FACE: Feature-Preserving CAD Model Surface Reconstruction

Shuxian Cai, Yuanyan Ye
School of Informatics
Xiamen University
caishuxian@stu.xmu.edu.cn

Juan Cao
School of Mathematical Sciences
Xiamen University
juancao@xmu.edu.cn

Zhonggui Chen
School of Informatics
Xiamen University
chenzhonggui@xmu.edu.cn

Abstract

Feature lines play a pivotal role in the reconstruction of CAD models. Currently, there is a lack of a robust explicit reconstruction algorithm capable of achieving sharp feature reconstruction in point clouds with noise and non-uniformity. In this paper, we propose a feature-preserving CAD model surface reconstruction algorithm, named FACE. The algorithm initiates with preprocessing the point cloud through denoising and resampling steps, resulting in a high-quality point cloud that is devoid of noise and uniformly distributed. Then, it employs discrete optimal transport to detect feature regions and subsequently generates dense points along potential feature lines to enhance features. Finally, the advancing-front surface reconstruction method, based on normal vector directions, is applied to reconstruct the enhanced point cloud. Extensive experiments demonstrate that, for contaminated point clouds, this algorithm excels not only in reconstructing straight edges and corner points but also in handling curved edges and surfaces, surpassing existing methods.

Keywords: surface reconstruction, feature lines, point cloud enhancement, CAD models

1. Introduction

The technique of CAD surface reconstruction finds widespread application in various industries, including mechanical, electronic, architectural, and aerospace. Its primary objective is to transform point clouds obtained from CAD models into triangular meshes. These models typically exhibit distinctive geometric features, such as corners, edges, and sharp details, which require careful preservation during reconstruction.

However, several factors, such as instrument precision, lighting variations, occlusions, and human errors, introduce noise, outliers, data missing, and uneven point cloud distribution into the acquired data. Consequently, achieving high-quality results while preserving these sharp features proves challenging when attempting to perform surface reconstruction directly on the raw point cloud data.

In prior research, efforts to faithfully replicate model geometry have primarily revolved around two core aspects: point cloud processing and reconstruction. The key to achieving reconstruction results that preserve geometric features lies in having high-quality input point clouds. Raw input point clouds tend to be sparse, typically containing only a few points situated along the edges of geometric features. Some existing techniques [18, 39, 29, 28, 23] attempt to enhance these features by increasing point density in edge regions. However, they often fall short of clearly defining feature lines due to imprecise or insufficient point additions.

In the reconstruction phase, the effective utilization of feature lines proves paramount. Traditional reconstruction methods fall into two categories: implicit and explicit. Implicit methods [11, 4, 3] represent surfaces using implicit functions and then construct results using isosurfaces. While robust against noise, they struggle to preserve sharp features. Explicit methods [37, 40], conversely, explicitly connect scattered input points following specific rules. Although they can recover geometric structures, they are sensitive to noise and require a minimum point density to reconstruct sharp features along feature edges.

Hence, the prevailing approach for feature-preserving reconstruction involves using processed point clouds. While some improved methods [36, 14] prioritize connecting points within feature regions, they still encounter challenges when reconstructing intricate structural areas, leading to certain deficiencies in the final outcomes.

In this paper, we propose a novel approach for **fea**ture-preserving **C**AD model surface **r**econstruction, named FACE, which consists of two parts: point cloud consolidation and surface reconstruction. Initially, we preprocess the point cloud to obtain a clean and uniformly resampled version, effectively removing noise. Subsequently, we employ discrete optimal transport to identify feature regions within the point cloud. We then update point normals and positions within these regions using information from neighboring points, generating new points to increase point density along feature lines. Finally, we apply an advancing-front surface reconstruction method, incorporating normal direction conditions, to reconstruct the enhanced point cloud, re-

sulting in a mesh that faithfully preserves sharp features.

In summary, our contributions are as follows:

1. We propose a novel feature-preserving framework that enhances and enriches input point clouds, enabling the reconstruction of model surfaces with distinct geometric features.

2. We devise preprocessing operations for point cloud denoising and resampling, which improve point cloud uniformity, providing high-quality, noise-free point clouds for subsequent feature enhancement and preventing uneven reconstruction results.

3. We incorporate normal direction conditions into the advancing-front surface reconstruction method that do not consider normals, enhancing the algorithm's adaptability to various geometric structures in models and significantly boosting overall performance and efficiency.

The rest of this paper is structured as follows. Section 2 provides a review of previous studies in the field. In Section 3, we describe the proposed framework. Implementation details and experimental results are shown in Section 4, then we draw a conclusion in Section 5.

2. Related work

2.1. Point cloud consolidation

The raw point cloud obtained from scanning is frequently disordered, sparse, and irregular. Previous methods relies on parameter-heavy approaches such as local mappings [1] and network-based solution [23]. In pursuit of parameter-free solutions, the locally optimal projection operator (LOP) [26] and the improved weighted LOP (WLOP) [17] are developed to accommodate non-uniform data. Additionally, low-rank matrix approximation algorithms [28] and objective functions grounded in winding number requirements [35] are employed to estimate robust normals and enhance point cloud quality. However, these methods struggle with both noise and non-uniform point clouds. In our approach, we address this issue during point cloud preprocessing by selecting denoising levels based on noise intensity and creating uniform point clouds through resampling.

Feature lines within point clouds provide vital geometric information for various applications. Prior research employs multi-scale clustering operators by Pauly *et al*. [30] and refinement techniques like robust moving least squares and Newton's method [7]. Some methods enhance edge density but are sensitive to normal accuracy [18]. Deep learning approaches, such as EC-Net [39], Self-sample [29], and PCED-Net [16], have been introduced to analyze shapes and detect edges. PIE-NET [34] and PC2WF [27] focus on edge detection with region proposal and feedforward modules, respectively. Other methods like MFLE [2] use anisotropic contraction and dual-branch structures for feature refinement. Similarly, SED-Net employs a dual-branch structure to fuse extracted edge and corner features, creating straightforward and distinct feature lines [24]. However, existing methods face challenges with sparse and scattered points along feature lines. We adopt a discrete optimal transport approach to accurately identify feature regions and generate dense points along edges, enhancing subsequent reconstruction.

2.2. Feature preserving surface reconstruction

Reconstructing feature-preserving meshes from complex point clouds is a well-explored challenge. Researchers extended moving least squares (MLS) with robust statistics [13] and non-linear kernels [41] to model sharp surfaces. Salman *et al*. used Voronoi cell covariance matrices and modified Delaunay refinement for sharp edge preservation [33]. Fast locally optimal projection (LOP) operators via kernel density estimation (KDE) also maintain features effectively [25]. Dey *et al*. combined Gaussian-weighted graph Laplacians with weighted Delaunay triangulations for feature safeguarding [9].

Network-based methods employ strategies like deformation [15], local/global priors [11], and 2D Delaunay triangulation properties [31] for mesh construction. Neural marching cubes (NMC) [4] and neural dual contouring (NDC) [3] enhance traditional techniques, with NDC excelling in sharp feature preservation. ComplexGen uses neural networks for probabilistic structure prediction [14]. It optimizes B-Rep chain complexes under structural validity constraints. SECAD-Net generates compact CAD models but suits specific structures [22]. A recent implicit neural network employs a two-stage training process with tailored loss functions to learn input point sharp features and surface details effectively [12]. For symmetrical models, global alignment can fit the input point cloud while preserving constraint relationships, but its application is somewhat limited [38].

Compared to prediction-based methods, RFEPS is a recent algorithm that optimizes points and normals initially [36]. It then detects feature regions, enhances feature lines, and interpolates polygonal surfaces using restricted power diagrams. In addition, feature-preserving reconstruction also employs optimal transport theory [10, 37] and quadratic error metrics [40]. However, these methods have various strengths and weaknesses, including robustness issues, limited suitability for small point clouds, and difficulties preserving sharp features in complex areas. In contrast, our approach leverages enhanced feature points, enabling the creation of sharp features in intricate regions, with superior efficiency due to straightforward connectivity rules.

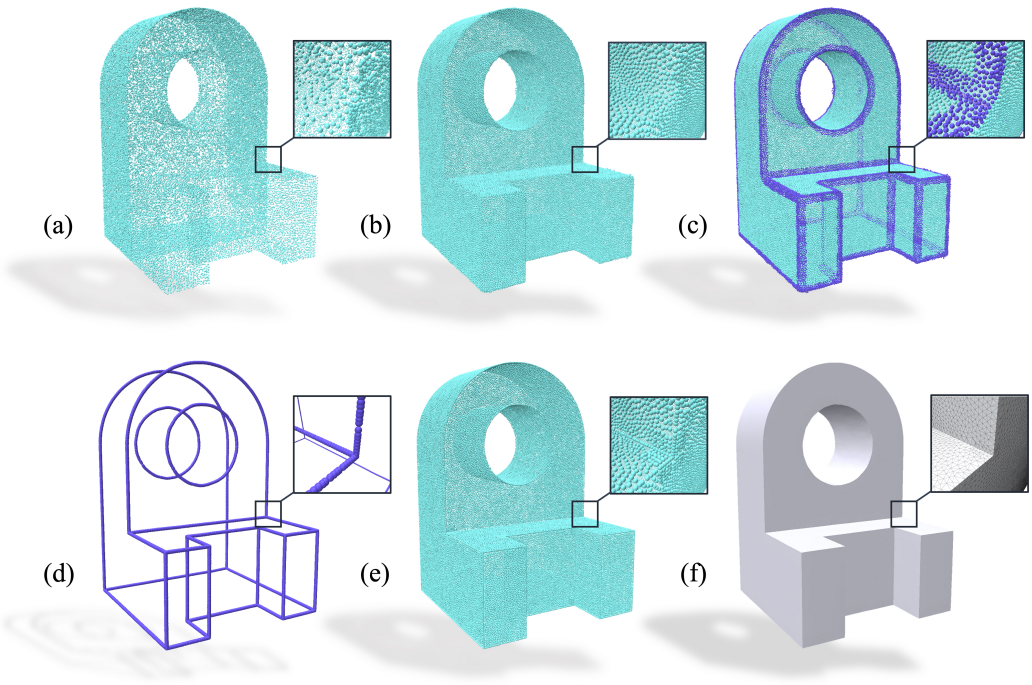


Figure 1. Pipeline of the proposed method. (a) Non-uniform input point cloud with noise; (b) preprocessed point cloud with noise-free and uniformity; (c) detected feature region (in purple points); (d) extracted feature lines; (e) enhanced point cloud with distinct feature lines; (f) reconstructed result.

3. Method

The utilization of feature lines and feature points in CAD models is essential to achieving the reconstruction of sharp features. Given a point cloud with noise and uneven distribution, preprocessing yields a point cloud with noiseless, uniform distribution and dense points on feature lines. Then, a triangular mesh with sharp features is created by employing an explicit reconstruction method. The overall algorithm is depicted in Figure 1, and the following is a brief description of the process.

1. Initialize the normal vectors, denoise the point cloud, and remove outliers. Then, resample the point cloud using the centroidal Voronoi tessellation method. Figure 1(b) displays the preprocessed point cloud.

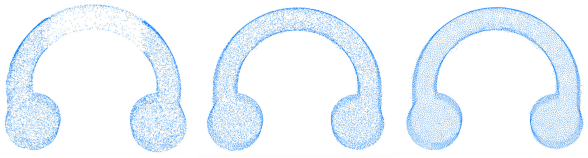
2. Detect point cloud features using optimal transport theory, as shown in Figure 1(c) (in purple points). After that, update point normals and positions based on neighborhood information, and generate new points for feature lines, which are presented in Figure 1(d).

3. The optimized point cloud after the above steps is shown in Figure 1(e). Reconstruct the point cloud using the advancing-front method with the normal condition, and the result is provided in Figure 1(f).

3.1. Point cloud preprocessing

To mitigate the impact of low-quality data on reconstruction results, it is crucial to conduct denoising, outlier removal, and resampling operations during point cloud preprocessing. We utilize the method described in [36] to denoise the data, which enables the selection of an optimal denoising level based on noise intensity.

In addition to noise, outliers in the input point cloud also can introduce additional structure into the results. To remove these outliers, we define points with fewer than five neighbors as outliers. The neighborhood is often defined using a spherical region with a radius of r, adjusted for point density. Uniform point clouds utilize a consistent neighborhood definition, but non-uniform cases require distinct definitions for sparse and dense regions. To prevent inadvertent removal of points in sparse regions, we employ an adaptive method to identify neighborhood points. Initially, we use a kNN tree to search for the 50 nearest points to p_i. For the 50 points found, calculate the average minimum distance d_i by computing the distance between each point and its nearest neighbor. Next, use a sphere with a radius $r_i = 2d_i$ to search for neighbors of p_i. After calculating the number of neighbors, check whether the number of neighbors at any point reaches a threshold, typically set between 80 and 120. If yes, stop the search. Otherwise, increase the



(a) (b) (c)

Figure 2. Procedure of resampling. (a) Initial point cloud; (b) initial resampling points; (c) final resampling points.

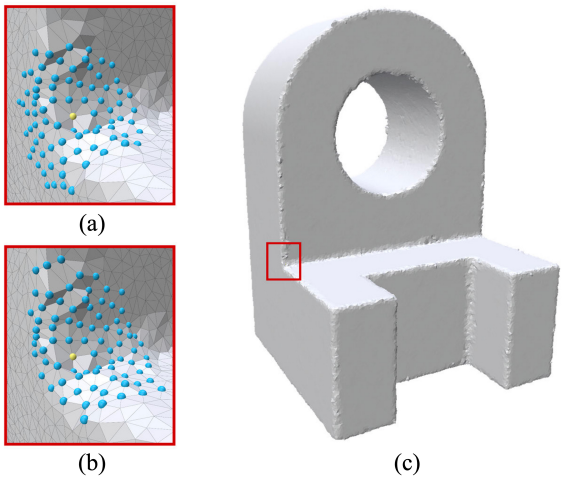
search radius by $0.5d_i$ and initiate a new round of search until the point count in any point's neighborhood reaches the threshold. After stopping the search, classify points with fewer than 5 neighbors as outliers based on the number of neighbors per point.

Furthermore, the density of the point cloud has an impact on the reconstruction of sharp features. To address non-uniform point clouds, this algorithm employs a method proposed by Chen *et al*. to resample the denoised point cloud, thereby improving the uniformity of point distribution [5]. Initially, this method generates sampling points on the restricted Voronoi diagram of the point cloud. Subsequently, it applies Lloyd's algorithm to calculate the centroid Voronoi diagram and relocates point p to the centroid of its Voronoi cell. Repeated iterations of this process result in resampling, typically requiring fewer than 5 iterations to obtain high-quality points. The results of each step are displayed in Figure 2. We only utilized uniform sampling to enhance the quality of the point cloud. For more complex structures, adaptive sampling can also be employed to ensure a more rational distribution of points.

3.2. Feature enhancement

Clear and continuous feature lines are essential for the successful reconstruction of features in CAD models. Despite the uniform and noise-free distribution of the preprocessed point cloud, its feature lines may not be readily apparent, and point arrangements can appear somewhat scattered. To enhance the point cloud's features, we draw inspiration from [36]. We detect the feature regions by using discrete optimal transport followed the idea from [36], and optimize and densify the points within these areas, ultimately achieving well-defined feature lines.

In Figure 1(c), the identified feature points are displayed, closely matching the actual feature edges. To enhance these features, we initially used the method shown in [36] to update point normals and positions, generating dense points along feature lines. However, during the update process, this method exhibited issues of incorrect point displacement in some complex regions. Upon further investigation, we identified the root cause as the improper selection of spherical neighborhoods.



(a) (b) (c)

Figure 3. Comparison of different neighborhood selections. (a) Spherical neighborhood points; (b) neighborhood points obtained by restricted Delaunay triangulation; (c) restricted Delaunay triangulation of resampling points. The blue point is the neighborhood point of the yellow point.

Spherical neighborhoods have limitations as they can mistakenly group points based solely on spatial proximity, disregarding topology. For example, in the corner region outlined in Figure 3(c), defining neighborhoods as shown in Figure 3(a) can lead to an erroneous update of yellow points' normal vectors. Since there are more neighborhood points on the side, more mass is transferred to the side normal vector. This can cause the normal vector of the yellow point to be incorrectly updated in the outward direction, affecting the update of the position of subsequent points. Thus, a more thoughtful approach to determining point neighborhoods is necessary.

The restricted Delaunay triangulation (RDT) obtained from resampling provides topological connectivity. To ensure precise normal vector updates, point neighborhoods are defined as follows: Initially, for a point p_i, its neighborhood $Neigh(p_i)$ includes 1-ring neighbors in the RDT. Then, for each $p_j \in Neigh(p_i)$, any point within p_j's 1-ring neighborhood, with a normal vector angle to p_i less than $\pi/3$, is added to $Neigh(p_i)$. This iterative process expands the neighborhood until the count of points reaches a threshold, typically between 80 and 120. Figure 3(b) illustrates these neighborhoods for yellow points, preventing incorrect normal vector updates.

After updating normal vectors, we optimize points similarly to the denoising part in Section 3.1. Since the number of points along these edges is limited, making it possible to project points within the feature region onto the plane intersections to generate new feature points. For points $p \in \mathcal{F}$ in the feature region, each neighboring point $p_j \in Neigh(p)$ generates a tangent plane passing through point p_j and per-

pendicular to the normal vector $\mathbf{n_j}$. These tangent planes intersect at the CAD model's feature edges. Optimizing the distance from point p to the tangent planes of its neighboring points creates new points at the intersections of these tangent planes.

To prevent excessive point density around feature points hindering sharp feature reconstruction, we remove non-feature points within $2l$ distance, with l as the average point spacing. In the feature-rich region, where points are densely distributed, elongated triangular facets can occur after the removal operation. To improve mesh quality, we downsample feature points using a spatial threshold. In the downsampled point cloud, the minimum distance between two points must meet or exceed the threshold, typically set at 0.0025.

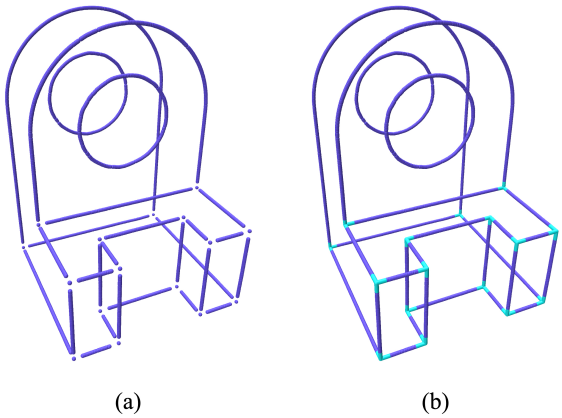


Figure 4. Comparison of results (a) before and (b) after the post-processing step for point generation between corners and neighbors.

The method mentioned above for generating feature points still has a problem: all feature points around corners project onto the corners, creating a shortage of points along the adjacent feature lines, as seen in Figure 4(a). To solve this, we introduce a post-processing step. It identifies feature points at corners and generates new ones between these corners and their neighbors.

Suppose the newly generated points are denoted as $\mathcal{F}'$. For each q in $\mathcal{F}'$, we search for the nearest m points within $\mathcal{F}'$, typically setting m between 20 and 25. If three points q_1, q_2, and q_3 are found among these m points, forming any two vectors qq_i and qq_j $(i, j \in \{1, 2, 3\}$ and $i \neq j)$ with angles between them ranging between 80° and 110°, it indicates a corner point. New feature points are sampled along the line segment between q and q_i, generating a new point every $5w$, where w is the average spacing of the resampled point cloud, as depicted by the blue lines in Figure 4(b).

Additionally, dense points are added on planes around inflection points to ensure enough points for sharp feature reconstruction. For the three nearest points to the corner point q identified earlier, any two points q_i and q_j among them exist in the same plane as the corner point q. Similar to Section 3.1, the average spacing z of the existing point cloud is calculated. Within an area of $\|qq_i\| \times \|qq_j\|$, starting from q, we alternate along the directions of qq_i and qq_j, adding a point every kz distance (where k is a coefficient controlling the density of added points, typically set to 10). This ensures an even and dense distribution of points within the specified range, thereby preserving the reconstruction of sharp features.

3.3. Surface reconstruction

In the last phase of this framework, we reconstruct the model surface with well-defined feature lines from the processed point cloud data. Explicit methods, such as advancing-front surface reconstruction (AFSR) [8], are highly effective for this purpose.

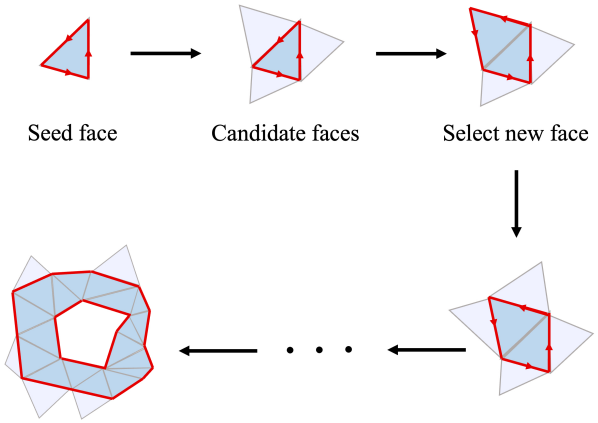


Figure 5. Main process of AFSR. Initiate with the seed face and incorporate selected faces from candidate faces along the reconstructed boundary until the process is complete.

Figure 5 demonstrates the primary process of AFSR. The approach initially computes the Delaunay tetrahedron of the input point cloud, obtaining a triangular faces set $\mathcal{D} = \{T_1, T_2, \ldots, T_n\}$. For every triangle $T_i = \{p_{i1}, p_{i2}, p_{i3}\}$ within $\mathcal{D}$, the method calculates its radius r_i, where r_i is the radius of the smallest sphere that passes through the vertices of T_i and enclosing no other sampling points. The method then chooses the triangular face with the smallest radius from $\mathcal{D}$ as the initial seed face.

Since the traditional AFSR method favors smaller triangles, this may lead to errors in subsequent reconstruction results, especially in models like those shown in Figure 6, where closely spaced points create connectivity challenges between double-layer thin surfaces. In order to tackle this issue, we introduce a normal condition during the selection of triangles. This condition necessitates prior knowledge of vertex normals and involves calculating the initial normal of the triangle based on vertex coordinates. When choosing a

triangle, if its initial normal or its reverse normal orients on the same side as the normals of the three vertices of the triangle, then the final normal of the triangle is the one that is oriented on the same side as the vertex normal. Otherwise, we do not calculate the normal of the triangle and discard it. By integrating information on radius and normal, we can accurately select the correct seed triangle.

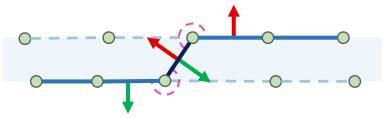


Figure 6. Illustration of double-layer connection. The double-layer thin surface is mistakenly connected as a single-layer thin surface.

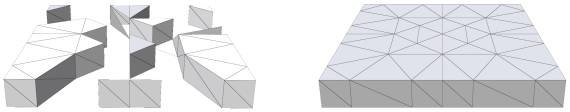
After selecting the seed triangle, it is added to the reconstruction result $\mathcal{S}$, and the three edges constituting the seed triangle are included in the boundary set $\partial\mathcal{S} = \{e_1, e_2, \ldots, e_n\}$. For each edge e_i on the boundary, if the normal of the selected triangle to which e_i belongs forms an angle smaller than a threshold (usually set to $5\pi/6$) with the normals of the unselected triangles adjacent to e_i in $\mathcal{D}$, then from these unselected triangles satisfying the angle condition, the triangle with the smallest radius is chosen as the most suitable candidate triangle for that edge. The candidate triangles chosen for each edge form the set of candidate triangles.

Afterward, from the set of candidate triangles, we carefully select the most reliable triangle as the newly added triangle and integrate it into the reconstruction result $\mathcal{S}$. When choosing the new triangle, to prevent connectivity errors, we consider not only the plausibility grade condition and extension types outlined in [8] but also the previously mentioned condition where the triangle's normal aligns with the vertex normal on the same side. The boundary set and the set of candidate triangles undergo updates based on $\mathcal{S}$, initiating a new round of selection and addition. This iterative process continues until all candidate triangles have been considered, indicating the conclusion of the reconstruction algorithm.

Figure 7 illustrates the comparison. The original method, ignoring normal vector conditions and favoring smaller triangles, fails to achieve accurate reconstruction. Conversely, the improved method, guided by normal directions, accurately selects triangles, achieving proper reconstruction for thin structures while preserving sharp features.

4. Experiments

Our algorithm was implemented in C++ on a computer with a 3.2GHz ARM-based 8-core CPU, 16GB RAM, and an NVIDIA RTX 3090Ti graphics card for some deep learning experiments. Experimental data were obtained from



(a) (b)

Figure 7. Comparison of reconstruction results (a) before and (b) after the improvement. The traditional AFSR [8], lacking consideration of normals, tends to favor smaller triangles, resulting in unnecessary connections. In contrast, the improved AFSR, guided by normals, achieve accurate reconstruction results.

the ABC dataset, which includes diverse CAD models. We used noise and non-uniformly sampled point clouds generated by Huang *et al*. [19] from the ABC dataset, normalizing all data to the [0, 1] range.

To evaluate the precision of consolidated point clouds, we randomly selected 50 models from the ABC dataset. We utilized the one-sided Chamfer Distance (OCD) and one-sided Edge Chamfer Distance (OECD) [36] to measure the proximity of the consolidated point cloud to the ground-truth surface. Regarding the accuracy of reconstructed meshes, three metrics are utilized, including Chamfer Distance (CD) [11], F-score (F1) [21], and Normal Consistency (NC) [32].

4.1. Improved advancing-front reconstruction

The traditional advancing-front surface reconstruction method did not take into account surface normals. In order to extend the applicability of this approach to a broader range of models, the enhanced version of this method incorporates normal considerations. Figure 8 provides a visual comparison between the original and enhanced methods. Since the purpose is to verify the validity of the reconstruction, the input point cloud is guaranteed to be uniform and noise-free. As the complexity of model reconstruction increases, it becomes evident that the improved method not only preserves the capabilities of the original one but also effectively addresses more complicated scenarios. In the case of models featuring thin-sheet structures, the original method would erroneously connect certain points due to the lack of normal vector information, leading to inaccuracies in the resulting faces along edges and gaps. However, with correct normals as a prerequisite, the enhanced version can accurately determine whether new faces can be generated based on the orientation of triangles, thus enabling the successful reconstruction of challenging models.

4.2. Sharp feature-preserving

Figure 9 illustrates the outcomes of point cloud processing and reconstruction achieved by our comprehensive algorithm applied to CAD models. It is obvious that this al-

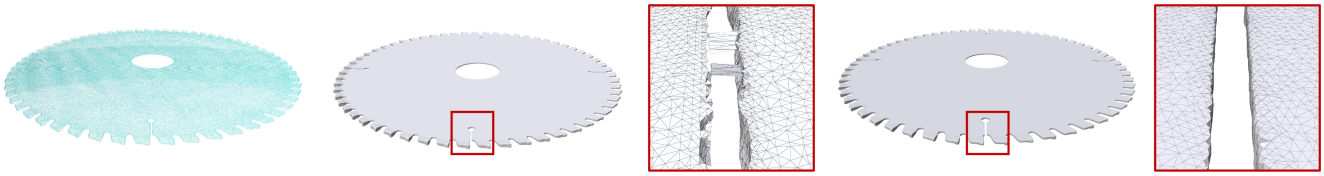


Figure 8. Comparison of reconstruction results on the model with complicated structure.

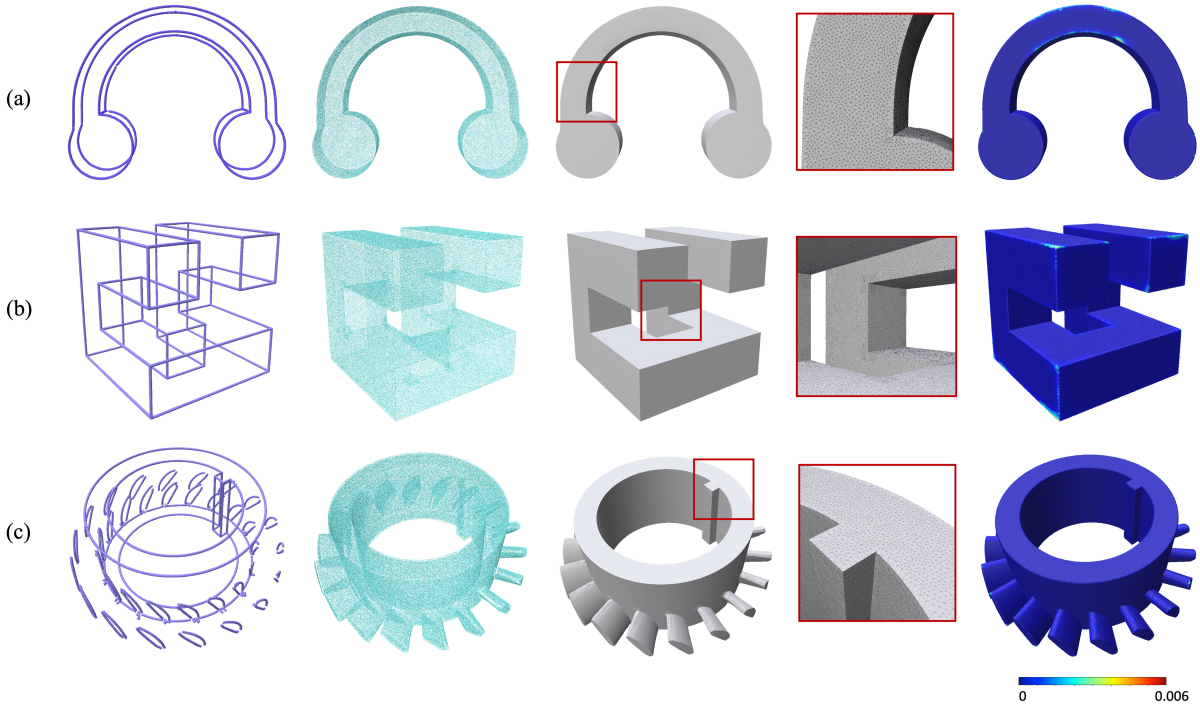


Figure 9. Reconstruction process of our algorithm on CAD models. From left to right: feature points acquired through feature enhancement, point clouds following feature enhancement, reconstruction results, local details, and reconstruction errors.

gorithm possesses the capability to effectively reconstruct sharp geometric features and smooth surfaces, while also accommodating high-genus data. Following multiple processing stages, the point cloud exhibits well-defined feature lines and a uniform point distribution. Moreover, dense feature points are generated in corner regions, bolstering subsequent reconstruction efforts.

During CAD model reconstruction, our algorithm focuses on two main types of sharp features: sharp edges and corner points. Sharp edges result from plane intersections, and our algorithm enhances features in the feature region, generating points along the intersection lines of cutting planes within point neighborhoods to reconstruct them effectively. Corner points, which consist of corners depicted in Figure 9(b) and sharp points shown in Figure 9(c), formed by the intersection of three or more planes, pose a greater challenge. While our optimization can generate corner points, it reduces the number of surrounding feature points. However, our post-processing methods effectively restore corner point features.

To evaluate reconstruction accuracy, we randomly sampled 200,000 points from the true model and measured their distances to the reconstructed mesh. Figure 9's final column displays the error distribution. Despite non-uniform input data with minor noise, the algorithm exhibits minor errors and most of the sharp features can be reconstructed.

4.3. Comparisons

The framework comprises two parts: point cloud consolidation and surface reconstruction. The former aims to achieve a uniform point cloud with clear feature lines, while the latter leverages feature points to preserve sharp features.

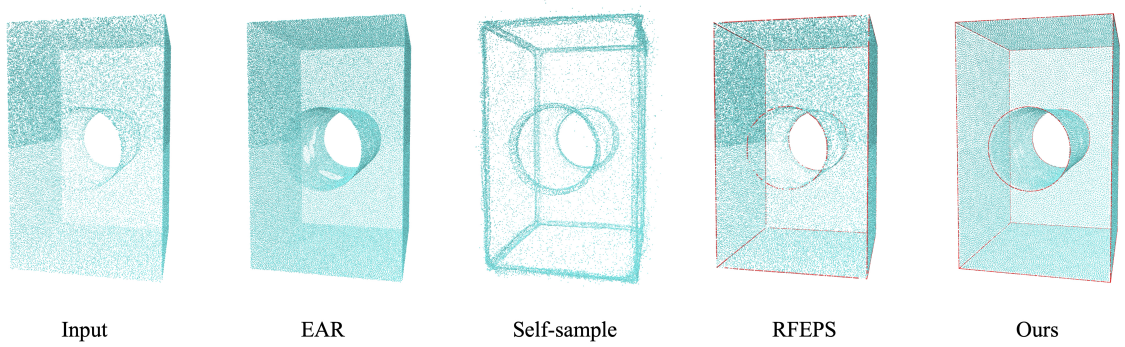


Figure 10. Comparison of results from different point cloud consolidation methods, where points on feature lines are highlighted in red.

4.3.1 Point cloud consolidation quality

The effectiveness of point cloud consolidation is evaluated based on the uniformity of the processed point cloud and the clarity of feature lines, which are crucial for subsequent reconstruction. In experiments with non-uniform point cloud inputs, we compared our method with EAR [18], Self-sample [29], and RFEPS [36], using the authors' provided code. Figure 10 illustrates the results, and the statistics for OCD and OECD can be found in Table 1.

EAR struggles to achieve uniform point density in non-uniform regions and may lead to point cloud loss due to its base selection mechanism. Although increasing the size can address this issue, the main limitation of EAR is its feature point generation scheme. EAR gradually upsamples points on both sides of the edge and projects them onto the potential surface, arranging them uniformly and densely in the vicinity of sharp edges. However, there is no assurance that they can align perfectly with feature lines, leading to challenges in reconstructing sharp edges.

Self-sample, while effective at densifying points in feature areas, can lead to blurry feature lines and lack clarity in corner points and sharp edges. On the other hand, RFEPS struggles with non-uniform scenarios due to the sparsity of the lower part of the point cloud, leading to unclear edges on the lower curved edge of the cylinder. Additionally, RFEPS tends to converge newly generated points at corner points after optimization, leaving no feature points around them.

In contrast, our method incorporates a resampling step that generates a uniform point cloud and precisely identifies feature regions to enhance the sharp features of straight edges, curved edges, and corner points. It excels not only in handling low-density point clouds, ensuring continuous feature lines and the presence of feature points around corner points, but also in generating point clouds closest to the ground-truth.

Table 1. Accuracy assessment of various point cloud consolidation methods.

Methods	OCD ↓	OECD ↓
EAR	0.649	0.151
Self-sample	0.783	0.228
RFEPS	0.554	0.147
Ours	**0.445**	**0.143**

4.3.2 Surface reconstruction quality

The main advantage of this framework lies in its capacity to seamlessly combine point cloud consolidation methods with reconstruction techniques, resulting in model surfaces that retain sharp features. Consequently, the meticulous selection of the optimal combination becomes paramount. Diverse point cloud processing methods may exhibit varying levels of compatibility with distinct reconstruction techniques. Figure 11 demonstrates the outcomes achieved through various method pairings. The input point cloud is non-uniform and noisy. The "w.o." denotes cases where no point cloud consolidation was applied.

The reconstruction methods compared in this study comprise RIMLS [41], screened Poisson surface reconstruction (SPSR) [20], Point2Mesh (P2M) [15], and RFEPS [36]. RIMLS and SPSR are both implicit methods, implemented and yielding results through MeshLab [6]. P2M employs deep learning techniques to iteratively simplify the convex hull towards the target model, while RFEPS is an explicit method. The results for both methods were obtained by executing the code provided by their respective authors.

The insights from Figure 11 are compelling, showing that point cloud consolidation consistently enhances reconstruction quality compared to untreated data. When paired with more suitable methods, it further improves feature

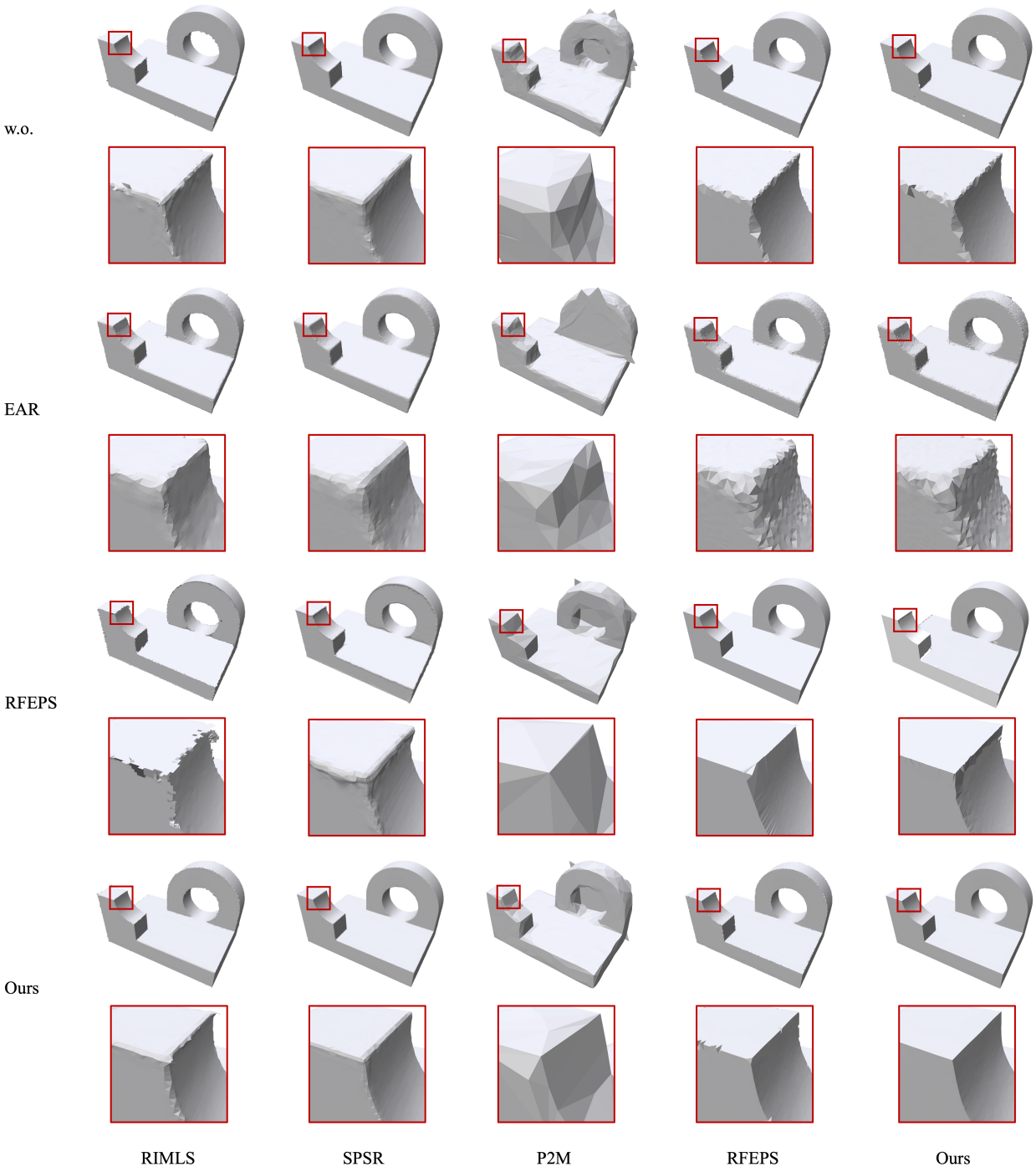


Figure 11. Comparison of results obtained by the combination of various point cloud consolidation methods with different reconstruction methods.

preservation. However, P2M is an exception, better suited for void-free structures and specific point cloud quantities, making it less effective for larger datasets. It successfully reconstructs some corner points but struggles with most straight and curved edges.

When applying RIMLS and SPSR to data processed by EAR and our method, respectively, they exhibit the ability to retain certain feature regions, but their implicit reconstruction nature tends to yield smoother results, making it arduous to fully recover sharp features.

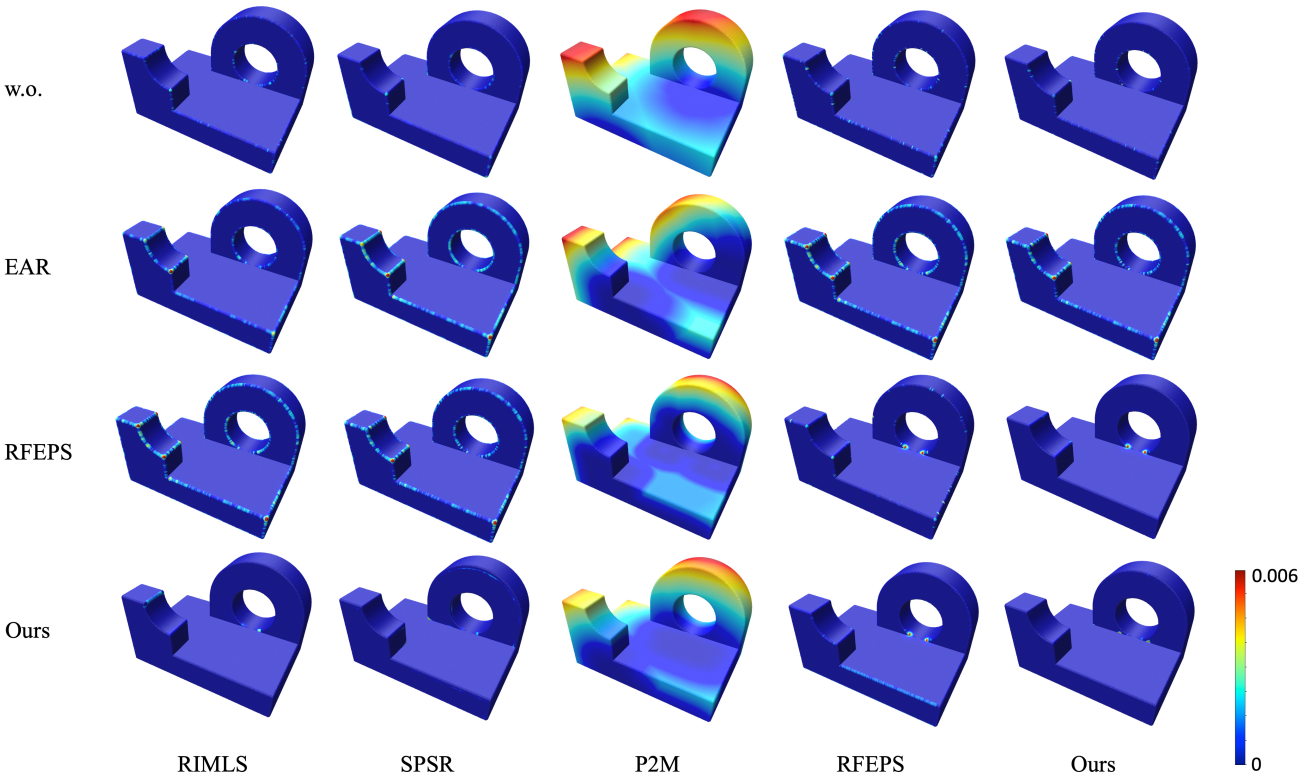


Figure 12. Comparison of error distributions in different reconstruction results.

RFEPS relies heavily on feature line point density, which our method enhances. However, simply increasing a certain of density still does not meet the demands of RFEPS. Excessive density can lead to elongated triangles and surface irregularities. Combining RFEPS with point cloud processing methods designed for it remains challenging due to its spherical neighborhood approach, which can include irrelevant points, causing inaccuracies in updates.

In contrast, our method calculates neighborhoods using the RDT of resampled point clouds, effectively mitigating the issue of erroneous point updates. After processing the point cloud with our algorithm and performing reconstruction, it consistently and accurately restores various features while ensuring a smooth and uniform surface.

For a quantitative comparison, we employed color coding to visualize disparities between ground truth and reconstructed surfaces. Figure 12 presents reconstruction errors. Our approach outperforms others, excelling visually and quantitatively, including global and local evaluations. Although certain results may exhibit potential competitiveness with our method in overall metrics, they frequently fall short when subjected to detailed local scrutiny, encountering difficulties in faithfully reconstructing sharp features.

To further highlight the method's superiority, Table 2 provides more specific and detailed comparative results at the numerical level. It is evident that our proposed combination method outperforms others in terms of the Chamfer Distance and F-score, closely approximating the ground-truth mesh. Regarding normal consistency, the point cloud reconstructed by applying SPSR to the output of RFEPS exhibits a slight advantage over our combination method, but the difference between the two is minimal.

Additionally, we applied uniform denoising to mitigate noise impact, enhancing the clarity of the feature enhancement's effectiveness. In Figure 13, reconstruction results are compared after using the same denoising method. RIMLS and SPSR utilize enhanced point clouds processed through EAR, while RFEPS and our method employ their respective frameworks. The input point cloud processed by EAR has undergone denoising before enhancement, resulting in smooth mesh surfaces reconstructed by RIMLS and SPSR. Despite their improvements, the absence of targeted feature line handling in corner regions hinders the accurate reconstruction of sharp corner features. Due to the use of incorrect neighborhood determination rules, RFEPS exhibits surplus facets in corner regions, preventing it from accurately reconstructing sharp features, unlike our method.

Table 2. Accuracy evaluation of reconstruction results for different combination methods. The point cloud consolidation section includes w.o. (representing no consolidation method), EAR, RFEPS, and our consolidation method. The surface reconstruction section comprises RIMLS, SPSR, P2M, RFEPS (RVD/RPD), and ours (advancing-front).

Methods	CD ↓				NC↑				F1↑			
	w.o.	EAR	RFEPS	Ours	w.o.	EAR	RFEPS	Ours	w.o.	EAR	RFEPS	Ours
RIMLS	0.824	0.475	0.593	0.513	0.847	0.903	0.912	0.915	0.396	0.435	0.498	0.461
SPSR	**0.223**	**0.126**	**0.095**	0.643	**0.988**	**0.986**	**0.992**	0.985	0.405	0.486	0.521	0.480
P2M	1.302	0.649	0.616	0.645	0.457	0.437	0.436	0.448	0.263	0.147	0.239	0.221
RFEPS	0.646	0.647	0.555	0.644	0.905	0.889	0.951	0.955	**0.554**	**0.525**	0.524	0.569
Ours	0.268	**0.126**	**0.095**	**0.065**	0.921	0.911	0.955	**0.991**	0.486	0.376	**0.554**	**0.630**

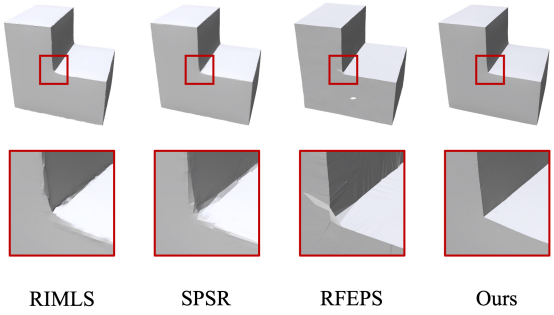


Figure 13. Comparison of reconstruction results after using the same point cloud denoising method.

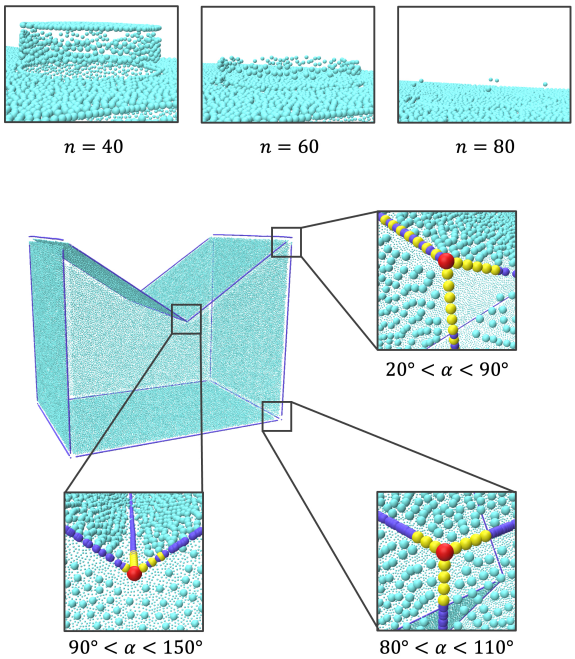


Figure 14. Impact of different parameter settings. Top row: n is used to define the maximum number of nearest neighbors; bottom row: the range of angle α influences corner point recognition. The identified corner points are marked with red, initial feature lines are represented by purple points and yellow points denote points added during post-processing step.

4.4. Robustness analysis

4.4.1 Robustness to parameter setting

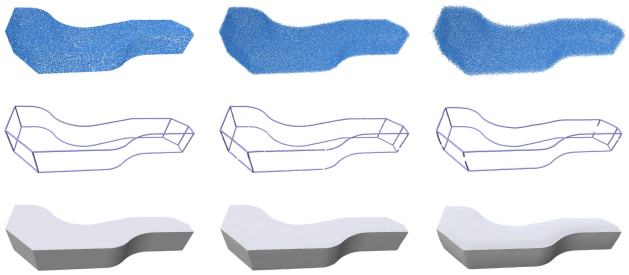
The proposed framework requires configuring multiple parameters, and different values can impact the results. Figure 14 illustrates the effects of two main parameters: the maximum number of nearest neighbors n and the angle α that corner points should have. For models with fine and small structures, setting n too high not only affects outlier removal during preprocessing but also causes some points to be incorrectly moved during subsequent feature processing, leading to the neglect of local structures. Smaller angle ranges are more effective in recognizing acute corner points, while larger ranges are better at identifying obtuse corner points.

Table 3 provides the impact of various parameter values on quantified accuracy. Although the performance is optimal when n is 40, which is suitable for models with various small local structures, a value of 60 for n can meet basic requirements in general scenarios. The selection of the angle range directly influences the effectiveness of corner point identification. For points with smaller angles, it is necessary to decrease the lower limit of the range, and vice versa for points with larger angles. After multiple experiments, we set the default value for the angle range to be 80 to 110 degrees. The influence of different parameter values on the overall results is minor, indicating that our method is relatively robust to parameter variations. However, for specific models, parameter adjustment based on experience may be required to achieve the best results.

Table 3. Impact of parameters n and α on the reconstruction results.

Parameters	CD ↓	NC↑	F1↑
$n = 40$	**0.973**	**0.989**	**0.577**
$n = 60$	0.982	**0.989**	0.573
$n = 80$	0.984	0.988	0.512
$20 < \alpha < 90$	1.008	0.982	0.504
$80 < \alpha < 110$	**0.982**	**0.989**	**0.573**
$90 < \alpha < 150$	0.991	**0.989**	0.527

4.4.2 Robustness to noise degree



(a) σ = 0.0% (b) σ = 0.6% (c) σ = 1.2%

Figure 15. Reconstruction results for different noise data. Top: input point cloud; middle: extracted feature lines; bottom: reconstruction results.

To assess algorithm robustness to noise, experiments with synthetic point clouds containing varying Gaussian noise levels were conducted. The noise data was introduced by applying a random offset, drawn from a Gaussian distribution with a mean of 0 and a standard deviation of σ, to each point within the original point cloud. The point cloud, containing 80,000 points, was standardized within the [0,1] range. Figure 15 shows results for noise-free and noisy data (standard deviations σ of 0.6% and 1.2%). It is evident that the algorithm maintains strong performance even in the presence of noise, preserving sharp features and delivering smooth surface reconstructions.

While this algorithm is robust to moderate noise, excessive noise can lead to imperfect reconstruction of sharp features. Figure 15(c) demonstrates that when the noise standard deviation σ reaches 1.2%, features near dihedral angles close to π may not be fully reconstructed, resulting in the absence of sharp features in that region.

Table 4. Execution times for different point cloud consolidation methods.

Point size	EAR	Self-sample	RFEPS	Ours
20K	**11.24s**	8267.03s	37.61s	30.22s
40K	**13.53s**	23483.36s	39.95s	38.57s
60K	**19.39s**	26674.17s	60.67s	53.50s
80K	**30.26s**	30687.79s	85.63s	71.08s
100K	**80.36s**	44378.21s	249.81s	145.27s
120K	**91.65s**	69251.47s	263.15s	197.41s

4.4.3 Robustness to real-world object

To provide a comprehensive assessment of our algorithm's resilience and applicability, we conducted experiments on real-world scanned point clouds. The reconstruction results using scan data (a) generated by the EinScan-SE 3D scanner, as well as scan data (b) and (c) generated by the Artec 3D scanner, are depicted in the Figure 16. Drawing from previous comparative results, we chose two frameworks that demonstrated superior performance for comparison with our method. Evidently, our method can effectively reconstruct sharp features and retain intricate details even in noisy scanned point clouds.

4.5. Runtime performance

In addition to assessing result accuracy, the program's runtime stands as a crucial metric to consider. Table 4 and Table 5 display execution times for various point cloud consolidation and reconstruction methods across scales from 20K to 120K points, using 50 randomly selected models in Section 4.3 with a maximum neighborhood size of 60 points.

Table 4 shows that while our method excels in enhancing point cloud features, its runtime is inferior to EAR, which performs best due to its simpler process.

Regarding reconstruction time, our approach showcases superior efficiency compared to alternative methods. It maintains a runtime of approximately 5 seconds, even as the point cloud scale increases. Thus, our algorithm outperforms existing method combinations in terms of runtime efficiency.

5. Conclusion

In this paper, we propose a feature-preserving CAD model surface reconstruction algorithm. The procedure of this algorithm is to procure high-quality point clouds by means of point cloud preprocessing and feature enhancement, then generate dense points along feature lines. Finally, we employ an improved advancing-front method to

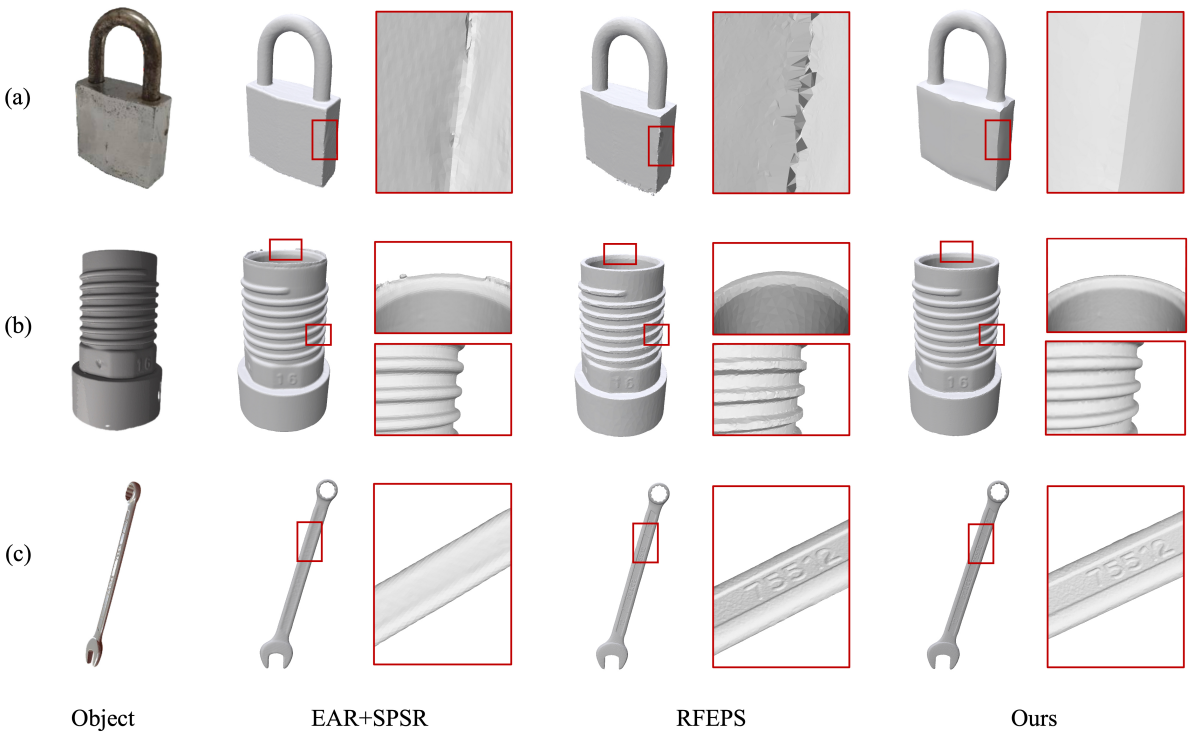


Figure 16. Comparison of reconstruction results from the real-world scanned objects.

Table 5. Execution times for different surface reconstruction methods.

Point size	RIMLS	SPSR	P2M	RFEPS	Ours
20K	3.18s	2.35s	1524.38s	17.80s	**0.21s**
40K	4.42s	3.68s	1790.41s	38.04s	**0.36s**
60K	4.76s	4.28s	1916.32s	41.39s	**0.64s**
80K	4.90s	4.72s	2163.07s	52.48s	**0.83s**
100K	5.23s	5.18s	2745.83s	76.14s	**4.75s**
120K	6.05s	**5.31s**	2963.44s	95.34s	5.67s

construct a triangulated mesh. Experimental results substantiate the algorithm's proficiency in reconstructing various sharp features and its compatibility with point cloud data featured by noise and non-uniform sampling densities.

While our proposed method has demonstrated its effectiveness, it does have some limitations. It can successfully detect feature regions with dihedral angles within the range of $[\pi/6, 5\pi/6]$ but struggles with regions close to π. For regions with dihedral angles smaller than $\pi/6$, feature detection and point generation can lead to erroneous enhancement due to the thinness. Although these issues can be alleviated by iteratively adjusting parameters, the current method's dependence on manual parameter tuning poses inconvenience. Additionally, there is a lack of rules to ensure that the generated points can form smooth curves, especially for various complex curve features. Therefore, future research could delve into exploring adaptive parameter control schemes, whether to address these marginal cases or to fit curves and generate feature points along them.

Acknowledgement

The research was supported by the National Natural Science Foundation of China (Nos. 61972327, 62272402, 62372389), the Natural Science Foundation of Fujian Province (No. 2022J01001), and the Fundamental Research Funds for the Central Universities (No. 20720220037).

References

[1] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. Silva. Point set surfaces. In *Proceedings Visualization, 2001. VIS '01.*, pages 21–29, 537, 2001. 2

[2] H. Chen, Y. Huang, Q. Xie, Y. Liu, Y. Zhang, M. Wei, and J. Wang. Multiscale feature line extraction from raw point clouds based on local surface variation and anisotropic contraction. *IEEE Transactions on Automation Science and Engineering*, 19(2):1003–1016, 2022. 2

[3] Z. Chen, A. Tagliasacchi, T. Funkhouser, and H. Zhang. Neural dual contouring. *ACM Trans. Graph.*, 41(4), jul 2022. 1, 2

[4] Z. Chen and H. Zhang. Neural marching cubes. *ACM Trans. Graph.*, 40(6), dec 2021. 1, 2

[5] Z. Chen, T. Zhang, J. Cao, Y. J. Zhang, and C. Wang. Point cloud resampling using centroidal Voronoi tessellation methods. *Computer-Aided Design*, 102:12–21, 2018. Proceeding of SPM 2018 Symposium. 4

[6] P. Cignoni, M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. MeshLab: An open-source mesh processing tool. In V. Scarano, R. D. Chiara, and U. Erra, editors, *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008. 8

[7] J. I. Daniels, L. K. Ha, T. Ochotta, and C. T. Silva. Robust smooth feature extraction from point clouds. In *IEEE International Conference on Shape Modeling and Applications 2007 (SMI '07)*, pages 123–136, 2007. 2

[8] F. D. David Cohen-Steiner. A greedy Delaunay-based surface reconstruction algorithm. *The Visual Computer*, 20:4–16, 2004. 5, 6

[9] T. K. Dey, X. Ge, Q. Que, I. Safa, L. Wang, and Y. Wang. Feature-preserving reconstruction of singular surfaces. *Computer Graphics Forum*, 31(5):1787–1796, 2012. 2

[10] J. Digne, D. Cohen-Steiner, P. Alliez, F. Goes, and M. Desbrun. Feature-preserving surface reconstruction and simplification from defect-laden point sets. *J. Math. Imaging Vis.*, 48(2):369–382, feb 2014. 2

[11] P. Erler, P. Guerrero, S. Ohrhallinger, N. J. Mitra, and M. Wimmer. Points2Surf learning implicit surfaces from point clouds. In *Computer Vision – ECCV 2020*, pages 108–124. Springer International Publishing, 2020. 1, 2, 6

[12] Y.-F. Feng, L.-Y. Shen, C.-M. Yuan, and X. Li. Deep shape representation with sharp feature preservation. *Computer-Aided Design*, 157:103468, 2023. 2

[13] S. Fleishman, D. Cohen-Or, and C. T. Silva. Robust moving least-squares fitting with sharp features. *ACM Trans. Graph.*, 24(3):544–552, jul 2005. 2

[14] H. Guo, S. Liu, H. Pan, Y. Liu, X. Tong, and B. Guo. ComplexGen: CAD reconstruction by B-Rep chain complex generation. *ACM Trans. Graph.*, 41(4), jul 2022. 1, 2

[15] R. Hanocka, G. Metzer, R. Giryes, and D. Cohen-Or. Point2Mesh: A self-prior for deformable meshes. *ACM Trans. Graph.*, 39(4), aug 2020. 2, 8

[16] C.-E. Himeur, T. Lejemble, T. Pellegrini, M. Paulin, L. Barthe, and N. Mellado. PCEDNet: A lightweight neural network for fast and interactive edge detection in 3D point clouds. *ACM Trans. Graph.*, 41(1), nov 2021. 2

[17] H. Huang, D. Li, H. Zhang, U. Ascher, and D. Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM Trans. Graph.*, 28(5):1–7, dec 2009. 2

[18] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. R. Zhang. Edge-aware point set resampling. 32(1), feb 2013. 1, 2, 8

[19] Z. Huang, Y. Wen, Z. Wang, J. Ren, and K. Jia. Surface reconstruction from point clouds: A survey and a benchmark, 2022. 6

[20] M. Kazhdan and H. Hoppe. Screened Poisson surface reconstruction. *ACM Transactions on Graphics*, 32(29):1–13, 2013. 8

[21] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans. Graph.*, 36(4), jul 2017. 6

[22] P. Li, J. Guo, X. Zhang, and D. ming Yan. SECAD-Net: Self-supervised CAD reconstruction by learning sketch-extrude operations, 2023. 2

[23] R. Li, X. Li, P.-A. Heng, and C.-W. Fu. Point cloud upsampling via disentangled refinement, 2021. 1, 2

[24] Y. Li, S. Liu, X. Yang, J. Guo, J. Guo, and Y. Guo. Surface and edge detection for primitive fitting of point clouds. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, New York, NY, USA, 2023. Association for Computing Machinery. 2

[25] B. Liao, C. Xiao, and L. Jin. Efficient feature-preserving local projection operator for geometry reconstruction. In N. Avis and S. Lefebvre, editors, *Eurographics 2011 - Short Papers*. The Eurographics Association, 2011. 2

[26] Y. Lipman, D. Cohen-Or, D. Levin, and H. Tal-Ezer. Parameterization-free projection for geometry reconstruction. *ACM Trans. Graph.*, 26(3):22–es, jul 2007. 2

[27] Y. Liu, S. D'Aronco, K. Schindler, and J. D. Wegner. PC2WF: 3D wireframe reconstruction from raw point clouds, 2021. 2

[28] X. Lu, S. Schaefer, J. Luo, L. Ma, and Y. He. Low rank matrix approximation for 3D geometry filtering. *IEEE Transactions on Visualization and Computer Graphics*, 28(4):1835–1847, 2022. 1, 2

[29] G. Metzer, R. Hanocka, R. Giryes, and D. Cohen-Or. Self-sampling for neural point cloud consolidation. *ACM Trans. Graph.*, 40(5), sep 2021. 1, 2, 8

[30] M. Pauly, R. Keiser, and M. Gross. Multi-scale feature extraction on point-sampled surfaces. *Computer Graphics Forum*, 22(3):281–289, 2003. 2

[31] M.-J. Rakotosaona, P. Guerrero, N. Aigerman, N. Mitra, and M. Ovsjanikov. Learning Delaunay surface elements for mesh reconstruction, 2021. 2

[32] K. Sadekar, A. Tiwari, and S. Raman. Shadow art revisited: A differentiable rendering based approach. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 29–37, January 2022. 6

[33] N. Salman, M. Yvinec, and Q. Merigot. Feature preserving mesh generation from 3D point clouds. *Computer Graphics Forum*, 29(5):1623–1632, 2010. 2

[34] X. Wang, Y. Xu, K. Xu, A. Tagliasacchi, B. Zhou, A. Mahdavi-Amiri, and H. Zhang. PIE-NET: Parametric inference of point cloud edges. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. 2

[35] R. Xu, Z. Dou, N. Wang, S. Xin, S. Chen, M. Jiang, X. Guo, W. Wang, and C. Tu. Globally consistent normal orientation for point clouds by regularizing the winding-number field. *ACM Trans. Graph.*, 42(4), jul 2023. 2

[36] R. Xu, Z. Wang, Z. Dou, C. Zong, S. Xin, M. Jiang, T. Ju, and C. Tu. RFEPS: Reconstructing feature-line equipped polygonal surface. *ACM Trans. Graph.*, 41(6), nov 2022. 1, 2, 3, 4, 6, 8

[37] Y. Ye, Y. Wang, J. Cao, and Z. Chen. Watertight surface reconstruction method for CAD models based on optimal transport. In *Computational Visual Media 2023 Conference Proceedings*, 2023. 1, 2

[38] H. Yu, J. Cao, X. Liu, and Z. Chen. Regularity-constrained point cloud reconstruction of building models via global alignment. *The Visual Computer*, 2024. 2

[39] L. Yu, X. Li, C.-W. Fu, D. Cohen-Or, and P.-A. Heng. EC-Net: An edge-aware point set consolidation network, 2018. 1, 2

[40] T. Zhao, L. Busé, D. Cohen-Steiner, T. Boubekeur, J.-M. Thiery, and P. Alliez. Variational shape reconstruction via quadric error metrics. In *ACM SIGGRAPH 2023 Conference Proceedings*, SIGGRAPH '23, 2023. 1, 2

[41] A. C. Öztireli, G. Guennebaud, and M. Gross. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum*, 28(2):493–501, 2009. 2, 8