Digital and automatic design of free-form single-layer grid structures

Zhi Li[1], Jun Ye[2,*], Boqing Gao [1], Qisheng Wang [1], Guan Quan[1], Paul Shepherd[3]

1. College of Civil Engineering and Architecture, Zhejiang University, Hangzhou 310058, China.
2. Department of Civil & Environmental Engineering, University of Strathclyde, Glasgow, G1 1XJ, UK
3. Department of Architecture & Civil Engineering, University of Bath, Bath, BA2 7AY, UK
**Corresponding author: Assistant Professor, jun.ye@strath.ac.uk;*

Abstract: Grid shells have been widely used in various long-span public buildings, and many of them are defined over free-form surfaces with complex boundaries. This emphasizes the importance of general and digitalised grid generation and optimization methods in the initial design stage to achieve visually sound grid shells. In this paper, a framework is presented for the development of a digital tool and to generate regular and fluent grids for structural design over free-form surfaces, especially those with complex boundaries. Both triangular and quadrilateral grid generation are addressed. To generate regular and fluent grids for free-form surfaces, a simple yet practical framework is proposed based on a spring-mass model. Firstly, an initial casual quadrilateral grid is tiled on the surface based on surface discretization and mesh parameterization. Secondly, the distribution of the initial grid vertices is adjusted by a dynamic relaxation procedure, assuming the grid as a spring-mass system. Thirdly, the grid vertices corresponding to the adjusted particles in the equilibrium state are then reconnected to produce a grid with a predefined pattern (triangular or quadrilateral). Finally, the generated grid is relaxed with the spring-mass model, alongside additional geometric operations including grid size adjustment and filtering techniques, to further improve the grid regularity and fluency. As part of its contribution, this paper also broadens the application scope of the fluency index, which can be used to quantitatively evaluate the suitability of a given triangular or quadrilateral grid for architectural and structural expression. Examples are presented and show that the proposed framework is effective for the triangular and quadrilateral grid generation over various surfaces and to optimize the resulted grids along complex boundaries. The method proposed can be useful for rapid design and performance evaluation of free-form grid structures.

Key words: Free-form surface; Grid structure; Parametric design; Dynamic relaxation; Grid quality.

1. Introduction

Grid structures as long span roof shells are often one of the most striking parts of a building in terms of structural efficiency or architectural appearance. Grid structures with simple shapes such as cylinder, sphere, and paraboloid have been widely applied in design practice, where designers often use analytical equations to determine the positions and connections of joints for structural design [1,2]. With the introduction of digital Computer-aided Design applications, designers can model almost any continuum shape (curves, surface, or volume) imaginable. Some buildings with fantastic and inspired shapes have been successfully erected in recent years, such as British Museum [3] with triangular grid cells and Chadstone Shopping Centre with quadrilateral grid cells, as presented in Fig. 1. To transform an architectural model with a free-form yet continuum surface into a real building, thin-walled, efficient grid structures may be the best choice due to their potentials for material reduction and internal space increase.

(a) Triangular grids on British Museum Great Court roof, London, UK

(b) Quadrilateral grids on Chadstone Shopping Centre roof, Melbourne, Australia

Fig. 1 Free-form structures in engineering practice with triangular and quadrilateral grids (Photographed by the authors).

To design a grid structure, grid generation is a vital step. However, it is always not easy to generate a grid that meets the requirements of designers, especially when the surface has complex boundaries. Designers often require the grid to be of fluent grid lines and regular grid cells to achieve visually sound architectures. In terms of fluency, each continuous member should fluidly pass over the surface and avoid singular vertices. As shown in Fig. 2(a-c), the structured triangular grid of which the internal vertices all have the same number of adjacent cells automatically forms continuous lines-sets in three different directions (green, blue, red lines in Fig. 2(a-c)). The three-directional lines-sets are distributed over the whole design domain and exhibit nearly little bending. This arrangement of grid lines enables the grid structure to be visually sensible and fluent. However, the singular vertices in fig. 2(d) interrupt the continuity of the grid lines, and some lines-sets are severely bent, such as the two lines-set formed by black and gray lines in Fig. 2(d). The fluency of the structured triangular grid has been defined as the overall bending degree of the lines-set in [4]. In terms of regularity, the narrow grid cell results in a small angle between two adjacent bars, which will bring difficulties to construction; therefore, grid cells should all be well-shaped and avoid distortion to ease connections. Detailed quantitative quality metrics of fluency and regularity will be illustrated in Section 8 of this paper.

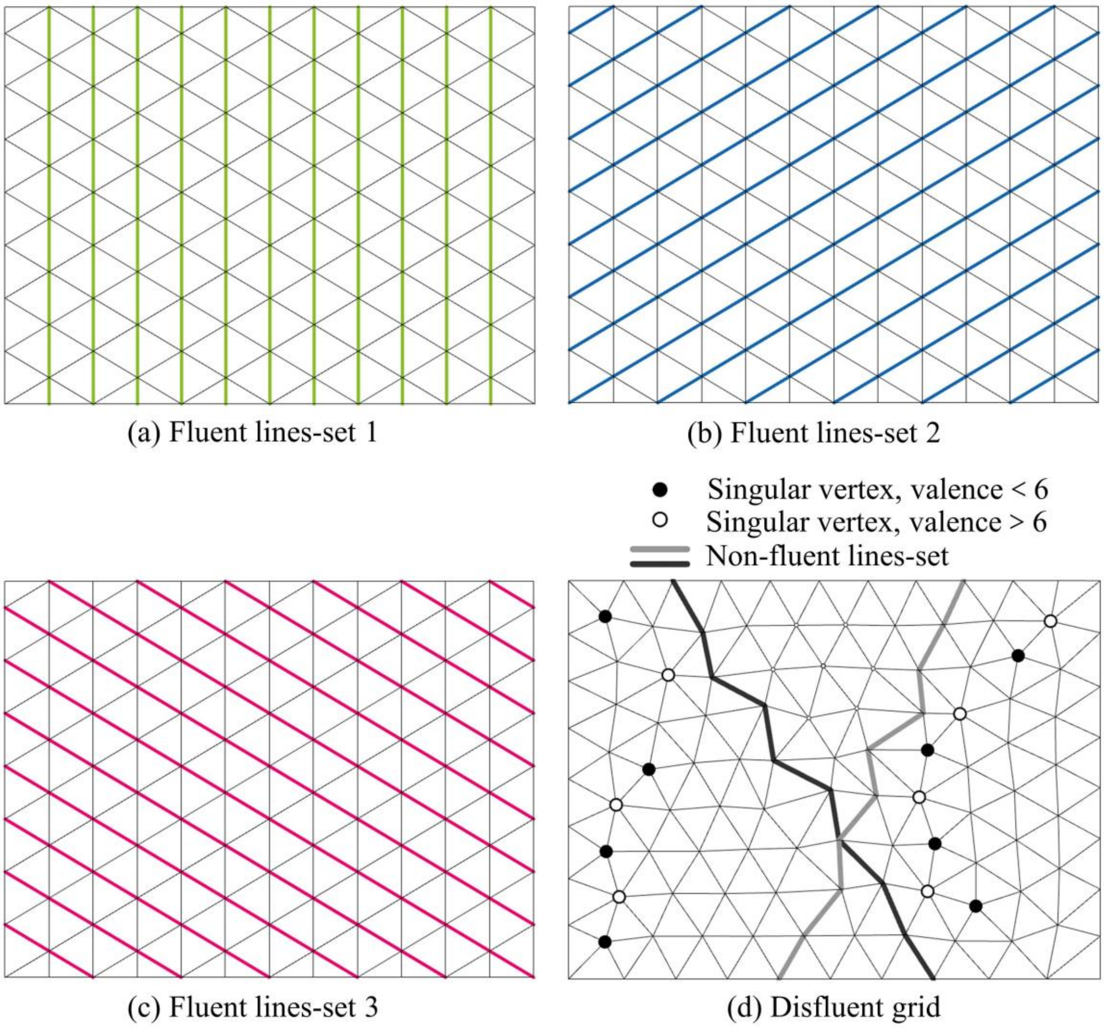


Fig. 2 Grid with curve fluidity and disfluent grid in a shell structure.

Existing instances of grid patterns on free-form shells have generally been generated manually using computer-aided design tools or automatically using purpose-built programming scripts that are tailored to each project. As such structures are becoming more popular and more challenging, particularly with complex internal/external boundaries and varying curvatures, perhaps more practical tools that can efficiently generate a structured grid that meets the requirements of designers on a given free-form surface are needed to facilitate the design process, especially in the early stage.

Most of the previous research has addressed grid generation methodologies for visualization or finite element analysis purposes. Owen [5] presented an excellent literature review of unstructured grid generation technologies in finite element analysis, including the Delaunay triangulation [6], the Advancing Front Technique [7], the Mapping Method [8,9], and their combinations. In finite element analysis, grid generation is concerned with the trade-off between calculation accuracy and speed. Grid generation in the field of architecture pays more attention to the aesthetic aspect of the grid, which is typically reflected by regular grid cells and fluent grid lines. As a result, the traditional grid generation approach cannot be used directly to build a free-form grid.

To generate efficient and practical grids for free-form structures, Winslow et al. [10] presented a design tool for the optimization of grid structures using structural performance as the objective. They employed a multi-objective genetic algorithm to vary rod directions over the surface, and a process of grid homogenization was used to calculate the mechanical performance of discretized grid structures that are composed of repeating grid cells. However, free-form surfaces with complicated boundaries were not considered. Su et al. [11] proposed an improved advancing front technique using the main stress trajectories to arrange rod directions of the grids. Although the grids generated by this method have better structural performance for a single load case, the fluency of the grid is bad, and some distorted grid cells exist.

To generate a more aesthetic grid, many studies have reported the grid generation progress over a free-form surface. Shepherd and Richens [12] employed an improved method based on the technique of surface subdivision to generate grids for structural design. A very coarse control grid with only a small number of vertices is first sketched over the given surface, and then the coarse grid is subdivided over a number of iterations to fit the original surface. Similarly, EvoluteTools [13], a plugin in Rhino, first generates a grid with a few large-size cells and then subdivides these cells at the same level to obtain the final grid. Because the subdivision operation generates structured and well-shaped sub-grids within each coarse cell, these two methods [12,13] can improve the fluency and regularity of the grid. However, their main limitation is that designing appropriate coarse grids manually over complex free-form surfaces is extremely difficult. Based on the concept of "guide line" by Gao et al. [14], a "guide line method" was developed to generate grids with rods of balanced length and fluent lines. The process began with a number of guide-curves on the surface, which determine the directions of the "rods" of the grid. The generated grids were demonstrated to have similar rod lengths. Gao et al. [15,16] improved the "guide line method" by incorporating a surface flattening technique, which reduced the grid shape irregularity by up to 47%. However, these strategies [14–16] did not succeed in improving grid fluency and eliminating the small rod members near the complex boundaries. Most recently, Oval et al. [17] proposed a feature-based topology finding of patterns for shell structures. The method is based on a generation procedure for singularity meshes as a start, followed by the boundaries of a surface as well as point and curve features, using a topological skeleton or medial axis. Despite the fact that the designed patterns are highly structured, a small number of singular vertices remain.

Many researchers have also utilized some force-based methods to generate grids. Shimada et al. [18] introduced a bubble-like approach for meshing trimmed parametric surfaces for finite element analysis. They viewed nodes as the centers of packed bubbles and optimized their placement iteratively by solving the bubbles' force equilibrium.

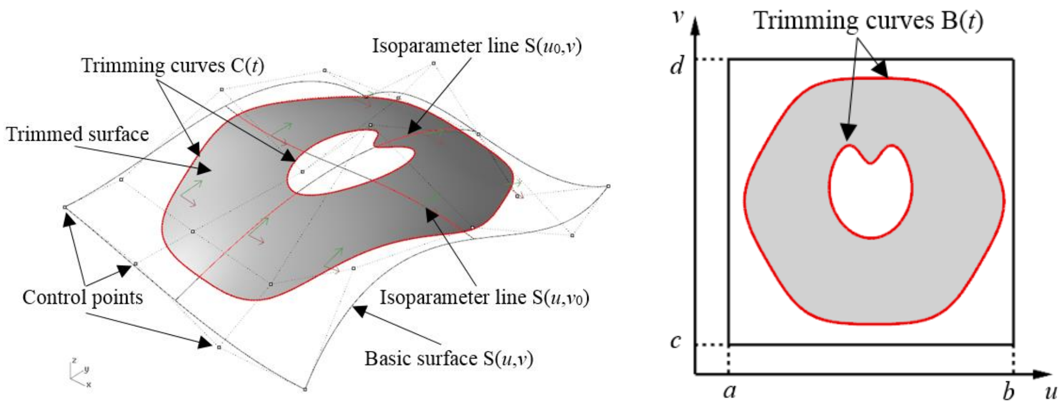
Similarly, Zheleznyakova et al. [19,20] presented a molecular approach for generating finite element meshes. In this method, nodes were treated as charged interacting particles that could be moved to ideal places using molecular dynamics in a NURBS surface parametric design domain. The Delaunay triangulation technique was used to connect these particles into well-shaped triangles. In a similar fashion to the molecular method, Wang et al. [21] suggested a grid generation strategy based on a mapping technique and a truss-like method. Delaunay triangulation was used to construct a planar triangular grid, which was then optimized by treating the grid as a truss in the parametric domain. The planar grids were then mapped back to the original surface to create a more regular grid. Combining the bubble method with edge operations, Wang et al. [4] developed a framework for triangular grid generation, resulting in highly structured grids. While these force-based methods [4, 18-21] have been developed and employed for grid generation, the primary goals have been grid uniformity and regularity, with a dearth of effective actions to account for grid fluency.

In addition to the above methods, topology optimization can also be regarded as an effective method to generate grids over free-form surfaces. Topology optimization is a mathematical method for optimizing the layout of materials inside a given design domain, under specified loads, boundary conditions, and constraints, with the purpose of maximizing the system performance [22]. Some scholars have made substantial progress in this field. For example, Wang et al. [23] proposed a method that combined the multiscale isogeometric topology optimization method and the progressive homogenization method, and applied it to the design of periodic lattice material structures.. Numerical examples show the advantages of this method in calculation accuracy, efficiency, and convergence. Zhang et al. [24] proposed a topology optimization method using B-spline curves to describe the geometry of the holes in the structure and developed the corresponding numerical solution technology, which achieved good results in topology optimization problems considering geometric features. Based on the Hamilton-Jacobi equation, Park and Youn [25] proposed the AIF (Adaptive Inner-Front) level set method and used the linear quadrilateral shell element based on Reissner-Mindlin theory to realize the application of the level set topology optimization method in the hyperbolic shell structure. Kang and Youn [26] proposed the TSA (Trimmed Surface Analysis) method, using topological derivatives as the criteria for judging the formation of new holes and determining the location of the holes, and obtained results with smooth boundaries and robust convergence. However, topology optimization does not take architectural aesthetics into account, and the generated grid is so coarse that it often destroys the shape of the original surface. Furthermore, the topology optimization results in complex configurations that are difficult to produce industrially.

All the studies reviewed so far, however, suffer from the fact that they fail to address grid regularity and fluency of free-form surfaces with complex boundaries. This paper is, therefore, innovative by developing a framework to generate a regular and fluent grid over a free-form surface based on the physical analogy between the grid and a spring-mass model. The framework can also deal with the surface with complex boundaries. The classical spring-mass method [27] has been widely employed in the form-finding of spatial structures since the 1980s, but it is demonstrated that the method can be used to solve grid design problems in free-form grid roofs with a simple initial grid. Compared with the classical spring-mass method, this paper designs different kinds of springs, such as grid edge spring, three types of constraint spring (surface constraint spring, curve constraint spring, and point constraint spring), especially the face edge spring and visual spring, to generate required grids. The framework starts with tiling an initial casual quadrilateral grid on the surface based on surface discretization and mesh parameterization to obtain a list of nodes spanning the design domain. A force-based spring-mass model is then applied to uniformize node distribution. After that, the grids are generated by connecting the resulted grid nodes according to a user-defined rule. With additional operations included, like grid size adjustment and grid filtering, a regular and fluent grid will be obtained. The proposed framework can be robustly applied to generate triangular or quadrilateral grids over various surfaces, possessing excellent adaptability effectiveness. Case studies are presented to demonstrate the effectiveness of the proposed framework. The resultant grids are proved to be of regular cells and fluent lines, thereby satisfying aesthetic demands. The resulted geometry of the grid shell can be further used in finite element analysis for the design of ultimate limit state analysis, local/global stability analysis, or serviceability limited state analysis. However, such research has been observed in many publications or engineering practices, which is not the focus of this paper.

2. Representation of surfaces and grids

Many mathematical models for the representation of surfaces have been proposed. However, free-form surfaces are mostly modelled by Non-Uniform Rational B-Splines (NURBS) technique [28]. NURBS technique realizes a free-form surface by control points and knot weights and guarantees positional accuracy with small data. A NURBS surface is a bivariate vector-valued piecewise rational function and thus establishes a mapping relation between the 3D domain and the parametric domain. A trimmed surface is composed of a basic surface (untrimmed) and trimming curves. Fig. 3 illustrates the composition of a trimmed NURBS surface.



(a) In 3D domain (b) In parametric space

Fig. 3 Trimmed NURBS surface.

A multiple surface consists of more than one closely adjacent trimmed or untrimmed NURBS surfaces. The boundary curves of its member surfaces which make up the boundary of the multiple surface are named naked boundary curves, while other boundary curves in the interior of the multiple surface are interior boundary curves. For example, the multiple NURBS surface shown in Fig. 4 consists of four member surfaces, and its boundary curve is fitted from seven naked boundary curves.

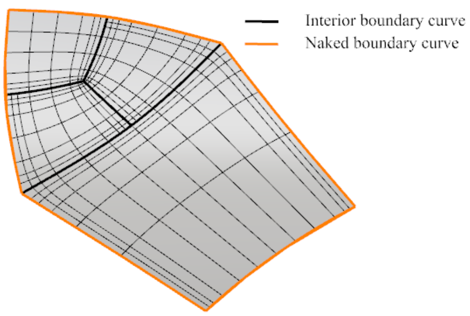


Fig. 4 Multiple NURBS surface.

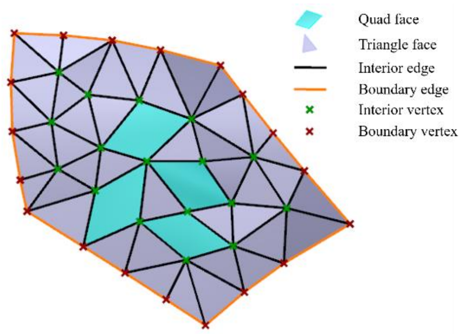


Fig. 5 Simple grid.

Discretized forms of surfaces (meshes/grids) are composed of a number of connected vertices, edges, and faces, as shown in Fig. 5. An edge (the connection between a pair of vertices) that forms only one face is defined as a boundary edge, whilst an edge that defines two faces is an interior edge. The endpoints of boundary edges are boundary vertices, whilst other endpoints are all interior vertices.

3. Overview of the framework

The procedure of the grid generation framework is summarized in Fig. 6. The steps are as follows:

(1) A free-form surface defined by an architect for a structural engineer is input to create a grid.

(2) Based on surface discretization and mesh parameterization, an initial simple quadrilateral grid over the surface is obtained.

(3) The distribution of the initial grid vertices is adjusted by a dynamic simulation procedure, assuming the grid as a spring-mass system.

(4) The adjusted grid vertices are reconnected to produce a grid with a predefined pattern (triangular or quadrilateral).

(5) The grid size control method can be optionally used to make the grid edges varied and adaptive to boundary curves.

(6) For the surface with internal or external boundaries, filtering techniques and dynamic simulation are employed to further improve the grid regularity and fluency.

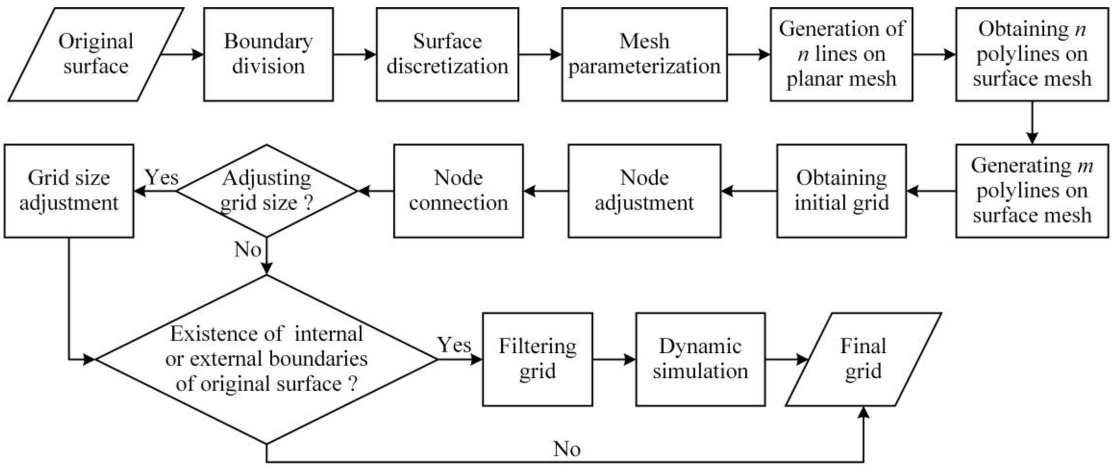


Fig. 6 Flow chart of the grid generation framework.

4. Force-based design model

The spring-mass model is commonly used for the simulation of dynamic systems in computer graphics due to its simplicity for implementation and relatively high computational efficiency [29,30]. In this paper, a grid is generated first, and then an algorithm is used to relax the generated grid. The algorithm is based on the physical analogy between the generated grid and the spring-mass model. In the physical analogy process, each grid vertex corresponds to a particle with a lumped mass, and each grid edge corresponds to an elastic spring with a stiffness. Besides the grid edge springs, some other types of springs are added to connect or restrict particles according to the design requirements of the grid, such as preventing particles from moving out of the surface so that the grids can approximate the given surface. With the slack length of a spring defined by the designer, unbalanced forces may develop due to the unequal length of the springs. The unbalanced forces express the difference between the current grid and the desired grid in a sense. With reasonable forces defined, a high-quality grid will be obtained by solving the equilibrium state of the spring-mass system. And with changes in forces, different grid characteristics can be achieved.

4.1. The spring-mass analogy for various edges

As stated above, the particles with a lumped mass are connected by analogically defined physical springs with a stiffness in the spring-mass model. Each spring has a force-displacement relationship which depends on its current length and its original length. A force-displacement function for linear elastic spring is used as the basic function in this paper, as shown in Eq. (1) and Fig. 7. According to their different roles, the springs involved in the model are divided into three types: grid edge springs, face edge springs, and constraint springs. The constraint springs can be subdivided into surface constraint springs, curve constraint springs and point constraint springs according to the type of the source object of the constraint. Each spring type will be introduced in the following sections. The general force to displacement relationship can be defined as:

$$\vec{T} = k(e - |\vec{d}|) \cdot \frac{\vec{d}}{|\vec{d}|}, \tag{1}$$

where $\vec{T}$ is the spring force; $\vec{d}$ is the length vector of the spring. k is the elastic coefficient, and e is the slack length of the spring.

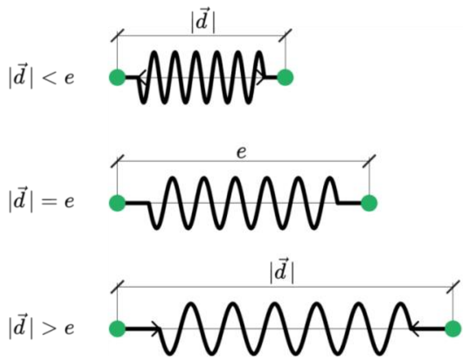


Fig. 7 Schematic diagram of spring force.

4.1.1. Grid edge spring

As shown in Fig. 8, each grid edge corresponds to an elastic spring, called grid edge spring. The grid edge spring is to maintain the overall uniformity of grid size without changing the topology of the grid. To achieve this goal, all grid edge springs have the same original length proportional to the average value of all current edge lengths $\bar{l}$:

$$e_{\mathrm{g}} = f_{\mathrm{g}} \bar{l}, \tag{2}$$

where f_{g} is a parameter that is smaller than 1.0 (a good default is 0.8). As $\bar{l}$ will only change slightly during the dynamic simulation, e_{g} can be set as a constant less than $\bar{l}$ for simplicity. Thus, based on Eq. (1), the grid edge spring force is:

$$\vec{T}_{\mathrm{g},ij} = k_{\mathrm{g}}(e_{\mathrm{g}} - |\vec{d}_{ij}|) \cdot \frac{\vec{d}_{ij}}{|\vec{d}_{ij}|}, \tag{3}$$

where k_{g} is the elastic coefficient of all grid edge springs.

The resultant force of grid edge spring forces on p_i is calculated by:

$$\vec{T}_{\mathrm{g},i} = \sum_{j \in J_{\mathrm{g}}} \vec{T}_{\mathrm{g},ij}, \tag{4}$$

where J_{g} is the set of particles connected to p_i.

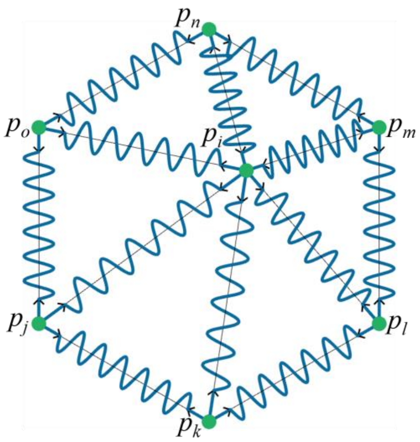


Fig. 8 Grid edge spring.

4.1.2. Face edge spring

Considering an individual face of a grid, the edges of the face are also regarded as springs, called face edge springs. The face edge spring aims to locally adjust each grid cell, thereby further improving the quality of the grid. The force of a face edge spring is defined as:

$$\vec{T}_{\mathrm{f},ij} = k_{\mathrm{f}}(e_{\mathrm{f}} - |\vec{d}_{ij}|) \cdot \frac{\vec{d}_{ij}}{|\vec{d}_{ij}|}, \tag{5}$$

where k_{f} is the elastic coefficient of all face edge springs; e_{f} is the average value of the current edge lengths of the face; $\vec{d}_{ij}$ is the displacement vector from the i-th particle to j-th particle.

A triangular face with equal-length sides is a regular triangle, while a quadrilateral face with equal-length sides is a rhombus. Since a flat rhombus is not regular, visual springs that correspond to the diagonal lines of the quadrangle are added to regulate the quadrilateral face. The force of each visual spring is:

$$\vec{T}_{\mathrm{v},ij} = k_{\mathrm{v}}(e_{\mathrm{v}} - |\vec{d_{ij}}|) \cdot \frac{\vec{d}_{ij}}{|\vec{d}_{ij}|}, \tag{6}$$

where k_{v} is the elastic coefficient of all visual springs; e_{v} is the average value of the current edge lengths of the two diagonal lines.

Fig. 9 shows the face edge spring of the triangular grid, and Fig. 10 illustrates the face edge spring and visual spring of the quadrilateral grid. It should be noted that, although both face edge spring and grid edge spring correspond to grid edges, there are differences between a grid edge spring and a face edge spring. With the purpose of obtaining the uniformly distributed grid nodes, the slack length of the grid edge spring is the average of the lengths of all grid edges, and each grid edge corresponds to one grid edge spring. However, the slack length of the face edge spring is defined as the average of the edge lengths of the single grid cell, with the corresponding edge of the face edge spring belonging to this grid cell. Thus, for an interior grid edge that belongs to two grid cells, there are two corresponding face edge springs, and a boundary edge has one corresponding face edge spring. The face edge spring is defined to locally adjust the regularity of the grid cell.

The resultant force of face edge spring forces on a vertex in a face, called face force, is calculated by:

$$\vec{T}_{\mathrm{f},i} = \sum_{j \in J_{\mathrm{f}}} \vec{T}_{\mathrm{f},ij}, \tag{7}$$

where J_{f} is the set of particles connected to p_i by face edges or visual edges in the face.

A grid vertex belongs to more than one face, so the resultant force of face forces on a vertex is

$$\vec{T}_{\mathrm{F},i}=\sum_{\mathrm{F}}\vec{T}_{\mathrm{f},i}\ , \tag{8}$$

where F is the set of the faces to which p_i belongs.

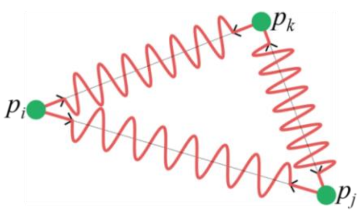


Fig. 9 Face edge spring of triangular grid.

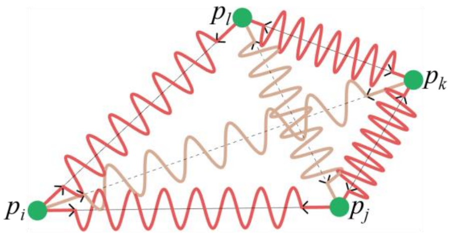


Fig. 10 Face edge spring and visual spring of quadrilateral grid.

4.1.3. Constraint spring

The continuum surface is usually predefined by architects, and the generated grids should approximate the original shape. To achieve this, the grid vertices need to be located on the surface, and the boundary vertices need to be located on the boundary curves. Therefore, it is necessary to define the corresponding constraint spring to constrain the position of the grid vertices.

To keep the particles (correspond to the grid vertices) on the surface, each particle is connected to the target surface by a spring, defined as a surface constraint spring. As shown in Fig. 11, the endpoints of the surface constraint spring are composed of a grid vertex and the point closest to the grid vertex on the surface. The force of the surface constraint spring can be:

$$\vec{P}_{\mathrm{s},i}=k_{\mathrm{s}}\cdot\vec{d}_{\mathrm{s},i} \tag{9}$$

where k_{s} is the elastic coefficient of the surface constraint spring; $\vec{d}_{\mathrm{s},i}$ is the displacement vector from p_i to its closest point on the surface.

As to the boundary particles (correspond to the boundary vertices), there are curve constraint springs to constrain the boundary particles on the boundary curves, as shown in Fig. 11. One end of the curve constraint spring is a boundary vertex, and the other end is the point closest to the boundary vertex on the boundary curve. The force of the curve constraint spring is:

$$\vec{P}_{\mathrm{c},i} = \begin{cases} k_c \cdot \vec{d}_{\mathrm{c},i}, & p_i \in P_{\mathrm{bou}} \\ 0, & p_i \in P_{\mathrm{int}} \end{cases}, \tag{10}$$

where k_{c} is the elastic coefficient of the curve constraint spring; $\vec{d}_{\mathrm{c},i}$ is the displacement vector from p_i to its closest point on the curve; P_{bou}, P_{int} are the sets of the boundary vertices and the set of the interior vertices, respectively.

Due to the demands of the load path or the surface modelling, some special positions on the surface are supposed to be fixed bearings in the grid structure, called anchoring points. To position the grid vertex at the anchoring point, the particle (correspond to the grid vertex) closest to an anchoring point or the specified particle is connected to the anchoring point by a point constraint spring (Fig. 11), subjected to the anchoring force of:

$$\vec{P}_{\mathrm{p},i} = \begin{cases} k_{\mathrm{p}} \cdot \vec{d}_{\mathrm{p},i}, & p_i \in P_{\mathrm{fixed}} \\ 0, & p_i \in P_{\mathrm{free}} \end{cases}, \tag{11}$$

where k_{p} is the elastic coefficient of the point constraint spring; $\vec{d}_{\mathrm{p},i}$ is the displacement vector from p_i to its corresponding anchoring point; P_{fixed} is the set of fixed vertices, and P_{free} is the set of other vertices.

The coefficients k_{s}, k_{c}, and k_{p} control the constraint intensity from the surface, the boundary, and the fixed points. They are much larger than the coefficients k_{g}, k_{f}, and k_{v}.

If the constraint is strict, the corresponding k is required to be infinite, which can be achieved by projecting points onto its relevant geometric object.

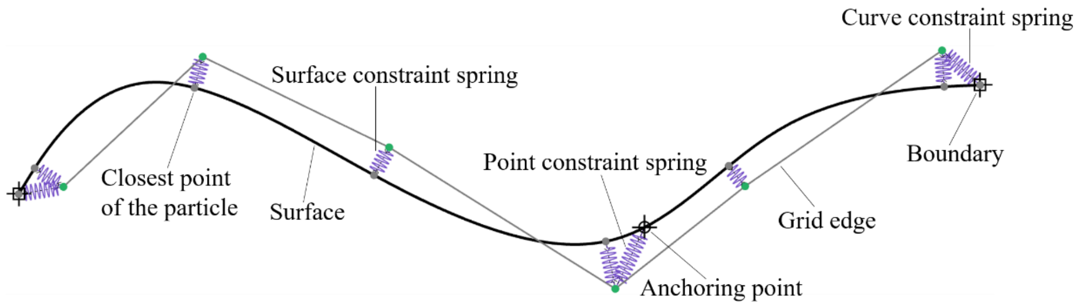


Fig. 11 Constraint spring.

4.2. Resultant force

As mentioned above, all the forces acting on a particle come from interactions with other particles, from external geometric objects like the surface, the boundary curve, and the fixed point. An interior edge of the grid corresponds to one grid edge spring and two face edge springs, while a boundary edge of the grid corresponds to one grid edge spring and one face edge spring. As shown in Fig. 12, these springs and particles make up the spring-mass system for the grid. In general, the forces of grid edge springs are to uniformize the edge length, the forces of face edge springs are to regularize the shape of grid faces, and three kinds of constraining forces are constraint conditions. In brief, the grid is like a tensile spring net stretched by the boundary and attached to the surface. Besides, each particle experiences a drag force which dissipates the potential energy of the system gradually:

$$\vec{f}_i = -k_{\mathrm{ve}} \cdot \vec{v}_i \, , \tag{12}$$

where k_{ve} is the resistance coefficient; $\vec{v}_i$ is the velocity.

Finally, in the spring-mass system, the resultant force acting on the i-th particle is computed by:

$$\overrightarrow{T}_i = \overrightarrow{T}_{\mathrm{g},i} + \overrightarrow{T}_{\mathrm{F},i} + \overrightarrow{P}_{\mathrm{s},i} + \overrightarrow{P}_{\mathrm{c},i} + \overrightarrow{P}_{\mathrm{p},i} + \overrightarrow{f}_i , \tag{13}$$

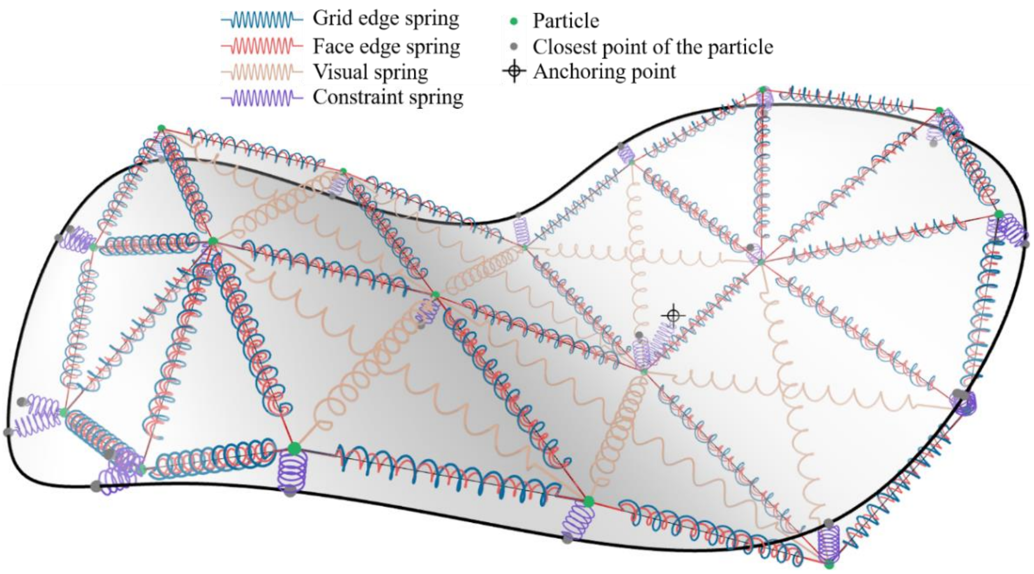


Fig. 12 The spring-mass system of a grid.

4.3. Dynamic simulation

The definitions of the spring forces in the spring-mass model are all based on the linear spring model. But the settings of the original lengths of different springs are not identical and largely related to the positions of the particles, so the forces are essentially nonlinear in the dynamic simulation.

Based on the force equations of particles and Newton's equations of motion, an explicit time integration method called the Verlet algorithm [31] is applied to numerical integration of equations to solve the equilibrium position of the spring-mass system. In the beginning, the particles are at rest. Then the unbalanced forces push them to new positions at a discretized artificial time which is a very small value. During an artificial time, the forces are assumed to be the same. After an artificial time, the forces of nodes in new positions are updated and continue moving the nodes iteratively. The system moves to an equilibrium state within a relatively short time due to the action of drag forces and the fast calculation of the solver.

To perform the above dynamic simulation, reasonable forces of springs are required and can be evaluated by the result of dynamic simulation. Since the slack lengths have been defined, some trials need to determine reasonable elastic coefficients of various springs. In the field of virtual simulation, to simulate real objects, a great deal of research has been carried out on the physical parameters of springs to make the characteristics of digitized objects more similar to those of real objects [32–34]. However, in this research, the elastic coefficients of springs have no relevant physical characteristics and are just utilized to solve the equilibrium state under different spring forces to obtain a high-quality grid.

In general, the elastic coefficients of constraint springs are much larger than those of other springs, while the coefficients of the grid edge springs, the face edge springs, and the virtual springs do not differ much. After some trials, the reasonable range of elastic coefficient is 2800-3200 N/m for surface and curve constraint springs, 8000-10000 N/m for point constraint springs, 260-300 N/m for grid edge springs, face edge springs, and virtual springs, and 25-30 kg/s for resistance coefficient. The framework typically gives a good result when each coefficient is within the corresponding range. Within each range, the corresponding coefficient can be further adjusted interactively to obtain different results. However, the differences between the results are extremely small and are usually acceptable for the building grid.

It should be noted that if a certain constraint is strict, its corresponding elastic coefficient k is required to be infinite, and the corresponding displacement vector $\vec{d}_{\mathrm{p},i}$ would be zero, which is achieved through geometrical projection. In the actual operation, each particle's motion is calculated under forces except the forces of strict constraints in each time step. Then the particle constrained strictly is projected to the object imposing the strict constraint.

5. Grid generation

Based on the spring-mass model, a method is proposed to generate grids for free-form surfaces, named the spring net-like method. The main algorithm of the method includes three processes: initial grid generation, node adjustment, and node connection. The surface in Fig. 13a was taken as an example to explain these processes.

5.1. Initial grid generation

To create a visually pleasing grid, the initial grid is required to be structured. Directly creating structured grids over free-form surfaces is quite challenging. The single surface has a natural planar parameter domain, and the initial grid may be created by the mapping relationship between the surface and the planar domain. However, the multiple surface has no natural parameter domain, and the initial grid cannot be generated through mapping. To establish a planar domain for both single surface and multiple surface, surface discretization technique [35] and mesh parameterization technique [36] are used.

Surface discretization is a fast and effective method to approximate a continuous surface by using a surface mesh. The surface mesh is composed of a large number of discrete triangular faces and is frequently clumsy and does not fulfill architectural standards, yet it effectively expresses the surface shape. Surface discretization is widely utilized in CAD tools such as Rhino and Solidworks. Mesh parameterization usually obtains a one-to-one mapping between a surface mesh and a simple parameter domain by assigning each grid vertex a 2D coordinate [36]. Therefore, the surface mesh can be converted to a planar mesh through mesh parameterization. Among the parameterization methods, LSCM [37] is reasonably quick and robust. It features low angular distortion, ensuring parameterization bijectivity. Furthermore, because it is a natural border parameterization technique, there is no non-conformality distortion at the boundary. As a result, the LSCM method is used here to implement the bidirectional mapping between the surface mesh and the planar mesh. The detailed process of initial grid generation is shown in Fig. 13.

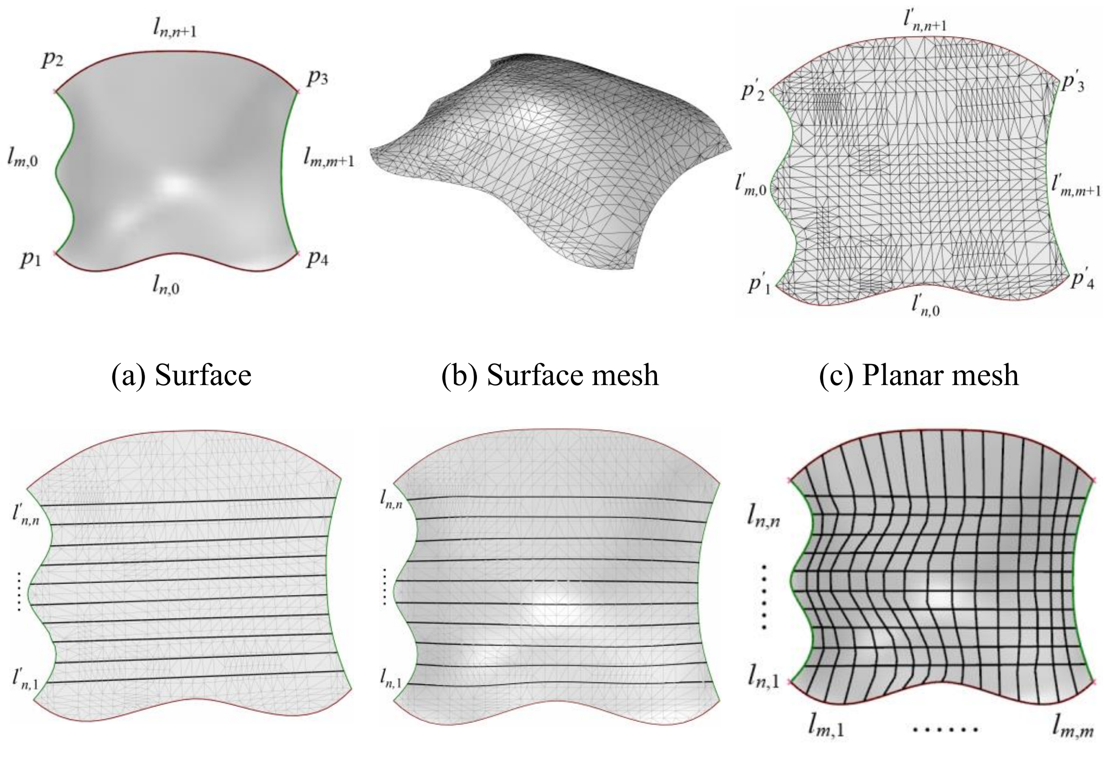
Four boundary curves of the surface were divided into two pairs (Fig. 13(a)). Then the surface was discretized into a surface mesh, as shown in Fig. 13(b). The LSCM algorithm was applied to obtain the planar form of the surface mesh. Corresponding to original surface, the planar mesh also contains two pairs of boundary curves (Fig. 13(c)). A pair of boundary curves $l'_{m,0}$ and $l'_{m,m+1}$ (green curves in Fig. 13(c)) were divided into n+1 segments. The other n lines (noted $l'_{n,1}$, $l'_{n,2}$,…, $l'_{n,n}$ in Fig. 13(d)) were acquired by connecting pairs of segment points at the same relative positions. n lines were uniformly sampled [28], with two adjacent sampling points spacing ε on each line. ε can be determined by Eq. (14).

$$\varepsilon=\rho\frac{d_{\min}}{m+1} \tag{14}$$

where $0 < \rho < 0.2$, and $d_{\min}$ is the length of the shortest line among n lines.

Through the point location algorithm [38], the index of the planar triangular cell containing each sampling point can be found. According to the spatial coordinates corresponding to the three vertices of the triangular cell, the spatial position of the sampling point was calculated by barycentric interpolation [39], as shown in Fig. 14. The sampling points on the surface mesh were connected to obtain n polylines (noted $l_{n,1}$, $l_{n,2}$,…, $l_{n,n}$ in Fig. 13(e)). These n polylines and the other pair of boundary curves (noted $l_{n,0}$ and $l_{n,n+1}$) were divided into m+1 segments, and m polylines on the other direction (noted $l_{m,1}$, $l_{m,2}$,…, $l_{m,m}$) were attained by connecting these segment points on the same relative locations. The n+m polylines and four boundary curves formed the initial grid, as shown in Fig. 13(f).

After this process, all the vertices of the initial grid are located on the surface mesh instead of the initial surface. The regularity and uniformity of the grid are not good, especially near the boundary. Therefore, the initial grid does not meet the requirements. However, the initial grid nodes are well-positioned (i.e., the valence of each internal node is the same), which will be necessary to determine the trend of the final grids.



(a) Surface (b) Surface mesh (c) Planar mesh

(d) n lines on planar mesh (e) n polylines on surface mesh (f) Initial grid with m = 15, n = 11

Fig. 13 Initial grid generation.

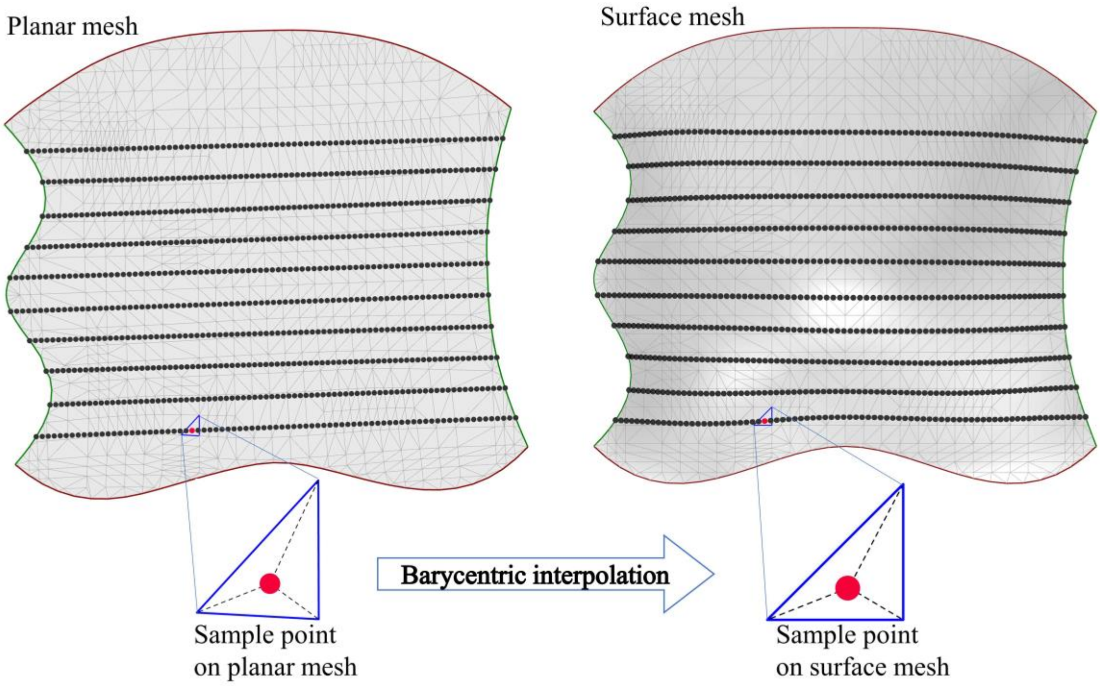


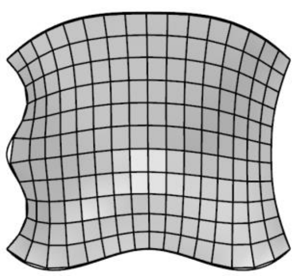
Fig. 14 Calculating the spatial position of the sampling point by barycentric interpolation.

5.2. Node adjustment

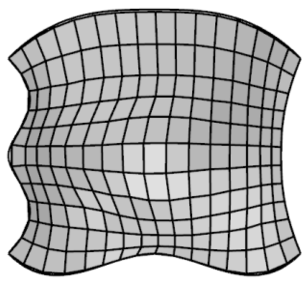
The initially generated grids are usually not of high quality. Therefore, some post-processes are needed to improve the overall quality of the elements. A nodal adjustment process is used here to adjust the node locations of the initial grid.

In this process, the initial grid was regarded as a spring net using the spring-mass model introduced in Section 4. The coefficients of various springs were given by some initial defaults and could be adjusted by some trials interactively. By using the Verlet algorithm, the particles of the spring net moved over the surface iteratively in the dynamic simulation. The convergence of the spring-mass model has been proven in [40]. The termination condition is that the maximum of all particle displacements in an iteration step is less than a given threshold or the number of iterations exceeds the maximum number of iterations. Fig. 15(a, b) presented the grids after one iteration and five iterations, respectively. When the termination condition was met, the final grid node distribution was acquired. As shown in Fig. 15(d), the grid nodes in the (m+2) × (n+2) matrix are distributed uniformly with all vertices subject to the surface and boundary vertices subject to the boundary. It should be noted that different grids require different iterations. Generally, the more grid nodes, the more iterations are required. The initial grid shown in Fig. 13(f) reaches convergence after 187 iterations, but the position of the grid vertices hardly changes after 20 iterations, as illustrated in Fig. 15(c, d). This shows that the process can quickly approach the state of convergence.

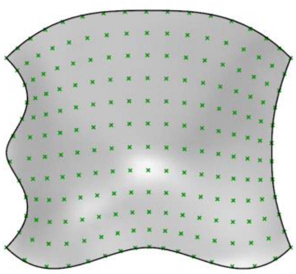
As stated in Fig. 2(d), a significant difference among the lengths of the grid rods connected to a node can break the continuum of the rod segments of the grid shell, affecting the fluency of the grid shells. After the node adjustment process, the nodes are thus uniformly distributed to ensure that the lengths of the grid edges next to the nodes are approximately the same, ensuring grid fluency.



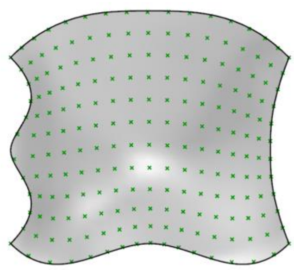
(a) Grid after one iteration



(b) Grid after five iterations



(c) Node distribution after 20 iterations



(d) Node distribution after 187 iterations

Fig. 15 Node adjustment by the spring net-based dynamic simulation.

5.3. Node connection

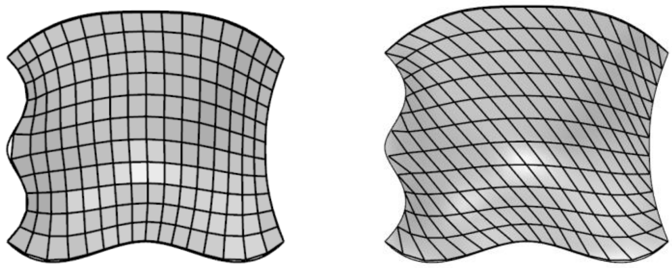
Grid shells come in many forms and are generally composed of triangular or quadrilateral grid cells. After the node adjustment, the grid nodes need to be reconnected to acquire the grid in the desired pattern.

In this process, the mode of node connections of the smallest similar unit was determined by the user and then tiled to the whole grid by traversing all units. For example, four types of grid cells shown in Fig. 16 are used to connect the nodes in Fig. 15(d), and four corresponding grids were obtained in Fig. 17. Although different forms of grids can be generated to give designers more choices, structured triangular and quadrilateral grids are more commonly used in buildings. This paper mainly discusses the generation and quality evaluation of these two forms of grids.

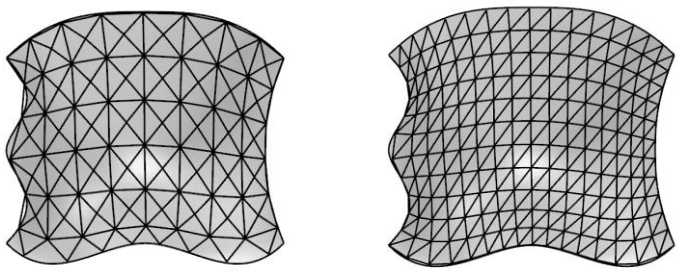
Connecting nodes with predefined grid cells is an efficient and robust process. A specific form of node connection ensures that the grid topology, which explains the relationships between the vertices, edges, and cells, is perfectly regular, so there will be no singular vertices that are detrimental to the fluency of the grid. The resulted grids consist of repeating unit cells and are usually regular and fluent, satisfying the design requirements and providing designers with high-quality options.

(a) Quadrilateral mode M_1 (b) Quadrilateral mode M_2 (c) Triangular mode M_3 (d) Triangular mode M_4

Fig. 16 Various predefined grid cell types are used for tilting the adjusted nodal set.



(a) Quadrilateral grid in M_1 (b) Quadrilateral grid in M_2



(c) Triangular grid in M_3 (d) Triangular grid in M_4

Fig. 17 Generated grids with various cell types.

6. Grid size control method

Sometimes, the designer may would like the generated grids to be of various lengths along with the boundary curves of the given surface. Taking the surface with a slender waist as an example, the initial grid was generated by the algorithm in the first process in Fig. 18(a). In the dynamic simulation process, all grid edge springs have the same slack length proportional to the average value of all current edge lengths, as illustrated in Eq. (2). As shown in Fig. 18(a), in the waist area, the lengths of the initial grid edges are smaller than those in other areas, and the corresponding slack lengths are substantially greater than the initial lengths. As a result, the grid edges in this area tend to elongate, with relatively large ratios of elongated lengths to initial lengths. However, the boundary constraint springs prevent the grid edges from extending beyond the surface, causing the grid to overlap (Fig. 18(b)).

To improve the grid regularity and avoid overlaps, the grid edges are desired to be varied and adaptive to the surface shape. The grid size is mainly controlled by grid edge springs. In Section 4.1.1, the force of the grid edge spring has been introduced and is to get a uniform grid. However, to get a non-uniform and adaptive grid, the force of the grid edge spring is redefined as:

$$\vec{T}_{t,ij} = k_{t,j}(e_{t,ij} - |\vec{d}_{ij}|) \cdot \frac{\vec{d}_{ij}}{|\vec{d}_{ij}|}. \tag{15}$$

$e_{t,ij}$ is the original length of the i-th edge on the curve $l_{t,j}$, defined as:

$$e_{t,ij} = f_g l_{t,j} \frac{l_{s,i} + l_{s,i+1}}{\sum_{i=0}^{s}(l_{s,i} + l_{s,i+1})}, i \in [0,s], j \in [0,t]. \tag{16}$$

where $t = m$, $s = n$ or $t = n$, $s = m$; f_g is the same as the one in Eq. (2).

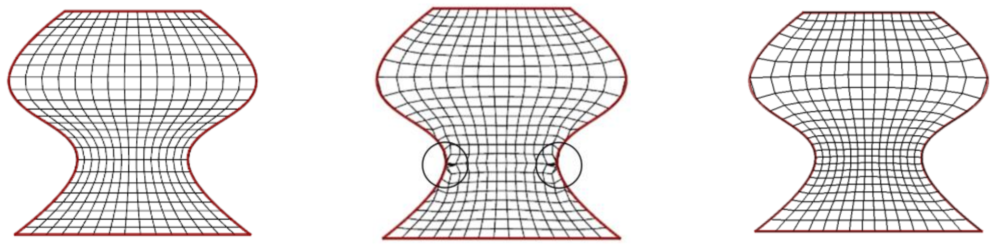
$k_{t,j}$ is the elastic coefficients of grid edge springs on the curve $l_{t,j}$, that is

$$k_{t,j} = k_g \frac{\sum_{i=0}^{t+1}(l_{t,j})}{(t+2)l_{t,j}}, j \in [0,t+1], \tag{17}$$

where k_g is the basic elastic coefficient.

The original lengths of grid edge springs are varied and related to their positions. The elastic coefficients of grid edge springs are also adjusted and not all the same. Only the grid edge springs on the same grid curve have the same elastic coefficient. The shorter the grid curve, the larger the elastic coefficient of grid edge springs on this curve.

Except for the grid edge spring forces, other forces and processes of the spring net-like method are not changed. Fig. 18(c) shows that the grid generated by the locally adjusted method is quite regular and fluent without overlaps of grids.



(a) Initial grid (b) Grid by the original method (c) Grid by the locally adjusted method

Fig. 18 Grids for a surface with a slender waist.

7. Adaption to complex boundary curves

The above-proposed grid generation algorithm mainly focuses on the curved surface composed of four different boundary curves. Although the framework is effective and efficient, and the resulted grids are of high quality over such surfaces, the application scope is rather limited. To broaden the application scope of the proposed, some extensional operations are introduced to handle surfaces with more complex boundary curves.

7.1. Ringed surface

Ringed surfaces are relatively common in free-form grid structural design, as shown in the example in Fig. 19 (a). Fig. 19 (a) is a free-form grid shell that is to be built in Taizhou, China. A ringed surface has two disjunct closed boundaries. The two boundary curves are first divided into n segments. And n lines are acquired by connecting pairs of segment points at the same relative positions. Then these n curves are divided into m+1 segments relatively, and m polylines in the other direction are obtained by connecting these segment endpoints, following the same rules. Therefore, the n lines, m polylines, and two boundary curves form the initial grid. The nodal positions are then adjusted through the same process presented in Section 5. After the node adjustment, grid nodes in the $(m+2) \times n$ matrix are achieved and connected into ringed grids.

The extended method was applied to the ringed surface presented in Fig. 19(b). The generated quadrilateral grid was obtained as presented in Fig. 19(c). It is shown that the grid quality is much better than the original grid presented in Fig. 19 (a) at the position of the surface with sharp curvature.

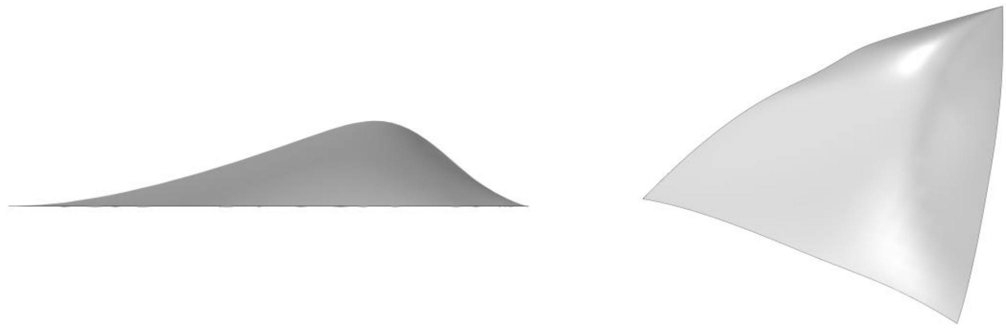
(a) Architectural rendering (b) Free-form surface model (c) Quadrilateral grid

Fig. 19 Grid generation for a drop-shaped and ringed surface roof.

7.2. Free-form surface with odd number of boundary curves

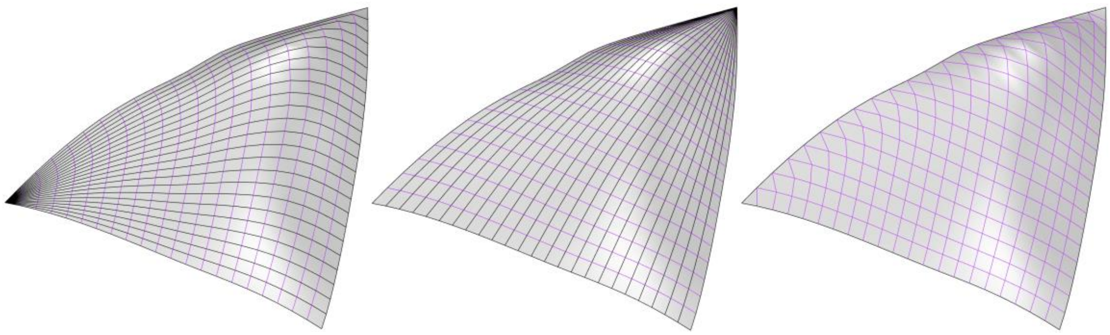
Surfaces with three or more than four boundary curves are frequently observed in some projects. This type of surface cannot be meshed directly, and the boundary curves must be processed in order to convert the surface into a pseudo surface with four boundary curves.

For a surface with three boundary curves (Fig. 20), the corner of the surface can be regarded as a degenerate edge with a very short length, and the surface can then be converted into a surface with four boundary curves for grid generation. As shown in Fig. 21(a, b), two different corners of the surface are regarded as degenerate edges, and grids are generated accordingly. However, the two grids are of poor quality and cannot be constructed with the grid cells clustered at the degraded edge. To avoid the above situation, the polylines over the two grids that do not intersect with the degraded edge (purple polylines in the Fig. 21(a, b)) are extracted respectively and recombined into the final grid (Fig. 21(c)). It should be noted that the designer has the option of generating the final grid based on any two corners. Different choices will result in different grid line directions, yet all final grids are more visually appealing, and the grid's quality has substantially improved, particularly at the corners.



(a) Front view (b) Perspective view

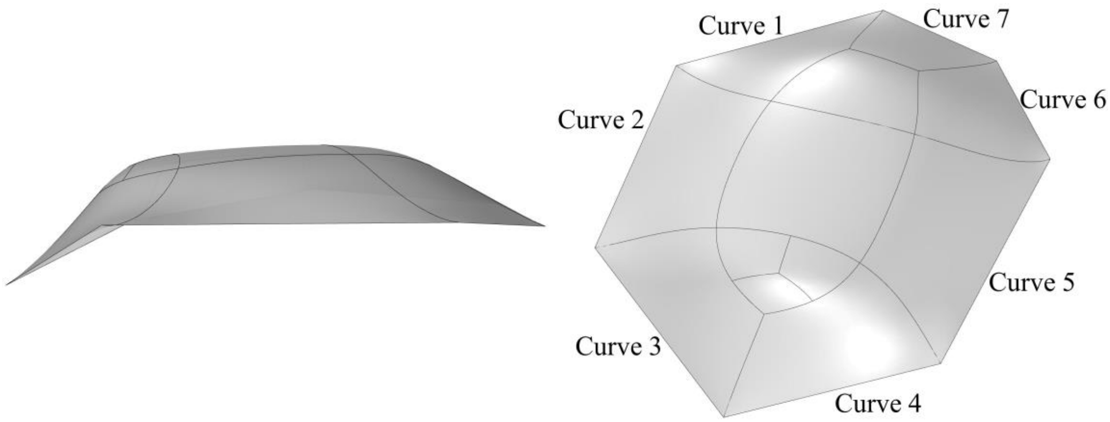
Fig. 20 A surface with three boundary curves.



(a) Grid with the lower left corner as the degenerate edge (b) Grid with the upper right corner as the degenerate edge (c) Final grid

Fig. 21 Grid generation for a surface with three boundary curves.

For surfaces with more than four boundary curves, some boundary curves need to be merged to reduce the number of boundary curves to four. As shown in Fig. 22, the multiple surface is made up of 12 single surfaces and contains seven naked boundary curves. The boundary curves 1 and 2, the boundary curves 4 and 5, are merged into $l_{n,0}$ and $l_{n,n+1}$ respectively, while $l_{m,0}$ consists of the boundary curves 6 and 7, as illustrated in Fig. 23. The surface is then thought to have four boundary curves. Different from the real surface with four boundary curves, when the initial grid is generated, due to the existence of vertices in the merged curve, the boundary edges of the grid near the vertices often deviate from the surface boundary curves, as shown in Fig. 24. To ensure the accurate representation of the surface by the grid, point constraint springs that only work on the boundary vertices of the grid are set at the vertices of each merged boundary curve. The subsequent steps are the same as for the four-sided surface, and the final grid is shown in Fig. 25. Furthermore, designers can choose different combinations of boundary curves according to their preferences. For example, the boundary curves 2 and 3 are merged into a curve, as are the boundary curves 5, 6, and 7, and the final grid is shown in Fig. 26.



(a) Front view (b) Perspective view

Fig. 22 Surface with seven boundary curves.

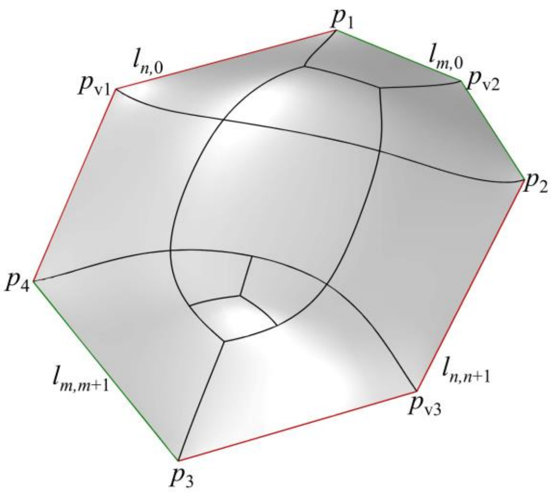


Fig. 23 Surface with boundary curves merged.

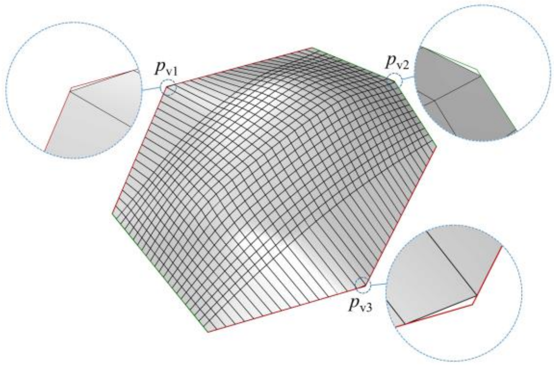


Fig. 24 Initial grid.

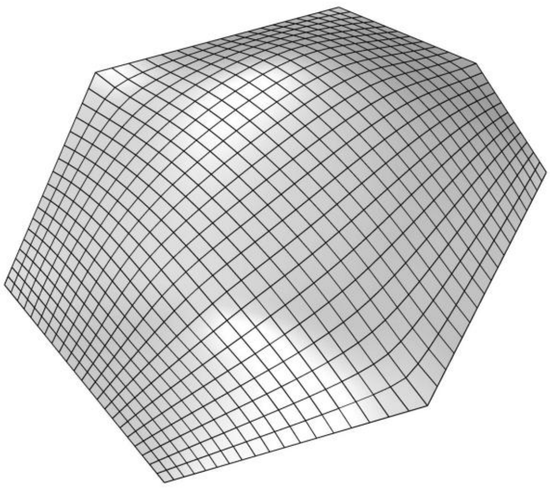


Fig. 25 Final grid.

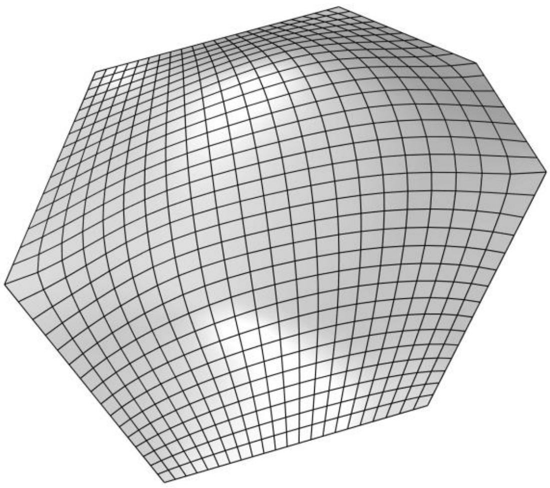


Fig. 26 Grid.

7.3. Free-form surface with internal boundary

For grid generation over a trimmed surface with internal boundary curves, one method is to generate an extended grid using the spring net-like method over the original complete surface without considering the internal boundary curves. Then remove all redundant edges that are inside of the inner boundary. The shortcoming of such a geometry operation is that the resulted grid has nodes and non-uniform edges adjacent to the boundary curves, making it difficult to satisfy the requirements of architectural aesthetics.

An improved technique is used to generate the extended grid over the basic surface. The grids are firstly generated over the complete surface. Then, the grid is filtered to get rid of the unnecessary grid cells that are outside the definition of the trimmed surface. The key to filtering the grid is to decide whether to keep or delete the grid cells intersecting with the boundary curves. The cases of intersection between a quadrilateral grid cell and the boundary curve are shown in Fig. 27, and the filtering criterion of a quadrilateral grid cell is defined as:

$$\begin{cases} \dfrac{S_{\text{in}}}{S_{\text{out}}} \le \dfrac{1}{3}, & \text{remove} \\ \dfrac{1}{3} < \dfrac{S_{\text{in}}}{S_{\text{out}}} < 3, & \text{if } \dfrac{l_{\text{BE}}}{l_{\text{BC}}} \le \dfrac{l_{\text{AF}}}{l_{\text{AD}}}, \text{ keep } \Delta\text{ABD and remove } \Delta\text{BDC} \\ \dfrac{S_{\text{in}}}{S_{\text{out}}} > 3, & \text{keep} \end{cases} \tag{18}$$

where S_{in} and S_{out} are the areas of the grid cells inside and outside the given free-form surface, respectively.

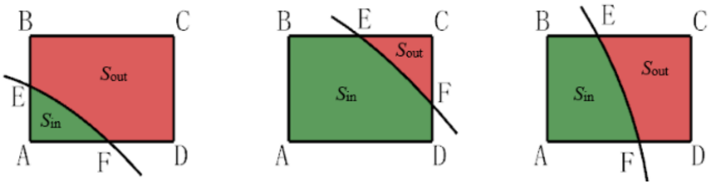


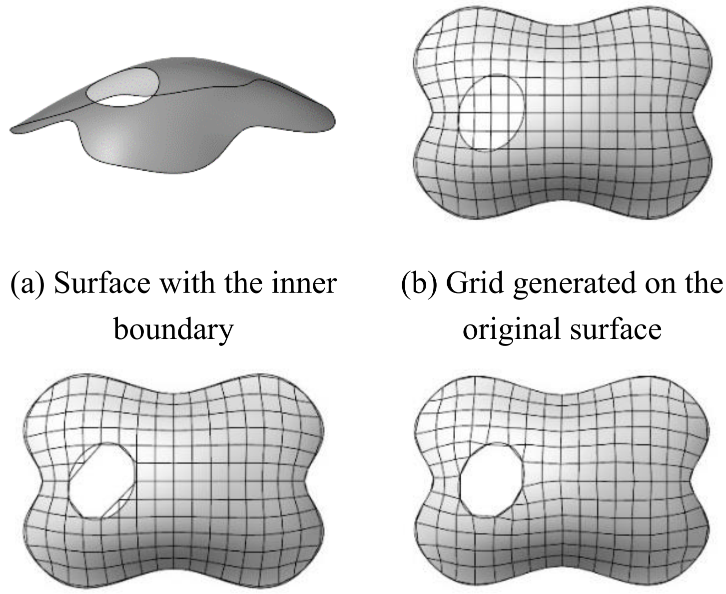
Fig. 27 The intersection between a quadrilateral grid cell ABCD and the boundary curve EF.

A triangular grid is filtered according to Eq. (19). Then the edges around the boundary are adjusted to approximate the boundary curves as much as possible by connecting the related nodes or eliminated edges:

$$\begin{cases} \dfrac{S_{\text{in}}}{S_{\text{out}}} \le 1, & \text{remove} \\ \dfrac{S_{\text{in}}}{S_{\text{out}}} > 1, & \text{keep} \end{cases} \tag{19}$$

After filtered, the generated grids are relaxed by the spring-based dynamic simulation introduced in Section 5.2, and the final grid over the given surface is obtained.

Based on the above grid filtering technique, a surface with an inner boundary is meshed according to the proposed framework, and the processes are shown in Fig. 28(a-d). The resulted grid (Fig. 28(d)) is regular and fluent and complies with the internal boundary curve, which demonstrates that the proposed framework can be well adapted to the surface with internal boundary curves.



(a) Surface with the inner boundary

(b) Grid generated on the original surface

(c) Grid with redundant edges removed

(d) Relaxed grids

Fig. 28 Grid generation over a surface with an internal boundary curve.

7.4. Free-form surface with external boundary

Similarly, given a surface with complex external boundary curves, the surface is firstly extended and trimmed according to the original one to form a surface with quadrilateral boundary curves. Then grids are generated for the quadrilateral surface with the spring net-like method in Section 5. The grid elements outside the given surface are removed according to the principle introduced in Section 7.3. Then the filtered grids are relaxed by employing the dynamic simulation algorithm.

As shown in Fig. 29(a-d), the grids over a surface with a complex outer boundary are generated accordingly. The resulted grids are shown in Fig. 29(d) with regular and fluent cells. The example illustrates the effectiveness of the proposed framework applying to the surface with complex outer boundary curves.

(a) Surface with a complex outer boundary

(b) Grid generated on the extended surface

(c) Grid with redundant edges removed

(d) Relaxed grid

Fig. 29 Grid generation over a surface with an external boundary curve.

8. Grid quality indexes

Traditionally, architects or engineers evaluate grids generated in their design through a visual check. This requires the designers' experience to assess the quality of grids in terms of regularization and fluency. Quantitative methods are essential to evaluate the quality of architectural grids of free-form surfaces. Quantitative metrics on the evaluation of the grid quality of grid shells can be borrowed from early studies of Finite Element Analysis (FEA) mesh element distortion [41,42]. Traditional grid quantitative indexes, such as face shape quality and edge length, can provide an overall description of the quality, but these indexes are mainly focused on FEA applications and are not appropriate in the context of structural grid shells. Therefore, an index is used to assess the fluency of the structured grid, whereby improved fluency means a better visual expression of a grid shell as required in most architectural applications.

Since grids used in the field of architecture are mostly triangular or quadrilateral, the quality evaluations discussed below are mainly for triangular or quadrilateral grids, even though grids in other patterns can be generated by our method.

8.1. Regularity index

To quantify the grid regularity, the shape quality index of the triangular or quadrilateral grid face is used and defined as:

$$q = \begin{cases} q_{\text{tri}}, & \text{triangle} \\ q_{\text{qua}}, & \text{quadrangle} \end{cases}, \tag{20}$$

where the triangular shape index q_{tri} is defined by Eq. (21) [43] and the quadrilateral shape index q_{qua} is defined by Eq. (22) [15].

$$q_{\text{tri}} = 4\sqrt{3}\frac{S_{\Delta\text{ABC}}}{l_{\text{AB}}^{2} + l_{\text{BC}}^{2} + l_{\text{AC}}^{2}} , \tag{21}$$

where $S_{\Delta\text{ABC}}$ is the triangle area; l_{AB}, l_{BC}, and l_{AC} are the side lengths. $q_{\text{tri}} \in [0,1]$. For an equilateral triangle, $q_{\text{tri}} = 1$, and for a degenerate triangle (three points collinear), $q_{\text{tri}} = 0$. Approximately equilateral triangles are desired. For quadrilateral shape:

$$\begin{aligned} q_{\text{qua}} = {} & 4\sqrt[4]{\frac{S_{\Delta\text{ABC}} \cdot S_{\Delta\text{BCD}}}{\left(l_{\text{AB}}^{2} + l_{\text{AD}}^{2}\right)\cdot\left(l_{\text{AB}}^{2} + l_{\text{BC}}^{2}\right)}} \\ & \times\sqrt[4]{\frac{S_{\Delta\text{CDA}} \cdot S_{\Delta\text{ABD}}}{\left(l_{\text{BC}}^{2} + l_{\text{CD}}^{2}\right)\cdot\left(l_{\text{CD}}^{2} + l_{\text{AD}}^{2}\right)}} \end{aligned}, \tag{22}$$

where $S_{\Delta\text{ABC}}$, $S_{\Delta\text{BCD}}$, $S_{\Delta\text{CDA}}$, $S_{\Delta\text{ABD}}$ denote the area of the triangles ΔABC, ΔBCD, ΔCDA, ΔABD; l_{AB}, l_{BC}, l_{CD}, and l_{AD} are the side lengths of the quadrangle. $q_{\text{qua}} \in [0,1]$.

The higher the q_{qua}, the better the shape quality of the quadrangle. If $q_{\text{qua}} = 1$, the quadrangle is a square whose shape quality is the best.

The average value $\bar{q}$ defined in Eq. (23) and the standard deviation s defined in Eq. (24) are employed to evaluate the whole grid regarding regularity. The larger $\bar{q}$ and the smaller s, the more regular the grid:

$$\bar{q} = \frac{\sum_{i=1}^{N} q_i}{N}, \tag{23}$$

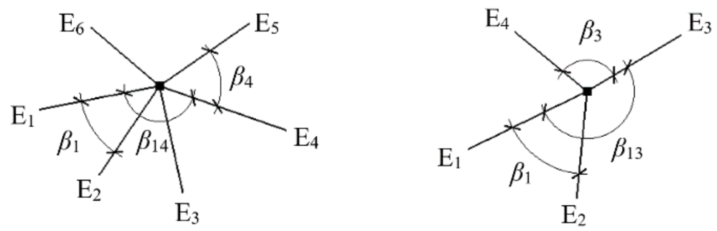
$$s = \sqrt{\frac{\sum_{i=1}^{N} (q_i - \bar{q})^2}{N-1}} \tag{24}$$

where N is the total number of objects, and q_i is the value of the i-th object.

8.2. Fluency index

Grid fluency is an essential aspect in evaluating a free-form grid shell. Wang et al. [4] presented an index to assess the fluency of a structured triangular grid based on angles of interior vertices. We broaden the application scope of this index to make it also applicable to a quadrilateral grid.

For a structured grid, the number of edges connected to any interior vertex noted as d is the same. $d = 6$ for a triangular grid while $d = 4$ for a quadrilateral grid. The factors that can affect fluency are the angles between two opposite edges and the opposite angles at the interior vertex. As shown in Fig. 30, if the opposite edges (i.e., E_1 and E_4 in a triangular grid and E_1 and E_3 in a quadrilateral grid) are in a straight line (β_{14}=180° and β_{13}=180°, respectively), a more fluent grid will be achieved. It is also expected that the opposite angles would be the same in a more fluent grid (i.e., $\beta_1 = \beta_4$ in a triangular grid and $\beta_1 = \beta_3$ in a quadrilateral grid). Thus, the fluency index of an interior vertex is defined in Eqs. (25-27).



(a) Triangular grid (b) Quadrilateral grid

Fig. 30 Angles of an interior vertex in structured grid

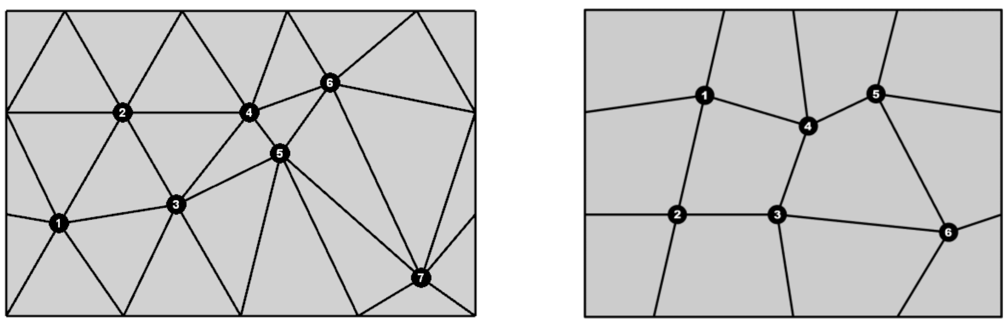
$$\sigma_i=\sqrt{\frac{\sum_{j=1}^{r}(\beta_{jk}-180°)^2}{r}},k=j+r, \tag{25}$$

$$\tau_i=\sqrt{\frac{\sum_{j=1}^{r}(\beta_j-\beta_{j+r})^2}{r}}, \tag{26}$$

$$\delta_i=\sqrt{\sigma_i^2+\tau_i^2}, \tag{27}$$

where δ_i denotes the fluency index of the i-th vertex; r = 0.5d, that is r = 3 for a triangular grid and r = 2 for a quadrilateral grid; β_{jk} is the angle between the j-th edge and the k-th edge and β_j is the angle between the j-th edge and the (j+1)-th edge (if j+1 > d, replaced by the 1-st edge).

The smaller the δ_i, the more fluent the grid at the i-th vertex. The smaller the mean value of the fluency index, the better the fluency of the structured grid.



(a) Triangular grid with 7 interior vertices (b) Quadrilateral grid with 6 interior vertices

Fig. 31 Simple planar grids.

Two simple grids in Fig. 31 are evaluated using this index as examples. In the triangular grid (Fig. 31(a)), p_2 is an ideal point with δ = 0°. The points p_5 and p_7 are visually non-ideal, and their fluency indexes are as large as 61.9° and 85.9°, respectively. Similarly, in the quadrilateral grid (Fig. 31(b)), point p_2 is also an ideal point whose fluency index equals 0°, and the fluency is the worst around point p_6 with δ = 78.0°. As shown in Table. 1 and Table. 2, the magnitude of δ reflects the grid fluency around each point.

Table. 1 Fluency index of interior vertices of the triangular grid.

Point number	p_1	p_2	p_3	p_4	p_5	p_6	p_7
σ (°)	13.5	0.0	21.1	15.7	45.9	21.0	61.3
τ (°)	12.2	0.0	12.3	15.6	41.5	20.1	60.2
δ (°)	18.2	0.0	24.4	22.1	61.9	29.1	85.9

Table. 2 Fluency index of interior vertices of the quadrilateral grid.

Point number	p_1	p_2	p_3	p_4	p_5	p_6
σ (°)	17.3	0.0	20.4	35.2	38.1	45.0
τ (°)	24.5	0.0	28.9	49.8	53.9	63.7
δ (°)	30.0	0.0	35.4	61.0	66.0	78.0

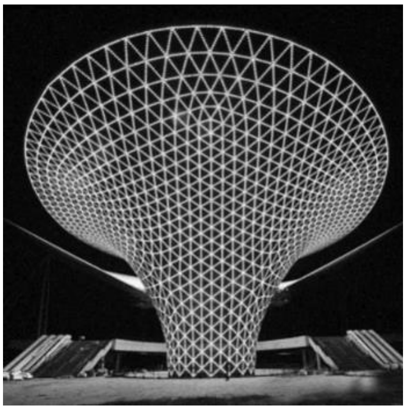
9. Additional case study

In previous sections, all the components of the framework for grid generation have been introduced. Apart from the three main processes (initial grid generation, node adjustment, and node connection), the framework includes additional geometry operations to handle surfaces with complex boundary curves. Grid quality indexes such as regularity and fluency indexes are also utilized to evaluate the generated grids. The framework described in the previous sections has been made available as a grid generator in a plugin named Grasshopper which is a Rhinoceros-based geometric modelling tool, providing a parametric modelling environment. In this section, the grid generator is applied to an existing project, and the grid quality indexes are compared between the proposed framework and other methods. In addition, mechanical performance analysis is also carried out.

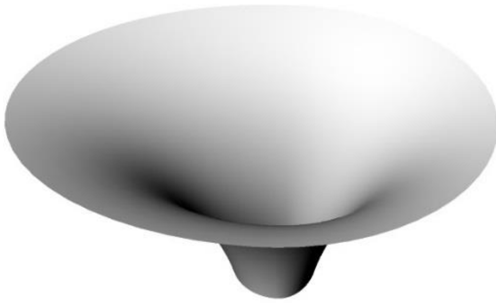
9.1. Grid generation

The Sun Valley of Expo Axis is a typical free-form grid structure (noted G_0) in Shanghai, China, as shown in Fig. 32(a). G_0 has good regularity, but there are several singular vertices. These singular vertices destroy the fluency of the whole grid and reduce the architectural beauty. A corresponding surface model has been established based on the Sun Valley (Fig. 32(b)). The surface is a single and ringed NURBS surface. Its top boundary is approximately an ellipse with a 100 m long axis and an 80 m short axis, while its bottom boundary is approximately an ellipse with a 30 m long axis and a 27 m short axis, and the height is 40 m. The surface is meshed by the mapping method and the proposed framework, respectively. As Fig. 32(c) and Table. 3 illustrate, the grid G_1 generated by the mapping method is very fluent without any singular vertex, and its $\bar{\delta}$ = 3.69°. But G_1 has many sliver triangles, and the regularity of the gird is not good obviously with $\bar{q}$ =0.838 and s = 0.157. As Fig. 32(d) and Table. 3 illustrate, the grid G_2 by the proposed framework is not only fluent with $\bar{\delta}$ = 4.82°, but also regular with $\bar{q}$ = 0.990. G_2 has the best visual expression among G_0, G_1, and G_2.

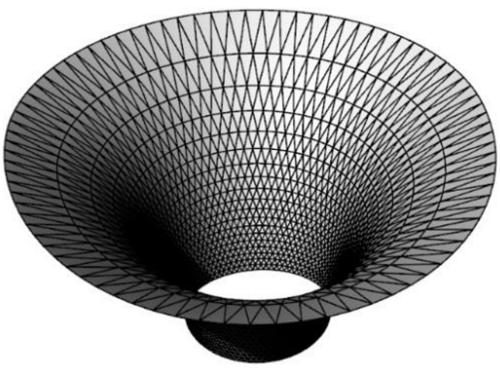
Achieving the harmony of fluency and regularity, the proposed framework is better than both the mapping method whose grid is not regular and the method [44] previously used for the Sun Valley whose grid is not fluent.



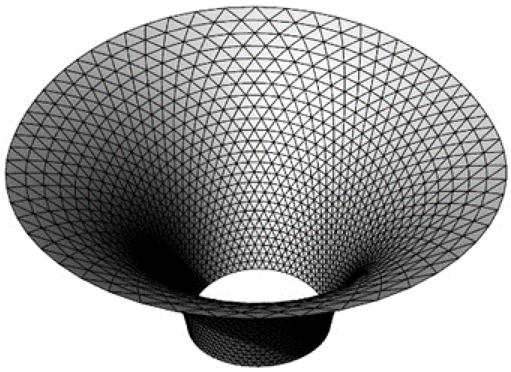
(a) Sun Valley of Expo Axis



(b) Surface for Sun Valley



(c) Grid G_1 by the mapping method



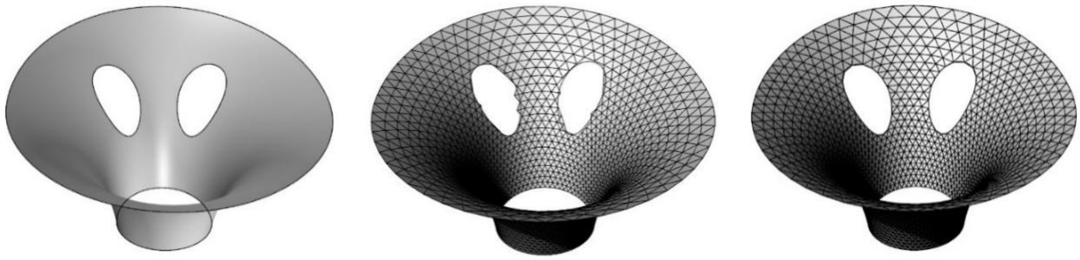
(d) Grid G_2 by the proposed framework

Fig. 32 Grid generation for the Sun Valley.

Table. 3 Grid quality indexes.

Grid	$\bar{l}\,(m)$	$\bar{q}$	$s\times10^{-2}$	$\bar{\delta}(°)$
G_1 (Fig. 32(c))	2.49	0.838	15.7	3.69
G_2 (Fig. 32(d))	2.54	0.990	0.954	4.82
G_3 (Fig. 33(c))	2.55	0.987	1.54	5.24

To be more challenging, the surface was trimmed by two closed curves, as shown in Fig. 33(a). As introduced in Section 7, the filtered grid (Fig. 33(b)) was attained by filtering the grid G_2 based on the trimmed surface. The final grid G_3 is acquired by relaxing the filtered grid. As Fig. 33(c) and Table. 3 illustrate, the grid G_3 that is regular with $\bar{q}$ = 0.987 and fluent with $\bar{\delta}$ = 5.24° expresses the trimmed surface adequately.



(a) Trimmed Surface (b) Filtered grid (c) Relaxed grid G_3

Fig. 33 Grid generation for the trimmed surface.

9.2. Mechanical performance

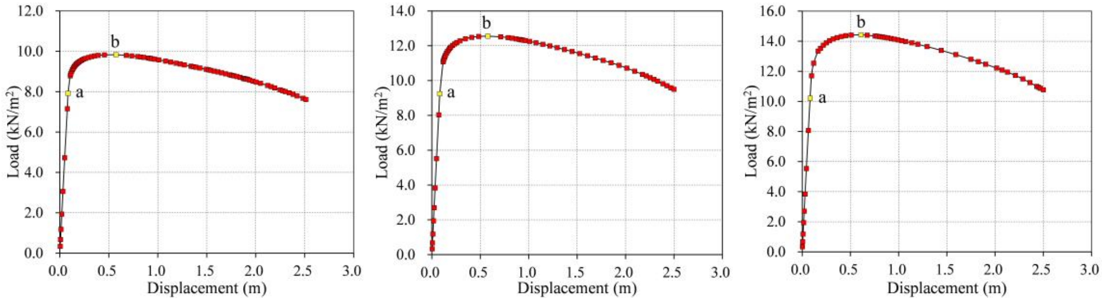
Many researchers have studied the mechanical properties of classic grid shells [45–47]. As a result, it is worthwhile to investigate the mechanical performance of free-form grid shells. To evaluate the mechanical performance, detailed geometric and material non-linear finite element analyses taking into account the imperfections (GMNAI) are performed using ANSYS [48].

The free-form grid shell shown in Fig. 32(d) is used to create three finite element models for analysis. All members of each model have identical cross-sections, as indicated in Table. 4. The three finite element models are developed using the BEAM188 element. This element is based on Timoshenko beam theory and takes shear deformation effects into account, and each member is simulated with three elements. The structural boundary conditions are hinged.

Table. 4 Cross-sections of members of the three models.

Designation	Grid member (diameter × thickness (mm))
Model 1	ϕ180 × 12
Model 2	ϕ219 × 12
Model 3	ϕ245 × 12

The elasto-plastic constitutive model is used in finite element analysis. The yield strength and Young's modulus of the steel are 235 MPa and 2.1×10^5 MPa, respectively. The vertical load is applied uniformly over the whole span of the three models. Furthermore, geometric imperfections are accounted for in the finite element analysis by scaling the first elastic buckling modes to a particular amplitude and superimposing it on the initial perfect geometry. The amplitude of the imperfections is taken as 1/300 of the span, with the amplitude of the imperfections set to 1/300 of the span.The load-displacement curve of each model is obtained from the GMNAI, as shown in Fig. 34. The displacement represents the maximum vertical displacement of all nodes in each model. As shown in Fig. 34, the load-displacement curve of each model has two characteristic times, denoted by time "a" and time "b", respectively. Time "a" is defined as the time when the member yielding first initiates, and time "b" is the time when the model reaches its ultimate bearing capacity. It can be seen that the ultimate bearing capacities of the three models are 9.83 kN/m^2, 12.55 kN/m^2, and 14.42 kN/m^2, respectively. As the cross-section grows larger, so does the ultimate bearing capacity of the model. Even Model 1, with the smallest cross-section, has a high ultimate bearing capacity. Additionally, from time a to time b, the displacements of corresponding nodes of the three models are all relatively large, indicating that the three models do not fail suddenly under this load case.



(a) Model 1 (b) Model 2 (c) Model 3

Fig. 34 Load-displacement curves of the three models.

10. Conclusion and future research

To mesh a free-form surface with complex boundary curves into a regular and fluent grid for the preliminary design of grid shells, this paper proposes a new grid generation framework. The framework relies on a spring-mass model to achieve regular and fluent triangular or quadrilateral grids over free-form surfaces. The framework can also handle surfaces with complex boundaries. First, a quadrilateral grid is decorated on the surface based on surface discretization and mesh parameterization. Secondly, the distribution of the initial grid vertices is adjusted by assuming the grid as a spring-mass system. Thirdly, high-quality grids are created by connecting the nodes in an equilibrium state with a predefined pattern. Finally, the generated grid is relaxed with the spring-mass model, alongside additional geometric operations including grid size adjustment and filtering techniques, to further improve the grid regularity and fluency. In the spring-mass model, spring forces between connected particles control the grid size; spring forces of faces regularize the grid shape; surface attraction forces to particles keep the spring net on the surface; boundary attraction forces to the boundary particles make the spring net cover the whole surface; anchoring forces can fix some particles. The proposed framework is robust, effective, and can be applied to free-form surfaces with complex boundary conditions. In addition to the conventional quantitative measurements of grid quality in terms of grid shape, we broaden the application scope of the fluency index to make it applicable to both triangular and quadrilateral grids, so a more concrete perception of the grid quality is obtained. Examples show that the framework can be applied to diverse free-form surfaces and the generated grids are fluent and regular in harmony with the requirements of architectural aesthetics. Compared with topology optimization, the grid generated by the proposed framework can better meet the requirements of architectural aesthetics and industrial production. This framework can be a useful tool to generate structured grids for the design of free-form grid shells.

It should be pointed out that the proposed framework can mainly generate grids in harmony between regularity and fluency, while the grid size may be non-uniform. However, the uniformity is also of great importance to the architectural grid, and a large difference in grid size is not conducive to the section design of bars and the construction cost for grid shells. It is necessary to do further research to realize the harmony of grid uniformity, regularity, and fluency based on the spring-mass model in the future. Besides, other specific requirements, such as planarization of polygonal grids, should also be considered to improve and extend the method.

Acknowledgements

This research was sponsored by the National Natural Science Foundation of China under Grant 51678521, 51778558 and by the Natural Science Foundation of Zhejiang Province LY15E080017. The project is also supported by the Foundation of Zhejiang

Provincial Key Laboratory of Space Structures, Grant 21705. The authors would like to thank them for their financial support.

References

[1] C.Y. Cui, B.S. Jiang, A morphogenesis method for shape optimization of framed structures subject to spatial constraints, Engineering Structures. 77 (2014) pp. 109–118. https://doi.org/10.1016/j.engstruct.2014.07.032.

[2] J.N. Richardson, S. Adriaenssens, R. Filomeno Coelho, P. Bouillard, Coupled form-finding and grid optimization approach for single layer grid shells, Engineering Structures. 52 (2013) pp. 230–239. https://doi.org/10.1016/j.engstruct.2013.02.017.

[3] C.J.K. Williams, The analytic and numerical definition of the geometry of the British Museum Great Court Roof, Mathematics & Design. (2001) pp. 434–440. http://opus.bath.ac.uk/14111/.

[4] Q. Wang, J. Ye, H. Wu, B. Gao, P. Shepherd, A triangular grid generation and optimization framework for the design of free-form gridshells, Computer-Aided Design. 113 (2019) pp. 96–113. https://doi.org/10.1016/j.cad.2019.04.005.

[5] S. Owen, A survey of unstructured mesh generation technology, 7th International Meshing Roundtable. (1998) pp. 239–267. http://ima.udg.edu/~sellares/ComGeo/OwenSurv.pdf.

[6] B. Wang, B.C. Khoo, Z.Q. Xie, Z.J. Tan, Fast centroidal Voronoi Delaunay triangulation for unstructured mesh generation, Journal of Computational and Applied Mathematics. 280 (2015) pp. 158–173.

https://doi.org/10.1016/j.cam.2014.11.035.

[7] S.H. Lo, Dynamic grid for mesh generation by the advancing front method, Computers & Structures. 123 (2013) pp. 15–27. https://doi.org/10.1016/j.compstruc.2013.04.004.

[8] W.A. Cook, W.R. Oakes, Mapping method for generating three-dimensional meshes: past and present, Los Alamos National Lab., NM (USA), 1982. https://www.osti.gov/biblio/5255207.

[9] J. Cuillière, An adaptive method for the automatic triangulation of 3D parametric surfaces, Computer-Aided Design. 30 (1998) pp. 139–149. https://doi.org/10.1016/S0010-4485(97)00085-7.

[10] P. Winslow, S. Pellegrino, S.B. Sharma, Multi-objective optimization of free-form grid structures, Structural and Multidisciplinary Optimization. 40 (2010) pp. 257–269. https://doi.org/10.1007/s00158-009-0358-4.

[11] L. Su, S. Zhu, N. Xiao, B. Gao, An automatic grid generation approach over free-form surface for architectural design, Journal of Central South University. 21 (2014) pp. 2444–2453. https://doi.org/10.1007/s11771-014-2198-7.

[12] P. Shepherd, P. Richens, The case for Subdivision Surfaces in building design, Journal of the International Association for Shell and Spatial Structures. 53 (2012) pp. 237–245.

[13] EvoluteTools, (2021) http://www.evolute.at/software-en/software-overview. Last access: 6th July 2021.

[14] B. Gao, T. Li, T. Ma, J. Ye, J. Becque, I. Hajirasouliha, A practical grid

generation procedure for the design of free-form structures, Computers & Structures. 196 (2018) pp. 292–310. https://doi.org/10.1016/j.compstruc.2017.10.006.

[15] B. Gao, C. Hao, T. Li, J. Ye, Grid generation on free-form surface using guide line advancing and surface flattening method, Advances in Engineering Software. 110 (2017) pp. 98–109. https://doi.org/10.1016/j.advengsoft.2017.04.003.

[16] L. Tierui, Y. Jun, S. Paul, W. Hui, G. Boqing, Computational Grid Generation for the Design of Free-Form Shells with Complex Boundary Conditions, Journal of Computing in Civil Engineering. 33 (2019) pp. 4019004. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000828.

[17] R. Oval, M. Rippmann, R. Mesnil, T. Van Mele, O. Baverel, P. Block, Feature-based topology finding of patterns for shell structures, Automation in Construction. 103 (2019) pp. 185–201. https://doi.org/10.1016/j.autcon.2019.02.008.

[18] K. Shimada, D.C. Gossard, Automatic triangular mesh generation of trimmed parametric surfaces for finite element analysis, Computer Aided Geometric Design. 15 (1998) pp. 199–222. https://doi.org/10.1016/S0167-8396(97)00037-X.

[19] A.L. Zheleznyakova, S.T. Surzhikov, Molecular dynamics-based unstructured grid generation method for aerodynamic applications, Computer Physics Communications. 184 (2013) pp. 2711–2727. https://doi.org/10.1016/j.cpc.2013.07.013.

[20] A.L. Zheleznyakova, Molecular dynamics-based triangulation algorithm of free-

form parametric surfaces for computer-aided engineering, Computer Physics Communications. 190 (2015) pp. 1–14. https://doi.org/10.1016/j.cpc.2014.12.018.

[21] Q. Wang, B. Gao, T. Li, H. Wu, J. Kan, B. Hu, A triangular mesh generator over free-form surfaces for architectural design, Automation in Construction. 93 (2018) pp. 280–292. https://doi.org/10.1016/j.autcon.2018.05.018.

[22] I.P. Rosinha, K. V Gernaey, J.M. Woodley, U. Krühne, Topology optimization for biocatalytic microreactor configurations, in: K. V Gernaey, J.K. Huusom, R.B.T.C.A.C.E. Gani (Eds.), 12 International Symposium on Process Systems Engineering and 25 European Symposium on Computer Aided Process Engineering, Elsevier, 2015: pp. 1463–1468. https://doi.org/10.1016/B978-0-444-63577-8.50089-9.

[23] Y. Wang, H. Xu, D. Pasini, Multiscale isogeometric topology optimization for lattice materials, Computer Methods in Applied Mechanics and Engineering. 316 (2017) pp. 568–585. https://doi.org/10.1016/j.cma.2016.08.015.

[24] W. Zhang, L. Zhao, T. Gao, S. Cai, Topology optimization with closed B-splines and Boolean operations, Computer Methods in Applied Mechanics and Engineering. 315 (2017) pp. 652–670. https://doi.org/10.1016/j.cma.2016.11.015.

[25] K.S. Park, S.K. Youn, Topology optimization of shell structures using adaptive inner-front (AIF) level set method, Structural and Multidisciplinary Optimization. 36 (2008) pp. 43–58. https://doi.org/10.1007/s00158-007-0169-4.

[26] P. Kang, S.K. Youn, Isogeometric topology optimization of shell structures

using trimmed NURBS surfaces, Finite Elements in Analysis and Design. 120 (2016) pp. 18–40. https://doi.org/10.1016/j.finel.2016.06.003.

[27] A. Kilian, J. Ochsendorf, Particle-spring systems for structural form finding, Journal of the International Association for Shell and Spatial Structures. 46 (2005) pp. 77–84.

[28] L. Piegl, W. Tiller, The NURBS Book, Springer Berlin Heidelberg, Berlin, Heidelberg, 1997. ISBN 978-3-540-61545-3. https://doi.org/10.1007/978-3-642-59223-2.

[29] S. Natsupakpong, M. Cenk Çavuşoğlu, Determination of elasticity parameters in lumped element (mass-spring) models of deformable objects, Graphical Models. 72 (2010) pp. 61–73. https://doi.org/10.1016/j.gmod.2010.10.001.

[30] T. Liu, A.W. Bargteil, J.F. O’Brien, L. Kavan, Fast simulation of mass-spring systems, ACM Transactions on Graphics. 32 (2013) pp. 1–7. https://doi.org/10.1145/2508363.2508406.

[31] V. Bulatov, W. Cai, Computer Simulations of Dislocations, Oxford University Press, 2006. ISBN 9780198526148. https://doi.org/10.1093/oso/9780198526148.001.0001.

[32] H. Wang, J. Chen, T. Nagayama, Parameter identification of spring-mass-damper model for bouncing people, Journal of Sound and Vibration. 456 (2019) pp. 13–29. https://doi.org/10.1016/j.jsv.2019.05.034.

[33] S. Dong, Z. Tang, X. Yang, M. Wu, J. Zhang, T. Zhu, S. Xiao, Nonlinear Spring-Mass-Damper Modeling and Parameter Estimation of Train Frontal Crash Using CLGAN Model, Shock and Vibration. 2020 (2020) pp. 1–19.

https://doi.org/10.1155/2020/9536915.

[34] D. Wu, C. Lv, Y. Bao, An improved vascular model based on mass spring model and parameters optimization by Gaussian processes, in: 2016 IEEE International Conference on Mechatronics and Automation, IEEE, 2016: pp. 2425–2430. https://doi.org/10.1109/ICMA.2016.7558946.

[35] G.V.V. Ravi Kumar, P. Srinivasan, K.G. Shastry, B.G. Prakash, Geometry based triangulation of multiple trimmed NURBS surfaces, Computer-Aided Design. 33 (2001) pp. 439–454. https://doi.org/10.1016/S0010-4485(00)00095-6.

[36] K. Hormann, K. Polthier, A. Sheffer, Mesh parameterization: Theory and practice, in: ACM SIGGRAPH ASIA 2008 Courses, SIGGRAPH Asia'08, ACM Press, New York, New York, USA, 2008: pp. 1–87. https://doi.org/10.1145/1508044.1508091.

[37] B. Lévy, S. Petitjean, N. Ray, J. Maillot, Least squares conformal maps for automatic texture atlas generation, ACM Transactions on Graphics. 21 (2002) pp. 362–371. https://doi.org/10.1145/566654.566590.

[38] M. de Berg, O. Cheong, M. van Kreveld, M. Overmars, Computational Geometry, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-77973-5. https://doi.org/10.1007/978-3-540-77974-2.

[39] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Levy, Polygon Mesh Processing, A K Peters/CRC Press, 2010. ISBN 9781439865316. https://doi.org/10.1201/b10688.

[40] S. Sastry, Nonlinear Systems, Springer New York, New York, NY, 1999. ISBN 978-1-4757-3108-8. https://doi.org/10.1007/978-1-4757-3108-8.

[41] S.H. Lo, A new mesh generation scheme for arbitrary planar domains, International Journal for Numerical Methods in Engineering. 21 (1985) pp. 1403–1426. https://doi.org/10.1002/nme.1620210805.

[42] P.M. Knupp, Algebraic Mesh Quality Metrics, SIAM Journal on Scientific Computing. 23 (2001) pp. 193–218. https://doi.org/10.1137/S1064827500371499.

[43] D.A. Field, Qualitative measures for initial meshes, International Journal for Numerical Methods in Engineering. 47 (2000) pp. 887–906. https://doi.org/10.1002/(SICI)1097-0207(20000210)47:4<887::AID-NME804>3.0.CO;2-H.

[44] C. Li, D. Lu, Study of intelligent layout design of single-layer lattice shell of free form surface, Tumu Gongcheng Xuebao/China Civil Engineering Journal. 44 (2011) pp. 1–7.

[45] J. Yan, F. Qin, Z. Cao, F. Fan, Y.L. Mo, Mechanism of coupled instability of single-layer reticulated domes, Engineering Structures. 114 (2016) pp. 158–170. https://doi.org/10.1016/j.engstruct.2016.02.005.

[46] L. Tian, J. Wei, Q. Huang, J.W. Ju, Collapse-Resistant Performance of Long-Span Single-Layer Spatial Grid Structures Subjected to Equivalent Sudden Joint Loads, Journal of Structural Engineering. 147 (2021) pp. 04020309. https://doi.org/10.1061/(ASCE)ST.1943-541X.0002904.

[47] L. Tian, J. He, C. Zhang, R. Bai, Progressive collapse resistance of single-layer latticed domes subjected to non-uniform snow loads, Journal of Constructional Steel Research. 176 (2021) pp. 106433.

https://doi.org/10.1016/j.jcsr.2020.106433.

[48] ANSYS R19.0, (2018), https://www.ansys.com. Last access: 5th September 2021.