Patterns for Masonry Vault Design

Robin Oval, Matthias Rippmann, Tom van Mele, Olivier Baverel, Philippe Block

September 25th - 28th, 2017
INTERFACES
architecture . engineering . science IASS 2017 hamburg

Proceedings of the IASS Annual Symposium 2017
"Interfaces: architecture.engineering.science"
25 - 28th September, 2017, Hamburg, Germany
Annette Bögle, Manfred Grohmann (eds.)

Patterns for Masonry Vault Design

Robin OVAL[a,b] *, Matthias RIPPMANN[a], Tom VAN MELE[a], Olivier BAVEREL[b], Philippe BLOCK[a]

* robin.oval@enpc.fr / roval@arch.ethz.ch

[a] ETH Zurich, Institute of Technology in Architecture, Block Research Group
Stefano-Franscini-Platz, HIB E 45, 8093 Zurich, Switzerland

[b] École des Ponts ParisTech, Laboratoire Navier, MSA
6-8, avenue Blaise Pascal, 77455 Champs-sur-Marne, France

Abstract

This paper presents a methodology to generate structural patterns for masonry vault design. First, a quad-dominant block decomposition is proposed based on a medial axis pruning/rebranching method from an input that comprises outer and inner boundaries as well as point and curve features, representing a point load or a crease in the structure, for instance. The meshing and smoothing of the resulting set of patches is straightforward and the mesh densities can be controlled globally and locally. The resulting meshes can be processed for form finding and further optimisation. Second, fabrication-and construction-aware rules to convert these form-found patterns into a tessellation are proposed.

Keywords: structural design, masonry vaults, meshing, topology, block decomposition, medial axis, tessellation.

1. Introduction

The computational form finding of masonry vaults, and more generally funicular shells, is based on methods that usually compute discrete meshes or networks instead of continuous surfaces, like Thrust Network Analysis (TNA) (Block and Ochsendorf [1], Rippmann *et al.* [2] and Rippmann and Block [3]). The work flow of TNA, as well as of other discrete shell form-finding methods, is as follows: 1. defining the boundaries, 2. designing a planar mesh, 3. setting the constraints, and 4. form finding.

The topology of the patterns, which serve as input to the form-finding process, is crucial as the achievable form-found geometries, i.e. the design space, is directly related to the chosen pattern topology. Our aim is to provide a methodology to generate patterns suitable for funicular shell form finding that fit the main features of the design. Moreover, a tessellation eventually has to be generated and it appears natural to derive this from the force pattern used for form finding, since sliding failure between the voussoirs is prevented by aligning the force flow with the interface normals between the voussoirs (Rippmann [4]). Additionally to these force-based constraint, other fabrication and construction constraints come into play, like the absence of concave corners in the voussoirs and the number of different voussoirs coming together at one point.

Generating a proper pattern is more or less straightforward, depending on the topological input of the project (outer and inner boundaries, additional features), on the structural typology and other project criteria. From the experience of the Block Research Group in masonry vault form finding using TNA and RhinoVAULT, both for simple (Figure 1a) and complex topologies (Figure 1b), we derive a general set of topological heuristics to guide our methodology for automatic mesh generation. The generated meshes should:

- be structured meshes, that provide a regular connectivity,
- be quad-dominant meshes, that can be interpreted as principal directions,

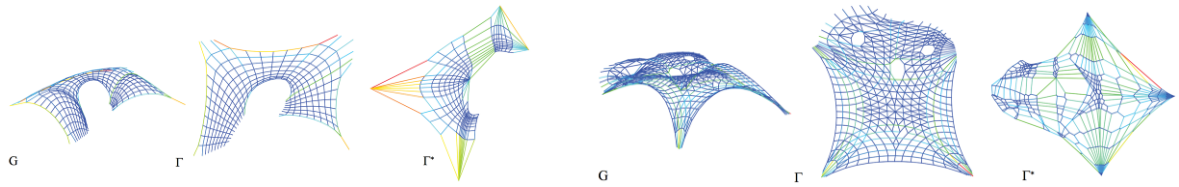
• have a low number of singularities, and

• have no boundary singularities.

Additionally, the methodology should have the ability to include additional features other than the boundaries, such as point and curve features. These are either related to discontinuities in the shell's shape (peaks and creases) or to force concentrations (loads, vertical/horizontal support reactions). The generated meshes should adapt to these features by having:

- a continuous set of edges representing the curve features,
- no singularities on the curve features, to provide feature alignment,
- poles at the extremities of the curve features that do not lie on the boundaries, and
- poles at the point features.

Adding poles with high valency at the point features and at the end of the curve features provides a high degree of hyperstaticity and a large number of load paths. This was initially motivated by requirements for funicular form fitting under load combinations including point loads (Van Mele *et al.* [7]).



(a) Simple topology (b) Complex topology

Figure 1: Two masonry vault projects with different boundary complexity with (G) the form-found vault, (Γ) the form diagram and (Γ*) the force diagram: (a) the free-form Catalan thin-tile vault in Zurich, Switzerland (Davis *et al.* [5]), and (b) MLK Jr. Park Vault, Austin, TX, USA (Rippmann and Block [6]).

2. Background

Extensive research has been done on quad mesh generation. We present the most interesting approaches for our application.

Vector-field integration and optimisation through energy minimisation are standard methods in the fields of computer graphics to generate quad meshes. Some of these researches focused on funicular form fitting of a target surface: Vouga *et al.* [8] use a remeshing process based on conjugate curve networks to design a planar quad mesh and Panozzo *et al.* [9] use one based on heuristic rules to align the mesh with unsupported boundaries, sharp features and the principal curvature directions in anticlastic areas.

However, these methods require an input geometry to compute these vector fields. These are typically not available in an open-ended design approach. Although Panozzo *et al.* [9] use some heuristics that are independent from the form-found geometry, it does not allow control of the mesh topology and its singularities and pushes for full quad patterns that do not allow polar patterns around point features for instance. These approaches relate more to backward processes like optimisation, rationalisation and post-processing than forward processes like mesh generation, exploration and design.

To generate proper meshes, some methods in computational fluid dynamics are based on block decomposition of complex topologies into blocks with simpler topologies. Fogg *et al.* [10] propose a block decomposition based on the shape's medial axis (or topological skeleton). Additional rules are required to further subdivide this block decomposition in order to directly mesh each block. They use the topological information contained in the medial axis as well as the geometrical information to handle concavities and derive rules to add singularities in order to provide smoother meshes. Rigby [11] propose

a simpler method with fewer rules than Fogg *et al.* [10], based on the topological information of the shape's medial axis only, that result in a simpler block de composition with less singularities.

3. Methodology

This methodology has been developed using the CAD software Rhino3D, its API with IronPython and compAS, the Block Research Group's framework and its library on halfedge mesh data structure.

The meshing methodology is split into two main steps. The first step is related to the topology of the mesh: the surface represented by the boundaries is split into a set of topologically simpler patches. The second step is related to the geometry of the mesh: each patch is subdivided and meshed with a simple pattern and the generated mesh is smoothed using a relaxation technique.

We present our methodology on a simple topological example throughout this paper: a quad with a disc-shaped opening. The patterns generated for this topology have a low density for the sake of readability and understandability. As a result, high geometrical distorsions can occur. However, the case study presents a topologically more complex example with a higher density.

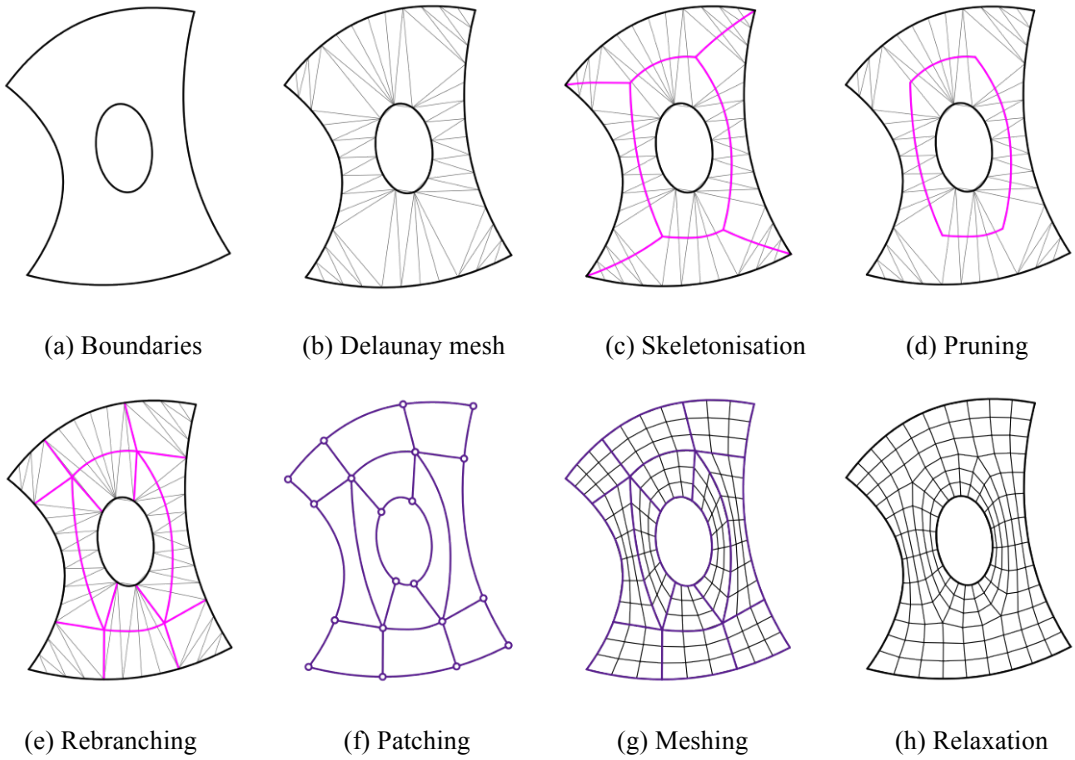
3.1. Topology

The first step is based on the generation of the topological skeleton from which branches are removed (pruning) and to which others are added (rebranching) to obtain a block decomposition. Saha *et al.* [13] provide a survey of the different strategies to generate a topological skeleton. We use a method based on a Delaunay triangulation. Both the generated topological skeleton and the underlying Delaunay triangulation are used to obtain our decomposition into a set of patches.

The input boundary curves shown in Figure 2a are discretised into a set of points. The discretisation spacing is set as a percentage of the length of the diagonal of the bounding box, with values between 1% to 5% usually providing good results. The spacing should be chosen considering the geometrical features that have to be captured: the denser the set of points approximating the curves, the more sensitive the generation of the topological skeleton is to small local changes of curvature. Based on this set of points, a Delaunay triangulation is generated from which the mesh faces lying outside the outer boundaries or inside the inner boundaries are removed to obtain the mesh shown in Figure 2b. In this mesh, we call regular faces those that have two adjacent faces, singular faces those with three adjacent faces and corner faces those with only one adjacent face.

The discrete topological skeleton is directly generated by connecting the circumcircle centres of adjacent faces. We make an exception for the corner faces that are not connected by the centre of their circumcircle but by their corner vertex. The discrete skeleton is split into branches at the singularities, i.e. the centre of the circumcircles of the singular faces. The resulting discrete topological skeleton is not smooth, especially for low discretisation spacings but each branch can be partially interpolated by a spline to give a smooth skeleton, as shown in Figure 2c.

The skeleton has now to be modified to give a proper set of mainly four-sided patches, similarly to Rigby [11]. We derived two heuristic rules based on our experiences to generate proper block decompositions. Firstly, the skeleton branches connected to corner faces are pruned, as shown in Figure 2d. Secondly, new branches are added by connecting each singularity to each vertex of the dual singular face, as shown in Figure 2e. These two steps combined permit to generate new local patches at the boundary corners and to split long N-sided patches into shorter four-sided patches. As a result, the set of patches in Figure 2f defines the topology of the future mesh, with mesh singularities corresponding to the ones of the topological skeleton.



(a) Boundaries (b) Delaunay mesh (c) Skeletonisation (d) Pruning

(e) Rebranching (f) Patching (g) Meshing (h) Relaxation

Figure 2: Medial-axis-based generation of a smooth quad mesh without boundary singularities.

3.2. Geometry

The consecutive geometrical step takes the previous set of patches as input and generates a mesh that is then smoothed. First, a pattern has to be applied for each patch. The topological complexity is controlled during the patching step with the generation of the set of patches. Therefore, we consider only the two elementary patterns shown in Figure 3: a quad without pole and a triangle with a pole, respectively for four- and three-sided patches. The triangle with a pole can be seen as a quad without pole, where an edge has been collapsed to a point that becomes the pole. In each three-sided patch, the pole is set to the unique naked vertex (on the boundaries) or the unique non-naked vertex (on the skeleton branches) among the three corners vertices of the patch.

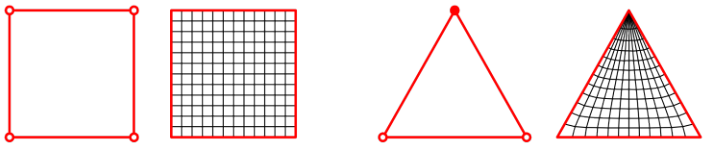


Figure 3: Quad and triangle meshing.

Based on the set of patches and their respective patterns, the edges sharing the same number of subdivisions are grouped. The number of edge groups represents the exact number of degrees of freedom available on the mesh density, for a given set of patches and patterns. The number of subdivisions of each group can be chosen by the designer or set based on a target length and the average or maximum length of the edges in the group, for instance. An additional constraint on the minimum number of subdivisions can also be taken into account. Now that a proper number of subdivisions has been assigned to each edge of each patch, these patches can be separately meshed, using discrete Coons patches. In the case of a four-sided patch, the four corners and edges are bilinearly interpolated to generate a proper quad mesh filling the patch. In the case of a three-sided pattern with a pole, discrete Coons patches can be applied by

inserting a zero-length edge at the pole. For more information see Coons [14]. The resulting set of meshes is welded to obtain a mesh that complies with our requirements: a fully structured quad mesh with a small number of singularities and no boundary singularities, as shown in Figure 2g.

However, the mesh is only piecewise smooth, since the position of the vertices in a patch was computed independently from the ones in the adjacent patches. Moreover, the mesh density can appear irregular, depending on the choices made by the designer when setting the number of subdivisions per edge group. For instance, a high density can be chosen in areas that will require particular attention further in the design process. Post-processing is required to smoothen the generated mesh, as shown in Figure 2h. For instance, a uniform Laplacian smoothing can be applied, which iteratively moves each vertex towards the barycentre of its neighbouring vertices (for more information see Botsch *et al.* [12]). Constraints are added to respect the boundaries: vertices at the boundary corners remain fixed and other vertices lying on the boundaries are projected back on it after each iteration, to allow them to slide along it. Usually, a small number of iterations is sufficient to obtain a globally smooth mesh.

4. Additional features

4.1. Points features

Although boundary point and inner point features have different impacts on the meshing methodology, they are treated in the same manner. Point features are added to the set of points used to generate the initial Delaunay triangulation and are labeled as 'feature point' to differentiate them from regular points.

Boundary point features do not modify the Delaunay triangulation nor the topological skeleton shown in Figure 5a. However, boundary point features are used to edit the set of patches and change the topology. If a four-sided patch has a corner labeled as 'feature point', then the diagonal corresponding to this corner is added, to split the four-sided patch into two three-sided patches, as shown schematically in Figure 4 and on the example in Figure 5b. If a three-sided patch has a corner labeled as 'feature point', this corner becomes the pattern pole, as shown in Figure 5c. The resulting smooth mesh finally features high valency vertices, as shown in Figure 5d. Such features could represent the concentrated horizontal thrust at the corners of the shell.

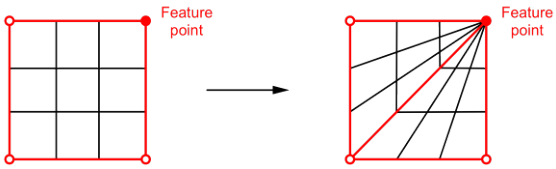
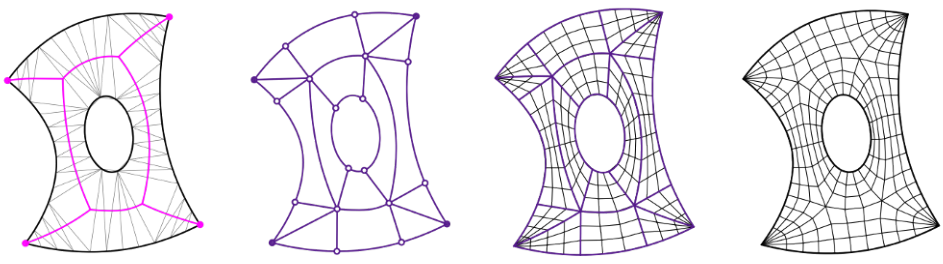


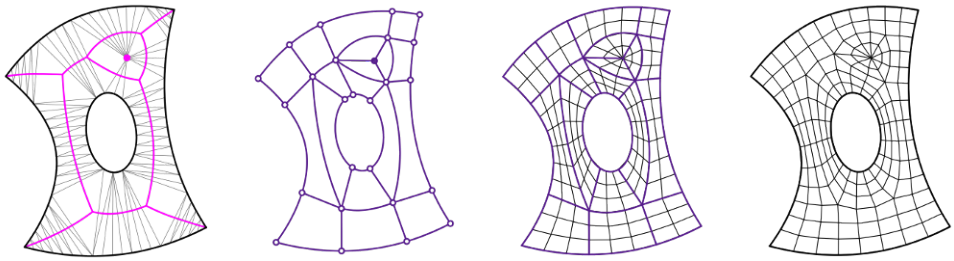
Figure 4: Transformation rule for four-sided patches with point feature.



(a) Skeletonisation (b) Patching (c) Meshing (d) Relaxation

Figure 5: Meshing with boundary point features

On the contrary, inner point features modify the Delaunay triangulation, as shown in Figure 6a, and the resulting set of patches, mainly by adding three-sided patches, as depicted in Figure 6b. The corners labeled 'point feature' of the three-sided patches become the pattern pole, as shown in Figure 6c. During the smoothing process, mesh vertices labeled 'point feature' remain fixed, as shown in Figure 6d. Such features could represent a point load or a nodal support like a column.



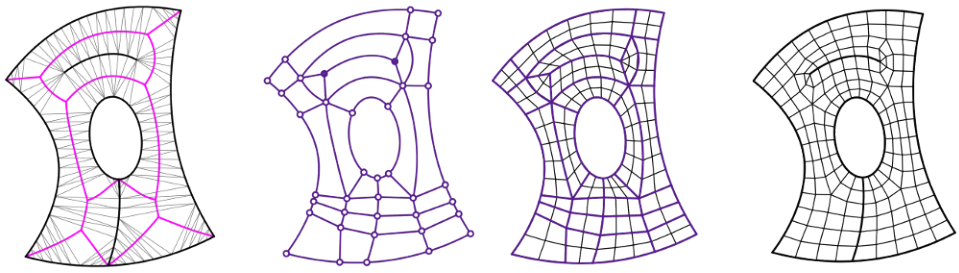
(a) Skeletonisation (b) Patching (c) Meshing (d) Relaxation

Figure 6: Meshing with inner point features

4.2. Curve features

The curve features are discretised and the resulting points added to the set of points used for the Delaunay triangulation, as for the boundaries. The extremities of the curve features that do not lie on the boundaries are labeled 'curve feature extremity' and the other points of the curve features are labeled 'curve feature point'.

An important transformation to the Delaunay triangulation is required to generated a proper topological skeleton: two adjacent faces of the Delaunay triangulation that have their shared edge lying on the curve feature are not considered as adjacent, i.e. a topological cut is made along the curve features. Otherwise, undesired branches would result from this adjacency by connecting the two faces and crossing the curve feature. Therefore, the topological skeleton in Figure 7a captures the curve features properly and does not cross them. As a consequence, the topological cut has to be repaired after the rebranching/pruning of the skeleton, by spreading the interrupted branches at the T-junctions on the curve features, to obtain the set of patches in Figure 7b. As for the point features, the corners labeled 'curve feature extremity' of the three-sided patches become the pattern pole. However, four-sided patches generally treat them as regular vertices, as shown in Figure 7c. During the smoothing process, point and curve constraints are added, as shown in Figure 7d. Such features could represent a crease in the shell's geometry or a line load.



(a) Skeletonisation (b) Patching (c) Meshing (d) Relaxation

Figure 7: Meshing with curve features.

Sometimes, there are no three-sided patches adjacent to the ends of curve features to yield poles. To do so, edges are added in order to split these four-sided patches into a quad and a triangle and yield the

proper connectivity shown schematically in Figure 8. These new edges connect the curve feature extremity with the midpoint of the edge opposite to the curve feature edge.

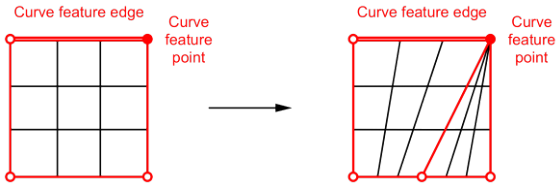
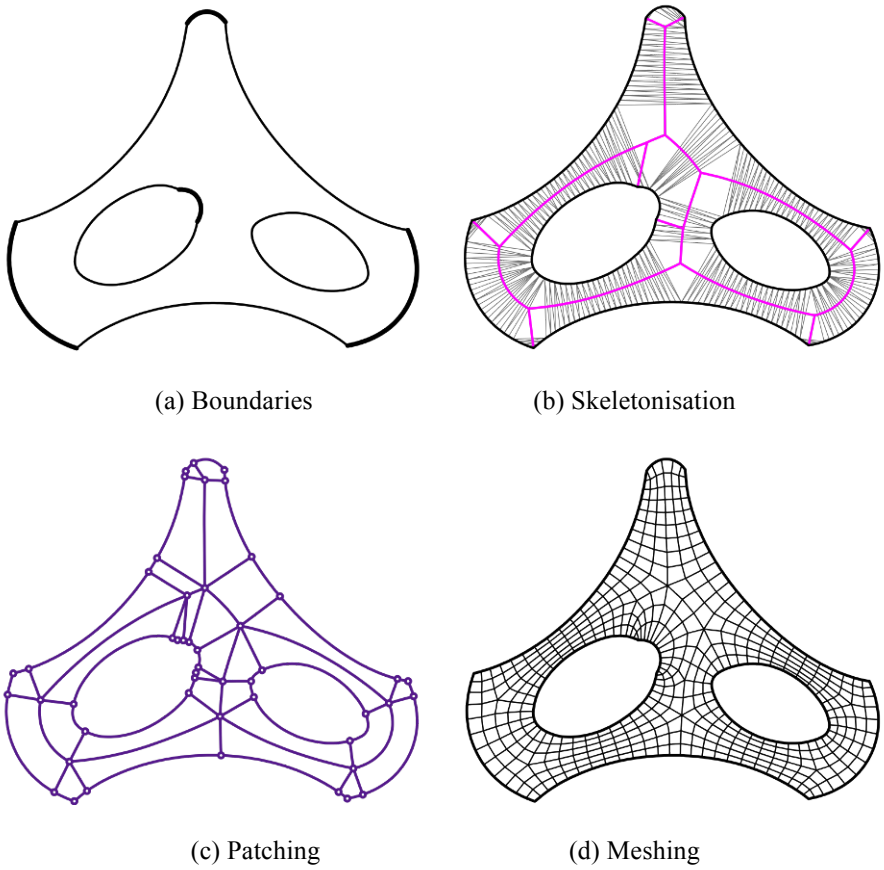


Figure 8: Transformation rule for four-sided patches with curve feature points.

5. Case study: the Armadillo Vault

This section focuses on the application of this work to the design of a masonry vault. We will show how to generate a form diagram for the funicular form-finding tool RhinoVAULT using TNA and how to convert the form-found geometry into a structure, i.e. a tessellation from the form-found thrust network. The Armadillo Vault has been designed and built for the 2016 Venice Biennale by the Block Research Group, the engineering office Ochsendorf DeJong & Block, and the construction company Escobedo Group (Rippmann *et al.* [15]). It is a unreinforced, dry-set, cut-stone vault comprised of 399 voussoirs. The shape of the vault is the result of an iterative form-finding process that evenutally used Best-Fit TNA to fit a target surface.



(a) Boundaries (b) Skeletonisation

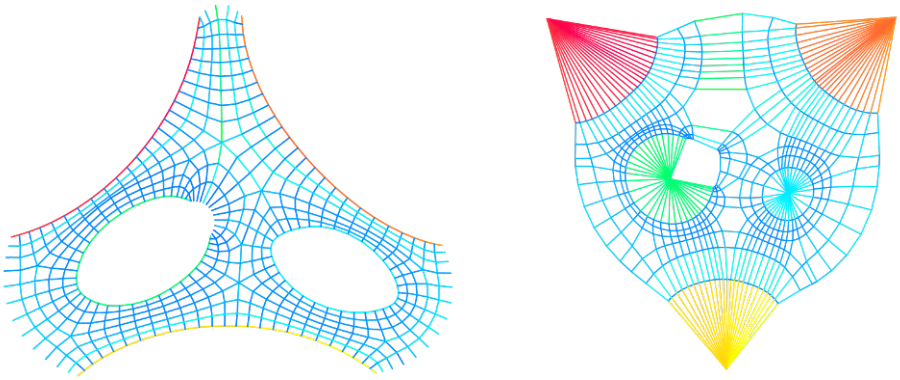
(c) Patching (d) Meshing

Figure 9: Block decomposition and meshing of the Armadillo Vault.

As presented previously, we start with the outer and inner boundaries in Figure 9a as input (the supported edges are represented as thick lines), the topological skeleton can be drawn, as shown in Figure 9b, from

which we derive the block decomposition in Figure 9c. After meshing and smoothing, we obtain the mesh shown in Figure 9d.

To convert this mesh into a proper form diagram that can serve as input for RhinoVAULT, we simply remove the mesh edges that are fully supported, i.e. edges with both end points bearing thrust. Then the form diagram (Figure 10a) is computed with RhinoVAULT to obtain the reciprocal force diagram (Figure 10b), and to achieve horizontal equilibrium, as the first step of the form-finding algorithm.



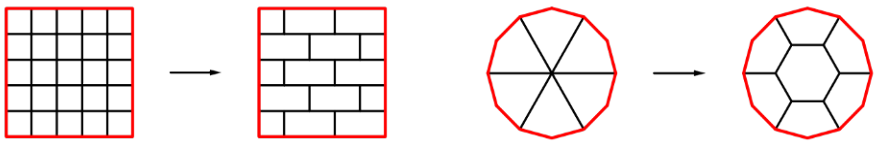
(a) Form diagram (b) Force diagram

Figure 10: Reciprocal form and force diagrams of the Armadillo Vault.

As a second step of the form-finding algorithm, the vertical equilibrium is computed to generate the final funicular geometry (Figure 13a). To tessellate the masonry vault, we take into account two constraints.

The first constraint is related to forces. The interface normals between the voussoirs and the force flow have to be aligned to prevent sliding failure (Heyman [16]). This suggests to base the tessellation on the same mesh or network used to compute the static equilibrium in a form-finding process. Sliding failure is particularly a matter along the unsupported arches and openings in the vault. A staggered pattern along the unsupported boundaries can be achieved by removing every second edge in a preferential direction of the mesh, as suggested in Panozzo *et al.* [9] and shown schematically in Figure 11a. The closest boundary serves as this locally preferential direction. However, we do not remove edges when this would result in strong voussoir concavities. Indeed, L-shaped voussoirs would be more difficult to position correctly and the sharp corners would induce sress concentrations and possibly local crushing, even though the general stress ratio is low.

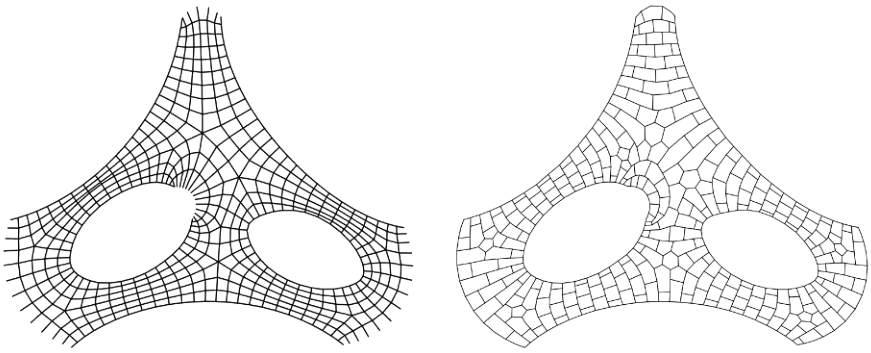
The second constraint is related to construction. Having many voussoirs coming to the same point makes the assembly task more difficult, or even not feasible, because of the fabrication tolerances on the cutting of the voussoirs and control over the geometry. Therefore, the starting pattern should not have singularities with very high valencies. As a strategy, singularities with a valency of five or more are transformed into their dual face, as shown schematically in Figure 11b. This keystone voussoirs replace the singularities and transforms the complex connection points of N voussoirs into N simpler connection points of 3 voussoirs.



(a) Regular areas (b) Singularities

Figure 11: Transformation rules of (a) regular areas and (b) singularities of the mesh into a tessellation

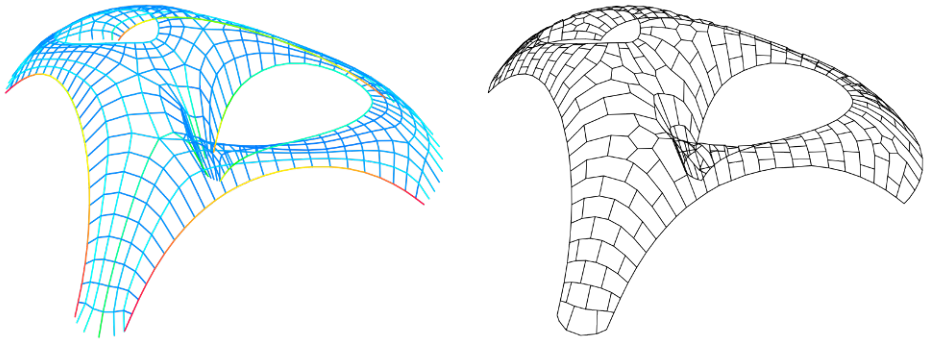
This way, the form-found thrust network of the Armadillo Vault in Figure 12a and 13a generates the tessellation shown in plan in Figure 12b and in space in Figure 13b.



(a) Thrust network (b) Tessellation

Figure 12: Conversion of the thrust network into a tessellation of the Armadillo Vault.

Regarding fabrication, shape and size uniformity between the voussoirs is prefered. However, high differences can occur in areas with important slope. Between form finding and fabrication, another step of uniform Laplacian smoothing can be done on the spatial network this time, constrained to the form-found surface. Additionally, a change of division rhythm can be applied by keeping every four edge instead of every second for instance, as suggested in Rippmann [4]. This allows to do form finding with a denser mesh, where the mesh faces are smaller than the tessellation's voussoirs.



(a) Form-found geometry (b) Tessellated vault

Figure 13: Form-found thrust network and vault tessellation of the Armadillo Vault.

The real project followed a different approach, both for the design of the form diagram and the tessellation. In the form diagram, the singularities are dualised into dense three-sided patches and some singularities lie on the boundary. The tessellation scheme does not derive directly from the thrust network and additional constraints are taken into account, such as the stone cutting technology. For more information, see Rippmann *et al.* [15].

6. Conclusion

This paper proposed a methodology for the design and exploration of structured quad-dominant meshes based on transformation rules. The generated meshes adapt to complex outer and inner boundaries, as well as point and curve features. The topological complexity is solved with a medial-axis-based block decomposition into patches simple to mesh and to smooth. This paper also proposed a strategy to derive force-, fabrication- and construction-aware tessellations from the form-found thrust networks.

The geometrical distorsions are partly a result of the limitations from Rigby [11], as pointed out by Fogg *et al.* [10] that solve this automatically using geometrical information of the shape's medial axis. However, we want to provide the designer a means to explore other topologies. Future work includes further use of the medial-axis-based block decomposition that expresses the pattern topology explicitely. Transformation grammar rules would allow an efficient exploration of the resulting mesh topology and generate new patterns. More uniform tessellations could be achieved. Other discrete shell typology like cable nets or gridshells and their specific structural constraints are also worth investigating.

References

[1] Block P. and Ochsendorf J., Thrust network analysis: a new methodology for three-dimensional equilibrium, *Journal of the International Association for Shell and Spatial Structures*, 2007.

[2] Rippmann M., Lachauer L. and Block P., Interactive vault design, *International Journal of Space Structures*, 2012.

[3] Rippmann M. and Block P., Funicular shell design exploration, *Proceedings of the 33rd Annual Conference of the ACADIA*, 2013.

[4] Rippmann M., *Funicular Shell Design: Geometric Approaches to Form Finding and Fabrication of Discrete Funicular Structures*, ETH Zurich, Department of Architecture Zurich, 2016.

[5] Davis L., Rippmann M., Pawlofsky T. and Block P., Innovative funicular tile vaulting: a prototype in Switzerland, *The Structural Engineer*, 2012.

[6] Rippmann M. and Block P., Rethinking structural masonry: unreinforced, stone-cut shells. *Proceedings of the ICE - Construction Materials*, 2013.

[7] Van Mele T., Panozzo D., Sorkine-Hornung O. and Block P., Best-fit thrust network analysis - Rationalization of freeform meshes. In Adriaenssens S., Block P., Veenendaal D. and Williams C. (ed.), *Shell Structures for Architecture: Form Finding and Optimization*, Routledge, 2014.

[8] Vouga E., Höbinger M., Wallner J. and Pottmann H., Design of self-supporting surfaces, *ACM Transaction on Graphics*, 2012.

[9] Panozzo D., Block P. and Sorkine-Hornung O., Designing unreinforced masonry models, *ACM Transactions on Graphics*, 2013.

[10] Fogg, H. J., Armstrong C. G. and Robinson T. T., Enhanced medial-axis-based block-structured meshing in 2D, *Computer-Aided Design*, 2016.

[11] Rigby D. L., Topmaker: a technique for automatic multi-block topology generation using the medial axis, *Technical report NASA/CR-213044*, 2004.

[12] Botsch M., Kobbelt L., Pauly M., Alliez P. and Levy B., *Polygon Mesh Processing*, AK Peters, 2010.

[13] Saha P. K., Borgefors G. and Sanniti di Baja G., A survey on skeletonization algorithms and their applications, *Pattern recognition letters*, 2012.

[14] Coons S. A., Surfaces for computer-aided design of space forms, *Project MAC Report MAC-TR-41,* MIT, 1967.

[15] Rippmann M., Van Mele T., Popescu M., Augustynowicz E., Méndez Echenagucia T., Calvo Barentin C., Frick U. and Block P., The Armadillo Vault: computational design and digital fbarication of a freeform stone shell, *Advances in Architectural Geometry*, 2016.

[16] Heyman J., *The Stone Skeleton: Structural Engineering of Masonry Architecture,* Cambridge University Press, 1997.