2025 International Conference on Intelligent Computing
July 26-29, Ningbo, China
https://www.ic-icc.cn/2025/index.php

NeuLF-Net: A Neural Latent Fusion Network for 3D Surface Reconstruction

Boren Li[1], Xuhua Shi[1,*], Haizhen Yu[1]

[1] Faculty of Electrical Engineering, Ningbo University, China
shixuhua@nbu.edu.cn

Abstract. Learning-based implicit neural networks have achieved inspirational performance on point cloud surface reconstruction. To reconstruct continuous surfaces from raw, discrete point clouds, existing methods typically project point clouds to grid latents or directly encode them as point latents. However, these methods rarely combine grid latents and point latents effectively and typically only perform simple topological transformations that ignore the spatial positional information of points, which seriously restricts the ability to capture fine details. In addition, traditional linear interpolation fails to sufficiently consider the global spatial information when inferring features of spatial points in sparse regions, resulting in a complete loss of expressiveness in some regions. In this paper, we propose a novel neural latent fusion network, named NeuLF-Net. The network serves as an end-to-end surface reconstruction framework, efficiently retaining the spatial encoding advantages of grid latents while capturing the fine-grained descriptive power of point latents. Specifically, we introduce a Neighbor Grid Enhancement Layer, which fully utilizes the neighbor information of the grid latents and point latents to enable enhancement of the two latents type. Furthermore, we design a novel adaptive interpolation strategy that exhibits better adaptability for point cloud spatial feature extraction. We extensively evaluate our pipeline with previous methods on three datasets including ShapeNet, Synthetic Rooms and ScanNet. Both quantitative and qualitative analyses demonstrate that NeuLF-Net substantially enhances the overall quality of point cloud reconstruction. From a visual perspective, the reconstruction results appear more realistic.

Keywords: NeuLF-Net, Implicit surface reconstruction, Point clouds, Adaptive interpolation strategy.

1 Introduction

Point clouds are one of the representations of 3D shapes, typically obtained by devices such as 3D scanners or LiDAR. They are characterized by sparsity, irregularity, and lack of topological structure [1]. Due to their discrete nature, raw point clouds data often difficult to meet practical requirements when directly applied to scientific and engineering applications. Therefore, point cloud surface reconstruction is pivotal in transforming scattered point clouds to continuous surfaces to fit various application scenarios.

The field of point cloud surface reconstruction has received extensive research. Traditional 3D point cloud reconstruction methods generally rely on geometric and optimization techniques [2]-[7], but these methods usually require prior knowledge of the point clouds and are limited in their ability to handle complex topological structures. Recently, deep learning-based point cloud surface reconstruction methods have demonstrated their advantages. These approaches learn complex surface patterns from point cloud data in an end-to-end manner, through deep neural networks, thereby generating accurate and detail-rich 3D surfaces. In particular, learning-based implicit representations have become a popular direction because they use implicit functions to represent continuous surfaces and can naturally handle arbitrary complex topological structures. This theory was first introduced in ONet [8]. In order to learn implicit function fields, Existing major methods typically project the input point clouds to a grid latents [23]-[32], or directly process as point latents [33]-[39], and finally generate continuous surfaces from the implicit fields through surface extraction algorithms.

However, projecting point clouds to grid latents often leads to shortcomings such as loss of accuracy, lack of flexibility, and loss of some details. In contrast, directly processing point latents retains more details but incurs high computational complexity and sensitive to sampling density, making it difficult to maintain topological integrity. Specifically, although [29] proposes an alternating topology approach, it remains fundamentally grid-based. The paper aims to solve the challenges. We propose a Neighbor Grid Enhancement Layer (NGEL), which enhances the feature representation of point latents by utilizing neighbor grid information and subsequently applies the enriched point latents to refine the grid latents. By embedding the NGEL module into the U-Net [40] architecture, we achieve multi-scale information fusion between points and grids, facilitating the effective processing of detail-rich point clouds data. The module is not simply about extracting neighbor points or girds for linear interpolation. It fuses point position encoding, point latents, and neighbor grid latents, calculating enhanced point latents through a learned approach. Then, in the inference phase, we present an adaptive interpolation strategy that adopts different strategies based on the spatial location of the query point, computing the corresponding high-dimensional features, and ultimately, the occupancy function value is derived. Finally, we construct a new network architecture called NeuLF-Net, which integrates the NGEL module and the adaptive interpolation strategy into the network to efficiently process point latents. The entire training process is end-to-end. Experimental results on the ShapeNet, Synthetic Rooms and ScanNet datasets demonstrate that NeuLF-Net outperforms other SOTA methods, achieving superior performance. The main contributions are summarized below:

- We proposed a Neighbor Grid Enhancement Layer, which integrates with a U-Net architecture to facilitate multi-scale feature enhancement.
- We designed an adaptive interpolation strategy that produces more accurate interpolated features for the query points.
- We integrated the NGEL and the adaptive interpolation strategy into the proposed NeuLF-Net, maintaining an end-to-end processing paradigm.

2 Related Works

2.1 Convolutions for Explicit Representations

The explicit representations of 3D shapes mainly include voxel, point clouds, and mesh representations, all of which are discretizations of the spatial domain. Voxel representations [9]-[11] as a 3D extension of pixel representations, are processed by extending 2D convolution to 3D convolution and were among the earliest representations applied in point clouds reconstruction. However, voxel representations are limited by resolution and memory constraints, leading to relatively coarse surface reconstructions. Point clouds representations [12],[13], characterized by their unstructured and unordered nature, lack topological information, which restricts their applicability in practical scenarios. Mesh representations [14]-[18], consisting of triangles connected by shared vertices or edges, provide strong generality. Nevertheless, directly generating triangular meshes from neural networks remains a significant challenge.

2.2 Convolutions for Implicit Representations

In contrast to explicit representation methods, implicit representation methods learn a continuous implicit function to predict the occupancy function value or signed distance function value for any arbitrary 3D coordinate in space. The surface is subsequently extracted using the classical Marching Cubes algorithm [19]. Mescheder et al. [8] introduced ONet, a method that implicitly represents three-dimensional surfaces through a deep network. This approach employs a neural network based on PointNet [36] to generate a set of feature vectors representing the entire shape and incorporates the Multiresolution IsoSurface Extraction (MISE) algorithm for efficient surface extraction. Park et al. [21] proposed DeepSDF, which utilizes a continuous signed distance function to compute the distance from any point in space to the surface, generating the final surface by calculating the zero-level set. Giebenhain et al. [22] presented AIR-Nets, which encode input point clouds into a set of local latent representations of 3D space. However, the representational capacity of vector-based encodings remains significantly constrained.

To address this issue, one approach projects point clouds into feature grids. Peng et al. introduced ConvONet [23], which combines a convolutional encoder with an implicit occupancy decoder, effectively aggregating both local and global feature information. This method maps input point clouds onto 2D feature planes or 3D feature volumes and uses bilinear or trilinear interpolation to retrieve features for arbitrary points, predicting their occupancy probability values. Tang et al. proposed SA-ConvONet [24], which leverages symbolic oblivious optimization for implicit fields in its input. S. Lionar et al. designed DP-ConvONet [25], introducing dynamic plane convolution operations into the model. Gropp et al. [26] proposed implicit geometric regularization techniques, significantly improving the smoothness and quality of the generated shapes. Jiang et al. [27] partitioned the scene into multiple regions and employed local SDF representations within each region, enabling efficient local reconstruction and global integration. Although there is extensive research on grid-based surface reconstruction methods, projecting point clouds to grid for processing leads to the loss of fine details, which is determined by the inherent characteristics of the grid structure.

Another approach is to directly handle point latents. Erler et al. [33] proposed the Points2Surf method, which extracts local geometric features from input point clouds through local sampling, effectively capturing surface details. Wang et al. [34] introduced RangeUDF, which employs point convolution techniques, enabling the model to utilize more raw surface points at the input stage, thereby improving reconstruction performance, especially in large-scale scenes. Boulch et al. [35] proposed POCO, which extracts local features for every point through point convolution operations and aggregates them by neighbor information. Additionally, the method introduces an attention-based weighting module to decode these features and predicts the occupancy function value of each point. Fan et al. [36] proposed Bifusion, which is based on point and voxel representations but is limited by the resolution of the voxels. Ummenhofer et al. [37] proposed AdaConv, which uses the point convolution features with the aid of oriented normals of the point clouds. However, point-based methods typically rely on neighbor features to infer global topology. When the point cloud is sparse, it leads to incomplete reconstructions, particularly in complex scenes. In order to solve these problems, our work is devoted to combining the advantages of two types of latents, to generate a more realistic and detailed surface.

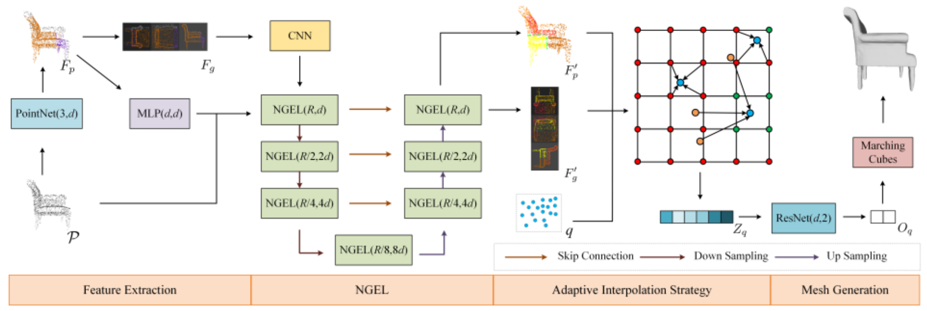

Fig. 1. The architecture of NeuLF-Net. There are four main stages in our architecture. We first extract point latents and grid latents from input point clouds, which are processed by the U-Net network composed with the NGEL to generate enhanced point and grid latents. Ultimately, the adaptive interpolation module valuates the occupancy function value of query points. R and d represent the grid resolution and the number of feature channels, respectively.

3 Methods

Goals. Given a set of three-dimensional discrete points $\mathcal{P} = \{p_i \in R^3\}_{i=1}^N$ as input, our goal is to construct a continuous implicit function $\mathcal{F}_\theta$: $R^3 \to [0,1]$, such that for any query point $q \in R^3$ in space, the value of the occupancy function $O_q = \mathcal{F}_\theta(q)$ can be calculated, where N represents the number of sampled points in the point cloud. By sampling points in 3D space and labeling them, we learn the implicit function $\mathcal{F}_\theta$ to make the reconstructed surface as close as possible to the ground truth surface.

3.1 NeuLF-Net Architecture

NeuLF-Net consists of four key parts: feature extraction, NGEL module, adaptive interpolation strategy and mesh generation. The entire network structure is illustrated in Fig. 1.

We first use PointNet [20] to extract the initial point latents $F_p = \{c_i \in R^d\}_{i=1}^N$, where d represents the dimension of the point latents. Then we normalize the point cloud and use orthogonal projection to map it to the grid space, initializing the grid latents F_g. In this case, we use a plane as grid latents, which results in $F_g \in R^{R\times R\times d}$, where R is the grid resolution. In this way, the initialization of the feature is completed. During the encoding phase, we apply the U-Net architecture to encode the two latents type. Each layer is an NGEL module, designed to leverage the neighborhood information of the grid latents to enhance the point latents representation ability, and reapply the point latents to the grid through spatial mapping, achieving interactive enhancement. When down-sampling is performed, the grid resolution is halved and the number of characteristic channels is doubled. The upsampling stage is the opposite. In the inference phase, we propose an adaptive interpolation strategy, which demonstrates greater robustness compared to traditional linear interpolation in the architecture. Finally, the mesh is generated using the Marching Cubes algorithm.

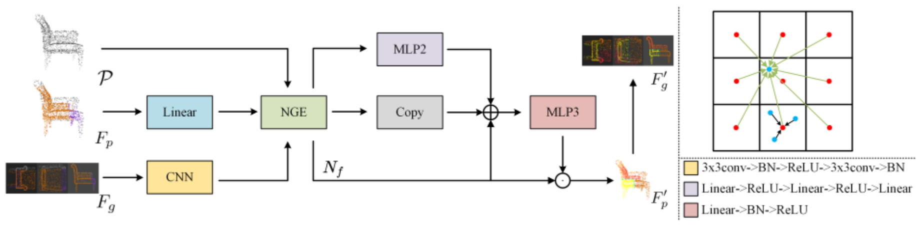

Fig. 2. Left: NGEL module. It extracts neighboring grid features, fuses positional encoding information, and employs an attention mechanism to compute enhanced point features that are subsequently fed back into the grid. **Top Right:** Neighbor Grid Extraction. Red dots represent grid features, and blue dots represent point features. The module extracts local grid and positional information to support the subsequent enhancement of point features. **Bottom right**: Legend indicating the specific details of the model architecture.

3.2 NGEL

We propose the Neighbor Grid Enhancement Layer (NGEL), as illustrated in Fig. 2, which combines the advantages of both latent representations. To illustrate this, we take the projection onto the xy-plane grid as an example, with a similar process applying to voxel grid.

Point Clouds Spatial Mapping. For any normalized coordinate point $p(x_0, y_0, z_0)$ in space, we calculate the normalized grid coordinates $p_{\text{nor}}(x, y)$, as well as the corresponding grid index $\text{index}(i, j)$, with the calculation formula as follows:

$$p_{\text{nor}}(x, y) = \left(\frac{x_0}{1+\varepsilon}, \frac{y_0}{1+\varepsilon}\right), \tag{1}$$

$$\text{index}(i,j) = (\lfloor x \times R \rfloor, \lfloor y \times R \rfloor), \tag{2}$$

where R is the grid resolution, and ε is a small constant. The center coordinates of the grid $c_{i,j}$ can be computed using the index information as follows:

$$c_{i,j} = \text{index}(i,j)/R. \tag{3}$$

Neighbor Grid Extraction(NGE). For the grid $\text{index}(i,j)$ of $p_{\text{nor}}(x,y)$, we extract the neighbor grid set $N_{i,j}$ where the point is located. The definition of $N_{i,j}$ is as follow:

$$N(i,j) = \left\{ \text{index}\left(i + \Delta_i, j + \Delta_j\right) \mid \Delta_i, \Delta_j \in \{-1,0,1\} \right\}. \tag{4}$$

Considering the boundary conditions, it is necessary to clip the neighbor grid information to obtain the actual neighbor. When $i + \Delta_i < 0$ or $i + \Delta_i \geq R$, the actual grid coordinate i_f to be taken is:

$$i_f = \min(\max(i + \Delta_i, 0), R - 1), \tag{5}$$

similarly, j_f is:

$$j_f = \min\left(\max\left(j + \Delta_j, 0\right), R - 1\right). \tag{6}$$

This guarantees that the neighbor grid remains within the valid range. The processed actual neighbor grid index set is defined as:

$$N_f(i,j) = \left\{ \text{index}\left(i_f, j_f\right) \mid \Delta_i, \Delta_j \in \{-1,0,1\} \right\}. \tag{7}$$

Thus, the corresponding neighbor grid latents set $\mathcal{F}_g$:

$$\mathcal{F}_g = \left\{ F_g(i,j) \mid (i,j) \in N_f(i,j) \right\}, \tag{8}$$

where $F_g(i,j)$ is the grid latents at the $\text{index}(i,j)$.

Position Encoding. To incorporate the positional information of the point cloud into the latent space, we calculate the relative position encoding of the neighbor grid centers with respect to the point cloud. We take the set of center coordinates of the neighbor grid for this point C:

$$C = \left\{ c_{i,j} \mid (i,j) \in N_f(i,j) \right\}. \tag{9}$$

Since the coordinates of the point are fixed, the formula for calculating the position encoding is as follows:

$$\mathcal{F}_{enc} = \left\{ MLP\left(c_{i,j} - p_{\text{nor}}(x,y)\right) \mid (i,j) \in N_f(i,j) \right\}, \tag{10}$$

where MLP denotes the Multi-Layer Perception.

Latent Update. To incorporate the grid latents into the point cloud, we consider a simple attention mechanism. we compute the importance of each neighbor grid relative to point p, and obtain the normalized weight $\widetilde{w}$, the calculation formula is as follows:

$$\widetilde{w} = \mathrm{softmax}\Big(\mathrm{MLP}\big(\mathcal{F}_{\mathrm{p}} - \mathcal{F}_g + \mathcal{F}_{enc}\big)\Big), \tag{11}$$

Finally, the point latents is updated, and the enhanced point latents F_p' is:

$$F_p'(i,j) = \sum_{(i,j)\in N_f(i,j)} \widetilde{w}(i,j) \cdot F_g(i,j). \tag{12}$$

To feedback the enhanced point latents into the grid latents and obtain the enhanced grid latents F_g', we rescan the grid space and apply average pooling to points in the same grid to update the grid latents:

$$F_g'(i,j) = \frac{1}{|I_{i,j}|} \sum_{p\in \mathrm{index}(i,j)} F_p', \tag{13}$$

where $I_{i,j}$ indicates the number of points in the index(i,j) grid. The NGEL module is seamlessly integrated with the U-Net module, leveraging skip connections to further enhance the feature extraction and expressive ability of the module. The encoder will eventually output the point latents and the grid latents.

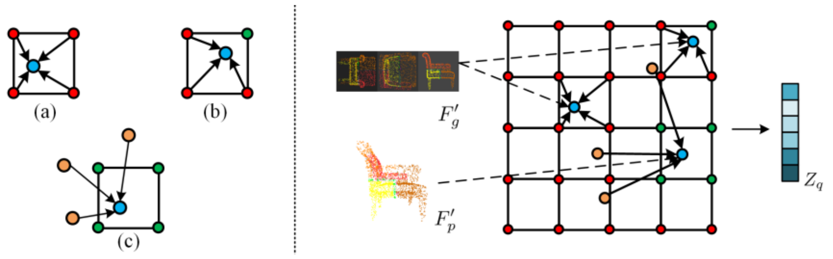

Fig. 3. Left: Three cases of interpolation. Red points represent grid corner points, green points denote zero-feature grid corners, blue points indicate query points, and orange points refer to other sampled points. The detailed explanation is presented in Sec 3.3. **Right:** Adaptive interpolation strategy. The input includes sampled points features and grid corners features, and the output is the feature vector of the query point.

3.3 Adaptive Interpolation Strategy

Given the point latents, grid latents, and any query point $q \in R^3$ in the space, our goal is to compute the learned features and estimate the occupancy function value. The typical approach is to use linear interpolation algorithm to extract the point features from the grid and decode it through point-wise MLP directly. However, in the sampling stage, many query points fall into grid regions without any features. When linear interpolation only considers the corner features of the current grid, yields a zero vector, indicating that the query point is not on the surface and neglecting its spatial positional information. In fact, the sampling point is merely located farther from the surface, and we can address this issue by leveraging its neighboring points. The network structure is illustrated in Fig. 3.

We consider the following cases with 2D grid:

(a). If the center of the grid containing the query point q has features, its computation is equivalent to linear interpolation. The formula is as follows:

$$(x_d, y_d) = \left(\frac{x-x_0}{x_1-x_0}, \frac{y-y_0}{y_1-y_0}\right), \tag{14}$$

$$z_q = c_{11}(1-x_d)(1-y_d) + c_{21}x_d(1-y_d) + c_{12}(1-x_d)y_d + c_{22}x_d y_d. \tag{15}$$

Here, c_{11}, c_{21}, c_{12}, c_{22} denote the corner points of the grid.

(b). If the corner points of the grid containing query point q are not all zero-featured, it indicates that the query point is located near the surface of the point cloud. Traditional linear interpolation is affected by zero features, resulting in unstable feature representations near grid boundaries. Therefore, in this case, we need to improve the linear interpolation algorithm that disregards the effect of zero vectors. Since traditional interpolation assigns weights summing to one for a query point, the total weight should remain unchanged after discarding the weights associated with zero vectors. To ensure that the remaining valid weights comply with the normalization requirement, we apply a normalization process. Accordingly, we introduce the sgn function to handle this issue.

$$\operatorname{sgn}(c) = \begin{cases} 1 & \text{if } c \text{ is zero feature} \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

Therefore, the calculation formula of weight is as follows:

$$w_q = \operatorname{sgn}(c_{11})(1-x_d)(1-y_d) + \operatorname{sgn}(c_{21})x_d(1-y_d) \tag{17}$$
$$+\operatorname{sgn}(c_{12})(1-x_d)y_d + \operatorname{sgn}(c_{22})x_d y_d.$$

Finally, the updated feature of the query point z'_q is obtained as:

$$z'_q = z_q / w_q. \tag{18}$$

(c). If all the corner points of the grid containing query point q have zero features, it indicates that the query point is some distance away from the surface of the point cloud. In this case, linear interpolation yields a zero feature for the query point, which to some extent neglects the information from other sampled points in the surrounding space. Inspired by POCO [35], we posit that the features of a query point are strongly influenced by nearby sampled points. In the decoding stage, rather than using learned weights, we adopt the Laplace weighting method. Let the K-nearest neighbor set of the sampled point q be denoted as $\mathcal{N}_K(q) = \{q_k \in R^3\}_{k=1}^{K}$. Then:

$$w_i = e^{-\lambda|q-q_i|}, \tag{19}$$

where λ denotes the weight decay rate, and w_i represents the weight of the i-th neighbor of q. This computation ensures that distant neighbors are assigned lower weights, while closer points receive higher weights. Similarly, we normalize the weights and perform a weighted sum with the neighboring features to obtain the interpolated feature of q:

$$z_q = \sum_{i=1}^{K} \mathrm{softmax}(w_i) \cdot p_{q_i}. \tag{20}$$

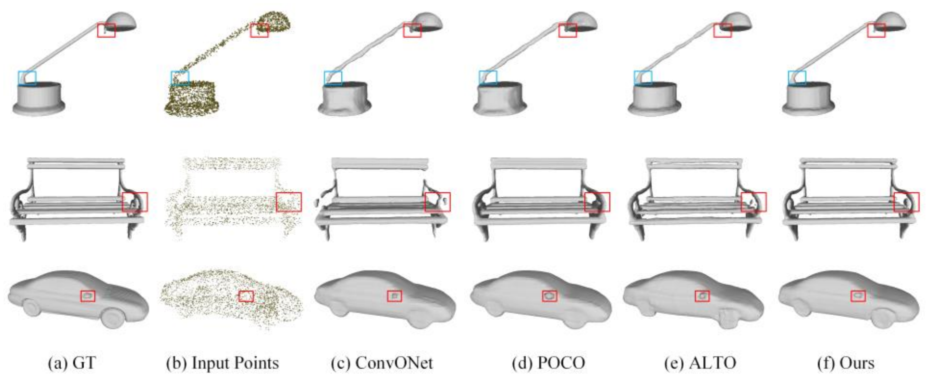

Fig. 4. Object-level comparisons on ShapeNet with 3k input points. GT represents the ground truth watertight mesh surface. The red and blue boxes highlighting the key differences in the comparison results across different methods. Comparison of our model to ConvONet, POCO and ALTO.

In this context, the softmax function is used for normalization, and p_{q_i} refers to the feature of the neighbor point q_i.

In summary, the proposed adaptive interpolation strategy robust interpolated features under diverse query point sampling conditions. Then we continue to use ResNet [41] to decode the occupancy values and generate continuous surfaces via the Marching Cubes algorithm.

Table 1. Object-level quantitative comparison on the ShapeNet dataset with different point density levels. σ is the standard deviation of the Gaussian noise.

Method	**3K points and $\sigma = 0.005$**				**1K points and $\sigma = 0.005$**			
	IoU↑	**CD↓**	**NC↑**	**FS ↑**	**IoU↑**	**CD↓**	**NC↑**	**FS↑**
ONet	0.761	0.87	0.891	0.785	0.772	0.81	0.894	0.801
ConvONet	0.884	0.44	0.938	0.942	0.859	0.50	0.929	0.918
SA-ConvONet	0.884	0.45	0.942	0.966	0.842	0.55	0.918	0.903
POCO	0.926	0.30	0.950	0.984	0.884	0.40	0.928	0.950
ALTO	0.930	0.30	0.952	0.980	0.905	0.35	0.940	0.964
Ours	**0.943**	**0.28**	**0.954**	**0.987**	**0.918**	**0.34**	**0.942**	**0.966**

3.4 Training and Inference

To train the network, we refer to previous work [8] and use the binary cross-entropy loss function to compute the distance between the predicted value and the ground truth. The formula is as follows:

$$\mathcal{L}(\widehat{O_q}, O_q) = -\frac{1}{N}\sum_{q=1}^{N}\left[O_q \log(\widehat{O_q}) + (1 - O_q)\log(1 - \widehat{O_q})\right]. \tag{21}$$

The proposed model is implemented in PyTorch, and trained using the Adam optimizer with a learning rate of 5×10^{-5}. During the experiments, we set the convolutional resolution R to 64, the dimension of the point latents d to 32, and the weight decay rate λ to 5. The training is conducted on a Nvidia RTX4080 SUPER GPU.

Table 2. Object-level quantitative comparison on the ShapeNet dataset with different noise levels and 3K input points.

Method	$\sigma = 0.005$		$\sigma = 0.025$	
	IoU↑	**CD↓**	**IoU↑**	**CD↓**
ConvONet	0.884	0.44	0.787	0.73
SA-ConvONet	0.884	0.45	0.736	0.90
POCO	0.926	0.30	0.817	0.58
Bifusion	0.920	**0.27**	0.810	**0.49**
ALTO	0.930	0.30	0.830	0.51
Ours	**0.943**	0.28	**0.840**	0.52

4 Experiments

In this section, we describe the datasets, baselines and evaluation metrics. After that we validate the proposed NeuLF-Net and perform both quantitative and qualitative comparisons with SOTA methods. Finally, we validated the effectiveness of the method through ablation study.

4.1 Datasets, Baselines and Metrics

Datasets. ShapeNet [42] is a classic object-level dataset for point cloud surface reconstruction tasks, containing 13 categories of object watertight meshes. Similar to the processing method in ONet [8], we sampled 3000 points and applied Gaussian noise with a standard deviation of 0.005. For reconstructing scene-level datasets, we employed the Synthetic Rooms dataset, first introduced in [23], which includes data from 5000 synthetic room scenes. Each scene is composed of objects, ground, and randomly sampled walls from the ShapeNet. We sampled 10000 points and similarly applied Gaussian noise with a standard deviation of 0.005. ScanNet [43] is a real-world scene dataset containing more than 1500 scans. We set the same parameters as the Synthetic Rooms dataset for testing.

Baselines. To assess the effectiveness of the proposed NeuLF-Net, we conduct a comparative analysis against both grid-based and point-based approaches. The grid-based methods include ONet [8], ConvONet [23], SA-ConvONet [24], and ALTO [29], while the advanced point-based method include POCO [35], Bifusion [36]. To ensure fairness, we set the same number of sampled points and noise levels when comparing the performance of different models.

Evaluation Metric. To assess the quality of the surface reconstruction results and compare them with previous studies, we use the following primary evaluation metrics: IoU: Measures the overlap between reconstructed surface and GT surface. Chamfer

Distance: Assesses the similarity between point clouds. Normal Consistency: Measures the similarity between the normal vectors of the reconstructed surface and the ground truth surface. F-score: A comprehensive evaluation of the model’s detail recovery capability and coverage ability with threshold value 1%.

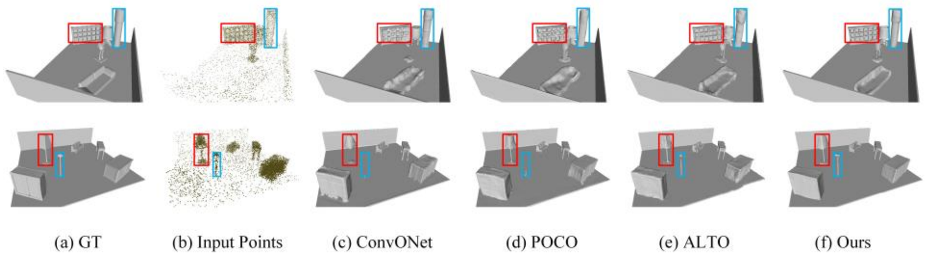

Fig. 5. Scene-level comparisons on Synthetic Rooms with 10k input points. Qualitative comparison of point clouds reconstruction in indoor scenes.

4.2 Object-level Reconstruction

Quantitative Evaluation. Firstly, we evaluated our approach using a simple shape dataset. As shown in Table 1, we conducted performance evaluations on the ShapeNet dataset with different sampling densities. We considered three different input point cloud sample sizes: 3000 and 1000, for comparison with the metrics of other models. We observed that our approach outperforms other techniques based on points and grids. Notably, it demonstrates a significant advantage in the IoU metric.

We further examine the impact of varying noise intensities. Notably, the model demonstrates robust stability even under increased noise conditions. The corresponding experimental findings are presented in Table 2.

Qualitative Evaluation. The ShapeNet dataset reconstruction results are shown in Fig. 4. In the first row, for the lamp reconstruction, other methods display significant irregularities and lack of smoothness on the lampshade, especially in certain detailed areas (as shown in the red box), where either the details are not reconstructed or they are extremely blurry. In contrast, our method can precisely reconstruct these details, presenting a smoother and more continuous geometric shape. Similar issues are also reflected in the chair contour reconstruction in the second row and the car rearview mirror reconstruction in the third row. Compared with other methods, our approach can more clearly exhibit the geometric contours, preventing blurriness and missing details. In general, our method shows more realistic and accurate surface reconstruction, with higher geometric precision and stability than other models.

4.3 Scene-level Reconstruction

Quantitative Evaluation. For the reconstruction of the scene-level datasets, we used the Synthetic Rooms Dataset for surface reconstruction and compared it with the baseline on the same dataset. We considered a scenario with 10,000 input sample points and Gaussian noise with a standard deviation of 0.005 for the comparison. Our results are presented in Table 3, and it is evident that, compared to the ShapeNet dataset, our model exhibits a more pronounced superiority in the complicated scene dataset.

Table 3. Comparison on the Synthetic Rooms dataset with 10K input points and σ = 0.005.

Methods	**IoU↑**	**CD↓**	**NC↑**	**FS ↑**
ONet	0.475	2.03	0.783	0.541
ConvONet	0.849	0.42	0.915	0.964
SA-ConvONet	0.850	0.42	0.912	0.961
POCO	0.884	0.36	0.919	0.980
ALTO	0.914	0.35	0.921	0.981
Ours	**0.920**	**0.33**	**0.936**	**0.985**

Qualitative Evaluation. We visualize the qualitative comparisons of Synthetic Rooms dataset in Fig. 5, it can be observed that the grid-based method yields relatively stable results but cannot correctly reconstruct the detailed part of the lamppost (blue box). The point-based method can properly display the lamppost, but it generates some incorrect holes. Our method performs well in both stability and detail reconstruction, demonstrating that it effectively combines the advantages of the both latents.

To validate the generalization ability of our model, we train it on Synthetic Rooms and test it on ScanNet. We visualize the qualitative comparisons in Fig. 6. In the first and second lines, the ConvONet processes complex wall structures in a very rough manner, while our method is clearer and reveals more detailed information. In the third row, ConvONet even fails to reconstruct the staircase, while our method produces a more complete result, capturing even the small table and lamp details. The above results indicate that our method can effectively process point cloud data from the real world, fully demonstrating the superiority of our approach.

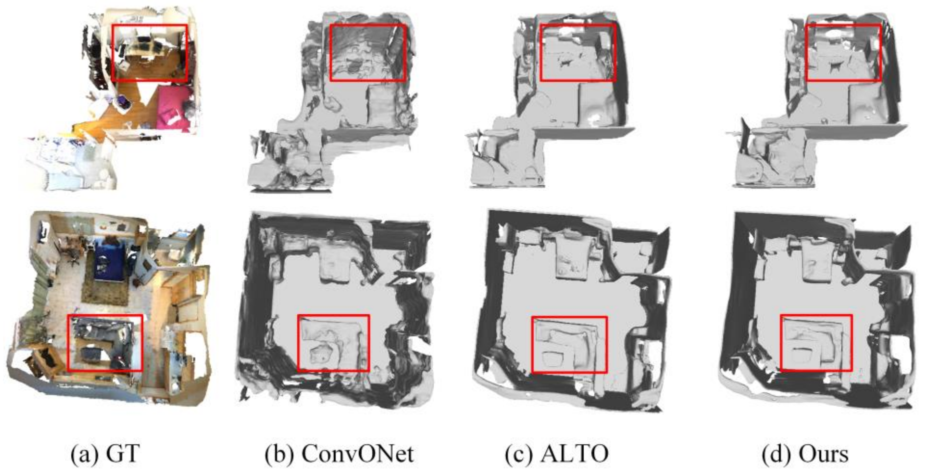

Fig. 6. Scene-level comparisons on ScanNet with 10k input points. Our method retains more details of the room and walls.

4.4 Ablation Study

In this section, we evaluate the effectiveness of the proposed model and compare it with the baseline model, with the specific results shown in Table 4. Our NGEL module significantly improved performance. This indicates that, compared to methods that only use grid latents, using neighbor information to update both grid and point latents enables the learning of more advanced point clouds spatial representations.

Moreover, although the adaptive interpolation strategy takes more time than linear interpolation method, it improves reconstruction accuracy, making the use of this method worthwhile. Compared with other KNN-based methods, although our method has the same worst-case time complexity, due to its adaptive strategy, the worst-case time complexity is not reached in most cases, and the inference time is improved compared with other two methods. This can be demonstrated in Table 5.

Table 4. Ablation study on Synthetic Rooms dataset with 3K points. We investigate different designs including NGEL and adaptive interpolation strategy.

Methods	**IoU↑**	**CD↓**	**NC↑**	**FS ↑**
ConvONet	0.818	0.46	0.906	0.943
POCO	0.801	0.57	0.904	0.812
ALTO	0.882	0.39	0.911	0.969
ConvONet+NGEL	0.900	0.38	0.911	0.973
Ours	**0.905**	**0.34**	**0.913**	**0.978**

Table 5. Runtime comparison of different interpolation strategy. N represents the number of points. While K represents the number of nearest neighbor nodes.

Method	**Strategy**	**Parameters**	**Time complexity**	**Inference time(s)**
ConvONet	Linear	4166657	$O(N)$	2.35
POCO	KNN	12790454	$O(KN)$	42.6
ALTO	KNN	4787905	$O(KN)$	5.28
Ours	Adaptive	7124983	$O(KN)$	4.79

5 Conclusion

In this paper, a novel network NeuLF-Net has been proposed for point clouds implicit surface reconstruction, which efficiently fuses point latents and grid latents. First, we designed the NGEL to enhances point latents through neighbor grid information, and re-applies the latents to grid through spatial mapping, overcoming the limitations of a single latents in processing point cloud features. Then, we proposed an adaptive interpolation strategy that analyzes the sampled points, fully exploits the advantages of both latents, and adaptively computes high-dimensional features, demonstrating robustness. Comprehensive experiments and comparisons demonstrated its effectiveness and Generalizability on the ShapeNet, Synthetic Rooms and ScanNet dataset, showcasing its strong potential.

Acknowledgments. This work was supported in part by the Ningbo Key Research and Development Program under Grant 2023Z067, and in part by the National Natural Science Foundation of China under Grant 61773225.

References

1. Sulzer, R., Marlet, R., Vallet, B., & Landrieu, L. (2023). A survey and benchmark of automatic surface reconstruction from point clouds. arXiv preprint arXiv:2301.13656.
2. Lee, D.-T., & Schachter, B. J. (1980). Two algorithms for constructing a Delaunay triangulation. International Journal of Computer & Information Sciences, 9(3), 219–242. Springer.
3. Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., & Taubin, G. (1999). The ball-pivoting algorithm for surface reconstruction. IEEE Transactions on Visualization and Computer Graphics, 5(4), 349–359. IEEE.
4. Hanocka, R., Metzer, G., Giryes, R., & Cohen-Or, D. (2020). Point2mesh: A self-prior for deformable meshes. arXiv preprint arXiv:2005.11084.
5. Maćkiewicz, A., & Ratajczak, W. (1993). Principal components analysis (PCA). Computers & Geosciences, 19(3), 303–342. Elsevier.
6. Kazhdan, M., Bolitho, M., & Hoppe, H. (2006). Poisson surface reconstruction. In Proceedings of the fourth Eurographics Symposium on Geometry Processing (Vol. 7, No. 4).
7. Kazhdan, M., & Hoppe, H. (2013). Screened Poisson surface reconstruction. ACM Transactions on Graphics (ToG), 32(3), 1–13. ACM New York, NY, USA.
8. Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., & Geiger, A. (2019). Occupancy networks: Learning 3D reconstruction in function space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 4460–4470).
9. Choy, C. B., Xu, D., Gwak, J., Chen, K., & Savarese, S. (2016). 3D-R2N2: A unified approach for single and multi-view 3D object reconstruction. In Computer Vision--ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII (pp. 628–644). Springer.
10. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., & Xiao, J. (2015). 3D shapenets: A deep representation for volumetric shapes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1912–1920).
11. Häne, C., Tulsiani, S., & Malik, J. (2017). Hierarchical surface prediction for 3D object reconstruction. In 2017 International Conference on 3D Vision (3DV) (pp. 412–420). IEEE.
12. Achlioptas, P., Diamanti, O., Mitliagkas, I., & Guibas, L. (2018). Learning representations and generative models for 3D point clouds. In International Conference on Machine Learning (pp. 40–49). PMLR.
13. Lin, C.-H., Kong, C., & Lucey, S. (2018). Learning efficient point cloud generation for dense 3D object reconstruction. In Proceedings of the AAAI Conference on Artificial Intelligence, 32(1).
14. Sharp, N., & Ovsjanikov, M. (2020). Pointtrinet: Learned triangulation of 3D point sets. In Computer Vision--ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII (pp. 762–778). Springer.
15. Liu, M., Zhang, X., & Su, H. (2020). Meshing point clouds with predicted intrinsic-extrinsic ratio guidance. In Computer Vision--ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VIII (pp. 68–84). Springer.
16. Sulzer, R., Landrieu, L., Marlet, R., & Vallet, B. (2021). Scalable surface reconstruction with Delaunay-graph neural networks. Computer Graphics Forum, 40(5), 157–167. Wiley Online Library.

17. Rakotosaona, M.-J., Guerrero, P., Aigerman, N., Mitra, N. J., & Ovsjanikov, M. (2021). Learning Delaunay surface elements for mesh reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 22–31).
18. Groueix, T., Fisher, M., Kim, V. G., Russell, B. C., & Aubry, M. (2018). A papier-mâché approach to learning 3D surface generation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 216–224).
19. Lorensen, W. E., & Cline, H. E. (1998). Marching cubes: A high resolution 3D surface construction algorithm. In Seminal Graphics: Pioneering Efforts That Shaped the Field (pp. 347–353).
20. Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). PointNet: Deep learning on point sets for 3D classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 652–660).
21. Park, J. J., Florence, P., Straub, J., Newcombe, R., & Lovegrove, S. (2019). DeepSDF: Learning continuous signed distance functions for shape representation. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 165–174).
22. Giebenhain, S., & Goldlücke, B. (2021). Air-nets: An attention-based framework for locally conditioned implicit representations. In 2021 International Conference on 3D Vision (3DV) (pp. 1054–1064). IEEE.
23. Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., & Geiger, A. (2020). Convolutional occupancy networks. In Computer Vision--ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III (pp. 523–540). Springer.
24. Tang, J., Lei, J., Xu, D., Ma, F., Jia, K., & Zhang, L. (2021). SA-ConvONet: Sign-agnostic optimization of convolutional occupancy networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 6504–6513).
25. Lionar, S., Emtsev, D., Svilarkovic, D., & Peng, S. (2021). Dynamic plane convolutional occupancy networks. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 1829–1838).
26. Gropp, A., Yariv, L., Haim, N., Atzmon, M., & Lipman, Y. (2020). Implicit geometric regularization for learning shapes. arXiv preprint arXiv:2002.10099.
27. Jiang, C., Sud, A., Makadia, A., Huang, J., Niessner, M., Funkhouser, T., et al. (2020). Local implicit grid representations for 3D scenes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6001–6010).
28. Peng, S., Jiang, C., Liao, Y., Niemeyer, M., Pollefeys, M., & Geiger, A. (2021). Shape as points: A differentiable Poisson solver. Advances in Neural Information Processing Systems, 34, 13032–13044.
29. Wang, Z., Zhou, S., Park, J. J., Paschalidou, D., You, S., Wetzstein, G., Guibas, L., & Kadambi, A. (2023). Alto: Alternating latent topologies for implicit 3D reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 259–270).
30. Lombardi, S., Oswald, M. R., & Pollefeys, M. (2020). Scalable point cloud-based reconstruction with local implicit functions. In 2020 International Conference on 3D Vision (3DV) (pp. 997–1007). IEEE.
31. Wu, W., Wang, Q., Wang, G., Wang, J., Zhao, T., Liu, Y., ... & Wang, H. (2024, September). Emie-map: Large-scale road surface reconstruction based on explicit mesh and implicit encoding. In European Conference on Computer Vision (pp. 370-386). Cham: Springer Nature Switzerland.
32. Siddiqui, Y., Thies, J., Ma, F., Shan, Q., Niessner, M., & Dai, A. (2021). RetrievalFuse: Neural 3D scene reconstruction with a database. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 12568–12577).

33. Erler, P., Guerrero, P., Ohrhallinger, S., Mitra, N. J., & Wimmer, M. (2020). Points2surf learning implicit surfaces from point clouds. In European Conference on Computer Vision (pp. 108–124). Springer.
34. Wang, B., Yu, Z., Yang, B., Qin, J., Breckon, T., Shao, L., Trigoni, N., & Markham, A. (2022). RangeUDF: Semantic surface reconstruction from 3D point clouds. arXiv preprint arXiv:2204.09138.
35. Boulch, A., & Marlet, R. (2022). Poco: Point convolution for surface reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 6302–6314).
36. Fan, C., Zhao, C., Xue, K., & Duan, Y. (2024, June). Point Voxel Bi-directional Fusion Implicit Field for 3D Reconstruction. In Proceedings of the 50th Graphics Interface Conference (pp. 1-11).
37. Ummenhofer, B., & Koltun, V. (2021). Adaptive surface reconstruction with multiscale convolutional kernels. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 5651–5660).
38. Boulch, A., Puy, G., & Marlet, R. (2020). FKAConv: Feature-kernel alignment for point cloud convolution. In Proceedings of the Asian Conference on Computer Vision.
39. Wang, Y., Huang, Z., Shamir, A., Huang, H., Zhang, H., & Hu, R. (2023). ARO-Net: Learning implicit fields from anchored radial observations. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 3572–3581).
40. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention--MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III (pp. 234–241). Springer.
41. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 770–778).
42. Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. (2015). ShapeNet: An information-rich 3D model repository. arXiv preprint arXiv:1512.03012.
43. Dai, A., Chang, A. X., Savva, M., Halber, M., Funkhouser, T., & Nießner, M. (2017). ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.