Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Unsupervised Learning of Neural Implicit Functions from Point Clouds via Graph Filtering

An Liu
Institute of Automation, Chinese Academy of Sciences
Rudong Xue
Nanjing Research Institute on Simulation Technique Co., Ltd
Bingxi Liu
Southern University of Science and Technology
Fulin Tang
tangfulin2015@ia.ac.cn

Institute of Automation, Chinese Academy of Sciences

Research Article

Keywords:

Posted Date: November 25th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11597473/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Unsupervised Learning of Neural Implicit Functions from Point Clouds via Graph Filtering

An Liu[1], Rudong Xue[2,3], Bingxi Liu[4,5], Fulin Tang[1*]

[1]Institute of Automation, Chinese Academy of Sciences, Beijing, 100190, China.
[2]Suzhou University, Suzhou, 215006, China.
[3]Nanjing Research Institute on Simulation Technique Co., Ltd, Nanjing, 211200, China.
[4]Southern University of Science and Technology, Shenzhen, 518055, China.
[5]Pengcheng Laboratory, Shenzhen, 518000, China.

*Corresponding author(s). E-mail(s): tangfulin2015@ia.ac.cn;
Contributing authors: liuan2026@ia.ac.cn; xuerudong@alu.suda.edu.cn; liubx@pcl.ac.cn;

Abstract

This paper presents an unsupervised method for directly learning neural implicit functions from 3D point clouds, aiming to tackle the core challenge of lacking ground-truth implicit values for supervisory training. Current approaches typically rely on global geometric constraints to learn implicit functions; however, this strategy often causes the optimization process to fall into local minima, resulting in over-smoothing artifacts, loss of fine geometric details, and blurring of sharp features. Although hand-crafted filters can be applied to point clouds to capture high-frequency details—thereby mitigating over-smoothing of geometric structures and avoiding the generation of coarse surfaces—their performance is highly dependent on manual parameter tuning, which severely restricts their generalization ability across diverse 3D shapes. To address these limitations, we propose a dynamic graph filter that integrates the adaptability of graph neural networks with the geometric characteristics of hand-crafted bilateral filters, enabling the learning of high-quality neural implicit functions from 3D point clouds. Our approach dynamically evolves graph features during the training process, automatically capturing high-frequency geometric details without the need for manual parameter adjustment. Ultimately, high-quality surface reconstruction can be achieved from the learned neural implicit function, even in the complete absence of ground-truth supervision. Experimental results on multiple datasets demonstrate that our proposed method effectively preserves complex geometric details and sharp features, achieving superior surface reconstruction quality compared to state-of-the-art approaches.

Keywords: Neural Implicit Function, Graph Neural Network, Point Clouds, Surface Reconstruction, Bilateral Filtering

1 Introduction

Implicit functions are widely used in 3D computer vision and graphics as powerful representations for modeling and manipulating complex shapes and surfaces [1–3]. A surface mesh can be effectively extracted from an implicit function by applying the Marching Cubes algorithm [4] to the zero level set of the function, enabling high-quality surface reconstruction. As a result, significant research efforts have been devoted to learning implicit functions directly from 3D point

clouds—particularly through neural implicit functions, which offer improved flexibility, accuracy, and detail preservation for reconstructing complex geometries.

Learning the implicit function only from 3D point clouds without supervision is difficult but important. However, only 3D point clouds will be available in most cases, which means there is an absence of ground-truth implicit function values for supervision [5]. Current methods employ various techniques to address these challenges. These approaches often involve learning implicit fields with additional constraints to infer signed distances for arbitrary query points in the 3D space. For instance, methods may use geometric regularization [6], sign-agnostic learning with specific loss functions [7], gradient constraints [8], pulling sampled queries toward the surface [5, 9]. However, these methods often struggle to balance surface smoothness and detail preservation, leading to over-smoothed geometric features, bumpy surfaces, or inconsistent reconstructed surfaces.

The issues of over-smoothed geometric features, bumpy surfaces, or inconsistent reconstructions in surface reconstruction largely stem from the global optimization strategies used by most existing methods. These approaches only constrain the input points on the zero-level set of the target implicit function in a global manner, often overlooking critical local geometric details. This global optimization strategy will result in underfitting for regions with sharp features or complex detailed geometry. Additionally, when the point cloud contains noise, global optimization can lead to overfitting, introducing artifacts or bumpy surfaces that degrade reconstruction quality. Hand-crafted filters, such as the bilateral filter [10, 11], attempt to address this by emphasizing local neighborhoods to preserve high-frequency geometric details as a supplement to global optimization. However, their performance heavily depends on manual parameter settings, such as neighborhood size and normal variance. Tuning these parameters is non-trivial, and a fixed set of values may fail to generalize effectively across diverse shapes, limiting their applicability.

We draw inspiration from the recent success of graph neural networks (GNNs) in various fields [12, 13], particularly in capturing high-frequency features. In the context of learning neural implicit functions, sharp features or fine geometric details correspond to these high-frequency components. This motivates us to leverage GNNs for learning neural implicit functions, aiming to avoid over-smoothing or inconsistency in these high-frequency geometric features. Unlike prior approaches [5, 6, 11, 14–17], our method employs dynamic graph filters to preserve the high-frequency geometric features. Specifically, our dynamic graph is constructed by using the *EdgeConv* layers, which dynamically update the graph of relationships from layer to layer, extracting robust graph features from the point cloud. These graph features are then transformed to predict an adaptive normal-angle parameter for the bilateral filter [10, 11], while its spatial scale is determined from the point cloud geometry. This dynamic graph filter adjusts automatically to the input point cloud, allowing for the preservation of high-frequency geometric details. Importantly, our learning framework is entirely unsupervised, requiring no ground truth values for supervision. Yet, it achieves superior retention of fine geometric details and sharp features in the learned neural implicit function.

The overall pipeline is illustrated in Fig. 1.

In summary, the main contributions of this paper are as follows:

- We propose a new approach to learning high-quality neural implicit functions directly from raw 3D point clouds in an unsupervised manner, leveraging a dynamic graph filter applied to the input point clouds.
- Our dynamic graph filter generalizes the traditional hand-crafted bilateral filter by dynamically updating graph relationships based on the input point clouds, enabling better preservation of geometric details and sharp features.
- We evaluate our approach on multiple datasets. Our proposed method shows competitive results compared to selected baselines [5, 6, 11, 14–17] both quantitatively and qualitatively, demonstrating the effectiveness of the proposed dynamic graph filter for learning neural implicit functions.

2 Related Work

With the rapid advancement of deep learning, neural networks have shown great promise in surface

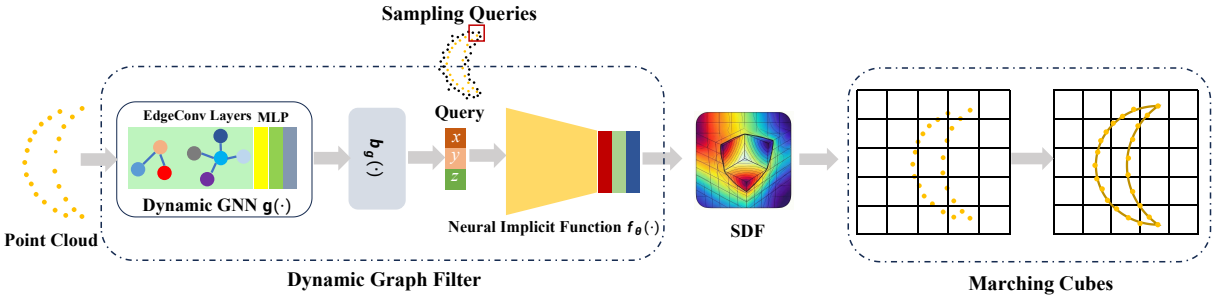

Fig. 1 The pipeline for learning our proposed neural implicit function $f_\theta(\cdot)$ proceeds as follows. First, we construct a dynamic graph neural network (GNN) $g(\cdot)$, which is built upon the *EdgeConv* layer, and integrate it into the bilateral filter to form a graph bilateral filter. This graph filter is then applied to the sampled queries, which are generated by random sampling around the input point cloud. The neural implicit function and the dynamic GNN filter are trained jointly to enhance the model's ability to capture fine geometric details. Finally, the marching cubes algorithm is applied to the learned neural implicit function to extract the surface mesh from predicted signed distance function (SDF) values

reconstruction from 3D point clouds. In the following, we provide a brief review of representative methods related to neural implicit function learning, encompassing both supervised and unsupervised approaches. We further discuss techniques for sharp feature preservation and the application of graph neural networks (GNNs) for point cloud processing.

2.1 Supervised Implicit Function Learning

The straightforward strategy for learning neural implicit functions is to learn priors in a data-driven manner. These methods rely on signed distances or occupancy labels as 3D supervision to learn either global priors or local priors. For example, the occupancy network and its convolutional variant [18, 19] predict an occupancy probability on a grid. A mesh is then extracted from the grid based on these occupancy predictions, typically using marching cubes [4] to convert the occupancy probabilities into a triangular mesh. Points2Surf [14] employs neural networks to regress signed distance functions, using ground-truth meshes as supervision. Similarly, POCO [20] encodes each input point into a latent vector and estimates occupancy based on ground-truth data. While these approaches can perform well on shapes similar to those in the training dataset, their generalization ability to unseen cases with significant geometric variations is often limited. Furthermore, these models typically assume fixed input formats, making it challenging to adapt them to varying point cloud sizes, which restricts scalability and flexibility in practical applications.

2.2 Unsupervised Implicit Function Learning

Learning neural implicit functions from 3D point clouds in an unsupervised manner differs significantly from methods that rely on direct 3D supervision [14, 18–20]. Unlike supervised approaches, the core idea behind learning neural implicit functions from 3D point clouds is to overfit neural networks to individual point clouds—either globally or locally—to derive signed implicit functions [5, 7, 8, 15, 21–25]. For instance, NeuralPull [5] learns implicit functions by pulling queries along the gradient of the implicit function towards the zero level set, while PCP [15] refines a pre-trained local context prior by learning predictive queries. Later, GridPull [21] infers a discrete distance field by pulling queries onto the surface within grids of interest. These unsupervised methods rely on the network's capacity to infer implicit functions. As a result, beyond the original input point clouds, they often require additional constraints on gradient directions [5, 15, 21, 23], gradient norms [7, 8, 25], specially designed priors [15, 24], or a differentiable Poisson solver [22], to accurately infer the signed implicit function. However, this unsupervised approach heavily relies on the

neural network's ability to fit and maintain continuity, which can lead to inaccuracies in the learned implicit function, especially when there is insufficient guidance across the discrete point cloud space. Consequently, accurately deducing geometry in free space becomes particularly challenging. Unlike previous works, our method utilizes the hand-crafted bilateral filter as a geometric prior and combines it with graph neural networks to capture local information from neighboring points, allowing for more precise geometry inference.

2.3 Sharp Feature Preserving

Preserving sharp features is crucial for high-fidelity surface reconstruction. Early methods [10, 26] addressed this problem through point cloud consolidation, enhancing point quality via local projection operations (LOP) [27] and its variants [10, 26, 28, 29], which provide stable and generalizable filtering strategies. Later, learning-based approaches [30, 31] incorporated edge awareness into point consolidation, but typically relied on separate reconstruction techniques [32] to produce final meshes. More recently, neural implicit representations [16, 17, 33, 34] have been proposed to embed high-frequency geometric features directly into the network architecture—using, for example, feature-frequency networks (FFN) [35], sinusoidal activations (SIREN) [16], or isosurface distance fields (IDF) [36]—or to regularize the learned field via geometric priors such as divergence constraints, as in DIGS [17] and EPI [33]. In contrast, our method introduces local geometric awareness through a dynamic GNN-based filtering mechanism, enabling more accurate optimization of the implicit field and producing reconstructions that better preserve sharp features and fine-scale geometric details.

2.4 GNN for Point Clouds

Research on graph neural networks (GNNs) [13, 37] extends the convolutional neural network paradigm, which is typically applied to regular data like images, to graph representations, where data is irregular. Generally, a GNN iteratively updates vertex features by aggregating information from connected vertices along edges, facilitating more complex feature interactions. In 3D computer vision, several approaches represent point clouds, which are irregular, as graphs. For instance, Qi et al. [38] employ a recurrent GNN for semantic segmentation of RGBD data, while Landrieu et al. [39] segment point clouds by partitioning them into simple geometric shapes and linking these shapes in a graph. Additionally, Wang et al. [40] propose the dynamic graph CNN (DGCNN) with *EdgeConv* layers for point cloud classification and segmentation. However, there is limited research on designing GNNs specifically for neural implicit functions.

3 Method

A neural implicit function is a signed implicit function that maps a query point, $\boldsymbol{x} \in \boldsymbol{R}^3$, to a signed distance $s \in \boldsymbol{R}$. It can be denoted by the function $s = f_\theta(\boldsymbol{x})$, where f_θ represents the neural network and θ denotes its parameters. The target is to learn f_θ from a given point cloud $\boldsymbol{P} = \{\boldsymbol{p}_i\}_1^N$ without ground truth of signed distance function (SDF) values for supervision.

3.1 Preliminary

To learn a neural implicit function, a natural constraint is that the input point cloud $\boldsymbol{P}$ should lie on the zero level set of the implicit function, which is enforced by minimizing the following loss:

$$L_1 = \frac{1}{N}\sum_{i=1}^{N} |f_\theta(\boldsymbol{p}_i)|. \tag{1}$$

However, this constraint is only applied to the input points and often leads to optimization getting stuck in local minima, resulting in suboptimal surface reconstruction. To address this, additional constraints can be imposed on the gradients of the implicit function [5, 11, 21]. Specifically, we can sample a query point set $\boldsymbol{X} = \{\boldsymbol{x}_i\}_1^M$ around the input point cloud, and pull the queries to the zero level set along the gradient by:

$$\boldsymbol{q}_i = \boldsymbol{x}_i - f_\theta(\boldsymbol{x}_i)\nabla f_\theta(\boldsymbol{x}_i)/||\nabla f_\theta(\boldsymbol{x}_i)||. \tag{2}$$

This operation will generate a new point set $Q = \{\boldsymbol{q}_i\}_1^M$. Ideally, the new point set Q should also lie on the zero level set of the implicit function [5, 11, 21]. Thus, we can put constraints on the gradients by minimizing the chamfer distance

between P and Q:

$$L_2 = \frac{1}{N}\sum_{\boldsymbol{p}_i \in \boldsymbol{P}} \min_{\boldsymbol{q}_j \in \boldsymbol{Q}} ||\boldsymbol{p}_i - \boldsymbol{q}_j||_2 + \frac{1}{M}\sum_{\boldsymbol{q}_j \in \boldsymbol{Q}} \min_{\boldsymbol{p}_i \in \boldsymbol{P}} ||\boldsymbol{q}_j - \boldsymbol{p}_i||_2. \tag{3}$$

A neural implicit function can be learned by minimizing the losses L_1 and L_2 to approximate the target surface. However, relying solely on these losses often results in over-smoothing and the loss of fine geometric details or sharp features. This is because the optimization primarily focuses on fitting the global input point clouds, which tend to overlook the local geometric information important for preserving geometric details and sharp features.

3.2 Dynamic Graph Filter

To learn a neural implicit function that also considers local geometry and topology, we propose the design of an automatic dynamic graph filter to capture local information during the training process. While the bilateral filter has shown to be effective in preserving sharp features [10, 11], it relies heavily on a parameter-sensitive weight function. Our goal is to leverage dynamic graph neural networks to automatically adjust the bilateral filter to preserve fine geometric details and shape features. We refer to this as the dynamic graph filter. The traditional bilateral filter $b(\boldsymbol{x})$ is defined as:

$$b(\boldsymbol{x}) = \frac{\sum_{\boldsymbol{y}\in\mathcal{N}(\boldsymbol{x})} d(\boldsymbol{x},\boldsymbol{y})w(\boldsymbol{x},\boldsymbol{y})}{\sum_{\boldsymbol{y}\in\mathcal{N}(\boldsymbol{x})} w(\boldsymbol{x},\boldsymbol{y})}, \tag{4}$$

where $d(\boldsymbol{x},\boldsymbol{y}) = |\boldsymbol{n}^T(\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})| + |\boldsymbol{n}^T(\boldsymbol{x})(\boldsymbol{y}-\boldsymbol{x})|$ is the projection distance, $\boldsymbol{n}(\cdot) = \nabla f_\theta(\cdot)/||\nabla f_\theta(\cdot)||$ denotes the normalized gradient, $\mathcal{N}(\boldsymbol{x})$ denotes the neighbors, and $w(\boldsymbol{x},\boldsymbol{y})$ denotes the weight function. The hand-crafted bilateral filter can preserve local geometric information through carefully designed weight functions. However, incorrect weight function choices can lead to excessive smoothing or poor detail preservation. Furthermore, different point clouds may exhibit varying densities and geometric characteristics, making finding a universal weight function complex. To overcome these limitations, we propose using a GNN to dynamically adjust the weight function for the bilateral filter, enabling the neural implicit function to preserve fine geometric details better.

Specifically, we construct a dynamic GNN $g(\cdot)$ built upon stacked *EdgeConv* [40] layers. This dynamic GNN takes the whole point cloud $\boldsymbol{P}$ as input to extract geometric features in a dynamic manner. We then leverage those geometric features to adjust the bilateral filter. Let $\mathcal{G} = (\mathcal{V}^l, \mathcal{E}^l)$ denote the knn graph, where $\mathcal{V}^l = \{\boldsymbol{p}_1^l, \ldots, \boldsymbol{p}_N^l\} \subseteq \boldsymbol{R}^F$ represents the vertices for lth layer, and $\mathcal{E}^l$ represents the edges. The edges $\mathcal{E}^l$ are constructed dynamically based on the k-nearest neighbors of $\mathcal{V}^l$ in the feature space $\boldsymbol{R}^F$. At each layer, the graph structure evolves with the vertex features $\mathcal{V}^l$, allowing the GNN to capture local geometric information dynamically. Initially, $F = 3$ corresponds to the spatial coordinates (x, y, z) of the input points. Besides, each edge is also associated with an edge feature e_{ij}^l derived from vertex features, which is calculated by:

$$e_{ij}^l = \mathrm{ReLU}(\boldsymbol{\gamma}(\boldsymbol{p}_j^l - \boldsymbol{p}_i^l) + \boldsymbol{\phi}(\boldsymbol{p}_i^l)), \tag{5}$$

where γ and ϕ are MLPs. The output of the *EdgeConv* layer is:

$$\boldsymbol{p}_i^{l+1} = \max_{j:(i,j)\in\mathcal{E}} e_{ij}^l. \tag{6}$$

Note that the knn graph is dynamically constructed at each layer. The final output of $g(\cdot)$ is a global feature vector $\boldsymbol{g} \in \boldsymbol{R}^{64}$ by max-pooling of the vertex features at the last layer:

$$\boldsymbol{g} = \mathrm{MaxPooling}(\boldsymbol{p}_1^e, \ldots, \boldsymbol{p}_N^e), \tag{7}$$

where $\boldsymbol{p}_i^e$ denotes the vertices of the last *EdgeConv* layer. We thus define a GNN-based weight function as:

$$w_g(\boldsymbol{x},\boldsymbol{y}) = \exp\left(-\frac{||\boldsymbol{x}-\boldsymbol{y}||^2}{\sigma_p^2}\right) \times \exp\left(-\frac{1-\boldsymbol{n}^T(\boldsymbol{x})\boldsymbol{n}(\boldsymbol{y})}{1-\cos(\mathcal{O}(\boldsymbol{g}))}\right), \tag{8}$$

where σ_p is $\sigma_p = \max_{\boldsymbol{x},\boldsymbol{y}}||\boldsymbol{x}-\boldsymbol{y}||$, and $\mathcal{O}$ is an MLP transforming $\boldsymbol{g}$ into a normal angle variance.

Finally, our dynamic graph filter is defined as:

$$b_g(\boldsymbol{x}) = \frac{\sum\limits_{\boldsymbol{y}\in\mathcal{N}(\boldsymbol{x})} d(\boldsymbol{x},\boldsymbol{y}) w_g(\boldsymbol{x},\boldsymbol{y})}{\sum\limits_{\boldsymbol{y}\in\mathcal{N}(\boldsymbol{x})} w_g(\boldsymbol{x},\boldsymbol{y})}, \tag{9}$$

3.3 Loss Functions

We employ four loss components to train the proposed neural implicit function.

The first component is defined by Eq. (1), which enforces that the points $\boldsymbol{P}$ lie on the zero level set of the neural implicit function f_θ. The second loss item is the Chamfer loss, as specified in Eq. (3), which aims at minimizing the distance between the generated point set $\boldsymbol{Q}$ and the input point set $\boldsymbol{P}$.

The third loss item applies the dynamic graph filter on the generated point set Q:

$$L_3 = \frac{1}{M}\sum_{\boldsymbol{q}_i\in\boldsymbol{Q}} b_g(\boldsymbol{q}_i), \tag{10}$$

The fourth loss term applies the dynamic graph filter to the sampled query points in $\boldsymbol{X}$:

$$L_4 = \frac{1}{M}\sum_{\boldsymbol{x}_i\in\boldsymbol{X}} b_g(\boldsymbol{x}_i), \tag{11}$$

Following [11], the neighbor points of $\mathcal{N}(\boldsymbol{x})$ on the level of $f_\theta(\boldsymbol{x})$ is obtained by:

$$\mathcal{N}(\boldsymbol{x}) = \{\hat{\boldsymbol{x}} | \hat{\boldsymbol{x}} = \boldsymbol{p} + f_\theta(\boldsymbol{x})\frac{\nabla f_\theta(\boldsymbol{p})}{||\nabla f_\theta(\boldsymbol{p})||}, \boldsymbol{p}\in\mathcal{N}_{\boldsymbol{P}}(\boldsymbol{x})\}, \tag{12}$$

where $\mathcal{N}_{\boldsymbol{P}}$ denotes the nearest neighbors in $\boldsymbol{P}$ for the query $\boldsymbol{x}$. Finally, the total loss is the sum of the four items:

$$L = \alpha_1 L_1 + \alpha_2 L_2 + \alpha_3 L_3 + \alpha_4 L_4, \tag{13}$$

where α_1, α_2, α_3, and α_4 are the balance weights, and we set them to 1.0.

4 Experiments

4.1 Implementation Details

We implement our approach using the PyTorch framework and conduct all experiments on NVIDIA V100 GPUs. The training is optimized using the Adam optimizer, with an initial learning rate of 10^{-3} that decays over time to ensure stable convergence. To evaluate the effectiveness of our proposed dynamic graph filter for surface reconstruction, we perform experiments on three widely used benchmark datasets: the FAMOUS dataset [14], the ABC dataset [41], and the ShapeNet dataset [42]. These datasets offer diverse geometric structures and varying complexity, providing a comprehensive evaluation of our method's robustness and reconstruction quality.

4.2 Comparisons

We compare our method against several state-of-the-art approaches, including Points2Surf [14], IGR [6], NeuralPull [5], PCP [15], SIREN [16], DIGS [17], and ImplicitFiltering [11]. We keep similar experiment settings with ImplicitFiltering [11]. For the choice of the evaluation metrics, we use L1 Chamfer distance (CD_{L1}) and L2 Chamfer distance (CD_{L2}), F-score (F-S.), and normal consistency (NC) to measure the surface reconstruction quality.

Specifically, we denote M_g and M_p as the ground-truth mesh (or point clouds) and the predicted mesh, respectively. Let $\mathcal{X} = \{\mathbf{x}_1,\ldots,\mathbf{x}_{N_g}\}$ and $\mathcal{Y} = \{\mathbf{y}_1,\ldots,\mathbf{y}_{N_p}\}$ represent randomly sampled points on these two meshes (or point clouds). The projection operators $\mathcal{P}_{g\to p}(\mathbf{x}) = \arg\min_{\mathbf{y}\in\mathcal{Y}} \|\mathbf{x}-\mathbf{y}\|_2$ and $\mathcal{P}_{p\to g}(\mathbf{y}) = \arg\min_{\mathbf{x}\in\mathcal{X}} \|\mathbf{y}-\mathbf{x}\|_2$ are defined to find the closest points. Here, N_g and N_p are the numbers of sampled points from M_g and M_p, respectively, and $\mathbf{n}(\cdot)$ denotes the normal vector operator for a given point.

The symmetric mean Euclidean Chamfer distance is defined as:

$$\begin{aligned}\mathrm{CD}_{\mathrm{E}}(\mathcal{X},\mathcal{Y}) = &\frac{1}{2N_g}\sum_{i=1}^{N_g}\left\|\mathbf{x}_i - \mathcal{P}_{g\to p}\left(\mathbf{x}_i\right)\right\|_2 + \\ &\frac{1}{2N_p}\sum_{i=1}^{N_p}\left\|\mathbf{y}_i - \mathcal{P}_{p\to g}\left(\mathbf{y}_i\right)\right\|_2,\end{aligned} \tag{14}$$

The reported CD_{L1} and CD_{L2} variants follow the comparison protocol of ImplicitFiltering [11]. Lower values indicate better reconstruction fidelity.

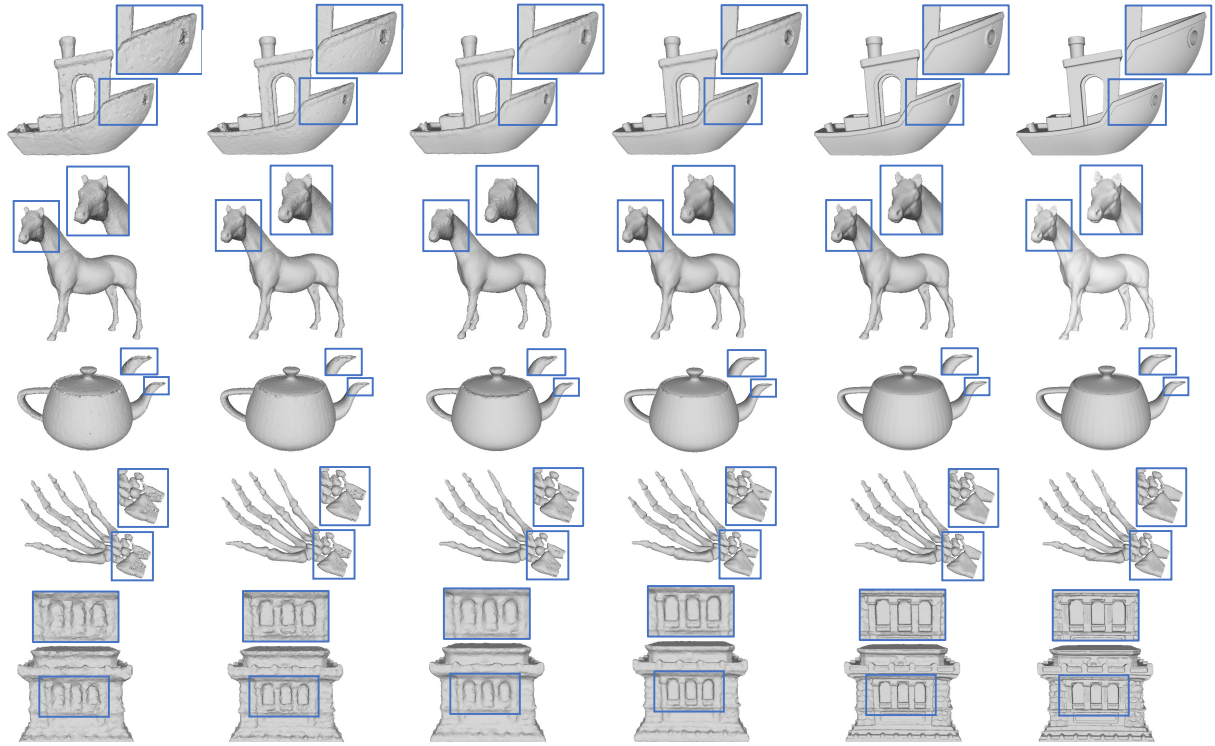

Fig. 2 Visual comparisons of surface reconstruction. From left to right, the columns show Points2Surf, IGR, NeuralPull, ImplicitFiltering, our method, and the ground truth (GT). Point clouds sampled from the reference mesh are used as input for our method and the baselines; the reference mesh serves as the ground truth for evaluation

Table 1 Comparisons on the ABC, Famous, and ShapeNet datasets. Lower CD_{L1} and CD_{L2} means better performance, while higher F-S. and NC indicate superior results.

Methods	FAMOUS				ABC				ShapeNet			
	CD_{L1}	CD_{L2}	F-S.	NC	CD_{L1}	CD_{L2}	F-S.	NC	CD_{L1}	CD_{L2}	F-S.	NC
Points2Surf [14]	0.008	0.012	0.752	0.891	0.015	0.298	0.598	0.867	0.007	0.010	0.765	0.889
IGR [6]	0.044	1.474	0.573	0.903	0.063	2.675	0.448	0.883	0.034	1.058	0.622	0.922
NeuralPull [5]	0.012	0.100	0.746	0.921	0.011	0.095	0.673	0.891	0.013	0.056	0.698	0.914
PCP [15]	0.014	0.037	0.435	0.941	0.023	0.252	0.373	0.905	0.012	0.021	0.587	0.951
SIREN [16]	0.012	0.025	0.561	0.959	0.012	0.022	0.493	0.916	0.010	0.019	0.674	0.962
DIGS [17]	0.008	0.015	0.772	0.974	0.010	0.021	0.667	0.935	0.007	0.010	0.796	0.968
ImplicitFiltering [11]	0.007	0.008	0.778	0.973	0.009	0.011	0.691	0.942	0.005	0.007	0.801	0.979
Ours	**0.006**	**0.007**	**0.801**	**0.979**	**0.008**	**0.010**	**0.719**	**0.956**	**0.003**	**0.004**	**0.856**	**0.986**

The Normal Consistency (NC) is given by:

$$\begin{aligned} \mathrm{NC}(\mathcal{X},\mathcal{Y}) =& \frac{1}{2N_g}\sum_{i=1}^{N_g}\left|\mathbf{n}\left(\mathbf{x}_i\right)\cdot\mathbf{n}\left(\mathcal{P}_{g\to p}\left(\mathbf{x}_i\right)\right)\right| + \\ & \frac{1}{2N_p}\sum_{i=1}^{N_p}\left|\mathbf{n}\left(\mathbf{y}_i\right)\cdot\mathbf{n}\left(\mathcal{P}_{p\to g}\left(\mathbf{y}_i\right)\right)\right|, \end{aligned} \tag{15}$$

which measures the alignment of surface normals. Higher NC values indicate more faithful geometric preservation.

The F-score (F-S.) is defined as the harmonic mean of precision and recall. Precision is the percentage of points on M_p that lie within a distance τ of M_g, and Recall is the percentage of points on

M_g that lie within a distance τ of M_p:

$$\mathrm{FS}(\mathcal{X}, \mathcal{Y}) = \frac{2 \cdot \mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}. \quad (16)$$

Higher F-S. values indicate better correspondence between predicted and ground-truth surfaces.

The quantitative comparisons on the FAMOUS, ABC, and ShapeNet datasets are reported in Table 1, while visual results are shown in Fig. 2. From our experiments, our method consistently achieves competitive performance across all metrics. Moreover, qualitative results show that our method preserves better fine geometric structures and sharp features.

4.3 Ablations

We conduct ablation studies on the FAMOUS dataset to comprehensively evaluate the effectiveness of our proposed dynamic graph filter by analyzing the impact of each individual loss term on surface reconstruction quality. The evaluation is performed using four metrics: CD_{L1}, CD_{L2}, F-score (F-S.), and normal consistency (NC), with detailed results summarized in Table 2. Notably, the loss terms L_3 and L_4, which are directly associated with our dynamic graph filter, are critical for preserving sharp features and capturing fine geometric details. Removing either of these terms leads to a clear decline in performance across all metrics, indicating their essential role in enhancing local geometric fidelity. In contrast, using the full loss formulation that includes L_1 through L_4 consistently yields the best reconstruction results, demonstrating the importance of the dynamic graph filter in guiding the learning of accurate and detail-preserving neural implicit functions.

Table 2 Ablations on different loss items.

L_1	L_2	L_3	L_4	CD_{L1}	CD_{L2}	F-S.	NC
	✓	✓	✓	0.009	0.010	0.783	0.961
✓		✓	✓	0.011	0.075	0.699	0.955
✓	✓			0.012	0.089	0.752	0.959
✓	✓		✓	0.009	0.024	0.755	0.963
✓	✓	✓		0.008	0.013	0.767	0.968
✓	✓	✓	✓	0.006	0.007	0.801	0.979

4.4 Robustness Tests

We further evaluate the robustness of our method using the FAMOUS dataset by assessing its reconstruction quality under different levels of Gaussian white noise. Specifically, we add Gaussian noise to the input point clouds, where the noise level is defined as the standard deviation of the Gaussian distribution normalized by the maximal diagonal of the point cloud's bounding box. To simulate varying degrees of noise, we incrementally increase the noise level from 0.01 to 0.05. The quantitative results under these noise conditions are reported in Table 3, demonstrating the stability and resilience of our method in handling noisy inputs.

Table 3 Robustness tests.

noise-level	CD_{L1}	CD_{L2}	F-S.	NC
no-noise	0.006	0.007	0.801	0.979
0.01	0.007	0.008	0.795	0.967
0.02	0.008	0.010	0.782	0.948
0.03	0.009	0.011	0.757	0.929
0.04	0.010	0.012	0.746	0.904
0.05	0.013	0.015	0.732	0.885

5 Conclusion

This paper presents a novel method for learning neural implicit functions from raw 3D point clouds by introducing a dynamic graph filter. Unlike previous approaches that depend on global optimization or manually tuned, hyperparameter-sensitive operators, our method leverages a dynamic graph neural network to adaptively adjust a bilateral filter, allowing the implicit function to better capture fine local geometric details. Extensive experiments on multiple benchmark datasets demonstrate that our approach achieves competitive performance and superior detail preservation compared to existing baseline methods.

Statements and Declarations

Funding The authors declare that no funding was received for this work.

Conflict of interest The authors declare no conflict of interest.

Ethics approval Not applicable.

Consent for publication Not applicable.

Data availability The datasets used in this study (FAMOUS, ABC, ShapeNet) are publicly available.

Materials availability Not applicable.

Code availability Code will be made available upon acceptance.

Author contribution Fulin Tang and An Liu conceived and designed the study. Fulin Tang implemented the method and conducted the experiments. Rudong Xue and Bingxi Liu assisted with data preparation and evaluation. All authors read and approved the final manuscript.

References

[1] Fu, R., Wen, C., Li, Q.: Imfit: Normal estimation via learning neural implicit surface. In: ICASSP 2024-2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 2605–2609 (2024). IEEE

[2] Fu, R., Hormann, K., Alliez, P.: Lfs-aware surface reconstruction from unoriented 3d point clouds. IEEE Transactions on Multimedia (2024)

[3] Fu, R., Zheng, J., Yu, L.: Consistent normal orientation for 3d point clouds via least squares on delaunay graph. In: Proceedings of the Computer Vision and Pattern Recognition Conference, pp. 16932–16942 (2025)

[4] Lorensen, W.E., Cline, H.E.: Marching cubes: A high resolution 3d surface construction algorithm. In: Seminal Graphics: Pioneering Efforts that Shaped the Field, pp. 347–353 (1998)

[5] 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. Proceedings of Machine Learning Research, vol. 139, pp. 7246–7257 (2021)

[6] 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. Proceedings of Machine Learning Research, vol. 119, pp. 3789–3799 (2020)

[7] 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, pp. 2565–2574 (2020)

[8] Atzmon, M., Lipman, Y.: SALD: sign agnostic learning with derivatives. In: 9th International Conference on Learning Representations, ICLR 2021 (2021)

[9] Chibane, J., Mir, A., Pons-Moll, G.: Neural unsigned distance fields for implicit function learning. In: Advances in Neural Information Processing Systems (NeurIPS) (2020)

[10] Huang, H., Wu, S., Gong, M., Cohen-Or, D., Ascher, U., Zhang, H.: Edge-aware point set resampling. ACM transactions on graphics (TOG) **32**(1), 1–12 (2013)

[11] Li, S., Gao, G., Liu, Y., Gu, M., Liu, Y.-S.: Implicit filtering for learning neural signed distance functions from 3d point clouds. In: Computer Vision – ECCV 2024. Lecture Notes in Computer Science, vol. 15064, pp. 234–251 (2024). https://doi.org/10.1007/978-3-031-72658-3_14

[12] Simonovsky, M., Komodakis, N.: Dynamic edge-conditioned filters in convolutional neural networks on graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3693–3702 (2017)

[13] Ju, W., Yi, S., Wang, Y., Xiao, Z., Mao, Z., Li, H., Gu, Y., Qin, Y., Yin, N., Wang, S., et al.: A survey of graph neural networks in real world: Imbalance, noise, privacy and ood challenges. arXiv preprint arXiv:2403.04468 (2024)

[14] Erler, P., Guerrero, P., Ohrhallinger, S., Mitra, N.J., Wimmer, M.: Points2Surf: Learning implicit surfaces from point clouds.

In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.-M. (eds.) Computer Vision – ECCV 2020, pp. 108–124. Springer, Cham (2020)

[15] Ma, B., Liu, Y.-S., Zwicker, M., Han, Z.: Surface reconstruction from point clouds by learning predictive context priors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6326–6337 (2022)

[16] Sitzmann, V., Martel, J.P., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. Neural Information Processing Systems,Neural Information Processing Systems (2020)

[17] Ben-Shabat, Y., Hewa Koneputugodage, C., Gould, S.: Digs: Divergence guided shape implicit neural representation for unoriented point clouds. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 19323–19332 (2022)

[18] 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, pp. 4460–4470 (2019)

[19] Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: Computer Vision–ECCV 2020, pp. 523–540 (2020). Springer

[20] Boulch, A., Marlet, R.: Poco: Point convolution for surface reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6302–6314 (2022)

[21] Chen, C., Liu, Y.-S., Han, Z.: Gridpull: Towards scalability in learning implicit representations from 3d point clouds. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV) (2023)

[22] Peng, S., Jiang, C., Liao, Y., Niemeyer, M., Pollefeys, M., Geiger, A.: Shape as points: A differentiable poisson solver. Advances in Neural Information Processing Systems **34**, 13032–13044 (2021)

[23] Koneputugodage, C.H., Ben-Shabat, Y., Campbell, D., Gould, S.: Small steps and level sets: Fitting neural surface models with point guidance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 21456–21465 (2024)

[24] Ma, B., Liu, Y.-S., Han, Z.: Reconstructing surfaces for sparse point clouds with on-surface priors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6315–6325 (2022)

[25] Marschner, Z., Sellán, S., Liu, H.-T.D., Jacobson, A.: Constructive solid geometry on neural signed distance fields. In: SIGGRAPH Asia 2023 Conference Papers, pp. 1–12 (2023)

[26] Liao, B., Xiao, C., Jin, L., Fu, H.: Efficient feature-preserving local projection operator for geometry reconstruction. Computer-Aided Design **45**(5), 861–874 (2013)

[27] Lipman, Y., Cohen-Or, D., Levin, D., Tal-Ezer, H.: Parameterization-free projection for geometry reconstruction. ACM Transactions on Graphics (TOG) **26**(3), 22 (2007)

[28] Huang, H., Li, D., Zhang, H., Ascher, U., Cohen-Or, D.: Consolidation of unorganized point clouds for surface reconstruction. ACM transactions on graphics (TOG) **28**(5), 1–7 (2009)

[29] Preiner, R., Mattausch, O., Arikan, M., Pajarola, R., Wimmer, M.: Continuous projection for fast l1 reconstruction. ACM Trans. Graph. **33**(4), 47–1 (2014)

[30] Yu, L., Li, X., Fu, C.-W., Cohen-Or, D., Heng, P.-A.: Ec-net: an edge-aware point set consolidation network. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 386–402 (2018)

[31] Zhang, D., Lu, X., Qin, H., He, Y.: Pointfilter: Point cloud filtering via encoder-decoder modeling. IEEE Transactions on Visualization and Computer Graphics **27**(3), 2015–2027 (2020)

[32] Fleishman, S., Cohen-Or, D., Silva, C.T.: Robust moving least-squares fitting with sharp features. ACM transactions on graphics (TOG) **24**(3), 544–552 (2005)

[33] Wang, X., Cheng, Y., Wang, L., Lu, J., Xu, K., Xiao, G.: Edge preserving implicit surface representation of point clouds. arXiv preprint arXiv:2301.04860 (2023)

[34] Lindell, D.B., Van Veen, D., Park, J.J., Wetzstein, G.: Bacon: Band-limited coordinate networks for multiscale scene representation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16252–16262 (2022)

[35] Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. Advances in Neural Information Processing Systems **33**, 7537–7547 (2020)

[36] Wang, Y., Rahmann, L., Sorkine-Hornung, O.: Geometry-consistent neural shape representation with implicit displacement fields. In: The Tenth International Conference on Learning Representations (2022). OpenReview

[37] Yin, N., Wang, M., Chen, Z., De Masi, G., Xiong, H., Gu, B.: Dynamic spiking graph neural networks. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, pp. 16495–16503 (2024)

[38] Qi, X., Liao, R., Jia, J., Fidler, S., Urtasun, R.: 3d graph neural networks for rgbd semantic segmentation. In: 2017 IEEE International Conference on Computer Vision (ICCV), pp. 5209–5218 (2017). https://doi.org/10.1109/ICCV.2017.556

[39] Landrieu, L., Simonovsky, M.: Large-scale point cloud semantic segmentation with superpoint graphs. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[40] Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (tog) **38**(5), 1–12 (2019)

[41] Koch, S., Matveev, A., Jiang, Z., Williams, F., Artemov, A., Burnaev, E., Alexa, M., Zorin, D., Panozzo, D.: Abc: A big cad model dataset for geometric deep learning. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)

[42] Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)