fh@alibaba-inc.com.

Manuscript received: 2023-01-02; accepted: 2023-02-28

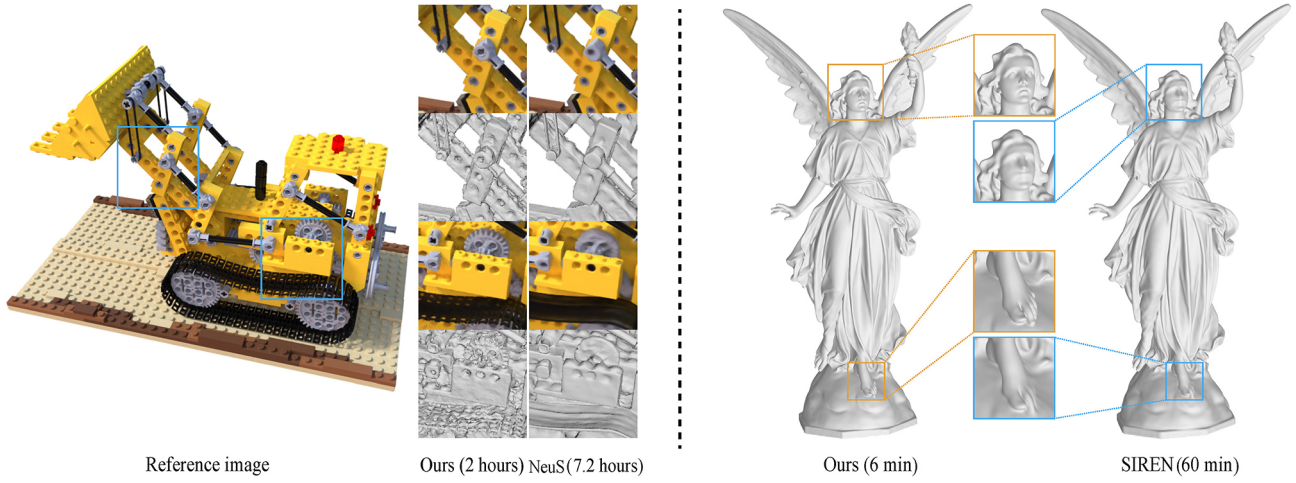


Fig. 1 Left, right: comparison of multi-view and point cloud reconstruction results using our method to the state-of-the art. It is 3.6 times faster than NeuS [1], and provides significantly improved reconstruction accuracy. Compared to SIREN [2], our method is considerably faster with improved point cloud reconstruction accuracy.

geometric reconstruction.

Many works have proposed how to improve neural geometry representation capability. Some, like SIREN [2], try to design more powerful activation functions. Others focus on positional encoding: encoding spatial location into a high-dimensional space using a given set of sinusoidal or spline functions [8–10]. These methods consider how to represent the high-frequency details of a given surface shape, but still fail to reconstruct geometric details accurately and efficiently. To tackle this problem, learnable positional encoding was introduced to further encode local geometric details around each given point using a predefined voxel grid. In particular, *learnable multi-scale hash encoding* [11] can efficiently obtain multi-scale geometric information. However, reconstruction artifacts may appear when directly applying this encoding strategy to neural geometry reconstruction tasks, for two reasons. Firstly, under weak supervision, it does not satisfy eikonal constraints well; explicit discrete grid-based neural geometry representations have poor gradient continuity, leading to a poor approximation of the signed distance field. Secondly, it requires careful initialization to help network optimization.

We propose a novel geometry representation based on multi-scale hash encoding to address these issues. Specifically, we move the hash encoding to the hidden layer as part of the input of the connected layer, and introduce Fourier position encoding as the input of the first layer to encode consecutive spatial locations, enhancing the gradient continuity of the geometric representation. We also initialize the geometry network using a modified version of SAL [12], in which optimization of the geometry starts from an approximate sphere.

To verify the effectiveness of our geometry representation in multi-view reconstruction, we use NeuS [1] as a baseline framework for comparisons of reconstruction accuracy and efficiency. In previous volume rendering frameworks, geometric features of points in 3D are extracted from the last layer of the geometry network. Thus, hash encoding of geometry representation encodes geometry features. This causes the reconstructed geometry to be inconsistent with the multi-view image rendered by the light field, due to the fast learning ability of the hash encoding operator. We thus move feature extraction to the connection layer of the geometry network, so that the multi-scale hash encoding only represents the geometry.

Extensive experiments demonstrate that our neural geometry representation outperforms the state-of-the-art neural geometry representations in terms of both speed and accuracy of reconstruction from point clouds and multi-view images. Compared to existing neural implicit function-based point cloud reconstruction methods, our method is at least 10 times faster and significantly more accurate. In multi-view image reconstruction, benefiting from the modified rendering framework, our approach can recover fine geometric detail at a

reconstruction speed at least 3.6 times faster than the state-of-the-art.

2 Related work

2.1 Neural geometry representation

Recently, coordinate-based neural networks ($\Phi(\theta, \boldsymbol{x})$, $\boldsymbol{x} \in \mathbb{R}^3$), which represent a 3D object as a continuous geometric shape, have attracted much attention. Refs. [13–15] utilize a neural network to represent the 3D shape as a (signed) distance field (SDF), while Refs. [16, 17] represent it as an occupancy field. Our work is closer to IGR [4], which uses simple coordinate-based MLPs to recover the SDF of a 3D shape. However, an implicit representation using coordinate-based MLPs cannot represent high-frequency details well, due to its limited representation ability [8]. Many methods have been proposed to address this issue. Xiao et al. [18] give a more detailed consideration of representation.

SIREN [2] uses sin as the activation function and suggests a suitable initialization method for optimization. Some works [19–23] divide complex shapes or large-scale scenes into regular subregions and replace the global MLP with local MLPs, so as to improve the geometric representation. It has been shown that using sinusoidal positional encoding can improve the performance of MLPs with ReLU in radiance field fitting [7]. In following work, Hertz et al. [9] propose a spatially adaptive progressive encoding (SAPE) scheme based on sinusoidal positional encoding, allowing an MLP-based representation to better fit the target signals with complex frequencies. As an alternative to sinusoidal functions, uniform parametric spline basis functions have also been utilized for position encoding with the aim of improving local and high-frequency geometric information [10].

Another strategy is to decompose the learnable feature or domain based on an explicit 3D data structure. To accelerate training, EG3D [24] encodes the 3D position of geometric rendering features by projecting it into a tri-plane with learnable features. ACORN [25] applies tree subdivision to the domain, with a large learnable auxiliary coordinate encoder neural network trained to output dense feature grids. Features in these dense grids are used to represent the positional encoding of any point in space. NGLOD [26] represents neural implicit functions using an octree-based position encoding, which adaptively fits shapes with multiple discrete levels of detail (LOD).

2.2 Hash encoding-based methods

As an efficient encoding tool, hash encoding is also widely used in geometry reconstruction. Voxel hashing is utilized in Ref. [27] in an online system for large, fine-scale volumetric reconstruction. A dynamic spatially-hashed truncated signed distance field is applied in Ref. [28] to contribute to a real-time house-scale dense 3D reconstruction system. Recently, a learnable multi-resolution hash encoding framework [11] has been proposed which encodes 3D position; it has been successfully applied to tasks for fast training of neural radiation fields and SDF fitting. In Ref. [29], a series of neural radiance fields is learnt as a facial expression basis by hash encoding, to enable semantic control over personalized semantic NeRF.

2.3 Point cloud reconstruction

Given a point cloud (possibly with normals), reconstructing the corresponding 3D shape is a classical problem in digital geometry processing. A parametric RBF representation may be utilized to reconstruct the surface by point cloud fitting [30, 31]. A further widely used approach is Poisson surface reconstruction [32], which solves a Poisson equation on a discrete volume, based on the given points and normals. Further related works can be found in the survey in Ref. [33].

Recently, surface reconstruction based on coordinate-based neural implicit representations has achieved great progress. DeepSDF [13] utilizes a neural implicit function to decode the SDF of 3D position in a bounding volume. Points2Surf [34] decomposes the neural geometry representation into a global sign function and local absolute distance function. Based on the eikonal equation, IGR [4] provides a new paradigm for computing high-fidelity implicit neural representations directly from raw 3D points.

SALD [35] advocates a novel sign agnostic regression loss, which incorporates both point-wise values and gradients of the unsigned distance function. Neural-Pull [36] uses the predicted signed distance and gradient at query locations to train a high-quality

neural geometry representation. To solve problems unrestricted by topology or type of input 3D signal, Chen et al. [37] propose a new data-driven approach for mesh reconstruction based on dual contouring.

2.4 Multi-view image reconstruction

Traditional MVS algorithms focus on neighbor view selection algorithms and photometric error measures. Robust neighbor view selection and visibility consistency algorithms are discussed in depth in Refs. [38] and [39], respectively. A currently popular MVS system, COLMAP [40], jointly estimates depth and surface normal, uses photometric and geometric priors for pixel-wise view selection, and uses geometric consistency for simultaneous refinement. We refer readers to Ref. [41] for a comprehensive overview of classical multi-view stereo reconstruction algorithms. Classical learning-based MVS methods attempt to replace certain components of the traditional MVS pipeline. Some works learn to match 2D features across views [42–44] or infer depth maps from multi-view images based on a data-driven framework [45–47]. Others [48–51] discuss in depth how to reduce memory needed by 3D convolution and how to increase inference speed of the model.

Recently, inverse rendering-based approaches have achieved great success in multi-view reconstruction. DVR [52] proposes a differentiable rendering approach to directly optimize the shape and texture of the input RGB images. IDR [5] utilizes the neural implicit function to simultaneously learn geometry and camera parameters, while neural rendering approximates the light reflected towards the camera. Wang et al. [53] use priors to extend IDR to 3D head reconstruction. NeRF [7] proposes a novel view synthesis framework which optimizes an underlying continuous volumetric scene function using multi-view images. VolSDF [54] attaches volume rendering techniques to IDR and eliminates the need for mask information. UNISURF [55] proposes a new multi-view framework in which implicit surface models and radiation fields can be formulated in a unified way, permitting surface and volume rendering from the same model. NerfingMVS [56] proposes a multi-view framework with learning-based priors to guide the NeRF optimization process. NeuralRecon [57] offers a neural network to directly reconstruct local surfaces represented as sparse TSDF volumes for each video fragment sequentially. MVSDF [58] jointly optimizes an SDF and a surface light field appearance model, directly supervised by geometry from stereo matching, and refined by multi-view feature consistency and fidelity of rendered images. DI-Fusion [59] proposes a local implicit function based framework for online 3D reconstruction with a commodity RGB-D camera. NeuS [1] proposes an unbiased density representation to recover high-quality surface shape with the help of differentiable volume rendering.

3 Method

3.1 Neural geometry representation

3.1.1 Background

The popular coordinate-based geometry representation uses the SDF to represent 3D geometric shape: $\Phi : \mathbb{R}^3 \to \mathbb{R}$, $\mathrm{SDF}(\boldsymbol{x}) = \Phi(\theta, \boldsymbol{x})$, where θ denotes learnable parameters. Using this continuously differentiable geometric representation, traditional geometric reconstruction can be directly performed using an end-to-end optimization framework, starting from, e.g., point clouds or multi-view images. The two most important aspects of a reconstruction method are its accuracy and speed. The recent NGP method [11] utilizes a learnable multi-scale hash encoding $\mathrm{enc}(\theta_\mathrm{h}, \boldsymbol{x})$ to encode 3D positions in space into the learnable features θ_h of a hash table. This improves the expressive ability of geometry representations and convergence speed of geometric fitting tasks such as SDF fitting and neural radiance field fitting. Thus, we introduce it for geometric reconstruction tasks, under weak supervision.

Our geometry representation utilizes multi-scale hash encoding $\mathrm{enc}(\theta_\mathrm{h}, \boldsymbol{x})$ derived from NGP. Specifically, we first construct hash tables arranged in L levels, with each level l containing up to T_l learnable features of dimension F. Each level independently and conceptually stores feature vectors at the vertices of a grid of a given resolution. As in NGP, the resolution at each level is progressively set to a value between the coarsest and finest resolutions $[N_\mathrm{min}, N_\mathrm{max}]$; N_max is a predefined target resolution:

$$N_l = \lfloor N_\mathrm{min} b^l \rfloor, \qquad b = 2^{\log_2(N_\mathrm{max}/N_\mathrm{min})/L}$$

where b is the scale for the level. We set N_max and N_min to be 2048 and 16 respectively in our experiments.

For any point $\boldsymbol{p} \in \mathbb{R}^3$ in space, we compute

$\mathrm{enc}(\theta_\mathrm{h}, \boldsymbol{p})$ by first finding the corresponding position on the grids at different levels through the predefined mapping, and then extract the learnable features from features stored in hash tables at different resolutions. Specifically, consider the hash table at a single level l. Point $\boldsymbol{p}$ is first scaled by the level's grid resolution and then rounded down and up $\lfloor \boldsymbol{p}_l \rfloor = \lfloor \boldsymbol{p} N_l \rfloor$, $\lceil \boldsymbol{p}_l \rceil = \lceil \boldsymbol{p} N_l \rceil$. $\lfloor \boldsymbol{p}_l \rfloor$ and $\lceil \boldsymbol{p}_l \rceil$ span a cell, whose 8 vertices are at integer coordinates in $\mathbb{Z}^d$. Then, the $\lfloor \boldsymbol{p}_l \rfloor$ and $\lceil \boldsymbol{p}_l \rceil$ are mapped into the hash-table using mapping function $h : \mathbb{Z}^3 \to \mathbb{Z}_{T_l}$. This maps each point into the feature grid via a one-to-one mapping when $N_l^3 \leqslant T_l$. At the fine level, due to storage limitations, we use a hash function to determine position in the feature hash table in a similar way to the strategy used in NGP.

Finally, the feature vectors of the point $\boldsymbol{p}$ at each level are obtained via tri-linear interpolation according to the relative position of $\boldsymbol{p}$ within its cell of the hash table. To improve continuity of high-level features, we use a second-order continuous interpolation weight function:

$$d(x) = 6x^5 - 15x^4 + 10x^3$$

3.1.2 Baseline geometry representation (NGP)

Figure 2 provides motivation for our geometry representation. Let us consider using the baseline network structure in NGP for point cloud reconstruction. This takes a hash encoding of a 3D point as input and outputs a 1-dimensional scalar value with several hidden layers. However, this leads to non-smooth reconstruction results; the gradient of the geometry representation has very poor continuity (also see Table 4 later). The reconstructed snowflake curve is wrong, resulting in a large difference between the predicted SDF and ground truth. The obvious cause is that here, supervisory information is much weaker than in NGP, which has ground truth supervision. This also greatly impacts multi-view image reconstruction (see Fig. 11 later). However, these observations have two deeper reasons. Firstly, only using the weak supervision of the multi-scale hash encoding neural network structure does not permit easy minimization of the eikonal constraint. Secondly, the neural geometry representation based on an explicit grid has gradient discontinuities, made worse by the high compression provided by hash map h. See Fig. 3: while points p and q are adjacent in space, due to the discontinuous nature of the features on the grid, they are far away in feature space. While it is straightforward to increase the number of sampling points when computing the eikonal equation constraint, this does not provide a good solution.

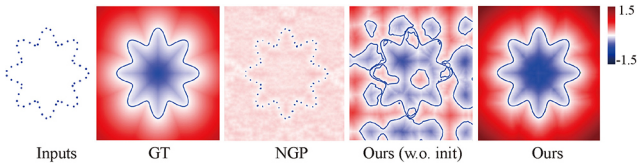


Fig. 2 Neural geometry representation ability, demonstrated by reconstructing a 2D point cloud sampled from a snowflake curve. Left to right: input (2D point cloud with normals), GT (ground truth), NGP (baseline hash encoding), our method without initialisation, our method. Blue curves: zero level set extracted by marching cubes. Color indicates the reconstructed implicit function value. In the last three columns, the eikonal constraints D_eik are 0.028, 0.0093, and 0.0041, with grad-error $D_\mathrm{grad\text{-}error}$ of 0.29, 0.09, and 0.04.

3.1.3 Our neural geometry representation

As Fig. 4(a) shows, based on the baseline network (NGP), we first introduce the connected layer, a hidden layer in the network used to connect features between different layers. Next, we move the hash-encoding $\mathrm{enc}(\theta_h, \boldsymbol{x})$ to the middle layer of the MLP as the input to the connected layer. Then the low-dimensional Fourier position encoding $\mathrm{enc}_\mathrm{ff}(\boldsymbol{x})$ is added as the input to the first layer, to reduce the discontinuity of learnable features caused by hash encoding. Finally, the modified initialization from SAL [12] is utilized to initialize our geometry network. As Fig. 5 shows, this initializes our network parameters to give an approximately spherical shape. As Fig. 2 and Table 4 show, the final reconstruction results are smoother, and the eikonal constraints are better satisfied. Our

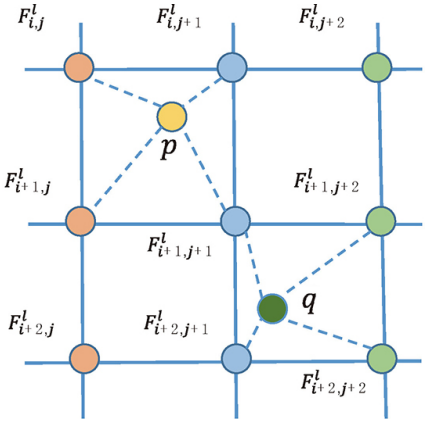


Fig. 3 Two spatially adjacent points p and q obtain interpolated features from the grid at level l. These features are different, due to the discontinuous nature of features on the discrete grids.

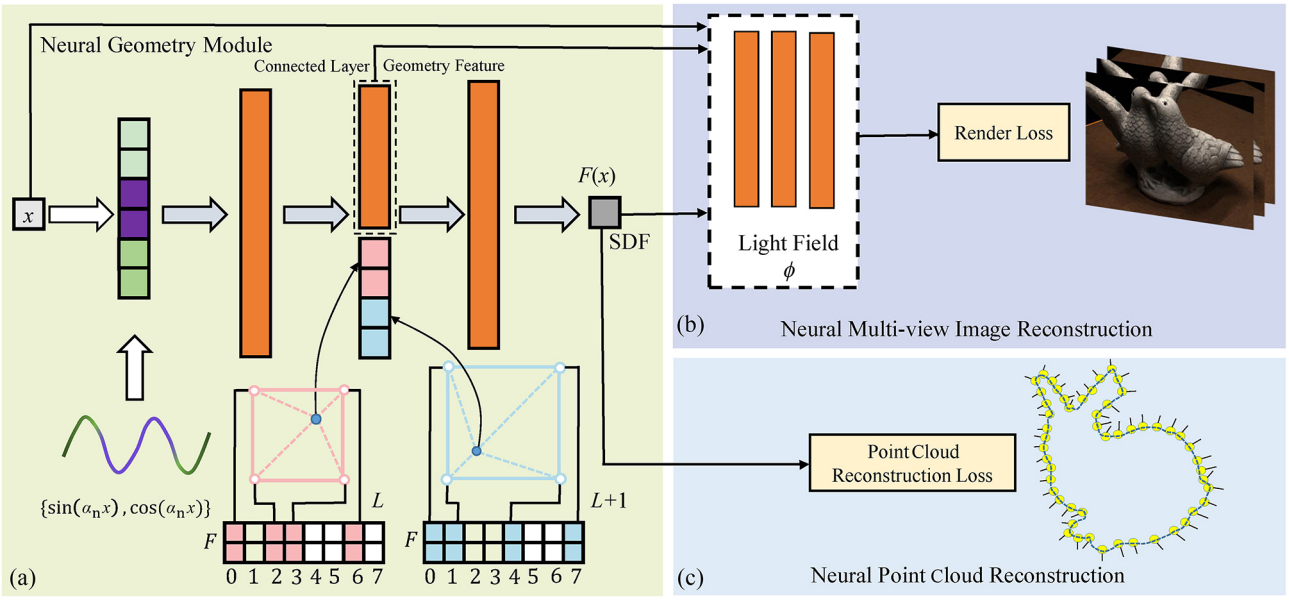


Fig. 4 Left: our geometry representation. Right: its application to geometric reconstruction. We combine Fourier position encoding and multi-scale hash encoding for neural geometry representation. We extract the geometry features of the multi-view rendering framework from the connected layer (inside the dashed box), so that hash encoding is avoided (for reasons of ambiguity) in geometry learning (the diagram for hash encoding follows Ref. [11]).

final neural geometry representation is expressed as $\Phi(\theta, \mathrm{enc}_{\mathrm{ff}}(\boldsymbol{x}), \mathrm{enc}(\theta_{\mathrm{h}}, \boldsymbol{x}))$.

3.1.4 Modification of geometry features

A neural rendering-based multi-view reconstruction framework usually comprises a geometry network and a color network (see Section 3.2.2). The color network predicts the RGB value of a given point with its geometric features, which are previously extracted from the last layer of the geometry network. In our geometry representation, due to the fast learning ability of hash encoding, the hash encoding of geometric features may encode rendering properties. As a result, the geometry module and the color module cannot be readily decoupled. Thus, the reconstructed geometry may be inconsistent with the rendered image, as we show in Section 5.3.2. To avoid this problem, we directly extract the geometry features from the connected layer in the geometry network, thus ensuring that hash encoding only encodes the geometry (SDF) of the object: see Fig. 4(a).

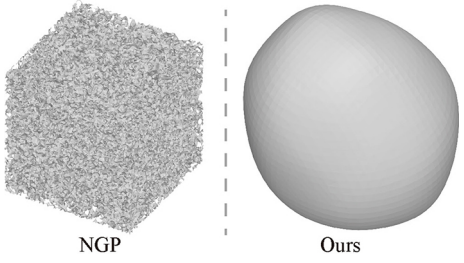


Fig. 5 Initial results for NGP and our geometry network with default initialization.

3.2 Geometry representation applications

3.2.1 Neural point cloud reconstruction

Like Refs. [2, 13], we use our neural geometry representation $SDF(\boldsymbol{x}) = \Phi(\theta, \mathrm{enc}_{\mathrm{ff}}(\boldsymbol{x}), \mathrm{enc}(\theta_h, \boldsymbol{x}))$ in the classical point cloud reconstruction task. Given an input point cloud $\{\boldsymbol{p}_i | \boldsymbol{p}_i \in \mathbb{R}^3\}$ with normals $\{\boldsymbol{n}_i | \boldsymbol{n}_i \in \mathbb{R}^3\}$ for an underlying surface S, the neural point cloud reconstruction task aims to infer a neural signed distance function of the surface S within a bounded volume Ω ($\{\boldsymbol{p}_i\} \subset \Omega$). As stated in Eq. (1), the constraints $\{G_k(\cdot)\}$ encourage $\{\boldsymbol{x}_i\}$ to be on the surface; the gradient of the implicit surface at $\{\boldsymbol{x}_i\}$ should be identical with the given normal $\{\boldsymbol{n}_i\}$. Specifically, we optimize our neural geometry representation using the loss terms in Eq. (2):

$$L = L_{\mathrm{data}} + \lambda_1 L_{\mathrm{eikonal}} + \lambda_2 L_{\mathrm{off}} \tag{2}$$

where

$$L_{\mathrm{data}} = \int_{\Omega_0} |\Phi(\boldsymbol{x})| + \lambda_3 (|1 - < \nabla_{\boldsymbol{x}} \Phi(\boldsymbol{x}), \boldsymbol{n}(\boldsymbol{x})) > |) \mathrm{d}\boldsymbol{x} \tag{3}$$

$$L_{\mathrm{eikonal}} = \int_{\Omega} (\|\nabla_{\boldsymbol{x}} \Phi(\boldsymbol{x})\| - 1)^2 \mathrm{d}\boldsymbol{x} \tag{4}$$

$$L_{\mathrm{off}} = \int_{\Omega \setminus \Omega_0} \psi(\Phi(\boldsymbol{x}), \beta_0) \mathrm{d}\boldsymbol{x} \tag{5}$$

Here, Ω_0 represents the input point set $\{\boldsymbol{p}_i\}$, and the function $\psi(x, \beta_0)$ can be formulated as $\exp(-\beta_0|x|)$, $\beta_0 \gg 1$, and $\{\lambda_i\}_{i=1}^4$ are weights. The data term L_{data} constrains the implicit function Φ by using oriented points sampled from the point cloud. The off-surface term L_{off} encourages points off the surface to have non-zero values.

3.2.2 Neural multi-view image reconstruction

Given calibrated multi-view images, neural multi-view reconstruction decouples geometry and appearance from them, representing geometry and appearance by implicit signed distance function (SDF) and a light field, respectively. In our work, the SDF is represented by our neural geometry representation $\text{SDF}(\boldsymbol{x}) = \Phi(\theta, \text{enc}_{\text{ff}}(\boldsymbol{x}), \text{enc}(\theta_{\text{h}}, \boldsymbol{x}))$. To optimize the parameters of our geometry representation, we utilize the state-of-the-art volume rendering framework NeuS [1] to render the 2D images based on the neural implicit SDF and light field, and then minimize the difference between the rendered images and the inputs. It should be noted that the volume rendering framework contains a color network $c : \mathbb{R}^3 \times \mathbb{S}^2 \to \mathbb{R}^3$, which encodes the color associated with geometric properties of a point $\boldsymbol{x} \in \mathbb{R}^3$ and view direction $\boldsymbol{v} \in \mathbb{S}^2$.

We render the proposed geometry representation with the corresponding light field to 2D images via a volume rendering framework, and then measure the difference between the rendered images and the input images for network supervision. Specifically, given a pixel from the input image I_t, we denote the ray from the center of the camera through this pixel as $\{\boldsymbol{r}(s) = \boldsymbol{o} + s\boldsymbol{v} | s \geqslant 0\}$, where $\boldsymbol{o}$ is the center of the camera and $\boldsymbol{v}$ is the unit direction vector of the ray. We integrate color along the ray using:

$$\hat{C}(\boldsymbol{r}) = \int_{s_{\text{n}}}^{s_{\text{f}}} T(t)\sigma(\boldsymbol{r}(t))c(\boldsymbol{r}(t), \boldsymbol{v})\text{d}t \tag{6}$$

$$\hat{M}(\boldsymbol{r}) = \int_{s_{\text{n}}}^{s_{\text{f}}} T(t)\sigma(\boldsymbol{r}(t))\text{d}t \tag{7}$$

where $\hat{C}(\boldsymbol{r})$ is the output color of this pixel, $\hat{M}(\boldsymbol{r})$ is the sum of the transmittance weights along the camera ray, s_{n} and s_{f} represent near and far bounds of the ray $\boldsymbol{r}$ respectively. $T(t) = \exp\left(-\int_{s_{\text{n}}}^{t} \sigma(\boldsymbol{r}(u))\text{d}u\right)$ denotes the accumulated transmittance along the ray, and $c(\boldsymbol{r}(t), \boldsymbol{v})$ is the color at the point $\boldsymbol{r}(t)$ along with the viewing direction $\boldsymbol{v}$. We formulate $\sigma(\boldsymbol{r}(t))$ to be an unbiased and occlusion-aware function, as in NeuS [1]. Finally, we use the loss functions in Eq. (8) to optimize the network parameters of the geometry module and the color module:

$$L = L_{\text{color}} + \alpha L_{\text{eikonal}} + \beta L_{\text{mask}} + \gamma L_{\text{off}} \tag{8}$$

where α, β, and γ are weights, and the color term L_{color} is defined as

$$L_{\text{color}} = \frac{1}{\#\mathcal{R}} \sum_{\boldsymbol{r}\in\mathcal{R}} \|M(\boldsymbol{r})(\hat{\boldsymbol{C}}(\boldsymbol{r}) - \boldsymbol{C}(\boldsymbol{r}))\|_1 \tag{9}$$

where

$$\mathcal{R} = \mathcal{R}(\{\mathcal{K}_i\}, \{\mathcal{T}_i\})$$
$$\#\mathcal{R} = \sum_{\boldsymbol{r}\in\mathcal{R}} M(\boldsymbol{r})$$

Here, $\mathcal{R}(\{\mathcal{K}_i\}, \{\mathcal{T}_i\})$ represents the ray set constructed based on the pixels from all images, and $\{\mathcal{K}_i\}, \{\mathcal{T}_i\}$ are the intrinsic and extrinsic parameters of the camera, respectively. $C(\boldsymbol{r}) \in \mathbb{R}^3$ and $M(\boldsymbol{r}) \in \{0, 1\}$ are ground truth color and object mask value for the ray $\boldsymbol{r}$, respectively.

The eikonal term, which regularizes the geometry representation $\Phi(\theta, \text{enc}_{\text{ff}}(\boldsymbol{x}), \text{enc}(\theta_{\text{h}}, \boldsymbol{x}))$ to be an SDF, is defined as

$$L_{\text{eikonal}} = \frac{1}{\#\mathcal{X}} \sum_{\boldsymbol{p}\in\mathcal{X}} (\|\nabla_{\boldsymbol{x}}\Phi(\boldsymbol{p})\|_2 - 1)^2 \tag{10}$$

where $\mathcal{X}$ is the sample point set on rays of set $\mathcal{R}(\{\mathcal{K}_i\}, \{\mathcal{T}_i\})$, and $\#\mathcal{X}$ is the number of points in $\mathcal{X}$.

The mask term L_{mask} is optional, and defined as

$$L_{\text{mask}} = \frac{1}{\#\mathcal{R}} \sum_{\boldsymbol{r}\in\mathcal{R}} \text{BCE}(\hat{M}(\boldsymbol{r}), M(\boldsymbol{r})) \tag{11}$$

where BCE is the binary cross entropy loss.

The off-surface loss L_{off} is defined as

$$L_{\text{off}} = \frac{1}{\sum_{\boldsymbol{x}\in\Omega} 1} \sum_{\boldsymbol{x}\in\Omega} \psi(\Phi(x), \beta_0) \tag{12}$$

where Ω is the bounding volume of the object. We uniformly sample 500 points per iteration in Ω, ψ is explained in Section 3.2.1, and β_0 is 100.

4 Datasets and implementation

4.1 Datasets

For point cloud reconstruction, we evaluate our approach and the baseline methods on the public *FAMOUS* dataset released by Points2Surf [34], with further cases from the Standard 3D Scanning Repository at `http://graphics.stanford.edu/data/3Dscanrep/` and the online 3D data library at `https://www.turbosquid.com/`, giving a total of 19 models. To show high-quality point cloud

reconstruction results, we preprocess the original meshes by subdividing each to give millions of points with normals. We these normalize these point clouds into $[-1,1]^3$.

For multi-view reconstruction, following Refs. [1, 5, 55], we evaluate our approach and baseline methods on 15 scenes from the DTU dataset [60]. Each scene is represented by 49 or 64 images from different perspectives with corresponding extrinsic parameters and a foreground mask provided by IDR [5]. The resolution of each image is 1200×1600, and the intrinsic camera parameters for each scene are given. This dataset is particularly challenging for reconstruction algorithms due to its diverse materials, appearances, geometry, non-Lambertian reflectance, and thin structures. We also conducted experiments and evaluations on some challenging scenes from the low-res set from the BlendedMVS dataset [61], a large dataset containing multi-view images with given camera extrinsic and intrinsic parameters. The selected cases have 31–143 images with a resolution of 768×576 and corresponding masks. Some scenes from a large multi-view face image dataset, FaceScape [62], were also evaluated. For each scene, we selected 32 images with 900×600 resolution; corresponding camera parameters, and a manually annotated rough foreground mask were also given.

4.2 Evaluation metrics

For point cloud reconstruction, to extract the fine geometry, we set the volume resolution to 2048^3 and used the marching cubes algorithm [3]. For each scene, we evaluated the quality of the 3D surface reconstruction result by calculating the chamfer-L_2 distance between the 10^7 uniformly sampled points on the reconstructed surface and the ground truth point cloud:

$$D_{\mathrm{scd}}(P,Q) = \frac{1}{\#P}\sum_{\boldsymbol{p}\in P}\min_{\boldsymbol{q}_t\in Q}\|\boldsymbol{p}-\boldsymbol{q}_t\|_2^2 \tag{13}$$

$$D_{\mathrm{cd}}(P,Q) = D_{\mathrm{scd}}(P,Q) + D_{\mathrm{scd}}(Q,P) \tag{14}$$

where P and Q are two point clouds, respectively.

The non-scale Laplace metric

$$D_{\mathrm{lap}}(\{V,E\}) = \frac{\sum_{\boldsymbol{v}\in V}\|\sum_{\boldsymbol{u}\in\mathcal{N}(\boldsymbol{v})}\omega_{\boldsymbol{v},\boldsymbol{u}}\boldsymbol{u}-\boldsymbol{v}\|_2^2}{\text{ave-edge}(\{V,E\})} \tag{15}$$

measures the non-scale smoothness of a triangle mesh $\{V,E\}$), with vertex and edge sets V and E respectively. Here, ave-edge$(\{V,E\})$ represents the average edge length of the triangle mesh, $\mathcal{N}(\boldsymbol{v})$ represents the set of neighbors of vertex $\boldsymbol{v}$, and $\omega_{\boldsymbol{v},\boldsymbol{u}}$ are area weights of the discrete Laplacian operator as defined in Ref. [63].

The further metrics

$$D_{\mathrm{eik}}(\Phi) = \frac{1}{\#\mathcal{X}}\sum_{\boldsymbol{p}\in\mathcal{X}}(\|\nabla_{\boldsymbol{x}}\Phi(\boldsymbol{p})\|_2-1)^2 \tag{16}$$

$$D_{\mathrm{grad}}(\Phi) = \frac{1}{\#\mathcal{X}}\sum_{\boldsymbol{p}\in\mathcal{X}}\|\nabla_{\boldsymbol{x}}\Phi(\boldsymbol{p})-\nabla_{\boldsymbol{x}}\Phi(\boldsymbol{p}+\delta)\|_2 \tag{17}$$

measure the degree of satisfaction of the eikonal constraint and the continuity of the gradient of the neural geometry representation Φ respectively, $\mathcal{X}$ is the uniformly sampled point set in the bounding volume Ω; it holds 20,000 points in our experiments. $\delta = \{10^{-3}, 10^{-3}, 10^{-3}\}$ represents a small displacement.

To assess multi-view image reconstruction results, we follow Refs. [1, 5, 55, 58] in choose 512^3 as the resolution used in the marching cubes algorithm to extract the final geometry. We use the formal *surface* evaluation script from the DTU dataset [60] to evaluate 3D surface reconstruction results. We further use a higher resolution 2048^3 to show details of reconstructed results, and we synthesize novel view images by performing volume or surface rendering of the reconstructed geometry using the given novel view parameters. For all methods, to measure the reconstruction quality of the light field, we report PSNR using pixels located within the predefined masks linking the rendered images and the reference images.

4.3 Implementations

4.3.1 Comparator methods

For the point cloud reconstruction task, we compare our approach to several state-of-the-art neural point cloud reconstruction methods, including IGR [8], SIREN [2], SplinePE [10], EG3D [24], and SAPE [9]. We conducted all experiments using Pytorch [64] on a GeForce RTX3090 GPU with 24 GB memory except for SplinePE [10], whose reference implementation requires a more powerful Tesla V100 GPU with 32 GB memory. For SAPE [9], we refer to the point cloud reconstruction framework used in IGR [4], and reproduce it for point cloud reconstruction. For EG3D [24], we reproduce the second-order gradient of the tri-plane position encoding operator using

Pytorch and embed it as a learnable position encoding in the geometry reconstruction framework. To obtain a fair comparison with IGR [4], we add a Fourier-position encoding [8] layer with encoding dimension 6 into their geometry network in the reference implementation. For simplicity, we denote these three reproduced methods EG3D*, SAPE*, and IGR(PE), respectively.

For the multi-view image reconstruction task, we compare our approach to IDR [5], NeuS [1], UNISURF [55], VolSDF [54], and NeRF [7]. Again, we conducted all experiments on a GeForce RTX 3090 GPU using their reference implementations.

4.3.2 Our approach

We use a hash encoding $\mathrm{enc}(\theta_\mathrm{h}, \boldsymbol{x})$ with 16 layers, and the dimension of each grid feature in each layer is 2. The frequency domain dimension of Fourier position encoding $\mathrm{enc}_\mathrm{ff}(\boldsymbol{x})$ is 6. In addition, we modify the network initialization strategy used in Ref. [12] and apply it to initialize our geometry network. Following NeuS [1], we use a hierarchical sampling strategy to sample points on rays in the multi-view image reconstruction task, and then use the mean of the SDF of sampled points as a threshold to eliminate various invalid sampled points on each ray.

5 Experiments

5.1 Neural point cloud reconstruction

5.1.1 Architecture

We use a 4-layer MLP with softplus activation functions to represent the geometry network in all neural point cloud reconstruction experiments. Each hidden layer contains 128 units, and the parameters of the softplus activation functions are set to $\beta = 100$. The input to the first layer is the Fourier-position encoding of spatial location $\mathrm{enc}_\mathrm{ff}(\boldsymbol{x})$, and the input to the third layer concatenates the hash encoding of spatial location $\mathrm{enc}(\theta_\mathrm{h}, \boldsymbol{x})$ and the output of the second hidden layer.

5.1.2 Hyperparameters

We trained our neural network for 1500 iterations with reconstruction loss Eq. (2). Following IGR [4], on each iteration, we sampled 65,536 points from the unorganized input 3D point cloud and 65,536 points from the bounding volume uniformly to optimize our network. In the objective loss functions in Eq. (2), we set β_0 and weights λ_1, λ_2, λ_3 to 100, 0.1, 0.05, and 1, respectively.

5.1.3 Point cloud reconstruction results

We assess reconstruction quality using the chamfer distance metric and record the required training time for each method. As Table 1 shows, in these selected challenging cases, our approach achieves significantly greater accuracy and a 10-time faster training speed. A detailed qualitative comparison using the *Thai Statue* is provided in Fig. 6. For IGR and SIREN, modifying the activation function and the spatial position encoding based on the ReLU-MLPs can improve the accuracy to a certain extent. But for IGR, the limited expressiveness of sinusoid functions results in a generally smooth reconstruction result with fewer details, and slow convergence. SAPE and SplinePE adopt a novel progressive learning strategy from low-frequency information to high-frequency information, which can reconstruct more details, but takes a long time to train. EG3D introduces great adaptability for high-frequency and low-frequency information in the reconstructed objects via the learnable position encoding: its reconstruction results have rich details. However, as highlighted in the blue rectangles in Fig. 6, EG3D is prone to noise in high-frequency details due to a lack of proper initialization and poor continuity of the explicit discrete representation. Like EG3D, our hash-encoded geometry representation employs a learnable positional encoding with multiple resolution layers, which is then compressed into a hash table with learnable features. Our geometry representation and initialization design give our approach better performance.

5.2 Neural multi-view image reconstruction

5.2.1 Architecture

In the multi-view image reconstruction task, our geometry network architecture is similar to that for point cloud reconstruction, except that 6 MLP layers are used. In addition, our light field c for color

Table 1 Reconstruction quality and computational cost using different methods for point cloud reconstruction. CD = chamfer-L_2 distance. T = time. I = number of iterations. M = memory usage

	IGR (PE)	SPAE*	SplinePE	SIREN	EG3D*	Ours
CD (10^{-6})	4.41	1.97	2.04	1.48	1.02	**0.59**
T (min)	18	50	1200	60	150	**6**
I (10^2)	500	500	200	400	500	**15**
M (GB)	6.6	11.31	6.8	13	6.2	**6.0**

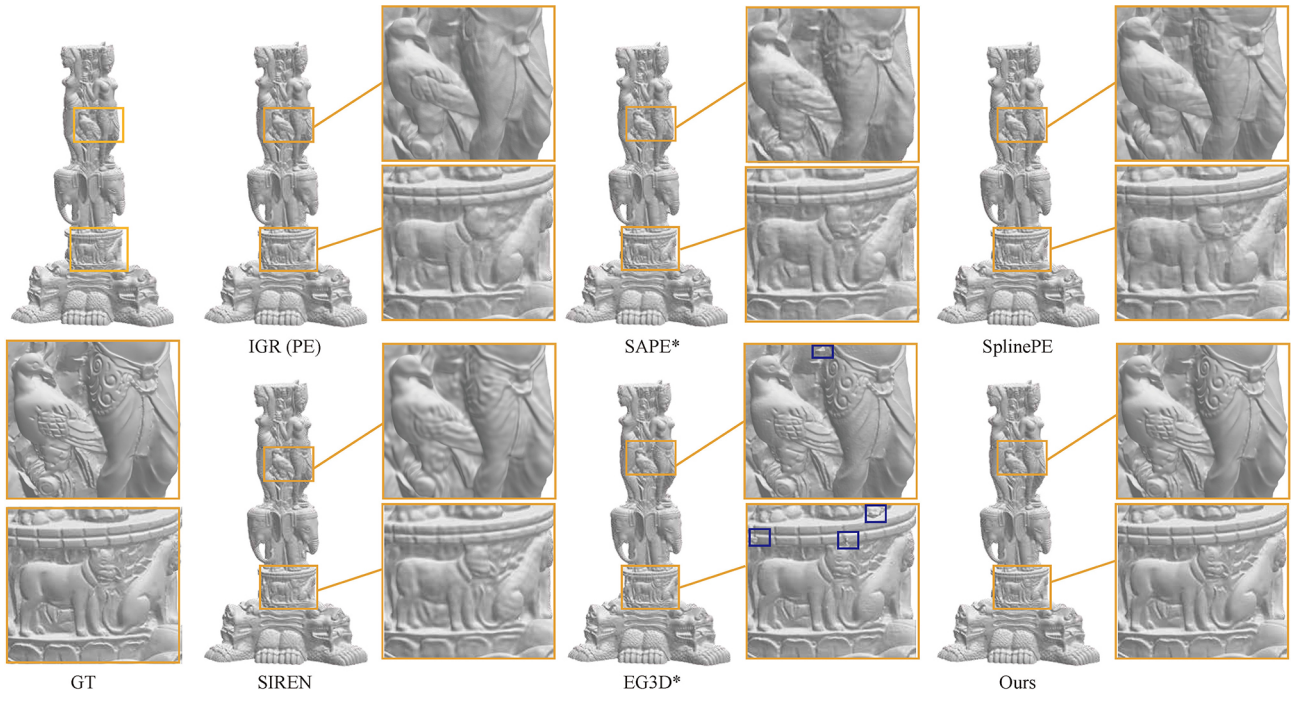


Fig. 6 Example results using different neural point cloud reconstruction approaches. GT = ground truth. Yellow boxes show close-ups to better appreciate detailed differences. Our method clearly provides superior reproduction accuracy. Blue boxes highlight various anomalous reconstruction results.

prediction is modeled using an MLP with 4 hidden layers, each containing 256 units. The inputs for the light field are the Fourier positional encoding of view direction $\boldsymbol{v}$, gradient $\boldsymbol{n}$, and the geometry feature vector output from the connected layer of the geometry network.

5.2.2 Hyperparameters

Here, we set the weights α, β, and γ in Eq. (8) to 0.1, 0.1, and 5×10^{-4}, respectively.

5.2.3 Multi-view image reconstruction results

In the multi-view image reconstruction task, for each scene, IDR and NeuS reconstruct the foreground object only, with a given mask, while NeRF, UNISURF, and VolSDF reconstruct the entire 3D scene. We evaluate reconstruction quality using the chamfer distance metric and the DTU dataset. We refer directly to existing results for IDR, NeuS, NeRF, UNISURF, and VolSDF, which were reported in the original papers [1, 54]. Scores are reported in Table 2, and show that our approach outperforms other baseline methods for these selected scenes. In addition, we compare the time and memory consumed in training by our approach and baseline methods in Table 3: it fairly and comprehensively shows that our approach requires less memory and training is faster.

We also conducted qualitative comparisons on the DTU and BlendedMVS datasets: see Fig. 7. NeuS, IDR, VolSDF, and UNISURF perform poorly in textureless areas of scene DTU40. Because of the lack of direct constraints on volume density, the geometry reconstructed by NeRF is relatively rough, with obvious noise. Compared to the other baselines, our approach has the ability to reconstruct more geometric detail, which is evident in the results for scenes DTU24, bmvs-clock and bmvs-stone.

We further compared our approach to NeuS [1] on scene DTU106. Note how the bird's feathers in Fig. 8 show better high-frequency detail consistent with the multi-view images when reconstructed by our method.

5.2.4 Novel view synthesis results

Novel view synthesis is a direct application of our neural multi-view image reconstruction framework: after using the existing neural volume rendering technique, we can synthesize a new image corresponding to the new view. PSNR values between reference images from the DTU dataset and synthesized images rendered from the reconstructed light field (for the foreground mask region only) are given in Table 2. They indicate that the quality of

Table 2 Multi-view reconstruction metrics for various methods, using from the DTU dataset

ScanID	IDR		NeuS		UNISURF		VolSDF		NeRF		Our method	
	CD ↓	PSNR ↑	CD ↓	PSNR ↑	CD ↓	PSNR ↑	CD ↓	PSNR ↑	CD ↓	PSNR ↑	CD ↓	PSNR ↑
scan24	1.63	23.29	0.83	26.73	1.32	25.51	1.14	24.16	1.90	26.02	**0.64**	**29.67**
scan37	1.87	21.36	0.98	23.42	1.36	23.26	1.26	21.29	1.60	**24.78**	**0.90**	24.23
scan40	0.63	24.39	0.56	26.32	1.72	25.79	0.81	24.93	1.85	27.83	**0.40**	**28.86**
scan55	0.48	22.96	0.37	24.92	0.44	25.53	0.49	22.78	0.58	26.36	**0.35**	**29.86**
scan63	**1.04**	23.22	1.13	30.49	1.35	28.12	1.25	28.99	2.28	**31.48**	**1.04**	30.97
scan65	0.79	23.94	**0.59**	32.55	0.79	30.38	0.70	28.68	1.27	31.92	0.72	**32.99**
scan69	0.77	20.34	**0.60**	29.03	0.80	28.78	0.72	27.67	1.47	**30.46**	0.71	28.53
scan83	1.33	21.87	1.45	**33.51**	1.49	30.78	**1.29**	31.50	1.67	33.31	1.39	33.45
scan97	1.16	22.95	0.95	**27.65**	1.37	25.93	1.18	22.57	2.05	26.43	**0.90**	27.49
scan105	0.76	22.71	0.78	31.20	0.89	30.83	**0.70**	30.56	1.07	31.07	0.76	**31.63**
scan106	0.67	22.81	0.52	32.13	0.59	30.68	0.66	29.50	0.88	32.26	**0.47**	**33.53**
scan110	**0.90**	21.26	1.43	28.85	1.47	29.03	1.08	27.11	2.53	28.19	1.01	**29.77**
scan114	0.42	25.35	0.36	28.42	0.46	28.06	0.42	26.60	1.06	29.08	**0.36**	**29.40**
scan118	0.51	23.54	**0.45**	34.97	0.59	32.31	0.61	28.60	1.15	34.86	0.49	**36.58**
scan122	0.53	27.98	**0.45**	34.81	0.62	33.03	0.55	31.60	0.96	32.95	0.57	**35.91**
Mean	0.90	23.20	0.77	29.66	1.02	28.53	0.86	27.11	1.49	29.80	**0.72**	**30.85**

Table 3 Computational costs of different methods of multi-view image reconstruction

	IDR	NeuS	UNISURF	VolSDF	NeRF	Ours
Time (h)	5.2	7.2	21	9	9.1	**1.8**
Rays	2048	512	1024	1024	1024	**512**
Iterations (10^3)	128	300	400	128	200	**120**
Memory (GB)	6.5	7	6	13.8	8	**5**

our reconstructed images are significantly better than those of other neural multi-view image reconstruction methods. Our results are also better than the results from the state-of-the-art novel view synthesis method, NeRF.

Focusing on novel view synthesis, we visually compare results of our approach to those of NeuS and NeRF for scene DTU55 *(rabbit)* in Fig. 9. We observe that our results are more realistic and detailed than those of NeuS, and even better than NeRF's, as shown by the fine-grained texture. In some places, such as for the yellow light in the background, NeRF's

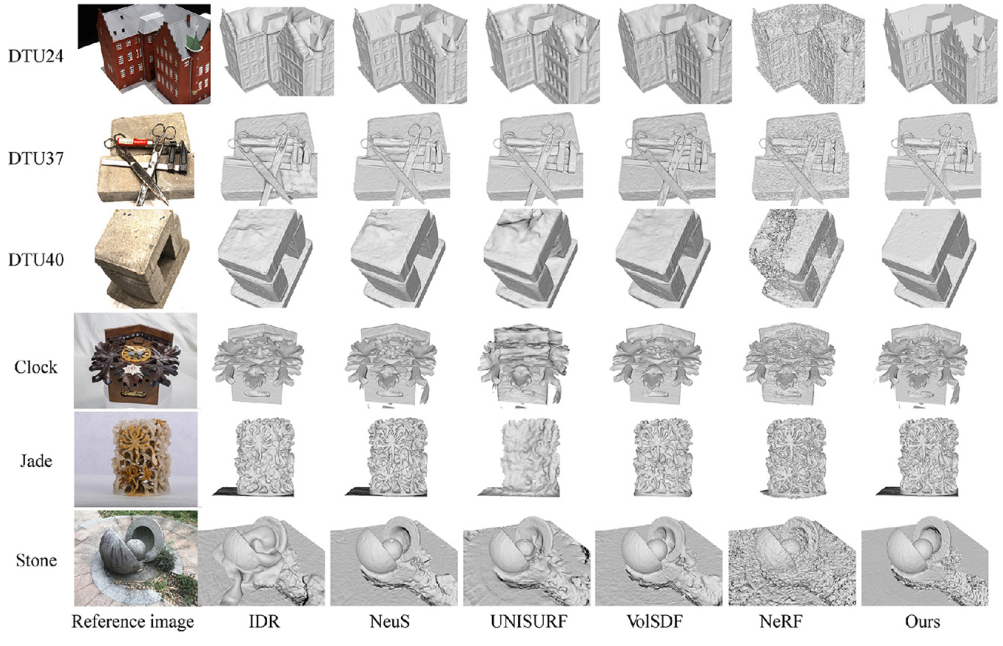


Fig. 7 Example reconstruction results provided by various methods, using multi-view images from the DTU and MVS-blender datasets.

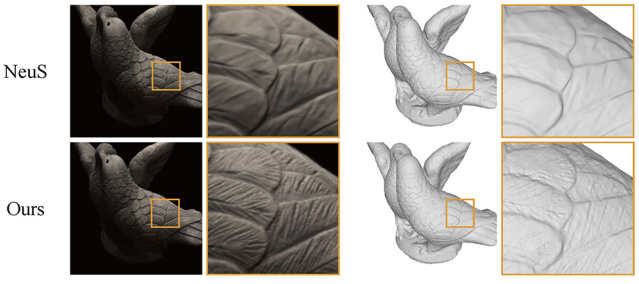


Fig. 8 Multi-view image reconstruction of DTU106 by NeuS and our method. Left, right: renderings of the reconstructed light field and geometry.

output displays evident artifacts due to incorrect learned geometry, while our approach still works well. In addition, our training time is 1/5 of that needed by NeRF.

5.3 Ablation study

5.3.1 Network structure for geometry representation

We next conduct a thorough ablation analysis for the effectiveness of our geometry representation. As explained in Section 3.1, the geometry representation, which only uses learnable hash features as a positional encoding (NGP), is unable to satisfy the constraints of the eikonal equation well under weak supervision, and it is prone to providing a non-smooth solution to the point cloud reconstruction and multi-view image reconstruction tasks. Table 4 gives comparative numerical results for network structure design; they indicate that NGP's fitting accuracy to the point cloud is very high, but the reconstructed surface is very rough. Specifically, the non-scale Laplace metric $D_{\text{lap}}(\{V_{\text{ori}}, E_{\text{ori}}\})$ of original mesh is 0.132, is the closest to the result reconstructed by our method. In particular, with regard to the continuous metric of the neural implicit representation in the whole space, the eikonal constraint D_{eik} and the spatial gradient continuity D_{grad} of NGP are both poor. The reason for the discontinuity of the spatial gradient of the geometry representation is that the features exist on discontinuous explicit grids, as shown in Fig. 3. To alleviate the above issue, we introduce Fourier position encoding [8] as input to the first layer of the geometry network to encode 3D position information in space, improving the continuity of geometry representation. We further use the network initialization from SAL [12] in our network to better satisfy eikonal constraints. Figures 10 and 11 show how using Fourier positional encoding and the learnable hash positional encoding together in our geometry representation results in smoother reconstruction results.

Table 4 Ablation of geometry representation network structure: processed point cloud datasets (contained 19 original models before upsampling). S: original point cloud. T: reconstructed mesh. $D_{\text{scd}}(\text{S},\text{T})$ and $D_{\text{scd}}(\text{T},\text{S})$ values are to be multiplied by 10^{-7} and 10^{-4}, respectively

	$D_{\text{scd}}(\text{S},\text{T})\downarrow$	$D_{\text{scd}}(\text{T},\text{S})\downarrow$	D_{lap}	$D_{\text{eik}}\downarrow$	$D_{\text{grad}}\downarrow$
NGP	4.060	1.640	0.166	0.023	0.27
Ours (no init)	3.962	1.630	0.145	0.009	0.11
Ours (full)	**3.242**	**0.758**	**0.129**	**0.002**	**0.06**

5.3.2 Framework for neural volume rendering

In most existing neural rendering frameworks, such as Refs. [1, 5, 7], estimating the light field of each input 3D point $\boldsymbol{p} \in \mathbb{R}^3$ requires the geometry feature and

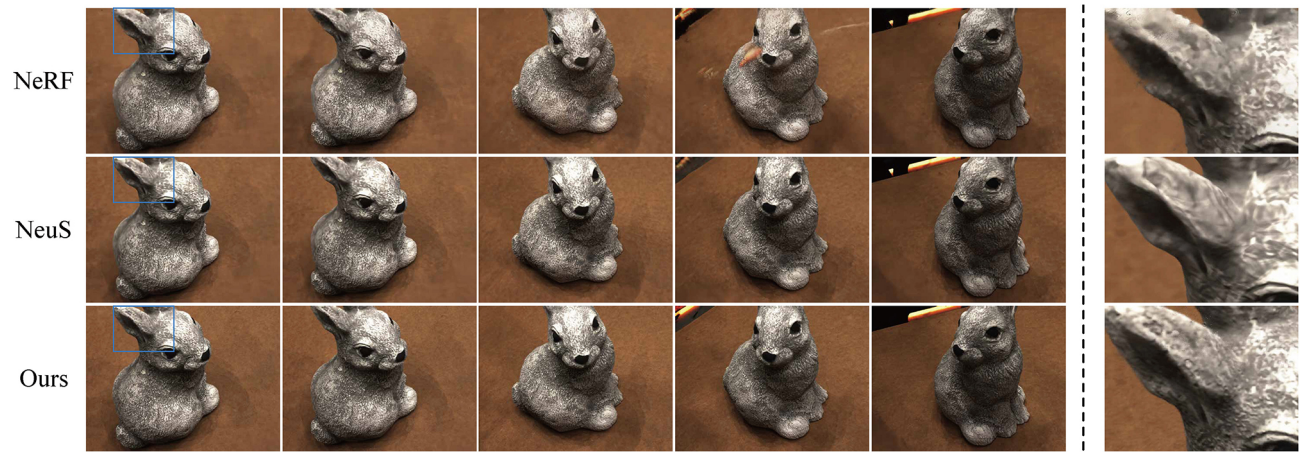


Fig. 9 Comparison of different methods on novel view synthesis. It shows the continuous interpolation of randomly selected two views to synthesize an unknown view image for DTU55. From the first column to the sixth column are the synthesis image of different novel views. The last column is a partial enlargement of the blue box in the first column of images.

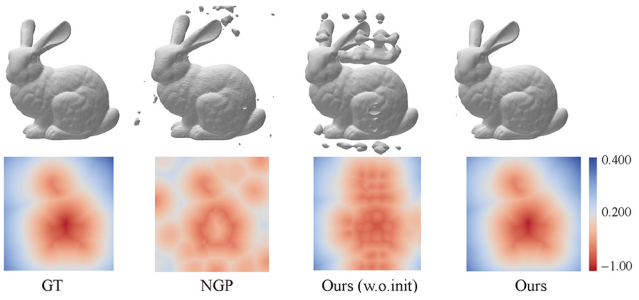


Fig. 10 Ablation study of geometry network designs for point cloud reconstruction. Left to right: ground truth, baseline with hash encoding (NGP), Fourier position encoding added, properly initialization also added (our full method). Above: zero level set extracted by marching cubes. Below: slice view of SDF, with eikonal constraints $D_{\text{eik}} = 0.012$, 0.006, and 0.00091 from columns 2 to 4, and values of $D_{\text{grad}} = 0.22$, 0.10, 0.005. Note that the NGP result has a double-layer structure.

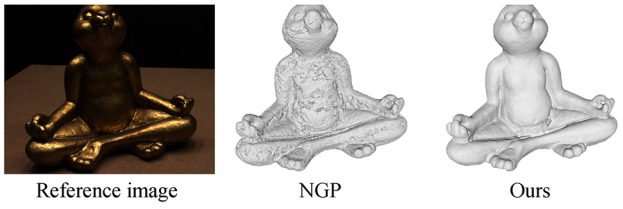


Fig. 11 Improvements in multi-view image reconstruction by using Fourier positional encoding and learnable hash positional encoding.

density, which are generated by the geometry network. Usually, the geometry feature is a part of the output of the last network layer. However, it is easy for the hash encoding to participate in learning color attributes in the multi-view geometry reconstruction task, which makes optimization of geometric density ambiguous. The main reasons are its strong expressiveness and speed of learning. As indicated in Table 5, encoding only geometric properties can improve geometry reconstruction and quality of rendering. Thus, in our geometry representation, we extract the geometry feature from the connected layer of our geometry network, where the hash encoding is exclusively used to represent the geometry (SDF) of objects. This avoids ambiguity and achieves more consistent geometric results with multi-view images, so our reconstructed geometry is in better agreement with the rendered image, as shown in Fig. 12.

Table 5 Assessment of whether hash encoding only encodes geometry (SDF), using multi-view reconstruction on the DTU dataset. Ours− and Ours extract the geometry feature without modification and modified as per our design, respectively

	NGP		Ours−		Ours	
	CD ↓	PSNR ↑	CD ↓	PSNR ↑	CD ↓	PSNR ↑
Mean	0.94	29.45	0.86	29.89	**0.72**	**30.85**

Fig. 12 Effect of location of geometric features in the rendering module of the neural multi-view framework. Left: reference image. Center: usual feature extraction location. Right: result when changing the extraction location to the connected layer.

5.3.3 Initialization of the geometry network

In neural geometry reconstruction, reasonable initialization plays a critical role in optimization of the network, as discussed by the authors of NeuS and IDR. Table 4 shows that initialization of the geometry representation is a crucial module. Figure 13 shows how using a geometry network without proper initialization leads to poor results in complex areas, e.g., with highlights or rapidly changing geometry.

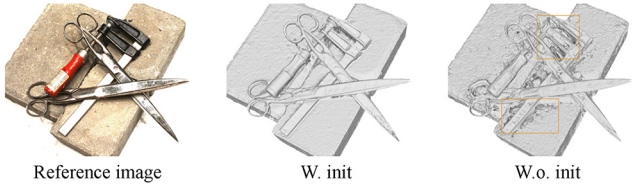


Fig. 13 Benefit of initialization strategy on multi-view reconstruction. Left: reference. Center: with. Right: without.

6 Discussion

Although our method moderately improves the reconstruction accuracy of textureless regions for the neural multi-view reconstruction tasks, it still needs further improvement to better cope with shadows and highlights. This is mainly due to the fact that the rendering representation cannot be perfectly decoupled from the geometry representation. Thus, it is crucial to design a more powerful rendering representation, and decoupling method to solve these problems; see Refs. [65, 66]. In addition, it is worth exploring how applying our ideas to dynamic geometry reconstruction and single-image based geometry reconstruction can improve speed and precision; see Refs. [67–69].

7 Conclusions

In this work, we have proposed a hash encoding-based neural geometry representation, and applied it

to recovery of the surface's signed distance function from an input point clouds or multi-view images. In our geometry network, we further combine our method with low-dimensional Fourier positional encoding and network initialization from SAL [12]. Meanwhile, for multi-view reconstruction, we redesign extraction of geometry features to avoid confusion between geometries and color values. Extensive experimental results have demonstrated that our method can achieve at least 10 times speedup for point cloud-based surface reconstruction, and significantly improve the accuracy and efficiency of multi-view reconstruction.

Acknowledgements

This research was partially supported by the National Natural Science Foundation of China (Nos. 62122071 and 62272433), the Fundamental Research Funds for the Central Universities (No. WK3470000021), and the Alibaba Innovation Research Program (AIR). The authors thank Peng Wang (the University of Hong Kong) for providing the script for evaluation of multi-view reconstruction, and Xueying Wang and Yuxin Yao (both from University of Science and Technology of China) for their help with paper writing.

Declaration of competing interest

The authors have no competing interests to declare that are relevant to the content of this article.

References

[1] Wang, P.; Liu, L.; Liu, Y.; Theobalt, C.; Komura, T.; Wang, W. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In: Proceedings of the 35th Conference on Neural Information Processing Systems, 27171–27183, 2021.

[2] Sitzmann, V.; Martel, J.; Bergman, A.; Lindell, D.; Wetzstein, G. Implicit neural representations with periodic activation functions. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, Article No. 626, 7462–7473, 2020.

[3] Lorensen, W. E.; Cline, H. E. Marching cubes: A high resolution 3D surface construction algorithm. *ACM SIGGRAPH Computer Graphics* Vol. 21, No. 4, 163–169, 1987.

[4] Gropp, A.; Yariv, L.; Haim, N.; Atzmon, M.; Lipman, Y. Implicit geometric regularization for learning shapes. In: Proceedings of the 37th International Conference on Machine Learning, 3789–3799, 2020.

[5] Yariv, L.; Kasten, Y.; Moran, D.; Galun, M.; Atzmon, M.; Basri, R.; Lipman, Y. Multiview neural surface reconstruction by disentangling geometry and appearance. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, Article No. 210, 2492–2502, 2020.

[6] Rahaman, N.; Baratin, A.; Arpit, D.; Draxler, F.; Lin, M.; Hamprecht, F. A.; Bengio, Y.; Courville, A. C. On the spectral bias of neural networks. In: Proceedings of the 36th International Conference on Machine Learning, 5301–5310, 2019.

[7] Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; Ng, R. NeRF: representing scenes as neural radiance fields for view synthesis. In: *Computer Vision – ECCV 2020. Lecture Notes in Computer Science, Vol. 12346*. Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J. M. Eds. Springer Cham, 405–421, 2020.

[8] Tancik, M.; Srinivasan, P. P.; Mildenhall, B.; Fridovich-Keil, S.; Raghavan, N.; Singhal, U.; Ramamoorthi, R.; Barron, J. T.; Ng, R. Fourier features let networks learn high frequency functions in low dimensional domains. In: Proceedings of the 34th International Conference on Neural Information Processing System, 7537–7547, 2020.

[9] Hertz, A.; Perel, O.; Giryes, R.; Sorkine-Hornung, O.; Cohen-Or, D. SAPE: Spatially-adaptive progressive encoding for neural optimization. In: Proceedings of the 35th Conference on Neural Information Processing Systems, 8820–8832, 2021.

[10] Wang, P. S.; Liu, Y.; Yang, Y. Q.; Tong, X. Spline positional encoding for learning 3D implicit signed distance fields. In: Proceedings of the 30th International Joint Conference on Artificial Intelligence, 1091–1097, 2021.

[11] Müller, T.; Evans, A.; Schied, C.; Keller, A. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics* Vol. 41, No. 4, Article No. 102, 2022.

[12] Atzmon, M.; Lipman, Y. SAL: Sign agnostic learning of shapes from raw data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2562–2571, 2020.

[13] Park, J. J.; Florence, P.; Straub, J.; Newcombe, R.; Lovegrove, S. DeepSDF: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 165–174, 2019.

[14] Liu, S. L.; Guo, H. X.; Pan, H.; Wang, P. S.; Tong, X.; Liu, Y. Deep implicit moving least-squares

functions for 3D reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1788–1797, 2021.

[15] Chibane, J.; Mir, A.; Pons-Moll, G. Neural unsigned distance fields for implicit function learning. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, Article No. 1816, 21638–21652, 2020.

[16] Mescheder, L.; Oechsle, M.; Niemeyer, M.; Nowozin, S.; Geiger, A. Occupancy networks: Learning 3D reconstruction in function space. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4455–4465, 2019.

[17] Chen, Z. Q.; Zhang, H. Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5932–5941, 2019.

[18] Xiao, Y. P.; Lai, Y. K.; Zhang, F. L.; Li, C. P.; Gao, L. A survey on deep geometry learning: From a representation perspective. *Computational Visual Media* Vol. 6, No. 2, 113–133, 2020.

[19] Peng, S. Y.; Niemeyer, M.; Mescheder, L.; Pollefeys, M.; Geiger, A. Convolutional occupancy networks. In: *Computer Vision – ECCV 2020. Lecture Notes in Computer Science, Vol. 12348*. Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J. M. Eds. Springer Cham, 523–540, 2020.

[20] Jiang, C. Y.; Sud, A.; Makadia, A.; Huang, J. W.; NieBner, M.; Funkhouser, T. Local implicit grid representations for 3D scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 6000–6009, 2020.

[21] Chabra, R.; Lenssen, J. E.; Ilg, E.; Schmidt, T.; Straub, J.; Lovegrove, S.; Newcombe, R. Deep local shapes: Learning local SDF priors for detailed 3D reconstruction. In: *Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science, Vol. 12374*. Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J. M. Eds. Springer Cham, 608–625, 2020.

[22] Genova, K.; Cole, F.; Sud, A.; Sarna, A.; Funkhouser, T. Local deep implicit functions for 3D shape. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 4856–4865, 2020.

[23] Liang, R.; Sun, H.; Vijaykumar, N. CoordX: Accelerating implicit neural representation with a split MLP architecture. *arXiv preprint* arXiv:2201.12425, 2022.

[24] Chan, E. R.; Lin, C. Z.; Chan, M. A.; Nagano, K.; Pan, B. X.; de Mello, S.; Gallo, O.; Guibas, L.; Tremblay, J.; Khamis, S.; et al. Efficient geometry-aware 3D generative adversarial networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 16102–16112, 2022.

[25] Martel, J. N. P.; Lindell, D. B.; Lin, C. Z.; Chan, E. R.; Monteiro, M.; Wetzstein, G. Acorn: Adaptive coordinate networks for neural scene representation. *ACM Transactions on Graphics* Vol. 40, No. 4, Article No. 58, 2021.

[26] Takikawa, T.; Litalien, J.; Yin, K. X.; Kreis, K.; Loop, C.; Nowrouzezahrai, D.; Jacobson, A.; McGuire, M.; Fidler, S. Neural geometric level of detail: Real-time rendering with implicit 3D shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 11353–11362, 2021.

[27] Nießner, M.; Zollhöfer, M.; Izadi, S.; Stamminger, M. Real-time 3D reconstruction at scale using voxel hashing. *ACM Transactions on Graphics* Vol. 32, No. 6, Article No. 169, 2013.

[28] Klingensmith, M.; Dryanovski, I.; Srinivasa, S.; Xiao, J. Z. Chisel: Real time large scale 3D reconstruction onboard a mobile device using spatially hashed signed distance fields. In: Proceedings of the Robotics: Science and Systems, 2015.

[29] Gao, X.; Zhong, C. L.; Xiang, J.; Hong, Y.; Guo, Y. D.; Zhang, J. Y. Reconstructing personalized semantic facial NeRF models from monocular video. *ACM Transactions on Graphics* Vol. 41, No. 6, Article No. 200, 2022.

[30] Carr, J. C.; Beatson, R. K.; Cherrie, J. B.; Mitchell, T. J.; Fright, W. R.; McCallum, B. C.; Evans, T. R. Reconstruction and representation of 3D objects with radial basis functions. In: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, 67–76, 2001.

[31] Walder, C.; Schölkopf, B.; Chapelle, O. Implicit surfaces with globally regularised and compactly supported basis functions. In: Proceedings of the 19th International Conference on Neural Information Processing System, 273–280, 2006.

[32] Kazhdan, M.; Bolitho, M.; Hoppe, H. Poisson surface reconstruction. In: Proceedings of the 4th Eurographics Symposium on Geometry Processing, 61–70, 2006.

[33] Berger, M.; Tagliasacchi, A.; Seversky, L. M.; Alliez, P.; Guennebaud, G.; Levine, J. A.; Sharf, A.; Silva, C. T. A survey of surface reconstruction from point clouds. *Computer Graphics Forum* Vol. 36, No. 1, 301–329, 2017.

[34] Erler, P.; Guerrero, P.; Ohrhallinger, S.; Mitra, N. J.; Wimmer, M. Points2Surf learning implicit surfaces from point clouds. In: *Computer Vision – ECCV 2020. Lecture Notes in Computer Science, Vol. 12350.*

Vedaldi, A.; Bischof, H.; Brox, T.; Frahm, J. M. Eds. Springer Cham, 108–124, 2020.

[35] Atzmon, M.; Lipman,Y. SALD: Sign agnostic learning with derivatives. In: Proceedings of the 9th International Conference on Learning Representations, 2021.

[36] Ma. B.; Han, Z.; Liu, Y. S.; Zwicker, M. Neural-pull: Learning signed distance functions from point clouds by learning to pull space onto surfaces. In: Proceedings of the 38th International Conference on Machine Learning, 7246–7257, 2021.

[37] Chen, Z. Q.; Tagliasacchi, A.; Funkhouser, T.; Zhang, H. Neural dual contouring. *ACM Transactions on Graphics* Vol. 41, No. 4, Article No. 104, 2022.

[38] Furukawa, Y.; Ponce, J. Accurate, dense, and robust multiview stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 32, No. 8, 1362–1376, 2010.

[39] Langguth, F.; Sunkavalli, K.; Hadap, S.; Goesele, M. Shading-aware multi-view stereo. In: *Computer Vision – ECCV 2016. Lecture Notes in Computer Science, Vol. 9907*. Leibe, B.; Matas, J.; Sebe, N.; Welling, M. Eds. Springer Cham, 469–485, 2016.

[40] Schönberger, J. L.; Zheng, E. L.; Frahm, J. M.; Pollefeys, M. Pixelwise view selection for unstructured multi-view stereo. In: *Computer Vision – ECCV 2016. Lecture Notes in Computer Science, Vol. 9907*. Leibe, B.; Matas, J.; Sebe, N.; Welling, M. Eds. Springer Cham, 501–518, 2016.

[41] Furukawa, Y.; Hernández, C. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision* Vol. 9, Nos. 1–2, 1–148, 2015.

[42] Kar, A.; Häne, C.; Malik, J. Learning a multi-view stereo machine. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, 364–375, 2017.

[43] Wang, F.; Galliani, S.; Vogel, C.; Speciale, P.; Pollefeys, M. PatchmatchNet: Learned multi-view patchmatch stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 14189–14198, 2021.

[44] Chen, R.; Han, S. F.; Xu, J.; Su, H. Point-based multi-view stereo network. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 1538–1547, 2019.

[45] Yang, J.; Mao, W.; Alvarez, J. M.; Liu, M. Cost volume pyramid based depth inference for multi-view stereo. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 44, No. 9, 4748–4760, 2022.

[46] Yao, Y.; Luo, Z. X.; Li, S. W.; Shen, T. W.; Fang, T.; Quan, L. Recurrent MVSNet for high-resolution multi-view stereo depth inference. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5520–5529, 2019.

[47] Peng, R.; Wang, R. J.; Wang, Z. Y.; Lai, Y. W.; Wang, R. G. Rethinking depth estimation for multi-view stereo: A unified representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8635–8644, 2022.

[48] Xu, H. F.; Zhang, J. Y. AANet: Adaptive aggregation network for efficient stereo matching. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1956–1965, 2020.

[49] Yang, Z. P.; Ren, Z. L.; Shan, Q.; Huang, Q. X. MVS2D: Efficient multiview stereo via attention-driven 2D convolutions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8564–8574, 2022.

[50] Cheng, X.; Zhong, Y.; Harandi, M.; Dai, Y.; Chang, X.; Li, H.; Drummond, T.; Ge, Z. Hierarchical neural architecture search for deep stereo matching. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, Article No. 1858, 22158–22169, 2020.

[51] Wang, F.; Galliani, S.; Vogel, C.; Speciale, P.; Pollefeys, M. PatchmatchNet: Learned multi-view patchmatch stereo. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 14189–14198, 2021.

[52] Niemeyer, M.; Mescheder, L.; Oechsle, M.; Geiger, A. Differentiable volumetric rendering: Learning implicit 3D representations without 3D supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 3501–3512, 2020.

[53] Wang, X. Y.; Guo, Y. D.; Yang, Z. Q.; Zhang, J. Y. Prior-guided multi-view 3D head reconstruction. *IEEE Transactions on Multimedia* Vol. 24, 4028–4040, 2022.

[54] Yariv, L.; Gu, J.; Kasten, Y.; Lipman, Y. Volume rendering of neural implicit surfaces. In: Proceedings of the 35th Conference on Neural Information Processing Systems, 4805–4815, 2021.

[55] Oechsle, M.; Peng, S. Y.; Geiger, A. UNISURF: Unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 5569–5579, 2021.

[56] Wei, Y.; Liu, S. H.; Rao, Y. M.; Zhao, W.; Lu, J. W.; Zhou, J. NerfingMVS: Guided optimization of neural radiance fields for indoor multi-view stereo. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 5590–5599, 2021.

[57] Sun, J. M.; Xie, Y. M.; Chen, L. H.; Zhou, X. W.; Bao, H. J. NeuralRecon: Real-time coherent 3D reconstruction from monocular video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 15593–15602, 2021.

[58] Zhang, J. Y.; Yao, Y.; Quan, L. Learning signed distance field for multi-view surface reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, 6505–6514, 2021.

[59] Huang, J. H.; Huang, S. S.; Song, H. X.; Hu, S. M. DI-fusion: Online implicit 3D reconstruction with deep priors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8928–8937, 2021.

[60] Jensen, R.; Dahl, A.; Vogiatzis, G.; Tola, E.; Aanæs, H. Large scale multi-view stereopsis evaluation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 406–413, 2014.

[61] Yao, Y.; Luo, Z. X.; Li, S. W.; Zhang, J. Y.; Ren, Y. F.; Zhou, L.; Fang, T.; Quan, L. BlendedMVS: A large-scale dataset for generalized multi-view stereo networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1787–1796, 2020.

[62] Yang, H. T.; Zhu, H.; Wang, Y. R.; Huang, M. K.; Shen, Q.; Yang, R. G.; Cao, X. FaceScape: A large-scale high quality 3D face dataset and detailed riggable 3D face prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 598–607, 2020.

[63] Wardetzky, M.; Mathur, S.; Kälberer, F.; Grinspun, E. Discrete Laplace operators: No free lunch. In: Proceedings of the 5th Eurographics Symposium on Geometry Processing, 33–37, 2007.

[64] Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An imperative style, high-performance deep learning library. In: Proceedings of the 33rd International Conference on Neural Information Processing Systems, Article No. 721, 8026–8037, 2019.

[65] Tiwary, K.; Klinghoffer, T.; Raskar, R. Towards learning neural representations from shadows. In: *Computer Vision – ECCV 2022. Lecture Notes in Computer Science, Vol. 13693*. Avidan, S.; Brostow, G.; Cissé, M.; Farinella, G. M.; Hassner, T. Eds. Springer Cham, 300–316, 2022.

[66] Suhail, M.; Esteves, C.; Sigal, L.; Makadia, A. Light field neural rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 8259–8269, 2022.

[67] Cai, H.; Feng, W.; Feng, X.; Wang, Y.; Zhang, J. Neural surface reconstruction of dynamic scenes with monocular RGB-D camera. In: Proceedings of the 36th Conference on Neural Information Processing Systems, 967–981, 2022.

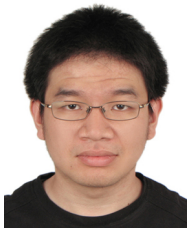
[68] Jiang, B. Y.; Hong, Y.; Bao, H. J.; Zhang, J. Y. SelfRecon: Self reconstruction your digital avatar from monocular video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 5595–5605, 2022.

[69] Deng, Z.; Liu, Y.; Pan, H.; Jabi, W.; Zhang, J. Y.; Deng, B. L. Sketch2PQ: Freeform planar quadrilateral mesh design via a single sketch. *IEEE Transactions on Visualization and Computer Graphics* Vol. 29, No. 9, 3826–3839, 2023.

Zhi Deng is a Ph.D. student in the School of Data Sciences, University of Science and Technology of China. His research interests include computer vision and computer graphics.

Haiyao Xiao is a postgraduate student in the School of Mathematical Sciences, University of Science and Technology of China, where he also obtained his bachelor degree in 2021. His research interests include computer vision and computer graphics.

Yining Lang received his bachelor degree in computing and economics from Beijing Institute of Technology in 2017, where he also received his master degree in computer science in 2020. He is currently an algorithm engineer in Alibaba Artificial Intelligence Governance Laboratory. His research interests include computer vision, computer graphics and virtual reality.



Hao Feng received his B.E. degree from the School of Computing, Beijing University of Technology, in 2007, and his Ph.D. degree in pattern recognition from the Image Processing Center, School of Astronautics, Beihang University, Beijing, China, in 2014. He is currently an algorithm engineer in the Intime Department, Alibaba Group, Beijing, China.

His research interests include computer vision, video understanding and their application to e-commerce and the retail industry.

Juyong Zhang is a professor in the School of Mathematical Sciences at the University of Science and Technology of China, where he received his B.S. degree in 2006. He has his Ph.D. 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*.