



# Procedural Metamaterials: A Unified Procedural Graph for Metamaterial Design

LIANE MAKATURA and BOHAN WANG, MIT, USA  
 YI-LU CHEN, ISTA, Austria  
 BOLEI DENG, MIT, USA  
 CHRIS WOJTAN and BERND BICKEL, ISTA, Austria  
 WOJCIECH MATUSIK, MIT, USA

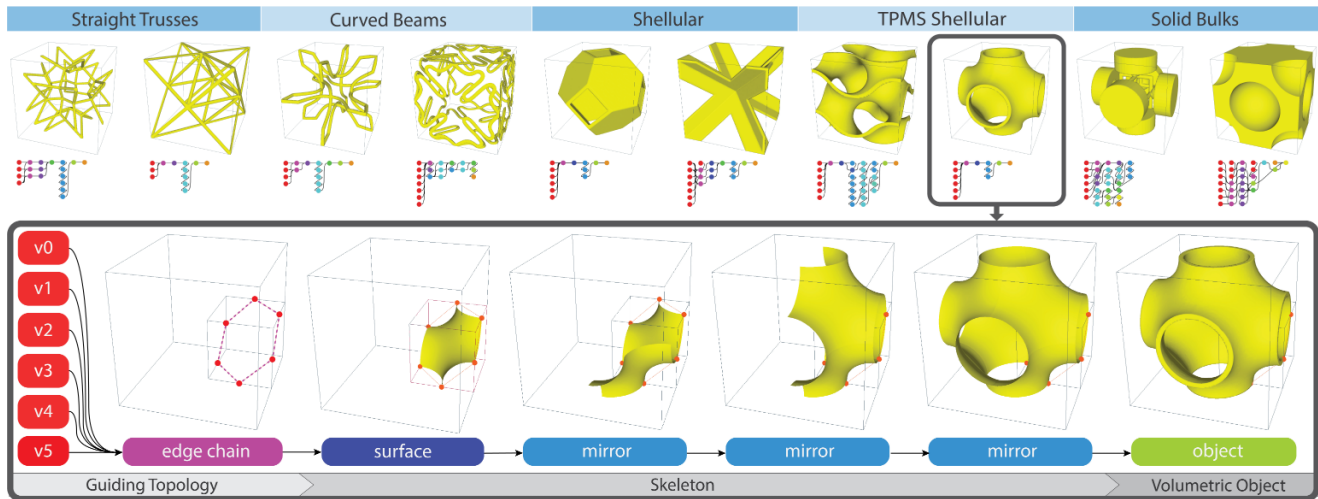


Fig. 1. (Top) Structures spanning five major classes of cellular architectures, all of which can be expressed compactly via our procedural graphs (beneath each structure, nodes colored by operation type). (Bottom) Expanded view of our procedural graph for the Schwarz P structure, with a visualization of the construction process: our operations transform simple guiding topology into a skeleton that is solidified according to a spatially varying thickness function.

We introduce a compact, intuitive procedural graph representation for cellular metamaterials, which are small-scale, tileable structures that can be architected to exhibit many useful material properties. Because the

L. Makatura and B. Wang contributed equally to this research. This material is based upon work supported by the National Science Foundation (NSF) Graduate Research Fellowship under Grant No. 2141064; the MIT Morningside Academy for Design Fellowship; the Defense Advanced Research Projects Agency (DARPA) Grant No. FA8750-20-C-0075; the ERC Consolidator Grant No. 101045083, “CoDiNA: Computational Discovery of Numerical Algorithms for Animation and Simulation of Natural Phenomena”; and the NewSat project, which is co-funded by the Operational Program for Competitiveness and Internationalisation (COMPETE2020), Portugal 2020, the European Regional Development Fund (ERDF), and the Portuguese Foundation for Science and Technology (FTC) under the MIT Portugal program. Authors’ addresses: L. Makatura, B. Wang, B. Deng, and W. Matusik, MIT, 77 Massachusetts Ave, Cambridge, MA, 02139, USA; emails: {makatura, bohanw, boleiden}@mit.edu, wojciech@csail.mit.edu; Y.-L. Chen, C. Wojtan, and B. Bickel, ISTA, Am Campus 1, 3400 Klosterneuburg, Austria; emails: {yi-lu.chen, chris.wojtan, bernd.bickel}@ist.ac.at.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).  
 © 2023 Copyright held by the owner/author(s).  
 0730-0301/2023/07-ART168 \$15.00  
<https://doi.org/10.1145/3605389>

structures’ “architectures” vary widely—with elements such as beams, thin shells, and solid bulks—it is difficult to explore them using existing representations. Generic approaches like voxel grids are versatile, but it is cumbersome to represent and edit individual structures; architecture-specific approaches address these issues, but are incompatible with one another. By contrast, our procedural graph succinctly represents the construction process for any structure using a simple skeleton annotated with spatially varying thickness. To express the highly constrained **triply periodic minimal surfaces (TPMS)** in this manner, we present the first fully automated version of the conjugate surface construction method, which allows novices to create complex TPMS from intuitive input. We demonstrate our representation’s expressiveness, accuracy, and compactness by constructing a wide range of established structures and hundreds of novel structures with diverse architectures and material properties. We also conduct a user study to verify our representation’s ease-of-use and ability to expand engineers’ capacity for exploration.

CCS Concepts: • **Computing methodologies** → **Modeling methodologies; Shape modeling**; • **Applied computing** → *Computer-aided design*;

Additional Key Words and Phrases: Graph representation, cellular metamaterials, microstructures, shellular, triply periodic minimal surfaces (TPMS), truss structures, hybrid metamaterials, conjugate surface construction method, procedural graph, procedural representation

**ACM Reference format:**

Liane Makatura, Bohan Wang, Yi-Lu Chen, Bolei Deng, Chris Wojtan, Bernd Bickel, and Wojciech Matusik. 2023. Procedural Metamaterials: A Unified Procedural Graph for Metamaterial Design. *ACM Trans. Graph.* 42, 5, Article 168 (July 2023), 19 pages. <https://doi.org/10.1145/3605389>

**1 INTRODUCTION**

*Metamaterials* are structures of long-standing interest, as they induce material properties that differ from those of their constituent base material(s). Metamaterials often exhibit behaviors that are not found in nature, such as tuneable compliant, chiral, auxetic and non-reciprocal behaviors [Jenett et al. 2020; Ou et al. 2018; Panetta et al. 2015], and impressive strength-to-weight ratios [Qin et al. 2017].

The behavior of a given metamaterial is primarily governed by its *cellular architecture*, which is the regular or random spatial arrangement of solid regions and voids used to fill a designated volume [Schaedler and Carter 2016]. The set of possible cellular architectures is uncountably large, even when we restrict our attention to, e.g., the space of regular cellular solids that fill a unit cube and tile periodically in  $\mathbb{R}^3$ , as shown in Figure 1. Figure 1 also displays the many architectural elements that occur within our subspace, such as straight/curved beams, thin shells, and solid bulks. This breadth is powerful, as each architectural class offers a unique set of strengths [Bertoldi et al. 2017; Gibson et al. 2010; Surjadi et al. 2019].

However, these distinct classes complicate metamaterial design, because no existing representation is well suited for all structures. Generic representations like voxel grids can express any structure, but the specification is cumbersome and difficult to edit, since even simple changes (e.g., thickening a beam) trigger many independent voxel updates. Class-specific representations are often more practical, as each structure’s description is compact and editable. However, the underlying specifications are as varied as the structures themselves: trusses and beams are often given by graphs; solid bulks stem from **constructive solid geometry (CSG)** operations; and thin-shell cellular (*shellular*) structures use surface meshes or implicit functions. These specifications are not immediately compatible with one another, so it is difficult to explore over more than one class.

In some cases, it is difficult to explore variations even *within* a given class, as each structure requires a unique derivation. For example, many shellular metamaterials are based on TPMS, which are precisely defined as the integral of an Enneper–Weierstraß function. For ease of use, TPMS are commonly approximated by, e.g., level sets of an implicit trigonometric function. However, both function types are structure specific and difficult (if not impossible) to derive. This limits the set of structures accessible to engineers and, thus, available for exploration.

To alleviate these challenges, we propose a procedural graph representation that streamlines the process of metamaterial design for a wide range of common classes, including generic TPMS. Our representation is *specific* to cellular metamaterial design, which allows us to capitalize on characteristics like symmetries while ensuring that our representation is equally suitable for all target classes. Moreover, our representation is *compact, intuitive,*

and *easily editable*, such that it is amenable to manual or automated design space exploration. Finally, it is *expressive* enough to represent not only *known* structures, as shown in Figure 1, but also *novel* structures containing elements from one or more class(es).

The most critical aspect of our representation is the unified *skeletal design space*, which uses simple elements like lines and surfaces to capture the wide range of shapes found in metamaterials. Each skeletal element is constructed from a small set of simple, high-level specifications comprised of vertices and edges, a bounding volume, and the type of skeletal element (e.g., line, surface) to be instantiated. Each element is also annotated with a spatially varying thickness function that determines how it should eventually be thickened into a physically realizable volume (e.g., beam, shell, or solid bulk).

Although this design space naturally accommodates most of our target classes, the TPMS shellulars present a considerable obstacle. Our TPMS approach is grounded in mathematical principles, yet compatible with the design space described above and intuitive enough to be used by novices. A critical element of our approach is the **conjugate surface construction method (CSCM)**, which is “one of the most powerful techniques to construct minimal surfaces with a proposed shape in mind” [Karcher and Polthier 1996]. However, existing CSCM algorithms are largely inaccessible, as they require extensive domain expertise and human intervention. We embed the CSCM in an optimization loop to realize the first fully automatic version of this pipeline, making it accessible to a wider audience.

In summary, our contributions include the following:

- a practical algorithm for TPMS via our extended CSCM,
- a unified skeletal design space that compactly expresses the thickness-annotated skeletons for a wide range of metamaterials, including the five major classes in Figure 1, and
- an intuitive procedural graph representation that facilitates the exploration and evaluation of novel structures.

We validate our approach by constructing hundreds of structures with diverse architectures. We also conduct a user study to verify our representation’s intuitiveness and ease-of-use. Finally, although we defer guided search strategies and physical property validation to future work, we show the potential of our approach by defining simple, random exploration schemes that automatically generate truss and shellular structures with a wide range of material properties.

**2 RELATED WORK****2.1 Cellular Architectures**

The notion of metamaterials is very broad: even within graphics, HCI, and ML, recent research includes two-dimensional (2D) metamaterial sheets that are embedded in 3D [Konaković et al. 2016; Martínez et al. 2019; Signer et al. 2021]; interactive metamaterial mechanisms [Ion et al. 2016, 2019, 2017]; functionally graded structures for, e.g., spatially varying elasticity [Schumacher et al. 2015]; and multi-material composites with engineered properties [Gongora et al. 2021]. Although our approach may also apply to these domains, we restrict our focus to static 3D cellular

metamaterials whose unit cells are regular (rather than random), tilable in  $\mathbb{R}^3$ , and confined to a unit cube.

**Trusses and Beams.** Trusses and beams are often used for metamaterials, as they are easy to specify yet widely varied. The space of truss topologies alone is large enough to demand its own taxonomy [Zok et al. 2016], even *before* accounting for continuous parameters like vertex positions or thickness profiles over beams and their junctions. To explore this space, Jenett et al. [2020] and Frenzel et al. [2017] hand-designed several curved-beam structures exhibiting chiral, auxetic, rigid, and compliant behaviors. For diverse Poisson’s ratio and Young’s modulus values, Panetta et al. [2015] described 1,205 truss-based topologies using a tetrahedral decomposition of a cube and a graph over 15 possible nodes. Bastek et al. [2022] created 262 topologies by deforming and superimposing seven fundamental lattice units. Chen et al. [2018] used topology optimization to compute candidate structures, over which they fit graph templates to create parametrized truss families with extremal properties. Our representation is reminiscent of these approaches and trivially captures truss- or beam-based topologies that reside in a unit cube. However, we expand this powerful graph-based representation by porting it to architectures that are traditionally less amenable to exploration.

**Solid Bulks.** Solid bulks appear in many forms, including non-periodic spinoid topologies [Kumar et al. 2020], foams [Ashby 2006], and open-cell porous structures [Tian et al. 2020]. However, we restrict ourselves to regular topologies, such as the topology-optimized, cubic-symmetric structures of Schumacher et al. [2015]. Chan et al. [2020] proposed another set of cubic-symmetric structures based on the level sets of 36 crystallographic space groups. Many CSG-like structures have also been hand-designed for specific properties like phononic bandgap [Muhammad and Lim 2021]. Our representation covers the vast majority of these regular structures.

**Shellulars.** Shellular units are often based on 2-manifold surfaces [Nguyen et al. 2016] or non-manifold surfaces like cubic lattices and honeycomb cells [Spadoni et al. 2014]. Several works also design functionally graded porous structures by spatially varying these base surfaces [Hu et al. 2022b; Lu et al. 2014]. Our representation captures a wide range of (non-)manifold surfaces, but the latter examples are out of scope, as we focus on individual cells.

**TPMS Shellulars.** TPMS are minimal surfaces that tile seamlessly along three mutually orthogonal directions in  $\mathbb{R}^3$ . Non-self-intersecting TPMS are of particular interest, as they partition the surrounding volume into two or more distinct labyrinthine channels. Shellulars based on TPMS are smooth and uniquely suitable for additive manufacturing [Hu et al. 2022b; Qin et al. 2017; Yan et al. 2021, 2020], making them ideal for tasks such as bone scaffolding [Ambu and Morabito 2019; Ataei et al. 2018], static mixing [Ouda et al. 2020], thermal energy management [Attarzadeh et al. 2022; Fan et al. 2022], and strength-preserving lightweighting [Qin et al. 2017]. TPMS are often created by Enneper–Weierstraß functions, specialized triangle meshes [Reitebuch et al. 2019], or level sets of implicit functions. The latter have enabled several TPMS explorer tools [Al-Ketan and Abu Al-Rub 2021; Hsieh and Valdevit 2020; Jones et al. 2021; Maskery et al. 2022] and methods for

hybrid TPMS creation. Feng et al. [2021] and Khaleghi et al. [2021] construct hybrid TPMS by extracting isosurfaces from weighted sums of TPMS’ trigonometric approximations, while Chen et al. [2019] and Feng et al. [2021] use mesh Booleans. Zhang et al. [2021] propose hierarchical TPMS structures by assembling small-scale TPMS cells into larger ones. Although promising, these methods are limited by their reliance on structure-specific TPMS representations, which are difficult to construct and combine with other classes. Our approach addresses both limitations.

Alternatively, Akbari et al. [2020, 2022] construct surface- and truss-based approximations of TPMS from physical principles by using 3D graphic statics to explore the dual graphs of geometric *form* and static *forces*. To design hybrid TPMS, they specify the target labyrinth(s), i.e., channels of negative space surrounding the surface, and then construct a surface that partitions the volume accordingly. This scheme is powerful, but even domain experts struggle to relate labyrinth–surface pairs [Schoen 2008]. Our approach takes input describing the surface itself rather than its voids.

As discussed in Section 1, a critical element of our TPMS approach is the CSCM, which was introduced by Lawson [1970] in  $\mathbb{S}^3$  and then adapted to  $\mathbb{R}^3$  by Karcher [1989] with a discrete approach by Pinkall and Polthier [1993]. This method (described in Section 4) leverages a surface’s *associate family* to construct complex TPMS indirectly. We introduce the first automated, easy-to-use pipeline for this method.

## 2.2 Relevant Modeling Methods

**Procedural Modeling.** Procedural models “encapsulate a large variety of shapes into a concise formal description that can be efficiently parametrized” [Krs et al. 2021], which lends them to a variety of tasks including 2D textures and shaders [Cook 1984; Hu et al. 2022a; Perlin 1985; Shi et al. 2020] and virtual world modeling [Prusinkiewicz and Lindenmayer 2004; Smelik et al. 2014; Whiting et al. 2009]. Graph-based models are of particular interest, as they are widely used in practice (e.g., SideFX Houdini, Blender, Adobe Substance Designer), and they are amenable to performance optimization [Boechat et al. 2016] and the intuitive specification of edits and constraints [Krs et al. 2021; Michel and Boubekeur 2021]. Our procedural graph builds on these ideas toward concise, intuitive metamaterial design.

**Skeleton-based Modeling.** Skeletons have long been explored for efficient shape representation, as many volumetric shapes are well approximated by lower-dimensional structures [Bærentzen and Rotenberg 2021; Blum 1967]. Tagliasacchi et al. [2016] survey the rich skeletonization literature, which generally tries to *reduce* a volume to a skeleton. By contrast, we specify skeletons to *construct* a volume. Although most approaches approximate shapes with *curve networks*, Tagliasacchi et al. [2012] observe that some shapes are best represented by *meso-skeletons* containing a mix of curves and surfaces. As such, we develop a concise meso-skeleton representation for shapes appearing in metamaterial design.

## 3 OVERVIEW

As suggested by Figure 1, regular cellular architectures are well suited for skeleton-based design, as they are often highly symmetric structures derived from lines, surfaces, and easily reducible solid primitives. We could imagine modeling any such

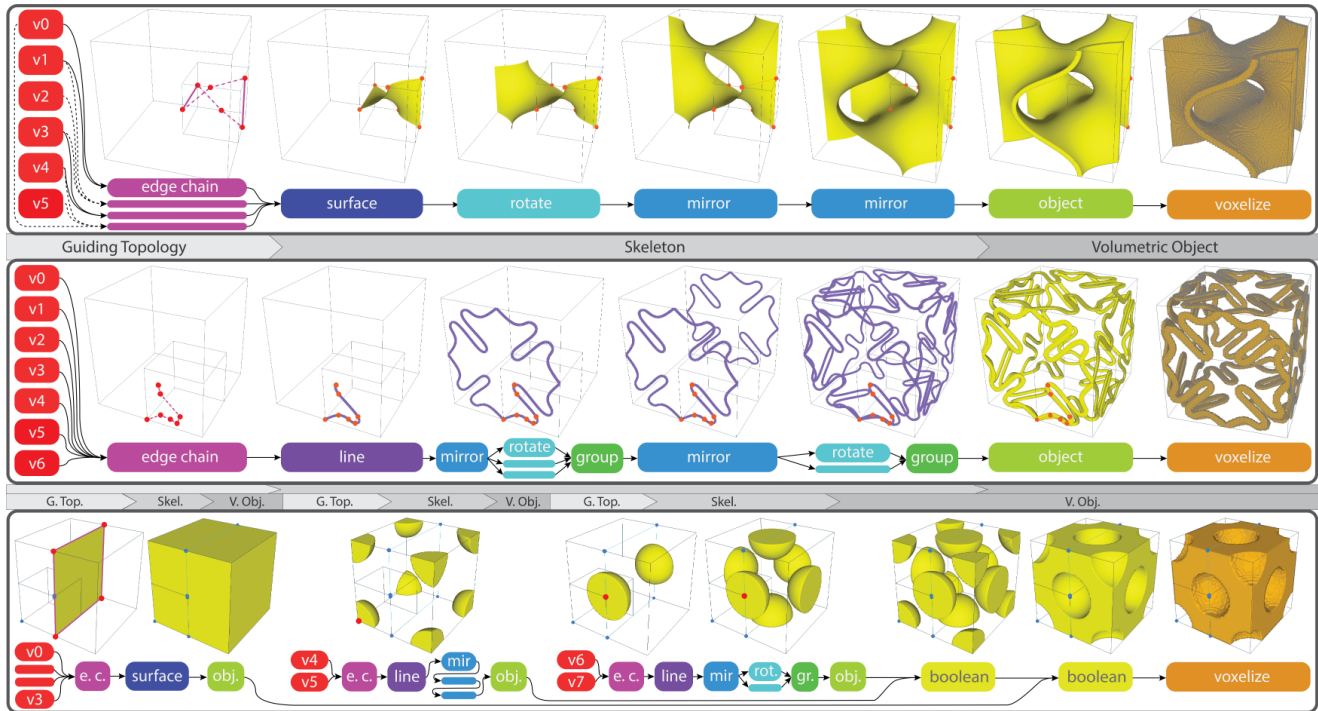


Fig. 2. **Metamaterial construction.** Our graphs succinctly create structures based on (top) a recently discovered TPMS [Chen and Weber 2021], (middle) auxetic curved beams [Jenett et al. 2020], and (bottom) a face-centered cubic solid [Lu et al. 2017]. Edge chains can be smooth (dashed) or non-smooth (solid).

metamaterial with four simple steps: (1) build the skeleton for a small fundamental piece of the structure, (2) assign a spatially varying thickness profile  $T(p)$  for each point  $p$  of the skeleton, (3) apply any transformations (e.g., mirroring, rotation) required to fill the tiling unit, and (4) realize the final volumetric object according to  $T(p)$ . To achieve this simple approach, we must address three challenges. First, we characterize the required skeletal elements and parametrize them in a concise, consistent manner. Then, we embed our boundary portion in a restricted skeletal design space in a representation that captures the four-step approach above. Finally, we envision a user-centric modeling process that is easy and intuitive.

### 3.1 Skeletal Design Space

Driven by our five target classes, our skeletal design space must accommodate straight/curved beams, planar/curved shells, and basic volumetric primitives such as cuboids and spheres. We posit that *line* and *surface* skeletons are sufficient to capture these shapes, when paired with a few annotations. To motivate and codify this design space, we briefly examine the needs of each shape category.

**Beams.** Straight and curved beams are well represented by line skeletons, which follow a path given by an ordered list of vertices. Curves can be created from concise vertex lists via, e.g., natural cubic spline interpolation. Thus, we need only introduce a “smoothness” flag to determine whether individual segments should be straight or smoothed. The final cross section of the thickened beams can be controlled by a spatially varying thickness profile over the line.

**Shells.** Shells are best represented by surface skeletons annotated with a spatially varying thickness profile, as above. Surface skeletons are also amenable to an ordered-vertex parametrization, as surfaces are frequently generated over a target boundary loop. For example, the widely studied *Plateau problem* spans a fixed boundary with a *minimal surface*, which locally minimizes surface area and has zero mean curvature everywhere [Harrison and Pugh 2016; Wang and Chern 2021]. Minimal surfaces can also be given by the *free boundary problem*, in which the input boundary is not fixed: each boundary portion is restricted to lie in its given *plane*, but its specific shape is inferred automatically as it “slides” along the plane to improve the metric [Karcher 1989]. For general TPMS construction, both boundary types are required. To capture them, we devise a pair of *sliding solvers* that take smoothness-annotated boundaries as input: smooth boundary portions are permitted to slide, while non-smooth portions remain fixed. The *mixed minimal sliding solver* is used when at least one fixed edge is present, as in Figure 2 (top). For fully sliding boundaries—which may otherwise degenerate—we introduce the *conjugate solver*, based on our extended CSCM. Finally, we introduce *direct solvers* to generate (not necessarily minimal) surfaces over fully fixed boundaries; here, non-smooth boundary portions remain straight while smooth portions are interpolated. All of our solvers assume disc topology (genus 0), as higher-genus surfaces can generally be decomposed. Critically, this also holds for TPMS: despite having  $\text{genus} \geq 3$  [Garbuz 2010; Meeks 1975], TPMS can be decomposed via their *symmetry lines* (see Section 4.1).

**Volumetric Primitives.** When combined with simple thickening methods, lines and surfaces yield many basic primitives. For

example, by offsetting the thickness along the skeleton’s normal direction(s), we can transform a line into a cylinder or a square-bounded planar surface into a cuboid. Similarly, by sweeping a sphere of the desired radius across our line/surface skeleton, we can create cylinders/cuboids with rounded ends/edges. This yields a sphere if the underlying line has length zero, as in Figure 2 (bottom).

*Unified Design Space.* In summary, line and surface skeletons capture our full set of target structures. Each skeletal element can be given by (1) a smoothness-annotated path over 3D vertices, (2) the skeletal type/solver, and (3) a spatially varying thickness profile over the element’s domain. For sliding solvers, we also provide (4) a set of bounding planes. Sections 4 and 5 explore the technical aspects of this design space.

### 3.2 Unified Procedural Graph

To facilitate the four-stage modeling process posed above, we create a procedural graph that unifies our skeletal design space with other pertinent operations. As shown in Figures 1 and 2, each graph node performs an operation such as vertex creation, line/surface inference, mirroring, or skeleton thickening. Each node also has properties that control its behavior. For example, an edge chain has a smoothness flag, and each surface node has a solver type and a thickness profile. By chaining and sequentially evaluating these nodes, we can form a variety of structures. We can also directly integrate nodes for performance evaluation, including, e.g., voxelization and simulations for material property prediction. Section 6 provides the full set of operations, along with their properties and implementation details.

### 3.3 User Design Process

To conceptualize a structure in our representation, users work backward through the stages of our procedural graph. First, users identify symmetries (e.g., mirrors, rotations) to reduce the structure to its smallest representative unit(s). After discounting thickness, they arrive at the structure’s **fundamental skeleton (FS)**, which resides in the **fundamental bounding volume (FBV)**. We support three scalable FBV primitives (cuboid, triangular prism, and tetrahedron) and custom FBVs. An FBV typically occupies a small part of the unit cube, though this need not be the case (Figure 2, bottom). Structures may also contain multiple FS, each residing in a unique FBV. Once each FS is identified, users can begin building the graph. The constituent vertices and edge chains for each FS are given by tracing its guiding/bounding path and classifying each portion as (non-)smooth. If a boundary contains both smooth and non-smooth portions, then multiple edge chains are required (Figure 2, top). The edge chains are then fed into a skeleton node, which can be transformed as needed to fill the unit cube and then thickened into an object. As the graph takes shape, users must verify that all nodes’ required conditions are met, e.g., for sliding solvers, all smooth boundary portions must lie on an FBV face. All such requirements are detailed in Section 6.

### 3.4 Outline

To build up a full understanding of our representation, we first explain our conjugate surface solver (Section 4), which is critical

for many popular TPMS structures, and also our primary technical challenge. In Section 5, we discuss the remaining surface and line solvers, together with our spatially varying thickness annotations. The procedural graph is detailed in Section 6 and then evaluated for expressiveness, compactness, and intuitiveness in Section 7, which includes a user study.

## 4 CONJUGATE SURFACE CONSTRUCTION

Our conjugate surface solver provides a stable approach for TPMS whose FS is the solution to a free boundary problem, i.e., the FS boundary consists entirely of sliding edges that are permitted to change in order to reduce surface area. Under standard approaches such as mesh-based gradient flows [Brakke 1992; Dziuk 1990], these problems are prone to degeneration (see Supplementary Materials and [Karcher and Polthier 1996]). However, by leveraging the *associate family* of a minimal surface, the CSCM provides a stable three-step approach to these problems, as shown in Figure 3: (1) Construct a related Plateau problem, (2) solve it, and then (3) apply a simple transformation to obtain the solution to our original problem [Karcher 1989; Karcher and Polthier 1996; Pinkall and Polthier 1993]. Although this approach is powerful, existing algorithms require considerable domain expertise. We introduce a novel optimization loop to automate the CSCM and make it accessible to novices. Although we defer a detailed treatment of the CSCM to prior works, we provide the critical intuition (Section 4.1) before discussing our extension (Section 4.2).

### 4.1 Background and Overview

An *associate family*  $F^\phi(u, v)$  is a set of minimal surfaces that can be continuously transformed into one another by varying the scalar  $\phi$ . Some well-known families transform a catenoid into a helicoid or a Schwarz P surface into a Schwarz D (Figure 4). For special pairs of surfaces  $S_1, S_2 \in F^\phi(u, v)$ , the family can be parametrized as follows:

$$\begin{aligned} F^\phi(u, v) &= \cos \phi \ S_1(u, v) \ - \sin \phi \ S_2(u, v), & (1) \\ &= \Re(e^{-i\phi} [S_1(u \ -iv) \ iS_2(u \ -iv)]), & (2) \end{aligned}$$

where  $\Re(z)$  returns the real part of a complex number  $z$ . As suggested by the complex formulation, the special surfaces  $S_1$  (at  $\phi = 0$ ) and  $S_2$  (at  $\phi = \frac{\pi}{2}$ ) are said to be *conjugate* to one another. More generally, any two members  $F^\theta$  and  $F^{\theta + \frac{\pi}{2}}$  are conjugate to one another and every minimal surface is part of such a pair. These surface pairs have several pertinent properties:

**PROPOSITION 4.1 (CONJUGATE SURFACE PROPERTIES).** *Let  $S$  be a minimal surface and  $C$  be its conjugate. Then the following are true:*

- For any arbitrary point  $(u_0, v_0)$  in the domain, the surface normals  $N_S(u_0, v_0)$  and  $N_C(u_0, v_0)$  are identical.*
- $S$  and  $C$  are isometric, so the angles at corresponding points  $(u_0, v_0)$  along the boundary are identical on both surfaces.*
- If some portion of  $S$  is bounded by a straight line, then the corresponding portion of  $C$  is bounded by a planar symmetry line and vice versa.*

The symmetry lines noted in Proposition 4.1(c) are critical for minimal surfaces (particularly TPMS), as they allow the surface to be extended while preserving smoothness:

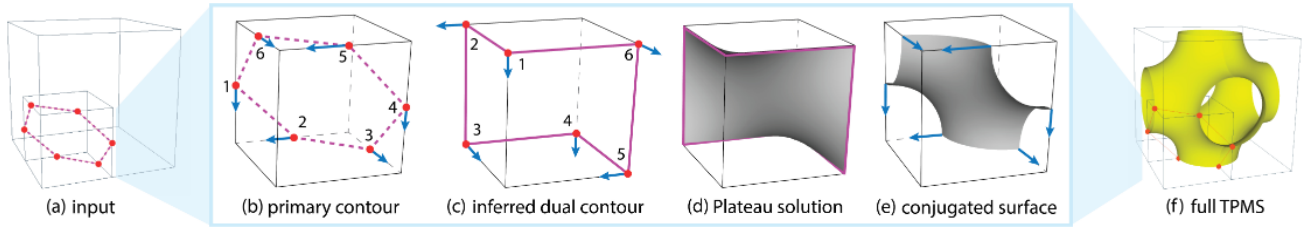


Fig. 3. **Conjugate surface construction.** Based on the (a) input, we (b) infer the angles and surface normals (blue) at each vertex and (c) construct a dual contour that respects these properties. Then, we (d) solve a Plateau problem over the dual contour and (e) conjugate the dual surface to arrive at the primary surface matching our original specifications. Finally, we (f) align the primary patch to the input and extend the patch to construct the full TPMS.

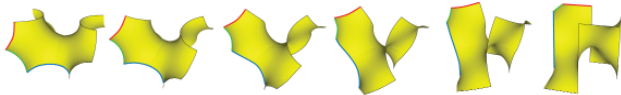


Fig. 4. **Associate family and symmetry lines.** The associate family transforms a Schwarz P surface patch (left) into its conjugate Schwarz D patch (right). The Schwarz P's planar symmetry lines (e.g., red/green/blue curves) morph into straight lines bounding the Schwarz D.

PROPOSITION 4.2 (MINIMAL SURFACE EXTENSION). *Let  $S$  be a minimal surface partially bounded by a symmetry line  $\ell$ . Then,*

- If  $\ell$  is a planar symmetry line in plane  $p$ , then  $\ell$  must be fully contained in  $p$ ,  $S$  must meet  $p$  orthogonally, and  $S$  can be extended by reflecting across  $p$ .
- If  $\ell$  is a straight line, then  $S$  can be extended by rotating  $180^\circ$  about  $\ell$ .

As hinted in Section 3, symmetry lines can also be used to decompose a complex surface into a fundamental patch  $P$  bounded by symmetries. Then, by Proposition 4.1(c),  $P$  has a conjugate  $\bar{P}$  bounded by the opposite symmetries. Moreover, it is possible to construct one surface from the other via a “conjugation” process [Karcher 1989; Pinkall and Polthier 1993]. Thus, the problems of solving for  $P$  and  $\bar{P}$  are equivalent. Since the term “conjugate” is overloaded—as the relationship between  $P$  and  $\bar{P}$  as well as the procedure that maps between them—we call  $P$  the *primary* surface and  $\bar{P}$  the *dual*.

The above equivalence is the foundation of the CSCM, which is most powerful when the dual  $\bar{P}$  can be solved for more easily than our primary target,  $P$ . For example, when  $P$  is a free boundary problem, its boundary  $\partial P$  contains only planar (sliding) symmetries; thus,  $\partial \bar{P}$  is a simple polygonal contour that admits a Plateau solution [Gray and Micallef 2007]. The CSCM easily recovers  $P$  by solving and conjugating  $\bar{P}$ . The CSCM has fewer advantages when  $\partial P$  has mixed symmetries, as  $\partial \bar{P}$  is similarly complex. As such, we only apply the CSCM (via our *conjugate solver*) when  $P$  is a free boundary problem; other cases are deferred to our *mixed minimal solver* (Section 5.1.1).

The CSCM (Figure 3) contains several steps that are well established and tangential to our extension; for a full description, we refer readers to our Supplementary Materials. Here, we need only discuss the construction process for  $\partial \bar{P}$  (from Figures 3(b) to (c)), as this is the major limitation of existing CSCM algorithms.

By Proposition 4.1(a), the surface normal at each dual vertex  $\bar{v} \in \partial \bar{P}$  must match that of the corresponding primary vertex

---

**ALGORITHM 1: MAINCONJUGATESURFACECONSTRUCTION**


---

**Input:** bounding planes  $B$ , closed edge loop  $L$   
**Output:** minimal surface  $P$  solving the free boundary problem

```

 $H_B \leftarrow$  convex polyhedron bounded by half-spaces of  $B$ 
Find co-planar verts and vert angles/normals for dual contour
 $vA \leftarrow$  EmptyMap(),  $vN \leftarrow$  EmptyMap(),  $V \leftarrow$  EmptyMap()
for vertex  $v$  in counterclockwise traversal of  $L$  do
   $p_1, p_2 \leftarrow$  planes of  $B$  containing  $v$ , with outward-facing normals
   $vA[v] \leftarrow$  dihedral angle between  $p_1, p_2$ 
   $vN[v] \leftarrow$  normalize((normal of  $p_1$ )  $\times$  (normal of  $p_2$ ))
   $V[p_1].append(v)$ ,  $V[p_2].append(v)$ 
end
Find the rigid transformation for bounding plane alignment
 $P^{test}, E^{test} \leftarrow$  CONTOURENERGY( $l^{init}, vA, vN, V, H_B, I$ )
 $B^{test} \leftarrow$  convex bounding volume of  $P^{test}$ 
 $R \leftarrow$  rotation that best aligns  $B$  and  $B^{test}$ 
Compute and post-process the best solution
 $P, E \leftarrow$  arg min $l$  CONTOURENERGY( $l, vA, vN, V, H_B, R$ )
return FIXBOUNDARY( $P$ )

```

---



---

**ALGORITHM 2: CONTOURENERGY**


---

**Input:** edge lengths  $l$ , vertex angles  $vA$ , vertex normals  $vN$ , coplanar vertex sets  $V$ , bounding polyhedron  $H_B$ , rotation  $R$   
**Output:** primary mesh candidate  $P_l$ , energy value  $E$

```

 $c \leftarrow$  SOLVECONTOUR( $l, vA, vN$ )
 $\bar{P}_l \leftarrow$  SOLVEPLATEAUPROBLEM( $c$ )
 $P_l \leftarrow R$  CONJUGATESURFACE( $\bar{P}_l$ )
 $E \leftarrow E_{contour}(P_l, V, H_B)$ 
return  $P_l, E$ 

```

---

$v \in \partial P$ . Moreover, since  $P$  must be orthogonal to both bounding planes incident on  $v$  (Proposition 4.2(a)), the normal at  $v$  must align with the intersection line on which  $v$  sits. By Proposition 4.1(b), the angle<sup>1</sup> between adjacent edges incident on  $\bar{v}$  equals the dihedral angle spanned by the planes incident on  $v$ . These facts prescribe everything about  $\partial \bar{P}$  except the length of each edge. Boundaries with four vertices are determined up to scaling due to the loop closure condition, but for  $N$  vertices, there are generally  $(N - 4)$  edge lengths that can be set arbitrarily.

<sup>1</sup>The angle is measured within the plane spanned by the pair of lines.

Previous works require users to manually compute these edge lengths using an iterative process that requires an understanding of the relationship between  $\partial\bar{P}$  and  $P$  [Karcher and Polthier 1996]. This is especially taxing when aligning  $P$  with a given FBV, as the edges' relative lengths determine the alignment—thus, they cannot be set arbitrarily. Improper edge lengths may also preclude a valid solution entirely: for the *period problem*, in which multiple portions of  $\partial P$  reside on the same bounding plane (see Table 2, “Wei’s genus 4”), most length assignments fail [Karcher and Polthier 1996]. Existing methods rely on a manual intermediate value argument that cannot readily expose the permissible values. Our approach identifies a valid solution while obscuring the CSCM details and reducing the required user expertise (as evidenced by the user study in Section 7.3).

## 4.2 Our Edge Length Optimization

To address the limitation of existing CSCMs, we devise a fully automated solver (Algorithm 1) for free boundary problems given by a set of boundary planes and an ordered edge loop  $L$  over these planes. We propose an optimization loop that finds edge lengths  $l$  for the dual contour  $\partial\bar{P}$ , such that the final surface  $P$  conforms to .

**4.2.1 Pre-processing.** To prepare for our optimization, we first infer the normals  $vN$  and angles  $vA$  at each vertex of  $\partial\bar{P}$ , as described in Section 4.1. In addition, we compute the convex polyhedron  $H_B$  that is enclosed by the half spaces given by . We also infer  $V := \{V_0, \dots, V_{K-1}\}$ , where  $K$  is the number of planes in and  $V_i$  is the set of vertices from  $L$  that belong to plane  $i$ . These vertex sets can be used w.r.t. any primary or dual surface, since we know the pointwise correspondences for any point in  $L$  by construction. Finally, we compute a rotation  $R$  that will align the candidate primary surfaces to for comparison. To do this, we first construct a sample primary mesh  $P^{\text{test}}$ . We fit a plane to each co-planar vertex set  $V_i$  along the boundary of  $P^{\text{test}}$ , which results in a set of  $K$  boundary planes  $\text{test}$ , as shown in Algorithm 1. We use the approach of Müller et al. [2005] to find the  $R$  that best aligns the normals of and  $\text{test}$ . We need only compute  $R$  once, because all future candidate primary surfaces (and thus, their bounding planes) will share the same orientation; only the scale will differ.

**4.2.2 Objective Function.** To judge each candidate contour with lengths  $l = \{l_i\}$ , we devise a two-term objective function,

$$E_{\text{contour}}(P_l, V, H_B) = E_{\text{BV}}(P_l, V, H_B) + E_{\text{coplanar}}(P_l, V),$$

where  $P_l$  is the primary mesh output by the CSCM on a contour with lengths  $l$ ,  $E_{\text{BV}}$  penalizes differences between the boundary planes of  $P_l$  and our target planes , and  $E_{\text{coplanar}}$  penalizes the mis-alignment of boundary vertices in  $P_l$  that are supposed to be co-planar.

First, we find the bounding planes  $\text{test}$  for  $P_l$  by fitting a plane to each co-planar vertex set  $V_i$ . Since  $R$  has already been applied to  $P_l$ , we set the normal  $n_i$  to be the normal of plane  $i$  in . The plane’s origin  $c_i$  is given by  $c_i = \frac{1}{|V_i|} \sum_{j \in V_i} p_j$ , where  $p_j$  is the position of the  $j$ th vertex. Then,  $E_{\text{coplanar}}$  can be defined as follows:

$$E_{\text{coplanar}} = \sum_{i=0}^{K-1} \sum_{j \in V_i} n_i^T (p_j - c_i)^2 \quad (3)$$

Finally, to evaluate  $E_{\text{BV}}$ , we compute the convex polyhedron that is enclosed by the half-spaces of . We use the **iterative closest point (ICP)** algorithm [Amberg et al. 2007] to align this polyhedron with our reference,  $H_B$ . We only permit translation, as the rotation  $R$  has already been accounted for. After alignment,  $E_{\text{BV}}$  is given by the mean squared distances between the two polyhedra. As we only consider translation, we did not encounter unconverged ICP results during our optimization.

The method above is designed for arbitrary convex bounding polyhedra. However, when the polyhedron  $H_B$  is an **Axis-Aligned Bounding Box (AABB)** (i.e., six axis-aligned planes forming a cuboid space) we can leverage a stronger objective. Namely, for any pair of points  $p_j, p_k$  on opposite, parallel planes of the AABB with normal  $n_a$ , we know the expected distance  $d_a$  between  $p_j$  and  $p_k$  along  $n_a$ . Thus, we can encode both  $E_{\text{BV}}$  and  $E_{\text{coplanar}}$  if we ensure that all such pairs  $p_j, p_k$  are distance  $d_a$  apart in the normal direction, i.e.,

$$E_{\text{contour}}^{\text{AABB}} = \sum_{a=1}^3 \sum_{j \in V_{a,1}} \sum_{k \in V_{a,2}} \left[ n_a^T (p_j - p_k) - d_a \right]^2, \quad (4)$$

where  $a$  is the axis index for the  $\{x, y, z\}$  directions,  $n_a$  is the unit axis- $a$  direction,  $V_{a,1}$  and  $V_{a,2}$  are the sets of boundary vertices on each opposing plane with normal  $n_a$ , and  $d_a$  is the target distance between the two planes in direction  $n_a$ .

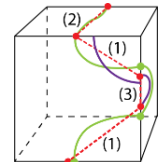
**4.2.3 Regularizing Edge Lengths.** On each evaluation of the energy function (Algorithm 2), we have a set of suggested lengths  $l^{\text{input}}$  over the contour that may or may not form a closed loop. We verify the closedness and/or adjust the lengths as necessary by solving the following mini-minimization problem during SOLVECONTOUR:

$$\arg \min_{l^{\text{base}}} \sum_i \left( l_i^{\text{base}} - l_i^{\text{input}} \right)^2, \quad (5)$$

$$\text{s.t. } \sum_i l_i^{\text{base}} e_i = 0, \quad (6)$$

where  $l_i^{\text{base}}$  is the edge length that will be used as the base guess for edge  $i$  on this iteration and  $e_i$  is the contour edge direction given by  $vA$  and  $vN$ . The linear constraint guarantees that  $l_i^{\text{base}}$  forms a closed loop. Since this is a quadratic energy with linear constraints,  $l_i^{\text{base}}$  is obtained by directly solving a small linear system.

**4.2.4 Initialization.** To seed our optimization and create  $P^{\text{test}}$ , we find an initial guess  $l^{\text{init}}$  for the contour lengths. By construction, all input vertices lie on the edges of the convex polyhedron , and contour edges lie on the faces of . As shown in the inset, we categorize each contour edge  $e_i$  based on its endpoints, which can be on (1) two neighboring polyhedron edges  $h_1, h_2$ ; (2) two non-neighboring polyhedron edges  $h_1, h_2$  on the same face; or (3) the same polyhedron edge  $h_1$ . The input edges for each case are shown in red, and the corresponding arcs along the primary surface  $P$  are shown in green. Although the green curves are not known *a priori*, each case behaves in a relatively consistent manner, as evidenced by, e.g., Proposition 4.2, which implies that final contour edges must intersect polyhedron



edges perpendicularly. The expected length of each green curve predicts the corresponding dual length, because lengths are preserved due to isomorphism (Proposition 4.1(b)). Thus, based on the expected shape of each final contour, we developed a simple heuristic (shown in purple) for the initial length  $l_i^{\text{init}}$  of  $e_i$  in each case. In case (1),  $l_i^{\text{init}}$  is the length of an arc between  $h_1$  and  $h_2$ , assuming the arc is part of the circle whose center is the intersection point of  $h_1$  and  $h_2$  and whose radius is the average distance from each endpoint of  $e_i$  to the circle center. In case (2),  $l_i^{\text{init}}$  is the distance between the endpoints of  $e_i$ . In case (3), we let the distance between the endpoints be the diameter of a circle whose center is the mid point of  $h_1$ ; then,  $l_i^{\text{init}}$  is half of the circle’s perimeter.

**4.2.5 Optimization.** Beginning from  $l^{\text{init}}$ , we optimize the energy given in Algorithm 2 using the gradient-free Nelder–Mead method [Nelder and Mead 1965]. Because it is common to have large differences in absolute edge lengths, the optimization converges faster if we optimize the *ratio* to each edge’s base length rather than the absolute length itself. By defining the optimization over this ratio space, we can also easily define generic lower and upper bounds for each edge length. We use 0.1 and 10, respectively.

**4.2.6 Post-processing.** After constructing a suitable primary mesh  $P$ , we apply our previously computed rotation to ensure that the returned surface aligns with  $\mathcal{P}$ . Although the overall agreement is quite high, numerical issues frequently cause the vertices of our conjugated surface boundary to be slightly offset from their intended planes. To resolve this issue, we “snap” all boundary vertices to their intended boundary planes. We do this via our direct surface solver (described in Section 5.1.2). For this, we treat  $P$  as our rest shape and compute the target position for each boundary vertex by projecting its current position to the target plane. Our small discrepancies only induce a slight deformation of  $P$ , so the accuracy of our result is unaffected, as shown in Section 7.1.

## 5 COMPLETE SKELETAL DESIGN SPACE

The conjugate surface solver described above lets us solve free-boundary problems in a stable manner. Now, we round out our skeletal design space by describing the remaining skeletal solvers and our spatially varying thickness specification.

### 5.1 Surfaces

To complete our surface design space, we describe the *mixed minimal* and *direct* surface solvers. Figure 5 highlights the variation within our surface space by showing the result of each solver over *identical* annotated boundary loops and planes.

**5.1.1 Mixed Minimal.** Our mixed minimal surface type is a sliding solver for boundaries that are at least partially fixed, such as the FS of the TPMS from Hao Chen’s  $o\Delta/t\Delta$  family (Figure 2, top). As discussed in Section 3, the shape and location of all non-smooth boundary segments are preserved, such that the boundary of the computed surface coincides precisely with all fixed input. Smooth (or “sliding”) segments have much more freedom, as they are treated like free boundaries that are permitted to deform within the given boundary plane. To achieve a minimal surface subject to these constraints, we perform a mean-curvature flow algorithm

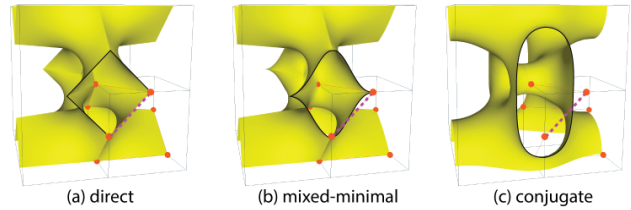


Fig. 5. **Surface types over identical input.** A given FBV and annotated boundary (where dashed edges are smooth and others are not) can generate many outputs based on the selected surface type: method (a) preserves the dashed line, method (b) allows it to deform into a curve within its plane, and method (c) allows sliding not only for this edge but for *all* edges, as our conjugate solver assumes fully sliding boundary loops regardless of annotations.

that imposes Dirichlet boundary conditions along any fixed boundaries and sliding constraints along sliding boundaries.

**5.1.2 Direct.** Direct surfaces yield more general (non-minimal) surfaces over fully fixed boundaries. Segments contained in a non-smooth edge chain remain straight, while each smooth edge chain is first interpolated into a curve according to the description in Section 5.2. Then, we apply a standard thin-shell model defined over a surface that is subject to fixed boundary constraints:

$$\arg \min_x E_{\text{inplane}}(x) + \alpha E_{\text{bend}}(x), \quad (7)$$

where  $E_{\text{inplane}}$  penalizes in-plane stretching,  $E_{\text{bend}}$  penalizes bending, and  $\alpha$  is the energy weight. By default, we set  $\alpha = 0.1$  to prioritize  $E_{\text{inplane}}$ . Although any such model would be suitable, we use the definition of Bouaziz et al. [2014] and the implementation of ShapeOp [Deuss et al. 2015] for fast performance.















To measure this deformation energy, we must define a rest shape for the surface patch. We could use a hole-filling algorithm to generate an initial surface that spans the boundary loop, but complex boundaries often yield poorly behaved or low-quality triangle meshes. Instead, we use a default rest shape for all boundary loops, namely a circular patch that is much smaller than the input boundary. This forces a large stretching deformation to meet the target boundary; coupled with a de-emphasized bending energy (e.g.,  $\alpha = 0.0$ ), this encourages a smooth surface that approximates a minimal surface. By adjusting  $\alpha$ , it is possible to deviate from this behavior to produce a wide range of surface patches.

### 5.2 Lines

Line skeletons are represented by a sequence of 3D points, which can form any open, non-branching path or simple closed loop (if the endpoints are identical). Along non-smooth edge chains, neighboring vertices are connected by straight lines. Each smooth edge chain is interpolated to form a natural cubic spline that passes through the input points with C2 continuity everywhere. We use natural cubic splines, because they are simple, intuitive, and familiar, thanks to their widespread use in other modeling tools. To address concerns that are specific to cellular metamaterial design, we adjust the standard spline solver to permit (1) C2 continuity along closed loops and (2) curves that tile smoothly across a periodic boundary. We address ill-defined curves (with fewer than



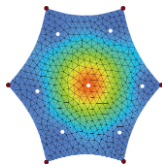
Table 1. **Node Types and Their Inputs.** Brackets Around an Input Indicate that there can be Multiple Nodes of the Same Type Fed into the Given Node

	Node Type	Color	Code	Input	Properties	Requirements
Topology	Vertex		v	-	position	-
	Edge Chain		e	{Vertex}	vertex order; smoothness	At least two vertices; branch-free
Skeleton	Line		l	{Edge Chain}	thickness profile	Edge Chains form a single branch-free path
	Surface		s	{Edge Chain}	surface type; thickness profile; energy weights (direct surface only)	Edge Chains must form a simple closed loop
	Dual Surface		dual	Surface	-	The input surface must be of type “conjugate”
	Mirror		mirror	Skeleton	mirror direction; origin; do copy	-
	Transform		t	Skeleton	origin; scale; rotation angle, axis; translation; do copy	-
	Associate Family		af	Skeleton × 2	angle $\phi$	Inputs 1 & 2 must belong to the same associate family
	Group		g	{Skeleton}	-	-
Solid	Object (Solidify)		obj	Skeleton	grid resolution, extrusion method	-
	CSG Boolean		bool	{Object}	boolean type	Inputs must have the same grid resolution
	Voxel		vox	Object	$E, v, \rho$	At most 1 voxel node in a graph
Material Properties	Material Matrix		mat	Voxel	-	-
	Phononic Bandgap		pbg	Voxel	#curves; #samples per curve; tol;	-

four vertices) by computing a quadratic spline (three vertices) or a straight line (two vertices).

### 5.3 Spatially Varying Thickness

To control how each skeletal element should be instantiated as a volumetric object, we include a spatially varying thickness function over the element’s domain. This function is given by sparse input: the user specifies the thickness at a few sample points, and then these values are interpolated over the full domain. The values are computed and stored w.r.t. a simple parametrization of the domain rather than actual vertex positions along the skeletal element. For lines, we use a 1D parametrization  $\gamma(t)$  for  $t \in [0, 1]$ . The full thickness profile is linearly interpolated from the provided sample points. For surfaces, which have disc-topology, we define thickness w.r.t. a UV-domain computed via the as-rigid-as-possible parameterization in libigl [Jacobson et al. 2018; Liu et al. 2008]. Then, as shown in the inset, we interpolate the thickness by treating each sample point (white dot) as a handle and computing bounded biharmonic weights over the domain [Jacobson et al. 2011].



## 6 PROCEDURAL GRAPH FOR METAMATERIALS

Building atop our skeletal design space, we introduce a procedural graph that facilitates the full metamaterial design process, from initial shape specification to material property prediction. We summarize our node types in Table 1, including their inputs, properties, and requirements. In this section, we expand on the design logic and implementation for each node.

### 6.1 Vertex and Edge Chain Nodes

A *vertex* node is the simplest node in our graph: it places a vertex at the 3D location given by its “position” property. Vertex nodes do not accept any input node connections. The next simplest node is the *edge chain*, which accepts a set of vertex nodes as input, and then instantiates a path over these vertices in the traversal order

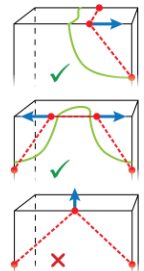
that is stored as a property of the edge chain node. As discussed, each edge chain also has a “smoothness” flag that is used by the subsequent line or surface node(s) to facilitate the variations described in Section 5.

### 6.2 Line Nodes

Our *line* node accepts a set of edge chains that are *continuously traversable*, i.e., the end of one edge chain is the start of another, such that they form a simple, non-branching path (open or closed). As discussed in Section 5, the shape of a line is determined by the smoothness of each constituent edge chain. A line node can accept any combination of smooth and fixed edge chains, as long as they are continuously traversable. For more complicated paths, multiple line nodes must be defined. The spatially varying thickness function for each line node is given as in Section 5.3.

### 6.3 Surface Nodes

Our *surface* node accepts a set of edge chains that form a simple closed edge loop, over which we instantiate a surface patch. Users first select a surface type from among *mixed minimal*, *conjugate*, and *direct*. This selection determines the algorithms used to interpret the boundary and solve for the final surface, as detailed in Section 5. It also dictates the requirements and properties for the surface node. For direct surfaces, the user need only provide the energy weight  $\alpha$  (see Section 5.1.2) and a non-degenerate, simple, closed boundary loop over vertices located anywhere in the FBV. For conjugate and mixed-minimal surfaces, each sliding segment must lie on an FBV face. Mixed-minimal surfaces permit vertices anywhere on an FBV face (see Table 2, “Deformed H”). Conjugate boundaries are more restricted to satisfy the properties of Section 4.1. Specifically, all vertices must lie on FBV edges and form property-respecting configurations, as shown in the inset. The bottom inset is invalid, because the surface normal (blue) cannot align with the FBV edge as required.



We also have *dual surface* and *associate family* nodes to explore the families given by Equation (1). This lets

us capture and explore intermediate surfaces like the gyroid (see Section 7.1).

#### 6.4 Mirror, Transform, and Group Nodes

To build full translational units from a structure’s FS, we provide standard geometric transformations such as the *mirror* node. We also support translation, rotation, and scaling via the *transform* node, as well as the ability to combine a set of skeletal elements into one unit via the *group* node. Each of these nodes takes one or more “skeletons” as input, which may be a line, a surface, or the output of one of the transformation nodes above. The user can select whether the transformation is applied to a copy of the input skeleton (“doCopy”=true) or to the input skeleton itself (“doCopy”=false).

#### 6.5 Object, CSG Boolean, and Voxel Nodes

To generate volumetric objects, our *object* node thickens each skeletal element based on its interpolated thickness profile. First, we instantiate a regular grid over a unit cube, which will store an indicator function that tracks whether each grid point is inside or outside of the object. The grid resolution is a property of the object node, along with the desired extrusion method. As suggested in Section 3, we support two extrusion approaches: *spherical* and *normal*. For spherical extrusion, we densely sample the skeleton and—for each sample point—we splat a sphere onto the indicator function grid, using a sphere diameter equal to the sample point’s prescribed thickness. For normal extrusion, the offsets are applied along the normal direction at each sample point. For surfaces, we offset each vertex by one-half of the desired thickness along each orientation ( $\pm$ ) of its normal. For lines, we sweep a scaled circular cross section along the skeleton to create a tube. These methods are illustrated in Supplementary Materials. After extrusion, we compute the indicator function for each resulting geometry and union them together. To preserve tileability, we require identical function values for each pair of corresponding grid points on opposite boundary faces. Finally, we perform marching cubes on the indicator function [Lorenson and Cline 1987] to extract the object mesh.

We also provide a *CSG Boolean* node that supports union, intersection, and difference operations on a set of volumetric objects, as shown in Supplementary Materials. For efficiency, we compute Boolean operations on the underlying indicator functions for each object; then we enforce tileability and perform marching cubes as before.

Finally, we use the indicator function to implement our *voxel* node, which creates voxelized meshes suitable for property prediction.

#### 6.6 Metamaterial Property Nodes

As a proof of concept for fully integrated design, we implement two property prediction nodes: material matrix and phononic bandgap.

The *material matrix* node computes the stiffness tensor of the volumetric structure given by our graph. We implement the homogenization approach of Panetta et al. [2015] to compute the equivalent stiffness tensor matrix  $C$ , assuming periodic boundary conditions and a linear elastic material over our voxel mesh. All

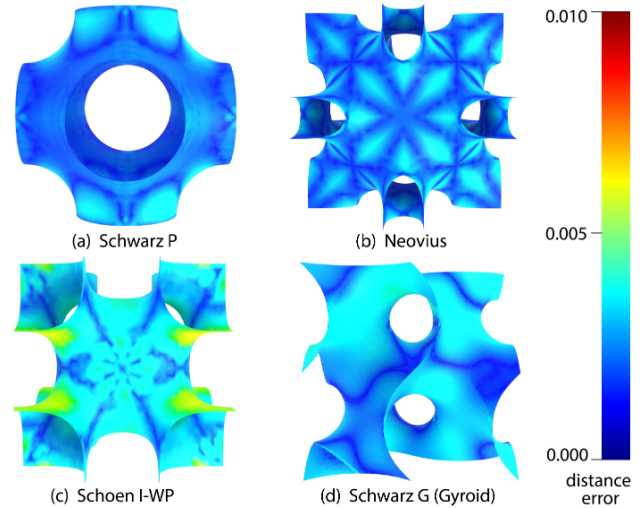


Fig. 6. **Conjugate surface accuracy.** We compare our conjugate TPMS to the ground-truth Enneper–Weierstraß functions integrated over a unit cell. The error is the distance between each vertex of our surface and its closest point on the analytical surface. The average errors are 0.001, 0.001, 0.003, and 0.002, with maximal errors of 0.005, 0.004, 0.007, and 0.007, respectively.

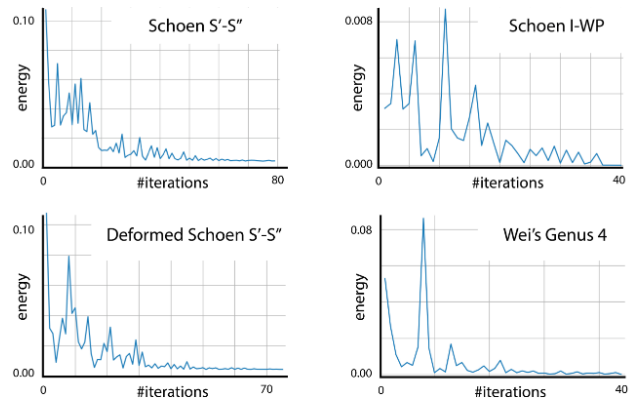


Fig. 7. **Convergence of edge length optimization.** We demonstrate convergence for the four TPMS with the longest optimization times, which use different FBVs: Schoen I-WP uses a prism, while the others use an AABB.

linear systems were solved using Intel MKL Pardiso [Schenk and Gärtner 2004]. We also use *C* to compute and display the “material sphere” that illustrates the (an)isotropy of each structure (Figure 11(c)).

The *phononic bandgap* node predicts a structure’s ability to prevent the propagation of waves in certain frequency ranges, toward applications such as frequency filters, beam splitters, waveguides, and sound/vibration protection devices. The blocked ranges are known as *bandgaps*, and they are often the result of structural frequency-filtering mechanisms such as Bragg scattering and local resonances. We predict the bandgap using the approach of Åberg and Gudmundson [1997], which generates a set of *dispersion curves* showing the structure’s eigenmodes over varying wave vectors

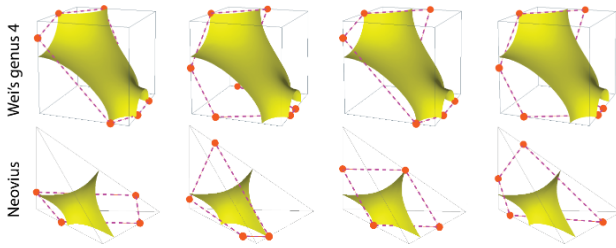


Fig. 8. **Robustness of edge length optimization.** Although different vertex positions (red) yield different initial edge lengths, our optimizer consistently converges to the same TPMS patch, even across different FBVs.

(Figure 12, light blue). We process these curves in search of horizontal bands through which no curves pass (Figure 12, grey rectangles). Each such area is a bandgap, as it indicates that the given frequency range has no viable transmission path through the structure.

## 7 RESULTS

We implement our procedural graph in C++, with an OpenGL-based GUI for interactive design. We performed all experiments on Ubuntu 20.04 using an AMD Ryzen 5950X CPU (16 cores) with 64 GB RAM.

### 7.1 Conjugate Surface Construction

To evaluate our CSCM, we reproduce a variety of TPMS whose FS are given by free boundary problems, including the popular Schwarz P, Neovius, and Schoen I-WP structures. As shown in Figure 6, our surfaces exhibit strong agreement with the ground-truth TPMS given by the Enneper–Weierstraß functions integrated over a unit cell. We obtained each ground-truth surface via a publicly available Mathematica notebook [Weber 2018a, b, c, d], from which we extracted and uniformly rescaled exactly one translational cell of unit size.

Figure 6 also shows our accurate reproduction of the gyroid, which is one of very few TPMS that do not contain any straight or planar symmetry lines. This seems to render the gyroid incompatible with our method. However, the gyroid is a known member of the Schwarz P/D associate family: assuming that  $D$  occurs at  $\phi = 0$ , the gyroid occurs at  $\phi \approx 38$  [Karcher 1989]. Thus, we can construct a patch of the gyroid by creating the P and D structures via our CSCM, then interpolating via the associate family node.

We ensure that our optimization converges reliably for all of our examples, including those in Table 2 and hundreds of randomly generated structures (Figure 9). Even the most intensive structures in Table 2 converge within 80 iterations, as shown in Figure 7. Moreover, Figure 8 demonstrates that our algorithm’s output is stable w.r.t. the input vertex positions and initial edge lengths  $l^{init}$ . The only difference is that more accurate initial guesses permit faster convergence, as evidenced by Tables 2 and 4.

### 7.2 Representing Established Cellular Structures

We use our procedural graph to recreate structures from literature that span all of our target classes. As a simple test, we first re-implement the exhaustive truss-based exploration strategy of Panetta et al. [2015] and arrive at the same collection of 1,205 valid topologies in our representation. We also recreate a num-

ber of other structures found in literature. Tables 2 and 3 show the final geometry for a selection of our structures, which were created from scratch in our GUI, following the design process of Section 3.3. Table 4 gives a detailed list of nodes used for each structure, along with the computation time required to evaluate our full procedural graph. The minimal, median, average, and maximal number of nodes used is 12, 16, 19.1, and 34, respectively. This demonstrates that our graph is both lightweight and versatile. Furthermore, the computation time shown in column  $t$  demonstrates that our system is able to quickly produce the final structure from the sparse graph input, with median, average, and max time costs being 0.72, 2.31, and 21.91 s, respectively. Thus, our method can be used to create and modify metamaterials at interactive rates.

### 7.3 User Study

To evaluate the expressiveness, compactness, and ease-of-use of our approach, we invited 10 participants to model several metamaterials using our procedural graph. We sought participants with varying degrees of expertise in metamaterials, 3D modeling, and minimal surface theory. Many participants self-identified as a novice in one, two, or all three domains, but nobody identified as an expert in all domains.

*Procedure.* The study generally took 3 to 4 hours per user and contained five stages: (1) a pre-survey; (2) a brief presentation introducing the project goal and our representation; (3) a guided modeling session to practice conceptualizing/building structures using our representation; (4) an independent modeling session, in which the participant constructed six target structures; and (5) a post-survey. Supplementary Materials provides a detailed account of each stage, as well as the participants’ responses/results. As noted to our users, the study is primarily concerned with the intuitiveness and flexibility of the procedural graph representation, not the interactive tool.

Our primary experiment occurred in stage (4), as participants independently modeled six structures given by target 3D meshes. The target structures spanned all of our major classes, so we could examine our method’s overall expressivity and ease-of-use. To reduce undue burden on the participants, we asked them to focus on reproducing each target’s main structure rather than precisely inferring continuous parameter values for, e.g., thickness or vertex positions.

*Main Results.* All 10 users successfully reproduced all six structures, independent of prior experience. A subset of the structures are shown in Table 5, along with statistics about the time and number of nodes required to represent them. Of the 60 total modeling tasks, 56 (93%) were completed in  $\leq 30$  minutes and 45 (75%) were completed in  $\leq 20$  minutes. In the post-survey, users also indicated high levels of confidence that they could implement unseen structures of the various classes in the future. Moreover, the overwhelming majority of users (90%) agreed that the process of modeling a diverse set of metamaterials would be easier/more intuitive in terms of our proposed procedural graph than it would be in terms any single other representation.

Curved shells presented the largest challenge, as users uniformly reported the lowest degree of confidence in—and highest

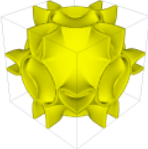
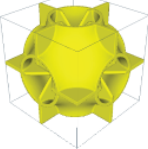
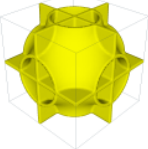
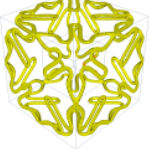
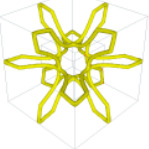
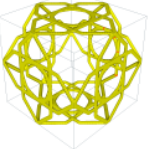
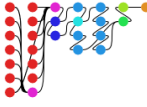
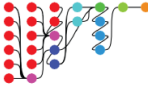
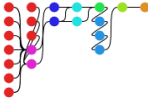


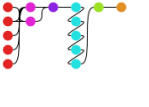

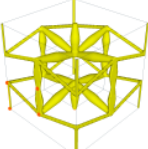
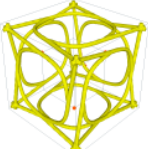
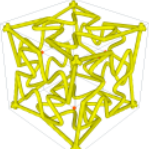
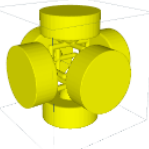
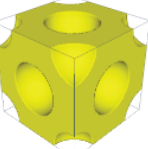
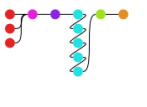
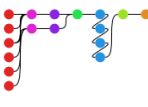
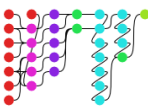
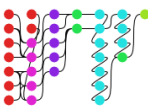
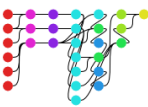

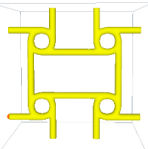

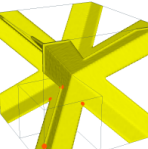
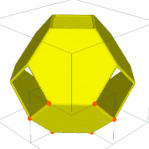
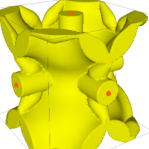
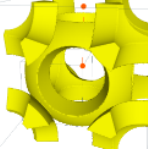
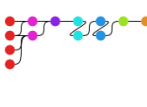

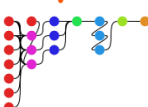


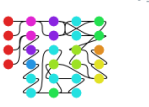
Table 2. **TPMS Boundary Loops.** We Represent Several Well-Known TPMS using a Small Set of Vertices and a Single Boundary Loop. Column “S” Indicates the Name and Type of Each Structure: A Blue Label Indicates a Direct Surface, Green Indicates a Mixed-Minimal Surface, and White Indicates a Conjugate Surface. For Each Structure, We show the Boundary Loop and Bounding Planes (“vtx/edge”), the Generated Surface Patch (“patch”), and the Final Volumetric Structure (“obj”)

S	vtx/edge	patch	obj	S	vtx/edge	patch	obj
Schwarz P				Wei's genus4			
Neovius				Schoen S'-S''			
Schoen I-WP				Deformed Schoen S'-S''			
Steffmann				Schwarz D			
Schwarz CLP				Schoen R2			
Hao Chen's $\alpha\Delta/r\Delta$				Deformed H			
Gyroid							
		primary	conjugate		$\phi \approx 52^\circ$		object

degree of difficulty with—these structures. Nevertheless, all users succeeded in the curved shell tasks, and 90% of them expressed that it would have been more difficult or impossible to represent these structures using any other approach. This is particularly

true of Hao Chen’s and Wei’s TPMS, as neither structure currently has a trigonometric approximation; this typically renders them inaccessible to designers, but our representation provided novice access to both. Moreover, the presence of a traditionally

Table 3. **Varied Cellular Metamaterials.** We Represent a Variety of Metamaterial Architectures, Including Some from Literature and Others of Our Own Creation

S	Schwarz P+ Neovius	Schwarz P+Cubic Open	Schwarz P+Cubic Closed	assembled	assembled2	assembled3
object						
graph						
S	octet	auxetic3d	twist-tilable	twist-tilable-bezier	bandgap	FCC lattice
object						
graph						
S	anti-chiral	star-shaped structure	non-manifold surface	polyhedral cell	our composition	6-hole BCC
object						
graph						

challenging period problem (see Section 4.1) in Wei’s genus 4 was a non-issue for our users; in fact, this structure required the lowest average modeling time of all. Overall, this user study confirms the expressiveness, compactness, and ease-of-use of our proposed representation, as even novice modelers can rapidly and faithfully realize a wide range of design intents.

## 8 APPLICATIONS

Armed with our procedural graph representation, we envision several exciting possibilities for future metamaterial exploration.

### 8.1 Automated Structure Generation

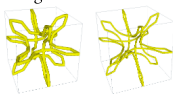
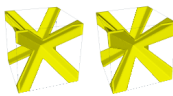
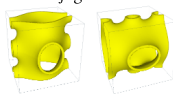
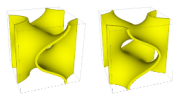
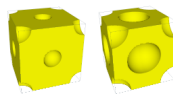
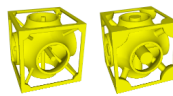
Due to its compact form, our representation is conducive to automatic exploration. As a proof of concept, we devise simple exploration strategies for structures in two classes: straight trusses

and shellulars. For a detailed explanation of these strategies, see the Supplementary Materials. Our truss exploration can generate a virtually unlimited number of structures due to the relatively unconstrained space for trusses. Valid shellular graphs are considerably more restricted, but our methods still generate hundreds of orthotropic and general asymmetric shellular structures. In particular, we obtained 1,000 direct structures in approximately 1 hour, 498 conjugate structures in 4 hours, and 500 general asymmetric structures in 3 hours. This exploration returned an enormous collection of unstudied direct shellulars and a mix of established and novel TPMS shellulars, as shown in Figure 9. The presence of established structures confirms that our representation encompasses critical regions of the cellular metamaterial design space, while the presence of novel structures indicates its potential for innovation. Our structures are also tilable by construction and (in all observed cases) physically realizable via inkjet-deposition additive

Table 4. **Procedural Graph Statistics.** For Each Structure, We List the Total Number of Nodes (#all) As Well As a Breakdown by Node Type: Vertex (#vtx); Edge (#e); Line (#l); Surface/Dual Surface (#s), Transformation/Mirror/Group (#T), Object (#o), and Voxel (#vox). We Also show Structure Type (“type”): “c” Refers to Conjugate Surface, “d” Denotes Direct Surface, “m” is Mixed Minimal Surface, and “l” is for Lines. Finally, We List the Time Cost to Evaluate Each Graph ( $t$  [s])

structure	#vtx	#e	#l	#s	#T	#o	#vox	#all	type	$t$	
Schwarz P <sup>(1)</sup>	6	1	0	1	3	1	1	13	c	0.44	† Our modification/design
Wei’s genus4 <sup>(1)</sup>	7	1	0	1	3	1	1	14	c	5.99	(1)[Weber n. d.]
Neovius <sup>(1)</sup>	4	1	0	1	8	1	1	16	c	0.72	(2)[Chen et al. 2019]
Schoen S’-S” <sup>(1)</sup>	7	1	0	1	3	1	1	14	c	14.16	(3)[Jenett et al. 2020]
Schoen I-WP <sup>(1)</sup>	6	1	0	1	4	1	1	14	c	3.20	(4)[Deshpande et al. 2001]
Deformed Schoen S’-S” <sup>(1)</sup>	7	1	0	1	3	1	1	14	c	21.91	(5)[Hsueh et al. 2019]
Steißmann <sup>(1)</sup>	6	1	0	1	3	1	1	13	d	1.33	(6)[Frenzel et al. 2017]
Schwarz D <sup>(1)</sup>	6	1	0	1	2	1	1	12	d	0.72	(7)[Muhammad and Lim 2021]
Schwarz CLP <sup>(1)</sup>	6	1	0	1	2	1	1	12	d	0.82	(8)[Lu et al. 2017]
Schoen R2 <sup>(1)</sup>	8	4	0	1	3	1	1	18	m	1.46	(9)[Wu et al. 2019]
Hao Chen’s $\alpha\Delta/t\Delta$ <sup>(1)</sup>	6	4	0	1	3	1	1	16	m	0.61	(10)[Mizzi et al. 2018]
Deformed H <sup>(1)</sup>	10	4	0	1	5	1	1	22	m	0.72	(11)[Han et al. 2015]
Gyroid <sup>(1)</sup>	6	1	0	2	16	1	1	27	c	0.57	(12)[Babae et al. 2013].
Schwarz P+Neovius <sup>(2)</sup>	13	2	0	2	9	1	1	28	c	1.56	
Schwarz P+Cubic Open <sup>(2)†</sup>	13	2	0	2	6	1	1	25	c+d	0.89	
Schwarz P+Cubic Closed <sup>(2)</sup>	10	2	0	2	6	1	1	22	c+d	0.93	
assembled <sup>(3)</sup>	7	1	1	0	9	1	1	20	l	0.28	
assembled2 <sup>(3)</sup>	4	2	1	0	6	1	1	15	l	0.21	
assembled3 <sup>(3)†</sup>	5	2	1	0	5	1	1	15	l	0.28	
octet <sup>(4)</sup>	3	1	1	0	5	1	1	12	l	0.24	
auxetic3d <sup>(5)</sup>	6	2	2	0	5	1	1	17	l	0.25	
twist-tilable <sup>(6)</sup>	8	5	5	0	13	1	1	33	l	0.23	
twist-tilable-bezier <sup>(6)†</sup>	9	5	5	0	13	1	1	34	l	0.25	
bandgap <sup>(7)</sup>	6	3	3	0	14	3	1	30	l	1.18	
3D FCC lattice <sup>(8)</sup>	6	3	3	0	3	3	1	19	l	0.43	
anti-chiral <sup>(9)</sup>	4	2	1	0	4	1	1	13	l	0.18	
star-shaped structure <sup>(10)</sup>	5	2	1	1	4	1	1	15	l+d	0.77	
non-manifold surface <sup>†</sup>	8	3	0	3	4	1	1	20	d	1.32	
polyhedral cell <sup>(11)</sup>	6	1	0	1	3	1	1	13	d	0.66	
our composition <sup>†</sup>	12	4	4	1	5	3	1	30	l+c	2.03	
6-hole BCC <sup>(12)</sup>	4	2	3	0	12	5	1	27	l	7.23	

Table 5. **User Study Results.** For Each of the Six Modeling Tasks, We Show Two Randomly Selected User-Created Structures (All Structures are Shown in the Supplement). The Construction Time and Number of Nodes Used for Each Structure are Reported As the “avg (min, max)” Measured Across All 10 Participants

Name Type	Assembled 2 Straight & curved beams	Non-Manifold Direct surface	Wei’s genus 4 Conjugate TPMS	Hao Chen’s $\alpha\Delta/t\Delta$ Mixed Minimal TPMS	FCC Solid Bulk	Combo All Classes
Results						
# Nodes	16 (15, 18)	14.4 (13, 20)	14.1 (14, 15)	17.1 (16, 19)	18.9 (15, 21)	38.2 (31, 52)
$t$ [min]	14.9 (10, 19)	14.2 (7, 28)	13.2 (5, 22)	26.4 (12, 42)	17.1 (11, 22)	21.8 (16, 33)

manufacturing.<sup>2</sup> Figure 10 shows photographs for four of our fabricated structures, which each feature a  $3 \times 3 \times 3$  tiling of our unit cell, with a total size of 9 cm in each dimension and a minimum wall thickness of 1 mm.

<sup>2</sup><https://inkbit3d.com/>.

## 8.2 Material Properties

Our work also presents opportunities for performance-driven design and shape optimization, as even our randomly generated structures exhibit a wide range of interesting material properties. We consider predictions for the stiffness tensor and phononic bandgap, using a base material with Young’s modulus  $E = 1$  Pa, mass density  $\rho = 1$  kg/m<sup>3</sup>, and Poisson’s ratio  $\nu = 0.45$ .

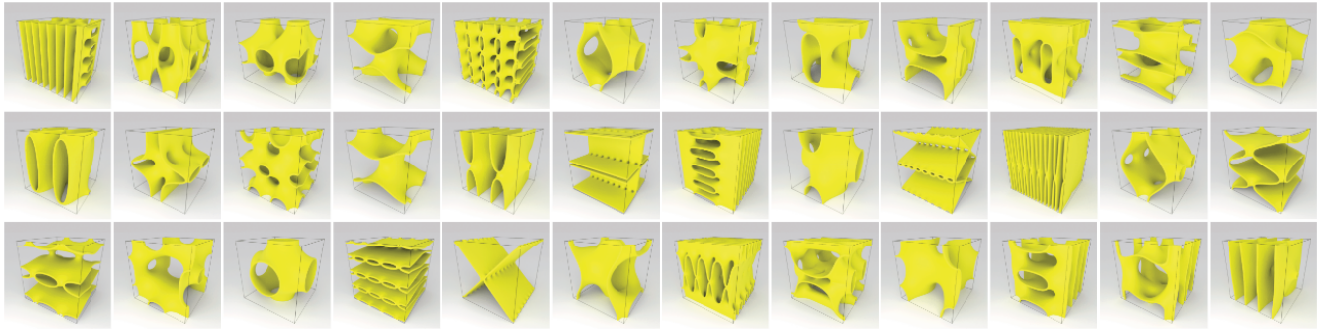


Fig. 9. **Randomly generated minimal surfaces.** Using our simple random exploration method over boundary loops, we have generated a large set of minimal surfaces, including both novel structures and classic TPMS such as the Schwarz P structure (bottom row, third column).

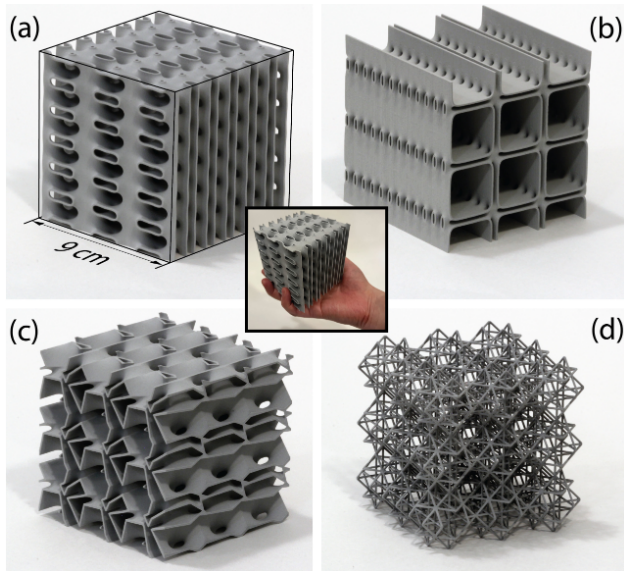


Fig. 10. **Fabricated structures.** We fabricated four structures based on our conjugate surface ((a) and (b)), direct surface (c), and truss-based (d) methods. We selected (a), (b), and (d) to maximize the ratio between Young’s modulus and density, subject to the constraints of the 3D printer ( $\geq 1$  mm thickness). Structure (c) is the result from Figure 11(c) (bottom). The middle photo shows the scale of our 3D-printed structures relative to the hand of an adult male.

**8.2.1 Stiffness Tensor.** We use homogenization to compute the stiffness tensors for all of our established and randomly generated structures. To generate each input mesh, we run voxelization with a grid resolution of  $100^3$ , such that each voxel is of size  $0.01^3$ .

We begin by examining the homogenized material properties for all of the orthotropic structures with thickness 0.02 (Figure 11(a)). The homogenization of an orthotropic material yields a  $6 \times 6$  matrix  $C$  of the following form:

$$C = \begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{21} & C_{22} & C_{23} & 0 & 0 & 0 \\ C_{31} & C_{32} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & C_{44} & 0 & 0 \\ 0 & 0 & 0 & 0 & C_{55} & 0 \\ 0 & 0 & 0 & 0 & 0 & C_{66} \end{bmatrix} \quad (8)$$

After inverting  $C$ , we can extract standard material properties such as  $E_i$ , which is the Young’s modulus along axis  $i \in \{x, y, z\}$ . Figure 11(a) plots each structure’s density against its average Young’s modulus,  $E_{\text{avg}} = (E_x + E_y + E_z)/3$ . Even from our undirected topological exploration, we are able to cover a broad region of this space and begin observing trends between the classes. For example, conjugate shellulars generally yield the best  $E_{\text{avg}}$  for a given density, but direct shellulars exhibit broader property coverage due to their less-constrained skeleton space. Furthermore, our coverage surpasses that of the established structures for a given density.

To examine the role of spatially varying thickness, we generate four additional variations of the truss structures: increasing each beam’s *uniform* thickness to 0.04 or 0.08 and increasing each beam’s *center* thickness to 0.04 or 0.08 (to mimic the well-known Pentamode [Milton and Cherkaev 1995]). As shown in Figure 11(b), the ratio between  $E_{\text{avg}}$  and density remains near constant as thickness increases. We observed a similar relationship for shellulars.

Finally, we examine the  $C$  matrices for our asymmetric structures, which *can* exhibit non-zero entries in the off-diagonal blocks. These entries yield interesting anisotropic material behaviors. Figure 11(c) shows this effect for two of our most extreme structures.

**8.2.2 Phononic Bandgap.** We evaluate the bandgap for half of our randomly generated symmetric shellular structures and select the structure with the largest gap (Figure 12, top). We also reproduce the state-of-the-art structure proposed by Muhammad and Lim [2021] (Figure 12, bottom) and compare the dispersion curves for each. The latter design is far superior, which indicates that we are unlikely to solve this task via uninformed exploration in a subset of the design space (e.g., only shellular structures). However, our concise representation of the state-of-the-art design suggests that an intelligent search over our full procedural graph space could reveal structures with comparable or even superior bandgaps.

## 9 LIMITATIONS AND FUTURE WORK

Although our approach covers a wide variety of metamaterial architectures, it has several limitations. For example, our thickening operations do not necessarily preserve separation between features, which may cause issues for, e.g., interpenetrating lattices [White et al. 2021] or kirigami structures with cuts. Our thickening

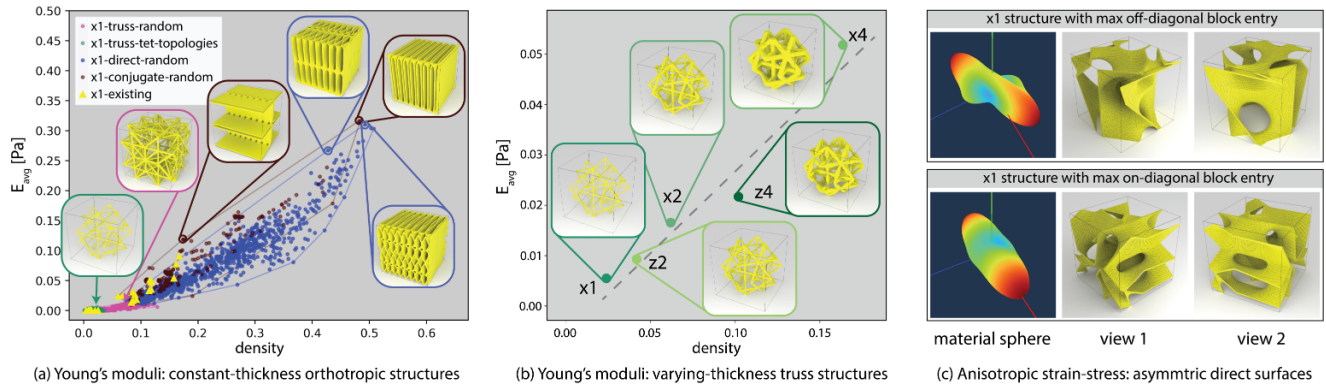


Fig. 11. **Stiffness tensor experiments.** Our preliminary studies suggest broad coverage of the stiffness tensor space. All subfigures assume a uniform baseline thickness of 0.02 (indicated by “x1”). (a) We plot density vs. average Young’s modulus ( $E_{avg}$ ) for structures in five classes: “truss-random,” “direct-random,” and “conjugate-random” are from our random exploration strategies; “truss-tet-topologies” are the topologies from Panetta et al. [2015]; and “existing” are other structures from literature. We show the convex hull of each class (except “existing”) along with structures that exhibit higher  $E_{avg}$  and/or lower density than comparable known structures. (b) We explore the effect of thickening on density vs.  $E_{avg}$ . Structures marked “x2”/“x4” are uniformly  $2\times/4\times$  the baseline thickness; for “x2”/“x4,” the beam centers are  $2\times/4\times$  the baseline. As thickness increases, both density and  $E_{avg}$  increase in a near-linear relationship. (c) Our asymmetric direct surfaces exhibit strong anisotropy due to their non-zero entries in all 21 **degrees of freedom (DOFs)** of  $C$ . The largest absolute entry in the off-/on-diagonal  $3\times 3$  block of  $C$  was exhibited by the top/bottom structure, respectively. The material spheres show their omni-directional strain–stress relationship, where color and distance to the center show the strain response to uniform stress in that direction.

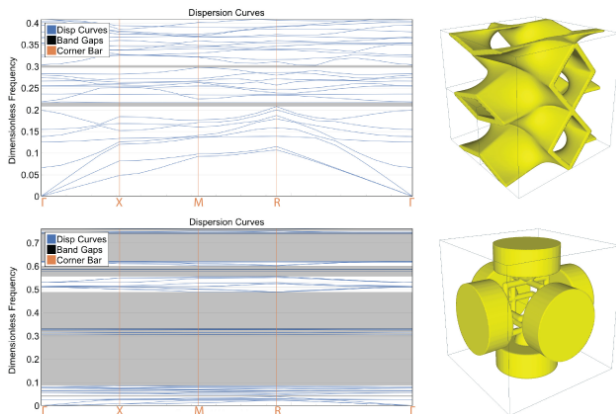


Fig. 12. **Selected phononic bandgaps.** We show the dispersion curves for two structures: (top) the randomly-generated shellular with the largest predicted bandgap and (bottom) the state-of-the-art metamaterial for phononic bandgap [Muhammad and Lim 2021]. Each dispersion curve plot shows normalized frequency values on the  $y$ -axis; the  $x$ -axis represents wave vectors along the irreducible first Brillouin zone for cubic-symmetric structures. Each vertical slice of the plot shows the structure’s eigenmodes under a particular wave vector; thus, any horizontal band without curves (gray) indicates a bandgap, i.e., a frequency range with no mode of transmission. Wide gaps in low frequency ranges are the most desirable.

operations are also presently limited to simple sphere- or normal-extrusion with spatially varying thickness. We could provide more flexibility via other solidification routines, such as user-defined cross-section profiles for generating triangular prisms or I-beams. To mitigate areas of high stress concentration, we could also introduce blending methods that control the shape of junctions between abutting skeletal elements. The convex-hull-restricted blending approach [Colli Tozoni et al. 2020; Panetta et al. 2017] offers one

such solution for beam structures, which could be integrated into our system by adding blending parameters on the vertices of line skeletons. However, this approach would need to be generalized to accommodate surfaces, solid bulks, and subtractive Boolean operations, as well as skeleton intersections that occur away from user-defined vertices.

Our work is also presently limited to a single cell of a regular metamaterial residing in a unit cube. We could expand this scope by exploring additional tilable cells such as hexagonal prisms or rhombic dodecahedral honeycombs. We could also integrate routines for stochastic porous structures like foams or for partitioning our graph into explicit, reusable subgraphs that could facilitate the design of complex hybrid structures or multi-scale designs [Rao et al. 2019; Zhang et al. 2021]. Finally, future work could consider ways to smoothly connect a set of distinct cells to facilitate the construction of larger volumes with, e.g., functional grading [Al-Ketan and Abu Al-Rub 2021; Hu et al. 2022b] or smooth transitions between structures of different classes (e.g., trusses to shells).

In a mathematical vein, we could also explore the limits of our CSCM approach w.r.t. the period problem discussed in Section 4.1. If robust, then our approach could facilitate the discovery of new examples or perhaps even a general solution for the problem of handle insertion. Alternatively, we could incorporate, e.g., *minimal twin surfaces* [Chen 2019] or *Willmore surfaces* [Willmore 1965], which are a superset of minimal surfaces with *constant* mean curvature. In conjunction with volume-preserving metrics, this could lead to a host of interesting new structures.

We also look forward to devising optimization schemes over our representation to permit the automatic discovery and interactive user-in-the-loop design of metamaterial structures with extremal properties. Toward this goal, our system would benefit from the development of a robust, principled GUI and expanded simulation capabilities, including nodes for, e.g., non-linear



simulators, stress–strain curves, and tetrahedralization. It would also be interesting to explore whether we could perform analysis and/or property prediction directly on our graph representation to permit faster exploration and optimization techniques free of meshing- or simulation-related bottlenecks and sensitivities.

## 10 CONCLUSION

We have presented a simple, compact procedural graph for the construction of a wide range of formerly disparate cellular metamaterial architectures: trusses, solid bulks, and shells, including TPMS-based structures. Within this, we have also developed a practical, easy-to-use implementation of a state-of-the-art construction method for TPMS and characterized a simple, unified design space for a wide range of thickness-annotated metamaterial skeletons. We have demonstrated our representation’s accuracy and generality by expressing a large collection of structures found in mechanical engineering and material science literature using only a few graph nodes. We have also verified our representation’s intuitiveness through an extensive user study. Finally, we have demonstrated our method’s potential w.r.t. a number of exciting applications and future works by generating thousands of structures with considerable diversity in terms of both visual structure and material properties.

## ACKNOWLEDGMENTS

The authors thank Mina Konaković Luković and Michael Foshey for their early contributions to this project, David Palmer and Paul Zhang for their insightful discussions about minimal surfaces and the CSCM, Julian Panetta for providing the Elastic Textures code, and Hannes Hergeth for his feedback and support. We also thank our user study participants and anonymous reviewers.

## REFERENCES

Mostafa Akbari, Armin Mirabolghasemi, Hamid Akbarzadeh, and Masoud Akbarzadeh. 2020. Geometry-based structural form-finding to design architected cellular solids. In *Proceedings of the Symposium on Computational Fabrication (SCF’20)*. Association for Computing Machinery, New York, NY, 11 pages.

Mostafa Akbari, Armin Mirabolghasemi, Mohammad Bolhassani, Abdolhamid Akbarzadeh, and Masoud Akbarzadeh. 2022. Strut-based cellular to shellular funicular materials. *Adv. Funct. Mater.* 32, 4 (2022), 2109725.

Oraib Al-Ketan and Rashid K. Abu Al-Rub. 2021. MSLattice: A free software for generating uniform and graded lattices based on triply periodic minimal surfaces. *Mater. Des. Process. Commun.* 3, 6 (2021), e205.

Brian Amberg, Sami Romdhani, and Thomas Vetter. 2007. Optimal step nonrigid ICP algorithms for surface registration. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1–8.

Rita Ambu and Anna Eva Morabito. 2019. Modeling, assessment, and design of porous cells based on schwartz primitive surface for bone scaffolds. *Sci. World J.* 2019 (2019), 7060847.

Michael F. Ashby. 2006. The properties of foams and lattices. *Philos. Trans. Roy. Soc. A: Math. Phys. Eng. Sci.* 364, 1838 (2006), 15–30.

Arash Ataee, Yuncang Li, Darren Fraser, Guangsheng Song, and Cuie Wen. 2018. Anisotropic Ti-6Al-4V gyroid scaffolds manufactured by electron beam melting (EBM) for bone implant applications. *Mater. Des.* 137 (2018), 345–354.

Reza Attarzadeh, Seyed-Hosein Attarzadeh-Niaki, and Christophe Duwig. 2022. Multi-objective optimization of TPMS-based heat exchangers for low-temperature waste heat recovery. *Appl. Therm. Eng.* 212 (2022), 118448.

Sahab Babae, Jongmin Shim, James C. Weaver, Elizabeth R. Chen, Nikita Patel, and Katia Bertoldi. 2013. 3D soft metamaterials with negative Poisson’s ratio. *Adv. Mat.* 25, 36 (2013), 5044–5049.

Andreas Bærentzen and Eva Rotenberg. 2021. Skeletonization via local separators. *ACM Trans. Graph.* 40, 5 (2021), 18 pages.

Jan-Hendrik Bastek, Siddhant Kumar, Bastian Telgen, Raphaël N. Glaesener, and Dennis M. Kochmann. 2022. Inverting the structure–property map of truss metamaterials by deep learning. *Proc. Natl. Acad. Sci. U.S.A.* 119, 1 (2022), e2111505119.

Katia Bertoldi, Vincenzo Vitelli, Johan Christensen, and Martin Van Hecke. 2017. Flexible mechanical metamaterials. *Nature Rev. Mater.* 2, 11 (2017).

Harry Blum. 1967. A transformation for extracting new descriptors of shape. In *Models for the Perception of Speech and Visual Form*. 19, 5 (1967), 362–380.

Pedro Boechat, Mark Dokter, Michael Kenzel, Hans-Peter Seidel, Dieter Schmalstieg, and Markus Steinberger. 2016. Representing and scheduling procedural generation using operator graphs. 35, 6 (2016), 12 pages.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph.* 33, 4 (2014), 11 pages.

Kenneth A. Brakke. 1992. The surface evolver. *Exp. Math.* 1, 2 (1992), 141–165.

Yu-Chin Chan, Faez Ahmed, Liwei Wang, and Wei Chen. 2020. METASET: Exploring shape and property spaces for data-driven metamaterials design. *J. Mech. Des.* 143, 3 (2020), 031707.

Desai Chen, Mélina Skouras, Bo Zhu, and Wojciech Matusik. 2018. Computational discovery of extremal microstructure families. *Sci. Adv.* 4, 1 (2018), eaao7005.

Hao Chen. 2019. Minimal twin surfaces. *Exp. Math.* 28, 4 (2019), 404–419.

Hao Chen and Matthias Weber. 2021. A new deformation family of Schwarz’ D surface. *Trans. Am. Math. Soc.* 374, 4 (2021), 2785–2803.

Zeyao Chen, Yi Min Xie, Xian Wu, Zhe Wang, Qing Li, and Shiwei Zhou. 2019. On hybrid cellular materials based on triply periodic minimal surfaces with extreme mechanical properties. *Mater. Des.* 183 (2019), 108109.

Davi Colli Tozoni, Jérémie Dumas, Zhongshi Jiang, Julian Panetta, Daniele Panozzo, and Denis Zorin. 2020. A low-parametric rhombic microstructure family for irregular lattices. *ACM Trans. Graph.* 39, 4 (July 2020), 20 pages.

Robert L. Cook. 1984. Shade trees. *SIGGRAPH Comput. Graph.* 18, 3 (Jan 1984), 223–231.

Vikram S. Deshpande, Norman A. Fleck, and Michael F. Ashby. 2001. Effective properties of the octet-truss lattice material. *J. Mech. Phys. Solids* 49, 8 (2001), 1747–1769.

Mario Deuss, Anders Holden Deleuran, Sofien Bouaziz, Bailin Deng, Daniel Piker, and Mark Pauly. 2015. ShapeOp – A robust and extensible geometric modelling paradigm. In *Modelling Behaviour*. Springer, Berlin, 505–515.

Gerhard Dziuk. 1990. An algorithm for evolutionary surfaces. *Numer. Math.* 58, 1 (1990), 603–611.

Zhaohui Fan, Renjing Gao, and Shutian Liu. 2022. Thermal conductivity enhancement and thermal saturation elimination designs of battery thermal management system for phase change materials based on triply periodic minimal surface. *Energy* 259 (2022), 125091.

Jiawei Feng, Bo Liu, Zhiwei Lin, and Jianzhong Fu. 2021. Isotropic porous structure design methods based on triply periodic minimal surfaces. *Mater. Des.* 210 (2021), 110050.

Tobias Frenzel, Muamer Kadic, and Martin Wegener. 2017. Three-dimensional mechanical metamaterials with a twist. *Science* 358, 6366 (2017), 1072–1074.

Darren Matthew Garbuz. 2010. *Isoperimetric Properties of Some Genus 3 Triply Periodic Minimal Surfaces Embedded in Euclidean Space*. Master’s Thesis. Southern Illinois University Edwardsville.

Lorna J. Gibson, Michael F. Ashby, and Brendan A. Harley. 2010. *Cellular Materials in Nature and Medicine*. Cambridge University Press.

Aldair E. Gongora, Siddharth Mysore, Beichen Li, Wan Shou, Wojciech Matusik, Elise F. Morgan, Keith A. Brown, and Emily Whiting. 2021. Designing composites with target effective Young’s modulus using reinforcement learning. In *Proceedings of the Symposium on Computational Fabrication (SCF’21)*. Association for Computing Machinery, New York, NY, 1–11.

Jeremy Gray and Mario Micalef. 2007. The Work of Jesse Douglas on Minimal Surfaces. arXiv:0710.5478 [math.HO].

Seung Chul Han, Jeong Woo Lee, and Kiju Kang. 2015. A new type of low density material: Shellular. *Adv. Mater.* 27, 37 (2015), 5506–5511.

Jenny Harrison and Harrison Pugh. 2016. *Plateau’s Problem*. Springer International Publishing, Cham.

Meng-Ting Hsieh and Lorenzo Valdevit. 2020. Minisurf—A minimal surface generator for finite element modeling and additive manufacturing. *Softw. Impacts* 6 (2020), 100026.

Chun-Hway Hsueh, Siegfried Schmauder, Chuin-Shan Chen, Krishan Kumar Chawla, Nikhilesh Chawla, Weiqiu Chen, Yutaka Kagawa, et al. 2019. *Handbook of Mechanics of Materials*. Springer.

Jiangbei Hu, Shengfa Wang, Baojun Li, Fengqi Li, Zhongxuan Luo, and Ligang Liu. 2022b. Efficient representation and optimization for TPMS-based porous structures. *IEEE Trans. Vis. Comput. Graph.* 28, 7 (Jul 2022), 2615–2627.

Yiwei Hu, Chengan He, Valentin Deschaintre, Julie Dorsey, and Holly Rushmeier. 2022a. An inverse procedural modeling pipeline for SVBRDF maps. *ACM Trans. Graph.* 41, 2 (Jan 2022), 17 pages.

Alexandra Ion, Johannes Frohnhofer, Ludwig Wall, Robert Kovacs, Mirela Alistar, Jack Lindsay, Pedro Lopes, Hsiang-Ting Chen, and Patrick Baudisch. 2016. Metamaterial mechanisms. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST’16)*. Association for Computing Machinery, New York, NY, 529–539.

Alexandra Ion, David Lindlbauer, Philipp Herholz, Marc Alexa, and Patrick Baudisch. 2019. Understanding metamaterial mechanisms. In *Proceedings of the CHI*

- Conference on Human Factors in Computing Systems (CHI'19). Association for Computing Machinery, New York, NY, 1–14.
- Alexandra Ion, Ludwig Wall, Robert Kovacs, and Patrick Baudisch. 2017. Digital mechanical metamaterials. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI'17)*. Association for Computing Machinery, New York, NY, 977–988.
- Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. 2011. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.* 30, 4 (2011), 99–106.
- Alec Jacobson, Daniele Panozzo, et al. 2018. libigl: A Simple C++ Geometry Processing Library. Retrieved from <https://libigl.github.io/>
- Benjamin Jenett, Christopher Cameron, Filippos Tourlomis, Alfonso Parra Rubio, Megan Ochalek, and Neil Gershenfeld. 2020. Discretely assembled mechanical metamaterials. *Sci. Adv.* 6, 47 (2020), eabc9943.
- Alistair Jones, Martin Leary, Stuart Bateman, and Mark Easton. 2021. TPMS designer: A tool for generating and analyzing triply periodic minimal surfaces. *Softw. Impacts* 10 (2021).
- Hermann Karcher. 1989. The triply periodic minimal surfaces of Alan Schoen and their constant mean curvature companions. *Manuscr. Math.* 64, 3 (1989), 291–357.
- Hermann Karcher and Konrad Polthier. 1996. Construction of triply periodic minimal surfaces. *Philos. Trans.: Math. Phys. Eng. Sci.* 354, 1715 (1996), 2077–2104.
- Saeed Khaleghi, Fayyaz N. Dehnavi, Mostafa Baghani, Masoud Safdari, Kui Wang, and Majid Baniassadi. 2021. On the directional elastic modulus of the TPMS structures and a novel hybridization method to control anisotropy. *Mater. Des.* 210 (2021), 110074.
- Mina Konaković, Keenan Crane, Bailin Deng, Sofien Bouaziz, Daniel Piker, and Mark Pauly. 2016. Beyond developable: Computational design and fabrication with auxetic materials. *ACM Trans. Graph.* 35, 4 (2016), 11 pages.
- Vojtěch Krs, Radomír Měch, Mathieu Gaillard, Nathan Carr, and Bedřich Benes. 2021. PICO: Procedural iterative constrained optimizer for geometric modeling. *IEEE Trans. Vis. Comput. Graph.* 27, 10 (2021), 3968–3981.
- Siddhant Kumar, Stephanie Tan, Li Zheng, and Dennis M. Kochmann. 2020. Inverse-designed spinoid metamaterials. *npi Comput. Mater.* 6, 1 (2020), 73.
- H. Blaine Lawson. 1970. Complete minimal surfaces in  $S^3$ . *Ann. Math.* 92, 3 (1970), 335–374.
- Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J. Gortler. 2008. A local/global approach to mesh parameterization. In *Computer Graphics Forum*, 27, 5 (2008), 1495–1504.
- William E. Lorensen and Harvey E. Cline. 1987. Marching cubes: A high resolution 3D surface construction algorithm. *ACM Siggraph Comput. Graph.* 21, 4 (1987), 163–169.
- Lin Lu, Andrei Sharf, Haisen Zhao, Yuan Wei, Qingnan Fan, Xuelin Chen, Yann Savoye, Changhe Tu, Daniel Cohen-Or, and Baoquan Chen. 2014. Build-to-last: Strength to weight 3D printed objects. *ACM Trans. Graph.* 33, 4 (July 2014), 10 pages.
- Yan Lu, Yang Yang, James K. Guest, and Ankit Srivastava. 2017. 3-D phononic crystals with ultra-wide band gaps. *Sci. Rep.* 7, 1 (2017), 43407.
- Jonàs Martínez, Mélina Skouras, Christian Schumacher, Samuel Hornus, Sylvain Lefebvre, and Bernhard Thomaszewski. 2019. Star-shaped metrics for mechanical metamaterial design. *ACM Trans. Graph.* 38, 4 (2019), 13 pages.
- Ian Maskery, Luke A. Parry, Daniel Padrão, Richard J. M. Hague, and Ian A. Ashcroft. 2022. Flatt pack: A research-focussed lattice design program. *Add. Manufact.* 49 (2022).
- William H. Meeks, III. 1975. *The Geometry and the Conformal Structure of Triply Periodic Minimal Surfaces in  $S^3$* . Ph.D. Dissertation. University of California, Berkeley.
- Élie Michel and Tamy Boubekeur. 2021. DAG amendment for inverse control of parametric shapes. *ACM Trans. Graph.* 40, 4 (July 2021), 14 pages.
- Graeme W. Milton and Andrej V. Cherkaev. 1995. Which elasticity tensors are realizable? *Journal of Engineering Materials and Technology* 117, 4 (10 1995), 483–493.
- Luke Mizzi, E. M. Mahdi, Kirill Titov, Ruben Gatt, Daphne Attard, Kenneth E. Evans, Joseph N. Grima, and Jin-Chong Tan. 2018. Mechanical metamaterials with star-shaped pores exhibiting negative and zero Poisson's ratio. *Mater. Des.* 146 (2018), 28–37.
- Muhammad and C. W. Lim. 2021. Phononic metastructures with ultrawide low frequency three-dimensional bandgaps as broadband low frequency filter. *Sci. Rep.* 11, 1 (2021), 7137.
- Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. 2005. Meshless deformations based on shape matching. *ACM Trans. Graph.* 24, 3 (2005), 471–478.
- John A. Nelder and Roger Mead. 1965. A simplex method for function minimization. *Comput. J.* 7, 4 (1965), 308–313.
- Ban Dang Nguyen, Jeong Shik Cho, and Kiju Kang. 2016. Optimal design of “Shellular”, a micro-architected material with ultralow density. *Mater. Des.* 95 (2016), 490–500.
- Jifei Ou, Zhao Ma, Jannik Peters, Sen Dai, Nikolaos Vlavianos, and Hiroshi Ishii. 2018. KinetiX - designing auxetic-inspired deformable material structures. *Comput. Graph.* 75 (2018), 72–81.
- Mariam Ouda, Oraib Al-Ketan, Nurshaun Sreedhar, Mohamed I. Hasan Ali, Rashid K. Abu Al-Rub, Seungkwon Hong, and Hassan A. Arafat. 2020. Novel static mixers based on triply periodic minimal surface (TPMS) architectures. *J. Environ. Chem. Eng.* 8, 5 (2020), 104289.
- Julian Panetta, Abtin Rahimian, and Denis Zorin. 2017. Worst-case stress relief for microstructures. *ACM Trans. Graph.* 36, 4 (July 2017), 16 pages.
- Julian Panetta, Qingnan Zhou, Luigi Malomo, Nico Pietroni, Paolo Cignoni, and Denis Zorin. 2015. Elastic textures for additive fabrication. *ACM Trans. Graph.* 34, 4 (July 2015), 12 pages.
- Ken Perlin. 1985. An image synthesizer. *SIGGRAPH Comput. Graph.* 19, 3 (July 1985), 287–296.
- Ulrich Pinkall and Konrad Polthier. 1993. Computing discrete minimal surfaces and their conjugates. *Exp. Math.* 2, 1 (1993), 15–36.
- Przemyslaw Prusinkiewicz and Aristid Lindenmayer. 2004. *The Algorithmic Beauty of Plants*. Electronic Edition.
- Zhao Qin, Gang Seob Jung, Min Jeong Kang, and Markus J. Buehler. 2017. The mechanics and design of a lightweight three-dimensional graphene assembly. *Sci. Adv.* 3, 1 (2017), e1601536.
- Mats Åberg and Peter Gudmundson. 1997. The usage of standard finite element codes for computation of dispersion relations in materials with periodic microstructure. *J. Acoust. Soc. Am.* 102, 4 (1997), 2007–2013.
- Cong Rao, Fan Xu, Lihao Tian, and Lin Lu. 2019. Bi-scale porous structures. In *Proceedings of the SMI 2019 Fabrication & Sculpting Event (FASE'19)*. 73–76.
- Ulrich Reitebuch, Martin Skrodzki, and Konrad Polthier. 2019. Discrete gyroid surface. In *Proceedings of Bridges 2019: Mathematics, Art, Music, Architecture, Education, Culture*. Tessellations, Phoenix, AZ, 461–464.
- Tobias A. Schaedler and William B. Carter. 2016. Architected cellular materials. *Annu. Rev. Mater. Res.* 46, 1 (2016), 187–210.
- Olaf Schenk and Klaus Gärtner. 2004. Solving unsymmetric sparse systems of linear equations with PARDISO. *Fut. Gener. Comput. Syst.* 20, 3 (2004), 475–487.
- Alan H. Schoen. 2008. On the graph (10-3)-a. *Notices of the AMS* 55, 6 (2008), 663. Available at <http://www.ams.org/notices/200806/tx080600663p.pdf> (accessed 02-22-2023).
- Christian Schumacher, Bernd Bickel, Jan Rys, Steve Marschner, Chiara Daraio, and Markus Gross. 2015. Microstructures to control elasticity in 3D printing. *ACM Trans. Graph.* 34, 4 (2015), 13 pages.
- Liang Shi, Beichen Li, Miloš Hašan, Kalyan Sunkavalli, Tamy Boubekeur, Radomir Mech, and Wojciech Matusik. 2020. Match: Differentiable material graphs for procedural material capture. *ACM Trans. Graph.* 39, 6 (November 2020), 15 pages.
- Madlaina Signer, Alexandra Ion, and Olga Sorkine-Hornung. 2021. Developable metamaterials: Mass-fabricable metamaterials by laser-cutting elastic structures. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI'21)*. Association for Computing Machinery, New York, NY, 1–13.
- Ruben M. Smelik, Tim Tuteneel, Rafael Bidarra, and Bedřich Benes. 2014. A survey on procedural modelling for virtual worlds. *Comput. Graph. Forum* 33, 6 (September 2014), 31–50.
- Alessandro Spadoni, Reinhard Höhler, Sylvie Cohen-Addad, and Vladimir Dorodnitsyn. 2014. Closed-cell crystalline foams: Self-assembling, resonant metamaterials. *J. Acoust. Soc. Am.* 135, 4 (2014), 1692–1699.
- James Utama Surjadi, Libo Gao, Hui Feng Du, Xiang Li, Xiang Xiong, Nicholas Xu-anlai Fang, and Yang Lu. 2019. Mechanical metamaterials and their engineering applications. *Adv. Eng. Mater.* 21, 3 (2019), 1800864.
- Andrea Tagliasacchi, Ibraheem Alhashim, Matt Olson, and Hao Zhang. 2012. Mean curvature skeletons. *Comput. Graph. Forum* 31, 5 (2012), 1735–1744.
- Andrea Tagliasacchi, Thomas Delame, Michela Spagnuolo, Nina Amenta, and Alexandru Telea. 2016. 3D skeletons: A state-of-the-art report. *Comput. Graph. Forum* 35, 2 (2016), 573–597.
- Lihao Tian, Lin Lu, Weikai Chen, Yang Xia, Charlie C. L. Wang, and Wenping Wang. 2020. Organic open-cell porous structure modeling. In *Proceedings of the Symposium on Computational Fabrication (SCF'20)*. Association for Computing Machinery, New York, NY, 12 pages.
- Stephanie Wang and Albert Chern. 2021. Computing minimal surfaces with differential forms. *ACM Trans. Graph.* 40, 4 (2021), 14 pages.
- Matthias Weber. [n. d.]. Triply Periodic Surfaces. Retrieved February 22, 2023 from <https://minimalsurfaces.blog/home/repository/triply-periodic/>
- Matthias Weber. 2018a. Mathematica Notebook for Gyroid. Minimal Surfaces Repository. Retrieved February 22, 2023 from <https://minimalsurfaces.blog/home/repository/triply-periodic/gyroid/>
- Matthias Weber. 2018b. Mathematica Notebook for Neovius Surface. Minimal Surfaces Repository. Retrieved February 22, 2023 from <https://minimalsurfaces.blog/home/repository/triply-periodic/neovius-surface/>
- Matthias Weber. 2018c. Mathematica Notebook for Schoen I-WP. Minimal Surfaces Repository. Retrieved February 22, 2023 from <https://minimalsurfaces.blog/home/repository/triply-periodic/schoen-i-wp/>

- Matthias Weber. 2018d. Mathematica Notebook for Schwarz P-Surface. Minimal Surfaces Repository. Retrieved February 22, 2023 from <https://minimalsurfaces.blog/home/repository/triply-periodic/schwarz-p-surface/>
- Benjamin C. White, Anthony Garland, Ryan Alberdi, and Brad L. Boyce. 2021. Interpenetrating lattices with enhanced mechanical functionality. *Add. Manufact.* 38 (2021), 101741.
- Emily Whiting, John Ochsendorf, and Frédo Durand. 2009. Procedural modeling of structurally-sound masonry buildings. In *ACM SIGGRAPH Asia 2009 Papers (SIGGRAPH Asia'09)*. ACM, New York, NY, 9 pages.
- Thomas J. Willmore. 1965. Note on embedded surfaces. *An. Sti. Univ. "Al. I. Cuza" Iasi Sect. I a Mat.(NS) B* 11, 493-496 (1965).
- Wenwang Wu, Wenxia Hu, Guian Qian, Haitao Liao, Xiaoying Xu, and Filippo Berto. 2019. Mechanical design and multifunctional applications of chiral mechanical metamaterials: A review. *Mater. Des.* 180 (2019), 107950.
- Chunze Yan, Liang Hao, Lei Yang, Ahmed Yussuf Hussein, Philippe G. Young, Zhaoqing Li, and Yan Li. 2021. *Tripoly Periodic Minimal Surface Lattices Additively Manufactured by Selective Laser Melting*. Elsevier Science.
- Xin Yan, Cong Rao, Lin Lu, Andrei Sharf, Haisen Zhao, and Baoquan Chen. 2020. Strong 3D printing by TPMS injection. *IEEE Trans. Vis. Comput. Graph.* 26, 10 (2020), 3037–3050.
- Lei Zhang, Zhiheng Hu, Michael Yu Wang, and Stefanie Feih. 2021. Hierarchical sheet tripoly periodic minimal surface lattices: Design, geometric and mechanical performance. *Mater. Des.* 26, 10 (2020), 3037–3050.
- Frank W. Zok, Ryan M. Latture, and Matthew R. Begley. 2016. Periodic truss structures. *J. Mech. Phys. Solids* 96 (2016), 184–203.

Received 19 October 2022; revised 8 March 2023; accepted 17 May 2023