International Journal of SPACE STRUCTURES

Article

International Journal of Space Structures
1–16

DOI: 10.1177/09560599231207650
journals.sagepub.com/home/sps
Sage

A vector encoding for topology finding of structured quad-based patterns for surface structures

R Oval[1,2,3], R Mesnil[2], T Van Mele[3], P Block[3] and O Baverel[2]

Abstract

Topology-optimisation strategies for structural design of discrete surface structures result in unstructured patterns that require post-rationalisation to limit the number and complexity of the various structural elements. Fabrication-related objectives still demand further processing. Designers need *topology exploration*, before *topology optimisation*, following a generative-design approach that is decoupled from density tuning and shape design, to produce high-quality patterns. Such an approach allows to flexibly embed multiple design constraints, benefit from state-of-the-art form-finding algorithms and explore design trade-offs of the multiple requirements related to architecture, engineering and construction. This research investigates a parameterisation strategy to encode topological exploration of structured patterns based on quad meshes. The focus is set on singular vertices, which connect an irregular number of blocks, cables, or beams. Singularities are independent from pattern density and geometry, and have a fundamental influence on the qualitative and quantitative performance of the structure. We introduce an L-system encoding a quad-mesh grammar into a string that describes topological transformations of these singularities. Design applications to a net and a gridshell demonstrate the influence of singularities on the design of surface structures, and highlight the flexibility and generality of the approach in terms of geometrical processing and performance metrics. Beyond exploration, this parameterisation strategy opens to novel applications of search and optimisation methods for generative design of singularities in structural patterns.

Keywords

Shell structures, structural design, generative design, trans-parametric design, structural topology, singularities, rule-based design, mesh grammar, formal grammar, multi-objective design

Introduction

This research tackles the encoding or parameterisation of design and optimisation of the topology – or connectivity – of patterns of surface structures. The approach follows a generic generative-design strategy based on a grammar of topological rules encoded in an L-system, whose application is driven by multiple objectives, which can relate to aesthetics, sustainability, structural efficiency and construction affordability.

Motivation

Surface patterns apply to a wide range of systems in structural design, particularly shells, from voussoir tessellations to beam layouts, or force patterns. These patterns are characterised by their high structuredness. Optimisation-oriented design methods focus on structural efficiency, generating both the topology and the geometry of

[1]Laboratoire Navier, UMR 8205, École des Ponts ParisTech, IFSTTAR, CNRS, UPE, Champs-sur-Marne, France
[2]ETH Zurich, Institute of Technology in Architecture, Block Research Group, Zurich, Switzerland
[3]Department of Civil and Environmental Engineering, Form Finding Lab, Princeton University, Princeton, NJ, USA

Corresponding author:
R Oval, Laboratoire Navier, UMR 8205, École des Ponts ParisTech, IFSTTAR, CNRS, UPE, Champs-sur-Marne, France.
Email: robin.oval@princeton.edu

the pattern.[1,2] However, strong fabrication and assembly constraints drive the construction feasibility and affordability of these patterns at an architectural scale.[3] Although multi-objective approaches exist for the design of their geometry,[4] generic methods are missing for the design of the topology of such patterns. Moreover, decoupling topological from geometrical design and optimisation would provide a more modular workflow, whose steps can easily be exchanged from one design project to another and even evolve during the design process with the modification and specification of the design requirements. A generic approach to topological exploration allows the designer – architect or engineer – to decide on the specific form-finding algorithms to apply and on the specific performance metrics to use for evaluation.

Literature review

Structural topology optimisation. Topology optimisation classically refers to either continuous topology optimisation, optimising the material distribution and density in a given domain,[1] or discrete topology optimisation, optimising the element selection and cross-section in a dense layout called ground structure.[2] These methods rely on structural analysis to provide a design that performs the best for the input support and load conditions. However, post-processing is necessary to make these ideal designs feasible, as fabrication constraints command their applicability for architectural-scale structures. Although post-rationalisation using geometrical optimisation potentially allows enforcing specific constraints related to fabrication, the requirements are challenging to generalise, as they are specific to the project and can evolve along the design process.

Surface field integration. Surface structures are a specific set of structural systems that can be represented by a mesh whose vertices, faces, and edges can represent nodes, panels and beams. They can efficiently be described using only structured quad meshes, made of quadrilateral faces and a few irregular nodes, as such discrete parameterisations are natural to map 2D objects like surfaces. These structured quad meshes have embedded the regularity that is necessary for a wide range of patterns of structural systems to be built, are easy to scale up computationally through mesh densification for long-span structures, and leverage the methods and algorithms in the field of mesh processing. Cross-field integration provides the opportunity to inform pattern design by various requirements. Indeed, integration of a stress field provides structural efficiency, whereas integration of a curvature field provides fabrication ease.[5] However, topology and geometry are still coupled and considering combinations of multiple objectives remains challenging.

Grammatical generative design. Leaving mathematical, geometrical, and structural optimisation, generative design offers solutions to control the design space and its exploration. Grammatical generative design relies on a set of grammar rules that perform modifications on an initial design to produce and explore a variety of designs by applying combinations of these rules. Such generative approaches have already been applied to an extensive range of designs, starting from shape grammars for visual arts[6] to architectural design like Palladian houses[7] and many more applications.[8] Specifically to the field of generative structural design, the literature includes grammars with rules that integrate functional and structural aspects for houses,[9] towers,[10,11] halls,[12] bridges,[13] trusses,[14,15] or gridshells.[16] Grammars originated with formal grammars from the field of natural language processing with pioneer work of Chomsky in the 1950s.[17–19] Formal grammars evolved into another field of computer-aided design called L-systems, introduced by Aristid Lindenmayer in 1968.[20] L-systems combined formal grammars as a string encoder and computer graphics as a string interpreter to initially generate and explore the morphology and growth of plants.[21] Since then L-systems have found popular applications like fractal generation. L-systems have also been applied to structural topology optimisation using genetic algorithms by evolving a structural layout within a domain[22–24] or by evolving a partition of a domain.[25,26] These methods generally yielded layouts that were too unstructured and coarse for the design of surface patterns

Problem statement

Existing methods for topology optimisation are powerful and efficient to inform the designer with a topological design optimised for a specific criterion, like material volume of a ground structure using layout optimisation or strain energy of a quad mesh using field integration. However, strategies are missing to explore structured designs, characterised by a regularity that is necessary for architectural and construction reasons, and tackle multi-objective exploration with flexibility in performance metrics. Generative methods that decouple topological and geometrical design allow such flexibility. Although L-systems have been used to generate structural patterns using grammars based on the addition of nodes and edges, they lacked structuredness for the design of shell systems like vaults, nets, or gridshells. The fundamental question lies in the encoding of the design problem to parameterise a design space of structured designs to search, that is, developing a L-system with a grammar that allows for the generation of viable surface structures for architecture.

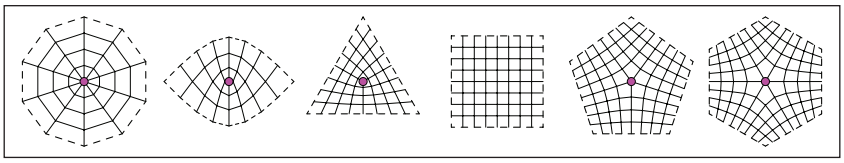

Figure 1. Singularities in quad meshes have a valency different from four. The singularities in pink include a high-valency pole singularity and singularities with valencies of two thee, five and six. Boundary singularities, which have a valency different from three, are not highlighted.

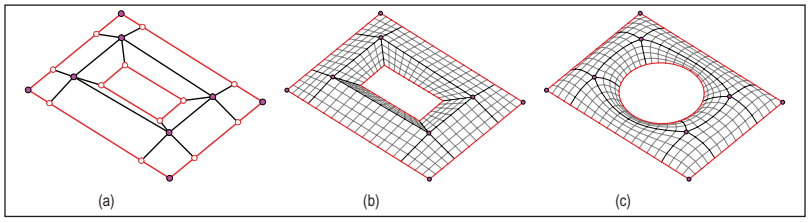

Figure 2. Approaching exploration of singularities in quad meshes via a coarse quad mesh in black to decouple topology from density and geometry. The boundaries are in red, the singularities in pink and the dense quad mesh in grey: (a) topology, (b) density, and (c) geometry.

Research objectives

We aim to develop an encoding strategy for the description of the topology of quad meshes. *Quad-mesh parameterisations* can describe a large variety of structural systems, including voussoir tessellation of vaults and beams layouts of gridshells. Moreover, the structuredness of quad meshes provide a key regularity to build structures at an architectural scale, while their *singularities* allow to locally change the layout of elements. A singularity, or an irregular vertex, is opposed to a regular node, which is adjacent to four other nodes in a quad mesh – or three if the node is on the boundary. Figure 1 shows a set of singularities in mesh patches, including a pole singularity and singularities with valencies of two, three, five and six. As for defects in crystals, singularities have a deep influence on the various performances of a design.

In a workflow as in Figure 2 where the topology of a *coarse quad mesh* is explored first, the pattern's density and shape then result from integer and real parameters, respectively, and can be designed independently with full control on the density and geometry of the resulting quad-mesh pattern.[27] This paper provides an *L-system* encoding, a type of formal grammar coupled with a decoding generation machine, to describe the singularity design of quad meshes. This strategy is compatible with form finding and multi-objective trade-off search as downstream steps of any design workflow. The encoding must allow for the description of all quad-mesh topologies, for *comprehensiveness*, and only quad-mesh topologies, for *efficiency*. Describing a quad-mesh topology as a succession of such rules or operations in a vector encoding offers then the potential to be interpreted with integer programming, genetic-algorithm or machine-learning processing.

Contributions

Achieving the desired granularity of a design space that includes all but only quad-mesh topologies is obtained by developing an L-system based on grammar rules that modify the fundamental strip structure in quad meshes, which indirectly modifies the singularities. Section 2 develops the methodology and its mathematical and computational foundation for the description of quad-mesh topology, with a quad-mesh grammar and its string encoding and decoding as an L-system. The formal grammar based on this encoding strategy is developed and the structure of the resulting design space discussed, along with its potential for grammatical topology optimisation. Section 3 applies this method to the structural design of patterns for different workflows and structural systems to demonstrate the relevance and flexibility of this approach. The two case studies are the form finding of a cable net and the multi-objective design of a steel-and-glass gridshell. They highlight the relation between the topology, the geometry and the structural and multi-objective performance of quad-mesh patterns. This work is implemented and shared publicly in compas_singular[28] as a Python package of COMPAS,[29] an open-source Python-based computational framework for collaboration and research in architecture, engineering and digital fabrication.

Mathematical and computational foundation

The proposed exploration approach decouples topology, density and geometry, allowing to focus on the topology of a coarse quad mesh before densification and geometrical processing. Figure 2 shows how a coarse quad mesh, in

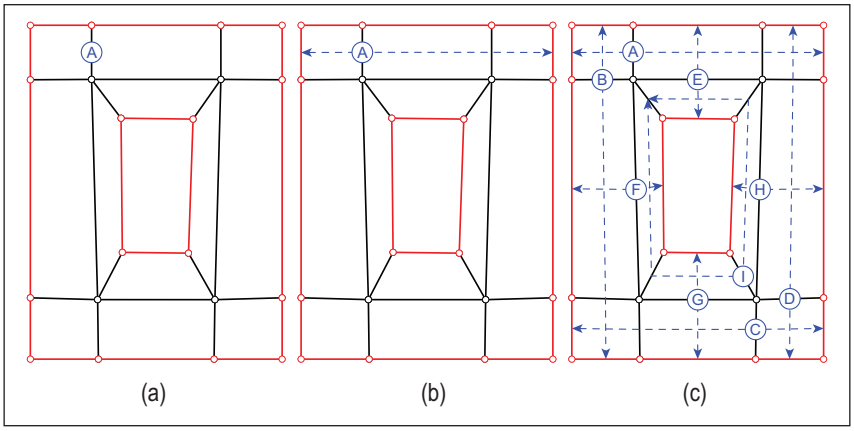

Figure 3. The strip structure, in blue, completely defines the topology of a quad mesh and its singularities, with nine strips from A to I here. The strip data is collected as the lists of topologically opposite edges across the quad faces: (a) get edge, (b) collect strip, and (c) repeat.

black with its boundary in red and singularities in pink, can describe the topology of a pattern before densification into a dense quad mesh, in grey, and constrained relaxation on a surface. A specific mesh grammar is developed for the exploration of the topology of singularities in quad meshes by modifying the connectivity of such coarse quad meshes. Then a formal grammar and L-system encoding of the mesh grammar is presented and its properties discussed for potential automatic search and optimisation.

Quad-mesh grammar

This grammar of rules applies topological modifications allowing to indirectly change the set of singularities in a quad mesh by changing the fundamental strip structure in quad meshes, in order to create a design space that encapsulates all but only quad-mesh topologies.

Strip structure. Beyond vertices, edges and faces, quad meshes contain strips that constitute a description of quad meshes at a larger scale. Strip-based modelling already found applications for digital[30,31] or physical[32,33] approaches, also referred to as loops, rings or chords. The strips also correspond to the independent parameters for densification of a quad mesh. The connectivity of these quad mesh strips depends on topology, not its geometry. The strips are constructed based on the relationship between pairs of opposite edges across quad faces. The strip data is collected as a list of edges, as illustrated in Figure 3:

1. **initiate data**: start with an initial complete list of edges;
2. **collect one strip**: initiate the collection of a strip by getting one edge from the list (label A in Figure 3(a)). Complete the strip by adding the edges across the adjacent quad faces in both directions (strip A in Figure 3(b)). Termination occurs when collection yields two boundary edges in the case of an open strip (strips A to H), or until it forms a loop in the case of a closed strip (strip I);
3. **update data**: store the strip data as the list of collected edges and remove them from the initial list of edges;
4. **collect all strips**: repeat from step 2 until the initial list of edges is empty (Figure 3(c)).

This structure offers the opportunity to develop a quad-mesh grammar that is tailored for the exploration of a comprehensive design space constrained to quad meshes.

Topological grammar. As the topology of a quad mesh is fully determined by its strip structure, only two low-level rules are necessary to be able to modify its topology: *adding* a strip and *deleting* a strip. Figure 4 illustrates how these two reciprocal rules apply to different configurations of strips and their corresponding polyedge – a continuous series of edges – in blue.

Strip addition. To add a strip along a polyedge described as a list of n adjacent vertices $[V_0 - ... - V_i - ... - V_{n-1}]$, the following operations are sequentially applied on the polyedge, as illustrated in Figure 5:

1. **first element**: two vertex copies of V_0, V_0' and V_0'', are created and the first edge $V_0 - V_1$ yields a triangular face (V_0', V_1, V_0'');
2. **regular elements**: $\forall i \in [\![1; n-3]\!]$, two vertex copies of V_i, V_i' and V_i'', are created, the edge $V_i - V_{i+1}$ yields a triangular face (V_i', V_{i+1}, V_i'') and the previous triangular face $(V_{i-1}', V_i, V_{i-1}'')$ becomes a quad face $(V_{i-1}', V_i', V_i'', V_{i-1}'')$.
3. **last element**: two pairs of vertex copies of V_{n-2} and V_{n-1}, V_{n-2}' and V_{n-2}'', and V_{n-1}' and V_{n-1}'',

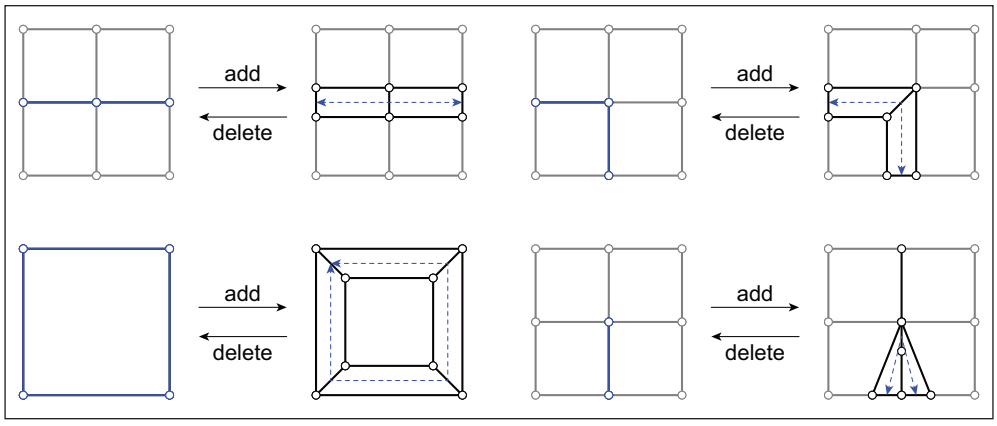

Figure 4. A low-level grammar rules of two reciprocal rules that add and delete strips and their corresponding polyedges in quad meshes, applied to different configurations of elements in blue.

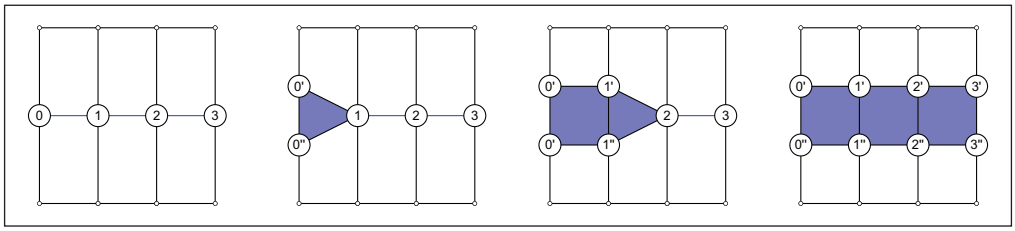

Figure 5. Detailed addition of a strip along a polyedge.

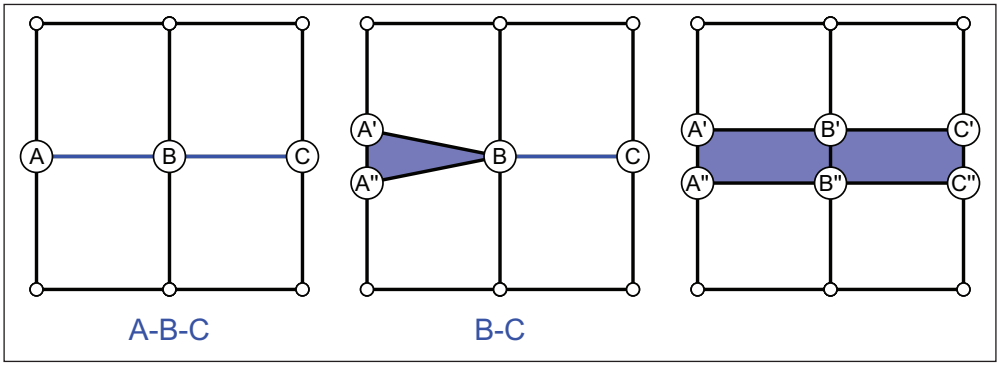

Figure 6. Add a strip changing the density.

respectively, are created and the last edge $V_{n-2}-V_{n-1}$ yields directly a quad face $(V'_{n-2},V'_{n-1},V''_{n-1},V''_{n-2})$;

4. **clean**: the old vertices, which are now disconnected, are deleted.

When a pair of new vertices V'_i and V''_i replaces an old vertex V_i, V_i is substituted for V'_i in the faces on the left side of the polyedge and for V''_i on the right side. This left/right convention is defined by the sense of the polyedge and the normal of the vertices. Therefore this method is specific to orientable meshes.

Visually, the polyedge is unzipped to become a strip. The temporary pseudo-quads have their poles oriented downstream the polyedge. In Figure 6, a strip is added along polyedge $(A-B-C)$ without modification of the singularities, only the density, because the polyedge is topologically parallel to an existing strip. In Figure 7, a strip is added along polyedge $(A-B-C)$ inducing new 3- and 5-valent singularities.

Some modifications are necessary for the addition rule to be valid for strips of any configuration.

To add a closed strip, marked with °, along a closed polyedge $[V_0-\ldots-V_i-\ldots-V_{n-2}-V_0^\circ]$, the last edge $(V_{n-2}-V_0)$ becomes the quad face $(V'_{n-2},V'_0,V''_0,V''_{n-2})$ using the vertices added from the first vertex, as illustrated in Figure 8 along polyedge $(A-B-C-D^\circ)$.

A closed polyedge $[V_0-\ldots-V_i-\ldots-V_{n-2}-\ldots-V_0]$, without °, marks the addition of an open strip. With V_0

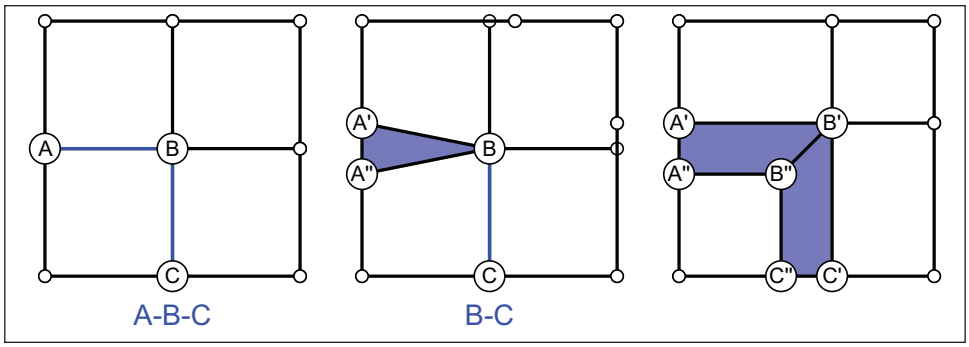

Figure 7. Add a strip changing the singularities.

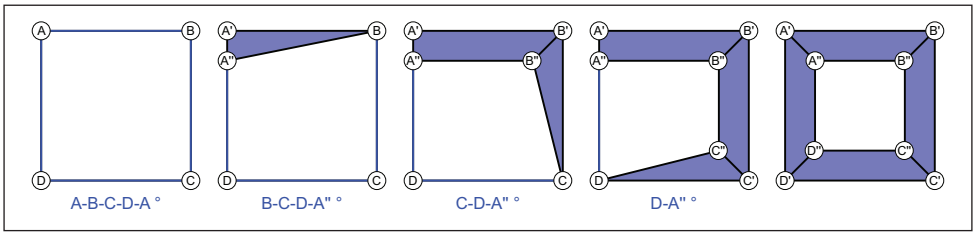

Figure 8. Add a closed strip.

occurring twice, this configuration is a case of a strip with repeated vertices.

Polyedges with repeated vertices have at least one vertex V_i occurring more than once in the polyedge. When V_i is replaced in the mesh by the vertices V_i' and V_i'', the polyedge updates and replaces the remaining V_i. When inserting a strip along the polyedge:

$$...-V_i-V_{i+1}-...-V_{j-1}-V_i-V_{j+1}-...$$

and after deletion of V_i, the remaining polyedge does not exist in the mesh anymore and is updated as:

$$V_{i+1}-...-V_{j-1}-f(V_i',V_i'')-V_{j+1}-...,$$

where $f(V_i',V_i'')$ is the combination of the new vertices V_i' and V_i'' that constitutes the shortest sub-polyedge from V_{j-1} to V_{j+1}, in order to reconstruct the polyedge. This shortest path is found using an A* search[34] in the graph made of the edges connected to vertices V_{j-1}, V_i', V_i'' and V_{j+1} only. If the repeated vertex is the last vertex of an open polyedge at the position $n-1$, then the sub-polyedge is from V_{n-2} to the boundary. The choice for the shortest polyedge is not constraining when combined with other rules to lengthen the polyedge. In Figure 9, the self-crossing polyedge $[A-B-C-D-E-B-F]$ yields a self-crossing strip. When deleting B, the shortest path from E to F is $[B'-B'']$. The remaining polyedge to add therefore updates to $[C-D-E-B'-B''-F]$.

In Figure 10, the self-overlapping polyedge $[A-B-A]$ yields a self-overlapping strip. When deleting A, the shortest path from B to the boundary is either A' or A''. The two options yield the same result.

In order to add a strip with pole singularities at one or two extremities, the corresponding face extremities become triangles with a pole at the extremity of the strip to count as pseudo-quad faces. The poles are marked as * like $V_0^*-...-V_{n-1}^*$ for strip with two poles, as shown in Figure 11. A strip with two pole extremities must stem from a polyedge with at least two edges. On the contrary to a regular extremity, the pole extremity of a strip is not necessarily on the boundary.

Strip deletion. To delete a strip by collapsing it into a polyedge, the following operations are sequentially applied on the strip, as illustrated in Figure 12:

1. get the edges of the strip to delete;
2. build a graph from these edges;
3. collect the disconnected parts of the graph as groups of vertices;
4. delete the faces of the strip;
5. merge vertices per group into a new vertex.

Visually, the strip edges are collapsed to zero-length edges, resulting in the collapse of the strip faces. This process applies to any configuration of strips, open, closed, with repeated vertices and with poles, as shown in Figure 13.

Deleting a strip causes the collapse of a boundary if less than three edges represent the boundary after deletion of the strip edges. This collapse changes the Euler's characteristic of the mesh and therefore its shape topology,

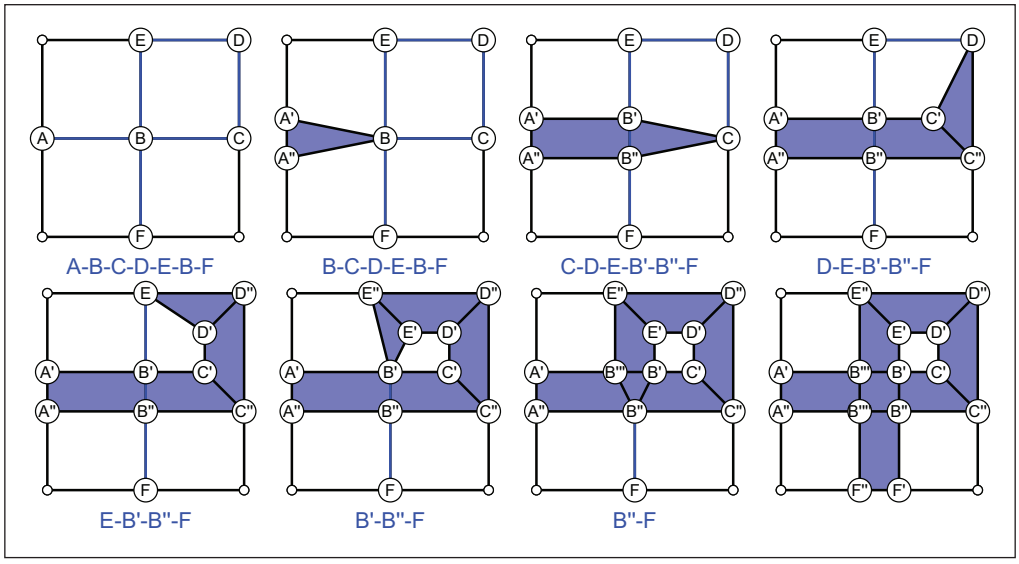

Figure 9. Add a self-overlapping strip with update of the polyedge.

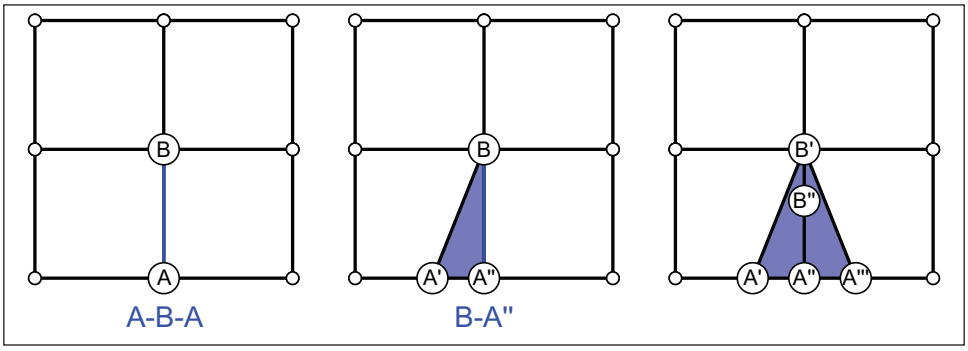

Figure 10. Add a self-crossing strip with update of the polyedge.

though this grammar should only modify the pattern topology, that is, its singularities. Figure 14 illustrates how the problem and its solution, obtained by refining some strips. Before deleting a strip, a verification predicts potential boundary collapse if:

$$| E_{strip} \cap E_{boundary} | < 3, \quad (1)$$

where E_{strip} is the set of edges of the strip and $E_{boundary}$ is the set edges of the boundary. The boundary edges to remain are refined to avoid boundary collapse by guaranteeing the minimum number of three edges. For one strip deletion, there is always at least one remaining edge to refine. Indeed, the minimum number of boundary edges is three, and the maximum number of boundary strip edges is two. Refining a strip does not add singularities, it only modifies its density, using a special case of strip addition when it is topologically parallel to an existing strip, as illustrated in Figure 6. Therefore, to avoid the collapse of the boundary marked in grey due to the deletion of the strip in blue in Figure 14, the remaining strips that end at this boundary are refined. One of the two polyedges along these strips serve as input for strip addition, to split a strip in two. Should one boundary edge remain, the strip is subdivided in three. Should two boundary edge remains, the two strips are both subdivided in two, to avoid any bias.

Strip data update. After the application of each rule, the strip data requires an update. Re-collecting all strips is not efficient, as only the added or deleted strip and the ones crossing the modified faces have been modified. Due to topological modifications, the labels of the vertices and faces change. Nevertheless, the labels of the strips are preserved in order to keep strip attributes and to combine multiple rule deletions for instance.

When the deletion rule is applied:

- the deleted strip is removed;
- the old vertices V_i' and V_i'' are replaced by the new one V_i;
- the collapsed edges (V_i', V_i''), which become (V_i, V_i), are removed;

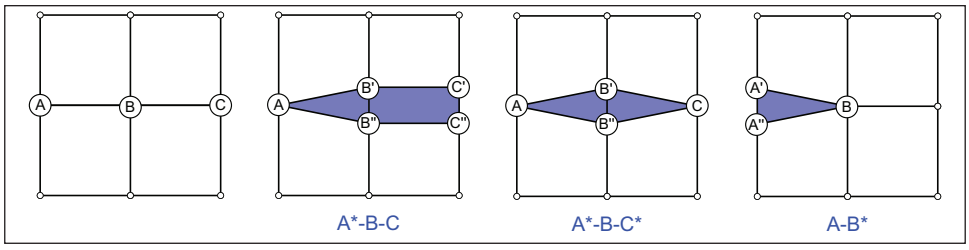

Figure 11. Add strips with poles.

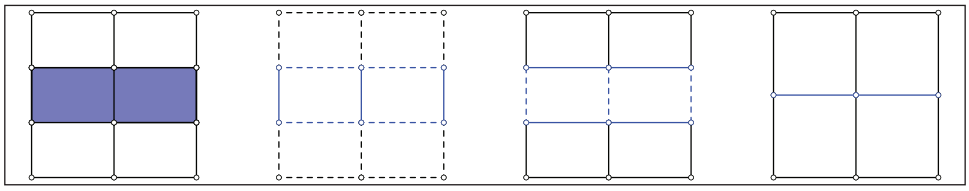

Figure 12. Delete a strip.

- the duplicated edges $(V_i',V_{i+1}'),(V_i'',V_{i+1}'')$, which become $(V_i,V_{i+1}),(V_i,V_{i+1})$, are merged to become (V_i,V_{i+1}).

When the addition rule is applied:

- the added strip is stored as $[(V_0',V_0''),...,(V_{n-1}'V_{n-1}''))]$;
- the old edges (V_i,V_{i+1}) are replaced by the pair of new edges (V_i',V_{i+1}') and (V_i'',V_{i+1}'') in the order to complete the strip;
- the old vertices V_i in the edges (V_i,V_j) are replaced by the new one among V_i' and V_i'' that is adjacent to V_j.

As illustrated in Figure 15, a designer can interactively select polyedges along which to add a strip, like the closed polyedge in blue, and select strips to collapse into a polyedge, like the two strips in green, changing the connectivity of the nodes and therefore the singularities, in pink.

Selecting these mesh elements allows interactive exploration. However, directly encoding these rules into a vector as a list of element labels, whether strips, edges or vertices, to modify is not possible. Indeed, the connectivity evolves with a varying number of elements along with their labels. Therefore, another strategy is necessary to encode the application of the grammar rules in a vector to be generated and read by exploration and optimisation algorithms.

Formal grammar

An approach based on L-systems permits to encode the mesh grammar into a string using a formal grammar, a set of characters specifying orders to a moving object that applies the two rules of the quad-mesh grammar. In the following figures, the moving object is depicted as a blue arrow marker.

Movement operations. Instead of moving freely within a blank domain, the marker moves along the edges of the mesh. A node V_i as position and an adjacent node V_j as direction define the marker (V_i,V_j). Two movement operations allow to relocate the markers anywhere in the mesh: *turn* moves the marker to the next edge of the leftward face following an anticlockwise rotation, shown in Figure 16(a), and *pivot* moves the marker to the next edge rightward to the position node following a clockwise rotation, shown in Figure 16(b). The orientations are defined by default by the mesh normal. Thereof, the marker can move to select polyedges and strips as inputs for the mesh-grammar rules.

Thanks to these movement operations, the marker visits the mesh edges based on a string encoding the sequence of these two operations.

Modification operations. While moving, the marker can select the mesh elements to which apply strip rules. Two modification operations allow the marker to modify the quad-mesh topology. The *add* operation, shown in Figure 17(a), toggles the collection of a polyedge, in red, as the successive positions of the marker when applying a combination of *turn* and *pivot* operations and adds a strip along it. The first *add* operation initiates the collection of the polyedge and the second one stops and adds the corresponding strip. The * and ° parameters are specific for the addition of strips with poles and closed strips, respectively. The *delete* operation, shown in Figure 17(b), deletes the strip transverse to the marker. Before deleting the strip, the marker is moved using the *pivot* operations until the marker lies on the polyedge that correspond to the collapsed strip.

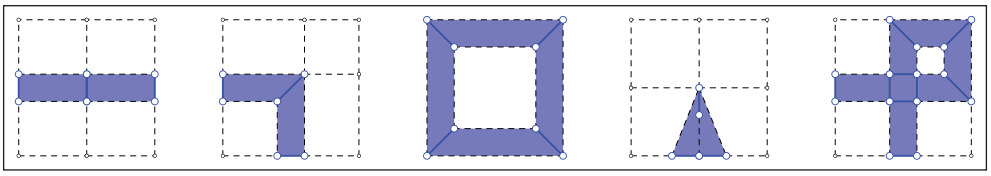

Figure 13. Disconnected graphs of vertices to merge for the strip deletion rule.

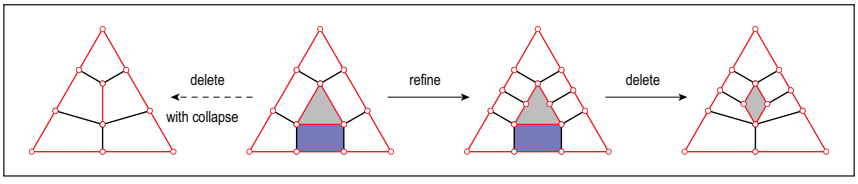

Figure 14. When deleting some strips, in blue, refining other strips avoids boundaries to collapse to less than three edges and close an opening, in grey.

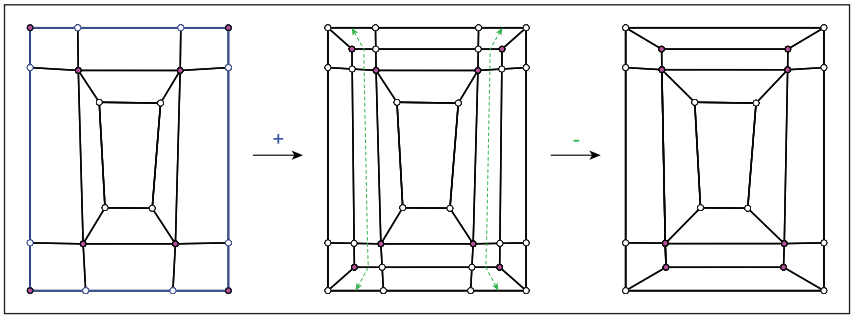

Figure 15. Selecting the polyedge in blue to add a strip and the strips in green to delete changes the connectivity of the singularities.

Figure 18 further illustrates the addition of a strip by representing the topological modification at different steps.

If a *delete* operation is applied during polyedge collection for the *add* operation, the collected polyedge must be updated. The old polyedge $[V_0,...,V_{n-1}]$ is updated by replacing the old vertices by the new vertices after strip deletions in a new polyedge $[V'_0,...,V'_{n-1}]$, and removing redundant vertices when two successive vertices are identical.

In the example in Figure 19, addition begins and the one-edge polyedge $[0,1]$, in red, is collected. Then, the strip in green is deleted, mapping the old vertex 1 to the new vertex 6. Therefore, the polyedge $[0,1]$ becomes $[0,6]$. Finally, addition ends, and the strip in red added.

Design space structure

Thanks to the L-system approach, the two-rule mesh grammar can be encoded in a string to describe a topological design, when the initial quad-mesh topology and position of the marker are known. The four-character formal grammar is *T* for turn, *P* for pivot, *D* for delete and *A* for add. The parameters for closed strips,°, and strips with poles, *, are variations of A. The relation between the string and the pattern is the one of genotype and phenotype, as used in evolutionary algorithms. Various aspects will influence the application of search algorithms of patterns via their string: the lack of isomorphism between the pattern phenotype and string genotype, the potential mutations on the string genotype space, and the resulting space metric – or distance.

Isomorphism. The string encoding can generate any quad-mesh pattern topology thanks to the underlying mesh grammar. There is therefore a surjection from the genotype to the phenotype. However, different strings can result in the same pattern. For instance, operation that move the marker without modifying the strips do not have an effect on the pattern, or a strip can be added starting from one extremity or the other. This redundancy means therefore that there is no injection from the genotype to the phenotype. Thus, there is no bijection, or isomorphism, between the genotype space of strings and the phenotype space of quad-mesh topologies for the proposed L-system.

This lack of isomorphism can be problematic for the application of evolutionary algorithms.[35,36] Indeed, a pool of varied genotypes does not then necessarily result in a

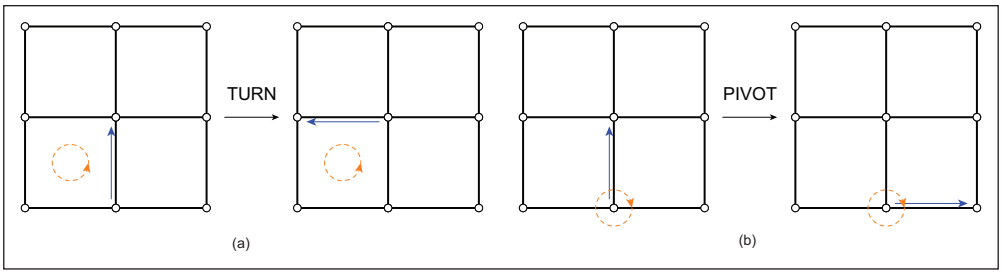

Figure 16. The two movement operations to relocate the marker, in blue, along the edges of the mesh: (a) *turn* operation and (b) *pivot* operation.

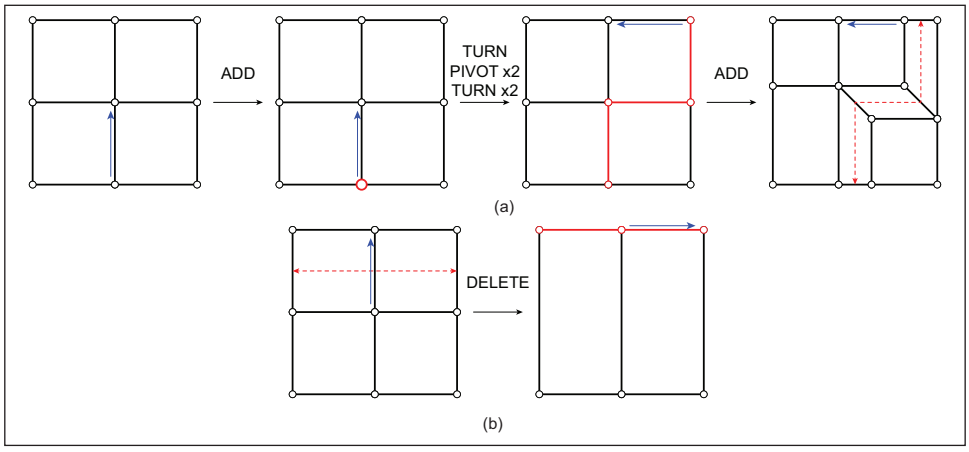

Figure 17. The two modification operations to modify the mesh based on the position of the marker and the collection of input polyedges or strips, in red: (a) *add* operation and (b) *delete* operation.

diverse pool of phenotypes. A lack of variety of designs and therefore of performances may cause the genetic algorithm to stagnate and stop improving. This risk depends on the degree of redundancy, that is, the proportion of genotypes resulting in the same phenotypes.

Mutations. For editing a string or combining different strings, modifications can be applied, also called mutations mutations, following the analogy with genotypes.

Three canonical modifications for strings stem from the work of Levenshtein in linguistics and information theory[37]:

1. *insert(X, i)*: insert character *X* at index *i*;
2. *remove(i)*: delete the character at index *i*:
3. *substitute(i, X)*: replace the character at index *i* by character *X*.

For instance, inserting a combination of Ts and Ps between two As will change the added strip, as well as the other rules.

Distances. The topological distance between two quad meshes is defined as the minimum number of strips to add and delete to obtain an isomorphism between the two quad meshes, as defined by Oval et al. [38] This definition of the distance applies to the level of the phenotype of the design. The allowed mutations can also help measure a distance between two strings, at the level of the genotype. The Levenshtein distance between a pair of strings is defined as the minimum number of modifications to apply using insertions, deletions and substitutions.[37] Other string metrics, like the Hamming distance[39] that only considers substitutions, can be considered. These distances verify the three properties of distances on the space of strings: the distance is symmetric; the distance from a string to itself is null and if their distance is null two strings identical; and the triangle inequality is respected. The phenotype and genotype distances structure their string and mesh design spaces, respectively, differently, meaning that similarity between two objects is different between these two spaces, which influences exploration and search. For instance, due to the lack of isomorphism between the string space and the pattern space, two different strings yielding the same mesh have a non-null genotype distance but a null phenotype distance.

Structural design applications

The presented grammar is applied to two illustrative structural systems: a net and a gridshell, to produce quad-based

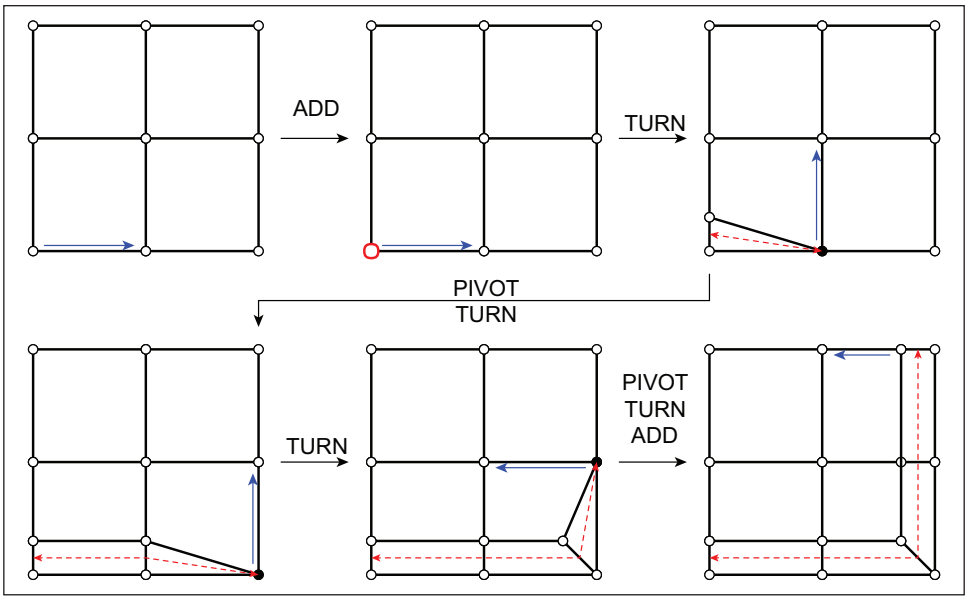

Figure 18. Illustrating the application of a pair of *add* operations by representing the topological modifications at different steps.

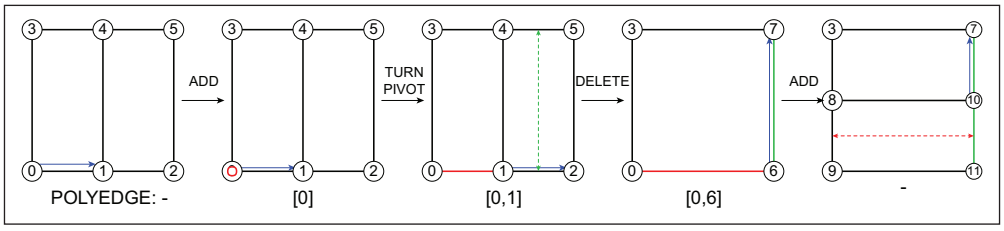

Figure 19. The collected polyedge in red for strip addition is updated after deleting the strip in green, by replacing the old vertices by the new ones.

patterns with different topologies. The influence of topology on geometry and performance is highlighted.

Form-finding of a cable-net

Cable nets are lightweight structures that can span large areas thanks to negative double curvature and prestress. To design such shapes, Frei Otto used experimental methods to obtain minimal surfaces resulting from a soap film based on a set of boundary conditions. However, geometries at an architectural scale are discretised. Therefore, the structure is not an isotropic and homogeneous material like soap but a finite set of cables in the case of a net. The surface must be discretised and the transition from a continuous surface with a stress distribution to a discrete pattern with a force distribution can influence the resulting form-found geometry in equilibrium. One of the most famous examples of Frei Otto is based on a rectangle and two offset circles as boundary conditions. Here, these boundaries serve as input for a skeleton-based decomposition algorithm[40] to obtain an initial coarse quad mesh with singularities at the farthest from the boundaries and at the kinks, shown in Figure 20(a). Four additional topologies are produced in Figure 20(b), resulting from different applied rules combining the addition of an intermediary strip between the rings and the deletion of the strips along the rectangular or the circular boundaries.

The coarse quad meshes are all systematically densified based on the same target length. Although a non-uniform distribution of prestress can be applied to control the shape, a uniform Laplacian smoothing, a mean curvature flow, is performed on the dense quad meshes to obtain a discrete minimal surface in equilibrium with these boundary conditions. Smoothing is applied for 500 iterations, enough to provide shape convergence, and with a damping value of 0.5. As form follows topology, the shapes in equilibrium differ although the same form-finding algorithm has been applied. Figure 21 shows the superposition of the five meshes with red lines highlighting the local range of deviations between them. The average and maximum deviations equal 29% and 67% of the average edge length, respectively.

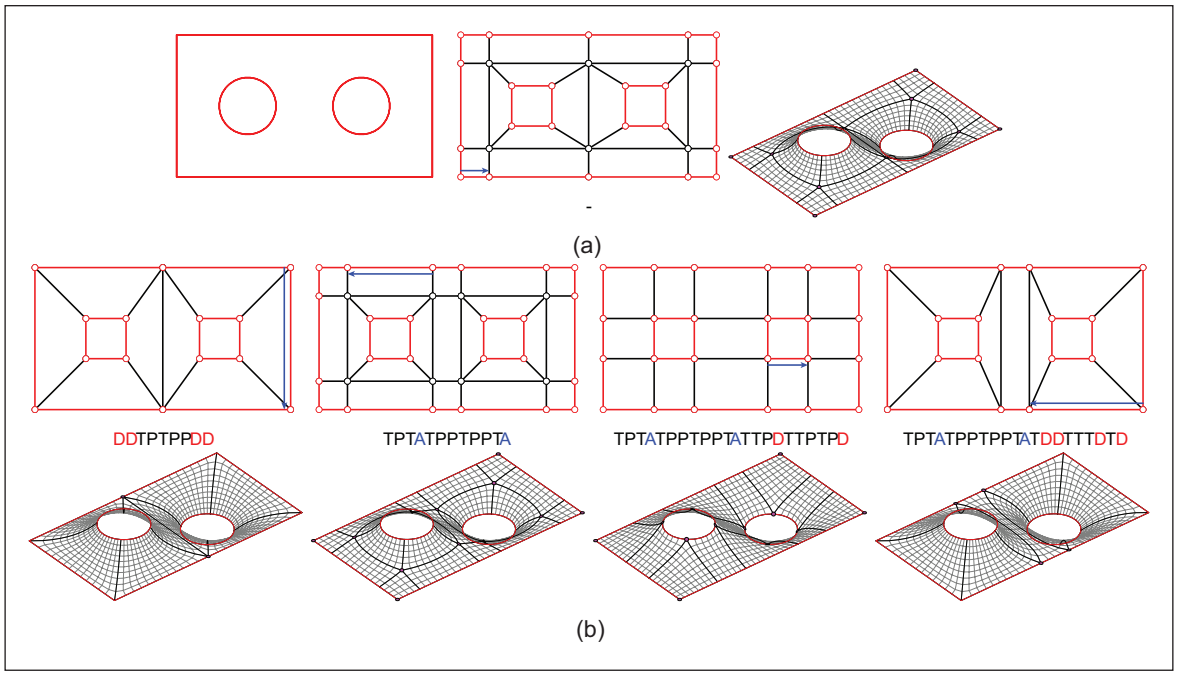

Figure 20. Five topological designs for the discretisation of Frei Otto's soap film based on a rectangular and two circular boundaries and minimal surface meshes in equilibrium for the five pattern topologies: (a) initial design and (b) four other designs with corresponding generative strings.

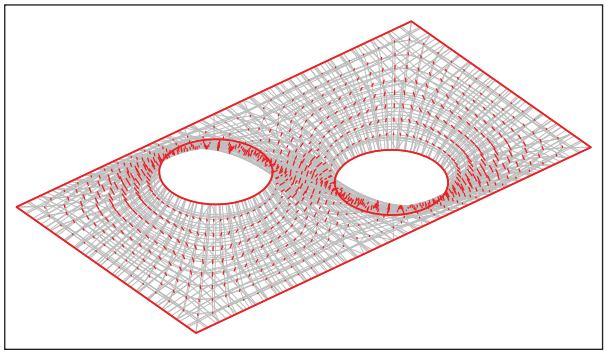

Figure 21. Local maximum deviations among the five meshes marked as red lines.

Other patterns than quad meshes may be relevant for cable nets. A family of structured patterns can be obtained by applying Conway et al. operators.[41] Here, the new patterns in Figure 22 are based on the initial quad-mesh parameterisation, with corresponding singularities. The three examples are quad dominant, with a dual, a diagonal (*join*) and a diagonal-dual (*ambo*) patterns, though an entire design space of hybrid tessellations of triangles, quads, pentagons and so on remains to be explored.[42,43]

Multi-objective design of a gridshell

The topology of a pattern influences its aesthetics, as well as its structural performance, its fabrication process, and its sustainability. The design choice of singularities can modify the trade-off between the multiple objectives without necessarily changing the density, that is, the number of elements to produce and assemble, nor the overall geometry. The gridshell of the Great Court Roof of the British Museum, shown in Figure 23, serves as case study to highlight this influence of topology on multi-objective trade-offs. This glazed steel gridshell spans between a rectangular and a circular boundaries and is supported vertically along its entire boundary and horizontally at the four external corners. The pattern of steel beams and glass panels is revisited as a quad-mesh pattern.

The designs in Figure 24, shown in top view, have different pattern topologies. Initial design 0 results from a skeleton-based decomposition of the surface.[40] Then, different combinations of grammar rules encoded in the descriptive strings produce the other designs 1–7. This open-ended exploration aims at generating highly different patterns in terms of topology and performance. The density is set to a 0.5 m target length and the quad mesh is relaxed using Laplacian smoothing on the original shape of the roof as a constraint,[44] for 100 iterations and a damping value of 0.5 with area weights. The same workflow is applied to each pattern topology. This workflow can be freely modified but the resulting multi-objective performance depends on it.

The engineering and construction details are found in Sischka et al.[45] The beams of the actual structure have a box cross-section with a width of 80 mm and a height varying from 80 to 200 mm, oriented with the surface normal.

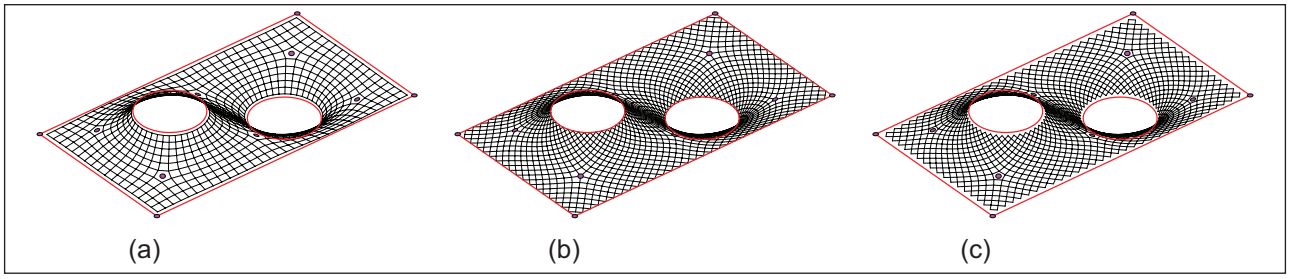

Figure 22. Other types of patterns resulting from the application of Conway operators with equivalent vertex or face singularities: (a) *dual* pattern, (b) *join* pattern, and (c) *ambo* pattern.

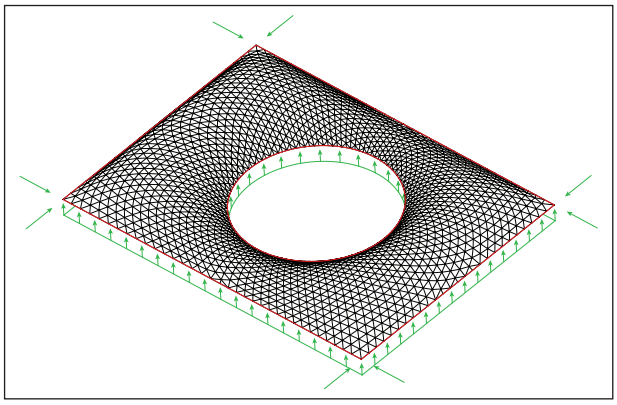

Figure 23. Shape, pattern and support conditions of the steel and glass gridshell of the British Museum.[44]

Here, the S355 steel beams all have the same cross-section to favour patterns with a homogeneous force flow, although another cross-section could have been considered to stiffen the simply-supported boundary.[46] The beams have built-in connections and must be stiffened due to the lack of triangulation of the quad mesh: they have a width of 250 mm, and an assumed wall thickness of 20 mm. The height of the beams is minimised to reduce the structural weight while complying with the relevant structural requirements. Several structural requirements apply to different load cases and combinations. The load cases are: the structural self-weight G; a downwards dead load $G' = 0.6\,\text{kN/m}^2$ for a 24 mm thick glazing; a downwards projected live load $Q = 0.5\,\text{kN/m}^2$ for snow loads, without taking into account geometry factors. The load combinations are: the Serviceability Limit State (SLS): 1.0(G+G′)+1.0Q; the Ultimate Limit State (ULS): 1.35(G+G′)+1.5Q. The structural requirements are: a maximal SLS deflection of 140 mm, corresponding to the maximal span over 200, though a pre-deformation compensates the deflection of the actual structure; a maximal ULS stress utilisation of 100%; a minimal ULS first load buckling factor of 4, as for the actual structure. The pre-deformation, as well as the imperfections, based on the first buckling mode with a maximal value of 140 mm, are not taken into account.

After topological exploration, geometrical processing, and cross-section minimisation, the following performance objectives are evaluated, additionally to structural mass, aiming to be minimised for the design of this steel and glass gridshell:

- structural objectives using second-order analysis with the Finite Element Analysis software Karamba3D[47]: stiffness as the maximum SLS deflection F; strength as the maximum ULS stress utilisation U; stability as the inverse of the ULS first load buckling factor B;
- fabrication objectives: general panel curvature as average curvature of quadrilateral panels C ; general panel skewness as average skewness of quadrilateral panels S; beam length disparity as edge length standard deviation L.

Figure 24 depicts the evaluation of these six performance metrics to minimise for the eight designs. The metrics X are normalised as X* by the maximum value, to be bounded by 1.00. The minimum value is marked by the dark shade area and the relative performance of the design for each metric is highlighted by a different gradient from red to green, from the worst to the best value among this set of designs, respectively. These numerical results highlight the trade-offs between the multiple objectives. Indeed, no design outperforms the other ones for each metric. For instance, design 1 performs among the best regarding buckling, panel skewness and beam length, while design 6 performs among the best regarding deflection, stress utilisation and panel curvature. It belongs to the designers to explore the Pareto front of trade-offs to decide on the most suitable design. Numerical results can also inform further exploration of design options and the generation of more topological designs using the presented L-system encoding. The designer could produce hybrids of the strings encoding designs 1 and 6, for instance. Or automatically produce variations of the description strings and their resulting designs thanks to state-of-the-art search algorithms.

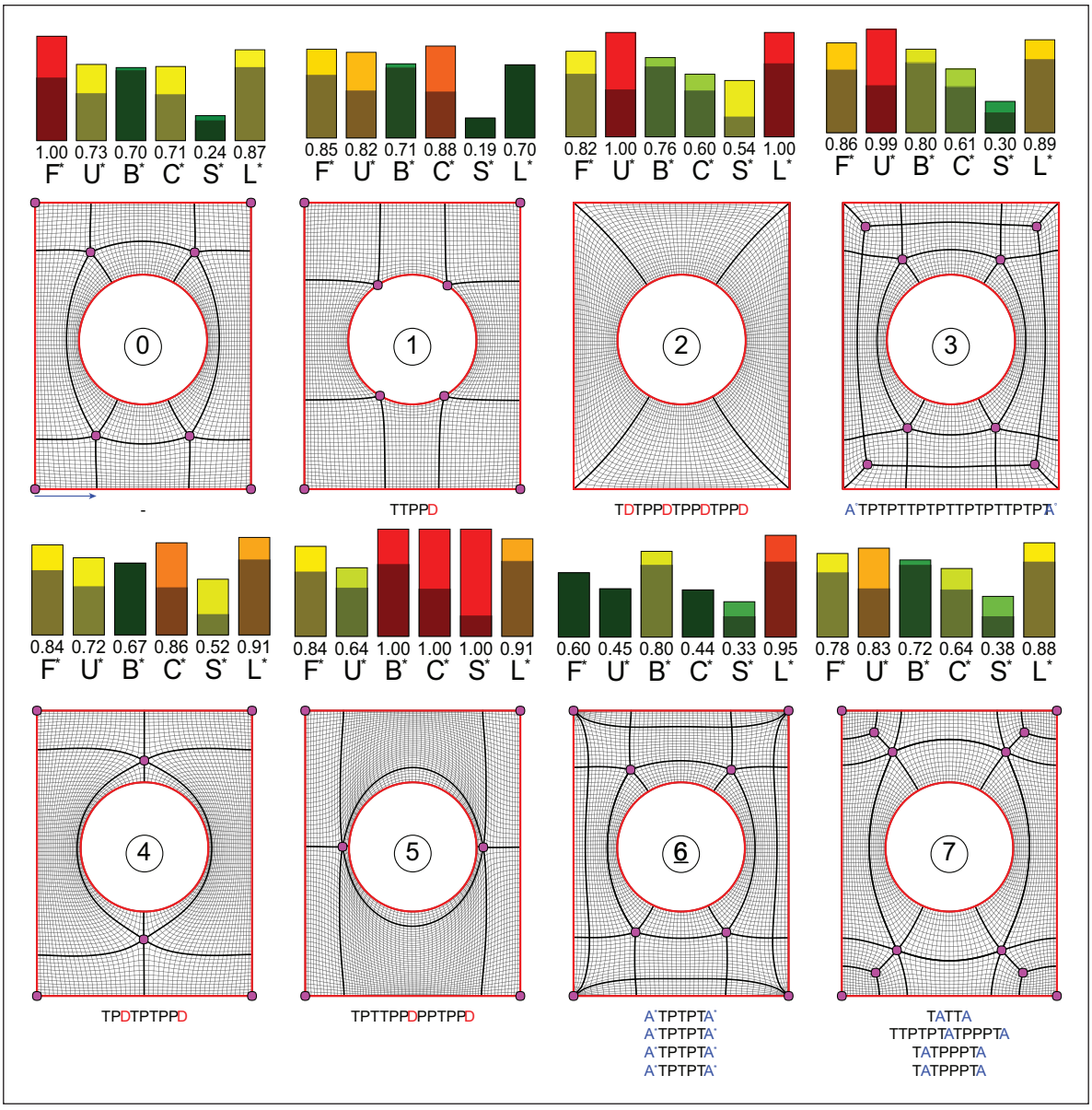

Figure 24. Topological exploration using the presented L-system and multi-objective performance comparison of quad-mesh patterns for the gridshell of the British Museum. The range of performance is represented by a colour gradient per objective, with the best designs in green and the worst ones in red. The minimum per metric is marked with a darker shade.

Conclusion

Our community of designers and builders is becoming versed in parametric design and the related algorithms for exploration and optimisation regarding geometry. However, strategies of the same quality are missing regarding topological exploration, which operates beyond parametric design. This paper presented such a generative-design approach for quad-mesh patterns. A mesh grammar was defined to modify their strip structure thanks to two low-level rules to add and delete these strips. The application of these two rules was extended into a four-rule formal grammar to encode topological operations in a string and decode them to generate designs using an L-system approach. This L-system approach was applied to the topological exploration of quad-based patterns for various systems for surface structures, a cable-net and a gridshell, further showing the influence of pattern topology on the geometry and the performance of the design. This work opens the door for the next generation of topology optimisation algorithms to design structured surface patterns with full flexibility regarding density selection, shape design, performance objectives, and search and optimisation solvers like integer programming, genetic algorithms or machine learning.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

ORCID iD


R Oval https://orcid.org/0000-0002-9701-9853

References

1. Bendsøe M and Sigmund O. *Topology optimization: theory, methods, and applications*. Berlin, Heidelberg: Springer Science + Business Media, 2013.
2. He L, Gilbert M and Song X. A Python script for adaptive layout optimization of trusses. *Struct Multidiscipl Optim* 2019; 60: 835–847.
3. Borgart A. New challenges for the structural morphology group. *J Int Assoc Shell Spatial Struct* 2010; 51(3): 183–189.
4. Winslow P, Pellegrino S and Sharma SB. Multi-objective optimization of free-form grid structures. *Struct Multidiscipl Optim* 2010; 40(1-6): 257–269.
5. Schiftner A and Balzer J. Statics-sensitive layout of planar quadrilateral meshes. In: C Ceccato, L Hesselgren, M Pauly, H Pottmann and J Wallner (ed.) *Proceedings of the Advances in Architectural Geometry*, 2010, pp.221–236. Springer.
6. George Stiny and James Gips. Shape grammars and the generative specification of painting and sculpture. In: *Proceedings of the Congress International Federation for Information Processing*, 1971, pp. 1460–1465.
7. Stiny G and Mitchell WJ. The palladian grammar. *Environ Plann B Plann Des* 1978; 5(1): 5–18.
8. Stiny G. *Shape: talking about seeing and doing*. Cambridge, Massachusetts: MIT Press, 2006.
9. Mitchell WJ. Functional grammars: an introduction. In: *Reality and Virtual Reality: Association for Computer Aided Design in Architecture Conference Proceedings*, 1991, pp. 167–176. University of California at Los Angeles.
10. Baldock R, Shea K and Eley D. Evolving optimized braced steel frameworks for tall buildings using modified pattern search. *Comput Civ Eng* 2005; 2005: 1–12.
11. Baldock R. *Structural optimisation in building design practice: case-studies in topology optimisation of bracing systems*. PhD thesis, University of Cambridge (2007, accessed 26 October 2018).
12. Geyer P. Multidisciplinary grammars supporting design optimization of buildings. *Res Eng Des* 2008; 18(4): 197–216.
13. Mueller CT. *Computational exploration of the structural design space*. PhD thesis, Massachusetts Institute of Technology (2014, accessed 01 September 2018).
14. Shea K, Cagan J and Fenves SJ. A shape annealing approach to optimal truss design with dynamic grouping of members. *J Mech Des* 1997; 119(3): 388–394.
15. Shea K and Cagan J. The design of novel roof trusses with shape annealing: assessing the ability of a computational method in aiding structural designers with varying design intent. *Des Stud* 1999; 20(1): 3–23.
16. Shea K and Cagan J. Innovative dome design: applying geodesic patterns with shape annealing. *Artif Intell Eng Des Anal Manuf* 1997; 11(5): 379–394.
17. Chomsky N. Three models for the description of language. *IRE Trans Inf Theory* 1956; 2(3): 113–124.
18. Chomsky N. *Syntactic structures*. Berlin: De Gruyter Mouton, 1957.
19. Chomsky N. *Aspects of the theory of Syntax*. Cambridge, Massachusetts: MIT press, 2014. Vol. 11.
20. Lindenmayer A. Mathematical models for cellular interactions in development i. Filaments with one-sided inputs. *J Theor Biol* 1968; 18(3): 280–299.
21. Prusinkiewicz P and Lindenmayer A. *The algorithmic beauty of plants*. Berlin, Heidelberg: Springer Science + Business Media, 2012.
22. Kobayashi MH. On a biologically inspired topology optimization method. *Commun Nonlinear Sci Numer Simul* 2010; 15(3): 787–802.
23. Bielefeldt BR, Reich GW, Beran PS, et al. Development and validation of a genetic L-system programming framework for topology optimization of multifunctional structures. *Comput Struct* 2019; 218: 152–169.
24. Hartl DJ, Reich GW and Beran PS. Additive topological optimization of muscular-skeletal structures via genetic L-system programming. In: *24th AIAA/AHS Adaptive Structures Conference*, 2016, p. 1569.
25. Hugo-Tiago C and Kobayashi M. On a cellular division method for topology optimization. *Int J Numer Methods Eng* 2011; 88(11): 1175–1197.
26. Stanford B, Beran P and Kobayashi M. Simultaneous topology optimization of membrane wings and their compliant flapping mechanisms. *AIAA J* 2013; 51(6): 1431–1441.
27. Oval R. *Topology Finding of Patterns for Structural Design*. PhD thesis, Université Paris-Est, 2019.
28. Robin Oval. compas_singular: a python framework for topology finding of singularities in patterns (2017, accessed 01 October 2023).
29. Van Mele T and Liew A. Tomás Méndez Echenagucia, Matthias Rippmann, et al. compas: A framework for computational research in architecture and structures (2017, accessed 01 September 2018).
30. Campen M, Bommes D and Kobbelt L. Dual loops meshing: quality quad layouts on manifolds. *Assoc Comput Mach Trans -gr* 2012; 31(4): 110.
31. Campen M and Kobbelt L. Dual strip weaving: interactive design of quad layouts using elastica strips. *Assoc Comput Mach Trans -gr* 2014; 33(6): 183.
32. Akleman E, Chen J and Gross JL. Strip sculptures. In: *2010 Shape Modeling International Conference*, 2010, pp. 236–240. IEEE.
33. Akleman E, Ke S, Wu Y, et al. Construction with physical version of quad-edge data structures. *Comput Graph* 2016; 58: 172–183.

34. Hart P, Nilsson N and Raphael B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans Syst Sci Cybern* 1968; 4(2): 100–107.
35. Ronald S, Asenstorfer J and Vincent M. Representational redundancy in evolutionary algorithms. In: *Proceedings of 1995 IEEE International Conference on Evolutionary Computation*, 1995, vol. 2, pp. 631–636. IEEE.
36. Echenagucia TM and Block P. Acoustic optimization of funicular shells. In: *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures*, 2015.
37. Levenshtein VI. Binary codes capable of correcting deletions, insertions, and reversals. *Sov Phys Dokl* 1966; 10(8): 707–710.
38. Oval R, Mesnil R, Van Mele T, et al. Two-colour topology finding of quad-mesh patterns. *Comput Aided Des* 2021; 137: 103030.
39. Hamming RW. Error detecting and error correcting codes. *Bell Syst Tech J* 1950; 29(2): 147–160.
40. Robin O, Rippmann M, Mesnil R, et al. Feature-based topology finding of patterns for shell structures. *Autom Constr* 2019; 103: 185–201.
41. Conway J, Burgiel H and Goodman-Strauss C. *The symmetries of things*. Boca Ration, Florida: CRC Press, 2016.
42. Shepherd P and Pearson W. Topology optimisation of algorithmically generated space frames. In: *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures*, 2013.
43. Koronaki A, Shepherd P and Evernden M. Layout optimization of space frame structures. In: *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures*, 2017.
44. Williams C. The analytic and numerical definition of the geometry of the British Museum Great Court Roof. In: Burry M, Datta S, Dawson A, et al. (eds) *Mathematics & design*. Geelong, Victoria, Australia: Deakin University, 2001, pp.434–440.
45. Sischka J, Brown S, Handel E, et al. Die Überdachung des Great Court im British Museum in London. *Stahlbau* 2001; 70(7): 492–502.
46. Venuti F and Bruno L. Influence of in-plane and out-of-plane stiffness on the stability of free-edge gridshells: A parametric analysis. *Thin-Walled Struct* 2018; 131: 755–768.
47. Preisinger C. Linking structure and parametric geometry. *Archit Des* 2013; 83(2): 110–113.