DOI: 10.1111/cgf.13922
EUROGRAPHICS 2020 / U. Assarsson and D. Panozzo
(Guest Editors)

Volume 39 (2020), Number 2

# Combinatorial Construction of Seamless Parameter Domains

J. Zhou[1] C. Tu[1] D. Zorin[2] M. Campen[3]

[1]Shandong University, China
[2]New York University, USA
[3]Osnabrück University, Germany

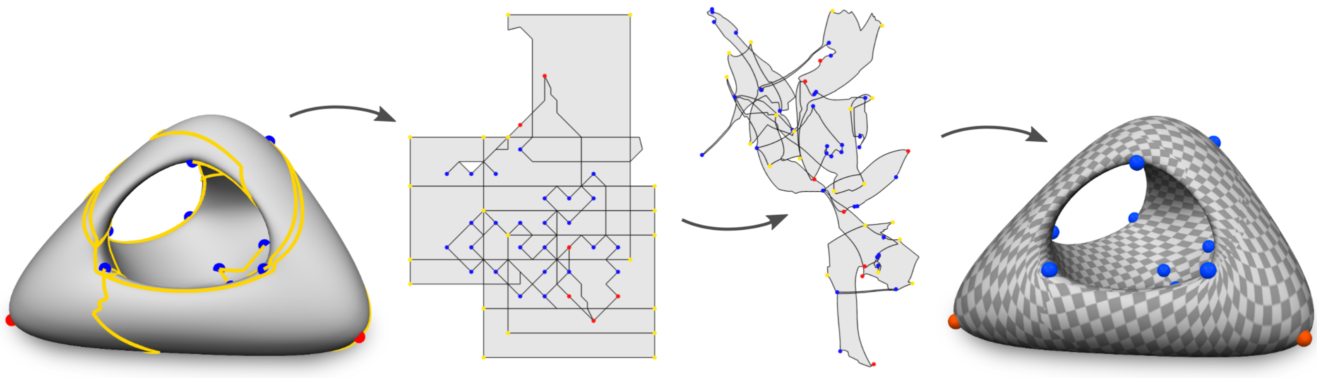

**Figure 1:** *Left: input surface (triangle mesh) with prescribed singularities (22 positions and indices, red: index* $1/4$*, blue:* $-1/4$*); a computed cut graph is shown in yellow. Center left: parameter domain (a* weakly self-overlapping polygon*) robustly constructed by our combinatorial method; yellow dots correspond to branch points of the cut graph. The boundary segments of this domain satisfy a number of hard constraints concerning relative lengths and angles. The cut input surface is mapped bijectively onto this domain with a constrained boundary map. By virtue of the specific domain boundary structure, this map is seamless. Center right: domain after optimizing this map for low distortion in a seamlessness-preserving manner. Right: grid texture pulled onto the surface using this map.*

**Abstract**

*The problem of seamless parametrization of surfaces is of interest in the context of structured quadrilateral mesh generation and spline-based surface approximation. It has been tackled by a variety of approaches, commonly relying on continuous numerical optimization to ultimately obtain suitable parameter domains. We present a general combinatorial seamless parameter domain construction, free from the potential numerical issues inherent to continuous optimization techniques in practice. The domains are constructed as abstract polygonal complexes which can be embedded in a discrete planar grid space, as unions of unit squares. We ensure that the domain structure matches any prescribed parametrization singularities (cones) and satisfies seamlessness conditions. Surfaces of arbitrary genus are supported. Once a domain suitable for a given surface is constructed, a seamless and locally injective parametrization over this domain can be obtained using existing planar disk mapping techniques, making recourse to Tutte's classical embedding theorem.*

**CCS Concepts**

• ***Computing methodologies*** $\rightarrow$ ***Computer graphics; Mesh models; Mesh geometry models;*** *Shape modeling;*

## 1. Introduction

We present a solution to the problem of constructing *seamless surface parametrizations* [MZ12] with prescribed singularities on surfaces of arbitrary topology without boundary. By construction, the resulting parametrizations are *locally injective*. This is a common prerequisite for the generation of valid *integer grid maps*, thus also quad meshes [BZK09] and spline surfaces [MAC19].

Specifically, we focus on the challenging problem of constructing suitable parameter domains for such parametrizations. Generally, the domain of such a seamless parametrization of a surface $M$ is a *weakly self-overlapping polygon* $\Omega$ in the plane, cf. Figs. 1 and 2, as discussed in detail in [WZ14]. The seamless parametrization can be viewed as a bijective map $f : M^c \leftrightarrow \Omega$, or (by an immersion of $\Omega$ in $\mathbb{R}^2$) as a locally injective map $f : M^c \rightarrow \mathbb{R}^2$. Here $M^c$ denotes the surface $M$ cut to disk topology along a cut graph.

While it is long known for which sets of prescribed singularities (numbers and indices/valences) locally injective seamless parametrizations (and in particular quad meshes) exist [JT73], actually computing them is a major challenge. Many works approach this problem by formulating it as a non-convex optimization problem, as detailed in Sec. 2. As a consequence, there is no guarantee that a solution is found, even when one is known to exist.

There are a few exceptions from this rule. For instance, for certain classes of singularity configurations, a linear program was shown to be suitable [AL15]. Furthermore, if strictly respecting the prescribed cone configuration is not essential, a surface partitioning technique in combination with multiple linear programs is a viable alternative [MPZ14].

Recently, an algorithm relying on convex optimization, based on discrete conformal mapping, was proposed in [CSZZ19]. This algorithm ensures that prescribed cones are respected in a fully general setting. However, while the problem is convex, it is nonlinear. Nonlinear optimization generally comes with potential numerical issues in practice, e.g., related to determining convergence or proper descent directions. In the specific conformal mapping context, additional challenges are due to the large scale variations common to conformal maps, as well as due to unresolved theoretical questions related to convergence.

In comparison, our method takes as input a surface together with prescribed singularity positions and indices, and constructs a suitable seamless parameter domain in a *combinatorial* manner—geometric computations and any kind of numerical optimization are taken into account only for non-crucial decisions (affecting initial quality but never validity). In this way we are not at risk of numerical issues affecting the output's validity. This is made possible by approaching the overall problem differently: instead of obtaining the domain as a byproduct of map optimization, we describe an explicit combinatorial domain construction. A parametrization over this domain can then be obtained in a second stage using existing planar disk mapping techniques [WZ14, SJZP19].

### Overall Idea

It is well-known that a genus $g > 0$ surface can be cut to a disk with $8g-4$ sides (cf. [Sti80, §1.3], *polygonal schema*). Moreover, the cut graph can be chosen in a way that exactly 4 cut curves meet at every branch point of the graph. We show an explicit combinatorial construction of a cone metric on this disk, i.e., a flat metric with a discrete set of points (cones) where curvature is concentrated. Such a metric induces a seamless parametrization (uniquely up to a rigid transformation). The constructed metric has the following properties: (a) the sides of the polygonal disk are straight, (b) pairs of sides corresponding to the same cut curve have equal lengths, (c) corners have right angles, so when the disk's paired sides are glued together, the branch points have a total angle of $2\pi$, i.e., are flat, (d) the number and curvature of cones in the interior of the disk matches an arbitrary prescribed configuration.

The cone metric is constructed based on a partition of the disk into quads, realized as unit squares. Hence, as a byproduct, the method yields a quad mesh connectivity for arbitrarily prescribed sets of (topologically admissible) extraordinary vertex valences.

We emphasize that our focus is on the validity, the seamlessness and local injectivity, of the parametrization. Quality optimization (e.g., distortion minimization) is delegated to existing injectivity-preserving map optimization methods to be applied subsequently. Our method's goal is to reliably provide a valid initialization.

## 2. Related Work

The problem of seamless surface parametrization with prescribed singularities has been considered in a long series of works, including [TACSD06, KNP07, BZK09, KMZ11, MZ13, MPZ14, CBK15, FLG15, CLW16, ESCK16, BCW17, ZCZ*18, FBT*18, HCW19]. Often, it is formulated as a numerical optimization problem—commonly a non-convex one. The challenges of non-convexity are dealt with, for instance, by omitting the non-convex constraints [KNP07, BZK09, KMZ11, MZ13, ESCK16, ZCZ*18] or by conservative convexification [Lip12, BCE*13, CBK15, BCW17, HCW19]. In the former case, results can be invalid (violating local injectivity requirements) [BZK09, EBCK13], in the latter case valid solutions are excluded—in the worst case leading to an infeasible problem.

Some methods, however, deviate from this common approach. It was shown that for certain special cases (in terms of surface genus and prescribed singularity configuration) the problem can be solved by convex linear programs [GY03, GGT06, AL15]. The method in [MPZ14] considers the general case; it likewise reduces the problem to (multiple, patch-wise) linear programs, based on a surface partitioning strategy. This strategy, unfortunately, cannot strictly preserve the prescribed singularity configuration in certain complicated cases. The recent method in [CSZZ19], which also handles the general case, does strictly respect the prescribed singularities. It employs a convex optimization problem—however, a nonlinear one (discrete conformal mapping), bringing about potential numerical challenges in practice. A similar idea is outlined in [CZK*19].

We note that in all these methods, the parameter domain is a byproduct of solving a parametrization optimization problem. Our method takes a different approach: in a first step, we explicitly focus on constructing a parameter domain (suitable for a seamless parametrization) without simultaneously finding or optimizing a parametrization. This distinct difference enables the robust combinatorial approach that we take.

## 3. Background & Approach

For a planar domain $\Omega$ and a continuous bijective map $f_{|\partial}$ between the boundary $\partial M^c$ of a disk topology surface $M^c$ (obtained by cutting the surface $M$ along a cut graph) and the domain boundary $\partial\Omega$, $f_{|\partial}$ can be extended to a map $f$ of the entire interior using, e.g., (discrete) dual-harmonic maps [WZ14, SJZP19].

The central goal of our work is the construction of a domain $\Omega$ for a given surface $M$, together with a boundary homeomorphism $f_{|\partial}$ between $\partial\Omega$ and $\partial M^c$, where $M^c$ is a cut version of $M$, using some cut graph $C$; then such an existing construction can be used to obtain a map $f$.

Note that there is a natural identification of boundary points of $\partial M^c$: along branches of the cut graph, there is a pairwise identification; at branch points, three or more boundary points are identified. Via a given map $f_{|\partial}$, this identification carries over to $\partial\Omega$.

**Domain Conditions**

The key challenge lies in the fact that the domain $\Omega$ to be constructed has to simultaneously satisfy multiple constraints—because we expect the resulting parametrization to be seamless and to exhibit exactly the prescribed singularities.

In particular, the following necessary and sufficient conditions on the domain boundary $\partial\Omega$ have to be satisfied for any seamless parametrization $f$ over $\Omega$ to exist:

(1) Inner angles around identified boundary points sum to prescribed values of the form $\frac{k\pi}{2}$, $k \in \mathbb{N}$.
(2) Identified boundary segments are isometric.
(3) The boundary is weakly self-overlapping.

Condition (1) ensures that the prescribed singularities with prescribed indices are respected; in combination with this, condition (2) ensures seamlessness [KNP07, BZK09, BCE*13]; condition (3) warrants that a bijective map onto the domain $\Omega$ (thus a locally injective map into the plane) exists, as shown in [WZ14].

**Domain Construction**

In a discrete, piecewise linear setting the boundary curve $\partial\Omega$ is a polygon. Techniques for the construction of polygons with prescribed corner angles [Har89, CR85] could be employed to satisfy condition (1), but they support neither the equal-length constraints of condition (2), nor the distinction of weakly self-overlapping from self-intersecting polygons required by condition (3).

[CSZZ19] describes a *padding* approach that we exploit here to reduce the problem to that of finding a polygon that satisfies conditions (1) and (3) only. This is made possible by using a specific class of cut graphs and a specific distribution of angles to the inner angles around identified boundary points along the cut graph. Under these circumstances, the padding approach allows modifying the polygon (or a collection of polygons) into one which satisfies condition (2) as well.

Our approach is based on generating parameter domains that satisfy conditions (1) and (3), such that through a combination with a discrete variant of the padding strategy we can ensure that all three conditions are satisfied. The boundary map $f_{|\partial}$ is then easily constructed via simple piecewise arc-length parametrization.

**Domain Interpretations**

Our approach is best understood by considering alternative interpretations of seamless parametrizations and their domains.

A seamless parametrization of a surface $M$ induces a cone metric with discrete holonomy ($k\pi/2$ turning numbers along any loop) [BCW17]. This metric is flat everywhere except at cone points, corresponding to singularities of the seamless parametrization. This metric uniquely defines the parametrization up to a rigid transformation and the (practically irrelevant) choice of cuts. The surface endowed with this cone metric defines a cone manifold, cf. Fig. 2 b. Ignoring the cuts, a seamless parametrization can be viewed as a homeomorphism between surface $M$ and this cone manifold.

Cutting the surface $M$ to a topological disk $M^c$ along a cut graph that avoids the singularities, and cutting the cone manifold along

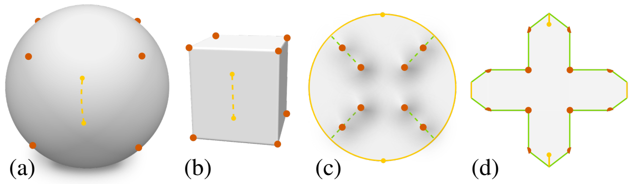


**Figure 2:** *Illustration of different interpretations of a seamless parametrization and its domain. (a) surface $M$ (here genus 0) with prescribed singularity points (orange dots). (b) a cone manifold with cones corresponding to these singularities; the configuration was chosen here in such a way that this manifold is embeddable in $\mathbb{R}^3$ isometrically (as a cube); in general, this is not possible. (c) cutting the cone manifold along a cut graph (yellow) yields a disk topology cone manifold D with boundary; note that its depiction here is non-isometric—D is not flat at the cones. (d) extending (green) the cut to include the cones yields a disk topology domain $\Omega$ that is flat everywhere, and can be embedded isometrically in the plane. In general, $\Omega$ is a weakly self-overlapping polygon.*

the image of this cut graph, one obtains a definition via a homeomorphism to a disk topology cone manifold $D$ with boundary, cf. Fig. 2 c. Across this boundary, seamless transition conditions between identified points are satisfied.

Further cutting the disk topology surface $M^c$ from the boundary to the singularities, and cutting $D$ along these cuts' images to the cones, one obtains the common definition via a homeomorphism to a domain $\Omega$ that is flat (as former cones are now on the boundary), cf. Fig. 2 d. This latter view is taken in most work on seamless surface parametrization for meshing and spline construction.

For our work the viewpoint considering the homeomorphism to the disk topology cone manifold $D$ (Fig. 2 c) is the most relevant and insightful one. Essentially, for a given surface cut graph, we explicitly construct a disk topology cone manifold (in form of a so-called *metapolygon*) such that its boundary structure is compatible with the combinatorial structure of this cut graph.

## 4. Overview

Before delving into the technical details, the following gives an advance summary of our method's algorithmic steps for orientation:

1. Cut input surface $M$ into one or more pieces of disk topology each, following Sec. 6.2.
2. Construct metapolygon per piece, with valences matching the prescribed singularities in the piece, as described in Sec. 5.2.
3. Subdivide each metapolygon to obtain meshes of quadrilaterals, following Sec. 5.4.
4. Perform discrete padding along the boundary of the mesh(es); padding widths computed as per Sec. 6.3.
5. Combine padded (therefore compatible) meshes to form one mesh $Q$, see Sec. 6.3.
6. Cut $Q$ to the cones, obtaining domain $\Omega$, and cut input surface $M$ to $M^c$ in a topologically identical manner, following Sec. 7.
7. Prescribe boundary mapping between $M^c$ and $\Omega$, exploiting the compatible cut, Sec. 8.
8. Extend boundary map to interior (equipped with unit square metric) using [WZ14], yielding a seamless parametrization.

The two central and technically most interesting operations in this are metapolygon construction (in step 2) and discrete padding (in step 4). Our main focus in the following is devoted to these. Their purpose in the context of our seamless parameter domain construction method is as follows:

- **Metapolygon Construction:** A metapolygon is a (combinatorial) polygon mesh of disk topology with a special boundary structure. We describe an algorithm that, given a list of singularity indices, creates a metapolygon whose subdivision yields a quad mesh with extraordinary vertices exactly corresponding to these singularity indices. Associating each abstract quad of this mesh with a unit square yields a disk topology cone manifold with prescribed cones.
- **Discrete Padding:** Adapting ideas from [CSZZ19], we show that it is possible to *pad* one or more (subdivided) metapolygons by additional layers of quads along their boundary and combine them so as to obtain a *seamless* metapolygon. This means that using several cuts it can be turned into a flat domain polygon that satisfies conditions (1), (2), and (3).

## 5. Metapolygon Construction

We start by defining a special kind of (combinatorial) polygon mesh. Via subdivision, it can be turned into a quad mesh, cf. Fig. 3, which we will make use of later. We emphasize that all constructions in this and the following section are combinatorial; mesh vertices do not have coordinates. Merely for purposes of illustration we embed these meshes in the plane in several of the figures.

**Definition 5.1 (Metapolygon)** A mesh $P$ of polygonal faces $p_i$, with all inner vertices of valence 4 and all boundary vertices of valence 1 or 2 we call a *metapolygon*. Valence here refers to the number of incident faces.

A *corner vertex* is a boundary vertex of valence 1, a *flat vertex* is a boundary vertex of valence 2, and a *concave vertex* is a boundary vertex of valence 3. A metapolygon does not have concave vertices, but these may occur at intermediate stages of its construction.

**Definition 5.2 (Meta-$k$-gon)** A metapolygon with $k$ corners (and any number of flat vertices) is called a meta-$k$-gon.

**Definition 5.3 (Excess)** A $k$-gon (polygon with $k$ vertices) $P$ is said to have valence $k$ and *excess* $e(P) = k-4$. A $k$-gon without excess (i.e. a 4-gon) is referred to as *regular*.

This notion extends to metapolygons via $e(P) = \sum_i e(p_i)$, the sum over the excesses of all polygons $p_i$ of $P$. Note that for the excess of a meta-$k$-gon $P$ it likewise holds $e(P) = k-4$ [PBJW14, §3.1]. We also use the definition $e(k) = k-4$. The empty polygon and the empty metapolygon are considered regular in the following.

### 5.1. Metapolygon Extension

In the following we define the key operation of our combinatorial construction. It takes a metapolygon containing a certain set of irregular faces and turns it into a metapolygon that contains exactly one additional irregular face, with a given valence $i$. More formally, let $\mathcal{I}(P)$ denote the unordered list of valences of the irregular faces of metapolygon $P$. Then the result shall be a metapolygon $Q$ with $\mathcal{I}(Q) = \mathcal{I}(P) \cup \{i\}$. The construction follows the idea of gluing an $i$-gon to the boundary of $P$ and filling any emerging concave corners with regular 4-gons so as to obtain a metapolygon again.

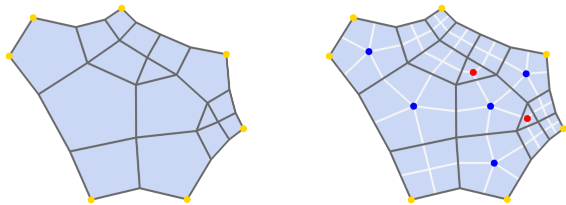

**Figure 3:** *Left: illustration of a* metapolygon. *Right: quad mesh obtained by subdividing the left metapolygon. Irregular vertices are marked red (valence 3) or blue (valence 5). In a (subdivided) metapolygon, each boundary vertex is either a* corner vertex *or a* flat vertex. *Here corner vertices are marked by yellow dots.*

The following definition makes this precise. In this, let $C(P')$ denote the clockwise cyclic sequence of non-flat boundary vertices of polygon mesh $P'$; for two such vertices $v$, $w$, let $d(v,w)$ denote the number of boundary edges between them in clockwise manner. The operation is illustrated in Fig. 4.

**Definition 5.4 (Metapolygon Extension)** The metapolygon extension $E(P,i)$ of a metapolygon $P$ by an $i$-gon is defined as follows:

- Input: meta-$k$-gon $P$ (possibly empty), $k > 0$, and integer $i > \max(1, 4-k)$, $i \neq 4$.
- Output: meta-$(k+i-4)$-gon $Q$, such that $\mathcal{I}(Q) = \mathcal{I}(P) \cup \{i\}$.

1. Glue an $i$-gon along one of its edges to an arbitrary boundary edge of meta-$k$-gon $P$. This yields polygon mesh $P'$. If $P$ is empty, $P'$ will be just the $i$-gon.
2. If there is a sub-sequence $v_0, v_1, v_2, v_3$ of $C(P')$ such that $v_1$ and $v_2$ are concave and $v_0$ and $v_3$ are corner:
   - glue a regular grid of 4-gons of size $d(v_1,v_2) \times \min(d(v_0,v_1), d(v_2,v_3))$ with three of its sides onto $P'$, aligning two of its corners with $v_1$ and $v_2$, respectively, yielding a new $P'$.
3. While there is a sub-sequence $v_0, v_1, v_2$ of $C(P')$ such that $v_1$ is concave and both, $v_0$ and $v_2$, are corner:
   - glue a regular grid of 4-gons of size $d(v_0,v_1) \times d(v_1,v_2)$ with two of its sides onto $P'$, aligning one of its corners with $v_1$, yielding a new $P'$ (ultimately $Q$).

**Theorem 5.5** Metapolygon extension $E(P,i)$ of a meta-$k$-gon $P$ is well-defined and its result is a meta-$(k+i-4)$-gon $Q = E(P,i)$ with $\mathcal{I}(Q) = \mathcal{I}(P) \cup \{i\}$.

*Proof*

- After step 1, $P'$ has only corner, flat, and concave (i.e., no valence 4+) boundary vertices; its inner vertices do not differ from $P$. At most two of the boundary vertices are concave (those adjacent to the glue edge). Let $0 \leq m \leq 2$ be this number of concave boundary vertices; then $|C(P')| = k+i-4+2m > 2m$ (due to $i > 4-k$).

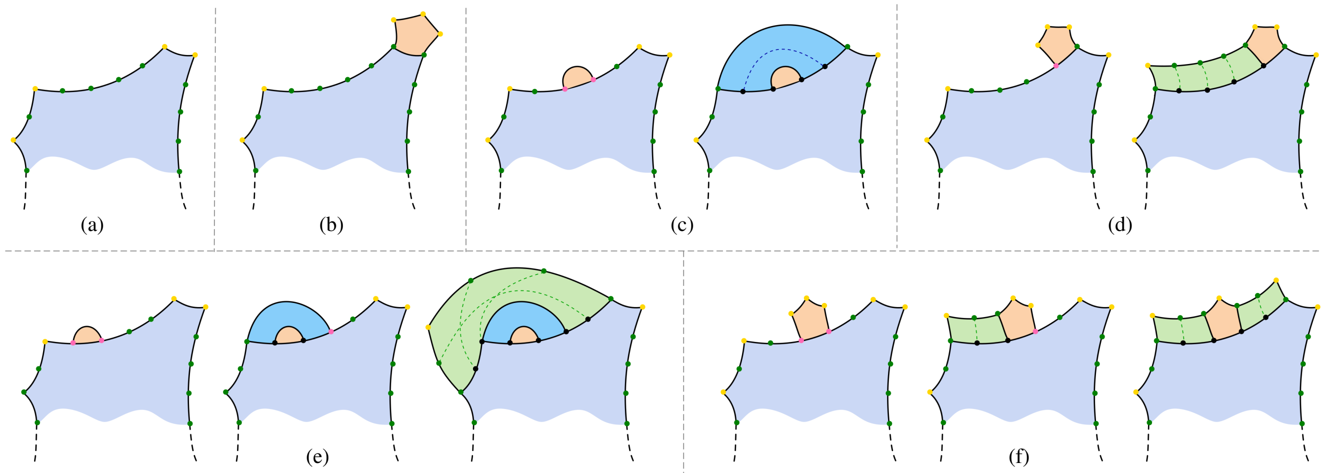


**Figure 4:** *Metapolygon Extension: illustration of cases. Polygons created in steps 1, 2, or 3 (Def. 5.4) are colored orange, blue, and green, respectively. Boundary vertex colors:* corner*: yellow;* flat*: green;* concave*: pink. Black dots are former boundary vertices that became regular inner vertices during extension. Dashed lines indicate the quad structure of attached regular grids. (a) input meta-k-gon P. (b) gluing an i-gon to a* corner-corner *boundary edge of P, no concave vertices are created; the construction terminates after step 1. (c) gluing a 2-gon to a* flat-flat *edge, two successive concave vertices emerge; because $d(v_0,v_1) = d(v_2,v_3)$ in step 2, one regular grid (blue) is attached in step 2. (d) gluing an i-gon to a* flat-corner *edge, one concave vertex emerges; one regular grid (green) is attached in step 3; (e) similar to (c), but with $d(v_0,v_1) \neq d(v_2,v_3)$ in step 2; a second regular grid (green) is attached in step 3; (f) gluing an i-gon ($i \neq 2$) to a* flat-flat *edge, two non-successive concave vertices emerge; two regular grids (green) are attached in step 3.*

- If step 2 applies, it reduces the number of corner and the number of concave vertices by 1 or by 2 (if $d(v_0,v_1) = d(v_2,v_3)$). All inner vertices introduced at step 2 are regular.
- Suppose 2 concave vertices are present in $C(P')$ before step 3. These vertices are not adjacent in $C(P')$: if after step 1 there are two concave vertices that are adjacent in $C$, there are (due to $|C(P')| > 4$) at least three corner vertices, such that step 2 applies (leading to less than 2 concave vertices). Therefore, step 3 can be applied, whether there are 1 or 2 concave vertices. Each iteration of step 3 reduces the number of corner and the number of concave vertices by 1; after the first iteration (if any) there is at most one concave vertex left—in which case another iteration is applicable. Hence upon termination, no concave boundary vertex is left. All newly introduced inner vertices are regular.

The result is a metapolygon. Because the only added non-4-gon is an $i$-gon, it specifically is a meta-$(k+i-4)$-gon with $\mathcal{I}(Q) = \mathcal{I}(P) \cup \{i\}$. $\square$

### 5.2. Metapolygons with Prescribed Valences

Repeated application of metapolygon extension, starting from an empty metapolygon, yields metapolygons containing a $k$-gon for any $k$ in a prescribed list of valences. This metapolygon can be converted to a quad mesh with the desired irregular vertices by one step of subdivision.

**Proposition 5.1** Given a sequence $S = (l_0, l_1, \ldots, l_{n-1})$ of $n$ integers $l_i > 1$, such that for all $j < n$ it holds $\sum_{i=0}^{j} e(l_i) > -4$, i.e. every prefix sum is greater than $-4$. Then repeated application of metapolygon extension allows to incrementally build a metapolygon $Q_{(n)}$ with $\mathcal{I}(Q_{(n)}) = S$, via $Q_{(i)} = E(Q_{(i-1)}, l_{i-1})$, $Q_{(0)} = \varnothing$.

*Proof* For sequences of length 0 this obviously holds. Assume it holds for sequences of length $j$; then $Q_{(j)}$ is a meta-$k$-gon with $k = 4 + \sum_{i=0}^{j-1} e(l_i) > 0$. Also: $\sum_{i=0}^{j} e(l_i) = k - 4 + e(l_j) = k - 4 + l_j - 4 > -4$. Hence, $l_j > 4 - k$, thus $Q_{(j+1)} = E(Q_{(j)}, l_j)$ can be constructed. $\square$

Note that, given an unordered list $L$ of valences, it can be ordered to form an admissible sequence $S = (l_0, l_1, \ldots)$ (i.e., such that for all $j < n$ it holds $\sum_{i=0}^{j} e(l_i) > -4$) iff $\sum_{l_i \in L} e(l_i) > -4$. For instance, start $S$ with all $l_i$ with $e(l_i) \geq 0$, followed by all with $e(l_i) < 0$.

### 5.3. Number of 4-gons

The size of the metapolygon obtained by repeated extension depends strongly on the choice of the glue edge in step (1) of the extension process. This choice affects the number of 4-gons that are used to fill up the concavities.

Let $l$ be the number of boundary edges of the metapolygon side onto which the next $i$-gon is glued. For $i > 4$, the number of 4-gons added in the process of metapolygon extension is $l-1$, cf. Fig. 4 b,d,f, regardless of where along this side the $i$-gon is glued.

For $i = 3$ and $i = 2$ this number, however, depends on the choice of glue edge. Let $x$ be the number of boundary edges between the glued $i$-gon and the nearest corner $c$, $0 \leq x \leq \lfloor (l-1)/2 \rfloor$. If $i = 3$, the number of required 4-gons is $x + (x+1)(l-x-1)$, so choosing a glue edge incident to a corner (i.e., $x = 0$) minimizes this number to $l-1$ (cf. Fig. 5). If $i = 2$, let $l' \geq 1$ be the number of edges on the other metapolygon side adjacent to corner $c$. The number of 4-gons is $x + (l-2x-1)(l'+1)$, so maximizing $x$ by choosing a glue edge centered between its two nearest corners minimizes the number of 4-gons to just $(l-1)/2$ or $l/2 + l'$, for $l$ odd or even (cf. Fig. 6).

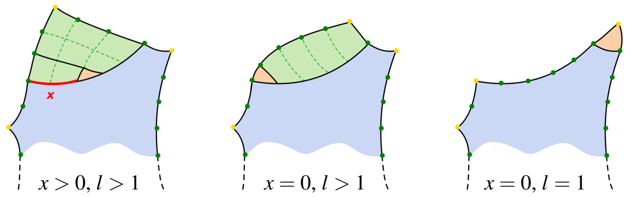


**Figure 5:** *Gluing a 3-gon to different boundary edges leads to more or less fill-up with 4-gons. Left: one grid of $x$ 4-gons and one grid of $(x+1)(l-x-1)$ 4-gons are glued to mesh $P'$. Middle: only one grid of $(l-1)$ 4-gons. Right: no 4-gons are needed.*

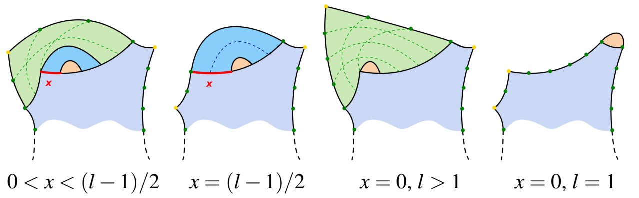


**Figure 6:** *Gluing a 2-gon to different boundary edges leads to more or less fill-up with 4-gons. Left: one grid of $x$ 4-gons and one grid of $(l-2x-1)(l'+1)$ 4-gons are glued to mesh $P'$. Center left: only one grid of $(l-1)/2$ 4-gons. Center right: only one grid of $(l-1)(l'+1)$ 4-gons. Right: no 4-gons are needed.*

### 5.4. Metapolygon to Cone Manifold

Subdividing each $k$-gon of a metapolygon into $k$ 4-gons yields a quad mesh, as illustrated in Fig. 3. For each $i$-gon, $i \neq 4$, of the metapolygon, this quad mesh contains a corresponding vertex of valence $i$; all other interior vertices are regular (valence 4). Endowing this quad mesh with a metric such that each quad is a unit square yields a disk topology cone manifold with boundary. Each valence $i$ vertex, $i \neq 4$, forms a cone of curvature $(4-i)\frac{\pi}{2}$.

## 6. Combinatorial Domain Construction

On a genus $g$ surface $M$, the total curvature of prescribed cones for a seamless parametrization must be $4\pi(1-g)$ (an implication of the Gauss-Bonnet theorem). For the list $L$ of corresponding valences (cone curvature $k\frac{\pi}{2}$ corresponds to valence $(4-k)$, cf. Sec. 5.4), this implies an excess $e(L) = 8g-8$. Therefore, for $g > 0$, valences of admissible cones can generally be ordered to satisfy the requirements of Prop. 5.1, i.e., we can construct a metapolygon for these. Note that the resulting metapolygon will be a meta-$(8g-4)$-gon. Subdividing this metapolygon and interpreting each quad as a unit square, we obtain a disk topology cone manifold $D$.

Assume we cut the surface $M$ to a disk topology surface $M^c$ using a cutgraph that has $4g-2$ branches; then $M^c$, like $D$, has $8g-4$ boundary sides (each corresponding to one side of a branch). It is then easy to establish a boundary bijection onto the boundary $\partial D$, side by side.

However, identified sides of $D$ (i.e. pairs of sides corresponding to the same cut graph branch) may have different lengths, due to different numbers of incident unit square quads. In other words, the quadrangulated cone manifold $D$ induced by the metapolygon does not, in general, glue to a closed *conforming* quad mesh of the same topology as $M$. A parametrization over this domain would therefore not induce a consistent metric on $M$ across the cut graph, in contrast to a seamless one; only for the broader class of similarity parametrizations this domain would be suitable [CZ17].

Note that in the process of metapolygon construction, we have no explicit control over the final side lengths. A very similar obstacle was described in [CSZZ19], where conformal map domains have analogous scale incompatibilities. In that work a *padding* technique is described to modify a parametrization using stretch and shift maps, equalizing the lengths of identified domain side.

Roughly speaking, our method can be viewed as following the same overall concept as that method, with two key differences: 1) the numerically challenging conformal mapping problem is replaced by our combinatorial metapolygon domain construction, 2) the padding idea is applied to combinatorially modify the domain in a discrete manner instead of continuously modifying a map.

### 6.1. Discrete Padding

Given a meta-$k$-gon with a side $s_i$ consisting of $l$ edges, one can glue a regular grid of $l \times m$ 4-gons along these edges ($m$-fold padding of side $s_i$). This yields a meta-$k$-gon with the number of edges of sides $s_{i-1}$ and $s_{i+1}$ increased by $m$. In this way the number of edges per side can be adjusted (though not independently) with the goal of achieving a state of pairwise equality. Fig. 7 shows an example of this operation applied to two sides.

As shown in [CSZZ19], due to the interdependencies, a state of pairwise length equality cannot be achieved in general. By working with two (three or four in special genus 2 cases) separate metapolygons, which are first padded and then glued to form one metapolygon, however, the desired state is achieved.

Alternatively, as we describe in Sec. 6.4, one can equivalently work with a single metapolygon by employing a slide operation in addition to the padding operation. This slide operation cuts one metapolygon into two pieces and recombines them differently.

### 6.2. Partitioning

To determine how many metapolygons are needed for which configuration, and how to distribute the prescribed singularities over these metapolygons, we directly follow the rules laid out in [CSZZ19, §4.2, §4.3]. Intuitively this can be pictured as partitioning the input surface into (commonly two) pieces $M_i^c$, and constructing a metapolygon $P_i$ for each piece following Sec. 5.2, considering those singularities that are contained in the piece. Afterwards, each metapolygon $P_i$ is subdivided to yield a quad mesh $Q_i$, as detailed in Sec. 5.4.

### 6.3. Padding Equation System

The numbers $\boldsymbol{w}$ of layers of quads that need to be glued to each side of the meshes $Q_i$ to match the lengths of identified sides can be computed by solving a linear system $A\boldsymbol{w} = \boldsymbol{b}$. Note that this is one global system, not an independent one per component. The system structure (and identification pattern) for each configuration of genus and singularities is given in [CSZZ19, Eq. (6), B.2, B.3].

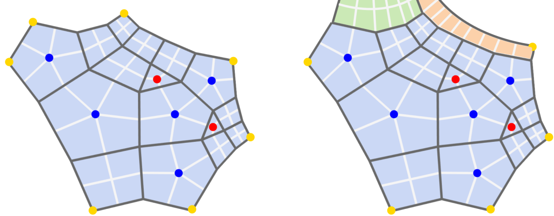

**Figure 7:** *Left: metapolygon $P$ subdivided to yield quad mesh $Q$. Right: top side of $Q$ padded by two layers of quads (green), top right side subsequently by one layer (orange). Notice that this padding affected the number of edges on four sides of the mesh.*

In contrast to that work, for our discrete combinatorial setting, we require an integer solution, $\boldsymbol{w} \in \mathbb{Z}^{|\boldsymbol{w}|}$. As $A$ and $\boldsymbol{b}$ are rational (in fact integer), the result $\boldsymbol{w}$ is rational. Let $d$ be the least common multiple of the denominators in $\boldsymbol{w}$, such that $d\boldsymbol{w}$ is integer.

Observe that, by linearity, $Ad\boldsymbol{w} = d\boldsymbol{b}$. Refining each quad in the meshes $Q_i$ into a $d \times d$ grid of quads yields a set of meshes $Q_i'$ for which $d\boldsymbol{b}$ is the right-hand side of the padding equation system, while $A$ depends only on the identification pattern, not the number of quads, thus remains unchanged. Hence, $d\boldsymbol{w}$ are equalizing *integer* padding numbers for these meshes $Q_i'$. In our experiments we have only encountered denominator 1, i.e., the results were generally in the integers right away. An interesting question is whether this is generally the case, due to the specific system structure. In any case, multiplication by the least common multiple of the denominators would yield an integer solution.

After the meshes $Q_i$ have been padded, they can be glued, according to the identification of their sides, to form one conforming quad mesh $Q$ of disk topology. Gluing *all* identified sides would yield a closed conforming quad mesh, but here we only glue a subset so as to yield a disk topology mesh $Q$.

### 6.4. Sliding

For the general case (genus 3+), where two pieces are used, we can alternatively construct one metapolygon (for the entire set of prescribed singularities), and split it afterwards. This approach is useful because it avoids the need to explicitly partition the surface into two pieces in a proper manner, which requires a relatively complex algorithm to ensure suitable subsets of the prescribed singularities lie in each piece [CSZZ19, §5.1]. Splitting the metapolygon (along a sequence of edges) is a much simpler combinatorial operation.

A metapolygon can be split into two metapolygons along any (combinatorially) straight sequence of interior edges that runs from boundary to boundary and does not self-intersect (purple in Fig. 8). Suitable edge sequences can be enumerated easily. Among these, we need to choose one which splits the metapolygon in such a way that the two resulting metapolygons have numbers $c_1, c_2$ of corners such that $c_i \bmod 4 \neq 0$. In this way the singularities are distributed between the two metapolygons according to the same rule employed by the *extra cut* used in [CSZZ19, Def. 4.2] to partition the surface into two suitable pieces.

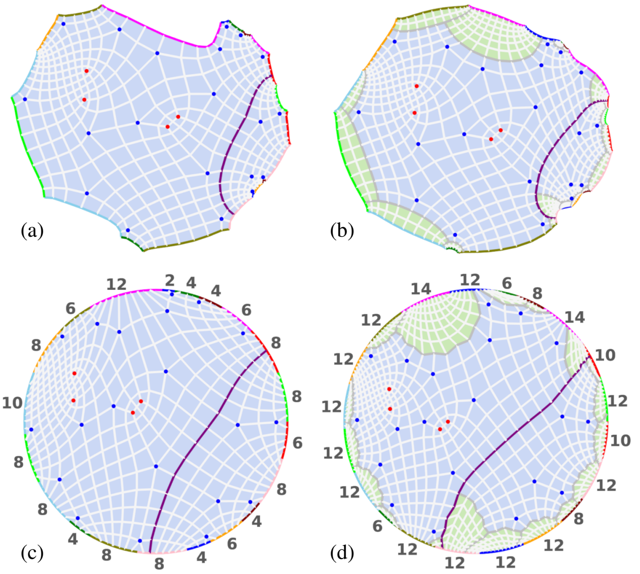


**Figure 8:** *Padding and sliding. (a) metapolygon $P$ subdivided to quad mesh $Q$. Pairs of identified sides corresponding to the same cut branch are marked in matching color. (b) after suitably padding all sides of $Q$ by regular grids of quads (light green) and sliding along an* extra cut *(purple, cf. Sec. 6.4) corresponding sides have equal lengths in terms of their number of edges. (c) and (d) show the meshes from (a) and (b) mapped to a disk, with uniform edge lengths along the boundary, so as to better illustrate the matching edge numbers (as labeled) per pair of sides after padding.*

Such a straight edge sequence always exists, unless all cones have valences that are multiples of 4 (in which case no split is necessary, cf. [CSZZ19, Prop. 4.3]). For instance, the maximal straight sequence of edges that contains the edge that the last $i$-gon, $i \bmod 4 \neq 0$, was glued onto during metapolygon construction, is one example of a valid choice, cf. Fig. 4.

Each of the two resulting metapolygons has one side corresponding to the split. As these two special sides will receive zero-padding in the process of Sec. 6.3 (in direct analogy to the above mentioned extra cut), they will be merged again when the two metapolygons are glued after padding. However, because different amounts of padding may have been applied on the adjacent sides, there will, in general, be some shift involved, cf. Fig. 8 d. Instead of a split, later followed by a merge, one can view this as a sliding operation. Effectively, we allow part of the metapolygon to slide (discretely) along a predetermined sequence of edges; this yields the additional degree of freedom required to make the padding problem feasible.

### 6.5. Genus 0

In the above we assumed $g > 0$. In the genus 0 case for the list $L$ of valences corresponding to the prescribed cones we have $e(L) = -8$, i.e., our metapolygon construction is not directly applicable. One can, however, split the set of cones into four subsets, each containing cones with a total excess of $-2$. Note that under the common assumption that no cones with valences $\leq 1$ are prescribed, such a partition is always possible. For each of the four pieces then a meta-2-gon can be constructed. In two pairs, these can be glued to form two meta-0-gons, i.e., metapolygons with only flat vertices on the boundary. If the numbers of edges along the sides to be glued do not match, the metapolygons (or their implied quad meshes) can be subdivided: assume the sides have $m$ and $n$ edges, respectively. Subdividing both meshes, replacing each quad with an $n \times n$ or $m \times m$ grid, respectively, yields two quad meshes which can be glued conformingly. In the same manner, the two meta-0-gon quad meshes can then be subdivided and glued to form a spherical mesh.

Cutting along one edge of this mesh yields a disk topology cone manifold (with two concave boundary points) as in Fig. 2 c.

## 7. Compatible Cutting

In Sec. 6 we assumed the surface $M$ is cut to a disk-topology surface $M^c$ using a cut graph that has $4g-2$ branches. In [CSZZ19] a so-called hole-chain cut graph is defined which has this number of branches, and whose side identification pattern implies a feasible padding problem (Sec. 6.3). After cutting the input mesh $M$ along this graph to a topological disk $M^c$, both $M^c$ and the padded metapolygon induced cone manifold $D$, have $8g-4$ sides. It remains to cut $D$ (in the form of quad mesh $Q$) to a flat domain $\Omega$ (cf. Fig. 2 c,d), and to compatibly extend the cut on $M^c$.

To this end, we choose a one-to-one correspondence between singularities prescribed on $M^c$ and cones in $D$ such that a singularity of index $\frac{k}{4}$ corresponds to an extraordinary vertex of valence $4-k$. In Sec. 9 we consider the problem of choosing geometrically reasonable correspondences; technically an arbitrary choice suffices. As both $M^c$ and $D$ have $8g-4$ corners, these can be brought into one-to-one correspondence, respecting cyclic order, as well.

In the quad mesh $Q$, we compute a discrete (edge-based) spanning tree of all singularities and one arbitrary flat boundary vertex, within the set of non-boundary edges. The set of non-boundary edges is connected by construction, hence such a tree exists.

To obtain a compatible spanning tree on $M^c$, for each *regular* branch vertex of the spanning tree of $Q$ we pick a distinct corresponding *non-singular* point on $M^c$, as well as a boundary point on the side of $M^c$ that corresponds (as per the corner correspondence) to the side of $Q$ that contains the spanning tree root. Then for each segment of the spanning tree of $Q$, we construct a path on $M^c$ between the two points corresponding to the segment's end points. These paths are chosen not to intersect each other. As $M^c$ with these paths removed remains a disk topology region throughout this process, such a path can always be found. When choosing a path, we need to ensure it reaches its endpoints in the proper sectors (as in [SAPH04, §4]). This is because we do not only need the spanning trees on $Q$ and on $M^c$ to be compatible as a graph, but as an embedded graph (i.e., their *rotation system* is relevant).

Cutting both, $Q$ and $M^c$, along the respective spanning trees, yields disk topology surfaces with all extraordinary vertices or singularities lying on the boundary—with corresponding entities in the same cyclic order around the boundary.

Using the unit square metric, the cut quad mesh $Q$ can be laid out isometrically in the plane, yielding the seamless parameter domain $\Omega$.

**Low Genus Special Cases** The hole-chain cutgraph is suitable for surfaces of genus 3 and higher. For genus 0 a trivial one-segment cut graph is sufficient, as discussed in Sec. 6.5. For the genus 1 and 2 case, suitable variations of the hole-chain cut graph are presented in [CSZZ19]. These cut the surface into 2 to 4 pieces, required to ensure feasibility of the padding problem.

## 8. Bijective Parametrization

We now consider how a bijective continuous map between the cut input surface $M^c$ and the domain $\Omega$ can be constructed such that it provides a locally injective seamless parametrization of $M$.

**Boundary Map** First, a bijective map between the boundaries $\partial M^c$ and $\partial\Omega$ is established. We already have a one-to-one correspondence of the corner vertices along the boundary. This corner map can be extended to a complete boundary map by mapping the sides (sequences of boundary edges) between corresponding pairs of neighboring corners according to (normalized) arc length.

One easily verifies that this boundary map is seamless: 1) domain sides of equal length are mapped to identified sides of $M^c$, using compatible, constant speed (arc length) parametrization; 2) due to the unit square metric, each side of $D$ forms a straight segment and the relative angle between any two sides' segments is a multiple of $\frac{\pi}{2}$. Furthermore, no additional singularities besides the intended ones are induced: 1) due to the sides of $D$ being straight, no curvature is induced along each branch of the cut graph; 2) due to adjacent sides of $D$ forming right angles at the corners, and the employed cut graph (Sec. 7) having branch points of degree 4 only, also at the branch points no cone is induced.

**Interior Map** Finally, we can extend the boundary map to the interior in a locally injective manner by direct application of the method described in [WZ14], with an efficient numerically robust variant of Tutte's embedding [SJZP19].

## 9. Geometric Guidance

In the previous sections we have, for clarity, described the algorithms for metapolygon construction, seamless domain construction, and compatible cutting in a purely combinatorial form. Choices of glued sides, glued edges, cut graphs, and singularity cuts are involved in these steps. We now describe how the geometry of the input surface and its prescribed singularities can be exploited as a guide to make these choices not arbitrarily but such that the distortion of the resulting initial parametrization is decreased.

### 9.1. Cut Graph

We construct the cut graph that cuts $M$ to $M^c$ as in [CSZZ19, §5.1], in the case of genus $g \geq 3$, out of $g$ discrete shortest loops and $2g-1$ shortest paths. As we found the provided implementation of the short handle loop method from [DFW13] to have some robustness limitations, we robustly compute the non-contractible non-intersecting loops using the tree-cotree-based algorithm of [EW05] instead. This algorithm is modified to avoid vertices marked singular; edges between two singular vertices are split to enable loops

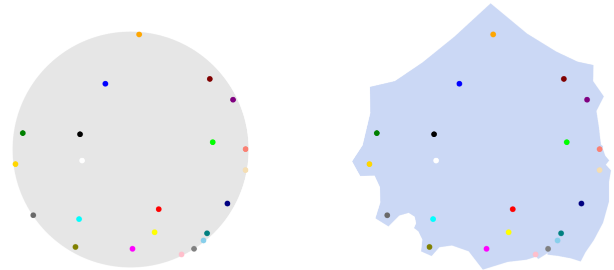

**Figure 9:** *Left: surface $M^c$ with prescribed singularity points (marked by colored dots). Right: the (subdivided) metapolygon constructed accordingly, with extraordinary vertices marked by correspondingly colored dots. Note the close similarity of the singularity and extraordinary vertex layouts.*

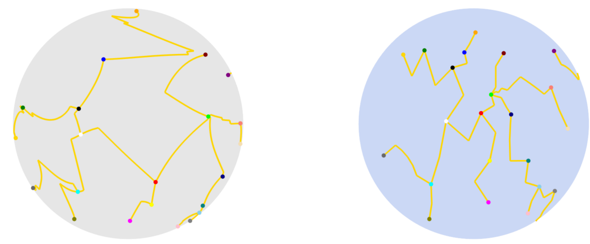

**Figure 10:** *Topologically compatible cut trees on the example from Fig. 9, on $M^c$ (grey) and cone manifold $D$ (blue). For visualization purposes, we embedded $D$, like $M^c$ on the left, in a unit disk. The jaggedness of paths is due to paths being sequences of edges, here in a low-resolution mesh.*

passing between them. For efficiency, we apply this algorithm to a sub-sampled set of vertices as base points, and greedily select the shortest non-intersecting loops from the resulting set. The $2g-1$ shortest paths connecting these loops are computed using Dijkstra's algorithm, again modified to circumvent singular vertices.

### 9.2. Metapolygon Construction

After partitioning the cut surface $M^c$ into pieces $M^c_i$, cf. Sec. 6.2, for each piece a metapolygon is constructed as follows. Note that in case sliding is used, cf. Sec. 6.4, we are dealing with just one piece in the general case of genus $\geq 3$.

We compute a map with minimal isometric distortion of the cut surface $M^c_i$ onto the unit disk. Fig. 9 left shows an example. The resulting positions of the prescribed singular vertices' images in this disk domain are of particular interest. Specifically, we aim to construct the metapolygon in such a way that it's irregular $i$-gon centers approximate the singularity layout. This initial singularity layout is taken into account in a twofold manner.

First, the ordered sequence $S$ of valences taken as input by the metapolygon construction algorithm from Sec. 5.2 is chosen based on this layout. Taking the center of the unit disk as reference point, we order the singularities (their corresponding valences) from closest to furthest. This is motivated by the fact that the metapolygon is constructed by extension from the center outwards. We call the resulting ordered sequence $S'$. Depending on the distribution of singularities, this order may violate the prefix-sum condition of 5.1. In this case, it can be reordered greedily by moving valences greater than 4 towards the front wherever necessary to satisfy the condition. As the total sum of indices, as well as the sum of indices for each part of the surface partition, respects the Poincaré-Hopf theorem by construction, the greedy approach is guaranteed to succeed.

Second, in each iteration of the metapolygon construction, i.e., in each instance $E(Q_i, l_i)$ of metapolygon extension, the side of the metapolygon $Q_i$ used for gluing the next $l_i$-gon is chosen based on the singularity layout in the unit disk. To this end, after each extension, we compute a geometric planar layout of the metapolygon as well. This is done in a free-boundary manner, using a least-squares conformal objective [LPRM02] for simplicity. The barycenters of $i$-gons that are already part of the metapolygon under construction, are pinned to the position of the corresponding singularities in the unit disk in that process. Fig. 9 right shows an example. Note that this layout does not need to have any strict properties, such as injectivity, for the overall algorithm to be correct, because it only serves as a guide for the choice of the next gluing side.

We glue the next $i$-gon to the side closest to the corresponding singularity in this unit disk embedding. In this way, the $i$-gon layout in the metapolygon mimics the singularity layout of the corresponding surface (piece). Algorithm 1 summarizes this procedure.

**Algorithm 1** Guided Metapolygon Construction

**Input:** valence sequence $S = (l_0, l_1, \dots, l_{n-1})$
unit disk singularity positions $C = (c_0, c_1, \dots, c_{n-1})$
**Output:** metapolygon $Q_{(n)}$
1: $Q_{(1)} \leftarrow E(\varnothing, l_0)$
2: **for** $i := 1$ to $n-1$ **do**
3: embed $Q_{(i)}$, pinning barycenters to $c_0, \dots, c_{i-1}$
4: find the boundary edge $e_i \in Q_{(i)}$ closest to $c_i$
5: $Q_{(i+1)} \leftarrow E(Q_{(i)}, l_i)$, using $e_i$ as glue edge
6: **end for**

### 9.3. Sliding

All straight edge sequences can easily be enumerated as they are uniquely defined by (any one of their) boundary vertices. Testing them for validity for the purpose of sliding is easy as well; checking for intersections and counting corners suffices.

Among all valid options, we choose one which implies the least number of additional quads added to the metapolygon in the padding process, to minimize the size of mesh $Q$ for efficiency. The number of additional quads for each valid option is easily obtained by solving the linear system of padding equations.

### 9.4. Cone Cuts

To construct the cut paths that connect the singularities and cone vertices to the boundary (Sec. 7) in a geometrically reasonable manner, we use geodesic paths. The following strategy proved beneficial in terms of achieving small total cut lengths on $M^c$ as well as on $Q$.

On the input mesh $M^c$ we connect the singularities by discrete, boundary-avoiding shortest paths in radial order, as determined by the unit disk embedding, starting from the center of the disk. In more detail, assuming the $i$ center-most singularities have already been connected, the $(i+1)$st is connected by the shortest possible paths (not intersecting the other paths) to any of these. Finally, the singularity closest to the boundary of the unit disk is connected to the closest boundary point by a shortest path.

On the quad mesh $Q$ these paths are replicated as discrete geodesic paths using Dijkstra's algorithm, constrained to the proper sectors. To prevent paths from blocking subsequent paths, we introduce Steiner vertices in $Q$ through edge splits where necessary, cf. [SAPH04]. To simplify implementation, $Q$ can be triangulated (splitting each quad into two right triangles). This is not an issue as ultimately only its boundary is of relevance, cf. Sec. 8.

We observed that additional improvement (in terms of the total lengths of resulting cuts) can be achieved by constructing not one global spanning tree, but a separate one per sub-metapolygon of the partition (cf. Sec. 6.2), with separate root points on the boundary. In the case of sliding being employed, two separate trees are constructed for the two sides of the slide edge sequence. The improvement can be attributed to the fact that the padding dictates how the sub-metapolygons are combined to form the global metapolygon (or how much sliding occurs), potentially making $Q$ and $M^c$ differ significantly geometrically along the interfaces. Cut paths that cross these interfaces can be short in $Q$ but potentially very long in $M^c$ or vice versa. Such paths are avoided when using multiple cut trees. Fig. 10 illustrates the resulting cut trees on an example.

## 10. Results

We applied our method to models from the dataset of [MPZ14], together with the cone prescription provided in that dataset. Table 1 reports the statistics for the 20 topologically most complex models from this dataset. In Fig. 11, we visualize the seamless domains $\Omega$ constructed by our method together with the corresponding models.

The extension of the boundary over the entire domain, as described in Sec. 8, can be done using a previous method [WZ14]. The implementation we tested is robust but not particularly efficient; while for the smaller cases it takes seconds, for the largest models (when $M^c$ and $\Omega$ combined have millions of vertices) more than an hour may be necessary.

The domains constructed by our method, together with the initial seamless parametrizations over these domains, can serve as valid initialization for optimization, e.g., for low parametric distortion. Fig. 1 shows an example of this. We note, however, that our experiments revealed that existing parametrization optimization techniques, such as [RPPSH17], have significant limitations when starting from initial parametrizations of high distortion in combination with low mesh quality (in the sense of badly shaped elements). For some of the initial parametrizations obtained as described in Sec. 8, the optimization converges very slowly, requiring hundreds or thousands of iterations. In some cases, the feasible step size becomes so small that limited numerical precision leads to a premature halt. One can conclude that the exploration of map optimization techniques that are more robust to issues of discretization and numerics is an important field for future work.

| | Input Model $M$ | | | Quad Mesh $Q$ | | Time |
|---|---|---|---|---|---|---|
| | cones | genus | faces | #F init | #F pad | |
| helmet | 9 | 3 | 1K | 220 | 1452 | 0.13s |
| genus3 | 22 | 3 | 13K | 348 | 576 | 0.25s |
| holes3 | 24 | 3 | 11K | 452 | 1120 | 0.3s |
| master_cyl. | 32 | 3 | 100K | 608 | 3284 | 1.9s |
| block | 46 | 3 | 4K | 2568 | 2288 | 0.7s |
| rolling_stage | 52 | 7 | 100K | 652 | 36620 | 6.5s |
| fertility | 60 | 4 | 27K | 2332 | 3952 | 1.2s |
| carter | 64 | 7 | 100K | 1048 | 7172 | 5s |
| botijo | 70 | 5 | 82K | 4K | 116K | 9s |
| chair | 98 | 7 | 100K | 8K | 56K | 8s |
| casting | 119 | 9 | 36K | 5K | 118K | 10s |
| elephant | 125 | 3 | 50K | 15K | 63K | 12s |
| pegaso | 131 | 6 | 30K | 19K | 138K | 19s |
| heptoroid | 140 | 22 | 100K | 2K | 119K | 15s |
| neptune | 212 | 3 | 105K | 84K | 44K | 82s |
| oil_pump | 212 | 4 | 100K | 224K | 266K | 252s |
| dancing_chil. | 212 | 8 | 100K | 59K | 365K | 85s |
| seahorse2 | 216 | 8 | 100K | 306K | 2203K | 375s |
| bozbezbozzel | 305 | 5 | 100K | 1276K | 2213K | 1574s |
| thai_statue | 366 | 3 | 80K | 3890K | 2390K | 7403s |

**Table 1:** *Result statistics. For each model, the number of prescribed cones, the genus, and the number of triangles is listed. The number of quads of the constructed metapolygon(s) before padding, and the number of additional quads due to padding is shown. The complete run time of our geometrically guided combinatorial construction of the seamless parameter domain $\Omega$ is given in the last column.*

As can be observed in Table 1, the size of the resulting domains can be quite large in terms of the number of quads. Fortunately, an explicit representation of these individual quads is not essential: any coarse tessellation (e.g., using one large rectangular face instead of a grid of many quads for each side's padding) is sufficient, as the mapping method [WZ14] only relies on the boundary information.

Note that the initial seamless parametrization obtained using our method are integer grid maps [BLP*13]: the translational component of the transitions across cuts is discrete, by our construction based on unit squares. The scale of this discreteness, however, cannot be controlled explicitly in our method. Nonetheless, the initial maps created by our method are valid input for quantization methods [CBK15, LCBK19] yielding integer grid maps with user-controllable scale.

## 11. Limitations & Future work

While due to the combinatorial nature of our domain construction there are no robustness issues in this part of the method, the current strategy clearly leaves room for improvements in terms of performance as well as initial result quality, as discussed in the following.

In the process of padding, the complexity of the metapolygon quad mesh sometimes hardly changes, but sometimes it increases by one or two orders of magnitude. A better understanding of the influence of the metapolygon structure on the implied amount of padding could help in preventing such cases.

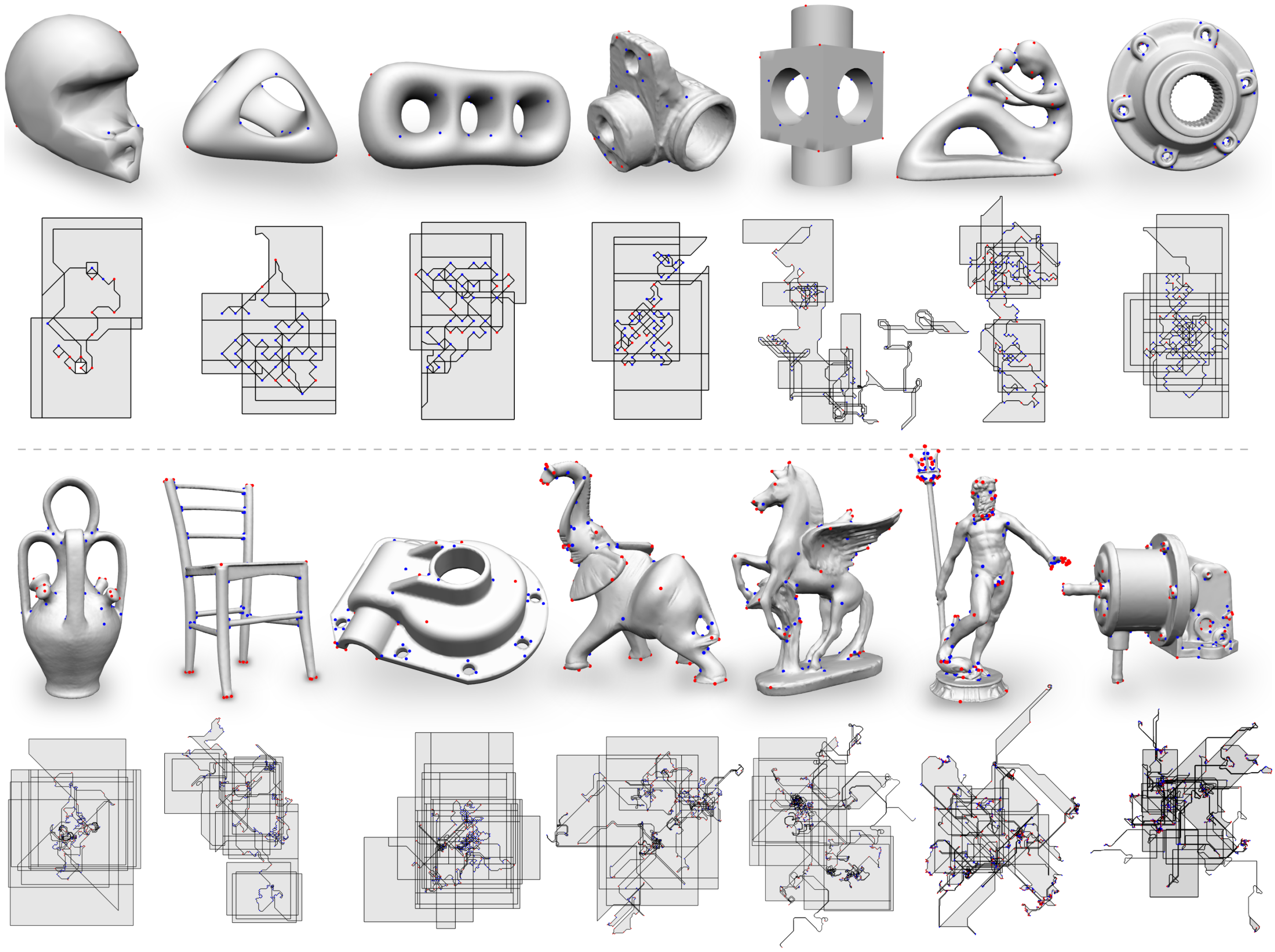

**Figure 11:** *Some example models with prescribed cones (red: valence 3 or lower, blue: valence 5 or higher). Below each model the seamless parameter domain constructed by our method is shown. Colored dots mark the boundary points corresponding to cone points. Note that axis-aligned straight boundary segments correspond to the branches of the cut graph that cuts $M$ to $M^c$; the rest of the boundary corresponds to the extended cut that connects the cones to the cut graph. The large number of diagonal segments is due to quad-triangle splitting (Sec. 9.4.)*

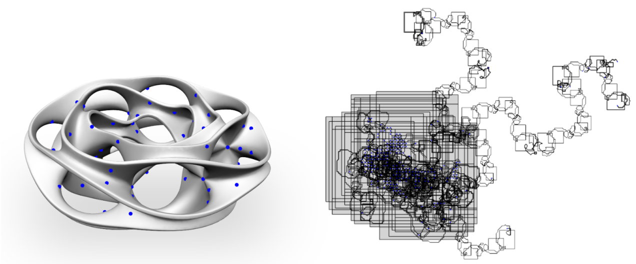

**Figure 12:** *For models with high genus (here 22) and a large number of prescribes cones (here 140), the constructed domain can become very complex—in size (due to a large number of elements forming the final padded metapolygon) as well as in shape (depending on the choice of cuts and their interplay).*

Another way to significantly reduce the size or complexity of the metapolygon domain could be the gluing of multiple $i$-gons to a metapolygon side, instead of generally filling everything up with 4-gons after each step. A challenge lies in properly treating all the possible cases that can occur depending on which combinations of singularity indices are involved. This could prevent extreme cases like the last example in Table 1, where the final domain has around 6 million quads.

The construction of cuts in such a way that they are topologically compatible and geometrically well-behaved on the surface and, at the same time, on the domain, is another area that deserves further attention. Like similar compatible embedded graph constructions in previous work, e.g. [SAPH04], ours follows a greedy strategy. It is easy to find cases where such greedy strategies yield highly sub-optimal results, i.e. overly long, badly shaped paths (cf. Fig. 12). This can significantly affect performance as the number of required Steiner vertices, and therefore the mesh complexity, can grow and the distortion of the initial map can be high.

As can be observed in Table 1, our current experimental implementation is slow for some complex models, given the relative simplicity of the algorithmic components, leaving room for optimizations. Additionally, an interesting avenue for future work is the investigation of multiresolution techniques in the general context of seamless parametrization. When a model has tens or hundreds of thousands of faces (like many of our test models) but only tens or hundreds of prescribed singularities, the high mesh resolution is not of high importance for the initial map computation.

Finally, as mentioned above, more robust, more flexible, less tessellation-dependent parametrization optimization techniques (supporting constraints, e.g., for seamlessness) would be of high value, in the present context and beyond.

## Acknowledgements

This work was supported by National Key Research and Development Project (No. 2017YFB1002603), NSFC Project (61772318). The authors thank Hanxiao Shen for his practical help with parametrization and optimization.

## References

[AL15] AIGERMAN N., LIPMAN Y.: Orbifold tutte embeddings. *ACM Trans. Graph. 34*, 6 (2015), 190:1–190:12. 2

[BCE*13] BOMMES D., CAMPEN M., EBKE H.-C., ALLIEZ P., KOBBELT L.: Integer-grid maps for reliable quad meshing. *ACM Trans. Graph. 32*, 4 (2013), 98:1–98:12. 2, 3

[BCW17] BRIGHT A., CHIEN E., WEBER O.: Harmonic global parametrization with rational holonomy. *ACM Trans. Graph.* 36,4(2017). 2, 3

[BLP*13] BOMMES D., LÉVY B., PIETRONI N., PUPPO E., SILVA C., TARINI M., ZORIN D.: Quad-mesh generation and processing: A survey. In *Computer Graphics Forum* (2013). 10

[BZK09] BOMMES D., ZIMMER H., KOBBELT L.: Mixed-integer quadrangulation. *ACM Trans. Graph. 28*, 3 (2009), 77. 1, 2, 3

[CBK15] CAMPEN M., BOMMES D., KOBBELT L.: Quantized global parametrization. *ACM Trans. Graph. 34*, 6 (2015), 192. 2, 10

[CLW16] CHIEN E., LEVI Z., WEBER O.: Bounded distortion parametrization in the space of metrics. *ACM Trans. Graph. 35*, 6 (2016). 2

[CR85] CULBERSON J., RAWLINS G.: Turtlegons: generating simple polygons for sequences of angles. In *Proc. Symp. Comp. Geom.* (1985), pp. 305–310. 3

[CSZZ19] CAMPEN M., SHEN H., ZHOU J., ZORIN D.: Seamless parametrization with arbitrary cones for arbitrary genus. *ACM Trans. Graph. 39*, 1 (2019). 2, 3, 4, 6, 7, 8

[CZ17] CAMPEN M., ZORIN D.: Similarity maps and field-guided t-splines: a perfect couple. *ACM Trans. Graph. 36*, 4 (2017), 91. 6

[CZK*19] CHEN W., ZHENG X., KE J., LEI N., LUO Z., GU X.: Quadrilateral mesh generation i: Metric based method. *Computer Methods in Applied Mechanics and Engineering 356* (2019), 652–668. 2

[DFW13] DEY T. K., FAN F., WANG Y.: An efficient computation of handle and tunnel loops via reeb graphs. *ACM Trans. Graph. 32*, 4 (2013), 32:1–32:10. 8

[EBCK13] EBKE H.-C., BOMMES D., CAMPEN M., KOBBELT L.: Qex: Robust quad mesh extraction. 168:1–168:10. 2

[ESCK16] EBKE H.-C., SCHMIDT P., CAMPEN M., KOBBELT L.: Interactively controlled quad remeshing of high resolution 3d models. *ACM Trans. Graph. 35*, 6 (2016), 218:1–218:13. 2

[EW05] ERICKSON J., WHITTLESEY K.: Greedy optimal homotopy and homology generators. In *Proc. ACM-SIAM Symp. on Discrete Algorithms* (2005), pp. 1038–1046. 8

[FBT*18] FANG X., BAO H., TONG Y., DESBRUN M., HUANG J.: Quadrangulation through morse-parameterization hybridization. *ACM Trans. Graph. 37*, 4 (2018), 92. 2

[FLG15] FU X.-M., LIU Y., GUO B.: Computing locally injective mappings by advanced mips. *ACM Trans. Graph. 34*, 4 (2015). 2

[GGT06] GORTLER S. J., GOTSMAN C., THURSTON D.: Discrete one-forms on meshes and applications to 3d mesh parameterization. *Computer Aided Geometric Design 23*, 2 (2006), 83 – 112. 2

[GY03] GU X., YAU S.-T.: Global conformal surface parameterization. In *Proc. Symp. Geometry Processing 2003* (2003), pp. 127–137. 2

[Har89] HARTLEY R. I.: Drawing polygons given angle sequences. *Information processing letters 31*, 1 (1989), 31–33. 3

[HCW19] HEFETZ E. F., CHIEN E., WEBER O.: A subspace method for fast locally injective harmonic mapping. In *Computer Graphics Forum* (2019), vol. 38, pp. 105–119. 2

[JT73] JUCOVIČ E., TRENKLER M.: A theorem on the structure of cell-decompositions of orientable 2-manifolds. *Mathematika 20* (1973). 2

[KMZ11] KOVACS D., MYLES A., ZORIN D.: Anisotropic quadrangulation. *Computer Aided Geometric Design 28*, 8 (2011), 449 – 462. 2

[KNP07] KÄLBERER F., NIESER M., POLTHIER K.: QuadCover: Surface Parameterization using Branched Coverings. *Computer Graphics Forum 26*, 3 (2007), 375–384. 2, 3

[LCBK19] LYON M., CAMPEN M., BOMMES D., KOBBELT L.: Parametrization quantization with free boundaries for trimmed quad meshing. *ACM Trans. Graph. 38*, 4 (2019), 51:1–51:14. 10

[Lip12] LIPMAN Y.: Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph. 31*, 4 (2012), 108:1–108:13. 2

[LPRM02] LÉVY B., PETITJEAN S., RAY N., MAILLOT J.: Least squares conformal maps for automatic texture atlas generation. In *ACM Trans. Graph.* (2002), vol. 21, ACM, pp. 362–371. 9

[MAC19] MARINOV M., AMAGLIANI M., CHARROT P.: Boundary conforming mesh to T-NURCC surface conversion. *Computers & Graphics 82* (2019), 95 – 105. 1

[MPZ14] MYLES A., PIETRONI N., ZORIN D.: Robust field-aligned global parametrization. *ACM Trans. Graph. 33*, 4 (2014). 2, 10

[MZ12] MYLES A., ZORIN D.: Global parametrization by incremental flattening. *ACM Trans. Graph. 31*, 4 (2012), 109. 1

[MZ13] MYLES A., ZORIN D.: Controlled-distortion constrained global parametrization. *ACM Trans. Graph. 32*, 4 (2013), 105. 2

[PBJW14] PENG C.-H., BARTON M., JIANG C., WONKA P.: Exploring quadrangulations. *ACM Trans. Graph.*, 1 (2014). 4

[RPPSH17] RABINOVICH M., PORANNE R., PANOZZO D., SORKINE-HORNUNG O.: Scalable locally injective mappings. *ACM Trans. Graph. 36*, 4 (2017). 10

[SAPH04] SCHREINER J., ASIRVATHAM A., PRAUN E., HOPPE H.: Inter-surface mapping. In *ACM Trans. Graph.* (2004), vol. 23, pp. 870–877. 8, 10, 11

[SJZP19] SHEN H., JIANG Z., ZORIN D., PANOZZO D.: Progressive embedding. *ACM Trans. Graph. 38*, 4 (2019), 32. 2, 8

[Sti80] STILLWELL J.: *Classical topology and combinatorial group theory*. Springer Verlag, 1980. 2

[TACSD06] TONG Y., ALLIEZ P., COHEN-STEINER D., DESBRUN M.: Designing quadrangulations with discrete harmonic forms. *Symposium on Geometry Processing* (2006), 201–210. 2

[WZ14] WEBER O., ZORIN D.: Locally injective parametrization with arbitrary fixed boundaries. *ACM Trans. Graph. 33*, 4 (2014), 75:1–75:12. 1, 2, 3, 8, 10

[ZCZ*18] ZHOU J., CAMPEN M., ZORIN D., TU C., SILVA C. T.: Quadrangulation of non-rigid objects using deformation metrics. *Computer Aided Geometric Design 62* (2018), 3–15. 2