Direct Extraction of High-Quality and Feature-Preserving Triangle Meshes from Signed Distance Functions

Longdu Liu
Shandong University
No.72 Binhai Rd, Qingdao, Shandong, China
liulongdu@163.com

Hao Yu
Shandong University
No.72 Binhai Rd, Qingdao, Shandong, China
yuh@mail.sdu.edu.cn

Shiqing Xin
Shandong University
No.72 Binhai Rd, Qingdao, Shandong, China
xinshiqing@sdu.edu.cn

Shuangmin Chen
Qingdao University of Science and Technology
No.99 Songling Rd, Qingdao, Shandong, China
csmqq@163.com

Hongwei Lin
Zhejiang University
No. 38 Zheda Road, Hangzhou, Zhejiang, China
hwlin@zju.edu.cn

Wenping Wang
Texas A&M University
400 Bizzell Street, United States
wenping@tamu.edu

Changhe Tu
Shandong University
No.72 Binhai Rd, Qingdao, Shandong, China
chtu@sdu.edu.cn

Abstract

With advancements in geometric deep learning techniques, neural signed distance functions (SDFs) have gained popularity for their flexibility. Recent studies show that neural SDFs can retain geometric details and encode sharp features. However, during the mesh extraction stage, methods like marching cubes may degrade these geometric details and sharp features, thus compromising the expressiveness of neural SDFs.

In this paper, we aim to develop a general-purpose mesh extraction method for both freeform and CAD models, assuming the availability of a SDF. Our goal is to produce a well-triangulated, resolution-adjustable mesh surface that preserves rich geometric details and distinct feature lines. Our approach is inspired by Centroidal Voronoi Tessellation (CVT) but introduces two key modifications. First, we extend CVT computation to implicit representations, where explicit surface decomposition is not available. Second, we propose a measure for estimating the likelihood that a point lies on feature lines, enabling the extraction of feature-aligned triangle meshes using power diagrams (with site weights positively correlated to the likelihood values). Comprehensive comparisons with state-of-the-art methods demonstrate the superiority of our approach in both feature alignment and triangulation quality.

Keywords: centroidal Voronoi tessellation signed distance function power diagram mesh extraction feature alignment.

1. Introduction

The Signed Distance Function (SDF) is a form of implicit surface representation, where the zero level set defines the underlying surface. It plays a crucial role in various geometric processing tasks, such as surface reconstruction [23, 49], generative modeling [7, 21, 17], shape completion [5, 27], and differentiable rendering [25, 46, 31]. A typical inverse process involves first inferring the SDF of the underlying shape and then extracting a mesh using existing solvers, most commonly the marching cubes algorithm.

In recent years, advancements in geometric deep learning have made neural SDFs [30, 38, 9] an expressive and flexible representation for these tasks. However, when extracting mesh surfaces from a neural SDF, the marching cubes algorithm often fails to capture the full expressiveness of the neural SDF due to limitations like poor triangulation quality and misaligned feature lines. Even improved mesh extraction techniques like Dual Contouring (DC) struggle to maintain geometric details and preserve sharp feature lines. This highlights the need for better mesh extractors, especially for both freeform and CAD models, assuming a SDF is available.

In this paper, we propose a set of strategies to address the aforementioned challenges, assuming the base surface is represented by a SDF. Our approach builds on the concepts of CVT and QEM. First, we introduce a numerical scheme for performing the integration of the objective function at each step. The key operation is to quickly create an umbrella-shaped triangle set for each movable site to sufficiently approximate the implicit surface. Second, while the dual of CVT generates a triangle mesh surface, it does not inherently guarantee feature alignment (as discussed above). To address this, we assign weights to the movable points based on the likelihood of their being located on feature lines. These weights enable the dual of the resulting power diagram to better align with the feature lines.

Our contributions can be summarized as follows:

1. We develop a general-purpose mesh extractor for both freeform and CAD models, assuming the availability of a signed distance function (SDF). It is optimization-driven and simultaneously considers triangulation quality and feature alignment.
2. Given the difficulty of decomposing an implicit surface using a Voronoi diagram, we propose a numerical integration scheme to estimate the objective function, which also supports running Centroidal Voronoi Tessellation (CVT) on implicit representations.
3. We transform the likelihood of a point being located on feature lines into a weight, enabling the extraction of a feature-aligned triangle mesh through power diagrams. Comprehensive comparisons with state-of-the-art methods show that our approach retains rich geometric details and distinct feature lines, yielding high-quality triangulations.

2. Related work

In this section, we review two categories of research works. The first category pertains to isosurface extraction from SDFs, while the second category focuses on mesh optimization that preserves features.

2.1. Isosurface Extraction

According to De Araújo et al. [8], we can further categorize the relevant methods for isosurface extraction into two groups: Spatial Decomposition and Surface Tracking.

Spatial Decomposition. This kind of approaches involves dividing the space into cells like cubes or tetrahedra. While the Marching Cubes (MC) algorithm serves as a fundamental technique, it struggles with topological ambiguities and sharp feature representation. Several improvements have been introduced to address these issues through enhanced lookup tables, precise interpolation, and increasing vertex/edge counts. Techniques like Dual Contouring [18] place vertices on boundaries for accurate isosurfaces, while Dual Marching Cubes [28] combines methods for better accuracy. Additionally, with advancements in geometric deep learning, network-based methods have shown promising results. Approaches like Deep Marching Cubes (DMC) [22], MeshSDF [32], Neural Marching Cubes [3], and FlexiCubes [34] use neural networks to provide differentiable isosurfacing procedures. Another common strategy is to integrate isosurfacing into end-to-end deep learning pipelines for applications such as shape completion [5, 27], model generation [7, 21, 17], or single-view reconstruction [23, 49].

Surface Tracking. This approach directly extracts the surface without spatial subdivision, starting from an isosurface vertex and generating triangles that satisfy specific constraints. Marching Triangles [15, 16] extends triangle edges to create new vertices while maintaining Delaunay constraints, forming a mesh that covers the entire surface. Subsequent studies have explored triangulation adaptivity [2, 19] and feature preservation [26].

2.2. Mesh Optimization

In fact, the requirements for mesh generation essentially align with those for mesh extraction. Common criteria for evaluating mesh quality include accuracy, triangle quality, and feature alignment.

High-Quality Triangulation. By minimizing the Centroidal Voronoi Tessellation (CVT) energy [10], it is possible to achieve an even distribution of movable points on the surface. There is substantial literature on using CVT to generate high-quality mesh surfaces [36, 37]. A notable implementation is proposed in [24], where high-quality meshing is accomplished by combining Restricted Voronoi Diagrams (RVD) with L-BFGS optimization. Additionally, various CVT variants [35, 12, 44, 11, 39] aim to improve triangulations in specific scenarios. However, most of these methods encounter difficulties in aligning with features.

Feature Preserving. Several approaches [40, 43, 45, 48] begin by pre-detecting features and then perform remeshing to preserve them. For instance, VoroCrust [1] suggests placing points symmetrically along pre-detected feature lines. However, feature line detection is impractical for organic models, as identifying weak features is significantly more challenging than detecting strong ones. A more

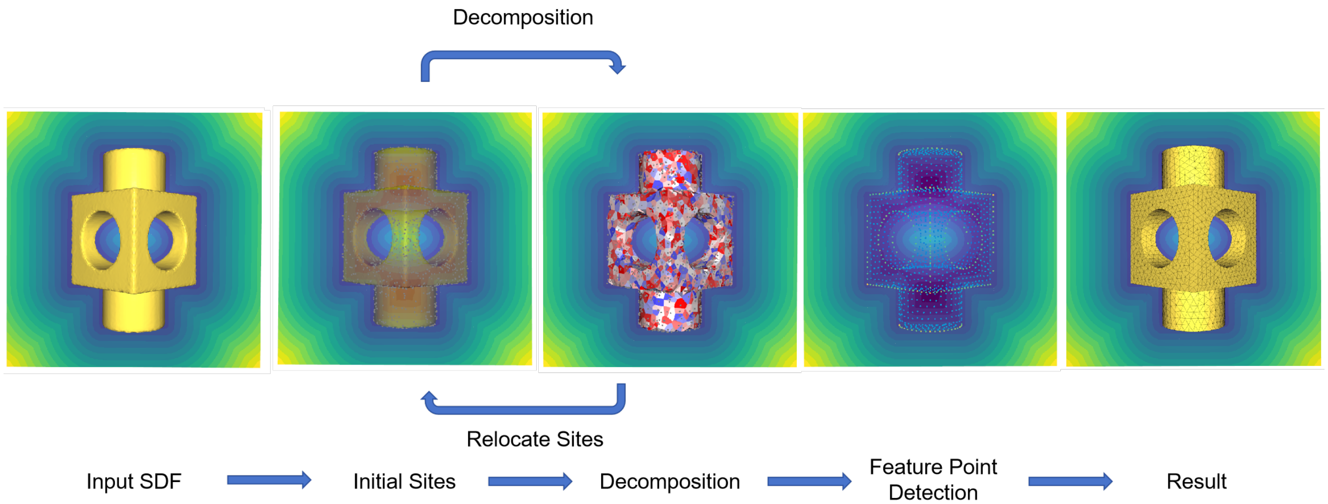

Figure 1. Algorithm Overview. Our algorithm iteratively optimizes a set of movable sites on the implicit surface until convergence. It concludes by constructing a power diagram, yielding a well-triangulated, feature-aligned polygonal surface.

promising strategy involves ensuring that movable points naturally align with underlying features. Many existing algorithms leverage the inherent property of the Quadric Error Metric (QEM), which effectively captures strong features. For example, Valette et al. [37] utilized QEM to guide the placement of Voronoi vertices, ensuring feature alignment. Similarly, Chiang et al. [6] proposed mesh quality enhancement through quadric error-based mesh relaxation. Additionally, Gao et al. [14] extended Optimal Delaunay Triangulation (ODT) to surface meshes by solving a quadratic optimization problem. While recent works [29, 47, 41] have employed QEM to preserve sharp features, they cannot deal with implicit representations. Furthermore, weak feature preservation remains a challenge for QEM-based methods.

3. Overview

We formulate isosurface extraction as an energy minimization problem. Similar to CWF [42], our objective function consists of two terms: E_{CVT} and E_{QE}, where E_{CVT} measures vertex uniformity, and E_{QE} represents feature alignment. As shown in Figure 1, the optimization process begins with sampling the implicit surface. During each iteration, we construct the approximate Voronoi partition of the implicit surface and estimate the objective function. At convergence, we compute the likelihood of a point being located on feature lines and transform it into a weight, then extract the final polygonal surface by computing the power diagram.

4. Algorithm

In this section, we will elaborate on the technical details of our algorithm, including the objective function , Voronoi decomposition, numerical integration, and the construction of the power diagram.

4.1. Objective function

Like CWF, the objective function is defined as follows:

$$E(\{\mathbf{x}_i\}_{i=1}^N) = \lambda_{\text{CVT}} E_{\text{CVT}} + \lambda_{\text{QE}} E_{\text{QE}}$$

where E_{CVT} represents the energy of the Centroidal Voronoi Tessellation (CVT), and E_{QE} measures normal anisotropy. During optimization, E_{CVT} attracts each movable site towards the centroid of its corresponding cell, while E_{QE} encourages each site to align with nearby feature points or lines. The weights λ_{CVT} and λ_{QE} balance these two effects.

The objective function can be rewritten in the following form:

$$E(\{\mathbf{x}_i\}_{i=1}^N) = \sum_{i=1}^{N} \int_{\Omega_i} (\mathbf{x} - \mathbf{x}_i)^{\mathrm{T}} M (\mathbf{x} - \mathbf{x}_i)\, \mathrm{d}s,$$

where the kernel matrix M is defined as:

$$M = \lambda_{\text{CVT}} M_{\text{CVT}} + \lambda_{\text{QE}} M_{\text{QE}}.$$

More specifically,

$$M_{\text{QE}} = \mathbf{n}_{\mathbf{x}} \mathbf{n}_{\mathbf{x}}^{\mathrm{T}}$$

and we simply set M_{CVT} to an identity matrix, which defines the Euclidean distance.

Gradient. The gradient of our objective function is [42]:

$$\nabla_{\mathbf{x}_i} E = \lambda_{\text{CVT}} \nabla_{\mathbf{x}_i} E_{\text{CVT}} + \lambda_{\text{QE}} \nabla_{\mathbf{x}_i} E_{\text{QE}}, \tag{1}$$

where

$$\nabla_{\mathbf{x}_i} E_{\text{CVT}} = \int_{\Omega_i} -2M_{\text{CVT}}(\mathbf{x} - \mathbf{x}_i)\mathrm{d}s, \tag{2}$$

and

$$\nabla_{\mathbf{x}_i} E_{\text{QE}} \approx \int_{\Omega_i} -2M_{\text{QE}}(\mathbf{x} - \mathbf{x}_i)\mathrm{d}s. \tag{3}$$

4.2. Decomposition and numerical integral

Suppose a set of movable points $\{\mathbf{x}_i\}_{i=1}^N$ determines a Voronoi diagram, where each site $\mathbf{x}_i$ dominates a 3D convex polyhedral cell. We can traverse the edges of these cells to identify all intersections with the implicit surface. Generally, the intersections form a polygonal loop $v_1 v_2 \cdots v_k$ that approximately encloses the dominating region of $\mathbf{x}_i$. However, this polygonal cell does not adhere to the implicit surface.

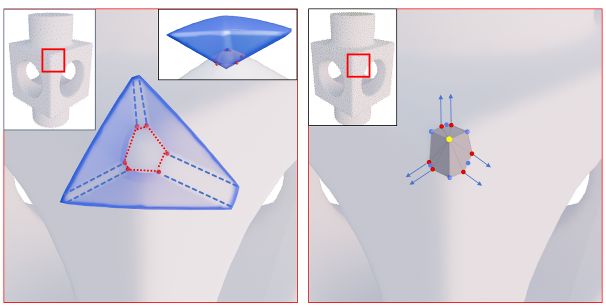

Figure 2. In this example, the polygonal loop consists of 6 vertices (see the left figure). We add a central vertex (yellow) and 6 boundary vertices (blue); See the right figure. The resulting polygonal cell is more accurately adhering to the surface.

Adding $k+1$ vertices. In the following, we will present a simple technique to quickly generate a more accurately adhering polygonal cell. Let $v_1 v_2 \cdots v_k$ denote the polygonal loop (see Figure 2), where v_j represents the intersection between an edge of the cell and the implicit surface, and $\boldsymbol{n}_j$ is the normal vector of v_j's tangent plane. The operation requires two steps. First, using the vertices $v_1, v_2, \ldots, v_k$ and their corresponding normal vectors, we can predict a new vertex that best fits the surrounding planes, similar to the approach used in QEM. Second, we add a vertex between each pair of successive vertices v_j and v_{j+1} in a similar manner.

Least squares solution using the Moore-Penrose inverse. As discussed in QEM method, the newly added vertices can be computed by solving a small-sized linear system. However, the coefficient matrix A may not be full rank, particularly when adding a vertex between a pair of successive vertices v_j and v_{j+1}. In this case, the solution is not unique, but we aim to find a position that is as close to $\frac{v_j+v_{j+1}}{2}$ as possible. Recall that the Moore-Penrose pseudo-inverse A^+ provides the minimum norm solution (closest to the origin). By reshaping $Av = b$ into

$$A\left(v - \frac{v_j + v_{j+1}}{2}\right) = b - A\frac{v_j + v_{j+1}}{2},$$

we can obtain the least squares solution:

$$v = A^+\left(b - A\frac{v_j + v_{j+1}}{2}\right) + \frac{v_j + v_{j+1}}{2}.$$

Numerical integral. As mentioned above, we add $k+1$ vertices, one of which is the central vertex. It is straightforward to divide the polygonal cell into $2k$ triangles, each rooted at the central vertex. As shown in Figure 2, the resulting polygonal representation better adheres to the surface. For computing the numerical integral, we use the Albrecht-Collatz quadrature over each triangle, which involves 6 points: 3 at the midpoints of the edges and 3 located inside the triangle.

4.3. Feature aligned mesh extraction

When the optimization terminates, some points are moved to feature lines while others are uniformly distributed elsewhere. According to CWF, the Delaunay triangulation among these points produces a feature-aligned polygonal surface. However, an occasional failure may occur where four sites—two (p_1, p_2) are located on the feature line and the other two (q_1, q_2) are on opposite sides. If these four sites share a common Voronoi vertex (also on the feature line), two possible triangulation configurations can result, each being a minimizer of our objective function. In fact, feature preservation is also violated if q_1 and q_2 share a common segment that aligns with the feature line. Consequently, connecting the movable sites based on the Voronoi diagram may lead to feature misalignment; See the inset figure.

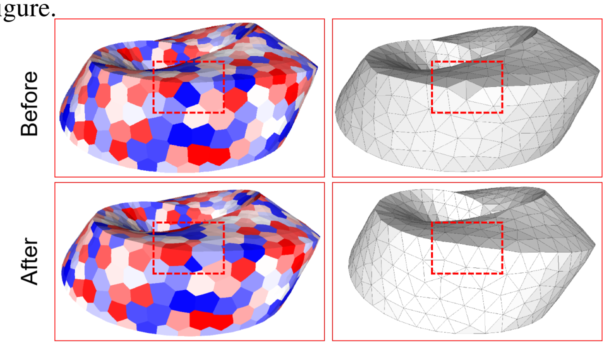

Figure 3. Power diagram and its dual.

To measure how the site $\mathbf{x}_i$ is positioned relative to a feature point or line, we compute the difference between $\mathbf{x}_i$'s normal and the normal vectors of the boundary curve:

$$W_i = \frac{\int_{\partial\Omega_i} \|\mathbf{x} - \mathbf{x}_i\|^2 \left(1 - \left(\mathbf{n}_{\mathbf{x}}^{\mathrm{T}} \mathbf{n}_{\mathbf{x}_i}\right)^2\right) \mathrm{d}s}{\int_{\partial\Omega_i} \mathrm{d}s}. \tag{4}$$

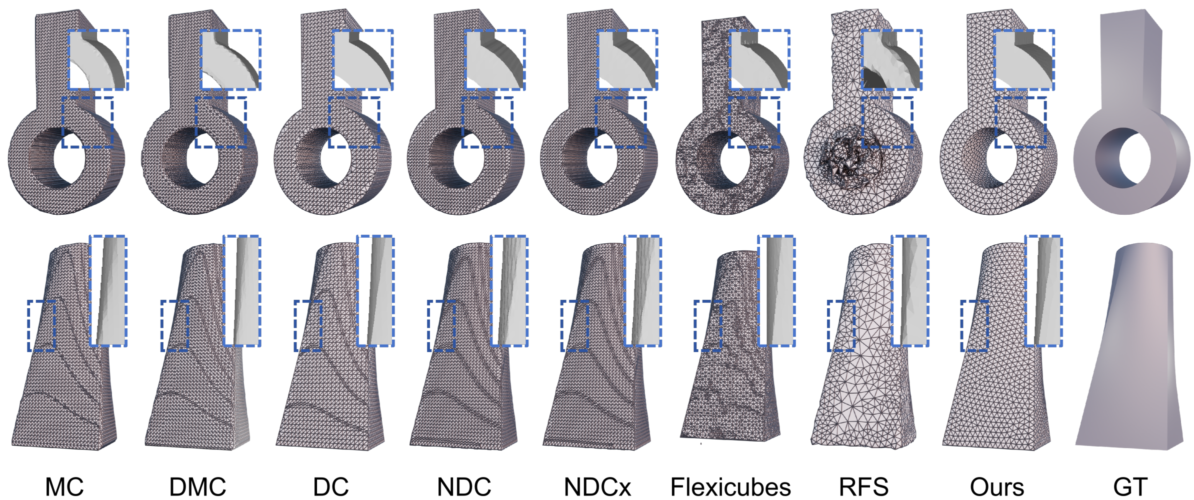

Figure 4. Comparison with state-of-the-art methods on the two CAD models.

It can be observed that when the cell is situated on a planar region, $W_i = 0$. Conversely, when the cell is located in a curved region, the use of squared distance in W_i aligns with the definition of power diagrams. Figure 3 illustrates that in the resulting power diagram, the feature-line sites control a larger region, making them more easily connectable.

5. Experimental Results and Evaluation

All of our experiments were conducted on a computer equipped with an AMD Ryzen 5995WX CPU and 128 GB of memory. The tests were performed on a total of 100 CAD models with distinct features, taken from the ABC dataset [20], and 21 organic models with weak features. For CAD models, we set the target vertex count to 2500, while for organic models, we set the target vertex count to 5000.

Evaluation Metrics. To assess triangle quality, we utilize the *TriangleQ* indicator [13], with a value closer to 1.0 indicating proximity to an equilateral triangle, i.e.,

$$TriangleQ(t) = \frac{6}{\sqrt{3}} \frac{S_t}{p_t h_t} \tag{5}$$

where S_t, p_t, and h_t represent the area, half-perimeter, and the longest edge length of the triangle t, respectively. Moreover, three indicators: *Chamfer Distance* (CD), *F-score* (F1), and *Normal Consistency* (NC) are employed to measure the difference between the extracted surface and the original version. In addition, *OpenB* represents the number of open mesh edges in the simplification result, and *NMV* denotes the number of non-manifold vertices. For CAD models, we employ *Edge Chamfer Distance* (ECD) and *Edge F-score* (EF1), proposed by NMC [3], to measure the extent to which the feature lines are preserved.

Parameters. In our experiments, the same parameters are set for all the models in this paper, including CAD models and organic models. At the beginning of the optimization, we set $\lambda_{\text{QE}} = \lambda_{\text{metric}} = 1.0$, where λ_{QE} remains unchanged throughout the optimization, but λ_{metric} undergoes a decaying process. The L-BFGS solver is utilized to solve the optimization problem, where the termination condition is set by referring to the gradient norm, with a tolerance of $1e^{-8}$.

Table 1. Quantitative comparison on 100 CAD models, taken from the ABC dataset [20]. Each CAD model has strong features. The **<u>best</u>** scores are emphasized in bold with underlining, while the **second best** scores are highlighted only in bold.

	Resolution	MC	DC	NDC	NDCx	DMC	Flexicubes	RFS	Ours
CD $(\times 10^4) \downarrow$	32	0.4310	**0.1366**	0.2463	0.2435	0.5519	2.7038	38.497	**<u>0.0855</u>**
	64	0.2308	0.5048	**0.1048**	0.1051	0.2399	25.8899	27.2147	**<u>0.0855</u>**
F1 $\uparrow$	32	0.7610	0.8401	0.8614	**0.8661**	0.7101	0.3975	0.3583	**<u>0.9328</u>**
	64	0.8830	0.8979	**0.9303**	0.9302	0.8716	0.4813	0.5656	**<u>0.9328</u>**
NC $\uparrow$	32	0.9575	**0.9833**	0.9731	0.9746	0.9546	0.9335	0.8488	**<u>0.9873</u>**
	64	0.9789	**0.9842**	0.9857	0.9865	0.9777	0.9242	0.8925	**<u>0.9873</u>**
ECD $(\times 10^2) \downarrow$	32	15.97	**0.3123**	0.3345	0.3237	21.32	0.6124	2.932	**<u>0.1431</u>**
	64	12.37	0.2013	**0.1732**	0.2193	21.96	1.837	2.998	**<u>0.1431</u>**
EF1 $\uparrow$	32	0.0503	1.0000	0.3906	**0.3974**	0.0518	0.1590	0.0401	**<u>0.5665</u>**
	64	0.1123	**0.5547**	0.5484	0.5514	0.0597	0.2886	0.0963	**<u>0.5665</u>**
TriangleQ $\uparrow$	32	0.6373	0.5591	0.6668	0.6640	0.7033	0.6136	**0.8241**	**<u>0.8876</u>**
	64	0.6520	0.6739	0.6772	0.6761	0.7046	0.6259	**0.8253**	**<u>0.8876</u>**
HD $(\times 10^2) \downarrow$	32	0.2732	0.5400	0.2215	**0.2201**	0.3759	1.376	2.155	**<u>0.0328</u>**
	64	0.1544	0.0989	0.1327	**0.0716**	0.1464	2.182	1.384	**<u>0.0328</u>**
OpenB $\downarrow$	32	**<u>0</u>**	38	207	206	**<u>0</u>**	64	111	**<u>0</u>**
	64	**<u>0</u>**	78	618	617	**<u>0</u>**	105	50	**<u>0</u>**
NMV $\downarrow$	32	10	41	67	65	**6**	42	1404	**<u>0</u>**
	64	77	96	201	224	**4**	88	6626	**<u>0</u>**

5.1. Comparison Methods

We compare our method with seven state-of-the-art (SOTA) methods: MC, DC, NDC, NDCx, DMC, Flexicude, and RFS. Marching Cubes (MC) [15] traverses voxel grids cube by cube, determining the surface based on vertex values and predetermined patterns. The resulting mesh vertices are guaranteed to lie on the voxel grid. Dual Contouring (DC) [18] transitions to a dual representation, extracting

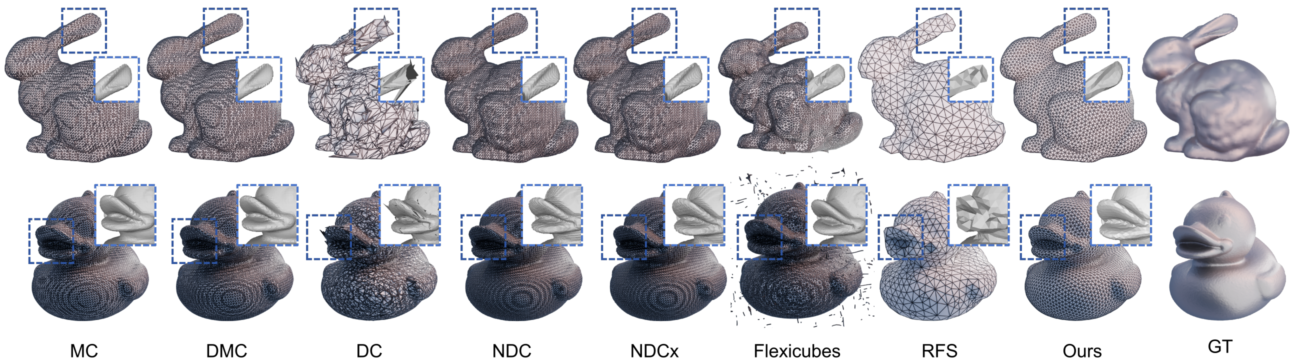

Figure 5. Comparison with state-of-the-art methods on the bunny model and duck model.

Table 2. Quantitative comparison on 21 organic models with weak features.

	Resolution	MC	DC	NDC	NDCx	DMC	Flexicubes	RFS	Ours
CD $(\times 10^4)\downarrow$	32	3.958	21.86	50.283	**0.2151**	4.309	2.646	92.28	**0.0479**
	64	**0.4352**	2.877	73.045	53.54	0.4667	48.27	109.8	**0.0479**
F1 $\uparrow$	32	0.6236	0.2649	0.6379	**0.7219**	0.4358	0.6152	0.1910	**0.9923**
	64	**0.9350**	0.5576	0.5713	0.6094	0.8591	0.7189	0.2717	**0.9923**
NC $\uparrow$	32	0.9340	0.8804	0.8956	0.9314	0.9064	**0.9458**	0.8241	**0.9855**
	64	0.9724	0.9367	0.8974	0.8971	**0.9620**	0.9076	0.8374	**0.9855**
ECD $(\times 10^2)\downarrow$	32	0.0741	**0.0347**	0.0355	0.1045	0.0668	0.0479	0.0572	**0.0193**
	64	0.0370	**0.0211**	0.0508	0.0661	0.0356	0.0269	0.0747	**0.0193**
EF1 $\uparrow$	32	0.3457	0.3456	**0.4083**	0.0000	0.3473	0.3402	0.3350	**0.5294**
	64	0.3929	0.3662	0.3032	0.2489	0.3587	**0.4248**	0.2878	**0.5294**
TriangleQ $\uparrow$	32	0.5853	0.4705	0.6683	0.6498	0.6966	0.6233	**0.8018**	**0.8873**
	64	0.5919	0.5019	0.6800	0.6733	0.7074	0.6318	**0.8428**	**0.8873**
HD $(\times 10^2)\downarrow$	32	1.283	1.943	1.986	1.995	1.120	**0.7008**	4.206	**0.0698**
	64	0.2609	1.066	**0.1023**	2.297	0.2511	0.6113	4.432	**0.0698**
OpenB $\downarrow$	32	**0**	289	5	3	**0**	24	631	**0**
	64	**0**	504	6	5	**0**	25	919	**0**
NMV $\downarrow$	32	21	226	15	13	**6**	51	7062	**0**
	64	14	645	14	12	**4**	74	9042	**0**

mesh vertices that typically locate within grid cells to more effectively capture sharp geometric features. Neural Dual Contouring (NDC) [4] replaces QEF solving with neural networks, enhancing the extraction quality from imperfect yet fixed scalar functions. NDCx is basically NDC with a more complex backbone network, which is slower than NDC but has better reconstruction accuracy. Dual Marching Cubes (DMC) [28] enhances the connectivity of the extracted mesh by extracting a mesh along the dual connectivity of the grid compared to Dual Contouring. FlexiCubes [34] adopt a specific Dual Marching Cubes formulation and introduce extra degrees of freedom to position each extracted vertex within its dual cell flexibly. In contrast to other methods, RFS [33] begins with a triangle mesh encompassing the surface and then shrinkwraps the underlying surface using a gradient flow to minimize an energy function.

We present the statistics of quantitative comparisons for the 100 CAD models and the 21 organic models with weak features in Table 1 and Table 2, and illustrate the qualitative comparisons in Fig 4 and Fig 5. In comparison with these methods, our approach simultaneously achieves the best performance in terms of accuracy, triangle quality, and feature alignment.

5.2. Comparsion with CWF.

The CWF[42] method can effectively preserve weak features during triangular mesh simplification and has a certain ability to recover features. Therefore, it is an intuitive idea to reconstruct the mesh by traditional reconstruction methods, and then simplify it while preserving features using CWF. However, the traditional RVD method used by CWF is sensitive to the number and quality of input meshes. Because the mesh extraction from SDFs always has an excessive number of triangles, CWF requires a considerable amount of time for intersection operations. Additionally, it may encounter non-manifold or self-intersecting situations, leading to algorithm failure. Therefore, our method based on direct SDF reconstruction is more efficient.

5.3. Using Trilinear Interpolation.

To further reduce the number of SDF evaluations, we can use trilinear interpolation instead of differentiable distance queries, although this may have a certain impact on the accuracy of our algorithm. We demonstrate this effect through two experiments. In one set, only the positions at the vertices of a regular grid are provided, while in the other set, we augment the fandisk model with additional normal information at each vertex. By comparing the visual results in Figure 6, we find that using trilinear interpolation can also yield good results. However, compared to differentiable distance queries, there is a potential risk of losing feature information.

5.4. Run-time Performance.

The run-time performance statistics are presented in Figure 7. The tests were conducted on the block model and bunny model, each with varying target numbers ranging from $0.5K$ points to $10K$ points. The total running time primarily includes those for the construction of RVDs and

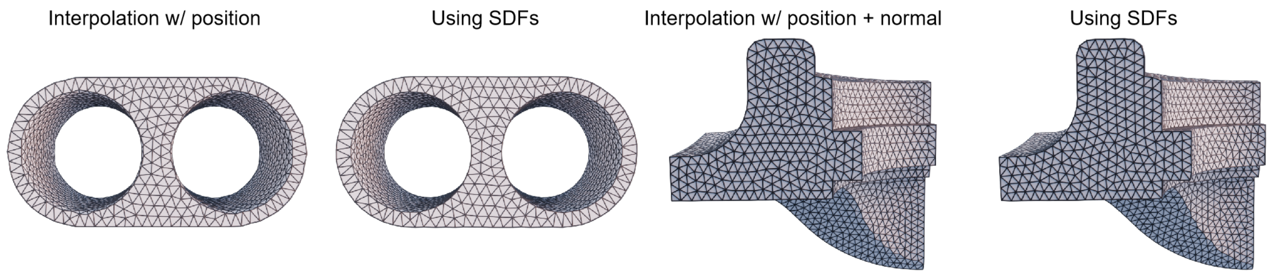

Figure 6. Comparing with Trilinear Interpolation.

the optimization, with the optimization being the most time-consuming operation. Additionally, it can be observed that the optimization typically requires 40 to 50 iterations. The practical timing cost is approximately $O(kn)$, where n is the target number of vertices, and k is the number of iterations.

6. Limitations and Conclusion

Limitations In its current form, our method has at least four limitations. Firstly, we have not demonstrated its potential for generating anisotropic meshes with feature alignment, as anisotropic Voronoi solvers are not yet available. Secondly, our implementation requires further GPU-based acceleration in the future. Additionally, due to the non-differentiability of the proposed meshing process, it is not suitable for integration into end-to-end learning frameworks that utilize neural SDFs as internal shape representations. Therefore, the proposed meshing technique can only function as a post-processing step following SDF learning. Lastly, our method is limited to closed surfaces. In the future, it is planned to extend the method to open surfaces by leveraging Unsigned Distance Fields (UDF).

Conclusion In this paper, we propose an iso-surface extraction method that simultaneously considers the requirements of triangle quality and feature alignment, by extending the mesh-based functional optimization framework to SDFs. The proposed method naturally supports running CVT on implicit surfaces. Moreover, a set of strategies are introduced to improve robustness, including surface-adhering projection and feature line consolidation using power diagrams. Finally, extensive comparison experiments validate its advantages over existing state-of-the-art (SOTA) methods.

References

[1] Ahmed Abdelkader, Chandrajit L Bajaj, Mohamed S Ebeida, Ahmed H Mahmoud, Scott A Mitchell, John D Owens, and Ahmad A Rushdi. Vorocrust: Voronoi meshing without clipping. *ACM Transactions on Graphics (TOG)*, 39(3):1–16, 2020. 2

[2] Samir Akkouche and Eric Galin. Adaptive implicit surface polygonization using marching triangles. In *Computer Graphics Forum*, volume 20, pages 67–80. Wiley Online Library, 2001. 2

[3] Zhiqin Chen and Hao Zhang. Neural marching cubes. *ACM Transactions on Graphics (TOG)*, 40(6):1–15, 2021. 2, 5

[4] Zhiqin Chen, Andrea Tagliasacchi, Thomas Funkhouser, and Hao Zhang. Neural dual contouring. *ACM Transactions on Graphics (TOG)*, 41 (4):1–13, 2022. 6

[5] Yen-Chi Cheng, Hsin-Ying Lee, Sergey Tulyakov, Alexander G Schwing, and Liang-Yan Gui. Sdfusion: Multimodal 3d shape completion, reconstruction, and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4456–4465, 2023. 1, 2

[6] Chien-Hsing Chiang, Bin-Shyan Jong, and Tsong-Wuu Lin. A robust feature-preserving semi-regular remeshing method for triangular meshes. *The Visual Computer*, 27:811–825, 2011. 3

[7] Gene Chou, Yuval Bahat, and Felix Heide. Diffusion-sdf: Conditional generative modeling of signed distance functions. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2262–2272, 2023. 1, 2

[8] Bruno Rodrigues De Araújo, Daniel S Lopes, Pauline Jepp, Joaquim A Jorge, and Brian Wyvill. A survey on implicit surface polygonization. *ACM Computing Surveys (CSUR)*, 47(4):1–39, 2015. 2

[9] Qiujie Dong, Rui Xu, Pengfei Wang, Shuangmin Chen, Shiqing Xin, Xiaohong Jia, Wenping Wang, and Changhe Tu. Neurcadrecon: Neural representation for reconstructing cad surfaces by enforcing zero gaussian curvature. *arXiv preprint arXiv:2404.13420*, 2024. 1

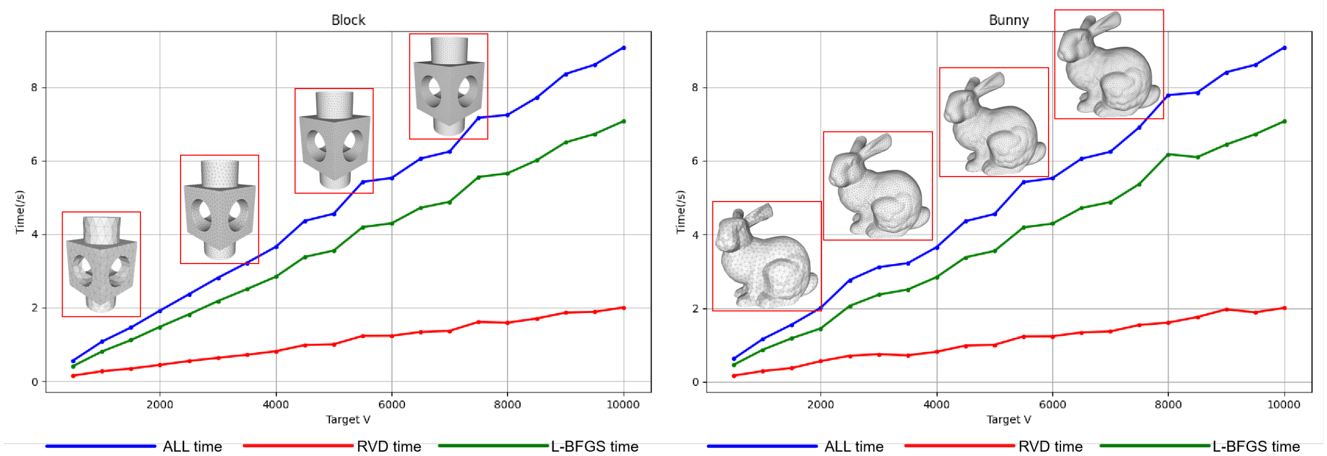

Figure 7. Run-time Performance.

[10] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM review*, 41(4):637–676, 1999. 2

[11] Xingyi Du, Xiaohan Liu, Dong-Ming Yan, Caigui Jiang, Juntao Ye, and Hui Zhang. Field-aligned isotropic surface remeshing. In *Computer Graphics Forum*, volume 37, pages 343–357, 2018. 2

[12] John Edwards, Wenping Wang, and Chandrajit Bajaj. Surface segmentation for improved remeshing. In *Proceedings of the 21st International Meshing Roundtable*, pages 403–418. Springer, 2013. 2

[13] Pascal J Frey and Houman Borouchaki. Surface mesh quality evaluation. *International journal for numerical methods in engineering*, 45(1):101–118, 1999. 5

[14] Zhanheng Gao, Zeyun Yu, and Michael Holst. Feature-preserving surface mesh smoothing via suboptimal delaunay triangulation. *Graphical Models*, 75 (1):23–38, 2013. 3

[15] Adrian Hilton, Andrew J Stoddart, John Illingworth, and Terry Windeatt. Marching triangles: range image fusion for complex object modelling. In *Proceedings of 3rd IEEE international conference on image processing*, volume 2, pages 381–384. IEEE, 1996. 2, 5

[16] Adrian Hilton, John Illingworth, et al. Marching triangles: Delaunay implicit surface triangulation. *University of Surrey*, 1997. 2

[17] Lutao Jiang, Ruyi Ji, and Libo Zhang. Sdf-3dgan: A 3d object generative method based on implicit signed distance function. *arXiv preprint arXiv:2303.06821*, 2023. 1, 2

[18] Tao Ju, Frank Losasso, Scott Schaefer, and Joe Warren. Dual contouring of hermite data. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 339–346, 2002. 2, 5

[19] Tasso Karkanis and A James Stewart. Curvature-dependent triangulation of implicit surfaces. *IEEE Computer Graphics and Applications*, 21(2):60–69, 2001. 2

[20] Sebastian Koch, Albert Matveev, Zhongshi Jiang, Francis Williams, Alexey Artemov, Evgeny Burnaev, Marc Alexa, Denis Zorin, and Daniele Panozzo. Abc: A big cad model dataset for geometric deep learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9601–9611, 2019. 5

[21] Muheng Li, Yueqi Duan, Jie Zhou, and Jiwen Lu. Diffusion-sdf: Text-to-shape via voxelized diffusion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12642–12651, 2023. 1, 2

[22] Yiyi Liao, Simon Donne, and Andreas Geiger. Deep marching cubes: Learning explicit surface representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2916–2925, 2018. 2

[23] Jingwang Ling, Zhibo Wang, and Feng Xu. Shadowneus: Neural sdf reconstruction by shadow ray supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 175–185, 2023. 1, 2

[24] Yang Liu, Wenping Wang, Bruno Lévy, Feng Sun, Dong-Ming Yan, Lin Lu, and Chenglei Yang. On cen-

troidal voronoi tessellation—energy smoothness and fast computation. *ACM Transactions on Graphics (TOG)*, 28(4):1–17, 2009. 2

[25] Yu-Tao Liu, Li Wang, Jie Yang, Weikai Chen, Xiaoxu Meng, Bo Yang, and Lin Gao. Neudf: Leaning neural unsigned distance fields with volume rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 237–247, 2023. 1

[26] Neil H McCormick and Robert B Fisher. Edge-constrained marching triangles. In *Proceedings. First International Symposium on 3D Data Processing Visualization and Transmission*, pages 348–351. IEEE, 2002. 2

[27] Paritosh Mittal, Yen-Chi Cheng, Maneesh Singh, and Shubham Tulsiani. Autosdf: Shape priors for 3d completion, reconstruction and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 306–315, 2022. 1, 2

[28] Gregory M Nielson. Dual marching cubes. In *IEEE visualization 2004*, pages 489–496. IEEE, 2004. 2, 6

[29] Dakshata Panchal and Deepak Jayaswal. Feature sensitive geometrically faithful highly regular direct triangular isotropic surface remeshing. *Sādhanā*, 47(2): 94, 2022. 3

[30] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 1

[31] Eduard Pujol and Antonio Chica. Rendering piecewise approximations of sdfs through analytic intersections. *Computers & Graphics*, page 103981, 2024. 1

[32] Edoardo Remelli, Artem Lukoianov, Stephan Richter, Benoit Guillard, Timur Bagautdinov, Pierre Baque, and Pascal Fua. Meshsdf: Differentiable iso-surface extraction. *Advances in Neural Information Processing Systems*, 33:22468–22478, 2020. 2

[33] Silvia Sellán, Christopher Batty, and Oded Stein. Reach for the spheres: Tangency-aware surface reconstruction of sdfs. In *SIGGRAPH Asia 2023 Conference Papers*, pages 1–11, 2023. 6

[34] Tianchang Shen, Jacob Munkberg, Jon Hasselgren, Kangxue Yin, Zian Wang, Wenzheng Chen, Zan Gojcic, Sanja Fidler, Nicholas Sharp, and Jun Gao. Flexible isosurface extraction for gradient-based mesh optimization. *ACM Trans. Graph.*, 42(4):37–1, 2023. 2, 6

[35] Feng Sun, Yi-King Choi, Wenping Wang, Dong-Ming Yan, Yang Liu, and Bruno Lévy. Obtuse triangle suppression in anisotropic meshes. *Computer Aided Geometric Design*, 28(9):537–548, 2011. 2

[36] Sébastien Valette and Jean-Marc Chassery. Approximated centroidal voronoi diagrams for uniform polygonal mesh coarsening. In *Computer Graphics Forum*, volume 23, pages 381–389. Wiley Online Library, 2004. 2

[37] Sébastien Valette, Jean Marc Chassery, and Rémy Prost. Generic remeshing of 3d triangular meshes with metric-dependent discrete voronoi diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 14(2):369–381, 2008. 2, 3

[38] Ruian Wang, Zixiong Wang, Yunxiao Zhang, Shuangmin Chen, Shiqing Xin, Changhe Tu, and Wenping Wang. Aligning gradient and hessian for neural signed distance function. *Advances in Neural Information Processing Systems*, 36, 2024. 1

[39] Yiqun Wang, Dong-Ming Yan, Xiaohan Liu, Chengcheng Tang, Jianwei Guo, Xiaopeng Zhang, and Peter Wonka. Isotropic surface remeshing without large and small angles. *IEEE Transactions on Visualization and Computer Graphics*, 25(7): 2430–2442, 2018. 2

[40] Lijun Xie, Ligang Chen, Jianjun Chen, and Yao Zheng. Surface remeshing with feature preservation. *Journal of Information & Computational Science*, 9 (9):2393–2400, 2012. 2

[41] Rui Xu, Zixiong Wang, Zhiyang Dou, Chen Zong, Shiqing Xin, Mingyan Jiang, Tao Ju, and Changhe Tu. Rfeps: Reconstructing feature-line equipped polygonal surface. *ACM Transactions on Graphics (TOG)*, 41(6):1–15, 2022. 3

[42] Rui Xu, Longdu Liu, Ningna Wang, Shuangmin Chen, Shiqing Xin, Xiaohu Guo, Zichun Zhong, Taku Komura, Wenping Wang, and Changhe Tu. Cwf: Consolidating weak features in high-quality mesh simplification. *ACM Transactions on Graphics (TOG)*, 43 (4):1–14, 2024. 3, 6

[43] Dong-Ming Yan and Peter Wonka. Gap processing for adaptive maximal poisson-disk sampling. *ACM Transactions on Graphics (TOG)*, 32(5):1–15, 2013. 2

[44] Dong-Ming Yan and Peter Wonka. Non-obtuse remeshing with centroidal voronoi tessellation. *IEEE*

Transactions on Visualization and Computer Graphics, 22(9):2136–2144, 2015. 2

[45] Dong-Ming Yan, Jianwei Guo, Xiaohong Jia, Xiaopeng Zhang, and Peter Wonka. Blue-noise remeshing with farthest point optimization. In *Computer Graphics Forum*, volume 33, pages 167–176, 2014. 2

[46] Mulin Yu, Tao Lu, Linning Xu, Lihan Jiang, Yuanbo Xiangli, and Bo Dai. Gsdf: 3dgs meets sdf for improved rendering and reconstruction. *arXiv preprint arXiv:2403.16964*, 2024. 1

[47] Tong Zhao, Laurent Busé, David Cohen-Steiner, Tamy Boubekeur, Jean-Marc Thiery, and Pierre Alliez. Variational shape reconstruction via quadric error metrics. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–10, 2023. 3

[48] Zichun Zhong, Xiaohu Guo, Wenping Wang, Bruno Lévy, Feng Sun, Yang Liu, Weihua Mao, et al. Particle-based anisotropic surface meshing. *ACM Transactions on Graphics (TOG)*, 32(4):99–1, 2013. 2

[49] Jingsen Zhu, Yuchi Huo, Qi Ye, Fujun Luan, Jifan Li, Dianbing Xi, Lisha Wang, Rui Tang, Wei Hua, Hujun Bao, et al. I2-sdf: Intrinsic indoor scene reconstruction and editing via raytracing in neural sdfs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12489–12498, 2023. 1, 2