Conformal Microstructure Synthesis in Trimmed Trivariate based V-reps

Q Youn Hong, Gershon Elber

Computer Science Dept., Technion, Israel Institute of Technology, Haifa, Israel

Abstract

We present a complete microstructure tiling paradigm for V-rep models, or volumetric models consisting of trimmed trivariates. Existing methods [10] are employed to tile individual primitive tensor product trivariates in a conformal way, only to handle the intersection of the microstructure tiles with the trimming surfaces of the trivariates. One-to-one and two-to-one bridging tiles are then constructed along the trimmed zones, while tile-clipping is completely avoided. The Boolean operation cases of subtraction, intersection and union are considered. The result is a set of, regular in the interior, possibly heterogeneous, trivariates, that defines the whole microstructure arrangement. This result is fully compatible with iso-geometric analysis as well as heterogeneous additive manufacturing. Examples are presented, including of 3D printed heterogeneous microstructures.

Keywords: Volumetric trivariate splines; Trimmed trivariates; Function composition; Porosity; Lattices; Heterogeneity; iso-geometric analysis; additive manufacturing

1. Introduction

We propose a method to continuously, conformally and adaptively place, possibly heterogeneous, spline micro tiles in a volumetric model (VModel). A VModel is a model in a volumetric representation (V-rep), based on trimmed trivariates. The placement of the tiles is processed hierarchically by separating the design of the macro-shape of the VModel from the micro-element or tile that will be filling the interior of the macro VModel [10]. The macro shape is represented by a set of V-Cells, which are typically trimmed, trivariate tensor product splines. A tile can be any set of curves, (trimmed) surfaces, or (trimmed) trivariate splines, all defined in the domain of a unit cube, $[0, 1]^3$, and hence can be modeled with high flexibility. In this work, we will focus on tiles that are formed of trivariates, the most complex tile type, while simpler geometries, like curves and surfaces, could be following a similar yet simpler path. The spline functions describing the tile are then functionally composed into the macro shape's spline VModel representation, numerous times, which makes the micro-structures repetitively and continuously paved inside the macro model. See Figure 1 for an example of embedding heterogeneous microstructure consisting of volumetric trivariate tiles, inside a trimmed trivariate based VModel. This heterogeneous microstructure from Figure 1 (c) is fully compatible with and can be immediately employed in iso-geometric analysis tools. Further, the boundaries of the constructed microstructures are fully compatible with the B-spline based boundary representation (B-rep) of contemporary geometric CAD systems and additive manufacturing [4].

The construction scheme of tile-based microstructures [10], while conformal to the macro VModel, is limited by the fact that the tensor product trivariate presents a cube-like topology, which also renders a cube-like topology of the final microstructure. In some cases, however, the shape of the macro models are more complex. Models can undergo multiple Boolean operations, resulting in sets of trimmed trivariates in a single volumetric representation or V-rep VModel [9].

When one aims to pave with tiles, a macro VModel consisting of trimmed trivariates following [10], some of the tiles in the full tensor product domain of some of the trivariates in the VModel might be trimmed away, possibly partially. Moreover, tiling a complete VModel, consisting of several trimmed trivariates, will require the proper management and the continuous stitching of the microstructures between adjacent (trimmed) trivariates, near the shared trimming boundary surface(s).

This work presents one possible paradigm to handle this tiling problem for a whole VModel. We start from the set of the trimmed trivariate primitives in the VModel. Given a trimmed trivariate, we first synthesize the microstructure in the tensor product trivariate primitive, ignoring the trimming, and following [10]. Some of the tiles are then eliminated if found (partially) outside the trimmed zone, whereas completely inside tiles are kept. Then, in zones that are near the trimming surfaces of the trimmed trivariates, special bridging tiles are introduced, to connect tiles to adjacent trimmed trivariates and/or to the boundary of the VModel.

The rest of this work is organized as follows. In Section 2, we survey related previous work. Then, Section 3 describes our entire algorithm to populate a general VModel with tiles. Some examples and results are presented in Section 4, only to conclude in Section 5.

2. Previous work

The design and manufacturing of 3D volumetric objects using porous repetitive microstructures, has gained recent

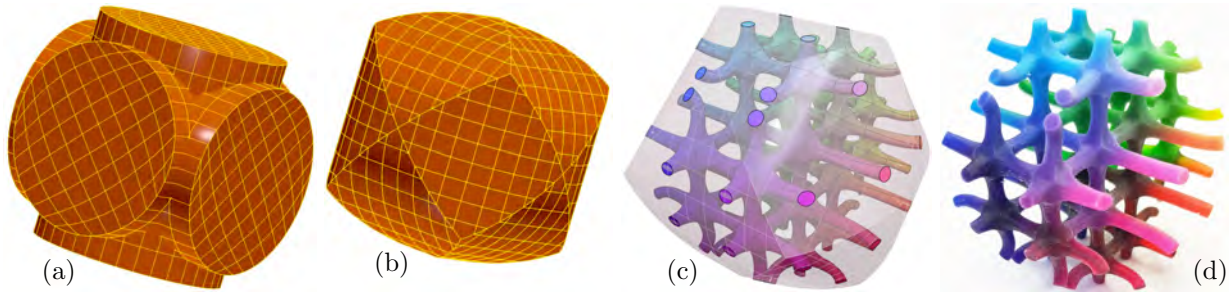

Figure 1: The intersection of three trivariate cylinders in (a), yields the shape in (b). Populating the interior, following the tile placement presented in this work, eliminates all intersecting tiles and builds bridging tiles to the trimmed surface boundaries, as is shown in (c). (d) presents a 3D-printed version of (c). In both (c) and (d), the heterogeneous graded colors were devised as a mapping from (x, y, z) to (R, G, B).

attention in various academic and industrial fields including computer-aided design, tissue engineering, aero-space engineering, and material science. Advances in additive manufacturing have facilitated the production of complex porous structures in a given model, while allowing heterogeneity within the model, with graded materials. Recent survey papers [16, 18, 13, 8] examined several aspects of this fabrication process, from designing the tile of the microstructure and populating the microstructure in the object, through selecting the material and printing technology suited for the porous structures, to validating the physical properties of the manufactured products. Our work here focuses on the geometric synthesis of microstructures and the population method. Hence, in this section, we will focus on related previous results.

Non-stochastic microstructures can be classified as uniform and/or conformal, based on the method used to populate microstructures in the macro shape [18, 14]. Uniform microstructures populate the template microstructures in the internal volume of the object without any guarantee on the continuity between the external boundary of the macro shape and the internal microstructures. Conformal microstructures, on the other hand, fill the macro shape with deformed microstructures so that the microstructures are populated more seamlessly.

Some researchers proposed 3D strut-and-node-based lattices that are conformal to the external boundary of the object. Wang et al. [21] generate semi-struts that connect the central nodes inside the macro shape using STL models. Ngyuen et al. [14] predefine four types of template lattice shapes and optimize the type and the size of the lattice based on the stress distribution of the macro shape. To populate the conformal microstructures, the input boundary mesh is divided into planar regions and offseted to make a trivariate volume between the original and offset surfaces and mapped to the ground truss structure.

Aremu et al. [2] proposed a voxel-based lattice microstructure that conforms to the external boundary of the shape. They start with designing a voxel-based uniform lattice and handle the discrepancies on the boundary of the object by trimming the lattice on the boundary.

Another related approach is presented in Sosin et al. [20] that builds a truss structure by placing the nodes of the structure at the centers of 3D packed spheres in the shape, striving for an optimal honeycomb-like topology. Regarding the design of beams in the strut-and-node-based microstructures, Gupta et al. [7] proposed to employ quadric-of-revolution surfaces to construct a beam of a single node or smoothly connect two sphere-shaped nodes. They also offer closed-form solutions to construct the 'hub' which joins two beams to a sphere-shaped node, which is an essential component in the construction of lattice-based microstructures.

Medeiros e Sá et al. [11] derive the internal structure of the volumetric object from a boundary mesh. They propose an adaptive algorithm to compute the internal microstructure from an input mesh using a primal-dual relationship between the mesh and inner structure. The resulting microstructure is heavily determined by the topology of the input mesh, which affects the control over the structure.

Instead of using explicit geometry as a unit tile, some researchers adopt implicit function-based methods. Pasko et al. [15] proposed a function-based representation to procedurally generate microstructures in a volumetric object. The microstructure is a uniform axis-aligned lattice expressed by infinite, periodic and implicit functions and is combined with the macro shape via set operations. Another example of implicit tiling is Feng et al. [6] that fills a volumetric object with triply periodic minimal surfaces (TPMS) and computes slice-by-slice intersection and interpolation with the macro shape.

Tang et al. [19] adopt a hybrid approach in modeling the microstructure of the volumetric object. They represent the macro shape as the topological skeleton that consists of vertices and links, and embed a unit microstructure in each vertex. The geometry of each strut is then computed using implicit functions.

Instead of using truss-like unit microstructures, Massarwi et al. [10] use a function composition-based tiling in trivariate volumetric V-Model. Micro-tiles in the domain are composed into the trivariate volume of the macro shape. Tiles can be regular grid-like tiling, randomized tiling and bifurcating tiling, in which the number of inlet and outlet surfaces are different. The same scheme is employed in [1] to optimize the geometry in a variety of physical applications, like heat transfer and stress analysis.

Another widely used and worthy of mentioning scheme in the design of porous geometries is topology optimization. [3]. This technique derives the porosity through, typically discrete, optimization with respect to desired physical properties.

Considering the format of the result, most offer polyg-

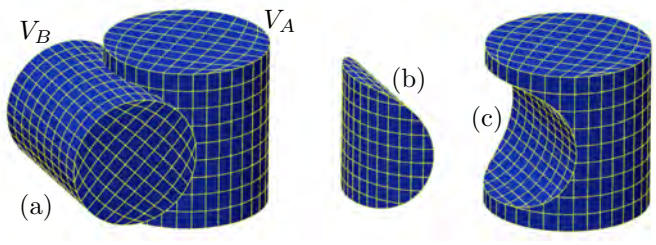

Figure 2: A typical Boolean operation between VModels V_A and V_B can be one of a union $V_A \cup V_B$ (a), an intersection $V_A \cap V_B$ (b) or a subtraction $V_A - V_B$ (c).

onal mesh [6, 11, 14, 21], or voxels or similar spatial discretizations [2, 3, 15, 19], as their output. In contrast, [7, 10, 20] employ spline or parametric function-based representations.

This work adopts some methods from [10], toward the generation of microstructures in tensor product trivariate forms, and extends it to handle general trimmed trivariate (V-rep) geometries. Tiles can be of general shapes, and of arbitrary dimensions. Further, the result is B-spline based and (its boundary) is fully compatible with modern B-rep geometric CAD tools on the one hand, and supports the encodements of additional volumetric fields, and heterogeneous 3D printing in specific, on the other.

3. The algorithm

In order to create a general shape in a geometric modeling tool, Boolean operations are typically applied. Herein, for VModels, Boolean operations are applied between volumetric primitives [9], and other VModels, recursively, forming a closure. Boolean union, intersection, and subtraction are typically exploited in this constructive process. We adopt the volumetric representation of [9], in which a VModel is a complex of mutually exclusive V-Cells. Each V-Cell is a volumetric zone inside the VModel that is completely contained in all tensor product trivariate splines that share this V-Cell. On the other hand, a V-Cell cannot cross boundaries of a tensor product trivariate. A V-Cell will also contain topological adjacency information on adjacent V-Cells, that together form the VModel, again as a complex. The volumetric zone of each such V-Cell is bounded by a set of trimming surfaces, much like the trimmed curves of a trimmed surface, in a lower dimension.

Consider two volumetric primitives, V_A and V_B. V_A and V_B are represented as tensor product trivariate splines. Using Boolean set operations, we can construct one of the following (trimmed trivariate) VModels: $V_A \cup V_B$ (a union), $V_A \cap V_B$ (an intersection) or $V_A - V_B$ (a subtraction). As an example, in Figure 2, we have two V-rep primitive cylinders, V_A and V_B, and these three different Boolean operations are portrayed.

In general, a VModel can be constructed by numerous Boolean operations between an arbitrary number of volumetric primitives. In Section 3.1, we first discuss how one can classify a tile as inside or outside a single V-Cell, or a trimmed trivariate. Tiles that transversely intersect with a trimming surface are also considered as outside, and are eliminated as well. In other words, all remaining tiles are fully inside the trimmed trivariate.

The construction of simple, one-to-one, bridging tiles is discussed in Section 3.2. Then, in Section 3.3, and given two volumetric primitives, V_A and V_B, we present full microstructure tiling in the simple Boolean operation case of $\mathcal{V} = V_A - V_B$, where one-to-one bridging tiles are constructed near and toward the trimming surfaces.

The problem is much more challenging at a shared trimming surface in a Boolean union, where tiles on both sides of that trimming surface must be matched and bridged. In Section 3.4, we discuss this matching problem, matching N (anchor-faces of) tiles on one side of the trimming surface to M (anchor-faces of) tiles on the other side of the trimming surface. The presented microstructure matching process is a variation on bipartite graph matching [12], where we allow one-to-one but also one-to-two matching ('bigamy'). Bifurcation bridging tiles are hence introduced in Section 3.5 to support such matchings. Finally, the problem of microstructure tiling in a Boolean union is considered in Section 3.6. Here, both the one-to-one and the one-to-two bridging tiles will be employed. Some additional comments are finally made in Section 3.7.

3.1. Classification of Tiles in a Trimmed Trivariate

Consider a tensor product trivariate, T, fully populated with tiles in the 3D grid, following [10]. If this trivariate is now trimmed, as a trimmed trivariate $\mathcal{T}$, some tiles will be inside, some outside, and some will transversely intersect with the trimming boundary of the trivariate. Hence, in each trimmed trivariate $\mathcal{T}$, tiles are classified into three types, depending on their membership with respect to the trimming surfaces. Clearly, the original six boundary surfaces of T, could be ignored in these intersection tests, even if they (partially) serve as trimming surfaces in $\mathcal{T}$. Then, in order to classify the tiles, we:

1. Test if the tile is intersecting with one of the (non-original boundary) trimming surfaces of trivariate. One should recall we do not seek precise intersection curves, only a firm answer if a tile and a (trimming boundary) surface intersect or not, a somewhat simpler problem.
2. Then, for tiles that are intersection free (with respect to the trimming surfaces), we further classify them as completely inside or completely outside, by sampling a single point p on each tile and testing p for inclusion in the trimmed domain.
3. Finally, tiles that are classified as inside, but have a neighboring tile that is detected as intersecting the trimming boundary, are classified as *to-be-bridged* inside tiles.

Figures 3 and 4 show one example of this classification process, for a subtraction case of $V_A - V_B$ (Recall also Figure 2 (c)). All outside tiles (in red) and all the tiles that intersect the trimming boundary (in cyan) are purged. See Figure 3. We are left only with the inside (green) tiles. However, as stated, some of the inside (green) tiles, that are closed to the trimming surfaces will be later connected with special bridging tiles to tiles on the other side of those trimming surfaces (or to the trimming boundary

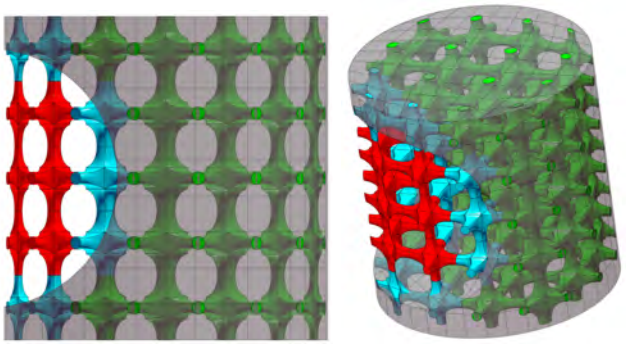

Figure 3: The tiles in this trimmed trivariate (from Figure 2 (c)) are classified into three groups: tiles that are inside the trimmed trivariate (green), outside the trimmed trivariate (red) and tiles that intersect with the boundary (cyan). Two views of the same arrangement.

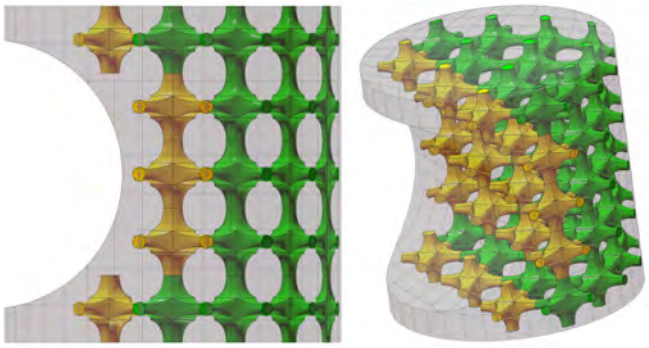

Figure 4: The tiles that are found inside this trimmed trivariate (from Figure 3), are further classified into regular-inside tiles (green) and to-be-bridged-inside tiles (in yellow). The yellow tiles are close to the trimming boundary and are candidates to be bridged to (the other side of) that trimmed surface. Two views of the same arrangement.

itself). Hence, we identify them as to-be-bridged inside, tiles. These to-be-bridged-inside tiles, are marked in yellow, in Figure 4. Finally, we also identify the anchoring faces (surfaces) for the bridging, on these tiles. An anchoring face is a boundary surface of a to-be-bridged tile that was exposed as its neighboring tile was purged (being either outside or intersecting the trimming surfaces of the trivariate),

3.2. The One-to-one Bridging Tiles

With the purging of the tiles that are fully or partially outside the trimmed trivariate, the to-be-bridged inside tiles are becoming our focus. The anchoring faces of these to-be-bridged tiles are typically not tangent to the trimming surface, as they are in the interior domain of the trivariate. (Recall Figure 4). Hence, we insert new tubular tiles that bridge these to-be-bridged tiles, connecting their anchoring faces to the nearby trimming surface. The cross-section of each constructed tubular bridging tile is following the shape of the anchoring surface of the corresponding to-be-bridged tile. While, for simplicity, we assume this cross-section is circular for most of this work, it clearly does not have to be, and we will demonstrate other cross-sections, toward the end.

A bridging tile is constructed by connecting an anchoring surface of a to-be-bridged tile to its closest trimming surface location. The anchoring surface is a boundary surface in a to-be-bridged and one to-be-bridged tile can have multiple anchoring surfaces (See Figure 5 (a)). Each anchoring surface is therefore connected to the trimming surface by sweeping the anchoring surface along some sweeping axis curve. Herein, the sweeping axis curve is set to be (initially) a quadratic Bézier curve, of which control points are composed of:

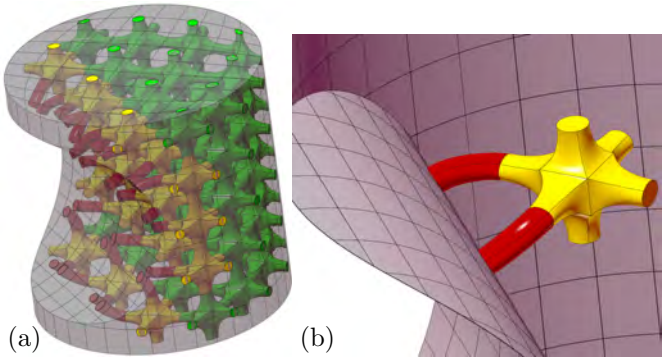

Figure 5: One-to-one bridging tiles connect the to-be-bridged-inside tiles to the closest trimming surface. In (a), all the to-be-bridged-inside tiles (in yellow), from Figure 4, are shown with one-to-one bridging tiles (in red) to the trimming surface. (b) shows one of the to-be-bridged-inside tiles from (a) (in yellow) with two anchoring surfaces, that are now bridged using two one-to-one bridging tiles (in red).

1. The center point, C, of the anchoring surface,
2. A middle point, M, in some direction $\vec{d}$ from the anchoring face toward the trimming surface, and,
3. The (closest) projection point, P, of M, on the trimming boundary surface.

The direction $\vec{d}$ can equal $\vec{n}$, a central normal of the anchoring face at C but also a partial derivative of the trivariate T holding the anchoring face, $\frac{\partial T}{\partial w}$, at C, or any combination thereof. More on that is Section 3.5 (see also Figure 9 (a)).

The bridging tile is a sweep along the axis curve of the anchoring face, and can connect the to-be-bridged-inside tile smoothly with G^1-continuity while joining the trimming boundary surface orthogonally. We introduce a scale parameter β, to scale the vector $\vec{d}$ from P toward M, to control the amount of the bending in the axis.

Figure 5 (a) shows all bridging tiles (in red), in the configuration from Figure 4. Figure 5 (b) shows one example of the to-be-bridged tile (in yellow) from Figure 5 (a) with two one-to-one bridging tiles (in red) that connect the tile at its two anchoring surfaces to the nearest trimming surface.

When the reconstructed axis curve, as discussed above, has high curvature, there is a chance that the swept bridge surface will self-intersect. We avoid such self-intersections, by adjusting (and adding new control points) the axis curve. Further, adjacent one-to-one bridging surfaces might intersect each other as well, another event where we locally adjust the axis curves. More on this in Section 3.3.

3.3. Populating a Single Subtraction V-rep with Microstructure Tiles

Consider the Boolean set operation of subtraction, $V_A - V_B$, between two tensor product trivariate V-rep primitives, V_A and V_B, and let $\mathcal{M}_\mathcal{A}$ be the set of microstructure tiles that populate V_A. Note that herein the microstructure $\mathcal{M}_\mathcal{B}$, in V_B, is not relevant. In other words, the micro tiles in $V_A - V_B$ will come solely from $\mathcal{M}_\mathcal{A}$.

The boundary trimming surfaces of VModel $V_A - V_B$ are divided into two groups depending on the source of the surfaces: one group originates from V_A and the other from V_B. Among these boundary surfaces and considering only set $\mathcal{M}_\mathcal{A}$, the surfaces that participate in the tile classification (i.e Section 3.1) are only the surfaces of V_B. Then, the to-be-bridged tiles are the interior tiles adjacent to the tiles that intersect the surface boundary of V_B and hence purged during tile classification process (recall the yellow tiles in Figure 4).

As each bridging tile is created independently, there is the possibility that these generated tiles will interfere with each other. Note that the original set of interior tiles, $\mathcal{M}_\mathcal{A}$ (green tiles in Figure 4), are collision-free as long as the provided micro-tile and macro-shape trivariate of the VModel are collision-free (positive Jacobian throughout). Therefore, we only need to consider the possibility of collisions between the newly created swept bridging tiles.

Collisions between adjacent bridging tiles can be detected by simply testing if any pair of (adjacent) bridging tiles collide with each other. In the specific case of circular cross-sections, two bridging tiles collide if the distance between the sweeping axes of the tiles is smaller than the sum of the radii of the cross sections of the sweeps. The latter can also be used for general shapes of cross sections, employing upper bounds on the (varying) radii, to accelerate the decision, only to test further when the distance between two bounding volumes are smaller than that bound.

As a first step before resolving a collision between two adjacent swept bridges, we raise the degree of the sweeping axis curve to a cubic and add one degree of freedom in the axis curve. By increasing the degree of the curve, we are able to construct the sweeping axis connecting the center point C of the anchoring surface to the arbitrary point on the trimming surface while maintaining the orthogonality of the axis curve with respect to the trimming surface.

The initial aim to resolve a collision, is by swapping the two end projection points, P, of the two adjacent bridges, but leaving the anchoring face points, C, intact. Figure 6 (c) shows this operation. In Figure 6 (b), we detect a pair of colliding tiles. From Figure 6 (b) to Figure 6 (c), we swap the two projection points of the colliding tiles. If the minimum distance between the axis curves of the tiles is becoming larger after the swapping, we consider this swapping beneficial and keep the swapping.

Indeed, not every swapping result that is kept, ends in two collision-free tiles. The pink and cyan bridging tiles in Figure 6 (c), for instance, still collide, after the swapping of end points, from Figure 6 (b). Hence, in such cases, we resort to an iterative process that employs repulsion forces on the control points of the two axis curves, $A_1(t)$ and $A_2(r)$, that are assumed regular (positive speed throughout). Let $A_1(t_0)$ and $A_2(r_0)$ be the location where A_1 and A_2 are the closest. Then, denote by $T_1 = A_1'(t)|_{t=t_0}$ and $T_2 = A_2'(r)|_{r=r_0}$ and let $V = T_1 \times T_2$ be the employed repelling direction. Note that while, in principle, this direction is similar to $A_1(t_0) - A_2(r_0)$, the magnitude of $A_1(t_0) - A_2(r_0)$ might be vanishing, in which case, direction $A_1(t_0) - A_2(r_0)$ might be unstable.

This repelling process affects the control points associated with the closest locations, t_0 and r_0. While it cannot happen near the to-be-bridged tiles, at the C points (as $\mathcal{M}_\mathcal{A}$ must be collision-free, as explained above), if the closest location occurs near or on the trimmed surface boundary, then the P points are repelled the most, etc.

Finally, one should realize that once we start to modify and/or repel some bridging tiles, new intersections might unfortunately form. After every such repelling iteration, one must verify that no new collision formed, in the close neighborhood.

In a case where the repelling fails, for example when three (or more) bridging tiles are simultaneously colliding, we also resolve collisions by eliminating one of the bridging tiles and repelling the others. These collisions might occur especially around the zone where the two trimming surfaces meet at G^0-continuity. Hence, the number of bridging tiles might be smaller than the number of matched anchoring faces. In the example in Figure 6, we end up with 153 bridging tiles out of 164 matched anchoring faces.

3.4. Handling Bridging Tiles along Shared Boundary Trimming Surfaces

In Section 3.3, we have discussed tiling the microstructures in a trimmed VModel that only contains a single V-Cell. A general trimmed VModel, however, will have several V-Cells. For instance, the trimmed VModel of $V_A \cup V_B$ in Figure 2 (a) consists of three V-Cells, $V_A - V_B$, $V_A \cap V_B$ and $V_B - V_A$ as is shown in Figure 7. When Boolean operations are applied to the primitive trivariate VModels or other trimmed VModels, recursively, each V-Cell in the trimmed VModel holds the list of the primitive trivariates that occupy (the volume of) that V-Cell [9]. Then, the microstructures in a V-Cell can be constructed based on one of the trivariates occupying and containing the V-Cell. In the case of the V-Cells in Figure 7, the microstructures in $V_A - V_B$ and $V_B - V_A$ will be generated from $\mathcal{M}_\mathcal{A}$ and $\mathcal{M}_\mathcal{B}$, respectively, recalling that $\mathcal{M}_\mathcal{A}$ and $\mathcal{M}_\mathcal{B}$ are the sets of microstructures in V_A and V_B. However, the V-Cell of $V_A \cap V_B$ can be tiled based on either $\mathcal{M}_\mathcal{A}$ or $\mathcal{M}_\mathcal{B}$.

The union of $V_A \cup V_B$ is mathematically equivalent to either one of the following three formulations:

1. $V_A \cup (V_B - V_A)$ or,
2. $V_B \cup (V_A - V_B)$ or,
3. $(V_A - V_B) \cup (V_A \cap V_B) \cup (V_B - V_A)$.

The synthesis of microstructures over the union will depend on the formulation that is employed. Giving V_A priority, $V_A \cup (V_B - V_A)$, employs every microstructure tile from $\mathcal{M}_\mathcal{A}$ but purges some of the microstructures in $\mathcal{M}_\mathcal{B}$ that are (partially) outside $V_B - V_A$. The symmetric situation, giving V_B priority, occurs for case 2, of $V_B \cup (V_A - V_B)$. At equal priority, as in case 3 of $(V_A - V_B) \cup (V_A \cap V_B) \cup (V_B - V_A)$, the microstructure of the union will partially employ microstructure tiles from both

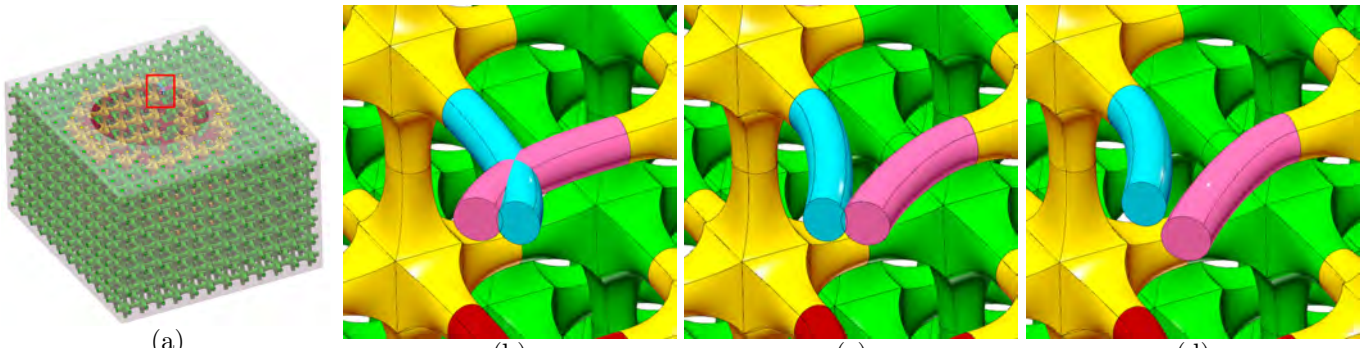

Figure 6: Collisions between neighboring bridging tiles are detected and resolved. (a) All the inside tiles in the trimmed VModel. (b) A zoom-in on the region marked in red in (a), with two neighboring bridging tiles (in cyan and pink) colliding with each other. (c) By swapping the end points of the sweeping axes, the collisions between the tiles are alleviated. However, the two tiles still intersect near the end points. (d) The sweeping axes of the colliding tiles are moved by applying a repulsion force, and eventually becoming collision-free.

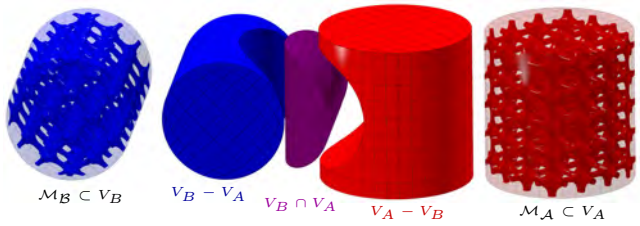

Figure 7: The VModel in Figure 2 (a) consists of three V-Cells, with the microstructures of the tensor product input primitive cylinder.

$\mathcal{M}_\mathcal{A}$ and $\mathcal{M}_\mathcal{B}$, which implies that the membership tests of the tiles are executed twice, for $V_A - V_B$ and $V_B - V_A$.

While only case 3 is symmetric, we allow the end-user to assign (numeric) importance priorities to the primitive trivariates and these priorities will affect the way the microstructures are synthesized, selecting between the three above options. Microstructures of a trivariate with a higher priority value will always be selected first.

All that said, we are now faced with an additional concern. We need to bridge the remaining tiles between two V-Cells, along their common trimming surface boundary, denoted S_B. Consider case 1, of $V_A \cup (V_B - V_A)$. We keep the full set of tiles in V_A, $\mathcal{M}_\mathcal{A}$ and build bridging tiles from them to the interior subset of the tiles of $\mathcal{M}_\mathcal{B}$ in $(V_B - V_A)$, through S_B. This bridging difficulty is more challenging for the fact that now we need to match and connect N anchoring faces in one side, say in V_A, to M anchoring faces in the other side, say in $(V_B - V_A)$, and N and M are likely to be different. In the rest of the section, we consider this combinatorial/geometric matching problem between anchoring faces of two microstructure sets, along a shared trimming surface boundary, S_B.

3.4.1. The Matching of Bridging Tiles

Along the shared trimming surface boundary, S_B, we have two sets of microstructures with anchoring faces that must be matched and bridged. Let the adjacent V-Cells that share boundary S_B, be denoted $\mathcal{VC}_1$ and $\mathcal{VC}_2$. Along S_B, $\mathcal{VC}_1$ might have N anchoring faces to bridge to, while $\mathcal{VC}_2$ might have M anchoring faces to bridge to, $M \neq N$.

Because M is typically not equal to N, a complete matching of this unbalanced bipartite graph is not a complete solution, and will leave some anchoring faces unmatched. Therefore, herein we allow bigamy or also two-to-one matchings, with the assumption that $M \leq 2N$ and $N \leq 2M$. There are two concerns to be addressed here:

1. We need to offer a geometric *bifurcation tile* that connects two anchoring faces on one side of S_B to one anchoring face on the other side of S_B. This geometric question will be addressed in Section 3.5.
2. We need to address the combinatorial question of this unbalanced bipartite graph matching, which also permits bigamy. In the rest of this section, we address this combinatorial question.

We are faced with a matching problem on the unbalanced bipartite graph $\mathcal{G}$ having M nodes, V_i, on one side and N nodes, V_j, on the other side. Each node in $\mathcal{G}$ represents an anchoring face. Hence after, nodes $V_i, V_j \in \mathcal{G}$ will also represent the corresponding geometric anchoring face, depending on the context.

Recall that $\vec{d}$ is the direction along which the bridging tile is constructed, at the anchoring face. Herein, bridging between two anchoring faces, these directions will be denoted $\vec{d_i}$ and $\vec{d_j}$, at V_i or V_j, respectively. Then, each edge, $E_{ij} \in \mathcal{G}$, between V_i and V_j, is assigned a weight, based on:

- The Euclidean distance between V_i and V_j, $Dist(V_i, V_j)$, striving to minimize these distances. Further, we limit the maximal allowed distance to be some constant $d^{maxdist}$ and assign an 'invalid' weight in such cases.
- The angles between vector $V_i - V_j$ and the bridging directions at anchoring faces V_i and V_j, $\vec{d_i}$ and $\vec{d_j}$, $Angle(V_i - V_j, \vec{d_i})$ and $Angle(V_j - V_i, \vec{d_j})$. In other words, we strive to construct bridging tiles that bend and deviate the least from $\vec{d_i}$ and $\vec{d_j}$, as we move away from the anchoring faces. Further, we do not allow these angles to span more than 90 degrees and assign an 'invalid' weight in such cases.

Allowing only one-to-one and two-to-one matches for the bridging tiles, the matching problem is reduced to the problem of finding a set of one-to-one and two-to-one bridges that minimizes the total sum of the weights while

maximizing the number of participants. In the special case of $M = N$, the problem is known as the *assignment problem* and we can obtain the $M = N$ matches (which is also of maximal cardinality) that minimize the sum of edge weights using the Hungarian algorithm [12].

In the rest of the section, we assume $M > N$. By applying the Hungarian algorithm, only one-to-one matchings are considered, and N matches can be found that minimize the sum of weights to the augmented bipartite graph. Then, we remove these N matched vertices from the anchoring faces in $\mathcal{VC}_2$ and apply the Hungarian algorithm again between $M - N$ vertices to the original N vertices on the other side, of $\mathcal{VC}_1$. This, two-staged, matching approach is summarized in Algorithm 1.

Algorithm 1 UnbalancedBipartiteMatch - Matching between M and N anchor faces, $M \geq N$

Input:
F^M: A set of V_i anchoring faces, $|F^M| = M$;
F^N: A set of V_j anchoring faces, $|F^N| = N$;
Output:
P: A set of all matches between F^M and F^N that minimize the sum of weights;
Algorithm
1: $E := \emptyset$;
2: **for** $V_i \in F^M$ **do**
3: **for** $V_j \in F^N$ **do**
4: *//The weight of edge E_{ij} into matrix W:*
5: $w(V_i, V_j) :=$ function of $Dist(V_i, V_j)$ and
6: $Angle(V_i - V_j, \vec{d_i})$ and
7: $Angle(V_j - V_i, \vec{d_j})$;
8: $E := E \cup (V_i, V_j)$;
9: $\mathcal{G}_1 :=$ The unbalanced bipartite graph $(F^M, F^N; E)$;
10: $\mathcal{P}_1 := HungarianAssign(\mathcal{G}_1, W)$;
11: $\overline{F}^M := \{V_i | V_i \in F^M$ and $(V_i, V_j) \in \mathcal{P}_1, V_j \in F^N\}$
12: $\widetilde{E} :=$ All edges in E between $(F^M \setminus \overline{F}^M)$ and F^N;
13: $\overline{E} := GeomFilter(F^M, F^N, \widetilde{E}, \mathcal{P}_1)$;
14: $\mathcal{G}_2 :=$ the unbalanced bipartite graph
15: $(F^M \setminus \overline{F}^M, F^N; \overline{E})$;
16: $\mathcal{P}_2 := HungarianAssign(\mathcal{G}_2, W)$;
17: $\mathcal{P} := \mathcal{P}_1 \cup \mathcal{P}_2$;
18: **Return** $\mathcal{P}$;

Some notes on Algorithm 1. In practice, we will have less than N successful matches in the first round (Line 10, Algorithm 1), as some of the matches will fail (being ∞). In Line 7, we use $Dist()$ and $Angle()$, to compute the distances/angles of vertices/bridging directions. As stated, we limit the maximal allowed distances to be some constant $d^{maxdist}$, and the angular deviations to be less than 90 degrees. Any violation of these conditions will assign the corresponding weight an 'invalid' weight, rendering an impossible match.

When the first round of matching is completed, we have at least $M - N$ unmatched anchoring faces from F^M. We run the Hungarian assignment algorithm once more (in Line 16), but this time, with $M - N$ anchoring faces from F^M and the same N anchoring faces of F^N. All the weights of these edges are inherited from the first stage, except for some edges that are becoming invalid due to geometric constraints, as identified using Algorithm 2.

Algorithm 2 GeomFilter - a filter for the removal of matched edges that are incompatible geometrically.

Input:
F^M: A set of V_i anchoring faces where $|F^M| = M$;
F^N: A set of V_j anchoring faces where $|F^N| = N$;
$\widetilde{E}$: Set of edges (V_i, V_j), $V_i \in F^M \setminus \overline{F}^M$ and $V_j \in F^N$;
$\mathcal{P}_1$: Set of one-to-one matches, computed using $HungarianAssign$;
Output:
$\overline{E}$: A subset of $\widetilde{E}$ where the geometrically invalid edges are filtered;
Algorithm
1: $\overline{E} := \widetilde{E}$;
2: **for** $(V_i, V_j) \in \widetilde{E}$ **do**
3: $\vec{d_j} :=$ bridging direction at V_j;
4: *//Proj. of $V_i - V_j$ onto a face orthogonal to $\vec{d_j}$.*
5: $Prj_j(V_i) = (V_i - V_j) - < V_i - V_j, \vec{d_j} > \vec{d_j}$;
6: **for** $V_k \in \overline{F}^M$ such that $(V_k, V_j) \in \mathcal{P}_1$ **do**
7: *//Proj. of $V_k - V_j$ onto a face orthogonal to $\vec{d_j}$.*
8: $Prj_j(V_k) = (V_k - V_j) - < V_k - V_j, \vec{d_j} > \vec{d_j}$;
9: **if** $Angle(Prj_j(V_i), Prj_j(V_k)) < \frac{\pi}{2}$ **then**
10: $\overline{E} := \overline{E} \setminus (V_i, V_j)$; *// An invalid edge.*
11: **Return** $\overline{E}$;

An edge (V_k, V_j) will be considered geometrically invalid at V_j, if its projection on the plane orthogonal to the bridging direction at V_j, $\vec{d_j}$, will form less than 90 degrees, in that plane, with a previously matched (projected) edge at V_j, (V_i, V_j), from the first step, $\mathcal{P}_1$ (Line 10, Algorithm 1). In other words, we reject bigamy matching at V_j, if the two outgoing arms of the bifurcation form less than 90 degrees.

Finally (in Line 17), the matching results from the two stages are combined together. The union of $\mathcal{P}_1$ and $\mathcal{P}_2$ yields bigamy matches as separate matched pairs, of $(V_{i_1}, V_j) \in \mathcal{P}_1$ and $(V_{i_2}, V_j) \in \mathcal{P}_2$, will be combined into $((V_{i_1}, V_{i_2}), V_j) \in \mathcal{P}$.

The proposed matching algorithm is heuristic. The algorithm does not guarantee optimal matching results, as it applies twice an (optimal) Hungarian algorithm but does not ensure optimality between these two stages. The first stage finds the best match of (up to) N pairs. Since now this set of matches is fixed in the second round, it can clearly happen that the second matching will yield a globally inferior result. Moreover, and as already stated, we can not bridge in full cases for which $M > 2N$ or $N > 2M$.

3.5. The Construction of Two-to-one Bridging Tiles

The matching results, following Section 3.4, include two-to-one bridging tiles between the two sets of microstructures. In this section, we explain how these bifurcation bridging tiles are geometrically constructed.

We have three anchoring faces that participate in the bifurcation tile. Denote them as an *inlet*, the one anchoring face from one microstructure set and two *outlet* anchoring faces from the remaining microstructure set. See Figure 8 (a) and (b). Each bifurcation tile will be composed of three trivariates; a *base trivariate* on top of the inlet surface (blue in Figure 8 (b)) and two branching trivariates that connect the base trivariate to the two outlet anchoring faces (red in Figure 8 (b)).

The most challenging step of constructing the bifurcation tile, is the construction of the base trivariate of the tile. See Figure 8 (b) and (c), in blue. The base trivariate is, in essence, the trivariate that creates the bifurcation. To construct it, a volumetric Boolean sum [5] is applied to six boundary surfaces, $T(u_{min}, v, w)$, $T(u_{max}, v, w)$, $T(u, v_{min}, w)$, $T(u, v_{max}, w)$, $T(u, v, w_{min})$, and $T(u, v, w_{max})$ (to be discussed below) as are shown in Figure 8 (d) and then shown exploded, in Figure 8 (e). Finally, recall that all the geometry is regular in the interior.

The inlet face is set to be $T(u, v, w_{min})$ (hence after denoted as the *bottom surface* of the base trivariate), making this joint conforming to the neighboring tile which shares the inlet face (the green tile throughout Figure 8). The local coordinate frame of the bottom surface is derived from the u and v derivatives (and the normal) at the center of the bottom surface. We denote these vectors as (x, y, z) of the frame, drawn as red, green and blue arrows in Figure 8 (e), respectively.

Because we have excluded bigamy matches where the two projected directions of the outlets are less than 90 degrees, we can always have the outlets connected at two opposite surfaces of the base trivariates. Either to $T(u_{min}, v, w)$ and $T(u_{max}, v, w)$, or to $T(u, v_{min}, w)$ and $T(u, v_{max}, w)$. Without loss of generality, hereafter we assume the configuration is as shown in Figure 8, namely the outlets are connected to $T(u, v_{min}, w)$, $T(u, v_{max}, w)$.

Let α be a user-specified angle to the construction of the base trivariate. The bottom surface is now copied and rotated twice along the x-axis (red in Figure 8 (e)): by α around point q_1 (See Figure 8 (e)) and by $-\alpha$ around q_2 to yield the initial two side surfaces of the base trivariate. The boundary edges holding the joint points q_i, of these two initial surfaces, now $T(u, v_{min}, w)$ and $T(u, v_{max}, w)$, are modified to interpolate the corresponding boundary of the bottom surface, completing the construction of these two side surfaces.

$T(u, v, w_{max})$ is constructed next as scaled version of $T(u, v, w_{min})$ that is forced to interpolate $T(u, v_{min}, w)$ and $T(u, v_{max}, w)$. Finally, the remaining two surfaces, $T(u_{min}, v, w)$ and $T(u_{max}, v, w)$, are derived as a Boolean sum of four boundary curves, now fully defined by $T(u, v_{min}, w)$, $T(u, v_{max}, w)$, $T(u, v, w_{min})$, and $T(u, v, w_{max})$.

We introduce another end-user degree of freedom, ϕ, in the form of the direction of the bridging tile, $\vec{d}$. ϕ can blend between two possible valid option: $\frac{\partial T}{\partial w}$, where T is the trivariate holding the anchoring face, and $\vec{n}_i$, the normal of the anchoring face, both evaluated at the center of the anchoring face. See Figure 9 (a).

Figure 9 shows the two extreme settings of the blending ratio ϕ, applied while constructing the base trivariate. When $\phi = 0$, the base trivariate of the bifurcation tile is aligned in the direction of the w-derivative of T, $\frac{\partial T}{\partial w}$, as is shown in Figure 9 (b). When $\phi = 1$, on the other hand, the base trivariate is constructed and aligned with the normal direction of the inlet anchoring surface, $\vec{n}_i$, as is shown in Figure 9 (c).

The bridging tiles are guaranteed to join with a G^0-continuity to the adjacent tiles as the inlet and outlet boundary surfaces are adopted directly from the surfaces of the neighboring trivariates. Higher continuity is also possible and depends on the continuity of the neighboring trivariates and the user-defined parameters such as ϕ and α, which determine the ending directions of the bridging tiles.

3.6. Construction of Tubular Bridging Tiles in a Union

When generating bridging tiles between two different sets of microstructure, in a union and along a shared boundary, we also employ tubular trivariates for one-to-one matches. The construction here of the tubular bridging tiles, however, is slightly different from the construction of these tubular tiles in the case of a subtraction operation (recall Section 3.3). Here, the one-to-one bridging tile connects two anchoring tile faces instead of joining an anchoring face and a point on the trimming boundary surface of the macro VModel, as was the case in subtraction. To ensure the continuity of the tubular face here, we construct the sweeping axis of the tile as a cubic Hermite, connecting the center locations of the anchoring surfaces while interpolating the starting and ending direction, following the prescribed ϕ blending parameter. Examples of this variant of one-to-one tubular tiles are shown in Figure 10.

As before, one should also handle the possibility of collisions between the bridging tiles. We detect and avoid collisions between tubular trivariates, for one-to-one matches, as well as two-to-one bifurcation tiles. The strategy to avoid collisions in the bridging, here, is restricted, compared to the Boolean subtraction case. We can no longer repulse the end positions of the sweeping axis curves as the end points are no longer on a trimmed surface. It is also undesirable to modify the starting and ending directions of the sweeping axis curves, that are ensuring the continuity of the bridging tiles to the neighboring tiles. Therefore, we increase the degree of the sweeping curves from a cubic to a quintic, obtaining two new degrees of freedom to deform the sweeping axis. Then, we avoid collision cases by repulsing the new internal control points. The new internal control points are iteratively repulsed in the repelling direction computed from the tangents of the axis curves at their closes points, similarly to the subtraction case.

Figure 11 (a) presents a case with two one-to-one tiles that are colliding. This collision is resolved, in Figure 11 (b), by repulsing the internal control points of a quintic sweeping axis (in opposite directions).

3.7. Managing general VModels

The proposed tiling scheme can also be applied to general VModel, created using other Boolean operations such as intersections and multiple Boolean operations. In a case of an intersection, $V_A \cap V_B$, one can employ either

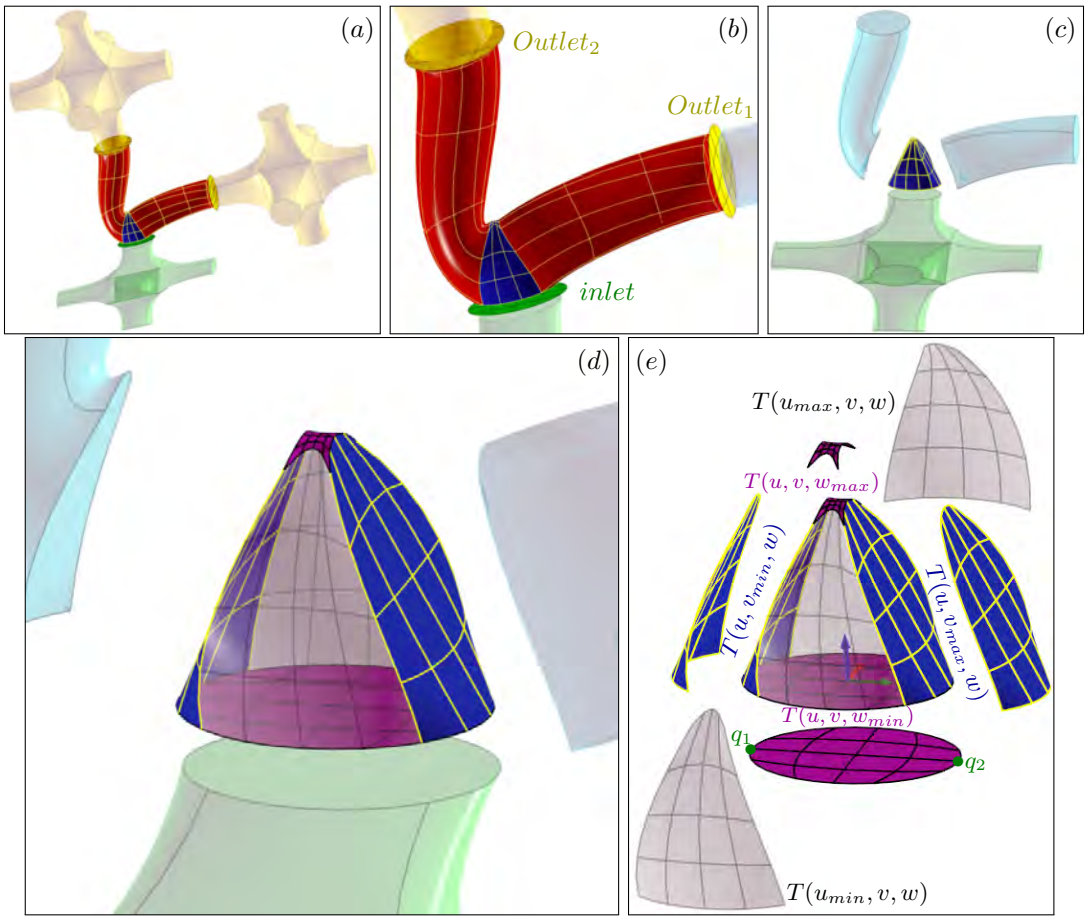

Figure 8: The construction of a bifurcation tile. (a) shows an example of a bifurcation tile consisting of three trivariates: a base (in blue) and two arms (in red). (b) zooms in on (a). (c) and (d) focus on the base trivariate, that is constructed using volumetric Boolean Sum, from six surfaces, as seen in (e).

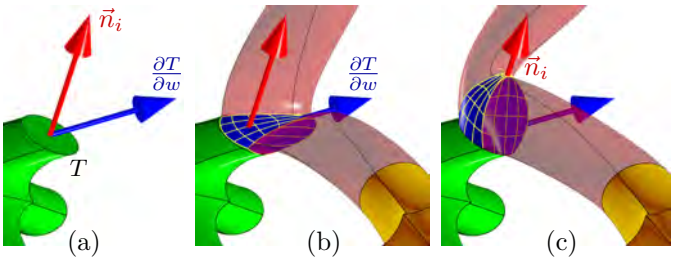

Figure 9: The direction of the base trivariate of the bifurcation tile is adjusted by a user blending parameter ϕ between $\vec{n}_i$, the normal of the anchoring face and $\frac{\partial T}{\partial w}$ as show in (a). In (b), $\phi = 0$ and the direction of $\frac{\partial T}{\partial w}$ is employed, and in (c), $\phi = 1$ and the direction of $\vec{n}_i$ is used.

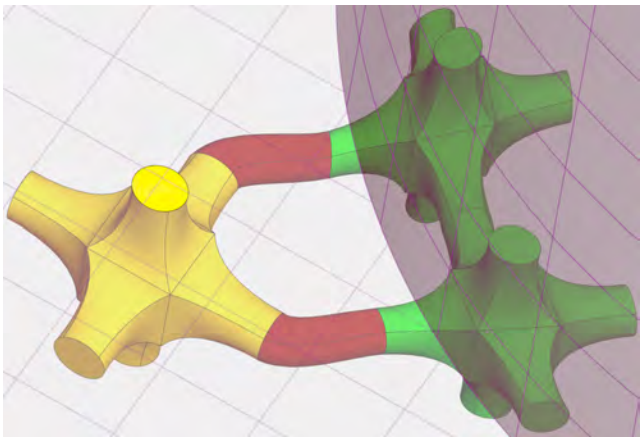

Figure 10: The tubular tiles (in red) bridge inlet and outlet anchoring faces from different primitive microstructure sets.

microstructure set, $\mathcal{M}_\mathcal{A}$ or $\mathcal{M}_\mathcal{B}$, from the primitive with the higher priority, say V_A. Here, the boundary surfaces of the trimmed VModel that originated from V_B (with lower priority) are identified and the microstructure is classified against these trimming surfaces. Bridging tiles are inserted from the anchoring faces of the interior tiles in V_A, to these trimming surfaces. In summary, the microstructure's processing of an intersection case is very similar to the subtraction case, of $V_A - V_B$.

Figure 12 shows two cases, with opposite priorities, for $V_A \cap V_B$. The volumetric primitive in Figure 12 (a) holds the set of microstructures $\mathcal{M}_\mathcal{B}$ (in blue), whereas the volumetric primitive in Figure 12 (d) holds the set of microstructures $\mathcal{M}_\mathcal{B}$ (also in blue). Figures 12 (b) and (c) show the different intersection results based on different primitive priorities. The interior tiles (in green) and the to-be-bridged tiles (in yellow) are different in the two cases

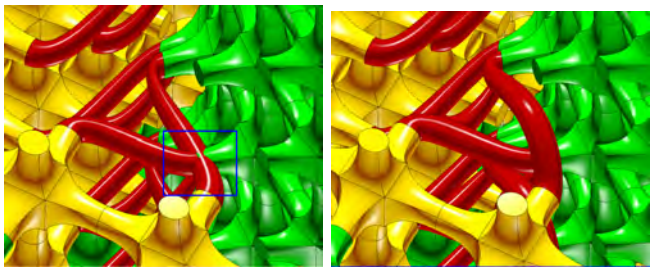

Figure 11: Collisions between neighboring bridging tiles are detected (left, in a blue frame), and resolved (right) by repulsion.

Example	TV_{ttl}	TV_{brg}	Time (sec)
$V_{box} \cup V_{cyl}$ (Fig. 6)	5992	154	18
$V_A \cup V_B$ (Fig. 13(b))	1400	28	35
$V_A \cup V_B - V_C$ (Fig. 13(c))	826	98	38
$V_A - V_B - V_C$ (Fig. 13(d))	384	48	17
$V_A \cap V_B \cap V_C$ (Fig. 14)	208	40	18
WingedDuck (Fig. 16)	39070	297	536

Table 1: Experimental results of the example models presented in this paper. TV_{ttl} represents the total number of trivariates in the VModel and TV_{brg} the number of bridging trivariate tiles. Time is the complete construction time, on the cited machine.

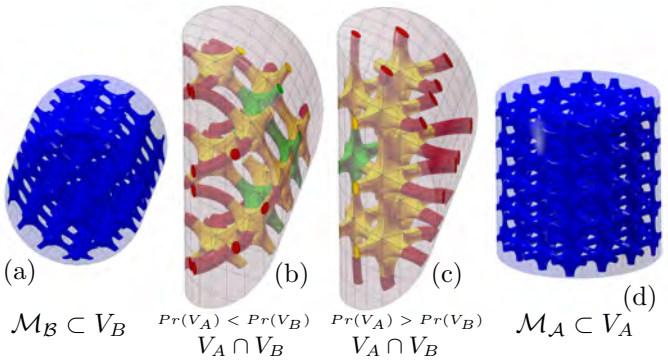

Figure 12: Two different configurations of the microstructures in $V_A \cap V_B$, depending on the priority, $Pr()$, of the input primitives. V_A and B along with their microstructures, $\mathcal{M}_\mathcal{A}$ and $\mathcal{M}_\mathcal{B}$, are shown in (a) and (d) respectively. (b) and (c) present the results for intersections with different primitive's priorities.

as are the bridging tiles (in red) that join different trimming surfaces.

Up until now, we have described our strategies to tile microstructures in a VModel that consists of two primitive tensor product trivariates. The presenting tiling strategies, however, can also be applied to more general VModels. This is obviously relevant and challenging in zones (V-Cells) of more than two trivariates. The key to consistently tiling a general VModel with microstructures is the assignment of priority values to all the primitive trivariates involved in the (Boolean operations in the) VModel. The microstructures are then tiled one V-Cell at a time, and according to these priorities.

As VModels are constructed, applying Boolean operations, a hierarchical tree is created with the final VModel at its root. In every step of these Boolean operations, and as a side operation, we record the information that will be needed for the microstructures' synthesis, such as priorities, the trivariates in each V-Cell, tiles' shape, and desired density of tiles. Consider the union of $V_A \cup V_B$, V_A and V_B are arbitrary VModels, consisting of an arbitrary number of V-Cells each. The V-Cells in $V_A \cup V_B$ are divided into three groups: $V_A - V_B$, $V_B - V_A$ and $V_A \cap V_B$. The primitive trivariates, their priorities, etc., as stored in V_A are propagated to the V-Cells in $V_A - V_B$ (unless a V-Cell of V_A is completely deleted by this subtraction). Similar propagations are executed for $V_A \cap V_B$ and for $V_B - V_A$.

Then, and given the complete VModel, the tiling of the microstructures starts from the V-Cells that hold the primitive trivariate of the highest priority. These V-Cells are processed and tiled first. Traversing the Booleans hierarchy tree, the handling of interior and boundary trimming surfaces is commenced following the description in previous sections, while keeping all tiles on the trimming surface side with the higher priority.

4. Results

In this section, we present several additional results. All were tested on an Intel Core i7-7700K 4.2GHz PC with 32GB of main memory. Table 1 shows the total number of trivariates in the microstructures, as well as the number of bridging trivariates, and the total execution time, in second. Each trivariate is formed of a trivariate spline of either quadratic or cubic degrees. The presented 3D printed examples were fabricated on a Stratasys printer, employing their VoxelPrint [17] interface.

Figure 13 shows microstructures tiled in various V-Models derived from three orthogonal intersecting cylinders. As is shown in Figure 13 (a), each cylinder can be tiled regularly with microstructures, generated by the function composition. Boolean union, subtraction or intersection are then applied to these cylinders to generate the (microstructures in the) VModels in Figure 13 (b)-(d).

Figure 14 shows three instances of microstructures in VModel $V_A \cap V_B \cap V_C$. Three different microstructures are revealed, depending on the priorities of the three cylinders. Compare with Figure 1. Figure 15 also shows a similar example, this time with double the resolution of the microstructure. Here also, the encapsulating VModel was 3D printing transparent, as can be seen in Figure 15 (b).

Finally, Figure 16 shows a winged duck VModel. The two trivariate wings were unioned with the trivariate of the body, only to subtract some hole near the tail. The wings had higher priority here and hence their microstructures were left unmodified. Because of the dense population of tiles in the body, there were some, fully inside the body, tiles that were located very close to the wings. As to-be-bridged tiles, the bridges from them were very short. Hence, we also allow a user-defined margin and purge such too-close to-be-bridged tiles, during the tile classification process.

Note that near the root of the wing, the resolution of the tiles of the body is much higher than the resolution of the microstructure of the wing and yet, with the aid of bifurcation bridging tiles, the result is quite pleasing. Also, in Figure 16 (e) and (f), fairly high-frequency colors were encoded into all trivariates, exemplifying, once more, the heterogeneity capability.

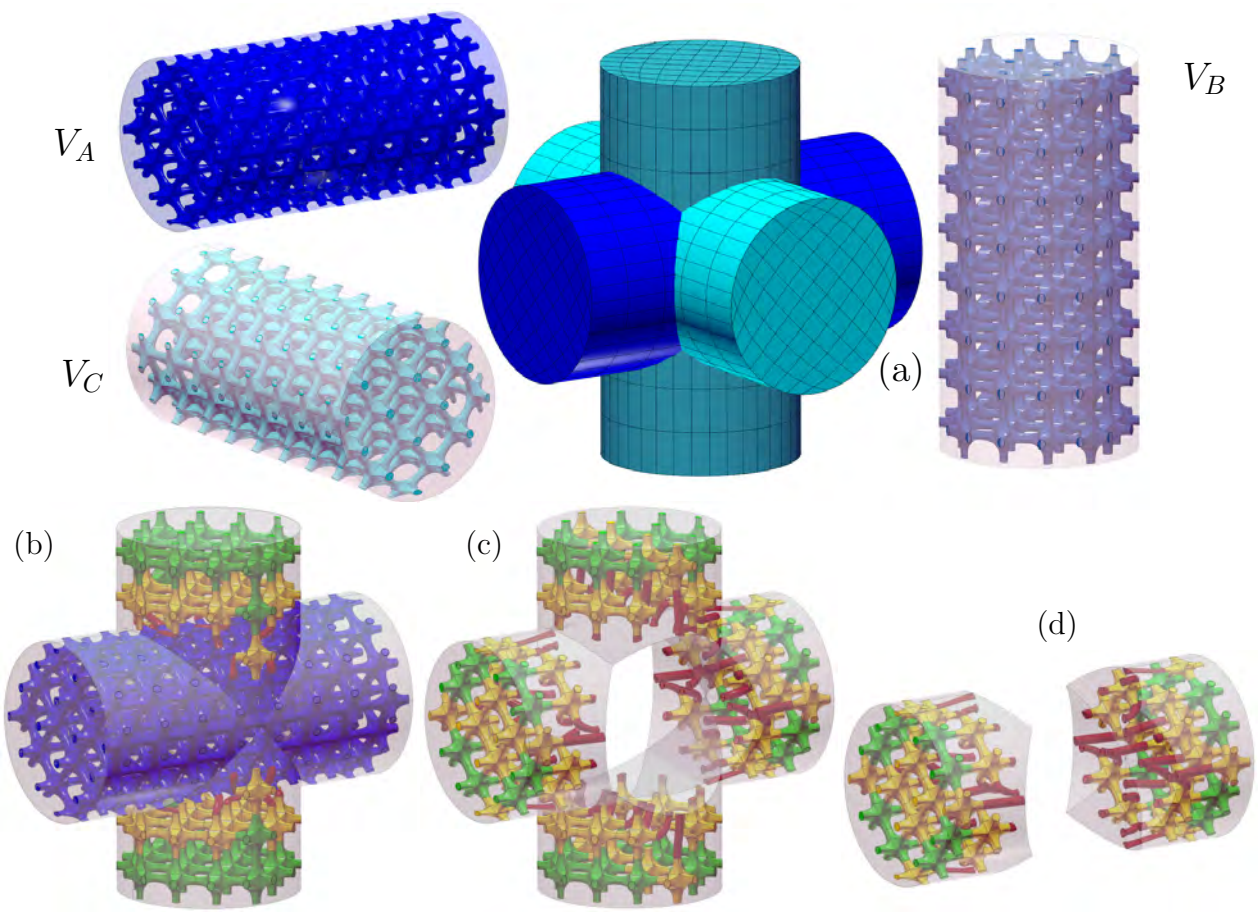

Figure 13: Microstructures in VModels undergoing different Boolean operation from three orthogonal cylinders, V_A, V_B and V_C, having (a) as input. (b) $V_A \cup V_B$, (c) $(V_A \cup V_B) - V_C$ and (d) $V_A - V_B - V_C$. The colors of the tiles in the trimmed V-Cells are set to green, in (b)-(d), if inside tiles, except for the to-be-bridged inside tiles that are set to yellow. In (b), all tiles of V_A are kept (no classification of tiles in V_A is needed) so they preserve their original color.

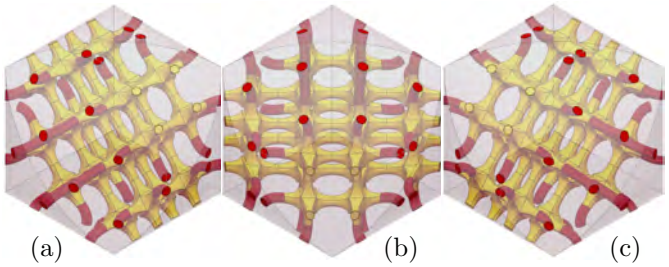

Figure 14: The microstructures in $V_A \cap V_B \cap V_C$ (three intersecting cylinders) where the three V_is are the same as in Figure 13. (a): V_A has highest priority, (b): V_B has highest priority, (c): V_C has highest priority. See also Figures 1 and 15

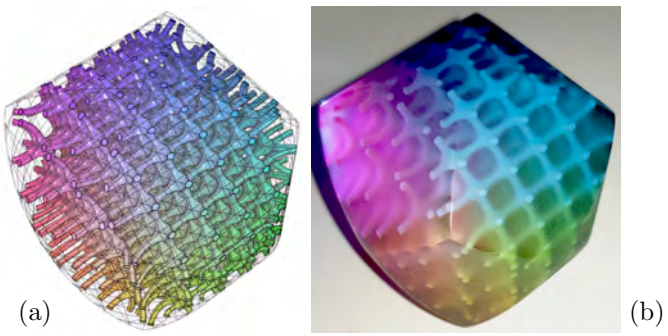

Figure 15: A similar result to Figure 1, 3D printed with double the microstructure resolution, and a transparent encapsulating VModel.

While barely discussed until now, this work is not limited to tiles with circular cross-sections. In Figure 16, the cross-sections of the tiles of the body are rectangles and those in the wing are circular. Yet, the bridging tiles are smoothly blended from one cross section to the other.

5. Conclusions and Future Work

This work presented a scheme to populate a full VModel consisting of several trimmed trivariates, using microstructures. The result is a set of tensor product trivariates that are fully compatible with iso-geometry. Extra care has been taken, to ensure that adjacent trivariates conform and match so no mortar will be necessary. Similarly, being a volumetric result, we could attach attributes (e.g., color) to the geometry in full (not just the boundaries) and 3D print the geometry with colors. While, in this work, we focused on trivariate tiles, the presented scheme can clearly be adapted to tiling with surface geometry and/or univariates (curves), being simpler cases.

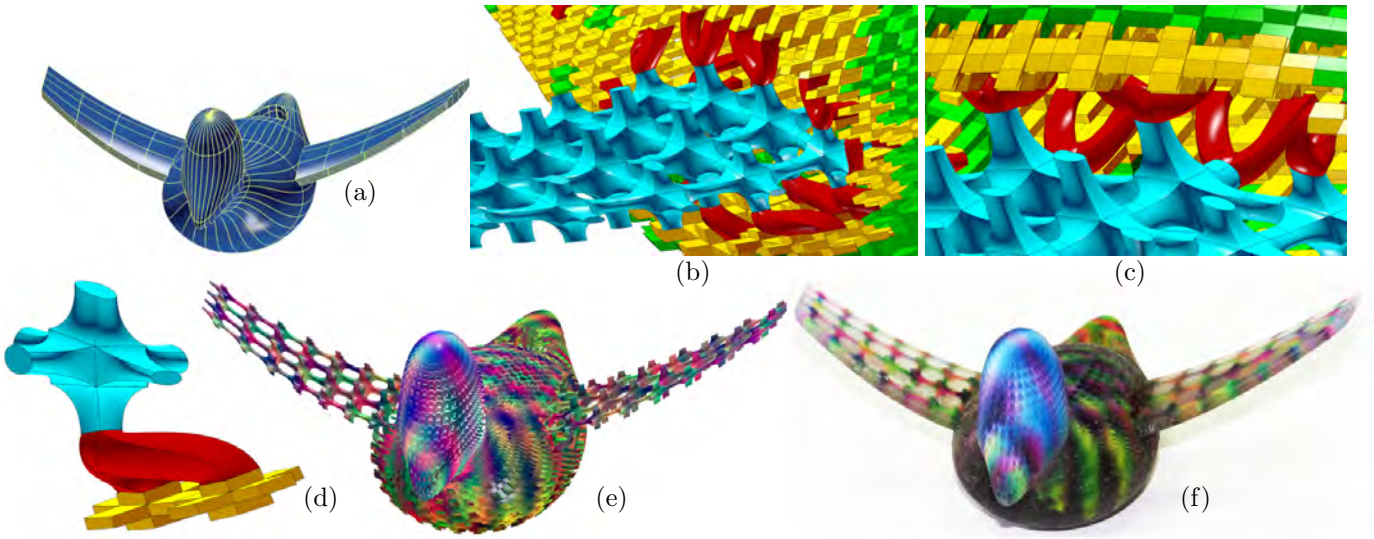

Figure 16: A VModel of a winged duck in (a), is populated with two types of tiles - rectangular cross-sections in the body and rounded in the wings, all unioned together and properly stitched. (b) and (c) show zoom-ins on the bridging tiles between the wing and the body (the wings had higher priority here and hence, their microstructures are unmodified and in cyan). One of the two-to-one bridging tiles is shown (in red) with the adjacent tiles in (d). Note the rectangle cross-section tile (in yellow) from the body and the rounded cross-section tile (in cyan) from the wing. (e) shows the VModel after random graded colors were encoded into its trivariates and (f) presents the final 3D printed sample, in a transparent encapsulating VModel.

Performing the microstructure synthesis as a post-process to the Booleans, and once the VModel is complete, allowed us to optimize the microstructure's synthesis and skip the computation of microstructures in primitives that were, for example, purged away. That is, ignore the microstructures of V_B in $V_A - V_B$.

In the case of synthesizing the bridging tiles in a union, the current scheme might leave some anchoring surfaces of tiles unconnected, when the resolution of the tile sets differ significantly. A remedy might be found by considering other configurations of matchings, such as three-to-one or even more generalized M-to-N matching, with all the geometric challenges that such generalized matchings can impose over the bridging tiles.

A clear next step will be to analyze such arrangement of microstructure and see how to optimize them on one side and ensure that the bridging tiles are not forming weak/bottleneck locations, being less regular. Clearly and while the principles were laid down in this work, the presented matching, bridging and repelling schemes can benefit from further research and improvement.

Acknowledgments

We would like to thank the anonymous reviewers for their comments and suggestions. This project has received funding in part by from the European Union Horizon 2020 research and innovation programme, under grant agreement No. 862025 and in part by the ISRAEL SCIENCE FOUNDATION (grant No. 597/18).

[1] Antolin, P., Buffa, A., Cohen, E., Dannenhoffer, J. F., Elber, G., Elgeti, S., Haimes, R., and Riesenfeld, R. Optimizing micro-tiles in micro-structures as a design paradigm. *Computer-Aided Design 115* (2019), 23–33.

[2] Aremu, A., Brennan-Craddock, J., Panesar, A., Ashcroft, I., Hague, R., Wildman, R., and Tuck, C. A voxel-based method of constructing and skinning conformal and functionally graded lattice structures suitable for additive manufacturing. *Additive Manufacturing 13* (2017), 1–13.

[3] Bendsoe, M. P., and Sigmund, O. *Topology Optimization: Theory, Methods, and Applications*. Springer, 2004.

[4] Dokken, T., Skytt, V., and Barrowclough, O. Trivariate spline representations for computer aided design and additive manufacturing. *Computers & Mathematics with Applications 78*, 7 (2019), 2168–2182. Simulation for Additive Manufacturing.

[5] Elber, G., Kim, Y.-J., and Kim, M.-S. Volumetric boolean sum. *Computer Aided Geometric Design 29*, 7 (2012), 532 – 540. Geometric Modeling and Processing 2012.

[6] Feng, J., Fu, J., Shang, C., Lin, Z., Niu, X., and Li, B. Efficient generation strategy for hierarchical porous scaffolds with freeform external geometries. *Additive Manufacturing 31* (2020), 100943.

[7] Gupta, A., Allen, G., and Rossignac, J. Quador: Quadric-of-revolution beams for lattices. *Computer-Aided Design 102* (2018), 160–170. Proceeding of SPM 2018 Symposium.

[8] Liu, Y., Zheng, G., Letov, N., and Zhao, Y. F. A survey of modeling and optimization methods for multi-scale heterogeneous lattice structures. *The International Journal of Advanced Manufacturing Technology 143*, 4 (2021), 040803.

[9] Massarwi, F., and Elber, G. A B-spline based framework for volumetric object modeling. *Computer Aided Design 78* (2016), 36–47.

[10] Massarwi, F., Machchhar, J., Antolin, P., and Elber, G. Hierarchical, random and bifurcation tiling with heterogeneity in micro-structures construction via functional composition. *Computer Aided Design 102* (2018), 148–159.

[11] Medeiros e Sá, A., Mello, V. M., Rodriguez Echavarria, K., and Covill, D. Adaptive voids. *The Visual Computer 31*, 6 (Jun 2015), 799–808.

[12] Munkres, J. Algorithms for the assignment and transportation problems. *Journal of the Society for Industrial and Applied Mathematics 5*, 1 (1957), 32 – 38.

[13] Nazir, A., Abate, K. M., Kumar, A., and Jeng, J.-Y. A state-of-the-art review on types, design, optimization, and additive manufacturing of cellular structures. *The International Journal of Advanced Manufacturing Technology 104*, 9 (2019), 3489–3510.

[14] Nguyen, J., Park, S.-i., Rosen, D., Folgar, L., and Williams, J. Conformal lattice structure design and fabrication. *23rd Annual International Solid Freeform Fabrication Symposium - An Additive Manufacturing Conference, SFF 2012* (01 2012), 138–161.

[15] Pasko, A., Fryazinov, O., Vilbrandt, T., Fayolle, P.-A., and Adzhiev, V. Procedural function-based modelling of volumetric microstructures. *Graphical Models 73*, 5 (2011), 165 – 181.

[16] Schaedler, T. A., and Carter, W. B. Architected cellular materials. *Annual Review of Materials Research 46*, 1 (2016), 187–210.

[17] Stratasys. Voxelprint technology, https://help.grabcad.com/article/230-guide-to-voxel-printing?locale=en, 2020.

[18] Tamburrino, F., Graziosi, S., and Bordegoni, M. The design process of additively manufactured mesoscale lattice structures: A review. *Journal of Computing and Information Science in Engineering 18*, 4 (2018), 040801.

[19] Tang, Y., Dong, G., and Zhao, Y. A hybrid geometric modeling method for lattice structures fabricated by additive manufacturing. *The International Journal of Advanced Manufacturing Technology 102*, 9 (06 2019), 4011–4030.

[20] van Sosin, B., Rodin, D., Sliusarenko, H., Barto, M., and Elber, G. The construction of conforming-to-shape truss lattice structures via 3d sphere packing. *Computer-Aided Design 132* (2021), 102962.

[21] Wang, H., Chen, Y., and Rosen, D. A hybrid geometric modeling method for large scale conformal cellular structures. In *Volume 3: 25th Computers and Information in Engineering Conference, Parts A and B* (January 2005).