Designing Quadrangulations with Discrete Harmonic Forms

Y Tong, Pierre Alliez, David Cohen-Steiner, Mathieu Desbrun

Eurographics Symposium on Geometry Processing (2006)
Konrad Polthier, Alla Sheffer (Editors)

Designing Quadrangulations with Discrete Harmonic Forms

Y. Tong[1] P. Alliez[2] D. Cohen-Steiner[2] M. Desbrun[1]
[1]Caltech [2]INRIA Sophia-Antipolis, France

Abstract

We introduce a framework for quadrangle meshing of discrete manifolds. Based on discrete differential forms, our method hinges on extending the discrete Laplacian operator (used extensively in modeling and animation) to allow for line singularities and singularities with fractional indices. When assembled into a singularity graph, these line singularities are shown to considerably increase the design flexibility of quad meshing. In particular, control over edge alignments and mesh sizing are unique features of our novel approach. Another appeal of our method is its robustness and scalability from a numerical viewpoint: we simply solve a sparse linear system to generate a pair of piecewise-smooth scalar fields whose isocontours form a pure quadrangle tiling, with no T-junctions.

1. Introduction

Partitioning a surface into quadrilateral regions is a common requirement in computer graphics, computer aided geometric design and reverse engineering. Such quad tilings are amenable to a variety of subsequent applications due to their tensor-product nature, such as B-spline fitting, simulation with finite elements or finite differences, texture atlasing, and addition of highly detailed modulation maps. Quad meshes are particularly useful in modeling as they aptly capture the symmetries of natural or man-made geometry, allowing artists to design simple surfaces using a quite intuitive placement of quad elements. Automatically converting a triangulated surface (issued from a 3D scanner for instance) into a quad mesh is, however, challenging. Stringent topological conditions make quadrangulating a domain or a surface a rather constrained and global problem [Ede00]. Application-dependent meshing requirements such as edge orthogonality, alignment of the elements with the geometry, sizing, and mesh regularity add further hurdles. In this paper, we propose a framework for quadrangle tiling of arbitrary triangulated surfaces that allows for a precise user-guided control over the design of the final pure-quad mesh. We show how to use discrete harmonic forms to solve for two piecewise-smooth scalar fields such that their respective isocontours create a mesh with well-shaped quadrangles at geometrically pertinent edge locations.

1.1. Previous Work

Due to their wide appeal in various communities, quad meshes have been the subject of a large number of papers presenting different algorithms for the generation of isotropic or anisotropic quad elements. Comprehensive reviews, found for instance in [ACSD*03, BMRJ04, AUGA05, DKG05], hint at a need for algorithms offering more *control* on the mesh regularity, as well as on the shape, size and alignment of the mesh elements with geometric or semantic features. A clustering-based method presented in [BMRJ04] manages to limit the number of extraordinary vertices in the final mesh, without guaranteeing the location of singularities. A recent Morse-theoretic approach in [DBG*06] provides improved results, but the modified relaxation [KLS03] involved in this method still allows no control over design, resulting in singularities at conspicuous places and elements of arbitrary shapes. Another technique allows fully regular quadrilateral meshes (except along a seam) through the use of holomorphic discrete 1-forms [GY03]. Unfortunately, this holomorphic requirement leaves little control over the local alignment of the mesh elements and creates potentially large area distortion, even after optimization [JWYG04]. A recent technique proposes a radically different approach to

Figure 1: *Scanned Hand. From a triangulated surface and a set of line singularities assembled into a singularity graph, our technique solves a linear, modified Laplace equation to get two potentials (top); The pair of 1-forms associated to the potential differentials is specified as either regular, reverse or switch across* singularity lines *(center). An isocontouring of these potentials results in a pure-quad mesh with non-integer index singularities capturing the geometry (see close-up, right), and no T-junction (bottom).*

conformal parameterization with *arbitrary* cone singularities [KSS06]: distortion is concentrated at carefully chosen places so as to allow better, global control of area distortion and hence over mesh sizing. Unfortunately, this non-linear method cannot directly control alignment with features, and/or guarantee proper matching of quads through patch boundaries.

A recent trend towards a better control of alignment focuses on vector field topology. A first approach proposed in [ACSD*03, MK04] consists in tracing *curvature lines*, thereby enforcing proper alignment of the mesh edges while creating a natural quad-dominant network. The placement of these lines are based on local decisions, resulting in numerous hanging lines all over the mesh: T-junctions and poor regularity of the mesh result from this greedy line selection. When targeting higher mesh regularity, a better approach defines these lines as isolevels and steepest descents of a *global* potential [DKG05]. As a result of this type of contouring, the lines are either closed curves (so-called isoparametric flow lines), or streamlines (gradient flow lines) obtained by numerical integration, leading to less T-junctions and irregularity on the final quad mesh. This method allows some design control through user-defined selection of a number of local extrema of the potential (be they points, or even polylines). However, each local extremum corresponds to an index 1 in the gradient field of the potential; because of the Hopf-Poincaré index theorem, this means that a number of *other* singularities (most likely saddles, of index -1) will be consequently created too, as the indices of all singularities of the vector field must sum up to the Euler characteristic of the surface. Therefore, design control does not scale nicely as each additional constraint increases shape distortion of the tiles on the rest of the surface.

Finally, Ray *et al.* [RLL*05] recently introduce another contouring technique performing a non-linear optimization of periodic parameters to best align directions along two given orthogonal vector fields, offering more freedom on the type of singularities than any previous approach. In particular, indices of type $1/2$ and $1/4$ can be introduced, allowing a satisfactory balance between area distortion and alignment control. However, even after a curl-correction step modulating the norm of the vector field to minimize the number of point singularities, this method does not provide *direct control* over the placement of singularities (*i.e.*, irregular vertices in the final remesh). Conspicuous imperfections may appear in the final mesh at seemingly random places.

1.2. Approach and Contributions

The recently introduced use of isolines as a basis for remeshing seems particularly appealing from a practical point of view: it naturally privileges regularity of the resulting quad mesh, and is numerically more robust than the use of streamlines as it alleviates the need for numerical integration. Isovalues can eventually be changed in order to adapt spacing between isolines and aspect ratios of the resulting quads. Therefore, we propose to design an algorithm to find two piecewise continuous scalar functions with harmonicity properties, such that *their respective contouring provide the final pure-quad mesh with no T-junctions*.

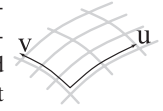
Based on the solid foundations of discrete differential forms, our technique improves upon previous methods in several unique respects. In particular, we provide control not only on the position of singularities and their (possibly fractional) indices, but also the way these singularities are interconnected in the final mesh. This information on the topological structure of the output mesh is encoded in what we call a *singularity graph*, specified by the user or semi-automatically computed using the curvature tensor of the surface. This singularity graph is also a convenient way to specify preferred directions for the quads depending on their locations. Moreover, the core of our algorithm relies on novel extensions of the *well-known cotangent formula* that enriches the space of discrete harmonic functions: we thus stay within the framework of linear algebra, avoiding non-linear minimization required in [RLL*05, KSS06] that can impair scalability.

2. Rationale and Theoretical Set-Up

We first describe our approach from a theoretical point of view. We will make use of the language of differential forms [DKT05] as most of the successful meshing techniques so far (using harmonic parameterizations or integral lines of orthogonal vector fields) can be elegantly formulated using the notion of *discrete differential forms*—a fact already noticed in [DKG05]. It is a trivial matter to discretize the equations written in this particular form; in particular the exterior derivative d of a node-based linear function is simply the difference of the node values for each oriented edge. See Appendix A for a brief overview.

2.1. Local Quadrangulation as Contouring

We start by using a "reverse-engineering" argument. Suppose that we *already* have a small surface patch composed of locally "nice" quadrangles, the notion of nice being highly application-dependent. From this mesh, we can first set a local (u,v) coordinate system (with directions e_u and e_v) of the surface to be aligned with the edges of the quads. We can then define a metric $\langle , \rangle$ such that $\langle e_u, e_v \rangle = 0$ everywhere, and such that lengths of each quad edge are unit. Thus, the mesh is locally defined by ***integer*** *u- and v-**isovalues***. In addition the gradients of the two parameters ∇u and ∇v are *orthogonal* in the prescribed metric. The way we have defined the metric also guarantees that we must have the magnitudes of the gradients equal to each other. The two conditions together are known as the Cauchy-Riemann equations for the parameters u and v of this patch:



$$\langle \nabla u, \nabla v \rangle = 0 \qquad \text{and} \qquad \langle \nabla u, \nabla u \rangle = \langle \nabla v, \nabla v \rangle.$$

These two equations can be elegantly formulated using the differentials of u and v, as well as the Hodge star induced by our metric. Indeed, the two 0-forms u and v simply satisfy:

$$du = \star dv$$

Notice that we can deduce (by applying d and $d\star$ to the previous equation) that $d \star dv = d \star du = 0$, hence du and

dv are both coclosed. Since $d \circ d = 0$, both are also closed. Therefore, du and dv must be *harmonic*. In more traditional notation, both gradient fields are curl- and divergence-free. Another consequence of the coclosedness of the two differentials is that both u and v are *also harmonic*, i.e., their Laplacian vanishes. These properties explain the popularity of harmonic functions in Euclidean space, where orthogonality means $\pi/2$ angles, hence leading to well-shaped quads [DKG05, GY03]. We will also stick to the Euclidean metric for now, to keep our explanations simple.

2.2. Towards Global Contouring

To extend the basic principle explained in the previous section from a local quad mesh to a global quad mesh, one needs to overcome a number of issues.

Necessity of discontinuities First, globally continuous harmonic scalar potentials are too restrictive for quad meshing purposes. In fact, for the frequent case of a genus-0 closed manifold , there are *no* globally continuous harmonic potentials other than the constant ones, of little worth. A classical way to deal with this problem is to add pole singularities, which amounts to piercing little holes at various locations on the surface. For example: pierce a sphere once at the top and once at the bottom; what remains is a globally continuous harmonic potential u, with extrema at the two poles, thus with flow lines defining longitudes. However, the corresponding v potential cannot be globally continuous since its derivative dv has closed flow lines, namely latitudes. Therefore, the only hope to truly extend the contouring approach is to allow the potentials to be *piecewise continuous*, i.e., only continuous inside non-overlapping patches of the manifold—akin to the traditional notion of charts [GH95, YZ04, KLS03]. We may find a potential v that is continuous everywhere except on a line joining the two poles, on which the jump of the v-value equals a constant. Note that in this case v is discontinuous but dv is not. However, we will see that continuous harmonic 1-forms are not sufficient in general. The reason is that they can only model singularities with *integer* indices, that is poles and saddles. As Figure 2 depicts, these types of singularities create significant distortion in the quadrangulation. To be able to generate fractional singularities, one needs to allow for certain types of discontinuities on 1-forms.

Compatibility conditions Acknowledging the lack of global continuity, we now assume that the 0-forms (the potentials u and v) can contain *singularities*, i.e., jumps along certain edges. Similarly, their differentials (the 1-forms du and dv, akin to the gradients of each potential) may have singularities at the same locations, i.e., the vector fields representing these 1-forms may jump across patch boundaries. We will denote such a boundary between two continuous patches a ***singularity line***. Fortunately, we will be able to set *simple compatibility conditions* on the jumps of potentials and of their differentials that will guarantee that a global contouring of u and v results in a proper tiling. More precisely, linear constraints of continuity on the two potentials

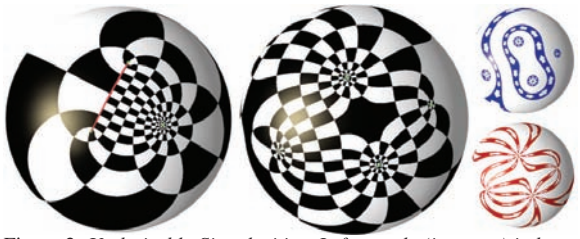


Figure 2: *Undesirable Singularities. Left: a pole (in green) induces too much distortion while a cut (in red) creates T-junctions in the final tiling. Middle and right: more than two poles (of index 1) on a genus-0 surface inevitably create singularities with negative index (saddles), creating large and distorted n-gons.*

can trivially ensure *continuity of the isolines*: indeed, if we trace all isolines with integer values, then the necessary and sufficient compatibility conditions are that the jumps of the potentials should be integer. On the other hand, the *smoothness of the isolines* will be ensured by a (tweaked, yet still linear) condition of "harmonicity" of the two potentials at patch boundaries. This last condition is, in fact, a continuity condition for 1-forms across the singularity line. We thus call this condition *singular continuity* to convey the notion of smoothness *modulo* the presence of a singularity.

Singular Continuity of Discrete Forms As mentioned above, obtaining a quad-dominant tiling on a disk-like patch through contouring two 0-forms u and v is rather easy. However, enforcing a proper tiling throughout the surface requires strong compatibility conditions at each singular line. Fortunately, only three different types of singular continuity across two neighboring patches can happen: *regular* (when both u and v directions individually match between the two patches), *reverse* (when both u and v directions change their orientations across the boundary), and *switch* (when the u and v directions are switched on the shared boundary). Only then can we get a globally consistent tiling of the surface.

2.3. Enforcing Singular Continuity of Forms

We now go over the various cases of continuity. As our technique uses two linear equations per vertex, we describe the different vertex types that we can encounter on a mesh: a vertex can be strictly within a patch, or on a particular type of singularity line.

Free Vertices When a vertex i is within a patch, i.e., not on any singularity line, we simply wish to enforce harmonicity of both 0-forms u and v. Consequently, the celebrated harmonicity condition [PP93] is imposed on this vertex, yielding:

$$\sum_{j \in \mathcal{N}(i)} w_{ij} \begin{pmatrix} u_i - u_j \\ v_i - v_j \end{pmatrix} = 0$$

where the index j goes through all the immediate neighboring vertices of i, u_k (resp., v_k) represents the value of u (resp., v) at vertex indexed k. For the Euclidean metric (as often used in graphics), the weights w_{ij} are the well-known sum of the cotangents of the angles opposite to edge ij.

Vertices with Regular Continuity When a vertex is on a *regular* singularity line between two patches, we assume that the fields u and v are smooth across the patch boundary *mod-*

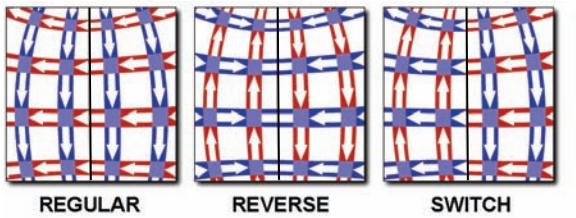
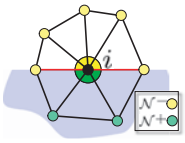


Figure 3: *Singular continuity. Three different types of continuity through a singularity line. Blue/red arrows are along isolines of u/v.*

ulo a constant offset. That is, if we call u^- (resp., v^-) the potential u of this vertex using its value from one of the patches, and u^+ (resp., v^+) the value at the same vertex but considering its value from the other patch, we wish to have:

$$u^- - u^+ = P_1 \qquad v^- - v^+ = P_2, \tag{1}$$

where P_1 and P_2 are two arbitrary integer constants associated with this particular patch boundary (we will discuss how to choose their values adequately later on). Obviously, enforcing this "equality modulo offset" will guarantee that integer isolines of u and v do match up at the boundary. Notice also that it corresponds to guaranteeing continuity of the 1-forms du and dv, as $d(u^+ - u^-) = du^+ - du^- = 0$. Finally, to ensure smoothness of these isolines, we enforce harmonicity of the two potentials taking the jump into account (see inset for conventions used):



$$\sum_{j\in\mathcal{N}^-(i)} w_{ij}\begin{pmatrix} u_i^- - u_j \\ v_i^- - v_j \end{pmatrix} + \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} u_i^+ - u_j \\ v_i^+ - v_j \end{pmatrix} = 0$$

Fortunately, one realizes that the above conditions can be rewritten using only *one* value of u and *one* value of v for our boundary vertex i—therefore alleviating the need for storing two different values, one on each side of the singularity line. Indeed, if we assume $u_i \equiv u_i^-$, and thanks to Eq. (1):

$$\boxed{\sum_{j\in\mathcal{N}(i)} w_{ij}\begin{pmatrix} u_i - u_j \\ v_i - v_j \end{pmatrix} = \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} P_1 \\ P_2 \end{pmatrix}.}$$

Notice that this equation is a simple variant of the former case, modifying the right hand side to impose the correct conditions on each side of the boundary.

Vertices with Reverse Continuity This time, we want the 0-forms u and v to change orientation when crossing the patch boundary. That is, we wish to have $du^+ = -du^-$, and $dv^+ = -dv^-$. These constraints are easily enforced by defining:

$$u^+ + u^- = Q_1 \qquad v^+ + v^- = Q_2,$$

where Q_1 and Q_2 are two integer constants associated to the boundary on which the vertex lies. We now enforce harmonicity of the two potentials at *i modulo* the reversal:

$$\sum_{j\in\mathcal{N}^-(i)} w_{ij}\begin{pmatrix} u_i^- - u_j \\ v_i^- - v_j \end{pmatrix} + \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} u_j - u_i^+ \\ v_j - v_i^+ \end{pmatrix} = 0$$

Again, one notices that a simpler expression using only one value for vertex i and a non-zero right-hand side, is:

$$\boxed{\sum_{j\in\mathcal{N}^-(i)} w_{ij}\begin{pmatrix} u_i - u_j \\ v_i - v_j \end{pmatrix} + \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} u_i + u_j \\ v_i + v_j \end{pmatrix} = \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} Q_1 \\ Q_2 \end{pmatrix}.}$$

This last expression preserves the *symmetric* nature of the Laplacian matrix. This is a particularly nice feature: state-of-the-art linear solvers have been shown to scale very well on such a problem [TCR05, BBK05].

Vertices with Switch Continuity Finally, for vertices on a singularity line on which we want u and v to switch, we simply enforce that $du^+ = dv^-$ and $dv^+ = -du^-$. Notice the extra minus sign, because switching u and v reverses one of the two directions. Again, these conditions are satisfied if:

$$v^- - u^+ = R_1 \qquad v^+ + u^- = R_2,$$

Finally, to ensure smoothness of these isolines, we enforce harmonicity of both potentials given this discontinuity through:

$$\sum_{j\in\mathcal{N}^-(i)} w_{ij}\begin{pmatrix} u_i^- - u_j \\ v_i^- - v_j \end{pmatrix} + \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} v_j - v_i^+ \\ u_i^+ - u_j \end{pmatrix} = 0$$

The resulting symmetric expression, using only one value for the vertex i and a non-zero right-hand side, is now:

$$\boxed{\sum_{j\in\mathcal{N}^-(i)} w_{ij}\begin{pmatrix} u_i - u_j \\ v_i - v_j \end{pmatrix} + \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} u_i + v_j \\ v_i - u_j \end{pmatrix} = \sum_{j\in\mathcal{N}^+(i)} w_{ij}\begin{pmatrix} R_2 \\ R_1 \end{pmatrix}.}$$

Notice there is an analogous formula for what we could call reverse-switch continuity vertices, namely when we want to switch u and $-v$.

2.4. Properties of Singular Continuity

Although quite simple, the four cases we discussed above provide an already rich repertoire of singularities. In particular, the previously mentioned case of a genus-0 object with two poles can be handled quite simply by linking the two poles with a singularity line: this "virtual" cut on the sphere creates one single patch touching itself along a *regular continuity* boundary. Now, the two potentials u and v can be computed per vertex by solving a modified Laplace equation, with vertices along the singularity line having different coefficients and non-zero right-hand sides.

Independence of Boundary Positions One remarkable property of the previous equations is that the exact position of the various boundaries between patches does *not* affect the final result: any boundary line in the same homology class as the original one will result in the same quad mesh. Although the 0-forms *will* be different (since their

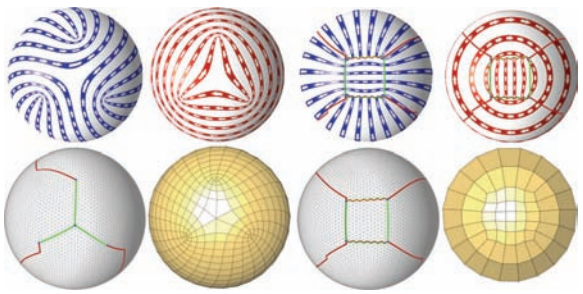


Figure 4: *Line Singularity as Basis of Many Singularities. Trisector (left) and square (right) singularities can be obtained by creating a graph of line singularities.*

Figure 5: *Line Singularity. From left to right: Piecewise-continuous harmonic potentials u and v (color-shaded); Red and blue arrows depict the direction of the potential gradients; a checkerboard is mapped onto the ellipsoid using (u,v) as texture coordinates; when the singularity line is wiggly, the two potential functions change, but their isolines remain exactly identical to the previous case.*

jumps will be located at distinct locations), their contouring will be exactly the same: only the local sign of their gradients will be affected in the reverse continuity case, while the gradient of u will become the gradient of v in the switch continuity case. In both cases, the union of the isolines of u and v remains the same! Therefore, the only real parameters are the set of constants, chosen for each boundary (that we called P_1, P_2, Q_1, Q_2, R_1, and R_2 previously). This is quite convenient, as no special effort needs to be spent on getting "straight" boundary lines (see Figure 5, right). In other words, only the *topology* of the patches is needed.

Other Typical Singularities Various other singularities can be achieved by designing a proper choice of boundary continuity between various patches. For instance, a trisector singularity, quite typical in direction fields, is obtained by assembling three concurrent lines, all of continuity type "reverse". A square singularity, *i.e.*, four index-$1/4$ poles forming a square-shaped index-1 singularity, is assembled from four lines in the shape of a square, with type "regular", "switch", "reverse", and "switch" in cyclic order (see Figure 4). Notice that these cases create significantly less distortion, and by design, no T-junctions. We will provide, in the next section, a simple implementation method to handle all these singularities (and more) in a unified manner using a singularity graph; but the equations provided above can already accommodate all these cases.

2.5. Discussion

Harmonic Forms Basis Globally continuous 1-forms are of interest only for objects of genus $g > 0$, the space of harmonic functions having dimension $2g$. Gu *et al.* [GY03, JWYG04] propose algorithms to compute and optimize those functions after having extracted the homology generators. In Appendix B, we explain how these computations can be derived in a simpler manner, simplifying both implementation and algorithmic complexity: we turn their asymmetric ExE linear solve into a symmetric VxV linear system. However, our novel treatment of harmonicity with singularities allows us to enrich these harmonic form bases significantly. One can indeed place a set of canonical singularities on the input mesh, then build a basis of the space of harmonic 1-forms (du and dv) that each singularity (point, line, or square) generates. This procedure requires solving a linear system per basis element, where only the right-hand side of the Laplace equations needs to be changed for every element. Nonetheless, finding a basis of all harmonic forms is only one step: the next issue is to find a *good* linear combination of these basis elements so as to compute our two potentials, and through contouring, a tiling. A user selection of coefficients for each basis element is highly non-intuitive since each 1-form has global support, making the design delicate at best—and certainly not scalable.

Using Optimization or Guidance Vector Fields An automatic selection of the coefficients could be achieved by providing a (most likely non-linear) optimization in order for the final mesh to best follow a given *guidance vector field* to achieve the same goal as in [RLL*05], or to improve control over sizing [JWYG04]. This approach can be also simplified by finding the best linear combination of basis elements by computing the L^2 *inner product* between each 1-form in the basis and the 1-form representing the guidance field [DKT05]—turning the non-linear optimization into a linear projection. Unfortunately, designing a guidance vector field on a manifold is, in itself, a challenge. In fact, we believe that designing a guidance field would be, in a sense, as complicated as performing a quad remeshing of the manifold: indeed, contouring does not require global orientability (only directions matter), whereas vector fields do.

Instead, we propose a radically different approach that allows an intuitive and scalable solution to quad mesh design. A whole *network of line singularities* is defined and leveraged to truly capture the topology of arbitrary manifolds. Next we present a way to design this network, the *singularity graph*, that is the backbone behind our mesh design.

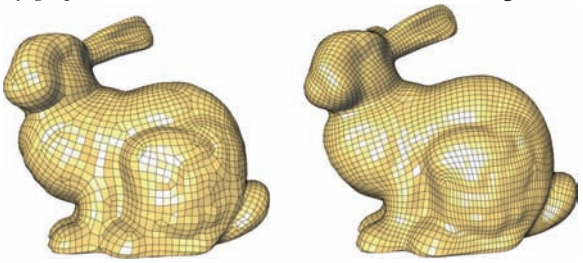


Figure 6: *Bunny: Comparison between a model (top, 6415 faces among which 124 are non-quads, 314 irregular vertices), courtesy of [RLL*05]) and our approach (bottom, 6575 quads). Notice the regularity of our mesh with only 34 irregular vertices (more views of a coarser version can be seen in Fig. 12).*

3. Designing a Singularity Graph

Allowing quad mesh design flexibility requires the use of a potentially large set of patches: the more patches we define, the better we can control the local alignment of edges as well as the local area distortion. Thus, we propose a user-guided (or automatic if needed) way to create this patch layout through the definition of a *singularity graph*, representing a topological "template" linking the singularities.

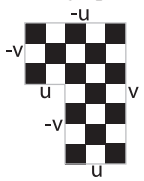
3.1. Definition of Singularity Graph

We call a *singularity graph* a meta-mesh whose meta-faces are non-overlapping patches of the original mesh, and whose meta-edges are assigned one of the *singularity continuity* conditions described in the previous section (Figure 4). This

structure is similar to the notion of *coarse mesh* in subdivision surfaces, as the resulting quad mesh based on this graph only have irregular vertices *at* meta-vertices.

Topological Structure of the Singularity Graph The vertices of the singularity graph (called meta-vertices to avoid ambiguity) are a subset of the vertices of the input triangle mesh. They should be thought of as salient points of the manifold, as they will live at the intersection of several regular patches in the final quad mesh—but additional meta-vertices can also be added virtually anywhere, allowing a *better control* of the output quad mesh as we will discuss later. Connectivity between meta-vertices define the meta-edges of the singularity graph. Each of these meta-edges are made out of two half-edges, oppositely oriented; these half-edges will be useful in explaining further details of our approach. Note that, as pointed out in Section 2.4, the exact positioning of the meta-edges will *not* affect the final mesh as demonstrated in our results (e.g., Figure 10). Finally, every cycle of half meta-edges defines a (meta-)face of the singular graph. Such a face corresponds to a patch in which our 0-forms will be smooth and continuous. In this section, we will call F (resp., E) the number of meta-faces (resp., meta-edges).

Determining Types of Singular Continuity Given a singular graph, we must assign to each meta-edge a particular *singular continuity* type as defined in Section 2. These assignments can be automatically obtained *if* we first tag each meta-halfedge as u, $-u$, v, or $-v$ according to their alignment with increasing or decreasing directions of the parameter (see Figure 7). Indeed, we would ideally like to map each meta-face of the graph to an *orthogonal polygon* in the parameter domain to *guarantee* the existence of a regular quadrangulation inside this patch. That is, the face should conceptually look like a simple polygon in the (u,v) parameter plane with only angles multiple of $\pi/2$ (see inset)—in fact, the choice of which edge is aligned with u vs. v does *not* affect the final quad mesh, so this polygon can be arbitrarily rotated by multiples of $\pi/2$ too. This condition imposes a constraint on the half-edge assignments, and we will provide an automatic procedure to enforce it on each face. After such an assignment is provided, the corresponding *singular continuity* types for all meta-edges becomes simple, as, for each pair of half-edge assignments, correspond the following continuity types:



- ***Regular***: $\{u,-u\}$, $\{-u,u\}$, $\{v,-v\}$, $\{-v,v\}$;
- ***Reverse***: $\{u,u\}$, $\{v,v\}$, $\{-u,-u\}$, $\{-v,-v\}$;
- ***Switch*** : $\{u,v\}$, $\{v,-u\}$, $\{-u,-v\}$, $\{-v,u\}$, (and symmetrically, $\{-u,v\}$, $\{-v,-u\}$, $\{u,-v\}$, $\{v,-u\}$.

3.2. Assisted Singularity Graph Creation

While the user can always either design or modify a given singularity graph and its current assignments interactively, an assisted creation of the graph is important when the manifold to remesh is quite complex. In particular, finding a geometrically-faithful graph is particularly important if the resulting mesh is expected to provide a concise approximation of the original mesh. The following generative procedures have been quite useful in our experiments—specific automatic graph design algorithms can be devised too.

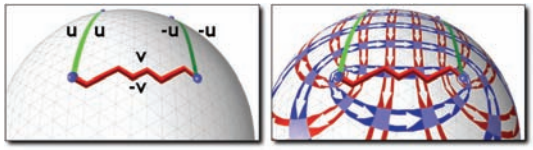


Figure 7: *Tags on half meta-edges. The continuity type of meta-edges is found by first tagging half-metaedges with $u, v, -u$, or $-v$.*

Placement of Meta-Vertices The *curvature tensor* offers a relevant guidance when it comes to alignment of the quad mesh edges with geometrically relevant directions. This tensor defines two orthogonal principal curvature directions $e_{\min}$ and $e_{\max}$ everywhere, except at the so-called umbilics. As we ideally seek a quad mesh with edges aligned with principal directions, it is natural to place the singularities of the mesh at the singularities of the principal foliations, i.e., the umbilics. To estimate the positions of these umbilics, we estimate the curvature tensor of the surface using the method advocated in [CSM03]. The support of this estimator must be chosen sufficiently large so as to avoid having too many unnecessary singularities in the presence of noise. The user can also define a maximum number of singularities.

Generation and Tagging of Meta-Edges We eventually want the u and v isolines to align with the principal directions of the surface. Since meta-edges represent a *template* of the final quad mesh, it is natural to take as meta-edge paths the ones that are most-aligned with the curvature lines of the input mesh. To find these paths, our technique performs two passes of a region-growing procedure over the input mesh triangles, starting at the one-rings of the meta-vertices. The growth procedure is in essence a multi-source Dijkstra algorithm over the dual graph, where each dual edge is weighted in accordance with its length multiplied by its alignment with $e_{\min}$ for the first pass, and $e_{\max}$ for the second pass. When two region fronts meet, their originating meta-vertices are connected with a meta-edge (if not already connected). The result is a graph connecting the meta-vertices, as desired. The resulting connectivity is a mere product of the geometry itself, as two meta-vertices are connected iff there is a path roughly aligned with a principal direction (see the examples depicted in Figure 8).

Tagging halfedges becomes simple: we can for instance compute the total geodesic curvature of every patch's boundary (i.e., the Gaussian curvature of the patch). We then compute the portion of geodesic curvature around each meta-vertex of a patch (from halfway of the previous meta-edge to halfway of the next meta-edge): this value, divided by the total geodesic curvature of the patch, scaled by 2π, and then rounded to the closest multiple of $\pi/2$ tells us how the tag of the next edge is linked to the previous tag. An initial meta-

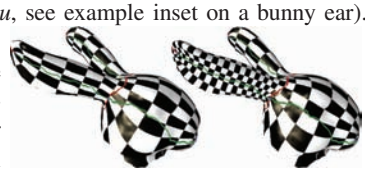
Figure 8: *Automatic Generation of Singularity Graph. Left: generation on the ellipsoid. The color map depicts the cumulative distance to the umbilics taking into account the alignment with κ_{min} and κ_{max}. Right: a similar process demonstrated on a cartoon hand.*

edge is tagged u arbitrarily (remember that rotations of the patch's assignments are equivalent). Since the sum of these values will be 2π, we have created a closed face in parameter space, with geometrically-derived angles.

3.3. Designing a Tiling

Designing the final quadrangulation amounts to deciding on an assignment of (u,v) values *at each corner of every meta-face*. These are, indeed, the *only* constraints that need to be fixed for the modified Laplace equation to be solvable: once these "meta-parameters" are known, all the constants defined per patch-boundary in the continuity equations described in Section 2 are determined. Thus, we begin by computing these values by solving a small meta-system. We will then inject the resulting meta-parameters into the system of (modified) Laplace equations for the final solve.

Meta-Parameters The constraints on these parameters depend only on the differences of u and v values on each meta-edge. Let Du (resp., Dv) be such a difference in u (resp., v) value over a meta-edge. In each meta-face, the sum of all the Du's of half-edges tagged as u *must* equal the sum of the Du's for those tagged as $-u$: the meta-face is a *closed* polygon in parameter space. The same argument applies for the sum of Dv of v edges and that of $-v$ edges (in the language of differential forms, this states that du and dv must be closed on the meta-mesh too). Similarly, as we wish to have isolines stitching properly across meta-faces, we must have equal differences for each half-edge of a meta-edge. Thus, these differences are E coefficients that need to be set, and there are $2F$ linear constraints on them (one for u and one for v per meta-face). We now set up in a small linear system these $2F$ constraint equations for E variables. However, the constraints can (and will often) be redundant. We thus use Gauss elimination to find the independent equations. This process is extremely fast since the graph contains typically three orders of magnitude fewer edges than the original mesh, and all the coefficients in this meta linear system are either 1 or -1 (since Du (or Dv) is computed as the simple difference between two parameter values). Additionally, notice that the $2F$ constraint equations sum to zero: we can thus guarantee that there will be *at least* E-$2F$+1 number of independent variables. Since a meta-face is homeomorphic to an orthogonal polygon, each meta-face has at least 4 edges. Therefore, $E \geq 2F$, and there is always at least one degree of freedom. More DOFs can be added by enriching the graph. The *free* meta-parameters now need to be set. Obviously, the user can enter values based on the number of isolines desired on those meta-edges (this is, indeed the geometric meaning of Du, see example inset on a bunny ear). Automatic assignment can also be done by simply entering the closest integer value to the actual meta-edge *length*: this provides a good approximation of the most geometrically-relevant number of isolines—as it helps providing a more isometric parameterization, therefore minimizing area distortion.



Code to assemble the ith line of the linear system $AU = B$
where A is the modified Laplacian matrix (note: V is the # of unconstrained vertices).
We use the conventions defined in Section 2.
For each half-edge (v_i, v_j)**, do**:

```
c ← w_ij; swap ← false; ε_u ← 1; ε_v ← 1; b_u ← 0; b_v ← 0
    if (i or j is on singularity line, but both not on the same line)
    switch (type(j))
        case On-Regular:
            getConstantsOfLine(P_1, P_2); b_u −= P_1 * c; b_v −= P_2 * c;
        case OnReverse:
            getConstantsOfLine(Q_1, Q_2);
            ε_u ← −ε_u; ε_v ← −ε_v; b_u −= Q_1 * c; b_v −= Q_2 * c;
        case OnSwitch:
            swap = true;  getConstantsOfLine(R_1, R_2);
            ε_v ← −ε_v; b_u −= R_1 * c_j; b_v −= R_2 * c_j;
        default: do nothing;
    switch (type(i))
        case OnRegular:
            getConstantsOfLine(P_1, P_2); b_u += P_1 * c; b_v += P_2 * c;
        case OnReverse:
            getConstantsOfLine(Q_1, Q_2);
            ε_u ← −ε_u; ε_v ← −ε_v;
            b_u ← −b_u; b_v ← −b_v; b_u += Q_1 * c * ε_u; b_v += Q_2 * c * ε_v;
        case OnSwitch:
            getConstantsOfLine(R_1, R_2);
            ε_v ← −ε_v; swap ←!swap;
            temp ← b_u; b_u ← −b_v; b_v ← temp;
            b_u += R_2 * c; b_v += R_1 * c;
        default: do nothing.
A[i,i] += c; A[i+V, i+V] += c; B[i] += b_u; B[i+V] += b_v;
if (swap)
    A[i, j+V] = −c * ε_u; A[i+V, j] = −c * ε_v;
else A[i, j] = −c * ε_u; A[i+V, j+V] = −c * ε_v;
```

Figure 9: *Pseudocode of the Laplacian Matrix Assembly.*

Final Solve Once the meta-parameters are set, we can finally assemble the global linear system for the 0-forms u and v of the original mesh. The system is created by assembling two linear equations per vertex v_i (depending if it is free or on a line singularity, as discussed in Section 2), and none for the vertices on corners of meta-faces, as they are already determined by the meta-parameters. Whenever one of these linear equations involves a meta-face corner, its value (i.e., one of the meta-parameters) is constrained and therefore substituted and moved to the right-hand side of the system. The matrix of this system is sparse and symmetric. *By construction, a contouring of the u and v potentials will stitch automatically into a pure quad mesh.* We provide pseudocode in Figure 9 for all possible cases of continuity conditions to facilitate this matrix assembly.

4. Implementation Details and Results

In our implementation, we use a half-edge data structure for the input mesh as well as for the singularity graph. The assembly of the modified Laplacian matrix is performed by going through each input mesh half-edge and checking the

type of vertices at each end. We used the supernodal multifrontal Cholesky factorization option of TAUCS [TCR05] as our linear solver since it handles sparse symmetric positive-definite systems very efficiently (the linear systems for the results shown in this paper took between 0.1 to 23s to solve, our running times being consistent with the ones reported in [BBK05]). Generating a singularity graph takes on the order of a minute or two, since only the corners need to be chosen by the user.

Controlling Alignment and Distortion The user can finely control the *alignment* of the mesh with features or semantically relevant directions. Inserting more meta-vertices along a given meta-edge, and tagging the newly-created meta-edge with the exact same tags as the original one will do the trick: the quads created nearby will align with these meta-edges as the isocurves are more constrained by the added meta-vertices. Similarly, while the presence of saddles is sometimes *unavoidable* on certain manifold (as stated by the Hopf-Poincaré theorem), the user can minimize their distortion by altering the singularity graph: adding a meta-face over a saddle will "split" it into four lower-index saddles, reducing the effective distortion quite significantly—at the price of three added irregular vertices.

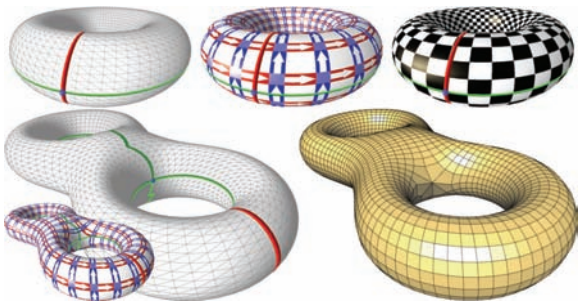


Figure 10: *Non-trivial Topology. Top: genus-*1 *example; Bottom: genus-*2 *model—a saddle (imposed by Hopf-Poincaré theorem) is present on a meta-vertex. The final meshes are still pure quad. Both singularity graphs were defined by homology generators.*

Controlling Quad Angles Notice that, unlike previous approaches [GY03], we did not require our pair of 1-forms du and dv to be holomorphic (*i.e.*, related through the Hodge star) to allow for more design flexibility. Therefore, using our method with an unreasonable choice of meta-parameters will result in regions where the quads are grossly non orthogonal. One way to palliate this issue is to find the meta-parameters so that the pair (du, dv) minimizes (in the $\mathcal{L}^2$ sense) the 1-form $du - \star dv$. We can also proceed as explained in Appendix C to truly design a stitched *conformal* parameterization—the slightly-modified singular continuity conditions involved in this variant will be explored in a future paper. An alternative approach is to play with another parameter: the Hodge star $\star$. Although beyond the scope of this paper, our method can indeed be applied for arbitrary metrics. Altering the metric locally amounts to change the coefficient w_{ij} used in the Laplacian operator. Theoretically speaking, the optimal quad shape for a *best* surface approximation derives from the curvature tensor—a research direction we wish to exploit soon.

Boundaries Boundaries are handled as follows. Once the assignments of (u, v) at each corner of the meta-faces are done, we go through the edges tagged as boundary between two such corners, and force the boundary values to be linearly interpolating the two corner values. This will force the isovalue of our potentials to follow the boundaries (Figure 11).

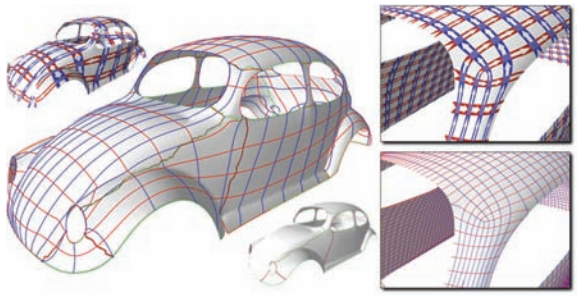


Figure 11: *Quad Remeshing with Boundaries. A triangulated beetle model with boundaries is remeshed with a singularity graph.*

Mesh Extraction The final mesh is easily extracted through integer contouring: we walk in the triangulation from an integer intersection of u and v to the next. Note that we may cross a line sigularity; in that case, we account for the singularity type to be able to resume the walk on the other side and find the next intersection. The output is, by construction, a *pure quadrilateral mesh*.

Comparisons & Limitations Our approach may look loosely related to the global linear system proposed in [KLS03] (recently extended to quads in [DBG*06]). However, the difference is significant: while these previous approaches try to make the excess angle of each corner (meta-vertex) be as close to 0 as possible, we instead fully embrace the specificity of the meta-vertices by forcing their Gaussian curvature to multiples of $\frac{\pi}{2}$ (i.e., specific cone singularities). To the best of our knowledge, *no other*

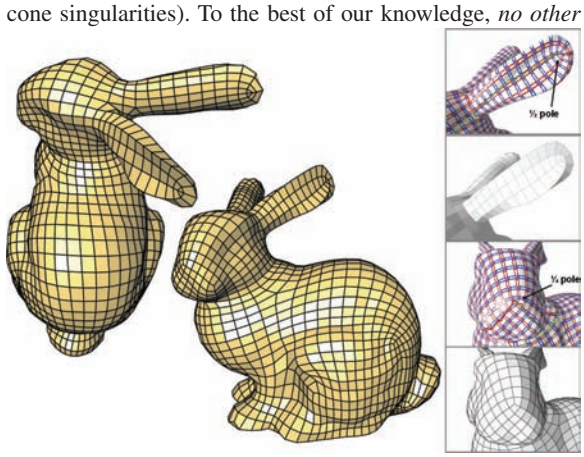


Figure 12: *Remeshing the Stanford Bunny. Left: Pure-quad remeshing. Right: detail of a half-pole, and a quarter-pole (using a switch and a regular line incident to the singularity). The half-pole becomes a degree-2 vertex, incident to two quads, with two nearly collinear edges. The degree-2 vertex can optionally be removed by merging its two incident quads into one.*

approach allows such a unique solution with a simple linear solve. Nonetheless, we acknowledge that if the user decides to significantly override the values on meta-vertices automatically prescribed by our technique, the resulting meshes may have folds and significant stretch. In our experience, except for vastly inappropriate values, the results are always satisfactory. Note that compared to previous work, the main strengths of our method are the simplicity (and scalability) of computations (a simple linear solve followed by isocontouring) and a direct *control* over the singularities. In particular, compared to [RLL*05], we do not resort to a non-linear minimization or to a curl-correction phase, and we allow a precise placement of the resulting (non-integer index) poles.

5. Conclusion

We have presented a theoretical and algorithmic approach for designing quadrangle tilings from arbitrary triangulated surface meshes. Our algorithm computes two piecewise smooth harmonic scalar functions, whose isolines tile the input surface into quadrangles, without any T-junctions. Our main contribution is an extension of the discrete Laplace operator which encompasses several types of line singularities. The resulting two discrete differential 1-forms are either regular, opposite or switched along the singularity graph edges. We show that this modification guarantees the continuity of the isolines across the lines, while the locations of the isolines themselves depend on the global solution to the modified Laplace equation over the whole surface. Design flexibility is provided through specification of the type of each line singularity of the graph, as well as the number of isolines along independent meta-edges to control quad sizes.

Besides the mere interest of creating quad meshes, useful in a slew of applications, we are interested in studying the theoretical consequences of the framework we proposed. In particular, understanding the feasibility and consequences of producing ***arbitrary cone singularities*** using only linear algebra is extremely valuable. This can be easily achieved by defining continuity conditions with arbitrary rotation of a 1-form. The relaxation proposed in [KLS03] could also be used to automatically reduce distortion near irregular vertices if a fully-automatic quad remeshing is sought after.

Acknowledgments The authors wish to thank Peter Schröder for his continuous help, and Alex McKenzie for his helpful comments. Sponsors include NSF (CAREER CCR-0133983, and ITR DMS-0453145), DOE (DE-FG02-04ER25657), and the EU Network of Excellence AIM@SHAPE (IST NoE No 506766).

References

[ACSD*03] Alliez P., Cohen-Steiner D., Devillers O., Lévy B., Desbrun M.: Anisotropic polygonal remeshing. *ACM Trans. Graph. 22*, 3 (2003).

[AUGA05] Alliez P., Ucelli G., Gotsman C., Attene M.: Recent advances in remeshing of surfaces. STAR AIM@SHAPE, January 2005.

[BBK05] Botsch M., Bommes D., Kobbelt L.: Efficient linear system solvers for mesh processing. In *IMA Conf. on Math. of Surfaces* (2005), pp. 62–83.

[BMRJ04] Boier-Martin I., Rushmeier H., Jin J.: Parameterization of triangle meshes over quadrilateral domains. In *Symp. on Geometry processing* (2004), pp. 193–203.

[CSM03] Cohen-Steiner D., Morvan J.-M.: Restricted delaunay triangulations and normal cycle. In *Proceedings of the Symp. on Computational Geometry* (2003), pp. 312–321.

[DBG*06] Dong S., Bremer P.-T., Garland M., Pascucci V., Hart J. C.: Spectral surface quadrangulation. to appear at ACM SIGGRAPH '06, July 2006.

[DKG05] Dong S., Kircher S., Garland M.: Harmonic functions for quadrilateral remeshing of arbitrary manifolds. *Computer Aided Design (Special Issue on Geometry Processing) 22*, 4 (2005), 392–423.

[DKT05] Desbrun M., Kanso E., Tong Y.: Discrete differential forms for computational modeling. In *Discrete Differential Geometry*. ACM SIGGRAPH Course Notes, 2005.

[Ede00] Edelsbrunner H.: Mathematical problems in the reconstruction of shapes, 2000. Talk at MSRI's Workshop on Computational Algebraic Analysis (`http://msri.mathnet.or.kr/`).

[EW05] Erickson J., Whittlesey K.: Greedy optimal homotopy and homology generators. In *SODA* (2005), pp. 1038–1046.

[GH95] Grimm C., Hughes J.: Modeling surfaces of arbitrary topology. In *Proceedings of ACM SIGGRAPH* (July 1995), pp. 359–369.

[GY03] Gu X., Yau S.-T.: Global conformal parameterization. In *Symposium on Geometry Processing* (2003), pp. 127–137.

[JWYG04] Jin M., Wang Y., Yau S.-T., Gu X.: Optimal global conformal surface parameterization. In *IEEE Visualization* (2004), pp. 267–274.

[KLS03] Khodakovsky A., Litke N., Schröder P.: Globally smooth parameterizations with low distortion. *ACM Trans. Graph. 22*, 3 (2003), 350–357.

[KSS06] Kharevych L., Springborn B., , Schröder P.: Discrete conformal mappings via circle patterns. *ACM Trans. on Graphics 25*, 2 (2006).

[MK04] Marinov M., Kobbelt L.: Direct anisotropic quad-dominant remeshing. In *Proceedings of the Pacific Graphics* (2004), pp. 207–216.

[PP93] Pinkall U., Polthier K.: Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics 2(1)* (1993), 15–36.

[RLL*05] Ray N., Li W. C., Lévy B., Sheffer A., Alliez P.: Periodic global parameterization. *Preprint found at* `www.loria.fr/~levy/publications/` (2005).

[TCR05] Toledo S., Chen D., Rotkin V.: TAUCS. Available at `http://www.tau.ac.il/~stoledo/taucs`, 2005.

[YZ04] Ying L., Zorin D.: A simple manifold-based construction of surfaces of arbitrary smoothness. *ACM Trans. on Graphics 23*, 3 (2004), 271–275.

Appendix A: Discrete Forms Glossary

For completeness, we provide a glossary of the terms traditionally involved in the use of discrete differential forms, more details and references can be found in [DKT05].

• *Discrete forms*: A discrete k-form on a piecewise-linear manifold assigns a real number to every oriented k-simplex of the manifold. 0-forms are discrete versions of continuous scalar fields, while 1-forms are discrete versions of vector fields (or of the differential 1-forms they represent).

• *Exterior Derivative*: The exterior derivative operator associates to each k-form ω a particular $(k+1)$-form $d\omega$. If ω is a 0-form (valued at each node), *i.e.*, a function on the vertices, then $d\omega$ evaluated on any oriented edge v_1v_2 is equal to $\omega(v_1)-\omega(v_2)$. If ω is a 1-form, then $d\omega(v_1v_2v_3) = \omega(v_1v_2)+\omega(v_2v_3)+\omega(v_3v_1)$ for any triangle $v_1v_2v_3$. The exterior derivative is thus similar to the gradient operator for 0-forms, and to the curl operator for 1-forms. It is not difficult to check that $d\circ d = 0$ (the curl of any gradient field is always null).

• *Closed and Exact Forms*: ω is called closed if $d\omega = 0$. For 1-forms, this means that the sum of the values on the directed edges around each face equals zero. Poincaré lemma states that every such closed 1-form has a local potential (a 0-form u with $\omega = du$) inside any disk-like patch. If there is such a potential such that $\omega = du$ on

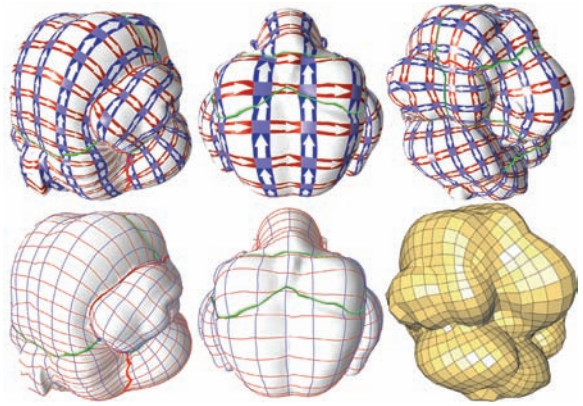


Figure 13: *Omotondo Model. With a simple singularity graph, a quad mesh following relevant geometric directions is created.*

the *whole* space, the 1-form ω is called exact. Since $d \circ d = 0$, exact forms are always closed.

• *Cohomology*: The difference between exactness and closedness of 1-forms on surfaces is determined by the topology of the surface. More precisely, two closed forms are said to belong to the same cohomology class if their difference is *exact*. Cohomology classes of 1-forms form a vector space whose dimension, called the first Betti number, counts the number of loops/holes in the surface.

• *Hodge star operator* $\star$: on 1-forms, it is the discrete analog of applying a rotation of $\frac{\pi}{2}$ to a vector field. Divergence can be written as $d*$ on a 1-form. Note that the Hodge star is metric dependent: if we change the metric, the Hodge star changes accordingly.

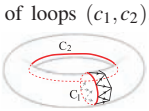
• *Harmonic Forms*: for $k > 0$, a k-form ω such that $\star\omega$ is closed is called *coclosed*. A k-form closed *and* coclosed is called harmonic. It can be shown that there is exactly one harmonic form in each cohomology class. Hence, harmonic 1-forms form a vector space of dimension equal to the first Betti number of the surface. A scalar function f is called harmonic if its Laplacian Δf vanishes. The exterior derivative of a scalar field is harmonic iff this field is harmonic.

Appendix B: Harmonic Form Bases

According to the continuous theory, there are $2g$ independent continuous harmonic 1-forms defined on a closed, orientable genus-g surface. Each element of the 1-form basis is associated with a loop, called the homology generator. These homology generators are basically $2g$ loops on the surface that allow to cut open the surface into a disk-like domain. For a detailed discussion, see [GY03]. Often, a surface M has a boundary ∂M, which can have m connected components (i.e., m boundaries). And naturally, if we look for quads orthogonal to the boundaries, we must have one of the 1-forms normal to the boundary, and the other tangential to the boundary. The continuous theory states that normal harmonic 1-forms are associated with the relative homology group $H_1(M, \partial M)$, which includes m-1 independent paths connecting the various boundaries, plus the aforementioned $2g$ genus-induced homological generators. Additionally, the space of tangential harmonic forms for a surface with m boundaries has $2g+m$-1 basis elements, associated with m-1 closed boundaries in addition to the $2g$ loops (all these loops forming the homology group $H_1(M)$).

We propose a simplified way of computing discrete counterparts of these 1-forms that necessitates *no double-covering*; in fact, it only requires solving a simple Poisson equation. First, any known method (for instance, [EW05]) can be used to find the homology generator loops, which are organized in pairs (each pair corresponds to one circle that will cut a handle and one circle which is around the hole of the handle, as in Figure 10 for the torus).

• For closed surfaces, there is one harmonic form in each cohomology class, so we propose to *pick a closed 1-form* ξ *from one cohomology class, and solve for a scalar field* f *such that* $d \star df = d \star \xi$. *Then* $\omega = \xi - df$ *is a harmonic 1-form*. Finding the independent ξ's is trivial thanks to *Poincaré duality*, as we now detail. The cohomology class associated with one loop is nothing but the set of all the closed forms that will integrate to 1 around that loop, and to 0 on all other loops. First take one pair of loops (c_1, c_2). Assigning the value 1 to edges *crossing* the loop (say c_1) in one given direction, and 0 to all other edges, defines ξ. Indeed, this is a closed form since the sum of the values will be 0 on all faces (for faces with non-zero edges, *i.e.*, near c_1, they always have a pair of them which cancel out; in other words, the loop always enter, then leave any triangle that it goes through). This 1-form integrates to 1 on c_2, and 0 on all other loops. Thus, ω calculated as above will be a harmonic form for c_2, and we can get the other $(2g-1)$ 1-form this way. To get the potential, traverse the surface starting from any seed vertex and integrate along edges (as done in [GY03]). The *gradient* of this potential corresponds to the continuous 1-form that we solved for.



• To compute the normal harmonic forms associated with the $2g$ loops, we use exactly the same method. The only difference is that we need to set the values of the potential at the boundary to 0. Now for the 1-forms associated with the $m-1$ paths, we put all vertices on the connected component of the boundary incident to the source of the path to 1, and assign 0 to all other boundary vertices (assuming, without loss of generality, that the paths start from the same boundary component). Solving the Laplace equation with Dirichlet boundary conditions gives us a harmonic 0-form whose gradient *is* the 1-form sought after. Although equivalent in result, our method is simpler to implement and *much* faster to compute than the double covering technique proposed in [GY03].

• Finally, tangential forms are easily dealt with based on Hodge and Lefschetz duality between homology and relative homology $H_1(M) \cong H_1(M, \partial M)$. Indeed, the Hodge star of the space of normal forms equals the space of tangential forms. Therefore, we can take the Hodge star of all $2g+m-1$ normal forms η_i as described above, compute their circulations around the $m-1$ loops along the boundary and the $2g$ loops for handles. These circulations form a matrix P, the so-called *period matrix*. The tangential 1-form corresponding to the i-th homology class in the basis is then $P^{-1}(e_i)$, with e_i the vector with 1 in the i-th entry and 0 elsewhere.

Notes The singularity graph presented in this paper is another simple alternative to what we just proposed that avoids the need for dual-primal discrete form conversion. Indeed, if we restrict ourselves to the use of "regular" singularity lines, we can get all global harmonic 1-forms directly. In particular, notice that when you use the homology generators as the singularity graph, you get exactly $2g$ free parameters.

Appendix C: Orthogonality & Quantization

To ensure orthogonality, we must have a pair of 1-forms that are Hodge star of each other up to a sign (the pair is then called holomorphic). In order to have quadrangles using integer-contouring, we must have the circulation around $2g+2(m-1)$ loops being integer numbers. This means orthogonality is not achievable in many cases (for instance, when there are irrational number entries in the period matrix). But as we can use rational numbers to approximate real numbers to arbitrary precision, we can increase the resolution of the mesh to get closer to orthogonality. More precisely, one can fix a denominator q and round the entries in the inverse period matrix to the closest multiple of $1/q$. Then we use $1/q$ as our contouring step. Better orthogonality is achieved for high values of q, giving a trade-off between orthogonality and coarseness of the resulting quad mesh.

In our approach using the singularity graph, we can obtain holomorphy of our forms by changing the linear system slightly. Instead of the singular continuity types that we used, we have to use conditions such as $du^+ = *du^-$ (and variants), resulting in a modified Laplace equation for u only. From the 0-form u, we can then compute $*du$, and assign v to its integral over each patch. Suppose that u is quantized is described above, all we need to do is to quantize the integral of $*du$ along each edge too. Using the corner values for (u_i, v_i) as the input for the meta linear system will lead to a ***stitched conformal mapping***. In other word, we can relax the constraint that each patch must be mapped to an orthogonal polygon in the parameter domain, and ensure the orthogonality between du and dv directly instead.