Voronoi diagram and Monte-Carlo simulation based finite element optimization for cost-effective 3D printing

A.Z. Zheng[2*], S.J. Bian[2], E. Chaudhry[1], J. Chang[1], H. Haron[3], L.H. You[1], and J.J. Zhang[1]

[1] The National Center for Computer Animation, Bournemouth University, UK
[2] Humain Ltd, UK
[3] Department of Computer Science, Universiti Teknologi Malaysia, Malaysia

Abstract

By extending the work published at ICCS 2020 [1], in this paper we propose a method to achieve cost-effective 3D printing of stiffened thin-shell objects. Our proposed method consists of three parts. The first part integrates finite element analysis, Voronoi diagram, and conformal mapping to obtain stiffener distribution. The second part combines finite element analysis with optimization calculations to determine the optimal sizes of stiffeners. And the third part introduces Monte-Carlo simulation to find a global optimum. The experiments made in this paper indicate that our proposed method is effective in minimizing 3D printing material consumption of stiffened thin-shell objects.

Keywords: Stiffened objects, 3D printing, Voronoi diagram, finite element optimization, Monte-Carlo simulation

Abbreviations

FEA - Finite Element Analysis
2D - Two-Dimensional
3D - Three-Dimensional
DKT - Discrete Kirchhoff Triangle
FSDT - First-order Shear Deformation Theory
LSCMs - Least Squares Conformal Maps

1. Introduction

With the rise of low-cost 3D printers, 3D printing is revolutionising the way products are manufactured, not only in large manufacturers such as Boeing and General Electric but also with personal users and small businesses. As one of the fastest growing industries, it is delivering significant impacts to the manufacturing sector, global economy, and quality of life.

Many different materials have been used for 3D printing, including both non-metallic and metallic materials. The price of desktop 3D printers has become more affordable to general customers - it has been possible to buy a desktop 3D printer at a price of less than £100 at the present time. Nowadays, personal users can make 3D prints easily with these affordable printers at their home. Apart from a lot of personal users, 3D printing has been widely applied in various sectors. The wide range of applications highlight the importance of minimizing the cost of 3D printing while satisfying the requirements of specific applications.

In order to minimizing the cost of 3D printing, thin-shell objects have been widely used to reduce the consumption of 3D printing materials. Since thin-shell objects have low strength and stiffness, various methods have been proposed to enhance thin-shell objects. In this paper, we use stiffeners to stiffen thin-shell objects and propose a finite element optimization framework based on Voronoi diagram and Monte-Carlo simulation to obtain stiffened thin-shell objects with minimum material consumption and required strength.

Our proposed framework achieves minimum material consumption through optimizing stiffener distribution and minimizing the cross-section sizes of stiffeners. In order to generate an optimal distribution of stiffeners, the stress field of input thin-shell objects under given loads and boundary conditions is calculated with the Finite Element Analysis (FEA). According to the calculated stress field, some points called seeds are placed randomly on the three-dimensional (3D) surface of thin-shell objects. Conformal mapping is used to map the 3D objects and seeds to a 2D space so that a Voronoi diagram can be generated from these mapped seeds. The generated Voronoi diagram is mapped back to the 3D space and the edges of the mapped Voronoi diagram represent the distribution of stiffeners. After that, cross-section sizes of stiffeners are optimized to minimize the volume of the stiffeners. Since the generation of seeds uses a uniform random process, which may not lead to a global optimal solution of

stiffener distribution, Monte-Carlo simulation is introduced and iterated a given number of times to avoid any local minimum.

2. Related work

Our proposed framework is related to 3D printing, finite element analysis, structural optimization, Voronoi diagram, conformal maps, and Monte-Carlo simulation. We briefly review them in this section.

3D printing The research on 3D printing is massive. Various aspects of 3D printing have been investigated in existing work. For example, the deformation problem was investigated in [2], the articulation of 3D printed models was examined in [3], mechanical movements of 3D printed objects were studied in [4, 5], and the appearance of 3D printed models was discussed in [6, 7]. The aim of this paper is to minimize material consumption of 3D printed thin-shell objects.

Finite element analysis There are enormous publications about finite element analysis. For example, the finite element method in solid and structures was introduced in [8]. The finite element analysis of stiffened plates was given in [9]. The finite element calculations of stiffened shell were presented in [10]. The vibration of stiffened plates was investigated with the finite element method in [11]. Stress analysis of stiffened composited plates was carried out in [12]. The plates and shells with geometrically linear and nonlinear problems were studied in [13]. And mesh distortions of plate and shell finite elements were examined in [14]. In this paper, finite element analysis will be used to determine stress distributions in unstiffened and stiffened thin-shell objects and their stiffeners.

Structural optimization Various optimization methods have been developed and widely applied [15]. For example, tracking control of an underactuated system was optimized in [16], a new trajectory synthesis and optimization scheme was proposed in [17], probabilistic movement primitives were used to improve local trajectory optimization in [18], principles and progresses of optimization methods in machine learning were introduced in [19], and adaptive neural network tracking control were developed for underactuated systems in [20].

In the field of structural optimization, there are a lot of publications. Here we only briefly review some representative literature on optimization of 3D printed objects. Three approaches: hollowing, thickening, and strut insertion were introduced in [21] to obtain structurally sound and lightweight 3D prints. Thickness parameters of shells were optimized in [22]. The number of struts in a skin-frame structure is minimized in [23]. The material consumption of honeycomb-like 3D models is reduced via a hollowing optimization algorithm in [24]. Stiffened objects were first investigated in [25]. A method to produce optimized structures for any input surface with any load configurations was examined in [26]. Structural optimization will be used in this paper to optimize stiffener distribution and stiffener sizes of stiffened thin-shell objects.

Voronoi diagrams Extensive research has been carried out about Voronoi diagrams and their applications. Applications and algorithms of centroidal Voronoi diagrams were discussed in [27]. The Voronoi diagram for graphs was used in [28] to analyse the structure of biological networks. Using graph Voronoi diagrams, a new geometric approach to graph community detection was proposed in [29]. Some new methods of constructing Voronoi diagrams were proposed in [30]. Based upon statistics with mean vector and covariance matrix, a Voronoi diagram was proposed in [31]. A Voronoi diagram was constructed in [32] to form a cloaked region and calculate the anchor point of the cloaked region for privacy preservation. A window-vertex-sorted triangle propagation algorithm was proposed in [33] to construct geodesic based Voronoi diagrams. With Voronoi diagrams, interactive design and manufacturing of a biomimetic bone scaffold was investigated in [34]. Using Voronoi diagrams to generate the centerlines of watercourses was presented in [35]. An algorithm called hexagon-based crystal growth was presented in [36] to extract generalized Voronoi diagrams from hexagonal grids. Through Voronoi diagrams, design and statistical analysis of irregular porous scaffolds for orthopedic reconstruction was examined in [37]. A new parametric method of designing Voronoi-based lattice porous structures was proposed in [38]. In this paper, we will use Fortune's algorithm to create Voronoi diagrams from the generated seeds.

Conformal maps A number of studies have investigated conformal maps. Relying on certain conformal mappings, an explicit method was presented in [39] to map any simply connected surface onto a sphere in a manner of preserving angles. First order finite difference approximations of Cauchy-Riemann equations were used in [40] for conformal maps. An efficient circle pattern algorithm was developed in [41] for discrete conformal mappings. Based on complex Hilbert barycentric coordinates, a new method was presented in [42] to compute C^{∞} conformal mappings. A weighted combination of conformal maps was used in [43] to generate candidate maps between two genus-0 non-isometric shapes. Extremal quasiconformal mappings were specifically designed in [44] to produce injective mappings with minimal amount of conformal distortion. A framework was given in [45] to calculate harmonic and conformal mappings in the plane with some mathematical guarantees. A fast iterative algorithm was developed in [46] to produce conformal maps between two simply connected planar domains without prescribing boundary correspondence. Not using a triangle mesh, regular polygon meshes of equilateral triangles, squares and hexagons were used in [47] to approximate continuous conformal maps. Conformal mapping will be applied in this paper to achieve the mapping between 3D surface points and 2D parametric points.

Monte-Carlo simulation Massive publications have discussed Monte-Carlo simulations and its applications. For example, Monte Carlo simulations were used in [48] to calculate the solubility of natural gas components in ionic liquids and Selexol. A Monte-Carlo method based on the Cauchy-Crofton formula from integral geometry was presented in [49] to compute hypersurface areas of n-ellipsoids. Monte-Carlo simulation techniques were discussed in [50]. It includes the methods such as direct inversion, rejection method, and Markov chain Monte Carlo to sample a probability distribution function. In addition, it also contains the methods for variance reduction to evaluate numerical integrals using the Monte Carlo simulation. Grid-free Monte Carlo methods were used in [51] to solve core problems in PDE-based geometry processing efficiently and reliably. A Monte Carlo simulation model was developed in [52] to represent the COVID-19 spread dynamics. The Behler-Parrinello neural networks was introduced in [53] as an effective Hamiltonian used in the self-learning Monte Carlo method. Monte-Carlo methods were applied to modelling important probabilistic influences on motorsport races in [54]. Monte-Carlo simulation will be introduced in this paper for global optimization of stiffener distributions.

The remaining parts of this paper are organized below. An algorithm overview is given in Section 3. The finite element formulation is presented in Section 4. The stiffener distribution is examined in Section 5. The size optimization of stiffeners is investigated in Section 6. Monte-Carlo simulation is carried out in Section 7. Experiments and results are given in Section 8. And Conclusion and future work are presented in Section 9.

3 Algorithm overview

Our proposed algorithm consists of three parts: stiffener distribution, size optimization of stiffeners, and Monte-Carlo simulation. Since stiffener distribution is based on random seed generation of Voronoi diagram guided by the calculated stress field, the obtained stiffener distribution and subsequent size optimization may not give a gobal optimum. In order to tackle this problem, Monte-Carlo simulation is introduced.

As shown in Fig. 1, the algorithm starts from the first iteration of Monte-Carlo simulation. The finite element calculation of the thin-shell object to be stiffened is conducted to find the stress field in the object (Fig. 1(a)). According to the obtained stress field, Voronoi diagram seeds are randomly dispersed in the high stress regions (Fig. 1(b)). We define high stress regions as those with $s_i > p^* \sigma_s$. Here, s_i is the stress in the i^{th} triangle of the model to be 3D printed, p^* is the probability threshold, and σ_s is the material strength. They are elaborated in Subsection 5.1. The surface of the object is represented with a triangle mesh. The generated seeds are mapped to a 2D parametric domain. And a Voronoi diagram is created from the generated seeds, which have been mapped to the 2D parametric domain (Fig. 1(c)). Then, the intersecting points between the edges of Voronoi diagram and the edges of triangles of the object are found and mapped back to the 3D surface to determine the stiffener distribution (Fig. 1(d)). After that, the finite element optimization is carried out to find the optimal cross-section sizes of stiffeners. By doing so, the total volume of all stiffeners is minimized, and strength requirements of both thin shell and stiffeners are satisfied (Fig. 1(c)). Next, the algorithm checks whether all the iterations of Monte-Carlo simulation have been completed. If yes, the algorithm stops. Otherwise, next iteration of Monte-Carlo simulation begins.

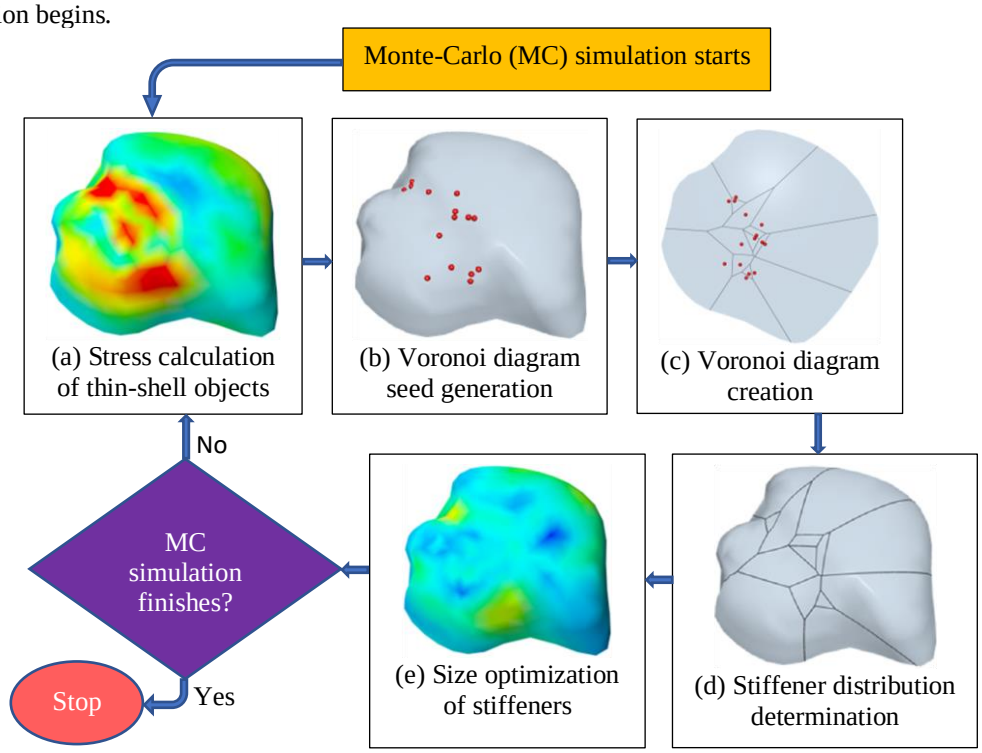


Fig. 1. Algorithm overview.

4. Finite element formulation

Since 3D printed objects may have both curved surfaces and flat surfaces, we follow the method given in [8] by Zienkiewicz and Taylor, which treats curved shells as an assembly of flat elements called flat shell elements. Such a treatment makes the method applicable to both flat plates and curved shells. The stiffeners are treated as beam elements. In order to ensure the stiffeners and shells to have the same deformations at their junctions, the same displacement functions are used for both flat shell elements and beam elements. The flat shell elements are divided into two types: Allman's plane stress triangle, which tackles in-plane deformations, and Discrete Kirchhoff triangle (DKT), which deals with lateral bending deformations. Beam elements can be placed anywhere within a shell element with arbitrary orientations. The von Karman's large deflection theory is used to address the large deflection problem, and geometric nonlinearity is solved with an iterative solution procedure. For the sake of completeness, we introduce this finite element analysis in the subsections below.

4.1 Kinematic equations

A typical flat shell element subjected to in-plane and lateral forces will have in-plane and bending deformations. Here "in-plane" means that the forces and deformations are in the plane of the flat shell element, and lateral forces and bending deformations are perpendicular to the plane.

We introduce the kinematics equations described in [55] by Neuyen-Van et al. and [13] by Cui et al. A local coordinate system x, y and z is used to indicate the directions of length, width, and thickness of a flat shell element, respectively. According to the first-order shear deformation theory (FSDT), shell kinematics is governed by mid-plane displacements $u_0 \quad v_0 \quad w_0$ and rotations θ_x and θ_y. Since the middle plane is parallel to the $x-y$ coordinate plane, all displacements and rotations at any points in the mid-plane are the functions of x and y only, i. e., $\boldsymbol{u}_0(x,y) = [u_0(x,y) \quad v_0(x,y) \quad w_0(x,y)]^T$, $\theta_x = \theta_x(x,y)$ and $\theta_y = \theta_y(x,y)$. The displacements at other points of the flat shell element are the functions of x, y and z, i. e., $\boldsymbol{u}(x,y,z) = [u(x,y,z) \quad v(x,y,z) \quad w(x,y,z)]^T$, which can be expressed as

$$\boldsymbol{u}(x,y,z) = \begin{Bmatrix} u_0(x,y) + z\theta_x(x,y) \\ v_0(x,y) + z\theta_y(x,y) \\ w_0(x,y) \end{Bmatrix} \tag{1}$$

where $u_0(x,y)$, $v_0(x,y)$ and $w_0(x,y)$ are the displacements in the mid-plane of the flat shell element in the x and y directions, and θ_x and θ_y are the rotations about the y and x axes, respectively.

Considering the von Kármán's large deflection assumption, the strains $\boldsymbol{\varepsilon} = [\varepsilon_x \quad \varepsilon_y \quad \varepsilon_{xy} \quad \varepsilon_{xz} \quad \varepsilon_{yz}]^T$ determined by the displacements (1) can be written as

$$\boldsymbol{\varepsilon} = \left[u_{,x} + 0.5w_{,x}^2 \quad v_{,y} + 0.5w_{,y}^2 \quad u_{,y} + v_{,x} + w_{,x}w_{,y} \quad \theta_x - w_{,x} \quad \theta_y - w_{,y}\right]^T \tag{2}$$

where $()_{,x} = \partial()/\partial x$ and $()_{,y} = \partial()/\partial y$.

Introducing Equation (1) into (2), the strains are changed into the following form

$$\begin{aligned} \boldsymbol{\varepsilon} = \big[&u_{0,x} + z\theta_{x,x} + 0.5w_{,x}^2 \quad v_{0,y} + z\theta_{y,y} + 0.5w_{,y}^2 \\ &u_{0,y} + v_{0,x} + z\theta_{x,y} + z\theta_{y,x} + w_{,x}w_{,y} \quad \theta_x - w_{,x} \quad \theta_y - w_{,y}\big]^T \end{aligned} \tag{3}$$

The strains $\boldsymbol{\varepsilon}$ can be divided into membrane strains $\boldsymbol{\varepsilon}_m$, bending strains $\boldsymbol{\varepsilon}_b$ and shear strains $\boldsymbol{\varepsilon}_s$. The membrane strains contain both linear and nonlinear parts. If we use $\boldsymbol{\varepsilon}_m^L$ to indicate the linear part, and $\boldsymbol{\varepsilon}_m^{NL}$ to denote the nonlinear part, the membrane strains $\boldsymbol{\varepsilon}_m$ can be written as

$$\boldsymbol{\varepsilon}_m = \boldsymbol{\varepsilon}_m^L + \boldsymbol{\varepsilon}_m^{NL} \tag{4}$$

where

$$\begin{aligned} \boldsymbol{\varepsilon}_m^L &= [u_{0,x} \quad v_{0,y} \quad u_{0,y} + v_{0,x}]^T \\ \boldsymbol{\varepsilon}_m^{NL} &= \left[0.5w_{,x}^2 \quad 0.5w_{,y}^2 \quad w_{,x}w_{,y}\right]^T = 0.5\boldsymbol{H}\boldsymbol{\theta} \end{aligned} \tag{5}$$

Comparing equations (3) and (5), the matrix $\boldsymbol{H}$ and the vector $\boldsymbol{\theta}$ are found to be

$$\begin{aligned} \boldsymbol{H} &= \begin{bmatrix} w_{,x} & 0 & w_{,y} \\ 0 & w_{,y} & w_{,x} \end{bmatrix}^T \\ \boldsymbol{\theta} &= [w_{,x} \quad w_{,y}]^T \end{aligned} \tag{6}$$

The bending strains $\boldsymbol{\varepsilon}_b$ and shear strains $\boldsymbol{\varepsilon}_s$ are determined by the following equations

$$\begin{aligned} \boldsymbol{\varepsilon}_b &= [\theta_{x,x} \quad \theta_{y,y} \quad \theta_{x,y} + \theta_{y,x}]^T \\ \boldsymbol{\varepsilon}_s &= [\theta_x - w_{,x} \quad \theta_y - w_{,y}]^T \end{aligned} \tag{7}$$

Comparing Equation (6) with (5) and (7), the strains $\boldsymbol{\varepsilon}$ are changed into

$$\boldsymbol{\varepsilon} = [\boldsymbol{\varepsilon}_m^L + \boldsymbol{\varepsilon}_m^{NL} + z\boldsymbol{\varepsilon}_b \quad \boldsymbol{\varepsilon}_s]^T \tag{8}$$

If we define a generalized strain vector $\bar{\boldsymbol{\varepsilon}} = [\boldsymbol{\varepsilon}_m \quad \boldsymbol{\varepsilon}_b \quad \boldsymbol{\varepsilon}_s]^T$ and a generalized stress vector $\bar{\boldsymbol{\sigma}} = [\boldsymbol{N} \quad \boldsymbol{M} \quad \boldsymbol{Q}]^T$ with in-plane forces $\boldsymbol{N} = [N_x \quad N_y \quad N_{xy}]^T$, in-plane bending moments $\boldsymbol{M} = [M_x \quad M_y \quad M_{xy}]^T$and in-plane shear forces $\boldsymbol{Q} = [Q_x \quad Q_y]^T$, the constitutive relationship can be formulated as

$$\bar{\boldsymbol{\sigma}} = \bar{\boldsymbol{D}}\bar{\boldsymbol{\varepsilon}} \tag{9}$$

The stiffness matrix $\bar{\boldsymbol{D}}$ in the above equation consists of the extensional stiffness $\bar{\boldsymbol{D}}_m$, the bending stiffness $\bar{\boldsymbol{D}}_b$, and the transverse shear stiffness $\bar{\boldsymbol{D}}_s$, i. e.,

$$\bar{\boldsymbol{D}} = \begin{bmatrix} \bar{\boldsymbol{D}}_m & 0 & 0 \\ 0 & \bar{\boldsymbol{D}}_b & 0 \\ 0 & 0 & \bar{\boldsymbol{D}}_s \end{bmatrix} \tag{10}$$

The extensional stiffness $\bar{\boldsymbol{D}}_m$, the bending stiffness $\bar{\boldsymbol{D}}_b$ and the transverse shear stiffness are determined by

$$\begin{aligned} \bar{\boldsymbol{D}}_m &= \frac{Eh}{(1-\nu^2)}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & (1-\nu)/2 \end{bmatrix} \\ \bar{\boldsymbol{D}}_b &= \frac{h^2}{12}\bar{\boldsymbol{D}}_m \\ \bar{\boldsymbol{D}}_s &= \frac{\kappa Eh}{2(1+\nu)}\boldsymbol{I}_2 \end{aligned} \tag{11}$$

where h , shown in Fig. 2(b), is the thickness of the flat shell element, E is Young's modulus, ν is Poisson's ratio, κ is the shear correction factor [14], and $\boldsymbol{I}_2$ is a 2×2 identity matrix.

4.2 Formulation of flat shell elements

In order to carry out the finite element analysis, we first discretize a 3D model into n_e flat shell finite elements and define generalized displacements $\bar{\boldsymbol{u}} = [u \quad v \quad w \quad \theta_x \quad \theta_y]^T$. If each element has n_p nodes, the nodal displacements $\boldsymbol{q}_n$ at the node n are $\boldsymbol{q}_n = [u_n \quad v_n \quad w_n \quad \theta_{xn} \quad \theta_{yn}]^T = [\boldsymbol{q}_{mn} \quad \boldsymbol{q}_{bn}]^T$ where $\boldsymbol{q}_{mn} = [u_n \quad v_n]^T$ and $\boldsymbol{q}_{bn} = [w_n \quad \theta_{xn} \quad \theta_{yn}]^T$. The displacements at any point of the element are connected to the nodal displacements by

$$\bar{\boldsymbol{u}} = \sum_{n=1}^{n_p} \boldsymbol{N}_n \boldsymbol{q}_n = \sum_{n=1}^{n_p} N_n \boldsymbol{I}_5 \boldsymbol{q}_n \tag{12}$$

where $\boldsymbol{N}_n$ is the matrix of shape functions N_n, and $\boldsymbol{I}_5$ is a 5×5 identity matrix.

Substituting the above equation into Equations (5) and (7), the following strain-displacement equations are obtained

$$\begin{aligned} \boldsymbol{\varepsilon}_m^L &= \sum_{n=1}^{n_p} \boldsymbol{B}_{mn}^L \boldsymbol{q}_{mn} \\ \boldsymbol{\varepsilon}_b &= \sum_{n=1}^{n_p} \boldsymbol{B}_{bn} \boldsymbol{q}_{bn} \\ \boldsymbol{\varepsilon}_s &= \sum_{n=1}^{n_p} \boldsymbol{B}_{sn} \boldsymbol{q}_{bn} \\ \boldsymbol{\varepsilon}_m^{NL} &= 0.5 \sum_{n=1}^{n_p} \boldsymbol{B}_{mn}^{NL} \boldsymbol{q}_{bn} = 0.5 \sum_{n=1}^{n_p} \boldsymbol{H}\boldsymbol{G}_n \boldsymbol{q}_{bn} \end{aligned} \tag{13}$$

where

$$\begin{aligned} \boldsymbol{B}_{mn}^L &= \begin{bmatrix} \boldsymbol{N}_{n,x} & 0 & \boldsymbol{N}_{n,y} \\ 0 & \boldsymbol{N}_{n,y} & \boldsymbol{N}_{n,x} \end{bmatrix}^T \\ \mathbf{G}_n &= \begin{bmatrix} \boldsymbol{N}_{n,x} & 0 & 0 \\ \boldsymbol{N}_{n,y} & 0 & 0 \end{bmatrix} \\ \boldsymbol{B}_{bn} &= \begin{bmatrix} 0 & \boldsymbol{N}_{n,x} & 0 \\ 0 & 0 & \boldsymbol{N}_{n,y} \\ 0 & \boldsymbol{N}_{n,y} & \boldsymbol{N}_{n,x} \end{bmatrix} \\ \mathbf{B}_{sn} &= \begin{bmatrix} -\boldsymbol{N}_{n,x} & \boldsymbol{N}_n & 0 \\ -\boldsymbol{N}_{n,y} & 0 & \boldsymbol{N}_n \end{bmatrix} \end{aligned} \tag{14}$$

We put the nodal displacements $\boldsymbol{q}_n$ for all nodes in a vector, i. e., $\boldsymbol{q} = [\cdots \quad \boldsymbol{q}_n \quad \cdots]^T$. Accordingly, the matrices in equations (13) and (14) are written as $\boldsymbol{B}_m^L = [\cdots \quad \boldsymbol{B}_{mn}^L \quad \cdots]^T$, $\boldsymbol{B}_m^{NL} = [\cdots \quad \boldsymbol{B}_{mn}^{NL} \quad \cdots]^T$, $\boldsymbol{G} = [\cdots \quad \boldsymbol{G}_n \quad \cdots]^T$, $\boldsymbol{B}_b = [\cdots \quad \boldsymbol{B}_{bn} \quad \cdots]^T$, and $\boldsymbol{B}_s = [\cdots \quad \boldsymbol{B}_{sn} \quad \cdots]^T$.

The finite element equations are derived from the weak form, i. e., the principle of virtual work. This principle states that the internal virtual work is equal to the external virtual work. With reference to the undeformed shell configuration in the total Lagrangian description, the principle of virtual work can be written as [55]

$$\Pi = \int_V (\boldsymbol{N}^T \delta\boldsymbol{\varepsilon}_m + \boldsymbol{M}^T \delta\boldsymbol{\varepsilon}_b + \boldsymbol{Q}^T \delta\boldsymbol{\varepsilon}_s)\, dV - \boldsymbol{f}^T \delta\boldsymbol{q} \tag{15}$$

where Π is the total potential energy in the domain, $\int_V (\boldsymbol{N}^T \delta\boldsymbol{\varepsilon}_m + \boldsymbol{M}^T \delta\boldsymbol{\varepsilon}_b + \boldsymbol{Q}^T \delta\boldsymbol{\varepsilon}_s)\, dV$ is the internal virtual work, $\boldsymbol{f}^T \delta\boldsymbol{q}$ is the external virtual work, and $\boldsymbol{f}^T$ is the nodal forces.

In order to calculate the total potential energy, we must first calculate the variations of strain components. This can be obtained below from Equations (4) and (13)

$$\delta\boldsymbol{\varepsilon}_m = \delta(\boldsymbol{\varepsilon}_m^L + \boldsymbol{\varepsilon}_m^{NL}) = (\boldsymbol{B}_m^L + \boldsymbol{B}_m^{NL})\delta\boldsymbol{q}$$

$$\delta\boldsymbol{\varepsilon}_b = \boldsymbol{B}_b\delta\boldsymbol{q}$$
$$\delta\boldsymbol{\varepsilon}_s = \boldsymbol{B}_s\delta\boldsymbol{q} \quad (16)$$

Substituting Equation (16) into (15), the equation for the principle of virtual work becomes

$$\Pi = \int_V [(\boldsymbol{N}^T(\boldsymbol{B}_m^L + \boldsymbol{B}_m^{NL}) + \boldsymbol{M}^T\boldsymbol{B}_b + \boldsymbol{Q}^T\boldsymbol{B}_s)dV - \boldsymbol{f}^T]\,\delta\boldsymbol{q} \quad (17)$$

Equation (17) defines a geometric nonlinearity problem, which can be solved with a Newton-Raphson iteration procedure. To do this, Equation (17) is first linearized through

$$\delta\Pi = \delta\boldsymbol{q}\{\int_V [(\boldsymbol{B}_m^L + \boldsymbol{B}_m^{NL})^T\delta\boldsymbol{N} + (\delta\boldsymbol{B}_m^{NL})^T\boldsymbol{N} + \boldsymbol{B}_b^T\delta\boldsymbol{M} + \boldsymbol{B}_s^T\delta\boldsymbol{Q}]dV - \boldsymbol{f}^T\} = 0 \quad (18)$$

The increments of the in-plane forces, in-plane bending moments, and in-plane shear forces $\delta\boldsymbol{N}$, $\delta\boldsymbol{M}$ and $\delta\boldsymbol{Q}$ can be obtained from Equations (9), (10) and (11)

$$\delta\boldsymbol{N} = \bar{\boldsymbol{D}}_m(\boldsymbol{B}_m^L + \boldsymbol{B}_m^{NL})\delta\boldsymbol{q}$$
$$\delta\boldsymbol{M} = \bar{\boldsymbol{D}}_b\boldsymbol{B}_b\delta\boldsymbol{q}$$
$$\delta\boldsymbol{Q} = \bar{\boldsymbol{D}}_s\boldsymbol{B}_s\delta\boldsymbol{q} \quad (19)$$

The integrand in the term $\int_V(\delta\boldsymbol{B}_m^{NL})^T\boldsymbol{N}dV$ can be calculated below by considering Equations (5) and (6)

$$(\delta\boldsymbol{B}_m^{NL})^T\boldsymbol{N} = \boldsymbol{G}^T\delta\boldsymbol{H}^T\boldsymbol{N} = \boldsymbol{G}^T\boldsymbol{B}_g\boldsymbol{G} \quad (20)$$

where

$$\boldsymbol{B}_g = \begin{bmatrix} N_x & N_{xy} \\ N_{xy} & N_y \end{bmatrix} \quad (21)$$

Substituting Equations (19) and (21) into equation (18), the integration of equation (18) leads to a tangent stiffness matrix below

$$\boldsymbol{K} = \boldsymbol{K}^L + \boldsymbol{K}^{NL} + \boldsymbol{K}^G \quad (22)$$

where $\boldsymbol{K}^L$ is the linear stiffness matrix, $\boldsymbol{K}^{NL}$ is the nonlinear stiffness matrix, and $\boldsymbol{K}^G$ is the geometric stiffness matrix.

If the transformation matrix from the local coordinate of a flat shell element to the global coordinate is $\boldsymbol{T}$, the three stiffness matrices $\boldsymbol{K}^L$, $\boldsymbol{K}^{NL}$, and $\boldsymbol{K}^G$ are determined with the following equations [10]

$$\boldsymbol{K}^L = \int_V \boldsymbol{T}^T(\boldsymbol{B}_m^L)^T\boldsymbol{D}_m\boldsymbol{B}_m^L\boldsymbol{T}dV$$
$$\boldsymbol{K}^{NL} = \int_V \boldsymbol{T}^T[(\boldsymbol{B}_m^L)^T\boldsymbol{D}_m\boldsymbol{B}_m^{NL} + (\boldsymbol{B}_m^{NL})^T\boldsymbol{D}_m\boldsymbol{B}_m^L + (\boldsymbol{B}_m^{NL})^T\mathbf{D}_m\boldsymbol{B}_m^{NL} + \boldsymbol{B}_b^T\bar{\boldsymbol{D}}_b\boldsymbol{B}_b + \boldsymbol{B}_s^T\bar{\boldsymbol{D}}_s\boldsymbol{B}_s]\boldsymbol{T}dV$$
$$\boldsymbol{K}^G = \int_V \boldsymbol{T}^T\boldsymbol{G}^T\boldsymbol{B}_g\boldsymbol{G}\boldsymbol{T}dV \quad (23)$$

4.3 Formulation of stiffener elements

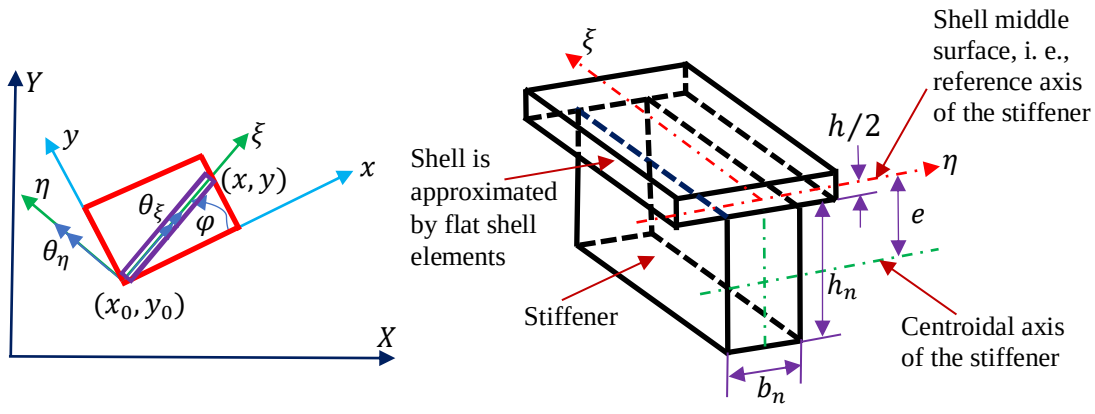
In order to consider the contribution of stiffeners to the shell, we present the finite element equations of stiffeners in this subsection.

As discussed in [12], the displacement field of the stiffener in the skew axes system ξ and η shown in Fig. 2(a) is defined by three translations and two rotations, i. e.,

$$\bar{\boldsymbol{u}}^s = \begin{bmatrix} u^s & v^s & w^s & \theta_\xi^s & \theta_\eta^s \end{bmatrix}^T \quad (24)$$

Taking the middle plane of the shell as a reference plane for the analysis of stiffeners, the relationship between the local coordinates ξ and η of stiffeners is connected to the local coordinates x and y of the flat shell element through

$$\xi = (x - x_0)\cos\varphi + (y - y_0)\sin\varphi$$
$$\eta = -(x - x_0)\sin\varphi + (y - y_0)\cos\varphi \quad (25)$$



(a) (b)

Fig. 2: Stiffener: (a) local coordinate system, (b) reference axis of the stiffener.

Solving equations (25) for x and y, the following equations are obtained

$$x = x_0 + \xi \cos\varphi - \eta \sin\varphi$$
$$y = y_0 + \xi \sin\varphi - \eta \cos\varphi \quad (26)$$

The deformation compatibility between the flat shell elements and stiffener elements requires the stiffeners to have the same displacements as the flat shell elements. This can be guaranteed by taking the shape functions of stiffener elements to be the same as those of flat shell elements, which leads to the following relationship between the local displacements of stiffeners and local displacements of flat shell elements

$$u^s = u\cos\varphi + v\sin\varphi$$
$$v^s = -u\sin\varphi + v\cos\varphi$$
$$w^s = w$$
$$\theta_\xi^s = \theta_x\cos\varphi + \theta_y\sin\varphi$$
$$\theta_\eta^s = -\theta_x\sin\varphi + \theta_y\cos\varphi \quad (27)$$

where u, v, w are displacements in the middle plane of the flat shell elements, and θ_x and θ_y of the flat shell elements are the rotations around the x and y axes, respectively.

The strains $\boldsymbol{\varepsilon}_s$ in stiffeners can be divided into linear strains $\boldsymbol{\varepsilon}_s^L$ and nonlinear strains $\boldsymbol{\varepsilon}_s^{NL}$, i. e.,

$$\boldsymbol{\varepsilon}^s = \boldsymbol{\varepsilon}_s^L + \boldsymbol{\varepsilon}_s^{NL} \quad (28)$$

where

$$\boldsymbol{\varepsilon}_s^L = \begin{bmatrix} u_{,\xi}^s & \theta_{\xi,\xi}^s & \theta_\xi^s - w_{,\xi}^s & \theta_{\eta,\xi}^s \end{bmatrix}^T$$

$$\boldsymbol{\varepsilon}_s^{NL} = 0.5\begin{bmatrix} \left(w_{,\xi}^s\right)^2 & 0 & 0 & 0 \end{bmatrix}^T \quad (29)$$

Substituting Equation (27) into the first one of Equation (29) and following the derivation given in [11], the linear strains in the stiffeners are determined by

$$\boldsymbol{\varepsilon}_s^L = \boldsymbol{T}^s\boldsymbol{\varepsilon}^p \quad (30)$$

where $\boldsymbol{T}^s$ is the transformation matrix for the stiffeners, which correlates the strains in the local coordinate of stiffeners to the local coordinate of flat shell elements, and $\boldsymbol{\varepsilon}^p$ is the strains in the mid-plane of flat shell elements, which are

$$\boldsymbol{\varepsilon}^p = [\boldsymbol{\varepsilon}_m^L \quad \bar{\boldsymbol{\varepsilon}}_b \quad \boldsymbol{\varepsilon}_s]^T \quad (31)$$

In the above equations, $\boldsymbol{\varepsilon}_m^L$ and $\boldsymbol{\varepsilon}_s$ are determined by equations (5) and (7). $\bar{\boldsymbol{\varepsilon}}_b$ is similar to $\boldsymbol{\varepsilon}_b$ in equation (7), but the last element in the vector has been written in the two elements, i. e.,

$$\bar{\boldsymbol{\varepsilon}}_b = [\theta_{x,x} \quad \theta_{y,y} \quad \theta_{x,y} \quad \theta_{y,x}]^T \quad (32)$$

Since $\boldsymbol{\varepsilon}_b$ is changed into $\bar{\boldsymbol{\varepsilon}}_b$, the matrix $\boldsymbol{B}_{bn}$ is accordingly changed into $\bar{\boldsymbol{B}}_{bn}$ below

$$\bar{\boldsymbol{B}}_{bn} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \boldsymbol{N}_{n,x} & 0 & \boldsymbol{N}_{n,y} & 0 \\ 0 & \boldsymbol{N}_{n,y} & 0 & \boldsymbol{N}_{n,x} \end{bmatrix}^T \quad (33)$$

According to [9], the transformation matrix $\boldsymbol{T}^s$ has the form of

$$\boldsymbol{T}^s = [T_1 \quad T_2 \quad T_3 \quad T_4]^T \quad (34)$$

where the vectors T_1, T_2, T_3 and T_4 are determined by the following equations

$$T_1 = [\cos^2\varphi \quad \sin^2\varphi \quad 0.5\sin 2\varphi \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]$$
$$T_2 = [0 \quad 0 \quad 0 \quad \cos^2\varphi \quad \sin^2\varphi \quad 0.5\sin 2\varphi \quad 0.5\sin 2\varphi \quad 0 \quad 0]$$
$$T_3 = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad \cos\varphi \quad \cos\varphi]$$
$$T_4 = [0 \quad 0 \quad 0 \quad -0.5\sin 2\varphi \quad 0.5\sin 2\varphi \quad \cos^2\varphi \quad \sin^2\varphi \quad 0 \quad 0] \quad (35)$$

Introducing Equation (13) into (30), the strains in stiffeners are related to nodal displacements of flat shell elements by

$$\boldsymbol{\varepsilon}_s^L = \left[\sum_{n=1}^{n_p}\boldsymbol{T}^s\boldsymbol{B}_{mn}^L\boldsymbol{q}_{mn} \quad \sum_{n=1}^{n_p}\boldsymbol{T}^s\bar{\boldsymbol{B}}_{bn}\boldsymbol{q}_{bn} \quad \sum_{n=1}^{n_p}\boldsymbol{T}^s\boldsymbol{B}_{sn}\boldsymbol{q}_{bn}\right]^T \quad (36)$$

Using N_s, M_s, Q_s and T_s to indicate the axial force, bending moment, shear force, and torsion moment, respectively, the generalized stresses $\bar{\boldsymbol{\sigma}}^s$ in stiffeners can be calculated from the strains in stiffeners through

$$\bar{\boldsymbol{\sigma}}^s = [N_s \quad M_s \quad Q_s \quad T_s]^T = \bar{\boldsymbol{D}}^s\bar{\boldsymbol{\varepsilon}}_s^L \quad (37)$$

In the above equation, $\bar{\boldsymbol{D}}^s$ is the elasticity matrix of the stiffeners, which can be written as

$$\bar{\boldsymbol{D}}^s = [\bar{\boldsymbol{D}}_1^s \quad \bar{\boldsymbol{D}}_2^s \quad \bar{\boldsymbol{D}}_3^s \quad \bar{\boldsymbol{D}}_4^s]^T \quad (38)$$

The elements $\bar{\boldsymbol{D}}_1^s$, $\bar{\boldsymbol{D}}_2^s$, $\bar{\boldsymbol{D}}_3^s$ and $\bar{\boldsymbol{D}}_4^s$ of the elasticity matrix $\bar{\boldsymbol{D}}^s$ are calculated by

$$\bar{\boldsymbol{D}}_1^s = [E_sA_s \quad E_sS_s \quad 0 \quad 0]$$
$$\bar{\boldsymbol{D}}_2^s = [E_sS_s \quad E_sI_s \quad 0 \quad 0]$$
$$\bar{\boldsymbol{D}}_3^s = [0 \quad 0 \quad S_s \quad 0]$$
$$\bar{\boldsymbol{D}}_4^s = [0 \quad 0 \quad 0 \quad G_sJ_s] \quad (39)$$

where E_s is Young's modulus of the stiffener, A_s is the cross-sectional area of the stiffener, S_s is the first moment of the stiffener cross-sectional area about the reference axis, i. e., the mid-plane of the flat sheet element, I_s is the

second moment of the stiffener cross-sectional area about the reference axis, G_s is the modulus of rigidity, and J_s is the polar moment of inertia of the stiffener cross-sectional area.

Using the same method as that for the flat shell element, and considering the transformation matrix $\boldsymbol{T}$ from the local coordinate of a flat shell element to the global coordinate, the stiffness matrix of stiffeners from the strains defined by the first of Equation (29) can be obtained as

$$\boldsymbol{K}^s = \int_{L_s} \boldsymbol{T}^T \boldsymbol{B}_s^T \, (\boldsymbol{T}^s)^T \boldsymbol{D}_s \boldsymbol{T}^s \boldsymbol{B}_s \boldsymbol{T} d\xi \tag{40}$$

Taking advantage of the chain rule $w^s_{,\xi} = w^s_{,x} x_{,\xi} + w^s_{,y} y_{,\xi}$, the second of Equation (29) becomes

$$\boldsymbol{\varepsilon}_s^{NL} = 0.5 \boldsymbol{A}^s \boldsymbol{R}^s \tag{41}$$

where

$$\boldsymbol{R}^s = \sum_{n=1}^{n_p} \boldsymbol{G}_n^s \, \boldsymbol{q}_n \tag{42}$$

and

$$\boldsymbol{G}_n^s = [0 \quad 0 \quad 0 \quad N_n \cos\varphi \quad N_n \sin\varphi] \tag{43}$$

With the similar treatment to that for flat shell elements and considering the transformation matrix $\boldsymbol{T}$ from the local coordinate of a flat shell element to the global coordinate, the stiffness matrix of stiffeners from the nonlinear strain has the form of

$$\boldsymbol{K}^s = \int_{L_s} \boldsymbol{T}^T \boldsymbol{G}_s^T \boldsymbol{D}_{sN} \boldsymbol{G}_s \boldsymbol{T} d\xi \tag{44}$$

where the matrix $\boldsymbol{G}_s$ is obtained by assembling the vector $\boldsymbol{G}_n^s$.

After adding the stiffness matrices (40) and (43) of stiffeners to the stiffness matrix (22), we reach the following equation

$$\boldsymbol{K}\boldsymbol{q} = \boldsymbol{f} \tag{45}$$

We solve the above equation to obtain all the nodal placements and determine deformations and stresses in the shell. Then we use the relationships between the nodal displacements of stiffeners and the displacements of the shell to calculate the deformations and stresses in the stiffeners.

5. Determination of stiffener distribution

Determining stiffener distribution consists of 4 steps. 1) Generate seeds of Voronoi diagram. 2) Map the 3D surface of an object and generated seeds to a 2D parametric domain. 3) Create Voronoi diagram in the mapped 2D parametric domain. 4) Obtain stiffener distribution from the created Voronoi diagram and map the extracted stiffeners from the 2D parametric domain to the 3D surface.

5.1 Seed generation of Voronoi diagram

In order to stiffen a thin-shell object optimally, we first specify the total iterations of Monte-Carlo simulation. Then, the finite element analysis is used to determine the stress field of the object subjected to given external forces and boundary conditions. Clearly, the regions with high stress should be stiffened with more stiffeners. To this aim, a given number of seeds used to create Voronoi diagram are distributed on the object through a probability that places more seeds in the areas with a higher stress.

We use n_t to stand for the number of triangles of the object mesh, s_i for the stress of a randomly selected triangle t_i, σ_s for the material strength, n_s for the number of expected seeds, and p^* for the probability threshold. We also use $s_i \,/\, \sigma_s$ to calculate the ratio of the stress s_i in the i^{th} triangle over the material strength σ_s. Since bigger is the ratio $s_i \,/\, \sigma_s$, higher is the stress in the triangle. Therefore, we use the ratio $s_i \,/\, \sigma_s$ together with the probability threshold p^* to determine whether a Voronoi diagram seed is valid or not.

First, the number n_s of the expected seeds used to create a Voronoi diagram is specified. Next, a triangle t_i is randomly selected from the total n_t triangles of the object. And finally, a probability p is randomly generated between 0 and 1. If the randomly generated probability p is bigger than the probability threshold p^* but smaller than the ratio $s_i \,/\, \sigma_s$, the triangle is seeded and marked. After the randomly selected triangle t_i has been seeded and marked, a new triangle is randomly selected and a new probability p is randomly generated between 0 and 1. The condition $p^* < p < s_i \,/\, \sigma_s$ is used to check whether the new triangle should be seeded and marked. This process is repeated until the number n_s of the expected seeds are reached. The algorithm is shown below.

Algorithm 1: Seed generation

Input: stress field $\{s_i\}$, material strength σ_s, number of expected seeds n_s and probability threshold p^*

Output: markedFaces

```
count = 0
markedFaces[n_t] =false
while count ≤ n_s do
    t_i = UniformRandom(1, ···, n_t)
    p = UniformRandom(0, 1)
    if not markedfaces[t_i] and p* < p < s_i/σ_s then
        count++
        markedFaces[t_i] = true
    end
end
```

5.2 Mapping a 3D surface to a 2D domain

After generating Voronoi diagram seeds on the 3D surface of a thin-shell object, the next step is to create a Voronoi diagram from these seeds and extract stiffeners from the created Voronoi diagram. However, directly creating a 3D Voronoi diagram from the generated seeds and extracting stiffeners from the created 3D Voronoi diagram on a 3D surface is more difficult than in a two-dimensional (2D) domain. In order to overcome the difficulty, we use the Least Squares Conformal Maps (LSCMs) [39] to map a 3D surface and the generated seeds to a 2D domain and create a 2D Voronoi diagram in the 2D domain.

As adopted in [39], this paper uses normal characters to stand for scalars, bold characters for vectors, capital characters for complex numbers, bold capital characters for vector of complex numbers, and cursive fonts for maps and matrices. With these notations, x is a scalar, $\boldsymbol{x} = [x, y]^T$ is a vector, $X = x + iy$ is a complex number, $\boldsymbol{X} = [X, Y]^T$ is a vector whose components are complex numbers X and Y, and $\mathcal{X}$ is a map or matrix.

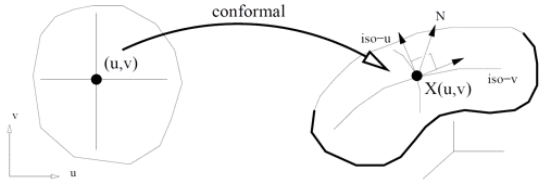


Fig. 3. Conformal mapping from a 2D point (u, v) to a surface point $X(u, v)$ [40].

In this paper, we use a conformal mapping to map a 3D surface and the generated seeds on the 3D surface to a 2D parametric domain. A conformal mapping, also called a conformal map, conformal transformation, angle-preserving transformation, or biholomorphic map, is a transformation that preserves local angles. Such local angle preservation enables to map an elementary circle in the (u, v) domain to an elementary circle on a surface. Therefore, a conformal mapping is also locally isotropic.

The mapping $\mathcal{X}(u, v)$ shown in Fig. 3 that maps a (u, v) domain to a surface is conformal since the tangent vectors to the iso-u and iso-v curves passing through $\mathcal{X}(u, v)$ are orthogonal and have the same norm for each (u, v). This property can be mathematically written as

$$N(u, v) \times \frac{\partial \mathcal{X}(u,v)}{\partial u} = \frac{\partial \mathcal{X}(u,v)}{\partial v} \tag{46}$$

where $N(u, v)$ is the unit normal to the surface.

For a thin-shell object represented with a triangle mesh consisting of n vertices and n_t triangles, we use $\mathcal{T}$ to stand for the set of n_t triangles and $\boldsymbol{p_j}$ $\{1 \le j \le n\}$ to denote the geometric location at vertex j of n vertices. For each of the n_t triangles, a local orthonormal basis is provided. The z-axis of the local orthonormal basis is in the normal direction of the triangle. And the coordinates of the three vertices of the triangle in its local orthonormal basis are $(x_1, y_1), (x_2, y_2)$, and (x_3, y_3). If two triangles share a edge, the local bases of two triangles are consistently oriented.

In the local orthonormal basis of triangle T with an area of A_T, the map $\mathcal{X}$: $(u,v) \rightarrow (x, y)$ can be written a complex function $\mathcal{X} = x(u, v) + iy(u, v)$ where the symbol i is an imaginary number. We use $\mathcal{U}$ to indicate the inverse map of $\mathcal{X}$. Similarly, the inverse map $\mathcal{U}$: $(x, y) \rightarrow (u, v)$ can also be written a complex function $\mathcal{U} = u(x, y) + iv(x, y)$. In the local orthonormal basis, Eq. (46) becomes

$$\frac{\partial \mathcal{X}}{\partial u} - i\frac{\partial \mathcal{X}}{\partial v} = 0 \tag{47}$$

According to the theorem on the derivatives of inverse functions, we obtain

$$\frac{\partial u}{\partial x} + i\frac{\partial u}{\partial y} = 0 \tag{48}$$

In general, Eq. (48) cannot be strictly enforced. The violation of the conformality condition can be minimized in the least squares sense, which defines the following criterion $C(T)$:

$$C(T) = \int_T \left|\frac{\partial u}{\partial x} + i\frac{\partial u}{\partial y}\right|^2 dA = \left|\frac{\partial u}{\partial x} + i\frac{\partial u}{\partial y}\right|^2 A_T \tag{49}$$

Summing over all the n_t triangles of the triangle mesh, the criterion to minimize the violation of the conformality condition can be written as

$$C(\mathcal{T}) = \sum_{T\in\mathcal{T}} C(T) \tag{50}$$

With the Least Squares Conformal Maps, the 3D surface of a thin-shell object to be stiffened and the generated seeds on the 3D surface are mapped to a 2D parametric domain.

5.3 Creating Voronoi diagram

Voronoi diagrams have widespread practical and theoretical applications in many fields. They are mainly applied in science and technology, but also in visual art including computational geometry, city planning, computer graphics, epidemiology, geophysics, and meteorology etc. Some application examples are: data compression in image processing, nearest neighbor queries for data structure problems in computational geometry, optimal quadrature rules, computational morphology such as modelling how fire spreads and crystals grow, optimal placement of resources, business applications such as determining where to locate a store so it is no closer to any existing store of its kind, finite difference schemes with optimal truncation errors, cell division, territorial behavior of animals, optimal representation, quantization and clustering, and applications of centroidal Voronoi tessellations in non-Euclidean metrics.

A Voronoi diagram is also called a Voronoi tessellation, a Voronoi decomposition, a Voronoi partition, or a Dirichlet tessellation. It is a partition of a plane into regions close to each of a given set of seeds (points, usually called sites). As shown in Fig. 4 below, if we are given a finite set of sites, i. e., points $\boldsymbol{P} = \{\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3, \ldots, \boldsymbol{p}_M\}$, the Voronoi diagram of $\boldsymbol{P}$ is the subdivision of the plane into M Voronoi cells $\boldsymbol{R}_i = \boldsymbol{R}(\boldsymbol{p}_i)$ (i=1, 2, 3,…, M) so that any point $\boldsymbol{p}$ lies in the cell $\boldsymbol{R}(\boldsymbol{s}_i)$ if $\|\boldsymbol{p} - \boldsymbol{p}_i\| < \|\boldsymbol{p} - \boldsymbol{p}_j\|$ for each $\boldsymbol{p}_j \in \boldsymbol{P}$ when $i \neq j$.

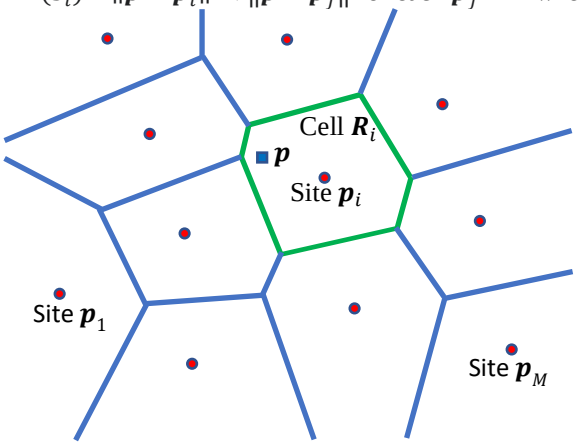


Fig. 4. Voronoi diagram.

With the seed generation algorithm described in Subsection 5.1, the seeds on the 3D mesh have been generated and shown in Fig. 8(a). These seeds are mapped to a 2D space with the mapping algorithm discussed in Subsection 5.2. The remaining problem is how to generate a Voronoi diagram from these seeds.

There are several different algorithms of generating Voronoi diagrams. The half plane intersection algorithm treats the edges of the Voronoi diagram as the segments taken from the perpendicular bisectors of the lines between the sites. These segments can be regarded as intersections of perpendicular bisectors, which divide the plane in half. The running time of the half plane intersection algorithm is $O(n^2 \log n)$. Fortune's algorithm constructs a Voronoi diagram as a horizontal line or vertical line sweeping the set of sites. The running time of Fortune's algorithm is $O(n \log n)$. Bowyer-Watson algorithm treats a Voronoi diagram as a dual graph of Delaunay triangulation. It generates a Voronoi diagram by connecting the centres of all the circumcircles of Delaunay triangulation. The running time of Bowyer-Watson algorithm is $O(n \log n)$ to $O(n^2)$. In this paper, we generate Voronoi diagrams by using Fortune's algorithm to sweep a horizontal line from top to bottom as discussed below.

Fortune's algorithm consists of two steps. The first step is to simulate the growth of the beach line as the sweep line moves downwards, and the second step is to trace the paths of the breakpoints as they travel along the edges of the Voronoi diagram.

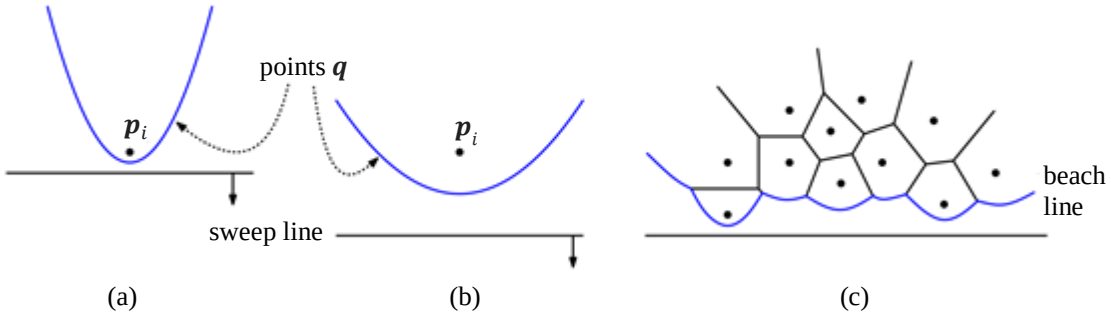


Fig. 5. The sweep line and beach line.

As shown in Fig. 5, the horizontal sweep line divides a plane into a top halfplane and a bottom halfplane, and the *x*-monotonic blue beach line divides the top halfplane into two regions. The points in the region above the blue beach line are closer to some site p_i above the sweep line than they are to the sweep line itself. And the points in the region below the beach line are closer to the sweep line than they are to any site above the sweep line.
The points q that are equidistant from the sweep line and the nearest site p_i above the sweep line is a parabola. When the sweep line passes through a new site, a new parabola is generated as shown in Fig. 5(a). When the sweep line moves downwards further, the parabola becomes "fatter" as shown in Fig. 5(b). A beach line consists of a lower envelope of these parabolas, one for each site, as shown in Fig. 5(c).

When two parabolas intersect, a breakpoint is generated. It is equidistant from two sites and the sweep line, and hence must lie on some Voronoi edge. For example, if two parabolas of sites p_i and p_j share a common breakpoint on the beach line, this breakpoint lies on the Voronoi edge between the sites p_i and p_j as shown in Fig. 6(a).

When the sweep line moves downwards, two events occur: one is site events, and the other is circle events also called Voronoi vertex events.

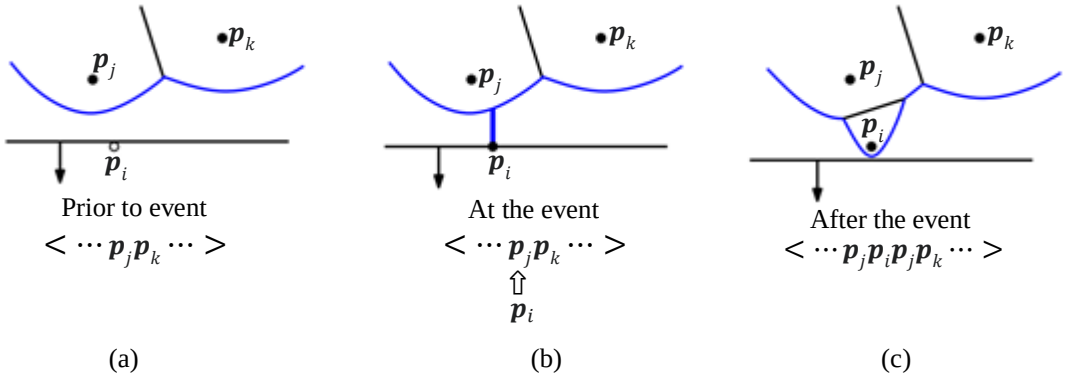
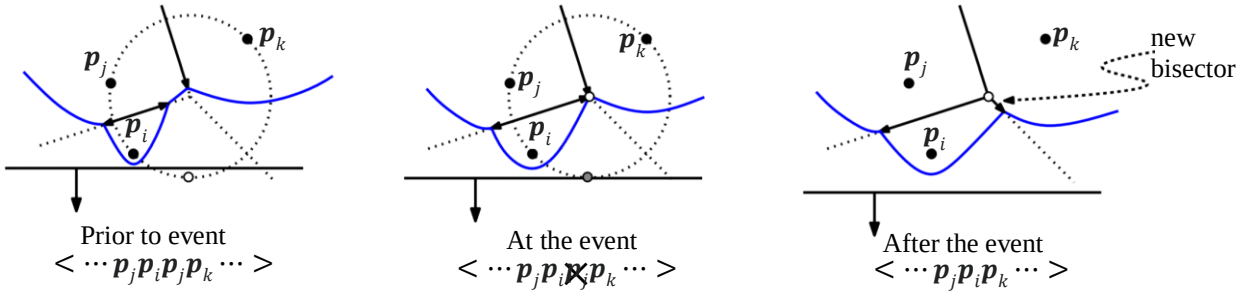


Fig. 6. Site event.

A site event occurs whenever the horizontal sweep line passes over a site. Here we use the sweep line passing through the site p_i to explain how a site event occurs. Before the sweep line touches the site p_i, the parabolic arc of the site p_j has been generated, and is connected to the parabolic arc of the site p_k. The sweep line status can be described with a list <…, p_j, p_k, …> as shown in Fig. 6(a). At the instant that the sweep line touches the site p_i, the associated parabolic arc of p_i degenerates to a vertical ray shooting up from the site p_i to the parabolic arc of the site p_j, and a new event p_i will be added to the list <…, p_j, p_k, …> as shown in Fig. 6(b). As the sweep line proceeds downwards, this ray widens into a parabolic arc along the beach line. As the sweep line sweeps on, the parabolic arc of the site p_i grows wider. The parabolic arc of the site p_j is split into two: one is on the left and the other is on the right of the parabolic arc of the site p_i, and the new event p_i is added to list <…, p_j, p_k, …> to change it into <…, p_j, p_i, p_j, p_k, …> as shown in Fig. 6(c).

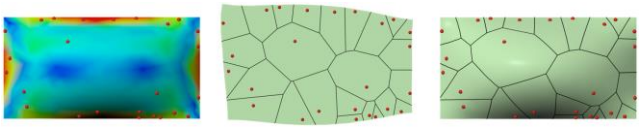


(a) (b) (c)

Fig. 7. Circle event.

Circle events are generated from triples of sites. We use Fig. 7 to explain how a circle event is generated. As shown in Fig. 7(a), any three consecutive sites $\boldsymbol{p}_i$, $\boldsymbol{p}_j$, and $\boldsymbol{p}_k$ define a circumcircle, and the small hollow circle is the lowest point of the circumcircle. The circumcircle contains no sites lying below the sweep line. When the sweep line is above the lowest point, the beach line contains the left parabolic arc of the site $\boldsymbol{p}_j$, the parabolic arc of the site $\boldsymbol{p}_i$, the right parabolic arc of the site $\boldsymbol{p}_j$, and the parabolic arc of the site $\boldsymbol{p}_k$, which can be described by the list <…, $\boldsymbol{p}_j$, $\boldsymbol{p}_i$, $\boldsymbol{p}_j$, $\boldsymbol{p}_k$, …>. At the instant when the sweep line falls to the lowest point, the circumcenter of the circumcircle is equidistant from all three sites $\boldsymbol{p}_i$, $\boldsymbol{p}_j$, and $\boldsymbol{p}_k$ and from the sweep line. Therefore, it is a Voronoi vertex. Since all three parabolic arcs of the sites $\boldsymbol{p}_i$, $\boldsymbol{p}_j$, and $\boldsymbol{p}_k$ pass through this circumcenter, the contribution of the parabolic arc from $\boldsymbol{p}_j$ disappears from the beach line, i. e., the length of the right parabolic arc of the site $\boldsymbol{p}_j$ becomes zero. In order to reflect this change, the right $\boldsymbol{p}_j$ in the list <…, $\boldsymbol{p}_j$, $\boldsymbol{p}_i$, $\boldsymbol{p}_j$, $\boldsymbol{p}_k$, …> is deleted as shown in Fig. 7(b). As the bisectors ($\boldsymbol{p}_i$, $\boldsymbol{p}_j$) and ($\boldsymbol{p}_j$, $\boldsymbol{p}_k$) have met each other at the Voronoi vertex, only a single bisector ($\boldsymbol{p}_i$, $\boldsymbol{p}_k$) remains. Accordingly, the triple of the consecutive sites $\boldsymbol{p}_i$, $\boldsymbol{p}_j$, $\boldsymbol{p}_k$ on the sweep-line status is replaced with $\boldsymbol{p}_i$, $\boldsymbol{p}_k$, and the list <…, $\boldsymbol{p}_j$, $\boldsymbol{p}_i$, $\boldsymbol{p}_j$, $\boldsymbol{p}_k$, …> becomes <…, $\boldsymbol{p}_j$, $\boldsymbol{p}_i$, $\boldsymbol{p}_k$, …> as shown in Fig. 7(c).

As the sweep line moves downwards, the above process is repeated to add new parabolic arcs through the site events and delete zeroed-length parabolic arcs through the circle events. After the sweep line completes the whole sweep, the Voronoi diagram of the site set $\boldsymbol{P}$ shown in Fig. 8(a) is created and depicted in Figs. 8(b) and 8(c).



(a) (b) (c)

Fig. 8. Generation of Voronoi diagram.

5.4 Extracting stiffeners

Having created the Voronoi diagram in 2D, the next work is to extract stiffeners from the Voronoi diagram. Suppose two ends of an edge of the Voronoi diagram is represented as p_a and p_b, respectively. And the edge intersects with the projected input mesh at m_i (i = 1, ⋯, I) where I is the number of intersections as shown in Fig. 9.

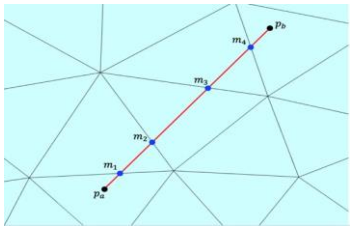


Fig. 9. Stiffener extraction.

The stiffener extraction step takes each edge from the Voronoi diagram. All local triangles t^l_i are iterated to detect all intersections p_1, p_2 in all triangles where p_1 stands for m_i, and p_2 stands for m_{i+1} (i=1, 2, …, I-1). In order to easily project 2D intersection points back to 3D, the obtained intersections p_1 and p_2 are encoded in area coordinates L_1 and L_2 using the local triangle t^l_i. After all edges of the Voronoi diagram have been processed, all intersections represented in local area coordinates are mapped back to 3D coordinates. The algorithm is summarized in Algorithm 2.

Algorithm 2: Stiffener Extraction

Input: voronoi diagram, local triangles t_i^l, global triangles t_i^g
Output: stiffeners

```
for each edge e_i in voronoi diagram do
    for each t_j^l do
        if e_i intersects with t_j^l then
            [p1, p2] = IntersectSegments(e_i, t_j^l)
            [L1, L2] = AreaCoords(p1, p2, t_j^l)
            stiffeners.append(L1, L2, t_j^l)
        end
    end
end
Project all 2D stiffeners back to 3D space
for stiffener in stiffeners do
    [p1, p2] = CartesianCoords(L1, L2, t_i^g)
end
```

6. Size optimization of stiffeners

Cost minimization of 3D printed objects can be treated as minimization of material consumption. In order to reduce material consumption, 3D printed objects are designed as thin shells. The required strength of thin shells is obtained through various enhancement structures. For thin shells stiffened with stiffeners, material consumption is determined by thin shells and stiffeners. In this paper, we take the wall thickness of 3D printed objects to be the minimum wall thickness of 3D printer. In doing so, minimization of wall thickness of shells is obtained, and material consumption minimization of 3D printed objects becomes volume minimization of stiffeners.

Stiffeners with a rectangular cross-section are widely applied in various thin-shell objects. In this paper, we investigate how to minimize the volume of this type of stiffeners. Assuming the length of the n^{th} stiffener is l_n, and the cross-section area of the stiffener is A_n, the number of the total stiffeners of a stiffened object is N, the total volume of the stiffeners of the stiffened object is $V = \sum_{n=1}^{N} l_n A_n$

The cross-section sizes of rectangular stiffeners are height h and width b. For stiffeners with a rectangular cross-section, if N stiffeners are required to stiffen a 3D printed thin-shell object, the volume minimization of stiffeners for the 3D printed thin-shell object involves $2N$ design variables: b_n, and h_n (n=1, 2, 3, …, N).

Stiffened objects should satisfy strength requirement. In engineering applications, von Mises stress is widely used in strength evaluation when structures or objects are subjected to a complicated loading condition. It is defined by the following equation

$$\sigma_v = \sqrt{\frac{(\sigma_{xx}-\sigma_{yy})^2+(\sigma_{yy}-\sigma_{zz})^2+(\sigma_{zz}-\sigma_{xx})^2+6(\tau_{xy}^2+\tau_{yz}^2+\tau_{zx}^2)}{2}} \tag{51}$$

where σ_{xx}, σ_{yy}, and σ_{zz} are three normal stresses, and τ_{xy}, τ_{yz}, and τ_{zx} are three shear stresses.

For the stiffeners with a rectangular cross-section, the total volume of the stiffeners is $V = \sum_{n=1}^{N} l_n h_n b_n$ where l_n (n=1, 2, 3, …, N) are known lengths of stiffeners determined in Subsection 5.4. Minimizing the total volume of stiffeners can be formulated as

$$\arg \min_{h_n, b_n} \sum_{n=1}^{N} h_n b_n \tag{52}$$

Having formulated the objective function, we formulate the optimization constraints. For thin-shell objects to be stiffened, the optimization constraints are: 1) user's specified lower bound $\underline{h}_n$ and upper bound $\overline{h}_n$ of the height of stiffeners, 2) user's specified lower bound $\underline{b}_n$ and upper bound $\overline{b}_n$ of the width of stiffeners, 3) the equivalent stress σ_n^{sh} in the shell is not more than the allowable stress σ_s, and 4) the equivalent stress σ_n^{st} in the n^{th} stiffeners is not more than the allowable stress σ_s. These optimization constraints can be formulated as

$$\begin{aligned} \underline{h}_n \le h_n \le \overline{h}_n \\ \underline{b}_n \le b_n \le \overline{b}_n \\ \sigma_n^{sh} \le \sigma_s \\ \sigma_n^{st} \le \sigma_s \end{aligned} \tag{53}$$

Volume minimization is to solve Eq. (52) subjected to the optimization constraints (53) for the stiffeners with a rectangular cross-section. In the section of Experiments and results, we will present many examples of obtaining the optimal sizes h_n and b_n together with the minimum volume of stiffeners by solving the above constrained optimization problem for the stiffeners with a rectangular cross-section.

7. Monte-Carlo simulation-based global optimization

As indicated in Algorithm 1, the seeded triangle t_i and probability $\boldsymbol{p}$ are both randomly generated from a uniform distribution. The stiffener distribution relies on the generated seeds from this algorithm, which may be a local minimum, not a global optimal solution. In order to tackle this problem, a Monte-Carlo simulation algorithm based on Monte-Carlo stochastic sampling is introduced.

Monte-Carlo sampling is one of the most classic sampling methods used to solve the problems such as evaluation of integrals, physical simulation, optimization and so on. With this sampling algorithm, n_m Monte-Carlo simulation iterations is specified, and then the process of determining the distribution of stiffeners and size optimizations of stiffeners is repeated n_m times with different randomly generated seeds r_s to search for a global optimal solution.

In this research, the number n_m of Monte-Carlo simulation iterations is set to be 100. The experiment indicates 100 Monte-Carlo simulation iterations are large enough to obtain a global optimal solution.

8. Experiments and results

In this section, we introduce the implementation and parameter setting of the proposed framework, effects of different probability thresholds and Monte-Carlo simulation, and 3D printed objects and the stress comparisons before and after they are stiffened with the method proposed in this paper.

8.1 Implementation and parameter setting

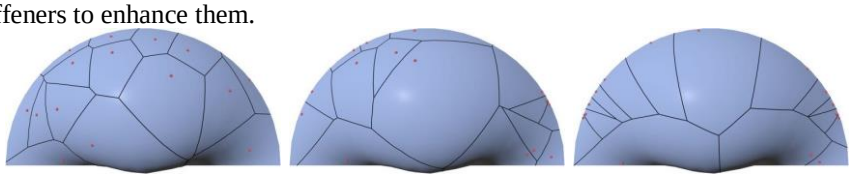
The proposed algorithm is implemented in MATLAB with FEM calculations compiled into MEX functions for speed reason. The simulations are conducted on a PC with an Intel Xeon E5 CPU and 32GB memory, running on Windows OS.

The minimal wall thickness allowed by the used printer is 1 mm. Therefore, both the $\underline{b}_n$ and $\underline{h}_n$ are set to be 1 mm. The material strength σ_s of the photosensitive resin used to print all the 3D objects is 42 N/mm^2 according to the information found from the link https://uk.3dsystems.com/sites/default/files/2020-03/3d-systems-figure-4-TOUGH-BLK-20-datasheet-usen-2020-03-16-web.pdf. Stiffeners act as beams. In order to avoid failure caused by a too large ratio of height to width of stiffeners, a maximum ratio of hight to width is specified. The maximum ratio of height to width found from the link www.ecourses.ou.edu/cgi-bin/ebook.cgi?topic=me&chap_sec=other&page=lumber_US &appendix=shapes is 6 in real applications. In this paper, the upper bounds $\overline{b}_n$ and $\overline{h}_n$ are taken to be 4 mm. When the width takes the minimum 1 mm and the height takes the maximum 4 mm, the maximum ratio of height to width is 4, which is less than 6 in real applications.

8.2 Effect of different probability thresholds

The probability threshold p^* is introduced here to control the spread of the seeds over the geometry. When p^* is set to a low value, the triangles with small probabilities will not be filtered out and marked as seeded ones, causing a wide spread of seeds over all triangles. On the contrary, if p^* is set to a high value, triangles with the stress $p\sigma_s$ no more than $p^*\sigma_s$ will never be selected, which guarantees the concentration of seeds around critical areas.

Figure 10 shows the effect of three different probability thresholds on the generated stiffeners with a rectangular cross-section where Figs. 10(a), 10(b) and 10(c) are from the probability thresholds 0, 0.3, and 0.5, respectively. It can be seen a small p^* such as $p^* = 0$ in Fig. 10(a) leads to a more uniform distribution of seeds over the mesh, while a large p^* such as $p^* = 0.5$ in Fig. 10(c) drives seeds towards the areas with higher stress and brings in more stiffeners to enhance them.



(a) $p^* = 0$ (b) $p^* = 0.3$ (c) $p^* = 0.5$

Fig. 10. Effect of different thresholds p^* on the distribution of seeds.

8.3 Effect of Monte-Carlo simulations

Figure 11 shows the effect of random number generator seed r_s of Monte-Carlo simulations on distributions of the stiffeners with a rectangular cross-section where the left, middle, and right images are from the random number

generator seeds 10, 20 and 30, respectively. With the same stress map and same number of seeds (n_s = 35), the distributions of seeds in Figs. 11(a), 11(b) and 11(c) are different, leading to different Voronoi diagrams shown in Figs. 11(d), 11(e) and 11(f) and different stiffener distributions shown in Figs. 11(g), 11(h), and 11(i), respectively.

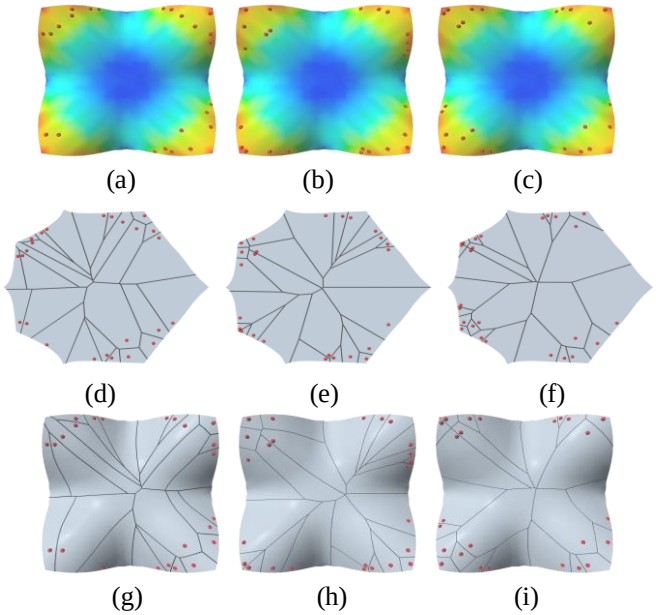


Fig. 11. Effect of Monte-Carlo simulations of a Guscio. The random number generator seeds r_s for each column are 10, 20 and 30, respectively.

8.4 3D printed objects and stress comparisons

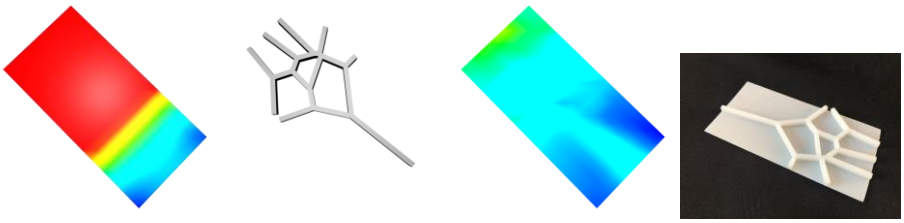
With the Voronoi diagram and Monte-Carlo based finite element optimization algorithm of stiffened objects proposed in this paper, the minimum stiffener volumes of some stiffened objects with a rectangular cross-section are obtained and the stress changes in the objects with and without the optimized stiffeners are shown in Fig. 13 -- Fig. 21, respectively. The 3D printed models stiffened with the optimized rectangular cross-section stiffeners are shown in Fig. 12.

Fig. 12. Printed 3D objects.

Table 1. Maximum stresses in unstiffened and stiffened thin-shell objects and total volume of stiffeners

	Maximum stresses (MPa)		Total volume (mm^3)
	Unstiffened	Stiffened	Stiffeners
Plate	278.198	24.6426	418.5148
Botanic	90.927	33.8706	418.856
Snail	33.273	28.3634	84.0108
Dome	59.028	34.3583	754.704
Bridge	94.4982	16.8744	535.109
Hemisphere	42.0198	31.2246	1961.93
Guscio	43.8379	29.5158	711.483
Lilium	52.0412	35.3578	227.294
Leaf	54.9437	24.6426	112.512

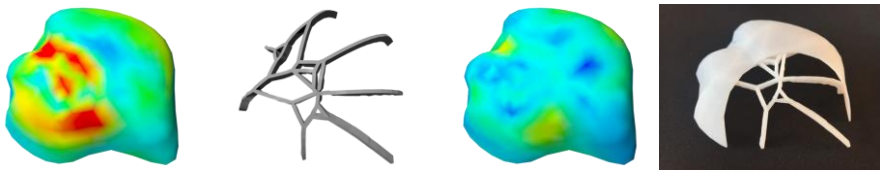
Figure 13 shows the stress distributions in an unstiffened and stiffened Plate, stiffeners, and 3D printed model. The maximum stresses in the unstiffened and stiffened Plate and the total volume of optimized stiffeners are given in Table 1. In the figure, (a) depicts the stress distribution in the flat plate without stiffeners with a maximum stress of 278.198 MPa, (b) shows the optimized stiffeners with a total volume of 418.5148 mm^3, (c) gives the stress distribution in the flat plate stiffened by the optimized stiffeners with a maximum stress 24.6426 MPa, and (d) is a photo of the 3D printed model of the stiffened plate. By applying the optimized stiffeners, the maximum stress reduces from 278.198 MPa to 24.6426 MPa.



(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed plate

Fig. 13. Unstiffened and stiffened Plate.

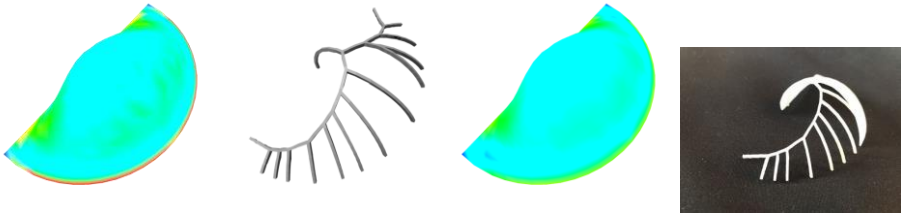
The example of a Botanic is given in Fig. 14 to show the stress distributions in an unstiffened and stiffened Botanic, stiffeners, and 3D printed model. The maximum stresses in the unstiffened and stiffened Botanic and the total volume of optimized stiffeners are given in Table 1. Fig. 14(a) shows the initial stress distribution in the Botanic without stiffeners with a maximum stress of 90.927 MPa, (b) shows the optimized stiffeners with a total volume of 418.856 mm^3, (c) gives the stress distribution in the Botanic stiffened by the optimized stiffeners with a maximum stress 33.8706 MPa, and (d) is a photo of the 3D printed model of the stiffened Botanic. By applying the optimized stiffeners, the maximum stress reduces from 90.927 MPa to 33.8706 MPa.



(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed botanic

Fig. 14. Unstiffened and stiffened Botanic.

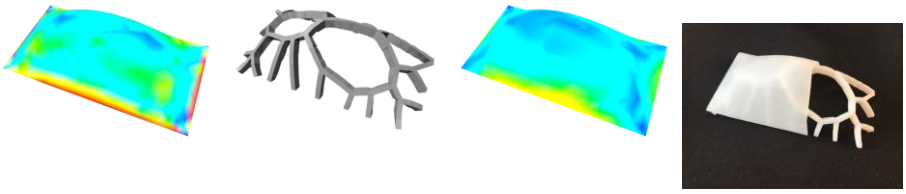
The stress fields in an unstiffened and stiffened Snail, stiffeners and 3D printed model are shown in Fig. 15. The maximum stresses in the unstiffened and stiffened Snail and the total volume of optimized stiffeners are given in Table 1. In the figure, the initial maximum stress in the Snail without any stiffeners is 33.273 MPa as shown in (a). After applying the stiffeners (b) with a total volume of 84.0108 mm^3 to the Snail, the maximum stress shown in (c) drops from 33.273 MPa to 28.3634 MPa in the final printed 3D model (d).



(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed snail

Fig. 15. Unstiffened and stiffened Snail.

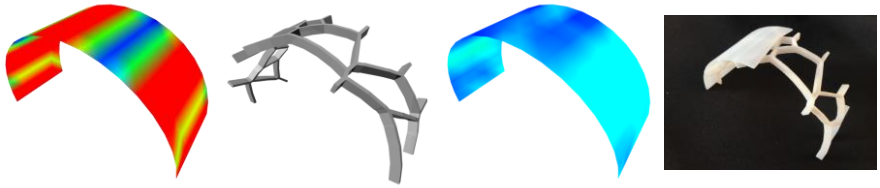
Figure 16 shows the stress distributions in an unstiffened and stiffened Dome, stiffeners, and 3D printed object. The maximum stresses in the unstiffened and stiffened Dome and the total volume of optimized stiffeners are given in Table 1. The maximum stress 59.028 MPa in the initial stress distribution (a) without any stiffeners is reduced to the maximum stress 34.3583 MPa in (c) by applying the stiffened stiffeners (b) with a total volume of 754.704 mm^3. (d) is a photo of the 3D printed model of the stiffened Dome.



(a) Initial stress (b) Stiffeners (c) Final stress (d) 3D printed dome

Fig. 16. Unstiffened and stiffened Dome.

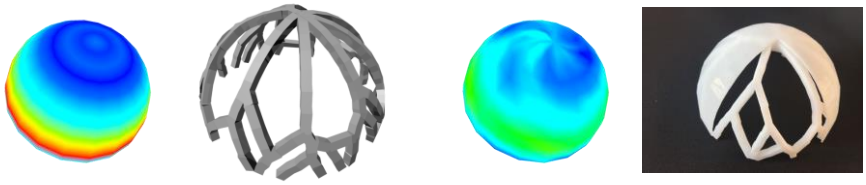
The stress fields in an unstiffened and stiffened Bridge, stiffeners and 3D printed model are shown in Fig. 17. The maximum stresses in the unstiffened and stiffened Bridge and the total volume of optimized stiffeners are given in Table 1. In the figure, the initial maximum stress in the bridge without any stiffeners is 94.4982 MPa as shown in (a). After applying the stiffeners (b) with a total volume of 535.109 mm^3 to the bridge, the final maximum stress (c) drops from 94.4982 MPa to 16.8744 MPa in the final printed 3D model (d).



(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed bridge

Fig. 17. Unstiffened and stiffened Bridge.

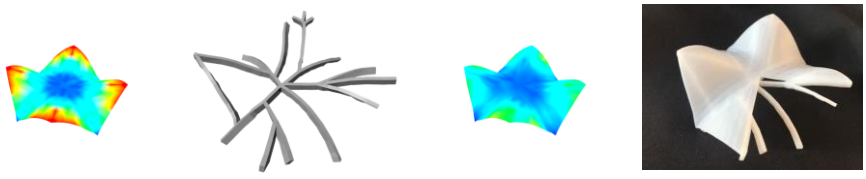
Figure 18 shows the stress distributions in an unstiffened and stiffened Hemisphere, stiffeners, and 3D printed object. The maximum stresses in the unstiffened and stiffened Hemisphere and the total volume of optimized stiffeners are given in Table 1. The initial stress distribution without stiffeners has a maximum stress of 42.0198 MPa shown in (a), (b) shows the optimized stiffeners with a total volume of 1961.93 mm^3, (c) gives the stress distribution in the Hemisphere stiffened by the optimized stiffeners with a maximum stress 31.2246 MPa, and (d) is a photo of the 3D printed model of the stiffened Hemisphere. The applied optimized stiffeners help to reduce to the maximum stress from 42.0198 MPa to 31.2246 MPa.



(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed hemisphere

Fig. 18. Unstiffened and stiffened Hemisphere.

Figure 19 shows the stress distributions in an unstiffened and stiffened Guscio, stiffeners, and 3D printed object. The maximum stresses in the unstiffened and stiffened Guscio and the total volume of optimized stiffeners are given in Table 1. The maximum stress 43.8379 MPa in the initial stress distribution (a) without any stiffeners is reduced to the maximum stress 29.5158 MPa in (c) by introducing the stiffeners (b) with a total volume of 711.483 mm^3. A photo of the 3D printed model of the stiffened Guscio is shown in Fig. 19(d).

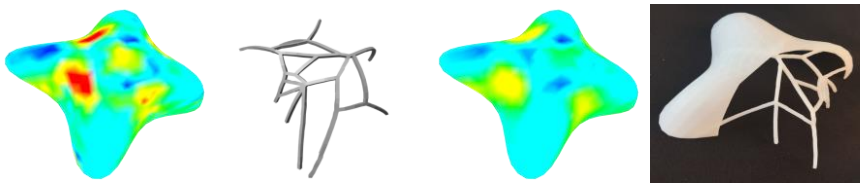


(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed guscio

Fig. 19. Unstiffened and stiffened Guscio.

Figure 20 shows the stress distribution in an unstiffened and stiffened Lilium, stiffeners, and 3D printed object of a Lilium. The maximum stresses in the unstiffened and stiffened Lilium and the total volume of optimized stiffeners are given in Table 1. The initial stress distribution without stiffeners has a maximum stress of 52.0412

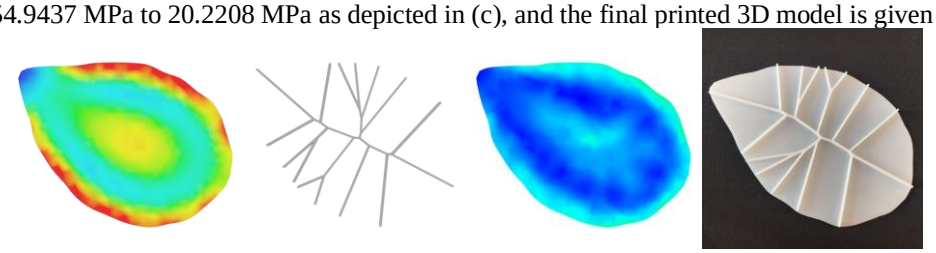
MPa shown in (a), (b) shows the optimized stiffeners with a total volume of 227.294 mm^3, (c) gives the stress distribution in the Lilium stiffened by the optimized stiffeners with a maximum stress 35.3578 MPa, and (d) is a photo of the 3D printed model of the stiffened Lilium. The applied optimized stiffeners help to reduce the maximum stress from 52.0412 MPa to 35.3578 MPa.



(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed lilium

Fig. 20. Unstiffened and stiffened Lilium.

The stress fields in an unstiffened and stiffened Leaf, stiffeners and 3D printed object are shown in Fig. 21. The maximum stresses in the unstiffened and stiffened Leaf and the total volume of optimized stiffeners are given in Table 1. In this example, the initial maximum stress in the Leaf without any stiffeners is 54.9437 MPa as shown in (a). After attaching the stiffeners (b) with a total volume of 112.512 mm^3 to the Leaf, the final maximum stress drops from 54.9437 MPa to 20.2208 MPa as depicted in (c), and the final printed 3D model is given in (d).



(a) Initial stress (b) Stiffener (c) Final stress (d) 3D printed leaf

Fig. 21. Unstiffened and stiffened Leaf.

We have also timed seed generation, Voronoi creation, stiffener extraction, and optimization calculations and listed the obtained time in Table 2. In the table, "Total" means the "Total time", which is the sum of the time spent on seed generation, Voronoi creation, stiffener extraction, and optimization calculations.

Table 2. Calculation time (milliseconds)

Object name	Total	Seed generation	Voronoi diagram creation	Stiffener extraction	Optimization
Botanic	12295.659	0.454	1.334	98.471	12195.400
Snail	8861.159	1.050	0.634	72.225	8787.250
Dome	7007.618	0.428	0.880	115.220	6891.090
Bridge	901.005	0.151	1.441	46.334	853.079
Hemisphere	4404.410	0.659	1.674	120.167	4281.910
Guscio	4855.962	0.719	0.607	60.876	4793.760
Lilium	21755.952	1.049	0.730	138.973	21615.200
Leaf	6656.774	0.401	0.682	120.461	6535.230

According to the data in Table 2, the optimization calculations took the most time, and seed generation took the least time. Among all the objects given in Table 2, the most time (21.756 seconds) was used to calculate Lilium, and the least time (0.901 seconds) was used to calculate Bridge. They indicate that the proposed method is very efficient in obtaining the minimum 3D printing material consumption of various stiffened thin-shell objects.

9. Conclusions and future work

In this paper, we have developed a finite element optimization framework based on Voronoi diagram and Monte-Carlo simulation to minimize the material consumption of 3D printing. Our developed framework consists of the finite element analysis to obtain the stress distribution in thin-shell objects, random generation of seeds guided by the obtained stress field, using conformal mapping to map the 3D objects and generated seeds to a 2D parametric domain to create a Voronoi diagram for optimizing the distribution of stiffeners. Apart from optimizing the stiffener distribution, the cross-section sizes of stiffeners are minimized to further save materials for 3D printing. The Monte-Carlo simulation is introduced to optimize the seed generation and achieve a global optimal solution.

A lot of experiments were carried out to demonstrate the effectiveness and advantages of the proposed method. The stress comparisons between the thin-shell objects with and without stiffeners demonstrate that thin-shell objects stiffened with the optimized distribution and cross-section sizes of stiffeners significantly reduce material consumption of 3D printed objects.

This paper only considers the stiffeners with a rectangular cross-section. In the future, other types of cross-sections such as tee-shaped and I/H (double-tee) cross-sections will be investigated. In addition, this paper assumes that cross-section sizes of stiffeners do not change along the length of stiffeners. In fact, the cross-section sizes of stiffeners in high stress regions should be bigger than the cross-section sizes in low stress regions. We will introduce varying cross-section sizes to further minimize material consumption of stiffeners in our following work. A small number of the expected seeds n_s will lead to a small number of stiffeners but large cross-section sizes of stiffeners. Oppositely, a large number of the expected seeds n_s will lead to a large number of stiffeners but small cross-section sizes of stiffeners. There should be an optimal number of the expected seeds, which will lead to the minimum volume of stiffeners. In the future, we will investigate how to optimize the number of the expected seeds n_s.

Conflicts of interest

The authors declare no conflicts of interest.

Acknowledgements

This research is supported by the PDE-GIR project that has received funding from the European Union Horizon 2020 research and innovation programme under the Marie Skodowska-Curie grant agreement No 778035.

References

[1] A.Z. Zheng, S.J. Bian, E. Chaudhry, J. Chang, H. Haron, L.H. You, J.J. Zhang, Minimizing material consumption of 3D printing with stress-guided optimization, V. V. Krzhizhanovskaya et al. (Eds.): ICCS 2020, Lecture Notes in Computer Science 12141 (2020) 588-603.

[2] M. Skouras, B. Thomaszewski, S. Coros, B. Bickel, M. Gross, Computational design of actuated deformable characters, ACM Transactions on Graphics 32(4) (2013) 82:1-9.

[3] J. Calì, D.A. Calian, C. Amati, R. Kleinberger, A. Steed, J. Kautz, T. Weyrich, 3D-printing of non-assembly, articulated models, ACM Transactions on Graphics 31(6) (2012) 130:1-8.

[4] L. Zhu, W. Xu, J. Snyder, Y. Liu, G. Wang, B. Guo, Motion-guided mechanical toy modeling, ACM Transactions on Graphics 31(6) (2012) 127:1-10.

[5] S. Coros, B. Thomaszewski, G. Noris, S. Sueda, M. Forberg, R.W. Sumner, W. Matusik, B. Bickel, Computational design of mechanical characters, ACM Transactions on Graphics 32(4) (2013) 83:1-12.

[6] Y. Dong, J. Wang, F. Pellacini, X. Tong, B. Guo, Fabricating spatially varying subsurface scattering, ACM Transactions on Graphics 29(4) (2010) 62:1-10.

[7] D. Chen, D.I. Levin, P. Didyk, P. Sitthi-Amorn, W. Matusik, Spec2Fab: a reducer-tuner model for translating specifications to 3D prints, ACM Transactions on Graphics 32(4) (2013) 135:1-9.

[8] O.C. Zienkiewicz, R.L. Taylor, The finite element method for solid and structural mechanics, Elsevier 2005.

[9] D.V. Rao, A.H. Sheikh, M. Mukhopadhyay, A finite element large displacement analysis of stiffened plates, Computers & Structures 47(6) (1993) 987-993.

[10] A. Samanta, M. Mukhopadhyay, Finite element large deflection static analysis of shallow and deep stiffened shells, Finite Elements in Analysis and Design 33(3) (1999) 187-208.

[11] A. Samanta, M. Mukhopadhyay, Free vibration analysis of stiffened shells by the finite element technique, European Journal of Mechanics-A/Solids 23(1) (2004) 159-179.

[12] R. Ojeda, B.G. Prusty, N. Lawrence, G. Thomas, A new approach for the large deflection finite element analysis of isotropic and composite plates with arbitrary orientated stiffeners, Finite Elements in Analysis and Design 43(13) (2007) 989-1002.

[13] X.Y. Cui, G.R. Liu, G.Y. Li, X. Zhao, T. Nguyen-Thoi, G.Y. Sun, A smoothed finite element method (SFEM) for linear and geometrically nonlinear analysis of plates and shells, Computer Modeling in Engineering and Sciences 28(2) (2008) 109-125.

[14] H. Nguyen-Van, N. Nguyen-Hoai, T. Chau-Dinh, T. Tran-Cong, Large deflection analysis of plates and cylindrical shells by an efficient four-node flat element with mesh distortions, Acta Mechanica 226(8) (2015) 2693-2713.

[15] G. R. Sinha, Modern Optimization Methods for Science, Engineering and Technology, IOP Publishing Ltd, 2020.
[16] P. Liu, H. Yu, S. Cang, Optimized adaptive tracking control for an underactuated vibro-driven capsule system, Nonlinear Dynamics 94 (2018) 1803-1817.
[17] P. Liu, H. Yu, S. Cang, Trajectory synthesis and optimization of an underactuated microrobotic system with dynamic constraints and couplings, International Journal of Control, Automation and Systems 16(5) (2018) 2373-2383.
[18] R.B.A. Shyam, P. Lightbody, G. Das, P. Liu, S. Gomez-Gonzalez, G. Neumann, Improving local trajectory optimisation using probabilistic movement primitives, Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau (China), 2019, pp. 2666-2671.
[19] S. Sun, Z. Cao, H. Zhu, J. Zhao, A survey of optimization methods from a machine learning perspective, IEEE Transactions on Cybernetics 50(8) (2020) 3668-3681.
[20] P. Liu, H. Yu, S. Cang, Adaptive neural network tracking control for underactuated systems with matched and mismatched disturbances, Nonlinear Dynamics 98 (2019) 1447-1464.
[21] O. Stava, J. Vanek, B. Benes, N. Carr, R. Měch, Stress relief: improving structural strength of 3D printable objects, ACM Transactions on Graphics 31(4) (2012) 48:1-11.
[22] H. Zhao, W. Xu, K. Zhou, Y. Yang, X. Jin, H. Wu, Stress-constrained thickness optimization for shell object fabrication, Computer Graphics Forum 36(6) (2017) 368-380.
[23] W. Wang, T.Y. Wang, Z. Yang, L. Liu, X. Tong, W. Tong, J. Deng, F. Chen, X. Liu, Cost-effective printing of 3D objects with skin-frame structures. ACM Transactions on Graphics 32(6) (2013) 177:1-10.
[24] L. Lu, A. Sharf, H. Zhao, Y. Wei, Q. Fan, X. Chen, Y. Savoye, C. Tu, D. Cohen-Or, B. Chen, Build-to-last: strength to weight 3D printed objects, ACM Transactions on Graphics 33(4) (2014) 97:1-10.
[25] W. Li, A. Zheng, L.H. You, X.S. Yang, J.J. Zhang, L. Liu, Rib-reinforced shell structure, Computer Graphics Forum 36(7) (2017) 15-27.
[26] F. Gil-Ureta, N. Pietroni, D. Zorin, Structurally optimized shells, arXiv preprint arXiv:1904.12240 (2019) 1-17.
[27] Q. Du, V. Faber, M. Gunzburger, Centroidal Voronoi tessellations: Applications and algorithms, SIAM Review 41(4) (1999) 637-676.
[28] M. Zivanica, O. Daescua, A. Kurdiab, S.R. Goodman, The Voronoi diagram for graphs and its application in the Sickle Cell Disease research, Journal of Computational Science 3**(**5**) (**2012**)** 335-343.
[29] D. Deritei, Z.I. Lázár, I. Papp, F. Járai-Szabó, R. Sumi, L. Varga, E.R. Regan, M. Ercsey-Ravasz, Community detection by graph Voronoi diagrams, New Journal of Physics 16 (2014) 063007:1-16.
[30] S. Li, L. Zhang, P. Li, D. Chen, New methods for the construction of Voronoi diagram and the nearest neighbor query, The 9th International Forum on Strategic Technology (IFOST), Cox's Bazar, Bangladesh, October 21-23, 2014, pp. 255-258.
[31] S. Kang, A generic statistics-based tessellation method of Voronoi diagram, Journal of Systems Science and Information 3(6) (2015) 568-576.
[32] H. Long, S. Zhang, J. Wang, C.-K. Lin, J.-J. Cheng, Privacy preserving method based on Voronoi diagram in mobile crowd computing, International Journal of Distributed Sensor Networks 13(10) (2017) 1-8.
[33] Y. Qin, H. Yu, J.J. Zhang, Fast and memory-efficient Voronoi diagram construction on triangle meshes, Computer Graphics Forum 36(5) (2017) 93-104.
[34] M. Fantini, M. Curto, Interactive design and manufacturing of a Voronoi-based biomimetic bone scaffold for morphological characterization, International Journal on Interactive Design and Manufacturing 12 (2018) 585-596.
[35] E. Lewandowicz, P. Flisek, A method for generating the centerline of an elongated polygon on the example of a watercourse, International Journal of Geo-Information 9(5) (2020) 1-20.
[36] F. Tang, X. You, X. Zhang, K. Li, Hexagon-based generalized Voronoi diagrams generation for path planning of intelligent agents, Mathematical Problems in Engineering 5750739 (2020) 1-13.
[37] Y. Du, H. Liang, D. Xie, N. Mao, J. Zhao, Z. Tian, C. Wang, L. Shen, Design and statistical analysis of irregular porous scaffolds for orthopedic reconstruction based on Voronoi tessellation and fabricated via selective laser melting (SLM), Materials Chemistry and Physics 239 (2020) 121968:1-9.
[38] H.-Y. Lei, J.-R. Li, Z.-J. Xu, Q.-H. Wang, Parametric design of Voronoi-based lattice porous structures, Materials and Design 191 (2020) 108607:1-10.
[39] S. Haker, S. Angenent, A. Tannenbaum, R. Kikinis, G. Sapiro, M. Halle, Conformal surface parameterization for texture mapping, IEEE Transactions on Visualization and Computer Graphics 6(2) (2000) 181-189.
[40] B. Lévy, S. Petitjean, N. Ray, J. Maillot, Least squares conformal maps for automatic texture atlas generation, ACM Transactions on Graphics 21(3) (2002) 362-371.
[41] L. Kharevych, B. Springborn, P. Schröder, Discrete conformal mappings via circle patterns, ACM Transactions on Graphics 25(2) (2006) 412-438.

[42] O. Weber, C. Gotsman, Controllable conformal maps for shape deformation and interpolation. ACM Transactions on Graphics 29(4) (2010) 78:1-11.
[43] V.G. Kim, Y. Lipman, T. Funkhouser, Blended intrinsic maps, ACM Transactions on Graphics 30 (2011) 79:1-12.
[44] O. Weber, A. Myles, D. Zorin, Computing extremal quasiconformal maps, Computer Graphics Forum 31 (2012) 1679-1689.
[45] R. Chen, O. Weber, Bounded distortion harmonic mappings in the plane, ACM Transactions on Graphics 34(4) (2015) 73:1-12.
[46] A. Segall, M. Ben-Chen, Iterative closest conformal maps between planar domains, Computer Graphics Forum 35(5) (2016) 33-40.
[47] R., Chen, C. Gotsman, Approximating planar conformal maps using regular polygonal meshes, Computer Graphics Forum 36 (2017) 629-642.
[48] M. Ramdin, Q. Chen, S.P. Balaji, J.M. Vicent-Luna, A.Torres-Knoop, D. Dubbeldam, S. Calero, T.W. de Loos, T.J.H. Vlugt, Solubilities of CO_2, CH_4, C_2H_6, and SO_2 in ionic liquids and Selexol from Monte Carlo simulation, Journal of Computational Science 15 (2016) 74-80.
[49] K.E. Khaldi, E.G. Saleeby, On the tangent model for the density of lines and a Monte Carlo method for computing hypersurface area, Monte Carlo Methods and Applications 23(1) (2017) 13-20.
[50] J. Qinag, Monte Carlo simulation techniques, Proceedings of the 2018 CERN-Accelerator-School Course on Numerical Methods for Analysis, Design and Modelling of Particle Accelerators, Thessaloniki, (Greece), 2018, pp. 1-11.
[51] R. Sawhney, K. Crane, Monte Carlo geometry processing: a grid-free approach to PDE-based methods on volumetric domains, ACM Transactions on Graphics 39(4) (2020) 123:1-18.
[52] G. Xie, A novel Monte Carlo simulation procedure for modelling COVID-19 spread over time, Scientific Reports 10 (2020) 13120.
[53] Y. Nagai, M. Okumura, A. Tanaka, Self-learning Monte Carlo method with Behler-Parrinello neural networks, Physical Review B 101 (2020) 115111.
[54] A. Heilmeier, M. Graf, J. Betz, M. Lienkamp, Application of Monte Carlo methods to consider probabilistic effects in a race simulation for circuit motorsport, Applied Sciences 10(12) (2020) 4229:1-21.
[55] H. Neuyen-Van, N. Neuyen-Hoai, T. Chau-Dunh, T. Tran-Cong, Large deflection analysis of plates and cylindrical shells by an efficient four-node flat element with mesh distortions, Acta Mechanica 226 (2015) 2693-2713.

Anzong Zheng is a R & D engineer at Humain Ltd, UK. He received his BSc and MSc degree from Tianjin University, China and a PhD degree from National Centre for Computer Animation, Bournemouth University, UK. His current interests are physics-based simulations, differential geometry processing, skinning and realistic face animation.

Shaojun Bian is currently a software engineer in HUMAIN Limited. She received her PhD degree from the National Centre for Computer Animation, Bournemouth University, UK. She received her BSc degree from China University of Petroleum and MSc degree from Ocean University of China. She worked as a research associate on the Knowledge Transfer Partnership between Bournemouth University and HUMAIN Limited from 19/02/2018 to 18/02/2020, awarded the highest grade of "Outstanding" by the KTP Grading Panel for its achievement in meeting KTP's Objectives. Her research interests include computer vision, deep learning, computer graphics, computer animation and geometric modelling.

Ehtzaz Chaudhry is a researcher at the National Centre for Computer Animation, Bournemouth University, UK. He received his BSc, MSc degree in Computer Science, another MSc degree from the University of Westminster, UK and a PhD degree from Bournemouth University, UK. Dr Chaudhry's research interests are in Computer Graphics, Character Animation, Geometric modelling, ODE-Based Dynamic Skin Deformation, VR/AR and Games.

Jian Chang is Professor at the National Centre for Computer Animation, Bournemouth University. He received his Ph.D. degree in computer graphics in 2007 at the National Centre for Computer Animation, Bournemouth University. His research focuses on physically based modelling, motion synthesis, virtual reality, and novel HCI (eye tracking, gesture control and haptic).

Habibollah Haron is a Professor in Soft Computing, Universiti Teknologi Malaysia (UTM). He has held various administrative positions including being the Head of Department of Department of Modelling and Industrial Computing and Department of Computer Science before assuming the post of Deputy Dean (Academic), Faculty of Computing. During his tenure, he was in charge of proposing

new curriculum and revising existing curriculum for bachelor and master program, managing and facilitate his academic staff for pursuing studies and preparing research proposal, and advising students on academic matters. He has taught various courses at the faculty including industrial computing related courses example robotic, automation, programming languages and computational mathematic related subject at both undergraduate and postgraduate levels. He also supervises both Master and PhD students in related fields. He has been written few books and Editor for few conference proceeding, Editorial Board and Reviewer of various journals related to the computer science, and has been appointed as keynote speaker for various conferences. His research interests include optimization in various domain such as medical problems, manufacturing, robotic and image processing. He has also project leader for various research projects at university, national and international level. Currently he is attached to the UTM Applied Industrial Analytics (ALIAS) research group.

Lihua You is Professor at the National Centre for Computer Animation, Bournemouth University, UK. He received his Ph.D. degree from Chongqing University, China, and another Ph.D. degree from Bournemouth University, UK. His current research interests are in computer graphics, computer animation, and geometric modelling.

Jian Jun Zhang is Professor of Computer Graphics at the National Centre for Computer Animation, Bournemouth University, UK where he leads the National Research Centre. He is also a co-founder of the UK's Centre for Digital Entertainment, funded by the Engineering and Physical Sciences Research Council. His research focuses on 3D virtual human modelling, animation, simulation and immersive technology.