

Design of Self-supporting Surfaces

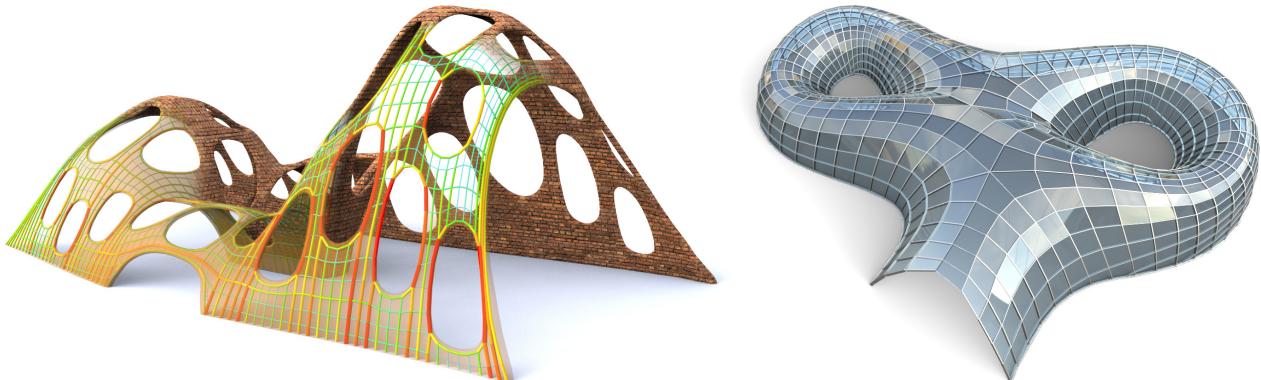


Figure 1: Left: *Surfaces with irregularly placed holes are hard to realize as masonry, where the mortar between bricks must not be subject to tensile stresses. The surface shown here has this surprising property – it has been found as the nearest self-supporting shape from a given freeform geometry. The fictitious thrust network used in our algorithms is also shown, with edges' cross-section and coloring visualizing the magnitude of forces (warmer colors represent higher stresses.)* Right: *Curvature analysis with respect to the so-called Airy stress surface tells us how to remesh shapes by self-supporting quad meshes with planar faces, which guide steel/glass constructions with low moments in nodes.*

1 Abstract

Self-supporting masonry is one of the most ancient and elegant techniques for building curved shapes. Because of the very geometric nature of their failure, analyzing and modeling such structures is more a geometry processing problem than one of classical continuum mechanics. This paper uses the thrust network method of analysis and presents an iterative nonlinear optimization algorithm for efficiently approximating freeform shapes by self-supporting ones. The rich geometry of thrust networks leads us to close connections between different topics of discrete differential geometry, such as a finite-element discretization of the Airy stress potential, perfect graph Laplacians, and computing admissible loads via curvatures of polyhedral surfaces. This geometric viewpoint allows us, in particular, to remesh self-supporting shapes by self-supporting quad meshes with planar faces. This leads to another application of the theory: steel/glass constructions with low moments in nodes.

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

Keywords: Discrete differential geometry, architectural geometry, self-supporting masonry, thrust networks, reciprocal force diagrams, discrete Laplacians, isotropic geometry, mean curvature

1 Introduction

Vaulted masonry structures are among the simplest and at the same time most elegant solutions for creating curved shapes in building construction. For this reason they have been an object of interest since antiquity; large, non-convex examples of such structures include gothic cathedrals. They continue to be an active topic of research today.

Our paper is concerned with a combined geometry+statics analysis of *self-supporting* masonry and with tools for the interactive modeling of freeform self-supporting structures. Here “self-supporting”

means that the structure, considered as an arrangement of blocks (bricks, stones), holds together by itself, with additional support present only during construction. This analysis is based on the following assumptions, which follow the classic [Heyman 1966]:

Assumption 1: Masonry has no tensile strength, but the individual building blocks do not slip against each other (because of friction or mortar). On the other hand, their compressive strength is sufficiently high so that failure of the structure is by a sudden change in geometry and not by material failure.

Assumption 2 (The Safe Theorem): If a system of forces can be found which is in equilibrium with the load on the structure and which is contained within the masonry envelope then the structure will carry the loads, although the actual forces present may not be those postulated by that system.

Our approach is twofold: We first give an overview of the continuous case of a smooth surface under stress, which turns out to be governed locally by the so-called Airy stress function. This mathematical model is called a membrane in the engineering literature and has been applied to the analysis of masonry before. The surface is self-supporting if and only if stresses are entirely compressive. For computational purposes, stresses are discretized as a fictitious *thrust network* [Block and Ochsendorf 2007] contained in the masonry structure; this network is a system of forces in equilibrium with the structure’s deadload. It can be interpreted as a finite element discretization of the continuous case, and it turns out to have very interesting geometry, with the Airy stress function becoming a polyhedral surface directly related to a reciprocal force diagram.

While previous work in architectural geometry was mostly concerned with aspects of rationalization and purely geometric side-conditions which occur in freeform architecture, the focus of this paper is design with *statics* constraints. In particular, our contributions are the following:

Contributions. • We present an optimization algorithm, based on the theory of thrust networks and Airy potentials, for efficiently finding a self-supporting surface near a given arbitrary reference surface (§3), and build a tool for interactive design of self-support-

ing surfaces based on this algorithm (§4). Freeform masonry is based on such surfaces.

- The discrete “stress Laplacian” derived from a thrust network with compressive forces is a so-called perfect one (§2.2). We use it to argue why our discretizations are faithful to the continuous case.
- We connect the physics of self-supporting surfaces with the geometry of isotropic 3-space, and express the equations governing self-supporting surfaces in terms of curvatures (§2.3) and (§2.4). Likewise we establish a connection between the stress Laplacian and mean curvatures of polyhedral surfaces. This theoretical part of the paper is a contribution to Discrete Differential Geometry.
- We use the geometric knowledge we have gathered to find particularly nice families of self-supporting surfaces, especially planar quadrilateral representations of thrust networks (§5). This leads to steel/glass structures with low bending and torsion moments.

Related Work. Unsupported masonry has been an active topic of research in the engineering community. The foundations for the modern approach were laid by Jacques Heyman [1966] and are available as the textbook [Heyman 1995]. The theory of reciprocal force diagrams in the planar case was studied by Maxwell [Maxwell 1864]; a unifying view on polyhedral surfaces, compressive forces and corresponding “convex” force diagrams is presented by [Ash et al. 1988]. F. Fraternali [2002], [2010] established a connection between the continuous theory of stresses in membranes and the discrete theory of forces in thrust networks, by interpreting the latter as a non-conforming finite element discretization of the former.

Several authors have studied the problem of finding discrete compressive force networks contained within the boundary of masonry structures; previous work in this area includes [O’Dwyer 1998] and [Andreu et al. 2007]. Fraternali [2010] proposed solving for the structure’s discrete stress surface, and examining its convex hull to study the structure’s stability and susceptibility to cracking. Philippe Block’s seminal thesis introduced *Thrust Network Analysis*, which pioneered the use of thrust networks and their reciprocal diagrams for efficient and practical design of self-supporting masonry structures. By first seeking a reciprocal diagram of the top view, guaranteeing equilibrium of horizontal forces, then solving for the heights that balance the vertical loads, Thrust Network Analysis linearizes the form-finding problem. For a thorough overview of this methodology, see e.g. [Block and Ochsendorf 2007; Block 2009]. Recent work by Block and coauthors extends this method in the case where the reciprocal diagram is not unique; for different choices of reciprocal diagram, the optimal heights can be found using the method of least squares [Van Mele and Block 2011], and the search for the best such reciprocal diagram can be automated using a genetic algorithm [Block and Lachauer 2011].

Other approaches to the design of self-supporting structures include modeling these structures as damped particle-spring systems [Kilian and Ochsendorf 2005; Barnes 2009], and mirroring the rich tradition in architecture of designing self-supporting surfaces using hanging chain models [Heyman 1998]. Alternatively, masonry structures can be represented by networks of rigid blocks [Livesley 1992], whose conditions on the structural feasibility were incorporated into procedural modeling of buildings [Whiting et al. 2009].

Algorithmic and mathematical methods relevant to this paper are work on the geometry of PQ meshes [Glymph et al. 2004; Liu et al. 2006], discrete curvatures for such meshes [Pottmann et al. 2007; Bobenko et al. 2010], in particular curvatures in isotropic geometry [Pottmann and Liu 2007]. Schiftner and Balzer [2010] discuss approximating a reference surface by a quad mesh with planar faces, whose layout is guided by statics properties of that surface.

2 Self-supporting Surfaces

This section is the theoretical basis of the paper. §2.1 and §2.2 explain the mathematical model for unsupported masonry and its discretization, which is needed in our modeling algorithms. The connection with isotropic geometry (§2.3 and §2.4) is important for the later Section 5, which deals with self-supporting PQ meshes and moment-free steel/glass constructions.

2.1 The Continuous Theory

We model masonry as a surface given by a height field $s(x, y)$ defined in some planar domain Ω . We assume that there are vertical loads $F(x, y)$ — usually F represents the structure’s own weight. By definition this surface is self-supporting if and only if there exists a field of compressive stresses which are in equilibrium with the acting forces. This is equivalent to existence of a field $M(x, y)$ of 2×2 symmetric positive semidefinite matrices satisfying

$$\operatorname{div}(M\nabla s) = F, \quad \operatorname{div} M = 0, \quad (1)$$

where the divergence operator $\operatorname{div} \begin{pmatrix} u(x,y) \\ v(x,y) \end{pmatrix} = u_x + v_y$ is understood to act on the columns of a matrix (see e.g. [Fraternali 2010], [Giaquinta and Giusti 1985]).

The condition $\operatorname{div} M = 0$ says that M is locally the Hessian of a real-valued function ϕ (the *Airy stress potential*): With the notation

$$M = \begin{pmatrix} m_{11} & m_{12} \\ m_{12} & m_{22} \end{pmatrix} \iff \widehat{M} = \begin{pmatrix} m_{22} & -m_{12} \\ -m_{12} & m_{11} \end{pmatrix}$$

it is clear that $\operatorname{div} M = 0$ is an integrability condition for \widehat{M} , so locally there is a potential ϕ with

$$\widehat{M} = \nabla^2 \phi, \quad \text{i.e.,} \quad M = \widehat{\nabla^2 \phi}.$$

If the domain Ω is simply connected, this relation holds globally. Positive semidefiniteness of M (or equivalently of \widehat{M}) characterizes *convexity* of the Airy potential ϕ . The Airy function enters computations only by way of its derivatives, so global existence is not an issue.

Remark: Stresses at boundary points depend on the way the surface is anchored: A fixed anchor means no condition, but a free boundary with outer normal vector \mathbf{n} means $\langle M\nabla s, \mathbf{n} \rangle = 0$.

Stress Laplacian. Note that $\operatorname{div} M = 0$ yields $\operatorname{div}(M\nabla s) = \operatorname{tr}(M\nabla^2 s)$, which we like to call $\Delta_\phi s$. The operator Δ_ϕ is symmetric. It is elliptic (as a Laplace operator should be) if and only if M is positive definite, i.e., ϕ is strictly convex. The balance condition (1) may be written as $\Delta_\phi s = F$.

2.2 Discrete Theory: Thrust Networks

We discretize a self-supporting surface by a mesh $\mathcal{S} = (V, E, F)$ (see Figure 2). Loads are again vertical, and we discretize them as force densities F_i associated with vertices \mathbf{v}_i . The load acting on this vertex is then given by $F_i A_i$, where A_i is an area of influence (using a prime to indicate projection onto the xy plane, A_i is the area of the Voronoi cell of \mathbf{v}'_i w.r.t. V'). We assume that stresses are carried by the edges of the mesh: the force exerted on the vertex \mathbf{v}_i by the edge connecting $\mathbf{v}_i, \mathbf{v}_j$ is given by

$$w_{ij}(\mathbf{v}_j - \mathbf{v}_i), \quad \text{where} \quad w_{ij} = w_{ji} \geq 0.$$

The nonnegativity of the individual weights w_{ij} expresses the compressive nature of forces. The balance conditions at vertices then

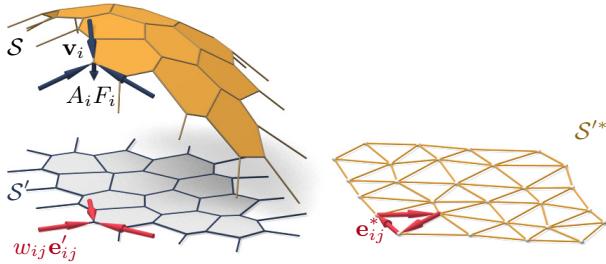


Figure 2: A thrust network \mathcal{S} with dangling edges indicating external forces (left). This network together with compressive forces which balance vertical loads $A_i F_i$ projects onto a planar mesh \mathcal{S}' with equilibrium compressive forces $w_{ij} \mathbf{e}'_{ij}$ in its edges. Rotating forces by 90° leads to the reciprocal force diagram \mathcal{S}'^* (right).

176 read as follows: With $\mathbf{v}_i = (x_i, y_i, s_i)$ we have

$$\sum_{j \sim i} w_{ij}(x_j - x_i) = \sum_{j \sim i} w_{ij}(y_j - y_i) = 0, \quad (2)$$

$$\sum_{j \sim i} w_{ij}(s_j - s_i) = A_i F_i. \quad (3)$$

177 A mesh equipped with edge weights in this way is a discrete *thrust* 178 network. Invoking the safe theorem, we can state that a masonry 179 structure is self-supporting, if we can find a thrust network with 180 compressive forces which is entirely contained within the structure.

181 **Reciprocal Diagram.** Equations (2) have a geometric interpretation: 182 with edge vectors

$$\mathbf{e}'_{ij} = \mathbf{v}'_j - \mathbf{v}'_i = (x_j, y_j) - (x_i, y_i),$$

183 Equation (2) asserts that vectors $w_{ij} \mathbf{e}'_{ij}$ form a closed cycle. Rotating 184 them by 90 degrees, we see that likewise

$$\mathbf{e}'^*_{ij} = w_{ij} J \mathbf{e}'_{ij}, \quad \text{with } J = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix},$$

185 form a closed cycle (see Figure 2). If the mesh \mathcal{S} is simply connected, 186 there exists an entire *reciprocal diagram* \mathcal{S}'^* which is a 187 combinatorial dual of \mathcal{S} , and which has edge vectors \mathbf{e}'^*_{ij} . Its 188 vertices are denoted by \mathbf{v}'^*_i .

189 *Remark:* If \mathcal{S}' is a Delaunay triangulation, then the corresponding 190 Voronoi diagram is an example of a reciprocal diagram.

191 **Polyhedral Stress Potential.** We can go further and construct a 192 convex polyhedral “Airy stress potential” surface Φ with vertices 193 $\mathbf{w}_i = (x_i, y_i, \phi_i)$ combinatorially equivalent to \mathcal{S} by requiring that 194 a primal face of Φ lies in the plane $z = \alpha x + \beta y + \gamma$ if and only if 195 (α, β) is the corresponding dual vertex of \mathcal{S}'^* (see Figure 3). Ob- 196 viously this condition determines Φ up to vertical translation. For 197 existence see [Ash et al. 1988]. The inverse procedure constructs 198 a reciprocal diagram from Φ . This procedure works also if forces 199 are not compressive: we can construct an Airy mesh Φ which has 200 planar faces, but it will no longer be a convex polyhedron.

201 The vertices of Φ can be interpolated by a piecewise-linear function 202 $\phi(x, y)$. It is easy to see that the derivative of $\phi(x, y)$ jumps by the 203 amount $\|\mathbf{e}'^*_{ij}\| = w_{ij} \|\mathbf{e}'_{ij}\|$ when crossing over the edge \mathbf{e}'_{ij} at right 204 angle, with unit speed. This identifies Φ as the Airy polyhedron in- 205 troduced by [Fraternali et al. 2002] as a finite element discretization 206 of the continuous Airy function (see also [Fraternali 2010]).

207 If the mesh is not simply connected, the reciprocal diagram and 208 the Airy polyhedron exist only locally. Our computations do not 209 require global existence.

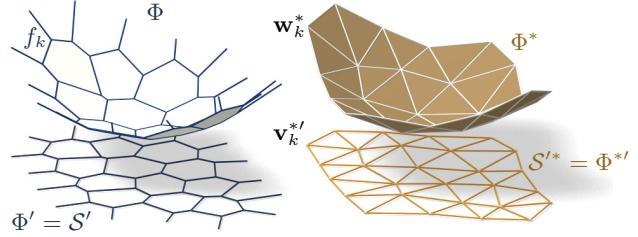


Figure 3: Airy stress potential Φ and its polar dual Φ^* . Φ projects onto the same planar mesh as \mathcal{S} does, while Φ^* projects onto the reciprocal force diagram. A primal face f_k lies in the plane $z = \alpha x + \beta y + \gamma \iff$ the corresponding dual vertex is $\mathbf{w}_k^* = (\alpha, \beta, -\gamma)$.

210 **Polarity.** Polarity with respect to the *Maxwell paraboloid* $z =$
211 $\frac{1}{2}(x^2 + y^2)$ maps the plane $z = \alpha x + \beta y + \gamma$ to the point $(\alpha, \beta, -\gamma)$.
212 Thus, applying polarity to Φ and projecting the result Φ^* into the xy
213 plane reconstructs the reciprocal diagram $\Phi'^* = \mathcal{S}'^*$ (see Fig. 3).

214 **Discrete Stress Laplacian.** The weights w_{ij} may be used to de-
215 fine a graph Laplacian Δ_ϕ which on vertex-based functions acts as

$$\Delta_\phi s(\mathbf{v}_i) = \sum_{j \sim i} w_{ij}(s_j - s_i).$$

216 This operator is a perfect discrete Laplacian in the sense of [War-
217 detzky et al. 2007], since it is symmetric by construction, Equa-
218 tion (2) implies linear precision for the planar “top view mesh” \mathcal{S}'
219 (i.e., $\Delta_\phi f = 0$ if f is a linear function), and $w_{ij} \geq 0$ ensures
220 semidefiniteness and a maximum principle for Δ_ϕ -harmonic func-
221 tions. Equation (3) can be written as $\Delta_\phi s = AF$.

222 Note that Δ_ϕ is well defined even when the underlying meshes are
223 not simply connected.

2.3 Surfaces in Isotropic Geometry

224 It is worthwhile to reconsider the basics of self-supporting surfaces
225 in the language of dual-isotropic geometry, which takes place in \mathbb{R}^3
226 with the z axis as a distinguished vertical direction. The basic ele-
227 ments of this geometry are planes, having equation $z = f(x, y) =$
228 $\alpha x + \beta y + \gamma$. The gradient vector $\nabla f = (\alpha, \beta)$ determines the
229 plane up to translation. A plane tangent to the graph of the function
230 $s(x, y)$ has gradient vector ∇s .

232 There is the notion of *parallel points*: $(x, y, z) \parallel (x', y', z') \iff$
233 $x = x', y = y'$.

234 *Remark:* The Maxwell paraboloid is considered the unit sphere of
235 isotropic geometry, and the geometric quantities considered above
236 are assigned specific meanings: The forces $\|\mathbf{e}'^*_{ij}\| = w_{ij} \|\mathbf{e}'_{ij}\|$ are
237 dihedral angles of the Airy polyhedron Φ , and also “lengths” of
238 edges of Φ^* . We do not use this terminology in the sequel.

239 **Curvatures.** Generally speaking, in the differential geometry of
240 surfaces one considers the *Gauss map* σ from a surface S to a con-
241 vex unit sphere Φ by requiring that corresponding points have par-
242 allel tangent planes. Subsequently mean curvature H^{rel} and Gaus-
243 sian curvature K^{rel} relative to Φ are computed from the derivative
244 $d\sigma$. Classically Φ is the ordinary unit sphere $x^2 + y^2 + z^2 = 1$, so
245 that σ maps each point to its unit normal vector.

246 In our setting, parallelity is a property of *points* rather than planes,
247 and the Gauss map σ goes the other way, mapping the tangent

planes of the unit sphere $z = \phi(x, y)$ to the corresponding tangent plane of the surface $z = s(x, y)$. If we know which point a plane is attached to, then the Gauss map is determined by the plane's gradient. So we simply write

$$\nabla\phi \xrightarrow{\sigma} \nabla s.$$

By moving along a curve $\mathbf{u}(t) = (x(t), y(t))$ in the parameter domain we get the first variation of tangent planes: $\frac{d}{dt}\nabla\phi|_{\mathbf{u}(t)} = (\nabla^2\phi)\dot{\mathbf{u}}$. This yields the derivative $(\nabla^2\phi)\dot{\mathbf{u}} \xrightarrow{d\sigma} (\nabla^2s)\dot{\mathbf{u}}$, for all $\dot{\mathbf{u}}$, and the matrix of $d\sigma$ is found as $(\nabla^2\phi)^{-1}(\nabla^2s)$. By definition, curvatures of the surface s relative to ϕ are found as

$$K_s^{\text{rel}} = \det(d\sigma) = \frac{\det\nabla^2s}{\det\nabla^2\phi},$$

$$H_s^{\text{rel}} = \frac{1}{2}\text{tr}(d\sigma) = \frac{1}{2}\text{tr}\left(\frac{M}{\det\nabla^2\phi}\nabla^2s\right) = \frac{\Delta_\phi s}{2\det\nabla^2\phi}.$$

The Maxwell paraboloid $\phi_0(x, y) = \frac{1}{2}(x^2 + y^2)$ is the canonical unit sphere of isotropic geometry, with Hessian E_2 . Curvatures relative to ϕ_0 are not called "relative" and are denoted by the symbols H, K instead of $H^{\text{rel}}, K^{\text{rel}}$. The observation

$$\Delta_\phi\phi = \text{tr}(M\nabla^2\phi) = \text{tr}(\widehat{\nabla^2\phi}\nabla^2\phi) = 2\det\nabla^2\phi$$

together with the formulas above implies

$$K_s = \det\nabla^2s, K_\phi = \det\nabla^2\phi \implies H_s^{\text{rel}} = \frac{\Delta_\phi s}{2K_\phi} = \frac{\Delta_\phi s}{\Delta_\phi\phi}.$$

Relation to Self-supporting Surfaces. Summarizing the formulas above, we rewrite the balance condition (1) as

$$2K_\phi H_s^{\text{rel}} = \Delta_\phi s = F. \quad (4)$$

Let us draw some conclusions:

- Since $H_\phi^{\text{rel}} = 1$ we see that the load $F_\phi = 2K_\phi$ is admissible for the stress surface $\phi(x, y)$, which is hereby shown as self-supporting. The quotient of loads yields $H_s^{\text{rel}} = F/F_\phi$.
- If the stress surface coincides with the Maxwell paraboloid, then *constant loads characterize constant mean curvature surfaces*, because we get $K_\phi = 1$ and $H_s = F/2$.
- If s_1, s_2 have the same stress potential ϕ , then $H_{s_1-s_2}^{\text{rel}} = H_{s_1}^{\text{rel}} - H_{s_2}^{\text{rel}} = 0$, so $s_1 - s_2$ is a (relative) minimal surface.

2.4 Meshes in Isotropic Geometry

A general theory of curvatures of polyhedral surfaces with respect to a polyhedral unit sphere was proposed by [Pottmann et al. 2007; Bobenko et al. 2010], and its dual complement in isotropic geometry was elaborated on in [Pottmann and Liu 2007]. As illustrated by Figure 4, the mean curvature of a self-supporting surface \mathcal{S} relative to its discrete Airy stress potential is associated with the vertices of \mathcal{S} . It is computed from areas and mixed areas of faces in the polar polyhedra \mathcal{S}^* and Φ^* :

$$H^{\text{rel}}(\mathbf{v}_i) = \frac{A_i(\mathcal{S}, \Phi)}{A_i(\Phi, \Phi)}, \quad \text{where}$$

$$A_i(\mathcal{S}, \Phi) = \frac{1}{4} \sum_{k: f_k \in 1\text{-ring}(\mathbf{v}_i)} \det(\mathbf{v}'_k, \mathbf{w}'_{k+1}) + \det(\mathbf{w}'_k, \mathbf{v}'_{k+1}).$$

The prime denotes the projection into the xy plane, and summation is over those dual vertices which are adjacent to \mathbf{v}_i . Replacing \mathbf{v}'_k by \mathbf{w}'_k yields $A_i(\Phi, \Phi) = \frac{1}{2} \sum \det(\mathbf{w}'_k, \mathbf{w}'_{k+1})$.

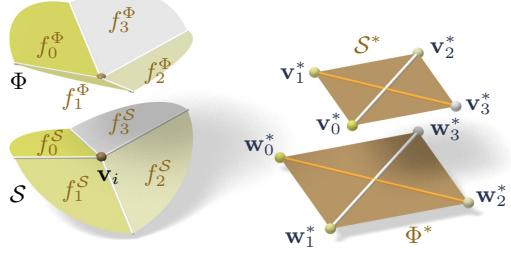


Figure 4: Mean curvature of a vertex \mathbf{v}_i of \mathcal{S} : Corresponding edges of the polar duals \mathcal{S}^* , Φ^* are parallel, and mean curvature according to [Pottmann et al. 2007] is computed from the vertices polar to faces adjacent to \mathbf{v}_i . For valence 4 vertices the case of zero mean curvature shown here is characterized by parallelity of non-corresponding diagonals of corresponding quads in \mathcal{S}^* , Φ^* .

Proposition. If Φ is the Airy surface of a thrust network \mathcal{S} , then the mean curvature of \mathcal{S} relative to Φ is computable as

$$H^{\text{rel}}(\mathbf{v}_i) = \frac{\sum_{j \sim i} w_{ij}(s_j - s_i)}{\sum_{j \sim i} w_{ij}(\phi_j - \phi_i)} = \frac{\Delta_\phi s}{\Delta_\phi\phi}|_{\mathbf{v}_i}. \quad (5)$$

Proof. It is sufficient to show $2A_i(\mathcal{S}, \Phi) = \sum_{j \sim i} w_{ij}(s_j - s_i)$.

For that, consider edges $\mathbf{e}'_1, \dots, \mathbf{e}'_n$ emanating from \mathbf{v}'_i . The dual cycles in Φ^* and \mathcal{S}^* without loss of generality are given by vertices $(\mathbf{v}'_1, \dots, \mathbf{v}'_n)$ and $(\mathbf{w}'_1, \dots, \mathbf{w}'_n)$, respectively. The latter has edges $\mathbf{w}'_{j+1} - \mathbf{w}'_j = w_{ij}J\mathbf{e}'_j$ (indices modulo n).

Without loss of generality $\mathbf{v}_i = 0$, so the vertex \mathbf{v}'_j by construction equals the gradient of the linear function $\mathbf{x} \mapsto \langle \mathbf{v}'_j, \mathbf{x} \rangle$ defined by the properties $\mathbf{e}'_{j-1} \mapsto s_{j-1} - s_i$, $\mathbf{e}'_j \mapsto s_j - s_i$. Corresponding edge vectors $\mathbf{v}'_{j+1} - \mathbf{v}'_j$ and $\mathbf{w}'_{j+1} - \mathbf{w}'_j$ are parallel, because $\langle \mathbf{v}'_{j+1} - \mathbf{v}'_j, \mathbf{e}'_j \rangle = (s_j - s_i) - (s_j - s_i) = 0$. Expand $2A_i(\mathcal{S}, \Phi)$:

$$\begin{aligned} & \frac{1}{2} \sum \det(\mathbf{w}'_j, \mathbf{v}'_{j+1}) + \det(\mathbf{v}'_j, \mathbf{w}'_{j+1}) \\ &= \frac{1}{2} \sum \det(\mathbf{w}'_j - \mathbf{w}'_{j+1}, \mathbf{v}'_{j+1}) + \det(\mathbf{v}'_j, \mathbf{w}'_{j+1} - \mathbf{w}'_j) \\ &= \frac{1}{2} \sum \det(-w_{ij}J\mathbf{e}'_j, \mathbf{v}'_{j+1}) + \det(\mathbf{v}'_j, w_{ij}J\mathbf{e}'_j) \\ &= \sum \det(\mathbf{v}'_j, w_{ij}J\mathbf{e}'_j) = \sum w_{ij} \langle \mathbf{v}'_j, \mathbf{e}'_j \rangle = \sum w_{ij}(s_j - s_i). \end{aligned}$$

Here we have used $\det(\mathbf{a}, J\mathbf{b}) = \langle \mathbf{a}, \mathbf{b} \rangle$. \square

In order to discretize (4), we also need a discrete Gaussian curvature, usually defined as a quotient of areas which correspond under the Gauss mapping. We define

$$K_\Phi(\mathbf{v}_i) = \frac{A_i(\Phi, \Phi)}{A_i},$$

where A_i is the Voronoi area of vertex \mathbf{v}'_i in the projected mesh \mathcal{S}' used in (3).

Remark: If the faces of the thrust network \mathcal{S} are not planar, the simple trick of introducing additional edges with zero forces in them makes them planar, and the theory is applicable. In the interest of space, we refrain from elaborating further.

Discrete Balance Equation. The discrete version of the balance equation (4) reads as follows:

Theorem. A simply-connected mesh \mathcal{S} with vertices $\mathbf{v}_i = (x_i, y_i, s_i)$ can be put into static equilibrium with vertical nodal forces $A_i F_i$ if and only if there exists a combinatorially equivalent

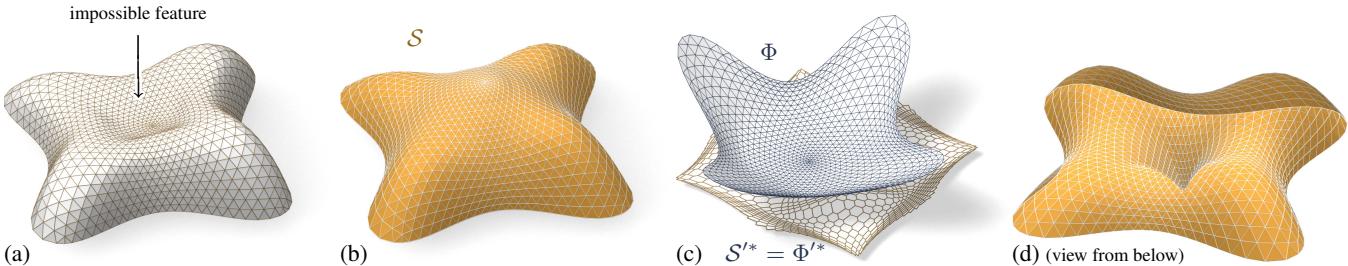


Figure 5: The top of the Lilium Tower (a) cannot stand as a masonry structure, because its central part is concave. Our algorithm finds a nearby self-supporting mesh (b) without this impossible feature. (c) shows the corresponding Airy mesh Φ and reciprocal force diagram S'^* . (d) The user can edit the original surface, such as by specifying that the center of the surface is supported by a vertical pillar, and the self-supporting network adjusts accordingly.

mesh Φ with planar faces and vertices (x_i, y_i, ϕ_i) , such that curvatures of \mathcal{S} relative to Φ obey

$$2K_\Phi(\mathbf{v}_i)H^{\text{rel}}(\mathbf{v}_i) = F_i \quad (6)$$

at every interior vertex and every free boundary vertex \mathbf{v}_i . \mathcal{S} can be put into compressive static equilibrium if and only if there exists a convex such Φ .

Proof. The relation between equilibrium forces $w_{ij}\mathbf{e}_{ij}$ in \mathcal{S} and the polyhedral stress potential Φ has been discussed above, and so has the equivalence “ $w_{ij} \geq 0 \iff \Phi$ convex” (see e.g. [Ash et al. 1988] for a survey of this and related results). It remains to show that Equations (2) and (6) are equivalent. This is the case because the proposition above implies $2K(\mathbf{v}_i)H^{\text{rel}}(\mathbf{v}_i) = 2\frac{A_i(\Phi, \Phi)}{A_i} \frac{A_i(\Phi, \mathcal{S})}{A_i(\Phi, \Phi)} = \frac{1}{A_i}(\sum_{j \sim i} w_{ij}(s_j - s_i)) = \frac{1}{A_i} A_i F_i$. \square

2.5 Convergence.

When considering discrete thrust networks as discretizations of continuous self-supporting surfaces, the following question is important: For a given smooth surface $s(x, y)$ with stress potential ϕ , does there exist a polyhedral surface \mathcal{S} in equilibrium approximating $s(x, y)$, whose top view is a given planar mesh \mathcal{S}' ? We restrict our attention to triangle meshes, where planarity of the faces of the discrete stress surface Φ is not an issue. Equivalently, we ask:

- Does \mathcal{S}' have a reciprocal diagram whose corresponding Airy polyhedron Φ approximates the continuous Airy potential ϕ ? (if the surfaces involved are not simply connected, these objects are defined locally).
- Does \mathcal{S}' possess a “perfect” discrete Laplace-Beltrami operator Δ_ϕ in the sense of Wardetzky et al. [2007] whose weights are the edge length scalars of such a reciprocal diagram?

From [Wardetzky et al. 2007] we know that perfect Laplacians exist only on regular triangulations which are projections of convex polyhedra. On the other hand, previous sections show how to appropriately re-triangulate: Let Φ be a triangle mesh convex hull of the vertices $(x_i, y_i, \phi(x_i, y_i))$, where (x_i, y_i) are vertices of \mathcal{S}' . Then its polar dual Φ^* projects onto a reciprocal diagram with positive edge weights, so Δ_ϕ has positive weights, and the vertices (x_i, y_i, s_i) of \mathcal{S} can be found by solving the discrete Poisson problem $(\Delta_\phi s)_i = A_i F_i$.

We expect, but we don’t prove, that the discrete Δ_ϕ approximates its continuous counterpart for reasonable sampling (after all it is directly derived from $\phi(x, y)$). This implies that solving the discrete Poisson equation leads to a mesh approximating its continuous counterpart $s(x, y)$, and we have convergence as the sampling density increases. A rigorous analysis is a topic for future research.

3 Thrust Networks from Reference Meshes

Consider now the problem of taking a given reference mesh, say \mathcal{R} , and finding a combinatorially equivalent mesh \mathcal{S} in static equilibrium approximating \mathcal{R} . The loads on \mathcal{S} include user-prescribed loads as well as the dead load caused by the mesh’s own weight. Conceptually, finding \mathcal{S} amounts to minimizing some formulation of distance between \mathcal{R} and \mathcal{S} , subject to constraints (2), (3), and $w_{ij} \geq 0$. For any choice of distance this minimization will be a nonlinear, non-convex, inequality-constrained variational problem. Our experience with black-box solvers [Wächter and Biegler 2006] is that they perform well for surfaces without complex geometry or for polishing reference meshes close to self-supporting, but fail to converge in reasonable time for more complicated shapes such as the one of Fig. 1, left. We therefore propose the following specialized, staggered linearization for solving the optimization problem:

0. Start with an initial guess $\mathcal{S} = \mathcal{R}$.
1. Estimate the self-load on the vertices of \mathcal{S} , using their current positions.
2. Fixing \mathcal{S} , locally fit an associated stress surface Φ .
3. Alter positions \mathbf{v}_i to improve the fit.
4. Repeat from Step 1 until convergence.

Remark: This staggered approach shares the several advantages of solving the full nonlinear problem: a nearby self-supporting surface is found given only a suggested reference shape, without needing to single one of the many possible top view reciprocal diagrams or needing to specify boundary tractions – these are found automatically during optimization. Although providing an initial top view graph with good combinatorics remains important, by not fixing the top view our approach allows the thrust network to slide both vertically and tangentially to the ground, essential to finding faithful thrust networks for surfaces with free boundary conditions.

Step 1: Estimating Self-Load. The dead load due to the surface’s own weight depends not only on the top view of \mathcal{S} , but also on the surface area of its faces. To avoid adding nonlinearity to the algorithm, we estimate the load coefficients F_i at the beginning of each iteration, and assume they remain constant until the next iteration. We estimate the load $A_i F_i$ associated with each vertex by calculating its Voronoi surface area on each of its incident faces (note that this surface area is distinct from A_i , the vertex’s Voronoi area on the top view), and then multiplying by a user-specified surface density ρ .

Step 2: Fit a Stress Surface. In this step, we fix \mathcal{S} and try to fit a stress surface Φ subordinate to the top view \mathcal{S}' of the primal mesh. We do so by searching for dihedral angles between the faces of Φ which minimize, in the least-squares sense, the error in force



Figure 6: A freeform surface (left) needs adjustments around the entrance arch and between the two pillars in order to be self-supporting; our algorithm finds the nearby surface in equilibrium (right) that incorporates these changes.

Fig.	Vertices	Edges	Time (s)	Iterations	Max. Rel. Error
5b	1201	3504	21.6	9	4.2×10^{-5}
5d	1200	3500	26.5	10	8.5×10^{-5}
6	1535	2976	17.0	21	2.7×10^{-5}
7	752	2165	8.0	9	5.8×10^{-5}
9	2358	4302	19.5	9	3.0×10^{-4}
16	527	998	5.7	25	2.4×10^{-5}

Table 1: Numerical details about our examples. We show the clock time needed by an Intel Xeon 2.3GHz desktop PC with 4 GB of RAM to find a self-supporting thrust network and associated stress surface from the example’s reference mesh; we also give the number of outer iterations of the four steps in (§3). The maximum relative error is the dimensionless quantity $\max_i \|A_i F_i - \sum_{j \sim i} w_{ij}(\mathbf{v}_j - \mathbf{v}_i)\| / \|A_i F_i\|$ (the maximum is taken over interior vertices \mathbf{v}_i).

equilibrium (6) and local integrability of Φ . Doing so is equivalent to minimizing the squared residuals of Equations (3) and (2), with the positions held fixed. We define the *equilibrium energy*

$$E = \sum_i \left\| \begin{pmatrix} 0 \\ 0 \\ A_i F_i \end{pmatrix} - \sum_{j \sim i} w_{ij}(\mathbf{v}_j - \mathbf{v}_i) \right\|^2, \quad (7)$$

where i runs through interior and free boundary vertices, and solve

$$\min_{w_{ij}} E, \quad \text{s.t. } 0 \leq w_{ij} \leq w_{\max}. \quad (8)$$

Here w_{\max} is an optional maximum weight we are willing to assign (to limit the amount of stress in the surface). This convex, sparse, box-constrained least-squares problem [Friedlander 2007] always has a solution. If the objective is 0 at this solution, the faces of Φ locally integrate to a stress surface satisfying (6), and this Φ certifies that \mathcal{S} is self-supporting – we are done. Otherwise, \mathcal{S} is not self-supporting and its vertices must be moved.

Step 3: Alter Positions. In the previous step we fit as best as possible a stress surface Φ to \mathcal{S} . There are two possible kinds of error with this fit: the faces around a vertex (equivalently, the reciprocal diagram) might not close up; and the resulting stress forces might not be exactly in equilibrium with the loads. These errors can be decreased by modifying the top view and heights of \mathcal{S} , respectively. It is possible to simply solve for new vertex positions that put \mathcal{S} in static equilibrium, since Equations (2) and (3) with w_{ij} fixed form a square linear system that is typically nonsingular.

While this approach would yield a self-supporting \mathcal{S} , this mesh is often far from the reference mesh \mathcal{R} , since any local errors in the stress surface from Step 2 amplify into global errors in \mathcal{S} . We propose instead to look for new positions that decrease the imbalance in the stresses and loads, while also penalizing drift away from the reference mesh:

$$\min_{\mathbf{v}} E + \alpha \sum_i \langle \mathbf{n}_i, \mathbf{v}_i - \mathbf{v}_i^0 \rangle^2 + \beta \|\mathbf{v} - \mathbf{v}_P^0\|^2,$$

where \mathbf{v}_i^0 is the position of the i -th vertex at the start of this step of the optimization, \mathbf{n}_i is the starting vertex normal (computed as the average of the incident face normals), \mathbf{v}_P^0 is the projection of \mathbf{v}^0 onto the reference mesh, and $\alpha > \beta$ are penalty coefficients that are decreased every iteration of Steps 1–3. The second term allows \mathcal{S} to slide over itself (if doing so improves equilibrium) but penalizes drift in the normal direction. The third term, weaker than the second, regularizes the optimization by preventing large drift away from the reference surface or excessive tangential sliding.

Implementation Details. Solving the weighted least-squares problem of Step 3 amounts to solving a sparse, symmetric linear system. While the MINRES algorithm [Paige and Saunders 1975] is likely the most robust algorithm for solving this system, in practice we have observed that the method of conjugate gradients works well despite the potential ill-conditioning of the objective matrix.

Limitations. This algorithm is not guaranteed to always converge; this fact is not surprising from the physics of the problem (if the boundary of the reference mesh encloses too large of a region, w_{\max} is set too low, and the density of the surface too high, a thrust network in equilibrium simply does not exist – the vault is too ambitious and cannot be built to stand; pillars are needed.)

We can, however, make a few remarks. Only Step 1 can increase the equilibrium energy E of Equation (7). Step 2 always decreases it, and Step 3 does as well as $\beta \rightarrow 0$. Moreover, as $\alpha \rightarrow 0$ and $\beta \rightarrow 0$, Step 3 approaches a linear system with as many equations as unknowns; if this system has full rank, its solution sets $E = 0$. These facts suggest that the algorithm should generally converge to a thrust network in equilibrium, provided that Step 1 does not increase the loads by too much at every iteration, and this is indeed what we observe in practice. One case where this assumption is guaranteed to hold is if the thickness of the surface is allowed to freely vary, so that it can be chosen so that the surface has uniform density over the top view.

If the linear system in Step 3 is singular and infeasible, the algorithm can stall at $E > 0$. This failure occurs, for instance, when an interior vertex has height z_i lower than all of its neighbors, and Step 2 assigns all incident edges to that vertex a weight of zero: clearly no amount of moving the vertex or its neighbors can bring the vertex into equilibrium. We avoid such degenerate configurations by bounding weights slightly away from zero in (8), trading increased robustness for slight smoothing of the resulting surface. Attempting to optimize meshes that have self-intersecting top views (i.e., aren’t height fields), have too many impossible features, or are insufficiently supported by fixed boundary points can also result in errors and instability.

4 Results

Interactive Design of Self-Supporting Surfaces. The optimization algorithm described in the previous section forms the basis of an interactive design tool for self-supporting surfaces. Users manipulate a mesh representing a reference surface, and the computer searches for a nearby thrust network in equilibrium (see e.g. Figure 8). Features of the design tool include:

- Handle-based 3D editing of the reference mesh using Laplacian coordinates [Lipman et al. 2004; Sorkine et al. 2003] to extrude vaults, insert pillars, and apply other deformations to the reference mesh. Handle-based adjustments of the heights, keeping the top view fixed, and deformation of the top view, keeping the heights fixed, are also supported. The thrust network adjusts interactively to fit the deformed positions, giving

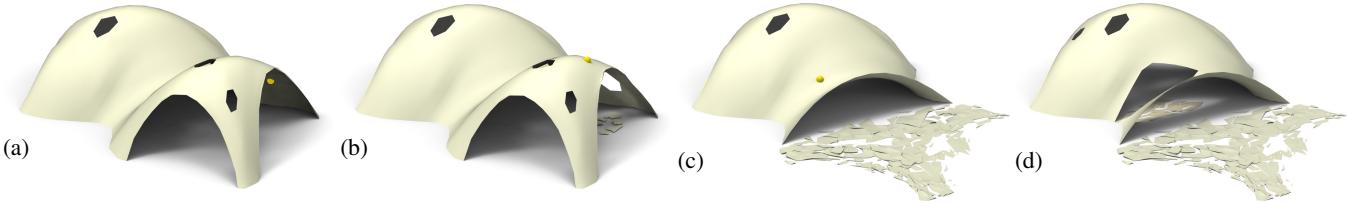


Figure 7: Destruction sequence. We simulate removing small parts of masonry (their location is shown by a yellow ball) and the falling off of further pieces which are no longer supported after removal. For this example, removing a certain small number of single bricks does not affect stability (a,b). Removal of material at a certain point (yellow ball in (b)) will cause a greater part of the structure to collapse, as seen in (c). (d) shows the result after one more removal (all images show the respective thrust networks, not the reference surface).

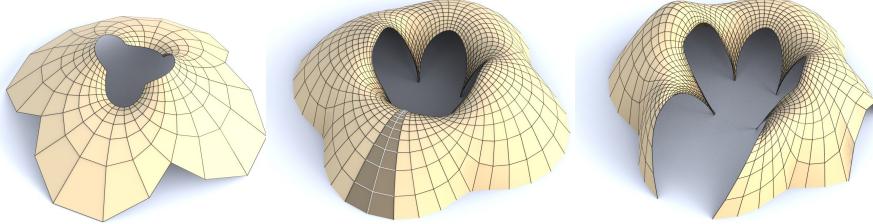


Figure 8: Interactive Edits. (a) shows a self-supporting thrust network – in fact it is a Koebe mesh as mentioned in §5. In (b) pillars have been added. Cutting along the highlighted edge and optimizing for the self-supporting property results in the mesh shown in (c).

the usual visual feedback about the effects of edits on whether or not the surface can stand.

- Specification of boundary conditions. Points of contact between the reference surface and the ground or environment are specified by “pinning” vertices of the surface, specifying that the thrust network must coincide with the reference mesh at this point, and relaxing the condition that forces must be in equilibrium there.
- Interactive adjustment of surface density ρ , external loads, and maximum permissible stress per edge w_{\max} , with visual feedback of how these parameters affect the fitted thrust network.
- Upsampling of the thrust network through Catmull-Clark subdivision and polishing of the resulting refined thrust network using optimization (§3).
- Visualization of the stress surface dual to the thrust network and corresponding reciprocal diagram.

Examples. *Top of the Lilium Tower.* Consider the top portion of the steel-glass exterior surface of the Lilium Tower, which is currently being built in Warszaw (see Figure 5). What is if we had wanted to build this surface out of masonry instead? This surface contains a concave part with local minimum in its interior and so cannot possibly be self-supporting without modification. Given this surface as a reference mesh, our algorithm constructs a nearby thrust network in equilibrium without the impossible feature. The user can then explore how editing the reference mesh – adding a pillar, for example – affects the thrust network and its deviation from the reference surface.

Example: Freeform Structure with Two Pillars. Suppose an architect’s experience and intuition has permitted the design of a nearly self-supporting freeform surface (Figure 6). Our algorithm reveals those edits needed to make the structure sound – principally around the entrance arch, and the area between the two pillars.

Interactive Editing: Figure 8 shows an example of the design and optimization workflow. Starting with a mesh, we first add pillars in the center and clean up the outer boundary (by fixing it to the floor). A subsequent cut needs a further round of optimization. This surface is neither convex nor simply connected, and exhibits a mix of boundary conditions, none of which cause our algorithm any difficulty; it always finds a self-supporting thrust network near the

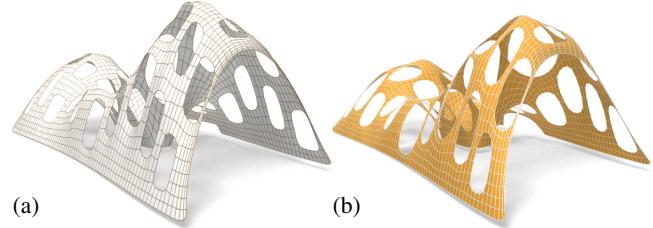


Figure 9: A mesh with holes (a) requires large deformations to both the top view and heights to render it self-supporting (b)

designed reference mesh. The user is free to make edits to the reference mesh, and the thrust network adapts to these edits, providing the user feedback on whether these designs are physically realizable [we generally refer to the accompanying video for interactive building and editing of freeform self-supporting shapes].

Example: Destruction Sequence. In Figure 7 we simulate removing parts of masonry and the falling off of further pieces which are no longer supported after removal. This is done by deleting the 1-neighborhood of a vertex and solving for a new thrust network in compressive equilibrium close to the original reference surface. We delete those parts of the network which deviate too much and are no longer contained in the masonry hull, and iterate.

Example: Swiss Cheese. Cutting holes in a self-supporting surface interrupts force flow lines and causes dramatic global changes to the surface stresses, often to the point that the surface is no longer in equilibrium. Whether a given surface with many such holes can stand is far from obvious. Figure 9a shows such an implausible and unstable surface; our optimization finds a nearby, equally implausible but stable surface without difficulty (see Figures 1, left and 9b).

Example: Stability Test: See Figures 10 and 11 for a series of images which visualize the effect of additional loads on a thrust network.

Example: Structural Glass. See Figure 16 for details on a self-supporting surface which is realized not as masonry, but as a steel/glass construction with glass as a structural element.

Figure 10: Stability Test. Left: Coloring and cross-section of edges visualize the magnitude of forces in a thrust network which is in equilibrium with this dome’s dead load. Right: When an additional load is applied, there exists a corresponding compressive thrust network which is still contained in the masonry hull of the original dome. This implies stability of the dome under that load.

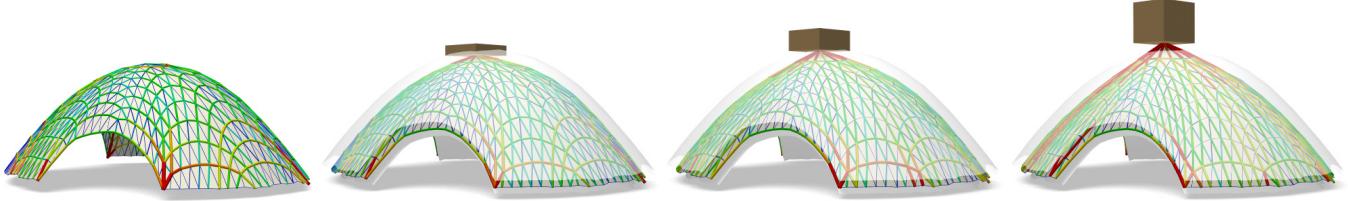


Figure 11: Stability test similar to Figure 10, but with a shell thickness of 1 m, in order to better visualize the way the thrust network starts to leave the masonry hull as the load increases. Additional loads are 0 kg, 5,000 kg, 10,000 kg, and 20,000 kg, resp., from left to right.

5 Special Self-Supporting Surfaces

PQ Meshes. Meshes with *planar* faces are of particular interest in architecture, so in this section we discuss how to remesh a given thrust network in equilibrium such that it becomes a quad mesh with planar faces (again in equilibrium). If this mesh is realized as a steel-glass construction, it is self-supporting in its beams alone, with no forces exerted on the glass (this is the usual manner of using glass). The beams constitute a self-supporting structure which is in perfect force equilibrium (without moments in the nodes) if only the deadload is applied.

Taking an arbitrary non-planar quad mesh and attempting naive, simultaneous enforcement of planarity and static equilibrium – either by staggering a planarity optimization step every outer iteration, or adding a planarity penalty term to the position update – does not yield good results, as shown in Figure 12. Indeed, as we will see later in this section, such a planar perturbation of a thrust network is not expected to generally exist.

Consider a planar quad mesh \mathcal{S} with vertices $\mathbf{v}_{ij} = (x_{ij}, y_{ij}, s_{ij})$ which approximates a given continuous surface $s(x, y)$. It is known that \mathcal{S} must approximately follow a network of conjugate curves in the surface (see e.g. [Liu et al. 2006]). We can derive this condition in an elementary way as follows: Using a Taylor expansion, we compute the volume of the convex hull of the quadrilateral $\mathbf{v}_{ij}, \mathbf{v}_{i+1,j}, \mathbf{v}_{i+1,j+1}, \mathbf{v}_{i,j+1}$, assuming the vertices lie exactly on the surface $s(x, y)$. This results in

$$\text{vol} = \frac{1}{6} \det(\mathbf{a}_1, \mathbf{a}_2) \cdot ((\mathbf{a}_1)^T \nabla^2 s \mathbf{a}_2) + \dots,$$

where $\mathbf{a}_1 = \begin{pmatrix} x_{i+1,j} - x_{ij} \\ y_{i+1,j} - y_{ij} \end{pmatrix}$, $\mathbf{a}_2 = \begin{pmatrix} x_{i,j+1} - x_{ij} \\ y_{i,j+1} - y_{ij} \end{pmatrix}$,

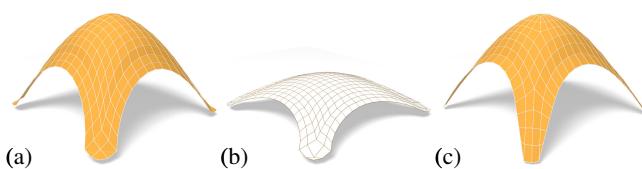


Figure 12: Directly enforcing planarity of the faces of even a very simple self-supporting quad-mesh vault (a) results in a surface far removed from the original design (b). Starting instead from a remeshing of the surface with edges following relative principal curvature directions yields a self-supporting, PQ mesh far more faithful to the original (c).

and the dots indicate higher order terms. We see that planarity requires $(\mathbf{a}_1)^T \nabla^2 s \mathbf{a}_2 = 0$. In addition to the mesh \mathcal{S} approximating the surface $s(x, y)$, the corresponding polyhedral Airy surface Φ must approximate $\phi(x, y)$; thus we get the conditions

$$(\mathbf{a}_1)^T \nabla^2 s \mathbf{a}_2 = (\mathbf{a}_1)^T \nabla^2 \phi \mathbf{a}_2 = 0.$$

$\mathbf{a}_1, \mathbf{a}_2$ are therefore eigenvectors of $(\nabla^2 \phi)^{-1} \nabla^2 s$. In view of §2.3, $\mathbf{a}_1, \mathbf{a}_2$ indicate the principal directions of the surface $s(x, y)$ relative to $\phi(x, y)$.

In the discrete case, where s, ϕ are not given as continuous surfaces, but are represented by a mesh in equilibrium and its Airy mesh, we use the techniques of Schiftner [2007] and Cohen-Steiner and Morvan [2003] to approximate the Hessians $\nabla^2 s, \nabla^2 \phi$, compute principal directions as eigenvectors of $(\nabla^2 \phi)^{-1} \nabla^2 s$, and subsequently find meshes \mathcal{S}, Φ approximating s, ϕ which follow those directions. Global optimization can now polish \mathcal{S}, Φ to a valid thrust network with discrete stress potential, where before it failed: we do so by taking the planarity energy $\sum_f (2\pi - \theta_f)^2$, where the sum runs over faces and θ_f is the sum of the interior angles of face f , linearizing it at every iteration, and adding it to the objective function of the position update (Step 3). Convexity of Φ ensures that \mathcal{S} is self-supporting.

Note that for each Φ , the relative principal curvature directions give the *unique* curve network along which a planar quad discretization of a self-supporting surface is possible. Other networks lead to results like the one shown by Figure 12. Figures 13 and 14 further illustrate the result of applying this procedure to self-supporting surfaces.

Remark: When remeshing a given shape by planar quad meshes, we know that the circular and conical properties require that the mesh follows the ordinary, Euclidean principal curvature directions [Liu et al. 2006]. It is remarkable that the self-supporting property in a similar manner requires us to follow certain *relative* principal directions. Practitioners’ observations regarding the beneficial statics properties of principal directions can be explained by this analogy, because the relative principal directions are close to the Euclidean ones, if the stress distribution is uniform and $\|\nabla s\|$ is small.

Koenigs Meshes. Given a self-supporting thrust network \mathcal{S} with stress surface Φ , we ask the question: Which vertical perturbation $\mathcal{S} + \mathcal{R}$ is self-supporting, with the same loads as \mathcal{S} ? As to notation, all involved meshes $\mathcal{S}, \mathcal{R}, \Phi$ have the same top view, and arithmetic operations refer to the respective z coordinates s_i, r_i, ϕ_i of vertices.

Figure 13: Planar quad remeshing of the surface of Fig. 5. (a) Relative principal directions, found from eigenvectors of $(\nabla^2 \phi)^{-1} \nabla^2 s$. (b) Quad mesh guided by principal directions is almost planar and almost self-supporting. (c) Small changes achieve both properties.

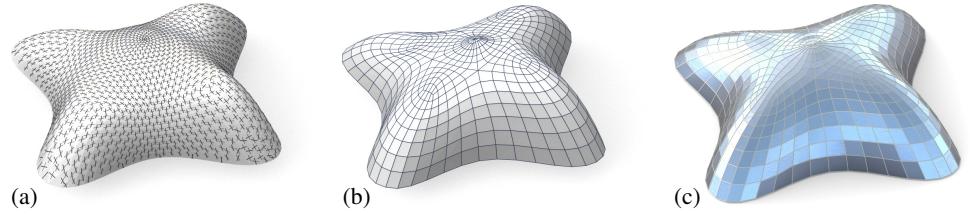


Figure 14: Planar quad remeshing of the surface of Figure 6. Left: Relative principal directions. Right: The result of optimization is a self-supporting PQ mesh, which guides a moment-free steel/glass construction (interior view, see also Fig. 1).



615 The condition of equal loads then is expressed as $\Delta_\phi(s + r) =$
 616 $\Delta_\phi s$ in terms of Laplacians or as $H_S^{\text{rel}} = H_{S+\mathcal{R}}^{\text{rel}}$ in terms of mean
 617 curvature, and is equivalent to

$$\Delta_\phi r = 0, \quad \text{i.e.,} \quad H_{\mathcal{R}}^{\text{rel}} = 0.$$

618 So \mathcal{R} is a minimal surface relative to Φ . While in the triangle mesh
 619 case there are enough degrees of freedom for nontrivial solutions,
 620 the case of planar quad meshes is more intricate: Polar polyhedra
 621 \mathcal{R}^* , Φ^* have to be Christoffel duals of each other [Pottmann and
 622 Liu 2007], as illustrated by Figure 4. Unfortunately not all quad
 623 meshes have such a dual; the condition is that the mesh is *Koenigs*,
 624 i.e., the derived mesh formed by the intersection points of diagonals
 625 of faces again has planar faces [Bobenko and Suris 2008].



Figure 15: A “Koebe” mesh Φ is self-supporting for unit dead load. An entire family of self-supporting meshes with the same top view is defined by $\mathcal{S}_\alpha = \Phi + \alpha \mathcal{R}$, where \mathcal{R} is chosen as Φ ’s Christoffel-dual.

626 **Koebe meshes.** An interesting special case occurs if Φ is a
 627 *Koebe* mesh of isotropic geometry, i.e., a PQ mesh whose edges
 628 touch the Maxwell paraboloid. Since Φ approximates the Maxwell
 629 paraboloid, we get $2K(\mathbf{v}_i)H^{\text{rel}}(\mathbf{v}_i) \approx 1$ and Φ consequently is
 630 self-supporting for unit load. Applying the Christoffel dual con-
 631 struction described above yields a minimal mesh \mathcal{R} and a family of
 632 meshes $\Phi + \alpha \mathcal{R}$ which are self-supporting for unit load (see Fig-
 633 ure 15).

641 with an interactive tool for evaluating the statics of freeform ge-
 642 ometries. The self-supporting property of a shape is directly rele-
 643 vant for freeform masonry. The actual thrust networks we use for
 644 computation are relevant e.g. for steel constructions, where equilib-
 645 rium of deadload forces implies absence of moments. This theory
 646 and accompanying algorithms thus constitute a new contribution to
 647 architectural geometry, connecting statics and geometric design.

648 **Future Work.** There are several directions of future research. One
 649 is to incorporate non-manifold meshes, which occur naturally when
 650 e.g. supporting walls are introduced. It is also obvious that non-ver-
 651 tical loads, e.g. wind load, play a role. There are also some direc-
 652 tions to pursue in improving the algorithms, for instance adaptive
 653 remeshing in problem areas. Probably the interesting connections
 654 between statics and geometry are not yet exhausted, and we would
 655 like to propose the *geometrization* of problems as a general solution
 656 paradigm.

657 **Acknowledgements.** This work was very much inspired by
 658 Philippe Block’s plenary lecture at the 2011 Symposium on Geom-
 659 etry Processing in Lausanne. Several illustrations (the destruction
 660 sequence of Fig. 7 and the maximum load example of Fig. 10) have
 661 real-world analogues on his web page [Block 2011].

622 References

- 663 ANDREU, A., GIL, L., AND ROCA, P. 2007. Computational anal-
 664 ysis of masonry structures with a funicular model. *J. Engrg.
 665 Mechanics* 133, 473–480.
- 666 ASH, P., BOLKER, E., CRAPO, H., AND WHITELEY, W. 1988.
 667 Convex polyhedra, Dirichlet tessellations, and spider webs. In
 668 *Shaping space (Northampton 1984)*. Birkhäuser, 231–250.
- 669 BARNES, M. R. 2009. Form finding and analysis of tension struc-
 670 tures by dynamic relaxation. *Int. J. Space Struct.* 14, 2, 89–104.
- 671 BLOCK, P., AND LACHAUER, L. 2011. Closest-fit, compression-
 672 only solutions for free form shells. In *IABSE — IASS 2011 Lon-
 673 don Symposium*, Int. Ass. Shell Spatial Structures. electronic.
- 674 BLOCK, P., AND OCHSENDORF, J. 2007. Thrust network analysis:
 675 A new methodology for three-dimensional equilibrium. *J. Int.
 676 Assoc. Shell and Spatial Structures* 48, 3, 167–173.

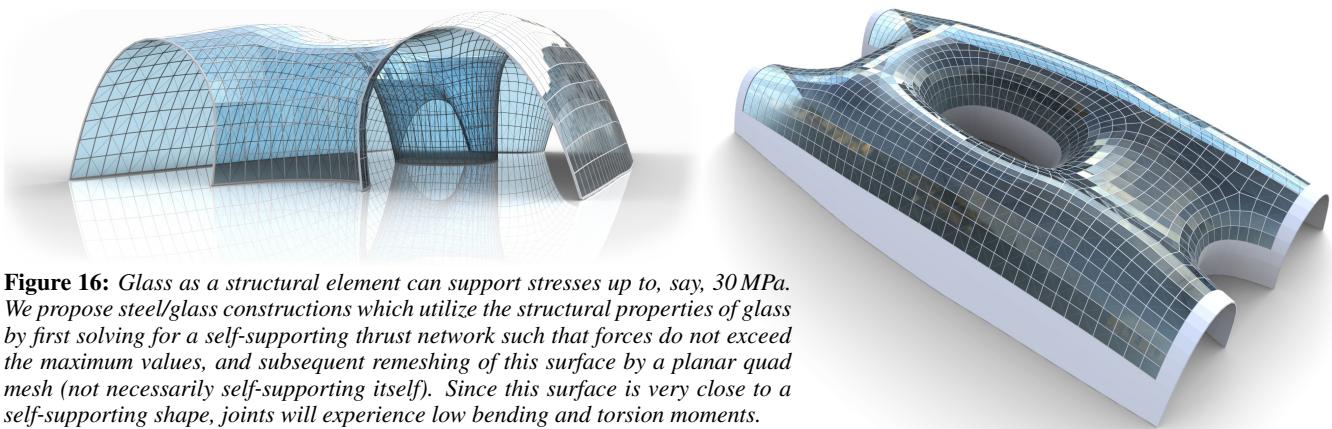


Figure 16: Glass as a structural element can support stresses up to, say, 30 MPa. We propose steel/glass constructions which utilize the structural properties of glass by first solving for a self-supporting thrust network such that forces do not exceed the maximum values, and subsequent remeshing of this surface by a planar quad mesh (not necessarily self-supporting itself). Since this surface is very close to a self-supporting shape, joints will experience low bending and torsion moments.

- 677 BLOCK, P. 2009. *Thrust Network Analysis: Exploring Three-
678 dimensional Equilibrium*. PhD thesis, Massachusetts Institute
679 of Technology.
- 680 BLOCK, P., 2011. Project webpage at [http://block.arch.ethz.ch/
681 projects/freeform-catalan-thin-tile-vaulting](http://block.arch.ethz.ch/projects/freeform-catalan-thin-tile-vaulting).
- 682 BOBENKO, A., AND SURIS, YU. 2008. *Discrete differential geometry: Integrable Structure*. American Math. Soc.
- 683 BOBENKO, A., POTTMANN, H., AND WALLNER, J. 2010. A
684 curvature theory for discrete surfaces based on mesh parallelity.
685 *Math. Annalen* 348, 1–24.
- 686 COHEN-STEINER, D., AND MORVAN, J.-M. 2003. Restricted
687 Delaunay triangulations and normal cycle. In *Proc. 19th Symp.
688 Computational Geometry*, ACM, 312–321.
- 689 FRATERNALI, F., ANGELILLO, M., AND FORTUNATO, A. 2002.
690 A lumped stress method for plane elastic problems and the
691 discrete-continuum approximation. *Int. J. Solids Struct.* 39,
692 6211–6240.
- 693 FRATERNALI, F. 2010. A thrust network approach to the equi-
694 librium problem of unreinforced masonry vaults via polyhedral
695 stress functions. *Mechanics Res. Comm.* 37, 2, 198 – 204.
- 696 FRIEDLANDER, M. P., 2007. BCLS: Bound constrained least
697 squares. <http://www.cs.ubc.ca/~mpf/bcls>.
- 698 GIAQUINTA, M., AND GIUSTI, E. 1985. Researches on the equi-
699 librium of masonry structures. *Archive for Rational Mechanics
700 and Analysis* 88, 4, 359–392.
- 701 GLYMPH, J., SHELDEN, D., CECCATO, C., MUSSLEM, J., AND
702 SCHOBER, H. 2004. A parametric strategy for free-form glass
703 structures using quadrilateral planar facets. *Automation in Con-
704 struction* 13, 2, 187 – 202.
- 705 HEYMAN, J. 1966. The stone skeleton. *Int. J. Solids Structures* 2,
706 249–279.
- 707 HEYMAN, J. 1995. *The Stone Skeleton: Structural Engineering of
708 Masonry Architecture*. Cambridge University Press.
- 709 HEYMAN, J. 1998. *Structural Analysis: A Historical Approach*.
710 Cambridge University Press.
- 711 KILIAN, A., AND OCHSENDORF, J. 2005. Particle-spring sys-
712 tems for structural form finding. *J. Int. Assoc. Shell and Spatial
713 Structures* 46, 77–84.
- 714 LIPMAN, Y., SORKINE, O., COHEN-OR, D., LEVIN, D., ROSSI,
715 C., AND SEIDEL, H. 2004. Differential coordinates for interac-
716 tive mesh editing. In *Proc. SMI*. IEEE, 181–190.
- 717 LIU, Y., POTTMANN, H., WALLNER, J., YANG, Y.-L., AND
718 WANG, W. 2006. Geometric modeling with conical meshes
719 and developable surfaces. *ACM Trans. Graphics* 25, 3, 681–689.
- 720 LIVESLEY, R. K. 1992. A computational model for the limit anal-
721 ysis of three-dimensional masonry structures. *Meccanica* 27,
722 161–172.
- 723 MAXWELL, J. 1864. On reciprocal diagrams and diagrams of
724 forces. *Philosophical Magazine* 4, 27, 250–261.
- 725 O'Dwyer, D. 1998. Funicular analysis of masonry vaults. *Com-
726 puters and Structures* 73, 187–197.
- 727 PAIGE, C. C., AND SAUNDERS, M. A. 1975. Solution of sparse
728 indefinite systems of linear equations. *SIAM J. Num. Analysis*
729 12, 617–629.
- 730 POTTMANN, H., AND LIU, Y. 2007. Discrete surfaces in isotropic
731 geometry. In *Mathematics of Surfaces XII*. Springer, 341–363.
- 732 POTTMANN, H., LIU, Y., WALLNER, J., BOBENKO, A., AND
733 WANG, W. 2007. Geometry of multi-layer freeform structures
734 for architecture. *ACM Trans. Graphics* 26, 3, #65,1–11.
- 735 SCHIFTNER, A., AND BALZER, J. 2010. Statics-sensitive layout
736 of planar quadrilateral meshes. In *Advances in Architectural Ge-
737 ometry 2010*, C. Ceccato et al., Eds. Springer, Vienna, 221–236.
- 738 SCHIFTNER, A. 2007. *Planar quad meshes from relative principal
739 curvature lines*. Master's thesis, TU Wien.
- 740 SORKINE, O., COHEN-OR, D., AND TOLEDO, S. 2003. High-
741 pass quantization for mesh encoding. In *Symposium Geometry
742 processing*. 42–51.
- 743 VAN MELE, T., AND BLOCK, P. 2011. A novel form finding
744 method for fabric formwork for concrete shells. *J. Int. Assoc.
745 Shell and Spatial Structures* 52, 217–224.
- 746 WÄCHTER, A., AND BIEGLER, L. T. 2006. On the implemen-
747 tation of a primal-dual interior point filter line search algorithm for
748 large-scale nonlinear programming. *Math. Progr.* 106, 25–57.
- 749 WARDETZKY, M., MATHUR, S., KÄLBERER, F., AND GRIN-
750 SPUN, E. 2007. Discrete Laplace operators: No free lunch.
751 In *Symposium on Geometry Processing*. 33–37.
- 752 WHITING, E., OCHSENDORF, J., AND DURAND, F. 2009. Proce-
753 dural modeling of structurally-sound masonry buildings. *ACM
754 Trans. Graphics* 28, 5, #112,1–9.
- 755