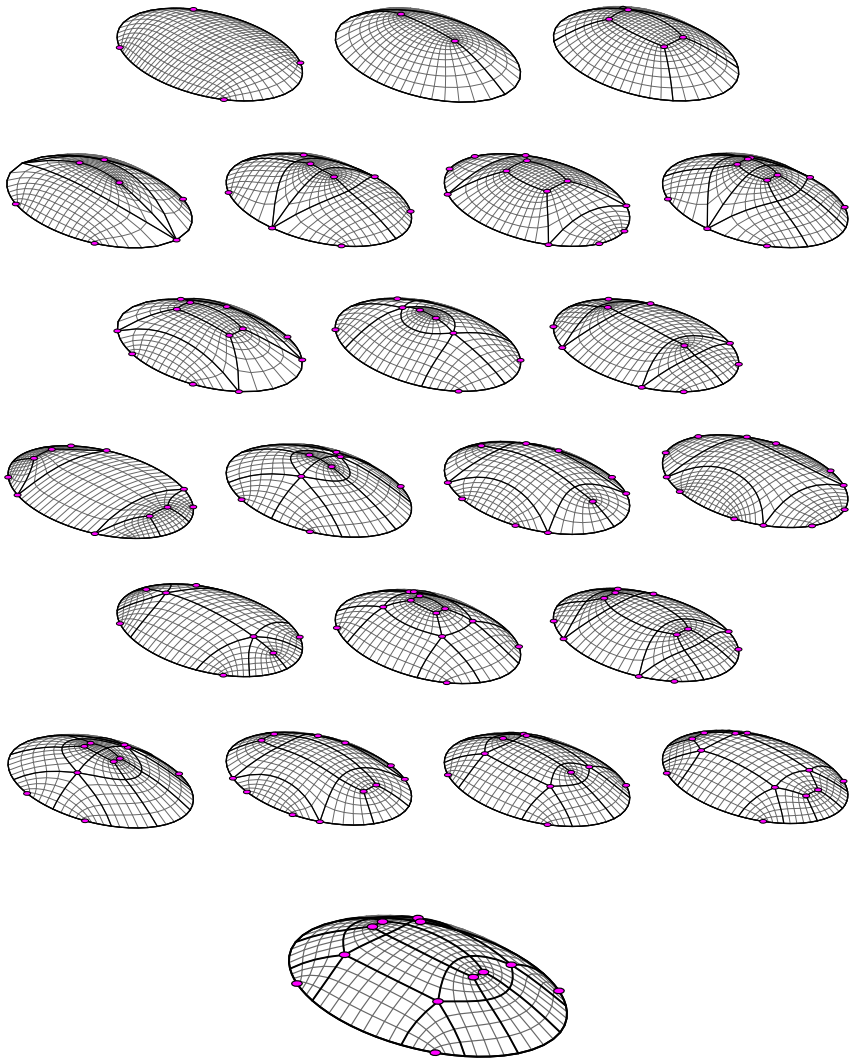
# Topology Finding of Patterns for Structural Design

Robin Oval



# Topology Finding of Patterns for Structural Design

Robin Oval

UNIVERSITÉ PARIS-EST

École doctorale Sciences, Ingénierie et Environnement

Thèse de doctorat

Spécialité Génie Civil

# Topology Finding of Patterns for Structural Design


présentée par

**Robin Oval**


Une collaboration École des Ponts ParisTech – ETH Zurich

Soutenue le 3 décembre 2019
devant le jury composé de:

| | | |
|---|---|---|
| **Rapporteurs** | Mark Pauly | EPF Lausanne |
| | Paul Shepherd | University of Bath |
| **Examinateurs** | Maurizio Brocato | ENSA Paris-Malaquais |
| | Bernard Maurin | Université de Montpellier |
| **Directeurs de thèse** | Olivier Baverel | École des Ponts ParisTech |
| | Philippe Block | ETH Zurich |
| **Co-encadrants** | Romain Mesnil | École des Ponts ParisTech |
| | Matthias Rippmann† | ETH Zurich |
| | Tom Van Mele | ETH Zurich |

*To Matthias*

# Acknowledgements

My PhD has been a nomadic journey between Paris and Zurich, which provided me the chance to meet so many people from whom I learned a lot, both on a personal and professional level.

First, I would like to thank l'École des Ponts and the Block Research Group who funded my PhD.

Thank you to my advisers in Paris. To Olivier Baverel, who saw me growing up from my first year at l'École des Ponts and who accepted my application as a PhD student and offered me the chance to embark in this great adventure with ETH Zurich. I always appreciated your trust, your passion and your curiosity. To Romain Mesnil, to whom I relate very much. Thanks for these years, for your support, starting even before the end of your PhD, and for all these interesting conversations. Also, thank you to Marie-Françoise Kaspi, Jean-François Caron and Karam Sab, who shield PhD students so we can work in the best possible conditions. A special thank to Cyril Douthe for the insightful discussions and teaching opportunities, and to Laurent Hauswirth for his introduction to discrete differential geometry.

Thank you to my advisers in Zurich. To Philippe Block, who got me into this beautiful adventure and helped me discover a different world. I am thankful for introducing me to your vision, sharing your enthusiasm and helping me improve in many ways. To Tom Van Mele, who I enjoyed working with. Thank you for the cohesion you bring to the group, your sharp expertise and your insight in framework development. To Matthias Rippmann, who supported me all along the way and who I miss very much. I am for ever grateful for everything you brought me, from the Venice Biennale to discussions about my future. Also, thank you to Noelle Paulson, who keeps the group running and who helped a lot in so many ways during my time

in Zurich. A special thank to Juney Lee, who introduced me to the art of graphic making, and to Mariana Popescu, who reviewed the graphics of this thesis.

I want to thank the members of the jury for the interesting discussions during the examination, the examiners Benard Maurin and Maurizio Brocato, and particularly the readers Paul Shepherd and Mark Pauly, for accepting to review my thesis.

Thank you very much to all my colleagues that constituted the two groups where I thrived so much. Without you, these three years would not have been the same. I will be missing these breaks, matches, drinks, grills, parties and travels. What I experienced with these two families goes beyond the research group and will last long afterwards towards new adventures. Particularly, thank you to Charlotte Allemand, Shajay Bhooshan, Marie Bonnet, Romain Boulaud, Cristian Calvo Barentin, Paul Carneau, Gregoire Corre, Julien Cravero, Alessandro Dell'Endice, Romain Duballet, Nicolas Ducoulombier, Leo Demont, Lionel du Peloux, Baptiste Durand, Lluis Enrique, Chadi El Boustani, Sandie Kate Fenton, Tristan Gobin, Philippe Hannequart, Gene Kao, Olivier Knepper, Katya Kuzmenko, Antonino Iannuzzo, Tran Le Hung, Nicolas Leduc, Juney Lee, Andrew Liew, Vianney Loing, David Lopez Lopez, Ricardo Maia Avelino, Sebastien Maitenaz, Koliann Mam, Pierre Margerit, Tomas Mendez, Nicolas Montagne, Mahan Motamedi, Rafael Pastrana, Mariana Popescu, Emma Radaelli, Francesco Ranaudo, Mark Tam, Xavier Tellier, Maryanne Wachter, Una Wang, and everyone else!

I am thankful to the countless number of people of the academic and practice community I had the chance to meet, who share my passion for structural design and had a positive impact on me, in various manners, many of which I consider as friends today.

Thank you to all my friends, in Paris, Zurich and elsewhere, whose friendship easily resisted my nomadic situation and strengthened me.

Thank you to my family for their love, particularly my sister and my parents. As always, you offered me the best conditions to pursue this adventure trouble-less, as my *repère* during these wandering years.

# Résumé

Les structures de type coque permettent de couvrir de larges espaces de façon élégante et efficace. Les maillages de quadrangles sont des motifs naturels pour représenter ces objets surfaciques, qui peuvent aussi servir à appliquer d'autres motifs. Les motifs pour ces coques, voûtes, résilles et autres réseaux peuvent representer une materialisation de la structure, un équilibre des efforts ou une carte de la surface. La topologie de ces motifs contraint qualitativement et quantitativement la liberté de modélisation pour l'exploration geométrique. À moins de réaliser une exploration topologique.

La conception paramétrique supportant l'exploration et l'optimisation de la geométrie des structures se répand au sein de la communauté de concepteurs et bâtisseurs. Malheureusement, la conception topologique est à la traîne, malgré des stratégies orientées vers l'optimisation pour des objectifs de conception spécifiques. Des stratégies, algorithmes et outils pour l'exploration topologique sont nécessaires pour s'attaquer aux multiples objectifs de l'architecture, de l'ingenierie et de la construction pour la conception de structures à l'échelle architecturale. La tâche de la conception des structures est riche et complexe, nécessitant des algorithmes interactifs orientés vers la co-conception entre l'humain et la machine. Une telle approche est complémentaire et renforcée par les méthodes éxistantes pour l'exploration geométrique et l'optimisation topologique.

Le travail présenté introduit la *recherche topologique* pour l'exploration efficace de l'espace de conception topologique. Cette thèse repose sur trois stratégies pour la recherche topologique de singularités dans des motifs de maillages de quadrangles, présentées de l'approche la plus haut niveau à celle la plus bas niveau.

*L'exploration encodée par des objets geométriques* repose sur la décomposition d'une surface par des patches à quatre côtés à l'aide de son squelette topologique, incluant des points et courbes attributs. Ces paramètres geométriques peuvent provenir d'heuristiques de conception à intégrer dans la conception, liées au système statique ou la courbure de la coque, par exemple.

*L'exploration encodée par des graphes de connectivité* repose sur les bandes topologiques dans les maillages de quadrangles. Une grammaire de règles permet l'exploration de cette structure de bandes pour explorer l'espace de conception. Un algorithme de recherche informé par la similarité trouve des structures avec différents degrés de similarité topologique. Des structures optimisées pour des objectifs uniques peuvent informer ce processus de génération pour obtenir des structures offrant différents compromis entre plusieurs objectifs. Un algorithme de recherche de deux-colorabilité trouve des structures qui remplissent un critère de maillage deux-colorable. Cette propriété topologique permet une partition des éléments du motif, que nécessite de nombreux systèmes structurels.

*L'exploration encodée par des chaînes de caractères* repose sur la traduction des règles de grammaire en des operations alphabétiques, changeant l'encodage d'un maillage phénotypique à une chaîne de caractères génotypique. Des modifications, ou mutations, de la chaîne transforment le génotype et change le phénotype de la structure. L'encodage en une chaîne, ou un vecteur, ouvre la voie à l'utilisation d'algorithmes d'exploration et d'optimisation, comme l'optimisation linéaire, les algorithmes génétiques ou l'apprentissage automatique.

**Mots-clefs**: conception structurale, exploration topologique, motifs, maillages de quadrangles, singularités, recherche de topologie, coques, résilles.

# Abstract

Shell-like structures allow to elegantly and efficiently span large areas. Quad meshes are natural patterns to represent these surface objects, which can also serve for mapping other patterns. Patterns for these shells, vaults, grid-shells or nets can represent the materialised structure, the force equilibrium or the surface map. The topology of these patterns constrains their qualitative and quantitative modelling freedom for geometrical exploration. Unless topological exploration is enabled.

Parametric design supporting exploration and optimisation of the geometry of structures is spreading across the community of designers and builders. Unfortunately, topological design is lagging, despite some optimisation-oriented strategies for specific design objectives. Strategies, algorithms and tools for topological exploration are necessary to tackle the multiple objectives in architecture, engineering and construction for the design of structures at the architectural scale. The task of structural design is rich and complex, calling for interactive algorithms oriented towards co-design between the human and the machine. Such an approach is complementary and empowered with existing methods for geometrical exploration and topology optimisation.

The present work introduces *topology finding* for efficient search across the topological design space. This thesis builds on three strategies for topology finding of singularities in quad-mesh patterns, presented from the most high-level to the most low-level approach.

*Geometry-coded exploration* relies on a skeleton-based quad decomposition of a surface including point and curve features. These geometrical parameters can stem from design heuristics to integrate into the design, related to the statics system or the curvature of the shell, for instance.

*Graph-coded exploration* relies on the topological strips in quad meshes. A grammar of rules allows exploration of this strip structure to search the design space. A similarity-informed search algorithm finds design with different degrees of topological similarity. Designs optimised for single objectives can inform this generation process to obtain designs offering different trade-offs between multiple objectives. A two-colour search algorithm finds designs that fulfil a two-colouring requirement of two-colouring. This topological property allows a partition of the pattern elements that many structural systems necessitate.

*String-coded exploration* relies on the translation of the grammar rules into alphabetical operations, shifting encoding from a phenotype mesh to a genotype string. Modifications, or mutations, of the string transform the genotype and change the phenotype of the design. String or vector encoding opens for the use of search and optimisation algorithms, like linear programming, genetic algorithms or machine learning.

**Keywords**: structural design, topological exploration, patterns, quad meshes, singularities, topology finding, shells, gridshells.

# Contents

# Part I

# Introduction

# Chapter 1

# Context

*'What matters is pattern and its connectivity.'*
Cecil Balmond
in Informal

*'Topology is the science of fundamental pattern and structural relationships of event constellations.'*
Richard Buckminster Fuller
in Operating Manual for Spaceship Earth

Architecture, structure, connectivity, topology and pattern are synonyms to describe the arrangement between elements. Patterns are ubiquitous in design, including architectural, structural design and appear in the design of different structural systems. For shell-like structures such as shells, vaults or gridshells, represented as a surface or a mesh, the pattern can represent the arrangement between beams, blocks or forces.

This chapter presents the context for structural design of patterns. Section 1.1 presents computational structural design. Section 1.2 highlights how patterns come into play in structural design. Section 1.3 lists the different design aspects of patterns. Section 1.4 sets the focus of this thesis on the structural design of quad meshes and their singularities.

## 1.1 Computational structural design

Structural design lies at the interface of architecture and engineering, between architectural design and structural analysis. Designing a structure re-

quires theoretical knowledge on geometry and mechanics, applied knowledge on fabrication technologies and construction codes, and a sense of aesthetics. Structures as a whole are not mass-produced, contrary to the automotive industry, for instance. A structure stems from a design process with specific design intents, engineering requirements and construction constraints. This rich and intense process is highly non-linear to achieve a design that meets all objectives and constraints. Hand drawing and physical modelling are insightful processes for the design of structures but were replaced by digital modelling to efficiently perform an iterative search. Computer-aided design empowers designers to improve the design process, save time, energy and money, or even make a project feasible. Indeed, computer-aided design allows rapidly exploring and evaluating a large number of designs to search for the most suitable ones, before physical prototyping and testing.

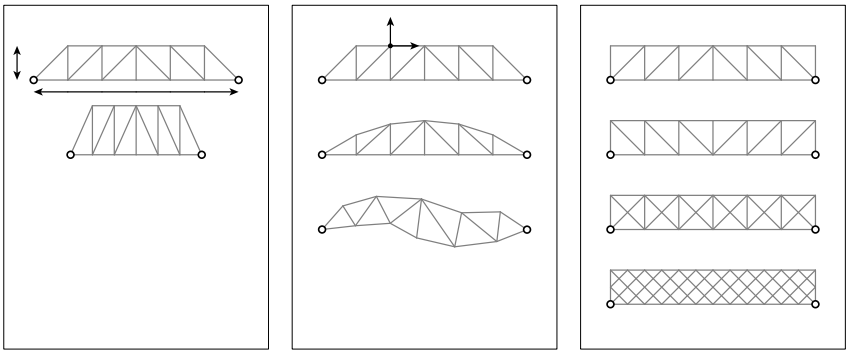
### 1.1.1 Design exploration

Computational models represent the geometry and the connectivity of the structure, on which the designer can run mechanical and other analyses. To benefit from such models beyond numerical analysis, building parametric models enables efficient and interactive design exploration via the modification of the parameters that describe the model.

#### 1.1.1.1 Parametric design

Setting a parametric model allows interactive design through the modification of values. Grasshopper3D for Rhino3D or Dynamo Studio by Autodesk provide such parametric environments, for instance. Regarding a truss, whose pattern is the beam layout, two simple parameters can be the span and the height, as in Figure 1.1a. Thereby, the designer can explore the geometry and the density of the truss. The models are re-generated when changing these values instead of re-drawing the model. The parameters define the flexibility of the model and therefore, the boundaries of the design space. A flexible parametric model of a truss can consider each coordinate of each node as a parameter to allow free-form shapes and not only rectangular ones, as in Figure 1.1b. However, the higher the number of parameters, the higher the dimension of the corresponding design space. And therefore, the more complex the exploration for the search for an optimal design. How much does adding a specific parameter enrich the design space? The definition of a parametric model is a trade-off between richness and complexity and must be carefully determined.

Parametric design enables geometric exploration but not a full exploration of a computational model. The geometry-related parameters are dependent on the connectivity of the structure that defines the organisation between its elements. An extended approach is required to embrace full exploration, which includes connectivity design of structures.



(a) Limited parametric design

(b) Flexible parametric design

(c) Trans-parametric design

Figure 1.1 – Parametric and trans-parametric design spaces for a truss.

#### 1.1.1.2 Trans-parametric design

Beyond parametric design, trans-parametric design uses discrete parameters, as opposed to continuous-valued parameters. Trans-parametric explores the connectivity of a design and traverses different parametric spaces, also called meta-parametric design by Harding [Harding et al., 2012, Harding and Shepherd, 2017]. These discrete parameters can define the layout of the beams of a truss, as in Figure 1.1c with different bracing configurations. The combinatorial nature of these parameters does not allow to directly enumerate them to define the dimension of the design space. Parametric design explores geometrical design spaces. On the contrary, trans-parametric design tackles topological design spaces, encapsulating multiple geometrical spaces. The parameters that perform the topological modifications must be carefully determined as well, as it defines the topological space to explore.

Density parameters, which can control the number of bracing elements in a truss, for instance, change the connectivity quantitatively but not qualitatively. The exploration of the density parameters is not combinatorial.

Although density parameters are not continuous but discrete, the finite set of density parameters enrich the parametric model.

The design space derives from the selected continuous parameters and discrete parameters. The higher the number of these, the larger the design space. However, exploring larger design spaces is more complex without ensuring to find more suitable designs. Interactive methods must support and guide the designer.

### 1.1.2 Design interactivity

Defining parametric and trans-parametric models can result in the computation of rich design spaces that exceed in size what a designer can comprehend. Nevertheless, the interaction between the human and the machine using specific search algorithms allows the designer to appreciate the design space and find the most relevant designs. These designs should respect the project constraints and optimise the design metrics.

#### 1.1.2.1 Constrained design

Not all and not even most designs in the design space are relevant. Instead of finding optimal designs in the general design space, constraining exploration to the relevant design subspaces allows finding their optimal designs more efficiently. Figure 1.2a illustrates this opposition between the general design space in white and the specific subspace in red.

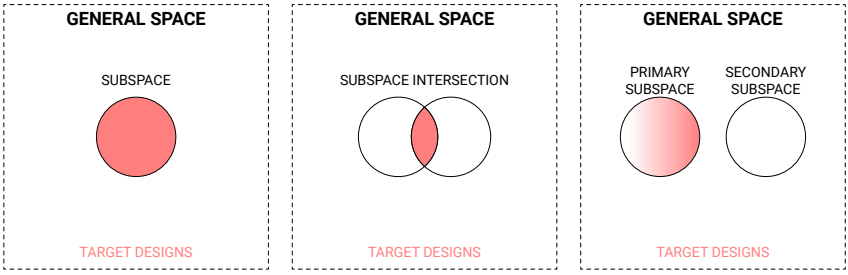
A first example is constraining exploration to compression-only networks in equilibrium with a given load case [Block and Ochsendorf, 2007] and then finding the one with the minimum load path that relates to the structure of minimum volume [Liew et al., 2019].

A second example is constraining exploration to structures with planar faces for flat panel fabrication [Mesnil et al., 2017e] and then performing structural optimisation [Mesnil et al., 2018b].

This constrained exploration approach drives the designer towards subspaces that contain the most relevant designs. These constraints can be of architectural, structural or constructive nature.

A structure often has to comply with multiple constraints. Constrained-exploration strategies can be combined to search for a potential intersection between the different subspaces to respect these multiple constraints. In Figure 1.2b, the intersection in red of the two subspaces further reduces the size of the space to explore. However, this intersection between subspaces may not exist. Nevertheless, a hierarchy usually exists between constraints. For a

steel and glass gridshell, a designer may favour affordable planar panels over compression-only equilibrium, although wishing to be close to such an equilibrium. For a stone vault, a design may on the contrary favour compression-only equilibrium over voussoir planarity, although wishing to have almost planar faces. Based on the constraint hierarchy, exploration can be constrained to the designs in the primary subspaces. Among these designs, exploration should favour the ones close to the secondary subspaces. In Figure 1.2c, not all designs of the primary subspace have the same relevance — the closer to the secondary subspace, the more relevant, as visualised with a white-to-red gradient.




(a) Unique subspace (b) Intersecting subspaces (c) Disjoint subspaces

Figure 1.2 – Constraining exploration of the general design space based on one or several subspaces that correspond to primary or secondary constraints to meet. The most relevant design space areas are highlighted in red.

#### 1.1.2.2 Informed design

Even constrained to a subspace, finding an optimal or heuristic design is challenging. Performance can be assessed during the search through the design space using mechanical and environmental analysis, for instance. Searching for efficient designs can be performed in an enumerative manner. Alternatively, the performance of tested designs can inform the generation of more efficient ones, following the judgement of the designer or using optimisation algorithms.

These computational approaches can apply to the design of patterns, which find many applications in the realm of structural design, including the design of shell-like structures.

## 1.2 Patterns in structural design

A pattern can represent different structural elements, like beams, walls or slabs. The focus of this thesis lies on the design of large-span discrete or continuous shell-like structures – or surface structures – like shells, vaults or gridshells. Shell-like structures can span large areas thanks to their double curvature, which provides geometrical stiffness. For their design, these large-scale structures become discrete patterns of smaller elements. The nature of these patterns can be of different types with different purposes.

### 1.2.1 Materialising elements

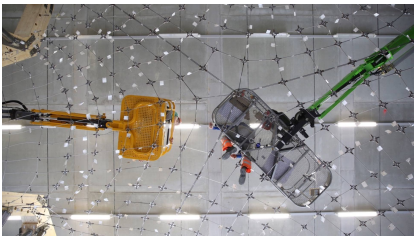
Patterns can be used to materialise the elements of the load-bearing and cladding system to fabricate and assemble. The elements of the pattern can represent nodes, beams or ribs, panels or voussoirs. Figure 1.3 features some of these patterns: a set of GFRP composite beams to form an elastic gridshell from a flat grid layout (Figure 1.3a); a tessellation of limestone voussoirs to form a compression-only stone vault (Figure 1.3b); a steel cable net to serve as flexible formwork for a concrete shell (Figure 1.3c); an assembly of timber beams to form a nexorade-like gridshell braced with panels (Figure 1.3d). The design of these material patterns relates to structural performance, fabrication, assembly and construction requirements, which relate to ecological and economic costs. The material pattern can represent the structure or the envelope alone if the two are decoupled [Mesnil, 2018].

### 1.2.2 Modelling equilibria

Patterns can be used to model the equilibrium between a set of forces in the structure, such as thrust networks [Block and Ochsendorf, 2007], load paths [Liew et al., 2018] or struts-and-ties [Schlaich and Schafer, 1991]. These force patterns can be used to find structures in pure compression or pure tension, for instance. They can also be used to confirm or infirm the structural safety of a design following simplified hypotheses, thanks to the lower-bound theorem in yield design [Heyman, 1997, Schlaich and Schafer, 1991]. A force pattern can become a material pattern like cable-net elements, which directly carry the forces. Alternatively, the force pattern can inform the design of a material pattern like masonry-vault voussoirs, whose normals should be as orthogonal as possible to the compressive forces to prevent sliding failure vaults [Rippmann and Block, 2018].

(a) Beam grid of the Ephemeral Cathedral in Créteil, France
[Du Peloux et al., 2015]
(Photo credits: thinkshell.fr)

(b) Voussoir tessellation of the Armadillo Vault in Venice, Italy
[Rippmann et al., 2016]
(Photo credits: Iwan Baan)



(c) Cable layout of the prototype for the NEST HiLo roof in Dübendorf, Switzerland
[Méndez Echenagucia et al., 2018]
(Photo credits: Naida Iljazovic)

(d) Beam network of a shell-nexorade hybrid at École des Ponts, Champs-sur-Marne, France
[Mesnil et al., 2018a]
(Photo credits: Romain Mesnil)

Figure 1.3 – Examples of material patterns of shell-like systems like gridhsells, vaults, nets and nexorades.

### 1.2.3 Parameterising surfaces

Patterns can apply to the parameterisation of surfaces. The pattern forms a digital discretisation of a continuous shell. They do not represent element or force distributions but provide a map of the surface. Parameterisation patterns can be used as control mesh for surfaces of NURBS or BREP [Pottmann et al., 2007a] to construct an underlying smooth surface. Parameterisation patterns can apply to form finding [Mesnil et al., 2018b], isogeometric analysis [Längst et al., 2017], continuous topology optimisation [Bendsøe and Sigmund, 2013] or finite element analysis. Moreover, folds [Mesnil et al., 2017a], corrugations [Norman et al., 2009] or perforations

[Schlaich, 2018] can be integrated. These parameterisation patterns allow storing any discrete data on the surface, like vector fields or cross fields, a natural manner of storing information in digital processing.

Patterns in structural design can be differentiated whether they relate to material, equilibrium or data. Nevertheless, the same aspects must be taken into account when designing these patterns.

## 1.3 Design aspects of patterns

The two main design aspects of a pattern are its geometry and its topology.

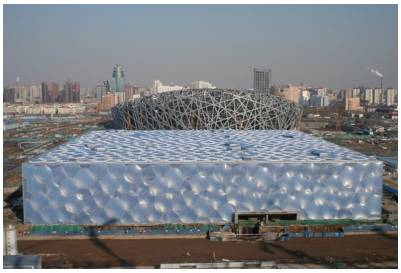
### 1.3.1 Geometrical aspects

The geometry of a pattern consists of the position of its vertices, the three coordinates x, y and z in the 3D space. A pattern with $n$ vertices has thus $3n$ geometrical degrees of freedom. These degrees of freedom are continuous parameters enabling the use of continuous algorithms for design and optimisation [Hojjat et al., 2014]. Grouping these parameters can bring coherence and constraints while reducing the size of the design space to search. Either implicitly using data analysis or explicitly using geometrical objects like curves or surfaces on which to constrain the vertices. Geometrical design methods aim at computing these parameters to fulfil specific goals. Such approaches usually assume a fixed topology.

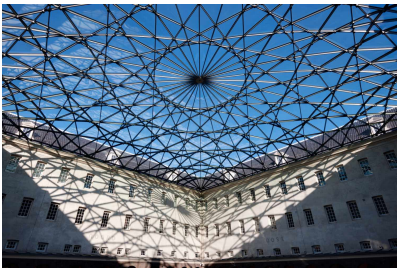
### 1.3.2 Topological aspects

In this thesis, the topology of a pattern refers to the connectivity between its vertices. The topology is a rich combinatorial problem controlled by a *flexible* number of *discrete* parameters. On the contrary, geometrical parameters consist of a fixed number of continuous parameters. The topology is embedded in the geometry. Indeed, two patterns with the same geometry entail that they have the same topology. However, two patterns with the same topology do not necessarily have the same geometry. Two patterns have the same topology if they can have the same geometry using a continuous, reversible geometrical transformation, known as *homotopy*. Such a transformation does not tear the shape or break the pattern. A square and a rectangle do not have the same geometry but have the same topology, whereas a disc and a sphere do not have the same geometry nor the same topology.

The design of the topology of a pattern can be decomposed into sub-problems to understand its regularity and irregularity.



(a) The frame network of the stadium and the Voronoi grid of the pool of the 2008 Beijing Olympic Games (Source: structurae.info)



(b) Polytope pattern of the courtyard roof of the Dutch Maritime Museum in Amsterdam, The Netherlands (Source: wbarchitectures.be)

(c) Timber gridshell with a semi-regular, Kagome pattern of the Centre Pompidou in Metz, France (Source: damienguillou.fr)

(d) Triangulated pattern of the geodesic dome of the Zeiss-Planetarium in Jena, Germany (Source: zendome.de)

Figure 1.4 – Non-structured patterns, non-regular patterns and regular patterns with some irregularities in structural design.

#### 1.3.2.1 Tessellations

The field of tessellations is a rich world that describes the fundamental module in a pattern [Grünbaum and Shephard, 1987]. Tessellations have logic, and therefore, are structured, characterised with high regularity. Tessellations are opposed to unstructured patterns where the types of and adjacency between elements are irregular. The applications of unstructured patterns

in structural design are rare. Figure 1.4 presents some examples with the Voronoi pattern and the network of curved beams of the Olympic swimming pool and stadium in Beijing, China, in Figure 1.4a, the Dutch Maritime Museum in Amsterdam, The Netherlands, in Figure 1.4b.

The logic of the tessellation of a pattern describes its repeated module. Regular tessellations are the ones where each polygon is identical. There are only three for the plane: tessellations of regular quads, triangles and hexagons, as shown in Figure 1.5. In each of these tessellations, faces have the same number of vertices, edges have the same length and vertices have the same number of neighbours.

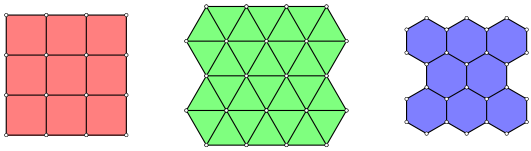


Figure 1.5 – The three regular tessellations of the plane.

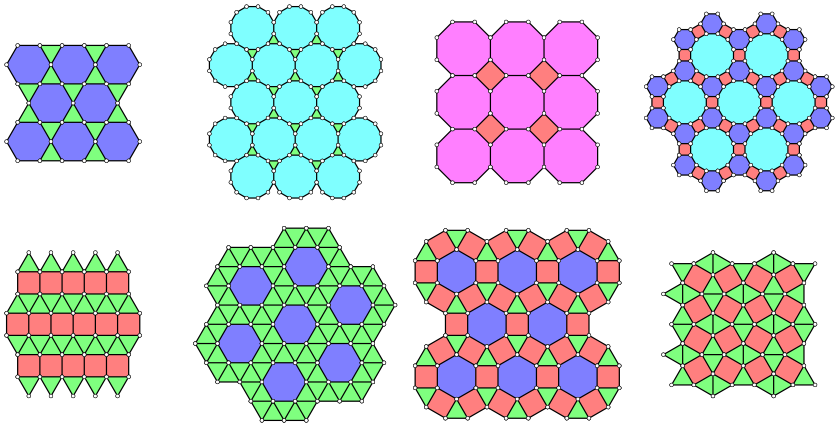


Figure 1.6 – The eight semi-regular tessellations of the plane. The colours represent the polygons with the same number of edges.

Semi-regular tessellations resort to two or more regular polygons. There are eight semi-regular tessellations for the plane, combining quads, triangles and hexagons, as well as octagons and dodecagons, as shown in Figure 1.6. Each vertex is adjacent to the same sequence of polygons. The regular and

semi-regular tessellations are different for the 3D space or non-Euclidean spaces like the sphere. The polygons of these tessellations are distorted on curved surfaces, with edges of different lengths. Mapping from the plane to the surface entails a loss in geometrical regularity. Nevertheless, they have the same topological regularity. The timber gridshell of the Centre Pompidou in Metz, France, shown in Figure 1.4c or the Yas Mall in Abu Dhabi, the United Arab Emirates, are based on such semi-regular tessellations consisting of hexagons and triangles, and quads and triangles, respectively.

The geodesic dome in Jena, Germany in Figure 1.4d is a particular example. The faces of the beam pattern are triangles with the same geometry. They can map the curved surface of a sphere thanks to a few irregular nodes that are adjacent to five faces instead of six, locally breaking the regularity of the tessellation. This specific organisation stem from a discretised icosahedron mapped to the sphere.

In construction, regularity in both geometry and topology matters [Mesnil et al., 2017b]. Different patterns can provide different properties. For instance, Kagome patterns have the same node complexity with shorter beams compared to quad meshes, offering interesting properties regarding fabrication and buckling [Mesnil et al., 2017b]. Nevertheless, the lack of modelling and design approach that tackles a large variety of tessellations is an obstacle for their exploration and implementation in structural design. The application of non-fully regular patterns in structural design as in Figures 1.4b and 1.4c are rare and regular patterns are preferred as in Figure 1.4d. The lack of a general topological design approach with tools that encapsulate the larger family of tessellations and structured patterns can explain their rarity. Nevertheless, some work in this direction resort to tessellation operators for the design of space frames [Koronaki et al., 2017].

#### 1.3.2.2 Density

The density of a pattern relates to the global number of elements. For a regular quad tessellation, the number of elements in each of the two grid directions characterises the density. The density parameters is a finite set of discrete values to explore when designing a pattern, as opposed to geometry-related continuous parameters.

#### 1.3.2.3 Singularities

Patterns and tessellations are synonyms of regularity. However, singular vertex or face elements can be included, locally breaking the pattern reg-

(a) Waffle slab without singularities (source: HOLEDECK)

(b) Ribbed slab with singularities (source: Pier Luigi Nervi)

Figure 1.7 – Slab systems with different geometrical and topological regularities due to singularities in the patterns.

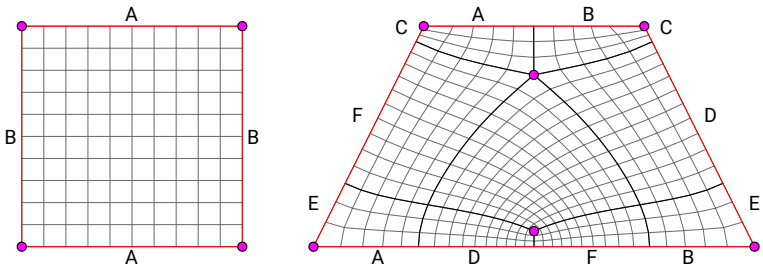
ularity. The tessellation of the geodesic dome in Figure 1.4d consists of regular triangles with regular nodes connecting six beams, except for some irregular nodes connecting five beams. These singular nodes are necessary to map the sphere.

In a quad pattern, regular faces and vertices have four edges, whereas irregular or singular vertices have a valency – a number of edges – different from four. These local irregularities have a global influence as the flow and the relation between elements are modified. In structural design, it relates to modifying the layout of structural and cladding elements in material patterns, the force layout in force patterns or the edge layout in parameterisation patterns. Figure 1.7 opposes two slab systems. The waffle slab in Figure 1.7a is highly regular to ease fabrication and assembly. Although, this pattern does not take the static system into account and reinforcement is necessary to avoid the columns to punch through due to shear forces. The ribbed slab of Pier Luigi Nervi in Figure 1.7b includes singularities to follow statics considerations and steer the beams towards the columns. This structural efficiency comes at a cost on pattern complexity, regarding topology and geometry, which impacts the fabrication process. Developments in digital fabrication with technologies like 3D printing provide new flexibility in the fabrication of complex structures. These technologies apply to a varied range of scales and set of materials, like the sand 3D-printed floor prototype of the Block Research Group [Rippmann et al., 2018]. The general work towards mass-customisation ease the integration of singular elements in the fabrication process. The design of the geometry and the topology is a dialogue with the many project-specific design aspects. These

influential aspects include aesthetics, statics, fabrication, assembly, as well as sustainability and cost.

Changing singularities in a quad mesh induces a different flow and relation between elements, as shown in Figure 1.8. In this thesis, the singularities in quad meshes are highlighted in pink, and the successive edges stemming from them are highlighted in black. The singularities in the quad meshes are vertices with an irregular valency, meaning different from four edges off the boundary and three on the boundary. From the quad mesh with four singularities in Figure 1.8a to the mesh with two additional ones in Figure 1.8b, the quantitative number of elements and their qualitative relation differ. The independent density parameters A to F along the boundary indicate these differences.

Takayama *et al.* [Takayama et al., 2014] present a set of quad-mesh patterns with different singularities for N-sided patches with a prescribed number of subdivisions on each side. Matching the prescribed number of subdivisions and the length of the sides allows to obtain quad meshes with a regular distribution of the boundary edges thanks to the singularities as in Figure 1.8.




(a) Two density parameters

(b) Six density parameters

Figure 1.8 – Different singularities in a quad mesh induce qualitatively different relations between its elements. The singularities are highlighted in pink and the successive edges stemming from them are highlighted in black.

These topological singularities occur also in the meshes used for Finite Element Analysis. However, they are different from the field singularities, which occur in the stress field, for instance, where the values do not converge and therefore the field is not defined.

Among the families of patterns, quad meshes are of particular interest

in structural design of shell-like structures thanks to their flexibility to map surfaces and represent different structural systems. Moreover, their singularities appear as a particularly influential design aspect.

## 1.4 Structural design of quad-mesh patterns

Quad-mesh patterns find a large set of applications in structural design of shell-like structures. Mechanical, fabrication and construction aspects must inform the design of their geometry and topology.

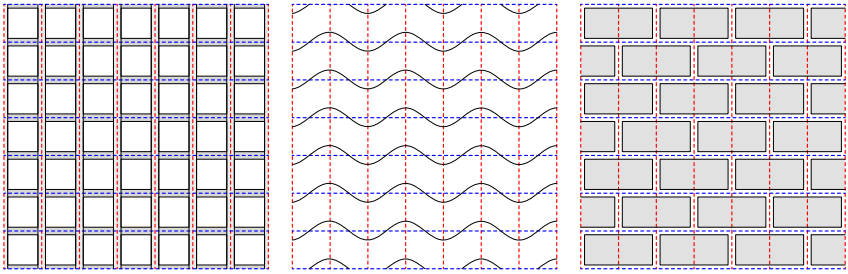
### 1.4.1 Quad-mesh patterns

Surfaces can be discretised using any pattern. However, surfaces are two-dimensional objects, naturally described with two parameters. Therefore, quad meshes and their bidirectional nature correspond to this natural description of surfaces.

Moreover, integration of cross fields on a shell yield quad meshes with correspondence of the two parameterisation directions. Principal curvature directions and principal stress directions are such cross fields, which architects and engineers are familiar with and use to interpret to understand their design. For instance, Nervi's floors [Nervi, 1965] are elegant and efficient examples of interpretation of principal stress directions to form a ribbed slab, as shown in Figure 1.7.

Furthermore, quad meshes can serve as an underlying representation of many structural systems. The structure stems either directly or indirectly from the quad-mesh elements, as illustrated in Figure 1.9 with continuous beams, corrugation patterns and staggered bricks.

Quad-mesh patterns offer other design options with different pros and cons to triangular-mesh patterns. As expressed by Schlaich and Schober [Schlaich and Schober, 2005, Schober, 2015], quad meshes for steel and glass gridshells are more transparent and have simpler nodes than a triangulated mesh. However, quad-mesh structures rely on member and node bending stiffness, especially against in-plane shear, whereas triangular-mesh structures rely on member axial stiffness, thanks to the stability of triangles. Moreover, a triangle is by definition planar, on the contrary to a quad, which can be curved. Stiff nodes or braced panels compensate for the lack of structural stability. Pre-or post-rationalisation strategies tackle the problem of panel curvature [Schober, 2015, Mesnil et al., 2017a, Liu et al., 2006].



(a) Axis of beams (b) Map of corrugations (c) Frame of bricks

Figure 1.9 – Quad meshes in dashed red and blue lines representing different structural systems in black and grey.

### 1.4.2 Geometrical design

Geometrical exploration aims at designing efficient and feasible structures by modifying the position of the mesh vertices to meet some requirements and optimise objectives. Form-finding methods can be force-based to guarantee membrane equilibrium, like Force Density Method [Schek, 1974], Dynamic Relaxation [Barnes, 1999], Thrust Network Analysis [Block and Ochsendorf, 2007] or Update Reference Strategy [Bletzinger and Ramm, 1999], or technology-based to ease fabrication and construction, like Scale-Trans Surfaces [Glymph et al., 2004] or Marionette Meshes [Mesnil et al., 2017e]. These strategies constrain design exploration to the relevant subspaces. These methods can be combined with optimisation strategies to further search these subspaces [Bletzinger and Ramm, 1993, Mesnil et al., 2018b]. Optimisation algorithms for architectural geometry allow rationalising forms based on multiple objectives, like ShapeOp [Deuss et al., 2015], based on projective dynamics [Bouaziz et al., 2014]. Nevertheless, the earlier the integration of design constraints, the easier the design process, which motivates the use of pre-rationalisation versus post-rationalisation [Mesnil et al., 2017a].

Yet, topology matters for exploration. Indeed, the topology of the pattern sets the bounds of the available geometrical design space within the more general design space. This available geometrical design space represents all the possible geometries for a given topology.

Figure 1.10 – The central singularity of the concrete shell of the CNIT in La Défense, France make the design feasible by orientating the stiffening corrugation pattern towards the supports (source: defense-92.fr).

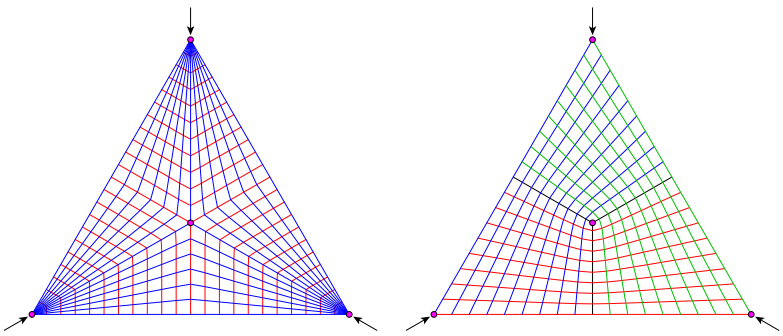
### 1.4.3 Topological design

Exploring different configurations of singularities modifies the flow of edges. These edges can represent the orientation of elements, the flows of forces or the directions of parameterisations. The definition of the geometry and the density depend on the singularities. As the most upstream design aspect, singularities define all the other downstream degrees of freedom and are therefore the most influential one. A poor topology will not yield efficient or even feasible geometrical designs. For instance, the CNIT shown in Figure 1.10 and designed by Nicolas Esquillan is a concrete shell that spans the concrete-shell world record of 218 meters thanks to stiffening corrugations [Motro and Maurin, 2011]. The central singularity directs the pattern between the supports. Thereof, the corrugations stiffen the shell in the proper direction, making such an achievement possible.

Many design objectives relate to geometry, like panel planarity or compression-only equilibrium. However, some topological properties, like maximum node valency, play an important role in structural design, independently from geometry. For instance, the CNIT can not be corrugated in both directions; otherwise, a resulting eggbox pattern would weaken both directions instead of reinforcing one. The condition for non-self-crossing corrugations depends on the singularities in the quad mesh and is not systematically guaranteed.

Figure 1.11 opposes the pattern of the CNIT in Figure 1.11a to another one that is not efficient nor feasible in Figure 1.11b. The pattern topology in Figure 1.11b does not allow to orientate the elements directly to the support. Moreover, it creates three groups of non-overlapping elements due to the central three-valent singularity. One group can not be chosen to form stiffening corrugations all over the surface. On the contrary, the pattern topology in Figure 1.11a integrates these requirements.

This example illustrates that the pattern, its topology and the related geometrical space may not contain efficient or even feasible designs if poorly designed.



(a) Efficient and feasible pattern topology

(b) Inefficient and unfeasible pattern topology

Figure 1.11 – Two different quad-mesh pattern topologies for a structure supported on three corners only, represent by black arrows. Efficiency comes from directing the continuous coloured elements towards the supports. Feasibility comes from the existence of one coloured group of non-overlapping continuous elements to introduce stiffening corrugations all over the surface.

## 1.5 Summary

Quad meshes can be used to model different objects for the design of structures. Significant work in architectural geometry, form finding and shape optimisation enable geometrical design of these patterns. Unfortunately, similar approaches are missing regarding the topology of these patterns, which has a strong influence on all design aspects. Exploring the topology of quad-mesh patterns necessitate a trans-parametric design approach, beyond parametric design. Among the different topological aspects, the singularities in the pattern have the most influence.

The next chapter presents the current strategies related to topological designs of patterns in general, with their benefits and their limitations, and complementary background on the topology of shapes and meshes.

# Chapter 2

# Literature review

The design of singularities in quad meshes is a critical aspect in structural design of patterns for shell-like structures. Additional background to architectural geometry and form finding is necessary to grasp the specific challenges of topological design of such objects.

Section 2.1 clarifies the pros and cons of current structural design approaches for the topology of quad meshes. Section 2.2 shows how rule-based design can be applied to topological trans-parametric design. Section 2.3 introduces the field of topology with graph theory. Section 2.4 presents the specific concepts and properties of quad meshes.

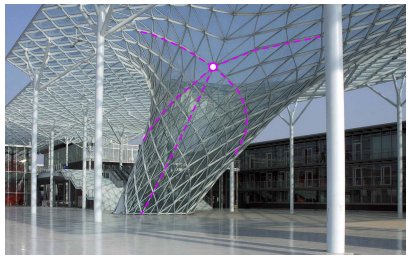
## 2.1 Design of patterns for structures

The current structural design approaches for the topology of quad-mesh patterns, both in practice and research, present several interests but different levels of limitations as well. The main approaches are presented, starting from the most simple ones.

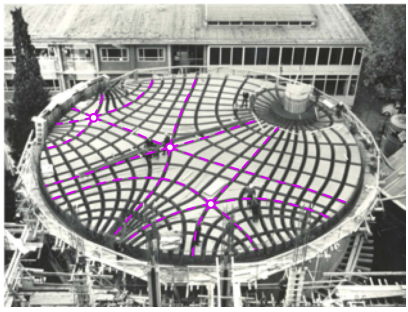
### 2.1.1 Trimming grids

A first approach is to design a regular quad-mesh grid on a large initial surface. Then, trimming along the boundary of the shell fits the pattern to a landscape or existing buildings, for instance. No singularities occur off the boundary of the shell, but the boundary includes a series of irregular vertex and face elements due to the trimming.

(a) The trimmed quad-mesh pattern of the Cabot Circus Shopping Centre roof in Bristol, England [Schober and Justiz, 2012]



(b) The sculpted triangulated quad-mesh pattern of the New Trade Fair in Milan, Italy [Schlaich et al., 2005]



(c) The statics-integrated quad-mesh pattern of the zoology lecture hall at the University of Freiburg, Germany [Antony et al., 2014]

(d) The topology-optimised quad-mesh pattern of the building bridge for the Zendai competition, China [Beghini et al., 2014]

Figure 2.1 – Projects with quad-mesh patterns following different structural design strategies.

The roof of the shopping centre of the Cabot Circus in Bristol, England, shown in Figure 2.1a, is a gridshell with a quad-mesh pattern of steel beams and glass panels. The design follows the geometry of a trans-scale surface to guarantee planar quad panels [Schober and Justiz, 2012]. The boundary of the surface is trimmed to follow the edge of the adjacent buildings. The pattern does not reflect the boundary and support conditions. Trimming induces irregular elements to fabricate and assemble, such as triangular or pentagonal faces or very short edges along the boundary. Trimming also interrupts the force flow carried by the beams, driving the loads

to the boundary to be carried by stiffer beams. Trimming is standard for such strategies where the designer does not entirely control the boundary [Schlaich and Schober, 1997, Schober, 2015]. These considerations are similar when trimming the quad-mesh patterns for elastic gridshells generated on a target surface thanks to the compass method [Otto et al., 1974, Du Peloux et al., 2013]. The compass method yields a Chebychev net, a quad mesh with equal-length edges, which eases the planar layout operation before the erection process. However, the method initiates from an intersection point between two guide curves on the surface and does not take into account the boundaries.

A quad mesh requires singularities to fulfil boundary alignment without trimmed elements, which influences structural and construction aspects. An example of beneficial influence is yielding membrane equilibrium in funicular form finding through alignment to unsupported boundary [Panozzo et al., 2013].

### 2.1.2 Sculpting meshes

Designers can sculpt meshes manually or semi-automatically following some heuristic rules in a trial-and-error approach. Sculpting coarse meshes before densification and application of different operators is an efficient design paradigm [Bhooshan et al., 2018]. Nevertheless, this approach relies on the experience and skill of the designer and still requires some manual labour that is not straightforward to transpose from one project to another.

The roof of the New Trade Fair in Milan, Italy, shown in Figure 2.1b is a steel and glass gridshell [Schlaich et al., 2005]. The structural pattern derives from a quad mesh with additional diagonals for triangulation of the quad faces in curved areas. Thanks to specific mesh modifications, the singularities provide a smooth transition from the top areas to the funnel areas. However, in the development of ad hoc modifications, the decisions are specific to the project context with limited application to other design problems.

### 2.1.3 Optimising layouts

Opposed to linear programming, which applies to continuous parameters, integer programming enables discrete optimisation [Schrijver, 1998]. The connectivity within a set of fixed vertices like the bar layout of a truss can be optimised using such solvers with heuristic methods like genetic algorithms. The problem is classically parameterised with point indices to

define the extremities of the bars [Suzuki and Knippers, 2017] or edge indices to the define the bar in a ground structure [He and Gilbert, 2016]. Shea and Cagan apply simulated annealing on shape grammars – or shape annealing – for optimisation of geodesic dome patterns [Shea and Cagan, 1997]. Bielefeldt *et al.* use genetic algorithms for topology optimisation of a bar layout [Bielefeldt et al., 2019a]. Including the position of the points in the optimisation process mixes discrete and continuous parameters, which can be tackled using mixed integer linear programming.

### 2.1.4 Integrating fields

Integration of cross fields for structural design allows generating informed patterns for different objectives. Integration of principal stress directions guarantees mechanical efficiency [Michell, 1904, Rozvany and Prager, 1976], while the integration of principal curvature directions ensures fabrication properties like panel planarity [Liu et al., 2006]. Heuristic rules can inform the design of a custom field. The integration of such fields provides a pattern that respects these heuristic rules, like boundary and feature alignment [Panozzo et al., 2013]. Quad-mesh generation following a cross-field is a complex problem. Although efficiently-formulated algorithms manage to yield high-quality meshes regarding geometrical quality and feature alignment rapidly [Kälberer et al., 2007, Bommes et al., 2009, Jakob et al., 2015].

The cross-field defines the geometry and topology of the resulting quad mesh. More specifically, the singularities have the position and the connectivity of the ones of the cross-field. A singularity in a field is a point where the field is not defined. Therefore, depending on the numerical precision and the processing scheme, they can appear directly in the output mesh or indirectly as dual faces resulting from the integration of surrounding curves.

Concrete plates and shells reinforced with ribs were made popular by the work of Pier Luigi Nervi [Nervi, 1965, Halpern et al., 2013]. The slab of the zoology lecture hall at the University of Freiburg, Germany, designed by Hans-Dieter Hecker [Hecker, 1969] and shown in Figure 2.1c is stiffened by a pattern of concrete ribs derived from the integration of the principal bending directions. The ribs provide structural efficiency but necessitate custom formwork, which can be hard and expensive to implement. Nevertheless, this system can compete with other ones, like hollow concrete slabs and prestressed concrete slabs, regarding sustainability assessment [Antony et al., 2014].

However, the design comes as a single result from the integration scheme. The topology of the quad mesh is not directly controlled, and post-processing

may be needed. Indeed, mild or substantial modifications of the topology in further design steps would allow integrating other requirements that the cross-field does not take into account.

The curvature analysis of the roof of the Visconti Courtyard for the department of Islamic Arts at the Louvre in Paris, France, yields a quad-dominant mesh [Zadravec et al., 2010, Wallner and Pottmann, 2011]. Even though all panels fulfil the planarity requirement for economical fabrication, the topology does not provide freedom for the optimisation procedure to fulfil the other requirements and constraints.

Schiftner and Balzer [Schiftner and Balzer, 2010] compare two designs for the steel and glass pattern on the geometry of the British Museum Great Court Roof, which was analytically defined by Williams [Williams, 2001]. The two quad-mesh patterns differ by their singularities, which stem from two different cross fields: the principal stress directions and the principal curvature directions. The former performs better regarding structural stiffness and the later regarding panel planarity, as shown in Figure 2.2. Both design aspects can be tackled simultaneously by form finding a shape that aligns the two cross fields [Kilian et al., 2017, Pellis and Pottmann, 2018]. However, this approach requires freedom in the design of the shape, and more project-specific design aspects must still come into play.

### 2.1.5 Concentrating distributions

Mechanical topology optimisation based on mechanical compliance yields both the geometry and the topology of the pattern as well [Bendsøe and Sigmund, 2013]. Continuous topology optimisation progressively removes the lower-stressed material in a continuous domain for given loading conditions, as for the bridge-building in Figure 2.1d. Discrete topology optimisation progressively removes the lower-stressed bars in a dense network for given loading conditions. Theoretically, the resulting ideal pattern should follow the principal stress lines and yield a quad-dominant mesh as in cross-field integration [Michell, 1904, Rozvany and Prager, 1976, He and Gilbert, 2016].

A topology-optimisation process returns highly optimised patterns regarding mechanical considerations for a specific load condition. Nevertheless, structures at the architecture scale must strike a compromise between the very varied combinations of load cases, such as dead loads, live loads, snow loads, wind loads, seismic loads. Moreover, these patterns are not necessarily feasible regarding construction considerations [Borgart, 2010]. In practice, the pattern is unstructured and does not even correspond to a

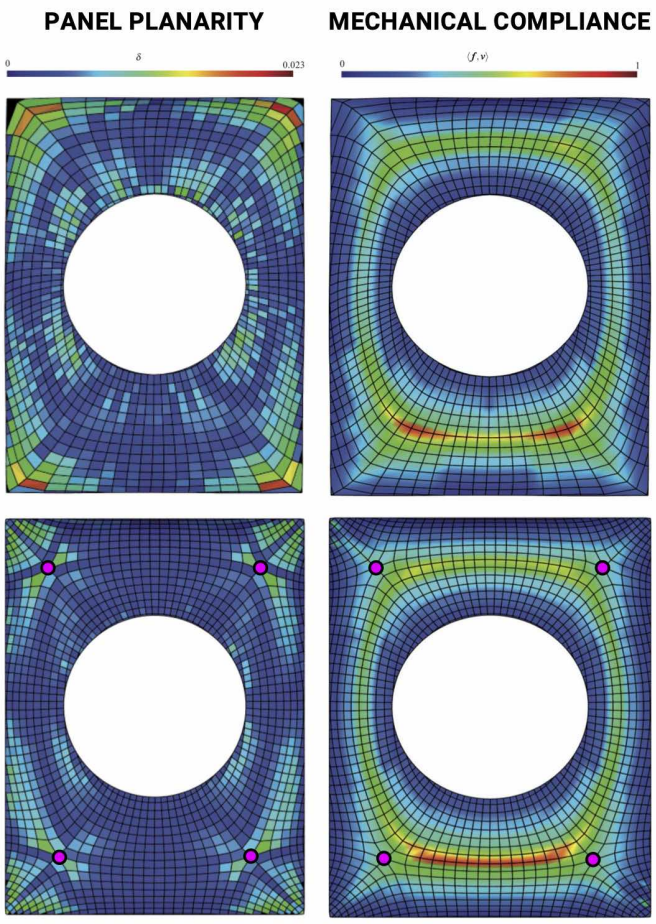



Figure 2.2 – Two pattern designs revisiting the British Museum roof. The topology at the top performs the best regarding the requirement on the right on minimising mechanical compliance. The topology at the bottom performs the best regarding the requirement on the left on minimising panel planarity [Schiftner and Balzer, 2010]. The singularities in a quad mesh, highlighted in pink, influence its efficiency and affordability, highlighting the importance of their design for the challenge of meeting trade-offs in multi-objective structural design.

mesh, which could be directly further processed.

However, optimisation approaches like field integration and topology optimisation can be used as a starting design for further exploration, and as a collaboration means between architects and engineers for integrated design [Beghini et al., 2014]. Such initial results can inform the designer, who could be empowered by the possibility of editing these results and combining them for multi-objective design.

Complementary to these main strategies, a rule-based design approach can be considered for the topological design of patterns for structures for shell-like structures, as used in many applications.

## 2.2 Exploration of topological spaces

Topological spaces do not have the same structure as geometrical spaces. Geometrical spaces rely on continuous parameters. Rule-based design and grammars compensate for this lack of continuous parameters to perform topological exploration.

### 2.2.1 Non-parametric spaces

Spaces are differentiated whether they have a metric or not.

#### 2.2.1.1 Parametric spaces

A set of independent continuous-valued parameters defines a parametric space. For instance, the set of the vertex coordinates describe the geometry of a mesh. Exploring these parameters within their defined ranges allows describing all the objects in the design space. Continuous optimisation algorithms can apply through the computation of these parameters.

For the fields of architectural and structural design, among others, Rhino's Grasshopper, or Autodesk's Dynamo, enable an active community of students, practitioners and researchers to perform parametric design. The parameters can be interactively modified while visualising the resulting design. Mechanical, light, thermal or acoustic analysis, a variety of form-finding and shape-optimisation algorithms or BIM software can provide additional information. Thanks to an open-source approach, hundreds of plugins with hundreds of thousands of downloads can be browsed through in food4Rhino, contributing to knowledge transfer [food4Rhino, 2019].

Parametric spaces are metric spaces because canonical metrics based on the $n$ parameters $x = [x_0, ..., x_{n-1}]^T$ can be defined such as the $L^2$ distance $d$:

$$d(x, y) = \sqrt{\sum_{i=0}^{n-1} (x_i - y_i)^2}, \tag{2.1}$$

which allows evaluating the distance – or similarity – between two objects. In the 2D or 3D Euclidean space, three coordinates describe each point, and the $L^2$ distance is the Euclidean distance, the length of the straight line between the two points.

Metrics – or distances – must verify several properties:

- the distance between any two objects is positive;
- if the distance between two objects $x$ and $y$ is zero, then $x$ and $y$ are identical, and reciprocally: $d(x, y) = 0 \iff x = y$;
- the distance is symmetric: the distance from an object $x$ to an object $y$ is the same as the distance from $y$ to $x$: $d(x, y) = d(y, x)$;
- the distance, respects the triangle inequality: the distance between two objects $x$ and $z$ is equal or smaller than the one between $x$ and any object $y$ plus the one between $y$ and $z$: $d(x, z) \leq d(x, y) + d(y, z)$.

#### 2.2.1.2 Topological spaces

However, some design spaces do not have such parameters, like the beam layout of a truss. For such topology-related spaces, parametric exploration is not possible as a finite set of continuous parameters is missing. Indeed, a combinatorial nature characterises such spaces.

Moreover, the definition of a canonical metric, based on continuous parameters, is not possible.

However, an alternative approach to parametric design enables topological exploration based on grammars of rules.

### 2.2.2 Rule-based design

Parametric design applies to continuous parameters. On the contrary, rule-based design applies topological modifications based on a set of rules grouped in a grammar. Rule-based design goes beyond parametric design and into

the field of trans-parametric design [Harding et al., 2012, Harding and Shepherd, 2017]. Thinking about topological design at an early stage of the design process unleashes new design possibilities in structural applications. For instance, Deleuran *et al.* [Deleuran et al., 2016] perform topological exploration of form-active hybrid structures using rules that modify the connectivity of a network of splines and cables.

### 2.2.3 Design grammars

Rule-based design encapsulates different approaches with design grammars that can apply to language, form and structure, through different levels of complexity, starting with formal grammars.

#### 2.2.3.1 Formal grammars

Chomsky introduces in the 1950s generative design applied to language using formal grammars [Chomsky, 1956, Chomsky, 1959]. A finite set of formal grammar rules apply modifications on a finite set of words, enabling the generation of an infinite set of sentences. The design rules define the language – or design space – to which these sentences – or designs – belong.

L-systems extend formal grammars for the generation of geometries.

#### 2.2.3.2 L-systems

Lindenmayer introduces in 1968 L-systems as a type of formal grammars to algorithmically describe the growth of plants, as shown in Figure 2.3 [Lindenmayer, 1968, Prusinkiewicz and Lindenmayer, 2012]. A set of rules applies to a string of characters that come from an alphabet. The interpretation of the string generates the corresponding shape, in the manner of turtle graphics [Goldman et al., 2004]. The rules are applied iteratively to generate different level of growth of the shape. The shape depends on the starting string axiom and the different parameters, in the case of parametric rules. L-systems can also describe the generation of patterns such as fractals.

L-systems can apply to structural design as parameterisation strategies for the generation of designs. The combination of L-systems and genetic algorithms provides a means for topology optimisation for the search for statics-optimised designs [Kobayashi, 2010, Bielefeldt et al., 2019a].

Shape grammar do not modify a string but directly a geometry and found many applications for exploration and design of general geometries.

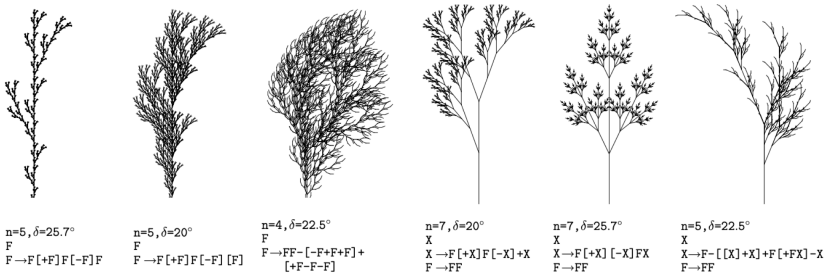



Figure 2.3 – L-systems are a specific type of formal grammars interpreted by turtle graphics designed for algorithmic generation of plants [Prusinkiewicz and Lindenmayer, 2012]. Here, the designs stem from a different number of iterations $n$, a different branching angle $\delta$, a starting string axiom and one or several rule to iteratively apply.

#### 2.2.3.3 Shape grammars

Stiny and Gips introduce in 1971 shape grammars for the generation of shapes in painting and sculpting [Stiny and Gips, 1971]. Subsequently, shape grammars found a large set of applications in many fields of design and engineering [Stiny, 2006, Knight and Stiny, 2015, Cagan, 2001]. A classification can be found in [Garcia, 2017].

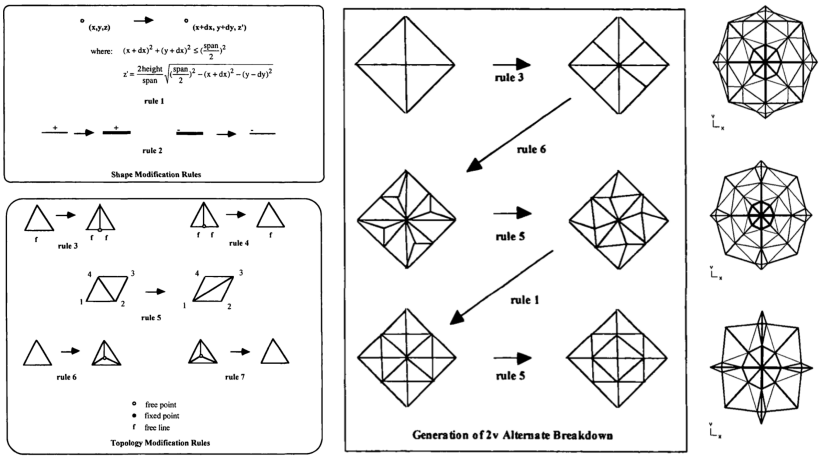
Shape grammars at the architectural scale develop into architectural grammars.

**Architectural grammars** Architectural grammars include a variety of applications, such as Palladian villas [Stiny and Mitchell, 1978], Frank Lloyd Wright's prairie houses [Koning and Eizenberg, 1981], Queen Anne Houses [Flemming, 1987], Yingazo Fashi Chinese buildings [Li et al., 2001] or Siza's houses in Malagueira [Duarte, 2005]. The grammar decodes each architectural style, formalised and structured into a set of rules to generate multiple other designs that share the same characteristics. Design grammars also apply at other scales than the building. Urban shape grammars tackle the district, like the Medina of Marrakesh [Duarte et al., 2006], or the city, like Praia in Cabo Verde [Beirão et al., 2011]. Product shape grammars tackle specific styles of chairs [Knight, 1980], coffeemakers [Agarwal and Cagan, 1998] cars [McCormack et al., 2004] or tableware [e Costa and Duarte, 2013], for instance, allowing mass customisation. An architectural grammar

for housing rehabilitation takes into account the varied information like new usage, among others, instead of decoding a design style [Eloy and Duarte, 2011].

Even though architectural grammars focus on the arrangement of shapes, grammars can include functional aspects other than geometry.

**Functional, structural and force grammars** Shape grammars evolve into functional and structural grammars to include non-geometrical data related to structures. This data can include structural-behaviour and construction-technology requirements. Engineering applications include houses [Mitchell, 1991], towers [Baldock et al., 2005, Baldock, 2007], halls [Geyer, 2008], bridges, [Mueller, 2014] or trusses [Shea et al., 1997]. Figure 2.4 highlights an application to a shell-like system with a grammar applied to the design of geodesic domes [Shea and Cagan, 1997, Shea and Cagan, 1999]. Some applications are specific to a fabrication technology, like CNC machines [Sass, 2006, Ertelt and Shea, 2009], instead of a structural system. Force grammars describe the organisation between forces, opposed to the one between spaces or elements, which can later materialise into a structure. Force grammars for graphic statics are introduced by Lee *et al.* applied to 2D edge networks [Lee et al., 2016b] and 3D cell decompositions [Lee et al., 2016a].




(a) Grammar (b) Examples (c) Designs

Figure 2.4 – Grammar for rule-based design of triangulated meshes for geodesic domes [Shea and Cagan, 1997].

The exploration of the topology of patterns and meshes resulted in the development of several operations, sometimes also framed as grammar rules, which are not specific to architecture and structures.

#### 2.2.3.4 Mesh grammars

The large range of applications of meshes sparked the design of very different sets of rules tuned for specific objectives, mainly for aesthetics purpose.

**Ornamental design** Hansmeyer and Dillenburger [Hansmeyer and Dillenburger, 2013] introduce a mesh grammar following a formal grammar approach that modifies the density and the geometry of meshes to generate highly-detailed ornamental shapes. This grammar focuses on shape. Indeed the pattern and its singularities are not modified.

**Visual optimisation** In the field of computer graphics, quad-mesh grammars modify the topology to improve the regularity of dense and unstructured meshes [Daniels et al., 2008, Daniels II et al., 2009, Tarini et al., 2010, Peng et al., 2011]. This regularity for modelling and representation concerns both geometry and topology. These quad-mesh grammars consist of a set of local rules that preserve the quad-mesh constraint, unlike other grammars for triangulated meshes for instance. Figure 2.5 illustrates some of these rules [Tarini et al., 2010].

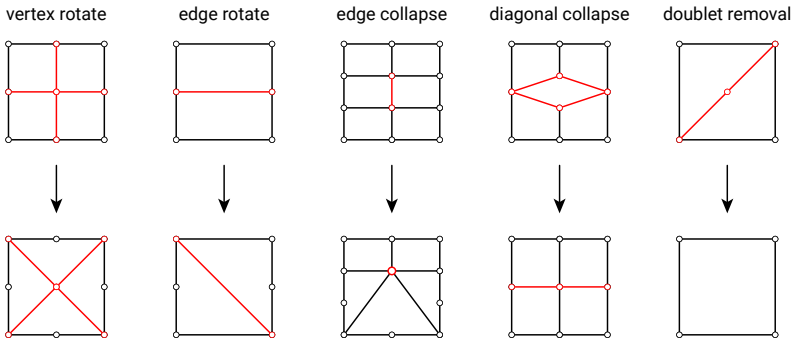



Figure 2.5 – A grammar rules for quad mesh visual-quality optimisation [Tarini et al., 2010].

Another family of quad-mesh grammars does not consist of rules but different patterns [Nasri et al., 2009, Takayama et al., 2014, Peng et al.,

2014]. The patterns feature different singularities to patch N-sided polygons with a prescribed number of subdivisions on each side. Combining these patterns allows completing the mesh of different shapes.

These rules apply to meshes for the field of computer graphics, animation and rendering. These rules have little use in the context of architectural and structural design where design does not start with a dense unstructured mesh and where many more objectives come into play.

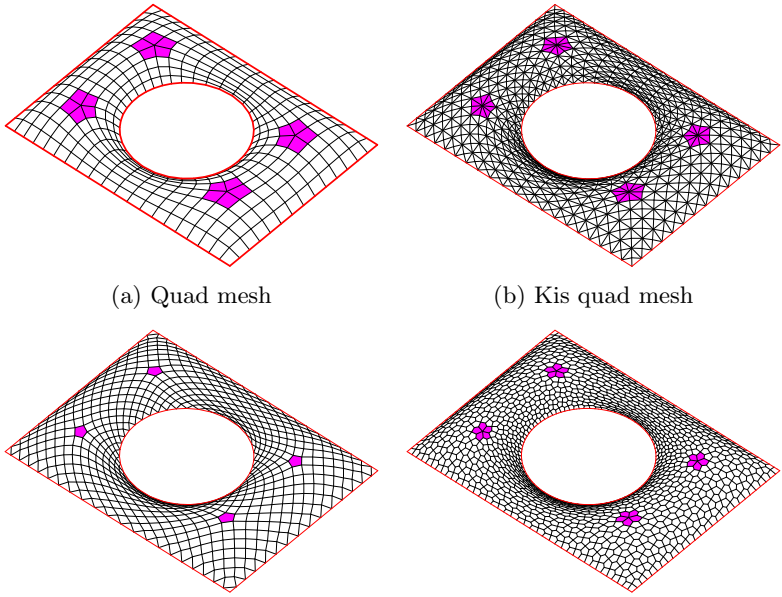
**Tessellation exploration** The Conway operators constitute a grammar developed by John H. Conway for the description of polyhedra [Conway et al., 2016]. Applying these operators modifies the tessellation and the density. However, these operators preserve singularities. Figure 2.6 illustrates some of the different tessellations a seed quad mesh can yield. The kis, ambo and gyro Conway operators yield a doubly-triangulated, a dual diagonal and a pentagonal pattern, respectively. The new patterns have equivalent vertex or face singularities, highlighted in pink. The five-valent singular vertices become ten-valent singular vertices, five-valent singular faces and five-valent singular vertices, respectively.

Therefore, Conway operators can be applied to translate one tessellation into another, already applied for the optimisation of space frame structures [Shepherd and Pearson, 2013, Koronaki et al., 2017]. Indeed, tessellations present different structural and fabrication properties worth investigating [Malek and Williams, 2013, Jiang et al., 2015, Mesnil et al., 2017b].

Developing grammars for the different types of tessellation does not make sense. Instead, a pattern grammar can combine a quad-mesh grammar and the Conway tessellation grammar. Consequently, quad-mesh grammars can modify the singularities in other patterns. The four pentagonal tilings in Figure 2.7 stem from quad meshes with different singularities. The singularities induce different distortions when mapped to the same disc surface. The faces around the equivalent vertex singularities are highlighted in pink.

**Solid modelling** Heisserman [Heisserman, 1994] presents a grammar on the boundary of solids represented by meshes for modelling of architectural spaces or volumetric objects. Modifications of the topology result from modifications of the geometry. When a movement modifies the adjacency or creates an overlap, the topology is updated. This grammar explores the topology of the shape, the underlying mesh being only a representation for computation.

Applying rule-based exploration to quad meshes must comply with spe-



(a) Quad mesh (b) Kis quad mesh

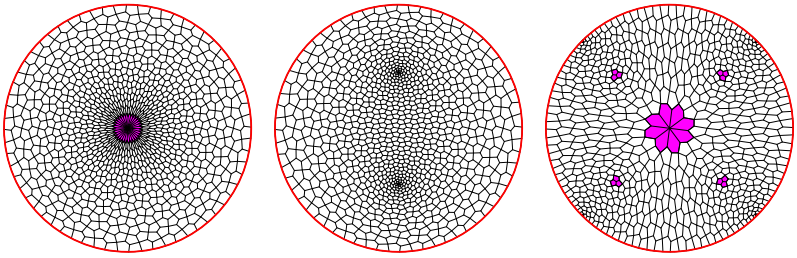
(c) Ambo quad mesh (d) Gyro quad mesh

Figure 2.6 – Applying Conway operators on a seed quad mesh generates patterns with different tessellations but equivalent vertex or face singularities in pink.

cific topological constraints, which general graphs encoding the connectivity between objects do not have.

## 2.3 Topology and graph theory

Patterns for representing structural elements, for describing force equilibria or for morphing surfaces can all be described by meshes. Work in geometrical processing of meshes came naturally into structural design, framed as *constructive geometry* [Sakarovitch, 2009] or *architectural geometry* [Pottmann et al., 2007a]. Beyond meshes, graphs are more general objects that describe a set of edges, as opposed to a set of faces. A graph is an efficient object to store high-dimension, non-Euclidian data, as opposed to vectors and matrices. Graphs have found applications for problem encoding and solving in very varied fields. Graph focus on connectivity, or topology, in-



(a) From one pole

(b) From two two-valent singularities

(c) From one eight-, four three-and four two-valent singularities

Figure 2.7 – Pentagonal tilings stemming from quad meshes with different singularities inducing different distortions when mapped to the same disc surface. The faces around the equivalent vertex singularities are highlighted in pink.

stead of geometry. Nevertheless, graphs can also describe a geometry, like a spatial network of forces in equilibrium [Schek, 1974, Block and Ochsendorf, 2007]. This thesis explores the connectivity of patterns, which calls to the rich theory on topology and graph objects.

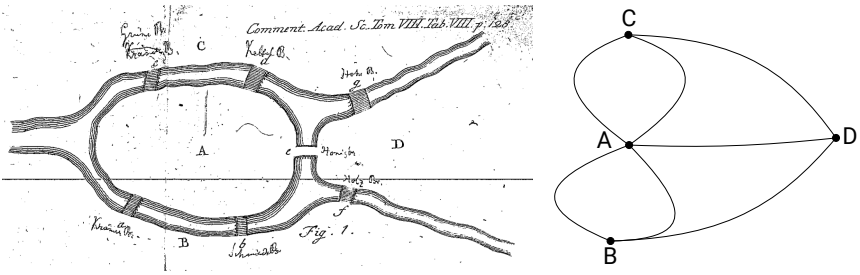
The two domains of topology and graph theory are historically connected.

### 2.3.1 Origin

In 1741, Leonhard Euler asked and answered a question about the city of Königsberg, today's Kaliningrad [Euler, 1741]. Königsberg had four districts separated by the Pregel River and connected by seven bridges, as shown in Figure 2.8a. Can someone walk such as to cross each bridge exactly once and finish at the starting point? This question relates only to the topology. Indeed, the solution is independent on the location or size of the districts, only on their connectivity.

Euler proved that no such loop exists. To solve this practical question, Euler abstracted the problem into a graph data structure as in Figure 2.8b. Each graph vertex represents a district of the city, and each graph edge represents a bridge connecting two districts. The graph encodes the necessary data of the problem to provide an answer.

The problem of the seven bridges of Königsberg gave birth to topology




(a) Map of the city [Euler, 1741]. (b) Graph encoding

Figure 2.8 – The problem of the seven bridges of the city of of Königsberg, encoded by Leonhard Euler in a graph, where each vertex represents a district and each edge represents a bridge between two districts.

and graph theory that are today ubiquitous in our daily lives. Indeed, graphs can abstract and encode many problems.

### 2.3.2 Data structure

The data in a graph consists of two elements. Vertices – or nodes – and edges connecting pairs of vertices. The vertex and edge elements can store attributes that are relevant for the problem considered. For a network of people, the graph vertices represent the persons, and the graph edges represent the relations. Vertex attributes can provide information on the persons (like the name or the age), and edge attributes can provide information on their relation (like the type of relation or the beginning of the relation).

### 2.3.3 Applications

The development of a large body of research on algorithms for different graph-based problems flourished since Euler [Gondran and Minoux, 1984], triggered by a large number of applications. Indeed, many problems are structured as such as they can not be represented using multi-dimensional matrices. Graphs better conceptualise such problems, with additional attributes stored on the vertices and edges. Applications for graph vertices and edges include social networks as people and connections; map colouring as countries and borders; airport connections as airports and connecting flights; molecules as atoms and bonds; neural networks as neurons and synapses. And the topology of patterns for structural design.

### 2.3.4 From graphs to meshes

A mesh is an adaptation of a graph. A mesh data structure is also based on vertices but replaces edges by faces to define the connectivity between these vertices. The edges in a mesh are secondary elements that result from the faces. Meshes have a more specific structure and are more suitable to define surfaces or manifolds. Similarly to graph data structures, vertices, faces and edges store attributes such as node coordinates, panel thicknesses or beam profiles, for construction applications.

Meshes can be seen as hybrids between graphs and surfaces as they share some properties with both.

## 2.4 Topology of quad meshes

Beyond geometrical properties, meshes, and more specifically quad meshes, have different topological properties. These properties exist at the level of the shape and the pattern. Moreover, theorems connect these two levels together and relate them to geometry.

### 2.4.1 Shape topology

Two shapes have the same topology, and therefore the same shape topological properties, if they are *homeomorphic*. Homeomorphism means that one surface can deform into the other one by continuous and reversible mapping. Stretching is allowed but not tearing. For instance, a doughnut and a mug have the same topology, more formally called torus, but a sphere, a torus or a double torus are topologically different, due to the different number of handles, as illustrated in Figure 2.9.

A classification sorts shapes according to their shape topological properties, which do not depend on a choice of mesh representation and their local properties. However, a mesh representation may be necessary to compute these global properties.

#### 2.4.1.1 Classification of shapes

Figure 2.10 summarises the main properties for topological classification of a shape. The shape can be:

1. *manifold* or not;
2. if so, *compact* or not;

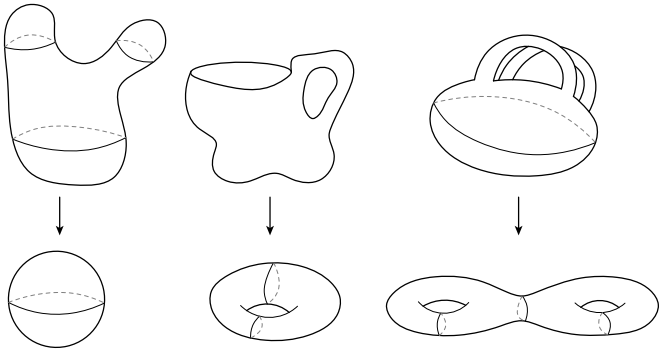


Figure 2.9 – Continuous and reversible deformation of a shape into another implies that they have the same shape topology. A sphere, a torus and double torus have different topologies. (Inspired from: learner.org)

3. if so, *orientable* or not.

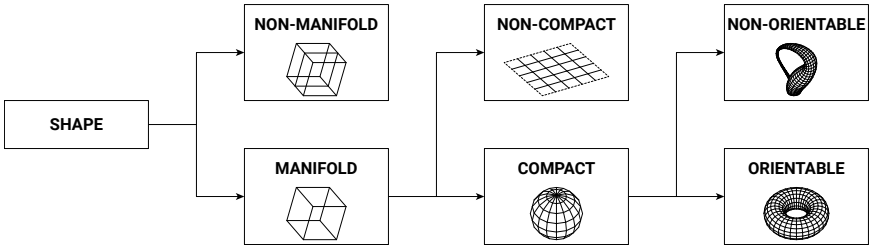



Figure 2.10 – Topological classification of shapes.

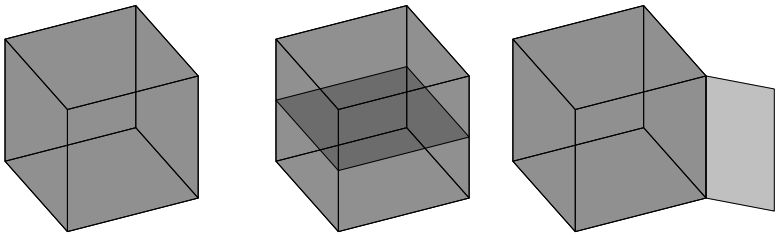
**Manifold versus non-manifold** Yamaguchi [Yamaguchi, 1997] classifies manifold and non-manifold shapes based on adjacency between vertex, edge, face and regions. The more general the shape, the more advanced the necessary data structures for representation and modelling:

- *manifolds* are shapes on which every point is locally homeomorphic to a disc (Figure 2.11a). Manifolds can be represented by a halfedge data structure (the least general one) [Mäntylä, 1988];
- *rigid-set solids* are single connected closed sets with a volume (Figure 2.11b). Rigid-set solids can be represented by a halfwedge data

structure [Requicha, 1977];

- *cell decompositions* are generic models of shapes with closed and open cells (Figure 2.11c). Cell decompositions can be represented by a feather data structure (the most general one) [Yamaguchi and Kimura, 1995].

A non-manifold shape is caused by edges with more than two adjacent faces, by faces with both sides adjacent to the same region, or by vertices adjacent to several non-connected fans of faces.



(a) Manifold

(b) Rigid-set solid (some points are not homeomorphic to a disc)

(c) Cell decomposition (some cells are open and a volume is not defined)

Figure 2.11 – Manifold and non-manifold shapes.

**Compact versus non-compact** A manifold shape can be compact or non-compact. A compact shape has a finite area, meaning triangulation is possible with a finite number of elements [Rowland, 2019]. A sphere or a torus are compact manifolds. However, the infinite plane, the infinite cylinder and the other shapes in Figure 2.12 have an infinite area and are therefore non-compact. Further classification of non-compact manifolds without and with boundaries can be found in [Richards, 1963] and [Prishlyak and Mischenko, 2007], respectively.

**Orientable versus non-orientable** A compact manifold shape is orientable if normals are consistently oriented. Adjacent faces in a mesh representation must have their normals oriented in the same direction relative to the surface. A sphere, a torus, a disc or a cylinder are orientable. On the contrary, the Möbius strip in Figure 2.13 or the Klein bottle are non-orientable shapes because their two sides are connected.

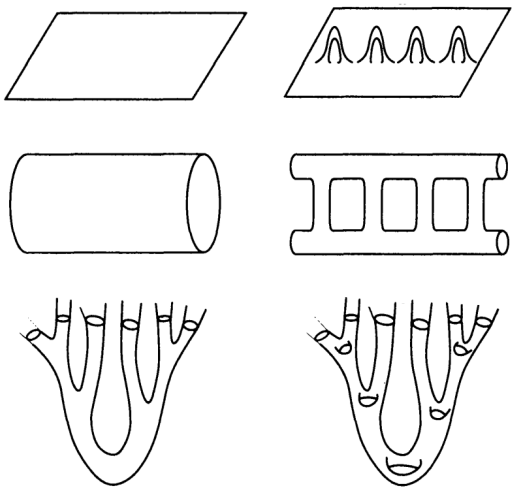


Figure 2.12 – Some non-compact manifold shapes like the infinite plane or the infinite cylinder [Ghys, 1995].

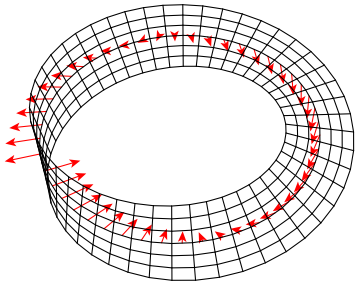


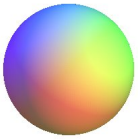
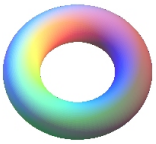
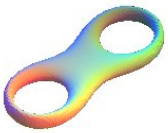
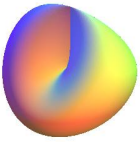
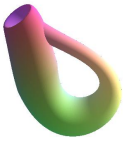
Figure 2.13 – The Möbius strip is a non-orientable shape because the normals in red can not be consistently oriented on the same side.

**Euler characteristic and genus** The compact manifold shapes can be further classified. Seifert and Threlfall proved this 1860s classification in 1934 [Seifert and Threlfall, 1934], improved by Conway in 1992 [Francis and Weeks, 1999].

The classification theorem states that a closed compact manifold shape is topologically equivalent to a sphere with a certain number of either handles

or cross-caps. A cross-cap is a self-intersection of a non-orientable shape. In other words, the genus, the number of handles or cross-caps, characterises orientable and non-orientable closed shapes, respectively. Table 2.1 shows the classification for shapes of genus from zero to two.

Table 2.1 – Classification of compact manifold shapes based on their orientability and their genus (source: mathcurve.com).

| genus | 0 | 1 | 2 |
|---|---|---|---|
| orientable |  sphere |  torus |  double torus |
| non-orientable | |  cross surface |  Klein bottle |

The genus $g$ relates to the Euler characteristic $X$. The Euler characteristic can be computed from any mesh representation of the shape as:

$$X = V - E + F, \tag{2.2}$$

where $V$ is the number of vertices, $E$ the number of edges and $F$ the number of faces in the mesh. The genus depends on the Euler characteristic as follows for orientable shapes:

$$X = 2 - 2g, \tag{2.3}$$

and as follows for non-orientable shapes:

$$X = 2 - g. \tag{2.4}$$

The doublet (orientability, genus) characterises a closed compact manifold shape.

**Closed versus open** Open shapes, which include boundaries, are classified using equivalent closed shapes, as illustrated in Figure 2.14:

1. the $N$ boundary components are collected;
2. a disc is zipped on each boundary component to close the open shape;
3. the orientability and the genus of the resulting closed shape are computed.

For instance, a cylinder has two openings and zipping a disc along each one yields a sphere topology. Therefore the cylinder is characterised as an orientable shape with zero handles and two boundary components. The triplet (orientability, genus, number of boundary components) characterises an open compact manifold shape.

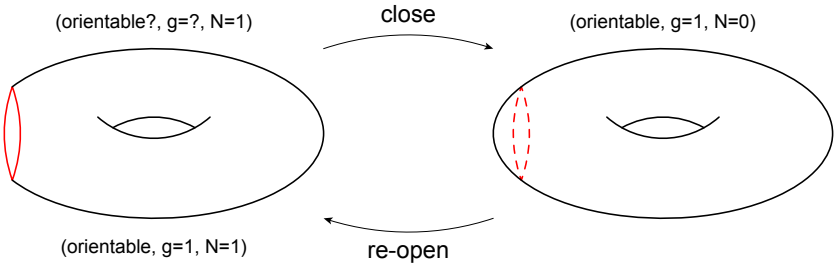



Figure 2.14 – Characterisation of an open compact manifold shape via the equivalent closed shape by temporarily zipping a topological disc along each boundary.

The Euler characteristic $X$ of an open compact manifold shape is computed without resorting to a mesh representation by correcting Equation 2.3 for orientable shapes as:

$$X = 2 - 2g - N, \tag{2.5}$$

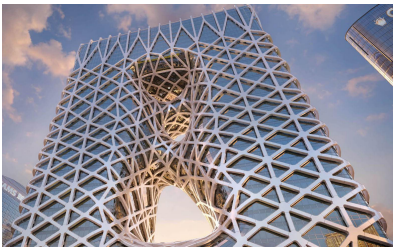
and Equation 2.4 for non-orientable shapes as:

$$X = 2 - g - N. \tag{2.6}$$

#### 2.4.1.2 Shape topology in construction

Different shape topologies and corresponding data structures apply to models in the construction industry. Building Information Modelling (BIM) of

(a) Orientable shape with zero handles of the British Museum in London, England
(Source: e-architect.co.uk)



(b) Orientable shape with a few handles of the Morpheus Hotel in Macau
(Source: macaubusiness.com)

(c) Orientable shape with many handles of the 2015/2016 ICD/ITKE Research Pavilion in Stuttgart, Germany
(Source: archdaily.com)

(d) Non-orientable shape of the Arnhem Central Station, The Netherlands
(Source: ice.org.uk)

Figure 2.15 – Built projects with different shape topologies.

slabs, walls, beams and columns necessitates topological modelling tools that handle non-manifold shapes [Aish et al., 2018].

Nevertheless, manifolds can represent most shell-like structures. Even constrained to orientable compact manifold shapes with a null or low number of handles, except for some following notable exceptions. Indeed, most roof structures are orientable shapes with one outer boundary and potentially inner boundaries, as for the British Museum in London, England (Figure 2.15a). Facade structures can have some handles due to connections between parts of a building, as for the Morpheus Hotel in Macau (Figure 2.15b). Multi-layer structures can have a high number of handles due to connection between the layers, as for the 2015/2016 ICD/ITKE Research

Pavilion (Figure 2.15c). Hybrid structures can be non-orientable, as for the Arnhem Central Station, The Netherlands (Figure 2.15d). However, a non-orientable shape as building skin questions the definition of interior and exterior, and therefore the habitability of the enclosed space.

Opposed to shape topology, local topological properties apply at the level of the mesh representation of the shape.

### 2.4.2 Pattern topology

Similarly to shapes, two patterns have the same topology if one can deform into the other without modifying the connectivity between its elements. For the same shape topology, different pattern topologies are possible. In the case of quad meshes, these topologies can differ by their singularities.

#### 2.4.2.1 Singularities

In a closed quad mesh, regular vertices have a valency of four, which means they have four edges connecting to adjacent vertices. On the boundary of an open quad mesh, regular vertices have a valency of three. Singularities are vertices with an irregular valency, meaning different from four off the boundary and three on the boundary. Singularities generate different degrees of deviations in the flow of elements in the mesh, as shown in Figure 2.16.

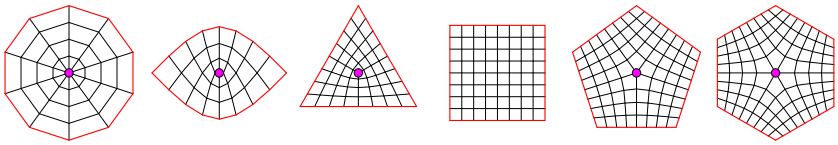


Figure 2.16 – Singularities in quad meshes have a valency different from four and induce different local flow deviations.

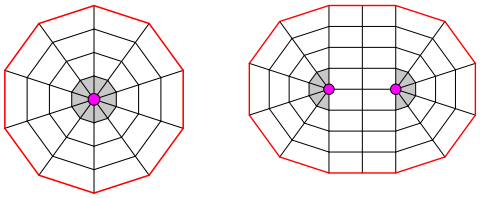
Equivalently, the valency of a singular vertex in a triangular or hexagonal mesh differs from six and three, respectively. Singularities can also refer to irregular faces, instead of irregular vertices. Kagome patterns consist of hexagonal and triangular faces, therefore, a five-or seven-valent face is singular [Ayres et al., 2018].

Structured meshes are meshes with a low number of singularities relatively to the number of regular vertices. Unstructured meshes result in disorganised flows of elements. Focus on structured patterns in structural

design is justified by its positive implications on aesthetics, statics and construction [Stephan et al., 2004].

#### 2.4.2.2 Poles

Poles, characterised by a local polar pattern, are a special type of singularities in quad meshes as they are adjacent to triangles, in grey in Figure 2.17a. The triangles should be interpreted as pseudo-quads with a double vertex at the location of the pole. The high valency of a pole varies as it depends on the mesh density. Partial poles are hybrid singularities adjacent to both quads and triangles, as shown in Figure 2.17b.



(a) Full pole (b) Partial pole

Figure 2.17 – Poles in quad meshes are a special type of singularities adjacent to triangles, in grey.

Pole points are of particular interest in structural design of patterns. Isostatic ribbed floors feature such poles. Such examples include the slab of Pier Luigi Nervi for the Gatti Wool factory in Rome, Italy, in Figure 1.7b [Nervi, 1965, Halpern et al., 2013], and the slab of Hans-Dieter Hecker for the lecture hall of the zoological department of the University of Freiburg, Germany, in Figure 2.1c [Hecker, 1969, Antony et al., 2014]. The principal stresses converge towards columns and walls thanks to poles in the pattern that attract forces to the supports, but detailing of these high-valency nodes is more complicated. The structural patterns of the gridshells of the courtyard roofs of the Dutch Maritime Museum [Adriaenssens et al., 2012] in Amsterdam, in Figure 1.4b, and of the British Museum [Williams, 2001] illustrate this trade-off. Indeed, they have similar support conditions, allowing thrust only at the four outer corners. However, the former features poles, whereas the latter does not.

Moreover, pole points in force patterns can provide an appropriately high number of loads paths at the location of concentrated loads and improve the

results of form fitting of target shapes with compression-only force patterns [Van Mele et al., 2014].

Poles can be understood as a combination of sources and sinks attracting and repelling elements, as done by Nsugbe and Williams [Nsugbe and Williams, 2001] to generate patterns using complex functions.

#### 2.4.2.3 Indices

Vertices are characterised by their valency, or equivalently by their index. The singularity index relates directly to the induced deviation in the flow of mesh elements. The index $i_v$ of a vertex $v$ equals the total signed variation of face orientation $d\theta_{ij}$ between the pairs of adjacent faces $f_i$ and $f_j$ in the adjacency of the vertex $F_v$, normalised by $2\pi$:

$$i_v = \frac{1}{2\pi} \sum_{f_i, f_j \in F_v} d\theta_{ij}. \tag{2.7}$$

Figure 2.18 shows this orientation variation for a six-valent singularity. The deviation adds up to a negative half-turn, thus an index of -1/2.

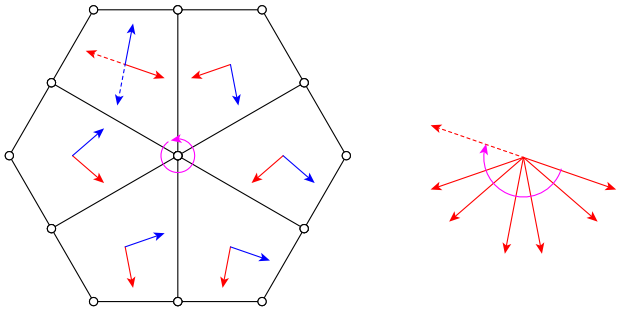


Figure 2.18 – A six-valent singularity has an index of -1/2 due to a negative half-turn of accumulated deviations of face orientations between pairs of adjacent faces.

In a quad mesh, the index can also be directly computed from the valency $n_v$ and its deviation from the regular valency $n_0$:

$$i_v = \frac{n_0 - n_v}{4}, \tag{2.8}$$

where the regular valency $n_0$ equals 4, or 3 for boundary vertices. For each face added or removed, the index is increased or decreased by -1/4,

respectively. The index of poles is computed using Equation 2.7: poles have an index of 1 outside the boundary and 1/2 on the boundary. The index of partial poles varies and is computed using Equation 2.7 or using Equation 2.8 after collapsing the triangles.

Table 2.2 summarises the indices of singularities, including poles $\star$.

Table 2.2 – Indices of singularities with valency from 2 to 9, general valency n, and poles $\star$.

| valency | index outside boundary | index on boundary |
|---|---|---|
| $\star$ | 1 | 1/2 |
| 2 | 1/2 | 1/4 |
| 3 | 1/4 | 0 |
| 4 | 0 | - 1/4 |
| 5 | - 1/4 | - 1/2 |
| 6 | - 1/2 | - 3/4 |
| 7 | - 3/4 | - 1 |
| 8 | - 1 | - 5/4 |
| 9 | - 5/4 | - 3/2 |
| n | (4 - n)/4 | (3 - n)/4 |

The index of a vertex has an equivalent in other structured meshes, or equivalent fields, such as triangular or hexagonal meshes. Generally, the index provides a measure of the irregularity induced by a singularity [Fogg et al., 2018].

Pattern and shape topology provide a direct relation between the vertex indices and the Euler characteristic.

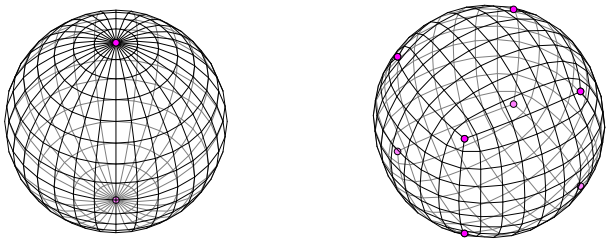
### 2.4.3 Pattern and shape topology

The Poincaré-Hopf theorem (PHT) links shape topology and pattern topology via the Euler characteristic $X$ and the sum of vertex indices $i_v$:

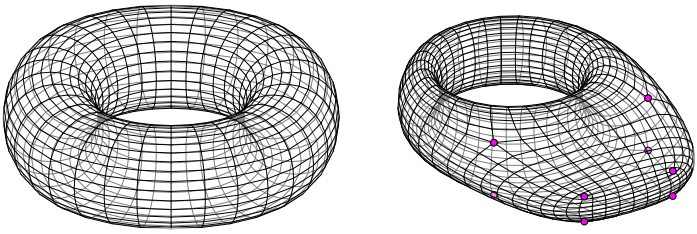
$$X = \sum_{v \in V} i_v, \tag{2.9}$$

where $V$ is the set of all the vertices. The Euler characteristic sets a constraint on the choice of combinations of singularities for a given shape topology. Nevertheless, multiple combinations are still permitted.

For a sphere, $X = 2$, so both pattern topologies in Figure 2.19a fulfil the PHT with two poles of index one as in a globe, or eight singularities of index 1/4 as in a cube. For a torus, $X = 0$, so no singularities are required but combinations of pairs of singularities with indices +1/4 and -1/4 also fulfil the PHT, as shown in Figure 2.19b.



(a) Sphere topology



(b) Torus topology

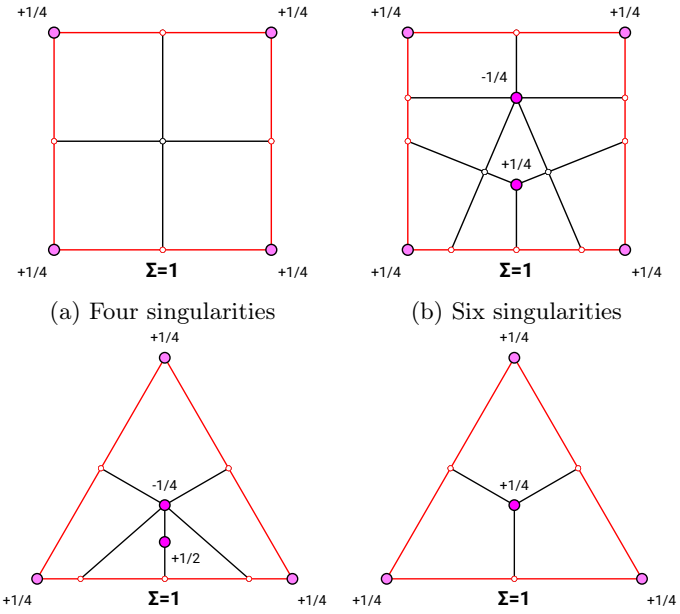
Figure 2.19 – Quad meshes with the same shape topology but different pattern topologies.

Differentiation of the PHT due to a topological modification of the pattern that preserves the shape topology yields:

$$0 = \sum_{v \in V^+} i_v - \sum_{v \in V^-} i_v + \sum_{v \in V^*} \Delta i_v, \tag{2.10}$$

where $V^+$ is the set of added vertices, $V^-$ the set of deleted vertices, $V^*$ the set of modified vertices, $i_v$ the vertex index and $\Delta i_v$ the modification of the vertex index. This equation shows that a singularity can not just be added or deleted independently from the others or it would break the PHT. Nevertheless, a 5-valent and a 3-valent singularity can be added together, as shown from Figure 2.20a to 2.20b. Indeed, the sum of their indices is null. A 6-valent and a 3-valent singularity can be merged into a 5-valent one, as

shown from Figure 2.20c to 2.20d. Indeed, this operation preserves the sum of the indices.




(a) Four singularities (b) Six singularities

(c) Five singularities (d) Four singularities

Figure 2.20 – For a given shape topology, the combination of singularities must respect the PHT. The circle-homotopic shapes have an Euler characteristic of one, therefore the sum of the vertex indices equals one.

Exploring the combinations of the local singularities must follow this global constraint. One can not directly add or delete a singularity. The development of a grammar modifying the singularities must apply at an intermediary scale between the vertex and the mesh. For instance, Peng *et al.* [Peng et al., 2011] base their connectivity editing operators on pairs of singularities. The operators simplify the pattern topology of unstructured dense quad meshes while preserving the quad-mesh and shape-topology constraints.

Pattern topology and geometry provide a direct relation between the vertex indices and curvature.

### 2.4.4 Pattern topology and geometry

The Gauss-Bonnet theorem (GBT) links topology and geometry via the Euler characteristic $X$ and the integral of Gaussian curvature $K$.

A point on a sufficiently smooth surface has two principal curvature directions that correspond to the maximum and minimum normal curvatures $k_1$ and $k_2$. The Gaussian curvature $K$ corresponds to the product of the principal curvatures $K = k_1 k_2$. A positive Gaussian curvature corresponds to a local sphere-like surface, a negative Gaussian curvature corresponds to a local saddle-like surface, and a null Gaussian curvature corresponds to a plane or cylinder-like surface.

The GBT links topology and geometry in the case of closed surfaces as follows:

$$2\pi X = \iint_S K dA, \tag{2.11}$$

with $X$ the Euler characteristic, $K$ the Gaussian curvature and $dA$ a unitary element of the surface $S$. For surfaces with a boundary, a correction term is added:

$$2\pi X = \iint_S K dA + \int_{\partial S} k_g ds, \tag{2.12}$$

where $k_g$ is the geodesic curvature and $ds$ a unitary element of the boundary $\partial S$ of the surface $S$.

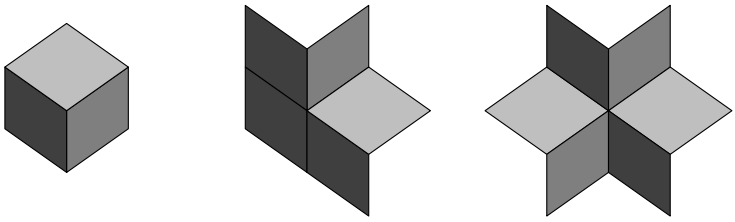
Combining the PHT and the GBT creates a link between pattern topology and geometry. Namely, a link connects indices and curvatures. Indeed, singularities with a positive index relate to positively curved areas, and the ones with a negative index to negatively curved areas.

This relation can be seen in the following expression of the discrete Gaussian curvature $K_v$ at a vertex $v$ in a mesh, derived from the GBT [Meyer et al., 2003]:

$$K_v = \frac{1}{A_v}(2\pi - \sum_{f \in F_v} d\theta_f), \tag{2.13}$$

where $F_v$ is the set of faces around the vertex $v$, $d\theta_f$ the angle of a incident face $f$ at the vertex $v$ and $A_v$ the total area of the adjacent faces. In a geometrically regular quad mesh with only $\pi/2$ angles between consecutive edges, the index and the curvature have the same sign. Figure 2.21 illustrates the relation between topology and geometry. A regular vertex adjacent to

four faces has a null curvature. A singularity with a valency lower than four has a positive index and curvature. A singularity with a valency higher than four has a negative index and curvature. Orthogonal networks of curves feature this right-angle property [Suris and Bobenko, 2008].



(a) Valency 3 – positive index and curvature

(b) Valency 5 – light negative index and curvature

(c) Valency 6 – strong negative index and curvature

Figure 2.21 – Relation between valency, index and curvature in quad meshes with right angles.

In structural design, a singularity is a natural way to introduce positive or negative curvature, as shown by the combination of the PHT and the GBT. Some structural systems, like elastic gridshells, are based on patterns of hexagons, triangles and quads with a high regularity of their geometry. These patterns feature equal spacing between the nodes to ease the construction process. Achieving curved geometries implies to release the equal-spacing constraint or to allow skewed faces. However, these distortions of geometry can be limited using properly placed singularities. The Dongdaemun Design Plaza in Seoul, South Korea, includes a five-valent singularity in the pattern of cladding in FIgure 2.22. This negative index singularity reduces the panel distortion in the negative-curvature area. Applications of this trade-off between geometrical and topological irregularity include woven structures [Martin, 2015, Ayres et al., 2018], elastic gridshells [Soriano et al., 2015, Masson, 2017, Avelino and Baverel, 2017] or auxetic materials [Konaković-Luković et al., 2018a, Konaković-Luković et al., 2018b].

## 2.5 Summary

Current strategies for topological design of quad-mesh patterns show several limitations. More specifically, exploration methods are missing for pattern

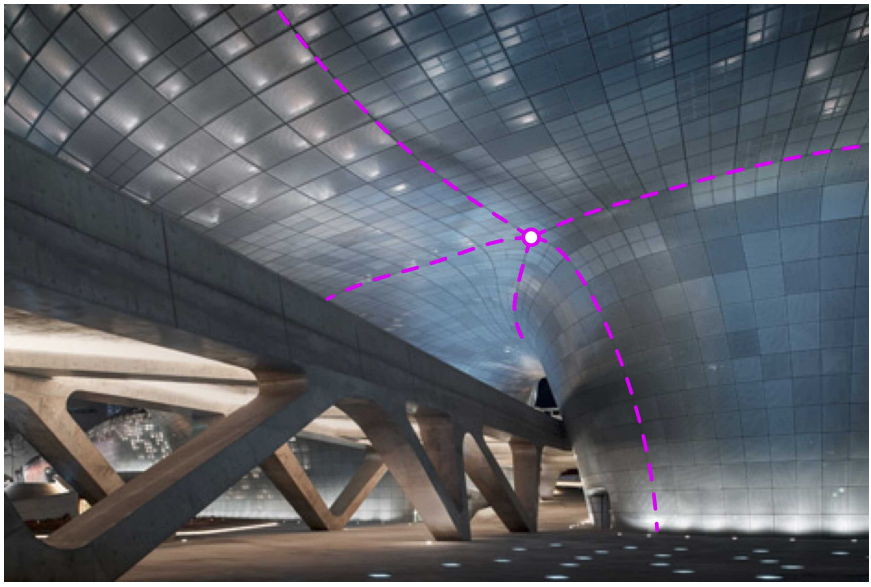


Figure 2.22 – The five-valent singularity in the cladding of the Dongdaemun Design Plaza in Seoul, South Korea, reduces panel distortion in the area with a negative curvature thanks to its negative index (source: dezeen.com).

singularities, despite their influential aspect on the design. Strategies exist for the exploration of topological spaces, which lack continuous-valued parameters. Such strategies rely on rule-based design using grammars and rules, potentially coupled with discrete-optimisation algorithms. Applications include a large variety of design topics, including structural design. However, the structure of meshes, and particularly quad meshes, results in several topological constraints to embed in the grammar, as opposed to unstructured networks. So far, such an approach does not exist for structural design of quad-mesh patterns, although it could support existing strategies.

The next chapter introduces the scope of this thesis with its research objectives, design approach and intellectual contribution for topological exploration of patterns in structural design.

# Chapter 3

# Thesis scope

If configuration processing provides a means to explore the geometry of space structures via a formex algebra [Nooshin, 1975, Nooshin, 2017], the aim of this thesis is to provide an algebra for the exploration of the topology of pattern of shell-like structures.

In Chapter 1, the context of structural design of quad-mesh patterns for shell-like structures has been set. In Chapter 2, background on topological design methods for patterns and the topology of quad meshes has been provided. This chapter provides the scope of this thesis. Section 3.1 highlights the research objectives. Section 3.2 presents the design approach. Section 3.3 summarises the intellectual contributions.

## 3.1 Research objectives

The topology of quad-mesh patterns in structural design, and particularly their singularity, influences all design aspects, aesthetics, structural, fabrication and construction. Indeed, the singularities define the density and geometry parameters available for exploration.

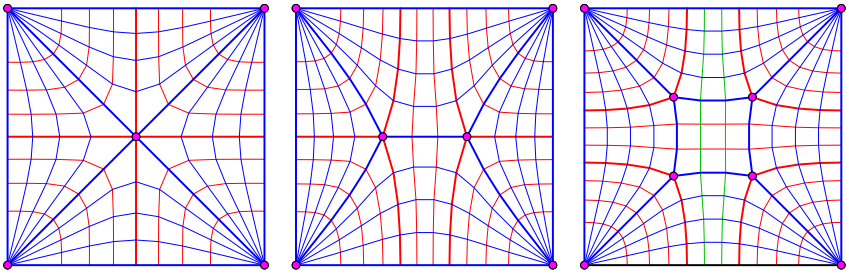
However, current design strategies for patterns in structures show limitations on the design, control and choice of singularities. Figure 3.1, revisiting Nervi's ribbed slabs, shows how different topologies impact different design aspects. In the context of multi-objective structural design, flexibility to accommodate several requirements is essential for the designer. However, existing strategies do not permit this.

Figure 3.1 presents three patterns for a four-sided plate supported on its

four corners under uniform loading. The patterns stem from the integration of the bending cross-field. The integration scheme yields three patterns due to slight numerical imprecision or geometrical asymmetry. They share similar features like the poles at the corners but have some dissimilarities with have pros and cons that depend on the considered structural system. The design in Figure 3.1a with its 8-valent singularity has the ideal topology for structural stiffness and preserves symmetry, which reduces the number of unique elements. However, high-valency nodes can be complex and expensive to build.

The design in Figure 3.1b with its two 6-valent singularities reduces the node complexity, although it does not preserve symmetry and deviates from the mechanically ideal design. The design in Figure 3.1c with its four 5-valent singularities deviates more from the mechanically ideal design but preserves symmetry and further reduces node complexity. However, the odd-valency induces a loss of the two-directional pattern structure, in red and blue, that stems from the cross-field. A third direction, in green, is necessary to avoid having two polyedges of the same colour overlapping each other. This property is essential for some structural systems with specific element organisations such as elastic gridshells with different layers of beams.

Depending on the structural system, the material and the construction technology, these designs with their similarities and dissimilarities can be more or less relevant. Moreover, quantifying the degree of similarity between two quad-mesh topologies would provide a metric for this design space to support exploration.



(a) One 8-valent singularity (b) Two 6-valent singularities (c) Four 5-valent singularities

Figure 3.1 – Three similar but different quad-mesh patterns offer different pros and cons for structural design.

Classically, exploration of topological spaces is tackled through rule-based design using grammar rules. Rule-based design performs beyond parametric design and its continuous-valued parameters. Structural design and optimisation already use such strategies. However, quad meshes have a specific structure that requires grammar of rules that preserve it.

Although common practice in other fields like computer graphics, topological modelling of quad meshes is not well spread in architecture, engineering and construction. Indeed, processing methods for geometry and topology mainly apply to dense and unstructured quad meshes. Moreover, they do not tackle the challenges of workflow integration the architect and the engineer have to face.

The goal of this thesis is to introduce and develop *topology finding* of patterns for structural design, focusing on singularities in quad meshes. Topology finding is meant as an analogy and as a complement to form finding and aims at reproducing its flexible design-oriented approach.

Topology finding performs beyond parametric design and optimisation, which include form finding, architectural geometry and shape optimisation, and beyond topology optimisation, which include discrete and continuous topology optimisation or cross-field integration. Figure 3.2 represents where topology finding stands in structural design. Topology finding does not aim at providing an optimised design in the topological space for specific objectives. Topology finding should offer a means for exploration as form finding offers it at the level of the geometrical space.

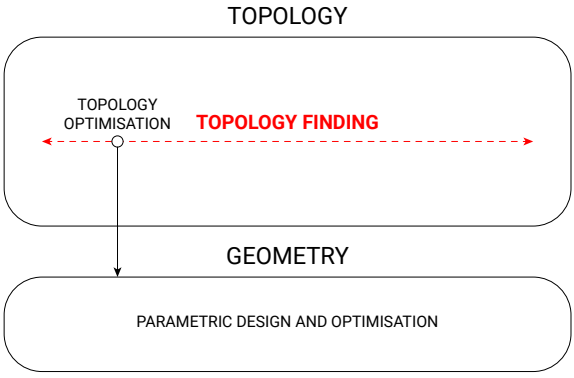



Figure 3.2 – Topology finding tackles topological design like topology optimisation but with the flexibility of parametric design at the level of geometrical design.

Nevertheless, topology finding is complementary to form finding and topology optimisation. Coupling these three approaches enable both exploration and optimisation of the geometry and the topology.

Three main objectives are expected from topology finding, illustrated as design directions in Figure 3.3:

1. **Comprehensive exploration**
   All designs must be accessible, in the realm of orientable compact manifold shapes. Any shape topology and any pattern topology. Exploration can then yield any potential mesh of the design space.

2. **Constrained exploration**
   Exploration must sort out the unfeasible designs if the required topological properties are not respected. Exploration can then be performed efficiently in the relevant design subspace.

3. **Informed exploration**
   Exploration must guide the designer towards efficient designs. Exploration informs the designer, who can interact with the machine during the design process, while the multiple design objectives evolve.

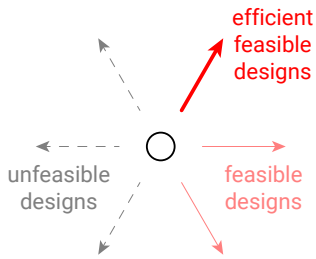



Figure 3.3 – The three objectives for topology finding of singularities in quad-mesh patterns in structural design: comprehensive exploration to reach all designs; constrained exploration to focus on feasible designs; and informed exploration to search for efficient designs.

The topology-finding algorithms tackling these research objectives are meant to apply at an early design stage for the exploration of a large variety of designs. The computation time of the algorithms should be fast enough for interactive design. This speed requirement can be assessed through comparison with other algorithms that occur later in the design process, like Finite

Element Analysis. Speed is vital for smooth user-machine interactivity and the automation or partial automation of the generation of numerous designs.

For such workflow flexibility and efficiency, the presented work follows a specific design approach.

## 3.2 Design approach

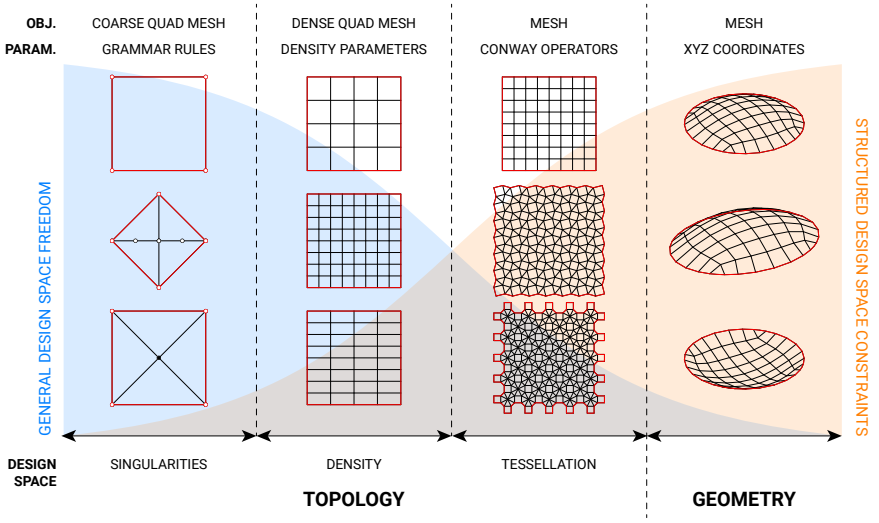



Figure 3.4 – General design space structure for exploration of the singularities in quad-mesh patterns. The design spaces feature a trade-off between generality and structuredness due to their objects and their parameters. The singularity design space is the most upstream one and therefore, the most influential one, the most general one but also the least structured one. Inspired by [Mueller, 2014, Mesnil et al., 2017a].

The different design aspects of patterns parameterised by quad meshes are decoupled and handled in different design spaces. This thesis focuses on the exploration of singularity design space. The exploration approach relies on a coarse quad mesh that encodes the data related to the singularities in the quad-mesh pattern. Such a strategy is common in computer graphics in mesh modelling environments to compute and model lighter meshes before applying smooth subdivision schemes like the Catmull-Clark subdivision

[Catmull and Clark, 1978]. Applying a few topological modifications on the coarse mesh is more efficient than applying multiple ones on a dense mesh. Some designers and researchers follow this approach for architectural and structural design, where the interest lies in performing design and optimisation on fewer global parameters [Shepherd and Richens, 2012, Bhooshan et al., 2018, Mesnil et al., 2017c].

The general design space follows the structure in Figure 3.4. From the most upstream to the most downstream one, the design space structure deals first with topology and second with geometry. Topology includes the singularities, the density and the tessellation of the pattern. The more upstream design spaces offer more exploration freedom but are less structured. The more downstream design spaces are more constrained but are more structured for exploration. Each design space defines the design parameters for exploration of the downstream ones. In this structure, the singularity design space is the most upstream one and therefore, the largest one and the most influential one. Nevertheless, exploration of this space relies on grammar rules, as opposed to continuous or discrete parameters.

1. **Singularity design space**
   Exploration of the singularity design space relies on a coarse quad mesh that defines the relationship between the singularities. Pseudo-quad faces allow including pole singularities. A pseudo-quad face has the geometry of a triangle and the topology of a quad. The exploration of such design space builds on a grammar of rules.

2. **Density design space**
   The coarse quad mesh yields a dense quad mesh after densification. Each strip of quad faces corresponds to a densification parameter. Therefore, exploration applies to non-null positive, discrete parameters. These multiple and independent density parameters are opposed to the unique global density parameter of the Catmull-Clark subdivision procedure [Catmull and Clark, 1978].

3. **Tessellation design space**
   The pattern topology comes from the potential application of Conway operators on the dense quad mesh, resulting in the equivalent density and singularities. The sequence of applied operators parameterises this design space. Other topological modifications can be applied, such as local edge operations like adding, deleting or trimming.

4. **Geometry design space**
   The coordinates of the vertices of the pattern serve as parameters to

the geometrical design space. Geometrical processing includes form finding or form optimisation, for instance. In this thesis, relaxation algorithms are necessary after geometry-blind topological processing that potentially yields a highly-distorted pattern or a pattern that does not respect boundary conditions [Williams, 2001, Williams, 2011]. Laplacian smoothing [Botsch et al., 2010] iteratively moves the vertices for a given number of iterations or until convergence below a given threshold displacement value. Each vertex of the pattern moves towards the centroid of its adjacent vertices at:

$$V_f = V_i + (1 - d)(\overline{V_i} - V_i), \tag{3.1}$$

where $V_f$ is the final position of the vertex, $V_i$ its initial position, $\overline{V_i}$ the centroid of its adjacent vertices, and $d$ a damping value between 0 and 1 for convergence stability, classically set to 0.5. Area-based weights take into account the common tributary area between the vertex and each of its neighbour when computing the centroid. Additionally, constraints are set on the vertices to project them back on objects like surfaces, meshes, curves and points after each iteration.

The design spaces are structured linearly because the upstream ones encompass the downstream ones. Nevertheless, the exploration of these design spaces is not necessarily linear. For instance, smoothing can be first applied on the lighter coarse quad mesh, then on the dense quad mesh, before application of Conway operators without further smoothing. Application of smoothing at different density levels tackles geometrical irregularities at different scales, or frequencies.

Following this design space structure, the contributions on topology finding apply on the coarse quad meshes that populate the singularity design space.

## 3.3 Intellectual contributions

The different intellectual contributions propose topology-finding algorithms. The flexibility of the design approach enables the designer to benefit from an interactive workflow with existing design and optimisation strategies.

Three encoding strategies are offered: geometry-, graph-and string-coded topology finding. They each provide complementary pros and cons for topology finding. This thesis introduces them from the most intuitive and high-level to the most fundamental and low-level.

- Geometry-coded topology finding: exploration is encoded with high-level geometrical parameters.
    - Chapter 4 introduces *feature-based topology finding*. A skeleton-based decomposition scheme allows generating topologies that respect alignment with boundaries. Additional point and curve features allow high-level indirect influence on the topology. This chapter contributes to comprehensive exploration of shape topologies, constrained exploration with feature integration and informed exploration thanks to heuristic features.
- Graph-coded topology finding: exploration is directly encoded with topological rules.
    - Chapter 5 presents *rule-based topology finding*. A grammar based on the strip of faces in quad meshes allows low-level editing of topologies. This chapter contributes to comprehensive exploration of pattern topologies.
    - Chapter 6 presents *similarity-informed topology finding*. A topological distance allows evaluating the similarity between topologies and the generation of hybrid topologies. This chapter contributes to informed exploration by combining heuristic designs stemming from any other method.
    - Chapter 7 presents *two-colouring topology finding*. The bipartite organisation in some structural system needs to respect a topological requirement related to two-colouring. A projection algorithm allows finding the subspaces of two-coloured topologies. This chapter contributes to constrained exploration with the integration of the two-colouring topological properties.
- String-coded topology finding: exploration is encoded with low-level alphabetical operations.
    - Chapter 8 presents *alphabet-based topology finding*. A string-encoding strategy of the strip rules opens towards evolutionary exploration based on the mutation of a genotype. This chapter contributes to comprehensive exploration with the possibility to generate any pattern topology.

This research is implemented in an open-source Python library called *compas_singular* [Oval, 2017]. compas_singular is a package of COMPAS

[Van Mele et al., 2017], an open-source, Python-based computational framework for research and collaboration in architecture, engineering, fabrication and construction. Chapter 9 provides further information on the implementation.

Chapter 10 summarises the contributions, their limitations and perspectives, before final conclusions.

# Part II

# Geometry-coded topology finding

# Chapter 4

# Feature-based exploration

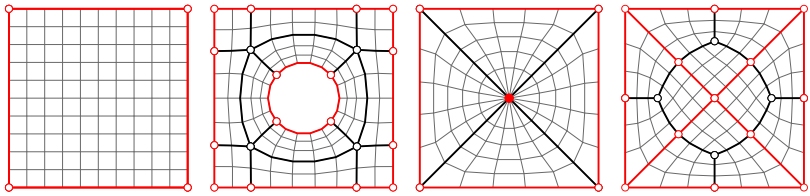
## 4.1 Motivations

Intents and constraints like a set of boundaries or a surface can drive or inform the design of a shell-like structure. The timber gridshell of the Solemar Therme in Bad Dürrheim, Germany aligns and concentrates the primary timber elements towards the oculi, which are supported by tree columns [Adriaenssens et al., 2014].

Integrating additional features can be substantial, like adding poles at the location of *point features* or aligning polyedges along with *curve features*. Nervi's slab for the Gatti Wool Factory and shell for the Palazzetto dello Sport in Rome, Italy, align the reinforcement rib pattern towards the columns and buttresses, respectively [Nervi, 1965]. The pattern of the NEST HiLo cable net is aligned with the facade for aesthetics and detailing [Méndez Echenagucia et al., 2018].

These geometrical features can stem from the static system with its support and load conditions. The features can also stem from geometry, like curvature concentrations or discontinuities. Designing a pattern whose topology respects these requirements, as shown in Figure 4.1, is not straightforward for non-experts and time-consuming for experts. The interest in a generation method for patterns that fit geometrical features is two-fold. Obtaining any shape topology for a pattern contributes to comprehensive exploration. Integrating design features in a pattern contributes to informed exploration.

This chapter introduces *feature-based topology finding*, using geometrical parameters to perform topological exploration. Section 4.2 develops an



(a) Outer boundary (b) Inner boundary (c) Point features (d) Curve features

Figure 4.1 – Integrating geometrical aspects like boundaries and features in the design of quad-mesh patterns.

algorithm to generate a coarse quad mesh decomposing a surface using its topological skeleton. Section 4.3 extends the algorithm to integrate point and curve features in the coarse quad mesh. Section 4.4 applies this generative method for feature-based topology finding. Point and curve features serve as geometrical design parameters for topological exploration.

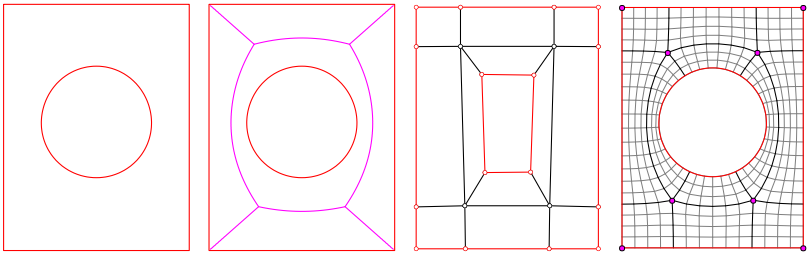
## 4.2 Surface decomposition

This section provides a method for decomposition of a surface into a pattern. Significant topological and geometrical aspects of the surface are the number of boundaries and the changes of curvature of the boundaries. The resulting mesh takes into account the topology and the geometry of the shape. The skeleton of the surface is used to find singularities away from the boundaries.

The skeletonisation and decomposition operations rely on the UV parameter space of the designed surface. Projection from the XY plane to the surface is not bijective if the surface overlaps or crosses itself. Nevertheless, mapping based on the UV parameterisation allows such flexibility on the input surface. The mapping of the surface is more general than projection and is therefore not constrained to height-field surfaces. The UV parameterisation influences the outcome and should represent the boundaries as faithfully as possible to yield a pattern that is not too distorted along the boundaries. For instance, conformal maps can yield strong distortions for highly-curved shapes [Botsch et al., 2010], but an angle-based flattening method can improve the results [Sheffer and de Sturler, 2001].

After skeletonisation and decomposition of the surface, the resulting coarse quad mesh can be densified and relaxed on the surface, as shown

in Figure 4.2. The pattern, thanks to the singularities stemming from the skeleton, is aligned with the boundary.



(a) Input (b) Skeleton (c) Decomposition (d) Mesh

Figure 4.2 – Skeleton-based decomposition of a surface in a coarse quad mesh that yields a quad mesh aligned with the boundary of the surface.

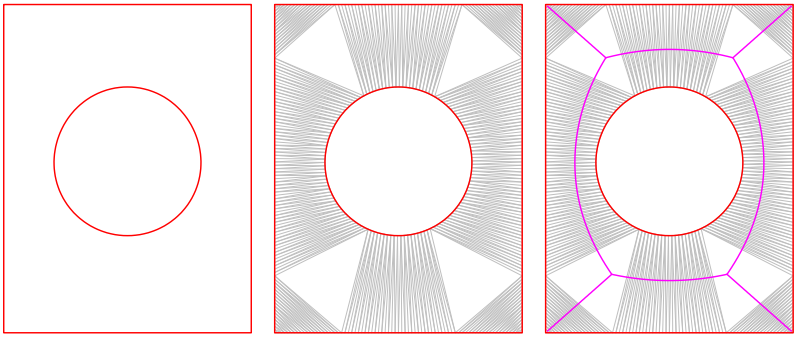
The methods and algorithms are further detailed.

### 4.2.1 Surface skeletonisation

The topological skeleton – or medial axis – is introduced by Blum [Blum, 1967] as a means to extract features from a surface. The skeleton provides dimension reduction of the surface into a set of curves. Still, the skeleton captures key aspects of the topology and the geometry of the surface. The skeleton is the locus of the points that are equidistant to several points on the boundaries. The skeleton consists mainly of points that are equidistant to two boundary points. The skeleton branches end at singular nodes that are equidistant to at least three boundary points. These singularities are away from the boundaries and even at the farthest possible location from them.

One of the strategies to generate a discrete skeleton of a surface is based on a Delaunay triangulation [Saha et al., 2016], as shown in Figure 4.3. A discrete approximation is sufficient for the decomposition and generation of a coarse quad mesh.

The surface is triangulated by a Delaunay mesh using points on the boundaries as vertices. The boundaries must be discretised into points densely enough to capture the relevant curvature changes but not too densely to avoid unnecessary heavy computation. Delaunay triangulation has a worst-case time complexity of $O(n \ln n)$ with a divide-and-conquer algorithm



(a) Input (b) Delaunay triangulation (c) Topological skeleton

Figure 4.3 – Generation of the topological skeleton of a surface by connecting the circumcentres of adjacent faces of a Delaunay triangulation.

[Leach, 1992]. The discretisation parameter $d_i$ for the number of points of each curve $i$ is the maximum value of:

- $d_{scale} = \lceil l_i/(\alpha D) \rceil$, the upper integer value of the length of the curve $l_i$ divided by a percentage $\alpha$ of a scale $D$ of the surface; and
- $d_{min}$, a minimum number of points:

$$d_i = \max(d_{scale}, d_{min}) \tag{4.1}$$

In practice, ranges of value of $d_{min}$ between 5 and 10 and of $\alpha$ between 0.01 and 0.05 of $D$ as the diagonal length of the total bounding box yield good results.

The Delaunay triangulation also deletes the faces lying outside the boundaries, to match the surface.

In the Delaunay mesh, three types of faces are distinguished depending on the number of adjacent faces:

- regular faces have two adjacent faces;
- singular faces have three adjacent faces; and,
- end faces have one adjacent face.

The segments connecting the circumcentres of all the pairs of adjacent Delaunay faces compose the skeleton. Groups of segments result in skeleton branches that end at singular faces and corner faces.

The topological skeleton serves as a basis for surface decomposition.

### 4.2.2 Skeleton decomposition

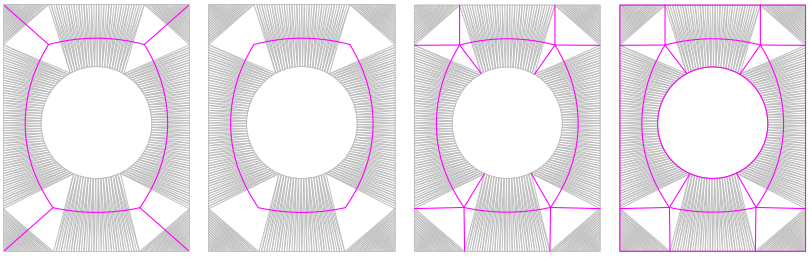
The topological skeleton separates regions of the surface, but these regions must be further decomposed into four-sided patches to obtain a coarse quad mesh, as shown in Figure 4.4.

Three topological operations are applied based on the connectivity of the Delaunay mesh:

1. pruning removes the branches connected to end faces;
2. grafting adds branches between the singular face circumcentres and their three vertices; and
3. closing adds boundary branches, split at the vertices of the singular faces and the two-valent boundary vertices.

At this stage, these operations have similar results to the ones in the work of Rigby [Rigby, 2003], yielding coarser quad meshes than other skeleton-based block decomposition approaches such as the ones of Fogg et al. [Fogg et al., 2016] used for meshing in Computational Fluid Dynamics.

The coarse quad mesh results from the connectivity of the four-sided patches.



(a) Skeleton (b) Pruning (c) Grafting (d) Closing

Figure 4.4 – The topological skeleton is converted into four-side patches by deleting and adding branches depending on their connectivity with the elements of the Delaunay mesh.

However, some modifications may be required to improve the quality of the decomposition.

### 4.2.3 Decomposition modification

Modifications on the set of patches are necessary to improve the resulting coarse quad mesh. These modifications ensure the validity of the topology and the geometry of the coarse quad mesh.

#### 4.2.3.1 Missed concavities

In Figure 4.5, the initial decomposition lost the concave kink in the surface boundary. On the contrary, the initial decomposition does integrate the four convex kinks. The skeleton marks convex but not concave kinks, and the added branches may not spot the concavities. Therefore, branches are added to include them in the decomposition, as illustrated in Figure 4.5. The kinks consist in the discontinuities of the boundary curves of the input surface.

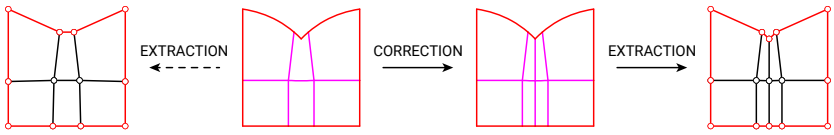



Figure 4.5 – Modifications of the decomposition for including the concavities missed by the topological skeleton by adding branches.

#### 4.2.3.2 Unwanted triangles

In Figure 4.6, the initial decomposition contains a triangle, although a quad decomposition is necessary to describe a quad mesh. If two adjacent singular faces have one or several coinciding vertices or a coinciding circumcentre, the missing branch yields triangular faces in the coarse mesh. Therefore, a branch is inserted to remove triangles from the decomposition, as illustrated in Figure 4.6. If two of the three patch corners are on the boundary, the branch is inserted at the other corner, and reciprocally if two of the three patch corners are off the boundary. Correcting concavities may yield unwanted triangles. Therefore, correcting unwanted triangles occurs afters correcting missed concavities.

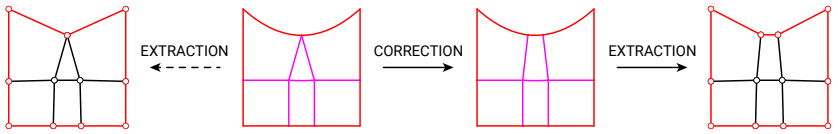



Figure 4.6 – Modifications of the decomposition for including the concavities missed by the topological skeleton by adding branches.

#### 4.2.3.3 Flipped patches

Strongly-curved branches can induce flipped patches in the coarse quad mesh with the right connectivity but with overlapping elements. Therefore, branches are added to refine flipped faces for geometrical quality, as illustrated in Figure 4.7. A face is flipped compared to its original patch if they have opposite normals $\boldsymbol{n_{face}}$ and $\boldsymbol{n_{patch}}$, respectively, checked thanks to the sign of their dot product:

$$\boldsymbol{n_{face}} \cdot \boldsymbol{n_{patch}} < 0. \tag{4.2}$$

After a patch subdivision, the check applies to the two resulting sub-patches. Correcting flipped patches requires a quad decomposition, so it occurs after correcting unwanted triangles.


Figure 4.7 – Modifications of the decomposition for refining the flipped patches with edge overlaps by adding branches.

#### 4.2.3.4 Collapsed boundaries

If less than three branches represent a boundary, the boundary collapses in the coarse quad mesh. Therefore, branches are added to subdivide the boundary, as illustrated in Figure 4.8. Adding branches for other corrections can solve collapsed boundaries. Therefore, correcting collapsed boundaries occurs last.

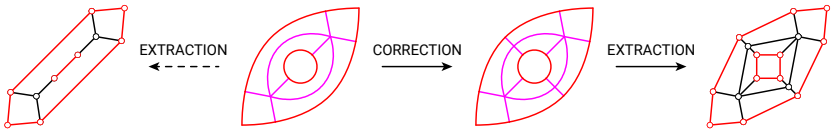



Figure 4.8 – Modifications of the decomposition for correcting the collapsed boundaries due to insufficient boundary branches by adding branches.

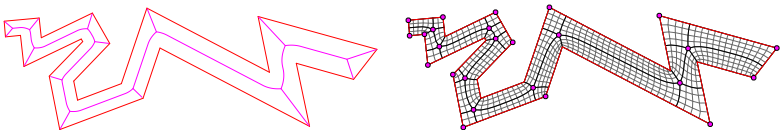
### 4.2.4 Examples

The skeleton-based decomposition is used to generate quad-mesh patterns for the roof of a couple of example projects: the Jewish Museum in Berlin, Germany in Figure 4.9, the Hiroshi Senju Museum in Karuizawa, Japan in Figure 4.10 and Solemar Therme in Bar Dürrheim, Germany in Figure 4.11. The boundaries are in red, the topological skeleton and the singularities in pink, the coarse quad mesh in black and the dense quad mesh in grey. The patterns integrate the geometrical features of the boundaries of the surface.

Distortions around the concave corners of the Jewish Museum show that the topology of the mesh is perfectible regarding geometrical regularity. For applications in Computational Fluid Dynamics, specific modifications improve this essential objective [Ruiz Gironès and Sarrate Ramos, 2007, Fogg et al., 2016]. Nevertheless, skeleton-based decomposition provides a starting topological design that captures the different characteristic of the surface.

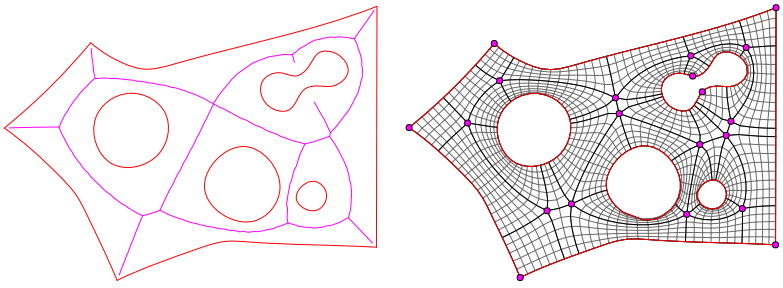
### 4.2.5 Non-null-genus shapes

This algorithm for skeleton-based decomposition does not apply to any shape topology. The shape must have zero handles $g$ and at least one boundary $N$ ($g = 0$, $N \geq 1$) so spheres and torus are excluded. This constraint is due to the Delaunay triangulation of the boundary points occurring on a planar map. Otherwise mapping would necessitate the integration of seams to map the shape.



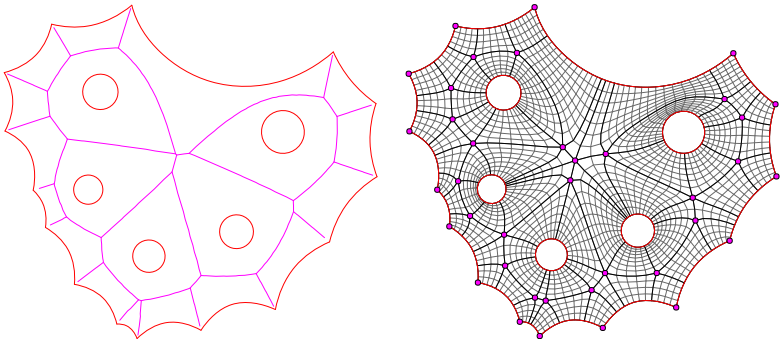
(a) Skeletonisation (b) Decomposition

Figure 4.9 – Revisiting the roof of the Jewish Museum in Berlin, Germany, with a quad-mesh pattern including 25 singularities.



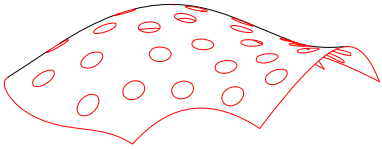
(a) Skeletonisation (b) Decomposition

Figure 4.10 – Revisiting the roof of the Hiroshi Senju Museum in Karuizawa, Japan, with a quad-mesh pattern including 20 singularities.

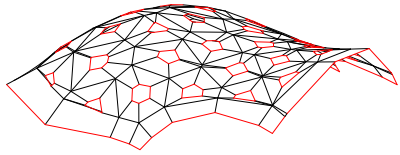


(a) Skeletonisation (b) Decomposition

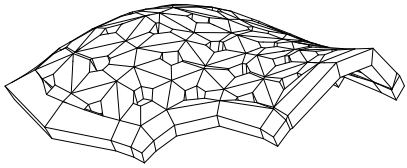
Figure 4.11 – Revisiting the roof of Solemar Therme in Bad Dürrheim, Germany, with a quad-mesh pattern including 38 singularities.



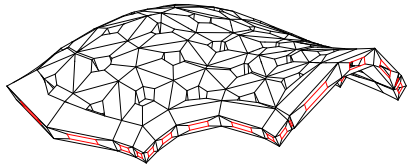
(a) Open null-genus medial surface



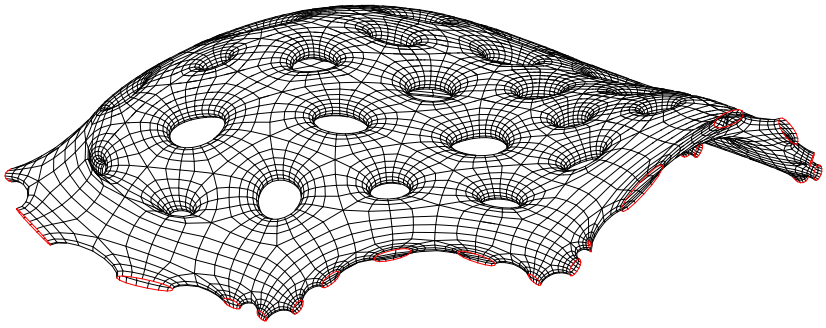
(b) Open null-genus coarse quad mesh after skeleton-based decomposition



(c) Closed non-null-genus coarse quad mesh after thickening



(d) Open non-null-genus coarse quad mesh after perforation



(e) Open non-null-genus quad mesh after densification and relaxation

Figure 4.12 – The generation of a high-genus pattern aligned with boundaries and handles is performed thanks to skeleton-based decomposition of a medial surface. The design is inspired by the ICD/ITKE Research Pavilion of 2015/16 [Alvarez et al., 2018].

Yet, an extension for shapes with any topology ($g \geq 0$, $N \geq 0$) is proposed. The ICD/ITKE Research Pavilion 2015-16 [Sonntag et al., 2017] serves as illustrative example with a sketch of its medial shape, shown in Figure 4.12a:

1. an open null-genus coarse quad mesh ($g = 0$, $N = a \geq 1$) is generated using the skeleton-based decomposition (Figure 4.12b);
2. the coarse quad mesh is thickened to obtain a closed non-null-genus coarse quad mesh ($g = a - 1 \geq 0$, $N = 0$) by offsetting it both directions and adding faces to join the boundaries together (Figure 4.12c);
3. the coarse quad mesh becomes an open non-null-genus ($g = a - 1 \geq 0$, $N = b \geq 0$) by perforating some of its faces (Figure 4.12d).

The resulting coarse quad mesh, and the pattern shown in Figure 4.12e after densification and geometrical processing, has a shape topology with a high number of handles and boundaries ($g = 33$, $N = 28$) for an Euler characteristic $X$ of -92. The pattern is aligned with the boundaries and the handles.

This approach requires the modelling of a medial surface. The results have the desired shape topology. Nevertheless, the pattern topology may not be suitable for shapes that are not faithfully represented by a medial surface due to high distortions. For instance, high-genus skeletal shapes are better to model via thickening of a network of lines [Srinivasan, 2004].

### 4.2.6 Computation time

The computation times for the skeleton-based surface decompositions of the presented examples are shown in Table 4.1. The discretisation of the boundaries corresponds to 3% of the length of the diagonal of the bounding box. The results compare the computation time for decomposition with the one for Delaunay triangulation, performed using the Python library NumPy. The computation time is low enough to allow interactive design with real-time editing of the surface, changing its geometry or adding openings.

The algorithm is deterministic as the same input surface always gives the same pattern topology – even though the Delaunay triangulation is not deterministic when co-circular vertices exist. The integration of additional geometrical features can drive the exploration of different pattern topologies constrained to a given surface. These features can stem from a design intent or various design aspects.

Table 4.1 – Computation times for skeleton-based surface decomposition compared with Delaunay triangulation.

| | triangulation [s] | decomposition [s] |
|---|---|---|
| British Museum (Figure 4.2) | 0.12 | 0.25 |
| Jewish Museum (Figure 4.9) | 0.16 | 0.38 |
| Hiroshi Senju Musem (Figure 4.10) | 0.11 | 0.37 |
| Solemar Therme (Figure 4.11) | 0.11 | 0.36 |
| ICD/ITKE Pavilion (Figure 4.12) | 0.16 | 0.87 |
| Rhön-Klinikum Campus (Figure 4.15) | 0.88 | 0.92 |

## 4.3 Additional features

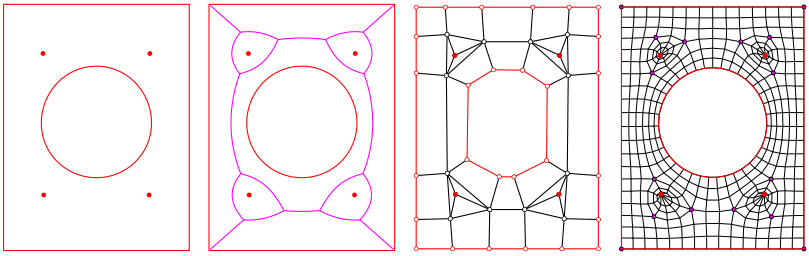
The same skeleton-based decomposition can integrate into the topology other important features represented as points or curves. Point features yield a pole singularity and curve features align the pattern in a specific direction. These geometrical features can stem from a design intent like a geometrical crease, a specific project feature like a point load. More importantly, these features allow the designer to influence the topology and explore different designs informed by design aspects.

### 4.3.1 Point features

Point features allow orientating the pattern around a pole, as shown in Figures 4.13 and 4.14.

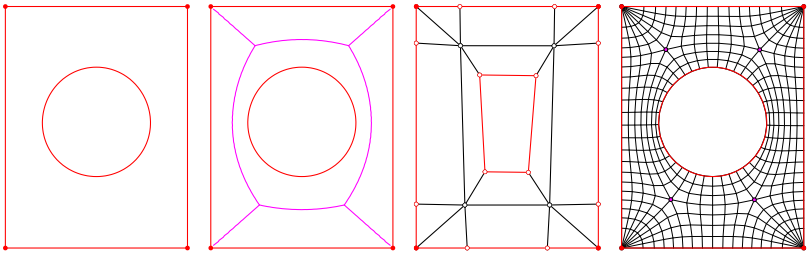
#### 4.3.1.1 Inner point features

Point features on the surface can be included in the coarse quad mesh to achieve a pattern featuring poles, as shown in Figure 4.13. The set of vertices for the Delaunay triangulation includes the points. The resulting skeleton



(a) Input (b) Skeleton (c) Decomposition (d) Mesh

Figure 4.13 – Skeleton-based decomposition of a surface in a coarse quad mesh to yield a pattern aligned with the boundary and point features off the boundary.



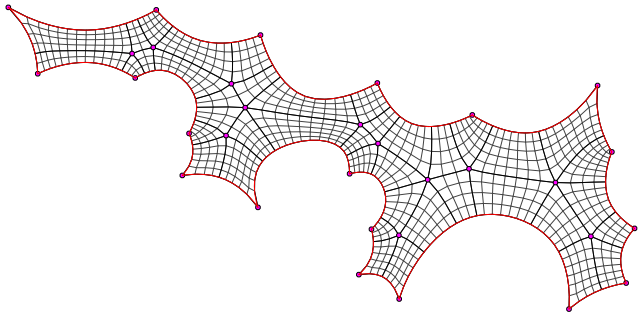
(a) Input (b) Skeleton (c) Decomposition (d) Mesh

Figure 4.14 – Skeleton-based decomposition of a surface in a coarse quad mesh to yield a pattern aligned with the boundary and point features on the boundary.

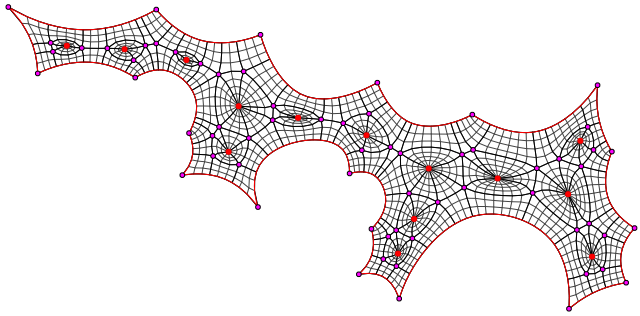
features new branches around the point features. Following the same modifications, the resulting coarse quad mesh yields pseudo-quad faces around the point features, resulting in a pattern with poles.

#### 4.3.1.2 Outer point features

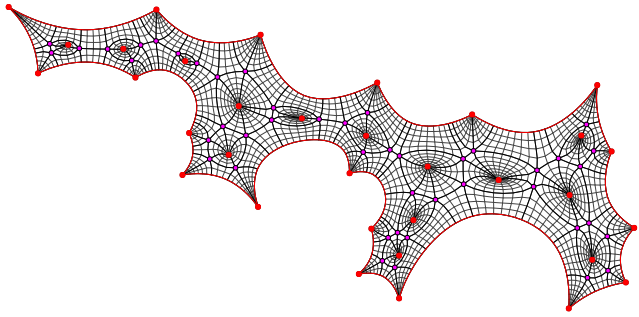
Point features lying on the boundary of the surface require additional modifications, as shown in Figure 4.14. Outer point features do not directly modify the coarse quad mesh since boundary points are already part of the Delaunay triangulation. Therefore, new edges split the quad faces adjacent to the point features. Each quad yields pseudo-quads in the coarse quad



(a) No point features, 31 singularities



(b) 14 point features at the inner mast supports, 78 singularities



(c) 33 point features at the inner mast and outer anchor supports, 78 singularities

Figure 4.15 – Revisiting the cable net of the Rhön-Klinikum Campus in Bad Neustadt, Germany, designed by Werner Sobek [Dürr, 2000], with skeleton-based generation of a quad-mesh pattern aligned with the boundaries and featuring poles based on the support conditions.

mesh by connecting the point feature to the face vertices to create boundary poles in the pattern.

The cable net of the Rhön-Klinikum Campus in Bad Neustadt, Germany, designed by Werner Sobek, is supported by the inner masts and the outer anchors [Dürr, 2000]. The cables are organised in an orthogonal quad grid, trimmed along the boundaries and around the supports. The pattern of the cable net is revisited using the skeleton-based decomposition, as shown in Figure 4.15. In Figure 4.15a, the pattern does not have trimmed elements and provides continuous cables along the unsupported boundary edges. In Figures 4.15b and 4.15c, the inner mast supports and the boundary anchor supports are successively added as point features to yield patterns that provide a high number of paths to them.

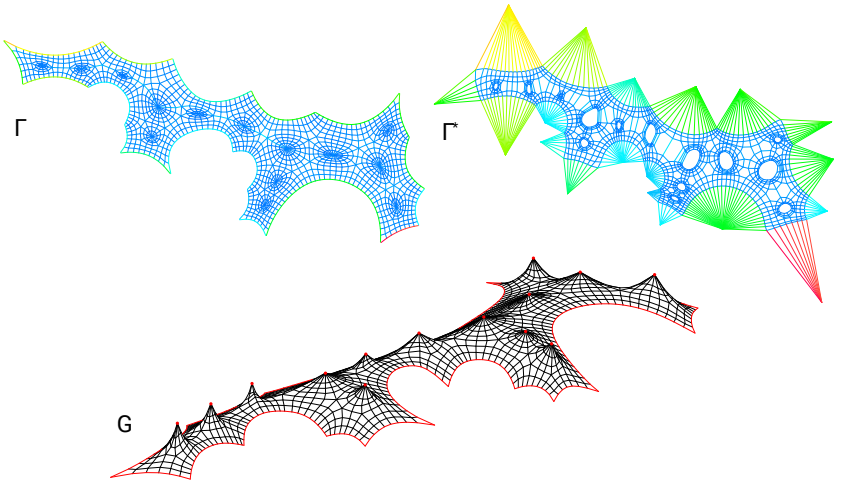


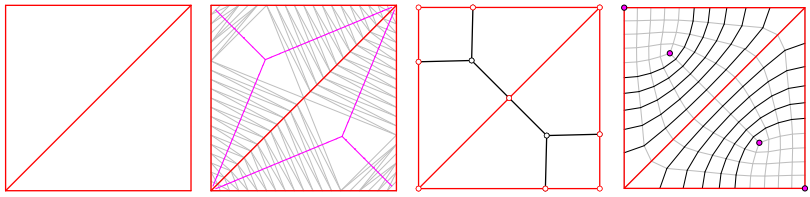

Figure 4.16 – Form diagram Γ, force diagram $\Gamma^*$ and form-found network $G$ for the Rhön-Klinikum Campus cable net using Thrust Network Analysis [Block and Ochsendorf, 2007] and RhinoVAULT [Rippmann et al., 2012, Rippmann and Block, 2013].

The structured boundary-and feature-aligned quad-mesh pattern provides a suitable input for funicular form finding with graphic statics using Thrust Network Analysis [Block and Ochsendorf, 2007] and RhinoVAULT [Rippmann et al., 2012, Rippmann and Block, 2013]. The form diagram Γ and force diagram $\Gamma^*$ in Figure 4.16 feature clear reciprocal areas for the edge arches and the pole hoops, providing a visual understanding of the

equilibrium in the structure to form find the tension-only network $G$.

### 4.3.2 Curve features

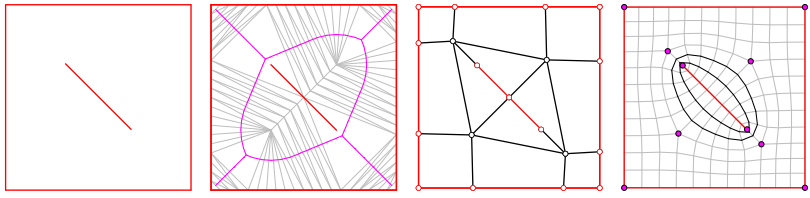
Curve features allow orienting a pattern along with several directions. A curve features can have its extremities on the boundary or not. The curve feature in Figure 4.17 as its two extremities on the boundary, which become three-valent boundary vertices. The resulting mesh provides a local alignment along with this curve feature, stressed in black.



(a) Input (b) Skeleton (c) Decomposition (d) Mesh

Figure 4.17 – Skeleton-based decomposition of a surface in a coarse quad mesh to yield a pattern aligned with curve features connected to the boundary.

The curve feature in Figure 4.18 as its two extremities off the boundary, which become two-valent vertices. The resulting mesh provides a local alignment around this curve feature, stressed in black. Curve features can be combined and superimposed.

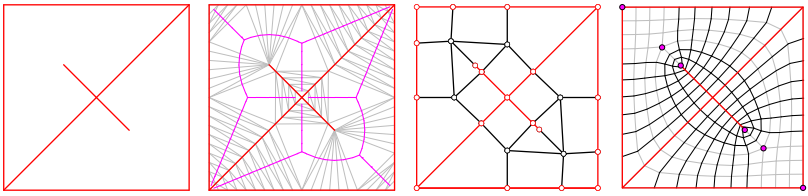


(a) Input (b) Skeleton (c) Decomposition (d) Mesh

Figure 4.18 – Skeleton-based decomposition of a surface in a coarse quad mesh to yield a pattern aligned with curve features disconnected to the boundary.

The curve features in Figure 4.19 are the same as in Figures 4.17 and

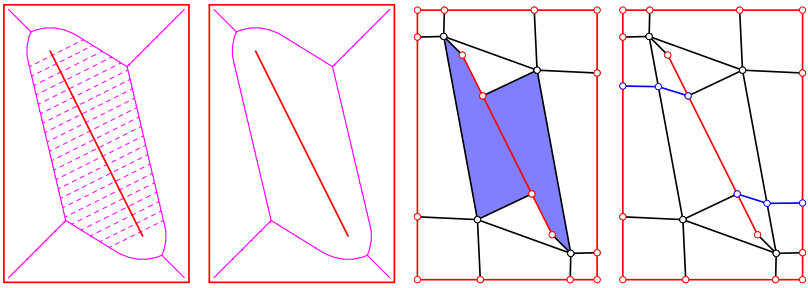
4.18. The resulting mesh provides alignment with both curve features as a hybrid between the two previous ones, stressed in black.



(a) Input (b) Skeleton (c) Decomposition (d) Mesh

Figure 4.19 – Skeleton-based decomposition of a surface in a coarse quad mesh to yield a pattern aligned with multiple curve features.

Additional steps to the original method are required, as illustrated in Figure 4.20. A set of points subdivide the curve features with their connecting edges set as constraints for the Delaunay triangulation [Chew, 1989].



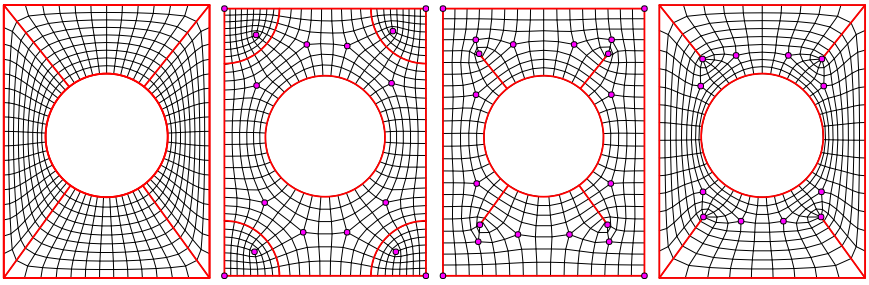
(a) Skeleton without cut (b) Skeleton with cut (c) Decomposition (d) Quad decomposition

Figure 4.20 – Additional steps for skeleton-based generation of coarse quad meshes including curve features. Curve features are unwelded to avoid the crossing skeleton branches in dashed pink curves. Pentagonal or higher-valency faces, in blue, resulting from the unwelding become quad faces by propagating the seams of the discrepancies on the curve features.

Topological cuts are made in the Delaunay mesh along the curve features to consider them as boundaries. Unwelding the curve features removes adjacency between faces on each side of the curve features, which would otherwise yield branches crossing the curve features as in Figure 4.20a instead of Figure

4.20b. After generating the skeleton and applying the same modifications, the topological cuts in the Delaunay meshes induce discrepancies creating polygonal faces, highlighted in blue in Figure 4.20c. The resulting polygonal faces become quad faces by propagating the seams of the discrepancies on the curve features, highlighted in blue in Figure 4.20d.

A curve extremity that lies on the boundary does not feature a singularity, as in Figures 4.21a and 4.21b. A curve extremity that does not lie on the boundary features a singularity. Either a two-valent singularity if adjacent to one singular face, as in Figure 4.21c, or a pole if adjacent to several singular faces, as in Figure 4.21d. A pole as in Figure 4.22a becomes a two-valent singularity as in Figure 4.22b due to the corrections for missed corners and unwanted triangles.



(a) (b) (c) (d)

Figure 4.21 – Exploring the topology of a quad-mesh pattern using skeleton-based decomposition with different curve features.

(a) A pole thanks to the missed-concavity operation

(b) A two-valent singularity thanks to the unwanted-triangle operation

Figure 4.22 – Modifying curve feature extremities between a pole and a two-valent singularity.

Point and curve features can serve as geometrical design parameters for topological exploration of the singularities in quad meshes.

## 4.4 Feature-based topology finding

Skeleton-based decomposition serves as a generative algorithm for feature-based topology finding of patterns. Point and curve features drive the topology of the pattern. These geometrical design parameters directly modify and influence the orientation of the pattern by modifying its singularities. Point and curve features can stem from design heuristics like the support conditions. The present application revisits the design of the steel gridshell of the British Museum.

### 4.4.1 Design problem

Figure 4.23 shows the geometry, the original pattern and the support conditions of the British Museum Courtyard Roof. The geometry is defined by Williams [Williams, 2001]. The pattern of the steel gridshell is a triangular mesh, but this study considers quad meshes with stiff nodes. Therefore the original pattern is not considered. Williams [Williams, 2011] relaxes some quad meshes on the same surface to reconcile the square and the circle, but these patterns result from the trimming of a dense regular grid and include boundary singularities only. Thrust is only permitted at the four corners, as the shell is supported along its boundary by sliding bearings to avoid applying thrust on the existing building. The study discards any additional stiffening system. Therefore, the analysis focuses on the difference of structural behaviour between patterns due to their singularities. Engineering and construction details are found in Sischka *et al.* [Sischka et al., 2001].

The beams of the actual structure have a box cross-section with a width of 80 mm and a height varying from 80 mm to 200 mm, oriented with the surface normal [Sischka et al., 2001]. Here, the S355 steel beams all have the same cross-section to favour patterns with a homogeneous force flow for the given support conditions. The beams must be stiffer 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. A complementary analysis could study the influence of the stiffness of the boundary edge. The boundary is not a free edge, as a vertical reaction applies. However, the boundary does not take thrust and stiffening

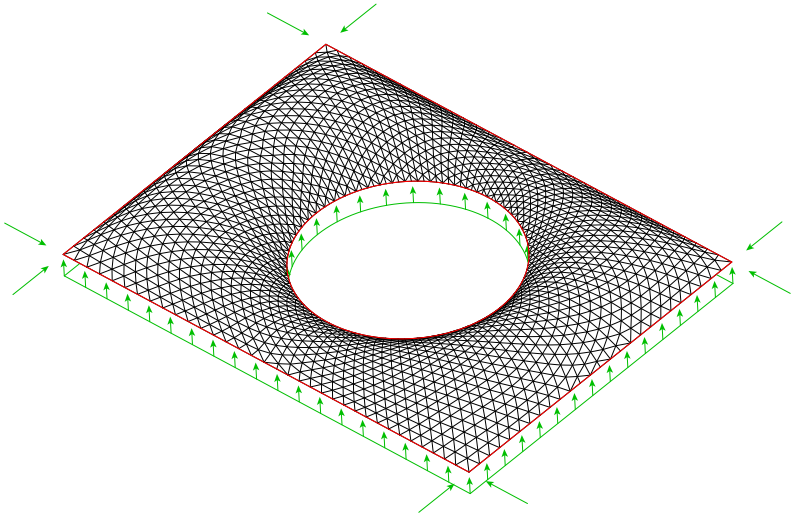


Figure 4.23 – Geometry, original pattern and support conditions of the roof of the Great Courtyard of the British Museum in London, England [Williams, 2001, Sischka et al., 2001]. The triangular-mesh gridshell is supported all along its boundaries on sliding bearings with thrust admitted at the four corners only.

the edge beam improves the structural behaviour of gridshells [Venuti and Bruno, 2018].

Several structural requirements apply to different load cases and combinations.

The load cases are:

- the structural self-weight G;
- a downwards dead load G'=0.6kN/m$^2$ for a 24 mm thick glazing [Sischka et al., 2001];
- a downwards projected live load Q=0.5kN/m$^2$ for snow loads, without taking into account geometry factors;

and 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 [Sischka et al., 2001];
- a maximal ULS stress utilisation of 100%;
- a minimal ULS first load buckling factor of 4, as for the actual structure [Sischka et al., 2001].

The pre-deformation, as well as the imperfections, based on the first buckling mode with a maximal value of 140 mm [Sischka et al., 2001], are not taken into account.

A second-order mechanical analysis is performed, using the Finite Element Analysis plugin Karamba for Grasshopper3D [Preisinger, 2013]. The cross-section optimisation is performed using Constrained Optimisation by Linear Approximation (COBYLA) [Powell, 1994, Powell, 1998] implemented in the NLOpt library for non-linear optimisation[Johnson, 2015]. Clune [Clune, 2013] and Mesnil *et al.* [Mesnil et al., 2018b, Mesnil et al., 2017a] use this library for structural shape optimisation, also made available in the parametric design environment Grasshopper thanks to the plugin goat. A penalty function allows to formulate the constrained-optimisation problem into an unconstrained one with the energy:

$$\frac{m}{m_0} + k(\max(0, \frac{f - f_{max})}{f_{max}}) + \max(0, \frac{u - u_{max}}{u_{max}}) + \max(\frac{0, b_{min} - b)}{b_{min}})) \quad (4.3)$$

with $m$ the weight and $m_0$ the initial weight for a cross-section height of 500 mm; $k$ a penalty coefficient set to $10^6$; $f$ and $f_{max}$ the measure and the criterion of maximum deflection; $u$ and $u_{max}$ the measure and the criterion of maximum stress utilisation; $b$ and $b_{min}$ the measure and the criterion of minimum first load buckling factor.

### 4.4.2 Pattern design

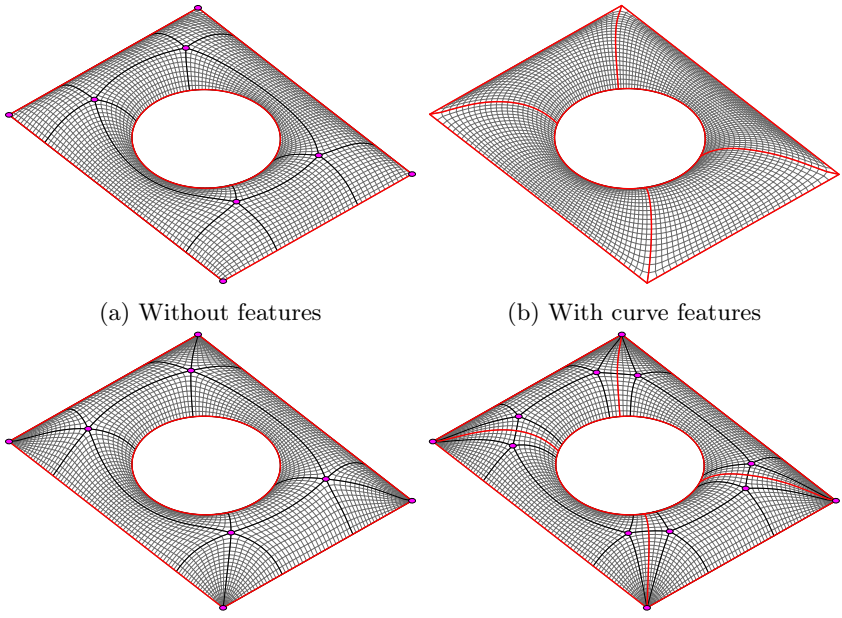
For the design of the patterns, topology stems from project-related considerations, density from a target length and geometry from the shape intent.

The quad-mesh pattern stemming from the stress field in Figure 2.2 is not taken into account as Schiftner and Balzer assume a fixed boundary [Schiftner and Balzer, 2010].

The support conditions of the roof incite to provide direct load paths to the corners. As they admit thrust, the corners are the stiffest parts of the shell. Two heuristic features are identified, supported by the analysis of the principal stress directions:

1. point features at the corners to concentrate force paths;
2. curve features along the longest spans towards the corners to align force paths.

Each of the two features can be applied or not, yielding the $2^2 = 4$ designs shown in Figure 4.24.



(a) Without features (b) With curve features

(c) With point features (d) With curve and point features

Figure 4.24 – Four quad-mesh patterns for the British Museum courtyard roof including some heuristic point and curve features to influence their topology.

The coarse quad meshes are densified resulting in quad-mesh patterns with an average beam length of 1.26 m to 1.34 m (6% difference) for a total of 7140 to 8264 beams (15% difference). The membrane stiffness of quad

gridshells depends on the edge length [Lebée and Sab, 2013, Mesnil et al., 2017d]. Therefore, the edge length must be similar in each design to evaluate the influence of the singularities in the pattern.

The quad meshes are relaxed on the actual surface as a design constraint using area-weighted Laplacian smoothing with constraints to re-project the vertices on the surface, its boundaries and its corners [Botsch et al., 2010, Williams, 2001, Williams, 2011].

### 4.4.3 Numerical results

Table 4.2 shows the numerical results from structural analysis and sizing optimisation. The performance metric is the ratio of the structural mass to the covered area of the gridshell. The pattern without features serves as a reference to assess the influence of the heuristic features on structural efficiency.

The ratio of structural mass per covered area is high. This lack of efficiency is due to the combination of the static system, the geometry and the quad grid that favour a bending behaviour, as opposed to a membrane behaviour.

Stress is the most decisive requirement. The choice for a unique cross-section penalises the designs that do not provide a uniform force flow for the stress utilisation of the beams. The maximum stresses occur around the corners. The limiting elements are either the hoops or the rays depending on the topology. Buckling is the least decisive requirement because the design favours a bending behaviour instead of a compression behaviour. However, buckling becomes closer to be decisive for the most efficient designs.

The most efficient designs are the ones with point features only and both curve and point features, with a weight decrease of 39%. The worst design is the one with curve features only, with a weight increase of 35%. The curve features reduce the deflection by orienting multiple load paths towards the corners but increase the stress utilisation. The point features reduce the stress utilisation by connecting multiple load paths directly to corners.

Taking into account the support conditions via heuristics such as point and curve features when designing the topology of the pattern can improve the structural behaviour and reduce the structural weight.

Table 4.2 – Structural performance after sizing optimisation of the heuristic patterns for the British Museum courtyard roof.

| | (a) without features | (b) with curve features | (c) with point features | (d) with curve and point features |
|---|---|---|---|---|
| Parameters | | | | |
| edge number [-] | 7140 | 7896 | 7364 | 8264 |
| beam height [mm] | 430 | 600 | 170 | 150 |
| Performance | | | | |
| ratio structural mass to covered area [kg/m$^2$] | 343 (100%) | 464 (135%) | 208 (61%) | 210 (61%) |
| Constraints | | | | |
| max SLS deflection [mm] | 123 | 81 | 137 | 114 |
| max ULS stress utilisation [-] | 0.99 | 0.99 | 0.96 | 0.99 |
| first ULS buckling load factor [-] | 27.9 | 40.6 | 6.3 | 4.7 |

## 4.5 Summary of contributions

This chapter introduced feature-based topology finding, using geometrical parameters to perform topological exploration:

- a skeleton-based decomposition algorithm was developed to generate patterns that align with the boundaries of a surface;
- an extension was added to integrate point and curve features in the topology of the pattern. These geometrical features are natural design parameters in architectural engineering, as illustrated by built examples;
- the relevance of features stemming from heuristic consideration was validated on a case study.

This topology-finding algorithm allows rapid computing and topological generation using geometrical parameters. Feature-based topology finding enables topological optimisation when coupled with any shape optimisation and performance analysis. Indeed, the number and position of the point and curve features can serve as encoding parameters for optimisation algorithms.

Boundary alignment provides aesthetic, structural and fabrication benefits. However, releasing locally this boundary-alignment constraint can provide orientation freedom. Therefore, such an extension allows improving geometrical regularity or alignment with a vector-or cross-field. Lyon *et al.* [Lyon et al., 2019] identify thirteen configurations for sub-quads that combine alignment and non-alignment along the boundaries of patches on a surface obtained using so-called motorcycle graphs [Campen et al., 2015]. An extension of this skeleton-based decomposition could provide such flexibility by considering only one part of the boundaries for alignment, creating a surface based on these boundaries for skeleton-based decomposition and trimming the resulting pattern along the other part of the boundaries.

The skeleton-based decomposition algorithm has some limitations for non-null-genus shapes. An extension adapted the algorithm for shapes with handles, modelled via shapes with corresponding boundaries. This assumption is suitable for shell structures, faithfully modelled as seamless surfaces, but significant geometrical distortions can occur for general envelopes. Adaptation of the algorithm, by adding seams for planar mapping, for instance, would release this constraint.

Approaching topology finding through geometrical features that can stem from design aspects of the project is intuitive for the designer. Constrained exploration allows integrating boundaries, curves or points, making important changes on the set of singularities. However, the topology is controlled indirectly through these geometrical parameters, and it is not guaranteed to yield any possible topology. Therefore, a complementary means for exploration is necessary to allow comprehensive exploration of the design space, which does not encode high-level geometrical parameters but low-level of topological rules.

# Part III

# Graph-coded topology finding

# Chapter 5

# Rule-based exploration

## 5.1 Motivations

Feature-based topology finding allows topological exploration with geometrical parameters. Although influencing the topology with curve and point features is intuitive, this approach does not give a direct control over the topology, which makes comprehensive exploration of the design space more challenging. Figure 5.1 shows how to combine point and curve features in the skeleton-based decomposition to obtain different designs. These geometrical parameters yield two of the three variations for the Nervi slabs shown in Figure 3.1. The choice of point and curve features to obtain the third one is not as intuitive. However, the three variations have similar topologies, and one could serve as a starting point for the generation of the other ones. Moreover, topological modifications are lighter than geometrical processing. Using the skeleton-based decomposition once before applying topological modifications is more efficient than resorting to the skeleton-based decomposition for the generation of each design. These aspects question the systematic use of a geometry-based generation means, instead of combining it with topological modifications.

The singularity design space relates to topology. On the contrary to a geometry-related space, a topological design space is not defined by continuous numeric parameters. Nonetheless, grammars allow for exploring topological spaces. Grammars consist of finite sets of rules that perform topological modifications.

The rules control the accessible design space for exploration. The higher

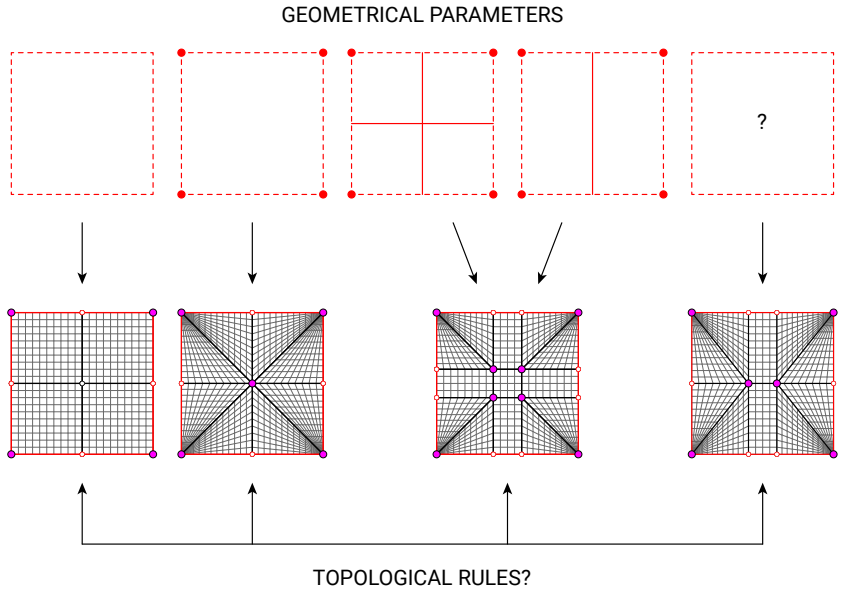



Figure 5.1 – Point and curve features in red serve as geometrical parameters in the skeleton-based decomposition for the generation of different designs inspired by the variations of Nervi slabs from Figure 3.1. The generation of these designs is not as intuitive, despite its topological similarity with the other designs.

the number of rules, potentially the larger the accessible design space but the more complex the use of the grammar. A low-level grammar allows performing comprehensive exploration with a small number of rules.

Prior work on quad-mesh grammars aims at improving the regularity of dense quad meshes, regarding both topology and geometry [Daniels et al., 2008, Daniels II et al., 2009, Tarini et al., 2010, Peng et al., 2011]. The proposed grammars apply to groups of varying numbers and structures of quad faces. These grammars are tailored for specific optimisation objectives and do not allow non-linear, comprehensive exploration. Indeed, most rules are subtractive and do not allow the generation of any quad-mesh topology. Moreover, these grammars include many rules. Restraining the grammar to fundamental operations like addition and deletion would reduce the number of rules without reducing the accessible design space. Nevertheless, these rules must apply to the right structure in a quad mesh.

This chapter introduces *rule-based topology finding*, using topological rules to perform topological exploration. Section 5.2 identifies a low-level strip structure in quad meshes. Section 5.3 introduces a corresponding low-level grammar of strip rules for quad-mesh singularities. Section 5.4 applies this grammar for rule-based topology finding for multi-objective design of patterns.

## 5.2 Quad mesh structure

General polygon meshes consist of faces with any number of vertices. Editing the connectivity of faces enables topological exploration, allowing changes in the number of vertices per face. However, quad meshes are by definition constrained to faces with exactly four vertices. A quad-mesh-specific topological operation must respect this constraint. The strips of quad faces provide a suitable structure in quad meshes for rules to perform non-linear, comprehensive exploration.

The following considerations on the structure of quad meshes apply to pseudo-quad meshes as well. These meshes include pseudo-quad faces, which have the geometry of triangles but the topology of quads. One of their vertices is a pole singularity, equivalent to a double vertex with a collapsed edge. As shown in Figure 5.2, a pseudo-quad mesh stands as a pseudo-quad mesh with poles as virtual boundary edges for topological exploration.

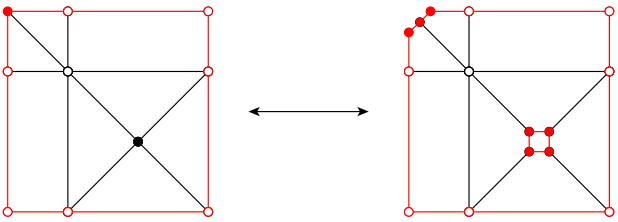


Figure 5.2 – Interpreting a pseudo-quad mesh as a quad mesh with poles as virtual boundary edges.

### 5.2.1 Quad mesh strips

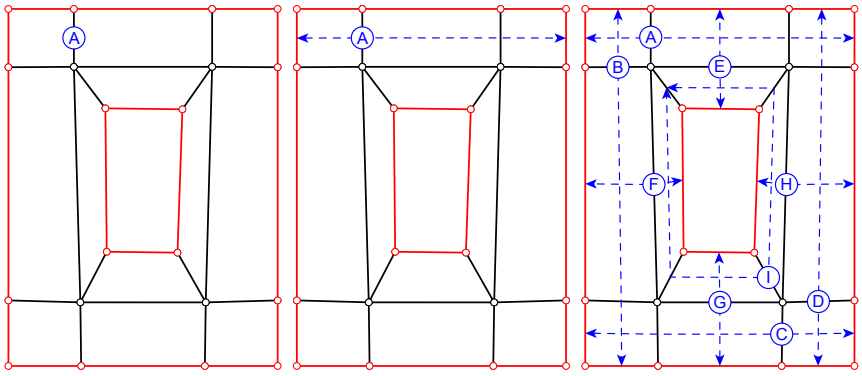
Quad meshes contain a structure of strips of quad faces. These strips constitute a description of quad meshes at a larger scale than faces themselves. Strips already apply in some ways for digital [Campen et al., 2012, Campen

and Kobbelt, 2014] or physical [Akleman et al., 2010, Akleman et al., 2016] modelling approaches, also referred to as loops, rings or chords.

The strips correspond to the independent parameters for densification of a quad mesh.

Quad mesh strips depend on the topology of a quad mesh, not its geometry. 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 5.3:

1. start with an initial complete list of edges;
2. initiate the collection of a strip by getting one edge from the list (Figure 5.3a);
3. complete the strip by adding the edges across the adjacent quad faces in both directions (Figure 5.3a). Termination occurs when collection yields two boundary edges in the case of open strips (strips A to H), or until it forms a loop in the case of closed strips (strip I);
4. store the strip data as the list of collected edges and remove them from the initial list of edges;
5. repeat from step 2 until the initial list of edges is empty (Figure 5.3c).




(a) Get edge (b) Collect strip (c) Repeat

Figure 5.3 – Collecting the strip data in a quad mesh as lists of opposite edges across the quad faces.

The strip data structure consists of lists of edges, an edge being a pair of vertices. Each edge appears exactly once. The strip data structure can convert into a list of faces, potentially more suitable depending on the use. Each face appears precisely twice, as shown in Figure 5.4 with an exploded map of the nine strips and their face data. A face can appear in the same strip if the strip crosses itself.

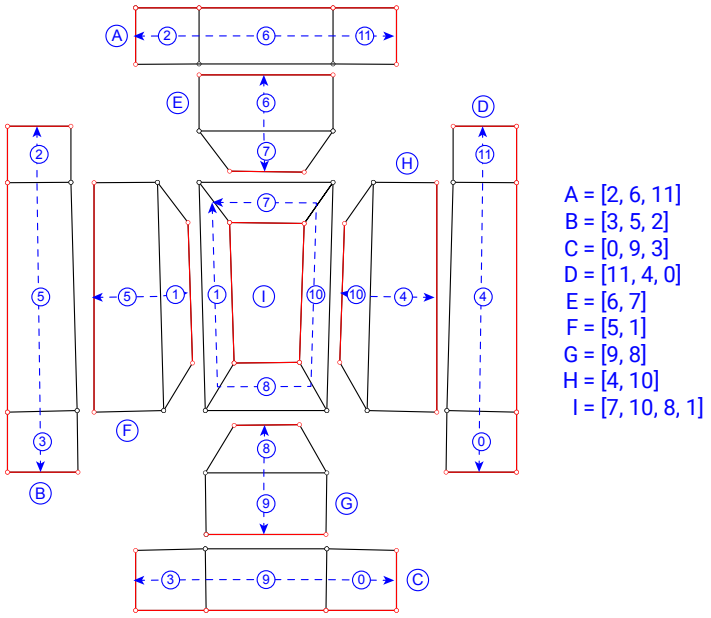



Figure 5.4 – Exploded map of the strips with their face data.

If the strip has a pole extremity, the strip data encodes the pole as a pseudo-edge starting and ending at the pole. Poles being virtual boundary edges, they interrupt strips.

### 5.2.2 Counting strips

How does the number of strips relate to the number of vertices, edges and faces, and can it be computed without the strip-collection algorithm?

The number of strips $S$ corresponds to the number of open strips $S_o$ and the number of closed strips $S_c$:

$$S = S_o + S_c. \tag{5.1}$$

The number of open strips $S_o$ in a quad mesh actually derives directly from the number of edges $E$ and the number of faces $F$.

The number of edges $E$ is divided into the number of boundary edge $E'$ and the number of non-boundary edge $E''$:

$$E = E' + E''. \tag{5.2}$$

Each boundary edge corresponds to one of the two extremity edges of an open strip, therefore:

$$S_o = E'/2. \tag{5.3}$$

Summed across the four edges for all faces, boundary edges occur once and non-boundary edges occur twice:

$$4F = E' + 2E'', \tag{5.4}$$

which becomes thanks to Equation 5.2:

$$4F = 2E - E', \tag{5.5}$$

which allows to express Equation 5.3 as:

$$S_o = E - 2F. \tag{5.6}$$

A different approach yields this relation. Figure 5.5 shows that when initially considering one open strip for each edge, this number is reduced by two for each face. Indeed the two pairs of opposite edges in a quad face each connect two open strips or close an open strip.

This relation is in agreement with the case of closed meshes. These meshes consist of closed strips only. Due to the lack of boundary, each edge is adjacent to two faces, therefore:

$$E = 2F. \tag{5.7}$$

No relation is provided here to count the number of closed strips $S_c$ and know the total number of strips $S$. Nevertheless, the number of strips and the number of closed strips relate to each other as:

$$S - S_c = S_o = E - 2F. \tag{5.8}$$

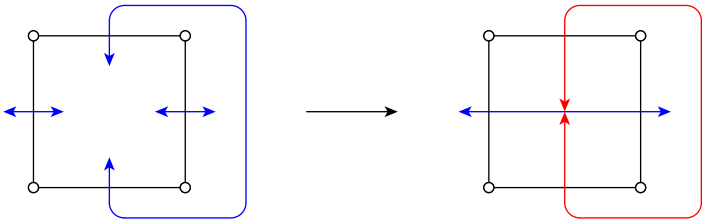


Figure 5.5 – Counting open strips $S_o$ as one per edge $E$ minus two per face $F$: $S_o = E - 2F$. Starting with one strip per edge, the connectivity of quad faces removes two open strips by merging open strips (in blue) or closing open strips (in red) for each pair of opposite edges.

The relation in Equation 5.6 is similar to equations on geometrical and mechanical properties of structures.

First, it relates to the number of degrees of freedom $D$ for parallel transformations, applied to planar-mesh offsetting [Pottmann et al., 2007b, Mesnil et al., 2017e], up to the three translation degrees of freedom:

$$D = E - 2F + 3. \tag{5.9}$$

Second, it relates to the extension of Maxwell's rule [Maxwell, 1864] by Calladine [Calladine, 1978] for computing the difference of states of self-stress $s$ and the number of mechanisms $m$ for a dual 2D structure of bars:

$$s - m = E^* - 2F^* + 3. \tag{5.10}$$

Indeed, a primal and a dual graph have the same number of edges $E^* = E$ and dual numbers of vertices and faces $V^* = F$ and $F^* = V$. The three extra parameters correspond to the 3 degrees of freedom for rigid-body motions in 2D. To find $s$ and $m$, Pellegrino [Pellegrino, 1993] resorts to Singular Value Decomposition of an equilibrium matrix in which $s$ is the dimension of the nullspace. This approach also applies to algebraic graphic statics [Van Mele and Block, 2014].

### 5.2.3 Strip graph

A graph encoding the strip connectivity compresses the data of quad-mesh strips. The strip connectivity consists of the crossings between the strips over the faces.

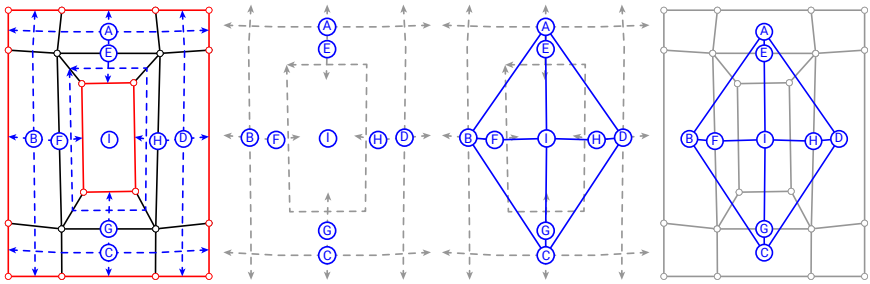
The strip graph is constructed as illustrated in Figure 5.6:

1. collect the strips in the quad mesh (Figure 5.6a);

2. for each mesh strip, a graph vertex is added at the centroid of the mesh strip: $S_{mesh} = V_{graph}$ (Figure 5.6b);

3. for each mesh face, a graph edge is added connecting the two crossing strips, potentially the same strip in the case of self-crossings: $F_{mesh} = E_{graph}$ (Figure 5.6c).

The quad mesh and the strip graph, as shown in Figure 5.6d, feature a different type of duality from the one between a primal and a dual mesh, where a vertex corresponds to a face $V_{primal} = F_{dual}$, and reciprocally $F_{primal} = V_{dual}$, and an edge correspond to an edge $E_{primal} = E_{dual}$.




(a) Quad mesh elements (b) Graph vertices (c) Graph edges (d) Strip graph

Figure 5.6 – Construction of the strip graph of a quad mesh.

The regular quad-mesh grid in Figure 5.7 illustrates the increase in the number of graph elements with the density. An $N \times N$ grid mesh has $2N$ strips and $N^2$ faces, therefore its strip graph has $2N$ vertices and $N^2$ edges.

The graph encodes some but not all of the topological data. Indeed, two topologically different meshes can have the same strip graph. There is no isomorphism between the two. Therefore, the strip graph is a simplified object derived from the quad mesh. Moreover, the question is open whether for any graph corresponds a quad-mesh strip structure or what are the conditions for it.

Based on this fundamental strip structure, the next section introduces a low-level grammar for the quad-mesh singularities.

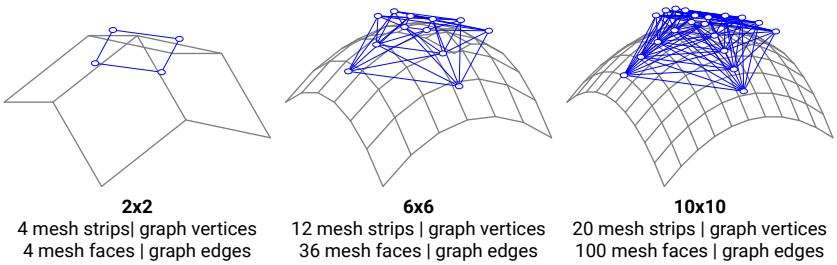



Figure 5.7 – The number of elements in the strip graph increases with the density of the quad mesh. An $N \times N$ grid mesh has $2N$ strips and $N^2$ faces, and its strip graph has $2N$ vertices and $N^2$ edges.

## 5.3 Quad-mesh grammar

A quad-mesh grammar with two low-level rules for *addition* and *deletion* of strips is introduced. This grammar is purely topological, independent from geometry. Figure 5.8 illustrates the two reciprocal rules for different configurations of strips with the insertion of strips along a polyedge and collapse of a strip into a polyedge, in blue.

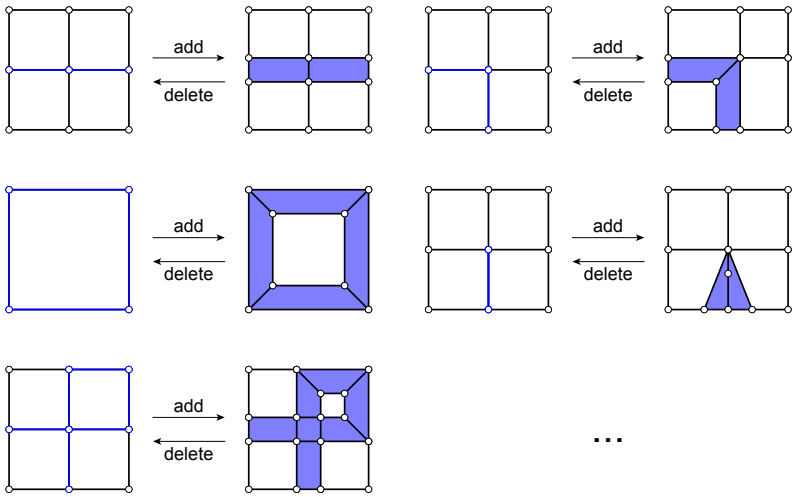



Figure 5.8 – Quad-mesh grammar of low-level rules for addition and deletion of strips with different configurations.

This grammar allows non-linear exploration as any rule can be undone by its reciprocal rule. Moreover, this grammar allows comprehensive exploration. Indeed, the rules can transform any quad mesh into another. An inefficient manner consists in first adding the strips of the final quad mesh, and second, deleting the strips of the initial quad mesh. For a polyedge to be the reciprocal of a strip, it must have both extremities on the boundary or close on itself. Nevertheless, self-crosses and self-overlaps are allowed. This grammar also applies to pseudo-quad meshes thanks to their quad-mesh interpretation with poles as virtual boundary edges. The quad mesh can have any shape topology, with multiple boundaries and handles or cross-caps.

The implementation of the rules is further detailed.

### 5.3.1 Strip addition

To add a strip along a polyedge $(V_0 - ... - V_i - ... - V_{n-1})$, the following operations are sequentially applied on the polyedge, as illustrated in Figure 5.9:

1. 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. $\forall i \in [\![1; n-3]\!]$, 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. 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 $V_{n-1}''$, respectively, are created and the last edge $V_{n-2} - V_{n-1}$ yields directly a quad face $[V_{n-2}', V_{n-1}', V_{n-1}'', V_{n-2}'']$.

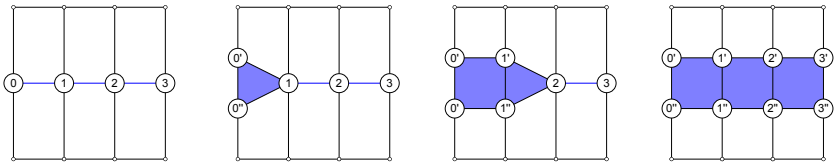



Figure 5.9 – Detailed addition of a strip along a polyedge.

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 convention is defined by the sense of the polyedge and the normal of the vertices.

Finally, the disconnected old vertices are deleted.

Visually, the polyedge is unzipped to become a strip. The temporary pseudo-quads have their poles oriented downstream the polyedge. In Figure 5.10, a strip is added along polyedge $A - B - C$ without modification of the singularities, only the density, because the polyedge exactly follows an existing strip. In Figure 5.11, a strip is added along polyedge $A - B - C$ inducing new 3-and 5-valent singularities.

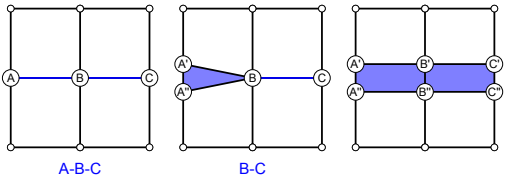



Figure 5.10 – Add a strip changing the density.

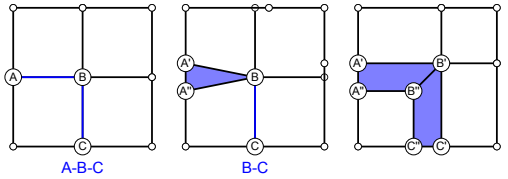



Figure 5.11 – Add a strip changing the singularities.

Some modifications are necessary for the addition rule to apply to strips in any configuration.

To add a closed strip, marked with °, along a closed polyedge $(V_0 - ... - V_i - ... - V_{n-2} - V_0°)$, 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 5.12 along polyedge $A - B - C - D°$.

A closed polyedge $(V_0 - ... - V_i - ... - V_{n-2} - ... - V_0)$, without °, marks the addition of an open strip. With $V_0$ 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:

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

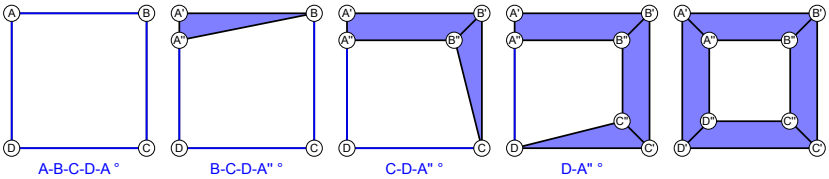



Figure 5.12 – Add a closed strip.

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 a breadth-first-search [Eppstein, 2007] 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 5.13, 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 $C - D - E - B' - B'' - F$.

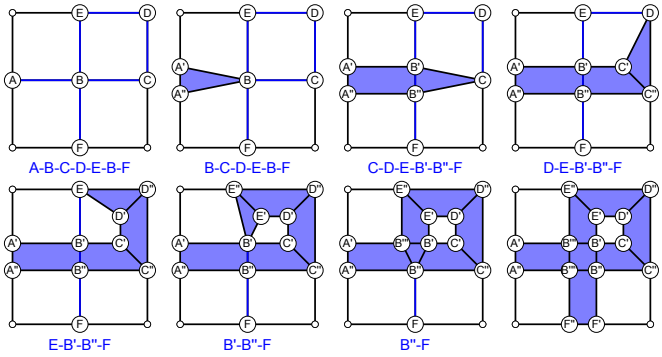



Figure 5.13 – Add a self-overlapping strip with update of the polyedge.

In Figure 5.14, 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 provide the same result.

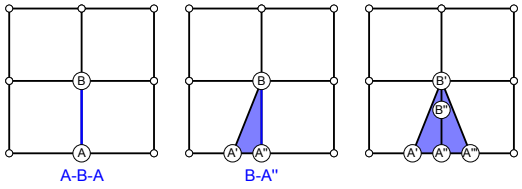



Figure 5.14 – Add a self-crossing strip with update of the polyedge.

In order to add a strip with poles 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 5.15. 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.

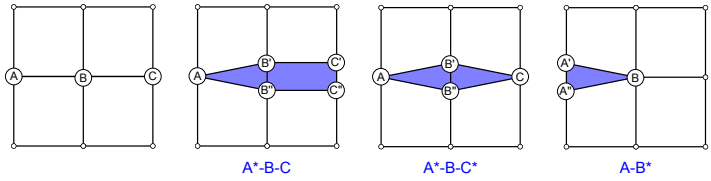



Figure 5.15 – Add strips with poles.

### 5.3.2 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 5.16:

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.

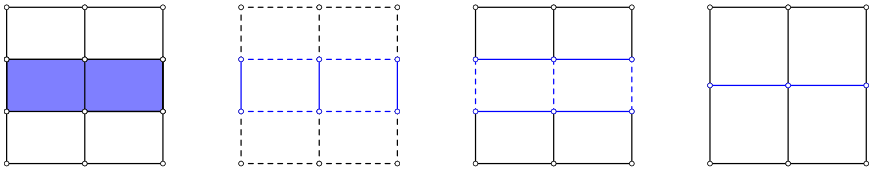


Figure 5.16 – Delete a strip.

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 5.17.

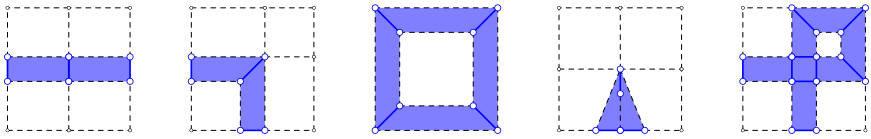


Figure 5.17 – Disconnected graphs of vertices to merge for the strip deletion rule.

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, though this grammar should only modify the pattern topology. Figure 5.18 illustrates how this problem is solved, by refining some strips.

Before deleting a strip, a verification predicts potential boundary collapse if:

$$|E_{strip} \cap E_{boundary}| < 3, \tag{5.11}$$

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 edge to remain to refine. Indeed, the minimum number of boundary edges is three, and the maximum number of boundary

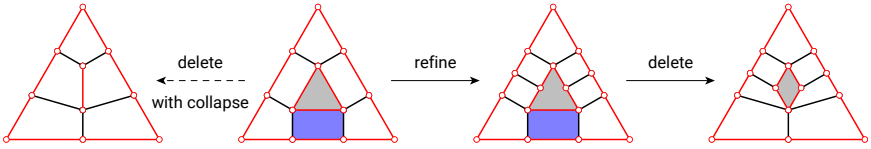



Figure 5.18 – When deleting some strips, in blue, refining other strips avoids boundaries to collapse to less than three edges and close an opening, in grey.

strip edges is two. Refining a strip does not add singularities, it only modifies its density. The addition rule in Figure 5.10 creates a new strip on a polyedge along the existing strips. Therefore, the rule does not modify the set of singularities, only the density, opposed to the rule in Figure 5.11, for instance, which does not fully follow an existing strip. Such addition rules apply in Figure 5.18. To avoid the collapse of the boundary marked in grey due to the deletion of the strip in blue, the remaining strips that end at this boundary are refined. One of the two polyedges along these strip serves 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.

### 5.3.3 Strip data update

After the application of each rule, the strip data updates. Re-collecting all strips is not efficient, as only the added or deleted strip and the ones crossing the modified faces need to be updated. 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.

If 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 deleted edges $(V_i', V_i'')$, which become $(V_i, V_i)$, are removed;
- 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 $(V_i, V_{i+1})$.

If 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 connected to $V_j$.

### 5.3.4 Strip graph relation

Adding or deleting a strip in a quad mesh have corresponding operations on its strip graph.

In Figure 5.19, adding strip $E$ along the polyedge crossing strips $A$ and $C$ induces a new vertex $E$ in the graph connected to vertices $A$ and $C$. Deleting strip $E$, deletes vertex $E$ and edges $E-A$ and $E-C$ in the graph.

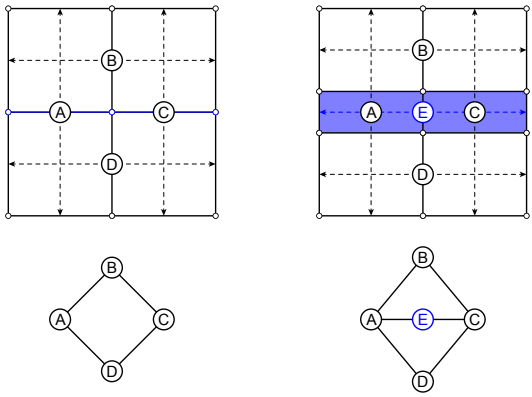


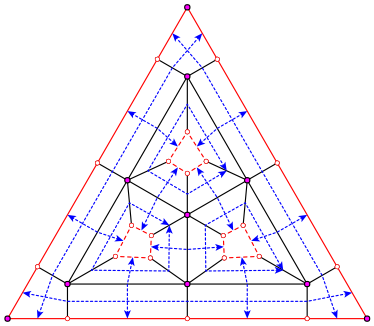

Figure 5.19 – Corresponding strip graph operations due to the addition and deletion of the quad-mesh strip E.

Adding a mesh strip along a polyedge induces addition of a graph vertex connected to the graph vertices of the strips crossing the polyedge. Deleting a mesh strip with its faces induces deletion of the graph vertex with its edges.

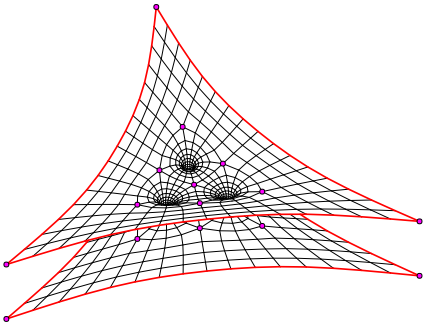
### 5.3.5 High-genus example

This approach applies to manifold quad meshes with any shape topology. The quad mesh in Figure 5.20 has two boundaries and two handles. The

coarse quad mesh in Figure 5.20a shows a part of the strip structure and the dense quad mesh in Figure 5.20b shows the full pattern, for clarity. The dashed red boundaries represent the connection to the other part of the mesh. Figure 5.21 shows the spatial strip graph for such a topology.



(a) Partial coarse quad mesh. Dashed boundaries connect to the other part of the coarse quad mesh.



(b) Quad-mesh pattern

Figure 5.20 – Example of the strip structure in a quad-mesh pattern with a non-null genus.

The grammar rules also applies to such meshes. The quad meshes in Figures 5.22 and 5.23 result from the quad mesh in Figure 5.20 after the addition of three and nine strips in blue, respectively. The three quad-mesh patterns have the same genus but different sets of singularities.

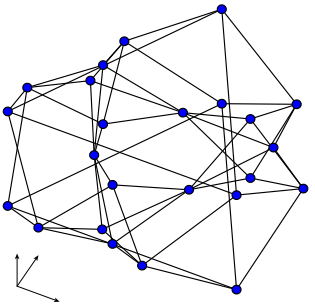
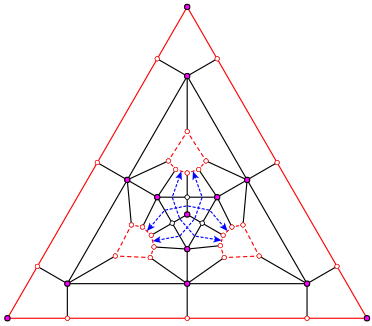
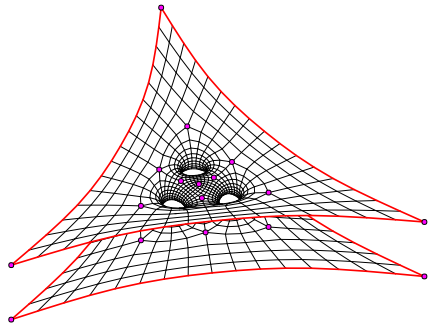


Figure 5.21 – Example of the strip graph in a quad-mesh pattern with a non-null genus.

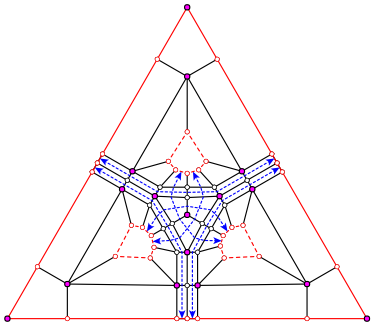


(a) Partial coarse quad mesh. Dashed boundaries connect to the other part of the coarse quad mesh.

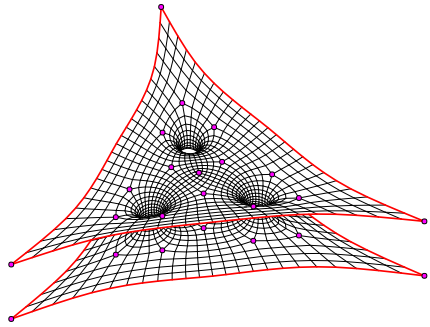


(b) Quad-mesh pattern

Figure 5.22 – Adding three strips, in blue, to a non-null-genus mesh.



(a) Partial coarse quad mesh. Dashed boundaries connect to the other part of the coarse quad mesh.



(b) Quad-mesh pattern

Figure 5.23 – Adding nine strips, in blue, to a non-null-genus mesh.

Thanks to this quad-mesh grammar, a designer can perform rule-based topology finding of quad-mesh patterns.

## 5.4 Rule-based topology finding

The designer can apply the addition and deletion of strips on a coarse quad mesh to explore quad-mesh patterns. Instead of geometrical parameters, topological rules allow topological exploration. The strip parameterisation guarantees that any quad-mesh topology is achievable while preserving the shape topology. The low-level rules guarantee non-linear exploration. Indeed, a rule can be cancelled by its reciprocal rule without resuming exploration.

### 5.4.1 Design problem

The same design problem as in Section 4.4 on feature-based topology finding is investigated: the Courtyard Roof of the British Museum.

However, all designs have the same unique box cross-section with a height of 500 mm, a width of 250 mm and a thickness of 20 mm. The statics-related metrics are the maximum SLS deflection $F$, the maximum ULS stress utilisation $U$ and the inverse of the ULS first load buckling factor $B$. These metrics are normalised by the structural weight.

Fabrication-related metrics are added for multi-objective design of steel and glass gridshells [Mesnil et al., 2017b].

A first metric is the regularity of the beams. The variation of beam lengths must be minimised to avoid fabrication of too long or too short elements. The metric on edge length disparity $L$ is the standard deviation of the edge length in the mesh.

A second metric is the curvature of the panels. Curved panels must be minimised to avoid expensive bending processes. The metric on face curvature $C$ is computed as:

$$C = \frac{1}{A_{mesh}} \sum_{face} A_{face} C_{face}, \tag{5.12}$$

with $A_{mesh}$ the mesh area, $A_{face}$ the face area and $C_{face}$ the curvature of the face as:

$$C_{face} = \frac{dL}{(L_1 + L_2)/2}, \tag{5.13}$$

where $L_1$ and $L_2$ are the lengths of the diagonals of a quad face and $dL$ the shortest distance between them.

A third metric is the straightness of the panels. Skewed panels must be minimised to reduce material loss when cutting the panels. The metric on face skewness $S$ is computed as:

$$S = \frac{1}{A_{mesh}} \sum_{face} A_{face} S_{face}, \tag{5.14}$$

with $S_{face}$ the face skewness as:

$$S_{face} = \max\left(\frac{\theta_{max} - 90}{90}, \frac{90 - \theta_{min}}{90}\right), \tag{5.15}$$

where $\theta_{min}$ and $\theta_{max}$ are the minimal and maximal angles in degrees between two consecutive edges in the quad face.

### 5.4.2 Pattern design

Here, topological exploration is performed freely by combining addition and deletion strip rules starting with the topology stemming from the skeleton-based generation procedure without features. The generation process follows the graph in Figure 5.24. From design 0 directly result seven designs (1 to 7). From design 2, a denser design 2' allows then the generation of three designs (8 to 9). Finally, a succession of four designs (11 to 14) stem from design 6.

Figure 5.25 presents the resulting designs.

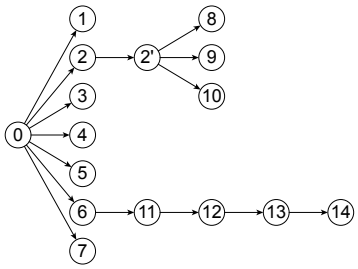



Figure 5.24 – Graph of the generation process although exploration is open and non-linear. Fourteen new designs stem from design 0 in a combination of tree branches of different width and depth.

Although presenting the generation process as a unidirectional tree, the grammar allows non-linear exploration thanks to the reciprocal rules. Indeed, different permutations of the same set of rules applied to the same

initial design can generate the same final design. For instance, topology 13 can come from topology 11 by first deleting strips A and C and then deleting strips B and D, resulting in another intermediary topology than topology 12.

The resulting quad-mesh patterns with different topologies are shown in Figure 5.26, after densification and relaxation as in Section 4.4.

### 5.4.3 Numerical results

The numerical results are displayed in Table 5.1. The number of edges and their total length show the density disparity. The six metrics $\overline{X}$ on statics and fabrication result from normalisation by their respective maximum:

$$\overline{X_i} = X_i / \max_i X_i, \tag{5.16}$$

where $X_i$ is the value of the metric $X$ for the design $i$. The lower the metric, the more efficient the design regarding the metric. The best and worst designs per metric are highlighted in Table 5.1 in green and red, respectively. The values of the minimum, maximum, average and standard deviation provide statistics on these results. The results exclude design 2' because it has the same singularities as design 2.

Design 1 performs the best for two of the three fabrication-related metrics: edge length disparity and face curvature skewness. Design 14 performs the best for two of the three statics-related metrics: maximum deflection and stress utilisation. Its topology is the same as the one resulting from skeleton-based decomposition with curve and point features. Nevertheless, all designs feature different performances to explore.

In multi-objective optimisation, the Pareto front $P$ is made of the designs that are non-dominated. To be non-dominated, a design must perform at least as well for each objective and perform better for at least one objective than each of the other design [Deb, 2001]:

$$x \in P \iff \forall y, \forall i, x_i \leq y_i, \exists i / x_i < y_i, \tag{5.17}$$

where $x_i$ is the performance of design $x$ for the metric $i$ to be minimised.

The Pareto front comprises designs 0, 1, 3, 4, 6, 8, 9, 11, 12, 13 and 14, underlined in Table 5.1 and Figure 5.26. Only 2, 5, 7 and 10 are dominated designs, meaning that at least one of the other designs performs at least the same for all metrics and better for one metric. All of the other design are worth considering as they offer different trade-offs between the multiple objectives.

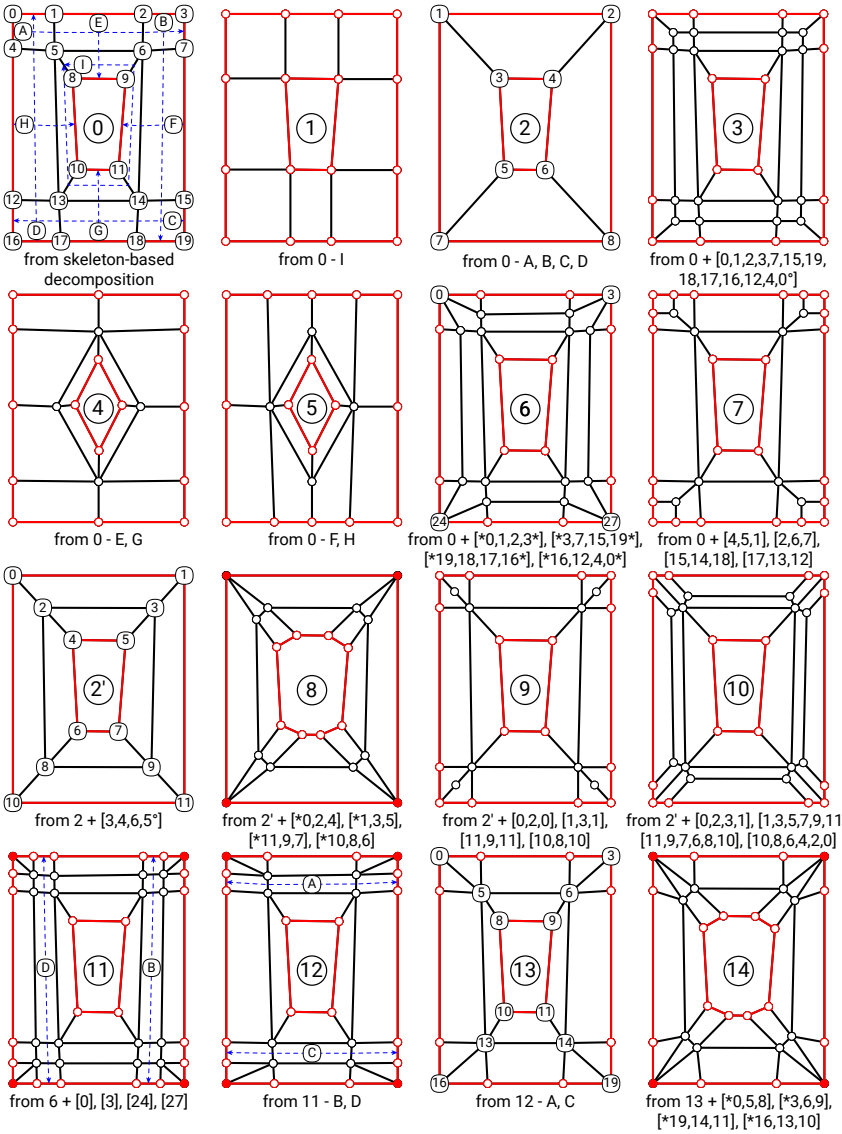



Figure 5.25 – Rule-based exploration of coarse quad meshes.

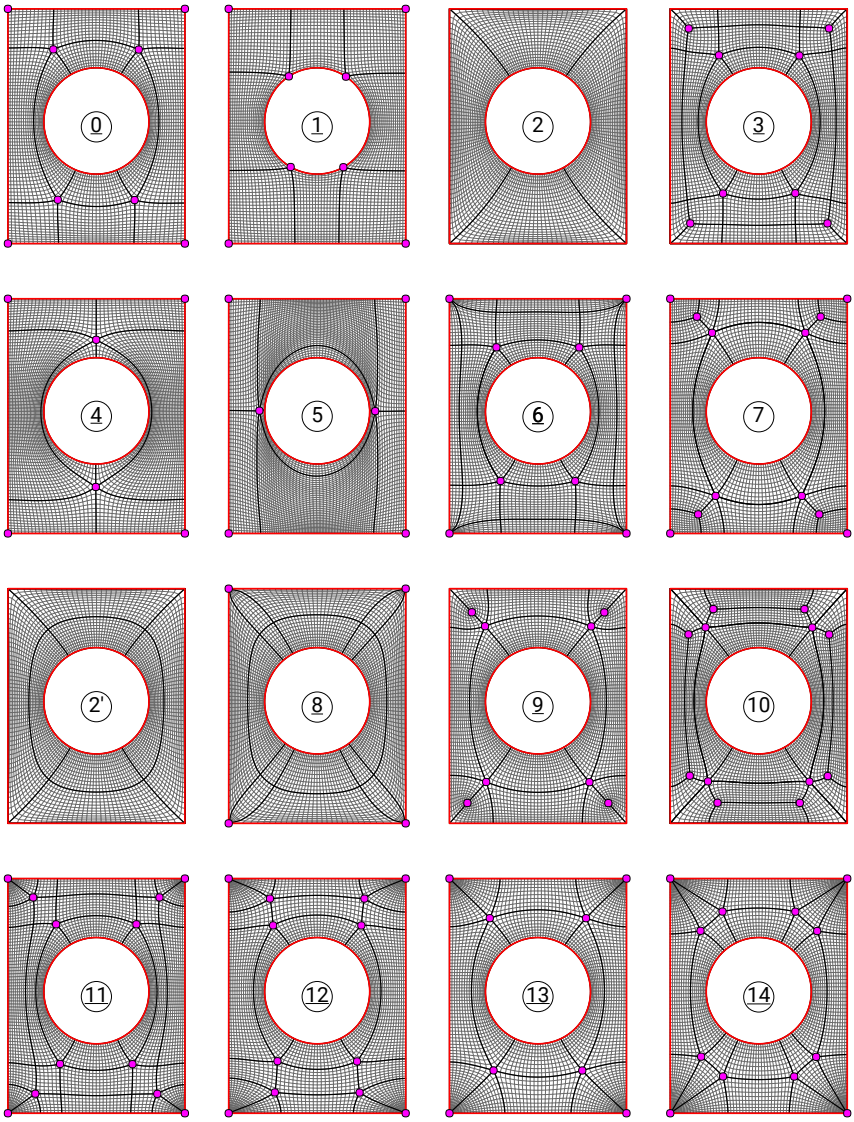



Figure 5.26 – Resulting quad-mesh patterns with different topologies. The labels of the Pareto designs are underlined.

Table 5.1 – Statics and fabrication metrics for the quad-mesh patterns with different singularities. Some metrics $X$ are normalised by the maximum value to obtain a metric $\overline{X}$ with a worst, maximum of 1. The Pareto designs are underlined.

| design | number edges $\lvert E\rvert$ | total edge length $E_{l,tot}$ [m] | edge length disp. $\overline{L}$ [-] | face curv. $\overline{C}$ [-] | face skew. $\overline{S}$ [-] | max. SLS defl. $\overline{F}$ [-] | max. ULS stress util. $\overline{U}$ [-] | ULS first buckl. load factor $\overline{B}$ [-] |
|---|---|---|---|---|---|---|---|---|
| 0 | 7140 | 9591 | 0.8 | 0.71 | 0.24 | 1 | 0.72 | 0.7 |
| 1 | 7537 | 9866 | 0.64 | 0.88 | 0.19 | 0.85 | 0.8 | 0.71 |
| 2 | 7896 | 10247 | 0.92 | 0.6 | 0.54 | 0.82 | 0.98 | 0.76 |
| 3 | 7278 | 9687 | 0.82 | 0.61 | 0.3 | 0.86 | 0.97 | 0.8 |
| 4 | 8532 | 10625 | 0.84 | 0.86 | 0.52 | 0.84 | 0.71 | 0.67 |
| 5 | 9760 | 12028 | 0.84 | 1 | 1 | 0.84 | 0.63 | 1 |
| 6 | 7887 | 10310 | 0.87 | 0.44 | 0.33 | 0.6 | 0.44 | 0.8 |
| 7 | 7868 | 10091 | 0.81 | 0.64 | 0.38 | 0.78 | 0.81 | 0.72 |
| 8 | 8060 | 10632 | 1 | 0.38 | 0.74 | 0.71 | 0.55 | 0.79 |
| 9 | 8474 | 10475 | 0.8 | 0.51 | 0.39 | 0.65 | 0.83 | 0.71 |
| 10 | 8622 | 10659 | 0.87 | 0.76 | 0.42 | 0.69 | 1 | 0.8 |
| 11 | 8602 | 10605 | 0.81 | 0.42 | 0.35 | 0.64 | 0.32 | 0.72 |
| 12 | 7886 | 10191 | 0.87 | 0.35 | 0.41 | 0.62 | 0.31 | 0.81 |
| 13 | 7364 | 9786 | 0.92 | 0.43 | 0.39 | 0.62 | 0.28 | 0.84 |
| 14 | 8264 | 10442 | 0.9 | 0.45 | 0.27 | 0.5 | 0.28 | 0.84 |
| min. | 7140 | 9591 | 0.64 | 0.35 | 0.19 | 0.5 | 0.28 | 0.67 |
| max. | 9760 | 12028 | 1 | 1 | 1 | 1 | 1 | 1 |
| avrg. | 8060 | 10336 | 0.85 | 0.61 | 0.44 | 0.74 | 0.66 | 0.78 |
| st. dev. | 650 | 569 | 0.08 | 0.2 | 0.2 | 0.13 | 0.27 | 0.08 |

For visualisation, the six-dimension performance space is mapped to a two-dimension space in Figure 5.27 using a Self-Organising Map (SOM) with the implementation of Harding for Grasshopper3D is used [Harding, 2016]. SOMs are a type of artificial neural networks for dimensionality reduction of an N-dimension space to a lower dimension [Kohonen, 2013]. Classically, the target dimension is two or three for visualisation purpose. A performance-

based SOM is different from a parameter-based SOM. For instance, the SOM of Fuhrimann *et al.* [Fuhrimann et al., 2018] uses the form and force parameters that generate the geometry of the designs using Combinatorial Equilibrium Modelling [Ohlbrock et al., 2017]. A few similar applications of SOMs in structural design exist [Saldana Ochoa et al., 2019, Pan et al., 2019].

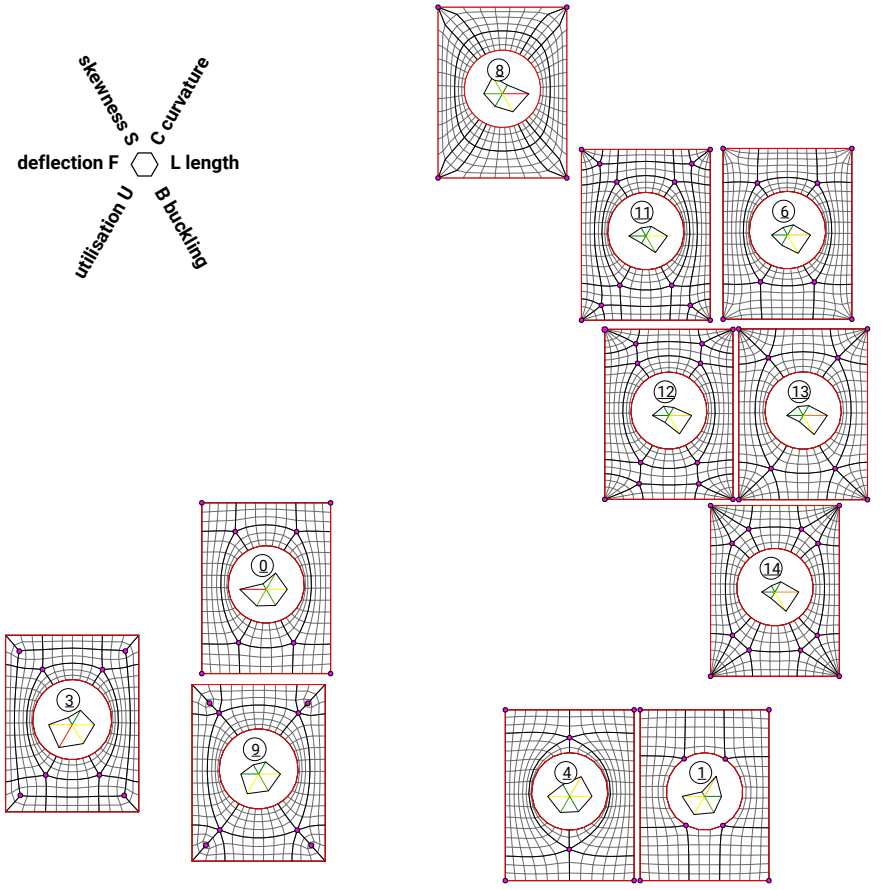



Figure 5.27 – Self-organising map for performance-based visualisation in 2D of the designs of the high-dimension 6D Pareto front. The spider graphs represent the performance of each design. Non-Pareto designs are excluded. Coarser meshes are shown for clarity.

During this open exploration process informed by performance evaluation, a map provides a visualisation of the performance similarity between the designs. Indeed, the closer two designs, the more similar their respective performances. The designs form clusters with similar performances, such as designs 6, 11, 12, 13 and 14. These designs share pole points at the corners, including both radial and ortho-radial elements, contrary to design 8 that has radial but not ortho-radial elements.

The map helps to understand the consequences of a topological design choice during the open exploration process. Moreover, the map suggests that designs with similar topologies are likely to have a similar performance.

## 5.5 Summary of contributions

This chapter introduced rule-based topology finding, using topological rules to perform topological exploration:

- a topological strip structure in quad meshes was identified as a fundamental data structure;
- a quad-mesh grammar of low-level rules for reciprocal addition and deletion of these strips was created;
- this grammar was used for multi-objective exploration using Pareto fronts and Self-Organising Maps to visualise the results.

More generally, the strip data structure could serve for more specific computing of quad meshes, as opposed to general face-based or edge-based data structures.

The grammar allows comprehensive exploration because the low-level rules apply to fundamental elements. Non-linear exploration is possible across the design space of quad-mesh singularities.

However, the relationship between exploration and performance is unilateral. The designer generates topologies, processes them and evaluates them. It belongs to the designer to decide which rule to apply next. In the context of multi-objective design, only applying rules and cancelling the ones that do not improve the general performance is not an option as one seeks for trade-offs. The performance of previous designs should inform the generation of the next ones. The SOM hints at some similarities between topology and performance in some designs, with the formation of clusters. Therefore, informed exploration can shift the problem of obtaining designs

with a similar performance to designs with a similar topology. Informed exploration based on design performance would close the loop initiated by open exploration.

# Chapter 6

# Similarity-informed exploration

## 6.1 Motivations

Rule-based topology finding allows comprehensive exploration of the singularity design space of quad meshes. Through open exploration, the designer can search for the rules that yield topologies with strips that improve performance. However, carrying out multi-objective design is driven by the exploration of trade-offs to fit best the context of the project. Figure 6.1 illustrates this challenge based on the patterns of the waffle slab system in Figure 1.7a and the ribbed slab system in Figure 1.7b, designed for fabrication affordability and structural efficiency, respectively.

A heuristic suggested by the study on rule-based topology finding in Section 5.4 is that designs with similar topologies have similar performances. Finding designs with different degrees of topological similarity should offer different degrees of performance similarity for multi-objective design. This assumption relaxes the problem from the performance space to the topological space. Input designs that each perform well for a single objective inform the generation of topologically similar designs. These input designs can stem from design intuition or any exploration and optimisation strategy, like the integration of a stress field for structural efficiency.

This chapter introduces *similarity-informed topology finding*. Section 6.2 clarifies the topological equality between two quad meshes based on their strip structure. Section 6.3 defines a topological distance between

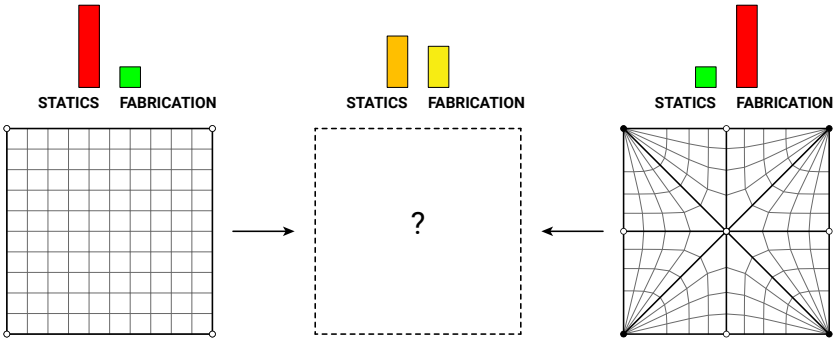



Figure 6.1 – Challenge for informed exploration of trade-offs between multiple objectives. The search for similarity in performance moves to searching for similarity in topology.

two quad meshes to assess their similarity. An algorithm allows computing the distance and the strip rules to convert one mesh into another. Section 6.4 develops a strategy to generate hybrid quad meshes sharing different degrees of similarity with a set of input quad meshes to combine. Section 6.5 applies this approach to similarity-informed topology finding of designs based on single-objective heuristic or optimal quad meshes to perform multi-objective design.

## 6.2 Quad-mesh equality

Before evaluating topological similarity, topological equality between quad meshes is tested based on graph-isomorphism checks. The algorithm of Cordella *et al.* [Cordella et al., 2001] for graph isomorphism is used, implemented in the Python library NetworkX on the structures, dynamics and functions of networks [Hagberg et al., 2008]. The isomorphism can integrate vertex and edge attribute matching.

### 6.2.1 Graph isomorphism

Isomorphism only depends on topology. Two graphs are isomorphic if a bijection exists that maps the vertices of one graph to the other while preserving the edge connectivity. If two graphs are isomorphic, several bijections may be possible.

In Figure 6.2, the two-edge graph A, B, C is isomorphic to the two-edge graph 1, 2, 3 because a bijection exists, which maps the pairs of vertices A - 1, B - 2, C - 3. A second bijection exists, which maps A - 3, B - 2 and C - 1. On the contrary, the two-edge graph A, B, C is not isomorphic to the three-edge graph 1, 2, 3, because no bijection would preserve the edge connectivity, here due to the third edge. For two graphs to be isomorphic, having the same number of vertices and edges is a necessary but non-sufficient condition.

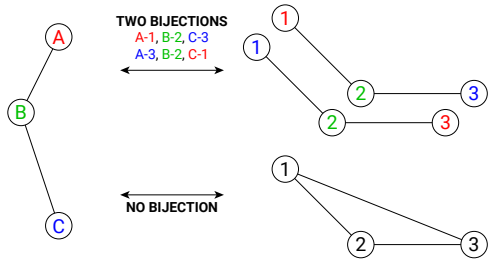



Figure 6.2 – Two graphs are isomorphic if a bijection exists between the vertices of two graphs that preserves the edge connectivity.

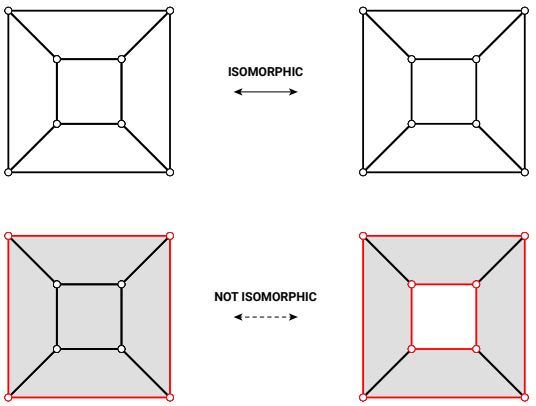



Figure 6.3 – Two quad meshes are isomorphic if their graph of edges are isomorphic while preserving edge boundary attribute marked in red. The corresponding mesh faces are marked in grey for clarity.

### 6.2.2 Mesh isomorphism

Mesh isomorphism relies on graph isomorphism. An edge-based graph loses the face data compared to a face-based mesh. Therefore, two meshes are isomorphic if the graphs of their edges are isomorphic with matching edge boundary attributes. Figure 6.3 presents the difference due to the integration of edge boundary attributes in red.

### 6.2.3 Quad-mesh isomorphism

Strip-graph isomorphism is not sufficient to guarantee quad-mesh isomorphism. The data reduction of a quad mesh to a strip graph creates a loss of isomorphism between quad meshes and strip graphs. Indeed, a strip graph can be the same for two different quad meshes, as illustrated in Figure 6.4.

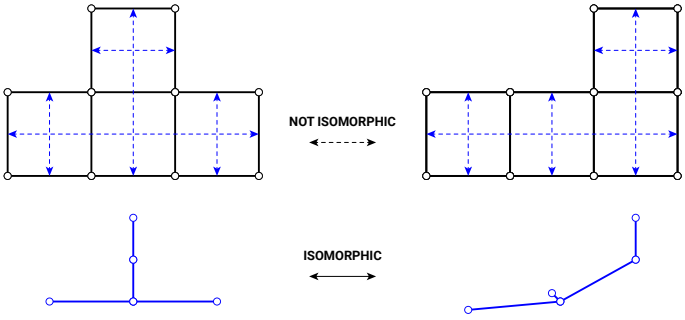



Figure 6.4 – Strip-graph isomorphism is not sufficient for quad-mesh isomorphism due to the lack of isomorphism between quad meshes and strip graphs.

Nevertheless, strip graphs have fewer elements than quad meshes, and isomorphism check is faster for strip graphs than quad meshes. For instance, the mesh in Figure 6.5 has thirteen edges and its strip graph only four edges. A search algorithm can check strip-graph isomorphism before quad-mesh isomorphism if most of isomorphism checks are likely to fail.

Similarly, a search algorithm can check necessary but non-sufficient conditions before isomorphism check. If two graphs do not have the same number of vertices with a valency of three, then they are not isomorphic. If two graphs have the same number of vertices for each valency, they could be isomorphic. These lighter checks can include diverse local conditions.

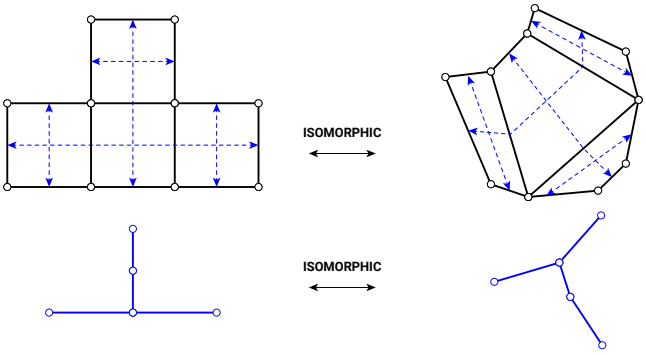



Figure 6.5 – A quad-mesh with thirteen edges is heavier than its strip graph with four edges only. In general, strip graphs have fewer elements than quad meshes. Checking isomorphism between two quad meshes is slower than between two strip graphs.

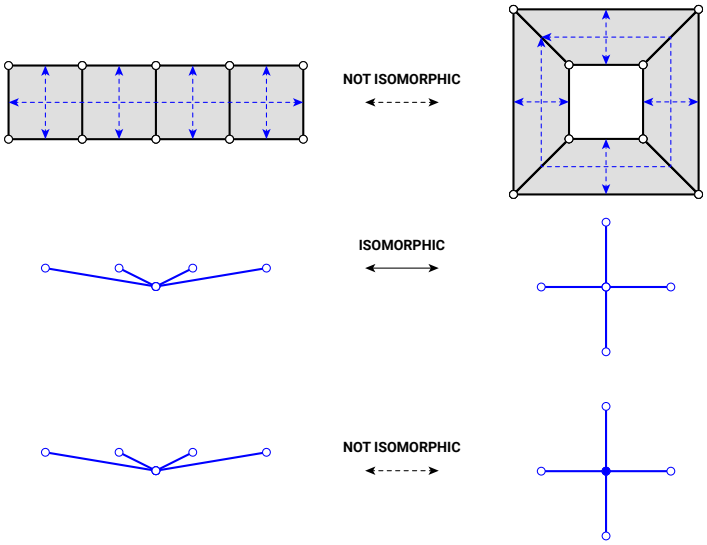



Figure 6.6 – Closed strip data are integrated as graph-vertex attributes marked as filled dots to reduce the cases of strip-graph isomorphism without quad-mesh isomorphism. The mesh faces are marked in grey for clarity.

Examples of such conditions are the distribution of valencies, the existence of triangles or cliques, which corresponds to a subset of vertices that are completely connected [Hagberg et al., 2008].

Graph-vertex attribute integrates closed strip data. This addition reduces the cases of false positives of strip-graph isomorphism without quad-mesh isomorphism, as shown in Figure 6.6.

### 6.2.4 Pseudo-quad-mesh isomorphism

The strip graph can not integrate pole data. If a strip has only one extremity with a pole as in Figure 6.7, an indeterminacy exists on which extremity. Indeed, the strip does not have an orientation. Therefore, a vertex attribute in the graph graphs can not specify the pole location.

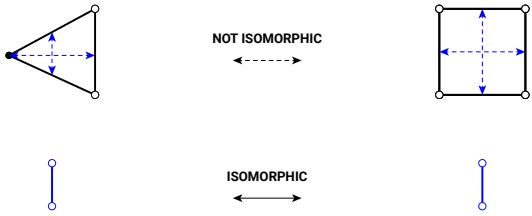



Figure 6.7 – Pole data can not be integrated as graph-vertex attributes due to the indeterminacy of the location of the pole at the extremities of the strip.

However, the mesh graph can integrate pole data. For a pseudo-quad mesh, the mesh graph stems from the equivalent quad mesh with additional boundary edges at the location of the poles.

The evaluation of equality serves as a basis for defining and computing the topological similarity between two quad meshes.

## 6.3 Quad-mesh similarity

A distance – or metric – measures the similarity between the topology of two quad meshes.

This distance could rely directly on the singularities, based on their valency or index. Histograms that encode the type and number of singularities provide a means for comparison. The multiple histogram distances in the

literature include the Chi-Square Distance or the Earth Moving Distance, for instance, [Pele and Werman, 2010].

However, a distance based on pure topology would not provide an answer on how to get from one quad mesh to another. Therefore, a topological distance based on the strip grammar is used to assess the topological similarity.

In computer science and information theory, string metrics found many applications to compare discrete data. Some string metrics evaluate the distance between two strings based on the number of operations – or rules – to apply to convert one string into the other. The different string metrics differ by the allowed operations: the Hamming distance [Hamming, 1950] allows substitutions only whereas the Levenshtein distance [Levenshtein, 1966] allows substitutions, insertions and deletions. Other string metrics tackle the evaluation of similarity differently, but they all have in common to respect the triangle inequality. The triangle equality states that the distance between two objects is equal or shorter than the sum of the distances between these two objects and any third object.

The definition of the quad-mesh distance follows these principles.

### 6.3.1 Definition

A topological distance is defined to measure the similarity between two quad meshes based on the strip grammar as a means for exploration. The distance is defined as the minimal number of strips to add and delete to go from one quad mesh to another. The minimality condition is necessary to exclude reciprocal addition and deletion of the same strip.

As the strip grammar modifies the pattern topology, not the shape topology, the distance applies to orientable compact manifold quad meshes with the same number of handles and the same number of boundaries. The grammar and the distance can be extended to include topological modifications on the shape topology.

The distance $d$ applies to two elements of the space $E = E_{g,N}$ of quad meshes with the same number of handles $g$ and boundaries $N$ and yields a positive integer in $\mathbb{N}^+$:

$$d : E \to \mathbb{N}^+. \tag{6.1}$$

The three distance properties are verified:

- the distance is symmetric. The same minimal number of rules apply from a topology $A$ to a topology $B$ and from $B$ to $A$, as for each

addition or deletion strip rule there is a reciprocal one:

$$\forall (A, B) \in E^2, d(A, B) = d(B, A); \tag{6.2}$$

- the distance from a topology to itself is null because no rules need to be applied and if no rules needs to be applied then two topologies are the same:

$$\forall (A, B) \in E^2, d(A, B) = 0 \iff A = B; \tag{6.3}$$

- the triangle inequality is satisfied: the same or a lower number of rules have to be applied to go directly from a topology $A$ to a topology $C$ than going through an intermediary topology $B$:

$$\forall (A, B, C) \in E^3, d(A, C) \leq d(A, B) + d(B, C). \tag{6.4}$$

  Equality occurs if no rules between $A$ and $B$ and between $B$ and $C$ are reciprocal and can compensate each other.

During rule-based exploration, successive application of strip rules generally increases by one the distance and therefore decreases by one the similarity with the initial design. Only application of a reciprocal rule cancelling a previous rule reduces by one the distance. The rules applied in Figure 6.8 add two strips and delete two other strips. The distance increases by one for every rule, resulting in a final distance of four. The strip graphs highlight similarity, even though strip graphs and quad meshes are not isomorphic.

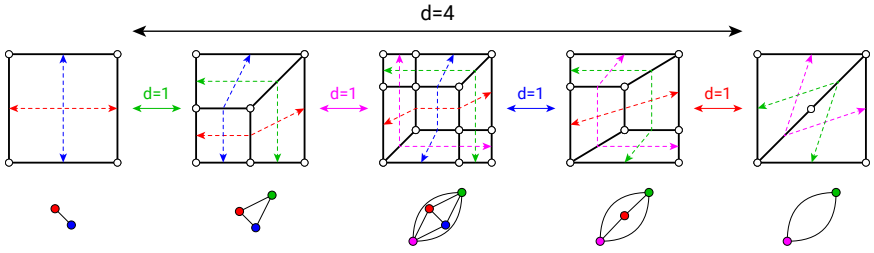



Figure 6.8 – Applying independent rules on a quad mesh increases the distance by one, and decreases the similarity by one, for a final distance of four, highlighted by the strip graphs.

Deleting some strips can cause collateral deletion of other strips whose faces are all included in the deleted strips. In Figure 6.9, deleting the strip in pink in a four-strip quad mesh results in a two-strip quad mesh at a distance

of two due to the secondary deletion of the strip in green. Although one rule is applied, the distance is not one but two. Therefore, the application of a deletion rule counts as the application of multiple deletion rules, equal to one plus the number of collateral deletions.

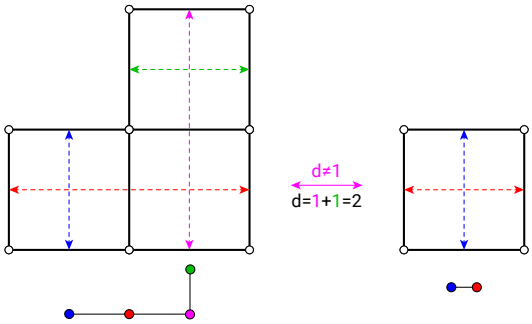



Figure 6.9 – Collateral strip deletion occurs when deleting one or several strips, like the one in pink, results in the deletion of other strips, like the one in green. Applying such a deletion rule counts as multiple deletions rules, two in this case.

Computing the existence and number of collateral deletions $n_{col}$ can occur before or after applying the deletion rules. Before deletion by collecting the faces of the strips to delete and checking if other strips have all their faces part of the faces to delete. After deletion by comparing the number of deletion rules applied $n_{del}$ with the difference of strips after $s_1$ and before $s_0$:

$$n_{col} = n_{del} - (s_1 - s_0). \tag{6.5}$$

Computing the distance along with the similar and dissimilar strips provides the rules to go from one topology to another.

### 6.3.2 Computation

The computation of the topological distance checks isomorphism between two meshes while trying combinations of an increasing number of strip deletions.

#### 6.3.2.1 Approach

The distance between two quad meshes is defined as the minimum number of strips to add and delete to go from one to another. Searching for combinations of addition and deletion rules to apply on one quad mesh involves the infinite combinatorial possibilities for strip additions. However, the number of strip deletions is bounded by $2^n$ for a quad mesh with $n$ strips. Therefore, the distance results from finding the two minimum sets of strip deletions to apply on the two quad meshes $M_1$ and $M_2$ that yield the same submesh $M_0$, up to an isomorphism. In Figure 6.10, the deletion of the strips in pink yields two isomorphic submeshes. This submesh represents the largest similar structure of strips in common between the two quad meshes. Two strip deletion rules are not reciprocal and can not compensate each other. This property guarantees having the minimum number of rules. Therefore, finding such a submesh yields the equality case of the triangle inequality:

$$d(M_1, M_2) = d(M_1, M_0) + d(M_0, M_2), \tag{6.6}$$

with $M_1$ and $M_2$ the input meshes and $M_0$ their submesh.

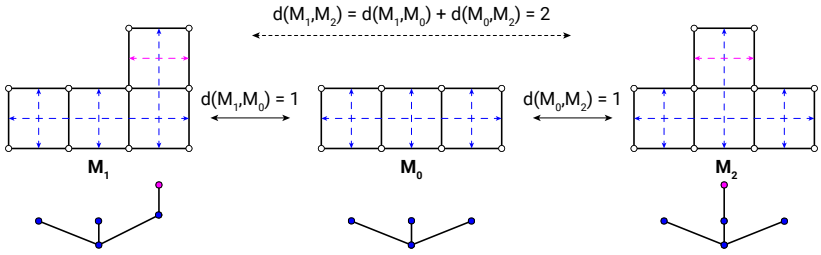



Figure 6.10 – Finding the distance between two quad meshes $M_1$ and $M_2$ via a common submesh $M_0$ using strip deletion rules only. The deleted strips are highlighted in pink.

Let $n_1$ and $n_2$ the number of strips of the two quad meshes $M_1$ and $M_2$, respectively, and supposedly $n_1 \geq n_2$. To find the submesh, an increasing number of strips are deleted on both meshes in parallel. The meshes have a minimum distance $d_{min} = |n_1 - n_2| = n_1 - n_2$ due to a potentially different number of strips. Starting with $k = 0$, all pairs of combinations to delete $k + \max(0, n_1 - n_2) = k + d_{min}$ strips in mesh $M_1$ and $k + \max(0, n_2 - n_1) = k$ strips in mesh $M_2$ are tested. If a pair of combinations yields two isomorphic quad meshes after deletion of the respective combinations of strips, then the

process stops. Otherwise, $k$ increases by one, and the process repeats to find the submesh.

If there is no such submesh, then the largest similar structure is a single strip, a quad having at least two strips. The distance between the two quad meshes is then the maximum:

$$d_{max} = (n_1 - 1) + (n_2 - 1). \tag{6.7}$$

When the search finds such a submesh, applying Equation 6.6 yields the distance. Indeed, the enumeration of combinations of rules in an increasing order guarantees to find the largest submesh, resulting from the minimum number of rules needed. If the submesh $M_0$ has $n_0$ strips, then the distance equals:

$$\begin{aligned} d(M_1, M_2) &= d(M_1, M_0) + d(M_0, M_2) \\ &= n_1 - n_0 + n_2 - n_0 \\ &= n_1 + n_2 - 2n_0. \end{aligned} \tag{6.8}$$

The distance can be expressed by the number of iterations $k$ as well, as $n_0$ equals the number of strips in the quad mesh $M_2$ minus the number of strips deleted, still assuming $n_1 \geq n_2$:

$$n_0 = n_2 - k, \tag{6.9}$$

therefore:

$$d(M_1, M_2) = n_1 - n_2 + 2k. \tag{6.10}$$

Figure 6.10 shows the computation of the distance between two meshes with five strips each. The largest submesh has four strips, resulting from the deletion of one strip for each input mesh highlighted in pink. Therefore, the distance between the input meshes equals two.

Deletion rules with collateral deletions are discarded to prevent overestimation of the distance. Indeed, they are equivalent to the application of multiple deletion rules. These multiple rules appear for a higher value of $k$ and should not count at a lower value.

Several non-isomorphic submeshes with the same number of strips can occur. Figure 6.11 shows two different submeshes $M_0'$ and $M_0''$, both with four strips and at a distance of one the input meshes $M_1$ and $M_2$. Only one is sufficient to compute the distance between the input meshes. Nevertheless,

to collect all of these submeshes, the computation should not stop when one submesh is obtained but finish testing all of the other rule combinations for the current value of $k$. Considering these different submeshes enriches quad-mesh combination, in Section 6.4.

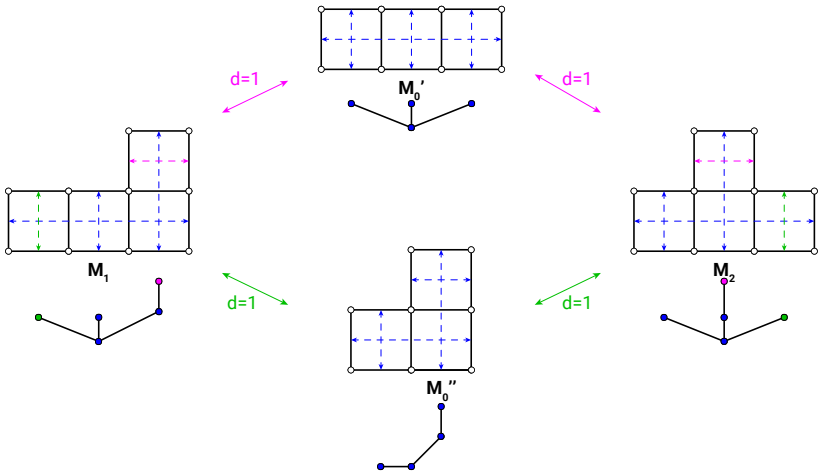



Figure 6.11 – Multiple submeshes $M_0'$ and $M_0''$ with the same number of strips and at the same distance of input quad meshes $M_1$ and $M_2$ can exist due to the deletion of different combinations of strips, differentiated in pink and in green.

#### 6.3.2.2 Algorithm

The algorithm for computation of the topological distance between two quad meshes is detailed in Algorithm 1. The input quad meshes are $M_1$ and $M_2$ with the assumption that $M_1$ has the highest number of strips. The algorithm skips the combinations of strip deletions resulting in collateral deletions. Isomorphism between the submeshes $M_{1,0}$ and $M_{2,0}$, resulting from strip deletions, is directly checked on the quad-mesh graph. The algorithm returns all largest non-isomorphic submeshes with the same number of strips. The algorithm also returns the strip deletions to apply to each input mesh to obtain the largest submesh $M_0$. If no submesh exist, the common structure is not a mesh, but any unique strip and the distance is maximal.

```
start empty list of results
n1, n2 = number of strips in M1 and M2
for k from 0 to n1 - 2 do
    for combinations of n1 - n2 + k strips s1 in M1 do
        copy M1 as M10
        delete strips s1 in M10
        if collateral strip deletion in M10 then
            skip this combination
        end
        for combinations of k strips s2 in M2) do
            copy M2 as M20
            delete strips s2 in M20
            if collateral strip deletion in M20 then
                skip this combination
            end
        end
        if meshes M10 and M20 are isomorphic then
            if M10 is not isomorphic to any submesh in results then
                add M10 to the list of submeshes
            end
            distance = n1 - n2 + 2k
            store submesh in results with distance, s1 and s2
        end
    end
    if results are not empty then
        return results
    end
    return max distance = n1 + n2 - 2
end
```

**Algorithm 1:** Algorithm for computing the topological distance and the dissimilar strips between two quad meshes $M_1$ and $M_2$, assuming that $M_1$ has the highest number of strips.

### 6.3.3 Validation

Several examples validate the algorithm. Table 6.1 shows the numerical results. The results provide the number of strips $n_1$, $n_2$ and $n_0$ of the input meshes and their submesh, the level of increment in the search $k$, the distance between the meshes $d$, the number of submeshes, the number of isomorphism checks and the computation time.

In Figure 6.12a, mesh $M_2$ is included in mesh $M_1$ after deletion of four strips. Therefore, $M_2$ is a submesh $M_0$, and the distance is equal to four. Deleting any combination of the vertical strips is possible, but they all yield isomorphic meshes. In Figure 6.12b, mesh $M_2$ is included in mesh $M_1$ after deletion of two strips. Therefore, $M_2$ is a submesh $M_0$ and the distance is equal to two. In Figure 6.12c, no submesh exists because the two meshes do not share any strip structure. They only have in common one of their strips. As they have four and two strips, their distance is equal to four. In Figure 6.12d, the two meshes have a submesh at a distance of two and one. Therefore their distance is equal to three. In Figure 6.12e, the two meshes have two non-isomorphic submeshes at a distance of five and three. Therefore their distance is equal to eight.

### 6.3.4 Limitation

The computation of the topological distance between two quad meshes is performed here in an enumerative manner. For large or far quad meshes, meaning with a high number of strips or a high distance, this algorithm has limitations for interactive design due to the computational time, as the number of rule combinations to explore is combinatorial. The examples in Figures 6.12b and 6.12e show the increase in computation time for a high distance between two quad meshes with a similar number of strips: 0.52 s for a distance of 2 between a 10-strip and a 8-strip meshes, and 12.4 s for a distance of 8 between a 8-strip and a 6-strip meshes. The examples in Figures 6.13a and 6.13b show the increase in computation time for a higher density: 0.62 s for a distance of 2 between an 8-strip and a 6-strip mesh and 9.4 s for a distance of 3 between a 12-strip and a 9-strip mesh.

Nevertheless, the presented algorithm allows the computation of the distance between quad meshes. Moreover, for coarse meshes that differ by a couple of strips, the computation time is under 1 s. Two improvements are possible for the algorithm. First, using parallel computing of the isomorphism checks. Multi-threading could improve the speed of the algorithm on the enumeration of pairs of combinations of strips. Second, using stochas-

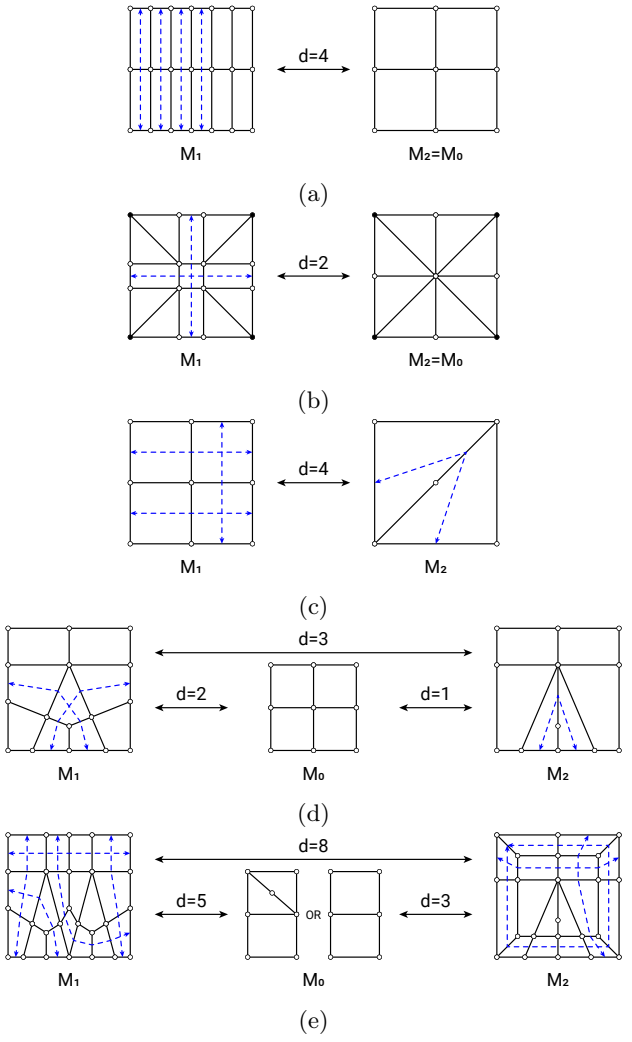
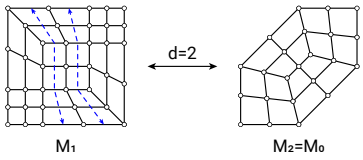
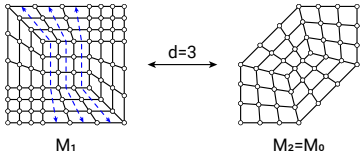



Figure 6.12 – Validation examples for computing the topological distance between two quad meshes. The distances between the meshes $M_1$ and $M_2$ and between their submesh $M_0$ is provided along with one of the possible combination of strips to delete in blue.




(a)




(b)

Figure 6.13 – Limitation examples for computing the topological distance between two dense quad meshes. The distances between the meshes $M_1$ and $M_2$ and between their submesh $M_0$ is provided along with one of the possible combination of strips to delete in blue.

tic search algorithms informed by a measure of the degree of isomorphism between two graphs, instead of a boolean check.

Table 6.1 – Numerical results for the validation and limitation examples for the computation of the distance between two quad meshes.

| Figure | 6.12a | 6.12b | 6.12c | 6.12d | 6.12e | 6.13a | 6.13b |
|---|---|---|---|---|---|---|---|
| $n_1$ | 8 | 10 | 4 | 6 | 8 | 8 | 12 |
| $n_2$ | 4 | 8 | 2 | 5 | 6 | 6 | 9 |
| $n_0$ | 4 | 8 | - | 4 | 3 | 6 | 9 |
| $k$ | 0 | 0 | 2 | 1 | 3 | 0 | 0 |
| $d$ | 4 | 2 | 4 | 3 | 8 | 2 | 3 |
| # submeshes | 15 | 1 | - | 1 | 2 | 1 | 1 |
| # iso. checks | 55 | 41 | 5 | 62 | 1516 | 28 | 220 |
| comp. time [s] | 0.58 | 0.52 | 0.025 | 0.30 | 12.4 | 0.62 | 9.4 |

Based on the computation of the distance between two meshes along with their similar and dissimilar strips, hybrid quad meshes can be generated to interpolate input quad meshes at different distances with different degrees of topological similarity.

## 6.4 Quad-mesh combination

Patterns with similar topologies perform similarly. Expressed differently, the less similar two patterns, the less likely they are to perform similarly. A pattern can be designed to fulfil a single objective. However, designing to fulfil multiple objectives is more challenging. Designing patterns that share different degrees of similarity with a set of patterns designed for single objectives is likely to provide multi-objective trade-offs.

Based on the definition of equality and similarity for quad meshes, a strategy is proposed to generate hybrid quad meshes based on a set of input quad meshes. The hybrid quad meshes provide a combination in the sense of the topological distance, with different degrees of similarity with the input quad meshes.

The challenge lies in converting deletion rules to additions rules through mapping of strips and polyedges from one mesh to another.

### 6.4.1 Approach

The approach for the generation of hybrid meshes is as follows:

1. the input meshes are created by any generation means;
2. the submeshes are computed, which comprise only the strips in common between the input meshes. The submesh is the *intersection* of the sets of strips;
3. the supermeshes are computed, which comprise all of the strips of the input meshes. The supermesh is the *union* of the sets of strips;
4. the intermediary meshes are generated, which comprise different combinations of the strips of the input meshes.

Figure 6.14 illustrates these steps. In this example, two input meshes with respectively three and four strips yield seven hybrid meshes that are not isomorphic and therefore offer different design options. These hybrid meshes include one submesh with two strips, three supermeshes with five strips and three intermediary meshes with three to four strips. The hybrid meshes can correspond to different input strips that yield isomorphic meshes. The results show only one of these combinations of input strips.

The coloured strips highlight the similarity between each hybrid mesh, provided as the similarity $s$, along with the distance $d$ to the input meshes.

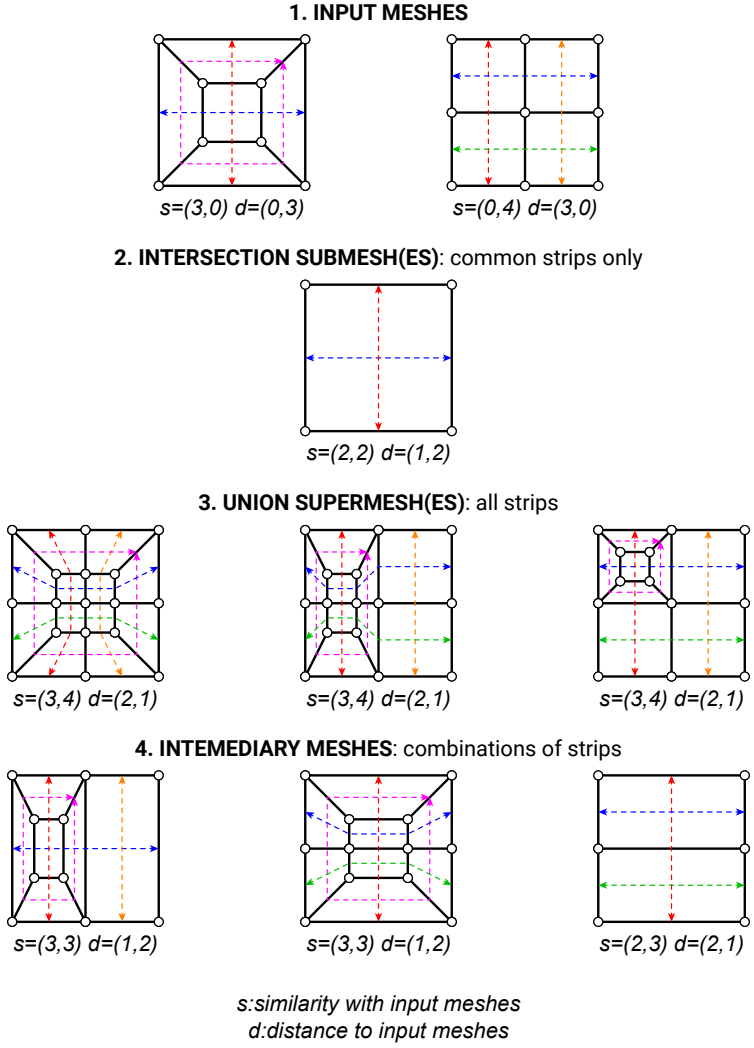



Figure 6.14 – Approach to generate hybrid quad meshes with different degrees of topological similarity with input quad meshes. The colours correspond to similar strips, supporting the values on the similarity and dissimilarity with the input meshes.

The similarity is the number of strips in the common strip structure, and the distance is the number of strips that are not part of the common strip structure.

For an input mesh $i$ with $n_i$ strips, the similarity $s_i$ and the distance $d_i$ are related as:

$$n_i = s_i + d_i. \tag{6.11}$$

They highlight the variation in the degree of similarity and dissimilarity with the input meshes.

The following sections present each in-between step.

### 6.4.2 Hybrid mesh generation

The generation of the hybrid meshes proceeds first with the intersection submeshes, then the union supermeshes and finally the intermediary meshes.

#### 6.4.2.1 Intersection submeshes

Computing the distance between the input meshes also yields the submeshes. All non-isomorphic submeshes with the highest number of strips are collected. This process is extended to multiple input meshes as follows:

1. the submeshes $M_0^{1,2}$ between two input meshes $M_1$ and $M_2$ are computed;
2. new submeshes $M_0^{1,2,3}$ between each submesh $M_0^{1,2}$ and another input mesh $M_3$ are computed;
3. new submeshes $M_0^{1,\ldots,i}$ between each submesh $M_0^{1,\ldots,i-1}$ and another input mesh $M_i$ are computed;
4. the final submeshes are $M_0^{1,\ldots,N}$, which are the non-isomorphic meshes that share the largest strip structure in common between the $N$ input meshes $M_1$ to $M_n$.

The submeshes contain all similar strips. The reciprocal polyedges of the dissimilar strips serve as input for an addition rule to obtain these disimilar strips. When deleting a strip $s_j$ in an input mesh $M_i$ to obtain a submesh $M_0^i$, a polyedge $P_{i,j}$ is stored. This polyedge updates during the deletion of the other strips. For each submesh, there is a set of polyedges that correspond to the addition rules to apply to revert the applied deletion rules. These rules are used to obtain supermeshes from the submeshes.

#### 6.4.2.2 Union supermeshes

Let $n_0$ be the number of strips in the submesh between two meshes $n_1$ and $n_2$. Exploring all combinations of strip deletions from two input meshes to a submesh provides a sum of the number of potential designs:

$$2^{n_1-n_0} + 2^{n_2-n_0}. \tag{6.12}$$

Instead, exploring all combinations of strip deletions and their reciprocal strip additions provides a product of the number of potential designs:

$$2^{n_1-n_0} \cdot 2^{n_2-n_0} = 2^{n_1-n_0+n_2-n_0}; \tag{6.13}$$

Full combination increases the potential number of hybrid meshes. In the general case of $N$ meshes, the number of combinations is:

$$\prod_{i=0}^{N-1} 2^{n_i-n_0} = 2^{\sum_{i=0}^{N-1}(n_i-n_0)}. \tag{6.14}$$

The supermesh combines all strips before exploring its combinations of strips.

Supermeshes are the non-isomorphic meshes that contain the strip structure of each input mesh. For each submesh, strip additions applies to all stored polyedges. After each strip addition, the polyedges for the remaining addition rules update. The addition rules on the longest polyedges are applied first as some polyedges may initially correspond to a single vertex.

#### 6.4.2.3 Intermediary meshes

The intermediary meshes result from the combination of deletions of dissimilar strips. Deleting no strips yields the supermesh. Deleting all strips yields the submesh. Other deletion combinations yield the intermediary meshes. For $n$ strips to delete in a supermesh, the combinations to apply equals:

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n. \tag{6.15}$$

The submeshes have the minimum number of strips:

$$n_{min} = n_0, \tag{6.16}$$

and the supermeshes have the maximum number of strips:

$$n_{max} = \sum_i (n_i - n_0), \tag{6.17}$$

with $n_i$ the number of strips in the input mesh $i$. The intermediary meshes have between $n_{min}$ and $n_{max}$ strips, strictly.

### 6.4.3 Data management

During the generation process, rules are transposed and reversed from one mesh to another using the input and output polyedges thanks to specific management of the polyedge and strip data.

#### 6.4.3.1 Updating polyedges

Applying strip rules modifies the topology, the number of elements and their connectivity in the quad mesh. As a result, the labelling of the vertices changes, as shown in Figure 6.15. Therefore, polyedges must update to apply to multiple rules.

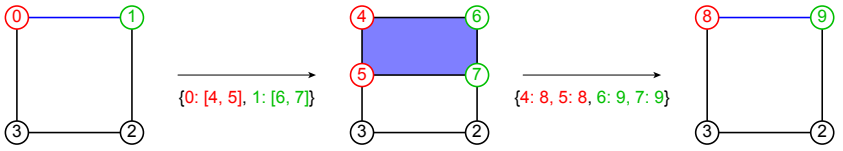



Figure 6.15 – Maps for vertex relabelling, in red and green, after strip addition and strip deletion rules, in blue.

Adding a strip duplicates the vertices of the input polyedge. In Figure 6.15, the old edge [0,1] branches to two possibilities with the new edges [4,6] and [5,7]. The new edges of a polyedge must be selected to rebuild a new continuous polyedge. Moreover, the polyedge must have its extremities on the boundary. Unless if the corresponding strip is closed or has pole extremities. The polyedge update can consider the several possibilities that fulfil these conditions. Here, the polyedge update considers only the shortest possibilities.

Deleting a strip merges the vertices of the edges of the input strip. In Figure 6.15, the old strip edges [4,5] and [6,7] result in the new vertices 8 and 9. Therefore, the old edges [4,6] and [5,7] become the same new edge [8,9].

#### 6.4.3.2 Mapping strips

The strips are mapped from one mesh to another by mapping polyedges for rule application. From the input meshes, strips are deleted to obtain the submesh. From the submeshes, strips are added to obtain the supermeshes. Figure 6.16 shows this process for two input meshes yielding two non-isomorphic submeshes, four non-isomorphic supermeshes and two non-isomorphic intermediary meshes. The strip structure of one of the input meshes is coloured in all meshes to highlight the similarity with this input mesh.

Figure 6.17 illustrates the mapping process of the polyedges from the input meshes to the submeshes, along which strips can be added to obtain the supermeshes, all shown in Figure 6.16. The input meshes have two non-isomorphic submeshes. Therefore, the polyedge-mapping process applies to each.

Deleting the dissimilar strips on the $N$ input meshes $M_i$ results in $N$ isomorphic submeshes $M_0^i$. These submeshes are isomorphic, but there can be multiple bijections that map one submesh to another. In Figure 6.17a for instance, the red polyedge in $M_0^1$ from $M_1$ and the blue polyedge in $M_0^2$ from $M_2$ can be combined in two ways. This situation results from the four possible bijections between the twofold symmetrical submeshes, two of which are not isomorphic.

The deleted strips provide reciprocal polyedges to apply the addition rule to obtain the supermesh. However, several possibilities exist to map the polyedges, two in this case. The multiple strips are added, while updating the other polyedges, resulting in the supermeshes. In Figure 6.17, the two submeshes have two polyedge maps each, resulting in the four supermeshes shown in Figure 6.16.

Deleting strips in the supermeshes yields the intermediary meshes shown in Figure 6.16. Here, only one strip is deleted, to obtain intermediary meshes with five strips, between the four-strip submeshes and the six-strip supermeshes. Most are isomorphic to the other meshes, except for two intermediary meshes that share different degrees of similarity with the input meshes and their strip structure.

If there is no strip structure in common and therefore no submesh to use for combination, a common strip structure must be created.

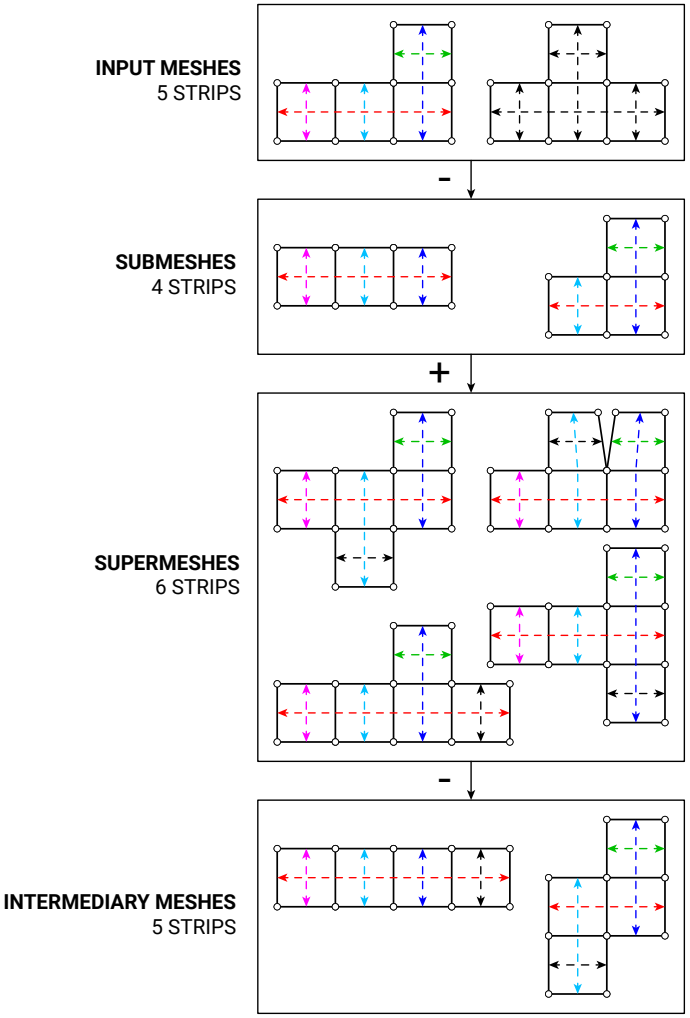
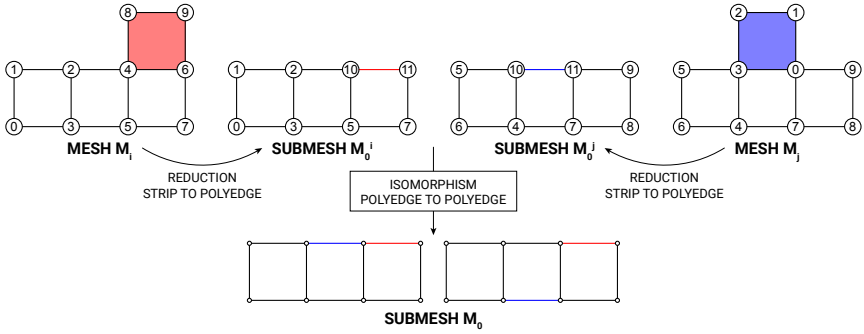
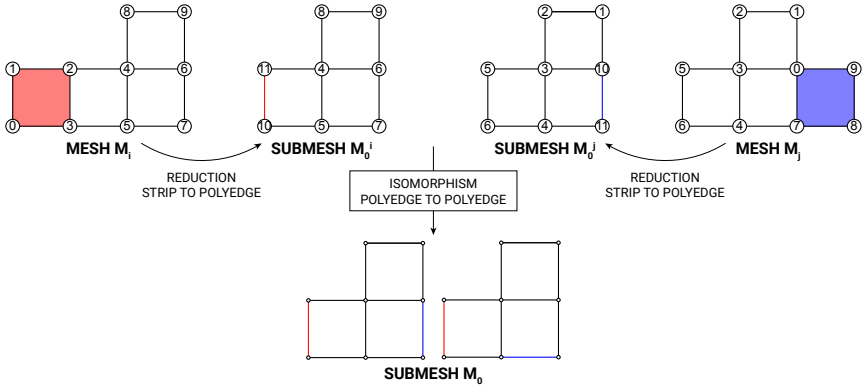



Figure 6.16 – Deleting the dissimilar strips between input meshes yields the submeshes. Adding the dissimilar strips on the submeshes yields the supermeshes. Deleting combinations of dissimilar strips yields the intermediary meshes. The strip structure of one of the input meshes is coloured in all hybrid meshes to highlight the similarity with this input mesh.




(a)




(b)

Figure 6.17 – Combining vertex maps to find polyedges across meshes with different numbers and labels of vertices. The strips become reduced polyedges through the rule maps from the input meshes $M_i$ and $M_j$to the submeshes $M_0^i$ and $M_0^j$, which are mapped across the submeshes $M_0$ via isomorphism search. Other submeshes between the same initial meshes result in different maps and therefore other possibilities for the generation of hybrid meshes.

### 6.4.4 Result prediction

Predicting the number of hybrid designs is difficult. It depends on the number of submeshes, supermeshes, intermediary meshes, as well as the existence and number of isomorphism between the different meshes.

This strategy for the combination of quad meshes is applied to similarity-informed topology finding for multi-objective design.

## 6.5 Similarity-informed topology finding

Based on quad-mesh patterns designed for single objectives, similarity-informed topology finding is performed using quad-mesh combination to design hybrid quad-mesh patterns for multi-objective design. In Section 6.5.1, designs are enumerated based on a unique design, whereas in Section 6.5.2, designs are combined based on multiple designs.

### 6.5.1 Similarity from one design

By applying $n$ rules on a design, $n$ other designs are generated. However, enumerating the $2^n$ combinations of these rules provides a wider set of designs sharing different degrees of similarity with the initial design. At a distance of $k$ from the initial design, $\binom{n}{k}$ potential designs can be generated, reduced by the redundant, isomorphic designs.

#### 6.5.1.1 Design problem

This section investigates the design of a quad-mesh pattern for a compression-only dome on an elliptic boundary of 15 m by 10 m. The structure consists of load-carrying beam elements covered by planar panels made of timber, for instance.

To design a compression-only structure, the Force Density Method is applied [Schek, 1974]. The force density $q_i$ in the element $i$ equals to the ratio of the force $f_i$ and the length of the element $l_i$:

$$q_i = \frac{f_i}{l_i}. \tag{6.18}$$

All the force densities equal one, $q_i = 1$. The load path $LP$ measures structural performance, which relates to the necessary structural volume:

$$LP = \sum_i f_i l_i. \tag{6.19}$$

The lower the load path, the more efficient the structure. Because the force densities equal one, the load path equals:

$$LP = \sum_i l_i^2. \tag{6.20}$$

Besides, the panels must be planar for fabrication affordability. Therefore, a planarisation process is applied, which induces deviation from the desired compression-only geometry. This deviation metric $D$ equals the sum of the movements of the vertices when enforcing planarisation. Moreover, the skewness of the panels should be low to reduce material loss during the cutting process. The curvature $C$ and the skewness $S$ metrics of the panels of the structure are evaluated as in Section 5.4.

#### 6.5.1.2 Pattern design

A couple of heuristics inform the rule-based exploration in Figure 6.18. To reduce face skewness, a quad mesh with two orthogonal strips serves as a starting point, with the addition of two other orthogonal strips. To reduce face curvature, the lines of curvature of an ellipsoid serve as a basis to apply three more rules. These rules recreate the strips for the pair of two-valent singularities due to the lines of curvature of an ellipsoid. Gaspard Monge uses these lines for the stereotomy of domes [Leroy, 1890]. Indeed, the lines of curvature form a conjugate network of curves that yield a discretisation with quasi-planar faces [Liu et al., 2006, Zadravec et al., 2010, Liu et al., 2011]. Figure 6.18 presents the addition rules with the dashed blue curves along the polyedges along which to add the strip. The dense quad mesh in grey helps for visualisation.

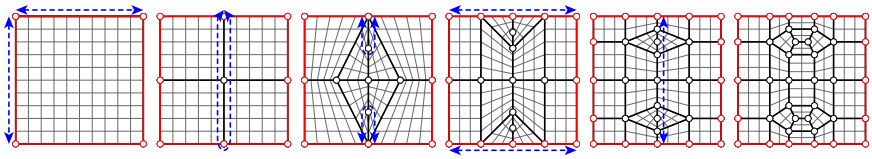


Figure 6.18 – Heuristic rule-based exploration with addition of strips along the polyedges highlighted with blue dashed lines.

The application of the eight rule additions in Figure 6.18 transforms the two-strip quad-mesh into a ten-strip quad mesh. This supermesh has ten strips gathered in six groups to preserve the twofold symmetry, represented by the colour scheme in Figure 6.19.

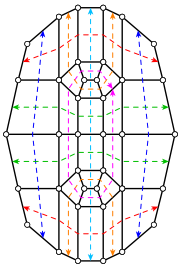


Figure 6.19 – The supermesh has ten strips, gathered in six coloured groups to preserve the twofold symmetry.

The enumeration of the $2^6 = 64$ deletions of the strip groups on the ten-strip quad mesh yields the 26 quad meshes in Figure 6.20. The ten-strip supermesh has the reference label 0.

Two reasons explain the 38 missing quad meshes. Some are isomorphic to one of the 26 quad meshes. For instance, several combinations of strip deletions yield the simple three-strip mesh 24. Some have a different and therefore invalid shape topology, that is not manifold or have a different number of boundaries. For instance, deleting all the strips but the one in red and orange results in two disconnected parts.

The set of designs does not include the initial two-strip quad mesh, as it is redundant with the four-strip quad mesh 23. Indeed, the two by two grid is equivalent to the one by one grid. Other quad meshes like the three-strip quad mesh 24, however, offer a different distribution of strips with a two by one grid, for instance.

Opposed to the results in Section 5.4 plotted based on performance similarity, Figure 6.20 organises the designs based on topological similarity, using the horizontal axis as distance to supermesh 0. The nine-strips designs are at a distance of one from the supermesh 0; the eight strips designs are at a distance of two; and so on. The distance to the supermesh directly equals to the different number of strips.

However, the organisation does not reflect the distance between the other meshes. The three-strip design 24 and the four-strip design 22 have a distance of one, corresponding to the pink strip. However, the three-strip design 25 and the four-strip design 23 have a distance of three, corresponding to the pink and blue strips. Since the distance between intra-group designs is not null, the distance between inter-group designs is not constant.

The densification of the coarse quad meshes yields the quad-mesh pat-

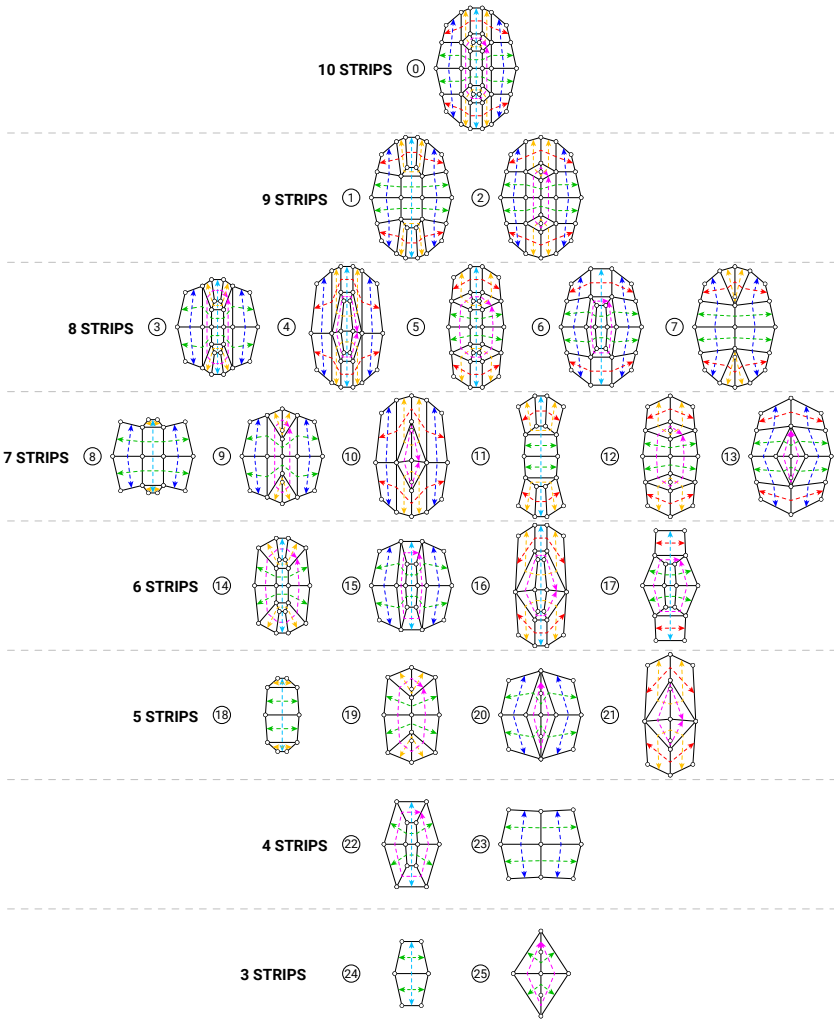



Figure 6.20 – Enumeration of the deletions of the coloured strip groups in quad mesh 0 to generate 25 other quad meshes. These quad meshes share a different number of these strips and, therefore, feature different degrees of similarity with quad mesh 0, which directly relates to the difference in the number of strips.

terns. The density parameters of the strips are set to the same value to get as close as possible to the target of 500 faces.

Geometrical design is subdivided in several steps.

1. area-weighted Laplacian smoothing with boundary vertices constrained to the elliptic boundary;

2. funicular form finding with the force density method with fixed boundary vertices, a uniform force density and a vertical load distributed on the vertices weighted by their projected tributary area;

3. planarisation with an iterative procedure that independently projects each face to their best-fit plane before merging the disconnected vertices. This approach is similar to the projection-based algorithm of ShapeOp for constrained geometrical modelling [Deuss et al., 2015], implemented in the second version of the plugin Kangaroo for Grasshopper [Piker, 2013].

Figure 6.21 show the resulting designs. The singularities are highlighted in pink, and the polyedges stemming from them are highlighted in black. The bar charts represent the performance for the different metrics.

#### 6.5.1.3 Numerical results

Table 6.2 shows the numerical results of each design for each objective. The lower the metrics, the better the design. The maximum values for each metric normalise the metrics. The results provide the statistics for the average, the standard deviation, and the minimum and maximum values, highlighted in green and red, respectively.

The designs present different trade-offs between the multiple objectives. The Pareto front consists of designs 0, 1, 14, 18 and 19. However, releasing the Pareto condition allows to consider designs that are quasi on the Pareto front. For a controlled performance loss, diversity in the pool of designs is maintained for upcoming considerations in the design process [Brown et al., 2015]. This quasi-Pareto condition is controlled by a factor $\alpha \leq 1$, the case $\alpha = 1$ yielding the Pareto front:

$$x \in P \iff \forall y, \alpha x_i \leq y_i \forall i, \exists i / \alpha x_i < y_i. \tag{6.21}$$

The quasi-Pareto front with $\alpha = 0.95$ adds designs 2, 5, 6, 7, 13, 22, 23, 24 and 25 to the strict Pareto front. The 95%-Pareto designs are underlined

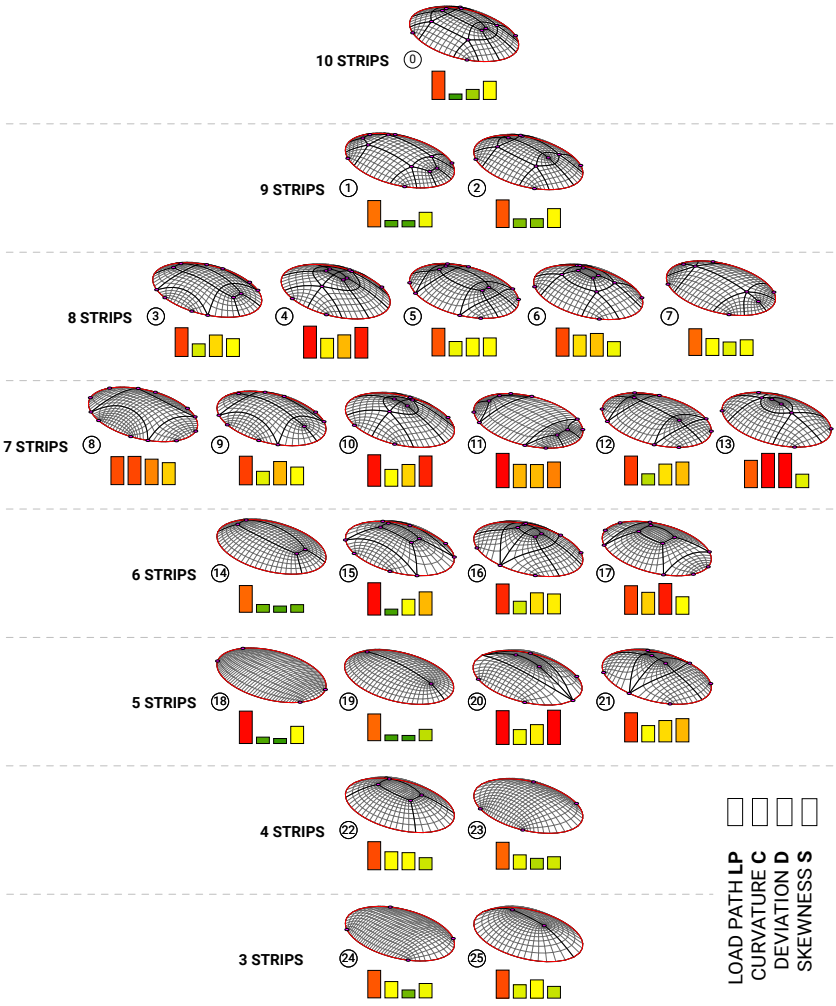



Figure 6.21 – Enumerating different topological designs for a dome on an elliptic boundary. The designs are organised based on topological similarity with design 0. All designs are close to a compression-only equilibrium against a vertical projected load and have quasi-planar faces. The bar charts show the multi-objective trade-offs. The 95%-Pareto designs are underlined.

Table 6.2 – Performance metrics of the 25 designs in Figure 6.21: normalised load path $\overline{LP}$, normalised panel curvature $\overline{C}$, normalised deviation due to planarisation $\overline{D}$ and normalised panel skewness $\overline{S}$. The 95%-Pareto designs are underlined.

| | $\overline{LP}$ [ - ] | $\overline{C}$ [ - ] | $\overline{D}$ [ - ] | $\overline{S}$ [ - ] |
|---|---|---|---|---|
| 0 | 0.83 | 0.16 | 0.29 | 0.53 |
| 1 | 0.77 | 0.18 | 0.18 | 0.42 |
| 2 | 0.81 | 0.25 | 0.26 | 0.55 |
| 3 | 0.84 | 0.37 | 0.63 | 0.52 |
| 4 | 0.93 | 0.58 | 0.68 | 0.89 |
| 5 | 0.81 | 0.42 | 0.52 | 0.53 |
| 6 | 0.82 | 0.62 | 0.66 | 0.42 |
| 7 | 0.78 | 0.49 | 0.39 | 0.46 |
| 8 | 0.82 | 0.82 | 0.74 | 0.64 |
| 9 | 0.84 | 0.39 | 0.68 | 0.52 |
| 10 | 0.92 | 0.49 | 0.64 | 0.88 |
| 11 | 1 | 0.68 | 0.67 | 0.75 |
| 12 | 0.84 | 0.32 | 0.61 | 0.67 |
| 13 | 0.8 | 1 | 1 | 0.39 |
| 14 | 0.78 | 0.23 | 0.18 | 0.22 |
| 15 | 0.94 | 0.17 | 0.46 | 0.68 |
| 16 | 0.87 | 0.37 | 0.61 | 0.58 |
| 17 | 0.83 | 0.64 | 0.9 | 0.51 |
| 18 | 0.95 | 0.19 | 0.15 | 0.5 |
| 19 | 0.78 | 0.17 | 0.15 | 0.33 |
| 20 | 0.99 | 0.43 | 0.58 | 1 |
| 21 | 0.85 | 0.48 | 0.62 | 0.68 |
| 22 | 0.82 | 0.53 | 0.5 | 0.35 |
| 23 | 0.78 | 0.42 | 0.32 | 0.36 |
| 24 | 0.8 | 0.48 | 0.23 | 0.42 |
| 25 | 0.82 | 0.39 | 0.53 | 0.34 |
| min. | 0.77 | 0.16 | 0.15 | 0.22 |
| max. | 1 | 1 | 1 | 1 |
| avrg. | 0.85 | 0.46 | 0.53 | 0.55 |
| st. dev. | 0.07 | 0.23 | 0.24 | 0.22 |

in Table 6.2 and Figure 6.21. The design options provide the designer with different choices in performance and shape.

The standard deviation is about three times lower for the load path metric than for the metrics. Indeed, for a fully supported structure, the structural performance of the different designs is similar.

### 6.5.2 Similarity from multiple designs

By combining designs with $n_i$ strips that have an intersection submesh with $n_0$ strips, $\prod_i 2^{n_i - n0}$ combinations of rules can be applied to explore designs with different degrees of similarity with the initial designs. However, the intersection submesh may not exist to create a reference to generate the union supermesh. Therefore, the designer can add strips to the initial designs to obtain the submesh and further hybrid meshes.

#### 6.5.2.1 Design problem

This section investigates the design of a quad-mesh pattern for a gridshell mapped on a pillow-like shape. The shape has a square boundary with a 10 m span and a 2 m height. The four corners are vertically and horizontally supported, and the structure has to withstand a central point load, as shown in Figure 6.22.

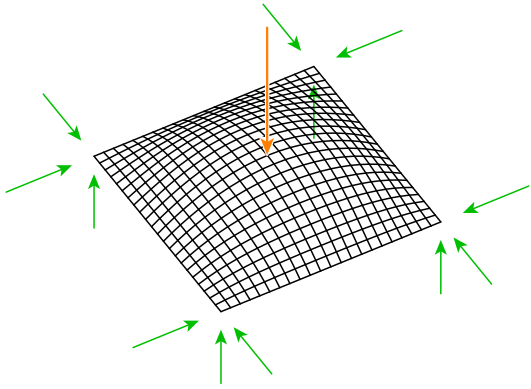


Figure 6.22 – Design of a pillow-shaped gridshell pinned at the corners and withstanding a point load.

The built-in steel S235 beams have an RO 114.3/4 tubular cross-section. The boundary cross-section RO 457.2/6.3 tubular cross-section, around a

100 times stiffer for the local stability of the free edge [Venuti and Bruno, 2018].

The performance objectives consist of three metrics to minimise. A relative edge-length variation $L$ measures geometrical regularity for fabrication:

$$L = (l_{max} - l_{min})/l_{tot}, \tag{6.22}$$

with $l_{max}$ the maximum edge length, $l_{min}$ the minimum edge length and $l_{tot}$ the total edge length. Two metrics measure the structural efficiency: the strain energy $E_M$ for a downward mesh load of 2 kN/m$^2$ and the strain energy $E_P$ for the point load of 100 kN. Comparing these two metrics allows differentiating the performance for the specific support conditions and loading conditions. The values are set to provide similar deflections while respecting the assumption for infinitesimal strain theory. A second-order mechanical analysis is performed, using the Finite Element Analysis plugin Karamba for Grasshopper3D [Preisinger, 2013].

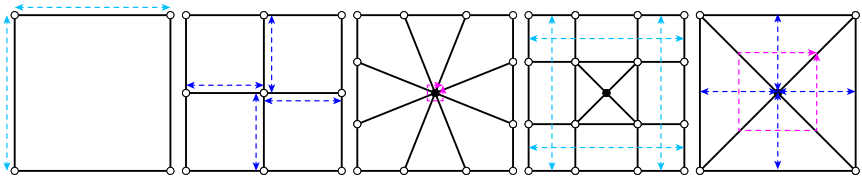
#### 6.5.2.2 Pattern design

The design starts from a regular quad-mesh grid. Figure 6.23 shows two heuristic designs obtained by applying addition and deletion rules on the initial design. Coloured dashed polylines highlight the input polyedges and strips of the rules. The colours show which strips are present in each mesh. The two-fold symmetry of the design problem is preserved by grouping the strips.

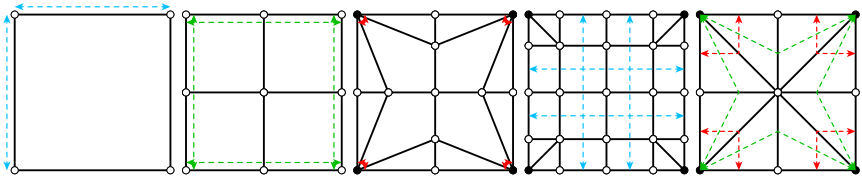
In Figure 6.23a, the temporary strips in cyan allow adding the strips in blue and pink to obtain a quad-mesh topology with a pole at the centre, to provide many load paths to the point load. Its distance from the initial mesh equals five, due to the strips blue and pink. In Figure 6.23b, the temporary strips in cyan allow adding the strips in green and red to obtain a quad-mesh topology with a pole at each corner, to provide many load paths to the supports. Its distance from the initial mesh equals eight, due to the strips green and red.

In order to combine these two topological designs, the intersection mesh (in Figure 6.24a) must be computed before obtaining the union supermesh (in Figure 6.24b). However, they do not share any strip structure. Therefore, addition rules create this common strip structure. This process is represented in Figure 6.25.

The intersection between the two initial designs does not yield any submesh because the set of their common strips is empty. Their distance equals

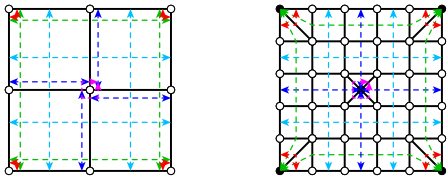


(a) Temporarily adding the strips in cyan to add the the strips in blue and pink to obtain a quad-mesh topology with a pole at the centre, to provide many load paths to the point load.



(b) Temporarily adding the strips in cyan to add the the strips in green and red to obtain a quad-mesh topology with a pole at each corner, to provide many load paths to the supports.

Figure 6.23 – Two quad-mesh topologies are generated starting from a simple coarse quad mesh.



(a) Intersection submesh (b) Union supermesh

Figure 6.24 – The reciprocal polyedges for addition in the submesh and strips for deletion in the supermesh. The colour scheme representing the strip groups takes into account the symmetry of the design.

eleven. Therefore, the four strips in cyan are added to provide this common strip structure. The intersection between the two new designs yields the submesh made of the four cyan strips in common. Their distance now equals thirteen. As a result, the union between the two new designs yields the supermesh with seventeen strips: the cyan strips of the intersection submesh, the blue and pink strips from one of the initial meshes and the green

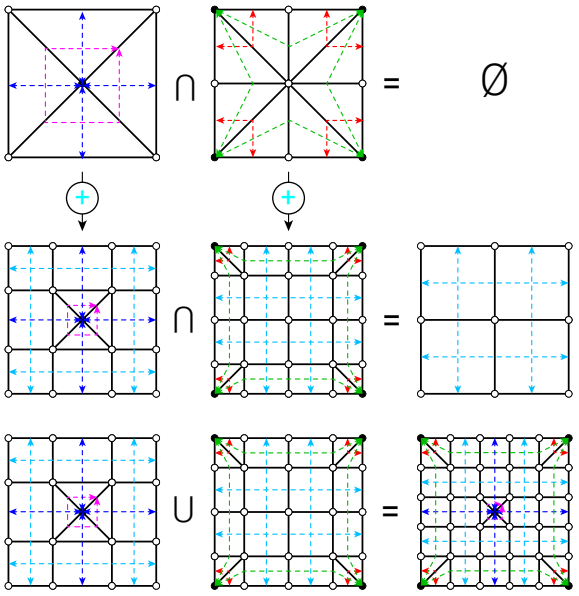


Figure 6.25 – The intersection between the initial designs does not yield any submesh due to the lack of common strip structure. Adding the strips in cyan provides such a structure to obtain an intersection submesh and a union supermesh.

and red strips of the other initial mesh.

The intersection submesh yields the union supermesh. The strip deletions to obtain the submesh yield polyedges that are used for the reciprocal strip additions to obtain the supermesh. Figure 6.24a represent the submesh and Figure 6.24b represents the supermesh with the coloured reciprocal polyedges and strips.

The seventeen strips in the super mesh represent five groups for $2^5 = 32$ combinations of strip deletions. The removal of the redundant isomorphic meshes and the meshes with a different shape topology results in the fourteen meshes with different strip structures in Figure 6.26. The two initial meshes 1 and 2 and the submesh 0 mark the corners of a triangular layout, the supermesh 13 marks the centre and the other meshes are arranged based on strip similarity. The fourteen meshes share different degrees of similarity with the initial meshes, which relate to fabrication and statics.

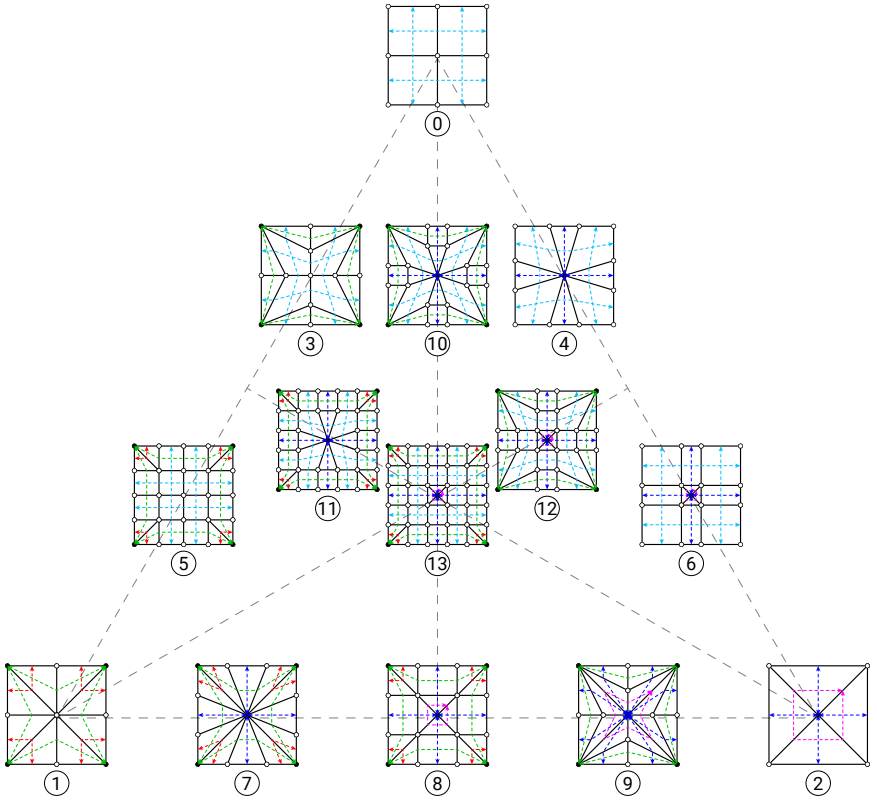



Figure 6.26 – Combination of the strip structures of the initial designs 0, 1 and 2. The fourteen designs are arranged based on topological similarity, highlighted by the common coloured strips.

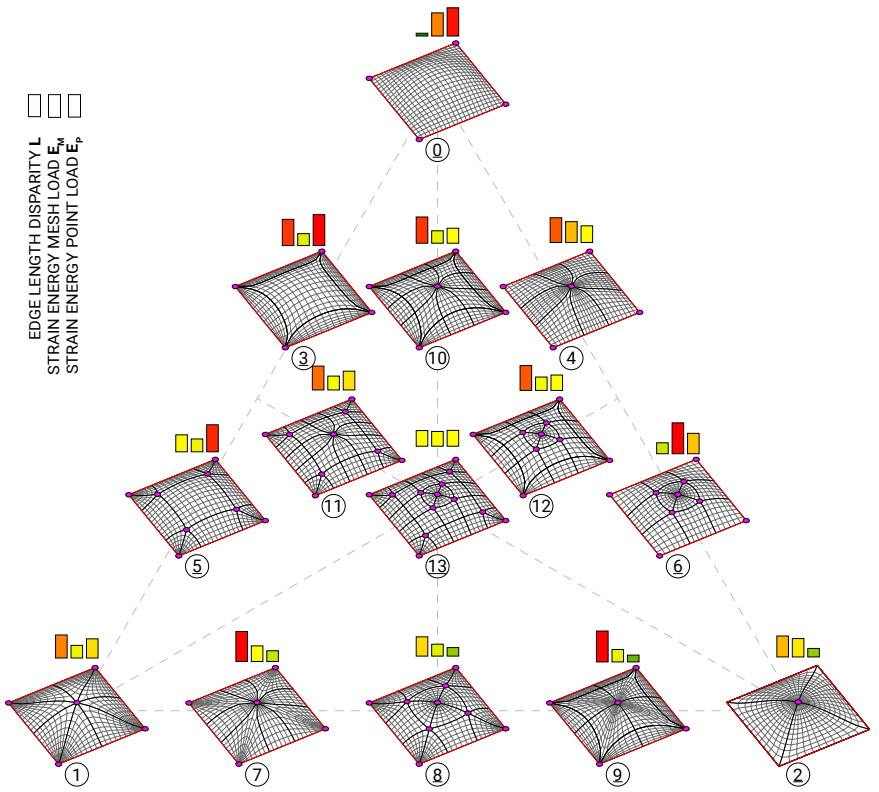



Figure 6.27 – Combining different topological designs for a pillow-shaped gridshell, supported on four corners. The designs are organised based on topological similarity with designs 0, 1 and 2. The designs provide different trade-offs between geometrical regularity and structural efficiency. The Pareto designs are underlined.

The coarse quad meshes are densified based on a target number of faces of 500, each strips having the same density parameter. Surface mapping and relaxation are performed using area-based Laplacian smoothing [Botsch et al., 2010]. The fourteen quad-mesh patterns are shown in Figure 6.27, arranged based on topological similarity, with their measured performance.

#### 6.5.2.3 Numerical results

The numerical results are provided in Table 6.3. The maximum value for each metric normalises the metrics. The lower the metric, the more efficient the design for this metric. The minimum and maximum values per metric are highlighted in green and red, respectively.

Table 6.3 – Performance metrics of the 14 designs in Figure 6.27: normalised edge-length disparity $\overline{L}$, normalised strain energy for a mesh load $\overline{E_M}$ and normalised strain energy for a point load $\overline{E_P}$. The Pareto designs are underlined.

| | $\overline{L}$ [ - ] | $\overline{E_M}$ [ - ] | $\overline{E_P}$ [ - ] |
|---|---|---|---|
| 0 | 0.09 | 0.75 | 0.91 |
| 1 | 0.75 | 0.41 | 0.62 |
| 2 | 0.68 | 0.59 | 0.28 |
| 3 | 0.85 | 0.39 | 1 |
| 4 | 0.8 | 0.67 | 0.54 |
| 5 | 0.55 | 0.42 | 0.87 |
| 6 | 0.35 | 1 | 0.66 |
| 7 | 0.97 | 0.51 | 0.35 |
| 8 | 0.62 | 0.39 | 0.28 |
| 9 | 1 | 0.4 | 0.22 |
| 10 | 0.85 | 0.4 | 0.49 |
| 11 | 0.77 | 0.45 | 0.61 |
| 12 | 0.8 | 0.43 | 0.5 |
| 13 | 0.51 | 0.48 | 0.51 |
| min. | 0.09 | 0.39 | 0.22 |
| max. | 1 | 1 | 1 |
| avrg. | 0.69 | 0.52 | 0.56 |
| st. dev. | 0.25 | 0.18 | 0.24 |

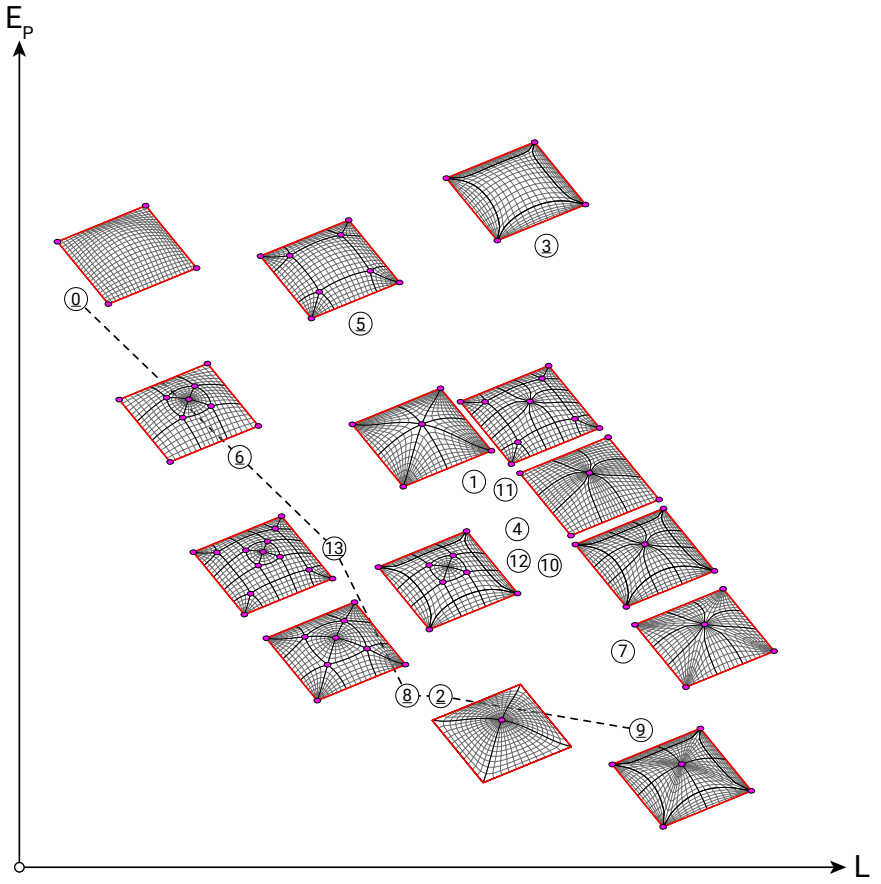



Figure 6.28 – Pareto front for the two metrics to minimised on edge-length disparity $L$ and strain energy for a point load $E_P$. The Pareto designs for the three metrics are underlined.

Designs 0, 1 and 2 are among the best designs for edge-length regularity, mesh-load energy and point-load energy, respectively. However, these designs do not perform well for the other two metrics. The designs offer different trade-offs between the multiple objectives. Design 13, from the supermesh, shows a balanced trade-off between the three objectives.

The Pareto front for the three metrics consists of designs 0, 2, 3, 5, 6, 8, 9 and 13, underlined in Table 6.3 and Figures 6.27 and 6.28. For the two metrics $L$ and $E_P$, the Pareto front consists of designs 0, 2, 6, 8, 9 and 13, represented as a dashed curve in Figure 6.28.

A data analysis provides a measure of the influence of each strip group on the different performance metrics. The influence measure is the covariance between the presence of a strip group in each design and its performance for each metric. The matrix in Table 6.4 provides whether a group of strips, represented by the same colour, are present in each of the designs.

These matrices provide the normalised covariance between the presence of the five strip groups and the performance of the three metrics. The resulting $5 \times 3$ matrix in Table 6.5 shows the relation between topology and performance. The covariance matrix highlights the positive, negative or lack of influence of a design choice on a performance design objective. In the matrix, a value $X_{ij}$ has a negative value if the strips $i$ reduce the performance metric $j$ and therefore improves the performance and vice versa.

The results are illustrated in Figure 6.29. The cyan strips from the initial quad-mesh grid improve the fabrication metric but worsen both statics metrics. The red and green strips from the design with corner poles worsen the fabrication metric but improve both statics metrics to different extents. The pink and blue strips of the design with a central pole improve the point-load strain energy, but the pink strips also improve the fabrication metric, and the blue strips worsen it.

The use of the covariance complements similarity-informed topology finding by informing quantitatively the designer on the relation between topology and performance.

Table 6.4 – Matrix of the strips per design. 0 and 1 mean that strips are absent or present, respectively.

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| cyan | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| red | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |
| green | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| pink | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| blue | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

Table 6.5 – Normalised covariance between topology and performance. The variance is computed for each metric per strip structure in the designs. Negative values, in green, mark an improvement and positive values, in red, mark a worsening.

| | norm. covar. $L$ [ - ] | norm. covar. $E_P$ [ - ] | norm. covar. $E_M$ [ - ] |
|---|---|---|---|
| cyan | - 1.5 | 1.9 | 4 |
| red | 0.3 | - 4.1 | - 0.5 |
| green | 4.4 | - 9.6 | - 0.8 |
| pink | - 0.6 | 1.5 | - 4.2 |
| blue | 2.7 | 1.2 | - 6.4 |

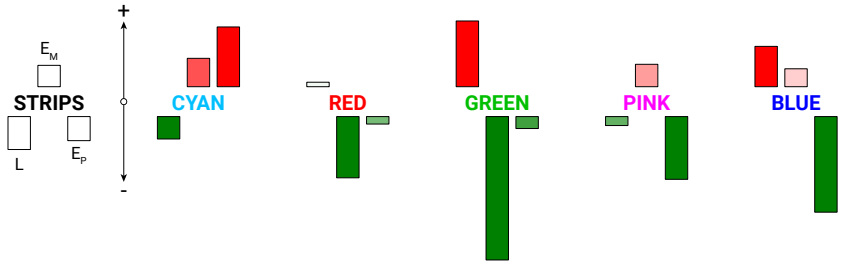



Figure 6.29 – Relation between topology and performance. The covariance provides data to see whether and how a group of strips of the same colour improves or worsens the different metrics in the considered design. The performance improves for a metric if it decreases, in green, and worsens if it increases, in red. For instance, the strips in cyan tend to improve the fabrication metric $L$ but worsen the statics metric $E_M$ and $E_P$.

## 6.6 Summary of contributions

This chapter introduced similarity-informed topology finding:

- an assessment was presented for topological similarity between two quad meshes with a topological distance;
- a strategy was developed for the combination of quad meshes to generate hybrid meshes with different degrees of similarity;
- this strategy was used to perform heuristic multi-objective design of quad-mesh patterns based on quad meshes designed for single objectives.

This approach enables informed exploration through a heuristic combination of previous designs that perform well for different objectives, completing initial open exploration.

The relation between topology and performance can be analysed using the covariance between strips and metrics among a set of designs for a specific application. Thereby, the designer is informed about the influence of design decisions.

Processing of the topological design influences the performance of the design. Figure 6.30 shows similarity-informed topology finding for a rib pattern on a plate supported on its four corners. The two load cases, symmetric loading and asymmetric loading, provide two sets of stress isolines for ideal ribs regarding these load cases. The singularities and their relation provide a coarse quad mesh and a pattern with equivalent topology. Figure 6.30 shows some of the intermediary meshes between these two topologies at a distance of fifteen. However, these meshes do not feature a trade-off in their structural performance for the two load cases. Indeed, the performance of the input and intermediary meshes is sensitive to the processing from topology to geometry. The position of the singularities, the density parameters and alignment to the cross-fields influence this performance. This example illustrates the challenge of geometry-sensitivity in performing similarity-informed topology finding. Combining topology finding and geometrical optimisation is necessary to full benefit from the parametric design space that relate to each topology [Maia Avelino et al., 2020].

Similarity-driven topology finding applies to the search of trade-offs between geometry-related objectives to improve. Nevertheless, some requirements are constraints to fulfil that depend on topology. Some structural systems require a specific organisation between their elements. However,

not any topology of a quad-mesh pattern fulfils this requirement. Therefore, exploration must be constrained to the subspace of feasible topological designs.

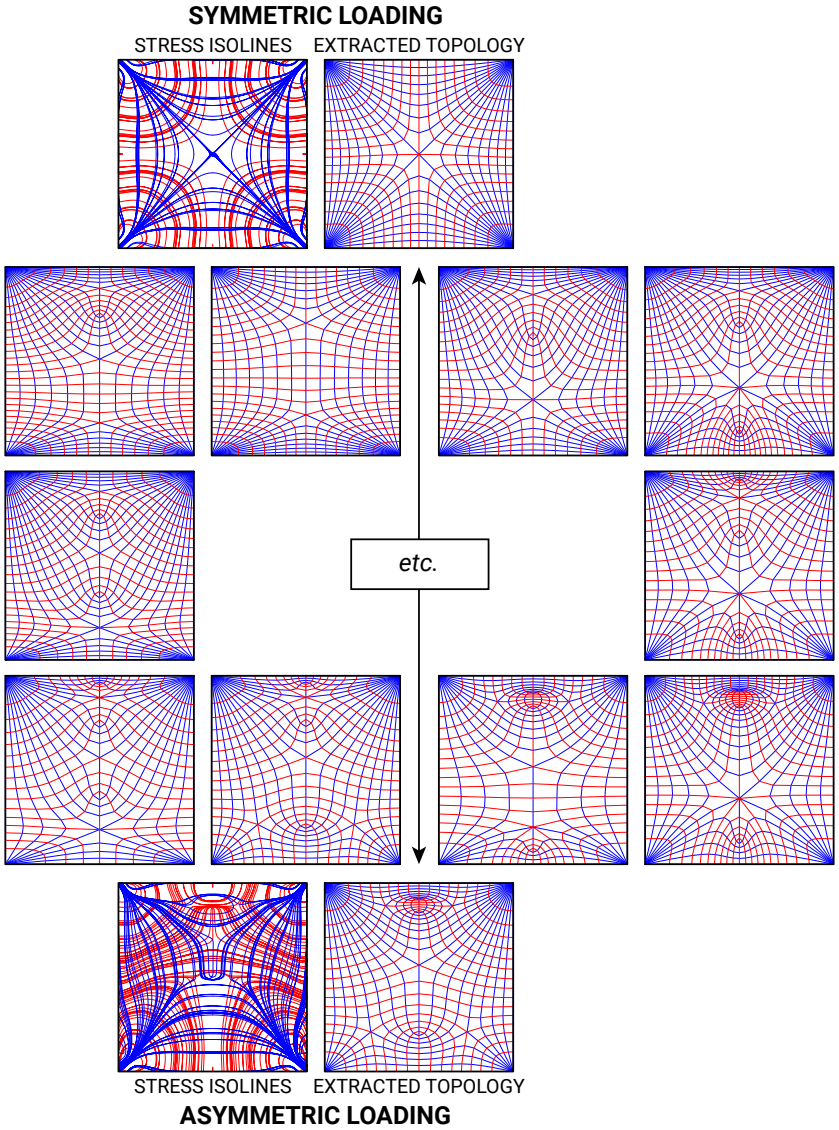



Figure 6.30 – Similarity-informed topology finding based on structural performance against multiple load cases. Obtaining a performance trade-off is sensitive to parameters of geometrical processing and may require geometrical optimisation.

# Chapter 7

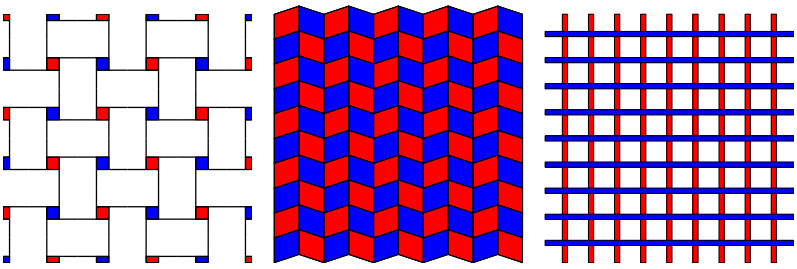
# Two-colouring exploration

## 7.1 Motivations

Several structural systems rely on a partition of their elements into two groups. Such an organisation relies on the partition of the elements into two groups without having two elements of the same group connected. This partition property is further called *two-colouring*. Two-colouring relates to colouring the elements using only two colours without having adjacent elements of the same colour. Figure 7.1 illustrates this property for different systems with partitions of nodes, panels or beams grouped in red and blue. This partition also occurs at the scale of the structure of orthotropic materials, for the directions with different properties, as in wood, composite or textile material.

This property is topological, independent from the geometry of the pattern.

A necessary condition for a nexorade to have an alternation of nodes turning right and left is the two-colouring of its nodes (Figure 7.1a). This alternation provides uniformity and preserves symmetry, as opposed to a design with non-alternated nodes [Mesnil et al., 2018a]. A necessary condition for origami to be flat-foldable is the two-colouring of its faces in a checkerboard pattern, as demonstrated by Maekawa's theorem [Hull, 1994] (Figure 7.1b). This property guarantees the alternation of panels facing upwards or downwards when folded. A necessary condition for an elastic gridshell to be made of two independent sets of continuous beams forming the top and the bottom layers, respectively, is the two-colouring of its beams (Figure



(a) Alternating nodes (b) Folding panels (c) Crossing beams

Figure 7.1 – Two-coloured organisation of elements in patterns based on the partition into two groups, in red and blue, without having elements of the same group connected to each other.

7.1c). This partition avoids having continuous beams weaving between the two layers, in order to ease the planar-layout process and avoid inducing additional bending pre-stresses due to the lifting process [Du Peloux et al., 2015].

Table 7.1 provides a literature review of the structural systems for which pattern elements respect a two-colouring property to provide a binary partition between two status.

However, not all quad meshes can be two-coloured. The three-valent singularities in Figure 7.2 do not allow this alternation between two groups of elements, in red and blue. Therefore, a third group, in green, is necessary to avoid having elements of the same group overlapping each other. This topological design does not allow the partition of continuous beams of an elastic gridshells into two layers with having some beams weaving between the two layers.

Quad-mesh patterns resulting from the integration of cross fields such as the principal curvature directions [Liu et al., 2006, Schiftner et al., 2012, Takezawa et al., 2016] and the principal stress directions [Schiftner and Balzer, 2010] fulfil this two-colouring property by definition. Indeed, two groups of non-overlapping elements partition the resulting polyedges. The groups correspond to the two directions in the cross-field, as in Figure 7.3 for two patterns stemming from the principal stress directions for different loading conditions. Note that the singularities in pink that are off the boundary have even valencies, mainly two or six, sometimes eight, a mathematical/physical property of these cross-fields. Nevertheless, designing two-coloured patterns relying on a cross-field is limited and not flexible.

Table 7.1 – Review of structural systems based patterns with two-coloured elements.

| Structural system | Pattern element | Binary status | References |
|---|---|---|---|
| Reciprocal structures | Vertex | Rightwards or leftwards turn | [Baverel, 2000, Mesnil et al., 2018a, Brocato and Mondardini, 2010, Estrin et al., 2011] |
| Origami | Face | Mountain or valley fold | [Hull, 1994, Tachi, 2009, Lebée, 2015] |
| Elastic gridshells | Polyedge | Top or bottom layer | [Otto et al., 1974, Douthe et al., 2017, Marquis et al., 2017, Avelino and Baverel, 2017] |
| Woven structures | Polyedge | Warp or weft direction | [Baverel and Popovic Larsen, 2011, Popescu et al., 2017, Ayres et al., 2018] |
| Membranes | Polyedge | Strip width or length | - |
| Corrugated shells | Polyedge | Creasing direction | [Norman et al., 2009, Malek and Williams, 2017] |
| Folded shells | Face | Folding direction | [Stitic and Weinand, 2015, Mesnil et al., 2017a] |
| Developable envelopes | Polyedge | Ruling direction | [Liu et al., 2006, Schiftner et al., 2012] |
| Circular and cyclidic meshes | Face | X or Y axis | [Liu and Wang, 2008, Bobenko and Huhnen-Venedey, 2014, Mesnil et al., 2017c] |

Caigui *et al.* [Caigui et al., 2019] generate two-coloured checkerboard patterns following the diagonals of quad meshes. This approach is equivalent to applying the ambo Conway operator on any mesh producing two families

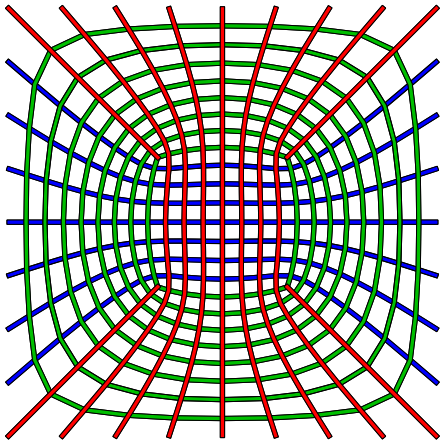
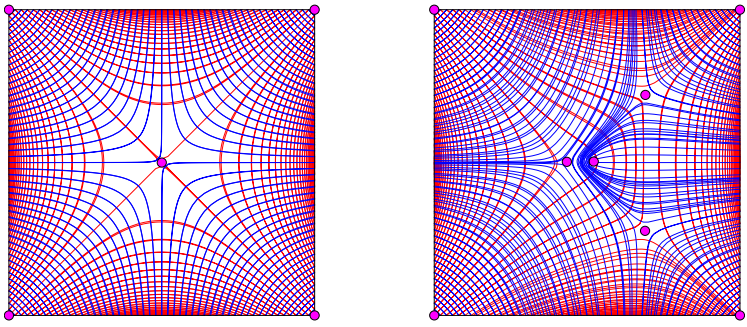


Figure 7.2 – A quad-mesh pattern with a three-valent singularity is an example of a topology that can not be two-coloured, in red and blue. A third group of elements, in green, is necessary to avoid having elements of the same group overlapping each other. This topology is not suitable for the design of an elastic gridshell made of continuous beams organised in two layers without having beams weaving between the two layers.



(a) Symmetric loading (b) Asymmetric loading

Figure 7.3 – Integration of cross fields yields two-coloured patterns by definition. Each cross field direction corresponds to one of the two groups of elements, in red or blue. The patterns stem from the principal stress directions for different loading conditions.

of faces: one from the initial vertices and one from the initial faces. However, the resulting patterns are not quad meshes due to singularities in the vertices or faces in the initial quad mesh, which translate into singular faces in the final mesh.

For the mentioned structural systems, the previous strategies for comprehensive and informed exploration must be coupled with search algorithms for constrained exploration to two-coloured quad-mesh patterns. These resulting patterns should share some degree of similarities with the initial patterns, according to the topological distance, to preserve the design intent. In order to provide exploration freedom to the designer, not one but several patterns should be provided to offer different design directions. This search relates to finding the projection of a pattern that can not be two-coloured onto the two-coloured subspace.

This chapter introduces *two-coloured topology finding*. Section 7.2 provides the requirements and characterisations for two-colouring of the different elements in quad meshes. Section 7.3 develops a projection algorithm to find the most similar two-coloured quad meshes from a quad mesh that can not be two-coloured. The algorithm is tested and analysed on some validation examples. Section 7.4 applies this search to two-coloured topology finding of patterns for folded shells.

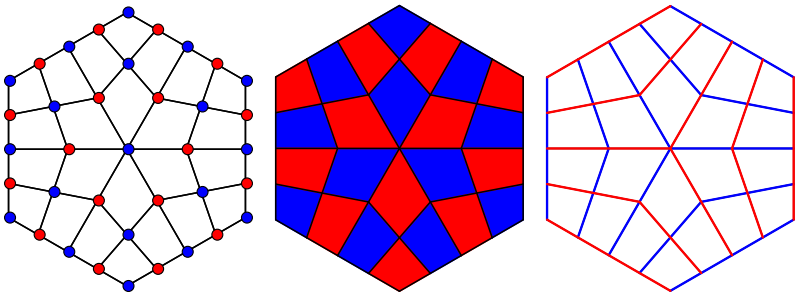
## 7.2 Two-colouring characterisation

For the search of two-coloured quad meshes, characterisation strategies on the requirements of the different types of two-colouring are necessary.

### 7.2.1 Types of two-colouring

Different types of two-colouring exist for quad meshes, which do not entail the same requirements. Map colouring inspires the term *colouring*, coming from graph theory [Gondran and Minoux, 1984]. The chromatic number of a graph is the minimum number of colours that can be used to colour all the vertices while no adjacent vertices have the same colour. In Figure 7.4a, the vertices of the mesh can be two-coloured as only two colours are necessary. Colouring is a labelling operation, where the colours can be replaced by the relevant binary status to encode specific data, like 'up' versus 'down' or 'left' versus 'right'.

Similarly, face colouring relates to colouring all faces while no adjacent faces have the same colour, as in Figure 7.4b. Polyedge colouring relates to



(a) Vertices (b) Faces (c) Polyedges

Figure 7.4 – The three types of two-colourability for quad-mesh patterns.

colouring all polyedges while no crossing polyedges have the same colour, as in Figure 7.4c. Connected extremities of polyedges do not count as crossings.

Previously, polyedges referred to a set of continuous edges in any mesh or graph. In this chapter, polyedges refer specifically to quad-mesh polyedges. Quad-mesh polyedges connect edges that are topologically opposite to each other across the regular four-valent vertices. They stop at singularities, at boundaries or when forming a loop. Quad-mesh polyedges are dual elements to quad-mesh strips. Figure 7.5 shows this relationship in a quad mesh $M$ and its dual mesh $M^*$ between two pairs of strips and polyedges $P$ and $S^*$, and $S$ and $P^*$ in blue and red.

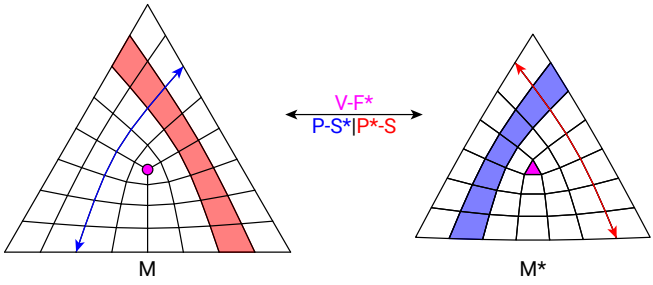



Figure 7.5 – Dual relation in a quad mesh $M$ and its dual $M^*$ between topologically continuous polyedges $P$ and strips $S$, in blue and red. Singular vertices $V$ become singular faces $F*$ in the dual mesh, which is therefore not a quad mesh but a quad-dominant mesh.

Quad mesh vertex and face colouring do not correspond to the same problem on two dual meshes, despite their dual relation. Indeed, the dual mesh of a quad mesh is, a priori, not a quad mesh, as singular vertices yield singular faces that are not quads. In Figure 7.5 the dual of the three-valent vertex $V$ is a three-valent face $F^*$ highlighted in pink, which breaks the quad-face condition, becoming a quad-dominant mesh with primarily quad faces.

Not any quad mesh guarantees these two-colouring properties.

### 7.2.2 Two-colouring requirements

A quad mesh can respect all, some or none of these three types of two-colouring.

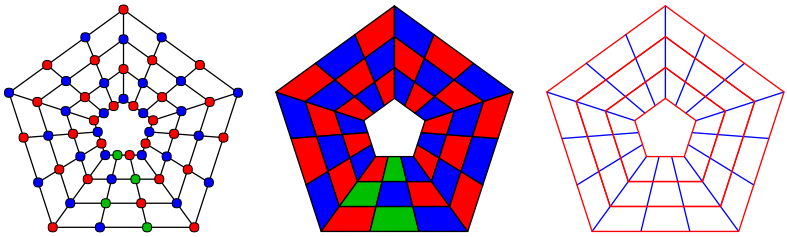
The quad mesh in Figure 7.6 has two-coloured polyedges. Nevertheless, its vertices and faces can not be two-coloured. Indeed, the closed strips have an odd subdivision and do not allow this binary alternation of the even number of vertices and faces along it. The quad mesh in Figure 7.7 has two-coloured vertices. Nevertheless, its faces and polyedges can not be two-coloured. Indeed, the singularity has an odd valency and does not allow this binary alternation of the five faces and polyedges around it. The quad mesh in Figure 7.8 mixes odd-subdivided strips and odd-valency singularities. Therefore, its vertices, faces and polyedges can not be two-coloured.

The different types of two-colouring depend on different topological aspects of a quad mesh: its singularities and its density. Table 7.2 summarises these dependencies.

Table 7.2 – Dependencies between two-colouring types and topological aspect in quad meshes.

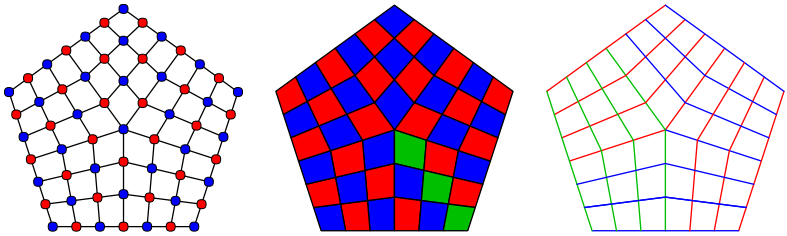
| Two-colourability | Vertices | Faces | Polyedges/strips |
|---|---|---|---|
| Singularity requirement | No | Yes | Yes |
| Density requirement | Yes | Yes | No |

The requirements for two-colouring are treated separately using a coarse quad mesh. On the one hand, the density requirement depends on the density of the strips. On the other hand, the singularity requirement depends on the combination of the strips.



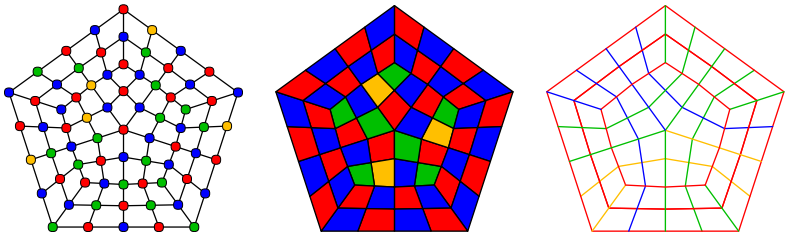
(a) Vertices (b) Faces (c) Polyedges

Figure 7.6 – Quad mesh with two-coloured polyedges but without two-coloured vertices or faces due to the odd number of elements along the closed strips.



(a) Vertices (b) Faces (c) Polyedges

Figure 7.7 – Quad mesh with two-coloured vertices but without two-coloured faces or polyedges due to the odd number of elements around the singularity.



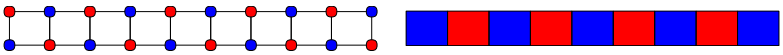
(a) Vertices (b) Faces (c) Polyedges

Figure 7.8 – Quad mesh without two-coloured vertices, faces or polyedges due to the odd number of elements along the closed strips and around the singularity.

#### 7.2.2.1 Density requirement

Vertex and face two-colouring depend on density, on the contrary to polyedge two-colouring.

If the quad mesh only has open strips, vertex and face two-colouring applies. In Figure 7.9, any density parameter can be chosen for the open strip.



(a) Vertices (b) Faces

Figure 7.9 – The vertex and face elements along an open strip can always be two-coloured.

If the quad mesh has closed strips, an even number of elements must cross each of them. This requirement sets a constraint on the sum of the density parameters $d_j$ of the strips $j$ crossing the closed strip $i$ to be even. A strip counts each time it crosses the closed strip, as there can be multiple crossings $x_{ij}$. This constraint formalises as:

$$\forall i \in S_c, \sum_{j \in S_i^{\perp}} x_{ij} d_j \propto 2, \tag{7.1}$$

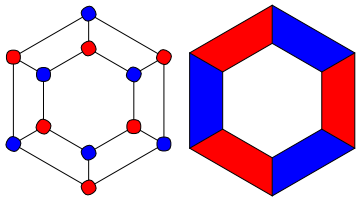
where $S_c$ is the set of closed strips and $S_i^{\perp}$ the set of strips crossing the strip $i$.

In Figure 7.10, the closed strip is crossed by six elements so its vertices and faces are two-coloured. However, in Figure 7.11, the closed strip is crossed by five elements and does not fulfil the constraint. Therefore, the density of the crossing strips is corrected to provide two-coloured vertices and faces.

Characterisation and fulfilment for the density requirement are more direct than for the singularity requirement.

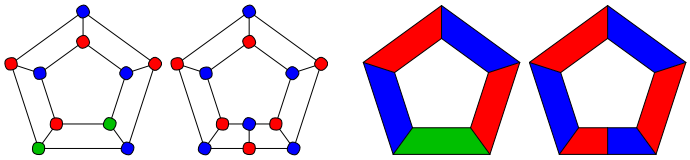
#### 7.2.2.2 Singularity requirement

Face and polyedge two-colouring depend on the singularities, on the contrary to vertex two-colouring. Indeed, the number of faces and polyedges looping around a singularity in a quad mesh is equal to its valency $n$ and is not necessarily even. On the contrary, the number of vertices is equal to twice its valency $2n$ and is therefore even.



(a) Vertices (b) Faces

Figure 7.10 – The vertex and face elements along a closed strip can be two-coloured if crossed by an even number of elements.



(a) Vertices (b) Faces

Figure 7.11 – The vertex and face elements along a closed strip can not be two-coloured if crossed by an odd number of elements. This number of elements can be tuned into an even number by modifying the density of the crossing strips.

A quad mesh fulfils the singularity requirement for two-colouring under two conditions. The number of strips both around each non-boundary vertex and along each boundary in the coarse quad mesh should be even. The requirement on the boundaries depends on all the singularities along each boundary. This requirement is equivalent to strip two-colouring: strips should be coloured using only two colours without having crossings between strips of the same colour.

The topology in Figure 7.12 fulfils this two-colouring requirement because there are six strips around the non-boundary vertex in pink and six strips along the boundary.

Otherwise, two-coloured alternation is not possible. In Figure 7.13, three colours are necessary to colour the three strips around the inner vertex in Figure 7.13a and along the boundaries in Figure 7.13b without crossings of strips of the same colour. Therefore, these topologies do not fulfil the singularity requirement.

Two complementary approaches characterise the singularity requirement

for two-colouring, with different pros and cons for the search of two-coloured quad-mesh patterns.

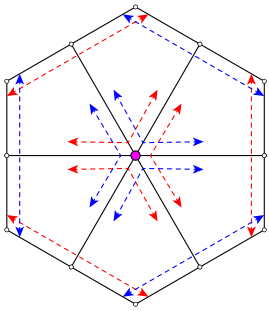
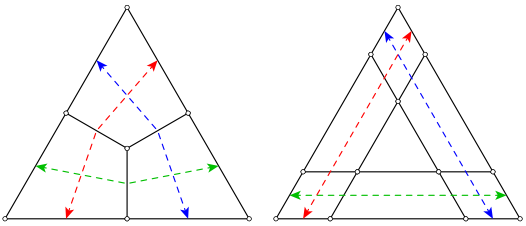


Figure 7.12 – The even numbers of strips around the non-boundary vertex in pink and along the boundary guarantee the fulfilment of the singularity requirement for two-colouring.



(a) Around a non-boundary vertex (b) Along a boundary

Figure 7.13 – Non-two-colourable coarse quad meshes because of odd numbers of strips.

### 7.2.3 Index-based characterisation

The two conditions on the even number of strips around the inner vertices and along the boundaries can be translated from the number of strips to the indices of the vertices, detailed in Section 2.4.2.3. An even number of strips around each vertex $v$ in the set the non-boundary vertices $V \setminus \partial V$ translates into their index $i_v$ being proportional to $1/2$:

$$\forall v \in V \setminus \partial V, i_v \propto 1/2, \tag{7.2}$$

and an even number of strips along each boundary $\partial V_i$ translates into the sums of vertex indices $i_v$ along each boundary being proportional to $1/2$:

$$\forall \partial V_i \in \partial V, \sum_{v \in \partial V_i} i_v \propto 1/2. \tag{7.3}$$

This vertex-based characterisation allows to directly assess if a quad mesh fulfils the singularity requirement for two-colouring. A metric can be derived to provide a measure on the degree of fulfilment of this requirement integrating the two conditions in Equations 7.2 and 7.3:

$$\sum_{v \in V \setminus \partial V} i_v \bmod 1/2 + \sum_{\partial V_i \in \partial V} \sum_{v \in \partial V_i} i_v \bmod 1/2. \tag{7.4}$$

This characterisation does not provide direct information on how to fulfil the singularity requirement for two-colouring, event though this metric can serve for stochastic search.

Another characterisation, more suitable for direct search, is expressed on the strip graph.

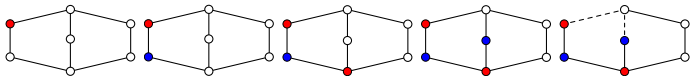
### 7.2.4 Graph-based characterisation

The singularity requirement can be expressed as colouring all the strips with only two colours without crossings between strips of the same colour. This problem is equivalent to vertex colouring of the strip graph, introduced in Section 5.2.3. Indeed, each graph vertex encodes a strip, and each graph edge encodes a strip crossing over a face.

Vertex colouring is a classic problem of graph theory that aims at finding the chromatic number. The existing algorithms offer different benefits and drawbacks in terms of speed and robustness to solve this NP-complete problem [Dailey, 1980, Gondran and Minoux, 1984]. However, determining the chromatic number is not necessary here. Assessing whether the graph is two-colourable or not is sufficient.

The two-colouring process starts by colouring one vertex. The iterative process colours the vertices that are not coloured and adjacent to a coloured vertex. If an uncoloured vertex can not be coloured because already adjacent to vertices of both colours, the process returns a False statement. If all vertices are assigned a colour, the process returns a True statement. In Figure 7.14a, two-colouring fails at the fifth vertex, already adjacent to

blue and red vertices, highlighted with dashed edges. In Figure 7.14b, two-colouring succeeds after colouring the six vertices.



(a) Fail if one vertex is adjacent to vertices of the two colours

(b) Success if all vertices are coloured

Figure 7.14 – Iterative two-colouring of the vertices of a graph.

The two-colouring algorithm is detailed in Algorithm 2.

```
store a starting vertex in a first-in, first-out list of vertices to colour
while the list of sources is not empty do
    get and remove one vertex from the list
    if the neighbour vertices already use both colours then
        return False
    end
    else
        colour the vertex with one colour not used by the neighbour
         vertices
        add the uncoloured neighbour vertices to the list of vertices
         to colour
    end
end
return True
```

**Algorithm 2:** Assessing the two-colourability of the vertices of a graph.

The complexity of the algorithm equals the number of graph vertices multiplied by their valency. The $N \times N$ quad-mesh grid in Figure 7.15 is a specific case of two-coloured quad mesh, as each strip of each group crosses exactly once each strip of the other group. The mesh has $2N$ strips and $N^2$ faces. Therefore, the strip graph has $2N$ vertices, one per strip, and $N^2$ edges, one per face. The strip graph has the topology of a complete bipartite graph, noted $K(N, N)$. Such a graph has two partitions of $N$ vertices, each connected to all the vertices of the other partition [Gondran

and Minoux, 1984]. A bipartite graph is another name for a graph with two-coloured vertices. The $2N$ vertices connect to the $N$ other vertices of the other partition. Therefore, the algorithm has a quadratic time complexity $O(N^2)$.

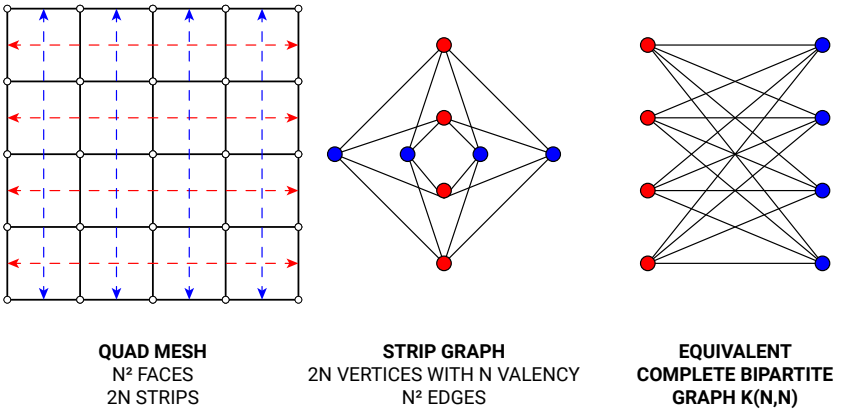


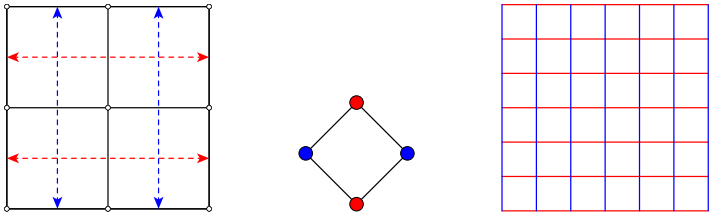

Figure 7.15 – A $N \times N$ quad-mesh grid has a strip graph equivalent to a complete bipartite graph $K(N, N)$, with $2N$ vertices connected to $N$ vertices, resulting in a quadratic time complexity $O(n^2)$ for the two-colouring algorithm.

In Figure 7.16, the coarse quad mesh has two-coloured strips, evidenced by its two-coloured strip graph. Therefore the dense quad mesh fulfils the singularity requirement as well and has two-coloured polyedges. This graph-based characterisation is in line with the index-based characterisation. Indeed, the four-valent inner vertex has an index of 0, which is proportional to $1/2$, and the sum of the indices of the four three-valent and four two-valent vertices along the boundary equals 1, which is proportional to $1/2$. Therefore the singularity requirement for two-colouring is fulfilled.

In Figure 7.17 on the contrary, the coarse quad mesh does not fulfil this requirement, as shown by its three-coloured strip graph. For representation, full colouring is completed using the greedy Welsh-Powell algorithm [Welsh and Powell, 1967]. This graph-based characterisation is in line with the index-based characterisation. Indeed, the five-valent inner vertices have an index of -1/4, which is not proportional to $1/2$. Therefore the singularity requirement for two-colouring is not fulfilled.

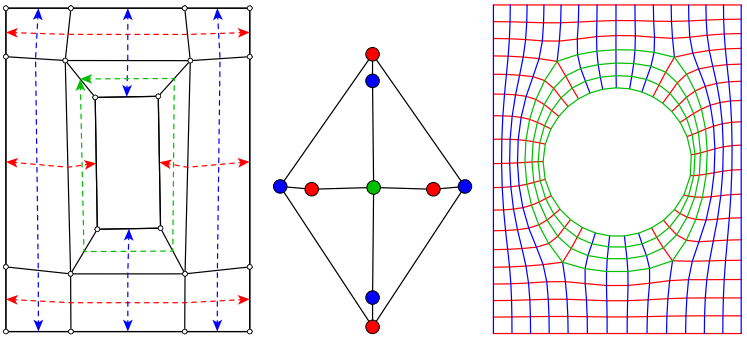
This graph-based characterisation of the singularity requirement for two-

colouring of quad meshes can drive exploration. A search algorithm is developed to find the most similar two-coloured quad-mesh patterns from an initial pattern that does not fulfil this requirement.



(a) Mesh (b) Graph (c) Polyedges

Figure 7.16 – Quad mesh with two-coloured polyedges characterised by the two-coloured strip graph of its coarse quad mesh.



(a) Mesh (b) Graph (c) Polyedges

Figure 7.17 – Quad mesh without two-coloured polyedges evidenced by the three-coloured strip graph of its coarse quad mesh.

## 7.3 Two-colouring search

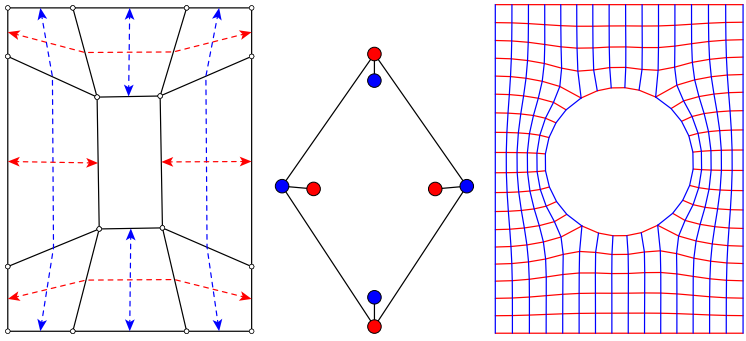
To perform two-coloured topology finding, a search algorithm is developed using the strip grammar rules, introduced in Chapter 5.4. Starting with a quad mesh that can not be two-coloured, the most similar two-coloured quad meshes are found, according to the topological distance defined in Chapter

6. This search for the closest two-coloured quad mesh is a projection onto the two-coloured design subspace. Moreover, the projection not only yields the closest design but the closest ones in several directions that provide different independent options to the designer.

The application of strip rules drive this search for two-coloured quad meshes, informed by the colouring of the strip graph.

### 7.3.1 Third-colour deletion

The algorithm provided in Algorithm 2 returns whether the vertices of a graph can be two-coloured. The design in Figure 7.16 fulfils this requirement. However, the design in Figure 7.17 does not. The graph can not be two-coloured, but completing the colouring process with a general colouring algorithm like the Welsh-Powell algorithm [Welsh and Powell, 1967] informs on a solution to make this design two-coloured. The graph has one vertex of the third colour green. Deleting the corresponding third-colour strip yields the design in Figure 7.18 that can be two-coloured, evidenced by the strip graph with one vertex less due to the strip deletion. Deleting the strip moved the five-valent non-boundary singularities to the boundary, along which the sum of the vertex indices is proportional to $1/2$, in line with the vertex-base characterisation.



(a) Mesh (b) Graph (c) Polyedges

Figure 7.18 – Quad mesh with two-coloured polyedges after deletion of the third-colour strips, evidenced by new two-coloured strip graph of its coarse quad mesh.

However, multiple combinations exist to colour a graph with three colours or more. Figure 7.19 hints at this richness that goes beyond the unique option yielded by a colouring algorithm. The nine coloured graphs suggest different sets of third-colour vertices. They each correspond to a different set of strip deletion rules to apply to yield a two-coloured quad mesh.

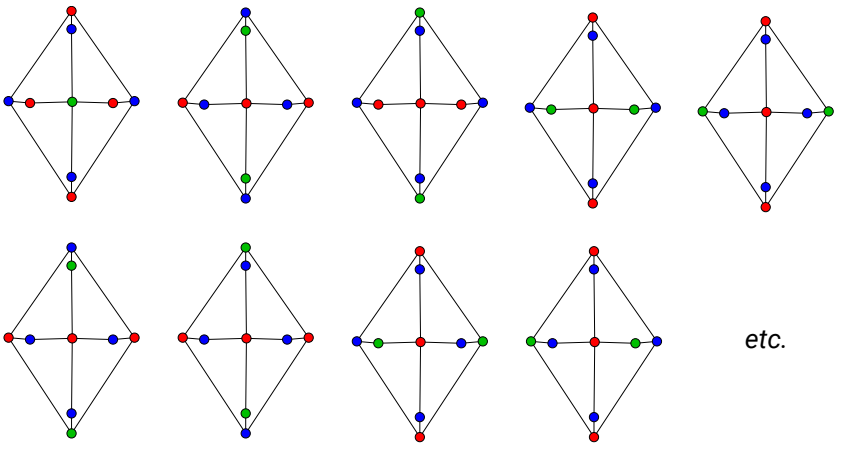



Figure 7.19 – The combinatorial richness in colouring a graph using more than two colours provides multiple combinations to delete third-colour strips.

This combinatorial richness opens up an opportunity to yield multiple quad meshes of the two-coloured subspace. The search algorithm proposes a strategy to sort out the most similar quad meshes that provide different design directions. This approach relates to a projection to the two-coloured subspace.

### 7.3.2 Two-colour projection

The distance between two quad meshes is the minimal number of low-level strip rules to apply to go from one to another.

The search algorithm only applies deletion rules. Indeed, adding a strip adds a vertex and edges to the strip graph. The old and new strip graph are a subgraph and a supergraph to each other, respectively. If the subgraph can not be two-coloured, nor can its supergraph.

The missing reciprocal addition rule does not allow a non-linear search.

The search is therefore oriented by the successive application of deletion rules, defining the search direction.

Figure 7.20 shows the directed graph for the search through the $2^3 = 8$ combinations for the deletion of three strips.

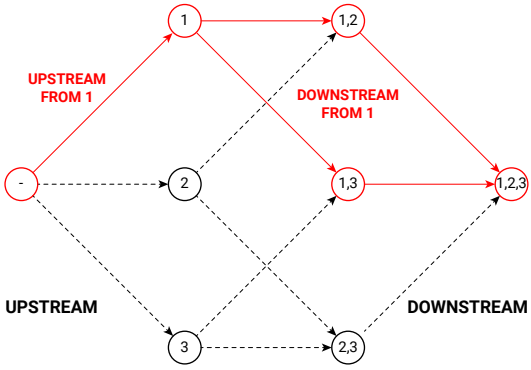



Figure 7.20 – This graph represents the search organisation through the combinations of three deletion rules 1, 2 and 3. The search algorithm applies only deletion rules, defining an orientation the upstream combination (-) to the downstream combination (1,2,3). For instance, (1,2) is in the downstream direction of (1) but (2,3) is not in the direction of (1).

The empty combination is upstream to all the combinations and the complete combination (1,2,3) is downstream to all the combinations. Considering a specific combination like (1), one part of the graph is upstream, namely the combination (-) and one part of the graph is downstream, namely the combinations (1,2), (1,3) and (1,2,3). Any other combination lies in a different direction of the graph compared to combination (1). The search algorithm limited to deletion rules can not obtain the mesh corresponding to combination (2) from the mesh corresponding to combination (1). Nevertheless, the search can obtain the mesh corresponding to combination (1, 2) from the mesh corresponding to combination (1) by deleting strip 2. Indeed, combination (1,2) is part of the downstream direction from combination (1).

Finding the closest or most similar two-coloured design is a projection to the design subspace of two-coloured designs. This projection $P$ applies to one element of the space of quad meshes $E$ and yields another one, or several ones if there is equidistance to several two-colourable designs:

$$P : E \to E. \tag{7.5}$$

If a quad mesh can already be two-coloured, applying two-colour projection yields the same quad mesh. Therefore, the idempotence definition of a projection is respected:

$$P^2 = P. \tag{7.6}$$

In order to find the closest designs first, the number of strip deletions applies in increasing order. All combinations of strip deletions are tested starting with one strip, then two strips, until the maximum number of strips. A seven-strip quad mesh that can not be two-coloured serves as example in Figures 7.21 to 7.23. The figures show completely coloured strip graphs for better understanding, though not needed for computing.

In Figure 7.21, some strip deletions yield two-coloured quad meshes, as deleting strips 1 and 2, but some do not, as deleting strip 3. Here, the resulting two-coloured quad meshes are at a distance of one from the input quad mesh.

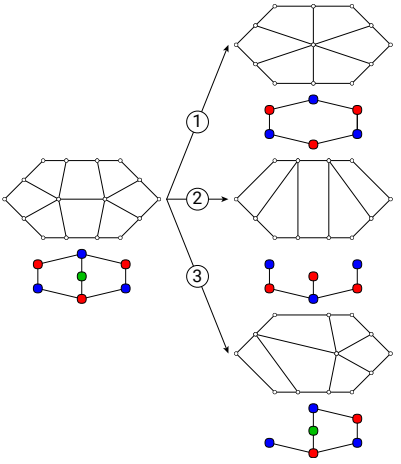



Figure 7.21 – Deleting one strip in a quad mesh that can not be two-coloured can yield two-coloured quad meshes or not.

The application of more deletion rules on a two-coloured quad mesh yields another two-coloured quad mesh. In Figure 7.22, deleting strips 1 yields a two-coloured quad mesh at a distance of one, therefore deleting strips 1 and 3 as well but at a distance of two. Such a combination of rules is redundant because it yields another two-coloured design in the same direction but at a farther distance in the two-coloured subspace.

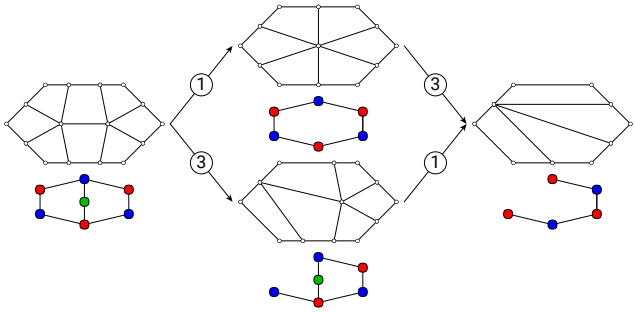



Figure 7.22 – Deleting a second strip of a two-coloured quad mesh is redundant because it yields another two-coloured quad mesh, but at a farther distance and in the same direction.

Nevertheless, combining multiple deletion rules allows finding two-coloured quad meshes at farther distances but in different directions. In Figure 7.23, deleting strips 3 and 6 separately does not yield a two-coloured quad mesh but combining them yields a two-coloured quad mesh at a distance of two. Deleting strip 1, for instance, provides a two-coloured quad mesh at a distance of one, but in another direction. Hence, the interest in such farther two-coloured quad meshes to provide an alternative to the designer.

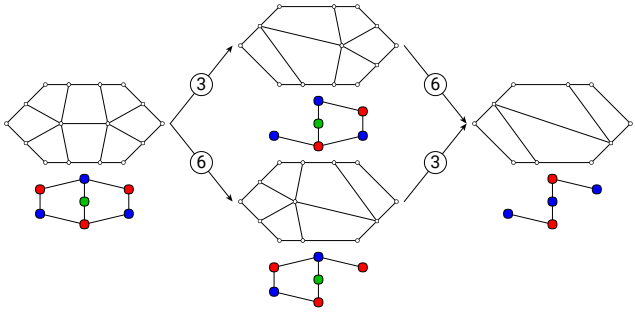



Figure 7.23 – Deleting two strips separately may not yield a two-coloured quad mesh, but can yield a two-coloured quad mesh at a farther distance in a new direction.

Figure 7.24 illustrates the principle of the algorithm for finding the closest designs in the different design directions in the search through the $2^3 = 8$ combinations for the deletion of three strips The result from strip deletions

in pink can be two-coloured, and the ones in black can not. Finding a two-coloured design removes all of the downstream designs in the same direction from the search algorithm, as highlighted by dashed circles and lines. The output is the closest two-coloured design in their respective directions, represented by full pink circles. Deleting strips (1) yields a two-coloured design, so the search does not investigate deleting strips (1,2), (1,3) and (1,2,3). Indeed, these combinations are part of the same direction, even though they can be two-coloured. Even though deleting strips (2) and (3) does not yield two-coloured designs, deleting strips (2,3) is investigated, as this combination is independent of combination (1). Deleting strips (2,3) yields a two-coloured design, so the search does not investigate deleting strips (1,2,3), although the search already discarded this combination. The algorithm, therefore, reduces computation to the combinations (1), (2), (3) and (2,3) only and returns the two-coloured design from the combinations (1) and (2,3).

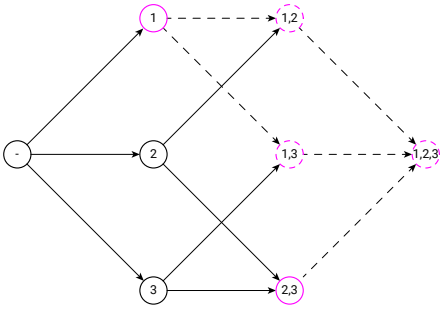



Figure 7.24 – Principle of the search algorithm for two-coloured quad meshes. The search yields only the closest two-coloured quad meshes in independent design directions. The two-coloured quad meshes are highlighted in pink, and the discarded quad meshes downstream are marked by dashes.

The implementation of the search algorithm is further detailed.

### 7.3.3 Search algorithm

For an input quad mesh with $n$ strips, the search algorithm iteratively tests and discard all strip combinations

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n. \tag{7.7}$$

Starting with $k = 0$, the $\binom{n}{k}$ combinations of $k$ strips among the total $n$ strips are enumerated and tested, $k$ is increased by one and the operation is repeated.

For each combination, the deletion applies to a copy of the input quad mesh and its strip graph with deletion of the mesh strips and the graph vertices corresponding to the combination. The search tests the design against two criteria.

#### 7.3.3.1 Criteria

For each combination, a decision is made based on the validity of the design against two criteria:

- on shape topology: the new mesh must have the same shape topology; and,
- on pattern topology: the new graph must be two-coloured.

**Shape topology** The new mesh must have the same shape topology (see Section 2.4.1). The mesh must be manifold, have the same Euler characteristic and have the same number of boundaries. This criterion is more decisive for high numbers of deleted strips. Indeed, deleting too many strips can result in a different shape topology. In Figure 7.25, deleting two strips results in a different shape topology, splitting the boundary in two. This combination is therefore not valid, although yielding a two-coloured design.

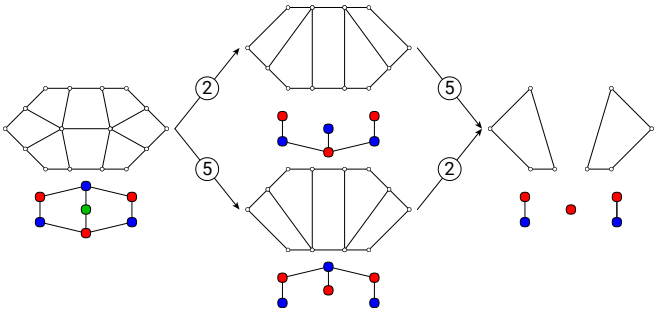



Figure 7.25 – Deleting some strips can yield a different shape topology, although yielding a two-coloured quad mesh.

This example highlights an isolated graph vertex resulting from a collateral strip deletion, mentioned in Section 6.3.1. The rule deletions do not

apply to this strip, but all of its faces are part of the deleted strips. This combination of two rules creates a distance of three. The combination yields a two-coloured quad meshes too far and too early. The search must check the other combinations deleting fewer strips before. Therefore, the search discards such combinations.

**Pattern topology** The new graph must be two-coloured. This criterion is more decisive for low numbers of deleted strips. Indeed, deleting strips deletes vertices in the graph and subgraphs of two-coloured graphs can be two-coloured.

The validity of the two criteria provides a reduction strategy on the search over the $2^n$ combinations.

#### 7.3.3.2 Search reduction

The search does not seek all two-coloured designs, only the closest ones for each design direction. The number of tests on downstream designs can, therefore, be reduced based on the results against the two criteria of the upstream designs. The downstream designs from a design are the ones at a farther distance in the same direction. A combination of strips $X$ is downstream another combination of strips $Y$ if $X$ is a subset of $Y$:

$$X \subset Y. \tag{7.8}$$

If a combination of strips yields a two-coloured quad mesh, the downstream combinations are discarded during the search, as they are farther in the same direction. If a combination of strips yields a different shape topology, the search discards the downstream combinations, as applying more strip deletions will not restore the shape topology.

Thereby, this scheme reduces the iterative enumeration and avoids testing all combinations.

#### 7.3.3.3 Termination criteria

Termination does not occur after $2^n$ test but earlier. Indeed, iterative enumeration terminates before $k$ reaches $n$, when the reduced search discards all of the downstream combinations. The designer can also specify custom termination criteria, like a maximum distance as $k_{max}$ or a minimum number of yielded designs. More generally, the designer can terminate the search when pleased with the current pool of design and does not wish to search further.

```
get a quad mesh with n strips
generate the strip graph of the mesh
start an empty list for the two-coloured topologies with the same
  shape topology
start an empty list for the discarding sub-combinations
initiate k = −1
while k < n do
    k+ = 1
    for each combination of k strips among the n strips do
        if the combination causes collateral strip deletions then
            discard combination
        end
        if one of the discarding sub-combinations is a subset of the
          combination then
            discard combination
        end
        else
            copy the original mesh
            delete the k strips of the copy mesh
            if the shape topology is different then
                add the combination to the list of discarding sub-
                  combinations
            end
            else
                copy the original graph
                delete the corresponding k vertices of the copy graph
                if the graph can be two-coloured then
                    add the copy mesh to the list of two-coloured
                      topologies with the same shape topology
                    add the combination to the list of discarding sub-
                      combinations
                end
            end
        end
    end
end
return the successful topologies
```

**Algorithm 3:** Projection of a quad mesh to the closest two-colourable quad meshes in different directions.

#### 7.3.3.4 Pseudo-code

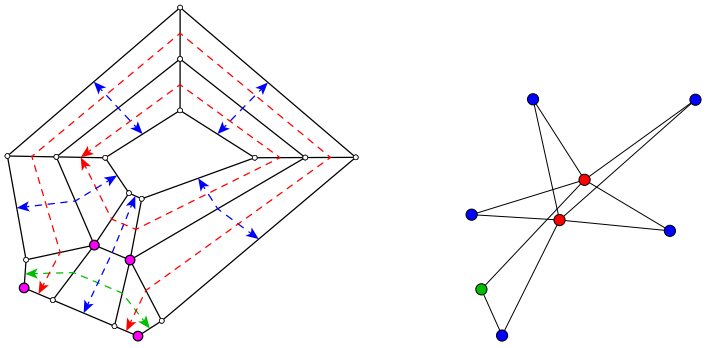
Algorithm 3 provides the pseudo-code of the two-colouring search.

### 7.3.4 Examples

Examples illustrate an application and validate the two-colour projection algorithm.

#### 7.3.4.1 Application

The coarse quad mesh in Figure 7.26 does not comply with the two-colouring requirements. The five-valent singularities do not allow to fulfil the singularity requirement. The three-coloured strip graph and the index-based characterisation evidence this statement.



(a) Coarse quad mesh (b) Strip graph

Figure 7.26 – A coarse quad-mesh that does not comply with the two-colouring requirements. The five-valent singularities does not allow to fulfil the singularity requirement, as evidenced by the three-coloured strip graph and the index-based characterisation.

Applying the two-colour projection yields the four coarse quad meshes with two-coloured strips in Figure 7.27. They all result from one strip deletion and are therefore at a distance of one from the initial coarse quad mesh. The topology fulfils the two-colouring requirement thanks to merging the initial five-valent singularities or moving them to the boundary. Integrating the density requirement allows for generating the checkerboard tiling with different singularities in Figure 7.27.

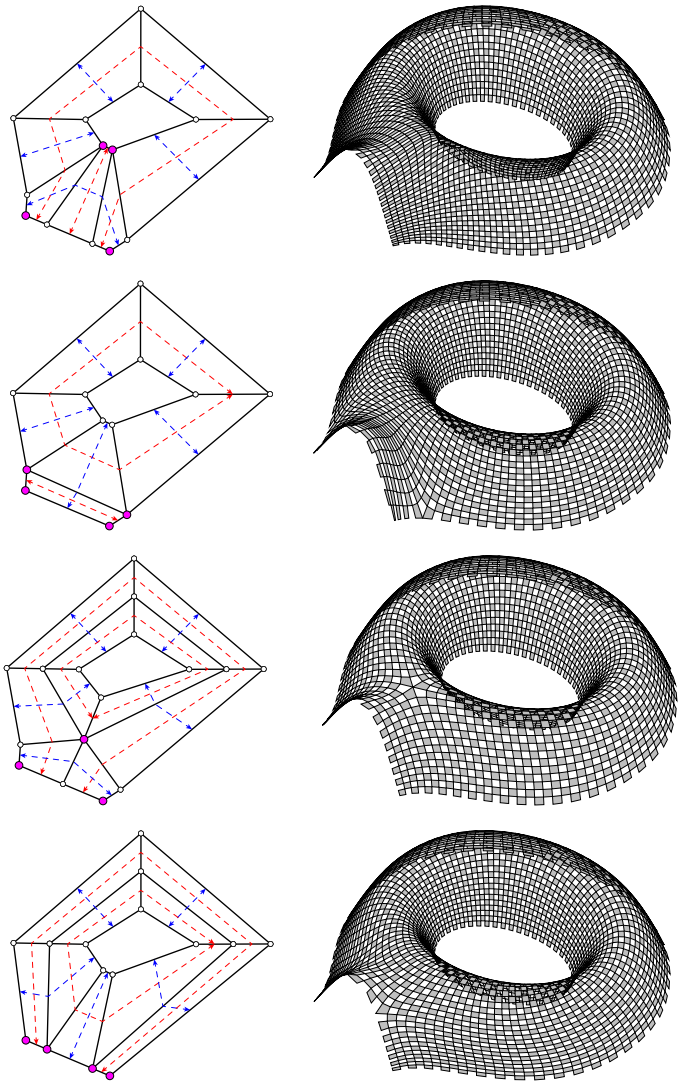


Figure 7.27 – Two-coloured coarse quad meshes resulting from two-coloured topology finding and their corresponding checkerboard tilings with different singularities.

#### 7.3.4.2 Validation

Several examples validate the two-colourability topology-finding algorithm. The validation tests provide some numerical results on the influence of the reduction scheme and the size of the two-coloured design subspace.

#### 7.3.4.3 Examples

In Figures 7.28 to 7.31, the found two-coloured designs are grouped by distance from the initial non-two-coloured design. No user-based termination criterion is set. Deleting strips coarsens the mesh and can cause faces to overlap. Nevertheless, the topology is correct and the overlaps fixed during geometrical processing.

The initial design for the hexagonal shape in Figure 7.28 can not be two-coloured because of the two five-valent singularities. Two-colour projection yields seven two-coloured topologies, three are at a distance of one and four at a distance of two. Two-colouring becomes possible thanks to either merging the two five-valent singularities into a six-valent one or moving them to the boundary.

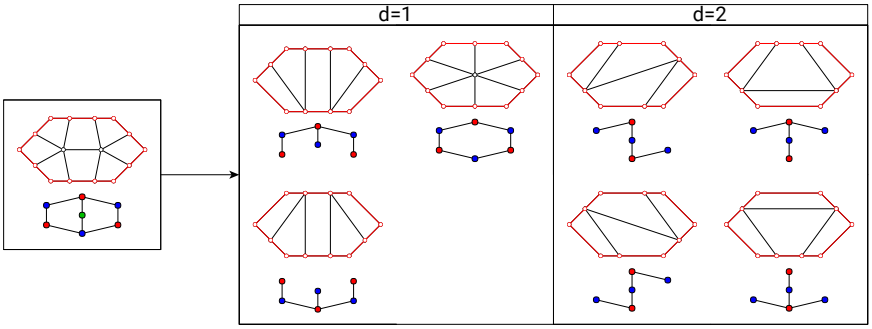
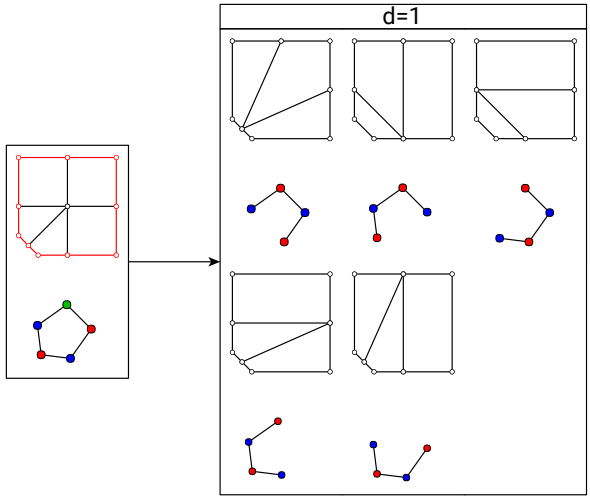
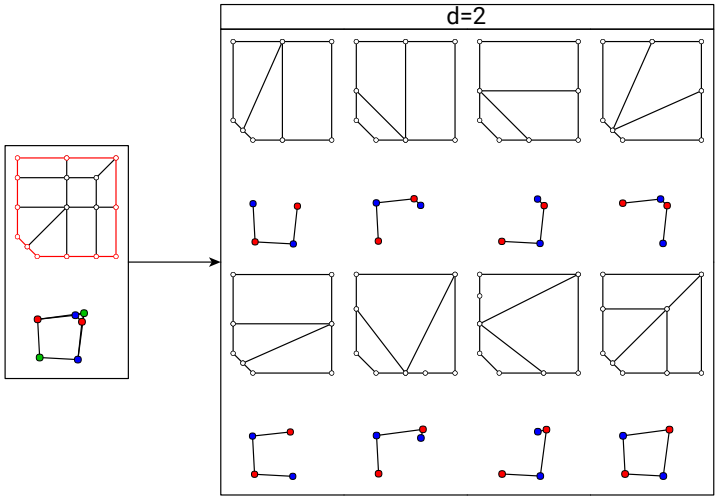



Figure 7.28 – Two-colour projection applied to a seven-strip quad mesh with a hexagonal shape.

The initial designs for the pentagonal shape in Figure 7.29 can not be two-coloured. They differ by one strip only, which was added from the design in Figure 7.29a to the one in Figure 7.29b. The projection of the five-strip topology yields five two-colourable topologies, all at a distance of one, while the projection of the six-strip topology yields eight two-colourable topologies, all at a distance of two. Adding a strip resulted in having more but farther two-coloured topologies.




(a) Five-strip quad mesh




(b) Six-strip quad mesh

Figure 7.29 – Two-colour projection applied to quad meshes with a pentagonal shape.

In Figure 7.30 and 7.31, the initial designs result from skeleton-based decomposition of the shape, presented in Chapter 4.

The initial topology for the rectangular shape with one opening in Figure 7.30 yields nine two-colourable topologies, only one a distance of one, and eight at a distance of two.

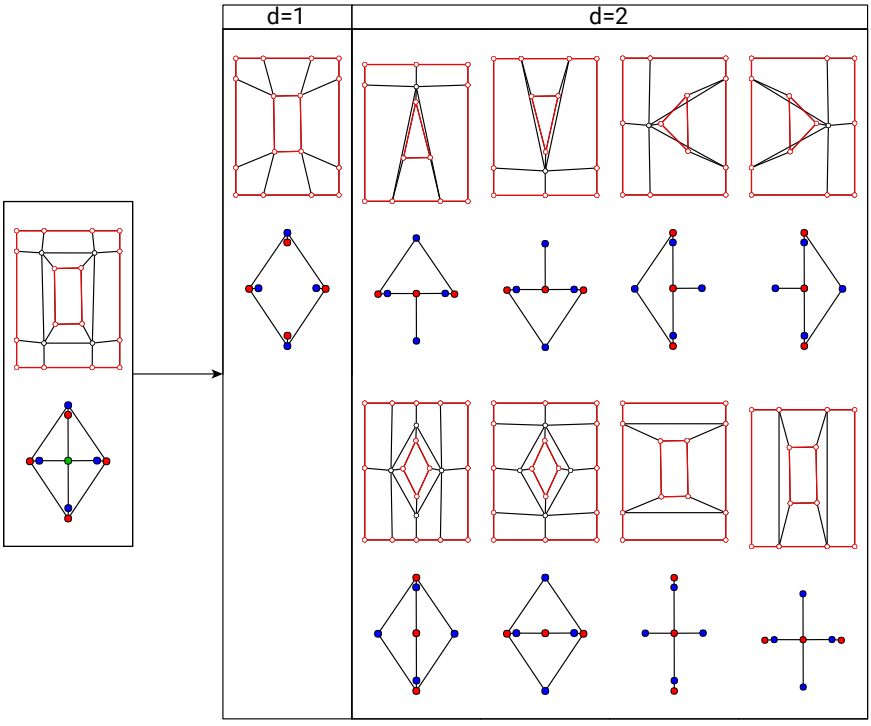



Figure 7.30 – Two-colour projection applied to a nine-strip quad mesh with a rectangular shape with one opening.

The initial topology for the rectangular shape with two openings in Figure 7.31 yields 31 two-colourable topologies.

#### 7.3.4.4 Results

The detailed results for each example are in Tables 7.8 to 7.12 in the Appendix of this chapter. Each table contains the number of potential combinations, the number of tested combinations due to search reduction and

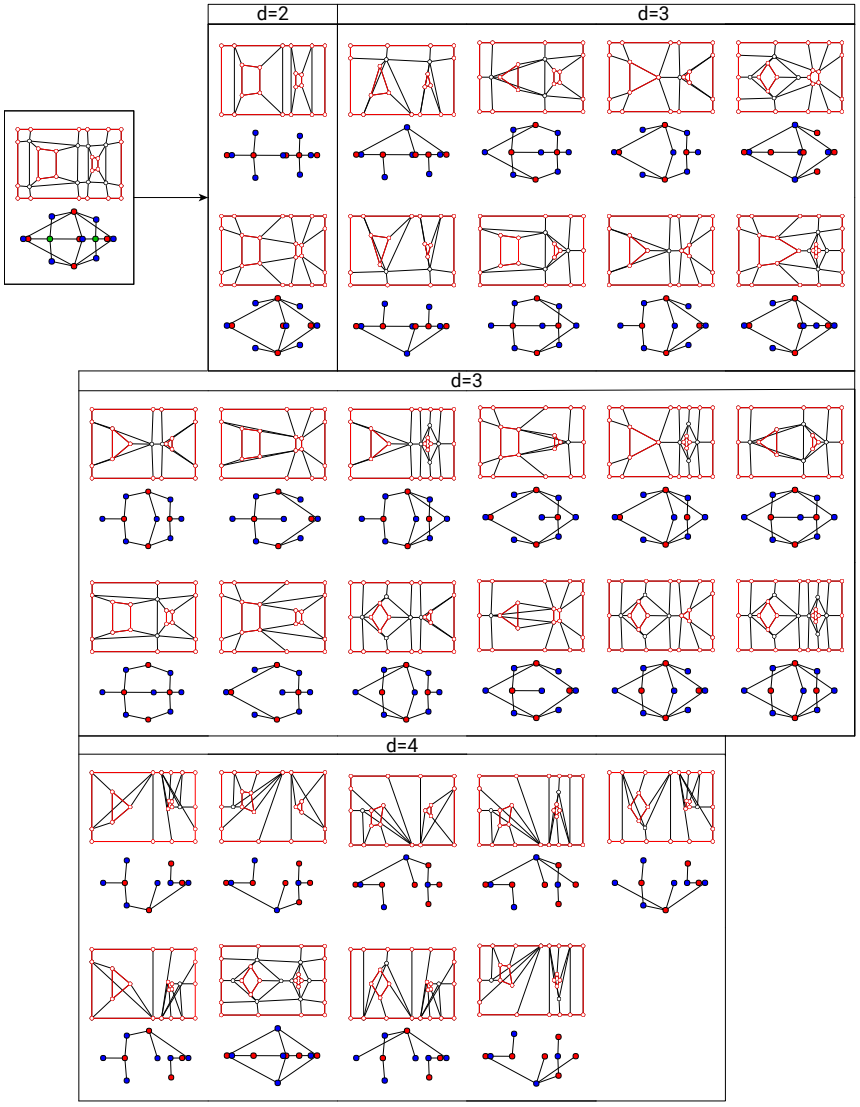



Figure 7.31 – Two-colour projection applied to a 14-strip quad mesh with a rectangular shape with two openings.

the results against the two validity criteria: the two-coloured topologies with the right shape topology (O), the non-two-coloured topologies with the right shape topology (-) and the the topologies with a wrong shape topology (X). The total number of yielded topologies is highlighted in green and the termination criterion in red.

Table 7.3 shows the summary results per example: the number of strips $n$, the total number of combinations for strip deletions $2^n$, the percentage of tested combinations and the percentage of yielded topologies.

Table 7.3 – Condensed results of two-colourable topology finding.

| Figure | 7.28 | 7.29a | 7.29b | 7.30 | 7.31 |
|---|---|---|---|---|---|
| n | 7 | 5 | 6 | 9 | 14 |
| $2^n$ | 128 | 32 | 64 | 512 | 16,384 |
| % tested | 10.2 | 15.6 | 37.5 | 11.1 | 6.1 |
| % yielded | 5.5 | 15.6 | 12.5 | 1.8 | 0.2 |

These results highlight:

- the significant decrease of tested combinations thanks to the search reduction, especially for the topology with many strips, with 6.1% for the 14-strip topology;
- the small part of combinations to yielded the closest two-coloured topologies in the different design directions, especially for the topology with many strips, with 0.2% for the 14-strip topology.

Termination occurs when there is no combination to be tested or when there is no non-two-coloured topology with the right shape topology at a value $k$. These results intuit another practical termination criterion: stopping the procedure when increasing $k$ stops yielding two-coloured topologies with the right shape topology. This criterion reduces by 34% and 27% the number of tests for the 9-strip and the 14-strip topology, respectively.

Comparing the results for the two topologies that differ by one strip in Figure 7.29 illustrates the challenge in including strip additions in two-colour projection. Although, adding strips can not make a topology two-coloured, projection yields then more results but at a farther distance. The fundamental limitation is the infinite combinatorial richness in adding strips. Nevertheless, an enumerative or stochastic search could integrate the addition rules.

This search algorithm for two-coloured quad meshes can apply to two-coloured topology finding of patterns for structural design.

## 7.4 Two-coloured topology finding

This section applies two-coloured topology finding to folded shells, a structural system that necessitates a two-coloured partition of the pattern to define the folding directions.

Folded or corrugated shells are continuous shells that can efficiently span large areas. For instance, the CNIT of Nicolas Esquillan at La Défense, Paris spans the world record for concrete structures of 218 meters [Motro and Maurin, 2011]. The folded plate in Figure 7.32 illustrates the differentiated behaviour induced by the folds.

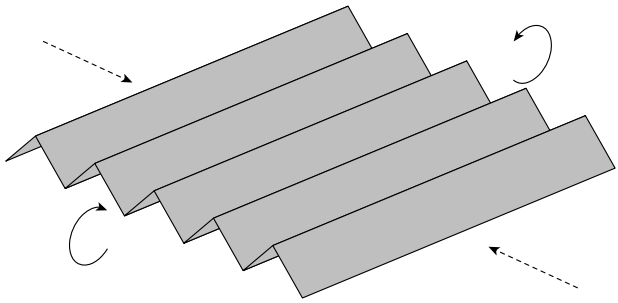


Figure 7.32 – Folded patterns increase the bending stiffness along the folds and decrease the membrane stiffness across the folds.

Along the folds, the structural height is increased, resulting in higher bending stiffness. Across the folds, the force eccentricity is increased, resulting in a lower membrane stiffness, as in an accordion.

Folding along two transverse directions yields an egg box pattern that weakens both directions. Therefore, the direction of the folding must be chosen carefully based on the statics system in order to strengthen the structure and not weaken it. Folding a structure relying mainly on membrane equilibrium like a fully supported dome weakens it. Folding a structure relying on bending can strengthen it, like a plate to reduce deflection or a shell to reduce buckling.

Previous research investigates the geometry of the folding pattern for structural performance and fabrication ease, mainly studying cylinder-like

surfaces [Norman et al., 2009, Malek and Williams, 2017, Stitic and Weinand, 2015, Mesnil et al., 2017a, Mesnil et al., 2018b]. The CNIT has a triangular footprint with supports at the corners only, highlighted in green in Figure 7.33.

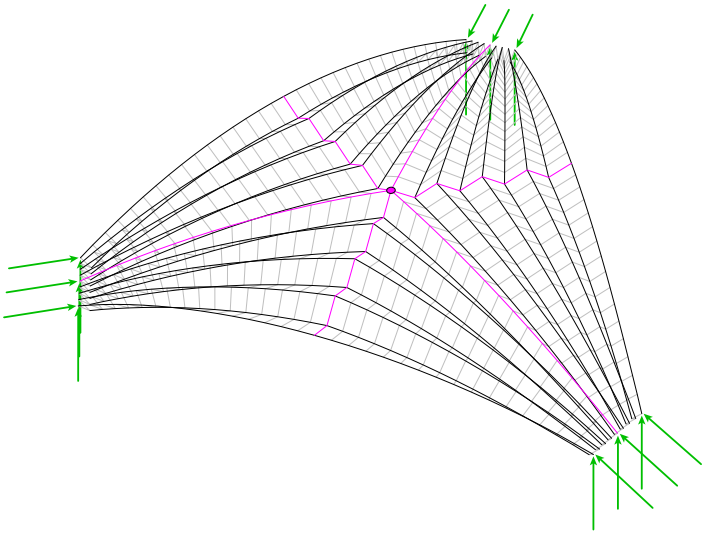


Figure 7.33 – Principle of the folding pattern of the CNIT. A six-valent singularity in pink provides a two-coloured pattern for stiffening the directions that carry the loads along the free edges directly to the supports in green. The stiff primary direction is in black and the secondary direction in grey.

The folding pattern is designed using a central singularity highlighted in pink to strengthen the free edges to carry the loads to the supports. The quad mesh serves as parameterisation of the continuous shell. The stiffened direction is in black, and the weakened direction is in grey.

The singularity in the quad mesh that represents and morphs the folded shell is six-valent. Its index follows the singularity requirement for two-colouring. Indeed, folds stiffen only one of the two directions. This differentiation in the two surface directions translates to a two-colouring property of the polyedges of the quad mesh.

Two-coloured topology finding can apply to the quad meshes used for the parameterisation of shells to explore feasible topologies of folding patterns. Starting from a pattern that can not be two-coloured, two-colouring topology

finding is applied. The two-coloured patterns obtained are the closest ones in different directions of the search for two-colour projection. Data analysis of the strips and equivalent rules applied is used to combine these design directions to find patterns at a farther distance but potentially more efficient.

### 7.4.1 Design problem

The 95 m by 74 m roof of the Great Courtyard of the British Museum, presented in details in Section 4.4, is vertically supported along its boundary but allows thrust only at the four corners [Sischka et al., 2001]. The original geometry does not stem from the statics system. The geometry is analytically defined to smoothly comply with the boundaries [Williams, 2001]. This statics system and the geometry favour bending behaviour.

Therefore, this section investigates a fold pattern to stiffen a continuous concrete shell. The quad mesh provides a parameterisation of the surface of the shell to morph the folds.

The constant shell thickness depends on its area. Indeed, the area changes with the amplitude and the pattern of the folds. A thickness of 20 cm for the average surface of 6000 $m^2$ defines the constant volume.

A C30/37 concrete is chosen with a characteristic strength $f_{ck}$ of 30 MPa and a Young modulus $E_{cm}$ of 33 GPa. The actual support conditions apply with full vertical support along the boundary and horizontal support at the four corners only. A unique load combination includes the self-weight $G$ and a uniform downward projected snow load Q=0.5kN/$m^2$. This analysis does not take into account the geometrical factors for snow distribution resulting from the curved and folded geometry.

A second-order mechanical analysis is performed, using the Finite Element Analysis plugin Karamba for Grasshopper3D [Preisinger, 2013]. The analysis mesh results from densification and triangulation of the quad-mesh pattern.

Four metrics evaluate the structural performance: the maximum displacement, the strain energy, the maximum stress utilisation and the first buckling load factor. The lower these metrics, the more structurally efficient the design, except for the first buckling load factor, which should be higher.

### 7.4.2 Pattern design

Section 4.4 performs feature-based topology finding, revisiting the pattern of the gridshell of the British Museum. Skeleton-based decomposition of the average surface, introduced in Section 4.2, does not yield a two-coloured

pattern. Indeed, the topology of the pattern, shown in Figure 7.34 with the strip labels, has four five-valent non-boundary singularities and does not fulfil the index-based characterisation.

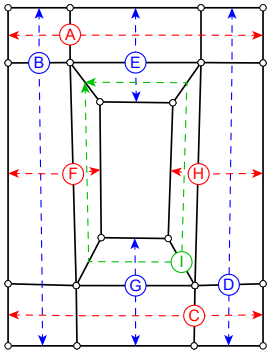



Figure 7.34 – Topology from skeleton-based decomposition of the surface with labelled strip. The resulting pattern can not be two-coloured due to the four five-valent non-boundary singularities.

Adding features yields some designs that can be two-coloured and are more efficient thanks to poles that concentrate elements at the corners. However, this concentration makes the detailing, fabrication and assembly of the folds complex. Therefore, the quad mesh from the skeleton-based decomposition of the surface without point features serves as starting topology.

Two-colouring projection applies to this starting topology. Nine two-coloured topologies are yielded, as shown in Figure 7.31 as part of the validation examples. Four of these topologies do not respect the two-fold symmetry of the project. Therefore, they are discarded, resulting in the pool of five two-coloured topologies in Figure 7.35.

These topologies are the closest two-coloured topologies in the different directions from the starting topology. Nevertheless, further deleting strips yields other two-coloured topologies, but at a farther distance in combined directions.

The topology needs to fulfil the singularity requirement to select a folding direction. However, the topology also needs to fulfil the density requirement to alternate up and down polyedges to form folds transversally to the closed polyedges. The density requirement imposes an even sum of the density of the strips transverse to the closed strips. The coarse quad meshes are densified based on a 1.5 m target length incorporating this density requirement.

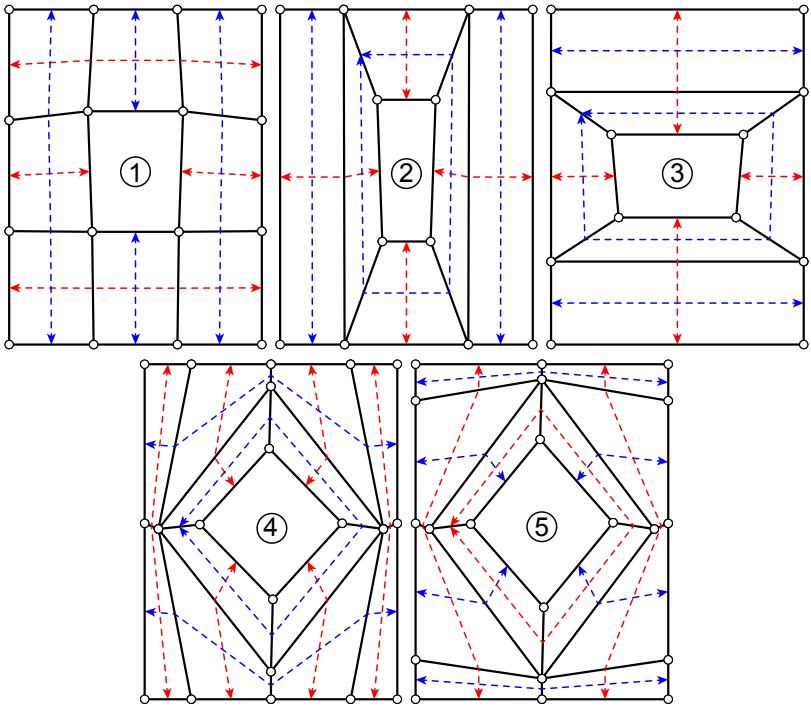



Figure 7.35 – Five symmetrical two-coloured topologies.

The folds of the CNIT are straight in the plan, resulting in kinks when folds meet along the polyedges stemming from the singularity represented in pink in Figure 7.33. The resulting forces in the transverse direction to the folds are taken over by beams. These beams serve as diaphragms to locally stiffen the transverse direction. They meet at the singularity and can redistribute forces along the folds and towards the supports. Here, the inner boundary hinders the use of straight folds, inducing curved folds. Therefore, the dense quad meshes are relaxed on the surface, resulting in curved folds without kinks along the edges from the coarse quad mesh.

The five resulting parameterisation quad meshes are shown in Figure 7.36 with the two potential directions to integrate folds differentiated by red and blue polyedges.

For each topology, the folds can follow either of the two coloured directions. For a given direction, the folds result from vertex offset of the

corresponding polyedges. The vertices on the same polyedge move towards the same side of the surface. The orthogonal offsets alternate up and down from an adjacent polyedge to another to provide a folded pattern. Three uniform amplitudes provide three designs for each topology and each direction: 0.5 m, 1 m and 1.5 m.

The shells in Figure 7.37 represent the ten folded patterns, five topologies times two directions. The ten designs are labelled X-Y, X representing the pattern topology and Y the folded direction.

To summarise, 35 designs for the continuous shell are analysed: one smooth pattern represented by the five topologies in Figure 7.36 and ten folded patterns in Figure 7.37 for three fold amplitudes.

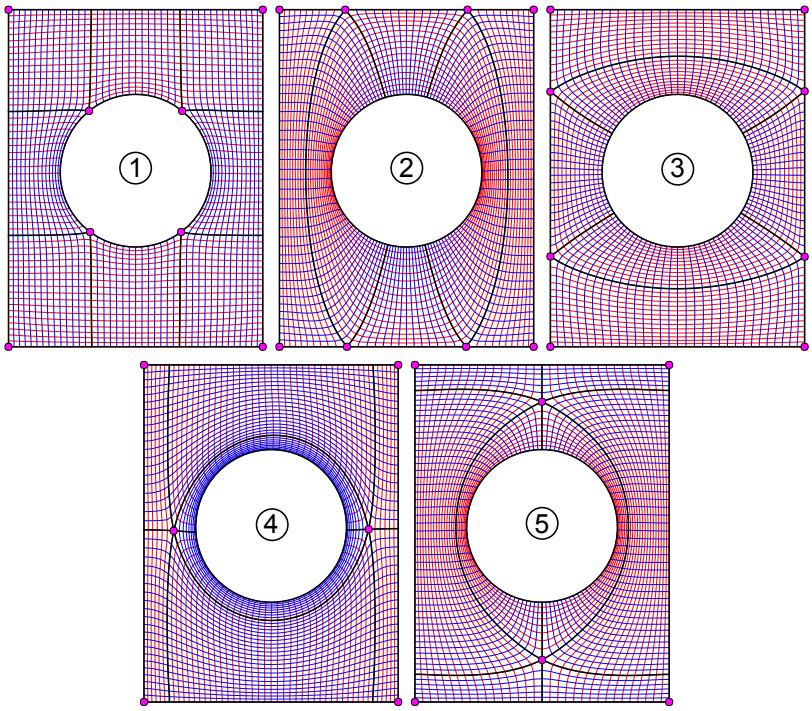



Figure 7.36 – The five smooth parameterisation quad meshes patterns with two-coloured polyedges.

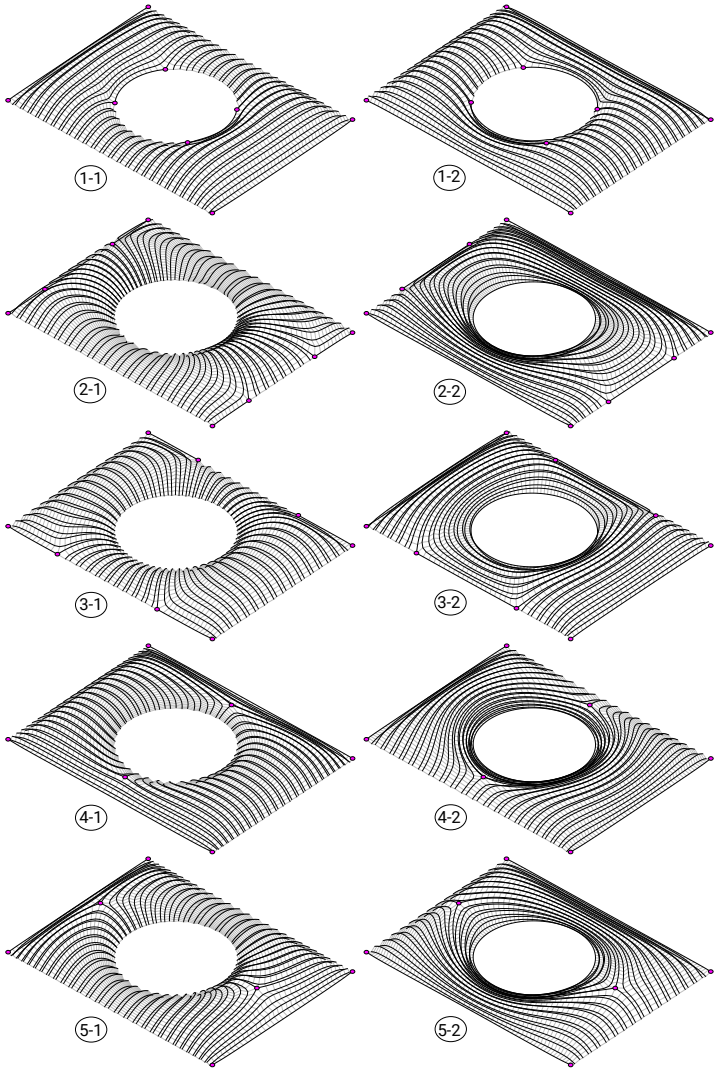



Figure 7.37 – The ten folded shells resulting from the five pattern topologies times the two folded directions, marked as X and Y in the X-Y labels, respectively.

### 7.4.3 Numerical results

The numerical results from structural analysis of the 35 designs are detailed in Table 7.13 in Appendix 7.6.2. For each topology from 1 to 5, direction 0 provides the smooth design, and directions 1 and 2 provide the two folded designs for amplitudes 0.5, 1.0 and 1.5 m.

The provided area and the thickness all result in the same volume.

The parameterisation patterns are refined to obtain the mesh for finite element analysis. First, subdivision refines the quad faces, similarly to a Catmull-Clark subdivision without smoothing. Second, triangulation splits the quad faces. A study of the convergence of the strain energy evaluates the necessary level of refinement, shown in Figure 7.38. The smooth design of topology 1 serves as a test case. Level one of subdivision provides sufficient precision for this analysis without resorting to a too large number of faces. All patterns are discretised following this rule. The resulting number of faces is featured in Table 7.13 in Appendix 7.6.2.

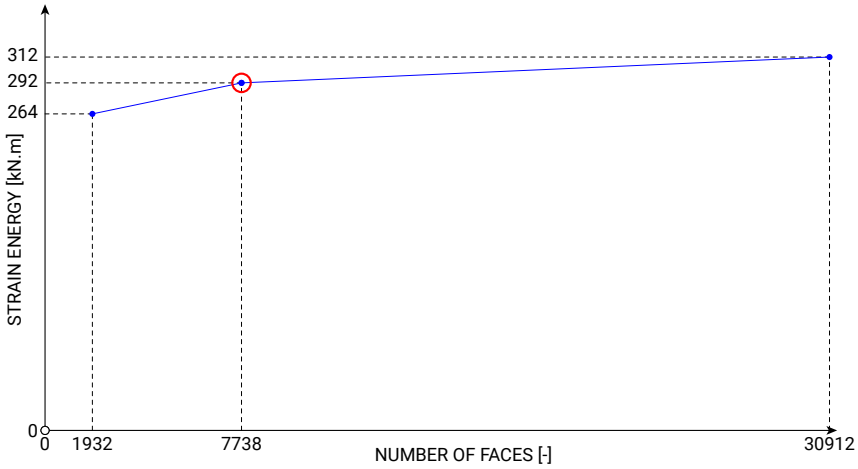



Figure 7.38 – Study of the convergence of the strain energy for the smooth design of topology 1. Level one of subdivision, highlighted in red, provides sufficient precision for this analysis without resorting to a too large number of faces.

Four metrics evaluate the structural performance: the maximum displacement, the strain energy, the maximum stress utilisation and the first buckling load factor. The results are detailed in Table 7.13 in Appendix

7.6.2.

The designs are compared based on their strain energy to assess global efficiency – the lower the strain energy, the better the design. Figure 7.39 compares the strain energy with the amplitude of the folds.

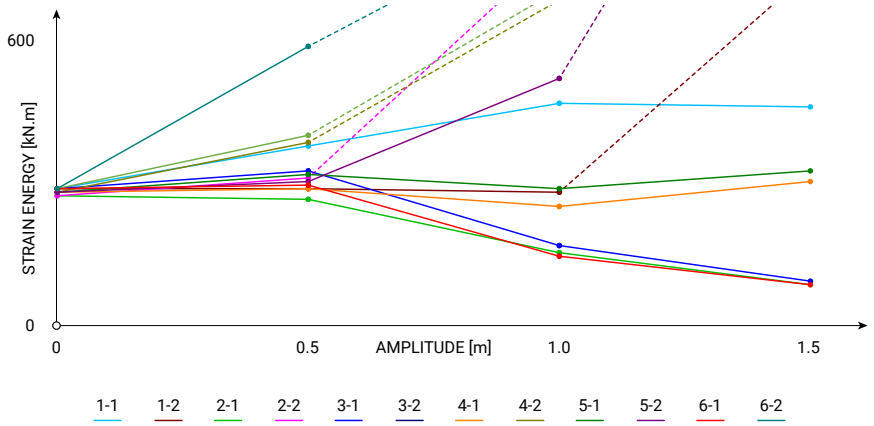



Figure 7.39 – Influence on the strain energy of introducing folds with different directions and amplitudes. Results above twice the initial strain energy are not shown, as evidenced by the dashed curves.

The smooth shells have similar strain energy with an average of 287 kN.m. Their standard deviation equals 5 kN.m, less than 2% of the average. Such precision is sufficient for comparison of the different fold patterns. Folding transverse directions tends to have an opposite qualitative effect on the structural behaviour, increasing or decreasing the strain energy. For instance, folds with an amplitude of 1.5 m in topology 2 decrease the strain energy to 93 kN.m in direction 1 and increase it to 1440 kN.m in direction 2. However, the increase or decrease is not monotonic with the amplitude. For instance, folds in topology 3 and direction 1 first increases the strain energy to 328 kN.m for an amplitude of 1.0 m before decreasing to 101 kN.m for an amplitude of 1.5 m. Most designs increase the strain energy for the high fold amplitude. Nevertheless, two folded designs significantly decrease the strain energy to about one-third of the smooth designs: topology 2 with direction 1 for an amplitude of 1.5 m resulting in a strain energy of 93 kN.m and topology 3 with direction 1 for an amplitude of 1.5 m resulting in a strain energy of 101 kN.m. These two designs also perform the best for the other metrics. They have the lowest deflection, 2.3 cm each; the lowest

stress utilisation, 0.7 and 0.6, respectively, the only ones below 1; and the highest first buckling load factor of 68 and 73, respectively.

The two most efficient designs, 2-1 and 3-1, seems to be the ones with the most direct folds from the outer boundary to the inner boundary. A data analysis compares this intuition to the evaluation of the positive or negative contributions in the structural performance of each folded strip across all designs. Specific matrices representing the fold pattern and the performance of each design provide this data.

Figure 7.34 shows the A to I labels of the nine strips. The matrix in Table 7.4 represents for each topology which strips are present or absent by a 1 and a 0, respectively.

Table 7.4 – Presence matrix of the nine strips for the five topologies: 1 if the strip is present, 0 otherwise.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| A | 1 | 0 | 1 | 1 | 1 |
| B | 1 | 1 | 0 | 1 | 1 |
| C | 1 | 0 | 1 | 1 | 1 |
| D | 1 | 1 | 0 | 1 | 1 |
| E | 1 | 1 | 1 | 1 | 0 |
| F | 1 | 1 | 1 | 0 | 1 |
| G | 1 | 1 | 1 | 1 | 0 |
| H | 1 | 1 | 1 | 0 | 1 |
| I | 0 | 1 | 1 | 1 | 1 |

The matrix in Table 7.5 represents for each direction of each topology which strips are folded by a 1 and which strips are not folded or absent by a 0.

The matrix in Table 7.6 provides the strain energy as a performance measure of each of the ten folded designs, averaged across the different values of amplitude.

For each strip, the covariance between the folding matrix and the performance matrix is computed in Table 7.7.

A negative covariance means that folding the strip decreases the strain energy and therefore improves structural behaviour on average, and vice versa. As displayed in Figure 7.40, strips E, F, G, H in green are the only ones that improve the performance on average. Only designs 2-1 and 3-1 fold all these strips. These designs are the most efficient ones regarding the strain-energy metric. These four strips are the only ones that connect the

outer and inner boundaries. The previous intuition is confirmed.

Table 7.5 – Folding matrix of the nine strips for the ten folded designs: 1 if the strip is present and folded, 0 otherwise.

| | 1-1 | 1-1 | 2-1 | 2-2 | 3-1 | 3-2 | 4-1 | 4-2 | 5-1 | 5-2 |
|---|---|---|---|---|---|---|---|---|---|---|
| A | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| B | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| C | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 |
| D | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 |
| E | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| F | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| G | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| H | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| I | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |

Table 7.6 – Performance matrix of the strain energy in kN.m of the ten folded designs averaged across the different values of amplitude.

| | 1-1 | 1-2 | 2-1 | 2-2 | 3-1 | 3-2 | 4-1 | 4-2 | 5-1 | 5-2 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0.0 | 292 | 292 | 279 | 279 | 291 | 291 | 288 | 288 | 287 | 287 |
| 0.5 | 380 | 289 | 272 | 316 | 328 | 406 | 292 | 392 | 321 | 305 |
| 1.0 | 474 | 287 | 156 | 803 | 169 | 715 | 252 | 694 | 294 | 521 |
| 1.5 | 464 | 752 | 93 | 1440 | 101 | 1294 | 304 | 1036 | 329 | 1394 |
| avrg. | 403 | 405 | 200 | 709 | 222 | 676 | 284 | 602 | 308 | 627 |

Table 7.7 – Covariance matrix in $kN^2.m^2$ of the energy performance of the folded strips, averaged across all designs.

| A | B | C | D | E | F | G | H | I |
|---|---|---|---|---|---|---|---|---|
| 21 | 25 | 21 | 25 | -66 | -64 | -66 | -64 | 84 |

The obtained two-coloured topologies are the closest ones in different directions from the initial topology. The design directions of topology 2 and 3 are the most promising ones for structural performance. These directions are combined to obtain a farther topology. Topology 2 and 3 result from the deletion of strips A and C, and B and D, respectively. Topology 6 in Figure 7.41 is obtained from the deletion of strips A, B, C and D. With

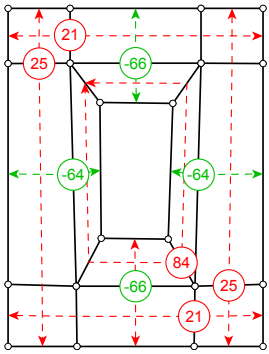



Figure 7.40 – The covariance in $kN^2.m^2$ of each strip with the structural performance when stiffened in the folded designs highlights their positive or negative influence.

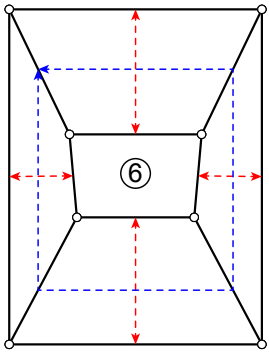



Figure 7.41 – Sixth symmetrical two-coloured topology resulting from the combination of topologies 2 and 3.

fewer strips, the topology can be two-coloured as well. This topology is at a distance of four from the input topology, as opposed to a distance of two for topologies 2 and 3.

The same generation and morphing processes are applied to obtain the two folded designs in Figure 7.42 represented for the amplitude of 1.0 m.

The complementary numerical results are shown in Table 7.14 in Appendix 7.6.2. Design 6-1 has folded strips E, F, G, H and form vertically curved but horizontally straight folds spanning from one boundary to another along the shortest spans. This design performs the best with a strain

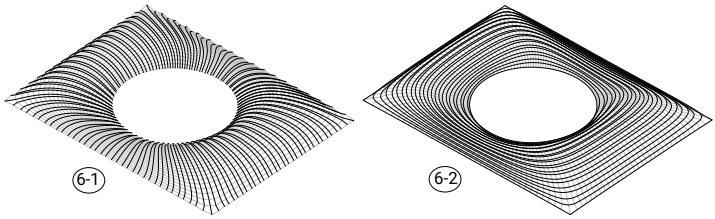



Figure 7.42 – Two additional folded shells resulting from the combination of topologies 2 and 3.

energy of 91 kN.m, a displacement of 2.2 cm, a stress utilisation of 0.6 and a first buckling load factor of 39, for the amplitude of 1.5 m. Except for the buckling metric, the other metrics are equal or slightly better than the ones of designs 2-1 and 3-1, for the fold amplitude of 1.5m.

The fold amplitudes are uniform. However, the spans and the magnitude of forces are not the same. Differentiating fold amplitudes based on the local bending moments could extend this study and further improve the designs, potentially providing different results.

Folding is not necessary for sufficiently stiff areas where the span is short or the forces low. Local folding could be sufficient, hinting at local two-colouring of the pattern.

Construction-related objectives are missing in this example. Nevertheless, the design process can integrate a fabrication requirement like panel planarity. Mesnil *et al.* use parallel transformations to preserve planarity in form finding of a folded shell [Mesnil et al., 2017a].

A simple performance analysis and data comparison allow evaluating a small set of topological designs. The results inform the generation of other topological designs that are farther in the design subspace of two-coloured patterns. This strategy provides an example to tackle the question of exploration of combinatorial spaces using data analysis.

## 7.5 Summary of contributions

This chapter introduced two-coloured topology finding:

- a topological, organisational property in a structural system based on quad meshes was identified as a two-colouring requirement;

- the two-colouring requirement was formalised and characterised using the strip graph;
- a search algorithm was developed to project topologies to the two-coloured design subspace using the strip grammar;
- this algorithm was applied to the design of folded shells with an example of data-based topological combinations to farther explore the two-coloured design subspace.

The algorithm only considers strip deletion rules. This approach simplifies the search as the number of these rules is finite for a given quad mesh. Nevertheless, considering strip addition rules can enrich the design directions, although yielding designs at a farther distance, demonstrated in the validation example in Figure 7.29.

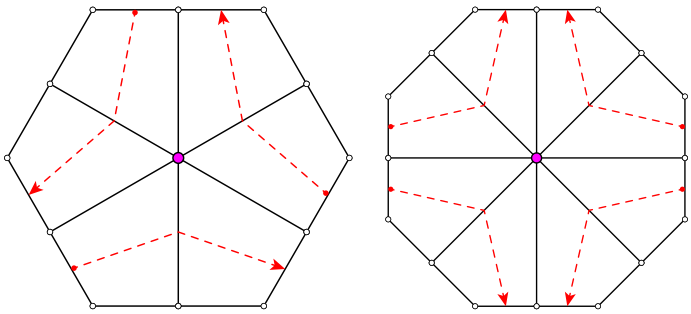
The enumerative approach for the search can be limited for quad meshes with a large number of strips. Particularly if not one but all different design directions are explored. A stochastic search of the two-coloured design is another strategy to tackle this problem, with the opportunity to embed other design objectives. Instead of considering two-colouring as a true-or-false binary property, a measure of the distance to the two-colouring requirement is necessary. The minimum number of third-colour vertices in the strip graph or the metric resulting from the index-based characterisation in Equation 7.4 can serve as such measures.

Two-coloured patterns apply to many structural as well as architectural and decorative applications, with differentiated directions. Moreover, an extension and adaptation of this constrained-exploration strategy can apply to patterns with other similar requirements.

A stronger requirement for quad-mesh patterns is to preserve the orientation between strips of the same colour. In Figure 7.43a, the six-valent singularity is adjacent to three strips per colour. This odd number does not allow matching directions, marked by red arrows. In Figure 7.43b, the eight-valent singularity is adjacent to four strips per colour. This even number allows matching directions.

Such a requirement constrains the pattern to have only 4k-valent vertices, or poles, which have an integer index, instead of 2k-valent vertices, which have an index proportional to $1/2$. Hu et al. [Hu et al., 2012] address some aspects of this requirement on patterns with specific symmetries, like the 17 plane-symmetry patterns of the wallpaper group [Fedorov, 1891, Polya, 1924, Grünbaum and Shephard, 1987]. Among other contributions, they

provide an operation preserving this property that reduces high-valent vertices. For instance, sixteen-valent singularities split into three eight-valent singularities. The strip addition rule can perform the same operation.



(a) Six-valent singularity without matching strip orientation

(b) Eight-valent singularity with matching strip orientation

Figure 7.43 – 2k-valent singularities are adjacent to odd number of strips per colour, which does not provide the symmetry for matching directions, represented by the red arrows, between the strips of the same colour, opposed to 4k-valent singularities, which are adjacent to even number of strips per colour.

Triangular-mesh patterns must fulfil a three-colouring requirement for organisations like the three-layer structure of beams in Figure 7.44. Triangular meshes also have a strip structure, with three strips crossing over each face, instead of two in the case of quad meshes. Similarly, triangular meshes can stem from coarse triangular meshes. Such coarse meshes encode the data related to the singularities. Then, the dense mesh result from triangle densification like the Loop subdivision [Loop, 1987]. Equivalently, an index also characterises the singularities in triangular meshes [Fogg et al., 2018]. In triangular meshes, the index is a measure from the deviation from the regular valency of six. The three-colouring requirement constrains the triangulated-mesh pattern to 3k-valent vertices, like the nine-valent singularity in the pattern in Figure 7.44.

The application on folded shells hinted at the potential of informed exploration provided by performance evaluation of different design directions through the combination of their different strips. However, considering a finite set of rules is constrained, even if it includes all the deletion rules. Considering any addition rules allows comprehensive exploration of the de-

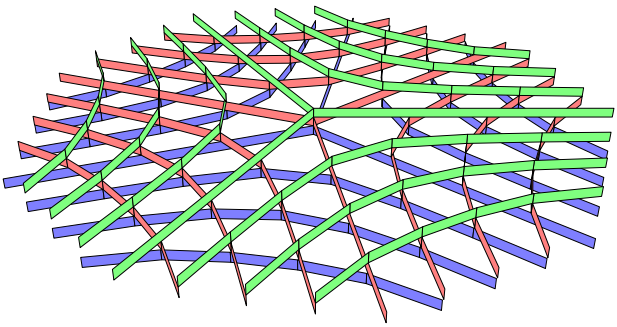


Figure 7.44 – A three-layer structure of beams organised in a three-coloured triangular pattern.

sign space. However, the encoding of these topological rules into a vector is necessary to serve as parameterisation for search algorithms, like genetic algorithms. Indeed, most search algorithms do not apply on graph data but on a vector of values or a string of characters.

## 7.6 Appendix

### 7.6.1 Algorithm validation results

Tables 7.8 to 7.12 in this appendix contains the detailed results for each validation example in Figures 7.28 to 7.31 in Section 7.3.4.2. Each table contains the number of potential combinations, the number of tested combinations due to the search reduction and the results against the two validity criteria: the two-coloured topologies with the right shape topology (O), the non-two-coloured topologies with the right shape topology (-) and the the topologies with a wrong shape topology (X). The total number of two-coloured designs with the right shape topology is highlighted in green and the termination criterion in red.

### 7.6.2 Structural application results

Tables 7.13 and 7.14 provide the numerical results of the performance analysis of the 42 folded shell designs in Section 7.4: six smooth designs plus six folded designs times two folding directions times three amplitudes. The

labels of each design describe topology (between 1 and 6), folding direction (1 or 2, or 0 if smooth) and folding amplitude (0.5 m, 1.0 m or 1.5 m). The provided shell area and the thickness respect a constant volume. The number of faces after discretisation for finite element analysis provides the density of the analysis mesh. The structural performance is evaluated based on the strain energy and completed by the maximum deflection, the maximum stress utilisation and the first buckling load factor.

Table 7.8 – Hexagonal shape (Figure 7.28)

| k | $\binom{n}{k}$ | ? | O | - | X |
|---|---|---|---|---|---|
| 1 | 7 | 7 | 3 | 4 | 0 |
| 2 | 21 | 6 | 4 | 2 | 0 |
| 3 | 35 | 0 | 0 | 0 | 0 |
| Σ | 63 | 13 | 7 | 6 | 0 |

Table 7.9 – Pentagonal shape (Figure 7.29a)

| k | $\binom{n}{k}$ | ? | O | - | X |
|---|---|---|---|---|---|
| 1 | 5 | 5 | 5 | 0 | 0 |
| Σ | 5 | 5 | 5 | 0 | 0 |

Table 7.10 – Pentagonal shape bis (Figure 7.29b)

| k | $\binom{n}{k}$ | ? | O | - | X |
|---|---|---|---|---|---|
| 1 | 6 | 6 | 0 | 6 | 0 |
| 2 | 15 | 15 | 8 | 7 | 0 |
| 3 | 20 | 3 | 0 | 3 | 0 |
| 4 | 15 | 0 | 0 | 0 | 0 |
| Σ | 56 | 24 | 8 | 16 | 0 |

Table 7.11 – Rectangular shape with one opening (Figure 7.30)

| k | $\binom{n}{k}$ | ? | O | - | X |
|---|---|---|---|---|---|
| 1 | 9 | 9 | 1 | 8 | 0 |
| 2 | 36 | 28 | 8 | 20 | 0 |
| 3 | 84 | 16 | 0 | 16 | 0 |
| 4 | 126 | 4 | 0 | 4 | 0 |
| 5 | 126 | 0 | 0 | 0 | 0 |
| Σ | 381 | 57 | 9 | 48 | 0 |

Table 7.12 – Rectangular shape with two openings (Figure 7.31)

| k | $\binom{n}{k}$ | ? | O | - | X |
|---|---|---|---|---|---|
| 1 | 14 | 14 | 0 | 14 | 0 |
| 2 | 91 | 91 | 2 | 83 | 6 |
| 3 | 364 | 282 | 20 | 236 | 26 |
| 4 | 1001 | 332 | 9 | 319 | 4 |
| 5 | 2002 | 202 | 0 | 202 | 0 |
| 6 | 3003 | 64 | 0 | 64 | 0 |
| 7 | 3432 | 10 | 0 | 10 | 0 |
| 8 | 3003 | 0 | 0 | 0 | 0 |
| Σ | 12910 | 995 | 31 | 928 | 36 |

Table 7.13 – Numerical results of the folded shells -part 1.

| top. | dir. | ampl. $[m]$ | area $[m^2]$ | thick. $[cm]$ | # faces | max. disp. $[cm]$ | strain energy $[kN.m]$ | max. stress util. $[-]$ | first buckl. load fact. $[-]$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 0 | 6001 | 20 | 7728 | 4.2 | 292 | 4.2 | 18 |
| 1 | 1 | 0.5 | 6210 | 19 | 7728 | 6.3 | 380 | 3.7 | 30 |
| 1 | 1 | 1 | 6800 | 18 | 7728 | 9.3 | 474 | 2.7 | 25 |
| 1 | 1 | 1.5 | 7686 | 16 | 7728 | 11.6 | 464 | 1.7 | 28 |
| 1 | 2 | 0.5 | 6279 | 19 | 7728 | 3.4 | 289 | 2.9 | 26 |
| 1 | 2 | 1 | 7047 | 17 | 7728 | 6.7 | 287 | 1.6 | 24 |
| 1 | 2 | 1.5 | 8149 | 15 | 7728 | 17.9 | 752 | 3 | 20 |
| 2 | 0 | 0 | 6000 | 20 | 10112 | 4 | 279 | 4.3 | 18 |
| 2 | 1 | 0.5 | 6242 | 19 | 10112 | 5.5 | 272 | 2.7 | 48 |
| 2 | 1 | 1 | 6894 | 17 | 10112 | 2.6 | 156 | 1.2 | 68 |
| 2 | 1 | 1.5 | 7838 | 15 | 10112 | 2.3 | 93 | 0.7 | 82 |
| 2 | 2 | 0.5 | 6600 | 18 | 10112 | 4.6 | 316 | 2.7 | 22 |
| 2 | 2 | 1 | 8024 | 15 | 10112 | 10.7 | 803 | 2.6 | 21 |
| 2 | 2 | 1.5 | 9896 | 12 | 10112 | 24.4 | 1440 | 3 | 12 |
| 3 | 0 | 0 | 6000 | 20 | 8320 | 4.2 | 291 | 4.3 | 18 |
| 3 | 1 | 0.5 | 6195 | 19 | 8320 | 7 | 328 | 3.1 | 38 |
| 3 | 1 | 1 | 6744 | 18 | 8320 | 2.7 | 169 | 1.2 | 61 |
| 3 | 1 | 1.5 | 7563 | 16 | 8320 | 2.3 | 101 | 0.6 | 73 |
| 3 | 2 | 0.5 | 6353 | 19 | 8320 | 5.2 | 406 | 3.4 | 21 |
| 3 | 2 | 1 | 7309 | 16 | 8320 | 9.4 | 715 | 2.7 | 21 |
| 3 | 2 | 1.5 | 8666 | 14 | 8320 | 18.7 | 1294 | 2.6 | 14 |

Table 7.14 – Numerical results of the folded shells -part 2.

| top. | dir. | ampl. $[m]$ | area $[m^2]$ | thick. $[cm]$ | # faces | max. disp. $[cm]$ | strain energy $[kN.m]$ | max. stress util. $[-]$ | first buckl. load fact. $[-]$ |
|---|---|---|---|---|---|---|---|---|---|
| 4 | 0 | 0 | 6001 | 20 | 10368 | 4.1 | 288 | 5 | 19 |
| 4 | 1 | 0.5 | 6310 | 19 | 10368 | 4.2 | 292 | 3.1 | 29 |
| 4 | 1 | 1 | 7156 | 17 | 10368 | 5.4 | 252 | 2.8 | 39 |
| 4 | 1 | 1.5 | 8366 | 14 | 10368 | 9.1 | 304 | 2.7 | 28 |
| 4 | 2 | 0.5 | 6482 | 19 | 10368 | 5.7 | 392 | 4.1 | 20 |
| 4 | 2 | 1 | 7673 | 16 | 10368 | 11.3 | 694 | 3.6 | 20 |
| 4 | 2 | 1.5 | 9262 | 13 | 10368 | 17.3 | 1036 | 3.6 | 11 |
| 5 | 0 | 0 | 6001 | 20 | 9984 | 4.2 | 287 | 4.6 | 18 |
| 5 | 1 | 0.5 | 6253 | 19 | 9984 | 5.4 | 321 | 3.4 | 31 |
| 5 | 1 | 1 | 6954 | 17 | 9984 | 7.2 | 294 | 1.9 | 44 |
| 5 | 1 | 1.5 | 7975 | 15 | 9984 | 9.3 | 329 | 1.7 | 27 |
| 5 | 2 | 0.5 | 6502 | 18 | 9984 | 5.8 | 305 | 3.2 | 20 |
| 5 | 2 | 1 | 7721 | 16 | 9984 | 8 | 521 | 2.2 | 25 |
| 5 | 2 | 1.5 | 9328 | 13 | 9984 | 24.6 | 1394 | 3.3 | 10 |
| 6 | 0 | 0 | 6001 | 20 | 12864 | 4.1 | 294 | 2.5 | 18 |
| 6 | 1 | 0.5 | 6329 | 19 | 12864 | 6.4 | 299 | 1.9 | 25 |
| 6 | 1 | 1 | 7204 | 17 | 12864 | 2.6 | 150 | 1 | 45 |
| 6 | 1 | 1.5 | 8440 | 14 | 12864 | 2.2 | 91 | 0.6 | 39 |
| 6 | 2 | 0.5 | 6611 | 18 | 12864 | 7.2 | 590 | 3 | 17 |
| 6 | 2 | 1 | 8120 | 15 | 12864 | 10.8 | 874 | 1.9 | 12 |
| 6 | 2 | 1.5 | 10090 | 12 | 12864 | 23.8 | 1860 | 1.9 | 9 |

# Part IV

# String-coded topology finding

# Chapter 8

# Alphabet-based exploration

## 8.1 Motivations

The quad-mesh grammar allows to add and delete strips. The application in Figure 8.1 of first an addition rule and second a deletion rule allows modifying the strips highlighted in blue. The added strips share some similarity with the deleted ones. The difference lies only in a pole, an extra face or a turn.

Nevertheless, the grammar rules do not allow modifying a strip, only to add and delete them. A set of operations that apply at a lower-level than the strips could allow such modifications. The definition of the concepts of similarity and distance would apply at a different level in searching for topological designs.

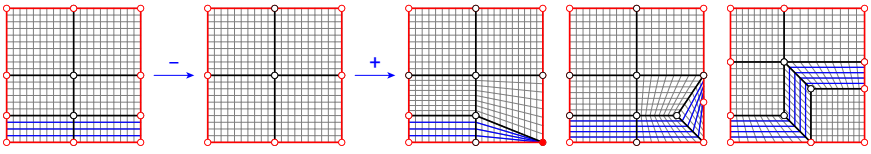



Figure 8.1 – The quad-mesh grammar allows deleting and adding the strips in blue. However, the grammar can not modify a strip, despite the similarity between the old and new strips.

Search algorithms like genetic algorithms enable to explore large design spaces for unstructured problems [Goldberg, 1989]. The flexibility of solvers

like multi-objective genetic algorithms allows to include a variety of goals and constraints. Evolutionary approaches rely on the survival of the fittest design during an evolution process of generation and selection of a pool of designs over multiple iterations. On the one hand, the designer can drive the selection process, which selects the preferred designs. The preference may be qualitative, such as the most beautiful, or quantitative such as the most efficient, if performance feedback informs the designer. On the other hand, the selection process can be automated by an algorithm, which selects the most efficient designs based on metrics defined by the designer. The selection process informs the generation of the next pool of designs by crossing over their genes. The set of genes forms the genotype. Interpretation of the genotype yields the phenotype of the design, its appearance and performance. Mutation of the genes can also occur to provide random modification in order to maintain the diversity of the pool of designs. Hybrid evolutionary approaches combine designer selection and algorithmic optimisation.

However, a string of values, characters or binaries encodes the genotype to feed the generation algorithms. For geometrical design, the continuous coordinates of the vertices of the pattern can compose such a string. For topological design, the rules must translate from a graph to a string of discrete values. For similarity-driven topology finding in Chapter 6 and two-coloured topology finding in Chapter 7, the $n$ strips stem from the input quad meshes. The combinations are therefore limited. A string can express the rules:

$$[\delta_i, \forall i \in 0, ..., n-1]^T, \tag{8.1}$$

where $\delta_i$ is a boolean value specifying if strip $i$ is to be added or deleted. This encoding strategy provides a combinatorial design space of $2^n$ strings. Even though the design space can be large, it remains bounded and does not offer the unlimited potential for exploration provided by the topological rules. Indeed, the unbounded number of strips, the unbounded length of the strips and their combinatorial nature poses a challenge for general string encoding of quad-mesh rules. Unlocking the full design space requires encoding any polyedge for the strip addition rules as well. Encoding the polyedges by their indices is not possible due to their infinite combinations. Encoding the polyedges by a list of vertex indices is not suitable neither as the combinatorics are higher, and only a small part of it represents actual polyedges. A list of $v$ vertices in a mesh within a total of $V$ vertices has a probability to yield a valid polyedge equal to the ratio of adjacent vertices

over total vertices for each vertex of the list:

$$(\frac{4}{V})^v, \tag{8.2}$$

for a uniform probability of selection of the vertices and an average valency of 4 in the quad mesh. Even for a simple case of a three-edge polyedge in a twenty-vertex mesh, a list of vertices has a probability below 1% to represent a polyedge. Moreover, the number and indices – or labels – of the mesh elements evolve due to the topological transformations. Therefore, keeping track of the elements is challenging.

Genetic algorithms is only an example of search algorithms that require string encoding in the realm of generative design and optimisation. Encoding the topological rules converts a shape grammar to a formal grammar.

This chapter introduces *alphabet-based topology finding*, using string operations to perform topological exploration. Section 8.2 introduces a strategy to encode the quad-mesh rules in a string of characters that represent operations. In analogy with genetic algorithms, the string represents the genotype of the topology of the design. Section 8.3 presents the modifications of the operations in the string. In analogy with genetic algorithms, these modifications of the strings act as mutations of the genotype. Section 8.4 defines a string distance and compares it to the topological distance. In analogy with genetic algorithms, these two distances relate to the genotype and the phenotype, respectively. Section 8.5 discusses application to evolution-driven topology finding through interactive exploration or objective optimisation.

## 8.2 String encoding

A string encodes the application of the strip rules on a quad-mesh strip. The string describes the visit of mesh half-edges to select strips to add and delete. In order to do so, operations are developed to describe the application of the rules.

### 8.2.1 Background

The inspiration for this string-encoding strategy to apply rules on a mesh stems from two concepts: turtle graphics programming and snake video game.

#### 8.2.1.1 Turtle graphics

Turtle graphics, related to the Logo programming language, find many applications in computer graphics and computational design and is praised for its educational value in programming [Goldman et al., 2004]. A cursor defined by its position and orientation is programmed to move in the 2D space. A turtle defined by its tail and its head can serve as equivalent interpretation. Examples of movement orders are 'move forward by 1 unit' or 'rotate leftward by 90°. Lines are drawn along the way, generating different patterns and geometries that stem from the sequence of orders. More advanced orders include moving by any distance, rotating by any angle, interrupting and resuming drawing or moving in the 3D space. Recursive application of the orders allows drawing fractal patterns, which are self-similar patterns at different scales.

#### 8.2.1.2 L-systems

Turtle graphics appear in L-systems, mentioned in Section 2.2.3.1, introduced by Aristid Lindenmayer in 1968 [Lindenmayer, 1968]. L-systems combine formal grammars as a string encoder and turtle graphics as a string interpreter to generate and explore the morphology and growth of plants [Prusinkiewicz and Lindenmayer, 2012]. L-systems have been applied to topology optimisation using genetic algorithms by evolving a structural layout within a domain [Kobayashi, 2010, Bielefeldt et al., 2019b, Hartl et al., 2016] or by evolving a partition of a domain [Pedro and Kobayashi, 2011, Stanford et al., 2013].

However, these generative strategies utilise turtle graphics to explore an empty domain. Topological exploration of patterns must integrate a mesh constraint. Bielefeldt *et al.* [Bielefeldt et al., 2019a] constrain exploration to a graph that represents a beam layout for evolutionary topology optimisation. Their graph-based turtle, or SPIDRS for spatial interpretation for the development of reconfigurable structures, moves between vertices and edges, along edges and across faces to add elements. However, the generated layouts are too unstructured and coarse for surface patterns.

#### 8.2.1.3 Snake game

The snake video game features a series of coloured pixels moving horizontally and vertically that grows when reaching a new coloured pixel. A similar collection process can be applied by a mesh-constrained turtle to apply strip

rules. Indeed, edges can be collected to constitute a polyedge to apply a strip addition rule while the turtle is moving along the mesh edges.

These concepts translate into a set of operations for a mesh-constrained turtle that applies strip rules. The hybrid between turtle and snake creates a lizard running along the edges of a mesh. The growing tail collects polyedges and gets cut after strip addition before growing back.

### 8.2.2 Movement operations

The movement operations along a mesh follow a halfedge data structure, illustrated in Figure 8.2. Each edge connecting two vertices $V_i$ and $V_j$ of the mesh produce two halfedges, in blue, oriented in opposite directions $(V_i, V_j)$ and $(V_j, V_i)$. The halfedges point towards one of the two adjacent faces. Except for boundary edges, in red, where one halfedge points to the virtual boundary face. In Figure 8.2, halfedge (1,4) points to face B, halfedge (4,1) points to face A, halfedge (1,0) points to face A as well, and halfedge (0,1) points to the virtual boundary face, for instance. The adjacency of the faces and the vertices are sorted and ordered in opposite directions here, illustrated by dashed orange circles. The mesh must be orientable for the definition of the halfedge data structure.

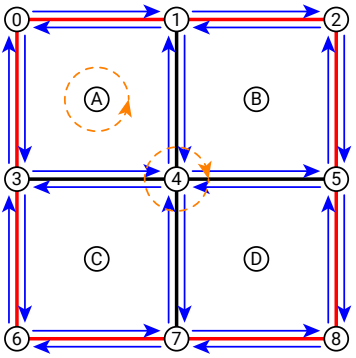



Figure 8.2 – Halfedge mesh data structure with edges producing oriented halfedges, in blue, pointing to the adjacent faces. The elements around the faces and the vertices are ordered and sorted in opposite directions, in orange.

The marker moves from one halfedge $(V_i, V_j)$ of the mesh to another, the start of the halfedge defining the position $S = V_i$ and the end of the

halfedge defining the direction $E = V_j$. In the different figures, the marker stands on one side of the edge to represent the corresponding halfedge.

An intuitive operation like 'forward' is not correctly defined. Figure 8.3 shows the ambiguity of such an operation for a marker directed towards a five-valent vertex.

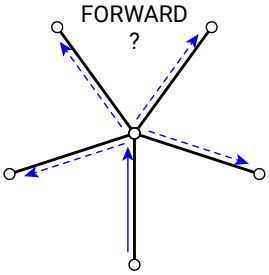



Figure 8.3 – Moving a marker forward is ambiguous: for a halfedge directed towards a n-valent vertex, as n-1 other halfedges may be chosen.

Therefore, two operations are created to move the marker along the halfedges:

- *turn* moves the marker to the next halfedge $(V_j, V_k)$ of the face of the current halfedge. The position and the direction become $V_j$ and $V_k$, respectively (Figure 8.4a);
- *pivot* moves the marker to the next halfedge of the current's vertex. The position does not change and the direction becomes $V_l$ (Figure 8.4b).

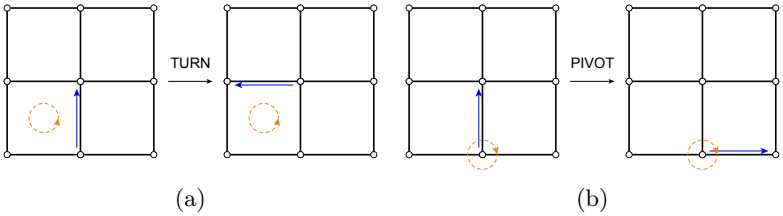
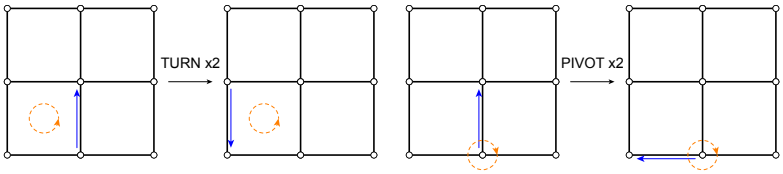



Figure 8.4 – The two movement operations to travel through the halfedges of a mesh.

The two operations can be extended with parameters to apply the operation multiple times:

- *turn(n)* applies *turn* n times, moving the marker to the next n-th halfedge of the face, as in Figure 8.5a for n=2;
- *pivot(n)* applies *pivot* n times to move the marker to the next n-th halfedge of the vertex, as in Figure 8.5b for n=2.

The parameters can be negative to yield the previous n-th halfedge by looping over the face or vertex adjacent elements in the reversed order.




(a) (b)

Figure 8.5 – Extension of the two movement operations with a parameter for multiple application of the operation.

These movement operations are not specific to a quad mesh and apply to any polygonal mesh. Thanks to these movement operations, the marker visits the mesh elements based on a string encoding the sequence of these operations. During these movements, the marker selects the mesh elements to apply strip rules.

### 8.2.3 Editing operations

To apply the rules of the quad-mesh grammar presented in Section 5.3, two editing operations are created for strip deletion and addition.

#### 8.2.3.1 Strip deletion operation

A *delete* operation deletes the strip to which the halfedge of the marker belongs. Figure 8.6 highlights the marker in blue, the transverse strip in a dashed red polyline and the resulting polyedge in red. Strip deletion deletes the edge of the marker. Therefore, the marker moves with the pivot operation to an edge of the polyedge resulting from the strip deletion.

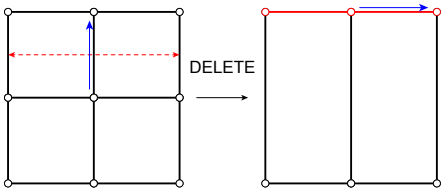



Figure 8.6 – The delete operation deletes the strip to which the halfedge, in blue, of the marker belongs to. The marker is beforehand positioned on the polyedge resulting from the deleted strip, in red.

If the edge is at an extremity of the strip, as in Figure 8.7, the pivot operation may not yield an edge of the polyedge. If so, the pivot operation with parameter -1 applies to move the marker to an edge in the other direction, guaranteeing the marker to be on the new polyedge, in red.

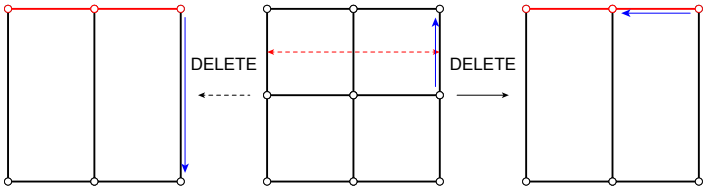



Figure 8.7 – When applying the delete operation, the pivot operation on the marker, in blue, may not yield a halfedge on the new polyedge, in red. Therefore, the marker is moved using the pivot operation with parameter -1, to bring the marker on the new polyedge.

#### 8.2.3.2 Strip addition operation

An *add* operation toggles an on-and-off state for the addition of a strip. Figure 8.8 provides an example where a three-face strip is added. The encoding string concatenates an add operation, several movement operations and another add operation to add a strip along the collected polyedge. When the addition state is toggled on, the collection of a polyedge starts. The first vertex is the position of the marker when the status is toggled, highlighted by a red circle. The polyedge grows with the successive positions of the marker, independently from its direction, highlighted by a red polyedge. Any turn and pivot operation can be applied to move the marker. Toggling

off the addition state adds a strip along the collected polyedge, highlighted by a dashed red polyline, and empties the collected polyedge. The marker stays on the same halfedge pointing to the same face. However, the labels of the vertices update due to the strip addition. The vertices of the new strip replace the vertices of the old polyedge.

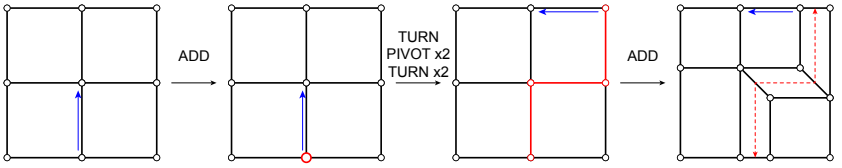



Figure 8.8 – The add operation toggles an on-and-off state to add a strip along a polyedge. Turning it on initiates the collection of a polyedge in red as the successive positions of the marker in blue. Turning it off adds a strip along the polyedge and empties the collected polyedge.

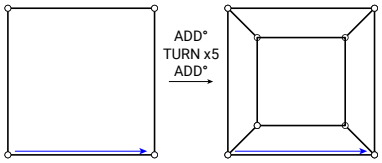
If the addition state is not on, the marker moves along the halfedges of the mesh without collecting a polyedge.

Additional parameters are necessary to include closed strips and strips with poles.

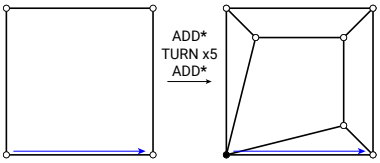
If a 'close' parameter, represented as °, is set to both add-on and add-off operations, the marker adds a closed strip, as in Figure 8.9a after collecting a closed four-edge polyedge. The polyedge must be closed for the 'close' parameter to apply.

If a 'pole' parameter, represented as *, is set to the add operation, the marker adds a strip with poles at the corresponding extremities. The add-on and add-off operations correspond to the start and the end of the added strip, respectively. The marker adds two poles if * is set to both add-on and add-off operations as in Figure 8.9b. The marker adds one pole to the start of the polyedge if * is set to the add-on operation as in Figure 8.9c. The marker adds one pole to the end of the polyedge if * is set to the add-off operation as in Figure 8.9d. If the start or the end is not on the boundary, the 'pole' parameter applies by default to the add-on or add-off operation, respectively.

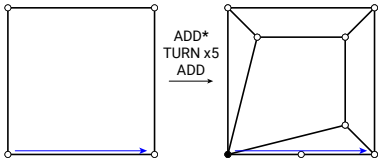
A variation adds the strip while collecting the polyedge, following the implementation of the strip addition rule, detailed in Section 5.3.1. In Figure 8.10, addition sets on and the strip faces, in red, are added after each turn movement, which moves the position of the marker and adds a vertex to the polyedge.



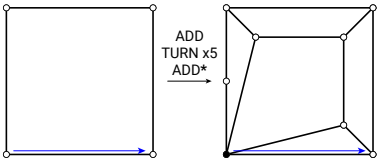

(a) Closed strip due to add-on and add-off operations with parameters °




(b) Two poles due to add-on and add-off operations with parameters *




(c) Pole at the start due to an add-on operation with parameter *




(d) Pole at the end due to an add-off operation with parameter *

Figure 8.9 – Extension of the add operation with parameters to add closed strips and strips with poles.

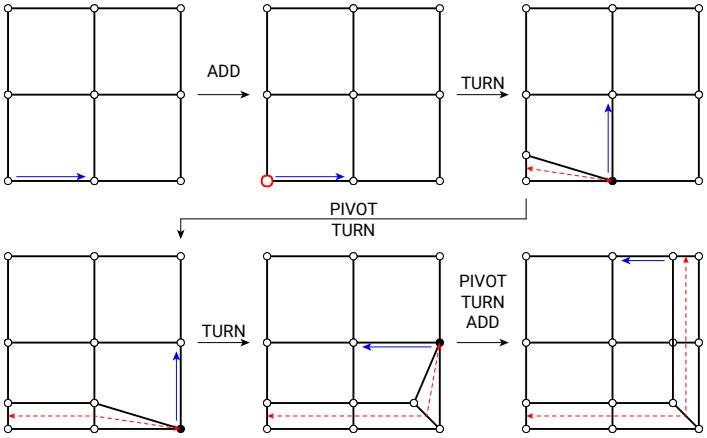



Figure 8.10 – The strips can be added progressively during the collection of the polyedge by adding a face of the strip after each turn movement, which moves the position of the marker and adds a vertex to the polyedge. The process follows the implementation of the strip addition rule.

#### 8.2.3.3 Combining add and delete operation

If a delete operation during polyedge collection for strip addition, the collected polyedge must be updated. The old polyedge $(V_0, ..., V_i, ...V_{n-1})$ is

updated by the function $f$ that maps the old vertices to the new vertices after strip deletions into the new polyedge $(f(V_0), ..., f(V_i), ...f(V_{n-1}))$. The new vertices replace the old vertices in the polyedge. If two consecutive vertices in the polyedge are identical, the update process removes one.

In Figure 8.11, 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.

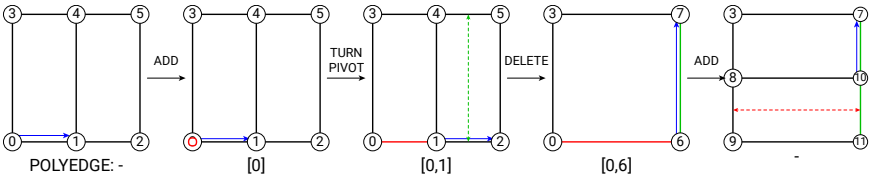



Figure 8.11 – The collected polyedge for strip addition, in red, updates when strip deletion applies, in green, by replacing the old vertices by the new ones.

### 8.2.4 Operation alphabet

The string encoding of the topological rules provides a genotype that describes the topological design. The genotype interpretation requires an initial quad-mesh topology and starting position and direction of the marker. The interpretation of the sequences of movement and editing operations can generate any quad-mesh topology, thanks to the application of the strip rules. Although, the shape topology is the same, meaning that the operations preserve the Euler characteristic and the number of boundaries. The operations are independent of the labels of the mesh vertices and faces.

The string – or genotype – encodes the sequence of operations as $T$ for turn, $P$ for pivot, $D$ for delete and $A$ for add. The parameters for closed strips, $^\circ$, and strips with poles, $^*$, are variations of A.

### 8.2.5 Examples

The generation of different strings based on this alphabet results in different quad-mesh topologies. In Figures 8.12 to 8.14, the strings are interpreted on the same initial quad mesh and same marker. The figures represent the added strips as dashed red curves. The previous positions of the marker are represented as dashed blue curves when multiple movement operations

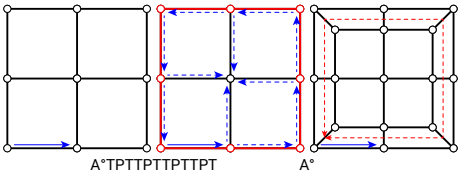



Figure 8.12 – The string A°TPTTPTTPTTPTA° adds a closed strip along the boundary.

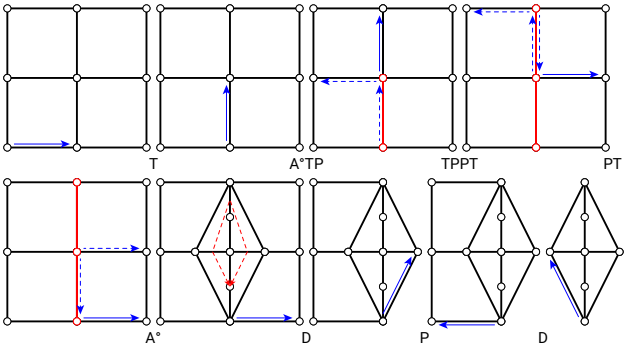



Figure 8.13 – The string TA°TPTPPTPTA°DPD adds another closed strip along the boundary and deleting two strips.

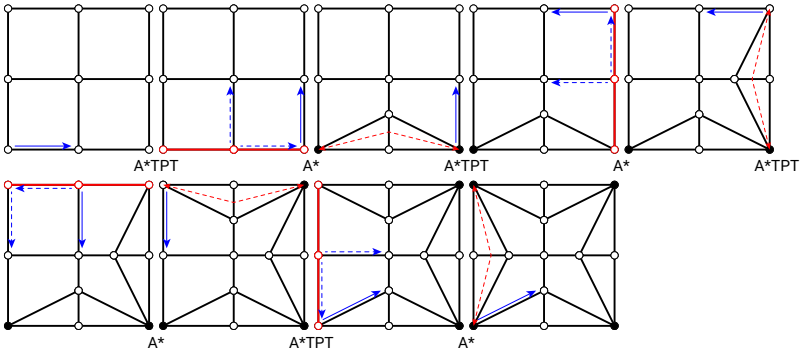



Figure 8.14 – The string A*TPTA*A*TPTA*A*TPTA*A*TPTA* adds four strips with poles along the boundary.

apply at once. The string A°TPTTPTTPTTPTA°, in Figure 8.12, adds a closed strip along the boundary. The string TA°TPTPPTPTA°DPD, in Figure 8.13, adds a different closed strip and deletes two strips. The string A*TPTA*A*TPTA*A*TPTA*A*TPTA*, in Figure 8.14, adds four strips with poles at both extremities along the boundary.

### 8.2.6 Lack of string-to-mesh isomorphism

This string encoding can generate any quad-mesh topology thanks to a combination of movement and editing operations. Meaning that there is a surjection from the string encoding – or genotype – to the mesh – or phenotype.

However, different combinations of movement and editing operations can result in the same quad-mesh topology. The string can describe the addition of a strip from any of its two extremities with the same result, for instance. Or the string can move the marker without applying strip rules with the same result, for instance. This redundancy means that there is no injection from the string to the quad-mesh topology.

Therefore, there is no bijection, or isomorphism, between the space of strings, i.e. the genotype, and the space of quad-mesh topologies, i.e. the phenotype, for the considered alphabet composing the string.

This redundancy in the genotypic encoding can be problematic for the application of genetic algorithms [Ronald et al., 1995, Echenagucia and Block, 2015]. Indeed, a pool of varied genotypes does not then necessarily result in a diverse pool of phenotypes. A lack of variety in designs and therefore in performances may cause the genetic algorithm to stagnate and stop improving. This risk depends on the degree of redundancy, i.e. the proportion of genotypes resulting in the same phenotypes.

Generating different strings by combining the letters of the alphabet generates different quad-mesh topologies. Nevertheless, modifications can apply on a starting string to generate and evolve other designs.

## 8.3 String modification

Applying modifications on the string allows controlling the changes made during topological exploration of quad meshes. The letter operations apply on the mesh, or phenotype, whereas, the string modifications apply on the string, or genotype. Following the analogy with genotypes, these modifications are equivalent to mutations.

For clarity, A°, for closed strips, and A*, for strips with poles, are not displayed. Nevertheless, the alphabet includes them along with A, D, T and P.

### 8.3.1 Definition

The modifications apply to the letters of the string to obtain different strings. In Figure 8.15, an initial string ATPTA is modified to obtain different topologies of quad meshes, starting from the same quad mesh and the same position and orientation of the marker in blue. The indices of the letters in the string start with 0. The modifications are highlighted in pink in the new strings. The added strips are represented as dashed red curves, the collected polyedges as red polyedges and the deleted strips as dashed green curves. The three types of modifications are: *insert*, *remove* and *substitute*. These three modifications allow converting one string into any other string. They allow to increase and decrease the length of the string and change its content. This set of modifications is the minimum one for maximum freedom.

#### 8.3.1.1 Insert

Mutation insert(X,i) inserts operation X at index i in the string. For instance, applying insert(T,4) on string ATPTA yields string ATPTTA. In Figure 8.15, the insertion of operations T lengthens the added strip. Inserting operation T before operation A to obtain ATPTTA only lengthens the strip, whereas inserting it earlier to obtain ATTPTA also changes the configuration of the strip. Inserting operation D to obtain ATPDTA deletes a strip and change the configuration of the added strip without changing its length.

#### 8.3.1.2 Remove

Mutation remove(i) removes the operation at index i in the string. For instance, applying remove(4) on string ATPTA yields string ATPT. In Figure 8.15, the removal of operations A cancels strip addition. If the second operation A is removed to obtain ATPT, the collected polyedge stays unchanged. Whereas if the first operation A is removed to obtain TPTA, the collected polyedge becomes a single vertex. The removal of an operation T to obtain ATPA shortens the added strip. The removal of the operation P to obtain ATTA modifies the configuration of the added strip. The strip ends with a

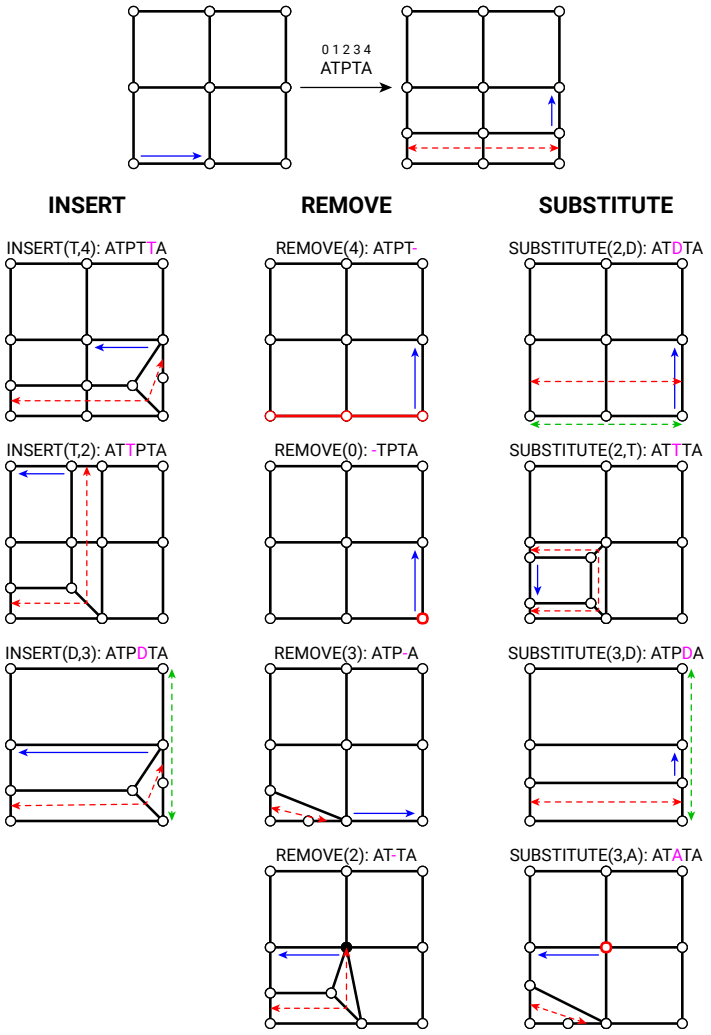



Figure 8.15 – Examples of application of the three types of letter modifications: insert, remove and substitute. The modifications in the initial string ATPTA are highlighted in pink. A dash marks the removed letters in the string. The added strip and collected polyedge are highlighted in red and the deleted strips in green.

pole because the extremity is no longer on the boundary and the strip does not close.

#### 8.3.1.3 Substitute

Mutation substitute(i,X) replaces the operation at index i in the string by operation X. For instance, applying substitute(3,D) on vector ATPTA yields string ATDTA. In Figure 8.15, the substitution of operation P by D to obtain ATDTA deletes one strip before adding a similar strip, cancelling each other. Substitution of operation P by T to obtain ATTTA lengthens the strip and changes its configuration. Substitution of one operation T by D to obtain ATPDA shortens the added strip and deletes a transverse strip. Substitution of operation P by A to obtain ATATA adds a shorter strip and starts collecting a polyedge for another strip addition from another position.

### 8.3.2 Combination

Modifications can be combined. In Figure 8.16, applying substitute(3,D) and remove(2) on ATPTA is performed in two steps: remove(2) is applied to obtain AT-TA and substitute(3,D) is applied to obtain AT-DA. The sequence of modifications updates the input indices when an insertion or removal modification modifies the length of the string. Therefore, applying substitute(3,D) on AT-TA becomes applying substitute(2,D) on ATTA.

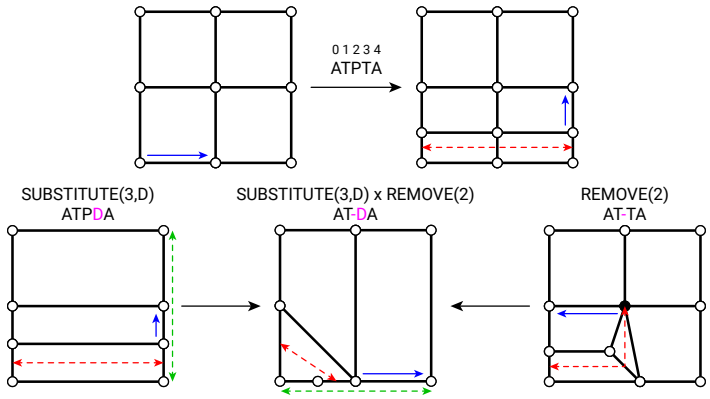



Figure 8.16 – Combining modifications: substitute(3,D) and remove(2) are applied on ATPTA to obtain ATDA. The results of the mutations are highlighted in pink.

The order of application of the modifications can yield different results. An example is insert(A,0) and insert(D,0) that insert two different operations at the same position. A second example is remove(0) and substitute(0,A) that apply two different mutations at the same position.

Therefore, a specific order is defined to combine modifications in a deterministic way. Per default, the different modifications apply in the following order:

1. insert;
2. remove;
3. substitute.

For instance, remove(0) applies before substitute(0,A). Per default, the same modifications with different operations apply in the following order:

1. T;
2. P;
3. D;
4. A.

For instance, insert(D,0) applies before insert(A,0)

A string distance is defined to measure the differences between two meshes at the level of the string, based on the string modifications.

## 8.4 String distance

Section 6.3.1 defines a topological distance between two quad meshes as the minimum number of strips to add and delete to obtain an isomorphism between the two quad meshes. This definition of the distance applies to the level of the phenotype of the design.

A distance that applies to the string, at the level of the genotype, can be defined as well.

### 8.4.1 Definition

In computer science and information theory, some string metrics evaluate the distance between two strings based on the number of modifications to apply

to convert one string into the other. The difference between these metrics lies in the set of allowed modifications: the Hamming distance [Hamming, 1950] allows substitutions only whereas the Levenshtein distance [Levenshtein, 1966] allows substitutions as well as insertions and deletions. Other string metrics follow different principles.

The distance between a pair of strings is defined as the minimum number of modifications to apply using insertions, deletions and substitutions. This distance corresponds to the Levenshtein distance. This distance verifies the three properties of distances on the space of strings $E$:

- the distance is symmetric:

$$\forall (A, B) \in E^2, d(A, B) = d(B, A); \tag{8.3}$$

- the distance from a string to itself is null and if their distance is null two strings identical:

$$\forall (A, B) \in E^2, d(A, B) = 0 \iff A = B; \tag{8.4}$$

- the triangle inequality is satisfied: the same or a lower number of modifications have to be applied to go directly from a string $A$ to a string $C$ than going first through an intermediary string $B$:

$$\forall (A, B, C) \in E^3, d(A, C) \leq d(A, B) + d(B, C). \tag{8.5}$$

The string distance and the mesh distance correspond to a genotype and a phenotype distance, respectively.

A fundamental difference exists between the two distance. The genotype distance is based on the applied modifications independently from the initial mesh, whereas the phenotype distance is based on the final mesh independently from the modification process. For instance, the difference in the final position of the marker or the remaining collected polyedge does not come into play.

Moreover, the influence of the rules and operations differ. The phenotype rules allow additions and deletions only. The genotype operations allow modifications as well, thanks to the substitute operation.

### 8.4.2 Examples

The genotype distance $d_G$ is evaluated on some examples and compared to the phenotype distance $d_P$.

The genotype distance is computed using the Wagner-Fischer algorithm, a dynamic-programming computation of the Levenshtein distance between two strings [Wagner and Fischer, 1974, Navarro, 2001].

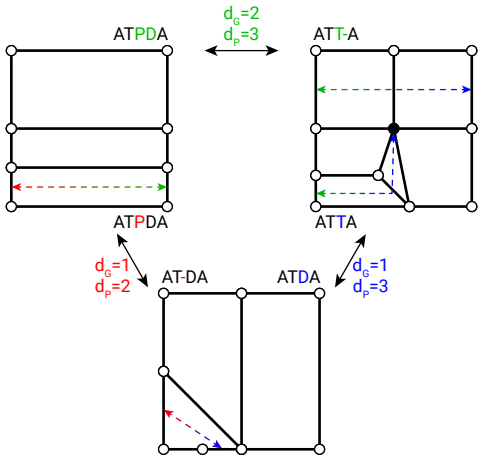



Figure 8.17 – Computing and comparing the genotype distance $d_G$ with the phenotype distance $d_P$ between three designs. The distances between the strings and meshes are highlighted by coloured letters and coloured strips, respectively. The two distances are not necessarily the same and one does not serve as a bound to the other.

Table 8.1 – Genotype distance $d_G$ and phenotype distance $d_P$ between a set of three strings and the resulting meshes. The results are symmetric.

| | ATPDA | ATTA | ATDA |
|---|---|---|---|
| ATPDA | $d_G = 0$<br>$d_P = 0$ | $d_G = 2$<br>$d_P = 3$ | $d_G = 1$<br>$d_P = 2$ |
| ATTA | | $d_G = 0$<br>$d_P = 0$ | $d_G = 1$<br>$d_P = 3$ |
| ATDA | sym. | | $d_G = 0$<br>$d_P = 0$ |

The two distances are computed for the strings ATPDA, ATTA and ATDA, and the resulting quad mesh after interpretation on the same input quad mesh with the same marker. The results are presented in Figure 8.17 and in Table 8.1. The differences are highlighted by coloured letters in the strings and coloured strips in the meshes.

The triangle inequality is verified for both distances. For instance,
$d_G(\text{ATPDA, ATDA}) = 1 \leq d_G(\text{ATPDA, ATTA}) + d_G(\text{ATTA, ATDA}) = 3$.

The genotype and phenotype distances between the same objects can differ from each other. For instance,
$d_G(\text{ATPDA, ATDA}) = 1 \neq d_P(\text{ATPDA, ATDA} = 2)$.

Moreover, one distance does not bound the other: in Figure 8.17
$d_G(\text{ATPDA, ATDA}) = 1 \leq d_P(\text{ATPDA, ATDA}) = 2$
and in Figure 8.18
$d_G(\text{ATPTA, ATTPPTTA}) = 3 \geq d_P(\text{ATPTA, ATTPPTTA}) = 2$.

The relation between the genotype and the phenotype distances is complex. The genotype applies at a lower-level than the phenotype. The string operations and the strip rules do not have the same impact. The string ATTA in Figure 8.17 adds only one strip but necessitates four operations. The two strings ATPDA and ATDA in Figure 8.17 both add one strip and delete one strip. Their genotype distance of one, as only one modification is necessary, is lower than their phenotype distance of two as they differ by two strips. In Figure 8.18, however, one strip is added in the designs ATPTA and ATTPPTTA. Their genotype distance of three is higher than the phenotype distance of two. Here, the configuration of the added strips is more different than the global configuration of all the strips.

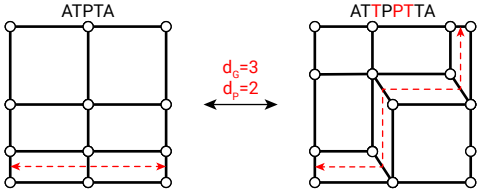



Figure 8.18 – Two designs with a genotype distance $d_G$ larger than the phenotype distance $d_P$.

Nevertheless, the sum of the delete and add-on/add-off operations in the pair of strings bounds the phenotype distance.

Due to the lack of isomorphism between the genotype and the phenotype, two different strings yielding the same mesh have a non-null genotype

distance but a null phenotype distance, as for ATPA and ATATA in Figure 8.15.

As evidenced by the comparison of the genotype and phenotype distances, topological exploration based on a string of operations instead of a configuration of strips is fundamentally different. This shift from a shape grammar to a formal grammar has implications in the exploration process. For instance, the similarity between the designs changes.

Nevertheless, string encoding opens topology finding to the wide set of search algorithms that require a vector input.

## 8.5 Towards evolution-driven topology finding

This strategy for alphabet-based exploration lies the groundwork towards evolution-driven topology finding, although some questions remain to be answered.

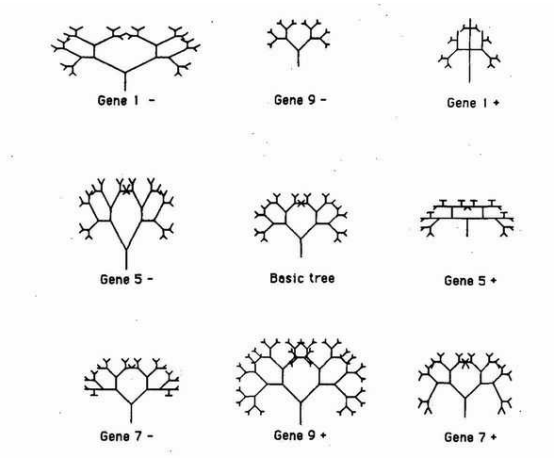
As expressed in this chapter, the string acts as genotype, the quad-mesh topology as phenotype and the modification as mutations.

Applying mutations on the genotypes results in mutated genotypes, and potentially mutated phenotypes. After selection of some of the mutated designs, the mutation-and-selection process repeats. In the case of genetic algorithms, the selection process automatically keeps the designs with the best performance.

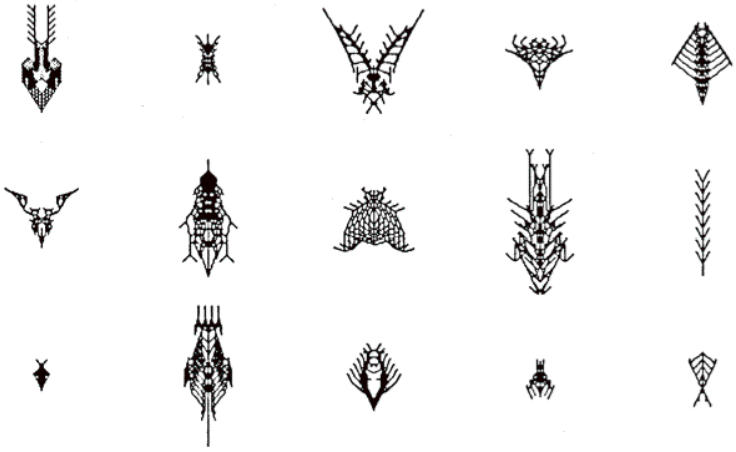
Nevertheless, a manual or partially-automated selection provides a means for the designer to include implicit design choices and engage in human-machine co-design. As such, the designer can influence a generation process based on randomness or optimisation to include various design aspects that can are not easily quantifiable like aesthetics [Mueller and Ochsendorf, 2015, Brown et al., 2015, Danhaive and Mueller, 2015, Turrin et al., 2011].

Beyond genetic algorithms, an exploration framework like the Biomorphs of Richard Dawkins [Dawkins, 1987] in Figure 8.19 allows such an evolution-driven design process. The goal is to replicate the step-by-step evolutionary process in nature. The interface in Figure 8.19a suggests eight designs that result from the mutation of the genotype of the current design in the centre. After many iterations with many mutations taking different routes, the obtained designs can be very different. Figure 8.19b shows such a set of designs that look like insects or plants.

The number of mutations simultaneously applied can start at a high value before decreasing, following a simulated annealing approach [Van Laarhoven and Aarts, 1987] to browse through a varied design space before converging.




(a) Selection of mutated designs



(b) Set of evolved designs

Figure 8.19 – Biomorph exploration by Richard Dawkins [Dawkins, 1987]. The evolutionary process is replicated through the iterative selection of mutated designs. Different routes result in different designs as varied as insect-like or plant-like designs.

Performance evaluation can inform the designer during the selection process.

Harding and Brandt-Olsen [Harding and Brandt-Olsen, 2018] present a Biomorpher to imitate this evolutionary design process in parametric design. Encoding topological exploration allows to apply such strategies to trans-parametric design.

Cross-over is an essential operation in evolutionary algorithms where two genotypes exchange sequences of their genes, as opposed to mutations that apply at the level of the gene. The use of cross-over raises the question of the definition of words made of several letters in this string-encoding strategy. For instance, the movement operations between a pair of add-on/add-off operations, which define the configuration of the added strip, can define a word. Taking into account this adjacency in the string creates a context-sensitive grammar, as opposed to the presented context-free grammar.

## 8.6 Summary of contributions

This chapter introduced alphabet-based topology finding, using string operations to perform topological exploration:

- a string-encoding strategy was introduced to describe the application of the strip rules on quad-meshes;
- operation modifications were used for the exploration of the string;
- a string distance based on the number of modifications was defined, and compared to the topological distance, which applies at the level of the mesh;
- potential application of this string-encoded strategy to evolution-driven topology finding was discussed, with the string as genotype and the mesh as phenotype.

On the one hand, the phenotype approach directly applies to the quad-mesh strips. On the other hand, the genotype approach uses lower-level operations encoding the application of the strip rules.

The complex relationship between the genotype and the phenotype, evidenced by the comparison of their two distances, should be analysed further for a better understanding of the genotype approach.

Finding the string that describes the operations between two input meshes would provide flexibility in the initial generation process. These input designs could stem from other strategies and combined with this string-encoding approach.

Tackling the current limitations would allow the application of these exploration strategies to genetic algorithms. These limitations include the lack of isomorphism between strings and meshes and lack of context-sensitivity of the operations.

More generally, string-based parameterisation opens topology finding to algorithms that require a vector for computing. The operations can be encoded as integers or binaries to convert the string into a vector of discrete values.

Section 5.4 utilises Self-Organising Maps (SOM) with vectors encoding performance as continuous values. Vectors encoding topology as discrete values can apply to SOMs as categorical data [Hsu, 2006]. SOMs can provide a visualisation means of the high-dimension space related to topology into a 2D map where close designs have similar topologies. Mixed vectors encoding discrete and continuous values can also be computed by SOMs to map designs based on their topological and performance similarity [Del Coso et al., 2015].

SOMs can be combined with clustering algorithms to reduce the number of designs to visualise using only one representative design per cluster. For instance, k-means clustering groups a set of vector-encoded designs into k clusters.

SOMs and k-means clustering are types of artificial neural networks and unsupervised machine learning techniques, respectively. Other algorithms from the field of artificial intelligence can enhance topology finding using the string encoding. Specifically, reinforcement learning could improve evolutionary co-design by starting with random mutations before suggesting mutations informed by the previous choices of mutations made by the designer.

# Part V

# Implementation

# Chapter 9

# compas_singular

The theoretical developments of the research in this thesis were supported by a continuous implementation, allowing testing, iterating, applying and sharing the different concepts.

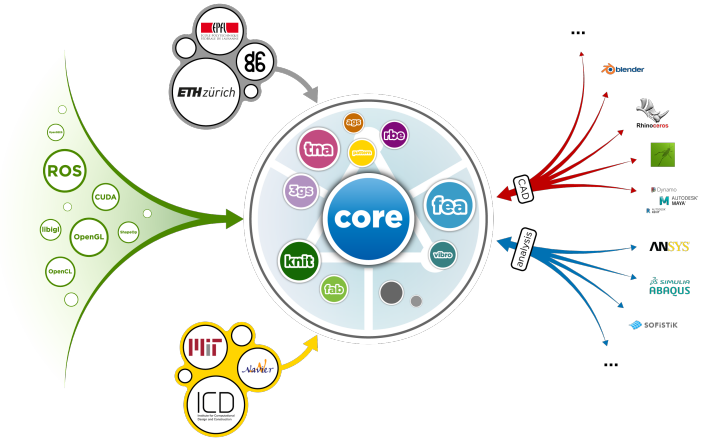



Figure 9.1 – The COMPAS framework developed by the Block Research Group of ETH Zurich. The package compas_singular results from the work in this thesis, in collaboration with the Laboratoire Navier in a project half-funded by l'École des Ponts ParisTech and half-funded by the Block Research Group. (Source: compas-dev.github.io)

## 9.1 Source

The implementation of this research on topology finding of patterns is made available in the public Github repository compas_singular [Oval, 2017].

https://github.com/BlockResearchGroup/compas_singular

compas_singular is a Python package of COMPAS, an open-source Python-based computational framework for research and collaboration in architecture, engineering, fabrication and construction. Figure 9.1 illustrates COMPAS' ecosystem [Van Mele et al., 2017].

### 9.1.1 Data structures

The data structure backing the topology-finding algorithms is a *coarse pseudo-quad mesh data structure*, a specific type of mesh. The coarse pseudo-quad meshes is the combination of coarse quad meshes and pseudo-quad meshes, which themselves stem from quad meshes. Figure 9.2 shows the progressive inclusion of a strip structure, density parameters and pole singularities in the initial mesh data structure.

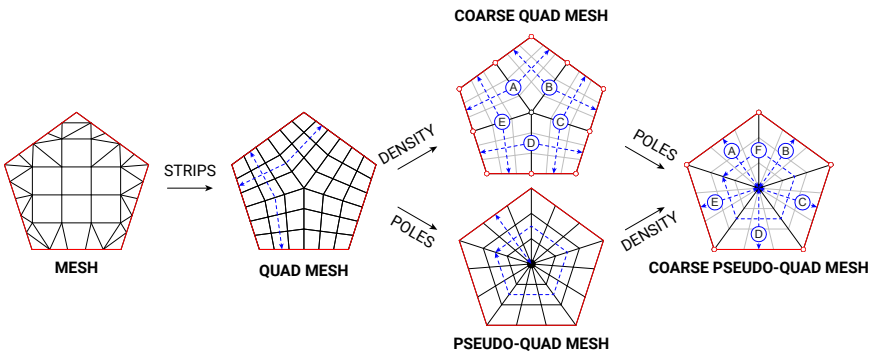



Figure 9.2 – From the data structures of meshes to coarse pseudo-quad meshes for topology finding of quad-mesh singularities. Data is added to integrate the strip structure in quad meshes, density parameters in coarse quad meshes and pole singularities in pseudo-quad meshes.

#### 9.1.1.1 Mesh

Data on both geometry and topology defines a mesh. The geometry stems from a list of vertices as point coordinates. The topology stems from a list of faces as lists of vertex keys. Based on this data, a more efficient half-edge mesh data structure can be computed [Botsch et al., 2010], as available in the core library of COMPAS.

#### 9.1.1.2 Quad mesh

Quad meshes are meshes with only quad faces, which are defined by exactly four vertices. In such meshes, regular vertices off the boundary have four adjacent vertices and regular vertices on the boundary have three adjacent vertices. Singularities are non-regular vertices. The specific structure in quad meshes allows the definition and computation of elements at an intermediary scale: quad-mesh strips and polyedges. Quad-mesh strips are consecutive edges opposite to each other in quad faces. These strips stop at boundaries or close on themselves. Quad-mesh polyedges are consecutive edges opposite to each other across the regular four-valent vertices. These polyedges stop at boundaries, at singularities or close on themselves.

#### 9.1.1.3 Coarse quad mesh

Coarse quad meshes are quad meshes with density parameters defined for each strip of quad faces for densification into a denser quad mesh. Parent-child relations exist between the different elements of the coarse and dense meshes: parent vertex to child vertex, parent edge to child polyedge, parent face to child faces and parent strip to child polyedges. These relations allow inheritance of data attributes and computing algorithms on dense meshes via their underlying coarse mesh.

#### 9.1.1.4 Pseudo-quad mesh

Pseudo-quad meshes are quad meshes allowing pseudo-quad faces. A pseudo-quad face has the geometry of a triangle but the topology of a quad. Triangular faces store the location of the pole among their three vertices. The location of the pole singularities influences the collection of the quad-mesh strips and polyedges.

#### 9.1.1.5 Coarse pseudo-quad mesh

Coarse pseudo-quad meshes combine coarse quad meshes and pseudo-quad meshes.

### 9.1.2 Algorithms

The algorithms include the research contributions presented in this thesis on topology finding of patterns in the directions of:

- geometry-encoded exploration;
- graph-encoded exploration; and
- string-encoded exploration.

### 9.1.3 Speed

Validation studies provide the computation times of the key algorithms, run on a 2.7 GHz Intel Core i5 processor.

The speed of the algorithms can be improved by:

- using a compiled language like C# or C++, instead of Python;
- applying parallel computing for the exploration of the multiple rule combinations;
- turning to a stochastic approach for search algorithms instead of an enumerative approach.

Nevertheless, the computation times prove that the topology-finding strategies are suitable for interactive design workflows on a normal machine. For instance, structural analysis and shape optimisation are more time consuming than topology finding in the presented applications.

## 9.2 Applications

Researchers and students have benefited from the presented research and provided important feedback during its development. Either using source codes, Rhino scripts or Grasshopper components. These motivated testers applied topology finding of patterns in the following research projects:

- optimisation of compression-only networks with Ricardo Avelino (ETH Zurich | SNSF);
- data-driven exploration of compression-only networks with Kam-Ming Mark Tam (ETH Zurich | ITA);
- supportless assembly of discrete structures with Gene Kao (ETH Zurich | NCCR DFAB);
- generation of robotic-arm tool paths for concrete 3D printing with Paul Carneau (Navier | ENPC);
- fabrication-aware mechanical optimisation of rebar cages for concrete structures with Sébastien Maitenaz (Navier | DiXite);

and in the following teaching programmes:

- Structural Design VI at ETH Zurich, Switzerland – Fall 2018;
- MIT Design-Build Workshop on Robotic Force Printing at Tongji University in Shanghai, China – January 2019;
- Advanced Master Design by Data at l'École des Ponts ParisTech, France – April 2019;
- Architecture of Weaving Workshop at Chalmers University of Technology in Gothenburg, Sweden – April 2019.

# Part VI

# Conclusion

# Chapter 10

# Conclusion

Crystals, as opposed to other materials, are characterised by a high degree of organisation, repeating a periodic pattern in the three directions of the physical space. Although the perfect pattern influences the material properties, the pattern imperfections have a more profound impact on these properties.

The closer to perfection an object or a shape, the more sensitive it is to imperfections.

Imperfections, irregularities or singularities in a regular surface pattern are a translation from 3D to 2D. Understanding and designing these singularities in the pattern allows having a profound qualitative and quantitative influence regarding both aesthetics and performance.

Concluding on this thesis, this chapter provides an overview of the presented research, which developed an algebra for the topological exploration of patterns for shell-like structures. Section 10.1 summarises the research contributions. Section 10.2 discusses the perspectives for future work in the light of the limitations and potentials of the current work. Section 10.3 concludes this thesis with final remarks.

## 10.1 Contributions

This thesis presents topology finding of patterns as a conceptual approach for trans-parametric design. The focus lies in the topological exploration of the set of singularities in quad-mesh patterns. Thereof, trans-parametric design shatters the inherent barriers in parametric design. Trans-parametric design enables structural exploration at another level across geometrical

design spaces. For the human to engage in interactive co-design with the machine, this thesis addresses the following challenges:

- comprehensive exploration of the design space: the designer can potentially reach any design in the topological space thanks to different encoding strategies using geometrical parameters, topological rules or string operations;
- constrained exploration of the design space: the designer can focus exploration on the subspaces that comply with topological requirements, like feature integration or pattern organisation;
- informed exploration of the design space: the designer can drive exploration based on feedback on the performance of designs to generate new designs at different degrees of similarity.

These new concepts, definitions and algorithms have been provided and evaluated on structural applications, including gridshells, vaults and shells using material patterns, force patterns or parameterisation patterns.

These contributions are shared via compas_singular, an open-source library, to allow the flexible combination with existing work like geometrical-exploration and topology-optimisation strategies, and to foster collaboration with other researchers and practitioners:

https://github.com/BlockResearchGroup/compas_singular.

This research opens several doors and triggers several questions to investigate.

## 10.2 Perspectives

Perspectives for future research are motivated by the current limitations of the research contributions and of the structural applications.

### 10.2.1 Richer architectural applications

Although enabled by the presented work, other applications could be investigated outside quad-mesh patterns and engineering and construction objectives.

The designer can explore the pattern topology of shapes with a non-null genus for building facades and envelopes, like the Morpheus Hotel in Macau.

Architectural objectives that relate to comfort like light, heat and acoustic can drive the design process.

Quad meshes can serve as parameterisation for other types of patterns, tessellations and tilings, like Conway operators. Pentagonal or hexagonal patterns mapped on quad meshes can be used for the design of a tessellated vault for instance. Exploring the singularities in the underlying quad mesh modifies the corresponding singularities in the mapped pattern.

Moreover, the boundary-alignment constraint can be fully or partially released thanks to trimming strategies, providing some flexibility in geometrical optimisation for instance.

### 10.2.2 Novel approach to topology optimisation

The different encoding strategies opens to the possibility of rethinking topology optimisation applied to large-scale shell-like structure based on any structural system and combined with geometrical optimisation. The optmisation algorithms depend on the encoding, with geometrical parameters, topological rules or string operations, which offer different pros and cons, and could therefore be combined.

### 10.2.3 Mapping and clustering for aided exploration

The designer can explore a potentially infinite number of designs in the topological space and their underlying parametric design spaces. Human-machine co-design can not be successful if the human is drowning under the amount of data computed by the machine. Thereof, mapping strategies can be used to organise designs in 2D maps based on topological and performance similarity. Clustering strategies further reduce visual complexity by providing a representative design of each cluster for the designer to visually and efficiently browse through design possibilities.

### 10.2.4 Data-driven exploration

The different encoding strategies enable the use of optimisation and learning algorithms. However, searching through a topological space is substantial, and relying on computational power should not replace computational efficiency. Finding heuristic instead of optimal designs is usually sufficient as structural and architectural design is a multi-objective design approach. Overfitting one objective with a perfect solution comes at a price on the other multiple objectives.

Moreover, exploration across the different design spaces on topology, density and geometry is a more challenging, mixed problem of continuous and discrete parameters. Probabilistic analysis and artificial intelligence techniques could be investigated to speed trans-space exploration. For instance, machine learning can predict results from geometrical optimisation for faster topological exploration. Informed exploration would not rely on the best performance but a probability of best performance after geometrical optimisation. Reinforcement learning applied at the level of topological exploration could guide the designer based on a mix of performance evaluation as well as design preferences.

### 10.2.5 From the discrete to the continuum

Homogenisation techniques [Winslow et al., 2010, Lebée and Sab, 2013] translate discrete structures into continuous models. To benefit from these techniques and combine them with the presented work, the influence of the pattern singularities has to be integrated in such models.

### 10.2.6 Advanced construction systems

Many structural systems rely on the simple assembly of geometrically regular patterns of elements that are easy to fabricate before deploying them during the construction process. The introduction of singularities in these patterns opens questions on the design and construction of these systems. Elastic gridshells [Soriano et al., 2015, Masson, 2017, Avelino and Baverel, 2017], woven structures [Martin, 2015, Ayres et al., 2018] or auxetic materials [Konaković-Luković et al., 2018a, Konaković-Luković et al., 2018b] are such examples. The initially regular elements become geometrically distorted once shaped into the final form. Introducing singularities can reduce geometrical distortions. The condition is to match singularities with positive or negative indices with areas of positive or negative curvature, respectively. As evidenced by the Poincaré-Hopf theorem and the Gauss-Bonnet theorem, matching the signs of indices and curvatures provide geometrically more regular patters on curved surfaces. On the contrary, not matching the signs would amplifying the geometrical distortions. These systems tightly connect topology and geometry. Non-Euclidean geometry offers a trade-off between irregularities in the geometry and the topology. A fundamental principle of these systems is to ease the fabrication, assembly and erection process. However, introducing singularities can cause problems in the deployment of these systems. Indeed, a singular element can block some kinematics. A

sequential construction process can solve such problems with a structure decomposed into singularity-free sub-structures.

### 10.2.7 Triangle-based patterns

Quad meshes are natural objects to model 2D surface-like objects. Quad meshes can serve as parameterisation maps for other patterns. However, the type of singularities stemming from quad meshes is limited. Indeed, quad meshes have singularities with indices as multiples of 1/4. However, the indices of singularities in triangular or hexagonal meshes are multiples of 1/6.

Designing a triangular-mesh pattern based on a quad-mesh pattern is therefore strongly limited. Moreover, some patterns like the Kagome patterns in woven structures derive more naturally from triangle meshes as parameterisation. The presented approach can adapt to the modelling of the family of triangle-parameterised patterns. Indeed, coarse triangular meshes can encode the data on the singularity. Moreover, equivalent strips exist in these triangular meshes to develop low-level grammar rules. Edges in triangular meshes belong to three strips and faces belong to three strips, instead of one and two, respectively, in the case of quad meshes. The coarse triangular meshes can be subdivided following the Loop subdivision algorithm [Loop, 1987], though providing only a unique global densification parameter. Optionally, the triangular mesh can serve as parameterisation map for another pattern. The mapped pattern features corresponding singularities and density as the triangular mesh.

Such a workflow allows designing Kagome patterns, which combine hexagons and triangles. Singular vertices in the coarse triangular meshes result in singular faces in the Kagome pattern like heptagons.

The two-colouring scheme can find an equivalent and adapt to triangular meshes. Indeed, a three-colouring property exists, which can be a requirement for the design of some systems.

### 10.2.8 Spatial patterns

Polyhedral meshes, spatial lattices or volumetric networks are 3D space-filling structures, like crystals, that can represent material at a mesoscale. The design of such 3D patterns of hexahedra also consists of regular elements with some singular elements that influence the orientation of the spatial pattern. The design of the singularities can influence the anisotropic mechanical behaviour. The design of a hexahedral-mesh grammar can apply

the addition and deletion of layers, the dimensional extension of strips as hyper-strips. Other polyhedra – or combinations of polyhedra – can constitute these 3D patterns.

### 10.2.9 Patterns in additive manufacturing

Automated fabrication using robotic arms can apply to additive manufacturing with the printing of concrete beam networks or shell layers [Duballet et al., 2019, Motamedi et al., 2019] or the winding of composite fibre structures [Prado et al., 2017, Spadea et al., 2017], for instance. The tool path of the robotic arm follows the pattern to create the structure However, the tool path is preferentially continuous. Indeed, temporarily interrupting the printing, or the winding process can be complicated or impossible in some applications, breaking or weakening the structural continuity. Integrating this construction-aware requirement for design a pattern for additive manufacturing implies the existence of an Eulerian path. An Eulerian path is a sequence of edges where each edge is visited exactly once. A stronger requirement may be to have a Eulerian cycle where the extremities meet to add several layers of material at the same time in a continuous manner. However, not all patterns or graphs feature such topological properties, as evidenced by Euler in the seven bridges of Königsberg problem [Euler, 1741]. The existence of a Eulerian cycle in a graph is equivalent to having an even valency at each vertex. This property relates only to the singularities in a quad mesh. Geometry does not play a role, neither do density as the regular vertices in a quad mesh already have a four valency. However, designing a pattern with such cycles must take into account material, geometry, static system and other aspects of the project.

### 10.2.10 Meshing applications in other fields

In addition to architectural and structural design applications, meshes have a wide range of computational applications where discrete data is stored and represented. Modelling in computer graphics and analysis in computational mechanics trigger extensive research in quad meshing. The presented quad-mesh grammar and the developed design algorithms can support existing meshing approaches, for the generation and optimisation of quad meshes.

## 10.3 Conclusions

Structure and pattern are both synonyms of order, organisation, harmony. Designing a structure means first designing the pattern underlying it. Structure and pattern are also synonyms of regularity. Nevertheless, regularity can be broken by irregularities that have a deep influence on the pattern. This thesis is about mastering these irregularities to bring them into the field of pattern design for structures.

Parametric design is spreading among students, researchers and practitioners. This thesis contributed to trans-parametric design for exploration across multiple parametric spaces. The evolving nature of the quantity and quality of parameters poses a profound challenge in comparing two parametric spaces with each other. The combinatorial nature of such problems is both a gift and a curse. Topology is a gift because it provides the designer with a rich space to explore, and a curse because its richness can be daunting and hard to master.

Deepening trans-parametric design raised the fundamental trade-off question for the development of concepts and algorithms for human-machine co-design between asking too much or too little from the designer. In co-design, the designer should be required to have a certain amount of knowledge, understanding and mastering of the proposed algorithmic design process rapidly.

Thanks to the increasing computational power, one can investigate the use of evolution-based or data-based algorithms to solve high-dimension problems. These tempting approaches still necessitate a long computation time or a large data set. Design algorithms should be the first approach to providing heuristic solutions to a problem.

Leonhard Euler stated that 'nothing takes place in the world whose meaning is not that of some maximum or minimum'. However, setting up optimisation algorithms to tackle all of the intricate problems of the world is not the right route. One shall not forget that structural design is embedded in a physical and societal reality. The former is immovable and will cause a design to be a failure, whereas the latter is evolving and will cause a design to be a disaster. The development of strategies and algorithms for design must be tailored for interactive and informed exploration to enable human-machine co-design that can flexibly evolve during the design process to build elegant, efficient, affordable and sustainable structures.

# List of Figures

# List of Tables

# Bibliography

[Adriaenssens et al., 2014] Adriaenssens, S., Block, P., Veenendaal, D., and Williams, C. (2014). *Shell structures for architecture: form finding and optimization*. Routledge.

[Adriaenssens et al., 2012] Adriaenssens, S., Ney, L., Bodarwe, E., and Williams, C. J. K. (2012). Finding the form of an irregular meshed steel and glass shell based on construction constraints. *Journal of Architectural Engineering*, 18(3):206–213.

[Agarwal and Cagan, 1998] Agarwal, M. and Cagan, J. (1998). A blend of different tastes: the language of coffeemakers. *Environment and Planning B: Planning and Design*, 25(2):205–226.

[Aish et al., 2018] Aish, R., Jabi, W., Lannon, S., Wardhana, N., and Chatzivasileiadi, A. (2018). Topologic: tools to explore architectural topology. In Hesselgren, L., Kilian, A., Sorkine-Hornung, O., Malek, S., Olsson, K.-G., and Williams, C. J. K., editors, *Advances in Architectural Geometry 2018*. Klein Publishing GmbH (Ltd.).

[Akleman et al., 2010] Akleman, E., Chen, J., and Gross, J. L. (2010). Strip sculptures. In *2010 Shape Modeling International Conference*, pages 236–240. IEEE.

[Akleman et al., 2016] Akleman, E., Ke, S., Wu, Y., Kalantar, N., Borhani, A., and Chen, J. (2016). Construction with physical version of quad-edge data structures. *Computers & Graphics*, 58:172–183.

[Alvarez et al., 2018] Alvarez, M. E., Martínez-Parachini, E. E., Baharlou, E., Krieg, O. D., Schwinn, T., Vasey, L., Hua, C., Menges, A., and Yuan, P. F. (2018). Tailored structures, robotic sewing of wooden shells.

In *Robotic Fabrication in Architecture, Art and Design*, pages 405–420. Springer.

[Antony et al., 2014] Antony, F., Grießhammer, R., Speck, T., and Speck, O. (2014). Sustainability assessment of a lightweight biomimetic ceiling structure. *Bioinspiration & biomimetics*, 9(1):016013.

[Avelino and Baverel, 2017] Avelino, R. and Baverel, O. (2017). Structural analysis of gridshells designed from singularities. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2017*.

[Ayres et al., 2018] Ayres, P., Martin, A. G., and Zwierzycki, M. (2018). Beyond the basket case: a principled approach to the modelling of kagome weave patterns for the fabrication of interlaced lattice structures using straight strips. In Hesselgren, L., Kilian, A., Sorkine-Hornung, O., Malek, S., Olsson, K.-G., and Williams, C. J. K., editors, *Advances in Architectural Geometry 2018*. Klein Publishing GmbH (Ltd.).

[Baldock, 2007] Baldock, R. (2007). *Structural optimisation in building design practice: case-studies in topology optimisation of bracing systems*. PhD thesis, University of Cambridge. Accessed on 26/10/2018.

[Baldock et al., 2005] Baldock, R., Shea, K., and Eley, D. (2005). Evolving optimized braced steel frameworks for tall buildings using modified pattern search. In *Computing in Civil Engineering (2005)*.

[Barnes, 1999] Barnes, M. R. (1999). Form finding and analysis of tension structures by dynamic relaxation. *International Journal of Space Structures*, 14(2):89–104.

[Baverel, 2000] Baverel, O. (2000). *Nexorades: a family of interwoven space structures*. PhD thesis, University of Surrey Guildford.

[Baverel and Popovic Larsen, 2011] Baverel, O. and Popovic Larsen, O. (2011). A review of woven structures with focus on reciprocal systems-nexorades. *International Journal of Space Structures*, 26(4):281–288.

[Beghini et al., 2014] Beghini, L. L., Beghini, A., Katz, N., Baker, W. F., and Paulino, G. H. (2014). Connecting architecture and engineering through structural topology optimization. *Engineering Structures*, 59:716–726.

[Beirão et al., 2011] Beirão, J. N., Duarte, J. P., and Stouffs, R. (2011). Creating specific grammars with generic grammars: towards flexible urban design. *Nexus Network Journal*, 13(1):73–111.

[Bendsøe and Sigmund, 2013] Bendsøe, M. P. and Sigmund, O. (2013). *Topology optimization: theory, methods, and applications*. Springer Science & Business Media.

[Bhooshan et al., 2018] Bhooshan, V., Bhooshan, S., and Block, P. (2018). Mayavault - A mesh modelling environment for discrete funicular structures. *Nexus Network Journal*, pages 1–16.

[Bielefeldt et al., 2019a] Bielefeldt, B. R., Akleman, E., Reich, G. W., Beran, P. S., and Hartl, D. J. (2019a). L-system-generated mechanism topology optimization using graph-based interpretation. *Journal of Mechanisms and Robotics*, 11(2):020905.

[Bielefeldt et al., 2019b] Bielefeldt, B. R., Reich, G. W., Beran, P. S., and Hartl, D. J. (2019b). Development and validation of a genetic l-system programming framework for topology optimization of multifunctional structures. *Computers & Structures*, 218:152–169.

[Bletzinger and Ramm, 1993] Bletzinger, K.-U. and Ramm, E. (1993). Form finding of shells by structural optimization. *Engineering with computers*, 9(1):27–35.

[Bletzinger and Ramm, 1999] Bletzinger, K.-U. and Ramm, E. (1999). A general finite element approach to the form finding of tensile structures by the updated reference strategy. *International Journal of Space Structures*, 14(2):131–145.

[Block and Ochsendorf, 2007] Block, P. and Ochsendorf, J. (2007). Thrust Network Analysis: A new methodology for three-dimensional equilibrium. *Journal of the International Association for Shell and Spatial Structures*, 48(3):167–173.

[Blum, 1967] Blum, H. (1967). A transformation for extracting new descriptors of shape. *Models for Perception of Speech and Visual Forms, 1967*, pages 362–380.

[Bobenko and Huhnen-Venedey, 2014] Bobenko, A. I. and Huhnen-Venedey, E. (2014). Curvature line parametrized surfaces and orthogonal coordinate systems: discretization with dupin cyclides. *Geometriae Dedicata*, 159(1):207–237.

[Bommes et al., 2009] Bommes, D., Zimmer, H., and Kobbelt, L. (2009). Mixed-integer quadrangulation. *Association for Computing Machinery Transactions On Graphics*, 28(3):77.

[Borgart, 2010] Borgart, A. (2010). New challenges for the structural morphology group. *Journal of the International Association for Shell and Spatial Structures*, 51(3):183–189.

[Botsch et al., 2010] Botsch, M., Kobbelt, L., Pauly, M., Alliez, P., and Lévy, B. (2010). *Polygon mesh processing*. CRC press.

[Bouaziz et al., 2014] Bouaziz, S., Martin, S., Liu, T., Kavan, L., and Pauly, M. (2014). Projective dynamics: fusing constraint projections for fast simulation. *Association for Computing Machinery Transactions on Graphics*, 33(4):154.

[Brocato and Mondardini, 2010] Brocato, M. and Mondardini, L. (2010). Geometric methods and computational mechanics for the design of stone domes based on Abeille's bond. In Ceccato, C., Hesselgren, L., Pauly, M., Pottmann, H., and Wallner, J., editors, *Advances in Architectural Geometry 2010*, pages 149–162. Springer.

[Brown et al., 2015] Brown, N., Tseranidis, S., and Mueller, C. (2015). Multi-objective optimization for diversity and performance in conceptual structural design. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2015*.

[Cagan, 2001] Cagan, J. (2001). Engineering shape grammars: where we have been and where we are going. In *Formal engineering design synthesis*, pages 65–92. Cambridge University Press.

[Caigui et al., 2019] Caigui, J., Peng, C.-H., Wonka, P., and Pottmann, H. (2019). Checkerboard patterns with black rectangles. *Association for Computing Machinery Transactions on Graphics*.

[Calladine, 1978] Calladine, C. (1978). Buckminster fuller's "tensegrity" structures and clerk maxwell's rules for the construction of stiff frames. *International journal of solids and structures*, 14(2):161–172.

[Campen et al., 2012] Campen, M., Bommes, D., and Kobbelt, L. (2012). Dual loops meshing: quality quad layouts on manifolds. *Association for Computing Machinery Transactions on Graphics*, 31(4):110.

[Campen et al., 2015] Campen, M., Bommes, D., and Kobbelt, L. (2015). Quantized global parametrization. *Association for Computing Machinery Transactions on Graphics*, 34(6):192.

[Campen and Kobbelt, 2014] Campen, M. and Kobbelt, L. (2014). Dual strip weaving: Interactive design of quad layouts using elastica strips. *Association for Computing Machinery Transactions on Graphics*, 33(6):183.

[Catmull and Clark, 1978] Catmull, E. and Clark, J. (1978). Recursively generated B-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10(6):350–355.

[Chew, 1989] Chew, P. L. (1989). Constrained Delaunay triangulations. *Algorithmica*, 4(1-4):97–108.

[Chomsky, 1956] Chomsky, N. (1956). Three models for the description of language. *IRE Transactions on information theory*, 2(3):113–124.

[Chomsky, 1959] Chomsky, N. (1959). On certain formal properties of grammars. *Information and control*, 2(2):137–167.

[Clune, 2013] Clune, R. P. (2013). *Algorithm selection in structural optimization*. PhD thesis, Massachusetts Institute of Technology. Accessed on 06/02/19.

[Conway et al., 2016] Conway, J. H., Burgiel, H., and Goodman-Strauss, C. (2016). *The symmetries of things*. CRC Press.

[Cordella et al., 2001] Cordella, L. P., Foggia, P., Sansone, C., and Vento, M. (2001). An improved algorithm for matching large graphs. In *3rd IAPR-TC15 workshop on graph-based representations in pattern recognition*, pages 149–159.

[Dailey, 1980] Dailey, D. P. (1980). Uniqueness of colorability and colorability of planar 4-regular graphs are np-complete. *Discrete Mathematics*, 30(3):289–293.

[Danhaive and Mueller, 2015] Danhaive, R. A. and Mueller, C. T. (2015). Combining parametric modeling and interactive optimization for high-performance and creative structural design. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2015*.

[Daniels et al., 2008] Daniels, J., Silva, C. T., Shepherd, J., and Cohen, E. (2008). Quadrilateral mesh simplification. *Association for Computing Machinery Transactions on Graphics*, 27(5):148.

[Daniels II et al., 2009] Daniels II, J., Silva, C. T., and Cohen, E. (2009). Localized quadrilateral coarsening. *Computer Graphics Forum*, 28(5):1437–1444.

[Dawkins, 1987] Dawkins, R. (1987). *The Blind Watchmaker*. Norton & Company, Inc.

[Deb, 2001] Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms*, volume 16. John Wiley & Sons.

[Del Coso et al., 2015] Del Coso, C., Fustes, D., Dafonte, C., Nóvoa, F. J., Rodríguez-Pedreira, J. M., and Arcay, B. (2015). Mixing numerical and categorical data in a self-organizing map by means of frequency neurons. *Applied Soft Computing*, 36:246–254.

[Deleuran et al., 2016] Deleuran, A. H., Pauly, M., Tamke, M., Tinning, I. F., and Thomsen, M. R. (2016). Exploratory topology modelling of form-active hybrid structures. *Procedia Engineering*, 155:71–80.

[Deuss et al., 2015] Deuss, M., Deleuran, A. H., Bouaziz, S., Deng, B., Piker, D., and Pauly, M. (2015). Shapeop—a robust and extensible geometric modelling paradigm. In *Modelling Behaviour*, pages 505–515. Springer.

[Douthe et al., 2017] Douthe, C., Mesnil, R., Orts, H., and Baverel, O. (2017). Isoradial meshes: Covering elastic gridshells with planar facets. *Automation in Construction*, 83:222–236.

[Du Peloux et al., 2013] Du Peloux, L., Baverel, O., Caron, J.-F., and Tayeb, F. (2013). From shape to shell: a design tool to materialize freeform shapes using gridshell structures. In *Rethinking Prototyping: Design Modelling Symposium Berlin 2013*.

[Du Peloux et al., 2015] Du Peloux, L., Tayeb, F., Caron, J.-F., and Baverel, O. (2015). The ephemeral cathedral of Créteil: a 350m2 lightweight gridshell structure made of 2 kilometers of GFRP tubes. In *Conférence Internationale de Géotechniques, des Ouvrages et Structures 2015: Innovations in Construction*. Accessed on 26/10/2018.

[Duarte, 2005] Duarte, J. P. (2005). A discursive grammar for customizing mass housing: the case of siza's houses at malagueira. *Automation in construction*, 14(2):265–275.

[Duarte et al., 2006] Duarte, J. P., Ducla-Soares, G., Caldas, L. G., and Rocha, J. (2006). An urban grammar for the medina of marrakech. In *Design computing and cognition'06*, pages 483–502. Springer.

[Duballet et al., 2019] Duballet, R., Baverel, O., and Dirrenberger, J. (2019). Space truss masonry walls with robotic mortar extrusion. *Structures*, 18:41–47.

[Dürr, 2000] Dürr, H. (2000). Seilnetze-Planung, Berechnung, Ausführung und Werkplanung. *Stahlbau*, 69(8):585–594.

[e Costa and Duarte, 2013] e Costa, E. C. and Duarte, J. P. (2013). Mass customization of ceramic tableware through digital technology. *Green Design, Materials and Manufacturing Processes*, pages 467–471.

[Echenagucia and Block, 2015] Echenagucia, T. M. and Block, P. (2015). Acoustic optimization of funicular shells. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2015*.

[Eloy and Duarte, 2011] Eloy, S. and Duarte, J. P. (2011). A transformation grammar for housing rehabilitation. *Nexus Network Journal*, 13(1):49–71.

[Eppstein, 2007] Eppstein, D. (2007). Lecture notes in ics 161: Design and analysis of algorithms. Accessed on 04/09/2019.

[Ertelt and Shea, 2009] Ertelt, C. and Shea, K. (2009). Generative design and cnc fabrication using shape grammars. In *ASME 2008 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 25–34. American Society of Mechanical Engineers Digital Collection.

[Estrin et al., 2011] Estrin, Y., Dyskin, A. V., and Pasternak, E. (2011). Topological interlocking as a material design concept. *Materials Science and Engineering: C*, 31(6):1189–1194.

[Euler, 1741] Euler, L. (1741). Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, pages 128–140.

[Fedorov, 1891] Fedorov, E. (1891). Simmetrija na ploskosti [symmetry in the plane]. *Zapiski Imperatorskogo Sant-Petersburgskogo Mineralogicheskogo Obshchestva [Proceedings of the Imperial St. Petersburg Mineralogical Society]*, 28:245–291.

[Flemming, 1987] Flemming, U. (1987). More than the sum of parts: the grammar of queen anne houses. *Environment and Planning B: Planning and Design*, 14(3):323–350.

[Fogg et al., 2016] Fogg, H. J., Armstrong, C. G., and Robinson, T. T. (2016). Enhanced medial-axis-based block-structured meshing in 2-D. *Computer-Aided Design*, 72:87–101.

[Fogg et al., 2018] Fogg, H. J., Sun, L., Makem, J. E., Armstrong, C. G., and Robinson, T. T. (2018). Singularities in structured meshes and cross-fields. *Computer-Aided Design*, 105:11–25.

[food4Rhino, 2019] food4Rhino (2019). Apps for rhino and grasshopper. Accessed on 10/10/2019.

[Francis and Weeks, 1999] Francis, G. K. and Weeks, J. R. (1999). Conway's zip proof. *The American mathematical monthly*, 106(5):393–399.

[Fuhrimann et al., 2018] Fuhrimann, L., Moosavi, V., Ohlbrock, P. O., and D'acunto, P. (2018). Data-driven design: Exploring new structural forms using machine learning and graphic statics. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2018*.

[Garcia, 2017] Garcia, S. (2017). Classifications of shape grammars. In *Design Computing and Cognition'16*, pages 229–248. Springer.

[Geyer, 2008] Geyer, P. (2008). Multidisciplinary grammars supporting design optimization of buildings. *Research in Engineering Design*, 18(4):197–216.

[Ghys, 1995] Ghys, É. (1995). Topologie des feuilles génériques. *Annals of Mathematics*, pages 387–422.

[Glymph et al., 2004] Glymph, J., Shelden, D., Ceccato, C., Mussel, J., and Schober, H. (2004). A parametric strategy for free-form glass structures using quadrilateral planar facets. *Automation in Construction*, 13(2):187–202.

[Goldberg, 1989] Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley Longman Publishing Co., Inc.

[Goldman et al., 2004] Goldman, R., Schaefer, S., and Ju, T. (2004). Turtle geometry in computer graphics and computer-aided design. *Computer-Aided Design*, 36(14):1471–1482.

[Gondran and Minoux, 1984] Gondran, M. and Minoux, M. (1984). *Graphs and algorithms*. Wiley.

[Grünbaum and Shephard, 1987] Grünbaum, B. and Shephard, G. C. (1987). *Tilings and patterns*. Freeman.

[Hagberg et al., 2008] Hagberg, A., Swart, P., and S Chult, D. (2008). Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States).

[Halpern et al., 2013] Halpern, A. B., Billington, D. P., and Adriaenssens, S. (2013). The ribbed floor slab systems of Pier Luigi Nervi. *Journal of the International Association for Shell and Spatial Structures*, 2013(23):1–7.

[Hamming, 1950] Hamming, R. W. (1950). Error detecting and error correcting codes. *The Bell system technical journal*, 29(2):147–160.

[Hansmeyer and Dillenburger, 2013] Hansmeyer, M. and Dillenburger, B. (2013). Mesh grammars. In Stouffs, R., Janssen, P., Roudavski, S., and Tunçer, B., editors, *Conference on Computer-Aided Architectural Design Research in Asia 2013*, pages 821–829.

[Harding, 2016] Harding, J. (2016). Dimensionality reduction for parametric design exploration. In Adriaenssens, S., Gramazio, F., Kohler, M., Menges, A., and Pauly, M., editors, *Advances in Architectural Geometry 2016*, pages 274–87. vdf Hochschulverlag AG.

[Harding and Brandt-Olsen, 2018] Harding, J. and Brandt-Olsen, C. (2018). Biomorpher: Interactive evolution for parametric design. *International Journal of Architectural Computing*, 16(2):144–163.

[Harding et al., 2012] Harding, J., Joyce, S., Shepherd, P., and Williams, C. J. K. (2012). Thinking topologically at early stage parametric design. In Hesselgren, L., Sharma, S., Wallner, J., Baldassini, N., Bompas, P., and Raynaud, J., editors, *Advances in Architectural Geometry 2012*. Springer.

[Harding and Shepherd, 2017] Harding, J. E. and Shepherd, P. (2017). Meta-parametric design. *Design Studies*, 52:73–95.

[Hartl et al., 2016] Hartl, D. J., Reich, G. W., and Beran, P. S. (2016). Additive topological optimization of muscular-skeletal structures via genetic l-system programming. In *24th AIAA/AHS Adaptive Structures Conference*, page 1569.

[He and Gilbert, 2016] He, L. and Gilbert, M. (2016). Automatic rationalization of yield-line patterns identified using discontinuity layout optimization. *International Journal of Solids and Structures*, 84:27–39.

[Hecker, 1969] Hecker, H.-D. (1969). Der Hörsaal des Zoologischen Instituts der Universität Freiburg. *Freiburger Universitätsblätter*, 25:49–52.

[Heisserman, 1994] Heisserman, L. (1994). Generative geometric design. *IEEE Computer Graphics and Applications*, 14(2):37–45.

[Heyman, 1997] Heyman, J. (1997). *The stone skeleton: structural engineering of masonry architecture*. Cambridge University Press.

[Hojjat et al., 2014] Hojjat, M., Stavropoulou, E., and Bletzinger, K.-U. (2014). The vertex morphing method for node-based shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 268:494–513.

[Hsu, 2006] Hsu, C.-C. (2006). Generalizing self-organizing map for categorical data. *IEEE transactions on Neural Networks*, 17(2):294–304.

[Hu et al., 2012] Hu, S., Xing, Q., Akleman, E., Chen, J., and Gross, J. (2012). Pattern mapping with quad-pattern-coverable quad-meshes. *Computers & Graphics*, 36(5):455–465.

[Hull, 1994] Hull, T. (1994). On the mathematics of flat origamis. *Congressus numerantium*, pages 215–224.

[Jakob et al., 2015] Jakob, W., Tarini, M., Panozzo, D., and Sorkine-Hornung, O. (2015). Instant field-aligned meshes. *Association for Computing Machinery Transaction on Graphics.*, 34(6):189–1.

[Jiang et al., 2015] Jiang, C., Tang, C., Vaxman, A., Wonka, P., and Pottmann, H. (2015). Polyhedral patterns. *Association for Computing Machinery Transactions on Graphics*, 34(6):172.

[Johnson, 2015] Johnson, S. G. (2015). The nlopt nonlinear-optimization package. Accessed on 06/02/2019.

[Kälberer et al., 2007] Kälberer, F., Nieser, M., and Polthier, K. (2007). Quadcover-surface parameterization using branched coverings. *Computer graphics forum*, 26(3):375–384.

[Kilian et al., 2017] Kilian, M., Pellis, D., Wallner, J., and Pottmann, H. (2017). Material-minimizing forms and structures. *Association for Computing Machinery Transactions on Graphics*, 36(6):173.

[Knight and Stiny, 2015] Knight, T. and Stiny, G. (2015). Making grammars: From computing with shapes to computing with things. *Design Studies*, 41:8–28.

[Knight, 1980] Knight, T. W. (1980). The generation of hepplewhite-style chair-back designs. *Environment and planning B: planning and design*, 7(2):227–238.

[Kobayashi, 2010] Kobayashi, M. H. (2010). On a biologically inspired topology optimization method. *Communications in Nonlinear Science and Numerical Simulation*, 15(3):787–802.

[Kohonen, 2013] Kohonen, T. (2013). Essentials of the self-organizing map. *Neural networks*, 37:52–65.

[Konaković-Luković et al., 2018a] Konaković-Luković, M., Konaković, P., and Pauly, M. (2018a). Computational design of deployable auxetic shells. In Hesselgren, L., Kilian, A., Sorkine-Hornung, O., Malek, S., Olsson, K.-G., and Williams, C. J. K., editors, *Advances in Architectural Geometry 2018*. Klein Publishing GmbH (Ltd.).

[Konaković-Luković et al., 2018b] Konaković-Luković, M., Panetta, J., Crane, K., and Pauly, M. (2018b). Rapid deployment of curved surfaces via programmable auxetics. *Association for Computing Machinery Transactions on Graphics*, 37(4):106.

[Koning and Eizenberg, 1981] Koning, H. and Eizenberg, J. (1981). The language of the prairie: Frank lloyd wright's prairie houses. *Environment and planning B: planning and design*, 8(3):295–323.

[Koronaki et al., 2017] Koronaki, A., Shepherd, P., and Evernden, M. (2017). Layout optimization of space frame structures. In *Proceedings*

*of the Annual Symposium of the International Association for Shell and Spatial Structures 2017.*

[Längst et al., 2017] Längst, P., Bauer, A. M., Michalski, A., and Lienhard, J. (2017). The potentials of isogeometric analysis methods in integrated design processes. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2017.*

[Leach, 1992] Leach, G. (1992). Improving worst-case optimal delaunay triangulation algorithms. In *4th Canadian Conference on Computational Geometry*, volume 2, page 15. Citeseer.

[Lebée, 2015] Lebée, A. (2015). From folds to structures, a review. *International Journal of Space Structures*, 30(2):55–74.

[Lebée and Sab, 2013] Lebée, A. and Sab, K. (2013). Homogenization of a space frame as a thick plate: Application of the bending-gradient theory to a beam lattice. *Computers & Structures*, 127:88–101.

[Lee et al., 2016a] Lee, J., Meled, T. V., and Block, P. (2016a). Form-finding explorations through geometric transformations and modifications of force polyhedrons. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2016.*

[Lee et al., 2016b] Lee, J., Mueller, C., and Fivet, C. (2016b). Automatic generation of diverse equilibrium structures through shape grammars and graphic statics. *International Journal of Space Structures*, 31(2-4):147–164.

[Leroy, 1890] Leroy, C. F. A. (1890). *Traité de stéréotomie: comprenant les applications de la géométrie descriptive à la théorie des ombres, la perspective linéaire, la gnomonique, la coupe des pierres et la charpente; avec un atlas composé de 76 planches infolio.* Gauthier-Villars et fils.

[Levenshtein, 1966] Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady*, 10(8):707–710.

[Li et al., 2001] Li, A. I. et al. (2001). *A shape grammar for teaching the architectural style of the Yingzao fashi.* PhD thesis, Massachusetts Institute of Technology. Accessed on 13/09/2019.

[Liew et al., 2019] Liew, A., Avelino, R., Moosavi, V., Van Mele, T., and Block, P. (2019). Optimising the load path of compression-only thrust networks through independent sets. *Structural and Multidisciplinary Optimization*, 60(1):231–244.

[Liew et al., 2018] Liew, A., Pagonakis, D., Van Mele, T., and Block, P. (2018). Load-path optimisation of funicular networks. *Meccanica*, 53(1-2):279–294.

[Lindenmayer, 1968] Lindenmayer, A. (1968). Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of theoretical biology*, 18(3):280–299.

[Liu et al., 2006] Liu, Y., Pottmann, H., Wallner, J., Yang, Y.-L., and Wang, W. (2006). Geometric modeling with conical meshes and developable surfaces. *Association for Computing Machinery Transactions on Graphics*, 25(3):681–689.

[Liu and Wang, 2008] Liu, Y. and Wang, W. (2008). On vertex offsets of polyhedral surfaces. In Pottman, H., Kilian, A., and Hofer, M., editors, *Proceedings of the Advances in Architectural Geometry 2008*, pages 61–64. TU Wien, Vienna.

[Liu et al., 2011] Liu, Y., Xu, W., Wang, J., Zhu, L., Guo, B., Chen, F., and Wang, G. (2011). General planar quadrilateral mesh design using conjugate direction field. *Association for Computing Machinery Transactions on Graphics*, 30(6):140.

[Loop, 1987] Loop, C. (1987). Smooth subdivision surfaces based on triangles. *Master's thesis, University of Utah, Department of Mathematics*.

[Lyon et al., 2019] Lyon, M., Campen, M., Bommes, D., and Kobbelt, L. (2019). Parametrization quantization with free boundaries for trimmed quad meshing. *Association for Computing Machinery Transactions on Graphics*.

[Maia Avelino et al., 2020] Maia Avelino, R., Oval, R., Liew, A., Van Mele, T., and Block, P. (2020). Topology exploration of compression-only networks for form and force optimisation. -. In preparation.

[Malek and Williams, 2013] Malek, S. and Williams, C. (2013). Structural implications of using cairo tiling and hexagons in gridshells. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2013*.

[Malek and Williams, 2017] Malek, S. and Williams, C. J. K. (2017). The equilibrium of corrugated plates and shells. *Nexus Network Journal*.

[Mäntylä, 1988] Mäntylä, M. (1988). *An introduction to solid modeling*. Computer science press.

[Marquis et al., 2017] Marquis, P., Cerosimo, A., and Douthe, C. (2017). Building elastic gridshells from patches. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2017*.

[Martin, 2015] Martin, A. G. (2015). A basketmaker's approach to structural morphology. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2015*.

[Masson, 2017] Masson, Y. (2017). *Existence et construction de réseaux de Chebyshev avec singularités et application aux gridshells*. PhD thesis, Université Paris-Est. Accessed on 01/02/2019.

[Maxwell, 1864] Maxwell, J. C. (1864). Xlv. on reciprocal figures and diagrams of forces. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 27(182):250–261.

[McCormack et al., 2004] McCormack, J. P., Cagan, J., and Vogel, C. M. (2004). Speaking the buick language: capturing, understanding, and exploring brand identity with shape grammars. *Design studies*, 25(1):1–29.

[Méndez Echenagucia et al., 2018] Méndez Echenagucia, T., Pigram, D., Liew, A., Van Mele, T., and Block, P. (2018). A cable-net and fabric formwork system for the construction of concrete shells: design, fabrication and construction of a full scale prototype. *Structures*.

[Mesnil, 2018] Mesnil, R. (2018). A re-parameterization approach for the construction of domes with planar facets. *Journal of the International Association for Shell and Spatial Structures*, 59(4):286–295.

[Mesnil et al., 2017a] Mesnil, R., Baverel, O., Douthe, C., Caron, J.-F., and Léger, B. (2017a). Structural morphology and performance of plated structures with planar quadrilateral facets. *Journal of the International Association for Shell and Spatial Structures*, 58(1).

[Mesnil et al., 2017b] Mesnil, R., Douthe, C., and Baverel, O. (2017b). Non-standard patterns for gridshell structures: Fabrication and structural optimization. *Journal of the International Association for Shell and Spatial Structures*, 58(4):277–286.

[Mesnil et al., 2018a] Mesnil, R., Douthe, C., Baverel, O., and Gobin, T. (2018a). Form finding of nexorades using the translations method. *Automation in Construction*, 95:142–154.

[Mesnil et al., 2017c] Mesnil, R., Douthe, C., Baverel, O., and Leger, B. (2017c). Generalised cyclidic nets for shape modelling in architecture. *International Journal of Architectural Computing*, 15(2):148–168.

[Mesnil et al., 2017d] Mesnil, R., Douthe, C., Baverel, O., and Léger, B. (2017d). Linear buckling of quadrangular and kagome gridshells: a comparative assessment. *Engineering Structures*, 132:337–348.

[Mesnil et al., 2017e] Mesnil, R., Douthe, C., Baverel, O., and Léger, B. (2017e). Marionette meshes: modelling free-form architecture with planar facets. *International Journal of Space Structures*, 32(3-4):184–198.

[Mesnil et al., 2018b] Mesnil, R., Douthe, C., Richter, C., and Baverel, O. (2018b). Fabrication-aware shape parametrisation for the structural optimisation of shell structures. *Engineering Structures*, 176:569–584.

[Meyer et al., 2003] Meyer, M., Desbrun, M., Schröder, P., and Barr, A. H. (2003). Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and mathematics III*, pages 35–57. Springer.

[Michell, 1904] Michell, A. G. M. (1904). Lviii. The limits of economy of material in frame-structures. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 8(47):589–597.

[Mitchell, 1991] Mitchell, W. J. (1991). Functional grammars: an introduction. In *Reality and Virtual Reality: Association for Computer Aided Design in Architecture Conference Proceedings 1991*, pages 167–176. University of California at Los Angeles.

[Motamedi et al., 2019] Motamedi, M., Oval, R., Carneau, P., and Olivier, B. (2019). Supportless 3d printing of shells: Adaptation of ancient vaulting techniques to digital fabrication. In *Impact: Design with all Senses: Design Modelling Symposium Berlin 2019*. Springer.

[Motro and Maurin, 2011] Motro, R. and Maurin, B. (2011). Bernard laffaille, nicolas esquillan, two french pioneers. In *IASS-IABSE Symposium: Taller, Longer, Lighter*.

[Mueller, 2014] Mueller, C. T. (2014). *Computational exploration of the structural design space*. PhD thesis, Massachusetts Institute of Technology. Accessed on 01/09/2018.

[Mueller and Ochsendorf, 2015] Mueller, C. T. and Ochsendorf, J. A. (2015). Combining structural performance and designer preferences in evolutionary design space exploration. *Automation in Construction*, 52:70–82.

[Nasri et al., 2009] Nasri, A., Sabin, M., and Yasseen, Z. (2009). Filling n-sided regions by quad meshes for subdivision surfaces. *Computer Graphics Forum*, 28(6):1644–1658.

[Navarro, 2001] Navarro, G. (2001). A guided tour to approximate string matching. *Association for Computing Machinery Computing Surveys*, 33(1):31–88.

[Nervi, 1965] Nervi, P. (1965). *Aesthetics and technology in building*. Charles Eliot Norton lectures. Harvard University Press.

[Nooshin, 1975] Nooshin, H. (1975). Algebraic representation and processing of structural configurations. *Computers & Structures*, 5(2-3):119–130.

[Nooshin, 2017] Nooshin, H. (2017). Formex configuration processing: A young branch of knowledge. *International Journal of Space Structures*, 32(3-4):136–148.

[Norman et al., 2009] Norman, A. D., Seffen, K. A., and Guest, S. D. (2009). Morphing of curved corrugated shells. *International Journal of Solids and Structures*, 46(7-8):1624–1633.

[Nsugbe and Williams, 2001] Nsugbe, E. and Williams, C. (2001). The generation of bone-like forms using analytic functions of a complex variable. *Engineering structures*, 23(1):22–28.

[Ohlbrock et al., 2017] Ohlbrock, P. O., D'acunto, P., Jasienski, J.-P., and Fivet, C. (2017). Constraint-driven design with combinatorial equilibrium modelling. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2017*.

[Otto et al., 1974] Otto, F., Schauer, E., Hennicke, J., and Hasegawa, T. (1974). *IL 10 Gitterschalen/Grid Shells: Bericht über das japanisch-deutsche Forschungsprojekt STI, durchgeführt von Mai 1971 bis Mai 1973*

*am Institut fur Leichte Flächentragwerke*. Seibu Construction Company / Institut für leichte Flächentragwerke / Krämer.

[Oval, 2017] Oval, R. (2017). compas_singular: a python framework for topology finding of singularities in patterns. Accessed on 12/05/2020.

[Pan et al., 2019] Pan, W., Sun, Y., Turrin, M., Louter, C., and Sariyildiz, I. (2019). Design exploration of architectural geometries and structural performance for sports arenas based on som-clustering and structural performance simulation. In *Structures and Architecture-Bridging the Gap and Crossing Borders: Proceedings of the Fourth International Conference on Structures and Architecture (ICSA 2019), July 24-26, 2019, Lisbon, Portugal*, page 342. CRC Press.

[Panozzo et al., 2013] Panozzo, D., Block, P., and Sorkine-Hornung, O. (2013). Designing unreinforced masonry models. *Association for Computing Machinery Transactions on Graphics*, 32(4):91.

[Pedro and Kobayashi, 2011] Pedro, H.-T. C. and Kobayashi, M. H. (2011). On a cellular division method for topology optimization. *International Journal for Numerical Methods in Engineering*, 88(11):1175–1197.

[Pele and Werman, 2010] Pele, O. and Werman, M. (2010). The quadratic-chi histogram distance family. In *European conference on computer vision*, pages 749–762. Springer.

[Pellegrino, 1993] Pellegrino, S. (1993). Structural computations with the singular value decomposition of the equilibrium matrix. *International Journal of Solids and Structures*, 30(21):3025–3035.

[Pellis and Pottmann, 2018] Pellis, D. and Pottmann, H. (2018). Aligning principal stress and curvature directions. In Hesselgren, L., Kilian, A., Sorkine-Hornung, O., Malek, S., Olsson, K.-G., and Williams, C. J. K., editors, *Advances in Architectural Geometry 2018*. Klein Publishing GmbH (Ltd.).

[Peng et al., 2014] Peng, C.-H., Barton, M., Jiang, C., and Wonka, P. (2014). Exploring quadrangulations. *Association for Computing in Machinery Transactions on Graphics*, 33(1):12.

[Peng et al., 2011] Peng, C.-H., Zhang, E., Kobayashi, Y., and Wonka, P. (2011). Connectivity editing for quadrilateral meshes. *Association for Computing Machinery Transactions on Graphics*, 30(6):141.

[Piker, 2013] Piker, D. (2013). Kangaroo: form finding with computational physics. *Architectural Design*, 83(2):136–137.

[Polya, 1924] Polya, G. (1924). Xii. über die analogie der kristallsymmetrie in der ebene. *Zeitschrift für Kristallographie-Crystalline Materials*, 60(1-6):278–282.

[Popescu et al., 2017] Popescu, M., Rippmann, M., Van Mele, T., and Block, P. (2017). Automated generation of knit patterns for non-developable surfaces. In *Humanizing Digital Reality: Design Modelling Symposium Paris 2017*, pages 271–284. Springer.

[Pottmann et al., 2007a] Pottmann, H., Asperl, A., Hofer, M., and Kilian, A. (2007a). *Architectural Geometry*. Bentley Institute Press.

[Pottmann et al., 2007b] Pottmann, H., Liu, Y., Wallner, J., Bobenko, A., and Wang, W. (2007b). Geometry of multi-layer freeform structures for architecture. *Association for Computing Machinery Transactions on Graphics*, 26(3):65.

[Powell, 1994] Powell, M. J. (1994). A direct search optimization method that models the objective and constraint functions by linear interpolation. In *Advances in optimization and numerical analysis*, pages 51–67. Springer.

[Powell, 1998] Powell, M. J. D. (1998). Direct search algorithms for optimization calculations. *Acta numerica*, 7:287–336.

[Prado et al., 2017] Prado, M., Dörstelmann, M., Solly, J., Menges, A., and Knippers, J. (2017). Elytra filament pavilion: Robotic filament winding for structural composite building systems.

[Preisinger, 2013] Preisinger, C. (2013). Linking structure and parametric geometry. *Architectural Design*, 83(2):110–113.

[Prishlyak and Mischenko, 2007] Prishlyak, A. O. and Mischenko, K. I. (2007). Classification of noncompact surfaces with boundary. *Methods of Functional Analysis and Topology*, 13(1):62–66.

[Prusinkiewicz and Lindenmayer, 2012] Prusinkiewicz, P. and Lindenmayer, A. (2012). *The algorithmic beauty of plants*. Springer Science & Business Media.

[Requicha, 1977] Requicha, A. (1977). Mathematical models of rigid solid objects. Technical report, University of Rochester. Technical memo 28.

[Richards, 1963] Richards, I. (1963). On the classification of noncompact surfaces. *Transactions of the American Mathematical Society*, 106(2):259–269.

[Rigby, 2003] Rigby, D. (2003). Topmaker: a technique for automatic multiblock topology generation using the medial axis. In *American Society of Mechanical Engineers/Japan Society of Mechanical Engineers 2003 4th Joint Fluids Summer Engineering Conference*, pages 1991–1997.

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

[Rippmann and Block, 2018] Rippmann, M. and Block, P. (2018). Computational tessellation of freeform, cut-stone vaults. *Nexus Network Journal*, 20(3):545–566.

[Rippmann et al., 2012] Rippmann, M., Lachauer, L., and Block, P. (2012). Interactive vault design. *International Journal of Space Structures*, 27(4):219–230.

[Rippmann et al., 2018] Rippmann, M., Liew, A., Van Mele, T., and Block, P. (2018). Design, fabrication and testing of discrete 3d sand-printed floor prototypes. *Materials Today Communications*, 15:254–259.

[Rippmann et al., 2016] Rippmann, M., Van Mele, T., Popescu, M., Augustynowicz, E., Méndez Echenagucia, T., Calvo Barentin, C., Frick, U., and Block, P. (2016). The Armadillo Vault: computational design and digital fabrication of a freeform stone shell. In Adriaenssens, S., Gramazio, F., Kohler, M., Menges, A., and Pauly, M., editors, *Advances in Architectural Geometry 2016*, pages 344–363. vdf Hochschulverlag an der ETH Zürich.

[Ronald et al., 1995] Ronald, S., Asenstorfer, J., and Vincent, M. (1995). Representational redundancy in evolutionary algorithms. In *Proceedings of 1995 IEEE International Conference on Evolutionary Computation*, volume 2, pages 631–636. IEEE.

[Rowland, 2019] Rowland, T. (2019). Compact surface. From MathWorld – a Wolfram web resource, created by eric w. weisstein. Accessed on 06/08/2019.

[Rozvany and Prager, 1976] Rozvany, G. I. N. and Prager, W. (1976). Optimal design of partially discretized grillages. *Journal of the Mechanics and Physics of Solids*, 24(2-3):125–136.

[Ruiz Gironès and Sarrate Ramos, 2007] Ruiz Gironès, E. and Sarrate Ramos, J. (2007). Automatic generation of quadrilateral structured meshes using linear programming and transfinite interpolation. In *6th Workshop on Numerical Methods in Applied Science and Engineering (NMASE 07), Vall de Nuria*.

[Saha et al., 2016] Saha, P. K., Borgefors, G., and di Baja, G. S. (2016). A survey on skeletonization algorithms and their applications. *Pattern Recognition Letters*, 76:3–12.

[Sakarovitch, 2009] Sakarovitch, J. (2009). Gaspard monge founder of "constructive geometry". In *Proceedings of the Third International Congress on Construction History*, pages 1293–1299.

[Saldana Ochoa et al., 2019] Saldana Ochoa, K., Ohlbrock, P. O., D'Acunto, P., and Moosavi, V. (2019). Beyond typology, beyond optimization. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2019*.

[Sass, 2006] Sass, L. (2006). A wood frame grammar: A generative system for digital fabrication. *International Journal of Architectural Computing*, 4(1):51–67.

[Schek, 1974] Schek, H.-J. (1974). The force density method for form finding and computation of general networks. *Computer Methods in Applied Mechanics and Engineering*, 3(1):115–134.

[Schiftner and Balzer, 2010] Schiftner, A. and Balzer, J. (2010). Statics-sensitive layout of planar quadrilateral meshes. In Ceccato, C., Hesselgren, L., Pauly, M., Pottmann, H., and Wallner, J., editors, *Proceedings of the Advances in Architectural Geometry 2010*, pages 221–236. Springer.

[Schiftner et al., 2012] Schiftner, A., Leduc, N., Bompas, P., Baldassini, N., and Eigensatz, M. (2012). Architectural geometry from research to practice: the eiffel tower pavilions. In Hesselgren, L., Sharma, S., Wallner, J., Baldassini, N., Bompas, P., and Raynaud, J., editors, *Advances in Architectural Geometry 2012*, pages 213–228. Springer.

[Schlaich and Schober, 2005] Schlaich, I. J. and Schober, I. H. (2005). Freeform glass roofs. In *Structures Congress 2005: Metropolis and Beyond*.

[Schlaich and Schafer, 1991] Schlaich, J. and Schafer, K. (1991). Design and detailing of structural concrete using strut-and-tie models. *Structural Engineer*, 69(6):113–125.

[Schlaich and Schober, 1997] Schlaich, J. and Schober, H. (1997). Glass roof for the Hippo House at the Berlin Zoo. *Structural Engineering International*, 7(4):252–254.

[Schlaich et al., 2005] Schlaich, J., Schober, H., and Kürschner, K. (2005). New trade fair in Milan – grid topology and structural behaviour of a free-formed glass-covered surface. *International Journal of Space Structures*, 20(1):1–14.

[Schlaich, 2018] Schlaich, M. (2018). Shell bridges-and a new specimen made of stainless steel. *Journal of the International Association for Shell and Spatial Structures*, 59(3):215–224.

[Schober, 2015] Schober, H. (2015). *Transparent shells: Form, topology, structure*. John Wiley & Sons.

[Schober and Justiz, 2012] Schober, H. and Justiz, S. (2012). Cabot Circus, Bristol - Ebene Viereckknetze für freigeformte Glasdächer. *Stahlbau*, 81(S1):28–42.

[Schrijver, 1998] Schrijver, A. (1998). *Theory of linear and integer programming*. John Wiley & Sons.

[Seifert and Threlfall, 1934] Seifert, H. and Threlfall, W. (1934). *Lehrbuch der topologie*. Chelsea.

[Shea and Cagan, 1997] Shea, K. and Cagan, J. (1997). Innovative dome design: applying geodesic patterns with shape annealing. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 11(5):379–394.

[Shea and Cagan, 1999] Shea, K. and Cagan, J. (1999). The design of novel roof trusses with shape annealing: assessing the ability of a computational method in aiding structural designers with varying design intent. *Design Studies*, 20(1):3–23.

[Shea et al., 1997] Shea, K., Cagan, J., and Fenves, S. J. (1997). A shape annealing approach to optimal truss design with dynamic grouping of members. *Journal of Mechanical Design*, 119(3):388–394.

[Sheffer and de Sturler, 2001] Sheffer, A. and de Sturler, E. (2001). Parameterization of faceted surfaces for meshing using angle-based flattening. *Engineering with computers*, 17(3):326–337.

[Shepherd and Pearson, 2013] Shepherd, P. and Pearson, W. (2013). Topology optimisation of algorithmically generated space frames. In *Proceedings of the Annual Symposium of the International Association for Shell and Spatial Structures 2013*.

[Shepherd and Richens, 2012] Shepherd, P. and Richens, P. (2012). The case for subdivision surfaces in building design. *Journal of the International Association for Shell and Spatial Structures*, 53(4):237–245.

[Sischka et al., 2001] Sischka, J., Brown, S., Handel, E., and Zenkner, G. (2001). Die Überdachung des Great Court im British Museum in London. *Stahlbau*, 70(7):492–502.

[Sonntag et al., 2017] Sonntag, D., Bechert, S., and Knippers, J. (2017). Biomimetic timber shells made of bending-active segments. *International Journal of Space Structures*, 32(3-4):149–159.

[Soriano et al., 2015] Soriano, E., Tornabell, P., Naicu, D. I., and Filz, G. H. (2015). Topologically based curvature in thin elastic shell networks. In Oñate, E., Bletzinger, K.-U., and Kröplin, B., editors, *Textiles composites and inflatable structures VII: proceedings of the VII International Conference on Textile Composites and Inflatable Structures, Barcelona, Spain. 19-21 October, 2015*, pages 167–176. International Centre for Numerical Methods in Engineering (CIMNE).

[Spadea et al., 2017] Spadea, S., Orr, J., Nanni, A., and Yang, Y. (2017). Wound frp shear reinforcement for concrete structures. *Journal of Composites for Construction*, 21(5):04017026.

[Srinivasan, 2004] Srinivasan, V. (2004). *Modeling high-genus surfaces*. PhD thesis, Texas A&M University. Accessed on 22/10/2018.

[Stanford et al., 2013] Stanford, B., Beran, P., and Kobayashi, M. H. (2013). Simultaneous topology optimization of membrane wings and their compliant flapping mechanisms. *AIAA journal*, 51(6):1431–1441.

[Stephan et al., 2004] Stephan, S., Sánchez-Alvarez, J., and Knebel, K. (2004). Reticulated structures on free-form surfaces. *Stahlbau*, 73(8):562–572.

[Stiny, 2006] Stiny, G. (2006). *Shape: talking about seeing and doing.* MIT Press.

[Stiny and Gips, 1971] Stiny, G. and Gips, J. (1971). Shape grammars and the generative specification of painting and sculpture. In *Proceedings of the Congress International Federation for Information Processing 1971*, pages 1460–1465.

[Stiny and Mitchell, 1978] Stiny, G. and Mitchell, W. J. (1978). The palladian grammar. *Environment and planning B: Planning and design*, 5(1):5–18.

[Stitic and Weinand, 2015] Stitic, A. and Weinand, Y. (2015). Timber folded plate structures–topological and structural considerations. *International Journal of Space Structures*, 30(2):169–177.

[Suris and Bobenko, 2008] Suris, Y. B. and Bobenko, A. I. (2008). Discrete differential geometry: Integrable structure. *American Mathematical Society.*

[Suzuki and Knippers, 2017] Suzuki, S. and Knippers, J. (2017). The design implications of form-finding with dynamic topologies. In *Humanizing Digital Reality: Design Modelling Symposium Paris 2017*, pages 211–223. Springer.

[Tachi, 2009] Tachi, T. (2009). Generalization of rigid-foldable quadrilateral-mesh origami. *Journal of the International Association for Shell and Spatial Structures*, 50(3):173–179.

[Takayama et al., 2014] Takayama, K., Panozzo, D., and Sorkine-Hornung, O. (2014). Pattern-based quadrangulation for N-sided patches. In *Proceedings of the Symposium on Geometry Processing 2014*, pages 177–184. Eurographics Association.

[Takezawa et al., 2016] Takezawa, M., Imai, T., Shida, K., and Maekawa, T. (2016). Fabrication of freeform objects by principal strips. *Association for Computing in Machinery Transactions on Graphics*, 35(6):225.

[Tarini et al., 2010] Tarini, M., Pietroni, N., Cignoni, P., Panozzo, D., and Puppo, E. (2010). Practical quad mesh simplification. *Computer Graphics Forum*, 29(2):407–418.

[Turrin et al., 2011] Turrin, M., Von Buelow, P., and Stouffs, R. (2011). Design explorations of performance driven geometry in architectural design using parametric modeling and genetic algorithms. *Advanced Engineering Informatics*, 25(4):656–675.

[Van Laarhoven and Aarts, 1987] Van Laarhoven, P. J. and Aarts, E. H. (1987). *Simulated annealing: Theory and applications*. Springer.

[Van Mele and Block, 2014] Van Mele, T. and Block, P. (2014). Algebraic graph statics. *Computer-Aided Design*, 53:104–116.

[Van Mele et al., 2017] Van Mele, T., Liew, A., Méndez Echenagucia, T., Rippmann, M., et al. (2017). compas: A framework for computational research in architecture and structures. Accessed on 01/09/2018.

[Van Mele et al., 2014] Van Mele, T., Panozzo, D., Sorkine-Hornung, O., and Block, P. (2014). Best-fit thrust network analysis – Rationalisation of freeform meshes. In Adriaenssens, S., Block, P., Veenendaal, D., and Williams, C. J. K., editors, *Shell Structures for Architecture: Form Finding and Optimization*, chapter 13, pages 157–169. Routledge, London.

[Venuti and Bruno, 2018] Venuti, F. and Bruno, L. (2018). Influence of in-plane and out-of-plane stiffness on the stability of free-edge gridshells: A parametric analysis. *Thin-Walled Structures*, 131:755–768.

[Wagner and Fischer, 1974] Wagner, R. A. and Fischer, M. J. (1974). The string-to-string correction problem. *Journal of the Association for Computing Machinery*, 21(1):168–173.

[Wallner and Pottmann, 2011] Wallner, J. and Pottmann, H. (2011). Geometric computing for freeform architecture. *Journal of Mathematics in Industry*, 1(1):4.

[Welsh and Powell, 1967] Welsh, D. J. and Powell, M. B. (1967). An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1):85–86.

[Williams, 2001] Williams, C. J. K. (2001). The analytic and numerical definition of the geometry of the British Museum Great Court Roof. In *Mathematics & Design*, pages 434–440. Deakin University.

[Williams, 2011] Williams, C. J. K. (2011). Patterns on a surface: the reconciliation of the circle and the square. *Nexus Network Journal*, 13(2):281–295.

[Winslow et al., 2010] Winslow, P., Pellegrino, S., and Sharma, S. (2010). Multi-objective optimization of free-form grid structures. *Structural and multidisciplinary optimization*, 40(1-6):257.

[Yamaguchi, 1997] Yamaguchi, Y. (1997). Classification of boundary representations for manifold and non-manifold topology. In *Product Modeling for Computer Integrated Design and Manufacture*, pages 104–115. Springer.

[Yamaguchi and Kimura, 1995] Yamaguchi, Y. and Kimura, F. (1995). Nonmanifold topology based on coupling entities. *IEEE Computer Graphics and Applications*, 15(1):42–50.

[Zadravec et al., 2010] Zadravec, M., Schiftner, A., and Wallner, J. (2010). Designing quad-dominant meshes with planar faces. *Computer Graphics Forum*, 29(5):1671–1679.

## About the author

# Robin Oval

| | |
|---|---|
| 2019 – | **University of Cambridge, United Kingdom** |
| 2016 – 2019 | **Université Paris-Est, France**<br>*Doctor of Philosophy* |
| 2016 | **ETH Zurich, Switzerland** |
| 2014 – 2015 | **Bollinger + Grohmann, Germany** |
| 2013 – 2014 | **École d'Architecture de Marne, France** |
| 2012 – 2016 | **École des Ponts & Chaussées, France**<br>*Diplôme d'ingénieur* |
| 2010 – 2012 | **Lycée Saint Louis, France** |