Computing Self-Supporting Surfaces by Regular Triangulation

Yang Liu* Hao Pan† John Snyder‡ Wenping Wang† Baining Guo*
*Microsoft Research Asia †The University of Hong Kong ‡Microsoft Research

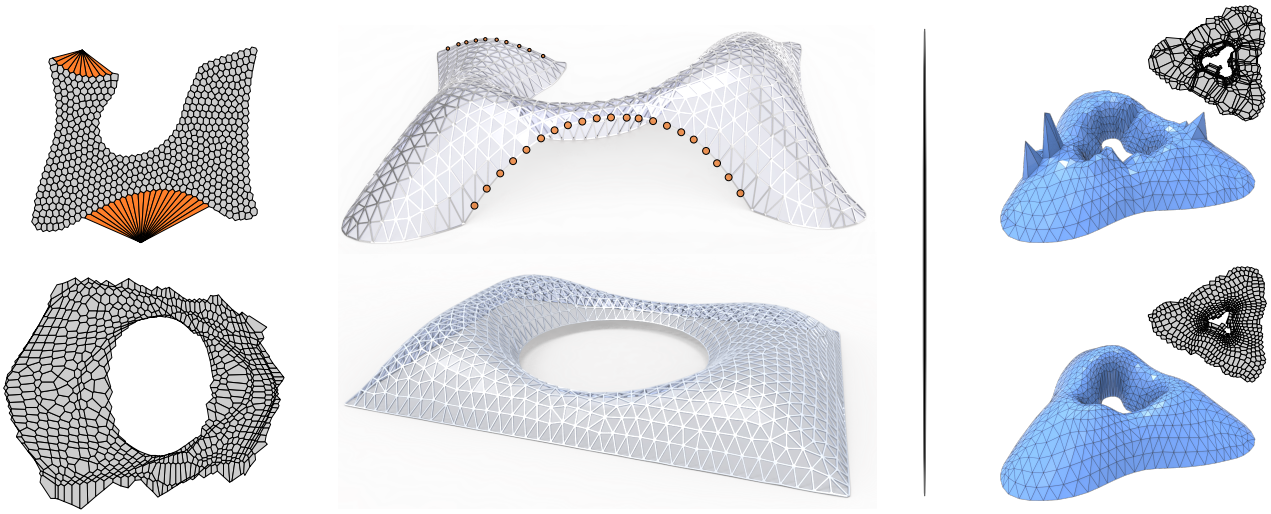


Figure 1: *Left: self-supporting surfaces with unsupported (top) and supported (bottom) boundary constraints. Unsupported boundary vertices and their corresponding power cells are colored in orange. Top right: initial self-supporting mesh. Spikes appear due to extremely small reciprocal areas. Bottom right: applying our smoothing scheme (5 iterations) improves mesh quality. The power diagrams (black) show how power cell area is distributed more evenly.*

Abstract

Masonry structures must be compressively self-supporting; designing such surfaces forms an important topic in architecture as well as a challenging problem in geometric modeling. Under certain conditions, a surjective mapping exists between a *power diagram*, defined by a set of 2D vertices and associated weights, and the reciprocal diagram that characterizes the force diagram of a discrete self-supporting network. This observation lets us define a new and convenient parameterization for the space of self-supporting networks. Based on it and the discrete geometry of this design space, we present novel geometry processing methods including surface smoothing and remeshing which significantly reduce the magnitude of force densities and homogenize their distribution.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Curve, surface, solid, and object representations

Keywords: power diagram, reciprocal diagram, stress potential

Links: DL PDF

ACM Reference Format
Liu, Y., Pan, H., Snyder, J., Wang, W., Guo, B. 2013. Computing Self-Supporting Surfaces By Regular Triangulation. ACM Trans. Graph. 32, 4, Article 92 (July 2013), 10 pages. DOI = 10.1145/2461912.2461927 http://doi.acm.org/10.1145/2461912.2461927.

DOI: http://doi.acm.org/10.1145/2461912.2461927

1 Introduction

A structure is *self-supporting* when it stands in static equilibrium without external support. This idealization is fundamental when designing masonry structures which can withstand compression but are weak against tensile stresses.

Many techniques have been employed to design self-supporting structures, including the force network method [ODwyer 1999], the hanging chain model [Kilian and Ochsendorf 2005], and the thrust network [Fraternali 2010]. Recently, Block and Ochsendorf [2007; 2009] developed *thrust network analysis* which decouples the 3D force equilibrium of a self-supporting network into horizontal (xy) and vertical (z) components. Horizontal equilibrium is characterized by the 2D network's force diagram, called the *reciprocal diagram* [Maxwell 1869; Cremona 1890]. By adjusting the reciprocal diagram, thrust network analysis provides a powerful way to design self-supporting structures. In particular, it allows a given shape to be approximated by a self-supporting one [Block and Lachauer 2011; Vouga et al. 2012].

An important limitation of existing approaches is that they fix the structure's network topology; i.e., the edge connectivity in a polygonal mesh. But the choice of topology influences both the shape of the result and the forces it generates under equilibrium. A non-optimal choice overestimates forces and underestimates the strength of a vaulted masonry structure [ODwyer 1999; Block 2009]. We use a *regular triangulation* (also known as *weighted Delaunay triangulation*) to parameterize the space of networks under equilibrium. Though the link between reciprocal diagrams and regular triangulations has long been understood [Aurenhammer 1987b], we exploit it computationally, to improve processing in the space of self-supporting meshes. Our parameterization implicitly encodes (rather than fixes) the network's connectivity, letting it adapt.

Our contributions are summarized as follows.

- We provide a novel and natural way to parameterize the space of networks under equilibrium. The parameterization is based on the 2D regular triangulation and its dual, the *power diagram*. It comprises a simple set of 2D vertices each with a scalar radius, and avoids encoding discrete connectivity variables.

- We analyze the dimensionality of the mapping from regular triangulations to reciprocal diagrams. We introduce a simple method that reduces the number of edge equations required to infer power diagram weights.

- We present a novel smoothing scheme based on isotropic mean curvature that removes irregularities in a self-supporting mesh. We also present a novel remeshing method that optimizes and reduces variation in force density by matching it to smoothly interpolated stress. The connectivity adaptation afforded by our parameterization is crucial to the effectiveness of these methods.

2 Related Work

Self-supporting design Various techniques have been developed since the nineteenth century; see [Block 2009, chapter 2] for a detailed review. The fundamental theory is summarized as the *Safe Theorem* [Heyman 1966; Heyman 1997] which states that if a set of internal forces can be found that equilibrate the external loads and are contained within the masonry, then the structure won't collapse. Computer-aided approaches for designing self-supporting structures are a recent development. Two notable methods are the hanging chain method [Kilian and Ochsendorf 2005] and thrust network analysis [Block and Ochsendorf 2007]. The former utilizes a particle-spring system to find the equilibrium positions of nodes. The latter combines the force density method [Schek 1974] with the idea of the reciprocal diagram. Both methods are discrete — based upon polygonal curves or networks. The continuous analog, *Airy stress potential*, is studied in [Fraternali et al. 2002; Fraternali 2010; Vouga et al. 2012]. By analyzing polyhedral stress potential and optimizing the topology of a discrete network via a regular triangulation, our method provides a good approximation to the continuous notion.

Self-supporting approximation To match a self-supporting surface to an arbitrary target shape, Block and Lachauer [2011] fix the input mesh's xy-coordinates and optimize the reciprocal diagram only. Vouga et al. [2012] simultaneously optimize the xy-coordinates of an input mesh and the edge lengths in the reciprocal diagram via an alternating optimization strategy. Both methods need an initial polyhedral mesh and do not change its connectivity as the optimization proceeds.

Network topology adaptation Mesh connectivity influences shape and determines force distributions, and is therefore critical in designing and analyzing compression-only (self-supporting) and tension-only (membrane) surfaces and frameworks [ODwyer 1999][Kilian 2006, chapter 5]. It is desirable to automatically evolve rather than specify the optimal network topology. On the other hand, topology adaptation strategies have been studied in surface and volume meshing. Methods include centroidal Voronoi tessellation (CVT) [Du et al. 1999; Liu et al. 2009], optimal Delaunay triangulation (ODT) [Chen and Xu 2004; Alliez et al. 2005] and Hodge-optimized triangulation (HOT) [Mullen et al. 2011]. CVT and ODT determine topology by a (restricted) Delaunay triangulation, while HOT utilizes the more powerful regular triangulation.

Regular triangulation in geometry processing Cheng and Dey use regular triangulations to eliminate slivers and protect features in a tetrahedral mesh [2004]. Mullen et al. propose HOT-meshes [2011] which encode the primal and dual of a regular triangulation, and increase accuracy in many meshing applications such as finite element analysis. Regular triangulation has also been applied to generate blue noise point sets [de Goes et al. 2012].

Statics-based geometric modeling Smith et al. present a nonlinear optimization method for designing truss structures [2002]. Umetani et al. consider physical stability in furniture design [2012]. The TNA method [Block and Ochsendorf 2007] provides a flexible way to design and analyze compression-only structures. Statics-aware planar quadrilateral (PQ) meshing is considered in [Schiftner and Balzer 2010] for architectural geometry. Whiting et al. apply static analysis on existing masonry buildings to improve their structural stability [2009; 2012]. Whiting et al. 's method supports structures with more general 3D topology but makes it difficult to guide or explore in the shape space. Our method inherits the flexibility of the TNA method for shape design as well as its restriction to structures with a 2D primal diagram.

3 Triangular Self-Supporting Surfaces

We introduce the thrust network approach [Block and Ochsendorf 2007] in Section 3.1 and regular triangulation in Section 3.2. We then show the theoretical connection between them, motivating our idea for using regular triangulation to parameterize self-supporting meshes in Section 3.3. Methods to manipulate this parameterization and handle boundary conditions are presented in Section 3.4.

3.1 Thrust network equilibrium and reciprocal diagram

We restrict our study to structures with manifold mesh topology. A 3D structure is represented by a polyhedral mesh $\mathcal{M}$. We assume that the vertical (gravitational) load applies only at vertices of $\mathcal{M}$, and that there is no external horizontal load. The self-supporting property requires that every vertex be in force equilibrium. At a specific mesh vertex $\mathbf{v}_i = (x_i, y_i, z_i)$, this condition is:

$$\sum_{j\sim i} r_{ij}(x_j - x_i) = \sum_{j\sim i} r_{ij}(y_j - y_i) = 0, \tag{1}$$

$$\sum_{j\sim i} r_{ij}(z_j - z_i) = F_i. \tag{2}$$

Here, $j \sim i$ denotes the index of one-ring neighbor vertices to $\mathbf{v}_i$. F_i is the vertical load applied to $\mathbf{v}_i$. r_{ij} is the *force density* on an edge: the ratio of horizontally projected force over horizontally projected edge length. It satisfies $r_{ij} = r_{ji}$. For a compression-only structure, $r_{ij} \geq 0$. $\mathcal{M}$ must be a height field, or at least locally a height field with possible global overlaps in xy, to ensure horizontal equilibrium.

Horizontal equilibrium in Eq. (1) implies that the set of density-weighted edges around a given vertex form a closed polygon. The direction and length of this polygon's j-th edge are respectively $\frac{\mathbf{v}'_j - \mathbf{v}'_i}{\|\mathbf{v}'_j - \mathbf{v}'_i\|}$ and $r_{ij}\|\mathbf{v}'_j - \mathbf{v}'_i\|$, where $\mathbf{v}'$ denotes orthogonal projection of $\mathbf{v}$ to the xy-plane. Assembling all these polygons forms a *reciprocal diagram* [Maxwell 1869; Cremona 1890].

Let $\mathcal{M}'$ be the 2D orthogonal projection of $\mathcal{M}$, called the *primal diagram*. Based on these definitions, it is easy to see that the primal and reciprocal diagrams have a simple relation: *any edge of the primal diagram has a dual edge in the reciprocal diagram and the two edges are parallel*. Fig. 2 illustrates their relationship.

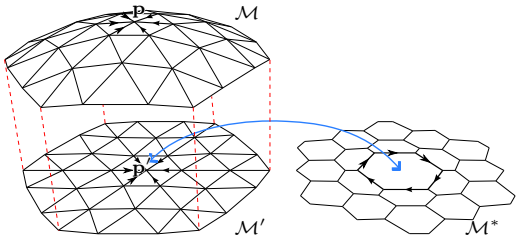


Figure 2: *A primal diagram (left) and its reciprocal diagram (right). Six axial forces at vertex* $\mathbf{p}$ *attain horizontal equilibrium and so form a closed polygon in the reciprocal diagram.*

Given a primal diagram $\mathcal{M}'$ and a reciprocal diagram $\mathcal{M}^*$, thrust network analysis finds a unique self-supporting surface $\mathcal{M}$ via the following simple method. The primal diagram already specifies the xy-components of vertices and their connectivity in $\mathcal{M}$. All that remains is to compute the height (z component) for each of its vertices. Force density is given by

$$r_{ij} = \|\mathbf{e}^*_{ij}\|/\|\mathbf{e}_{ij}\|, \tag{3}$$

where $\|\mathbf{e}^*_{ij}\|$ and $\|\mathbf{e}_{ij}\|$ are the lengths of the corresponding dual and primal edges. Vertex heights can then computed by solving the linear system represented by Eq. (2). Some vertices should be fixed in the structure to provide support.

The reciprocal diagram corresponding to a given primal diagram is not unique because of the choice of force densities; the space of reciprocal diagrams offers the proper flexibility for designing discrete self-supporting surfaces [Block and Ochsendorf 2007].

3.2 Power diagram and regular triangulation

In $\mathbb{R}^2$, $\mathrm{d}(\mathbf{x},\mathbf{p}) = \|\mathbf{x}-\mathbf{p}\|^2 - w_\mathbf{p}$ represents the *power distance* between points $\mathbf{x}$ and $\mathbf{p}$, where $w_\mathbf{p}$ is a weight associated to $\mathbf{p}$. This definition generalizes standard Euclidean distance. The tuple $(\mathbf{p}, w_\mathbf{p})$ can be regarded as a *power circle* with center $\mathbf{p}$ and radius $\sqrt{w_\mathbf{p}}$. Given a set of n such circles $\{(\mathbf{p}_i, w_i)\}$, Euclidean space can be partitioned into n regions with respect to the power distance:

$$\mathcal{V}_i \triangleq \{\mathbf{x} \in \mathbb{R}^2 \,|\, \mathrm{d}(\mathbf{x},\mathbf{p}_i) \le \mathrm{d}(\mathbf{x},\mathbf{p}_j), \forall j \neq i\}.$$

This partition is called a *power diagram* [Aurenhammer 1987a], $\mathcal{V}_i$ is called a *power cell*. A power cell can be empty; its associated vertex is called *hidden*.

The dual to the power diagram is constructed by connecting $\mathbf{p}_i$ and $\mathbf{p}_j$ if $\mathcal{V}_i$ and $\mathcal{V}_j$ are adjacent. The result is called the *regular triangulation*. Alternatively, the regular triangulation can be defined as the orthogonal projection (to the xy-plane) of the convex hull of 3D points $\{\widetilde{\mathbf{p}}_i\}$ where $\widetilde{\mathbf{p}}_i \triangleq (\mathbf{p}_i, (\|\mathbf{p}_i\|^2 - w_i)/2)$ [Aurenhammer 1987a]. The 3D point $\widetilde{\mathbf{p}}_i$ is called the *lifted point* of $\mathbf{p}_i$. The lifted point of a hidden vertex lies inside the interior of the convex hull of the other lifted vertices, so that it disappears from the triangulation. Fig. 3 shows a typical power diagram and its corresponding regular triangulation.

When all w_i are equal, the power diagram and regular triangulation revert to the standard Voronoi diagram and Delaunay triangulation. Adding a constant value to all weights yields the identical power diagram and regular triangulation.

Any triangle $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ in the regular triangulation possesses a *weighted circumcenter* $\mathbf{c}_{ijk}$ formed by the intersection of any two bisector lines $\mathrm{d}(\mathbf{x},\mathbf{p}_i) = \mathrm{d}(\mathbf{x},\mathbf{p}_j)$ and $\mathrm{d}(\mathbf{x},\mathbf{p}_i) = \mathrm{d}(\mathbf{x},\mathbf{p}_k)$. $\mathbf{c}_{ijk}$ is also the shared vertex of $\mathcal{V}_i$, $\mathcal{V}_j$, and $\mathcal{V}_k$. $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ and $\mathbf{c}_{ijk}$ are called *dual* to each other. The edge $\overline{\mathbf{c}_{ijk}\mathbf{c}_{ijl}}$ in the power diagram

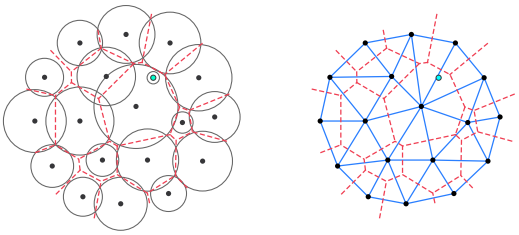


Figure 3: *Left: power circles of vertices and the resulting power diagram (edges in red). Right: the corresponding regular triangulation (edges in blue). The vertex colored in green is hidden and has no influence on the resulting regular triangulation.*

is dual to the shared edge $\mathbf{e}_{ij} \triangleq \overline{\mathbf{p}_i\mathbf{p}_j}$ between adjacent triangles $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ and $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_l$. From these definitions it is easy to see that dual edges must be orthogonal.

Local regularity and the orthogonal dual Given a planar triangle mesh $\mathcal{T}$ and a weight associated with each vertex, a dual diagram $\mathcal{D}$ can be constructed by connecting each pair of weighted circumcenters between two adjacent triangles. $\mathcal{D}$ is called an *orthogonal dual* of $\mathcal{T}$. If all the cells of $\mathcal{D}$ are convex, we say $\mathcal{T}$ preserves *local regularity* everywhere. Local regularity generalizes the previous definition of regular triangulation — a regular triangulation is identical to a locally regular $\mathcal{T}$ containing no holes or overlaps and having a convex boundary. Local regularity implies that there is positive (non-stretching) thrust on each edge and so correctly yields a self-supporting structure, even in the presence of holes or (non-height-field) overlaps. Hereafter, we use the term "regular" to mean "locally regular" and "power diagram" to refer to the dual diagram constructed in this way.

Constructing the regular triangulation Various methods exist for computing the 2D power diagram and regular triangulation [Aurenhammer 1987a]. The edge flipping algorithm [Lawson 1977] is a powerful method to convert an arbitrary planar triangulation to a Delaunay triangulation. It can also be generalized to regular triangulation. An edge of a triangle mesh is *flippable* if it is not locally regular and the quadrilateral formed by its adjacent triangles is convex. Local regularity of an edge $\mathbf{e}$ between $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ and $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_l$ is characterized by the signed edge length ratio $r_\mathbf{e} \triangleq \frac{|\mathbf{c}_{ijk}-\mathbf{c}_{ijl}|}{|\mathbf{e}|}$ [Wardetzky et al. 2007]: if $r_\mathbf{e} \ge 0$, $\mathbf{e}$ is locally regular. If no hidden vertices exist, flipping all such flippable edges iteratively leads to the regular triangulation [Glickenstein 2005]. Hidden vertices can be handled by the following augmented algorithm.

- Step 1. Flip all flippable edges iteratively until no flippable edge remains.
- Step 2. If there is an non-locally regular edge but the associated quadrilateral is concave, the vertex at the quadrilateral's concave corner must be a hidden point. Remove it and its adjacent triangles, re-triangulate the hole, and goto Step 1.
- Step 3. Stop and output the regular triangulation.

Although the above algorithm is inefficient (Step 1 runs in time $\mathcal{O}(n^2)$ in the worst case while an optimal method runs in time $\mathcal{O}(n \log n)$ [Aurenhammer 1987a]), it computes a locally regular triangulation and thus handles 2D triangle meshes with holes and overlaps.

3.3 Parameterizing thrust networks with regular triangulations

The space of regular triangulations and reciprocal diagrams are related. Let $\mathbf{P} \triangleq \{\mathbf{p}_i\}_{i=1}^n$ represent a set of 2D vertices in the primal diagram of a polygonal, self-supporting surface mesh, forming a

manifold with boundary and possible holes. We also make a reasonable assumption that no primal faces is degenerate, i.e., areas of faces are not zero and no three consecutive vertices on a face are collinear. Let $\mathcal{P}$ and $\mathcal{P}^*$ represent the space of regular triangulations and compression-only reciprocal diagrams on $\mathbf{P}$, respectively. The following proposition relates these two spaces.

Proposition 1 *A surjection exists from $\mathcal{P}$ to $\mathcal{P}^*$ if vertices on holes and concave parts of the exterior boundary are unsupported. A lower bound on the dimension of the kernel of this surjection is* 2.

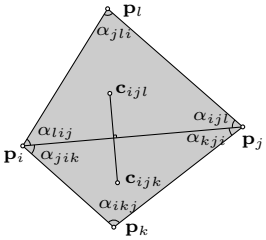
We sketch the proof as follows. (1) regular triangulation $\Longrightarrow$ reciprocal diagram. Since each edge of the power diagram is orthogonal to the primal edge and the power cells of interior vertices are closed, the power diagram must be a reciprocal diagram rotated 90° around the z-axis. (2) reciprocal diagram $\Longrightarrow$ regular triangulation. Without loss of generality, assume $\mathcal{M}'$ is triangular since any non-triangular primal diagram can be triangulated by adding edges. We add edges to triangulate the spaces formed by any holes and concave exterior segments, forming a simply-connected mesh with convex boundary, which we call the *extended* primal diagram. Added edges correspond to dual edges with zero lengths. The primal diagram so extended must be a regular triangulation according to a well-known theorem: *a reciprocal diagram for a triangulation exists if and only if the triangulation is regular* [Aurenhammer 1987b; Glickenstein 2005; Wardetzky et al. 2007]. This theorem assumes the reciprocal diagram is simply-connected; i.e., without holes.

We show how to compute power weights from the reciprocal diagram and count the dimensionality from $\mathcal{P}^*$ to $\mathcal{P}$. Applying the formula for the diagonal Hodge star operator in a regular triangulation [Mullen et al. 2011, page 7] to the shared edge $\mathbf{e}_{ij} = \mathbf{p}_i - \mathbf{p}_j$ between adjacent triangles $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ and $\triangle\mathbf{p}_i\mathbf{p}_j\mathbf{p}_l$, we obtain:

$$(\star^1)_{ij} \triangleq \operatorname{sgn}((\mathbf{c}_{ijk} - \mathbf{c}_{ijl}) \cdot \mathbf{e}_{ij}^{\perp}) \frac{\|\mathbf{c}_{ijk} - \mathbf{c}_{ijl}\|}{\|\mathbf{e}_{ij}\|}, \tag{4}$$

$$= \frac{1}{2}\Big(\cot\alpha_{ikj} + \cot\alpha_{jli} \tag{5}$$
$$+ (w_i - w_k)\frac{\cot\alpha_{kji}}{\|\mathbf{e}_{ij}\|^2} + (w_j - w_k)\frac{\cot\alpha_{jik}}{\|\mathbf{e}_{ij}\|^2}$$
$$+ (w_i - w_l)\frac{\cot\alpha_{ijl}}{\|\mathbf{e}_{ij}\|^2} + (w_j - w_l)\frac{\cot\alpha_{lij}}{\|\mathbf{e}_{ij}\|^2}\Big).$$

Here $\mathbf{e}_{ij}^{\perp}$ denotes an edge in the regular triangulation rotated 90° around the z-axis, and $\mathbf{c}_{ijk}$, $\mathbf{c}_{ijl}$ denote the weighted circumcenters of the adjacent triangles. Comparing Eqs. (3) and (4), and recalling that an edge in the power diagram is $\mathbf{e}_{ij}^* = \mathbf{c}_{ijk} - \mathbf{c}_{ijl}$, we see that the Hodge star operator yields a signed version of the edge's force density, r_{ij}. It is also known that the triangulation is regular if and only if all Hodge star edge evaluations are nonnegative [Wardetzky et al. 2007]. This implies that the reciprocal diagram has a corresponding power diagram, since $r_{ij} \geq 0$ for all its edges. We next show that there exists a set of weights in a regular triangulation such that each of its edge's Hodge star evaluations matches the reciprocal diagram's force density, or $(\star^1)_{\mathbf{e}} = r_{\mathbf{e}}$, via Eq. (5). On edges added to triangulate non-triangular cells, holes, and concave boundary, $(\star^1)_{\mathbf{e}} = 0$; no axial force should be applied. The linear equations formed by $\{(\star^1)_{\mathbf{e}} = r_{\mathbf{e}}, \forall\mathbf{e}\}$ characterizes the mapping $\mathcal{P}^*$ to $\mathcal{P}$.



Suppose that the extended primal diagram contains N_e edges, N_f faces, N_i interior vertices, and N_b boundary vertices. The total number of vertices is denoted $N_v = N_b + N_i$. Without loss of generality, assume no face contains only supported boundary vertices. The equations $(\star^1)_{\mathbf{e}} = r_{\mathbf{e}}$ form a $(N_e - N_b) \times N_v$ linear system with respect to the weights w_i. But in each dual cell polygon, two of its edges are determined by the rest. The reason is that the reciprocal diagram's edge directions are fixed regardless of the weight assignment, and its edge vectors sum to zero. So any two of the cell's edge vectors are uniquely determined by decomposing the negative sum of its remaining vectors along the two given edge directions. This property is called the *closure condition* of the polygon. Thus $2N_i$ of these equations can be eliminated from the linear system.

The number of rows thus reduces to $N_e - N_b - 2N_i = N_f - N_i - 1$.[1] By the shift invariance property of the power diagram, we can fix one weight to obtain a $(N_f - N_i - 1) \times (N_v - 1)$ linear system. The lower bound on the dimensionality of its solution space is $(N_v - 1) - (N_f - N_i - 1) = 2$. Deriving this last step again applies the edge-counting formula.■

We conjecture that *a unique weight assignment exists if there are no degenerate triangles and we fix the 2D translation of the reciprocal diagram*. In other words, the dimension of the kernel of the surjection is 2 exactly. We have numerically verified this conjecture by computing the SVD of the linear system in many examples. This means we can impose as a constraint the 2D location of one weighted circumcenter in the power diagram to make the solution of the above linear system unique. The power diagram so determined matches the given reciprocal diagram.

Limitation: When there are supported vertices on holes or concave parts of the exterior boundary, it is possible that no regular triangulation exists. For some reciprocal diagrams, interior vertices of the extended primal diagram cannot all be in 2D equilibrium, no matter how we extend the primal diagram to a simply-connected convex domain or assign forces on the added edges. In other words, we cannot extend the reciprocal diagram to be simply-connected. The linear system formed by Eq.(5) has no solution. This limitation is less significant than it may appear. In Section 3.4, we describe a way to find a set of power weights that directly fit a given self-supporting mesh rather than a reciprocal diagram. By allowing some flexibility in satisfying vertical equilibrium, we typically find a regular triangulation that reproduces the mesh. This implicitly amounts to perturbing the reciprocal diagram to find one that has a valid extension.

Remark 1. The definition of power diagram and regular triangulation can be generalized. Given a triangle mesh and a set of per-vertex weights, the *general power diagram* connects adjacent weighted circumcenters of triangles. Rotating it 90° horizontally, we obtain a generalized reciprocal diagram. Self-intersecting or nonconvex cells can appear. A dual edge whose direction is opposite to its corresponding primal edge implies a negative force density. This generalization makes it possible to model self-supporting structures allowing tension as well as compression.

3.4 Computing the parameterization

Computing power weights from a reciprocal diagram For a primal and reciprocal diagram pair satisfying the conditions of Prop. 1, we show how to compute a set of power weights that reproduce it. The following procedure selects $N_e - N_b - 2N_i$ linearly independent constraints from edges in the extended primal diagram.

[1] This can be derived by applying Euler's formula for a simple-connected triangular mesh with boundary: $N_f + N_v - N_e = 1$. The derivation also applies the formula $2N_e - N_b = 3N_f$ for triangular meshes, derived by counting edges on all triangles, which counts interior edges twice and boundary edges once.

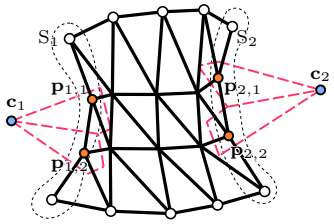


Figure 4: *Boundary conditions for a self-supporting mesh. A convex boundary vertex (white circle) must be supported. In order to achieve horizontal equilibrium, each unsupported concave boundary vertex (orange circle) must possess a closed power cell, drawn in red. All cells along the same segment must share a common dual vertex (blue circle).*

1. Initially we have an equation for each inner edge. By the closure condition, we can remove two equations for each inner vertex v_i corresponding to two of its non-parallel edges. We ensure that the edges we remove are new (not previously removed from a vertex visited earlier) by tagging edges.
2. We emit the equality constraint $(\star^1)_{\mathbf{e}} = r_{\mathbf{e}}$ for each selected edge $\mathbf{e}$ using Eq. (5), where $r_{\mathbf{e}}$ is the known force density on $\mathbf{e}$.

We then assemble these constraints as a linear system $AW = B$ where W is the vector of unknown weights. Due to the shift invariance property, we set one weight to zero. Based on Prop. 1 and our conjecture, we can solve the linear system directly by specifying the 2D coordinate of one weighted circumcenter.

Computing power weights from a self-supporting mesh For a self-supporting mesh $\mathcal{M}$, we can find a set of weights that approximates $\mathcal{M}$ by solving an inequality-constrained quadratic programming (QP) problem:

$$\arg\min_{W} \sum_{v_j} \sum_{j\sim i} ((\star^1)_{ij}(z_i - z_j) - F_i)^2$$

under $(\star^1)_{ij} \geq 0, \forall \mathbf{e}_{ij}$. Here W is the vector of unknown weights and v_j are the unsupported vertices. One weight is set to zero due to shift invariance. The objective function penalizes vertical disequilibrium. Inequalities maintain the connectivity of the input triangulation. Horizontal equilibrium of unsupported vertices is implicitly maintained by parameterizing with a regular triangulation; no additional equality constraints need be imposed. If the objective vanishes, then the above optimization finds a power diagram that is a 90° rotated reciprocal diagram for M.

Boundary conditions Weights and locations of vertices on the boundary (including its exterior extent and interior holes) of a primal diagram must satisfy certain constraints. We first consider the exterior boundary and then discuss holes. Boundary vertices can be supported (in contact with the ground) or unsupported (hanging freely above the ground) as shown on the left side of Fig. 1.

- Boundary vertices are categorized as *convex* and *concave*. Boundary edges at a convex vertex form an outside angle no less than π; the outside angle at a concave vertex is smaller than π. Convex vertices can not be in horizontal equilibrium with compression-only forces and must be supported.
- Unsupported concave boundary vertices can be divided into segments, separated by chains of supported vertices. Every segment includes two supported vertices at its ends. Refer to Fig. 4. For each segment $\mathrm{S}_k : \{\mathbf{p}_{k,i}\}$, the weighted bisector lines of each of its edges must intersect at a common point since the adjacent power cells share the same dual edge. More precisely, a point $\mathbf{c}_k = (a_k, b_k)$ must exist satisfying $(\mathbf{p}_{k,i} - \mathbf{c}_k)^2 - w_{k,i} = (\mathbf{p}_{k,i+1} - \mathbf{c}_k)^2 - w_{k,i+1}$ for $i = 1, \ldots, n_k - 1$. These equations expand to $\mathbf{p}_{k,1}^2 - w_{k,1} - 2\mathbf{c}_k \cdot \mathbf{p}_{k,1} = \cdots = \mathbf{p}_{k,n_k}^2 - w_{k,n_k} - 2\mathbf{c}_k \cdot \mathbf{p}_{k,n_k}$. Letting $h_k = (\mathbf{p}_{k,1}^2 - w_{k,1})/2 - \mathbf{c}_k \cdot \mathbf{p}_{k,1}$, we conclude that all lifted points $(\mathbf{p}_{k,i}, (\mathbf{p}_{k,i}^2 - w_{k,i})/2)$ lie on the plane $z = a_k\, x + b_k\, y + h_k$. The *segment tuple* (a_k, b_k, h_k) parameterizes weights and vertex locations on the boundary S_k; h_k is called the *segment plane offset*.
- Two segments sharing the same supported vertex must also share the same segment plane offset.
- (a_k, b_k) is actually a vertex in the power diagram. For a boundary primal edge $\mathbf{e}$ connecting to two unsupported vertices, connecting the circumcenter of its adjacent face to (a_k, b_k) determines the dual edge $\mathbf{e}^*$.

These constraints similarly apply to vertices on holes. There is one exception. When a hole has one or zero supported vertices, the corresponding segment has only one or zero end points. In this case, a single segment tuple parameterizes weights and locations of all vertices on the hole.

Computing the 3D mesh We also want to perturb self-supporting surfaces to explore their neighboring space. One possible method to do this is to first modify the vertices of the reciprocal diagram without considering the constraint of edge parallelism in the primal diagram, and then to apply this constraint by minimizing the change in force densities via an iterative algorithm [Rippmann et al. 2012]. The self-supporting surface is finally computed by solving Eq. (2). This method fixes the primal diagram's edge connectivity. Our approach using regular triangulation lets us edit weights and segment tuples freely while preserving equilibrium. Edge connectivity varies naturally as those weights change.

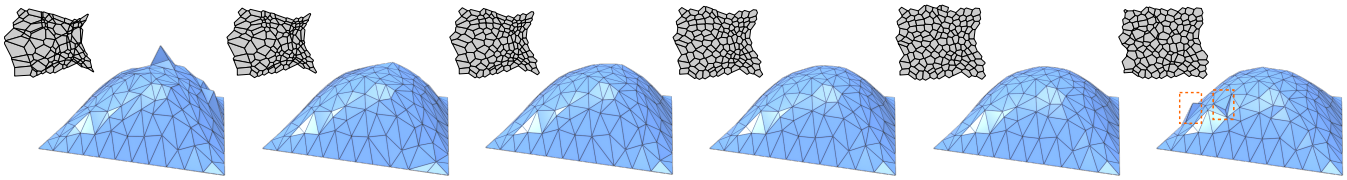
Given a set of 2D points with weights $(\mathbf{p}_i, w_i)$ that satisfy boundary conditions, boundary polylines, and the 3D coordinates of all supported points, we compute the corresponding self-supporting polyhedral mesh as follows:

1. Create a 2D initial triangulation connecting points and boundary lines via constrained Delaunay triangulation [cga].
2. Apply the edge flipping algorithm (see Section 3.2) to convert it to a regular triangulation. The result serves as the primal diagram.
3. Construct its dual. For each triangular face of the primal diagram, compute its weighted circumcenter. This gives the 90°-rotated reciprocal diagram.
4. Compute force densities via the edge length ratio in Eq. (3).
5. Solve the linear system in Eq. (2) by sparse LU decomposition to find the heights of unsupported vertices.

Hidden vertices can appear when some weights have especially small or large values relative to the rest. This wastes degrees of freedom: more hidden vertices lead to a lower-resolution mesh. The presence of hidden vertices is efficiently detected during the edge flipping algorithm.

4 Self-supporting geometry processing

The discrete differential geometry of self-supporting surfaces is studied in [Vouga et al. 2012]. Polyhedral stress potential plays an important role. We review and relate it to regular triangulation in Section 4.1, and then present our methods for self-supporting smoothing and remeshing in Sections 4.2 and 4.3. Smoothing changes both the stress potential and the shape of the result, while remeshing preserves shape while improving mesh topology to better match the stress distribution.



(a) (b) (c) (d) (e) (f)

Figure 5: *Smoothing results on a self-supporting surface with supported boundary. In this example, H=1 and all F_i=0.03. Spikes in the initial surface (a) result from variations in the size of power cells. The result of our smoothing method after 1, 5, 15, 35 iterations is shown in b, c, d, e respectively, with corresponding reciprocal diagrams. Simple averaging of weights via $w_i' = (\sum_{j\sim i} w_j)/(\sum_{j\sim i} 1)$ is shown in (f). Such averaging does not converge and even after 500 iterations fails to eliminate all the rough edges and spikes (dashed red boxes).*

4.1 Polyhedral stress potential

The polyhedral stress potential Φ is the discrete analog for polygonal meshes of the continuous (Airy) stress potential for a smooth self-supporting surface [Fraternali et al. 2002; Fraternali 2010]. Φ is represented as a convex polyhedral mesh with 3D vertices $(\mathbf{p}_i, \Phi_i)$ and edge connectivity identical to the self-supporting mesh $\mathcal{M}$. A face of Φ lies on the plane $z = \alpha\, x + \beta\, y + \gamma$ where (α, β) is the dual vertex in the 90°-rotated reciprocal diagram. Φ is related to the reciprocal diagram and force densities: $\|\mathbf{e}_{ij}^*\| = r_{ij}\|\mathbf{e}_{ij}\|$ represents the jump in Φ's derivative when crossing an edge [Vouga et al. 2012].

We can construct Φ based on a regular triangulation and power diagram. Lifting a primal face (i.e. a face in the regular triangulation) $\triangle \mathbf{p}_i\mathbf{p}_j\mathbf{p}_k$ yields a 3D triangle through $\widetilde{\mathbf{p}}_i$, $\widetilde{\mathbf{p}}_j$, and $\widetilde{\mathbf{p}}_k$. It is easy to verify that this face satisfies the plane equation $z = \mathbf{c}_x\, x + \mathbf{c}_y\, y + \gamma$ where $(\mathbf{c}_x, \mathbf{c}_y)$ is the triangle's weighted circumcenter (i.e. a vertex in the power diagram). Φ is the piecewise union of all such lifted triangles. It can be regarded as a piecewise-linear discretization of the continuous locally convex function $\Phi(x, y) = (x^2 + y^2 - w(x, y))/2$, where $w(x, y)$ is a continuous weight function.

4.2 Shape smoothing

Large variations in reciprocal cell size tend to produce a "rough" mesh with spikes and creases. Direct smoothing, such as by applying the Laplacian operator on vertices, does not respect the self-supporting property. We present a novel method that effectively smooths self-supporting meshes. To do this, we measure curvature and try to hold it constant.

The isotropic mean curvature of a self-supporting surface with respect to its stress potential is defined as $H^{\text{rel}}(\mathbf{v}_i) = \Delta\mathcal{M}(\mathbf{p}_i)/\Delta\Phi(\mathbf{p}_i)$ [Vouga et al. 2012], where Δ is the (discrete) Laplacian operator. The notation $\mathcal{M}(\mathbf{p})$ represents the height function of the mesh $\mathcal{M}$ and $\mathbf{v}_i = (\mathbf{p}_i, \mathcal{M}(\mathbf{p}_i)) = (\mathbf{p}_i, z_i)$ one of its 3D vertices. The curvature can be expressed as $H^{\text{rel}}(\mathbf{v}_i) = 2F_i/\mathcal{A}_i$ where $\mathcal{A}_i$ is the area of the power cell associated with $\mathbf{p}_i$. When H^{rel} is constant everywhere, we call the resulting surface *constant isotropic mean curvature* (CIMC) with respect to Φ. We conclude that a self-supporting triangular surface is CIMC if and only if

$$2F_i = \mathcal{A}_i\, H \tag{6}$$

at all vertices $\mathbf{v}_i$. The constant H represents the mean curvature.

Given a polyhedral self-supporting surface $\mathcal{M}$ parameterized by a regular triangulation, a curvature constant H, and a set of vertical loads F_i on vertices, we develop the following method to evolve $\mathcal{M}$ to an CIMC surface by incrementally updating weights. The xy-coordinates of vertices remain fixed through the iteration to preserve the original 2D design. The iteration produces much more uniformly-sized reciprocal cells and thus a smoother self-supporting mesh result. It evolves power diagram weights and updates the mesh using the method described in Section 3.4 ("Computing the 3D mesh").

Since $\frac{1}{4}(\star^1)_{ij}\,(\|\mathbf{e}_{ij}\|^2 - w_j + w_i)$ is the signed area of the triangle formed by $\mathbf{p}_i$ and the two weighted circumcenters of faces adjacent to edge $\mathbf{e}_{ij}$, it is not difficult to show that

$$\begin{aligned} 4\mathcal{A}_i &= \sum_{j\sim i}(\star^1)_{ij}\,\|\mathbf{e}_{ij}\|^2 - \sum_{j\sim i}(\star^1)_{ij}\,(w_j - w_i) \\ &= \sum_{j\sim i}(\star^0)_{ij}\,\|\mathbf{e}_{ij}\|^2 + \sum_{j\sim i}\left[(\star^1)_{ij} + (\star^0)_{ij}\right]\,(w_i - w_j) \end{aligned} \tag{7}$$

Here $(\star^0)_{ij} \triangleq \frac{1}{2}(\cot\alpha_{ikj} + \cot\alpha_{jli})$ is the cotan Laplacian operator (refer to the inset in Prop. 1) and $(\star^1)_{ij}$ is the Hodge star operator previously defined in Eq. (5). Applying Newton iteration on the equality constraint in Eq. (6) yields the following update rule for the vertex weight w_i:

$$w_i' = w_i + \lambda\frac{E_i - \sum_{j\sim i}\left[(\star^1)_{ij} + (\star^0)_{ij}\right](w_i - w_j)}{\sum_{j\sim i}\left[(\star^1)_{ij} + (\star^0)_{ij}\right] + \sum_{j\sim i}\partial_{w_i}(\star^1)_{ij}(w_i - w_j)}. \tag{8}$$

where $E_i = 8F_i/H - \sum_{j\sim i}(\star^0)_{ij}\,\|\mathbf{e}_{ij}\|^2$ and λ is the damping parameter with initial value 1. Iteratively applying this formula to each vertex yields a progressively smoother self-supporting mesh.[2] Our implementation currently fixes the weights at boundary vertices.

Hidden points may appear as the iteration proceeds. We do a simple adjustment after each iteration to handle them. It finds the triangle in which the hidden vertex lies and resets its weight so that its lifted point lies exactly on the corresponding lifted triangle.

The above method does not prevent changes of mesh topology which may be undesirable in some applications. λ can be decreased until no edge swap is detected when computing the regular triangulation.

If H is not specified, we can estimate it by the median curvature value from $\{2F_i/\mathcal{A}_i\}_{i=1}^n$. Fig. 5 shows the results of our weight updating scheme; a few iterations significantly reduce spikes and creases.

4.3 Shape remeshing

The force distribution of a self-supporting structure at equilibrium depends on the topology of its primal diagram [Block 2009, Chapter 4]. An arbitrary mesh can greatly exaggerate force densities. We

[2] Updating the weights can alter the connectivity implied in $j \sim i$ and changes the evaluation of Hodge star and Laplacian edge operators in subsequent iterations. The regular triangulation and its dual needs to be recomputed after each iteration.

seek a discretization that provides better guidance about the strength and load capacity of a smooth surface structure. Our remeshing method improves the accuracy of stress estimation and typically reduces the magnitude of force density. This avoids overdesign and allows a better tradeoff between strength and expense in the choice of building material.

We observe that a major source of error in stress and force density approximation is the fact that the stress potential has been polygonally discretized. Our method works by minimizing the difference between the polyhedral stress potential Φ and a smooth version of that same function, Φ_0. It comprises the following two steps.

1. Stress potential estimation A self-supporting surface mesh $\mathcal{M}$ is specified as input. If a smooth target stress potential Φ_0 is not specified directly, we can define it given per-vertex weights on $\mathcal{M}$ via

$$\Phi_i \triangleq \Phi(\mathbf{p}_i) \triangleq (x_i^2 + y_i^2 - w_i)/2. \tag{9}$$

Per-vertex weights w_i can be computed from $\mathcal{M}$ using the procedure in Section 3.4. We then employ modified butterfly subdivision [Zorin et al. 1996] to define a smoother mesh Φ_0 that interpolates these discrete values Φ_i. Our implementation subdivides the discrete stress potential twice and uses that result to evaluate Φ_0.

Weights can then be computed from the smooth potential Φ_0 at any xy-location by evaluating

$$w_0(x, y) \triangleq x^2 + y^2 - 2\,\Phi_0(x, y). \tag{10}$$

2. Discretization error minimization We measure error from stress potential discretization defined as

$$E \triangleq \int_\Omega |\Phi(x, y) - \Phi_0(x, y)|\,\mathrm{d}x\,\mathrm{d}y, \tag{11}$$

where Ω is the 2D domain (footprint) of the mesh. Since the vertices of Φ are sampled from Φ_0 and Φ_0 is (locally) convex, Φ is always above Φ_0 in general and the absolute value operator in E can be removed. E simplifies to [Chen and Xu 2004]:

$$E = \frac{1}{n+1}\sum_i \Phi_0(\mathbf{p}_i)|\Omega_i| - \int_\Omega \Phi_0\,\mathrm{d}x\,\mathrm{d}y. \tag{12}$$

Ω_i denotes the one-ring of triangles around $\mathbf{p}_i$; $|\Omega_i|$ denotes its area. n is the number of vertices of $\mathcal{M}$. If the triangulation of the mesh is optimal in the sense of minimizing E, then for an interior vertex $\mathbf{p}_i$, following Chen's and Xu's result [2004, Theorem 3.5], we have

$$\nabla E(\mathbf{p}_i) = -\frac{1}{|\Omega_i|}\sum_{\triangle_j \in \Omega_i}\left(\nabla_{\mathbf{p}_i}|\triangle_j| \sum_{\mathbf{p}_k \in \triangle_j, \mathbf{p}_k \neq \mathbf{p}_i} \Phi(\mathbf{p}_k)\right), \tag{13}$$

where $\triangle_j \in \Omega_i$ denotes the j-th triangle in the i-th one-ring. Inspired by the approach presented in Alliez et al. 's work [2005], we define an update rule for each vertex by enforcing $\nabla E(\mathbf{p}_i) = 0$:

$$\mathbf{p}_i' = \frac{\sum_{\triangle_j \in \Omega_i} |\triangle_j|\,\mathbf{c}_{\triangle_j}}{\sum_{\triangle_j \in \Omega_i} |\triangle_j|} + \frac{\nabla_{\mathbf{p}_i} w_0(\mathbf{p}_i)}{2}, \tag{14}$$

where $\mathbf{c}_{\triangle_j}$ is the weighted circumcenter of $\triangle_j$. The equivalence of $\nabla E(\mathbf{p}_i) = 0$ and Eq. (14) can be verified by substituting the formula for weighted circumcenters in terms of power weights.

Based on these results, our iterative remeshing method works as follows.

(a) For every interior vertex $\mathbf{p}_i$ of $\mathcal{M}'$, update its position by Eq. (14).
(b) Update all weights by $w_i' \leftarrow w_0(\mathbf{p}_i)$.
(c) Update the regular triangulation of $\mathcal{M}'$. Since w_i are interpolated from the locally convex function Φ_0, no hidden points appear.
(d) Measure the maximal movement of $\|\mathbf{p}_i' - \mathbf{p}_i\| + \|w_i' - w_i\|$. If greater than a threshold $10^{-4}\,L$ where L is the diagonal length of the bounding box of the primal diagram, goto (a), otherwise goto (e).
(e) Compute the z-coordinates of $\mathcal{M}$ by solving the vertical equilibrium constraints. To assign vertical loads on unsupported vertices, we first construct a piecewise-linear load distribution function $\mathcal{F}(x, y)$ from the initial triangle mesh $\mathcal{M}^0$. It associates a load density per area (pressure) of $F_i^0/A(\mathbf{p}_i^0)$ to each vertex $\mathbf{p}_i^0$ where $A(\mathbf{p}_i^0)$ is the projected 2D area of the one-ring region of $\mathbf{p}_i^0$. A load of $A(\mathbf{p}_i)\,\mathcal{F}(\mathbf{p}_i)$ is then assigned to vertex $\mathbf{p}_i$ during the remeshing iteration. This method preserves the initial load distribution.

Fig. 6a shows a self-supporting mesh having a grid layout, which yields a "polarized" distribution of force densities. This example assumes an analytic stress potential, $\Phi(x, y) = x^2 + 5y^2 + xy$, so that power weights are known at any vertex location. Fig. 6b&c shows two remeshing results using the analytic stress potential Φ and the estimated version Φ_0, respectively. Our method produces a more balanced distribution of densities (encoded by edge width) and reduces the average force density by 24% (b) and 25% (c). Fig. 7 shows another remeshing example where the stress potential is estimated from the initial mesh (left) using modified butterfly subdivision. The remeshing result (right) better approximates the estimated stress potential and reduces average force density by 10%.

Our remeshed result also yields a more accurate stress distribution compared to an underlying smooth surface, as shown in Fig. 8. This example assumes an analytic form for the self-supporting surface, $s(x, y) = (-2x^2 - 4y^2 - xy + 1)/4$, and its stress potential, $\Phi(x, y) = (x^2 + y^2)/2$. The stress tensor $\sigma(x, y)$ can also be computed analytically; see the derivation in [Vouga et al. 2012, Sec 2.1]. We estimate face stress tensors σ_f from the primal and reciprocal diagrams and compute their deviations from the true stress tensor $\sigma(x, y)$ using the Frobenius norm. Average errors are weighted by triangle area. The formula for computing face stress tensors is provided in a supplement. We note that the analytic Φ is only used for computing the ground truth stress tensor. Our remeshing is based on stress potential estimated from the initial mesh.

Remark 2. Our remeshing technique is actually a generalized version of *optimal Delaunay triangulation* (ODT) [Chen and Xu 2004; Alliez et al. 2005]. Eq. (14) generalizes the update for-

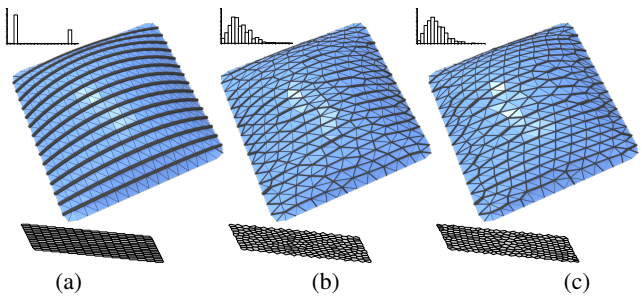


Figure 6: *Remeshing result. An initial self-supporting mesh and its reciprocal diagram are shown in (a). Using the analytic form of the stress potential, our remeshing result after 100 iterations is shown in (b). (c) shows our remeshing result using the stress potential estimated from (a). Edge thickness encodes force density; histograms of force densities are plotted above the meshes.*

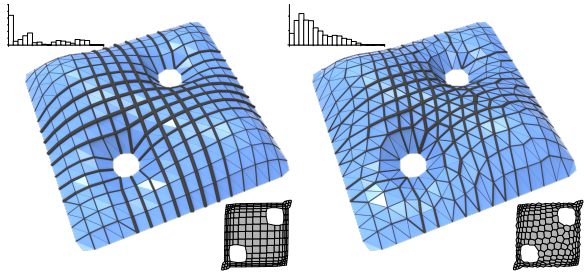


Figure 7: *Remeshing comparison. Left: a self-supporting mesh with two supported holes designed by our method as described in Section 5. The input primal mesh is a grid-like triangle mesh. Right: our remeshing distributes force density more evenly and reduces its largest magnitudes. Edge thickness encodes force density; the histograms of force densities are plotted above the meshes.*

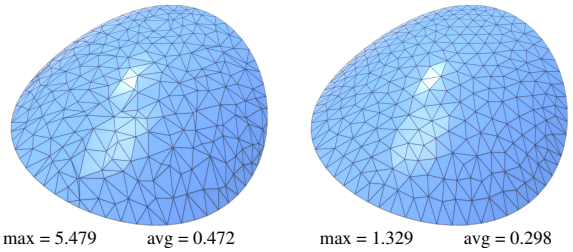
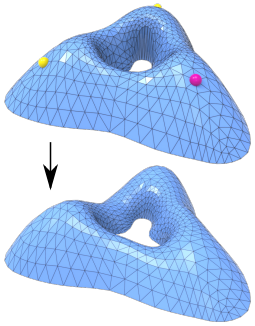


Figure 8: *Stress tensor comparison based on an analytic self-supporting surface. Left: initial mesh. Its primal mesh is computed by unstructured triangulation of the given polygonal domain. Right: remeshed result. Maximum and average deviations from the true (smooth surface) stress tensor are shown below the mesh. Remeshing provides a more accurate stress estimate.*

mula for ODT where w_0 vanishes and the weighted circumcenter degenerates to the circumcenter of the primal triangle. It can be used to remesh a 2D region equipped with an arbitrary, anisotropic metric $\nabla^2 f(x,y)$ where f is a convex function. ODT fixes this function as $f(x,y) = \frac{1}{2}(x^2+y^2)$. We simply update vertex positions using Eq. (14) iteratively with the weight function $w(x,y) = -f(x,y) + \frac{1}{2}(x^2+y^2)$.

5 Self-supporting surface editing

The regular triangulation provides a convenient parameterization for interactively creating and editing self-supporting surfaces.



Manipulating weights We provide a simple UI to directly control weights, represented by circles centered at vertices. The user selects a circle associated with one vertex $\mathbf{p}_{i^*}$ and changes its radius, corresponding to the square root of its weight. Height varies inversely with weight, falling when it rises, and rising when it falls, as a result of the vertical equilibrium condition (Eq. (2)). Weights of nearby vertices are also modified according to their distance to $\mathbf{p}_{i^*}$, via a simple exponential falloff in weight change: $\Delta w(\mathbf{p}) \triangleq \exp(-(\mathbf{p}-\mathbf{p}_{i^*})^2/\sigma)\,\Delta w(\mathbf{p}_{i^*})$. After updating the weights, a new self-supporting surface can be computed by solving for vertical equilibrium with the vertical loads using Eq. (2). Force densities r_{ij} in these equations are determined from the weights via Eqs. (3) and (5).

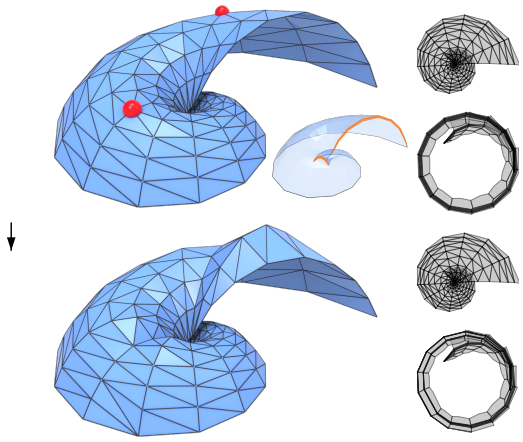


Figure 9: *Editing a sea-shell-like self-supporting mesh. The edit increases weights on the red vertices. Unsupported boundary curves are colored in orange in the middle inset. Vertical loads are kept constant. Upper: initial mesh with its primal and dual diagrams. Lower: resulting mesh after editing and two-steps smoothing with its primal and dual diagrams.*

If the mesh topology must be fixed during editing, we constrain the magnitude of the weight change via the following QP problem:

$$\underset{w_1,\ldots,w_n}{\arg\min} \sum_{i=1}^{n} (w_i - \hat{w}_i)^2, \qquad (15)$$
$$\text{Subject to } (\star^1)_{\mathbf{e}} > 0 \;\; \forall\, \mathbf{e}.$$

where $\hat{w}_i$ represents the desired weight after editing. The Hodge star inequalities preserve mesh topology. If unsupported boundary vertices exist, their segment plane constraints are appended to the edge inequalities: $(x_{k,i}^2 + y_{k,i}^2 - w_{k,i})/2 = a_k\, x_{k,i} + b_k\, y_{k,i} + h_k$. Segment plane coefficients a_k, b_k, h_k can be treated as unknown variables or fixed. Any boundary conditions and any desired vertex weights can be directly manipulated by the user.

The inset shows an example. The upper image is the initial self-supporting mesh. By increasing the weight of the red vertex and decreasing weights of the two yellow ones, we obtain the deformed self-supporting mesh shown on the bottom.

Fig. 9 shows weight editing on a sea-shell-like self-supporting mesh. The geometry in this example is not a height field globally; its primal diagram contains overlaps. A self-supporting result is correctly generated using our extended edge flipping algorithm.

Fig. 10 shows an example where we manipulate segment tuples directly and keep the connectivity of vertices. As shown in the upper power diagram, forces at unsupported boundary vertices (colored in orange) are much larger than interior forces. This happens because the dual vertices (colored in green) lie too far away from most power cells. We can directly move them towards interior power cells along the red arrows shown. Note that these dual vertices are the orthogonal projection of the segment tuples $\{(a_k, b_k, h_k)\}_{k=1}^3$ (see "Boundary conditions" in Section 3.4). Moving the green dual vertices is achieved simply by specifying the points $\{(a_k, b_k)\}$. To preserve connectivity, we solve the QP problem defined by Eq. (15) with h_1, h_2, h_3 added as variables to the set of weights. We append an additional term $\sum_{k=1}^3 (h_k - \hat{h}_k)^2$ to the objective function penalizing the change in h_k; $\hat{h}_k$ is the value before editing. The edited result (lower part of figure) shows that forces on the boundary are reduced, as expected.

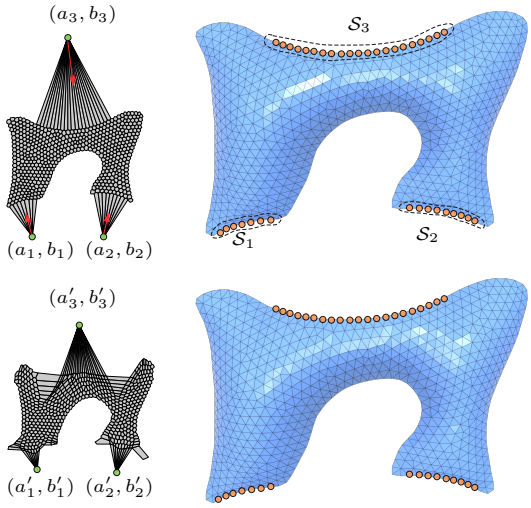


Figure 10: *Unsupported boundary edit. Upper: an initial self-supporting mesh and its power diagram. Orange circles indicate unsupported boundary vertices. All vertical loads are kept constant. Lower: the edited self-supporting mesh and its power diagram. The power diagram shows that this edit reduces large forces on the unsupported boundary.*

Creating sharp features Sharp surface features can be created by increasing or decreasing weights by the same amount over a feature curve in the 2D domain. We then fix these weights and minimize Eq. (15). Fig. 11 shows a result.

Manipulating vertex positions and mesh topology Changing vertex positions as well as weights provides greater freedom. The connectivity of the triangle mesh responds through the regular triangulation. We note that if an unsupported boundary vertex moves to become "convex", it must be supported to avoid horizontal disequilibrium.

Directly specifying mesh topology breaks horizontal equilibrium unless new weights are computed. To find them, we can again solve the QP problem in Eq. (15) with Hodge star inequality constraints. But this solver can fail: not all triangle meshes are regular [Aurenhammer 1987a]. A small random jitter on 2D vertex positions sometimes resolves the problem; otherwise, the topology operation should be rejected. This is not a problem with our parameterization, but rather the unavoidable price for directly manipulating mesh topology in the space of self-supporting surfaces.

6 System Performance

Interactive editing depends on QP solution when the mesh topology is specified. For a typical mesh with about 700 vertices, a QP problem can be solved in about 0.2 seconds using the OOQP library [Gertz and Wright 2003], on a desktop PC with a 2.83 GHz Intel Core Quad and 8 GB of RAM. This performance is acceptable for interactive design. If no predefined mesh topology is enforced, editing is possible in milliseconds; the computation involves only edge flipping to compute the regular triangulation and sparse linear equation solution to satisfy vertical equilibrium.

For self-supporting smoothing and remeshing, iterations to update weights and vertices are performed efficiently and only require computation of the regular triangulation and Hodge star edge operators. A QP problem is solved once in remeshing to generate the polyhedral stress potential; its performance depends on the size of the underlying mesh.

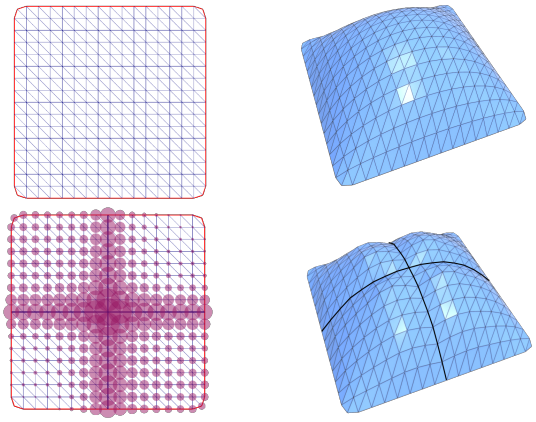


Figure 11: *Sharp feature creation. The upper image shows a primal diagram and its corresponding self-supporting surface. All the weights are zero and the boundary vertices are supported. vertical loads keep constant. The lower left image shows how weights are changed on two feature lines (drawn in black). Circle radii correspond to the square root of the weights. The resulting mesh exhibits creases as shown in the lower right.*

Figure	Vertices	Edges	Iterations	Time(seconds)
Fig. 5	172	468	35	0.05
Fig. 6b	289	800	90	0.3
Fig. 6c	289	800	100	0.1
Fig. 7	287	784	180	0.9
Inset in Section 5	758	2150	-	0.2
Fig. 9	180	483	-	0.03
Fig. 10	687	1911	-	0.2
Fig. 11	289	800	-	0.1
Fig. 1-right	758	2150	5	0.3

Table 1: *Statistics and timings.*

We list timing and other statistics from our experiments in Table 1.

7 Conclusion

A 2D regular triangulation provides a natural way to represent a 3D self-supporting mesh. Our parameterization has three advantages over previous ones. It requires fewer variables (a weight per vertex rather than a force density per edge). It encodes edge connectivity only implicitly and so lets it adapt. And it preserves equilibrium by construction.

Using this representation, we present new ways to smooth and optimize force density on a self-supporting mesh. We show how to directly manipulate our representation to interactively edit self-supporting surfaces. Our remeshing procedure provides a more faithful estimation to the capacity of self-supporting surfaces.

Our approach also inherits a basic limitation of thrust network analysis — it cannot handle non-manifold structures. It also neglects external horizontal forces. Study of a more generalized regular triangulation and its dual may lift some of these restrictions. We are interested in further investigation of discrete geometry processing for self-supporting surfaces and interactive design of stress potential. Exploring the design space of self-stressed surfaces under both tension and compression is another area for future work suggested by the generalized power diagram.

Acknowledgements

The authors would like to thank the reviewers for their valuable comments. The work of Wenping Wang was partially supported by the National Basic Research Program of China(2011CB302400), NSFC (61272019) and the Research Grant Council of HongKong (718209, 718010, 718311, 717012).

References

ALLIEZ, P., COHEN-STEINER, D., YVINEC, M., AND DESBRUN, M. 2005. Variational tetrahedral meshing. *ACM Trans. Graph. (SIGGRAPH) 24*, 3, 617–625.

AURENHAMMER, F. 1987. Power diagrams: properties, algorithms and applications. *SIAM J. Comput. 16*, 1, 78–96.

AURENHAMMER, F. 1987. A criterion for the affine equivalence of cell complexes in R^d and convex polyhedra in R^{d+1}. *Discrete Comp. Geom. 2*, 49–64.

BLOCK, P., AND LACHAUER, L. 2011. Closest-Fit, Compression-only solutions for freeform shells. In *Proc. of the IABSE-IASS Symp.*

BLOCK, P., AND OCHSENDORF, J. 2007. Thrust Network Analysis: a new methodology for three-dimensional equilibrium. *J. Int. Assoc. Shell and Spatial Structures 48*, 167–173.

BLOCK, P. 2009. *Thrust Network Analysis: Exploring Three-dimensional Equilibrium*. PhD thesis, Massachusetts Institute of Technology.

CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

CHEN, L., AND XU, J. 2004. Optimal Delaunay triangulations. *J. Comput. Math. 22*, 299–308.

CHENG, S.-W., AND DEY, T. K. 2004. Quality meshing with weighted Delaunay refinement. *SIAM J. Comput. 33*, 1, 69–93.

CREMONA, L. 1890. *Graphical Statics*. Oxford University Press.

DE GOES, F., BREEDEN, K., OSTROMOUKHOV, V., AND DESBRUN, M. 2012. Blue noise through optimal transport. *ACM Trans. Graph. (SIGGRAPH ASIA) 31*, 6, 171:1–171:11.

DU, Q., FABER, V., AND GUNZBURGER, M. 1999. Centroidal Voronoi tessellations: applications and algorithms. *SIAM Rev. 41*, 637–676.

FRATERNALI, F., ANGELILLO, M., AND FORTUNATO, A. 2002. A lumped stress method for plane elastic problems and the discrete-continuum approximation. *J. Int. Solids and Structures 39*, 6211–6240.

FRATERNALI, F. 2010. A thrust network approach to the equilibrium problem of unreinforced masonry vaults via polyhedral stress functions. *Mechanics Research Communications 37*, 198–204.

GERTZ, E. M., AND WRIGHT, S. J. 2003. Object-oriented software for quadratic programming. *ACM Trans. Math. Softw. 29*, 58–81.

GLICKENSTEIN, D. 2005. Geometric triangulations and discrete Laplacians on manifolds. `arXiv:0508188 [math.MG]`.

HEYMAN, J. 1966. The stone skeleton. *J. Int. Solids and Structures 2*, 249–279.

HEYMAN, J. 1997. *The Stone Skeleton: Structural Engineering of Masonry Architecture*. Cambridge University Press.

KILIAN, A., AND OCHSENDORF, J. 2005. Paticle-spring systems for structural form finding. *J. Int. Assoc. Shell and Spatial Structures 46*, 77–84.

KILIAN, A. 2006. *Design Exploration through Bidirectional Modeling of Constraints*. PhD thesis, Massachusetts Institute of Technology.

LAWSON, C. 1977. Software for C^1 surface interpolation. In *Mathematical Software III*, Academic Press, New York, 161–194.

LIU, Y., WANG, W., LÉVY, B., SUN, F., YAN, D.-M., LU, L., AND YANG, C. 2009. On centroidal Voronoi tessellation — energy smoothness and fast computation. *ACM Trans. Graph. 28*, 101:1–101:17.

MAXWELL, J. C. 1869. On reciprocal diagrams, frames and diagrams of forces. *Transactions of the Royal Society of Edinburgh 26*, 1–40.

MULLEN, P., MEMARI, P., DE GOES, F., AND DESBRUN, M. 2011. HOT: Hodge-optimized triangulations. *ACM Trans. Graph. (SIGGRAPH) 30*, 4, 103:1–103:12.

ODWYER, D. 1999. Funicular analysis of masonry vaults. *Computers & Structures 73*, 187–197.

RIPPMANN, M., LACHAUER, L., AND BLOCK, P. 2012. Interactive vault design. *J. Int. Space Structures 27*, 219–230.

SCHEK, H.-J. 1974. The force density method for form finding and computation of general networks. *Comput. Methods in Applied Mech. and Eng. 3*, 115–134.

SCHIFTNER, A., AND BALZER, J. 2010. Statics-sensitive layout of planar quadrilateral meshes. In *Advances in Architectural Geometry*, 221–236.

SMITH, J., HODGINS, J., OPPENHEIM, I., AND WITKIN, A. 2002. Creating models of truss structures with optimization. *ACM Trans. Graph. (SIGGRAPH) 21*, 3, 295–301.

UMETANI, N., IGARASHI, T., AND MITRA, N. J. 2012. Guided exploration of physically valid shapes for furniture design. *ACM Trans. Graph. (SIGGRAPH) 31*, 4, 86:1–86:11.

VOUGA, E., HÖBINGER, M., WALLNER, J., AND POTTMANN, H. 2012. Design of self-supporting surfaces. *ACM Trans. Graph. (SIGGRAPH) 31*, 4, 87:1–87:11.

WARDETZKY, M., MATHUR, S., KÄLBERER, F., AND GRINSPUN, E. 2007. Discrete Laplace operators: no free lunch. In *Symp. on Geom. Proc.*

WHITING, E., OCHSENDORF, J., AND DURAND, F. 2009. Procedural modeling of structurally-sound masonry buildings. *ACM Trans. Graph. (SIGGRAPH ASIA) 28*, 112:1–112:9.

WHITING, E., SHIN, H., WANG, R., OCHSENDORF, J., AND DURAND, F. 2012. Structural optimization of 3D masonry buildings. *ACM Trans. Graph. (SIGGRAPH ASIA) 31*, 159:1–159:11.

ZORIN, D., SCHRÖDER, P., AND SWELDENS, W. 1996. Interpolating subdivision for meshes with arbitrary topology. In *SIGGRAPH '96*, 189–192.