CSIAM Trans. Appl. Math.
doi: 10.4208/csiam-am.SO-2025-0071

Vol. **x**, No. x, pp. 1-28
xx 2025

Smooth Bijective Projection of Polygonal Meshes via a Cubic B-Spline Shell

Guilong He[1], Huibiao Wen[1], Shuangmin Chen[2,*], Lin Lu[1], Shiqing Xin[1] and Changhe Tu[1]

[1] *Shandong University, Qingdao 250100, P.R. China.*
[2] *Qingdao University of Science and Technology, Qingdao 266061, P.R. China.*

Received 1 July 2025; Accepted 28 July 2025

Abstract. Polygonal meshes are a fundamental surface representation, yet their resolution can vary significantly. Establishing a smooth, bijective projection between meshes of different resolutions is crucial for consistent attribute transfer but becomes challenging when handling sharp bends and complex geometric features.
This paper presents a novel approach to address this challenge by implicitly representing the shell enclosing two polygonal meshes using a cubic trivariate B-spline function, where the inner and outer bounding surfaces are formulated as level sets of a single cubic B-spline function. Our method enforces the bijective projection requirements on the cubic B-spline function, ensuring that the resulting gradient field naturally defines a robust bijective projection. Leveraging the favorable properties of cubic B-spline functions – namely, 1) sufficient smoothness while maintaining expressive representation, and 2) computational efficiency and ease of implementation, our approach efficiently computes a smooth and bijective projection even for challenging cases. Compared to existing shell-based bijective projection methods, our method consistently produces valid bijective projections, even in complex scenarios, outperforming state-of-the-art techniques. We further demonstrate its effectiveness in robust attribute transfer and precision-controlled shape manipulation.

AMS subject classifications: 68U05, 97R60

Key words: Cubic B-spline shell, attribute transfer, smooth bijective projection.

1 Introduction

Polygonal meshes are ubiquitous in digital geometry processing, offering a discrete representation for encoding 3D surfaces through vertices, edges, and facets. While they

*Corresponding author. *Email addresses:* hegl2533@gmail.com (G. He), ericvein@163.com (H. Wen), csmqq@163.com (S. Chen), llu@sdu.edu.cn (L. Lu), xinshiqing@sdu.edu.cn (S. Xin), chtu@sdu.edu.cn (C. Tu)

provide exceptional modeling flexibility, performing tasks such as mesh simplification [14, 42], real-time rendering [48], and collision detection [4, 43] consistently across different mesh triangulations remains challenging. Large-scale models or irregular tessellations often introduce unnecessary computational burdens, making efficient processing difficult. Research efforts have sought to address these limitations by exploring geometric proxies that balance computational efficiency with shape fidelity.

Recent research suggests that shell representations – volumetric domains bounded by two closely spaced surfaces with controlled separation – offer an effective solution to this challenge. Analogous to sandwich composites in continuum mechanics [24], shell structures provide several advantages, such as bounded geometric approximation errors [20], bijective attribute mappings [39], and unified surface-volume parametrizations [31]. These properties have led to widespread adoption across applications such as texturing [62], meshing [34], and physical simulation [7]. In these applications, attributes such as textures, displacements, and physical properties must be preserved during remeshing.

Existing explicit shell-based approaches, including linear shell [24] and high-order shell [39]†, employ piecewise bijective projections to maintain attribute integrity within the shell space. However, these methods struggle to ensure smooth bijectivity. As shown in Fig. 1, linear shells can introduce distortions. Furthermore, explicit shell representations require careful handling to prevent adjacent intervals from intersecting, as such intersections violate bijectivity and reduce smoothness.

To address these fundamental limitations, we propose a novel mathematical framework for constructing cubic B-spline shells (CBS) that ensures two key properties: 1) an analytic representation of sandwich-walled spaces using implicit functions, and 2) homeomorphically preserved projections within the shell domain. Our approach defines bounding surfaces as iso-surfaces of a carefully designed implicit function f, ensuring smoothness and continuity via a C^2-regular B-spline construction. The bijective projection requirement translates to a gradient constraint $|\nabla f| \geq \delta > 0$ throughout the shell region, which we enforce through analytic gradient bounding in our spline formulation. Furthermore, this can be efficiently solved using a sparse linear system. Compared to existing proxies, B-spline functions provide an optimal balance between geometric flexibility and computational efficiency, as their localized nature significantly reduces computational overhead by restricting function evaluations to a subset of control parameters. Additionally, we introduce a gradient-normal alignment term and prove that when the gradients of the implicit function align with the normal vectors at mesh edges, the surface-restricted gradient changes tend to be harmonic.

To demonstrate the effectiveness of CBS, we showcase two key applications: 1) mesh editing with global precision control, and 2) attribute transfer between surfaces within the shell. These applications highlight our framework's ability to maintain high-fidelity mappings while significantly reducing distortion compared to existing methods.

†Since the code is not open, we limit our comparison with linear shell.

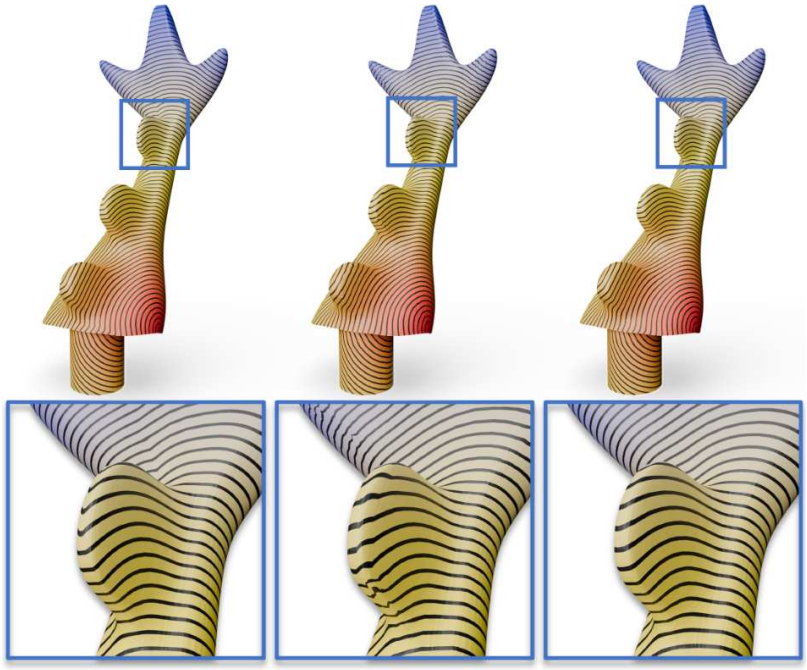

Figure 1: The geodesic distance fields computed via the heat method [10] on identical high-quality triangular mesh surfaces are subsequently transferred back to the original low-quality input mesh. Comparative analysis with exact geodesic fields demonstrates that our methodology maintains fidelity without distortion, whereas the linear shell exhibits significant artifacts and errors on the shared boundaries of neighboring prisms.

2 Prior works

In this section, we review the previous research most pertinent to our study, focusing on three key areas: implicit surface representations, shell generation for polygonal meshes, and attribute transfer.

2.1 Implicit representation

Implicitization serves as a powerful framework in digital geometry processing, representing shapes as level sets of continuous functions. These implicit functions offer smoothness and differentiability, making them particularly effective for applications that require robust, continuous representations.

One of the most prominent applications is implicit surface reconstruction, where a smooth function is fitted to point clouds to recover detailed geometry [27, 51, 61]. To ensure computational efficiency, the function is typically constrained within a structured function space and expressed as a linear combination of predefined basis functions. Common choices include B-splines [26, 45] and radial basis functions (RBFs) [36, 56], which provide both smooth approximation and efficient evaluation.

Implicit representations offer two notable advantages: (1) Continuity and differentiability: These properties enable the use of gradient-based optimization, PDE solvers, and physical simulations, which are central to tasks such as shape fitting and deformation [3, 6, 11]. (2) Topological flexibility: Unlike mesh-based methods, implicit models naturally support changes in topology, making them ideal for simulations, blending, and morphing [50, 63].

Thanks to these benefits, implicitization techniques have seen wide adoption in geometric modeling, physics-based simulation, and computer-aided design, forming a cornerstone of modern shape representation strategies.

2.2 Shell generation

Techniques for generating shell structures from polygonal meshes have evolved along two primary methodological axes: explicit geometric operations and implicit field representations. As these approaches address distinct computational challenges, their comparative analysis reveals critical trade-offs in precision, robustness, and implementation complexity.

Explicit methods. Explicit methods primarily rely on geometric displacement strategies, where the standard workflow encompasses topological surface duplication, directional offsetting along predefined vectors, and seamless topological gap closure to ensure facet connectivity while preventing intersection artifacts. Early implementations [47, 49, 62] established the foundational paradigm of normal-based offsetting, while later advancements [21, 24] incorporated gradient fields derived from generalized distance functions to refine displacement direction. The core challenge in these approaches lies in preserving manifold integrity during offset propagation and ensuring continuity at primitive junctions to avoid topological inconsistencies.

Although significant progress has been made through techniques such as voxel-based regularization [31], curvature-aware displacement [55], coarse tetrahedral meshes [25] and high-order elements [39], explicit methods remain inherently dependent on the quality of the input mesh. This reliance becomes particularly problematic for models containing non-manifold edges or degenerate triangles, where error propagation can escalate exponentially, ultimately compromising geometric fidelity and computational stability.

Implicit methods. Implicit formulations model shells as iso-surfaces of bounded scalar fields. The foundational works [2, 52] defined shells through truncated signed distance functions, enabling unified handling of complex topologies. Building upon this foundation, subsequent research has explored various approaches to enhance the fidelity and efficiency of implicit modeling. Wang *et al.* [59] demonstrated the effectiveness of representing polygonal surfaces with collections of simple solids for containment checks, highlighting the potential of implicitly defined shells in practical applications. In addition to the commonly used distance fields, some methods [5, 41] also use the winding number field and its generalized versions for implicit surface representation. The main

advantage of these methods is their stronger robustness to noise and sparse scenes. Unlike explicit methods that often require precise mesh generation, implicit formulations provide greater flexibility and robustness against variations in input quality.

2.3 Attribute transfer

Attribute transfer, a fundamental concept in computer graphics, involves mapping geometric and topological properties from one shape or structure to another. This process is particularly useful in applications such as mesh deformation, fluid simulation, and shape modification [28,35], where the ability to preserve certain properties while deforming or transforming shapes is crucial. We review here only relevant research in attribute transfer techniques, focusing on projection-based methods and parameterization-based approaches.

Projection-based methods. One of the most common approaches is projection-based attribute transfer, where attributes from one surface are projected onto another. Early methods relied on simple orthogonal, spherical, and cage-based projections [8,12], which, despite their efficiency, often introduce artifacts due to discontinuities and a lack of bijectivity. More advanced projection techniques, such as those leveraging continuously varying normal fields [12, 46], have been introduced to establish smoother correspondences between neighboring surfaces. However, these methods remain susceptible to discontinuities, making them unreliable for applications requiring bijectivity, such as finite element simulations or large-scale dataset processing.

Recent work has combined projection techniques with bijective operators to improve attribute transfer. The linear shell model [24] established a structured domain for continuous bijective projections, while subsequent high-order shells [39] introduced globally smooth projection fields, improving the accuracy and smoothness of transferring attributes such as vector fields and displacements.

Parameterization-based methods. An alternative approach involves mapping both source and target surfaces to a common parameterization domain. This strategy enables attribute transfer through composition of source-to-domain and domain-to-target mappings. Various works have explored different parameterization domains, including disks [13, 60], planar regions [23, 54], canonical coarse polyhedra [32], and orbifolds [1]. While these approaches allow attribute transfer between geometrically distinct surfaces, ensuring automatic, robust mappings with guaranteed output quality remains an open problem.

Another promising direction involves explicit tracking of bijective correspondences during remeshing operations. Methods such as MAPS-based successive self-parameterization [38] and intrinsic remeshing strategies [37] provide localized solutions to maintain correspondences. However, these techniques are tailored to specific remeshing algorithms and lack general applicability across arbitrary surfaces.

3 Preliminaries

3.1 Function generation

Tensor-product B-splines [57] are extensively employed for approximating the signed distance field (SDF) due to their local support and non-oscillation characteristics [16]. In this paper, we adopt a cubic trivariate tensor-product B-spline function in this paper to parameterize the SDF of input mesh, we impose explicit constraints on both the gradient and the Hessian of the function, ensuring that the resulting field adheres to desired differential properties. This approach allows the space of interest to be partitioned into regularly structured grid cells, with each basis function defined locally.

The standard univariate cubic uniform B-spline basis is expressed as

$$b(t)=\begin{cases} -\frac{1}{6}t^3+t^2-2t+\frac{4}{3}, & t\in[1,2], \\ \frac{1}{2}t^3-t^2+\frac{2}{3}, & t\in[0,1], \\ -\frac{1}{2}t^3-t^2+\frac{2}{3}, & t\in[-1,0], \\ \frac{1}{6}t^3+t^2+2t+\frac{4}{3}, & t\in[-2,-1], \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

Let the side length of a grid cell be denoted as w. For any point $\boldsymbol{v}\in\mathbb{R}^3$ with coordinates $(x,y,z)^\top$, the tensor-product B-spline basis centered at a grid point $\boldsymbol{g}$ is given by

$$B_{\boldsymbol{g}}(\boldsymbol{v})=b\left(\frac{x-x_g}{w}\right)b\left(\frac{y-y_g}{w}\right)b\left(\frac{x-z_g}{w}\right), \tag{3.2}$$

where $\boldsymbol{g}=(x_g,y_g,z_g)^\top$. Thus, any function spanned by the basis functions defined at the grid points can be expressed as

$$f(\boldsymbol{v})=\sum_{\boldsymbol{g}\in\mathcal{G}}\lambda_{\boldsymbol{g}}B_{\boldsymbol{g}}(\boldsymbol{v}), \tag{3.3}$$

where $\mathcal{G}$ denotes the set of all grid points, and $\lambda_{\boldsymbol{g}}$ represents the unknown coefficient to be determined. Since the linear system for solving the coefficients $\Lambda=\{\lambda_{\boldsymbol{g}}\}$ is sparse, it can be efficiently solved using the conjugate gradient method [18].

3.2 Supporting lemmas

In this subsection, we provide several technical lemmas that serve as foundational tools for the proofs in later sections.

Lemma 3.1. *Let $\mathcal{M}\subset\mathbb{R}^3$ be a closed polygonal mesh, and let $f:\mathbb{R}^3\to\mathbb{R}$ be a continuous function that parametrically approximates the SDF of $\mathcal{M}$. Due to the inherent approximation error introduced during the parameterization of a SDF, there always exist some points $\boldsymbol{v}\in\mathcal{M}$ such that $f(\boldsymbol{v})\neq 0$. Therefore, there must exist at least two points $\boldsymbol{v}_1,\boldsymbol{v}_2\in\mathcal{M}$ such that*

$$f(\boldsymbol{v}_1)<0,\quad f(\boldsymbol{v}_2)>0,$$

which implies that the zero level set of f does not coincide with the surface $\mathcal{M}$, and that $\mathcal{M}$ is enclosed between nonzero isosurfaces of f.

Lemma 3.2. *Let $\Omega_{[a,b]}=\{\boldsymbol{v}\mid f(\boldsymbol{v})\in[a,b]\}$. If $\nabla f(\boldsymbol{v})\neq 0$ for all $\boldsymbol{v}\in\Omega_{[a,b]}$, then the level sets defined by $f(\boldsymbol{v})=c$ for $c\in[a,b]$ are homeomorphic, as proposed in* [17].

Lemma 3.3. *Let the endpoints of an edge $\boldsymbol{e}\subset\mathbb{R}^3$ be denoted by $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. A point $\boldsymbol{v}\in\boldsymbol{e}$ can be parameterized as $\boldsymbol{v}(t)=t\boldsymbol{v}_1+(1-t)\boldsymbol{v}_2$, $t\in[0,1]$. Let $f:\mathbb{R}^3\to\mathbb{R}$ be a continuous function, and define the function $g_{\boldsymbol{e}}(t)=f(\boldsymbol{v}(t))$. Then the extrema of f along the edge $\boldsymbol{e}$ occur at the endpoints or at the roots of the derivative $g'_{\boldsymbol{e}}(t)=df(t\boldsymbol{v}_1+(1-t)\boldsymbol{v}_2)/dt$.*

If $g_{\boldsymbol{e}}(t)$ is a polynomial of degree n, then $g'_{\boldsymbol{e}}(t)$ is a polynomial of degree $n-1$, and its roots can be computed as the eigenvalues of the companion matrix $C\in\mathbb{R}^{(n-1)\times(n-1)}$, defined by

$$C=\begin{bmatrix}0&0&\cdots&0&-c_0\\1&0&\cdots&0&-c_1\\0&1&\cdots&0&-c_2\\\vdots&\vdots&\ddots&\vdots&\vdots\\0&0&\cdots&1&-c_{n-2}\end{bmatrix},$$

where $g'_{\boldsymbol{e}}(t)=c_0+c_1t+\cdots+c_{n-2}t^{n-2}+t^{n-1}$. The roots of $g'_{\boldsymbol{e}}(t)$ are precisely the eigenvalues of C [19].

Lemma 3.4. *Let $\Omega\subset\mathbb{R}^n$ be an open domain, and let $\mathcal{F}:\Omega\to\mathbb{R}^n$ be a continuously differentiable vector field such that*

$$\mathcal{F}(\boldsymbol{p})\neq 0,\quad\forall\boldsymbol{p}\in\Omega.$$

Then, for any point $\boldsymbol{p}\in\Omega$, the initial value problem

$$\frac{d\boldsymbol{c}(t)}{dt}=\mathcal{F}\big(\boldsymbol{c}(t)\big),\quad\boldsymbol{c}(t_0)=\boldsymbol{p}$$

admits a unique local solution in Ω. In other words, an integral curve passing through $\boldsymbol{p}$ exists and is unique. This result follows from standard ODE theory; see, e.g. [33].

4 Shell construction

Given a polygonal mesh surface $\mathcal{M}$ embedded in $\mathbb{R}^3$, our goal is to construct a shell that satisfies specific requirements. To begin, we must find an implicit function $f:\mathbb{R}^3\to\mathbb{R}$ that satisfies the following conditions:

1. Tight enclosure: The mesh $\mathcal{M}$ is confined between two isosurfaces, satisfying $\epsilon_1 \le f(\boldsymbol{v}) \le \epsilon_2$ for all points $\boldsymbol{v} \in \mathcal{M}$, where ϵ_1 and ϵ_2 represent the minimum and maximum values of f on $\mathcal{M}$, respectively.

2. Acute angle condition: $\angle(\nabla f, \mathbf{n}_T) < \pi/2$ for all facets $T \in \mathcal{M}$. This is a necessary condition for establishing bijective projection, as proved in Proposition 2.

3. Non-vanishing gradient: For all point $\boldsymbol{v} \in \Omega_{\mathcal{M}}$, the gradient $\nabla f(\boldsymbol{v}) \neq 0$.

The subsequent subsections will provide detailed explanations of the representation and construction of f.

Based on this implicit function f, the implicit shell space $\Omega_{\mathcal{M}}$ is defined as

$$\Omega_{\mathcal{M}} = \Big\{ \boldsymbol{v} \,|\, f(\boldsymbol{v}) \in [\epsilon_1, \epsilon_2], \epsilon_1 = \min_{\boldsymbol{u} \in \mathcal{M}} f(\boldsymbol{u}), \epsilon_2 = \max_{\boldsymbol{u} \in \mathcal{M}} f(\boldsymbol{u}) \Big\}. \tag{4.1}$$

4.1 Objective function

To ensure both topological consistency and geometric fidelity between the shell structure and the input mesh $\mathcal{M}$, the implicit function f is expected to approximate the SDF of $\mathcal{M}$. Accordingly, the first energy term can be formulated as follows:

$$E_{\text{SDF}}(f) = \sum_{\boldsymbol{p} \in P} \big(f(\boldsymbol{p}) - D(p, \mathcal{M}) \big)^2, \tag{4.2}$$

where P is a set of sample points uniformly distributed within the bounding box of the mesh $\mathcal{M}$, and $D(\boldsymbol{p}, \mathcal{M})$ denotes the signed distance from point $\boldsymbol{p}$ to the surface of $\mathcal{M}$.

To uphold the acute angle condition property, it is crucial to guarantee that the gradient direction of the implicit function f at every point on the mesh $\mathcal{M}$ forms an acute angle with the surface normal of the corresponding face. This condition ensures that the level sets of f intersect the surface from the correct direction, thereby preserving the consistency between the zero level set and the input geometry. To this end, we minimize the deviation between the gradient and the face normal over all surface facet, leading to the following energy:

$$E_{\text{angle}} = \sum_{T \in \mathcal{T}} \int_T \| \nabla f - \boldsymbol{n}_T \|^2 ds, \tag{4.3}$$

where $\mathcal{T}$ is the set of facets on $\mathcal{M}$ and $\boldsymbol{n}_T$ denotes the unit normal of each facet T.

To obtain a non-vanishing gradient property, we expect the implicit function f to exhibit desirable differential behavior along the normal direction within the narrow implicit shell region $\Omega_{\mathcal{M}}$. In particular, critical points should be avoided as much as possible within this region. To this end, we directly adopt the singular Hessian term proposed in [61], defined as

$$E_{\text{singularH}} = \int_{\Omega_{\mathcal{M}}} \| \det(\mathbf{H}_f(x) \| dx, \tag{4.4}$$

which encourages the Hessian matrix of f to be singular at every point within $\Omega_{\mathcal{M}}$.

According to Morse theory [17], such a degeneracy constraint effectively eliminates non-essential critical points in the implicit shell region, thereby improving the geometric expressiveness of the implicit function while preserving topological consistency. To validate the effectiveness of the $E_{\text{singularH}}$ term, we perform an exhaustive search method [63] for critical points, as demonstrated in Fig. 2, no critical points are observed within the implicit shells.

To ensure that the gradient aligns with the surface normals, we enforce the condition $\nabla f \cdot \boldsymbol{n}_T = 1$ on each facet T. Directly imposing this as a hard constraint in the optimization process is often impractical. Instead, we incorporate a penalty term that heavily penalizes any deviation from this condition. With a sufficiently large penalty weight, the solution will effectively satisfy the desired normal alignment

$$E_{\text{normal}} = \sum_{T \in \mathcal{T}} \int_T (\nabla f \cdot \boldsymbol{n}_T - 1)^2 ds. \tag{4.5}$$

Therefore, the problem reduces to constructing an implicit function that minimizes the total energy

$$E = E_{\text{SDF}} + \mu(E_{\text{angle}} + E_{\text{singularH}} + \zeta E_{\text{normal}}), \tag{4.6}$$

where the parameter μ serves to balance the influence between the SDF fitting energy and the regularization terms that enforce desirable first and second order differential properties of the implicit function f (set to 0.1 in our experiments). In our implementation, we set the penalty weight $\zeta = 10^5$ to strongly enforce the normal alignment condition while maintaining numerical stability.

Benefiting from the properties of the cubic B-spline function, the above optimization problem can be formulated as a large sparse linear system, which we then solve efficiently. As shown in Fig. 3, the resulting function f produces well-structured level sets, demonstrating that our method effectively captures the SDF.

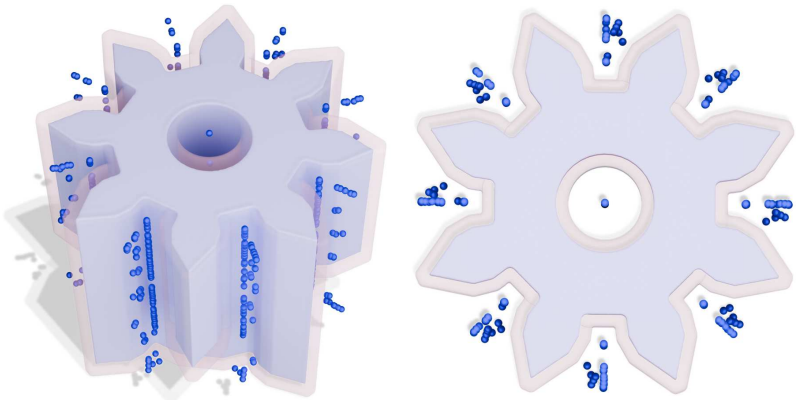

Figure 2: Critical points and the implicit shells for a given polygonal mesh in different views. We use the marching cubes [40] algorithm to visualize the implicit shells.

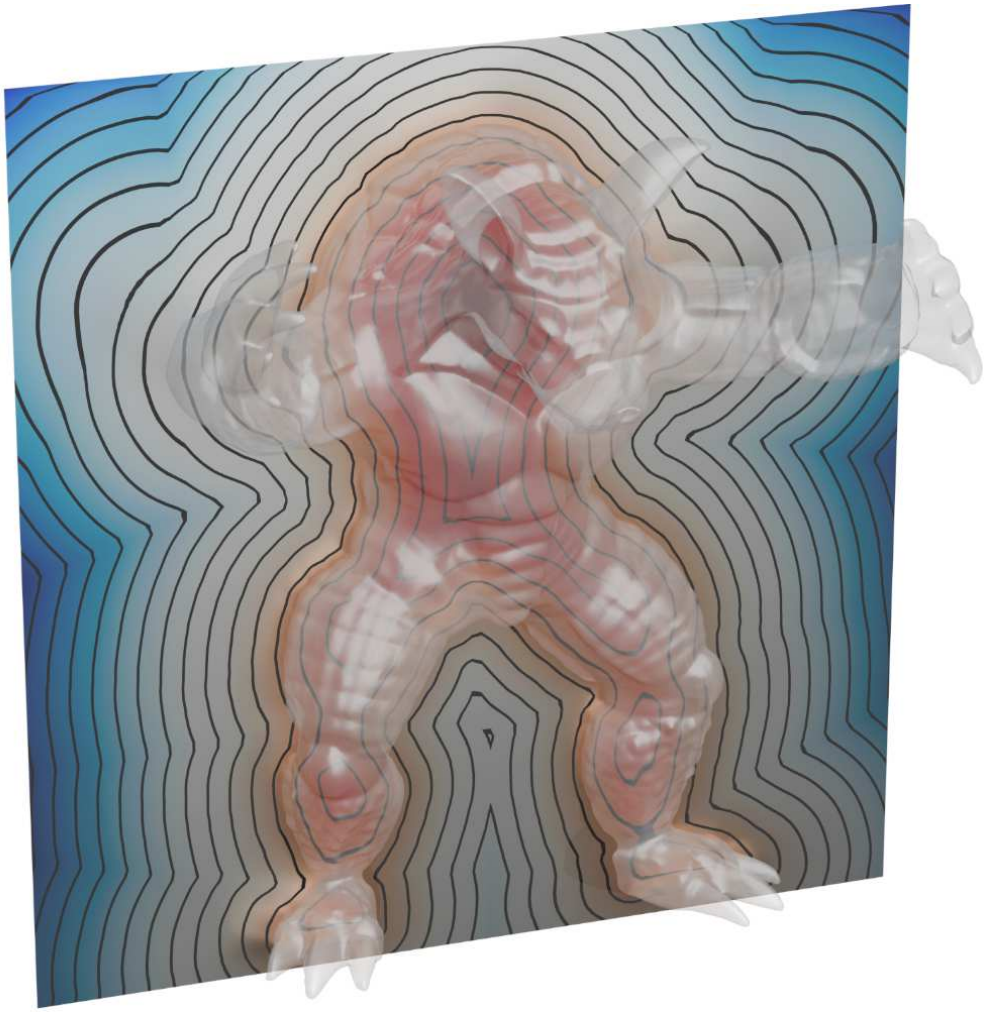

Figure 3: Visualization of the SDF approximated using tensor-product B-splines.

4.2 Bounding surfaces

Having obtained the function f that satisfies the properties previously introduced acute angle condition and non-vanishing gradient, we now describe how to compute the maximum and minimum values of f over the input mesh $\mathcal{M}$. This allows us to identify two isosurfaces that satisfy Tight enclosure. According to Lemma 3.1, the existence of such enclosing isosurfaces is guaranteed.

The local extremum of each facet T must lie within its boundaries rather than within its interior, according to the following proposition.

Proposition 4.1. *The projection of the gradient of the implicit function f onto the facet adheres to the harmonic property, as shown in Fig.* 4.

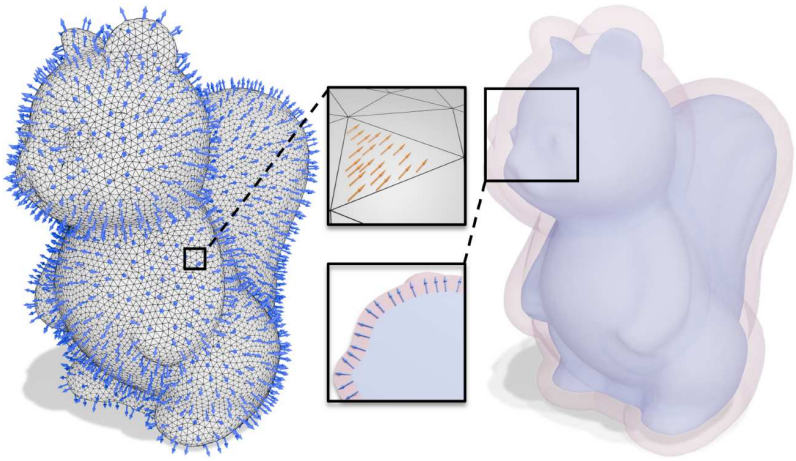

Figure 4: Left: The gradient vectors of the implicit function on a given polygonal surface and the close-window demonstrates that the projection of the vectors onto a facet are consistent. Right: The gradient vector within the shell space. For visualization purpose, the bounding surfaces are both extended eight times inwards and outward, respectively.

Proof. For each facet T on $\mathcal{M}$, the gradient ∇f can be decomposed into its normal and tangential components as follows:

$$\nabla f = (\nabla f \cdot \boldsymbol{n}_T)\boldsymbol{n}_T + \nabla f_T,$$

where ∇f_T represents the tangential projection of ∇f onto the plane of T. Substituting this decomposition into the energy functional in Eq. 4.3 yields

$$E_{\text{angle}} = \sum_{T\in\mathcal{T}} \int_T \|(\nabla f \cdot \boldsymbol{n}_T)\boldsymbol{n}_T + \nabla f_T - \boldsymbol{n}_T\|^2 ds.$$

Expanding the squared term and simplifying

$$E_{\text{angle}} = \sum_{T\in\mathcal{T}} \int_T (\nabla f \cdot \boldsymbol{n}_T - 1)^2 ds + \sum_{T\in\mathcal{T}} \int_T \|\nabla f_T\|^2 ds.$$

Since we incorporate the penalty term in Eq. (4.6), to enforce the condition $\nabla f \cdot \boldsymbol{n}_T = 1$, the first term in E_{angle} becomes negligible. Consequently, we have $E_{\text{angle}} \approx \int_T \|\nabla f_T\|^2 ds$, which represents the Dirichlet energy. This implies that the tangential projection of ∇f onto each facet exhibits the harmonic property. □

Therefore, the maximum and minimum values of f over the domain $\mathcal{M}$ must occur on the mesh edges of $\mathcal{M}$. According the Lemma 3.3, for any given edge $\boldsymbol{e}$, the extrema of f along $\boldsymbol{e}$ can be efficiently computed. Consequently, the global extrema of f over the all edges are obtained as follow:

$$\epsilon_1 = \min_{\boldsymbol{e}\in\mathcal{M}} \{ \min_{v\in\boldsymbol{e}} \{f(v)\}\}, \quad \epsilon_2 = \max_{\boldsymbol{e}\in\mathcal{M}} \{ \max_{v\in\boldsymbol{e}} \{f(v)\}\}, \tag{4.7}$$

where the ϵ_1 and ϵ_2 denote the minimum and maximum values, and define the implicit bounding surface of $\mathcal{M}$.

Remark 4.1. To optimize computational efficiency, we precompute the intersections between the grid cells of B-spline and the input mesh surface to generate a refined mesh $\mathcal{M}_{\text{refined}}$. This preprocessing step guarantees that every edge $\boldsymbol{e} \in \mathcal{M}_{\text{refined}}$, is strictly confined within a single grid cell. Consequently, the sparsity pattern of the resulting system matrix is constrained to a fixed per-voxel stencil of $4\times4\times4$ non-zero coefficients.

4.3 Bijective projection

Previously, we computed the two isosurfaces of the implicit function corresponding to ϵ_1 and ϵ_2, thereby obtaining the implicit shell space. In this subsection, we focus on proving the existence of a bijective projection within the shell space. This projection plays a crucial role in subsequent applications.

Proposition 4.2. *Let $\Omega_{\mathcal{M}_A}$ be a cubic B-spline shell that encloses the mesh surface $\mathcal{M}_A$, defined by a cubic B-spline function f. Consider an oriented, manifold, and watertight mesh $\mathcal{M}_B$ fully contained within $\Omega_{\mathcal{M}_A}$. If the following condition hold:*

$$\boldsymbol{n}_B \cdot \nabla f(\boldsymbol{p}_B) > 0, \quad \forall \boldsymbol{p}_B \in \mathcal{M}_B, \tag{4.8}$$

where $\boldsymbol{n}_B$ denote the unit normal at the point $\boldsymbol{p}_B$, then the projection $\mathcal{P}$ induced by the gradient field ∇f establishes a bijection between $\mathcal{M}_A$ and $\mathcal{M}_B$.

Proof. We prove that for any point $\boldsymbol{p}_B \in \mathcal{M}_B$, there exists a unique corresponding point $\boldsymbol{p}_A \in \mathcal{M}_A$ such that $\mathcal{P}(\boldsymbol{p}_A) = \boldsymbol{p}_B$.

Assume, for the sake of contradiction, that there exist two or more distinct points $\boldsymbol{p}_{Ai}$ on $\mathcal{M}_A$ (where $i > 1$) such that

$$\mathcal{P}(\boldsymbol{p}_{Ai}) = \boldsymbol{p}_B.$$

This would lead to two possible cases, as illustrated in Fig. 5.

Since the gradient field ∇f is non-zero within the shell, Lemma 3.4 ensures that through every point in $\Omega_{\mathcal{M}}$, there exists a unique integral curve following the gradient field.

If two or more distinct integral curves intersect at a junction, this contradicts Lemma 3.4, which guarantees uniqueness of the integral curves. If a single integral curve passes through multiple points $\boldsymbol{p}_{Ai}$, then by the continuity of the gradient field, there must exist a point $\boldsymbol{\xi}_A \in \mathcal{M}_A$ where the normal satisfies $\boldsymbol{n}_{\xi_A} \cdot \nabla f(\xi_A) \leq 0$, which contradicts acute angle condition.

Since the integral curves of the gradient field are reversible within the shell domain, the same holds for the projection $\mathcal{P}^{-1}$. Therefore, the gradient-aligned mapping $\mathcal{P}$ is bijective. □

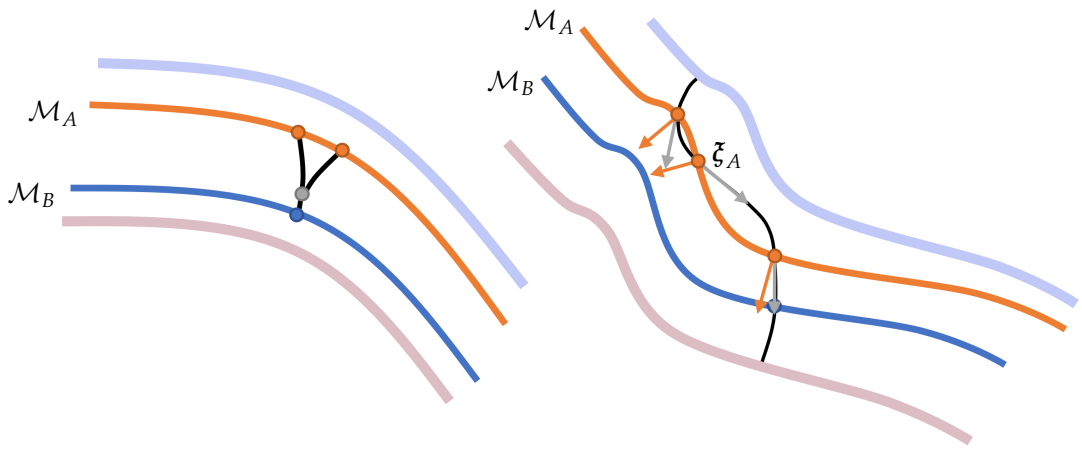

Figure 5: Two possible scenarios arise when projecting multiple points from $\mathcal{M}_A$ onto a single point on $\mathcal{M}_B$. The orange arrow indicates the normal direction, while the gray arrow represents the gradient direction.

5 Results

5.1 Implementation

Our experiments and computational tasks were conducted on an Intel i7-14700K @ 3.40GHz processor with 32GB of RAM. The algorithm is implemented in C++ utilizing the Eigen [15] library for efficient linear algebra operations and the Libigl [22] library for fundamental geometric computations. In addition to these core libraries, we employ the SymEngine [58] library for performing advanced symbolic mathematics and numerical evaluations when converting polygonal polynomials into their corresponding companion matrices, as elaborated in Section 4.2.

5.2 Robustness and validation

Since our algorithm relies on the normals of each facet in the input polygonal mesh, it requires that the input models be orientable. Furthermore, to ensure the accurate transfer of geometric properties, our algorithm requires the input models to be intersection-free and manifold.

To evaluate the robustness of our method, we tested it on the ABC dataset [30] and the Thingi10K dataset [64]. Fig. 6 shows several representative examples of challenging models from both datasets, featuring intricate geometric and topological structures. In all cases, our algorithm successfully generates well-defined implicit shells.

A key tunable parameter in our algorithm is grid cell size w within which B-splines are defined. During implementation, we normalize the model to unit size and set $w{=}1/\gamma$, where $\gamma{>}0$ represents the resolution of the structured grid. This resolution directly influences both the shell thickness and the level of geometric detail captured by the bounding

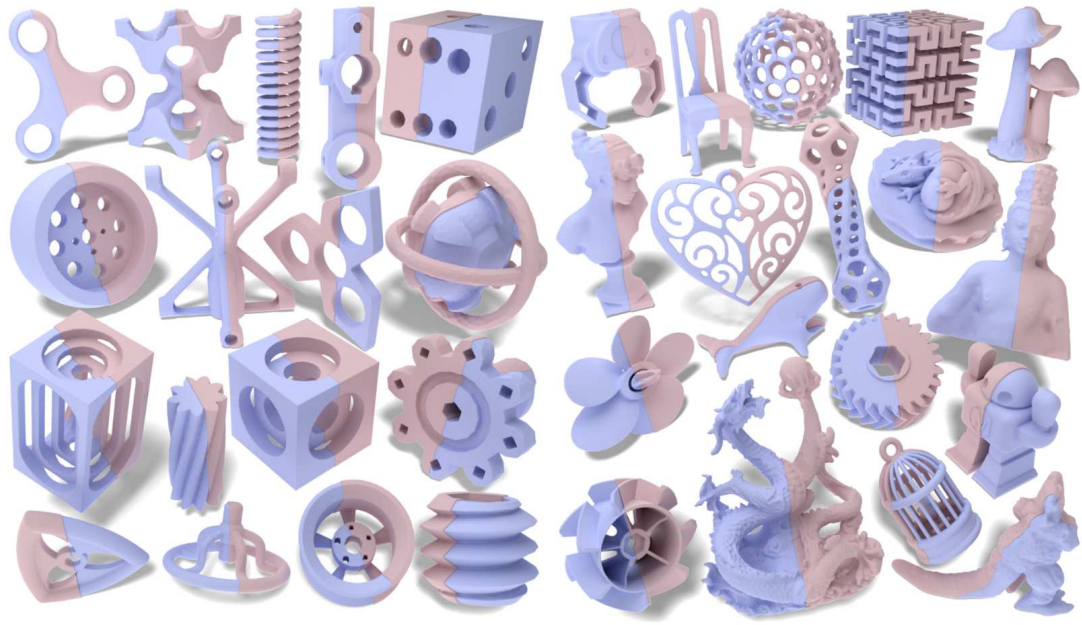

Figure 6: Gallery of shells constructed around models from ABC [30] and Thingi10K [64], with the inner and outer bounding surfaces highlighted in blue and red, respectively.

surfaces. To quantitatively analyze the impact of γ, we define the thickness of the shell structure as

$$d=\epsilon_2-\epsilon_1. \tag{5.1}$$

Additionally, to eliminate numerical errors that arise when computing function values for the inner and outer bounding surfaces, which may prevent the shell from strictly enclosing the input mesh surface, we refine the values of bounding surfaces using

$$\epsilon_1=\frac{\lfloor \epsilon_1\times\tau\rfloor}{\tau},\quad \epsilon_2=\frac{\lceil \epsilon_2\times\tau\rceil}{\tau}.$$

Here $\lfloor\cdot\rfloor$ and $\lceil\cdot\rceil$ represent the floor and ceiling functions, respectively. We set τ to 10^{10} in the experiments.

In Fig. 7, we illustrate the impact of the parameter γ on the shell structure. A larger γ results in shells that better preserve the geometric features of the input model, whereas a smaller γ leads to a loss of geometric details. Fig. 8 presents a parametric sensitivity analysis of γ, highlighting its dual impact on geometric fidelity (shell thickness) and computational efficiency. This analysis is conducted through a comparative evaluation of two benchmark datasets. The assessment framework utilizes curated subsets (100 models per dataset) with controlled topological complexity: Thingi10K models contain triangle counts ranging from 10K to 300K, whereas ABC dataset meshes range from 2K to 4K. All reported metrics are computed using statistically validated cross-model averages derived from curated subsets of these datasets.

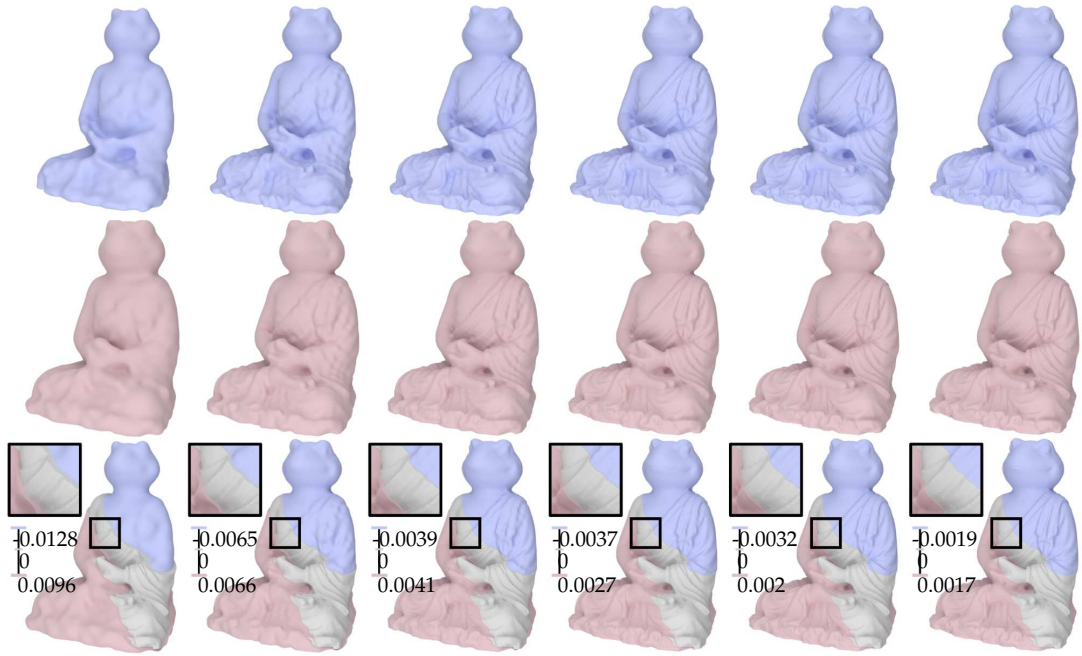

Figure 7: For the input model "Buddha Frog" from Thingi10K dataset, we visualize the inner (top row) and the outer (middle row) bounding surfaces. From left to right: The parameter γ varies as 30,50,70,90,110,130. To further analyze the behavior of the implicit function f, we highlight its extrema using a slice-cut visualization, as shown in the bottom row. When γ is relatively large, our method is expressive enough to closely approximate the original mesh. For example, when $\gamma = 130$, the maximum absolute value of f is 1.9×10^{-3}, indicating an almost negligible deviation from the original base surface.

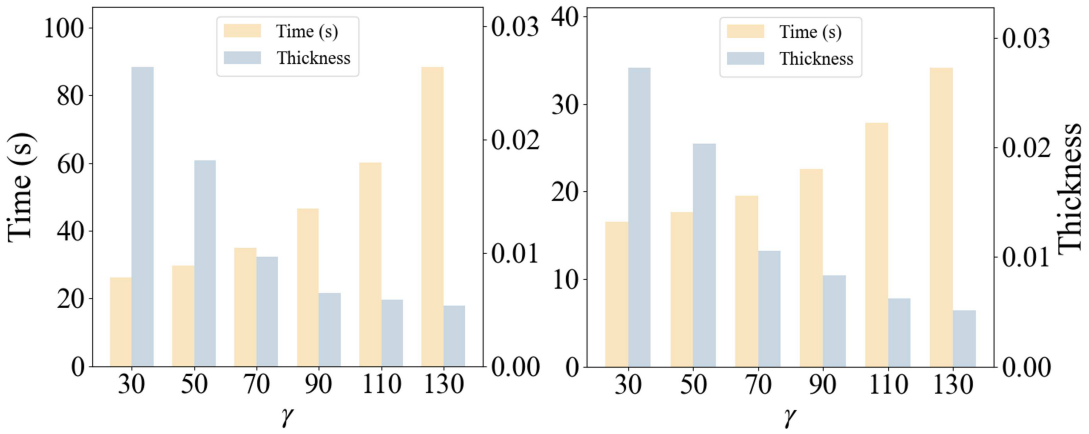

Figure 8: The impact of γ variation on temporal performance and geometric fidelity across benchmark datasets (left: Thingi10K, right: ABC).

To ensure the validity of shell structures generated by our algorithm, we establish a verification framework to test strict geometric containment through statistically rigorous sampling analysis. For an input manifold mesh $\mathcal{M}$ with verified orientation con-

sistency and absence of self-intersections, we implement Poisson-disk sampling [9] to obtain a quasi-uniformly distributed point set $P_s=\{\mathbf{p}_i\}_{i=1}^N$, where we set N to 100K in the experiments. The containment criterion for point p_i is formalized as

$$\mathcal{C}(\mathbf{p}_i)=\begin{cases}1, & \text{if} \quad f(\mathbf{p}_i)\in[\epsilon_1,\epsilon_2],\\ 0, & \text{otherwise.}\end{cases}$$

where f denotes the approximated signed distance. The global containment metric is then computed as

$$\mathcal{W}=\frac{1}{N}\sum_{i=1}^{N}\mathcal{C}(\mathbf{p}_i).$$

Our verification pipeline processes 500 certified models from both datasets under six discrete parameters ($\gamma=30,50,70,90,110,130$). Quantitative analysis reveals strict geometric containment ($\mathcal{W}=1$) in 99.8% of test cases (499/500 models). The residual 0.2% of cases exhibit near-unity containment metrics ($1-2.10\times10^{-4}\leq\mathcal{W}<1$).

5.3 Comparisons

We conduct a comprehensive comparative study with linear shell [24], the state-of-the-art shell generation method. While the baseline approach successfully produce the shell space using the prism-based method, it suffers from a limitation: discontinuous vector fields at the boundaries of prisms, which lead to distortion in attribute transfer when the remeshed surface crosses these regions.

As shown in Figs. 1 and 9, we evaluate the attribute transfer capabilities of our method in comparison to linear shell through a benchmarking process. The workflow begins with the remeshing of the input mesh, generating new meshes within all shells. The heat method [10] is then applied to these remeshed surfaces to compute geodesic distance fields from specified source points. The computed solutions are subsequently mapped back to the original geometry using the bijective projection operators, which will be introduced in Section 6.2. A significant limitation arises in the baseline method due to the prism-based vector field definitions: attribute transfer across prism boundaries leads to noticeable distortions. This issue stems from the discontinuous nature of piecewise-defined vector fields in explicit shell constructions.

5.4 Ablation study

Our algorithm involves the key parameter μ, which balances the SDF approximation and the angle condition. In this ablation study, we focus on the effect of varying μ on the bijectivity of our attribute transfer process. To evaluate bijectivity, we analyze the transfer performance of the heat method geodesic distance field [10]. For visualization, we display histograms of absolute errors (threshold 0.03) to highlight regions where bijectivity is compromised, as illustrated in Fig. 10.

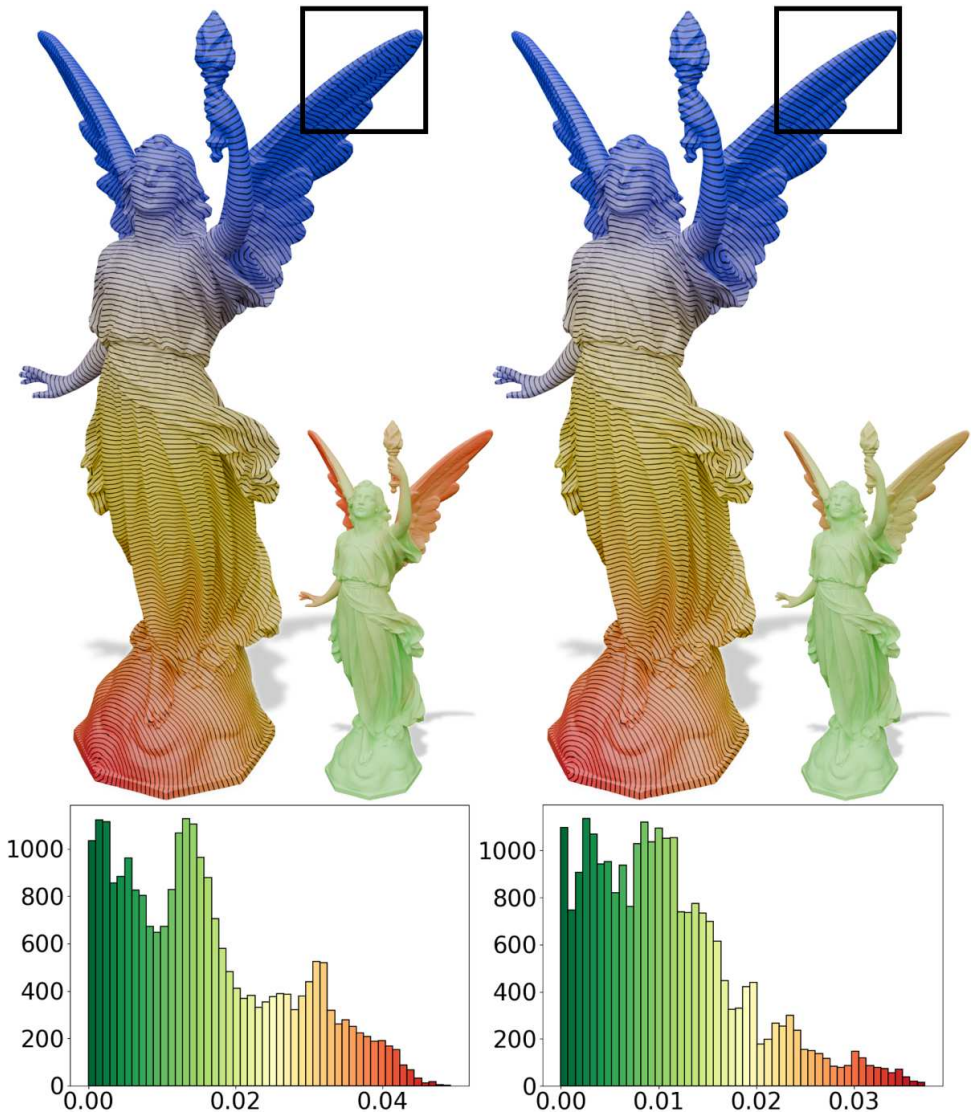

Figure 9: Comparison with linear shell in evaluating induced errors during geodesic distance field transfer operations from refined meshes to low-quality meshes.

Our experimental results reveal that for a small value of μ, regions with significant normal variation are not sufficiently constrained by the acute angle condition. As a consequence, the local gradient directions deviate from the requirements for bijectivity. In contrast, when μ is set too high, the shell loses its geometric fidelity, which impairs bijectivity and leads to minor fluctuations in the geodesic distance isolines. These findings demonstrate that proper tuning of μ is essential for preserving bijectivity and ensuring accurate attribute transfer.

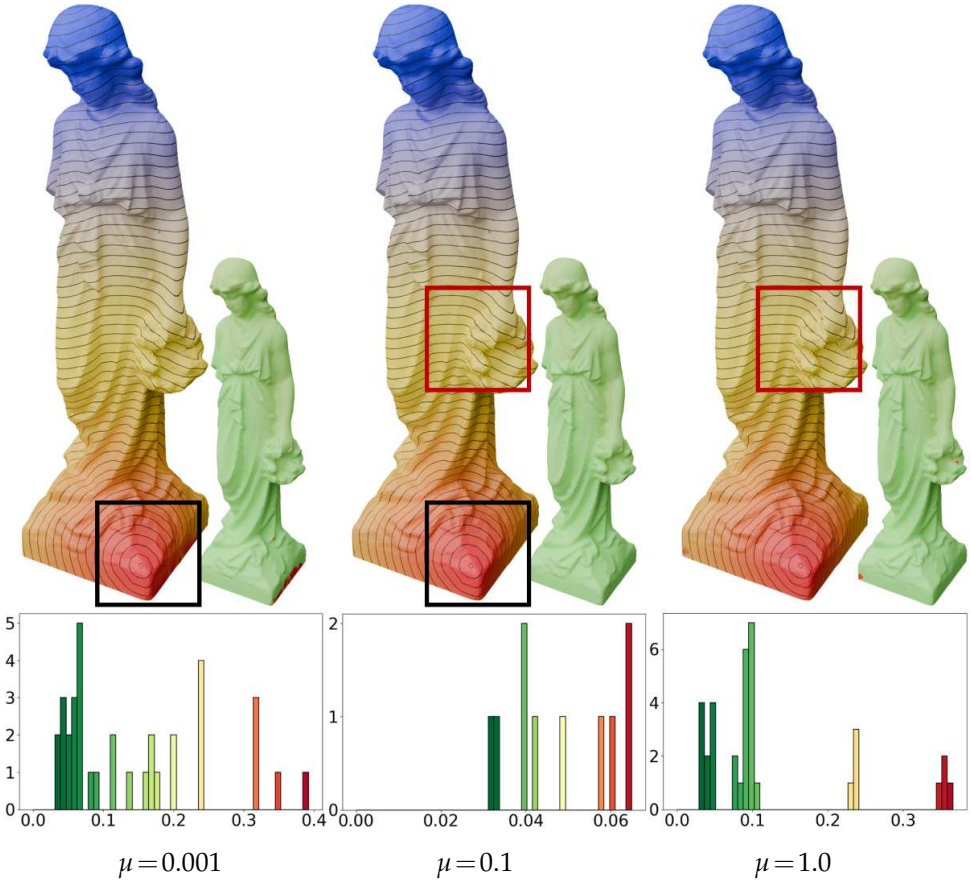

Figure 10: Impact of μ on attribute transfer (To visualize the effect, we set $\gamma = 20$), and we visualize absolute errors greater than 0.03 in histograms.

6 Applications

The geometrically encoded shells proposed in this paper inherently retain the desirable properties of implicit representations, allowing for seamless surface proximity verification via spatial point sampling. This mathematical formulation simultaneously functions as a surface approximation framework and a topological constraint mechanism, thereby providing a robust global regularization paradigm for mesh remeshing. Moreover, the constructed shell space features a bijective mapping structure, enabling efficient cross-mesh attribute transfer while preserving geometric consistency between the two surfaces within the shell space.

6.1 Mesh simplification

Quadric error metric (QEM) [14] is a well-known polygonal mesh simplification technique that employs an iterative edge collapse process guided by a priority queue. This

priority queue prioritizes edges based on quadratic error metrics, enabling the method to maintain geometric accuracy while reducing mesh complexity. However, as demonstrated in Fig. 11(b), the original QEM algorithm often neglects broader geometric considerations, potentially leading to unwanted topological artifacts such as intersections or improper triangle orientations.

To overcome these limitations, we introduce an enhanced version of QEM that integrates shell constraints into the decision-making process.

Our method constrains the mesh simplification process using an implicit shell generated from the input surface. Specifically, two key conditions are enforced to ensure geometric fidelity and support bijective correspondence between the simplified and original meshes: (1) Tight enclosure: The simplified mesh must remain strictly within the implicit shell region. During edge collapse, if any part of a newly formed triangle lies outside the shell(determined via point-in-shell queries) the operation is rejected. (2) Acute angle condition: The normals of newly generated triangles must form acute angles with the local gradient of the implicit function. Together, these constraints prevent self-intersections and excessive distortion, enabling high-fidelity simplification and facilitating consistent attribute transfer between the original and simplified surfaces.

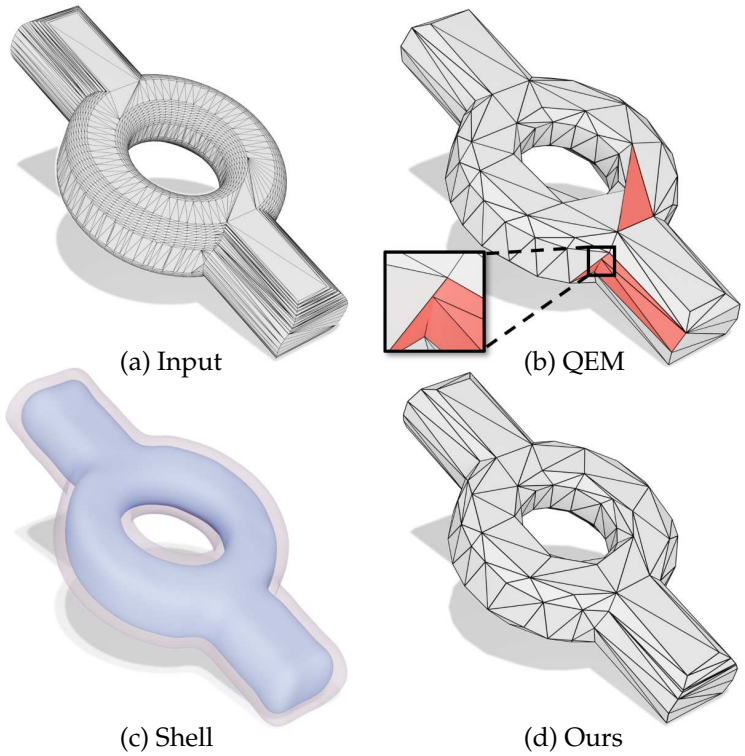

Figure 11: We simplify an input triangle mesh (a) from 6494 triangles to 300. While QEM [14] successfully reduces the triangle count (b), it introduces self-intersections (highlighted in red). To address this, we construct shell (c) around the input mesh and use them as constraints during simplification. Our approach (d) ensures a well-preserved geometric structure while preventing undesirable geometric artifacts.

As shown in Fig. 11(d), ours approach significantly reduces the likelihood of generating topologically inconsistent mesh structures while still achieving a high level of geometric preservation. Compared to traditional QEM methods, our algorithm demonstrates improved robustness in handling complex geometries and adhering to application-specific constraints.

6.2 Attribute transfer

Our cross-model attribute transfer framework establishes continuous correspondences through gradient-driven projection. When utilizing the CBS for attribute transfer, the source surface $\mathcal{M}_s \subset \mathbb{R}^3$, endowed with attributes A_s, and the target surface $\mathcal{M}_t \subset \mathbb{R}^3$ must satisfy Proposition 4.2 within their shared embedding shell. In the following applications, we set the parameter $\gamma = 100$.

Previous geometric constraint ensures that for any point $\mathbf{q} \in \mathcal{M}_s$, projecting along the integral curve of the gradient at $\mathbf{q}$, denoted as $\nabla f(\mathbf{q})$, will result in a unique corresponding point $\mathbf{p} \in \mathcal{M}_t$.

This process yields the attribute transfer operator

$$\pi : A_s(\mathbf{q}) \ \mapsto \ A_t(\mathbf{p}),$$

which maps the attribute A_s from the source surface to its corresponding counterpart A_t of the target surface.

This attribute transfer operator serves as the fundamental building block powering all subsequent applications. As illustrated in Fig. 12, our shell-based computational framework leverages the attribute transfer operator to facilitate bidirectional field mapping across two mesh surfaces within implicit shells. This operator ensures seamless transfer of both vector fields and cross-field, enabling smooth and consistent data propagation.

Proxy. Building upon the PDE-solving paradigm established in previous works [24, 39], we develop a unified computational pipeline for constructing proxy domains that facilitate stable PDE solutions on low-quality meshes as shown in Fig. 13. Given an input mesh surface $\mathcal{M}_{\text{orig}}$ with poor quality, our pipeline follows four principled stages: (1) The surface $\mathcal{M}_{\text{orig}}$ undergoes topology-preserving remeshing via our implicit shell construction, yielding a well-defined proxy mesh $\mathcal{M}_{\text{proxy}}$. (2) Boundary conditions are mapped onto $\mathcal{M}_{\text{proxy}}$ using the attribute transfer operator π. (3) A conforming tetrahedral mesh, generated by TetGen [53], serves as the computational domain for simulation of nonlinear elasticity. (4) The computed solution A_s is transferred back to the original geometry through the attribute transfer operator π. Our framework natively supports solving intrinsic surface PDEs through its geometrically consistent discretization scheme. Another representative example is that we implement the heat method [10] for geodesic distance computation, providing enhanced robustness. As demonstrated in Fig. 14, this approach effectively handles meshes with low-quality triangulations that would otherwise lead to significant inaccuracies and deviations.

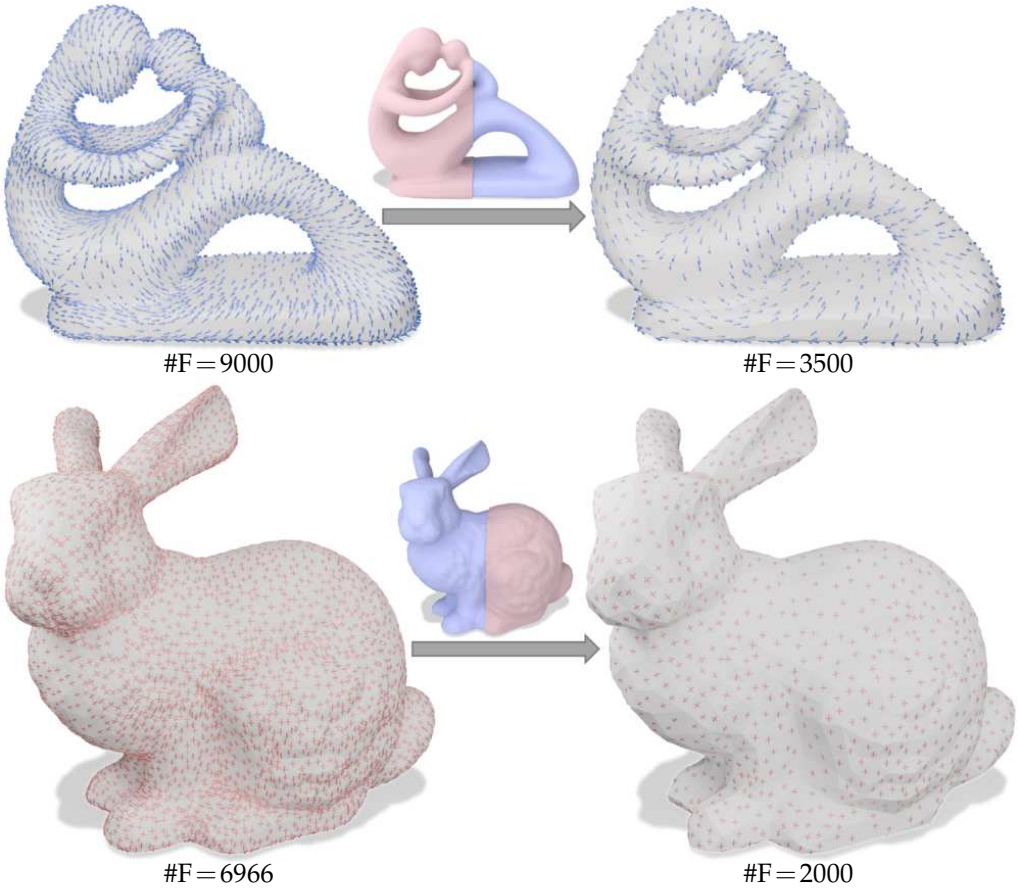

Figure 12: Top: Tangential vector field from the "Fertility" model, projected onto the simplified surface using the attribute transfer operator. Bottom: Cross-field computed on the "bunny" model and transferred to a simplified model within our shell. Both fields are computed following [29].

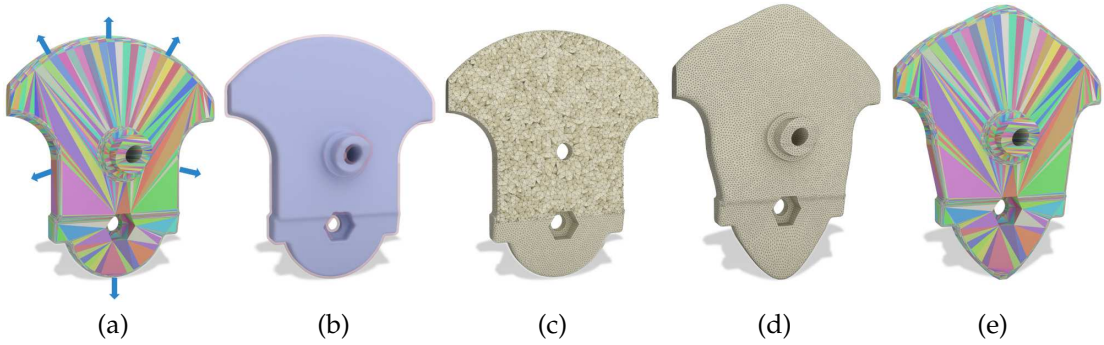

Figure 13: Given a low-quality triangle mesh with boundary conditions (a), we first perform remeshing within our implicit shell (b), ensuring a bijective correspondence between the remeshed geometry and the input mesh while transfering the boundary conditions. A tetrahedral mesh is generated using TetGen [53], as shown in the cross-sectional view (c). Next, we compute nonlinear plastic deformation on this volumetric representation. Finally, the solution is transferred back to the original mesh, yielding the final deformed result (e).

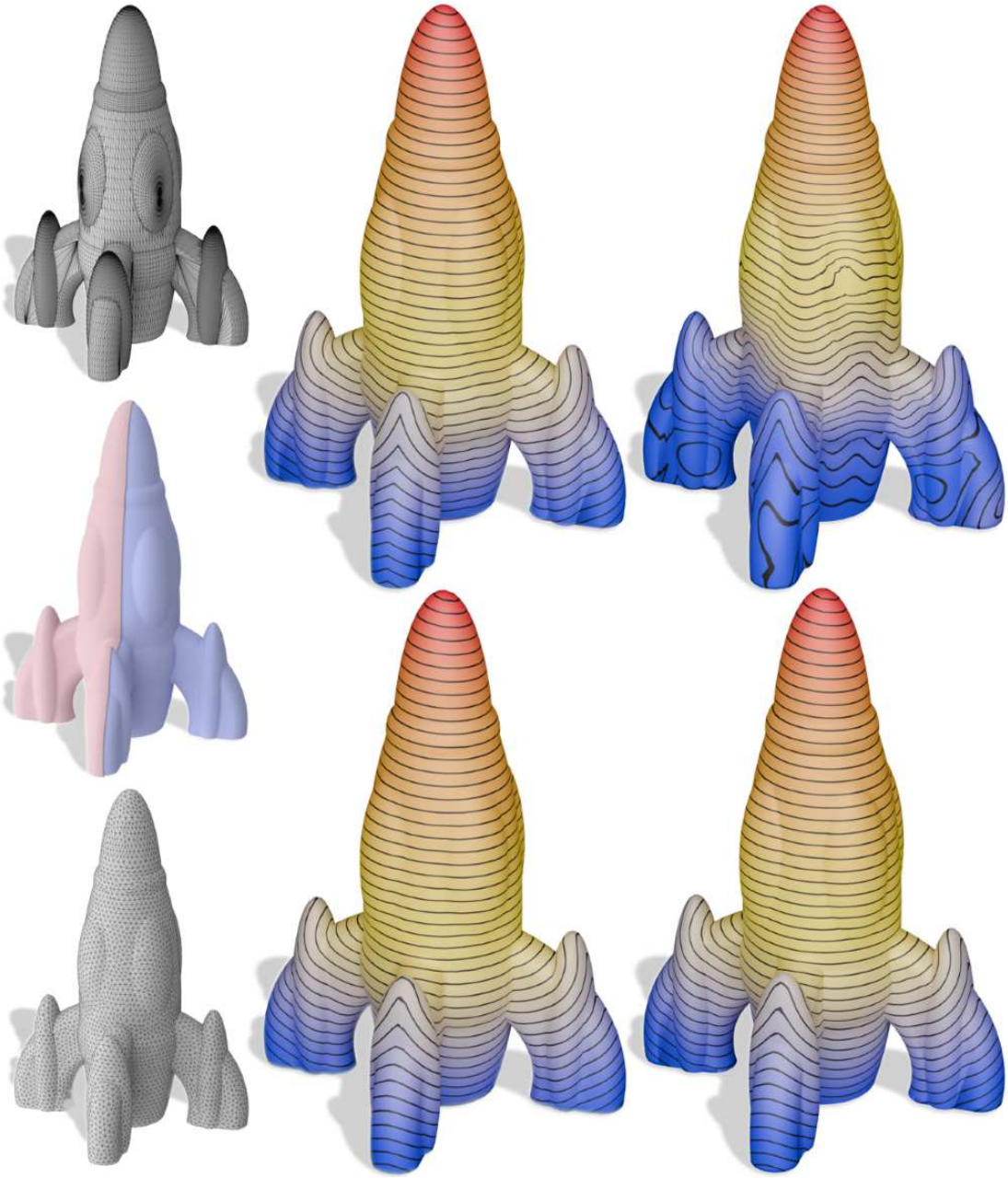

Figure 14: When applying the heat method [10] to a low-quality input mesh (top left), the resulting distance field (top right) is highly inaccurate and deviates significantly from the exact discrete geodesic distance [44] (top middle). To improve accuracy, we first generate a high-quality remeshing of the input model within our shell (middle left) and obtain a refined mesh (bottom left). The heat method is then applied to this improved mesh (bottom middle), and the computed distance field is mapped back to the original input mesh using our attribute transfer operator (bottom right). This refinement ensures that the transferred distance field better aligns with the exact geodesic computation.

Boolean operations. Constructive solid geometry (CSG) provides a procedural modeling paradigm through Boolean operations (union, intersection, difference) applied to primitives. While this approach maintains parametric control, Boolean operations inevitably introduce geometric singularities, particularly near intersection contours, that compromise computational robustness. Our framework resolves this limitation through a three-stage topology-aware processing pipeline: (1) For CSG nodes $\mathcal{N}_1$,$\mathcal{N}_2$, compute the Boolean model $\mathcal{M}_{\text{Bool}}$ while preserving their respective material attributes $\{A_1, A_2\}$. (2) Construct the implicit shell using our algorithm, followed by topology-preserving simplification within the shell. (3) Leverage the shell's attribute transfer operator to establish inter-node property correspondences. As demonstrated in Fig. 15 through mesh union operations, our method successfully maintains exact geometric correspondence between input surfaces, as evidenced by consistent color transfer across Boolean boundaries.

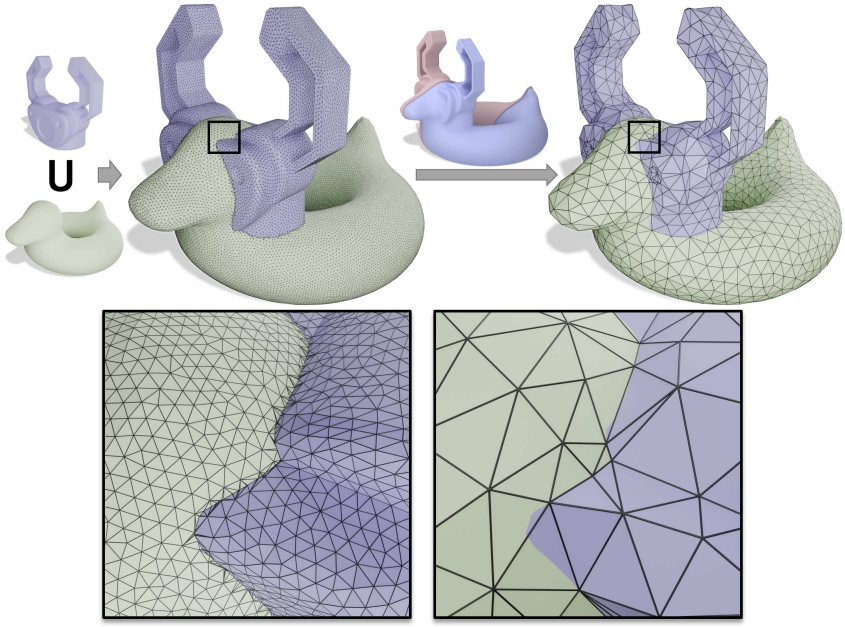

Figure 15: The union of two input models with distinct colors undergoes remeshing within our shell, and the resulting low-poly mesh effectively preserves the original color information.

7 Conclusion

This study presents a novel framework for cubic implicit shell representations, addressing key challenges in polygonal surface processing. By leveraging cubic B-spline functions and enforcing gradient constraints, the proposed method achieves both high precision and computational efficiency. The experimental results validate the effectiveness of

this approach in attribute transfer and mesh editing tasks, demonstrating its potential for applications in a wide range of domains.

Limitations. Our algorithm exhibits inefficiencies when processing models with extremely thin structures, highly curved regions, or narrow gaps. In these cases, achieving an accurate representation requires increasing the resolution of the structured grid cells, which, in turn, significantly escalates computational costs. Due to the limitations of B-spline functions, it is challenging to ensure that the generated shell can strictly enclose the input model with 100% accuracy.

Furthermore, for models containing self-intersections or other geometric artifacts, our method is capable of generating shells. However, it fails to ensure the homeomorphism property within the shell space, as illustrated in Fig. 16. This limitation arises because topological inconsistencies between the inner and outer bounding surfaces can disrupt the bijective correspondence, leading to potential issues in downstream applications that rely on a well-defined shell structure.

Future works. To overcome the aforementioned limitations, several promising directions for future research can be explored:

1. Improving efficiency for extreme cases: Develop advanced algorithms or optimization techniques to handle extremely thin structures and narrow gaps with higher efficiency, reducing computational costs while maintaining precision.
2. Handling self-intersecting and non-manifold models: Further investigate methods to handle self-intersecting and non-manifold models without compromising the homeomorphism property between inner and outer bounding surfaces, ensuring topological consistency.
3. Octree-based spatial partitioning: Implement octree-based space division to reduce computational complexity and accelerate processing, particularly for sparse regions.

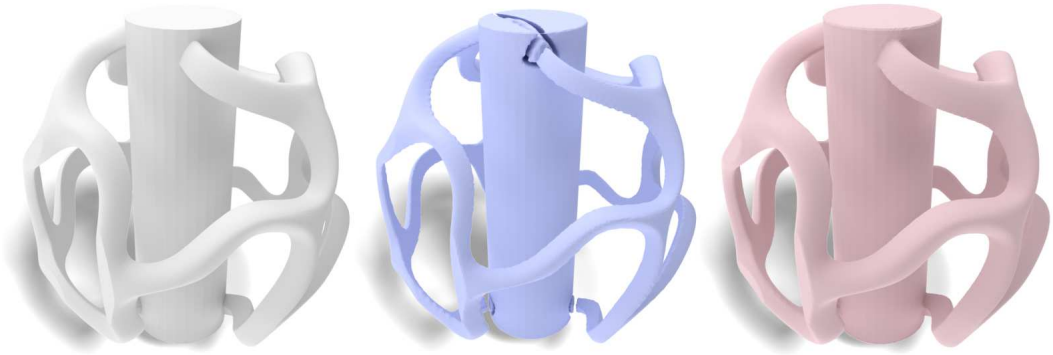

Figure 16: An example of a mesh composed of two intersecting sub-meshes is shown on the left. Our algorithm successfully generates the inner (middle) and outer (right) bounding surfaces. However, due to the topological discrepancy between the two surfaces, a bijective projection in the shell space cannot be guaranteed.

By addressing these areas, future advancements could significantly enhance the robustness and practicality of the proposed framework, making it more suitable for a wider range of applications in computational geometry and related fields.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions.

This work was supported by the National Key R&D Program of China (Grant No. 2022YFB3303200) and by the National Natural Science Foundation of China (Grant Nos. U23A20312, 62272277, and 62472258).

References

[1] N. Aigerman, S. Z. Kovalsky, and Y. Lipman, *Spherical orbifold tutte embeddings*, ACM Trans. Graph., 36:90, 2017.

[2] C. L. Bajaj, G. Xu, R. J. Holt, and A. N. Netravali, *Hierarchical multiresolution reconstruction of shell surfaces*, Comput. Aided Geom. Des., 19:89–112, 2002.

[3] A. Belhachmi, A. Benabbou, and B. Mourrain, *A spline-based regularized method for the reconstruction of complex geological models*, Math. Geosci., 57:89–114, 202.

[4] S. Calderon and T. Boubekeur, *Bounding proxies for shape approximation*, ACM Trans. Graph., 36(4):57, 2017.

[5] H. Chen, B. Miller, and I. Gkioulekas, *3D reconstruction with fast dipole sums*, ACM Trans. Graph., 43(6):192, 2024.

[6] L. Chen, B. Li, and R. de Borst, *The use of powell-sabin B-splines in a higher-order phase-field model for crack kinking*, Comput. Mech., 67:127–137, 2021.

[7] Y. Chen, T. Xie, C. Yuksel, D. Kaufman, Y. Yang, C. Jiang, and M. Li, *Multi-layer thick shells*, in: ACM SIGGRAPH 2023 Conference Proceedings, SIGGRAPH '23, ACM, 25, 2023.

[8] B. O. Community, *Blender – a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, 2018. `https://docs.blender.org/manual/en/latest/copyright.html`

[9] M. Corsini, P. Cignoni, and R. Scopigno, *Efficient and flexible sampling with blue noise properties of triangular meshes*, IEEE Trans. Vis. Comput. Graphics, 18:914–924, 2012.

[10] K. Crane, C. Weischedel, and M. Wardetzky, *Geodesics in heat: A new approach to computing distance based on heat flow*, ACM Trans. Graph., 32(5):152, 2013.

[11] Z. Deng, H. Xiao, Y. Lang, H. Feng, and J. Zhang, *Multi-scale hash encoding based neural geometry representation*, Comput. Vis. Media, 10:453–470, 2024.

[12] D. Ezuz, J. Solomon, and M. Ben-Chen, *Reversible harmonic maps between discrete surfaces*, ACM Trans. Graph., 38(2):15, 2019.

[13] M. S. Floater, *Parametrization and smooth approximation of surface triangulations*, Comput. Aided Geom. Des., 14:231–250, 1997.

[14] M. Garland and P. S. Heckbert, *Surface simplification using quadric error metrics*, in: Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., 209–216, 1997.

[15] G. Guennebaud et al., *Eigen: A C++ template library for linear algebra: Matrices, vectors, numerical solvers, and related algorithms*, 2010. `http://eigen.tuxfamily.org`

[16] C. A. Hall and W. Meyer, *Optimal error bounds for cubic spline interpolation*, J. Approx. Theory, 16:105–122, 1976.
[17] J. Hart, *Morse theory for computer graphics*, 1997. `https://www.researchgate.net/publication/2725761`
[18] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Stand., 49(6):409–436, 1952.
[19] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 1985.
[20] K. Hu, D.-M. Yan, D. Bommes, P. Alliez, and B. Benes, *Error-bounded and feature preserving surface remeshing with minimal angle improvement*, IEEE Trans. Vis. Comput. Graphics, 23:2560–2573, 2017.
[21] J. Huang, X. Liu, H. Jiang, Q. Wang, and H. Bao, *Gradient-based shell generation and deformation*, Computer Animation and Virtual Worlds, 18:301–309, 2007.
[22] A. Jacobson et al., *libigl: A simple C++ geometry processing library*, 2018. `https://libigl.github.io/`
[23] Z. Jiang, S. Schaefer, and D. Panozzo, *Simplicial complex augmentation framework for bijective maps*, ACM Trans. Graph., 36(6):186, 2017.
[24] Z. Jiang, T. Schneider, D. Zorin, and D. Panozzo, *Bijective projection in a shell*, ACM Trans. Graph., 39(6):247, 2020.
[25] Z. Jiang, Z. Zhang, Y. Hu, T. Schneider, D. Zorin, and D. Panozzo, *Bijective and coarse high-order tetrahedral meshes*, ACM Trans. Graph., 40(4):157, 2021.
[26] M. Kazhdan, M. Bolitho, and H. Hoppe, *Poisson surface reconstruction*, in: Proceedings of the Fourth Eurographics Symposium on Geometry Processing, SGP '06, Eurographics Association, 61–70, 2006.
[27] M. Kazhdan, M. Chuang, S. Rusinkiewicz, and H. Hoppe, *Poisson surface reconstruction with envelope constraints*, Comput. Graph. Forum, 39(5):173–182, 2020.
[28] L. Kharevych, P. Mullen, H. Owhadi, and M. Desbrun, *Numerical coarsening of inhomogeneous elastic materials*, ACM Trans. Graph., 28(3):51, 2009.
[29] F. Knöppel, K. Crane, U. Pinkall, and P. Schröder, *Globally optimal direction fields*, ACM Trans. Graph., 32(4):59, 2013.
[30] S. Koch, A. Matveev, Z. Jiang, F. Williams, A. Artemov, E. Burnaev, M. Alexa, D. Zorin, and D. Panozzo, *ABC: A big CAD model dataset for geometric deep learning*, in: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 9593–9603, 2019.
[31] S. Z. Kovalsky, N. Aigerman, R. Basri, and Y. Lipman, *Large-scale bounded distortion mappings*, ACM Trans. Graph., 34(6):191, 2015.
[32] V. Kraevoy and A. Sheffer, *Cross-parameterization and compatible remeshing of 3D models*, ACM Trans. Graph., 23(3):861–869, 2004.
[33] J. M. Lee, *Integral curves and flows*, in: Introduction to Smooth Manifolds, Springer, 434–463, 2003.
[34] Y.-S. Leung, X. Wang, Y. He, Y.-J. Liu, and C. C. L. Wang, *A unified framework for isotropic meshing based on narrow-band Euclidean distance transformation*, Comput. Vis. Media, 1:239–251, 2015.
[35] D. Li, Y. Fei, and C. Zheng, *Interactive acoustic transfer approximation for modal sound*, ACM Trans. Graph., 35(1):2, 2016.
[36] S. Lin, D. Xiao, Z. Shi, and B. Wang, *Surface reconstruction from point clouds without normals by parametrizing the Gauss formula*, ACM Trans. Graph., 42(2):14, 2022.
[37] H.-T. D. Liu, M. Gillespie, B. Chislett, N. Sharp, A. Jacobson, and K. Crane, *Surface simplification using intrinsic error metrics*, ACM Trans. Graph., 42(4):118, 2023.

[38] H.-T. D. Liu, V. G. Kim, S. Chaudhuri, N. Aigerman, and A. Jacobson, *Neural subdivision*, ACM Trans. Graph., 39(4):124, 2020.
[39] S. Liu, Y. Ji, J.-P. Guo, L. Liu, and X.-M. Fu, *Smooth bijective projection in a high-order shell*, ACM Trans. Graph., 43(4):59, 2024.
[40] W. E. Lorensen and H. E. Cline, *Marching cubes: A high resolution 3D surface construction algorithm*, ACM SIGGRAPH Computer Graphics, 21:163–169, 1987.
[41] W. Lu, Z. Shi, J. Sun, and B. Wang, *Surface reconstruction based on the modified Gauss formula*, ACM Trans. Graph., 38(1):2, 2018.
[42] M. Mandad, D. Cohen-Steiner, and P. Alliez, *Isotopic approximation within a tolerance volume*, ACM Trans. Graph., 34(4):64, 2015.
[43] F. J. Melero, Á. Aguilera, and F. R. Feito, *Fast collision detection between high resolution polygonal models*, Comput. Graph., 83:97–106, 2019.
[44] J. S. B. Mitchell, D. M. Mount, and C. H. Papadimitriou, *The discrete geodesic problem*, SIAM J. Comput., 16:647–668, 1987.
[45] M. Pan, W. Tong, and F. Chen, *Phase-field guided surface reconstruction based on implicit hierarchical B-splines*, Comput. Aided Geom. Des., 52-53:154–169, 2017.
[46] D. Panozzo, I. Baran, O. Diamanti, and O. Sorkine-Hornung, *Weighted averages on surfaces*, ACM Trans. Graph., 32(4):60, 2013.
[47] J. Peng, D. Kristjansson, and D. Zorin, *Interactive modeling of topologically complex geometric detail*, ACM Trans. Graph., 23(3):635–643, 2004.
[48] F. Policarpo, M. M. Oliveira, and J. L. D. Comba, *Real-time relief mapping on arbitrary polygonal surfaces*, ACM Trans. Graph., 24:935, 2005.
[49] S. D. Porumbescu, B. Budge, L. Feng, and K. I. Joy, *Shell maps*, ACM Trans. Graph., 24:626–633, 2005.
[50] M. Riso, E. Michel, A. Paris, V. Deschaintre, M. Gaillard, and F. Pellacini, *Direct manipulation of procedural implicit surfaces*, ACM Trans. Graph., 43(6):238, 2024.
[51] M. Rouhani, A. D. Sappa, and E. Boyer, *Implicit B-spline surface reconstruction*, IEEE Trans. Image Process., 24:22–32, 2015.
[52] C. Shen, J. F. O'Brien, and J. R. Shewchuk, *Interpolating and approximating implicit surfaces from polygon soup*, in: ACM SIGGRAPH 2004 Papers, SIGGRAPH '04, ACM, 896–904, 2004.
[53] H. Si, *Tetgen, a delaunay-based quality tetrahedral mesh generator*, ACM Trans. Math. Softw., 41(2):1–36, 2015.
[54] V. Surazhsky and C. Gotsman, *Morphing stick figures using optimized compatible triangulations*, in: Proceedings Ninth Pacific Conference on Computer Graphics and Applications. Pacific Graphics 2001, IEEE, 40–49, 2001.
[55] V. K. Suriyababu, C. Vuik, and M. Möller, *Towards a high quality shrink wrap mesh generation algorithm using mathematical morphology*, Comput. Aided Des., 164:103608, 2023.
[56] J. Süßmuth, Q. Meyer, and G. Greiner, *Surface reconstruction based on hierarchical floating radial basis functions*, Comput. Graph. Forum, 29:1854–1864, 2010.
[57] Y. Tang and J. Feng, *Multi-scale surface reconstruction based on a curvature-adaptive signed distance field*, Comput. Graph., 70:28–38, 2018.
[58] The SymEngine Project, *SymEngine: A fast symbolic manipulation library*, 2023. `https://github.com/symengine/symengine`
[59] B. Wang, T. Schneider, Y. Hu, M. Attene, and D. Panozzo, *Exact and efficient polyhedral envelope containment check*, ACM Trans. Graph., 39(4):114, 2020.
[60] Q. Wang, W.-X. Zhang, Y.-Y. Cheng, L. Liu, and X.-M. Fu, *Practical construction of globally injective parameterizations with positional constraints*, Comput. Vis. Media, 9:265–277, 2023.

[61] Z. Wang, Y. Zhang, R. Xu, F. Zhang, P.-S. Wang, S. Chen, S. Xin, W. Wang, and C. Tu, *Neural-singular-Hessian: Implicit neural representation of unoriented point clouds by enforcing singular Hessian*, ACM Trans. Graph., 42(6):274, 2023.

[62] K. Ye, K. Zhou, Z. Pan, Y. Tong, and B. Guo, *Low distortion shell map generation*, in: 2007 IEEE Virtual Reality Conference, IEEE, 203–208, 2007.

[63] Y. Zhang, Z. Wang, Z. Zhao, R. Xu, S. Chen, S. Xin, W. Wang, and C. Tu, *A Hessian-based field deformer for real-time topology-aware shape editing*, in: SIGGRAPH Asia 2023 Conference Papers, SA '23, ACM, 114, 2023.

[64] Q. Zhou and A. Jacobson, *Thingi10k: A dataset of 10,000 3D-printing models*, arXiv:1605.04797, 2016.