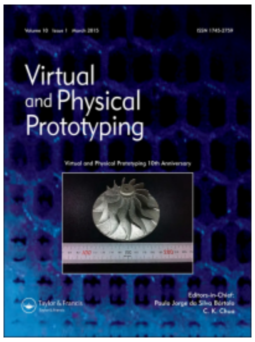

Virtual and Physical Prototyping

ISSN: 1745-2759 (Print) 1745-2767 (Online) Journal homepage: www.tandfonline.com/journals/nvpp20

Non-planar slicing for high-genus surfaces with non-coplanar interfaces

Mania Aghaei Meibodi , Abdallah Kamhawi , Ashkan Foroughi Dehnavi & Yichuan Li

To cite this article: Mania Aghaei Meibodi , Abdallah Kamhawi , Ashkan Foroughi Dehnavi & Yichuan Li (2026) Non-planar slicing for high-genus surfaces with non-coplanar interfaces, Virtual and Physical Prototyping, 21:1, e2603853, DOI: 10.1080/17452759.2025.2603853

To link to this article: https://doi.org/10.1080/17452759.2025.2603853

VIRTUAL AND PHYSICAL PROTOTYPING
2026, VOL. 21, NO. 1, e2603853 (28 pages)
https://doi.org/10.1080/17452759.2025.2603853

Non-planar slicing for high-genus surfaces with non-coplanar interfaces

Mania Aghaei Meibodi*, Abdallah Kamhawi*, Ashkan Foroughi Dehnavi* and Yichuan Li*

DART Laboratory, Taubman College of Architecture and Urban Planning, University of Michigan, Ann Arbor, MI, USA

ABSTRACT

3D printing enables efficient fabrication of Materially Optimized Reinforced Concrete Elements (MORCE), but conventional planar slicing methods struggle with their complex geometries, leading to layer misalignment and structural instability. Existing non-planar slicing techniques typically produce solid toolpaths with infill material, making them unsuitable for MORCE, which require surface-based geometries with hollow interiors to accommodate reinforcement, casting, and insulation. A few non-planar slicing models for surface-based complex geometries have appeared recently; however, they rely on heuristic methods that require manual input and involve intervention at multiple stages. This paper presents an analytical model for automated non-planar slicing that generates geometry conforming slices by extracting critical geometric features – such as saddle points at bifurcations and boundary edges. The method uses Morse theory to identify the critical points (i.e. saddle points) and geodesic distance fields to generate a geometry-conforming scalar field that accurately captures critical points and non-coplanar interfaces. A bias term is introduced to correct slice non-uniformity near saddle points, ensuring smooth and consistent slicing. The effectiveness of this approach is validated through comparative analysis on three topology-optimised beams. The proposed method achieves perfect interface boundary conformance across all segments and produces uniform layer transitions, demonstrating robust handling of bifurcations and non-coplanar interfaces.

ARTICLE HISTORY
Received 22 August 2025
Accepted 6 December 2025

KEYWORDS
3D concrete printing; non-planar slicing; materially optimised elements; multi-axis 3D printing; 3D printing complex geometries

1. Introduction

1.1. Paper motivation

The construction industry must build infrastructure for 2.3 billion people by 2050 [1–5], equivalent to a Paris-sized city every week [6,7]. At the same time, countries like the United States of America face a housing crisis [8–10] due to material shortages, labour shortages, and outdated construction methods [11,12]. Conventional construction methods are inefficient [13–17], labour intensive, and wastefully use excessive material [18–21]; thus, they struggle to meet the demands of rapid urbanisation. Reinforced concrete (RC) is vital for urban densification by enabling long-span, large, and multi-story structures [22,23]. However, the inefficient use of concrete makes it the second-most consumed material worldwide [24–26], which is exacerbated by the reliance of traditional concrete construction methods on costly, one-time formwork, that often exceed 50% of the budget and end up overloading landfills [27–31].

3D concrete printing (3DCP) has the potential to address the limitations of conventional concrete construction methods and the urgent need for rapid urbanisation by enabling waste-free production of materially optimised reinforced concrete elements (MORCE) that has the potential to reduce material use by 30%–70% [32–35], allowing for swift, efficient construction of multistory buildings to meet the demands of rapid urbanisation with minimal environmental impact. Creating MORCE involves techniques such as topology optimisation to reduce material use and weight while maintaining structural integrity. These elements often feature complex geometries featuring high-genus-number topologies, extreme overhangs, and multiple hollow ribs [1–4,36–38].

High-genus refers to surfaces or solid geometries that contain multiple topological holes or handles – formally, geometries with genus $g>0$. The genus quantifies the number of independent closed loops that can be drawn on a surface without cutting it. In other words, genus g denotes the number of independent surface cycles (first Betti number [39,40]). For example, a sphere has genus $g = 0$, a torus has $g = 1$, and topology-optimised structures often exhibit much higher genus (e.g. $g > 5$–20), characterised by multiple internal voids and interconnected channels. In this paper,

CONTACT Mania Aghaei Meibodi mania.meibodi@gmail.com
*These authors contributed equally to this work.

'high-genus' therefore denotes geometries with complex internal connectivity. Specifically, in the context of this paper, high-genus parts are those that exhibit $g \geq 2$ per printable segment. Current attempts at 3D printing MORCE suffer from the limitations of planar-slicing that directs the printer's printhead to deposit material only in layers parallel to the ground, restricting its application to simple forms and making it incompatible with the production of materially optimised parts that inherently entail complex geometries [38].

MORCE are often ribbed structures [3,36,41–45], or rib structures with thin shells between them [1] (Figure 1). These forms are generated through topology optimisation [3,4,36,46] or 2D/3D stress lines methods [1,36,47], which define the path of internal forces that are later translated into solid RC ribs [1,37,48]. To perform structurally, these ribs are translated as hollow, stay-in-place formwork for 3D printing, allowing for rebar and concrete infill. Typically, a MORCE, such as a beam, is not printed in one piece. Instead, they are conventionally segmented into a series of discretised parts that are assembled together after 3DCP. These segmented parts exhibit complex, high-genus topologies and include critical features that must be considered. For example, they may feature multiple bifurcations, merging areas with branches of varying lengths, non-coplanar interfaces, and hollow areas designed to accommodate inserts such as reinforcement or insulation.

Studies exploring 3DCP for MORCE (Figure 2), demonstrated the incompatibility and inefficiency of the widespread planar slicing and planar 3DP (P-3DCP) method with MORCE [41–43,49]. P-3DCP of geometrically complex features inherent to MORCE, like areas of bifurcation, merging, sharp cantilevers and non-coplanar interfaces, results in partially supported layers (staircase effect [50–54]), requiring external support to prevent collapse, thereby reducing productivity and increasing waste [55]. For instance, both the OptiBridge [42] and XtreeE column [41] projects required printing additional scaffolding membranes to support overhangs that were removed during post-processing, resulting in material waste and extra fabrication time (Figure 2(a–c)). In the Girder beam project a complex form was discretised into smaller parts to meet planar 3DP constraints. The resulting beam had reduced structural integrity due to the inaccurate assembly alignment of the small parts with imprecise interfaces (Figure 2(d)) [43,49]. Overall, planar slicing and P-3DCP restrict the architectural design to simpler geometries such as orthogonal walls and hinder waste-free manufacturing of MORCE.

To fully realise the potential of 3DCP in construction, advancements in material deposition techniques are needed to accommodate the complex geometries inherent to MORCE. A major challenge in achieving this is the absence of a generalisable non-planar slicing model, which limits the successful 3D printing of MORCE structures. This paper presents a robust approach for generating non-planar slicing tailored to MORCE forms.

The proposed geometry-conforming non-planar slicing method provides a practical computational framework for manufacturing topology-optimised parts with pronounced curvature and branching that are difficult or impossible to produce with conventional planar layers. At architectural and construction scales, it enables extrusion-based robotic 3D printing of structurally optimised beams, columns, and nodes – in concrete, continuous fibre, or metal – so that deposited material (derived from the slices) aligns with principal stress trajectories. This improves load transfer, strengthens interlayer bonding, reduces or eliminates supports and infill to cut waste and increase productivity, allows for embedded conduits or multimaterial inserts, and yields precise, assembly-ready interfaces. Beyond construction, the approach applies directly to aerospace and automotive lightweighting, where topology-optimised brackets and joints require smooth, continuous surfaces to minimise stress concentrations; and to biomedical implants and prosthetics, where organic, porous, high-genus morphologies benefit from curvature-conforming paths for enhanced mechanics and biocompatibility. In sum, the method bridges topology optimisation and fabrication by generating geometry-conforming slices across scales – from large robotic concrete printing to high-resolution additive manufacturing of intricate functional components.

1.2. State of the art on non-planar slicing methods

Six degrees of freedom (6-DOF) robotic arm manipulators are essential for non-planar 3D printing of parts with complex geometries. Although many approaches to non-planar slicing exist [56–64], they fail to accommodate hollow components with high-genus topologies without resorting to excessive infill, which prevents leveraging the hollow interiors for construction inserts (Figure 3). In architecture, surface-based geometries with hollow cores are often employed to integrate reinforcement bars, casting, and insulation (Figure 1) [56,57]. Additionally, existing methods do not consider complex non-coplanar interfaces, which are essential for seamless assembly of the intricate, discretised parts of MORCE.

This section surveys state-of-the-art methods for non-planar slicing. We categorise existing approaches into

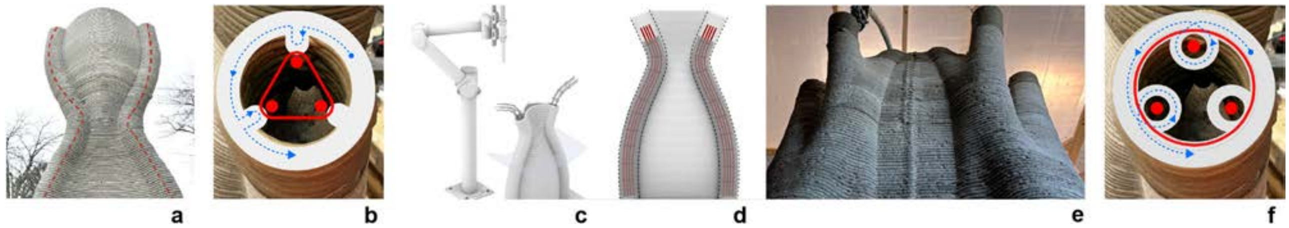

Figure 1. Non-planar slicing and print-path strategies from our prior work on robotic NP-3DCP at the DART Laboratory. (a) Printed RC wall prototype with non-planar layers demonstrating up to 60% material savings [3]. (b–d) Toolpaths (blue) enabling integration of a rebar cage (red). (e,f) Print-path strategy for achieving hollow channels within conduits that host single rebars and allow inserting interlayer circular stirrup.

two main groups: (a) techniques for solid geometry slicing [58–60,68,69] and (b) techniques for surface-based geometry slicing [56,57], also referred to as open mesh or single shell slicing. Section 1.2.1 discusses methods for solid mesh slicing, primarily used in small-scale manufacturing with infill and Section 1.2.2 focuses on techniques for open mesh slicing, emphasising the limited research on large-scale, surface-based geometries, which are significantly crucial for MORCE.

1.2.1. Closed mesh slicing methods for small scale parts

Four primary methods exist in the literature for closed mesh slicing: (a) space-deformation-based slicing (Figure 4(a)) [59,69] warps the entire model in a slicing space, where planar slices can be easily taken, and then maps these slices back onto the original geometry as smooth, curved layers. This approach effectively mitigates the 'staircase' effect and avoids nozzle collisions while printing, but it relies on having a reasonably thick or volumetric part that can accommodate the deformation, (b) voxel-based growth slicing (Figure 4(b)) [58] discretizes the entire object into voxels and defines a scalar growth field that determines how material can be added, layer by layer, without supports. Each curved layer corresponds to an isosurface of that field, ensuring collision-free deposition onto previously printed material. This is well suited for fully solid models but becomes problematic for hollow parts, which cannot support the interior-filling steps required by voxel-based methods, (c) stress-guided volumetric slicing (Figure 4(c)) [60] builds a tetrahedral mesh of the volume and computes a stress-aligned scalar field. Layers are then extracted from the resulting isosurfaces, placing filaments along principal stress directions to enhance strength. Although this yields mechanically optimised internal structures, the method again requires a significant internal volume for stress computation and filament alignment, and (d) skeleton-based volumetric slicing (Figure 4(d)) [68] employs a medial-axis or skeletonisation procedure to partition the solid volume into smaller sub-regions, each of which is fitted with curved surfaces (e.g. ellipsoids). This partitioning facilitates collision-free slicing and deposition. However, the method assumes a watertight, mesh and a volumetric interior to decompose – thus it struggles with thin-walled or hollow-core geometries that do not lend themselves to such skeleton-based subdivisions.

Each of these categories shares a common assumption: the presence of a substantial solid interior volume that can be deformed, voxelized, stress-analysed, or skeletonised. As a result, purely hollow or single-surface parts – especially those designed to remain open or accommodate features such as reinforcement bars – are not well served by these methods.

1.2.2. Surface-based geometry slicing methods

Surface-based slicing methods focus on generating slices that lie directly on a surface, avoiding the offsets, infills, and supports typically required for solid (closed-mesh) geometries. A recently developed method [56,57] focuses on the generation of non-planar slicing for open mesh surfaces. This method computes a geodesic-distance field on the surface or sub-surfaces from user-specified target curves (which can be placed at part's edges, saddle points, etc.). Next, they blend or union multiple distance fields if the shape has bifurcations, ensuring smooth transitions. Each chosen level-set (isocurve) of that distance field becomes one curved layer in the slicing. Finally, they resolve complex features such as saddle points – where isocontours would cause gaps – by implicit partitioning, effectively cutting the model near those saddle points so each part can have a continuous slicing without abrupt holes (Figure 5).

While the presented methods laid the foundation for research in surface-based non-planar slicing for building elements, there are still, several challenges remain in the current state of the art:

(a) The method relies on manual input for mesh segmentation and the placement of target curves that

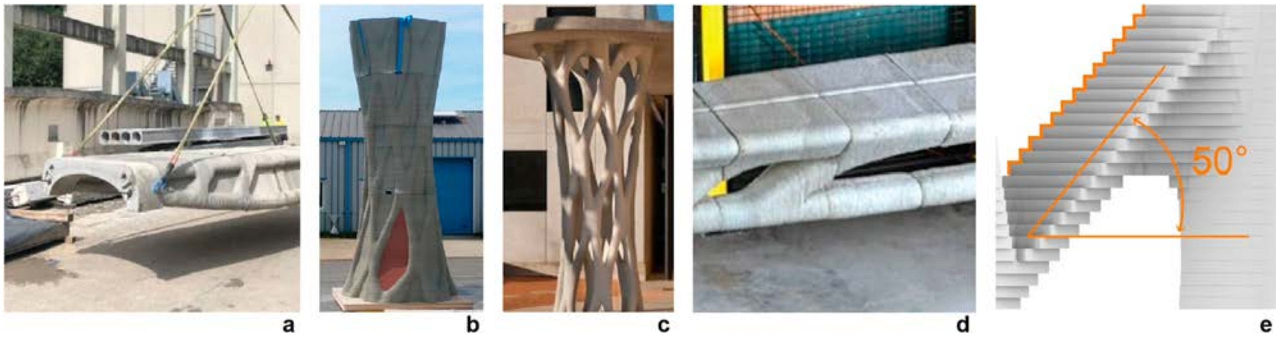

Figure 2. P-3DCP for (a) OptiBridge [42], (b) XtreeE column required printing support (red), XtreeE column after postprocessing [41], (d) Lack of alignment accuracy in the assembly of multiple simplified parts in Girder beam [43], and (e) Staircase effect inherent to P-3DCP.

are used for interpolating slices. This becomes tedious when the part has multiple branching and merging areas, as significant user input is needed to define targets correctly. This is critical in the case of complex high-genus geometries, where the manual extraction of target curves increasingly becomes tedious.

(b) The extraction of slices based on user input leads to a heuristic approach that approximates the position of the slices. While this can be faster and more intuitive, it does not guarantee finding the best solution. The resulting slices are often an approximation and rely on human experience or intuition.

(c) The approximation of the slices can result in non-uniform spacing around saddle points and can lead to inconsistent layer heights, making 3DCP challenging and resulting in defects and gaps in the final printed part.

1.3. Paper goal and contributions

Building on the existing body of literature, this paper presents a non-planar slicing model. The primary goal of the developed algorithm is to preserve geometric fidelity – ensuring accurate feature representation, homogeneity in layer heights across non-planar slices, and effective handling of bifurcations and merging regions – to accommodate MORCE's surface-based hollow-core systems.

The presented non-planar slicing model moves beyond the traditional heuristic, rules-of-thumb approach to non-planar slicing of surface-based geometries by providing an analytical, fully automated, and scalable solution. Instead of relying on manual input for segmentation, our approach leverages Morse Theory [70–72] and geodesic distance fields to generate optimal, geometric-conforming non-planar slices. A key contribution is the introduction of a bias term that integrates the influence of saddle points and boundary conditions, resulting in smooth, geometrically conforming slicing patterns with a more uniform distribution and minimal height variation around saddle points. This method effectively addresses the slicing challenges of hollow, high-genus geometries and significantly reduces the risk of collapse during 3DCP by overcoming critical limitations inherent in existing planar slicing techniques.

2. Methods: geometry-informed non-planar slicing model for complex parts

The modelling process involves creating a geometry-informed slicing algorithm that takes a mesh

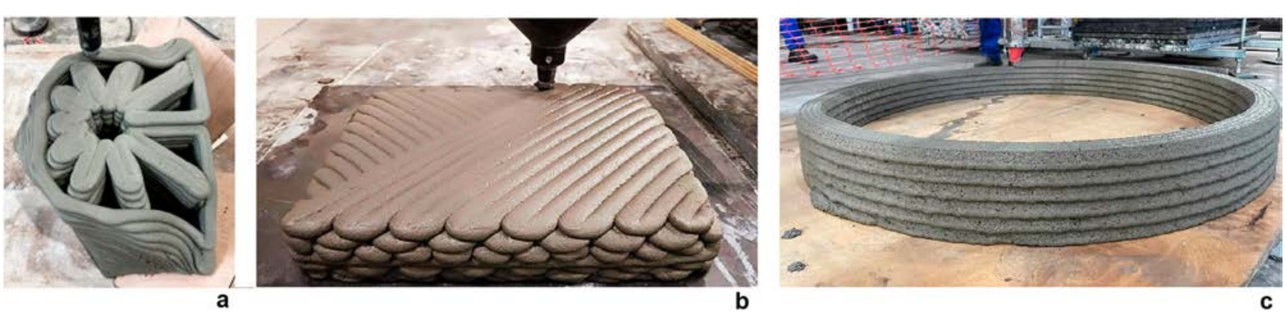

Figure 3. (a, b) 3D printing of solid volumes with infill material [65,66], (c) surface-base 3D printing with hollow cores [67].

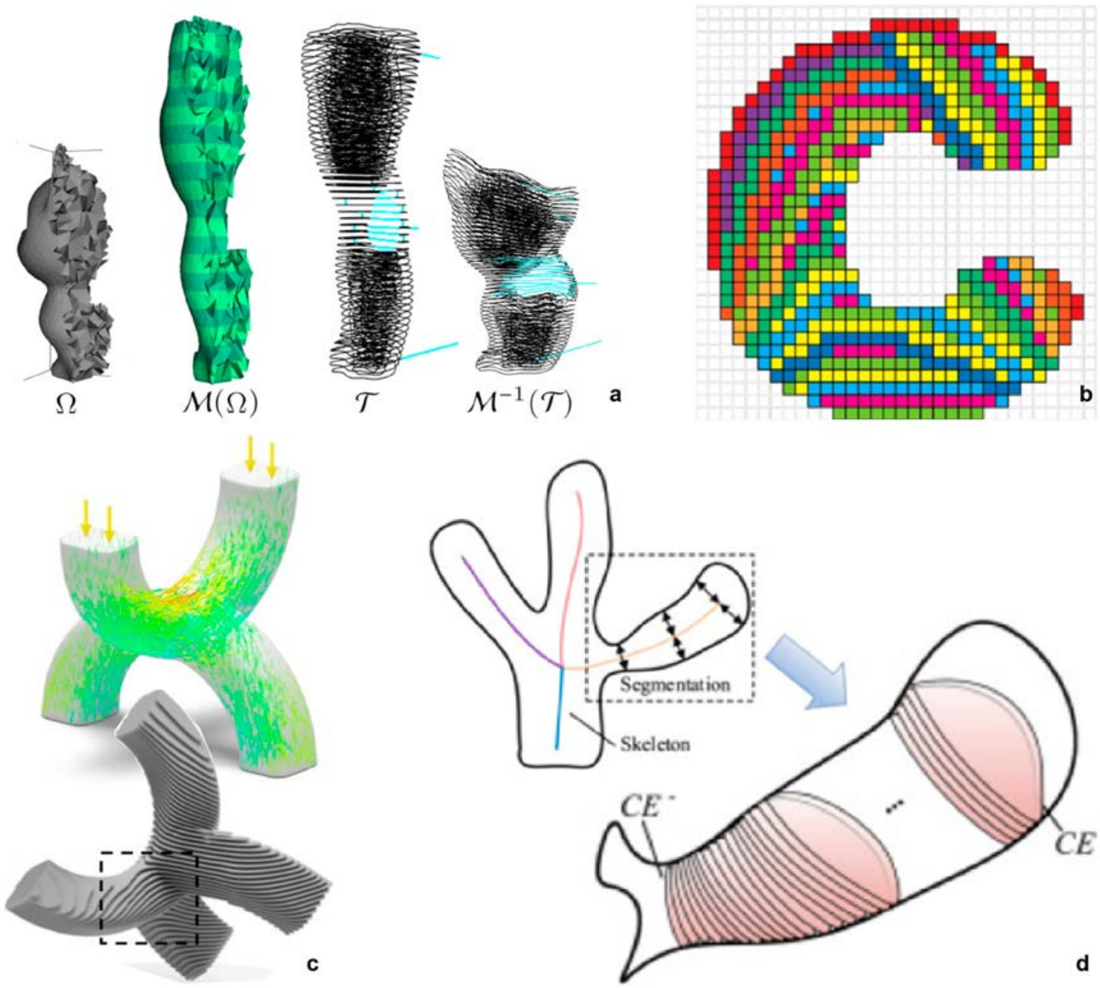

Figure 4. Closed mesh slicing models : (a) CurviSlicer [59], (b) Support-Free Volume Printing with Multi-Axis Motion [58], (c) Multi-Axis Filament Alignment for Controlled Anisotropic Strength [60], (d) Volume Decomposition for Multi-Axis Ellipsoidal Slicing [68].

representation of the part as input and outputs non-planar slices through the following stages (Figure 6):

(1) *Identification of Critical Geometric Features*: The algorithm identifies two types of critical features: vertices on boundary edges and saddle points. This step aims to robustly identify these features in any input complex geometry with a high genus number, as they will inform and guide the subsequent slicing process. The implementation employs discrete Morse theory, which extends continuous Morse theory to piecewise-linear functions on triangular meshes [70–74]. In this discrete setting, critical points are detected at mesh vertices through analysis of the scalar field's behaviour in the vertex's one-ring neighbourhood.

(2) *Modeling Geometry Conforming Scalar Field*:

To generate non-planar slices that follow the topology of complex geometries with high-genus number, we construct a scalar field that responds to the identified critical geometric features – such as boundary vertices and saddle points. First, we compute geodesic distance fields from mesh vertices to boundary vertices to establish a continuous measure of spatial proximity along the surface. Next, we introduce a bias term **B** that encodes the influence of saddle points. This bias locally modifies the field to reflect the surface's internal curvature transitions, which helps guide slicing layers to conform around regions of high geometric variation. The combined field – integrating geodesic distances and curvature-derived bias – is then normalised to a 0–1 range, producing a geometry-conforming scalar field. This field acts as a topological map that enables smooth, continuous, and curvature-aware extraction of non-planar slicing layers that adapt naturally to the form of the part.

(3) *Iso-Contour Extraction for Non-Planar Slices*: Finally, the slicing algorithm employs an adaptive iso-curve extraction strategy that maintains uniform

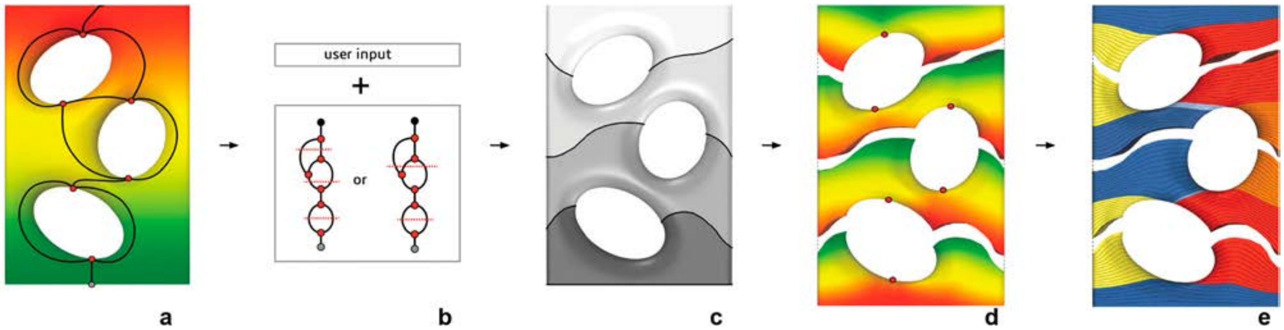

Figure 5. State of the art method for non-planar slicing of surface-based geometries [56]: (a) Typology manifestation using geodesic distance from the base as a scalar function, (b) Heuristics and user input as target curves, (c) Splitting the mesh into components, (d) Identification of the saddle points of the distance function, and (e) Secondary partitioning crossing saddle points.

layer spacing while respecting the geometry's critical features. The extraction process dynamically adjusts the iso-value increments based on local gradient information and the remaining normalised range. This ensures evenly distributed slices throughout complex geometries while maintaining smoothness at bifurcations and merging regions.

Method assumption: the method assumes an input 2-manifold surface, possibly with boundary, free of self-intersections and non-manifold elements, and without abrupt vertex outliers relative to their local neighbourhood.

2.1. Identification of critical geometric features

The points at critical features, such as bifurcations and the part's interface, strongly influence the geometry of the slices. We refer to these points collectively as *source points*. Let S denote the set of source points, defined as the union of three distinct subsets: saddle points S_s, upper boundary points S_u, and lower boundary points S_ℓ:

$$S = S_s \cup S_u \cup S_\ell \tag{1}$$

2.1.1. Saddle points (S_s)

In scalar fields on surfaces, a saddle point is a critical point where the function increases in some directions and decreases in others. We developed a saddle-point detection algorithm based on the sign-change criterion from discrete Morse theory. This detects bifurcation/merging loci that guide conforming non-planar slicing.

This algorithm operates on a vertex-defined scalar field, which can be specified in two ways. If an explicit scalar field is provided, the algorithm uses it directly for critical-point analysis. This allows adaptation to specific geometries or manufacturing constraints when a custom field better captures the relevant features. If no field is provided, the algorithm defaults to an axis-distance field, typically along the build z-axis for additive manufacturing.

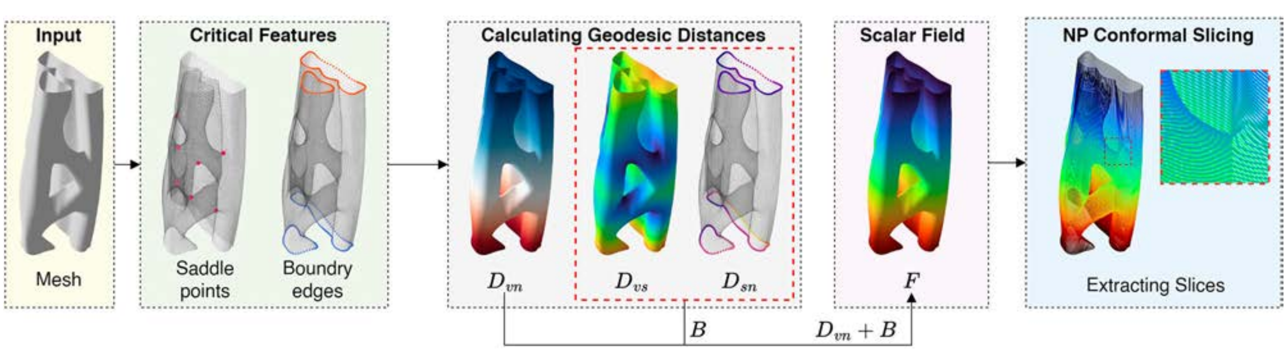

Figure 6. Geometry–informed non–planar slicing model. The algorithm takes an input mesh and identifies critical geometric features (boundary vertices and saddle points), using discrete Morse theory. Geodesic distance fields from boundary vertices are added to a bias term that captures the influence of saddle-points to construct a geometry-conforming scalar field. Non-planar slices are then extracted via adaptive iso-curve tracing, where iso-value steps are adjusted by the local field gradient to maintain near-uniform layer thickness and smooth transitions across bifurcations. The resulting slices conform to complex, high-genus geometries.

For each vertex v_i with position coordinates (x_i, y_i, z_i) and a chosen axis $a \in \{x, y, z\}$, the default scalar field value is:

$$f_i = p_i^a - \min_{j \in V} p_j^a \tag{2}$$

where p_i^a is the a-coordinate of vertex i and V is the set of all mesh vertices. For the common z-axis case, this simplifies to $f_i = z_i - z_{\min}$.

For a vertex v with scalar value f_v from the chosen field and ordered one-ring neighbours, we denote the neighbour values as $\{f_i\}_{i=1}^{k}$, where k is the number of neighbours in the one-ring and i indexes each neighbour vertex. We use the ordered one-ring to capture how f varies around v angularly. The differences between the central vertex value and its neighbours are computed as:

$$d_i = f_v - f_i \tag{3}$$

We count the number of sign changes C in the cyclic sequence of differences after filtering those with magnitude below a tolerance ε. Figure 7 illustrates this sign-change analysis process, showing how the algorithm traverses the ordered one-ring neighbourhood to identify alternating positive and negative differences that characterise saddle points. The classification of the vertex follows:

$$\text{Critical type} = \begin{cases} \text{maximum} & \text{if } C = 0 \text{ and all } d_i > \epsilon \\ \text{minimum} & \text{if } C = 0 \text{ and all } d_i < -\epsilon \\ \text{saddle} & \text{if } C \geq 4 \text{ and } C \text{ is even} \\ \text{regular} & \text{otherwise} \end{cases} \tag{4}$$

This sign-change criterion naturally captures the alternating up-down pattern characteristic of saddle points, where the scalar field rises in certain directions and falls in others.

The saddle point detection algorithm includes several refinements – adaptive epsilon computation, confidence scoring, and geodesic clustering – described below. These refinements improve the robustness of the saddle-point detection algorithm, especially when sign-change analysis yields multiple false positives due to mesh noise or errors.

2.1.1.1 Adaptive Epsilon Computation. rather than using a global tolerance, the algorithm computes a per-vertex epsilon based on local field variation. For vertex v with its set of neighbours denoted as $\mathcal{N}(v)$, let $\Delta_{\mathcal{N}} = \{|f_i - f_j| : i, j \in \mathcal{N}(v),\ i \neq j\}$ represent the set of all pairwise absolute differences between neighbour values. Figure 8 demonstrates how the adaptive epsilon varies across the mesh, with larger tolerances in noisy regions and smaller tolerances in smooth regions. The adaptive epsilon is computed as:

$$\epsilon_v = \max\left(\epsilon_{\text{global}}, \min(0.1 \cdot Q_{0.1}(\Delta_{\mathcal{N}}), 0.01 \cdot (\max_{\mathcal{N}} f - \min_{\mathcal{N}} f))\right) \tag{5}$$

where $Q_{0.1}$ denotes the 10th percentile function applied to the distribution of differences.

2.1.1.2 Confidence Scoring. each detected saddle receives a confidence score to quantify the reliability of the detection. Figure 9 visualises how confidence scores vary across detected saddle points, with higher scores indicating more reliable detections based on signal strength, pattern consistency, and ring completeness. Let $\mathcal{D}_v = \{|d_i| : |d_i| > \epsilon_v,\ i = 1, \ldots, k\}$ denote the set of significant absolute differences for vertex v. The signal strength σ_v is defined as:

$$\sigma_v = \begin{cases} \text{med}(\mathcal{D}_v) & \text{if } |\mathcal{D}_v| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

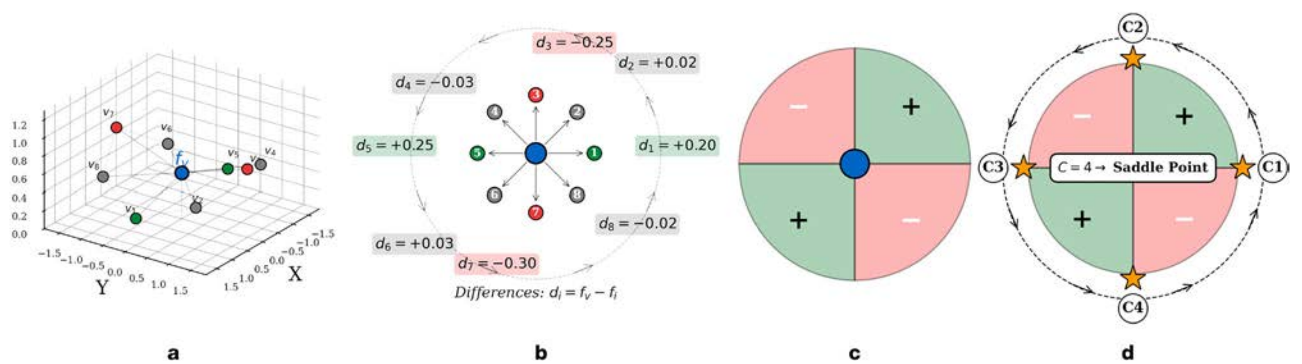

Figure 7. Sign-change analysis for saddle point detection. (a) One-ring neighbourhood of vertex v with scalar field values shown as heights. (b) Ordered traversal of neighbours showing field differences $d_i = f_v - f_i$. (c) Sign pattern after filtering differences below threshold ε, with alternating positive (green) and negative (red) regions. (d) Detection of $C = 4$ sign changes indicating a saddle point. The method identifies the characteristic up-down-up-down pattern around saddle points in discrete meshes.

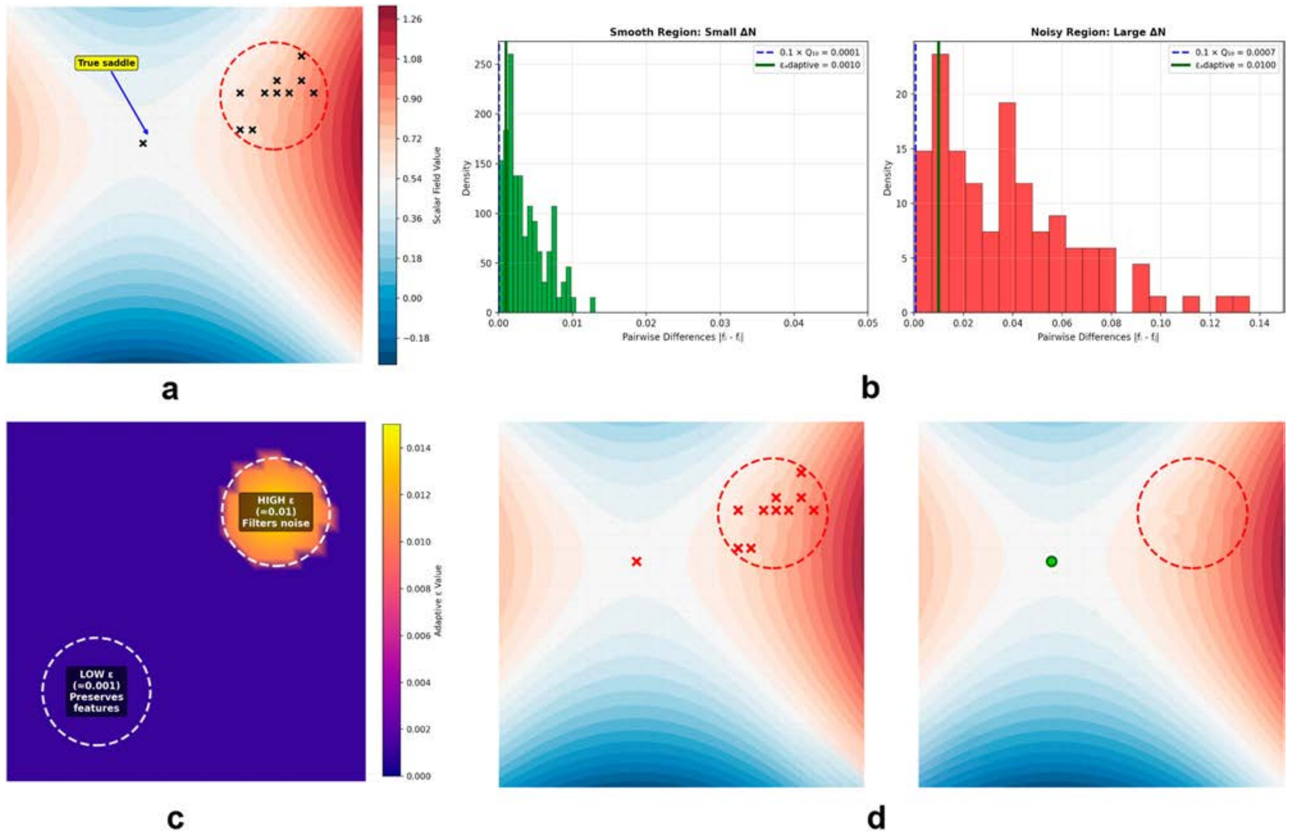

Figure 8. Adaptive epsilon computation across the mesh. (a) Global uniform epsilon leading to false positives in noisy regions and missed detections in smooth regions. (b) Local field variation analysis showing the distribution of pairwise differences $\Delta_{\mathcal{N}}$ in the one-ring neighbourhood. (c) Computed adaptive epsilon values ϵ_v visualised as a heat map on the mesh, with blue indicating small tolerances and red indicating large tolerances. (d) Improved saddle detection with adaptive epsilon, showing reduced false positives while maintaining sensitivity to true critical points.

where med denotes the median operator. The pattern consistency function $\phi : \mathbb{N} \to [0, 1]$ quantifies how well the sign-change count matches expected patterns:

$$\phi(C) = \begin{cases} 1.0 - \dfrac{\sqrt{\frac{1}{k}\sum_{i=1}^{k}(d_i - \bar{d})^2}}{\frac{1}{k}\sum_{i=1}^{k}|d_i| + 10^{-10}} & \text{if } C = 0 \\ \frac{\min(n_+, n_-)}{\max(n_+, n_-)} & \text{if } C = 2 \\ \frac{1}{1 + 0.1 \cdot |C - 4|} & \text{if } C \geq 4 \text{ and } C \text{ is even} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $n_+ = |\{i : d_i > \epsilon_v\}|$, $n_- = |\{i : d_i < -\epsilon_v\}|$, and $\bar{d} = \frac{1}{k}\sum_{i=1}^{k} d_i$. The ring quality factor $\rho_v \in \{0.7, 1.0\}$ accounts for neighbourhood completeness:

$$\rho_v = \begin{cases} 1.0 & \text{if ring is complete} \\ 0.7 & \text{if ring is incomplete} \end{cases} \tag{8}$$

The final confidence score $\mathcal{C}_v \in [0, 1]$ is:

$$\mathcal{C}_v = \min\left(1, \frac{\sigma_v \cdot \phi(C) \cdot \rho_v}{\sigma_v + 1}\right) \tag{9}$$

2.1.1.3 Geodesic Clustering. when multiple saddle points appear in close proximity, we employ geodesic-based clustering. Let $d_g : V \times V \to \mathbb{R}_{\geq 0}$ denote the geodesic distance function on the mesh. Saddles $s_i, s_j \in S_s$ belong to the same cluster if $d_g(s_i, s_j) < \tau$, where τ is the clustering radius. Within each cluster, the representative is selected as:

$$s^* = \arg\max_{s \in \text{cluster}} \mathcal{C}_s \tag{10}$$

with ties broken by selecting the vertex with minimum scalar field value.

2.1.2. Upper boundary points (S_u)

Upper boundary points are identified through analysis of mesh topology and position along a chosen axis a

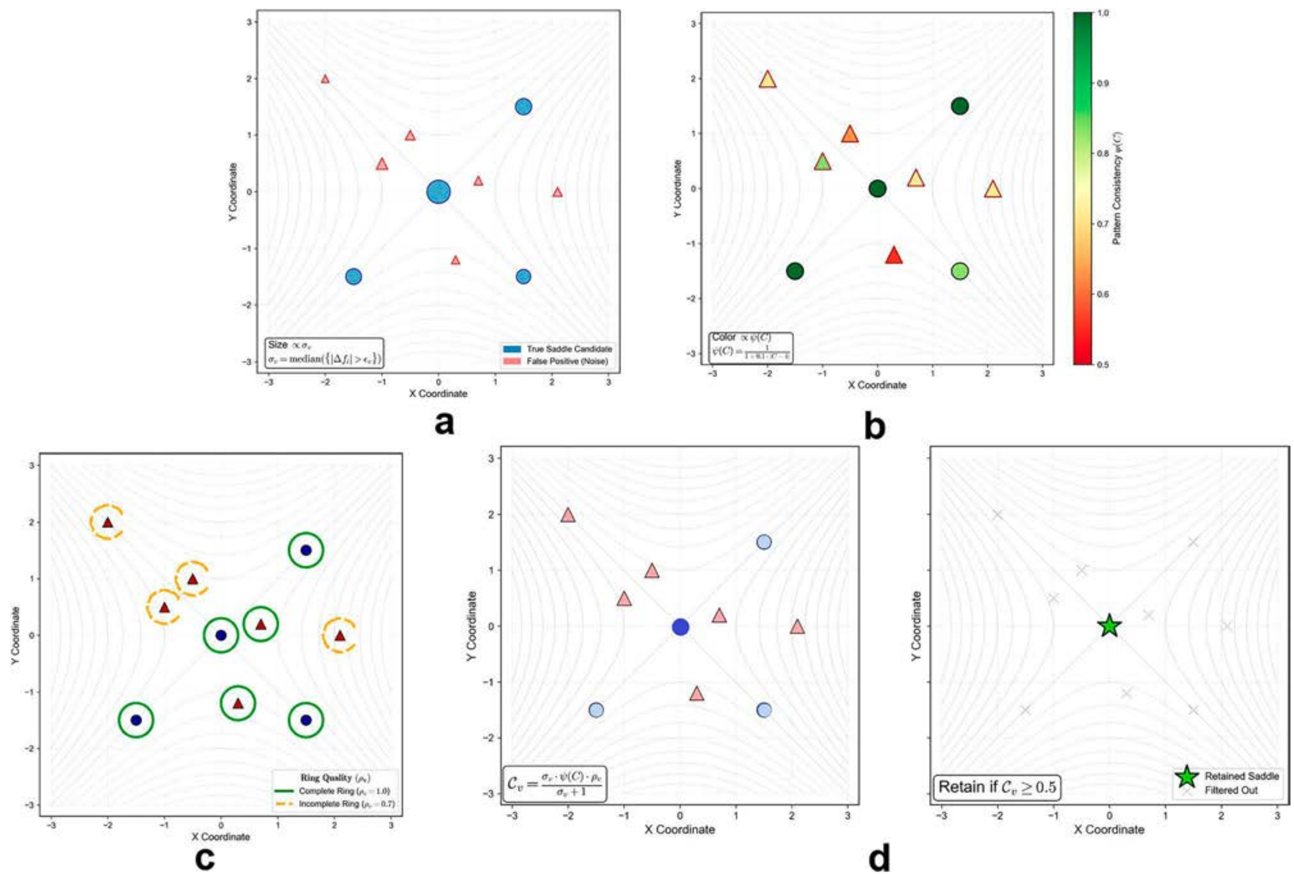

Figure 9. Confidence scoring for detected saddle points. (a) Signal strength σ_v computed from median of significant differences, shown as sphere sizes at each detected saddle. (b) Pattern consistency $\phi(C)$ visualised as colour coding, with green indicating ideal 4-change pattern and red indicating deviations. (c) Ring quality factor ρ_v showing complete (solid green) versus incomplete (dashed yellow) one-ring neighbourhoods. (d) Final confidence scores $\mathcal{C}_v$ combining all factors, with high-confidence saddles (dark blue) retained and low-confidence detections (light blue) filtered.

(typically the build axis, so that 'upper/lower' aligns with fabrication). Let $\mathcal{E}$ denote the set of mesh edges and $\mathcal{F}$ denote the set of faces. Boundary edges $\mathcal{E}_b \subset \mathcal{E}$ are those belonging to exactly one face:

$$\mathcal{E}_b = \{e \in \mathcal{E} : |\{f \in \mathcal{F} : e \in f\}| = 1\} \tag{11}$$

Here, $|\cdot|$ denotes the cardinality (number of elements) of a set.

To compare boundary loops in a scalar form – and classify them as upper or lower boundaries we compute the mean position along the build axis. Connected components of boundary edges form boundary loops $\mathcal{L}_1, \mathcal{L}_2, \ldots, \mathcal{L}_m$. For each loop $\mathcal{L}_j$, the mean position along axis a is:

$$\mu_j^a = \frac{1}{|\mathcal{L}_j|} \sum_{v \in \mathcal{L}_j} p_v^a \tag{12}$$

To account for geometric noise and small height variations – so that slightly uneven boundary loops are still recognised as part of the same upper boundary – we use a tolerance δ. Let $\delta \in (0, 0.5)$ denote the classification tolerance. Vertices in loop $\mathcal{L}_j$ belong to S_u if:

$$\mu_j^a > \max_i \mu_i^a - \delta \cdot (\max_i \mu_i^a - \min_i \mu_i^a) \tag{13}$$

2.1.3. Lower boundary points (S_ℓ)

Lower boundary points follow an analogous identification process. Vertices in loop $\mathcal{L}_j$ belong to S_ℓ if:

$$\mu_j^a < \min_i \mu_i^a + \delta \cdot (\max_i \mu_i^a - \min_i \mu_i^a) \tag{14}$$

When boundary loops fall in the middle range, they are assigned based on minimum geodesic distance to existing classified loops. This step ensures consistent topological grouping when the geometry contains intermediate openings or nested surfaces.

2.2. Modeling geometry conforming scalar field

To remove manual parameter tuning and ensure smooth transitions at critical features, this work develops a

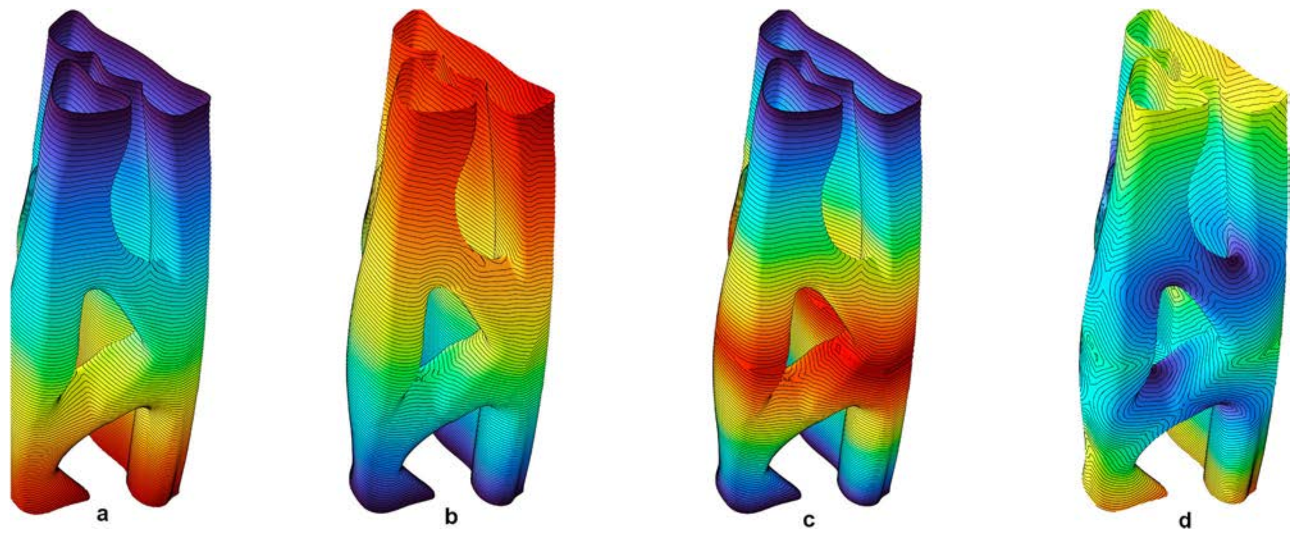

Figure 10. Comparison of geodesic distance fields for slicing a segment from a topology optimised beam. (a) mesh vertices to vertices on the upper boundary ($\mathbf{D}_{vn}[upper]$), (b) mesh vertices to vertices on the lower boundary ($\mathbf{D}_{vn}[lower]$), (c) mesh vertices to vertices to both upper and lower boundaries ($\mathbf{D}_{vn}$), and (d) mesh vertices to saddle points ($\mathbf{D}_{vs}$).

geometry-conforming scalar field that integrates influences from both boundary conditions and internal saddle points. Most existing approaches to constructing a scalar field yield non-uniform slices because they capture only a single aspect of the geometry's critical features typically the boundary-edges. A common strategy for non-planar slicing generates slices from geodesic distances between mesh vertices and boundary vertices, denoted $D_{vn} \in \mathbb{R}^{|V|\times|N_\partial|\times 2}$. Each entry $(D_{vn})_{i,j,b}$ represents the distance from vertex i to boundary segment j, with $b \in \{\text{upper}, \text{lower}\}$ distinguishing between upper vs. lower vertices on boundary edges. However, relying on D_{vn} alone disregards saddle points and produces discontinuities where branching or merging occurs. An alternative approach might consider purely vertex-to-saddle distances $D_{vs} \in \mathbb{R}^{|V|\times|S_s|}$, where S_s is the number of saddle points. Each of these captures only a subset of the critical structure, resulting in non-uniform slices (Figure 10).

In our earlier prototype of scalar-field modelling, we explored a naive weighted combination of individual distance fields. However, determining optimal weights requires computationally expensive tuning for each geometry. To address these limitations, we propose an analytical approach introducing a bias term **B** encoding saddle influences:

$$F = \mathcal{F}(\mathbf{D}_{vn} + \mathbf{B}) \tag{15}$$

where $\mathcal{F} : \mathbb{R}^{|V|\times|N_\partial|\times 2} \to [0,1]^{|V|}$ is a reparameterization operator that performs dimension reduction followed by normalisation.

The reparameterization operator $\mathcal{F}$ is defined through its component operations. Given an input tensor $\mathbf{T} \in \mathbb{R}^{|V|\times|N_\partial|\times 2}$, the operator is:

$$\mathcal{F}(\mathbf{T}) = \left[\psi(\mathcal{R}_{\text{upper}}(\mathbf{T}), \mathcal{R}_{\text{lower}}(\mathbf{T}))\right] \tag{16}$$

where the reduction operators $\mathcal{R}_{\text{upper}}, \mathcal{R}_{\text{lower}} : \mathbb{R}^{|V|\times|N_\partial|\times 2} \to \mathbb{R}^{|V|}$ extract the minimum distances to upper and lower boundaries respectively:

$$\mathcal{R}_{\text{upper}}(\mathbf{T})[i] = \min_{j\in\{1,\ldots,|N_\partial|\}} \mathbf{T}[i, j, \text{upper}] \tag{17}$$

$$\mathcal{R}_{\text{lower}}(\mathbf{T})[i] = \min_{j\in\{1,\ldots,|N_\partial|\}} \mathbf{T}[i, j, \text{lower}] \tag{18}$$

and the normalisation function $\psi : \mathbb{R}^{|V|} \times \mathbb{R}^{|V|} \to [0,1]^{|V|}$ is defined element-wise as:

$$\psi(\mathbf{u}, \mathbf{l})[i] = \frac{\mathbf{u}[i]}{\mathbf{u}[i] + \mathbf{l}[i] + \epsilon} \tag{19}$$

where **u** and **l** represent the reduced upper and lower distance vectors respectively, and $\epsilon > 0$ is a small constant preventing division by zero. This operator transforms the three-dimensional augmented distance tensor into a one-dimensional normalised scalar field that smoothly transitions from 0 near lower boundaries to 1 near upper boundaries, with intermediate values modulated by the saddle point influences encoded in the bias term.

The scalar field construction proceeds through three computational stages:

(1) *Computing geodesic distance fields.* We compute geodesic distances via Dijkstra's algorithm on the weighted mesh graph, yielding three distance tensors. Let $d_g : V \times V \to \mathbb{R}_{\geq 0}$ denote the geodesic distance function. The vertex-to-boundary tensor

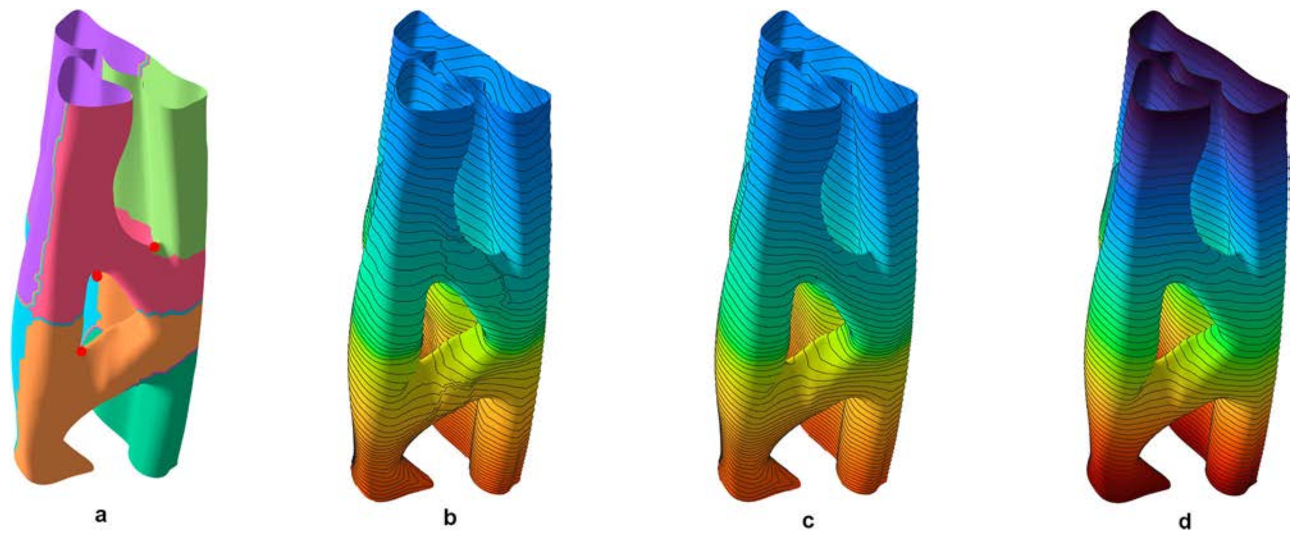

Figure 11. (a) Geodesic partition of the mesh by nearest saddle, with each saddle's influence radius $R_k = \min_{j,b} \mathbf{D}_{sn}[k, j, b]$ indicated; partitions mark where saddle guidance acts. (b–d) Influence profiles from the unnormalised weight $\tilde{W}_{ik} = csch(\sqrt{\max(\mathbf{D}_{vs}[i, k] + R_k^n - R_k, 0)})$ shown for $n = 0$, $n = 1$, and $n = 2$. After normalisation to $\mathbf{W}$ (partition of unity across saddles), larger n concentrates bias near saddle cores while smaller n spreads influence.

$\mathbf{D}_{vn} \in \mathbb{R}^{|V|\times|N_\partial|\times 2}$ stores distances from each mesh vertex to boundary vertices, indexed by boundary type $b \in \{\text{upper, lower}\}$. Similarly, $\mathbf{D}_{vs} \in \mathbb{R}^{|V|\times|S_s|}$ stores vertex-to-saddle distances, and $\mathbf{D}_{sn} \in \mathbb{R}^{|S_s|\times|N_\partial|\times 2}$ stores saddle-to-boundary distances. These tensors provide the foundation for constructing the geometry-conforming scalar field (see Appendix 1 for formal entry-wise definitions).

(2) *Computing bias incorporating saddle influences.* The bias term $\mathbf{B}$ ensures smooth transitions through saddle points by modulating how distances are perceived near these critical features. We introduce this bias so that the scalar field explicitly reflects interior topology, preventing discontinuities when slices branch or merge at saddle points. For each saddle point k, we first determine its influence radius R_k as the minimum geodesic distance to any boundary vertex.

This radius defines a region where the saddle should affect the scalar field to ensure smooth bifurcations (Figure 11(a) demonstrates the mesh partitioning with respect to the saddle points). We then construct a weight matrix $\mathbf{W} \in \mathbb{R}^{|V|\times|S|}$ that encodes how strongly each saddle influences each vertex. The unnormalised weights use the hyperbolic cosecant function to create a smooth decay from the saddle centre:

$$\tilde{W}_{ik} = \operatorname{csch}\left(\sqrt{\max(\mathbf{D}_{vs}[i, k] + R_k^n - R_k, 0)}\right) \quad (20)$$

where n (default $n = 2$, Figure 11(b–d) demonstrate the effect parameter n has on the resulting scalar field) controls the spatial decay rate, and csch provides rapid fall-off while maintaining smoothness. The term $R_k^n - R_k$ shifts the influence zone to begin at the saddle and extend approximately to radius R_k. After normalisation to ensure weights sum to unity at each vertex:

$$\mathbf{W}_{ik} = \frac{\tilde{W}_{ik}}{\sum_{k'=1}^{|S|} \tilde{W}_{ik'}} \quad (21)$$

The final bias tensor effectively pulls the computed distances toward saddle points, creating local minima in the scalar field that guide slices through bifurcations:

$$\mathbf{B}[i, j, b] = \sum_{k=1}^{|S|} \mathbf{W}[i, k] \cdot \mathbf{D}_{sn}[k, j, b] \quad (22)$$

This weighted combination ensures that vertices near saddles perceive boundaries as being closer through the saddle, naturally routeing iso-curves through these critical geometric features (Figure 11).

(3) *Field reduction and normalisation.* The augmented tensor $\mathbf{T} = \mathbf{D}_{vn} + \mathbf{B}$ undergoes reduction to produce the final normalised field $F \in [0, 1]^{|V|}$:

$$F[i] = \frac{\min_j \mathbf{T}[i, j, \text{upper}]}{\min_j \mathbf{T}[i, j, \text{upper}] + \min_j \mathbf{T}[i, j, \text{lower}] + \epsilon} \quad (23)$$

where $\epsilon > 0$ prevents division by zero.

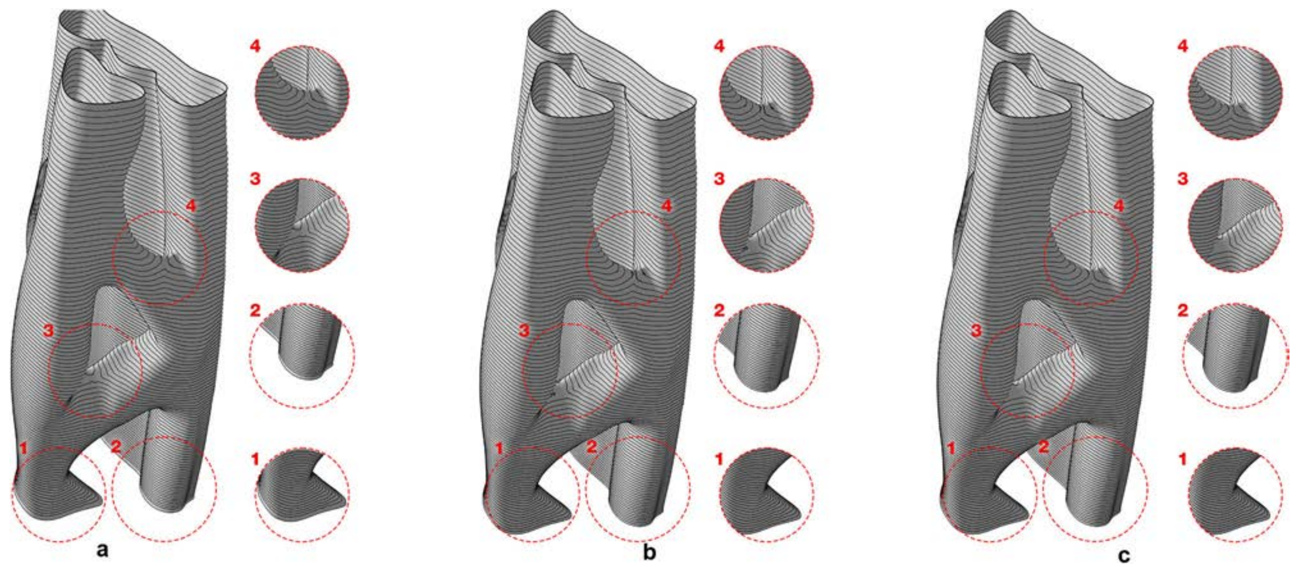

Figure 12. Comparison of slicing strategies for a branched topology-optimised geometry. (a) shows pure uniform stepping with equally spaced iso-values in the normalised field domain [0, 1], resulting in non-uniform physical layer spacing with excessive clustering in high-gradient regions (circles 1, 2) and sparse coverage in low-gradient regions (circles 3, 4), (b) demonstrates pure gradient-driven adaptive spacing that maintains approximately constant physical layer height but risks inadequate domain coverage in regions with numerical artefacts or anomalous field behaviour, potentially missing critical geometric features near boundaries, and (c) presents the hybrid adaptive method combining uniform and gradient-driven components through the weighted control law $\Delta\tau_{m+1} = (1-\alpha)\cdot\frac{1-\tau_m}{N_m} + \alpha\cdot\frac{h}{g_m}$, which achieves both complete normalised domain coverage and geometric adaptability. The adaptive approach produces well-distributed slices that respect local geometric complexity while maintaining consistent layer properties throughout bifurcations (circles 1, 2) and merging regions (circles 3, 4).

2.3. Extraction of non-planar slices

The slice extraction process employs an adaptive algorithm designed to maintain consistent physical layer spacing while accommodating complex geometry and topology. A naive extraction of iso-curves at uniform intervals in the normalised field domain, $F \in [0, 1]$, produces non-uniform physical layer spacing. This results in slice clustering in high-gradient regions and sparse coverage elsewhere. The algorithm developed in this work – *Topology-Aware Gradient-Adaptive Slicing (TAGAS)* – resolves the spacing issue through gradient-based adaptive control, which dynamically adjusts iso-value increments based on the local rate of change of the field, ensuring quasi-uniform physical layer height.

Beyond controlling spacing, the slice extraction process must robustly handle topological transitions within the geometry, such as bifurcations where a single iso value can be split between multiple disjoint components. The extraction at a given iso-level initially yields a raw, unordered collection of points where the iso-surface intersects the mesh faces. A naive assembly of these points via a nearest-neighbour traversal would fail in the presence of multiple components, as it would incorrectly connect points across distinct islands or branches, resulting in a single, topologically invalid toolpath.

To prevent this failure mode, the TAGAS algorithm implements a robust connectivity-based assembly procedure. This method first constructs a connectivity graph to cluster all raw points into topologically consistent and distinct components. Only after this grouping is complete does the algorithm proceed to order the points within each component to reconstruct the slices. This multi-stage approach guarantees topological correctness, correctly identifying separate features such as branches or internal holes before forming the final slice geometry.

2.3.1. Gradient-aware adaptive spacing

The TAGAS algorithm requires an initial estimate of the number of slicing intervals needed to maintain the desired physical layer spacing throughout the geometry. This estimate is obtained by characterising the scalar field's rate of change across the mesh through per-face gradient magnitude computation.

For each triangular face $t \in \mathcal{F}$ with vertex positions $(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k) \in \mathbb{R}^3$ and corresponding scalar field values (F_i, F_j, F_k), we compute the gradient magnitude $\|\nabla F\|_t$ using standard methods for piecewise-linear functions on triangulated surfaces [75] (see Appendix 2 for detailed equations). This magnitude represents the rate at which the normalised field changes per unit

Euclidean distance within the triangle's plane. By measuring how steeply the field changes, the algorithm identifies where finer slicing is needed to maintain uniform layer thickness. Faces with large gradient magnitudes correspond to regions where the field varies rapidly, requiring smaller iso-value increments to maintain uniform physical spacing. After computing gradient magnitudes for all faces, we characterise the global field behaviour through a robust statistical measure. Let $\mathcal{F}$ denote the set of all triangular faces in the mesh. The effective gradient g_{eff} is defined as:

$$g_{\text{eff}} = \text{median}\{\|\nabla F\|_t : t \in \mathcal{F}\} \tag{24}$$

The median is chosen over the mean to provide robustness against outliers arising from nearly degenerate triangles or numerical artefacts during gradient computation. This single scalar value summarises the typical rate of change experienced across the geometry.

Given a desired physical layer height h (measured in the same units as the mesh coordinates) and the effective gradient g_{eff}, the corresponding step size in the normalised field domain [0, 1] is approximately:

$$\Delta\tau_{\text{target}} = h \cdot g_{\text{eff}} \tag{25}$$

This relationship reflects the inverse proportionality between physical spacing and field gradient: regions with high gradients require smaller normalised increments to achieve the target physical spacing, while regions with low gradients can tolerate larger normalised steps. This proportionality ensures that slice spacing adapts naturally to geometric variation without over-sampling flat regions.

The initial number of slicing intervals N is determined by dividing the normalised field range by the target step size:

$$N_{steps} = \left\lfloor \frac{1}{\Delta\tau_{\text{target}}} \right\rfloor = \left\lfloor \frac{1}{h \cdot g_{\text{eff}}} \right\rfloor \tag{26}$$

This estimate provides a baseline for the subsequent adaptive extraction process, ensuring that the algorithm begins with a reasonable expectation of the total number of slices required. The floor operation guarantees that N is a positive integer, and the subsequent dynamic step size control (described in Section 2.3.2) refines this estimate based on local field behaviour encountered during extraction. Dynamic step size control, introduced later, refines this baseline in response to spatial variations detected during field traversal.

2.3.2. Dynamic step size control

The TAGAS algorithm employs a hybrid control strategy that dynamically adjusts the iso-value increment at each iteration to balance two competing objectives: maintaining uniform coverage of the normalised field domain and adapting to local geometric complexity. A purely uniform stepping strategy would produce N equally spaced iso-values in the normalised domain [0, 1], but this approach fails to account for the spatially varying relationship between normalised field increments and physical layer spacing. Conversely, a purely gradient-driven strategy that maintains constant physical spacing could fail to adequately cover regions where the field exhibits anomalous behaviour or numerical artefacts. The hybrid controller synthesises these approaches, ensuring both complete domain coverage and geometric adaptability (Figure 12).

To avoid reliance on global heuristics that may not hold locally, using an estimate derived from the current slice couples step sizing to the actually sampled geometry. At iteration m with current normalised position $\tau_m \in [0, 1]$, the controller must determine the next iso-value τ_{m+1} based on information extracted from the slice at level τ_m. This information is encoded in the gradient-weighted average g_m, which characterises the local rate of change experienced by the iso-curve as it traverses the geometry.

When the iso-level τ_m intersects a triangular face $t \in \mathcal{F}$, it produces a line segment within that triangle connecting two intersection points on the triangle's edges. Let $\mathbf{p}_{i,1}, \mathbf{p}_{i,2} \in \mathbb{R}^3$ denote the endpoints of segment i belonging to the set of intersection segments $\mathcal{S}_m$ extracted at iteration m. The three-dimensional Euclidean length of segment i is:

$$\begin{aligned} \ell_i &= \|\mathbf{p}_{i,2} - \mathbf{p}_{i,1}\| \\ &= \sqrt{(x_{i,2} - x_{i,1})^2 + (y_{i,2} - y_{i,1})^2 + (z_{i,2} - z_{i,1})^2} \end{aligned} \tag{27}$$

This length represents the physical extent of the intersection within the face and serves as a natural weighting factor for aggregating gradient information across the slice. Let $t_i \in \mathcal{F}$ denote the face containing segment i, with associated gradient magnitude $\|\nabla F\|_{t_i}$ computed as described in the previous section. The gradient-weighted average at iteration m is defined as:

$$g_m = \frac{\sum_{i=1}^{|\mathcal{S}_m|} \|\nabla F\|_{t_i} \cdot \ell_i}{\sum_{i=1}^{|\mathcal{S}_m|} \ell_i} \tag{28}$$

where $|\mathcal{S}_m|$ denotes the cardinality of the segment set at iteration m. This weighted average assigns greater influence to longer segments, which typically correspond to faces where the iso-curve makes a more substantial traversal. Faces that the iso-level barely intersects contribute proportionally less to the gradient estimate, providing a robust characterisation of the

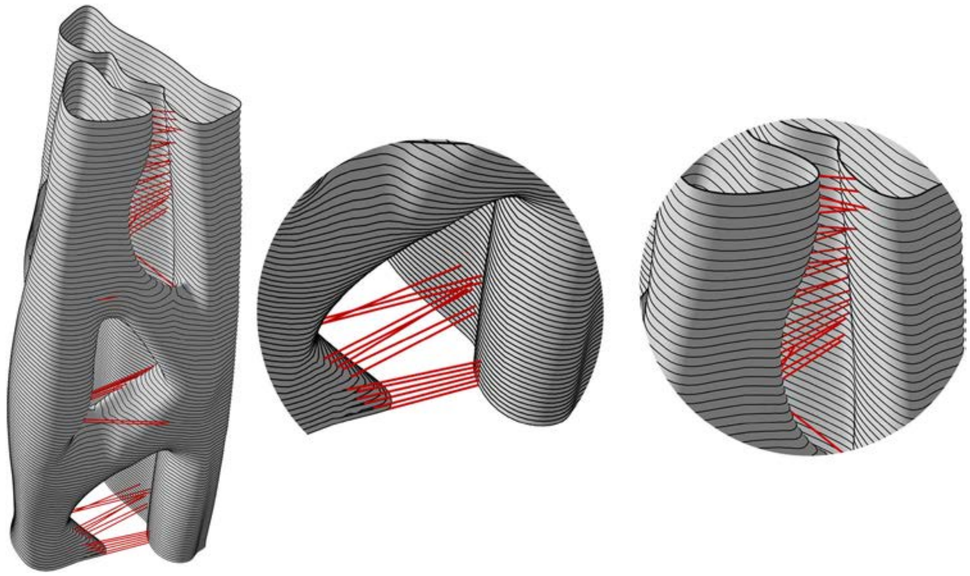

Figure 13. Failure modes of naive nearest-neighbour iso-curve assembly for branched topology-optimised geometries. Three views of the same sliced geometry demonstrate how a greedy nearest-neighbour traversal incorrectly connects intersection segments across topologically distinct regions in the presence of bifurcations and complex topology.

field's local behaviour weighted by geometric significance.

To make the estimator resilient to pathological intersections without discarding potentially informative segments, we weight by segment length: longer intersection segments generally indicate regions where the iso-curve samples a larger portion of the face's area, making the gradient estimate from that face more representative of the local field behaviour. Short segments may arise from grazing intersections near triangle vertices or from highly oblique angles between the iso-surface and the face – situations where the gradient estimate may be less reliable due to geometric or numerical sensitivity.

To guarantee that the step size remains between the two principled extremes – and to enable smooth transitions between coverage-dominant and adaptivity-dominant behaviour – we use a convex blend of two complementary strategies. Using the local gradient information encoded in g_m and the remaining normalised domain, the controller then computes the next step size.

To explicitly enforce a budget on the remaining steps – preventing undersampling of late-stage regions – we track the number of intervals remaining. Let N_m denote the number of intervals remaining to be processed after extracting the slice at level τ_m. This quantity is initialised to $N_m = \max(N - m, 1)$ where N is the total number of intervals determined from the initial gradient analysis. The remaining normalised range to be covered is $(1 - \tau_m)$, which must be distributed across the N_m remaining intervals to ensure the extraction process reaches $\tau = 1$ without premature termination.

The uniform stepping component ensures complete domain coverage by computing the step size required to traverse the remaining range in exactly N_m equal increments:

$$\Delta\tau_{\text{uniform}} = \frac{1 - \tau_m}{N_m} \tag{29}$$

This term guarantees that the algorithm will complete its traversal of the normalised field domain regardless of local gradient variations. Without this component, regions of unexpectedly low gradient could cause the algorithm to take excessively large steps that exhaust the interval budget prematurely, leaving portions of the domain unsampled. Thus, the uniform term serves as a coverage safeguard.

The adaptive component responds to local field behaviour by computing a step size inversely proportional to the measured gradient. As defined previously the normalised layer height h was defined during initialisation as $h = 1/N$ based on the effective gradient g_{eff}. This value represents the normalised step size that would achieve the desired physical layer spacing if the gradient were uniformly equal to g_{eff} throughout the geometry. The adaptive step size is:

$$\Delta\tau_{\text{adaptive}} = \frac{h}{g_m} \tag{30}$$

This term implements the fundamental relationship between normalised field increments and physical spacing: where the gradient is high (the field changes rapidly with position), smaller normalised steps are required to maintain constant physical layer height, and where the gradient is low (the field varies slowly), larger normalised steps are permissible. By scaling the baseline step h inversely with the local gradient g_m, this term adapts the extraction to the instantaneous geometric complexity encountered at each iteration.

The final step size is computed as a weighted average of these two components:

$$\begin{aligned}\Delta\tau_{m+1} &= (1-\alpha)\cdot\Delta\tau_{\text{uniform}} + \alpha\cdot\Delta\tau_{\text{adaptive}} \\ &= (1-\alpha)\cdot\frac{1-\tau_m}{N_m} + \alpha\cdot\frac{h}{g_m}\end{aligned} \tag{31}$$

where $\alpha \in [0, 1]$ is the controller blend factor. The default value $\alpha = 0.5$ provides equal weighting to both objectives, though this parameter can be adjusted to prioritise either coverage reliability or geometric adaptivity. Setting $\alpha = 0$ recovers purely uniform spacing in the normalised domain, while $\alpha = 1$ yields purely gradient-driven adaptive spacing. The term α makes the controller tuneable for different materials or hardware without altering core logic.

To prevent numerical instability and ensure robust convergence across diverse geometries, the computed step size undergoes clamping before application. The clamped step size is:

$$\Delta\tau_{m+1} = \min\big(\max\left(\Delta\tau_{m+1}, \delta_{\min}\right), \min\left(\delta_{\max}, 1-\tau_m\right)\big) \tag{32}$$

where $\delta_{\min} = 10^{-4}$ and $\delta_{\max} = 0.2$ are empirically determined bounds. The minimum bound $\delta_{\min}$ prevents the algorithm from taking arbitrarily small steps in regions of extremely high gradient, which could lead to excessive iteration counts and computational expense without meaningful improvement in slice quality. The maximum bound $\delta_{\max}$ prevents excessively large steps that might skip over important geometric features or produce slices with poor spatial resolution.

The additional constraint $\min(\delta_{\max}, 1-\tau_m)$ ensures that no step can exceed the remaining normalised range, preventing the algorithm from attempting to extract slices beyond $\tau = 1$. This safeguard is particularly important near the terminal iteration where the remaining range may be smaller than $\delta_{\max}$.

The updated position for the next iteration is computed as:

$$\tau_{m+1} = \tau_m + \Delta\tau_{m+1} \tag{33}$$

The algorithm continues this iterative process, extracting a slice at each computed iso-level and updating the step size based on the observed gradient, until the terminal condition $\tau_m \geq 1 - \delta_{\min}$ is satisfied. At this point, a final slice at $\tau = 1$ is extracted to ensure proper closure of the normalised field domain, as described in the convergence and quality assurance section.

2.3.3. Forming non-planar slices from segments of iso-curves

Forming non-planar slices proceeds by assembling/joining iso-curves at each iso-level τ in three phases: (1) triangle-level intersection to extract iso-curve segments, (2) connectivity-based clustering of these iso-curve segments to identify topologically distinct curve components, and (3) curve reconstruction with smoothing to yield the final slice geometry. This multi-phase pipeline robustly handles complex topological transitions in branched and high-genus geometries, where a single iso-level may intersect multiple disjoint regions of the mesh simultaneously.

A naive approach that attempts to assemble intersection points into a single continuous curve through nearest-neighbour traversal fails catastrophically in the presence of bifurcations, internal voids, or multiple disconnected components (Figure 13). Such an approach would incorrectly connect points across topologically distinct regions. The robust multi-segment assembly procedure described here prevents these failure modes by explicitly identifying and respecting the topological structure of the iso-surface before attempting to form the non-planar slice.

Phase 1 – Segment extraction. We detect triangle-iso intersections at level τ_m to form primitive iso-curve segments. The segment extraction phase operates on the mesh face set $\mathcal{F}$, examining each triangular face to identify edges that intersect the current iso-level $\tau_m \in [0, 1]$. For a triangle $t \in \mathcal{F}$ with vertices indexed as (a, b, c) and corresponding field values $(F_a, F_b, F_c) \in [0, 1]^3$, the algorithm tests each of the three edges for intersection with the iso-level.

Consider an edge connecting vertices a and b with positions $\mathbf{p}_a, \mathbf{p}_b \in \mathbb{R}^3$ and field values $F_a, F_b \in [0, 1]$. Define the signed distances from the iso-level as:

$$\Delta_a = F_a - \tau_m, \quad \Delta_b = F_b - \tau_m \tag{34}$$

The edge intersects the iso-level if and only if the field values bracket τ_m, which occurs when the signed distances have opposite signs:

$$\Delta_a \cdot \Delta_b < 0 \tag{35}$$

This condition ensures that the iso-level lies strictly between the two endpoint values. When this intersection condition is satisfied, we compute the exact intersection point along the edge by linearly interpolating between the two endpoint positions and field values.

For each triangle, we collect all intersection points on its edges; triangles with two intersection points contribute a single line segment representing the local portion of the iso-curve. The set of all segments extracted at iso-level τ_m is denoted S_m. The explicit interpolation formulas and notation are provided in Appendix 3.

Phase 2 – Connectivity clustering. The extracted segments $\mathcal{S}_m$ form an unordered collection of line segments that must be organised into topologically coherent curve components. This organisation is achieved through the construction of a connectivity graph that encodes spatial adjacency relationships between segments.

The algorithm constructs an undirected graph $\mathcal{G}_m = (\mathcal{V}_m, \mathcal{E}_m)$ where each vertex in $\mathcal{V}_m$ represents one of the extracted segments and each edge in $\mathcal{E}_m$ connects segments that should belong to the same curve component. Two segments $\mathbf{s}_i = (\mathbf{p}_{i,1}, \mathbf{p}_{i,2})$ and $\mathbf{s}_j = (\mathbf{p}_{j,1}, \mathbf{p}_{j,2})$ are considered connected if they satisfy either of two criteria.

The first criterion is exact endpoint matching within numerical tolerance. Segments share a common vertex in the mesh if their endpoints coincide:

$$\min\{\|\mathbf{p}_{i,1} - \mathbf{p}_{j,1}\|, \|\mathbf{p}_{i,1} - \mathbf{p}_{j,2}\|, \|\mathbf{p}_{i,2} - \mathbf{p}_{j,1}\|, \|\mathbf{p}_{i,2} - \mathbf{p}_{j,2}\|\} < \epsilon_{\text{round}} \tag{36}$$

where ϵ_{round} is a rounding tolerance (typically 10^{-6} in normalised coordinates) that accounts for floating-point arithmetic limitations. This criterion captures the common case where adjacent triangles share an edge, causing their intersection segments to meet at a shared endpoint.

The second criterion handles near-connections that may arise from numerical imprecision or mesh irregularities. Segments are considered spatially proximate if the minimum distance between any pair of endpoints falls below an adaptive threshold:

$$\min_{(i',j')\in\{1,2\}^2} \|\mathbf{p}_{i,i'} - \mathbf{p}_{j,j'}\| < \theta_m \tag{37}$$

where the spatial threshold θ_m is computed adaptively based on the characteristic scale of the intersection segments:

$$\theta_m = k_\theta \cdot \text{median}\{\|\mathbf{p}_{i,2} - \mathbf{p}_{i,1}\| : \mathbf{s}_i \in \mathcal{S}_m\} \tag{38}$$

with connectivity factor $k_\theta = 0.1$. This adaptive threshold ensures that the connectivity analysis scales appropriately with the local geometry, remaining sensitive to small gaps while avoiding spurious connections between distant segments.

For each pair of segments satisfying either connectivity criterion, an edge is added to $\mathcal{E}_m$, establishing their adjacency relationship. The resulting graph $\mathcal{G}_m$ partitions the segment set $\mathcal{S}_m$ into connected components through standard graph analysis. Depth-first search (DFS) or breadth-first search (BFS) identifies the connected components:

$$\{\mathcal{C}_{m,1}, \mathcal{C}_{m,2}, \ldots, \mathcal{C}_{m,k_m}\} \tag{39}$$

where k_m denotes the number of distinct curve components detected at iteration m, and each component $\mathcal{C}_{m,j} \subseteq \mathcal{S}_m$ is a maximal connected subset of segments. The primary component (largest by segment count or total length) typically represents the main iso-curve, while secondary components correspond to internal holes, isolated islands, or branches that have fully separated at the current iso-level.

Phase 3 – Geometric reconstruction (ordering). Within each connected component $\mathcal{C}_{m,j}$, the constituent segments must be ordered to form a continuous polyline suitable for downstream processing. This ordering step transforms the unordered segment set into a sequential point list that traces the curve's path through three-dimensional space.

The ordering algorithm begins by analyzing the connectivity structure within component $\mathcal{C}_{m,j}$. For each segment endpoint, the algorithm counts the number of adjacent segments that share that endpoint. An endpoint with exactly one adjacent segment is classified as a terminus, indicating that the curve is open and terminates at that location. An endpoint with exactly two adjacent segments is interior to the curve. Endpoints with degree greater than two indicate topological anomalies (typically numerical artefacts) and trigger special handling or component rejection.

For an open curve component (containing two terminus endpoints), the traversal begins at one terminus and follows the unique chain of adjacent segments until reaching the opposite terminus:

$$\{\mathbf{q}_{j,1}, \mathbf{q}_{j,2}, \ldots, \mathbf{q}_{j,n_j}\} \tag{40}$$

where $n_j = |\mathcal{C}_{m,j}| + 1$ is the number of distinct points (one more than the number of segments due to endpoint sharing). For a closed curve component (all endpoints have degree two), the traversal begins at an arbitrary segment and follows the cycle until returning to the starting point, with the first point repeated at the end to explicitly close the loop.

The traversal maintains a visited set to prevent infinite loops and ensures each segment is processed exactly once. At each step, the algorithm identifies the unique unvisited segment adjacent to the current endpoint and appends its opposite endpoint to the ordered point sequence. This greedy traversal succeeds when the component has valid curve topology (degree two

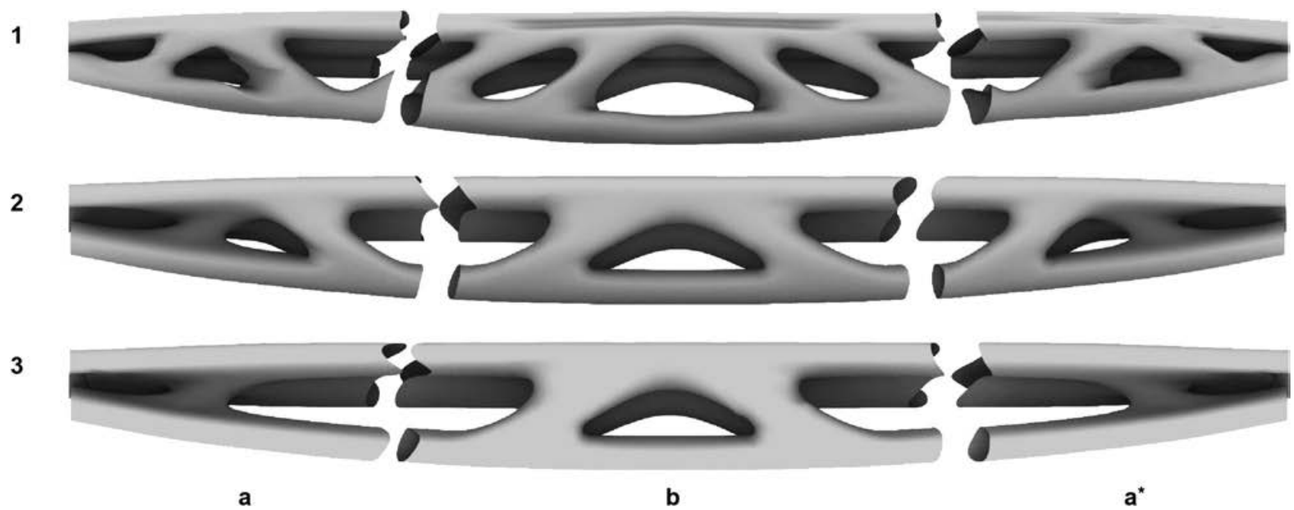

Figure 14. Segmented topology-optimised beams with varying levels of complexity and genus numbers. Each row represents a different beam design with genus numbers of 19 (top), 14 (middle), and 9 (bottom). The beams are divided into three segments (a, b, a*), with segments (a) and (a*) being symmetrically identical.

at all interior points, degree one at endpoints for open curves).

Phase 4 – Quality filters. We discard numerically spurious or vanishing components using point-count and length thresholds. Before proceeding to smoothing, the algorithm applies filtering criteria to remove spurious components arising from numerical artefacts or mesh irregularities. A component $\mathcal{C}_{m,j}$ is retained only if it satisfies:

$$n_j \geq n_{\min} \quad \text{and} \quad \ell_j = \sum_{i=1}^{n_j-1} \|\mathbf{q}_{j,i+1} - \mathbf{q}_{j,i}\| > \epsilon_{\text{length}} \tag{41}$$

where $n_{\min} = 3$ is the minimum point threshold and $\epsilon_{\text{length}} = 10^{-6}$ (in mesh units) is the minimum acceptable curve length. These criteria eliminate components that are too small to represent meaningful geometric features while preserving all significant curve elements.

Phase 5 – Smoothing. The ordered point sequences produced by segment traversal exhibit discretization noise arising from the piecewise-linear approximation of the iso-surface on the triangulated mesh. To produce smooth curves while preserving essential geometric features, the algorithm applies Savitzky-Golay filtering independently to each spatial coordinate.

For component $\mathcal{C}_{m,j}$ with ordered points $\{\mathbf{q}_{j,1}, \mathbf{q}_{j,2}, \ldots, \mathbf{q}_{j,n_j}\}$ where $\mathbf{q}_{j,i} = (x_{j,i}, y_{j,i}, z_{j,i}) \in \mathbb{R}^3$, the Savitzky-Golay filter operates by fitting a polynomial of degree p to a sliding window of w consecutive points centred at each interior point [76]. The window size w must be odd to ensure symmetric treatment; when $n_j < w$, the window size is reduced to the largest odd integer not exceeding n_j.

For each interior point index i satisfying $\lceil w/2 \rceil \leq i \leq n_j - \lfloor w/2 \rfloor$, the smoothed position $\tilde{\mathbf{q}}_{j,i}$ is computed as a weighted linear combination of points in the local neighbourhood:

$$\tilde{\mathbf{q}}_{j,i} = \sum_{\delta=-\lfloor w/2 \rfloor}^{\lfloor w/2 \rfloor} c_\delta \mathbf{q}_{j,i+\delta} \tag{42}$$

where the coefficients $\{c_\delta\}_{\delta=-\lfloor w/2 \rfloor}^{\lfloor w/2 \rfloor}$ are derived from least-squares fitting of a polynomial of degree p over the window indices. The Savitzky-Golay coefficients are computed analytically from the Moore-Penrose pseudoinverse of the Vandermonde matrix associated with the polynomial basis, ensuring optimal noise reduction while preserving polynomial features up to degree p.

The standard configuration uses window size $w = 7$ and polynomial degree $p = 2$, which provides effective smoothing of discretization noise while preserving quadratic curvature features. This choice balances noise reduction against feature preservation, avoiding the over-smoothing that would result from larger windows or lower-degree polynomials.

Points near the curve endpoints (within distance $\lfloor w/2 \rfloor$ from either end) cannot be centred in a full window and are handled through asymmetric windowing or by leaving them unsmoothed. For closed curves, the filtering wraps around the periodic boundary, treating the point sequence as cyclic.

Phase 6 – Resampling. The final reconstruction step resamples each smoothed curve to a uniform distribution

Table 1. Distribution of genus numbers across segments for the three test beams.

	Beam(1)	Beam(2)	Beam(3)
Segment(a)	5	4	2
Segment(b)	9	5	5
Segment(a*)	5	4	2

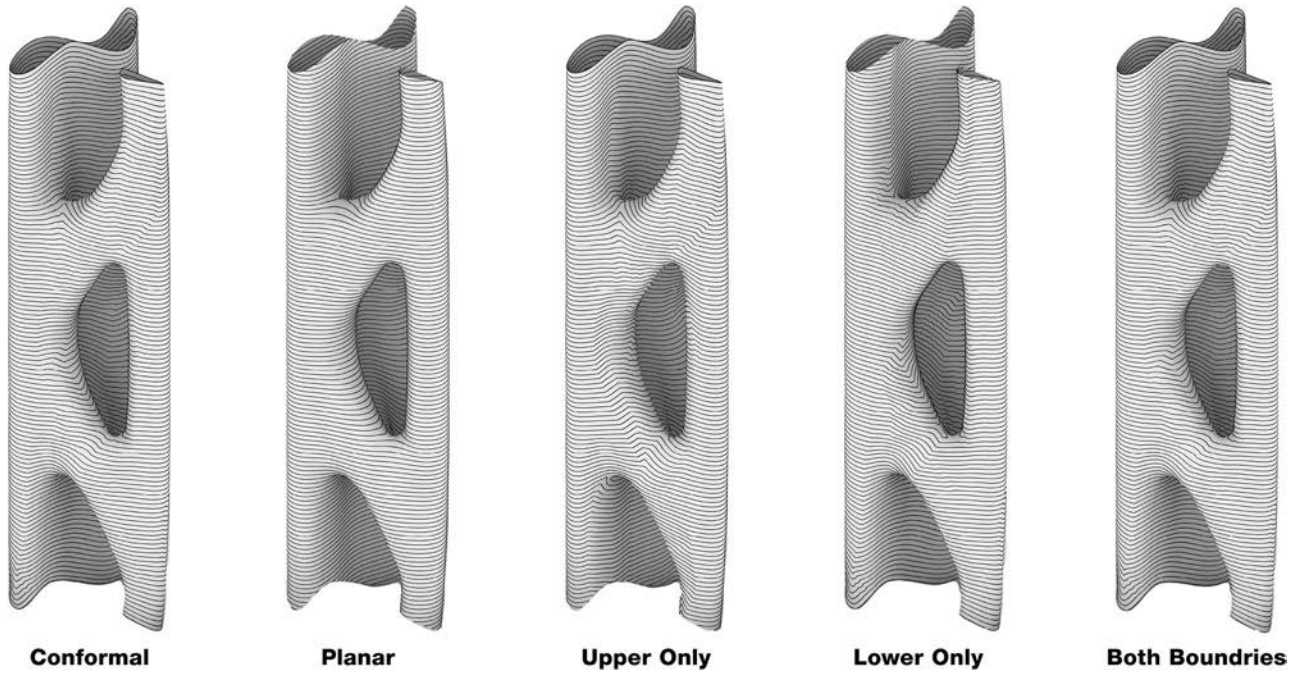

Figure 15. Comparison of slicing outputs for segment 3b from the five evaluated methods: Conformal, Planar, Upper Only, Lower Only, and Both Boundaries.

of points along the arc length using standard arc-length parameterisation, facilitating downstream geometric operations and toolpath generation. Given a target sample count s (default 200), the algorithm computes the cumulative arc length function for each component $\mathcal{C}_{m,j}$ with points $\{\tilde{\mathbf{q}}_{j,1}, \tilde{\mathbf{q}}_{j,2}, \ldots, \tilde{\mathbf{q}}_{j,n_j}\}$, and evaluates it at uniformly spaced arc length parameters.

For each target arc length, binary search identifies the containing segment, and linear interpolation yields the resampled point. The final resampled component consists of s uniformly spaced points, providing a consistent parameterisation suitable for B-spline fitting, curvature analysis, and toolpath generation. For closed curves, the resampling ensures that the first and last points coincide to maintain explicit closure. (See Appendix 4. for detailed equations.)

The complete set of reconstructed components at iteration m forms the geometric representation of the iso-slice at level τ_m, with each component characterised by its resampled point sequence, arc length, closure status, and topological relationship to other components.

3. Geometry-conforming non-planar slicing for topology-optimised beams: a comparative study

3.1. Experimental design

To evaluate the performance of the proposed geometry-conforming non-planar slicing algorithm, we conducted a systematic analysis against four alternative slicing strategies – (a) Conventional Planar Slicing, (b) Upper Boundary Non-Planar Slicing, (c) Lower Boundary Non-Planar Slicing, and (d) Boundary-Only Blended Field without the bias term. These comparisons assess the algorithm's ability to handle complex topological transitions while maintaining geometric fidelity in high-genus structures typical of materially optimised reinforced concrete elements.

The evaluation is conducted on three topology-optimised beams, each sharing identical bounding dimensions (length = 3.15 m, width = 0.3 m, depth = 0.4 m) while exhibiting distinct topological complexities (i.e. multiple internal voids, branche's angel, and non-uniform connectivity), with genus numbers of 19, 14, and 9, respectively. Each beam was segmented into three equal parts along the longitudinal axis considering the typical printing bounding area of robotic 3DP for large-scale manufacturing, yielding nine segments total.

Segmenting the beam is not dictated by the algorithm but by project-level constraints, and the interface between segments may be planar, non-planar, and non-coplanar. Interfaces are typically laid out with respect to internal force flow (to route reinforcement/post-tensioning and avoid stress concentrations), manufacturing limitations (robot reach, nozzle orientation, build envelope), and assembly logistics (handling, tolerances, transport). Segmentation is independent of our contribution; the method applies under any reasonable segmentation scheme. The proposed method operates locally on each segment and generates geometry-conforming non-planar slicing regardless of the interface geometry or the specific segmentation scheme.

In this study, we split each beam into three sections along its span to make the comparisons consistent. Due to structural symmetry, segments (a) and (a*) within each beam are geometrically identical, resulting in six unique test geometries (Figure 14). Table 1 details the genus distribution across segments, demonstrating how topological complexity varies both within and between beam designs. The selection of these test geoemtries ensures comprehensive coverage of challenging geometric features encountered in MORCE, including multiple bifurcations where single branches split into multiple paths, merging regions where separate branches converge, varying branch lengths that create asymmetric load distributions, and non-coplanar interfaces critical for accurate assembly of segmented components.

3.2. Comparative slicing methods

The evaluation framework compares five distinct slicing approaches to assess the relative performance of the proposed algorithm:

(1) *Geometry-Conforming Non-Planar Method*: The full implementation incorporating saddle point detection through discrete Morse theory, weighted bias computation, and adaptive iso-curve extraction as detailed in the methodology section.
(2) *Conventional Planar Slicing*: Traditional layer-wise slicing with planes perpendicular to the z-axis, representing the current industry standard for 3D concrete printing applications.
(3) *Upper Boundary Non-Planar Slicing*: Non-planar slicing using geodesic distances computed exclusively from upper boundary vertices ($\mathbf{D}_{vn}$[upper]), neglecting both lower boundaries and internal critical features.
(4) *Lower Boundary Non-Planar Slicing*: Non-planar slicing based solely on geodesic distances from lower boundary vertices ($\mathbf{D}_{vn}$[lower]), similarly omitting upper boundaries and saddle points.
(5) *Boundary-Only Blended Field*: Non-planar slicing using the normalised scalar field $\mathcal{F}(\mathbf{D}_{vn})$ that combines upper and lower boundary influences without incorporating the saddle point bias term $\mathbf{B}$, effectively implementing Equation (15) without the weighted bias computation described in Section 2.2.

3.3. Implementation parameters

All slicing methods were executed using consistent parameters to ensure fair comparison. The key configuration settings included a target physical layer height of 15 mm, appropriate for concrete extrusion systems with typical nozzle diameters of 20–25 mm [3,25,38]. The adaptive controller blend factor was set to $\alpha = 0.5$, providing equal weighting between uniform domain coverage and gradient-driven adaptivity. Each iso-curve was uniformly resampled to 1800 points to ensure adequate resolution for downstream toolpath generation while maintaining computational efficiency.

Additional parameters included an edge intersection tolerance of 10^{-6} for robust segment extraction, dynamic step size clamping between 10^{-4} and 0.2 in the normalised domain, and a minimum component size threshold of 3 points to filter numerical artefacts. The boundary classification used a tolerance 5% to distinguish the upper and lower boundaries based on their mean positions along the z-axis (Figure 15).

3.4. Evaluation metrics

Our objective is geometric conformity, evaluated in slicing without physical printing. To isolate the algorithmic effect from hardware/material confounders, we use two geometric conformity-oriented metrics. First, we analyze normalised effective layer-height gradients: low gradients indicate that successive slices adhere smoothly to the target surface rather than departing abruptly – i.e. local geometric conformity that is machine-agnostic. Second, we measure interface boundary conformance by verifying closed, interface-respecting curves across segments – i.e. global conformity to prescribed boundaries. These metrics are scale-free, do not assume printer-specific parameters, and thus provide a fair basis for comparing slicing strategies in the absence of fabrication tests. While our evaluation is simulation-based, future work will include print trials to correlate these geometry-level metrics with process outcomes (bead stability, gaps, and assembly tolerances).

3.4.1. Metric 1: normalised effective layer height gradient analysis

The first metric quantifies the smoothness of layer transitions through analysis of effective layer height variations, critical for maintaining geometric adherence between successive slices. For each point $\mathbf{p}_i$ on slice s_k, the effective layer height $h_{\text{eff}}(\mathbf{p}_i)$ represents the physical

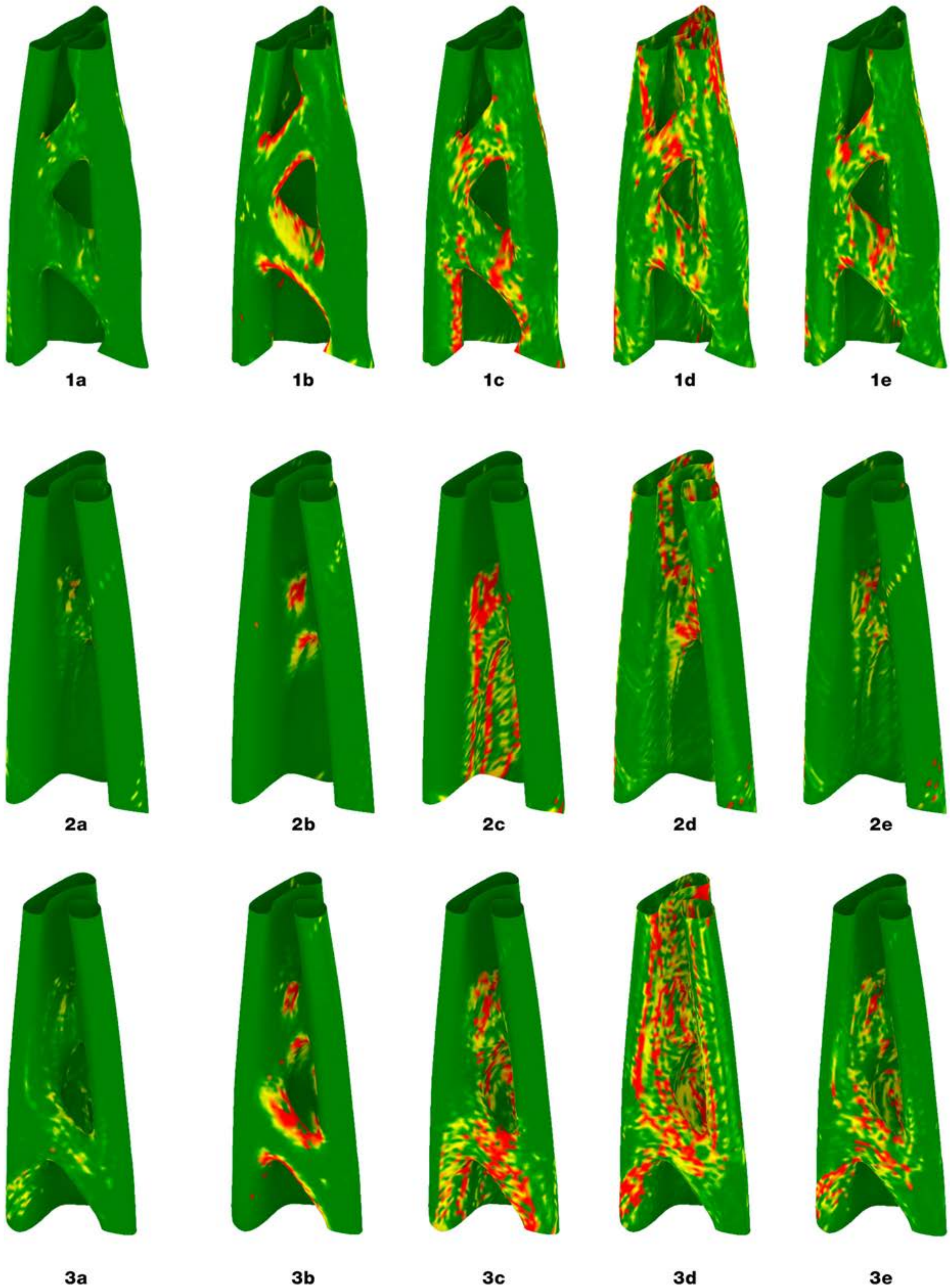

Figure 16. Visualization of the normalised effective layer height gradient for the outer beam segments (1a, 2a, 3a). Columns correspond to the slicing methods: (a) Conformal, (b) Planar, (c) Upper Only, (d) Lower Only, and (e) Both Boundaries. The colour scale on the heatmap represents the magnitude of the normalised gradient, ranging from 0 (green) for smooth layer transitions to 1 (red) for abrupt changes that may lead to fabrication defects.

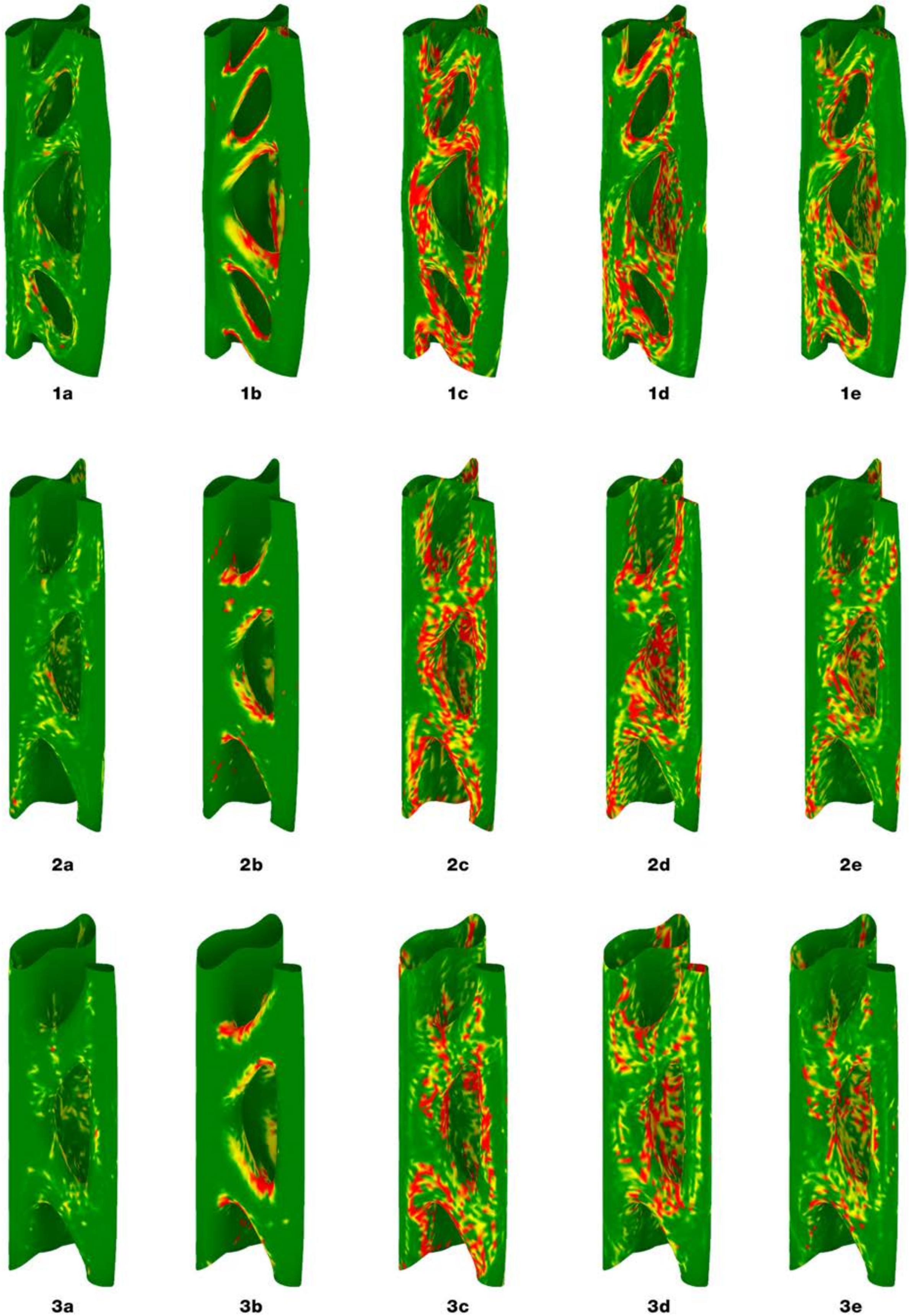

Figure 17. Visualization of the normalised effective layer height gradient for the topologically complex central beam segments (1b, 2b, 3b). Columns correspond to the slicing methods as in Figure 16. The colour scale on the heatmap represents the magnitude of the normalised gradient, ranging from 0 (green) for smooth layer transitions to 1 (red) for abrupt changes. The proposed Conformal method (a) maintains low gradients, whereas other methods exhibit significant high-gradient regions, particularly at bifurcations.

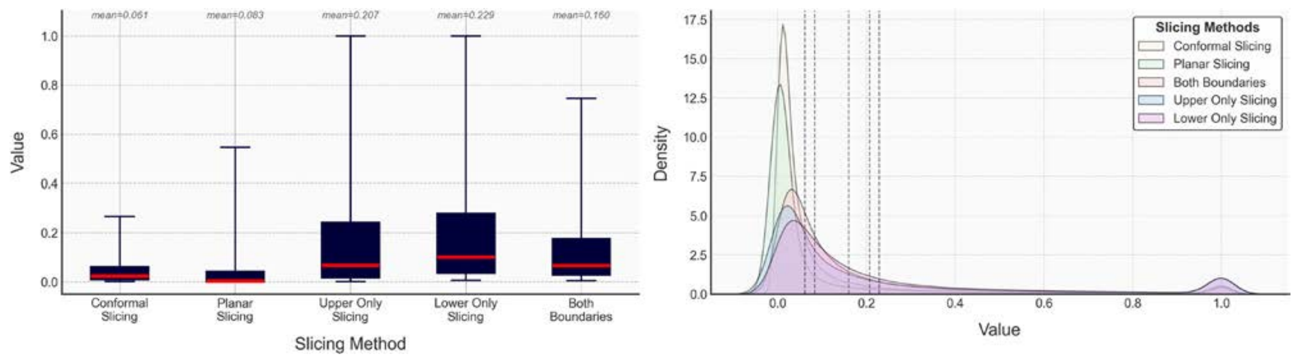

Figure 18. Statistical analysis of normalised effective layer height gradient distributions across all test segments. The box plot (left) and kernel density estimate (right) illustrate the median, variance, and distribution shape for each slicing method.

distance to the supporting material on the previous slice s_{k-1}.

The computation proceeds by constructing the binormal vector $\mathbf{b}_i$ at each point through the cross product of the interpolated surface normal $\mathbf{n}_i$ and the normalised tangent vector $\mathbf{t}_i$ along the slice curve:

$$\mathbf{b}_i = \mathbf{n}_i \times \mathbf{t}_i \tag{43}$$

The effective layer height is then determined by ray-casting along the binormal direction to find the intersection with the previous slice. Formally, let S_{k-1} denote the set of points on the previous slice s_{k-1} and let $\mathbf{b}_i$ be the unit binormal at point $p_i \in s_k$. We define the effective layer height as the minimal positive offset along the binormal direction:

$$h_{\text{eff}}(p_i) = \min_{q \in S_{k-1}(q-p_i)\cdot \mathbf{b}_i > 0} (q - p_i) \cdot \mathbf{b}_i,$$

i.e. the shortest distance from p_i to the previously deposited material measured along $\mathbf{b}_i$.

Non-planar slicing inherently produces spatially varying layer heights to accommodate geometric complexity. Therefore, rather than measuring absolute uniformity, we compute the gradient of effective layer height to identify discontinuous transitions that indicate potential loss of geometric conformity.

Table 2. Interface boundary conformance scores for different slicing methods across all test segments.

Segment	Conformal	Planar	Upper Only	Lower Only	Both Boundaries
1a	1.0	0.85	0.85	0.90	1.0
1b	1.0	0.85	0.92	0.90	1.0
2a	1.0	0.86	0.87	0.92	1.0
2b	1.0	0.84	0.90	0.91	1.0
3a	1.0	0.92	0.88	0.93	1.0
3b	1.0	0.88	0.90	0.92	1.0

The local gradient magnitude ∇h_i at point $\mathbf{p}_i$ is computed using finite differences:

$$\nabla h_i = \frac{|h_{\text{eff}}(\mathbf{p}_{i+1}) - h_{\text{eff}}(\mathbf{p}_i)|}{\|\mathbf{p}_{i+1} - \mathbf{p}_i\|} \tag{44}$$

To enable meaningful comparison across different scales and regions of the geometry, the gradient values are normalised with respect to the mean effective layer height along each slice. The normalised gradient $\bar{\nabla} h_i$ is computed as:

$$\bar{\nabla} h_i = \frac{\nabla h_i}{\bar{h}_{\text{eff}}} \tag{45}$$

where $\bar{h}_{\text{eff}}$ represents the mean effective layer height for the current slice. This normalisation ensures that gradient measurements remain scale-invariant and comparable across regions with different characteristic layer heights, providing a dimensionless metric that quantifies relative changes rather than absolute variations. High normalised gradient values indicate regions where geometric conformity degrades and where surface tracking may become unreliable.

3.4.2. Metric 2: interface boundary conformance

The second metric evaluates the slicing algorithm's ability to maintain closed curves throughout the segment, which is essential for watertight, interface-consistent contours and topological continuity. This metric directly quantifies the occurrence of open curves that represent failures in maintaining topological integrity.

For each segment and slicing method, we compute the open curve ratio:

$$\mathcal{R}_{\text{open}} = \frac{N_{\text{open}}}{N_{\text{total}}} \tag{46}$$

where N_{open} represents the number of open curves (incomplete loops that do not return to their starting point) and N_{total} denotes the total number of curves generated across all slices in the segment. The interface

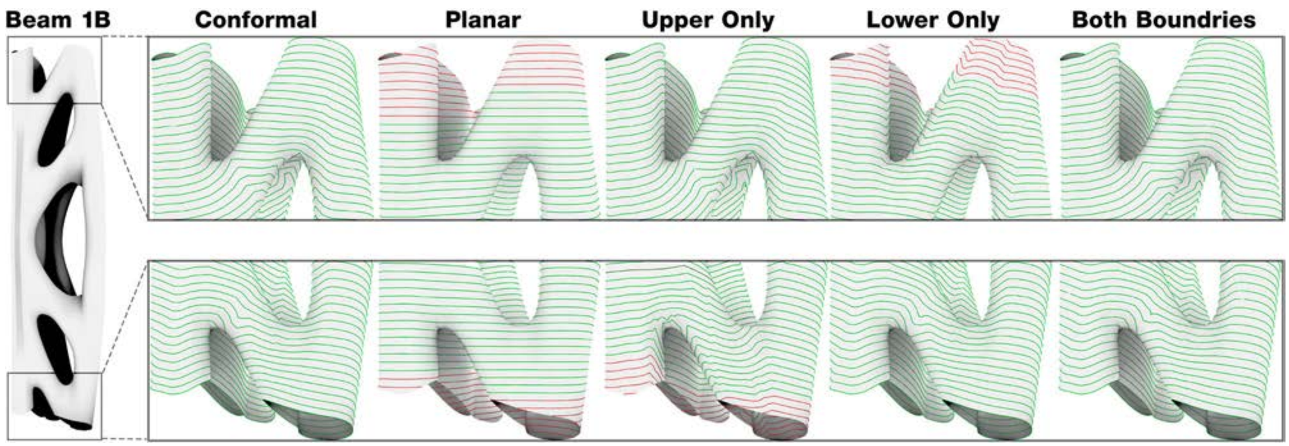

Figure 19. Qualitative comparison of slicing outputs at the interfaces within a complex bifurcation and merging region of Beam 1, Segment b. The visualisation uses colour to indicate topological integrity: closed curves are shown in green, while topologically incorrect open curves are highlighted in red.

boundary conformance score is then:

$$\mathcal{B}_{\text{conformance}} = 1 - \mathcal{R}_{\text{open}} \tag{47}$$

This formulation produces a score ranging from 0 to 1, where a score of 1 indicates perfect conformance with all curves properly closed, while lower scores quantify the proportion of topological failures. Open curves are particularly problematic at segment interfaces, where they create gaps and topological discontinuities relative to prescribed interfaces.

The metric provides direct insight into each method's ability to maintain topological consistency. Methods that explicitly incorporate boundary information in their scalar field construction should achieve higher conformance scores, while methods that neglect boundary constraints or rely solely on partial geometric features are expected to produce more open curves. This straightforward metric complements the gradient analysis by ensuring that slices maintain the topological properties required for geometry-conforming, interface-consistent slice sets.

4. Results

The comparative evaluation of the five slicing methods across six unique segment geometries reveals substantial performance differences in both layer height consistency and topological integrity. The geometry-conforming non-planar slicing method consistently outperformed all alternative approaches across both quantitative metrics.

The analysis of the normalised effective layer height gradient indicates that the proposed conformal method yields significantly more uniform layer transitions compared to the alternative non-planar approaches. Visualizations of the gradient fields are presented in Figures 16 and 17. For all tested segments, the conformal method (column a in both figures) consistently produced low gradient values, represented by a predominantly green heatmap. This uniformity persists even in regions of high geometric complexity, such as the bifurcations and mergers characteristic of the central beam segments (Figure 17, 1b, 2b, 3b).

In contrast, the other slicing strategies exhibited pronounced regions of high gradient values (yellow and red), indicating abrupt and non-conforming changes in effective layer height. Conventional planar slicing (column b) generated high gradients at curved surfaces and overhangs, consistent with planar slicing artefacts (staircase). The non-planar methods relying on partial boundary information – Upper Only (column c) and Lower Only (column d) – produced widespread, erratic high-gradient zones, reflecting a poor fit to the overall geometry. The Both Boundaries method (column e) showed improvement over the single-boundary approaches but still contained notable high-gradient areas around critical features that the proposed method's bias term reduces non-conforming gradients and lowers gradient magnitudes around critical features.

The statistical distributions of the normalised gradient, depicted in Figure 18, corroborate these findings. The box plot reveals that the conformal method achieves a low median gradient (0.027), comparable to that of planar slicing (0.023). However, its interquartile range is substantially smaller than those of the other non-planar methods (Upper Only, Lower Only, Both Boundaries), signifying a higher degree of consistency. The kernel density plot further illustrates this, showing the conformal method's distribution is tightly concentrated near zero, whereas the other non-planar methods have wider distributions with heavier tails, indicating a higher frequency of large, undesirable gradients. A low median combined with a small IQR and light tails implies uniformly geometry-adherent slices across the entire geometry – not merely on average – reducing local departures from the target surface and minimising interface-related topological glitches.

The interface boundary conformance metric assesses each method's ability to maintain closed curves throughout the segment, essential for accurate assembly. Table 2 presents conformance scores distinguishing between outer segments (1a, 2a, 3a) and central segments (1b, 2b, 3b). The conformal method and both-boundaries method both achieve perfect scores of 1.0 across all segments, ensuring every slice forms properly closed curves for precise assembly without gaps. Planar slicing demonstrates the poorest performance, with scores ranging from 0.84 to 0.92. Segment 1b achieves only 0.85 conformance, indicating that fifteen percent of slices fail to form closed curves. Single-boundary methods show intermediate performance, with upper-only achieving 0.85–0.88 in outer segments and lower-only performing slightly better at 0.90–0.92. Performance converges in central segments (0.87–0.93 range) as both methods struggle with increased topological complexity. Lower conformance implies frequent topological breaks – e.g. at 0.85, roughly one in seven slices is open – propagating gaps and misalignments along the segment and undermining interface conformity precisely where geometric complexity is highest.

Figure 19 visualises the open curve phenomenon for segment 1b. The conformal method generates exclusively closed curves throughout the segment, preserving topological continuity across nine internal holes. By contrast, the planar method produces multiple open curves (shown in red) where horizontal planes fail to capture the effects of the non-planar boundaries.

While the both-boundaries method achieves perfect conformance through bilateral boundary information, comparison with Figure 17 reveals that boundary information alone does not guarantee optimal layer height consistency. The conformal method's superior gradient performance demonstrates that explicit incorporation of saddle point information provides geometric refinement beyond boundary constraints alone.

5. Conclusion and discussion

We present an analytical, fully automated non-planar slicing algorithm that fills a critical gap in compatible slicing for 3D printing of geometrically complex, surface-based structures, with a focus on Materially Optimized Reinforced Concrete Elements (MORCE). Our core contribution is an algorithm that automatically constructs a scalar field capturing the critical features of complex geometry and provides procedures to derive non-planar slices from that same, consistent field. The method leverages discrete Morse theory to identify saddle points and integrates their influence into a geodesic-distance–based field via a novel bias term.

The results of our comparative study demonstrate the clear superiority of the proposed conformal method. It consistently produced slices with the smoothest layer height transitions, minimising the abrupt gradient changes that often lead to fabrication failures in other methods. Furthermore, it guaranteed topologically correct, closed-loop slices across all tested high-genus geometries, a critical requirement for the assembly of segmented structural components. In contrast, conventional planar slicing created problematic overhangs, while other non-planar approaches that lack a comprehensive understanding of the geometry's critical features resulted in distorted layers and topological failures.

These findings open significant avenues for 3D-printing components with voids for inserts – rebar, conduits, insulation, post-tensioning tendons, cables, sensors – both for reinforced concrete and for other materials. The presented evidence – low, tightly distributed effective layer-height gradients and closed, interface-respecting contours – indicates improved geometric conformity around saddle points and cavities, which could translate to reduced waste and better structural integration. We leave empirical verification of these implications to future print trials.

Future work can proceed along computational and process-integrated directions. Computationally, extensions could include (i) broader validation on diverse topology-optimised forms – beams, shells, and branching nodes – with a standardised benchmark (models, toolpaths, G-code, and metrics such as geometric error, path continuity, and curvature bounds); (ii) scalability and robustness via GPU/parallel implementations for iso-curve extraction, clustering, and smoothing, accompanied by formal complexity analysis and explicit handling of degeneracies and non-manifold cases; and (iii) integration with topology optimisation framewrok through a bilevel scheme in which stress fields seed iso-levels while manufacturability constraints (curvature limits, reachability, collision-free kinematics, support-free deposition) are enforced at the lower level. Open-sourcing code and datasets could also enable reproducible comparisons and data-driven 'suggestive' slicers trained on geometry–toolpath pairs.

On the materials/process side, future work can embed process constraints directly in the slicer's objective (e.g. cooling/setting windows, shrinkage, rheologyderived overhang limits) and evaluate the approach across concrete, clay, continuous-fibre thermoplastics, WAAM metals, and cementitious inks with admixtures. Physical prototyping with robotic 3DCP can assess path fidelity and build stability, including closed-loop control (height sensing, bead-width monitoring) to adapt layer height and speed, mitigate sagging, and limit cumulative drift. Together, these directions move the method from geometric proof-of-concept toward a reliable, validated, scalable, and fabrication-ready workflow for manufacturing surface-based, materially optimised structural elements.

Author contributions

CRediT: **Mania Aghaei Meibodi:** Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing; **Abdallah Kamhawi:** Conceptualization, Data curation, Formal analysis, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing; **Ashkan Foroughi Dehnavi:** Data curation, Formal analysis, Methodology, Validation, Writing – original draft; **Yichuan Li:** Data curation, Formal analysis, Methodology, Validation, Visualization, Writing – original draft.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This research was funded by the National Science Foundation CAREER Award [NSF CAREER-2441176 CMMI] to Dr. Mania Aghaei Meibodi and University of Michigan's Pressing Matters grants to Dr. Mania Aghaei Meibodi.

Data availability statement

The data that support the findings of this study are openly available at: https://github.com/DART-Research/NP-3DP

References

[1] Aghaei Meibodi M, Jipa A, Giesecke R, et al. Smart slab: Computational design and digital fabrication of a lightweight concrete slab. In: Acadia 2018 Recalibration: On Imprecision and Infidelity: Proceedings of the 38th Annual Conference of the Association for Computer Aided Design in Architecture. Mexico City, Mexico: Association for Computer Aided Design in Architecture (ACADIA); 2018. p. 434–443.

[2] Jipa A, Aghaei Meibodi M, Giesecke R, et al. 3D-printed formwork for prefabricated concrete slabs. In: 1st International Conference on 3D Construction Printing (3DcP). ETH Zurich: Digital Building Technologies (ITA); 2018.

[3] Lin Y, Bayramvand A, Aghaei Meibodi M. Towards lightweight structure: coupling topology optimization with non-planar 3D concrete printing. In: Proceedings of the 2024 DigitalFUTURES. Shanghai, China: Springer; 2024.

[4] Aghaei Meibodi M, Lin Y, Chen H. Hybrid approaches towards 3D concrete printing for lightweight reinforced concrete structures. In: RILEM International Conference on Concrete and Digital Fabrication. Hong Kong, China: Springer; 2024.

[5] Gheorghe A. A glimpse into the future: 3D printing for structural components. ser. The Art of Building, Splendid Isolation #11. Architektur Aktuell, 2017, no. 444.

[6] Chatham House. Making concrete change: innovation in low-carbon cement and concrete. 2018. [accessed 2024 Jul 20]. [Online]. Available from: https://www.chathamhouse.org/2018/06/making-concrete-change-innovation-low-carbon-cement-and-concrete

[7] Princeton University. Cement and concrete: the environmental impact. 2020. [accessed 2024 Jul 20]. [Online]. Available from: https://psci.princeton.edu/tips/2020/11/3/cement-and-concrete-the-environmental-impact

[8] Belaïd F. How does concrete and cement industry transformation contribute to mitigating climate change challenges? Resour Conserv Recycl Adv. 2022;15:200084. Available from: https://www.sciencedirect.com/science/article/pii/S2667378922000220

[9] Singla V. Cut carbon and toxic pollution: make cement clean and green. 2021. [accessed 2024 Jul 20]. [Online]. Available from: https://www.nrdc.org/bio/veena-singla/cut-carbon-and-toxic-pollution-make-cement-clean-and-green

[10] United Nations. Cutting carbon emissions from concrete will be critical to global climate action. 2019. [accessed 2024 Jul 20]. [Online]. Available from: https://news.un.org/en/story/2019/03/1035161

[11] Quint R. Young adults & the construction trades. HousingEconomics.com, National Association of Homebuilders; 2017.

[12] The Guardian. Concrete: The most destructive material on earth. 2019. [accessed 2024 Jul 20]. [Online]. Available from: https://www.theguardian.com/cities/2019/feb/25/concrete-the-most-destructive-material-on-earth

[13] Hossain MA, Zhumabekova A, Paul SC, et al. A review of 3D printing in construction and its impact on the labor market. Sustainability. 2020;12(20):8492, 1–21. doi: 10.3390/su12208492

[14] Jaillon L, Poon C-S, Chiang YH. Quantifying the waste reduction potential of using prefabrication in building construction in Hong Kong. Waste Manag. 2009;29(1):309–320. doi: 10.1016/j.wasman.2008.02.015

[15] Jiang L, Li Z, Li L, et al. A framework of industrialized building assessment in China based on the structural equation model. Int J Environ Res Public Health. 2018;15(8):1687, 1–15. doi: 10.3390/ijerph15081687

[16] Lachimpadi SK, Pereira JJ, Taha MR, et al. Construction waste minimisation comparing conventional and precast construction (mixed system and IBS) methods in high-rise buildings: a Malaysia case study. Resour Conserv Recycl. 2012;68:96–103. doi: 10.1016/j.resconrec.2012.08.011

[17] Law KK, Chang S, Siu M-FF. Factors influencing adoption of construction robotics in Hong Kong's industry: a multistakeholder perspective. J Manage Eng. 2022;38(2):04021096. doi: 10.1061/(ASCE)ME.1943-5479.0001011

[18] CBS News. Cement industry CO2 emissions: Brimstone's innovative approach to climate change. 2023. [accessed 2024 Jul 20]. [Online]. Available from: https://www.cbsnews.com/news/cement-industry-co2-emissions-climate-change-brimstone/

[19] Lamb Z. The world needs to build more than two billion new homes over the next 80 years. 2018. [accessed 2024 Jul 20]. [Online]. Available from: https://theconversation.com/the-world-needs-to-build-more-than-two-billion-new-homes-over-the-next-80-years-91794

[20] United Nations. Department of Economic and Social Affairs, Population Division: World population prospects 2017. 2017. [accessed 2024 Jul 20]. [Online]. Available from: https://www.un.org/development/desa/en/news/population/world-population-prospects-2017.html

[21] Intergovernmental Panel on Climate Change. Human settlements, infrastructure, and spatial planning. Geneva, Switzerland: IPCC; 2015.

[22] Dodman D, Hayward B, Pelling M, et al. Cities, settlements and key infrastructure. In: Pörtner H-O, Roberts D, Tignor M, Poloczanska E, Mintenbeck K, Alegría A, Craig M, Langsdorf S, Löschke S, Möller V, Okem A, Rama B, editors. Climate Change 2022: Impacts, Adaptation and Vulnerability. Cambridge, UK: Cambridge University Press; 2022.

[23] Zhang XQ. The trends, promises and challenges of urbanisation in the world. Habitat Int. 2016;54:241–252. doi: 10.1016/j.habitatint.2015.11.018

[24] MIT Sustainability. Cleaning one of the world's most commonly used substances. 2023. [accessed 2024 Jul 20]. [Online]. Available from: https://sustainability.mit.edu/article/cleaning-one-worlds-most-commonly-used-substances

[25] Kamhawi A, Brown J, Fahmy A, et al. Waste-free production of ultra-thin concrete panels via robotic 3D printing and cnc dynamic bed device. In: RILEM International Conference on Concrete and Digital Fabrication. Hong Kong, China: Springer; 2024.

[26] Kamhawi A, Varadharajan T, Khan MD, et al. Multi-robotic 3D printing wood-based slurry on construction scale. In: EAAE/ARCC Conference 2024: Architecture Into the Unknown, Milan, Italy; 2024.

[27] YouTube. How cement could solve the climate crisis. 2020. [accessed 2024 Jul 20]. [Online]. Available from: https://www.youtube.com/watch?v=awpmJriWcEw

[28] Van Damme H. Concrete material science: past, present, and future innovations. Cem Concr Res. 2018;112:5–24. doi: 10.1016/j.cemconres.2018.05.002

[29] Jayasinghe A, Orr J, Hawkins W, et al. Comparing different strategies of minimising embodied carbon in concrete floors. J Clean Prod. 2022;345:131177. doi: 10.1016/j.jclepro.2022.131177

[30] Mata-Falcón J, Bischof P, Huber T, et al. Digitally fabricated ribbed concrete floor slabs: a sustainable solution for construction. Rilem Tech Lett. 2022;7:68–78. doi: 10.21809/rilemtechlett.2022.161

[31] Flatt RJ, Wangler T. On sustainability and digital fabrication with concrete. Cem Concr Res. 2022;158:106837. doi: 10.1016/j.cemconres.2022.106837

[32] Ranaudo F, Van Mele T, Block P. A low-carbon, funicular concrete floor system: design and engineering of the hilo floors. System. 2021;2016:2016–2018.

[33] Garbett J, Darby A, Ibell TJ. Optimised beam design using innovative fabric-formed concrete. Adv Struct Eng. 2010;13(5):849–860. doi: 10.1260/1369-4332.13.5.849

[34] Stoiber N, Kromoser B. Topology optimization in concrete construction: a systematic review on numerical and experimental investigations. Struct Multidiscipl Optim. 2021;64(4):1725–1749. doi: 10.1007/s00158-021-03019-6

[35] Jewett JL, Carstensen JV. Topology-optimized design, construction and experimental evaluation of concrete beams. Autom Constr. 2019;102:59–67. doi: 10.1016/j.autcon.2019.02.001

[36] Jipa A, Bernhard M, Meibodi M, et al. 3D-printed stay-in-place formwork for topologically optimized concrete slabs. In: Proceedings of the 2016 TxA Emerging Design + Technology Conference. Austin, TX, USA: Texas Society of Architects; 2016. p. 97–107.

[37] Aghaei-Meibodi M, Bernhard M, Jipa A, et al. The smart takes from the strong: 3D printing stay-in-place formwork for concrete slab construction. Fabricate. 2017;3:210–217.

[38] Kamhawi A, Aghaei Meibodi M. Techniques and strategies in extrusion based 3D concrete printing of complex components to prevent premature failure. Automation in Construction. 2024;168:105278.

[39] Pranav P, Van de Weygaert R, Vegter G, et al. Topology and geometry of Gaussian random fields I: on bettinumbers, Euler characteristic, and Minkowski functionals. Mon Not R Astron Soc. 2019;485(3):4167–4208. doi: 10.1093/mnras/stz541

[40] Park C, Pranav P, Chingangbam P, et al. Betti numbers of gaussian fields. arXiv preprint arXiv:1307.2384; 2013.

[41] Gaudillière N, Duballet R, Bouyssou C, et al. Large-scale additive manufacturing of ultra-high-performance concrete of integrated formwork for truss-shaped pillars. In: Robotic Fabrication in Architecture, Art and Design 2018: Foreword by Sigrid Brell-Çokcan and Johannes Braumann, Association for Robots in Architecture. Zurich, Switzerland: Springer; 2019. p. 459–472.

[42] Ooms T, Vantyghem G, Tao Y, et al. The production of a topology-optimized 3D-printed concrete bridge. In: RILEM International Conference on Concrete and Digital Fabrication. Eindhoven, The Netherlands: Springer; 2022. p. 37–42.

[43] Vantyghem G, De Corte W, Shakour E, et al. 3D printing of a post-tensioned concrete girder designed by topology optimization. Autom Constr. 2020;112:103084. doi: 10.1016/j.autcon.2020.103084

[44] Leslie T. Form as diagram of forces: the equiangular spiral in the work of Pier Luigi Nervi. J Archit Educ. 2003;57(2):45–54.

[45] Tam K-MM, Mueller CT. Stress line generation for structurally performative architectural design. In: 35th Annual Conference of the Association for Computer Aided Design in Architecture (ACADIA). Cincinnati, Ohio: ACADIA; 2015 Oct. p. 19–25.

[46] Eschenauer HA, Olhoff N. Topology optimization of continuum structures: a review. Appl Mech Rev. 2001;54(4):331–390. doi: 10.1115/1.1388075

[47] Tam K-MM, Mueller CT. Additive manufacturing along principal stress lines. 3D Printing Addit Manuf. 2017;4(2):63–81. doi: 10.1089/3dp.2017.0001

[48] Gebhard L, Mata-Falcón J, Anton A, et al. Structural behaviour of 3D printed concrete beams with various reinforcement strategies. Eng Struct. 2021;240:112380. doi: 10.1016/j.engstruct.2021.112380
[49] Vantyghem G, Ooms T, De Corte W. Fem modelling techniques for simulation of 3D concrete printing. arXiv preprint arXiv:2009.06907; 2020.
[50] Wolfs R, Suiker A. Structural failure during extrusion-based 3D printing processes. Int J Adv Manuf Technol. 2019;104:565–584. doi: 10.1007/s00170-019-03844-6
[51] Chen Y, Chaves Figueiredo S, Yalçinkaya Ç, et al. The effect of viscosity-modifying admixture on the extrudability of limestone and calcined clay-based cementitious material for extrusion-based 3D concrete printing. Materials. 2019;12(9):1374, 1–14. doi: 10.3390/ma12091374
[52] Janusziewicz R, Tumbleston JR, Quintanilla AL, et al. Layerless fabrication with continuous liquid interface production. Proc Natl Acad Sci. 2016;113(42):11703–11708. doi: 10.1073/pnas.1605271113
[53] Elkaseer A, Schneider S, Scholz SG. Experiment-based process modeling and optimization for high-quality and resource-efficient FFF 3D printing. Appl Sci. 2020;10(8): 2899. 1–19. doi: 10.3390/app10082899
[54] Zhao D, Guo W. Shape and performance controlled advanced design for additive manufacturing: a review of slicing and path planning. J Manuf Sci Eng. 2020;142(1):010801. doi: 10.1115/1.4045055
[55] Lao W, Li M, Tjahjowidodo T. Variable-geometry nozzle for surface quality enhancement in 3D oncrete printing. Addit Manuf. 2021;37:101638.
[56] Mitropoulou I, Bernhard M, Dillenburger B. Nonplanar 3D printing of bifurcating forms. 3D Printing Addit Manuf. 2022;9(3):189–202. doi: 10.1089/3dp.2021.0023
[57] Mitropoulou I, Bernhard M, Dillenburger B. Print paths key-framing: Design for non-planar layered robotic FDM printing. In: Proceedings of the 5th Annual ACM Symposium on Computational Fabrication; ser. SCF '20. New York, NY, USA: Association for Computing Machinery; 2020. [Online]. doi: 10.1145/3424630.3425408
[58] Dai C, Wang CC, Wu C, et al. Support-free volume printing by multi-axis motion. ACM Trans Graph. 2018;37(4):1–14. doi: 10.1145/3197517.3201342
[59] Etienne J, Ray N, Panozzo D, et al. Curvislicer: slightly curved slicing for 3-axis printers. ACM Trans Graph. 2019;38(4):1–11. doi: 10.1145/3306346.3323022
[60] Fang G, Zhang T, Zhong S, et al. Reinforced FDM: multi-axis filament alignment with controlled anisotropic strength. ACM Trans Graph. 2020 Nov;39(6):204:1–15. doi: 10.1145/3414685.3417834
[61] Chakraborty D, Reddy BA, Choudhury AR. Extruder path generation for curved layer fused deposition modeling. Comput Aided Des. 2008;40(2):235–243. doi: 10.1016/j.cad.2007.10.014
[62] Llewellyn-Jones T, Allen R, Trask R. Curved layer fused filament fabrication using automated toolpath generation. 3D Printing Addit Manuf. 2016;3(4):236–243. doi: 10.1089/3dp.2016.0033
[63] Ezair B, Fuhrmann S, Elber G. Volumetric covering print-paths for additive manufacturing of 3D models. Comput Aided Des. 2018;100:1–13. doi: 10.1016/j.cad.2018.02.006
[64] Hu K, Jin S, Wang CC. Support slimming for single material based additive manufacturing. Comput Aided Des. 2015;65:1–10. doi: 10.1016/j.cad.2015.03.001
[65] Yabanigül MN, Gulec Ozer D. Exploring architectural units through robotic 3d concrete printing of space-filling geometries. Buildings. 2024;15(1):60, 1–18. doi: 10.3390/buildings15010060
[66] Cromwell. 3D concrete homes. [accessed 2025 Feb 13]. [Online]. Available from: https://cromwell.com/2022/03/01/3d-concrete-homes/
[67] PMSA. 3D concrete printing. [accessed 2025 Feb 13]. [Online]. Available from: https://pmsa.com/products/3d-concrete-printing/3d-concrete-printing/
[68] Xie F, Jing X, Zhang C, et al. Volume decomposition for multi-axis support-free and gouging-free printing based on ellipsoidal slicing. Comput Aided Des. 2022;143(C):103135. doi: 10.1016/j.cad.2021.103135
[69] Zhang T, Fang G, Huang Y, et al. S3-slicer: a general slicing framework for multi-axis 3D printing. ACM Trans Graph. 2022 Nov;41(6). 233:1–14. doi: 10.1145/3550454.3555516
[70] Ni X, Garland M, Hart JC. Fair morse functions for extracting the topological structure of a surface mesh. ACM Trans Graph. 2004;23(3):613–622. doi: 10.1145/1015706.1015769
[71] Benedetti B. Discrete morse theory for manifolds with boundary. Trans Am Math Soc. 2012;364(12):6631–6670. doi: 10.1090/tran/2012-364-12
[72] Goresky M, MacPherson R, Goresky M, et al. Stratified morse theory. Berlin, Heidelberg: Springer; 1988.
[73] Grove K. Critical point theory for distance functions. In: Proceedings of Symposia in Pure Mathematics. Vol. 54, no. 3. Santa Cruz, CA, USA: American Mathematical Society; 1993. p. 357–385.
[74] Edelsbrunner H, Harer J, Natarajan V, et al. Morse-smale complexes for piecewise linear 3-manifolds. In: Proceedings of the Nineteenth Annual Symposium on Computational Geometry, San Diego, CA, USA; 2003. p. 361–370.
[75] Botsch M, Kobbelt L, Pauly M, et al. Polygon mesh processing. CRC Press; 2010.
[76] Schafer RW. What is a savitzky-golay filter? IEEE Signal Processing Magazine [Online]. Piscataway, NJ, USA: IEEE. Available from: https://api.semanticscholar.org/CorpusID:15584363

Appendices

Appendix 1. Scalar field implementation

The geodesic distance tensors are defined as follows:

$$\mathbf{D}_{vn}[i,j,b] = d_g(v_i, n_j), \quad \mathbf{D}_{vn} \in \mathbb{R}^{|V|\times|N_\partial|\times 2}, \quad b \in \{\text{upper, lower}\} \tag{A1}$$

$$\mathbf{D}_{vs}[i,k] = d_g(v_i, s_k), \quad \mathbf{D}_{vs} \in \mathbb{R}^{|V|\times|S_s|} \tag{A2}$$

$$\mathbf{D}_{sn}[k,j,b] = d_g(s_k, n_j), \quad \mathbf{D}_{sn} \in \mathbb{R}^{|S_s|\times|N_\partial|\times 2} \tag{A3}$$

where $\mathbf{D}_{vn}$ stores vertex-to-boundary distances, $\mathbf{D}_{vs}$ stores vertex-to-saddle distances, and $\mathbf{D}_{sn}$ stores saddle-to-boundary distances.

Appendix 2. Per-face gradient computation of the scalar field

For each triangular face $t \in \mathcal{F}$ with vertex indices (i, j, k) and corresponding vertex positions $(\mathbf{p}_i, \mathbf{p}_j, \mathbf{p}_k) \in \mathbb{R}^3$, the gradient magnitude $\|\nabla F\|_t$ quantifies the local rate of change of the normalised scalar field F within the triangle's tangent plane. The computation proceeds by establishing a local two-dimensional coordinate system aligned with the triangle.

Let $\mathbf{e}_1 = \mathbf{p}_j - \mathbf{p}_i$ and $\mathbf{e}_2 = \mathbf{p}_k - \mathbf{p}_i$ denote the edge vectors emanating from vertex i. The triangle's unit normal vector is:

$$\hat{\mathbf{n}} = \frac{\mathbf{e}_1 \times \mathbf{e}_2}{\|\mathbf{e}_1 \times \mathbf{e}_2\|} \tag{A4}$$

To establish an orthonormal tangent basis, we normalise the first edge vector to obtain:

$$\hat{\mathbf{u}} = \frac{\mathbf{e}_1}{\|\mathbf{e}_1\|} \tag{A5}$$

The second basis vector is constructed via Gram-Schmidt orthogonalization:

$$\hat{\mathbf{v}} = \frac{\mathbf{e}_2 - (\mathbf{e}_2 \cdot \hat{\mathbf{u}})\hat{\mathbf{u}}}{\|\mathbf{e}_2 - (\mathbf{e}_2 \cdot \hat{\mathbf{u}})\hat{\mathbf{u}}\|} \tag{A6}$$

In this local coordinate system, the three vertices have planar coordinates:

$$\begin{aligned} (x_i, y_i) &= (0, 0) \\ (x_j, y_j) &= (\|\mathbf{e}_1\|, 0) \\ (x_k, y_k) &= (\mathbf{e}_2 \cdot \hat{\mathbf{u}}, \mathbf{e}_2 \cdot \hat{\mathbf{v}}) \end{aligned} \tag{A7}$$

Let F_i, F_j, and F_k denote the normalised scalar field values at the three vertices. The gradient components (a, b) in the tangent plane satisfy the linear system:

$$\begin{aligned} F_j - F_i &= a \cdot x_j \\ F_k - F_i &= a \cdot x_k + b \cdot y_k \end{aligned} \tag{A8}$$

Solving for the gradient components yields:

$$a = \frac{F_j - F_i}{x_j}, \quad b = \frac{F_k - F_i - x_k \cdot a}{y_k} \tag{A9}$$

The gradient magnitude for face t is then:

$$\|\nabla F\|_t = \sqrt{a^2 + b^2} \tag{A10}$$

Appendix 3. Iso-curve segment extraction on triangular meshes

The interpolation parameter $t \in [0, 1]$ is computed as:

$$t = \frac{\tau_m - F_a}{F_b - F_a} \tag{A11}$$

provided that $|F_b - F_a| > \epsilon_{\text{edge}}$, where ϵ_{edge} is a numerical tolerance (typically 10^{-6}) that prevents division by near-zero denominators when the edge is nearly parallel to the iso-surface. The three-dimensional coordinates of the intersection point are:

$$\mathbf{p} = \mathbf{p}_a + t(\mathbf{p}_b - \mathbf{p}_a) = (1 - t)\mathbf{p}_a + t\mathbf{p}_b \tag{A12}$$

This linear interpolation assumes that the scalar field varies linearly across each triangle, which is consistent with the piecewise-linear representation of the field on the triangulated mesh.

For each triangle, the algorithm collects all intersection points found on its edges. Since each triangle has exactly three edges and the iso-level is generic (not aligned with vertex positions), a triangle either has no intersections, exactly two intersections (when the iso-surface cuts through the triangle interior), or degenerately has one intersection (when the iso-level passes through a vertex, which is handled by the numerical tolerance). When exactly two intersection points are found on triangle t, they define a line segment $\mathbf{s}_t = (\mathbf{p}_1, \mathbf{p}_2)$ that represents the iso-curve's trajectory within that face.

The complete set of intersection segments extracted at iteration m is denoted:

$$\mathcal{S}_m = \{\mathbf{s}_t : t \in \mathcal{F}, |\text{intersections on } t| = 2\} \tag{A13}$$

where each segment $\mathbf{s}_t = (\mathbf{p}_{t,1}, \mathbf{p}_{t,2})$ consists of an ordered pair of three-dimensional points. The cardinality $|\mathcal{S}_m|$ represents the number of faces intersected by the iso-level, which varies with the geometric complexity and the position of τ_m within the normalised field domain.

Appendix 4. Arc-length resampling equations

For smoothed component $\mathcal{C}_{m,j}$ with points $\{\tilde{\mathbf{q}}_{j,1}, \tilde{\mathbf{q}}_{j,2}, \ldots, \tilde{\mathbf{q}}_{j,n_j}\}$, the piecewise cumulative arc length function is:

$$L_j(i) = \sum_{k=1}^{i-1} \|\tilde{\mathbf{q}}_{j,k+1} - \tilde{\mathbf{q}}_{j,k}\|, \quad i = 1, 2, \ldots, n_j \tag{A14}$$

with $L_j(1) = 0$ and total arc length $\ell_j = L_j(n_j)$. The algorithm generates s uniformly spaced arc length parameters:

$$\lambda_r = (r - 1) \cdot \frac{\ell_j}{s - 1}, \quad r = 1, 2, \ldots, s \tag{A15}$$

For each λ_r, the segment index i_r is identified via binary search such that:

$$L_j(i_r) \leq \lambda_r < L_j(i_r + 1) \tag{A16}$$

The local interpolation parameter within segment i_r is:

$$\xi_r = \frac{\lambda_r - L_j(i_r)}{\|\tilde{\mathbf{q}}_{j,i_r+1} - \tilde{\mathbf{q}}_{j,i_r}\|} \tag{A17}$$

and the resampled point is:

$$\mathbf{r}_{j,r} = (1 - \xi_r)\tilde{\mathbf{q}}_{j,i_r} + \xi_r\tilde{\mathbf{q}}_{j,i_r+1} \tag{A18}$$