Volume Parametrization Quantization for Hexahedral Meshing

HENDRIK BRÜCKLER, Osnabrück University, Germany
DAVID BOMMES, University of Bern, Switzerland
MARCEL CAMPEN, Osnabrück University, Germany

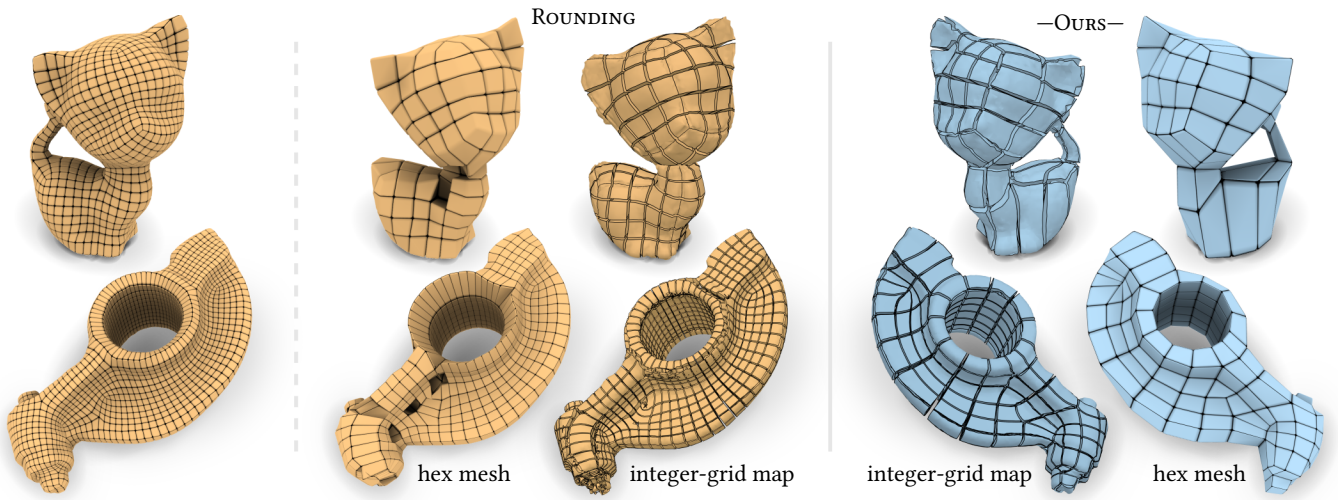


Fig. 1. Volumetric integer-grid maps for hexahedral meshing are commonly generated using a classical integer rounding procedure (orange). This rounding is fragile, its probability of success depends in particular on the desired grid resolution. While for sufficiently fine results it can be successful (far left), for a coarser target resolution, map degeneracies emerge (center right), implying defects in the mesh (center left); notice the holes, the missing tail, and an entire missing outermost layer, exposing interior singularities. Our method (blue), by contrast, is guaranteed to yield conflict-free integer values regardless of resolution, even if extremely coarse. Besides the meshes, the underlying integer-grid maps are visualized via cut out integer-iso-surfaces on the original model.

Developments in the field of parametrization-based quad mesh generation on surfaces have been impactful over the past decade. In this context, an important advance has been the replacement of error-prone rounding in the generation of integer-grid maps, by robust quantization methods. In parallel, parametrization-based hex mesh generation for volumes has been advanced. In this volumetric context, however, the state-of-the-art still relies on fragile rounding, not rarely producing defective meshes, especially when targeting a coarse mesh resolution. We present a method to robustly quantize volume parametrizations, i.e., to determine guaranteed valid choices of integers for 3D integer-grid maps. Inspired by the 2D case, we base our construction on a non-conforming cell decomposition of the volume, a 3D analogue of a T-mesh. In particular, we leverage the motorcycle complex, a recent generalization of the motorcycle graph, for this purpose. Integer values are expressed in a differential manner on the edges of this complex, enabling the efficient formulation of the conditions required to strictly prevent forcing the map into degeneration. Applying our method in the context of hexahedral meshing, we demonstrate that hexahedral meshes can be generated with significantly improved flexibility.

Authors' addresses: Hendrik Brückler, hendrik.brueckler@uos.de, Osnabrück University, Germany; David Bommes, david.bommes@inf.unibe.ch, University of Bern, Switzerland; Marcel Campen, campen@uos.de, Osnabrück University, Germany.

0730-0301/2022/7-ART60
https://doi.org/10.1145/3528223.3530123

CCS Concepts: • **Computing methodologies → Computer graphics**; **Mesh models**; **Mesh geometry models**; *Shape modeling*.

Additional Key Words and Phrases: block-structured, multi-block, T-mesh, hexahedral mesh, volume mesh, block decomposition, base complex

ACM Reference Format:
Hendrik Brückler, David Bommes, and Marcel Campen. 2022. Volume Parametrization Quantization for Hexahedral Meshing. *ACM Trans. Graph.* 41, 4, Article 60 (July 2022), 19 pages. https://doi.org/10.1145/3528223.3530123

1 INTRODUCTION

The discretization of 3D objects and domains by means of volumetric meshes is a cornerstone of many techniques in computer graphics, engineering, and further areas. The algorithmic generation of (semi-structured) hexahedral meshes [Pietroni et al. 2022] is a greater challenge than the generation of (unstructured) tetrahedral or polyhedral meshes, but the effort may pay off in terms of higher efficiency and accuracy observed in various contexts [Benzley et al. 1995; Blacker 2001; Bourdin et al. 2007; Cifuentes and Kalbag 1992; Ramos and Simões 2006; Sarrate Ramos et al. 2014; Tadepalli et al. 2011; Wang et al. 2004, 2021]—though there is ongoing debate about pros and cons [Schneider et al. 2019].

Integer-Grid Maps. A number of algorithmic strategies have been explored to this end. Parametrization-based hex mesh generation using 3D *integer-grid maps* has received particular attention lately, following major advances regarding quad mesh generation using

2D integer-grid maps, as defined by [Bommes et al. 2013]. The parametrization-based approach can offer flexible control over sizing, alignment, regularity, and anisotropy, and allows leveraging well-explored map distortion optimization techniques to promote high mesh quality.

An integer-grid map is a special restricted case of so-called seamless parametrizations (cf. Fig. 2): certain degrees of freedom need to be integer-valued—in accordance with the discrete nature of the finite quad or hex mesh it is supposed to imply. The choice of integer values is critical: it determines not only the mesh's resolution, also its structural quality, and in particular its validity (in the sense of absence of defects such as holes or non-hexahedral cells). The crucial problem of choosing integer values therefore has received repeated attention, starting from naive rounding, over a series of improvements in robustness, fidelity, and flexibility—at least for the now mature 2D case. In the 3D case, by contrast, fragile rounding strategies, originally described for this case in the seminal CubeCover paper [Nieser et al. 2011], have remained the state-of-the-art.

Quantization. In the 2D case, robustness was ultimately achieved by settling the integer degrees of freedom not individually but in a globally coordinated manner—referred to as quantization. The use of a differential representation (using differences as primary integer variables) on top of a non-conforming rectangular (T-mesh) partition of the surface was a key development [Campen et al. 2015]. The motorcycle graph [Eppstein et al. 2008] proved to be particularly suitable to generate an adequate T-mesh structure.

We generalize this approach to the 3D case, leveraging a recently proposed motorcycle complex [Brückler et al. 2022] that allows generating a 3D analogue of a T-mesh (cf. Fig. 3). In this way we provide a reliable and flexible solution to the volumetric parametrization quantization problem, offering the guarantee that the chosen integers do not force the integer-grid map into degeneration.

Challenges. While inspiration can be drawn from the 2D case, a solution for the volumetric 3D case requires major deviation from the established in multiple regards. A key difference, and cause of major challenges, is related to the *singularities*, responsible for extraordinary elements in the resulting mesh, which are the critical entities in the context of quantization. In 2D, these are simple isolated points. In 3D, they are complex networks of curves (cf. Fig. 3). Domain boundaries are the second type of critical entities. In 2D, they are simple curves. In 3D, they are surfaces, and they may interact with the singularity network. These differences have hindered the extension of the involved 2D algorithms to the 3D case.

Our method, outlined in the following, overcomes these challenges. It thereby closes one of the three main robustness gaps (discussed further in Sec. 2) that remained so far between parametrization-based surface quad meshing and volume hex meshing:

① singularity meshability,
② local injectivity,
③ quantization validity.

Namely, it closes gap ③, providing a first reliable solution to this problem. This brings the volume case one step closer to a mature and satisfactory situation, while multiple ongoing efforts to address the other two gaps can currently be witnessed.

1.1 Method Overview

Input. We follow a setting analogous to the 2D case [Campen et al. 2015; Lyon et al. 2019, 2021a] and take as input a seamless (but non-integer) parametrization as illustrated in Fig. 2 left, albeit a volumetric one on a tetrahedral mesh, defined formally in Sec. 3.1. Alternatively, as may be of interest for certain use cases (see Sec. 7.2.1), a hexahedral mesh could be taken as input; it trivially induces a seamless parametrization of its volume by mapping each hexahedron to the unit cube, with compliant transitions, so this case does not require any dedicated attention.

Goal. To generate a parametrization that is similar to the input, but, by contrast, is an integer-grid map (Fig. 2 right) and thereby readily usable for hexahedral mesh extraction, values for the relevant integer degrees of freedom need to be chosen. These are related to singularities and boundaries of the parametrization and need to be chosen carefully in such a way that a valid map for these exists. In particular, it must be prevented that these force topologically separate elements into parametric coincidence or degeneration. Besides this, low restrictions should be placed on the choice of integers, so as to maximize flexibility of control over density and quality of the resulting quantized parametrization, and—by implication—the hexahedral mesh extractable from it.

Approach. First, a parametrization-aligned motorcycle complex (MC) is constructed on the input, as described in [Brückler et al. 2022]. We then formulate a quadratic integer problem that assigns integer lengths to the arcs of the MC (cf. Fig. 3), subject to several types of constraints assuring the validity of the assignment under the requirement of, in particular, seamlessness and separation of critical entities. Using a tailored lazy constraint mechanism, we solve this problem efficiently. Finally, through integration of the resulting differential integer values, a valid quantization, i.e. an integer choice for the discrete degrees of freedom permitting a valid integer-grid map, is obtained.

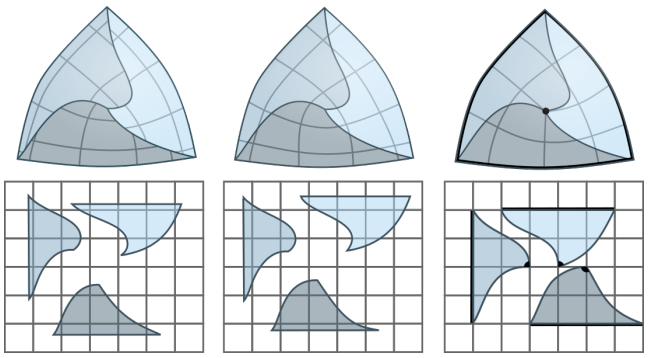


Fig. 2. A general seamless parametrization (left), a seamless parametrization with grid automorphism transitions (center), and a quantized seamless parametrization (or integer-grid map) (right) that in addition demands integer-alignment of boundaries and singularities. Note that extracting a mesh from the latter is a simple matter of determining the pre-image of the integer grid under the map.

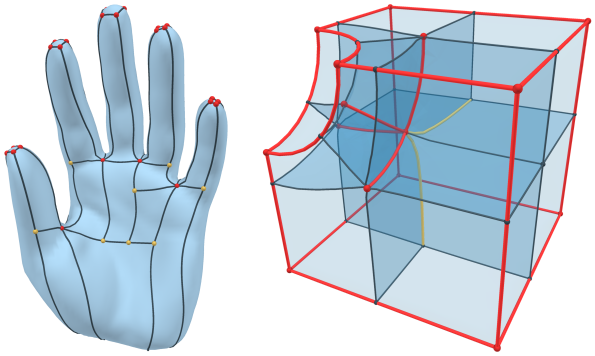


Fig. 3. Structured aligned decomposition of a surface by a motorcycle graph (left) and a volume by a motorcycle complex (right). Singularities (red) are points in the surface case, but form a network of curves in the volumetric case. T-junctions (left: points; right: curves) are marked in yellow.

We demonstrate how an integer-grid map subject to these integers can then be generated using established parametrization methods. Pursuant to one of the above mentioned gaps in the volumetric context (gap ②), existing parametrization methods provide no guarantee yet to always find a map that is locally injective even if it exists; importantly, our quantization result is guaranteed to permit one, and is therefore readily usable in conjunction with future advances on that field.

The general suitability of the motorcycle complex for the quantization task has been briefly demonstrated by [Brückler et al. 2022]. The main challenges, in particular dealing with non-positive quantization values, have been circumvented for simplicity in their proof of concept, though. Our result shows that the MC is suitable as the basis for a full-fledged solution, with equal or even higher flexibility than known from reliable solutions in the 2D case.

2 RELATED WORK

2D Integer-Grid Maps. The idea of parametrization-based semi-structured quadrilateral mesh generation goes back to works such as [Bommes et al. 2009; Kälberer et al. 2007; Tong et al. 2006]. The term integer-grid map was later coined to refer to the class of parametrizations implying boundary-conforming all-quad meshes [Bommes et al. 2013]. It refers to a chart-based map (an atlas) of a surface into $\mathbb{R}^2$ such that the preimage of the integer-grid forms such a mesh. To this end, chart transitions need to be grid automorphisms and the map needs to align the surface boundary and the map's singularities with the grid. Such maps are often generated by initially relaxing the problem (ignoring the integer requirements) and later determining integers based on the relaxed solution [Bommes et al. 2009; Kälberer et al. 2007].

2D Seamless Maps. This relaxed class of maps is referred to as seamless maps. These have found their own interest, for instance to define spline function spaces [Campen and Zorin 2017; Marinov et al. 2019], and their reliable generation has been investigated in dedicated works [Campen et al. 2019; Levi 2021, 2022; Myles et al. 2014; Zhou et al. 2020]. We formally define these various classes of maps in Sec. 3.2.

2D Quantization. The essential gap between a seamless map and an integer-grid map is the choice of a number of integers (related to singularities, boundaries, and transitions). This key problem was initially addressed by [Bommes et al. 2009, 2010; Kälberer et al. 2007]. The basic proposal is to first solve the relaxed (seamless) problem, then *round* the relevant variables to nearest integers (and finally resolve with these fixed). They also investigated incremental versions of this rounding strategy, with variations regarding order and batching. Unfortunately, this rounding approach is fragile and can lead to conflicts, implying map degeneration, as illustrated in [Bommes et al. 2013, Fig. 1] and [Campen et al. 2015, Fig. 3]. We expand on the underlying reason in Sec. 3.3.

More recent work has found ways to perform this task, also referred to as quantization, in a reliable manner. The method of [Bommes et al. 2013] effectively explores part of the space of integer assignments until a valid one (or the optimal one within that part) is found; the search is delegated to a general-purpose mixed-integer solver. A major step in efficiency was made by instead expressing the integer choice on top of a structured coarse decomposition of the surface [Campen et al. 2015; Couplet et al. 2021; Lyon et al. 2019, 2021a,b]. The motorcycle graph [Eppstein and Erickson 1999; Eppstein et al. 2008] is employed to obtain such a partition, as it offers beneficial properties regarding map-alignment and parsimony.

3D Integer-Grid Maps. The notion of integer-grid map generalizes from the 2D case to arbitrary higher dimensions. For the 3D case it was first spelled out explicitly by [Nieser et al. 2011]. The most significant difference to the 2D case is related to the possible singularities of such maps—in analogy to the more complex space of irregularities that can occur in a hex mesh compared to a quad mesh. This is discussed in detail by [Liu et al. 2018].

3D Quantization. In the field of volumetric parametrization-based meshing, the fragile method of rounding, as described by [Nieser et al. 2011] and slightly varied by [Jiang et al. 2014; Li et al. 2012], has remained the state-of-the-art. The above mentioned advances in terms of quantization robustness did not easily generalize to 3D. We show in Sec. 7 that, as could be expected, the rounding strategy is likewise non-robust in the 3D case. Nevertheless, it is still commonly employed, even in recent works in the context of hexahedral meshing [Corman and Crane 2019; Fang et al. 2016; Liu et al. 2018; Palmer et al. 2020; Solomon et al. 2017]—for lack of a better option.

For the restricted class of polycube maps, quantization was dedicatedly considered [Cherchi et al. 2016; Protais et al. 2020]. In this case the problem is simpler, because these maps are free of singularities in the interior as well as, in the setting of these works, free of chart transitions. Also for other restricted problem classes, for instance with user-provided high-level structural volume decomposition, integer selection strategies were discussed [Shepherd 1999].

Recently, the motorcycle graph that, as mentioned above, has helped robustly solving the quantization problem in 2D, was lifted to the 3D volumetric setting, in the form of a motorcycle complex [Brückler et al. 2022], see Fig. 3. The authors demonstrate that it can, in principle, be employed for quantization tasks, and conjecture

that it could serve as basis for a general solution in 3D. We provide such a general solution for the volumetric case.

Frame Fields. Related to but separate from the problem of quantization, is the problem of deciding on a suitable singularity structure for a given 3D domain. The use of optimized 3D frame-fields (in generalization of 2D cross-fields [Vaxman et al. 2016]) to derive such structures has been promising in this context from the start [Huang et al. 2011; Nieser et al. 2011]. A challenge lies in restricting to *meshable* singularity structures. Several techniques have been explored to this end, some relying on additional input, e.g. a metamesh [Nieser et al. 2011], or interaction [Liu et al. 2018]. Others use heuristics to cure invalid singular graphs [Huang et al. 2012; Jiang et al. 2014; Li et al. 2012], albeit without guarantees—leaving gap ① mentioned in Sec. 1. One can witness a continuing effort to advance the state of the art in frame field computation, including singularity graph generation, repair, and optimization [Corman and Crane 2019; Liu et al. 2018; Palmer et al. 2020; Ray et al. 2016; Reberol et al. 2019; Solomon et al. 2017; Viertel et al. 2016; Zhang et al. 2020].

Local Injectivity. Local injectivity is an important property of seamless maps. In the 2D case, the problem of promoting or maintaining local injectivity of maps has been addressed in various contexts [Bommes et al. 2013; Lipman 2012; Rabinovich et al. 2017; Schüller et al. 2013], and constructions with local injectivity guarantees are available by now [Campen et al. 2019; Levi 2021, 2022; Weber and Zorin 2014; Zhou et al. 2020]. For the case of volumetric maps, progress is being made on multiple fronts [Aigerman and Lipman 2013; Campen et al. 2016; Fu et al. 2021; Garanzha et al. 2021], but no guaranteed method is available, leaving gap ② so far.

Other Hex-Meshing Approaches. Methods for the generation of hexahedral meshing that follow other general ideas, not based on parametrization maps, are discussed in various surveys [Blacker 2000; Sarrate Ramos et al. 2014; Shepherd and Johnson 2008; Tautges 2001]. Recent developments include interactive and automatic (dual) decomposition-based techniques [Livesu et al. 2020; Takayama 2019] and improvements of extrinsic grid and octree based methods [Pitzalis et al. 2021]. The most recent survey [Pietroni et al. 2022] treats both, parametrization-based methods and other approaches, and discusses their relative advantages; while parametrization-based methods are associated with high flexibility and quality, in particular a higher level of robustness of some other (grid or polycube based) approaches is pointed out—an aspect that is, in part, addressed by our work. Remotely related are methods for hex-remeshing or simplification, such as [Gao et al. 2015a, 2017].

3 BACKGROUND

We consider the setting that the input domain is represented by a tetrahedral mesh M. Its elements are called tets, facets, edges, and vertices. For a facet f between tets t_1, t_2 the tuples (f, t_1) and (f, t_2) are called the half-facets of f; they can be understood as directed versions of f.

3.1 Volumetric Seamless Parametrization

A parametrization (or map) $\phi : M \to \mathbb{R}^3$ is represented in a piecewise linear manner, i.e. $\phi|_t$, the restriction of ϕ to any individual tetrahedron (tet) t, is affine. We are going to operate with maps that need not be continuous across the triangular faces (between tets), hence the map is generally expressed by $\boldsymbol{u} = (u, v, w)$-parameters associated with tet corners rather than vertices. Alternatively, we can consider a mesh M^c obtained by cutting M along all triangles with discontinuities, effectively duplicating vertices along the cut. On M^c, the map can be expressed using vertex-associated $\boldsymbol{u}$-parameters. Let U denote the set of all these individual per-vertex u, v, and w parameters.

Unless explicitly stated otherwise, we assume maps to be valid in the following, in the sense that each $\phi|_t$ is orientation-preserving and non-degenerate, i.e. for each tet's (constant) Jacobian J_t of $\phi|_t$:

$$\det J_t > 0 \tag{1}$$

Definition 1 (Transition). *For two half-facets h_1, h_2 of a facet, the function $\tau_{h_1} = \phi|_{h_2} \circ \phi|_{h_1}^{-1}$, i.e. such that $\tau_{h_1} \circ \phi|_{h_1} = \phi|_{h_2}$, is called the transition of h_1.*

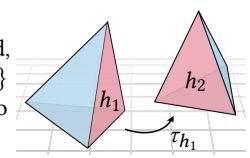
Notice that opposite half-facets of the same facet have inverse transitions, $\tau_{h_2} = \tau_{h_1}^{-1}$.

Definition 2 (Volumetric Seamless Map). *We say a map ϕ is seamless if each transition τ is of the form*

$$\tau\boldsymbol{u} = R_k\boldsymbol{u} + d, \quad \boldsymbol{u} = (u, v, w),\ R_k \in Oct,\ d \in \mathbb{R}^3, \tag{2}$$

where Oct is the octahedral group, the group of 24 rotations which preserve the octahedron (or the unit cube).

In other words, the transitions are rigid, and their rotational part $R_k, k \in \{1, \ldots, 24\}$ [Nieser et al. 2011], is merely allowed to permute the (signed) parametric axes.



Alignment. A facet (or edge) is called *aligned* if its image under ϕ is parallel to one coordinate plane (or axis respectively). A map is *boundary-aligned* if each facet on the boundary of M is aligned. A facet f (or edge e) is u-aligned, if $\phi(f)$ (or $\phi(e)$) is constant in u-direction; similar for v and w. Note that an aligned edge is aligned in two of these dimensions.

Singularities. Edges can be categorized by the sum of parametric dihedral tet angles incident on them. In a seamless map it necessarily is an integer multiple of $\pi/2$. Interior edges with an angle different from 2π and boundary edges with an angle different from π are *singular*, all other edges are *regular*.

Boundary-alignment and singularity-alignment, being important prerequisites in mesh generation, are generally assumed in the following.

Remark: Approaches for the generation of such parametrizations are often based on a two-stage principle: first decide on rotations R_k (which largely define the singularity structure), then obtain the map as the solution to some distortion or alignment optimization problem. The variables in the latter are the parameters $\boldsymbol{u}$ of each vertex $v \in M^c$, and the translational components d are implied.

3.2 Volumetric Integer-Grid Map

Let the subset $U_B \subseteq U$ (of per-vertex parameters on M^c) denote the set of *boundary parameters*; it contains a parameter w if its vertex is on a w-aligned boundary facet (analogously for u and v). The subset $U_S \subseteq U$ of *singularity parameters* contains a parameter w if its vertex is on a w-aligned singular edge (analogously for u and v).

Definition 3 (Volumetric Integer-Grid Map). *A map ϕ is a volumetric integer-grid map [Liu et al. 2018] if it is seamless and*

- *for each transition τ: translation $d \in \mathbb{Z}^3$,* (3)
- *for each boundary parameter $z \in U_B$: $z \in \mathbb{Z}$,* (4)
- *for each singularity parameter $z \in U_S$: $z \in \mathbb{Z}$.* (5)

In other words, an integer-grid map is a seamless map that additionally respects the integer grid, aligning the grid across discontinuities (the transitions are *grid automorphisms*), and aligning boundaries and singularities with the grid (Fig. 2). This kind of parametrization directly induces a hex mesh, obtainable by determining the inverse image of the parametric integer grid under ϕ [Lyon et al. 2016; Nieser et al. 2011]. Integer iso-surfaces of the parametrization consequently correspond to sheets of quads in the resulting hexahedral mesh while singularities of the parametrization manifest themselves as irregularities in the mesh.

Equivalence. We call two integer-grid maps equivalent if they imply the same mesh. Note that modifying a map by applying (per tet) an arbitrary integer translation yields an equivalent map, as the grid pre-image is invariant to such translations. This means that $n_T - 1$ of the n_F (three-dimensional) transition translation degrees of freedom d, where n_T is the number of tets and n_F the number of interior facets, are irrelevant under equivalence. For instance, for each facet f crossed by a spanning tree of the tet adjacency graph (a total of $n_T - 1$), we may fix $d_f = 0$ without restricting the space of representable meshes [Li et al. 2012].

3.3 Integer Feasibility

The remaining $n_F - n_T + 1$ translations, as well as the singularity and boundary related integer degrees of freedom are not independent, but related by the involved conditions (seamlessness and alignment). Overall, the number of independent integers depends on the complexity of the singularity network and the topological complexity of M, not on the number of elements in the mesh M. We exploit this in Sec. 6.1.

While independent with respect to the linear seamlessness, boundary alignment, and singularity alignment conditions, values of these integers cannot actually occur in arbitrary combinations in integer-grid maps. The reason is the additional (nonlinear) validity condition (1). In other words, for each consistent choice of integers, maps exist that are (possibly invalid) integer-grid maps, but not necessarily any valid integer-grid map exists. We call a choice of integer values *infeasible* if it does not permit a valid map (even under mesh refinement).

This is the key limitation of existing rounding-based methods to decide on the integer values [Jiang et al. 2014; Li et al. 2012; Nieser et al. 2011] in the context of integer-grid map generation; they take into account the simple linear dependencies, but ignore the intricate validity condition (1). They may therefore yield infeasible integers, and especially for coarse target grid resolutions they often do—for a sufficiently coarse target resolution they even necessarily will. This explains why example meshes shown in pertinent articles are sometimes of rather dense, high resolution. The simplest example of a conflict is the choice of coincident integer parameters for two spatially distinct singular or boundary vertices. This implies a parametric distance of zero in any compatible map, such that (1) cannot hold along some path between the two vertices.

The integers produced by our method, by contrast, are guaranteed to be feasible.

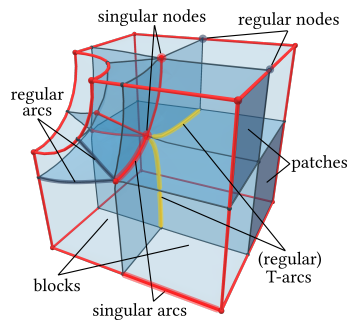


Fig. 4. Elements of a volume-T-mesh, e.g., a motorcycle complex.

3.4 Volume-T-Mesh

In the context at hand, the term *T-mesh* is used in the 2D case for a partition of a seamlessly parametrized surface into patches such that (i) each patch is four-sided, (ii) each patch interior is regular (free of singularities), and (iii) each patch boundary is aligned with the parametrization (i.e. piecewise lies on an isoline). A mesh representation of this partition in general is non-conforming, with T-joints, hence the name T-mesh. Note that this non-conformity allows the partition to be relatively simple, while exhibiting the above properties. Fig. 3 left shows an example.

The 3D analogue of this we refer to as volume-T-mesh. It forms a partition of a seamlessly parametrized volume M into blocks such that (i) each block is cuboidal, (ii) each block interior is regular, and (iii) each block boundary is aligned with the parametrization (i.e. piecewise lies on an isosurface). We refer to its elements as blocks, patches, arcs, and nodes (in order of decreasing dimension), as illustrated in Fig. 4. Singular arcs are those coincident with singularities, singular nodes are nodes with 1 or more than 2 incident singular arcs.

Analogously to the use of a T-mesh for quantization representation in 2D [Campen et al. 2015; Lyon et al. 2019, 2021a,b], we use a volume-T-mesh for this purpose in 3D. Concretely, we employ the following construction method.

Motorcycle Complex. The motorcycle complex [Brückler et al. 2022] is a generalization of the motorcycle graph [Eppstein et al. 2008], and yields a volume-T-mesh of the above type. While not yielding a minimal result, its aim is to produce a simple partition with few blocks. Conceptually speaking, isosurfaces are followed, starting from all singular edges, up to some termination condition. These

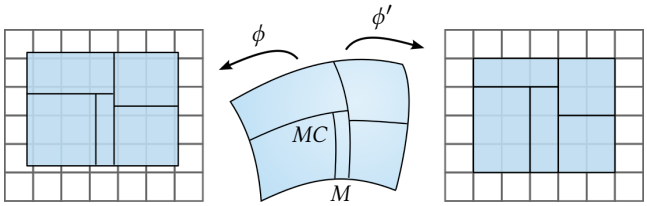


Fig. 5. 2D illustration of remapping a T-mesh decomposition to integer extents. The seamless map ϕ is conceptually transformed to achieve alignment with the integer grid.

partition the volume M and ultimately form the bounding patches of the implied blocks. For details we refer to the original paper.

With respect to the input parametrization, arcs are axis-aligned, patches are rectangular and blocks are cuboidal as well as fully regular in their interior. The singular graph of the parametrization coincides with arcs and nodes, while boundaries of M coincide with patches. Due to the non-conforming nature of the partition, patches may be bounded by more than four arcs and blocks may be bounded by more than six patches—due to (three-dimensional) T-joints.

Base Complex. Alternatively, consider an analogous but conforming partition of the volume, such as the base complex [Gao et al. 2015a]. This could also be used as basis for our method, but as shown by [Brückler et al. 2022], it is more expensive to construct, and in particular leaves less degrees of freedom for the quantization purpose compared to a coarser, non-conforming partition, unnecessarily restricting the space of achievable integer-grid maps.

4 APPROACH OVERVIEW

The classical rounding approach to (volumetric) integer-grid generation proceeds as follows: First, compute a seamless parametrization ϕ, i.e. an integer-grid map with all $\mathbb{Z}$-variables relaxed to $\mathbb{R}$. Then, round these variables' values to $\mathbb{Z}$. Finally, recompute a seamless parametrization ϕ' with these variables fixed accordingly. Optionally, rounding and recomputation may happen incrementally. The latter step inevitably fails if the rounded values are infeasible.

Instead of rounding, we consider a volume-T-mesh on the seamless parametrization. Imagine rescaling (or more generally transforming) the parametrization per block of this volume-T-mesh, in such a way that it remains aligned with the block boundary, the block assumes integer extents in each of the three parametric dimensions, and patches have integer extents as well. Up to a global translation, the result would be an integer-grid map, one that forms some $m{\times}n{\times}o$ grid of hexahedra per block, cf. Fig. 5. Our quantization method is based on this intuition, but avoids the explicit execution of this (expensive) process. Instead, we determine the choice of integer values that such a transformation would imply, while keeping it implicit. The knowledge that a corresponding transformation does exist, certifies the feasibility of the resulting quantization, i.e. the resulting integer values.

We adapt the idea of a differential quantization representation via a T-mesh [Campen et al. 2015] to our setting. Concretely, we represent a quantization by assigning integers to the arcs of a volume-T-mesh—conceptually expressing their parametric extent in the above imagined transformation.

Algorithmically, the following steps are performed:

Volume-T-Mesh Construction First, construct the motorcycle complex (MC) of the input seamless parametrization, a non-conforming partition into parametrically cuboidal blocks.

Quantization Computation Via a series of integer quadratic programs (IQP), assign integer lengths to the arcs of the MC. It aims to reproduce original scales, subject to linear constraints that keep patches rectangular and blocks cuboidal, and linear constraints that forbid degeneration.

Quantization Integration Summation of per-arc integers along certain paths in the MC, starting from a root node, yields the desired integer values, suitable for a volumetric integer-grid map.

Finally, we demonstrate the use of the resulting quantization values for the explicit construction of integer-grid maps:

Reparametrization Recompute or transform the parametrization to adopt the integer dimensions implied by the quantization, yielding an integer-grid map.

A procedure of this kind has also been outlined in [Brückler et al. 2022], with a focus on the first step. The other steps are addressed briefly, restricted to a simplified setting, limited in flexibility. We demonstrate the detrimental effect of this restriction in more detail in Sec. 7. In particular, we demonstrate that restricting to positive integers per arc—while making degeneration prevention trivial—cuts down the space of representable quantizations significantly, excluding practically important options. We allow zeros and even negative values. This requires additional attention in preventing degeneration, as detailed in Sec. 5, and in return enables higher fidelity.

Ultimately, applying our method to the generation of hexahedral meshes, we demonstrate that it allows for more robust mesh generation than previous, rounding-based approaches, and for more structurally flexible and geometrically apt mesh generation than the simple restricted strategy.

5 MOTORCYCLE COMPLEX QUANTIZATION

Given a valid seamless parametrization ϕ on M, let $MC = \{N, A, P, B\}$ be its motorcycle complex, composed from nodes N, arcs A, patches P and blocks B, cf. Fig. 4. Let singular arcs be denoted by $A^* \subseteq A$, singular nodes by $N^* \subseteq N$ and boundary patches by $\bar{P} \subseteq P$.

Definition 4 (Quantization). *We call MC quantization, or just quantization for short, a map*

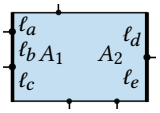
$$q\colon A \to \mathbb{Z}, \quad a \mapsto \ell_a \tag{6}$$

that assigns integer lengths to each arc of the MC.

A quantization is *consistent* if

$$\sum_{a \in A_1} \ell_a = \sum_{a \in A_2} \ell_a \tag{7}$$

for each pair A_1, A_2 of arc sequences (both consisting of one or more arcs) lining two opposite sides of some patch $p \in P$ of MC. This means each patch is rectangular under quantization q.



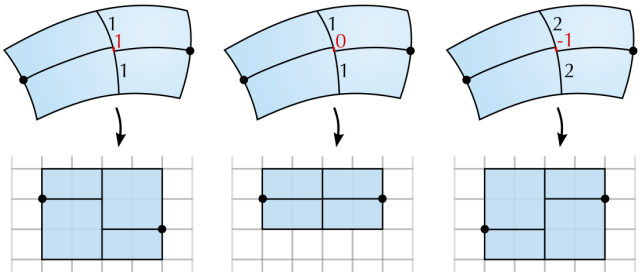


Fig. 6. 2D illustration of the effect of $\mathbb{Z}^{>0}$ vs $\mathbb{Z}^{\geqslant 0}$ vs $\mathbb{Z}$. In this example, two singularities (dots) are slightly offset in the input parametrization (top). For $\ell_a \in \mathbb{Z}^{>0}$, this misalignment is necessarily preserved, and even increased to at least 1 (left). Allowing $\ell_a \in \mathbb{Z}^{\geqslant 0}$ (center), the singularities can be aligned. Allowing negative values, $\ell_a \in \mathbb{Z}$ (right), even their relative order may be changed, in case this is beneficial for overall quantization quality.

Note that this definition is more general than in previous works (whether in 2D or 3D), where q maps into $\mathbb{Z}^{>0}$ or $\mathbb{Z}^{\geqslant 0}$. Fig. 6 illustrates the additional flexibility.

Such an MC quantization fully defines an integer assignment for all the integer degrees of freedom of an integer-grid map that adopts the singularity structure of ϕ. For this assignment to be feasible (note that, e.g., $q \equiv 0$ is consistent but not feasible), the quantization needs to satisfy additional validity criteria, spelled out in Sec. 5.1.

Local q-Charts. For any simply connected regular subset of MC (not containing any internal singularity), a quantization q implies an image in $\mathbb{R}^3$ in which each block is a cuboid of the size specified by its arcs values ℓ_a (see Fig. 7 left). This image is unique (up to a global rigid transformation): Map one node to $(0, 0, 0)$, one incident arc such that it aligns with the u-direction, and another adjacent arc such that it aligns with the v-direction. The remainder is implied by the arc lengths ℓ_a of q. In the following we will often argue about paths of arcs, and about the relative orientation of a path's arcs and the relative position of nodes. This is generally to be understood as being meant in such a coherent local parametric coordinate system (a q-chart). As only *relative* aspects are considered, the above mentioned rigid transformation is irrelevant, and the notion of arcs of a path being parallel or perpendicular is well-defined. We call a path a zero-path if its endpoints have distance zero in a q-chart.

5.1 Validity

First of all, we require that blocks have non-negative extent, i.e.

$$\sum_{a \in A_1} \ell_a \geqslant 0 \tag{8}$$

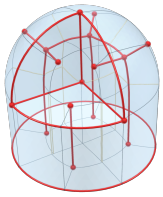
for each sequence A_1 of arcs forming one of the twelve edges of a block. Note that this does not require each individual arc to have a non-negative value (cf. Fig. 6 right).

Let us briefly discuss how a block with an assigned extent of zero (in one or multiple dimensions) fits the per-block rescaling intuition mentioned in Sec. 4. Of course, mapping a block (i.e. a piece of M) onto a parameter cuboid of zero volume would imply a degenerate map (Fig. 7 left). We can, however, consider the problem more globally. Instead of assuming that each block individually maps to a respective parameter cuboid, assume that a zero-block borrows a little parametric space from its surrounding nonzero-blocks, which shrink their parametric footprint accordingly, as illustrated in Fig. 7 center. More generally, a connected component of multiple zero-blocks can be assumed to collectively borrow from the surrounding nonzero-blocks.

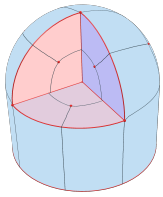
Inversely, rather than in parametric space, this can be viewed in actual 3D space as shrinking zero-blocks to zero volume in M, distributing their volume to neighboring nonzero-blocks that expand accordingly (Fig. 7 right). Mapping the resulting degenerate blocks onto degenerate parameter cuboids then is perfectly compatible with an injective map. We remark that in the work of [Lyon et al. 2019] on 2D quantization this principle is actually employed explicitly, performing collapses on the surface T-mesh.

In this sense, zero-blocks do not present a problem—as long as they, or rather each of their connected components, can be contracted (along the required dimensions). This can be impossible if a component is topologically non-contractible (not of ball topology, e.g., winding around a handle of M) or if it involves critical entities, as defined in the following.

Definition 5 (Singular Link). *A maximal sequence of one or more singular arcs connected via regular nodes is a singular link. Note that it may connect two singular nodes, form a loop rooted at one singular node, or form a cycle. We denote the set of singular links as L.*



Definition 6 (Boundary Region). *A maximal set of one or more boundary patches connected via regular boundary arcs (black) is a boundary region. Note that it can be a closed surface or be bounded by singular boundary arcs. We denote the set of boundary regions as R.*



The set of *critical entities* of the MC is defined as $C = L \cup R$. As these critical entities have a fixed spatial location in M, we must

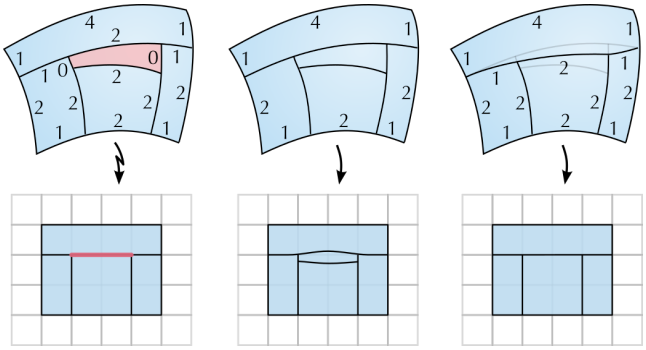


Fig. 7. Mental picture (in 2D) for dealing with a zero-block (red). Left: Directly mapping block-by-block according to the quantization would imply a non-injective map. Center: Expanding the image of the zero-block, borrowing parametric space from neighbors. Right: Equivalently, this can be viewed as collapsing the zero-block on M; the overall map is the same.

respect them in the imagined zero-block shrinking process. We may shrink *onto* them (or along/within them), but must not move them. A connected component of zero-blocks may therefore be non-collapsible due to involving multiple critical entities, or due to containing one entirely. Fig. 8 illustrates this. In the following we describe how to preclude such quantizations—and thereby address the main challenge of the problem at hand.

5.2 Strategy

We formulate the problem of finding a quantization q that is consistent and valid as an integer quadratic program (IQP). Its objective aims to reproduce the parametric extents observed in the input seamless parametrization:

$$\sum_{a \in A} (\ell_a - \|\phi(a)\|)^2 \rightarrow \min, \tag{9}$$

where $\|\phi(a)\|$ is the original parametric length of arc a. For consistency, we add linear equality constraints (7). For validity, we add linear inequalities (8) to preclude negative blocks and, in a lazy manner, further linear inequalities (Sec. 5.3) that preclude non-contractible situations, like critical entity conflicts.

We first solve the IQP without the lazy constraints. The result is then checked for violation of any of these (without the need to explicitly formulate them). A subset of those that are violated are then made explicit and added to the IQP, which is then resolved (with warm start). This is iterated if necessary.

This lazy approach allows for an efficient processing. In particular, in cases that are conflict-free right away, a single solve (with a low number of inequalities) is sufficient. Only in harder cases (roughly those that the classical rounding approach would likely fail on) additional effort (setting up further constraints, resolving the extended IQP) needs to be spent. Due to the careful way we select the lazy constraints, the number of iterations needed is small even in extreme cases, see Sec. 7.1.

5.3 Validity Constraints

Pursuant to the discussion in Sec. 5.1, the following situations must be prevented. In these, not all zero-blocks could be collapsed without moving critical entities in M or breaking the topology of M.

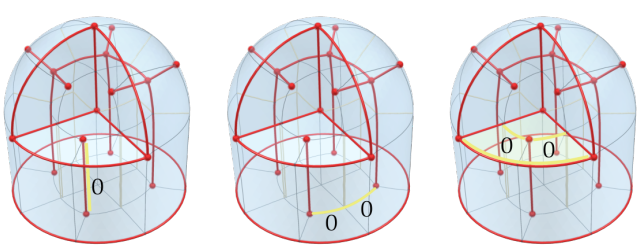


Fig. 8. Invalid quantizations on a volume-T-mesh of an example shape. Singular links and singular nodes are marked in red, regular arcs are light grey. Left: A singular link (highlighted in yellow) would collapse under the indicated local quantization. Center: Two singular arcs would collapse onto each other along the yellow path. Right: A boundary region collapses if the yellow arcs have zero values.

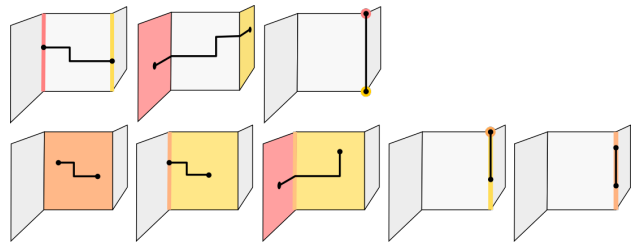


Fig. 9. Examples of paths between two points on critical entities. Shown are three square boundary regions with in-between vertical singular links. The confinement zone D_1 of the start point is marked in red, zone D_2 of the end point in yellow, their intersection in orange. Assume the shown paths are zero-paths under q. Then the top row shows violating paths, while those in the bottom row, lying in $D_1 \cup D_2$, are not violating.

i) A singular link (corresponding to a 1-manifold[1] curve in M) must not collapse to a point.
ii) Two points on critical entities must not collapse, except *within* each of their containing critical entities.
iii) A boundary region (corresponding to a 2-manifold[1] surface in M) must not collapse to a point or curve.
iv) Cycles that are topologically non-contractible within M must not collapse.

Given a consistent but possibly invalid quantization (from the first solve of the above IQP), we can check for violations of these conditions, and add constraints that will prevent them in the next solve iteration. This is detailed in the following for the above four types of conditions.

5.3.1 Type (i) Conditions. A singular link $l \in L$ would collapse entirely iff all its arcs have length zero under the quantization. To prevent the collapse, we simply need to require

$$\forall l \in L: \quad \sum_{a \in l} \ell_a > 0. \tag{10}$$

5.3.2 Type (ii) Conditions. To detect a violation of this condition, we search for a concrete path that is evidence of this. Let p_i be a critical point, i.e. a point contained in one or more critical entities $C_i = \{c_i^0, \dots\}$. Let $D_i = \cap_k c_i^k$ denote the intersection of these critical entities; this may be a boundary region, a singular link, or a singular node. We search for a path ρ between any two critical points p_1, p_2, that is a zero-path and not contained (under q) in the union $D_1 \cup D_2$, see Fig. 9.

The rationale for this is as follows: A zero-path between two critical points is evidence that they will become coincident in the zero-block collapsing process mentioned above. This can be validly possible or not. D_i is the zone within which p_i may conceptually be moved in the collapse process without it leaving the critical entities it lies on. If the points need to be moved only within their respective D_i, the collapse is validly possible. Therefore only zero-paths ρ not contained in $D_1 \cup D_2$ are violating. Containment here is meant with respect to q; a path may well leave this region in MC and return, but still be a distance of zero away all along according to q, thus be non-violating.

[1] with or without boundary

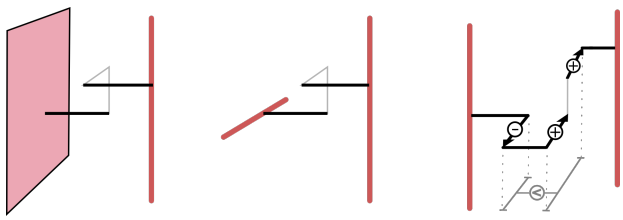


Fig. 10. Examples of arc paths ρ in M between various critical entities. Left: Boundary region and singular link. Center: Two (parametrically) perpendicular singular links. Right: two (parametrically) parallel singular links. Arcs that are in $\rho^{\perp}$ are shown bold. Their sign in constraint (11) is positive by default, except for arc sets that are dominated by longer, oppositely oriented arc sets, as indicated on the right.

In Sec. 5.4 we describe how to discover such critical paths. For a found critical path ρ between two critical entities c_i, c_j, we then add a constraint as follows to prevent the violation in the next iteration. Let $\rho^{\perp} \subseteq \rho$ be the subset of arcs that are perpendicular to both c_i and c_j (in a common local q-chart along ρ). Note that if c_i and c_j are parallel singular links, $\rho^{\perp}$ may contain arcs aligned with two axes; in case of perpendicular singular links or if boundary regions are involved, only one axis is involved, see Fig. 10. The following constraint ensures separation of the critical entities in the next iteration, by enforcing a non-zero parametric distance of the two critical points in direction perpendicular to the critical entities:

$$\sum_{a \in \rho^{\perp}} \operatorname{sign}(a, \rho) \ell_a > 0. \tag{11}$$

The restriction to the perpendicular direction(s) is important, because a distance in parallel direction would just shift the critical entities in parameter space along each other; the coincidence would merely move to another pair of points on the same entities.

The coefficient $\operatorname{sign}(a, \rho) \in \{-1, 1\}$ is chosen positive if a is of *dominant* orientation in ρ, negative otherwise. An arc is of dominant orientation if the total length (under ϕ) of (directed) arcs in ρ pointing in equal direction is greater than the total length of arcs pointing in opposite direction, see Fig. 10 right. In case of equality, one orientation is arbitrarily chosen as dominant.

Notice that this constraint flexibly permits separation in either of the two perpendicular dimensions in the case of two parallel singular links, as in Fig. 10 right—the solution space is not constrained by fixing a separation dimension. Furthermore, the above sign definition permits preserving the original relative parametric location of the two singularities; the opposite choice would enforce "switching places" in the respective separation dimension.

5.3.3 Type (iii) Conditions. Note that the boundary of boundary regions is formed by singular links, such that the above constraints (i)+(ii) already to a large extent prevent the collapse of a boundary region under the quantization. Also, a closed boundary region of genus 0 necessarily has incident singular arcs (implied by Gauss-Bonnet) whose associated constraints prevent its collapse. In the general case of a topologically non-trivial surface region, we can compute a set of homotopy generator cycles γ_i of the region [Erickson and Whittlesey 2005], within the contained arcs of MC, and require for each (again assuming a common local q-chart):

$$\forall k \in \{u, v, w\} : \quad \sum_{a \in \gamma_i^k} \operatorname{sign}(a, \gamma_i) \ell_a > 0, \tag{12}$$

where $\operatorname{sign}(a, \gamma)$ is defined as in the above, and $\gamma_i^k \subseteq \gamma_i$ contains the arcs aligned with parameter direction k. If the cycle is degenerate for some dimension k in the (valid) input parametrization, i.e. $\sum_{a \in \gamma_i^k} \operatorname{sign}(a, \gamma_i) \|\phi(a)\| = 0$, the respective constraint can be dropped; the input parametrization is witness that in this dimension separation is not necessary.

With these constraints, effectively the collapse of entire topological handles, along either parametric direction involved in the cycle, is prevented. We remark that these constraints are conservative, as, depending on the configuration, separation in one (rather than two or three) parametric directions along a cycle can be sufficient. However, as these constraints turn out to be practically irrelevant (see Sec. 5.3.5), further discrimination is of little use.

5.3.4 Type (iv) Conditions. In order to prevent a collapse along some non-contractible cycle, for a set of volumetric homotopy generator cycles γ_i of M (which can be computed on the cell complex formed by MC using the algorithm of [Kim et al. 2008]) we set up a constraint of the above type (12).

5.3.5 Lazy Strategy. Initially, we use constraints (i); these are inexpensive to set up, so is sensible to add all of them to the IQP from the start. The rest is added lazily. Concretely, in subsequent iterations we first add constraints (ii), as long as any violation is detected. Finally, constraints (iii) and (iv) are added. These latter constraints actually were relevant only for one contrived example during our investigation, one without any singularities—a rare situation only possible on a topological hollow torus (conceptually obtained by gluing a box along two pairs of opposite sides). We conjecture that constraints (i)+(ii) are generally sufficient except for this single completely regular special case—which only has three integer degrees of freedom and could therefore easily be handled specially.

5.4 Discovering Critical Paths

The set of critical points is infinite. We therefore consider entire critical entities (i.e. curves and surfaces) at once, and determine whether a zero-path exists between any two points on any pair of these.

Searching the Relevant Space. To this end, for each critical entity $c \in C$ for each parametric direction d perpendicular to it, we construct a spanning forest $T(c, d)$ in the graph of arcs A, rooted at the nodes of c_i. Note that a boundary region has just one perpendicular direction, while there are k outgoing directions at a singular link of valence k.

Let $R^0(c)$ denote the set of blocks of MC that, under q, have zero distance to c, i.e. those that after collapsing all zero-blocks would intersect c. We can restrict the construction of $T(c, d)$ to $R^0(c)$, as any relevant critical path ρ necessarily is contained in this zone. We furthermore stop the construction as soon as the first critical entity c' is reached in such a way that there is a path ρ in $T(c, d)$ that satisfies the criteria of violation spelled out in Sec. 5.3.2. The rationale is that enforcing parametric separation just between nearest neighbors, due

to the transitive effect of the inequality constraints, often is already sufficient to yield a valid quantization. If not, further violations will be detected and constraints be added in the next iteration.

We construct the forest $T(c, d)$ as a forest of shortest paths in ϕ. Arcs are conquered, starting from all nodes of c, in a Dijkstra-like manner, restricted to arcs on blocks from $R^0(c)$, and never walking in direction $-d$. The resulting spanning forest therefore is formed by (weakly) d-monotone paths. Fig. 11 illustrates such forests. Note that the direction d is well-defined only within a local regular chart; as we stop no later than when the first conflicting singularity is reached, no ambiguities arise. If a non-conflicting singularity is reached (as discussed next), it lies on the boundary of R^0, so the forest also in this case remains contained in a regular subset of R^0.

Checking for a Conflict. When a critical entity is reached, i.e. an arc c' of a singular link or a patch c' of a boundary region, and d is perpendicular also to c', we check for a conflict of c and c'. More precisely: whether any point of c and any point of c' collapse along the discovered path ρ from c to c' in $T(c, d)$ under the current quantization q.

To this end, recall that critical arcs and patches are *aligned*. Hence, their image under ϕ as well as under any parametrization consistent with q is an axis-aligned one- or two-dimensional box (a line segment or rectangle). Let $I(c')$ denote this box of c' in a local q-chart along ρ. We check it for intersection with $I(c)$ in the common chart. If they intersect, the critical entities are in conflict, the criteria of violation are satisfied, so a lazy constraint of type (11) is added for the next iteration. Note that paths contained in $D_1 \cup D_2$ (under q, which would not be violating, cf. Sec. 5.3.2) are not discovered because the forest does not have arcs within c, and the above restriction to weakly d-monotone paths prevents a return to c within R^0.

Fig. 12 sketches the iterative enforcement of separation between initially conflicting entities. Fig. 13 shows a volumetric example.

5.5 Constraint Feasibility

Analogous to the argument in appendix A.2 of [Campen et al. 2015], an assignment of strictly positive integers that are consistent exists: Scaling the rational parametric lengths of arcs in the input parametrization by their lowest common denominator s yields one example.

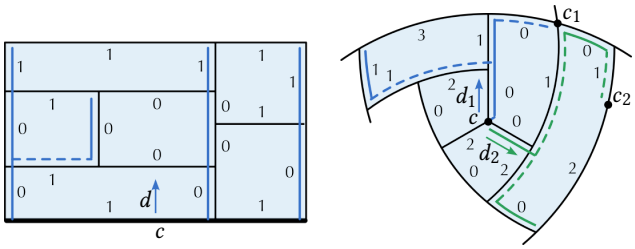


Fig. 11. Forests $T(c, d)$ in $R^0(c)$ in a perpendicular cross section view of (left) a boundary region (bold curve), and (right) a singular arc (central dot). On the right, two forests (in this case trees) for two different directions d_1, d_2 are shown (blue and green). Forest arcs that are not parallel but perpendicular to its defining direction are dashed. On the right, note that the encountered singularity c_1 is not in conflict with c (i.e. $I(c) \cap I(c_1) = \varnothing$), but c_2 is.

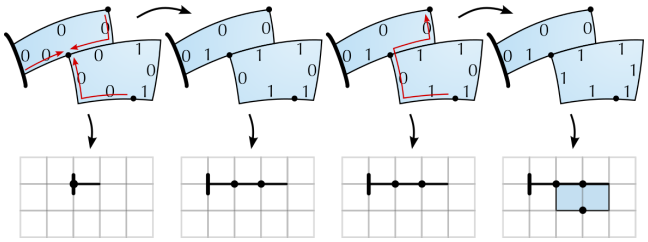


Fig. 12. Shown is an excerpt from a larger MC, together with a quantization q, and an image under q (bottom). Four singular links are involved, one parallel to the viewing plane (bold black curve), three perpendicular (black dots). From an initial, invalid state (left), a valid quantization (right) is derived by, here in two iterations, finding critical paths (red) and "inflating'" along them using constraints (11) in the subsequent solution.

For this assignment q^*, as all arcs have a positive value, all types of lazy constraints are satisfied: For those with possibly negative coefficients, (11) and (12), this may not immediately be obvious; however, note that the positive coefficients are associated with the arcs of dominant orientation (with larger total length in ϕ, thus also in $s\phi$ and under q^*). Therefore, the IQP remains feasible no matter what subset of lazy constraints is added in what order.

5.6 Algorithm Summary

The pseudocode in Algorithm 1 gives a concise summary of the described quantization method as a whole. The scaling factor s in the objective function (equivalent to globally scaling the input seamless parametrization ϕ) is a parameter that allows choosing the targeted grid or mesh resolution. The use of the resulting integers is addressed in the following section.

6 QUANTIZED REPARAMETRIZATION

Our method for the computation of valid quantizations can be used as a drop-in replacement for the rounding procedure in established integer-grid map generation pipelines [Fang et al. 2016; Jiang et al. 2014; Li et al. 2012; Liu et al. 2018; Nieser et al. 2011]. These first compute a seamless parametrization, then decide on integer values in various rounding-based ways, and finally recompute a parametrization (with identical singularity structure) with fixed integer values. The rounding step in the middle is to be replaced by our method.

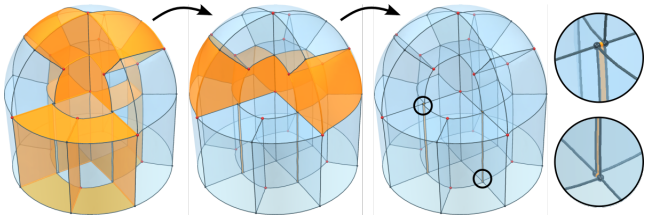


Fig. 13. An invalid quantization evolves due to the addition of constraints in two iterations. Marked in orange are arcs of length zero and patches of area zero under the quantization. In the end, four zero-arcs and two small zero-patches (see blow-ups) remain which cause no further conflicts.

Algorithm 1: Volumetric Quantization

Input: mesh M, valid seamless parametrization ϕ
Output: valid quantization $\ell_a \in \mathbb{Z}$ on MC of M

$MC \leftarrow$ volume-T-mesh(M) `// motorcycle complex`
$\hat{\ell}_a \leftarrow \|\phi(a)\| \in \mathbb{R}, \forall a \in A$ `// consistent input arc lengths`
foreach *patch* $p = \{A_1, A_2, A_3, A_4\} \in P$ **do**
  for $i \in \{1, 2\}$ **do**
    add constraint $\sum_{a \in A_i} \ell_a = \sum_{a \in A_{i+2}} \ell_a$ `// (7) rectangular patches`
∗ **foreach** *block* $b = \{A_1, \ldots, A_{12}\} \in B$ **do**
  for $i \in \{1, 2, 3\}$ **do**
    add constraint $\sum_{a \in A_i} \ell_a \geqslant 0$ `// (8) non-negative blocks`
foreach *singular link* $l \in L$ **do**
  add constraint $\sum_{a \in l} \ell_a > 0$ `// (i) link non-collapse`
repeat
  solve $\sum_{a \in A} (\ell_a - s\hat{\ell}_a)^2 \rightarrow \min,\ \ell_a \in \mathbb{Z},\ s.t.$ constraints
  foreach *critical entity* c **do** `// (ii) critical separation`
    foreach *direction* d **do**
      lazily build d-monotonic forest $T(c, d)$ in $R^0(c)$
      foreach *critical entity* $c' \perp d$ *reached* **do**
        $\rho \leftarrow$ path from c to c' in $T(c, d)$
        if $\tau_\rho I(c) \cap I(c') \neq \emptyset$ **then**
          add constraint $\sum_{a \in \rho^\perp} \mathrm{sign}(a, \rho)\ell_a > 0$
          break
  if *no new constraints were added* **then**
    $\Gamma \leftarrow$ generator cycles for boundary regions and volume
    foreach $\gamma \in \Gamma$ **do** `// (iii)+(iv) topology preservation`
      for $k \in \{u, v, w\}$ **do**
        if $\sum_{a \in \gamma_i^k} sign(a, \gamma_i)\hat{\ell}_a \neq 0$ **then**
          if $\sum_{a \in \gamma_i^k} sign(a, \gamma_i)\ell_a = 0$ **then**
            add constraint $\sum_{a \in \gamma_i^k} \mathrm{sign}(a, \gamma_i)\ell_a > 0$
until *no new constraints were added*
return *integers* ℓ_a *on* MC

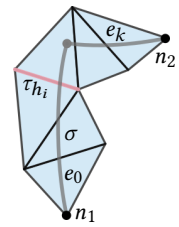
Due to compatible input, the only interfacing question to be addressed is how the integers computed by our method (expressed on arcs of a volume-T-mesh) can be fed into the final constrained reparametrization step. This is not trivial because of the presence of transitions. In a transition-free setting, we could just "integrate" the quantization values along arcs, starting from some node fixed to the origin, so as to yield integer coordinates for all other nodes. The integer values to be prescribed for boundaries (4) and singularities (5) in the final parametrization computation could then easily be read off from contained nodes.

In a general setting, we could still perform the above integration within a simply-connected global chart of MC (self-adjacent via transitions across patches). However, the obtained integers would only make sense for this particular choice of chart layout, and imposing the same on the tetrahedral mesh M for the final parametrization can be challenging because interior facets of MC do not necessarily coincide with facets of M or because the chart layout of the seamless parametrization from the first step is to be adopted (e.g. because a guiding frame field is represented in it).

6.1 Differential Constraints

These challenges can be circumvented by constraining differences rather than absolute integer values, in a manner that takes transitions into account.

For a directed arc path σ, starting at node n_1 and ending at node n_2, let $E_\sigma = \{e_0, \ldots, e_k\}$ denote a sequence of facet-adjacent tetrahedra of M such that it contains σ, e_0 contains n_1, and e_k contains n_2—as illustrated on the right in a 2D sketch. Let $H_\sigma = \{h_1, \ldots, h_k\}$ be the sequence of half-facets $h_i = (f_i, e_{i-1})$ in between, where f_i is the facet between e_{i-1} and e_i. Let $\tau_\sigma = \tau_{h_k} \circ \cdots \circ \tau_{h_0}$ denote the composed transition along this path.



Assume nodes n_1, n_2 lie on vertices v_1, v_2 of M^c. Let $\boldsymbol{r} \in \mathbb{Z}^3$ be the vectorial sum of axis-aligned arc vectors of σ, i.e. vectors of length ℓ_a and direction according to the arcs direction in the coordinate system of e_k. Using the following condition, we can constrain the parametrization to have the spacing between v_1 and v_2 that is intended by q:

$$\boldsymbol{u}_{v_2} - \tau_\sigma \boldsymbol{u}_{v_1} = \boldsymbol{r}. \tag{13}$$

Note that the transition τ_σ is not a constant but contains (translation) variables. In case a node does not lie on a vertex, we instead use a linear combination of up to four vertices (of a containing tet, using barycentric coordinates) to express the constraint—though this is rarely necessary as we will see in the following.

Choice of Constraint Paths. Setting up such a constraint for each individual arc of MC, and recomputing a seamless parametrization subject to these constraints, would then obviously yield an integer-grid map—up to a global translational degree of freedom.

This set of constraints is unnecessarily restrictive (constraining more than just the integer degrees of freedom, also the positions of points that coincide with regular nodes) as well as redundant, though. We can reduce it to a smaller set that only constrains the integer degrees of freedom (3)-(5).

First of all, if the paths of a set of such distance constraints form a contractible cycle, one of them is redundant (implied by the combination of the others together with (7)). Therefore, instead of all arcs, it suffices to consider a spanning tree, together with (in case M is not of ball topology) a generating set of non-contractible arc cycles—computable using the algorithm of [Kim et al. 2008].

Furthermore, there is no need to constrain nodes that do not lie on a singularity or the boundary, as regular interior nodes are not related to the integer degrees of freedom. Also, constraining multiple nodes per singular link or per boundary region is redundant. Therefore, we compute a set of arc paths σ_i that form a spanning tree of the critical entities, augmented by a set of cyclic generator paths. The root for the tree as well as the cycles is chosen at a singular node; if none exists, a node on the boundary is chosen.

Finally, note that the constraint (13) is three-dimensional. For each path of the tree it is sufficient to set up this constraint only for those of the three dimensions that are perpendicular to one of the two connected critical entities. This is because vertices on a singular link need integer parameters only in two dimensions (not the parallel, aligned one), vertices on a boundary region only in one.

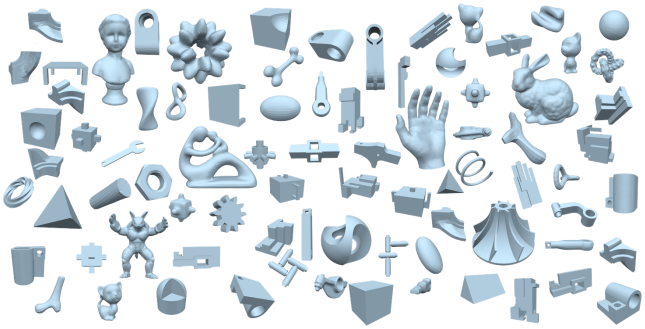


Fig. 14. Gallery of models from the datasets used for our experiments. Note that some may look like duplicates, but singularity structures differ.

Constrained Parametrization. We now can add these constraints to the volumetric seamless parametrization problem of [Nieser et al. 2011, Eq. (10)], together with one additional constraint pinning the root node to $(0, 0, 0)$. They effectively prescribe the integer degrees of freedom, so the resulting parametrization will be an integer-grid map.

Let us remind that the general problem of valid (locally injective) volumetric parametrization still lacks a fully robust solution (cf. Sec. 2), so also this particular optimization formulation is not guaranteed to yield a locally injective map (whether with or without the integer constraints). For orientation: for the most challenging, maximally coarse quantizations considered in the experiments in Sec. 7.1, reparametrization yields a fully valid map in 84% of the cases. (A typically mild increase in chosen target resolution, on average by $1.12\times$, leads to a quantization for which a valid map is obtained in all cases.) With the integer constraints determined by the proposed method, it is clear that an occasional failure is due to this general shortcoming, not due to an infeasible quantization.

In Sec. 7 we show, analyze, and discuss integer-grid maps and hexahedral meshes generated in the here described way.

7 RESULTS

We demonstrate the qualities of the proposed quantization method on the basis of a set of input seamless parametrizations and in comparison to a rounding-based strategy.

Datasets. As input (Fig. 14) for our experiments we use 15 seamless (non-integer) parametrizations generated using the approach of [Liu et al. 2018], provided by the authors, and 75 seamless (non-integer) parametrizations we generated using the approach from [Nieser et al. 2011], i.e. with a singularity structure derived from given meta-meshes. Meshes released with a number of papers [Cherchi et al. 2019, 2016; Corman and Crane 2019; Fang et al. 2016; Fu et al. 2016; Gao et al. 2015b; Gregson et al. 2011; Huang et al. 2014; Li et al. 2012; Livesu et al. 2017, 2015, 2013; Shang et al. 2017; Takayama 2019; Wu et al. 2018, 2017] served as meta-meshes in this context.

In Sec. 7.1 we consider aspects of the produced quantizations themselves, in Sec. 7.2 we inspect integer-grid maps and hexahedral meshes generated based on our quantizations as described in Sec. 6. The hexahedral meshes we show, extracted from the generated integer-grid maps using HexEx [Lyon et al. 2016], are visualized using HexaLab [Bracci et al. 2019]. The MC of the input is generated using the open source implementation of Brückler et al. [2022]. The IQP is solved using [Gurobi Optimization, LLC 2022].

7.1 Quantization

When aiming to create an extremely dense, high resolution integer-grid map or hexahedral mesh, the choice of approach to decide on integer values is uncritical. Whether MC-based or rounding-based, and regardless of parameter settings, for a sufficiently fine resolution they will eventually yield a valid assignment. Differences therefore become most apparent when targeting coarser resolutions.

We start by considering the extreme case, aiming for as little total parametric volume ("number of hexes") as possible. The rounding-based strategy consistently fails in this scenario, it cannot produce a valid integer-assignment for any input. Our MC-based quantization, even for a scaling factor $s = 0$ in Alg. 1, yields a valid solution by construction. We compare the effect of supporting either $\mathbb{Z}^{>0}$, $\mathbb{Z}^{\geq 0}$, or $\mathbb{Z}$ as value range of the quantization q. [Brückler et al. 2022] use the former (*positive*) for one of their demonstrations. Methods for the 2D case often support *non-negative* values. Our approach even supports *arbitrary* values, subject to non-negative block extents (8). Table 1 lists the parametric volume of the quantization results for these three options, when aiming for maximum coarseness.

Effect of Zeros. Significant differences can be observed in Table 1. Requiring strictly positive quantization values on arcs leads to results up to (in one case) $23\times$ more complex than the non-negative case. While the coarsest possible result may not be the one needed in concrete use cases, this difference illustrates the significantly higher general flexibility, the significantly larger space of valid integer assignments that can be represented and achieved when supporting zeros. This is part of the justification for the extra effort, essentially all of Algorithm 1 starting from the line marked by $*$ (except for the solve), which is necessary exactly to enable the safe use of zeros. Fig. 15 provides further insight on this matter.

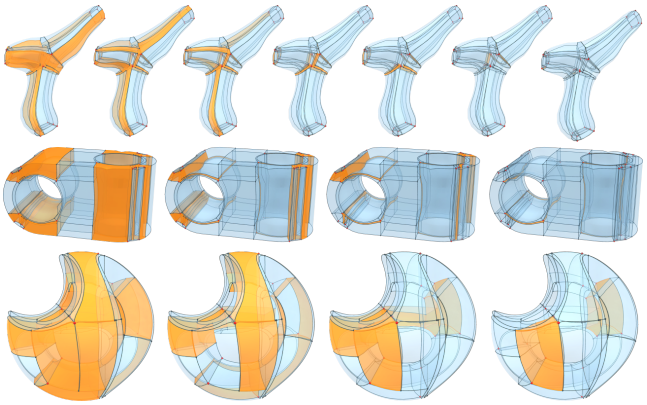


Fig. 15. Zero-arcs and zero-patches of valid quantizations computed by our method are highlighted in orange in a semi-transparent rendering of the MC. From left to right, the set target resolution is increased, thus the number of elements that are ideally quantized to zero decreases.

Table 1. For inputs from our two datasets, the smallest quantized parameter volume ("number of hexes") that is achieved depending on the employed value range ($\ell_a \in \mathbb{Z}^{>0}$, $\mathbb{Z}^{\geqslant 0}$, or $\mathbb{Z}$) is shown in grey. The last columns show the ratios of these numbers. (For complete lower table, see Table 4).

Model	$\mathbb{Z}^{>0}$	$\mathbb{Z}^{\geqslant 0}$	$\mathbb{Z}$	$\frac{\mathbb{Z}^{\geqslant 0}}{\mathbb{Z}^{>0}}$	$\frac{\mathbb{Z}}{\mathbb{Z}^{>0}}$
SCULPTURE	108	16	16	15%	15%
CYLINDER	26	5	5	19%	19%
JOINT	205	62	62	30%	30%
ARMADILLO	1832	596	670	33%	37%
ROCKERARM	1347	483	483	36%	36%
FANDISK	110	51	51	46%	46%
BROKEN BULLET	44	24	24	55%	55%
BONE	87	56	56	64%	64%
CAMILLE HAND	122	79	79	65%	65%
KITTEN	176	139	139	79%	79%
CUBE SPHERE	10	10	10	100%	100%
FANPART	5	5	5	100%	100%
PRISMA	3	3	3	100%	100%
SPHERE	7	7	7	100%	100%
TETRAHEDRON	4	4	4	100%	100%
2018 - FCBPSVDFHM EXAMPLE 2	16824	732	406	4%	2%
2019 - SPFPBHM DOUBLE HINGE WH	1826	81	78	4%	4%
2016 - PSFCLOSAV CUBESPIKES POLYCUBE IN	736	33	33	4%	4%
2016 - PSFCLOSAV CUBESPIKES POLYCUBE OUT	582	33	33	6%	6%
2016 - PSFCLOSAV CUBESPIKES MODEL IN	3687	241	241	7%	7%
2016 - PSFCLOSAV CUBESPIKES MODEL OUT	3460	334	284	10%	8%
2017 - ECMFGTS FEMUR SHELL1	190	19	20	10%	11%
2016 - AMUCIP FANCY RING-HEX	120	15	15	13%	13%
2016 - SVDVGS ROTELLIPSE PADDED	134	19	19	14%	14%
2016 - SVDVGS TWISTEDU	134	19	19	14%	14%
2012 - AMUSF SCULPTURE-A	108	18	18	17%	17%
2019 - SPFPBHM COLUMN	100	18	18	18%	18%
2011 - AMGVVPD BUMPY TORUS	3927	811	806	21%	21%
2017 - AGATMSMG EXAMPLE 3	5164	1189	1086	23%	21%
2012 - AMUSF ROD	301	64	64	21%	21%
⋮					

Effect of Negatives. When allowing even negative values, which, to the best of our knowledge, is considered here for the first time, even up to 41× simpler results can be achieved relative to the positive case. The benefit relative to the non-negative case in terms of maximum coarseness is less significant, though (a maximum of 1.8× in our tests). But considering that it does not require additional algorithmic effort, it is well worth using, especially because it has benefits in terms of enabling improved structural quality (Fig. 16).

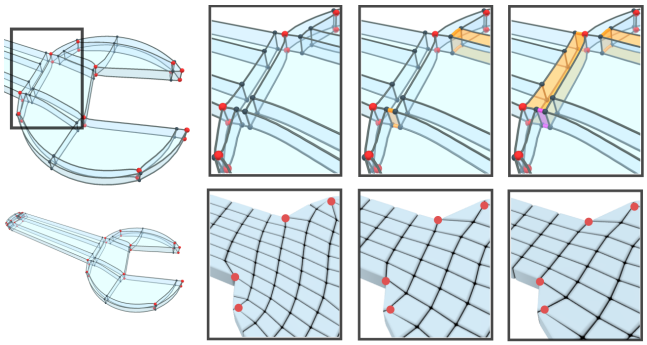


Fig. 16. Features, such as singularities, that are not perfectly aligned in the input parametrization may be forced to misalign even more when operating with only positive arc values (left). Supporting zeros (orange, center) or even negative values (pink, right) yields results (bottom) with improved structure.

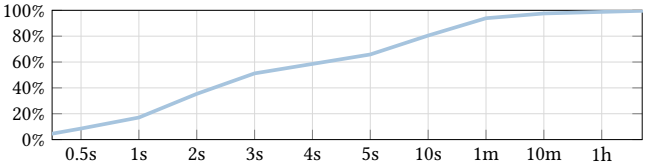


Fig. 17. Fraction of all input dataset cases for which quantization takes less time than indicated on the horizontal axis. The time taken is linked to the number of arcs (which is closely related to the input's complexity in terms of singularities); as a rule of thumb, 15ms per arc can typically be expected.

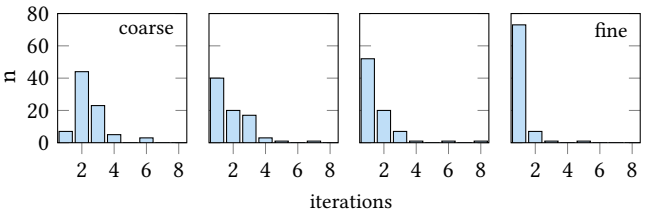
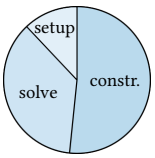


Fig. 18. Number of lazy constraint iterations needed. Reported are the number n of input cases from the dataset that needed the respective number of iterations. Each histogram shows the numbers for a different level of target coarseness, from maximally coarse (left) to fine (right).

Timing. The main contributors to the run time of our quantization method are the general setup (discovery of entities in MC, etc.), the lazy discovery of critical paths, and (repeatedly) solving of the IQP. The latter, being a discrete optimization problem with numerous equality as well as inequality constraints, may be expected to be the dominating part. However, note that the complexity of the MC, and therefore the number of variables and constraints, depends mainly on the complexity of the singularity network. It does not depend on the resolution of the input mesh M. In fact, on average the IQP solution or solutions took 36% of the total time (see inset). The main share is commonly taken by the constraint construction by means of critical path discovery. Only occasionally we observe the solver in total clearly dominating the overall run time (in 8% of the cases ⩾ 75%). Absolute values are indicated in Fig. 17



Due to our choice of lazy constraint ordering strategy, the number of repeated IQP solves, i.e. of iterations of lazily adding constraints, is typically quite low. As can be seen in Fig. 18, in the majority of cases one or two solves are sufficient, an initial solve with link constraints of type (i), followed by a second solve with critical separation constraints of type (ii). As expected, coarser target resolution cause more conflicts and require more iterations on average.

7.2 Integer-Grid Maps and Hex Meshes

Let us also compare to the rounding-based approach. We apply the best existing variant, incremental rounding with interleaved optimization [Jiang et al. 2014; Li et al. 2012], to the dataset of

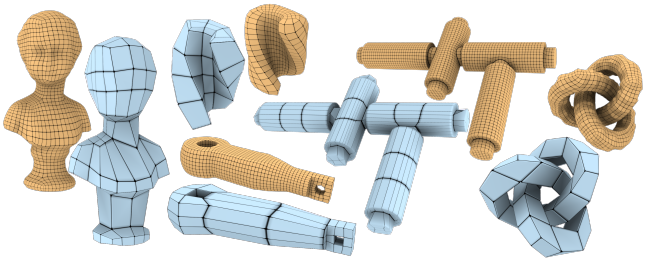


Fig. 19. Orange: Coarsest hex meshes for various models achievable using rounding. Any coarser target resolution leads to severely invalid integer-grid maps that cannot even be repaired by HexEx [Lyon et al. 2016]. Blue: Some much coarser hex meshes generated using our quantization method.

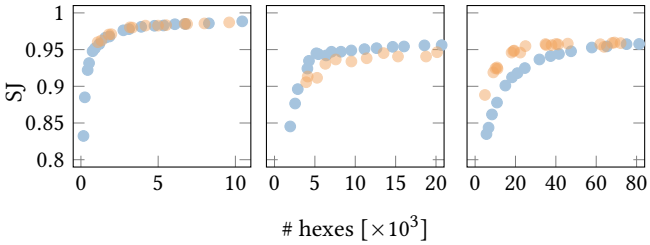


Fig. 20. Comparison between our method (blue) and rounding (orange) in terms of resulting hex mesh quality (mean scaled Jacobian) for varying output mesh resolution. Left: typical/average case (model AMuSF Joint). Center and right: best and worst cases (Fcbpvdfhm Example2 and AMUCIP Rockerarm) over the entire dataset. Also see the SJ columns in Table 2.

seamless parametrizations. Note that hence the problem structure (in terms of the singularity network, boundary alignment) is identical, the only difference is the integer determination strategy.

The only known way to verify the validity of an integer assignment generated by rounding is via a valid (locally injective) map that adheres to it and thereby certifies its validity. Given the remaining limitations of existing methods for 3D map computation (cf. Sec. 2), this can only conservatively be checked. For a fair comparison, for the purpose of this experiment we also consider the quantizations generated by our method valid only if a valid map for them is found using the same map computation setup. We use the setup of [Jiang et al. 2014, Sec. 6] including adaptive stiffening (raising the iteration limit from 5 to 30 to increase chances even further).

Coarseness. Using a binary search over the target resolution parameter, for both approaches (rounding and MC-based quantization) we determine the coarsest resolution for which they yield an integer assignment that is valid in the sense that a locally injective integer-grid map is achieved using the above setup. The results are reported in Table 2, and Fig. 19 shows some examples. The fragility of rounding becomes obvious: In some cases the simplest valid map obtainable with rounding is around 100× more complex than that obtainable based on our method.

Quality. As a further comparison, beyond validity, let us consider the relative quality of results, in terms of mesh quality measured in meshes extracted from the generated integer-grid maps. Fig. 20 shows plots of the mean scaled Jacobian quality measure, over output meshes of varying resolution. It can be observed that, for finer resolutions on which rounding also succeeds, our method does not differ much, i.e. its reliability does not come at a significant cost of quality in general. Fig. 22 shows some visual examples.

Another quality comparison (regarding the employed quantization value ranges, $\mathbb{Z}^{>0}$ or $\mathbb{Z}$) is shown in Fig. 21.

Remark: Hex Geometry. Let us remark that while a valid integer-grid map always implies a *structurally* valid hex mesh, the question of *geometric* validity is more intricate. The map associates each logical hex element with some unique non-degenerate region of the input object, but in practice often geometrically simple (e.g. trilinear) elements are assumed. These only approximate the map and may exhibit inversions even if the underlying map is valid. For instance, at the maximally coarse resolutions as reported in Tables 2 and 5, in 46% of the implied meshes all (trilinear) hex elements have positive scaled Jacobian values. With increasing resolution, this rate increases, e.g., at 1.5× higher quantization target resolution to 76%, at 2.0× to 82%. Note that this aspect is common to parametrization-based methods in general, not particular to our quantization approach, and deserves further attention in future work.

Table 2. For inputs from our two datasets, the number of hexes in the simplest mesh that is achieved with either rounding or our method (as well as their ratio) is shown. Furthermore, the columns SJ_r and SJ_q show a comparison of the mean scaled Jacobian quality between then simplest mesh achieved with rounding and a mesh of similar resolution generated using our quantization method. (For complete lower table, see Table 5).

Model	SJ_r	SJ_q	round	**ours**	$\frac{\text{ours}}{\text{round}}$
kitten	0.90	0.90	3276	139	4%
sculpture	0.93	0.93	386	18	5%
rockerarm	0.91	0.86	8056	571	7%
cylinder	0.78	0.78	60	5	8%
joint	0.95	0.95	715	62	9%
bone	0.80	0.80	628	70	11%
sphere	0.71	0.71	32	7	22%
fandisk	0.84	0.85	162	51	31%
prisma	0.85	0.85	9	3	33%
camille hand	0.81	0.79	1438	773	54%
armadillo	0.92	0.92	29306	16393	56%
broken bullet	0.80	0.80	36	24	67%
cube sphere	0.85	0.85	10	10	100%
fanpart	0.92	0.92	5	5	100%
tetrahedron	0.81	0.81	4	4	100%
2016 - AMUCIP knot-hex	0.84	0.84	4704	45	1%
2019 - SPfPBHM Chamfer L4	0.95	0.96	484	7	1%
2016 - SVDvGS twistedu	0.81	0.80	1227	19	2%
2017 - Agatmsmg Example 2	0.92	0.93	30334	695	2%
2016 - SVDvGS rotellipse padded	0.78	0.78	804	19	2%
2018 - Fcbpsvdfhm Example 5	0.99	0.99	37	1	3%
2019 - SPfPBHM Double hinge WH	0.95	0.94	2592	81	3%
2013 - PolyCut BU hex opt	0.89	0.87	8337	300	4%
2016 - Psfclosav table2 polycube out	1.00	1.00	340	13	4%
2016 - Psfclosav cubespikes model in	0.95	0.96	5895	241	4%
2017 - Agatmsmg Example 4	0.88	0.92	15799	660	4%
2019 - SPfPBHM Gear	0.95	0.95	1456	69	5%
2016 - EVPC Fertility hex-largel	0.89	0.89	7693	379	5%
2014 - l1CoPMfCS kitty	0.91	0.91	2522	137	5%
⋮					

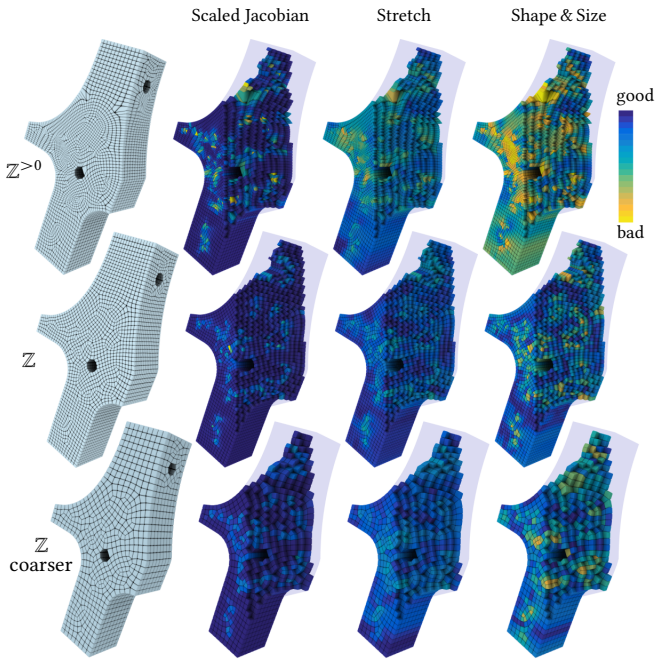


Fig. 21. Top: Coarsest hex mesh that can be achieved using $\ell_a \in \mathbb{Z}^{>0}$, together with visualization of three quality measures [Knupp et al. 2006]. Middle: A mesh of equal resolution generated using $\ell_a \in \mathbb{Z}$. Its quality is significantly better. Bottom: Using this larger value range, also coarser meshes can be achieved, still with higher quality than in the finer top row.

7.2.1 Hex Mesh Input. Finally, let us just briefly point out another potential use. As mentioned in Sec. 1.1, hex meshes can also be taken as input—they trivially induce a seamless parametrization. We can then generate an output hex mesh with identical singularity structure, but different resolution or different *base complex*. The base complex is the coarsest conforming partition of the mesh into regular pieces, i.e. blocks of some numbers $m \times n \times o$ of hexes. A mesh with sufficiently simple base complex is said to be block-structured, and is desirable in certain contexts [Armstrong et al. 2015].

To improve (i.e. simplify) the base complex, take a hex mesh as input, compute a coarsely quantized integer-grid map, and extract a hex mesh from it at varying sub-integer resolution, such that multiple hexes are created per parametric unit cube, aiming to match the input mesh resolution (or a possibly deviating user-defined resolution). To illustrate the potential, for a selection of hexahedral meshes from the repository of [Bracci et al. 2019] we report the number of blocks in the base complex before and after this processing in Table 3. Depending on the use case, of course further control over the base complex' quality beyond just simplicity may be needed, e.g, as explored for the 2D case by [Lyon et al. 2021a].

Let us note that [Gao et al. 2017] also consider a base complex simplification problem. They take a more powerful approach that allows the singularity structure to be modified, granting more flexibility. Assuming a scenario where the singularity structure is to be preserved, our approach may offer some advantage: the non-conforming structure of the MC our method builds on, allows for a broader range of (singularity-preserving) modifications than the conforming sheet collapses used in the above method. The relevance to the specific task at hand appears to be not huge, though; while we observed a further reduction relative to sheet collapses by a factor of $5\times$ in one case, in most other cases the factor is lower than $2\times$.

Table 3. Improvement of block-structuredness (base complex simplicity). Reported is the number of base complex blocks in the input (BC in), the number achieved using sheet collapses (BC sheet), the number achieved using our method (BC ours), as well as ratios of them.

Model	BC in	BC sheet	BC ours	ours / sheet	ours / in
2018 - Fcbpvdfhm Example 1	74331	59691	11065	19%	15%
2016 - EVPC Armadillo hex-b	3265	640	505	79%	15%
2016 - EVPC Bunny hex	1282	300	221	74%	17%
2016 - EVPC Armadillo hex-a	5960	1263	1031	82%	17%
2014 - l1CoPMfCS gargoyle	7563	2102	1419	68%	19%
2011 - AMGvVPD bunny hex	1324	254	258	102%	19%
2016 - AMUCIP pegasus-hex	9745	3945	2032	52%	21%
2016 - AMUCIP dragon-hex	12488	3299	2770	84%	22%
2014 - l1CoPMfCS dancing-children-2	5482	1656	1313	79%	24%
2012 - AMuSF impeller	878	326	224	69%	26%
2016 - Psfclosav bunny model in	637	260	165	63%	26%
2016 - EVPC Elephant hex	3105	970	835	86%	27%
2012 - AMuSF fertility	1352	934	375	40%	28%
2016 - EVPC Buste hex	1081	362	303	84%	28%

8 OUTLOOK & FUTURE WORK

Singularity Relaxation. We assumed singularities to be respected as given. If this is not strictly necessary, better quantizations (in terms of implied distortion) may be achieved by allowing singularities to merge where beneficial, as recently demonstrated for the 2D case by [Lyon et al. 2021b]. Enabling this in the 3D case will require some additional effort as not only simple singular points need to be moved in accordance with the quantization, but singular curves need to be rerouted. There may furthermore be restrictions on the types of singularities that can validly be merged perpendicularly.

Blockwise Reparametrization. The zero-block collapse concept employed as a mental picture in Sec. 5.1 could actually be executed explicitly in the end, yielding a volume-T-mesh MC' in M with strictly positive integer values. If also negative arc lengths are permitted, these require additional care. This MC' would enable establishing a valid integer-grid map in a potentially simpler blockwise manner. A strategy along these lines has been employed in the 2D surface setting [Lyon et al. 2019]. Exploring a generalization to 3D will be interesting, though the collapsing procedure is intricate already in 2D.

Objective Function. We employed the objective function (9), as also used in related work for the surface case. It seems plausible that other objectives could lead to results of even higher quality, depending on the relevant notion. For instance, it may be more important to stay close to the original positions than to their distances, distances may be measured on certain paths rather than on individual arcs, a weighted version could be used, or minimal volume may be targeted explicitly (which ours does only indirectly for $s = 0$).

Hex Layouts. Variations of the 2D T-mesh based quantization approach have recently been explored for the generation of coarse block structures for quad meshes (so-called quad layouts) [Lyon et al.

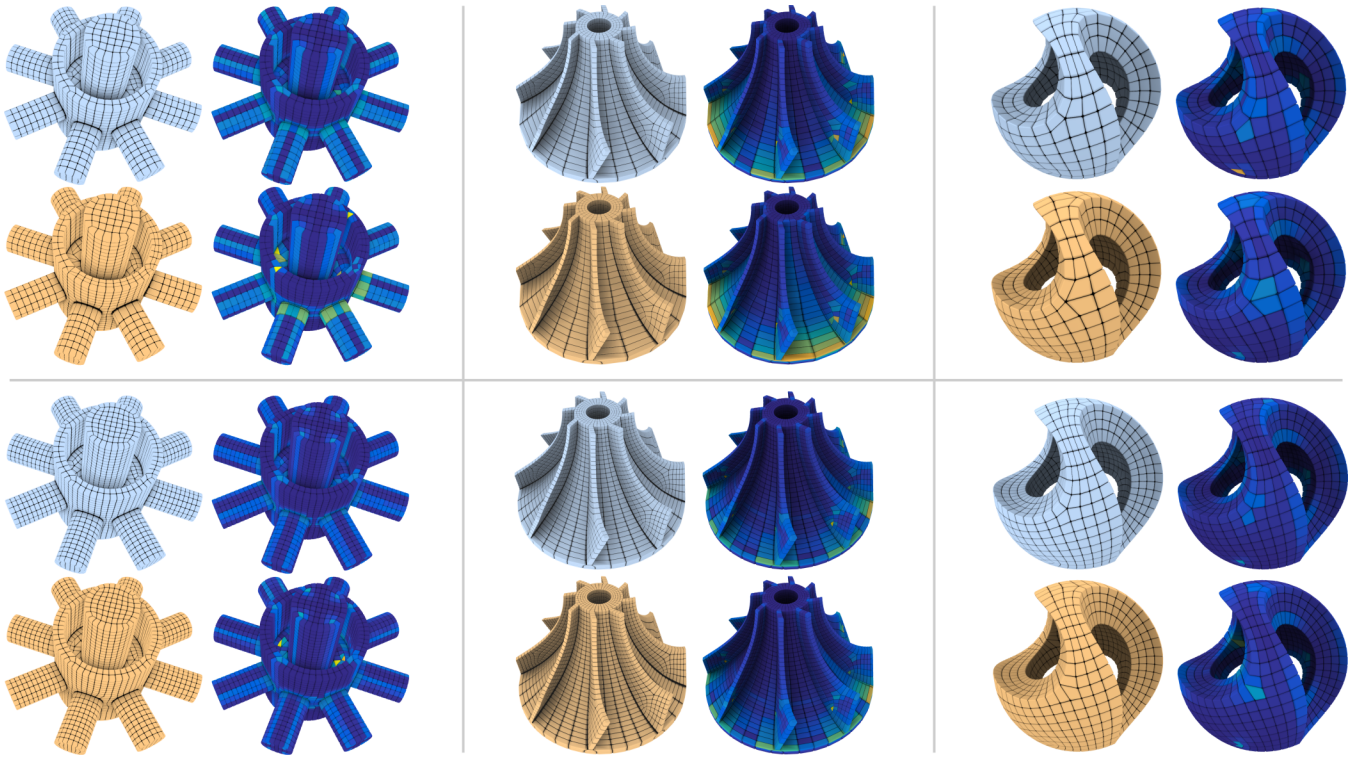


Fig. 22. Hex meshes of three example models, in a coarser (top) and finer (bottom) version. Integers were determined using our method (light blue) or rounding (orange). Next to each mesh, a visualization of per-element scaled Jacobian values is shown. Color scale as in Fig. 21. Notice that rounding behavior is not monotonic: In the bottom right case, despite being of higher resolution than the top case, the integer-grid map generated by rounding had defects; HexEx managed to extract a mesh anyway, albeit at the cost of disrespecting the input singularity structure (singularity merged into boundary in the front).

2021a,b]. Analogously, hex layouts may be of interest for the generation of block-structured hexahedral meshes [Armstrong et al. 2015]. While very coarse quantizations generated using the proposed methods may already be suitable for this purpose (cf. Sec. 7.2.1), higher quality could likely be achieved by a tailored variant.

Other Cell Complexes. The motorcycle complex assumes a valid seamless parametrization as input. It may be tempting to instead use a more general, non-aligned, perhaps tetrahedral cell complex to represent the quantization on. A challenge, however, is the expression of validity constraints and the preservation of feasibility on such a non-aligned complex, as already the constraint expressing non-inversion of cells becomes non-convex in this case.

Negative Blocks. Conceptually, there is no strict need to forbid blocks of negative extent in the quantization. They would grant additional flexibility, generally allowing singularities to switch their relative order in the parametric dimensions. This is unlikely to be beneficial in many cases, but for coarse target resolutions may pay off. It will be challenging, though, to support negative blocks in the critical path discovery, as the R^0 region cannot be explored in the described monotonic manner in this case.

Special Purpose Solver. We solve the IQPs using a generic solver. For the 2D case, a tailored strategy is described by [Campen et al. 2015]: Starting from a consistent quantization, violated inequalities are cured by "inflating" one of the contained arcs. To preserve consistency, this inflation needs to be propagated along a dual path until a cycle is closed (or the boundary is reached). An analogous process in 3D would require finding a *dual surface*, a significantly harder problem. Fortunately, we observe the generic solver—reported to be slow in the surface case—to perform well in the volume case (cf. Sec. 7.1). We believe this to be related to the fact that in particular on closed surfaces, arc variables often interdepend along complex, long-winding dual cycles. In the volumetric case, where we instead always have a boundary and a flat metric in M, dependencies between variables are simpler, benefiting the solver. Nevertheless, it can be interesting to investigate tailored strategies in the future.

Hexahedral Meshing. Finally, as laid out in Sec. 1, now that robustness gap ③ has found a solution, addressing gaps ① and ② is key to making parametrization-based hex meshing fully mature in the near future.

ACKNOWLEDGMENTS

This work was funded by the Deutsche Forschungsgemeinschaft (DFG) - 427469366; 456666331. D. Bommes has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (AlgoHex, grant agreement No 853343).

Table 4. Complete version of bottom part of Table 1

Model	$\mathbb{Z}^{>0}$	$\mathbb{Z}^{\geqslant 0}$	$\mathbb{Z}$	$\frac{\mathbb{Z}^{\geqslant 0}}{\mathbb{Z}^{>0}}$	$\frac{\mathbb{Z}}{\mathbb{Z}^{>0}}$
2018 - Fcbpsvdfhm Example 2	16824	732	406	4%	2%
2019 - SPfPBHM Double hinge WH	1826	81	78	4%	4%
2016 - Psfclosav cubespikes polycube in	736	33	33	4%	4%
2016 - Psfclosav cubespikes polycube out	582	33	33	6%	6%
2016 - Psfclosav cubespikes model in	3687	241	241	7%	7%
2016 - Psfclosav cubespikes model out	3460	334	284	10%	8%
2017 - ECMfGTS femur shell1	190	19	20	10%	11%
2016 - AMUCIP fancy ring-hex	120	15	15	13%	13%
2016 - SVDvGS rotellipse padded	134	19	19	14%	14%
2016 - SVDvGS twistedu	134	19	19	14%	14%
2012 - AMuSF sculpture-A	108	18	18	17%	17%
2019 - SPfPBHM Column	100	18	18	18%	18%
2011 - AMGvVPD bumpy torus	3927	811	806	21%	21%
2017 - Agatmsmg Example 3	5164	1189	1086	23%	21%
2012 - AMuSF rod	301	64	64	21%	21%
2016 - Psfclosav bunny polycube in	194	46	45	24%	23%
2019 - SPfPBHM Double hinge NH	679	165	165	24%	24%
2016 - Psfclosav block polycube out	93	24	24	26%	26%
2017 - Agatmsmg Example 4	2553	727	660	28%	26%
2016 - Psfclosav bunny polycube out	155	41	41	26%	26%
2012 - AMuSF rockerarm	1691	449	449	27%	27%
2019 - SPfPBHM Chamfer L4	26	7	7	27%	27%
2016 - AMUCIP hollow-eight-hex	1369	369	369	27%	27%
2016 - Psfclosav block polycube in	114	42	31	37%	27%
2019 - SPfPBHM Wrench	122	41	35	34%	29%
2016 - Psfclosav chinese dragon polycube in	253	76	76	30%	30%
2017 - Agatmsmg Example 2	1750	527	536	30%	31%
2016 - Psfclosav teapot polycube in	217	89	69	41%	32%
2012 - AMuSF joint	202	66	66	33%	33%
2017 - HMGvCQ pone.0177603.s003	202	66	66	33%	33%
2016 - Psfclosav rockerarm polycube out	520	211	187	41%	36%
2012 - AMuSF fandisk	80	30	30	38%	38%
2016 - Psfclosav teapot polycube out	223	88	84	39%	38%
2016 - Psfclosav chinese dragon polycube out	223	85	85	38%	38%
2019 - DSM fandisk	161	53	62	33%	39%
2016 - AMUCIP rockerarm-hex	1778	707	697	40%	39%
2013 - PolyCut bunny hex opt	524	271	207	52%	40%
2012 - AMuSF sculpture-B	69	28	28	41%	41%
2012 - AMuSF double	410	169	169	41%	41%
2019 - DSM fandisk.liu18	122	51	51	42%	42%
2016 - Psfclosav rockerarm polycube in	251	115	107	46%	43%
2015 - PHOvER impeller stresstest out	627	272	272	43%	43%
2016 - EVPC Fertility hex-largel	789	374	379	47%	48%
2017 - HMGvCQ fandisk	80	39	39	49%	49%
2017 - HMGvCQ pone.0177603.s002	80	39	39	49%	49%
2016 - Psfclosav fandisk polycube in	106	52	52	49%	49%
2016 - Psfclosav fandisk polycube out	98	52	52	53%	53%
2013 - PolyCut BU hex opt	564	291	300	52%	53%
2019 - SMF hex brokenbullet	44	24	24	55%	55%
2016 - Psfclosav hand polycube in	37	21	21	57%	57%
2011 - AMGvVPD asm001	366	212	212	58%	58%
2019 - SPfPBHM Gear	119	69	69	58%	58%
2016 - Psfclosav femur polycube out	43	34	25	79%	58%
2016 - Psfclosav table2 polycube in	22	13	13	59%	59%
2012 - AMuSF ellipsoid-A	52	34	34	65%	65%
2016 - Psfclosav hand polycube out	41	27	27	66%	66%
2014 - l1CoPMfCS rod	320	212	212	66%	66%
2016 - Psfclosav asm polycube out	47	32	32	68%	68%
2017 - ECMfGTS femur shell0	11	8	8	73%	73%
2016 - AMUCIP kitten-hex	176	139	139	79%	79%
2014 - l1CoPMfCS kitty	172	137	137	80%	80%
2016 - AMUCIP joint-hex	59	47	47	80%	80%
2012 - AMuSF hanger	50	41	41	82%	82%
2019 - DSM hanger	50	41	41	82%	82%
2016 - Psfclosav asm polycube in	78	66	66	85%	85%
2016 - Psfclosav table1 polycube in	60	52	52	87%	87%
2016 - Psfclosav table1 polycube out	53	53	46	100%	87%
2016 - AMUCIP knot-hex	50	45	45	90%	90%
2012 - AMuSF ellipsoid-B	7	7	7	100%	100%
2012 - AMuSF ellipsoid-C	7	7	7	100%	100%
2016 - AMUCIP nut-hex	12	12	12	100%	100%
2017 - ECMfGTS cylinder grid	1	1	1	100%	100%
2017 - HMGvCQ cube	17	17	17	100%	100%
2017 - HMGvCQ pone.0177603.s001	17	17	17	100%	100%
2018 - Fcbpsvdfhm Example 5	1	1	1	100%	100%
2019 - DSM double-torus	7	7	7	100%	100%
2019 - SMF hex tetrahedron	4	4	4	100%	100%

Table 5. Complete version of bottom part of Table 2

Model	SJ_r	SJ_q	round	**ours**	$\frac{\textbf{ours}}{\textbf{round}}$
2016 - AMUCIP knot-hex	0.84	0.84	4704	45	1%
2019 - SPfPBHM Chamfer L4	0.95	0.96	484	7	1%
2016 - SVDvGS twistedu	0.81	0.80	1227	19	2%
2017 - Agatmsmg Example 2	0.92	0.93	30334	695	2%
2016 - SVDvGS rotellipse padded	0.78	0.78	804	19	2%
2018 - Fcbpsvdfhm Example 5	0.99	0.99	37	1	3%
2019 - SPfPBHM Double hinge WH	0.95	0.94	2592	81	3%
2013 - PolyCut BU hex opt	0.89	0.87	8337	300	4%
2016 - Psfclosav table2 polycube out	1.00	1.00	340	13	4%
2016 - Psfclosav cubespikes model in	0.95	0.96	5895	241	4%
2017 - Agatmsmg Example 4	0.88	0.92	15799	660	4%
2019 - SPfPBHM Gear	0.95	0.95	1456	69	5%
2016 - EVPC Fertility hex-largel	0.89	0.89	7693	379	5%
2014 - l1CoPMfCS kitty	0.91	0.91	2522	137	5%
2016 - AMUCIP hollow-eight-hex	0.89	0.89	6757	384	6%
2013 - PolyCut bunny hex opt	0.93	0.92	7384	425	6%
2016 - Psfclosav bunny polycube out	1.00	1.00	685	41	6%
2011 - AMGvVPD bumpy torus	0.92	0.92	12234	806	7%
2011 - AMGvVPD asm001	0.92	0.88	3450	245	7%
2012 - AMuSF joint	0.96	0.96	864	66	8%
2017 - HMGvCQ pone.0177603.s003	0.96	0.96	864	66	8%
2019 - SPfPBHM Column	0.91	0.93	209	18	9%
2016 - AMUCIP rockerarm-hex	0.93	0.88	8850	796	9%
2016 - AMUCIP kitten-hex	0.77	0.79	1277	139	11%
2017 - ECMfGTS femur shell1	0.77	0.79	163	20	12%
2016 - AMUCIP fancy ring-hex	0.94	0.95	120	15	13%
2016 - Psfclosav teapot polycube out	0.99	0.99	503	69	14%
2016 - Psfclosav teapot polycube in	0.99	0.99	501	69	14%
2012 - AMuSF ellipsoid-B	0.69	0.68	50	7	14%
2014 - l1CoPMfCS rod	0.90	0.85	1646	245	15%
2019 - SPfPBHM Double hinge NH	0.93	0.91	1090	165	15%
2015 - PHOvER impeller stresstest out	0.86	0.88	1302	272	21%
2019 - DSM hanger	0.92	0.93	185	41	22%
2012 - AMuSF hanger	0.91	0.92	180	41	23%
2012 - AMuSF rod	0.90	0.90	261	64	25%
2017 - Agatmsmg Example 3	0.90	0.92	6411	1597	25%
2016 - Psfclosav chinese dragon polycube in	0.96	0.96	278	76	27%
2016 - Psfclosav asm polycube out	0.99	1.00	116	32	28%
2016 - Psfclosav femur polycube out	0.99	0.99	85	25	29%
2016 - Psfclosav bunny polycube in	0.99	0.99	151	45	30%
2012 - AMuSF fandisk	0.90	0.90	100	30	30%
2019 - DSM fandisk.liu18	0.89	0.90	164	51	31%
2012 - AMuSF rockerarm	0.79	0.71	1632	509	31%
2016 - AMUCIP joint-hex	0.95	0.94	150	47	31%
2012 - AMuSF double	0.76	0.77	488	169	35%
2016 - Psfclosav table1 polycube out	0.99	0.99	129	46	36%
2019 - DSM double-torus	0.72	0.83	14	5	36%
2016 - Psfclosav chinese dragon polycube out	0.99	0.99	225	85	38%
2017 - HMGvCQ fandisk	0.88	0.90	100	39	39%
2017 - HMGvCQ pone.0177603.s002	0.87	0.90	100	39	39%
2016 - Psfclosav block polycube out	0.96	0.98	60	24	40%
2016 - Psfclosav fandisk polycube in	0.96	0.93	129	52	40%
2019 - DSM fandisk	0.93	0.93	147	62	42%
2016 - Psfclosav cubespikes model out	0.86	0.85	622	284	46%
2016 - Psfclosav table1 polycube in	0.96	0.97	105	52	50%
2016 - Psfclosav block polycube in	0.94	0.97	60	31	52%
2016 - Psfclosav hand polycube in	0.96	0.96	40	21	53%
2012 - AMuSF sculpture-A	0.83	0.66	30	16	53%
2012 - AMuSF ellipsoid-A	0.71	0.72	64	34	53%
2016 - Psfclosav rockerarm polycube in	0.93	0.75	199	107	54%
2018 - Fcbpsvdfhm Example 2	0.92	0.91	3614	1947	54%
2019 - SPfPBHM Wrench	0.96	0.96	64	35	55%
2016 - Psfclosav asm polycube in	0.99	1.00	116	66	57%
2019 - SMF hex brokenbullet	0.80	0.80	36	24	67%
2016 - Psfclosav hand polycube out	0.96	0.96	36	27	75%
2016 - Psfclosav fandisk polycube out	0.97	0.95	69	52	75%
2016 - Psfclosav rockerarm polycube out	0.99	0.98	234	187	80%
2017 - HMGvCQ cube	0.94	0.94	18	17	94%
2017 - HMGvCQ pone.0177603.s001	0.94	0.94	18	17	94%
2012 - AMuSF sculpture-B	0.76	0.62	29	28	97%
2012 - AMuSF ellipsoid-C	0.56	0.56	7	7	100%
2016 - AMUCIP nut-hex	0.92	0.92	12	12	100%
2016 - Psfclosav cubespikes polycube in	0.98	0.98	33	33	100%
2016 - Psfclosav cubespikes polycube out	0.98	0.98	33	33	100%
2017 - ECMfGTS cylinder grid	0.98	0.98	1	1	100%
2017 - ECMfGTS femur shell0	0.92	0.92	8	8	100%
2019 - SMF hex tetrahedron	0.74	0.74	4	4	100%

REFERENCES

Noam Aigerman and Yaron Lipman. 2013. Injective and bounded distortion mappings in 3D. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–14.

Cecil G Armstrong, Harold J Fogg, Christopher M Tierney, and Trevor T Robinson. 2015. Common themes in multi-block structured quad/hex mesh generation. *Procedia Engineering* 124 (2015), 70–82.

Steven E. Benzley, Ernest Perry, Karl Merkley, Brett Clark, and Greg Sjaardema. 1995. A comparison of all hexagonal and all tetrahedral finite element meshes for elastic and elasto-plastic analysis. In *Proc. 4th International Meshing Roundtable*. 179–191.

Ted Blacker. 2000. Meeting the challenge for automated conformal hexahedral meshing. In *Proceedings of International Meshing Roundtable*. 11–20.

T Blacker. 2001. Automated conformal hexahedral meshing constraints, challenges and opportunities. *Engineering with Computers* 17, 3 (2001), 201–210.

David Bommes, Marcel Campen, Hans-Christian Ebke, Pierre Alliez, and Leif Kobbelt. 2013. Integer-Grid Maps for Reliable Quad Meshing. *ACM Trans. Graph.* 32, 4 (2013).

David Bommes, Henrik Zimmer, and Leif Kobbelt. 2009. Mixed-integer quadrangulation. *ACM Trans. Graph.* 28, 3 (2009), 77:1–77:10.

David Bommes, Henrik Zimmer, and Leif Kobbelt. 2010. Practical mixed-integer optimization for geometry processing. In *International Conference on Curves and Surfaces*. Springer, 193–206.

X. Bourdin, X. Trosseille, P. Petit, and P. Beillas. 2007. Comparison of tetrahedral and hexahedral meshes for organ finite element modeling: an application to kidney impact. In *Proc. 20th Int. Technical Conference on the Enhanced Safety of Vehicles*.

Matteo Bracci, Marco Tarini, Nico Pietroni, Marco Livesu, and Paolo Cignoni. 2019. HexaLab. net: An online viewer for hexahedral meshes. *Computer-Aided Design* 110 (2019), 24–36.

Hendrik Brückler, Ojaswi Gupta, Manish Mandad, and Marcel Campen. 2022. The 3D Motorcycle Complex for Structured Volume Decomposition. *Computer Graphics Forum* 41, 2 (2022). https://doi.org/10.1111/cgf.14470

Marcel Campen, David Bommes, and Leif Kobbelt. 2015. Quantized global parametrization. *ACM Trans. Graph.* 34, 6 (2015).

Marcel Campen, Hanxiao Shen, Jiaran Zhou, and Denis Zorin. 2019. Seamless Parametrization with Arbitrary Cones for Arbitrary Genus. *ACM Trans. Graph.* 39, 1 (2019).

Marcel Campen, Cláudio T Silva, and Denis Zorin. 2016. Bijective maps from simplicial foliations. *ACM Trans. Graph.* 35, 4 (2016), 1–15.

Marcel Campen and Denis Zorin. 2017. Similarity maps and field-guided T-splines: a perfect couple. *ACM Trans. Graph.* 36, 4 (2017), 1–16.

Gianmarco Cherchi, Pierre Alliez, Riccardo Scateni, Max Lyon, and David Bommes. 2019. Selective padding for polycube-based hexahedral meshing. In *Computer graphics forum*, Vol. 38. 580–591.

Gianmarco Cherchi, Marco Livesu, and Riccardo Scateni. 2016. Polycube simplification for coarse layouts of surfaces and volumes. In *Computer Graphics Forum*, Vol. 35. 11–20.

A.O. Cifuentes and A. Kalbag. 1992. A performance study of tetrahedral and hexahedral elements in 3-D finite element structural analysis. *Finite Elements in Analysis and Design* 12, 3 (1992), 313 – 318.

Etienne Corman and Keenan Crane. 2019. Symmetric Moving Frames. *ACM Trans. Graph.* 38, 4 (2019).

Mattéo Couplet, Maxence Reberol, and Jean-François Remacle. 2021. Generation of High-Order Coarse Quad Meshes on CAD Models via Integer Linear Programming. In *AIAA Aviation 2021 Forum*. 2991.

David Eppstein and Jeff Erickson. 1999. Raising roofs, crashing cycles, and playing pool: Applications of a data structure for finding pairwise interactions. *Discrete & Computational Geometry* 22, 4 (1999), 569–592.

David Eppstein, Michael T Goodrich, Ethan Kim, and Rasmus Tamstorf. 2008. Motorcycle graphs: canonical quad mesh partitioning. *Comp. Graph. Forum* 27, 5 (2008).

Jeff Erickson and Kim Whittlesey. 2005. Greedy optimal homotopy and homology generators. In *SODA*, Vol. 5. 1038–1046.

Xianzhong Fang, Weiwei Xu, Hujun Bao, and Jin Huang. 2016. All-Hex Meshing Using Closed-Form Induced Polycube. *ACM Trans. Graph.* 35, 4 (2016).

Xiao-Ming Fu, Chong-Yang Bai, and Yang Liu. 2016. Efficient Volumetric PolyCube-Map Construction. *Computer Graphics Forum* 35, 7 (2016).

Xiao-Ming Fu, Jian-Ping Su, Zheng-Yu Zhao, Qing Fang, Chunyang Ye, and Ligang Liu. 2021. Inversion-free geometric mapping construction: A survey. *Computational Visual Media* 7, 3 (2021), 289–318.

Xifeng Gao, Zhigang Deng, and Guoning Chen. 2015a. Hexahedral mesh re-parameterization from aligned base-complex. *ACM Trans. Graph.* 34, 4 (2015).

Xifeng Gao, Tobias Martin, Sai Deng, Elaine Cohen, Zhigang Deng, and Guoning Chen. 2015b. Structured volume decomposition via generalized sweeping. *IEEE transactions on visualization and computer graphics* 22, 7 (2015), 1899–1911.

Xifeng Gao, Daniele Panozzo, Wenping Wang, Zhigang Deng, and Guoning Chen. 2017. Robust Structure Simplification for Hex Re-Meshing. *ACM Trans. Graph.* 36, 6 (2017).

Vladimir Garanzha, Igor Kaporin, Liudmila Kudryavtseva, François Protais, Nicolas Ray, and Dmitry Sokolov. 2021. Foldover-free maps in 50 lines of code. *arXiv preprint arXiv:2102.03069* (2021).

James Gregson, Alla Sheffer, and Eugene Zhang. 2011. All-Hex Mesh Generation via Volumetric PolyCube Deformation. *Computer Graphics Forum* 30, 5 (2011).

Gurobi Optimization, LLC. 2022. Gurobi Optimizer. https://www.gurobi.com

Jin Huang, Tengfei Jiang, Zeyun Shi, Yiying Tong, Hujun Bao, and Mathieu Desbrun. 2014. ℓ1-based construction of polycube maps from complex shapes. *ACM Transactions on Graphics (TOG)* 33, 3 (2014), 1–11.

Jin Huang, Tengfei Jiang, Yuanzhen Wang, Yiying Tong, and Hujun Bao. 2012. *Automatic frame field guided hexahedral mesh generation.* Technical Report. Tech. report, State Key Lab of CAD & CG, College of Computer Science at Zhejiang University.

Jin Huang, Yiying Tong, Hongyu Wei, and Hujun Bao. 2011. Boundary aligned smooth 3D cross-frame field. *ACM Trans. Graph.* 30, 6 (2011).

Tengfei Jiang, Jin Huang, Yuanzhen Wang, Yiying Tong, and Hujun Bao. 2014. Frame field singularity correctionfor automatic hexahedralization. *IEEE Transactions on Visualization and Computer Graphics* 20, 8 (2014), 1189–1199.

Felix Kälberer, Matthias Nieser, and Konrad Polthier. 2007. QuadCover - Surface Parameterization using Branched Coverings. *Computer Graphics Forum* 26, 3 (2007).

Junho Kim, Miao Jin, Qian-Yi Zhou, Feng Luo, and Xianfeng Gu. 2008. Computing fundamental group of general 3-manifold. In *International Symposium on Visual Computing*. Springer, 965–974.

Patrick Michael Knupp, CD Ernst, David C Thompson, CJ Stimpson, and Philippe Pierre Pebay. 2006. *The verdict geometric quality library.* Technical Report. Sandia Nat. Lab.

Zohar Levi. 2021. Direct Seamless Parametrization. *ACM Trans. Graph.* 40, 1 (2021).

Zohar Levi. 2022. Seamless Parametrization of Spheres with Controlled Singularities. *Comp. Graph. Forum* (2022).

Yufei Li, Yang Liu, Weiwei Xu, Wenping Wang, and Baining Guo. 2012. All-hex meshing using singularity-restricted field. *ACM Transactions on Graphics* 31, 6 (2012), 177.

Yaron Lipman. 2012. Bounded distortion mapping spaces for triangular meshes. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–13.

Heng Liu, Paul Zhang, Edward Chien, Justin Solomon, and David Bommes. 2018. Singularity-constrained octahedral fields for hexahedral meshing. *ACM Trans. Graph.* 37, 4 (2018).

Marco Livesu, Marco Attene, Giuseppe Patané, and Michela Spagnuolo. 2017. Explicit cylindrical maps for general tubular shapes. *Computer-Aided Design* 90 (2017), 27–36.

Marco Livesu, Nico Pietroni, Enrico Puppo, Alla Sheffer, and Paolo Cignoni. 2020. LoopyCuts: Practical Feature-Preserving Block Decomposition for Strongly Hex-Dominant Meshing. *ACM Trans. Graph.* 39, 4 (2020).

Marco Livesu, Alla Sheffer, Nicholas Vining, and Marco Tarini. 2015. Practical Hex-Mesh Optimization via Edge-Cone Rectification. *ACM Trans. Graph.* 34, 4 (2015).

Marco Livesu, Nicholas Vining, Alla Sheffer, James Gregson, and Riccardo Scateni. 2013. Polycut: Monotone graph-cuts for polycube base-complex construction. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–12.

Max Lyon, David Bommes, and Leif Kobbelt. 2016. HexEx: Robust Hexahedral Mesh Extraction. *ACM Transactions on Graphics* 35, 4 (2016).

Max Lyon, Marcel Campen, David Bommes, and Leif Kobbelt. 2019. Parametrization Quantization with Free Boundaries for Trimmed Quad Meshing. *ACM Trans. Graph.* 38, 4 (2019).

Max Lyon, Marcel Campen, and Leif Kobbelt. 2021a. Quad Layouts via Constrained T-Mesh Quantization. *Computer Graphics Forum* 40, 2 (2021).

Max Lyon, Marcel Campen, and Leif Kobbelt. 2021b. Simpler Quad Layouts using Relaxed Singularities. In *Computer Graphics Forum*, Vol. 40. 169–179.

Martin Marinov, Marco Amagliani, Tristan Barback, Jean Flower, Stephen Barley, Suguru Furuta, Peter Charrot, Iain Henley, Nanda Santhanam, G. Thomas Finnigan, Siavash Meshkat, Justin Hallet, Maciej Sapun, and Pawel Wolski. 2019. Generative Design Conversion to Editable and Watertight Boundary Representation. *Computer-Aided Design* 115 (2019), 194 – 205.

Ashish Myles, Nico Pietroni, and Denis Zorin. 2014. Robust field-aligned global parametrization. *ACM Trans. Graph.* 33, 4 (2014).

M. Nieser, U. Reitebuch, and K. Polthier. 2011. CubeCover – Parameterization of 3D Volumes. *Computer Graphics Forum* 30, 5 (2011), 1397–1406.

David Palmer, David Bommes, and Justin Solomon. 2020. Algebraic Representations for Volumetric Frame Fields. *ACM Trans. Graph.* 39, 2 (2020).

Nico Pietroni, Marcel Campen, Alla Sheffer, Gianmarco Cherchi, David Bommes, Xifeng Gao, Riccardo Scateni, Franck Ledoux, Jean-Francois Remacle, and Marco Livesu. 2022. Hex-Mesh Generation and Processing: a Survey. https://arxiv.org/abs/2202.12670

Luca Pitzalis, Marco Livesu, Gianmarco Cherchi, Enrico Gobbetti, and Riccardo Scateni. 2021. Generalized adaptive refinement for grid-based hexahedral meshing. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–13.

François Protais, Maxence Reberol, Nicolas Ray, Etienne Corman, Franck Ledoux, and Dmitry Sokolov. 2020. Robust Quantization for Polycube Maps. (Dec. 2020). preprint.

Michael Rabinovich, Roi Poranne, Daniele Panozzo, and Olga Sorkine-Hornung. 2017. Scalable Locally Injective Mappings. *ACM Trans. Graph.* 36, 4 (2017).

A. Ramos and J.A. Simões. 2006. Tetrahedral versus hexahedral finite elements in numerical modelling of the proximal femur. *Medical Engineering & Physics* 28, 9 (2006), 916 – 924.

Nicolas Ray, Dmitry Sokolov, and Bruno Lévy. 2016. Practical 3D frame field generation. *ACM Trans. Graph.* 35, 6 (2016), 1–9.

Maxence Reberol, Alexandre Chemin, and Jean-Francois Remacle. 2019. Multiple Approaches to Frame Field Correction for CAD Models. In *Proc. 28th International Meshing Roundtable*.

Josep Sarrate Ramos, Eloi Ruiz-Gironés, and Francisco Javier Roca Navarro. 2014. Unstructured and semi-structured hexahedral mesh generation methods. *Computational Technology Reviews* 10 (2014), 35–64.

Teseo Schneider, Yixin Hu, Xifeng Gao, Jeremie Dumas, Denis Zorin, and Daniele Panozzo. 2019. A Large Scale Comparison of Tetrahedral and Hexahedral Elements for Finite Element Analysis. *arXiv:1903.09332* (2019).

Christian Schüller, Ladislav Kavan, Daniele Panozzo, and Olga Sorkine-Hornung. 2013. Locally injective mappings. *Computer Graphics Forum* 32, 5 (2013).

Feifei Shang, Yangke Gan, and Yufei Guo. 2017. Hexahedral mesh generation via constrained quadrilateralization. *PloS one* 12, 5 (2017).

Jason F Shepherd. 1999. *Interval matching and control for hexahedral mesh generation of swept volumes.* Master's thesis. Brigham Young University-Provo.

J. F. Shepherd and C. R. Johnson. 2008. Hexahedral mesh generation constraints. *Engineering with Computers* 24, 3 (2008), 195–213.

Justin Solomon, Amir Vaxman, and David Bommes. 2017. Boundary element octahedral fields in volumes. *ACM Trans. Graph.* 36, 4 (2017), 1.

Srinivas C. Tadepalli, Ahmet Erdemir, and Peter R. Cavanagh. 2011. Comparison of hexahedral and tetrahedral elements in finite element analysis of the foot and footwear. *Journal of Biomechanics* 44, 12 (2011), 2337 – 2343.

Kenshi Takayama. 2019. Dual Sheet Meshing: An Interactive Approach to Robust Hexahedralization. *Computer Graphics Forum* 38, 2 (2019), 37–48.

Timothy J Tautges. 2001. The generation of hexahedral meshes for assembly geometry: survey and progress. *Internat. J. Numer. Methods Engrg.* 50, 12 (2001), 2617–2642.

Y. Tong, P. Alliez, D. Cohen-Steiner, and M. Desbrun. 2006. Designing Quadrangulations with Discrete Harmonic Forms. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing (SGP '06)*. Eurographics Association, 201–210.

Amir Vaxman, Marcel Campen, Olga Diamanti, Daniele Panozzo, David Bommes, Klaus Hildebrandt, and Mirela Ben-Chen. 2016. Directional Field Synthesis, Design, and Processing. *Computer Graphics Forum* 35, 2 (2016).

Ryan Viertel, Matthew L Staten, and Franck Ledoux. 2016. *Analysis of Non-Meshable Automatically Generated Frame Fields.* Technical Report. Sandia National Lab.(SNL-NM), Albuquerque, NM (United States).

Erke Wang, Thomas Nelson, and Rainer Rauch. 2004. Back to elements-tetrahedra vs. hexahedra. In *Proceedings of the 2004 international ANSYS conference*.

Wei Wang, Yong Cao, and Tsubasa Okaze. 2021. Comparison of hexahedral, tetrahedral and polyhedral cells for reproducing the wind field around an isolated building by LES. *Building and Environment* 195 (2021), 107717.

Ofir Weber and Denis Zorin. 2014. Locally injective parametrization with arbitrary fixed boundaries. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–12.

Haiyan Wu, Shuming Gao, Rui Wang, and Jinming Chen. 2018. Fuzzy clustering based pseudo-swept volume decomposition for hexahedral meshing. *Computer-Aided Design* 96 (2018), 42–58.

Haiyan Wu, Shuming Gao, Rui Wang, and Mao Ding. 2017. A global approach to multi-axis swept mesh generation. *Procedia Engineering* 203 (2017), 414 – 426.

Paul Zhang, Josh Vekhter, Edward Chien, David Bommes, Etienne Vouga, and Justin Solomon. 2020. Octahedral Frames for Feature-Aligned Cross Fields. *ACM Trans. Graph.* 39, 3 (2020).

Jiaran Zhou, Changhe Tu, Denis Zorin, and Marcel Campen. 2020. Combinatorial construction of seamless parameter domains. *Computer Graphics Forum* 39, 2 (2020), 179–190.