Computer-Aided Design 134 (2021) 102992

Contents lists available at ScienceDirect

Computer-Aided Design

journal homepage: www.elsevier.com/locate/cad

Discrete Gaussian Curvature Flow for Piecewise Constant Gaussian Curvature Surface[☆]

Kazuki Hayashi [a,*], Yoshiki Jikumaru [b], Makoto Ohsaki [a], Takashi Kagaya [b], Yohei Yokosuka [c]

[a] *Department of Architecture and Architectural Engineering, Kyoto University, Japan*
[b] *Institute of Mathematics for Industry, Kyushu University, Japan*
[c] *Department of Architecture and Architectural Engineering, Kagoshima University, Japan*

A R T I C L E I N F O

Article history:
Received 27 August 2020
Received in revised form 30 November 2020
Accepted 2 January 2021

Keywords:
Form-finding
Discrete differential geometry
Linear Weingarten surface
Gaussian curvature flow
Energy minimization

A B S T R A C T

A method is presented for generating a discrete piecewise constant Gaussian curvature (CGC) surface. An energy functional is first formulated so that its stationary point is the linear Weingarten (LW) surface, which has a property such that the weighted sum of mean and Gaussian curvatures is constant. The CGC surface is obtained using the gradient derived from the first variation of a special type of the energy functional of the LW surface and updating the surface shape based on the Gaussian curvature flow. A filtering method is incorporated to prevent oscillation and divergence due to unstable property of the discretized Gaussian curvature flow. Two techniques are proposed to generate a discrete piecewise CGC surface with preassigned internal boundaries. The step length of Gaussian curvature flow is adjusted by introducing a line search algorithm to minimize the energy functional. The effectiveness of the proposed method is demonstrated through numerical examples of generating various shapes of CGC surfaces.

1. Introduction

Design of architectural free-form surfaces requires complex decision making in view of desirable geometry, structural performance, and constructability. To generate a smooth surface that is optimal in view of a certain performance measure, a number of methods have been proposed utilizing parametric representations of surfaces. Ramm et al. [1] used Bézier surface [2] to express the shape of a shell roof structure, and optimized its shape incorporating the coordinates of control points as design variables. Nagata and Honma [3] generated Pareto optimal solutions of parametric free-form shells using a heuristic approach. However, even if the architectural geometry is defined as a smooth surface, it is necessary to discretize it into meshes to apply the finite element method for simulating its structural performance. Isogeometric analysis is recently used so that structural analysis is carried out using the same basis function as the geometry representation [4]. However, in practical application, a shell roof surface is often constructed using planar panels for reduction of the cost and time for fabrication and construction [5]. Therefore, it is beneficial to directly design a nonparametric form of discrete (polyhedral) surface consisting of a set of planar panels (faces). Nevertheless, relatively few methods have been proposed for designing architectural geometry through direct handling of discrete surfaces [6,7] rather than modeling as parametric smooth surfaces.

Properties of the surface can be evaluated using various geometrical invariants such as Gaussian curvature and mean curvature. Rando and Rourier [8] proposed several fairness metrics for parametric curves and surfaces. Fu et al. [9] used Gaussian curvature for measuring the similarity between free-form surfaces. Fujita and Ohsaki [10] used triangular Bézier patch to express irregular shape of shell structures and evaluated properties of the surface shape using geometric invariants defined by the covariant and contravariant gradients and Hessian of the surface.

There are many piecewise-smooth free-form architectural roofs with internal boundaries, and the overall locations of internal boundaries are usually determined in view of architectural design and planning requirements. The surface has C^0 continuity at the internal boundary, which may be stiffened by beams to prevent collapse due to stress concentration. Therefore, it is practically and theoretically important to investigate the methods for generating surfaces with specified locations of internal boundaries of piecewise smooth surfaces. Ohsaki and Hayashi [11] used fairness metrics to generate piecewise smooth Bézier surface. Nakamura et al. [12] proposed a method for connecting

☆ This paper has been recommended for acceptance by Y.J. Zhang.
* Corresponding author.
E-mail address: hayashi.kazuki.55a@st.kyoto-u.ac.jp (K. Hayashi).

https://doi.org/10.1016/j.cad.2021.102992
0010-4485/© 2021 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

$(1, n)$-Bézier surfaces to generate piecewise developable surfaces of architectural roofs and facades.

Gaussian curvature is an important index for the convexity of the architectural roofs. In this study, we first formulate the energy functional so that its stationary point is the linear Weingarten (LW) surface [13]. The energy functional is the weighted sum of the total mean curvature, the total area, and the volume bounded by the surface. The LW surface has the property such that the weighted sum of mean and Gaussian curvatures is constant [14,15]. The energy functional is simplified as the sum of the total curvature of a surface and the product of target Gaussian curvature and the volume bounded by the surface by ignoring the total area. The negative gradient flow of this functional, called Gaussian curvature flow, is utilized to generate a piecewise smooth surface as an assembly of the constant Gaussian curvature (CGC) surfaces. Therefore, the CGC surface can be regarded as a special type of the LW surface, and it can be obtained using the Gaussian curvature flow derived from the first variation of the energy functional and moving the surface in the normal direction of the surface in accordance with the Gaussian curvature flow. Although mean curvature flow has been successfully applied to surface fairing in the field of computer graphics, only a few applications can be found for the Gaussian curvature flow [16]. Zhao and Xu [17] noted that the Gaussian curvature flow can be used only for a convex surface.

Most of the studies on optimization of free-form shell roofs in architectural design is intended to obtain a mechanically efficient form under vertical loads mainly representing the self-weight. Even when the convex dome-type shapes are desired, an HP-type surface with negative Gaussian curvature may be sometimes obtained as a result of optimization. Therefore, it is important to present a method for generating CGC surfaces with positive Gaussian curvatures to generate feasible solutions for mechanical optimization to limit the design domain. Convex surfaces are also desired for ensuring enough stiffness against horizontal earthquake and wind loads.

According to Refs. [18,19], the LW surface is closely related to an equilibrium shape of membrane shells, in which the bending moment and out-of-plane shear force can be neglected. If the equilibrium is expressed with respect to the lines of principal curvatures with vanishing shear stress and tension stresses proportional to the principal curvatures in the different directions, respectively, then the Gaussian curvature is proportional to the pressure loads in the normal direction of the surface. Therefore, the LW surface is expected to be a special class of surface for shape optimization of shell and membrane structures. Furthermore, the CGC surface has a special property such that the principal directions of the membrane stresses coincide with the curvature lines, when subjected to uniform pressure loads, and the values of principal stresses are proportional to the principal curvatures.

Since the developable surface is regarded as a special type of CGC surface, the methods developed for the CGC surfaces can be naturally applied to the developable surfaces that have many fields of application. The method of generating a developable surface with curved internal boundary may be directly used for design of origami structures with curved crease lines.

The discretized form of Gaussian curvature flow is obtained using the angle defect at each vertex of the triangular mesh [20]. The discrete LW surface can be regarded as a generalization of the discrete constant mean curvature surface [21]. Tellier et al. [22, 23] obtained the gradient of the energy functional of the LW surface after discretizing the surface. They used the dihedral angle of the adjacent pair of triangular faces to define the mean curvature, and derived the gradients of the total mean curvature by approximately differentiating the dihedral angle. It is also possible to solve a nonlinear programming problem for minimizing the norm of error of the Gaussian curvature from the specified target value to obtain a CGC surface. However, in this case, gradients should be evaluated by solving a set of large number of linear equations if the surface is discretized into many triangular faces. Therefore, it is important to develop a method based on energy minimization using a simple evaluation of gradients. This study aims at obtaining the gradient by discretizing the first variation of the energy functional so that the process of generating the CGC surface is fully compatible with minimization of the energy functional. For architectural roofs that have various boundary shapes, it is not realistic to try to have a constant Gaussian curvature in the whole domain of the surface. Therefore, it is important to develop a method for generating piecewise CGC surfaces, e.g., a piecewise developable surface may be generated using $(1, n)$-Bézier surfaces [24,25].

It is well known in the field of structural shape optimization of continuum structures that nonsmooth boundary shapes are generated if the locations of nodes of the finite element mesh are considered as independent design variables [26]. It is also well known that distribution of the material or thickness of plate elements becomes nonsmooth if the density of material or the thickness of each finite element is considered as independent design variables [27]. To prevent obtaining distorted shapes, various approaches of filtering have been proposed [28–30]. Filtering techniques are also used in the field of computer graphics [31].

In this paper, a method is presented for generating a discrete piecewise CGC surface. Although there exist many studies for generating CGC surfaces, the properties of surfaces with boundary remain to be investigated, and most of methods have been developed for the closed surfaces without boundary. The proposed method for generating a surface with internal boundaries can be extended and generalized for generating various types of piecewise continuous surface. In Section 2, an energy functional is first formulated so that its stationary point is the LW surface, and simplified by ignoring the area term to obtain the gradient as the first variation of the energy functional. In Section 3, the discretized forms of the energy functional and the Gaussian curvature flow are derived. An optimization method is developed in Section 4 for generating a discrete piecewise CGC surface using the Gaussian curvature flow. A filtering method is incorporated to prevent oscillation due to unstable property of the discretized Gaussian curvature flow. Two strategies are proposed to generate a discrete piecewise CGC surface with preassigned internal boundaries. The step length of Gaussian curvature flow is adjusted by introducing a line search algorithm to minimize the energy functional. Effectiveness of the proposed method is demonstrated in Section 5 through numerical examples of various CGC surfaces.

2. Gaussian curvature flow

In this section, the basics of LW surface and Gaussian curvature flow are explained for completeness of the paper. Although some results are presented in the literatures in mathematical forms including those in higher dimensional space, presenting basic formulas for special cases in this paper may be beneficial for understanding the details of our method.

2.1. Gaussian curvature flow for continuous surface

The Gaussian curvature flow for continuous surface is first introduced utilizing the properties of a special case of the LW surface. Let H and A denote the mean curvature and the area of a closed surface $\mathbf{X} \in \mathbb{R}^3$ in a three dimensional space. The volume bounded by the surface is denoted by V. These values are defined in Section 3.1 for the discretized surface. Using the constants α, β, and γ, the energy functional is formulated as

$$E = \int (\alpha H + \beta) dA + \gamma V \tag{1}$$

where the stationary point of E is the linear LW surface [13]. Note that a constant mean curvature surface is generated if E is minimized with $\alpha = 0$, $\beta \neq 0$, and $\gamma \neq 0$ [32].

Considering the special case of $\alpha = 1$, $\beta = 0$, and $\gamma = \bar{K}$ with a constant $\bar{K}$, Eq. (1) is rewritten as

$$\tilde{E} = \int H dA + \bar{K} V \tag{2}$$

The stationary condition of the energy functional $\tilde{E}$ is derived below for completeness of the paper. Let $\mathbf{n} \in \mathbb{R}^3$ denote the unit outward normal vector at a point $\mathbf{X}$ on the surface. We consider a variation of the surface $\mathbf{X} \to \mathbf{X} + t\delta\mathbf{X} = \mathbf{X} + t\psi\mathbf{n}$ in the normal direction, where ψ is an arbitrary function of the coordinates on the surface, and t is a scalar parameter. The following equations hold for the first variations of H and A [33,34]:

$$\delta H = \frac{1}{2}[-\Delta\psi + (4H^2 - 2K)\psi] \tag{3}$$

$$\delta dA = -2\psi H dA \tag{4}$$

where K is the Gaussian curvature. From the divergence theorem, we have

$$\int \Delta\psi dA = 0 \tag{5}$$

Therefore, the following equation is satisfied for the first variation of the total mean curvature:

$$\begin{aligned} \delta \int H dA &= \int (\delta H dA + H \delta dA) \\ &= \int (2H^2 - K - 2H^2)\psi dA \\ &= -\int K\psi dA \end{aligned} \tag{6}$$

The first variation of the volume is obtained as

$$\delta V = \int \psi dA \tag{7}$$

Hence, the first variation of the energy functional $\tilde{E}$ is expressed as

$$\delta\tilde{E} = -\int (K - \bar{K})\psi dA \tag{8}$$

Accordingly, the stationary point of $\tilde{E}$ needs to satisfy the following Euler–Lagrange equation:

$$K - \bar{K} = 0 \tag{9}$$

This implies that when $\bar{K}$ is specified as a target Gaussian curvature, minimization of the energy functional $\tilde{E}$ leads to the surface with the constant Gaussian curvature $\bar{K}$, if $\tilde{E}$ is convex with respect to the shape variation in the normal direction. It is known that the energy functional E in Eq. (1) is elliptic only when $\beta^2 + \alpha\gamma > 0$ is satisfied [35]. Tellier et al. [23] stabilized E for a discrete surface by replacing the total area with its square. In this paper, we consider the case of $\beta = 0$ and $\alpha = 1$; therefore, $\bar{K} > 0$ is assumed, because the Gaussian curvature flow can be applied only to a convex surface as noted by Zhao and Xu [17].

It is easily shown by minimizing $\delta\tilde{E}$ under constraint on the norm of ψ that the gradient of $\tilde{E}$ is $-(K - \bar{K})$; thus, a CGC surface can be obtained using the steepest descent method by moving the surface in the direction of the outward unit normal vector multiplied by $-(K - \bar{K})$, which is called Gaussian curvature flow.

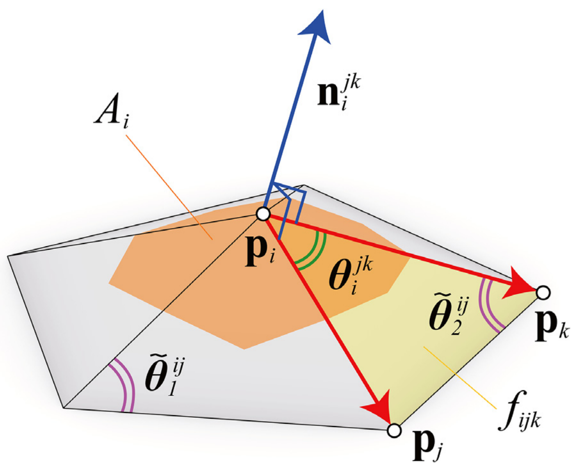

A_i
$\mathbf{n}_i^{jk}$
$\mathbf{p}_i$
$\boldsymbol{\theta}_i^{jk}$
$\tilde{\boldsymbol{\theta}}_2^{ij}$
$\mathbf{p}_k$
$\tilde{\boldsymbol{\theta}}_1^{ij}$
f_{ijk}
$\mathbf{p}_j$

Fig. 1. Definitions of vectors and angles at a vertex of a triangular mesh surface.

3. Gaussian curvature flow for discrete surface

In this section, the formulas used for the discrete Gaussian curvature flow are explained for completeness of the paper.

3.1. Geometrical properties of a triangular mesh surface

We consider a discretized closed or boundary-fixed open surface M with triangular faces (i.e., triangulated polyhedron) in 3-dimensional space. The position vector of vertex i is denoted as $\mathbf{p}_i \in \mathbb{R}^3$. The value corresponding to vertex i is indicated by the subscript i in the following.

Discrete Gaussian curvature K_i

Fig. 1 shows the geometrical properties used for computing the discrete Gaussian curvature flow. The area of Voronoi region of vertex i is denoted by A_i, and the triangular face f_{ijk} consists of the vertices i, j, k aligned anti-clockwise when viewed from the outside of the surface. The angle between the vectors $\mathbf{p}_j - \mathbf{p}_i$ and $\mathbf{p}_k - \mathbf{p}_i$ of the face f_{ijk} is denoted by θ_i^{jk}.

Among several definitions of the discrete Gaussian curvature, the angle defect defined as follows is used in this study:

$$K_i = \frac{1}{A_i}\left(2\pi - \sum_{f_{ijk}\in\mathcal{T}_i} \theta_i^{jk}\right) \tag{10}$$

where $\mathcal{T}_i$ denotes the set of indices of faces connected to vertex i. Note that the Gaussian curvature is to be computed at the internal vertices only.

Unit outward normal vector $\mathbf{n}_i$

An outward normal vector $\mathbf{n}_i^{jk}$ of the face f_{ijk} at vertex i is defined as

$$\mathbf{n}_i^{jk} = \frac{1}{2}(\mathbf{p}_j - \mathbf{p}_i) \times (\mathbf{p}_k - \mathbf{p}_i) \tag{11}$$

The length of $\mathbf{n}_i^{jk}$ is equal to the area of the face f_{ijk}. A normal vector $\mathbf{n}_i^0 \in \mathbb{R}^3$ at vertex i is computed as

$$\mathbf{n}_i^0 = \sum_{f_{ijk}\in\mathcal{T}_i} \mathbf{n}_i^{jk} \tag{12}$$

and the unit normal vector $\mathbf{n}_i$ at vertex i of the surface is obtained as

$$\mathbf{n}_i = \frac{\mathbf{n}_i^0}{\|\mathbf{n}_i^0\|} \tag{13}$$

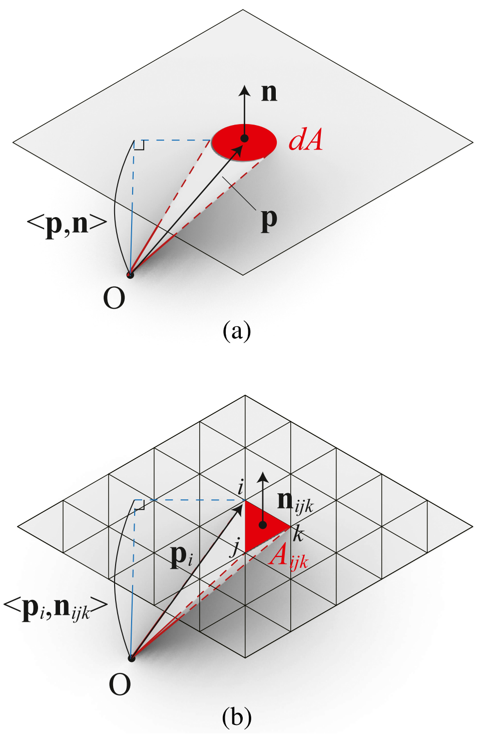


Fig. 2. Computation of volume bounded by the surface; (a) infinitesimal region of a continuous surface, (b) triangular element of a discrete surface.

Note that $\mathbf{n}_i^0$ is a normal vector as the mean of the normal vectors of the faces in $\mathcal{T}_i$ weighted by the area.

Total mean curvature $\int HdA$

Let $\mathcal{N}_i$ denote the set of indices of vertices adjacent to vertex i. The mean curvature vector $\mathbf{H}_i \in \mathbb{R}^3$ at vertex i is computed as follows using the discrete Laplace–Beltrami operator [36]:

$$\mathbf{H}_i = \frac{1}{A_i}\sum_{j\in\mathcal{N}_i}\frac{1}{2}(\cot\tilde{\theta}_1^{ij} + \cot\tilde{\theta}_2^{ij})(\mathbf{p}_j - \mathbf{p}_i) \tag{14}$$

where $\tilde{\theta}_1^{ij}$ and $\tilde{\theta}_2^{ij}$ are defined as shown in Fig. 1. Half of the norm of $\mathbf{H}_i$ corresponds to the absolute value of the mean curvature H_i as $|H_i| = \frac{1}{2}\|\mathbf{H}_i\|$ [37]. The sign of H_i is defined by the dot product between the unit outward normal vector $\mathbf{n}_i$ and the mean curvature vector $\mathbf{H}_i$; i.e., the mean curvature is positive/negative if the surface is concave/convex in the outward normal direction.

The total mean curvature of the continuous surface is equivalently computed as the summation of H_i multiplied by A_i over the discrete surface; i.e., the following relation is approximately satisfied:

$$\int HdA = \sum_{i\in\mathcal{N}} H_i A_i \tag{15}$$

where $\mathcal{N}$ is the set of indices of all vertices.

Volume bounded by the surface V

Figs. 2(a) and (b) show the volume corresponding to an infinitesimal region of a continuous surface and a triangular face of a discrete surface, respectively. For an open or a boundary-fixed continuous surface, the volume of the surface can be computed by surface integration over conical elements whose base is an infinitesimal surface area and the apex is the origin of the space [14,38]. Similarly, the volume bounded by a triangular mesh can be computed by summation of the triangular pyramids whose bases are their face element and apexes are the origin of the global coordinate system. Fig. 2(b) illustrates the triangular pyramid corresponding to face f_{ijk}, where A_{ijk} and $\mathbf{n}_{ijk} \in \mathbb{R}^3$ are the area and the unit normal vector of face f_{ijk}, respectively. By definition, the volume of the pyramid V_{ijk} is computed as:

$$V_{ijk} = \frac{1}{3}\langle \mathbf{p}_i, \mathbf{n}_{ijk}\rangle A_{ijk} \tag{16}$$

where $\langle \mathbf{a}, \mathbf{b}\rangle$ is the dot product of the vectors $\mathbf{a}$ and $\mathbf{b}$, and the unit normal vector $\mathbf{n}_{ijk}$ is given as

$$\begin{aligned}\mathbf{n}_{ijk} &= \frac{(\mathbf{p}_j - \mathbf{p}_i)\times(\mathbf{p}_k - \mathbf{p}_i)}{\|(\mathbf{p}_j - \mathbf{p}_i)\times(\mathbf{p}_k - \mathbf{p}_i)\|} \\ &= \frac{1}{2A_{ijk}}\left[\mathbf{p}_j\times\mathbf{p}_k + (\mathbf{p}_k - \mathbf{p}_j)\times\mathbf{p}_i\right]\end{aligned} \tag{17}$$

Because $(\mathbf{p}_k - \mathbf{p}_j)\times\mathbf{p}_i$ is perpendicular to $\mathbf{p}_i$, the dot product of these two vectors vanishes as

$$\langle(\mathbf{p}_k - \mathbf{p}_j)\times\mathbf{p}_i, \mathbf{p}_i\rangle = 0 \tag{18}$$

From Eqs. (16)–(18), the volume bounded by the discrete surface M is obtained as

$$V = \frac{1}{6}\sum_{f_{ijk}\in M}\langle \mathbf{p}_i, \mathbf{p}_j\times\mathbf{p}_k\rangle \tag{19}$$

In the numerical examples, we consider boundary-fixed surfaces. For the case of isolated fixed points with variable boundary edges, dummy meshes are generated outside the boundary.

3.2. Discrete Gaussian curvature flow

Discrete Gaussian curvature flow is a gradient flow of the energy functional $\tilde{E}$ that transforms a polyhedron with respect to its discrete Gaussian curvature. The gradient flow is discretized here, instead of discretizing the energy functional and differentiating it to obtain the gradient flow.

Let K_i, $\bar{K}_i$, and $\mathbf{n}_i \in \mathbb{R}^3$ denote the Gaussian curvature, its target value, and the unit outward normal vector at vertex i, respectively, which have been defined in Section 3.1. Although $\bar{K}_i$ should be the same at all vertices, we use the subscript i to indicate a value at vertex i. The gradient of $\tilde{E}$ is obtained as

$$\Delta\mathbf{K}_i = (K_i - \bar{K}_i)\mathbf{n}_i \tag{20}$$

and the locations of vertices are updated using the discrete Gaussian flow as

$$\mathbf{p}_i \longleftarrow \mathbf{p}_i - \Delta\mathbf{K}_i \tag{21}$$

However, to prevent oscillation and divergence, a small positive parameter μ is incorporated as

$$\mathbf{p}_i \longleftarrow \mathbf{p}_i - \mu\Delta\mathbf{K}_i \tag{22}$$

The strategy for determination of μ is explained in Section 4. The surface converges to a CGC surface by iteratively using Eq. (22) for updating the locations of vertices.

4. Optimization method for generating piecewise CGC surface

In this section, two approaches are presented for generating piecewise CGC surfaces divided by internal boundaries.

As noted in Introduction, the Gaussian curvature flow is not very stable compared with the mean curvature flow. Here, a simple filter that is used for structural topology optimization is utilized to maintain smoothness of the surface. The weighted average of the gradients of the Gaussian curvature among the neighborhood vertices is used at each vertex. Let d_{ij} denote the

Euclidean distance between vertices i and j. The upper-bound value $\bar{d}$ is given for d_{ij} to define a modified distance $\tilde{d}_{ij}$ as

$$\tilde{d}_{ij} = \min(d_{ij}, \bar{d}) \tag{23}$$

To generate piecewise CGC surfaces, the whole surface is divided into several regions by the edges along the internal boundary that is specified *a priori*. This does not mean that the locations of the vertices along the internal boundary are fixed. Let $\mathcal{N}_\mathrm{I}$, $\mathcal{N}_\mathrm{f}$, and $\mathcal{N}_\mathrm{p}$ denote the sets of indices of internal vertices, fixed (external boundary) vertices, and the vertices on the internal boundary, respectively. The locations of vertices in $\mathcal{N}_\mathrm{f}$ are fixed; however, those in $\mathcal{N}_\mathrm{p}$ are allowed to move in accordance with the adjacent internal vertices.

The weight coefficients w_{ij} are defined as

$$w_{ij} = \begin{cases} \left(\dfrac{\bar{d} - \tilde{d}_{ij}}{\bar{d}}\right)^p & \text{if } j \in \mathcal{N}_\mathrm{I} \\ 0 & \text{else} \end{cases} \tag{24}$$

where $p\ (> 0)$ is the parameter to determine relative importance of nearer vertices; we set $p = 3$ in the numerical examples. Consequently, effect of the gradients at the vertices in $\mathcal{N}_\mathrm{f}$ and $\mathcal{N}_\mathrm{p}$ are ignored. The weight coefficient w_{ij} is normalized to $\tilde{w}_{ij}$ as

$$\tilde{w}_{ij} = \frac{w_{ij}}{\sum_{j=1}^{n} w_{ij}} \tag{25}$$

where n is the number of vertices.

Using the weight coefficients, the smoothed gradient $\Delta\tilde{\mathbf{K}}_i$ is obtained as follows for the internal vertices:

$$\Delta\tilde{\mathbf{K}}_i = \sum_{j=1}^{n} \tilde{w}_{ij}\Delta\mathbf{K}_j \text{ for } i \in \mathcal{N}_\mathrm{I} \tag{26}$$

Note again that the smoothed gradient $\Delta\tilde{\mathbf{K}}_i$ does not depend on the gradients at the vertices in the external and internal boundaries.

An optimization problem is solved to obtain a CGC surface using the discrete Gaussian flow. The following two strategies are proposed for generating a piecewise CGC surface:

S1 The gradients of vertices along the internal boundary are multiplied by the parameter $s \in (0, 1)$ as

$$\Delta\tilde{\mathbf{K}}_i = s\sum_{j=1}^{n} \tilde{w}_{ij}\Delta\mathbf{K}_j \text{ for } i \in \mathcal{N}_\mathrm{p} \tag{27}$$

The vertices in $\mathcal{N}_\mathrm{p}$ move in the same manner as the adjacent internal vertices if s is close to 1. By contrast, those vertices move only slightly if s is close to 0.

S2 The gradients of vertices along the internal boundary are defined in the same manner as the internal boundary using Eq. (26). However, the update of vertex location is delayed for e steps; i.e., the locations of vertices along the internal boundary are fixed for the first e steps. It is shown in the numerical examples that a small e, e.g., $e = 5$, leads to significant difference in the final converged shapes.

The important feature of our optimization problem is that we minimize the energy functional to obtain the solution satisfying the stationary condition that corresponds to the CGC surface; i.e., we do not minimize the error of the Gaussian curvature from the target value. Another important feature is that we do not have to compute the gradients (sensitivity coefficients) of the Gaussian curvature analytically differentiating the governing equations. The piecewise CGC surface is obtained by simply modifying the locations of vertices using the discrete Gaussian curvature flow. However, in addition to the filtering technique, some techniques should be introduced to improve the computational efficiency.

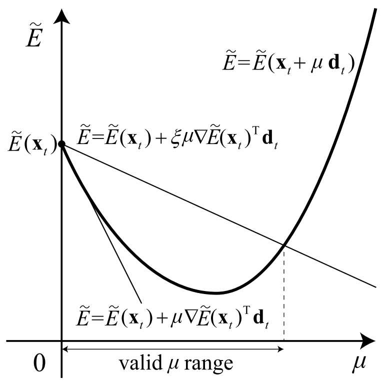

$\tilde{E}$
$\tilde{E}=\tilde{E}(\mathbf{x}_t+\mu\,\mathbf{d}_t)$
$\tilde{E}(\mathbf{x}_t)$
$\tilde{E}=\tilde{E}(\mathbf{x}_t)+\xi\mu\nabla\tilde{E}(\mathbf{x}_t)^\mathrm{T}\mathbf{d}_t$
$\tilde{E}=\tilde{E}(\mathbf{x}_t)+\mu\nabla\tilde{E}(\mathbf{x}_t)^\mathrm{T}\mathbf{d}_t$
0
valid μ range
μ

Fig. 3. Illustration of Armijo condition for line search.

As seen from Eq. (8), the norm of variation of the vertex position vector decreases as K_i converges to the target value $\bar{K}_i$ when a constant value is used for μ. The variation may be too large and the solution may diverge if the initial surface is not close to the CGC surface. Furthermore, unnecessarily many iterations may be needed when the solution approaches the final CGC surface. Therefore, we utilize line search method to dynamically adjust the amount of modification at each step of iteration with the constant value of the gradient.

Among various strategies of line search developed in the field of mathematical programming, the Armijo condition as illustrated in Fig. 3 is used in the numerical examples in the same manner as Tellier et al. [23]. Let $\mathbf{x} \in \mathbb{R}^m$ denote the vector of m design variables consisting of the coordinates of the internal vertices and the vertices in the internal boundaries. Important point here is that the value of the energy functional $\tilde{E}(\mathbf{x})$, which is the objective function of the optimization problem, is explicitly used for the termination condition of the line search.

The value of $\mathbf{x}$ at the beginning of the iterative step k and its search direction vector are denoted by $\mathbf{x}_k$ and $\mathbf{d}_k \in \mathbb{R}^m$, respectively. Note that $\mathbf{d}_k$ consists of the gradients $\Delta\mathbf{K}_i$ at the internal vertices. Using a constant $\xi \in (0, 1)$ and the gradient $\nabla\tilde{E}(\mathbf{x}_k)$ of $\tilde{E}(\mathbf{x})$ at $\mathbf{x} = \mathbf{x}_k$, the Armijo condition for μ is given as

$$\tilde{E}(\mathbf{x}_k + \mu\mathbf{d}_k) \le \tilde{E}(\mathbf{x}_k) + \xi\mu\langle\nabla\tilde{E}(\mathbf{x}_k), \mathbf{d}_k\rangle \tag{28}$$

Furthermore, to prevent divergence due to too large variation of the surface shape, the following criterion is added for determination of the step length, assuming that a CGC surface is generated from an initial surface with small Gaussian curvature:

$$K_i \le \zeta\bar{K}_i \ \text{ if } \ i \in \mathcal{N}_\mathrm{I} \tag{29}$$

where ζ is a constant slightly larger than 1.

In summary, the largest value of μ satisfying Eqs. (28) and (29) is determined at each iteration step using a line search. In the following examples, μ starts from a sufficiently large value μ_0 and is multiplied by a parameter $\tau \in (0, 1)$ until Eqs. (28) and (29) are satisfied or the number of substeps reaches the upper bound q. In the numerical examples, convergence property is also checked by the following error norm of the Gaussian curvature from the target value:

$$\kappa = \sqrt{\sum_{i\in\mathcal{N}_\mathrm{I}}(K_i - \bar{K}_i)^2} \tag{30}$$

Since the proposed method is based on energy minimization, it is guaranteed that the energy always decreases at each iteration using the steepest descent method with line search, and

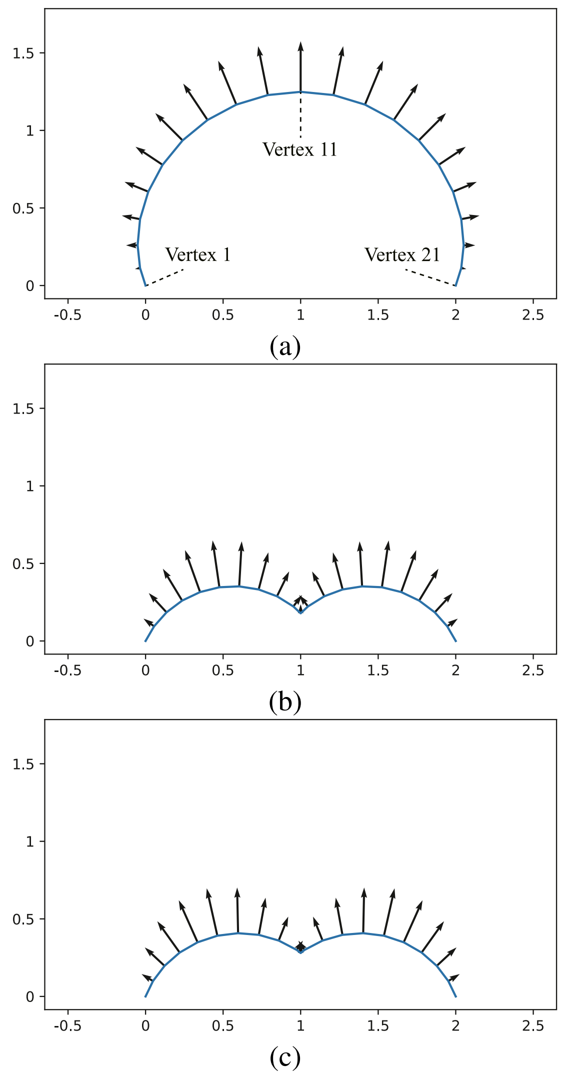


Fig. 4. Piecewise smooth curves with constant curvature (C1); (a) without internal boundary, (b) S1 with $s = 0.8$, (c) S2 with $e = 3$.

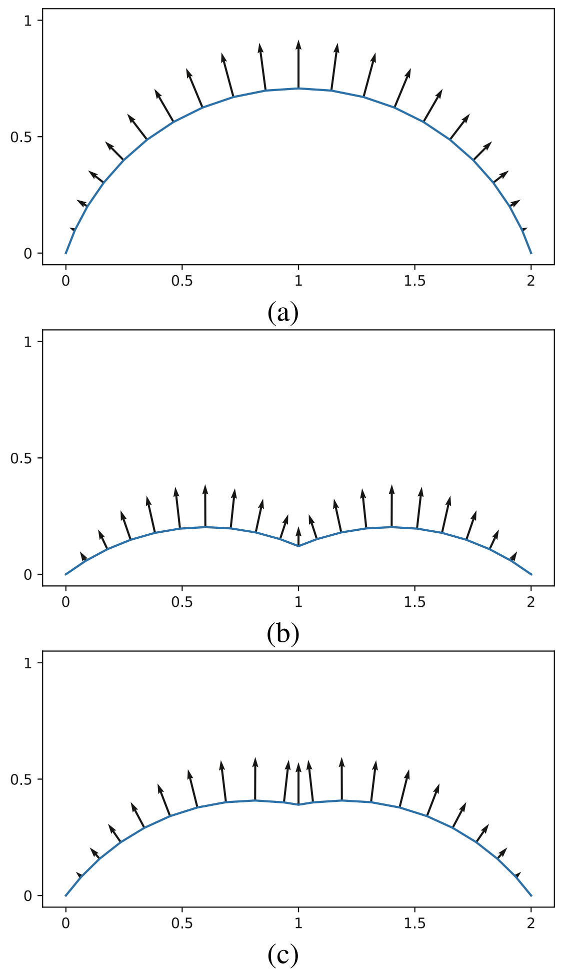


Fig. 5. Piecewise smooth curves with constant curvature (C2); (a) without internal boundary, (b) S1 with $s = 0.8$, (c) S2 with $e = 3$.

terminates at a local minimum. Accordingly, the convergence to the CGC surface is theoretically guaranteed. However, the error of Gaussian curvature from the target value slightly oscillates due to numerical error. The convergence theory for the Gaussian curvature flow starting from a convex and closed initial surface was studied in Ref. [39]. Although a theoretical convergence analysis is difficult for the proposed method for application to surfaces with internal and external boundaries, a linear convergence of the energy functional is verified in the numerical examples.

5. Numerical example

Several piecewise CGC surfaces are generated using the proposed method. The units are omitted because they are not important to describe the methodology. The computational cost and accuracy are discussed, and parameter assignment is explained in Section 5.8.

5.1. Example 1: Two dimensional curve with piecewise constant curvature

To verify the proposed approach in Section 4 for generating a CGC surface with internal boundary, a piecewise smooth curve with constant curvature is investigated on the 2-dimensional (X_1, X_2)-plane. The two strategies S1 and S2 are utilized for generating piecewise smooth arches with curvature discontinuity at the center vertex.

The line is divided into 20 edges, i.e., the line has 21 vertices. The initial shape is the straight line connecting the fixed vertices 1 and 21 indicated in Fig. 4(a) and located at $(X_1, X_2) = (0, 0)$ and (2,0), respectively. The edge j ($j = 1, \dots, 20$) connects vertices j and $j + 1$, and vertex 11 is located at the center as indicated in Fig. 4(a). Let L_i and Φ_i denote the length and the anticlockwise angle from the horizontal X_1-axis of edge i. We consider the following two types of definition of the curvature k_i at vertex i:

- C1: $k_i = \Phi_{i-1} - \Phi_i$
- C2: $k_i = \frac{\Phi_{i-1}}{L_{i-1}} - \frac{\Phi_i}{L_i}$

We do not discuss the proper definition of the curvature, because the purpose here is to demonstrate that different curve shapes are generated using the different definitions of the curvature. Therefore, we simply show the results using C1 and C2. For simplicity, the filter and the line search explained in Section 4 are not used here, and the gradient at vertex 11 is computed as the mean of those at vertices 10 and 12. The constant value $\mu = 0.05$ is used for updating the vertex locations.

Consider C1 with the target curvature $\bar{k}_i = 0.2$. The shapes after 500 steps are shown in Figs. 4(a)–(c), where the curvature of internal vertices converged to 0.2 for all three cases. No internal

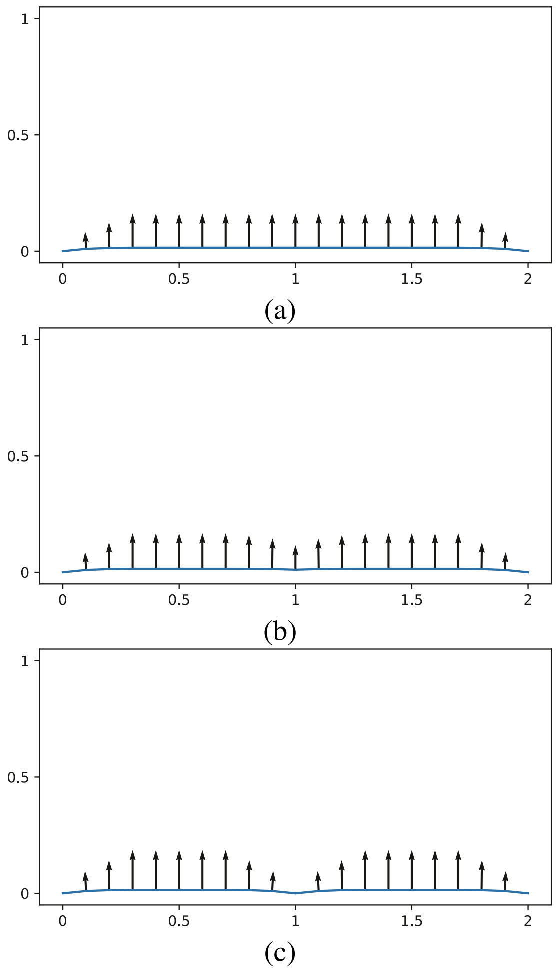


Fig. 6. Shapes at step 3 of C2; (a) without internal boundary, (b) S1 with $s = 0.8$, (c) S2 with $e = 3$.

boundary exists in Fig. 4(a). Since the angle between the adjacent edges is 0.2 rad. (= 11.46 deg.) and the number of internal vertices is 19, the angle between the edges 1 and 20 connected to the fixed vertices 1 and 21, respectively, is 3.8 rad. (= 217.72 deg.). Therefore, the curve becomes almost semicircle with a diameter larger than the distance between the fixed vertices.

Fig. 4(b) shows the result of strategy S1 with the parameter $s = 0.8$ at the center vertex 11. Since the number of internal vertices for each of the two arch segments is nine, the difference between the angles of the first and the last edges in each segment is 1.8 rad. (= 103.13 deg.). It should be noted that a completely different shape from the shape in Fig. 4(a) is generated by using the s value slightly smaller than 1. Fig. 4(c) shows the result of strategy S2 with the delay $e = 3$. It is notable that a shape similar to the strategy S1 with $s = 0.8$ is generated by delaying the update of the center vertex for only three steps.

The arrows in the figures represent the gradients of the energy functional, which are normalized by the maximum length in each figure. Note that the actual lengths of the gradients in Fig. 4 are very small. We can see from Figs. 4(b) and (c) that the lengths of the edges connected to the center vertex become very small especially if S2 is used.

Consider next C2 with $\bar{k}_i = 1.0$. The shape after 500 steps without internal boundary is shown in Fig. 5(a), where the curvatures at all internal vertices converged to 1.0. We can see by comparing Figs. 4(a) and 5(a) that a curve with a smaller height is obtained when C2 is used. Figs. 5(b) and (c) show the results of

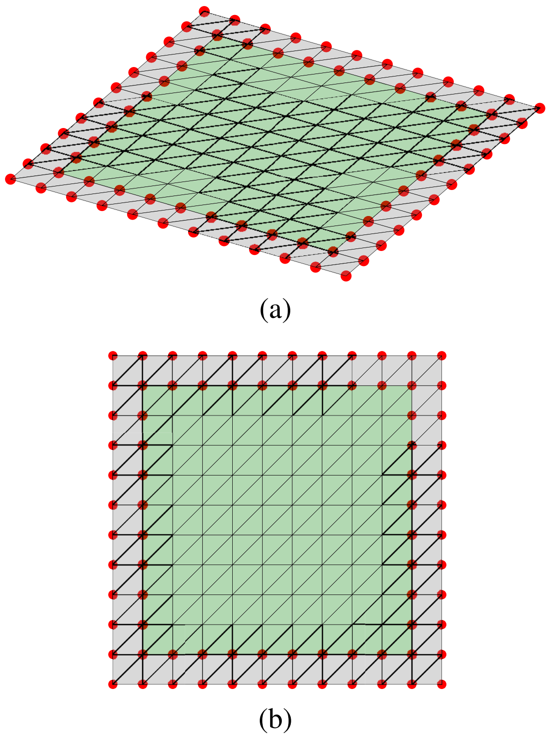


Fig. 7. Initial shape of Example 2; (a) diagonal view, (b) plan view.

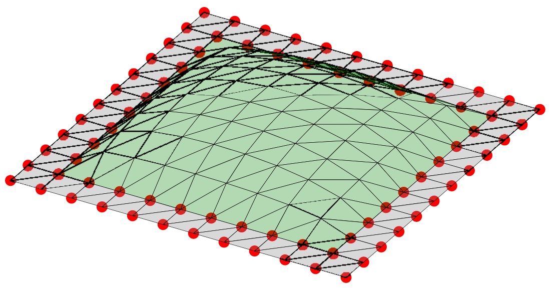
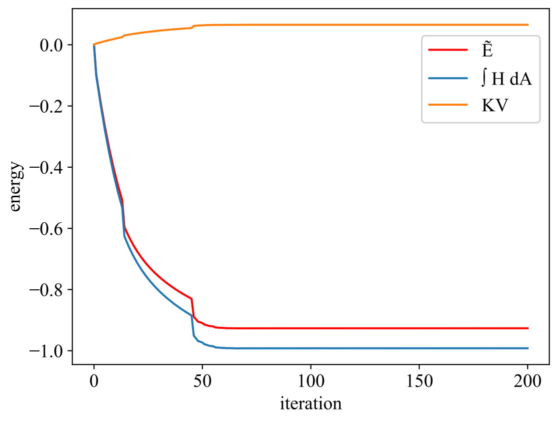


Fig. 8. Final shape of Example 2.

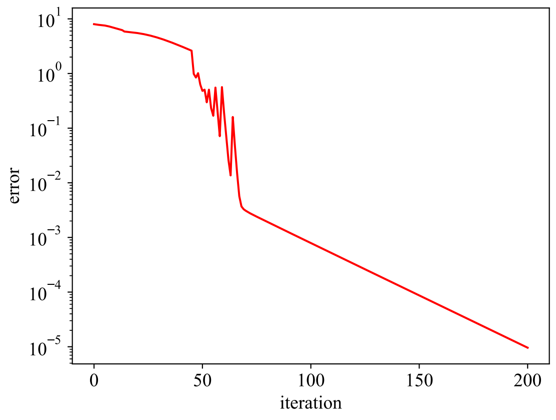
S1 with the parameter $s = 0.8$ and S2 with $e = 3$, respectively, at the center vertex 11. We can see from Figs. 4 and 5 that the differences of the total lengths in three cases are smaller for C2 than C1. The discontinuity in the angle at the center vertex is also small if C2 is used. The edge lengths are almost uniform including those of the edges connected to the center vertex. Figs. 6(a), (b), and (c) show the shapes and the gradients for the three cases at step 3 of updating the vertex locations from the horizontal line. We can see from Fig. 6(a) that the curvature propagates from each of the fixed vertices 1 and 21 if internal boundary does not exist. It is confirmed from Figs. 6(b) and (c) that the norm of gradient is small at the vertices near the center vertex that is considered as the internal boundary.

5.2. Example 2: 9 × 9 grid with fixed boundary

Consider a 9 × 9 grid as shown in Fig. 7, where each grid has a 0.1 × 0.1 square, and the red filled circle indicates a fixed vertex. The dummy meshes indicated with gray are added outside of the boundary so that the Gaussian curvature can be computed at the boundary vertices in the same manner as the internal vertices; consequently, the mesh has a 11 × 11 grid. Alternatively, the geodesic curvature may be used for the boundary vertices;



(a)



(b)

Fig. 9. Iteration history of Example 2; (a) objective function (energy functional) (b) error norm from the target value.

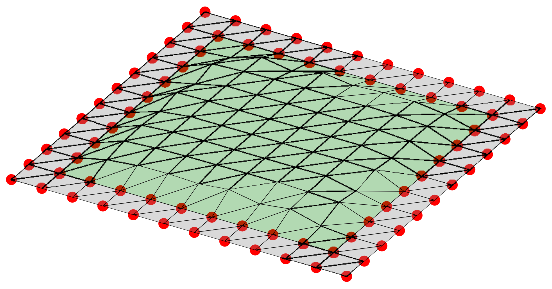
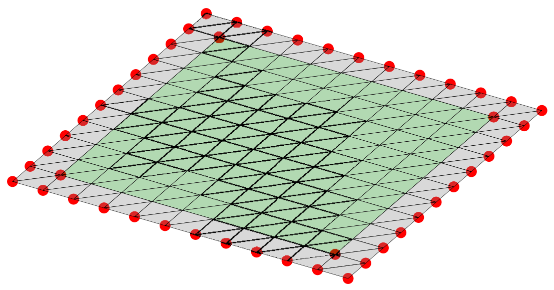


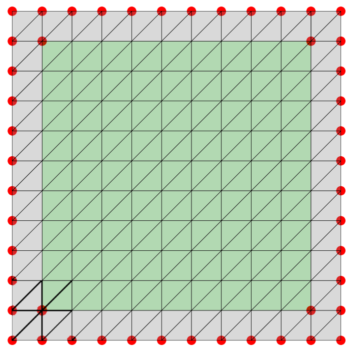
Fig. 10. Shape without filter of Example 2.

however, consistency between the geodesic curvature and the Gaussian curvature is not clearly defined for a polyhedral surface. All internal vertices are connected by six edges to ensure accuracy of the curvature values [20]; accordingly, the triangular faces are not symmetrically located, although the overall shape is symmetric. The target Gaussian curvature is $\bar{K}_i = 1.0$. The parameters for the line search and filter are $\xi = 0.01$, $\tau = 0.5$, $\mu_0 = 0.1$, $\zeta = 1.1$, $q = 5$, $p = 3$, and $\bar{d} = 0.5$. The convex surface with uniform Gaussian curvature 1.0 has been obtained within 200 steps of iteration.

The final shape is shown in Fig. 8, which verifies that a smooth shape can be generated using the filtering technique for the case without internal boundary. Convergence of the energy functional is confirmed in Fig. 9(a). The error of Gaussian curvature does not converge smoothly as shown in Fig. 9(b); however, the amount is very small because the vertical axis is in a logarithmic scale; i.e., a linear convergence is observed after 70 steps. A smooth surface could not be obtained if the filter is not used; Fig. 10 shows the shape at an intermediate step before the shape diverges.



(a)



(b)

Fig. 11. Initial shape of Example 3; (a) diagonal view, (b) plan view.

5.3. Example 3: 9 × 9 grid with fixed corners

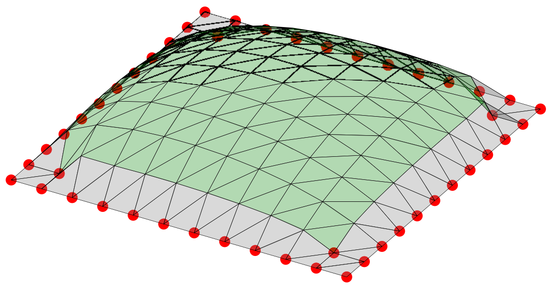
Consider next the same grid as Example 2 that is fixed at four corners. The initial shape is shown in Fig. 11. The dummy meshes are added also for this example. The target Gaussian curvature is $\bar{K}_i = 1.0$, and the same values as Example 2 are used for the parameters for line search and filter.

The final shape at the 200th step is shown in Fig. 12, which verifies that a smooth shape can be generated also for the case where four corners are fixed. Note that the shape near the corners is strongly dependent on the shapes of the triangular mesh. Convergence of the energy functional is confirmed in Fig. 13(a). The error of Gaussian curvature oscillates as shown in Fig. 13(b); however, a sufficiently small error is achieved at the final step. The shape without filter is shown in Fig. 14 before the divergence occurs. The use of filter is effective also for this example.

5.4. Example 4: 15 × 15 grid with internal boundary

Consider a 15 × 15 grid divided into 3 × 3 square regions that have piecewise smooth shapes. The size of each grid is 0.1 × 0.1. The initial shape is shown in Fig. 15. The four corners are fixed and the dummy meshes are added in the similar manner as Examples 2 and 3. The target Gaussian curvature is $\bar{K}_i = 4.0$, and the parameters for the line search and filter are the same as those for Examples 2 and 3. The thick lines in Fig. 15 represent the internal boundary, where the Gaussian curvature is not computed.

The final shape using the strategy S1 with $s = 0.8$ is shown in Fig. 16, which confirms that a piecewise smooth CGC surface


(a)

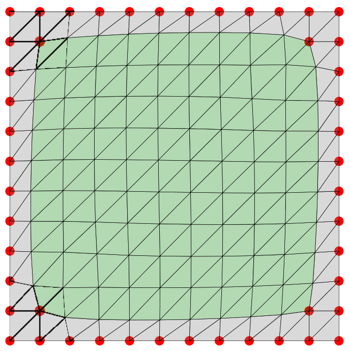
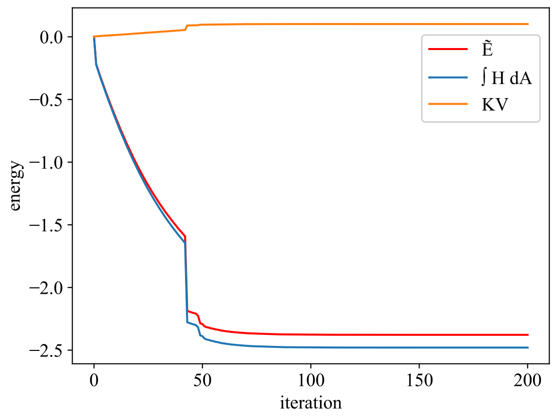
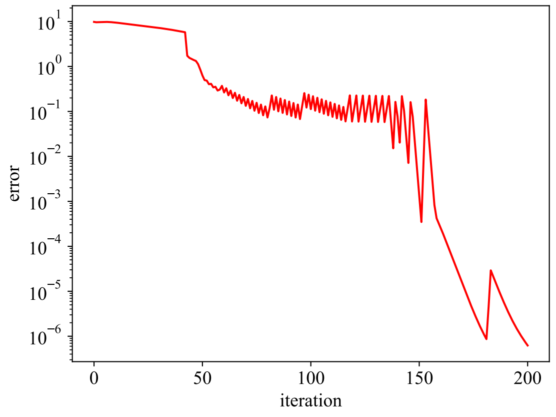

(b)

Fig. 12. Final shape of Example 3; (a) diagonal view, (b) plan view.



(a)



(b)

Fig. 13. Iteration history of Example 3; (a) objective function (energy functional) (b) error norm from the target value.

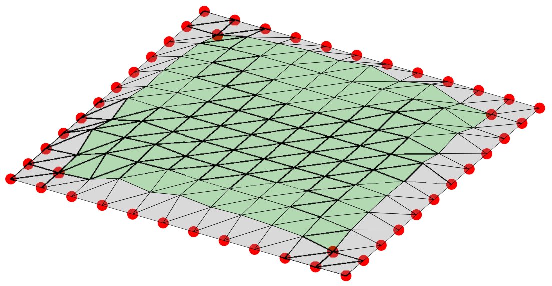
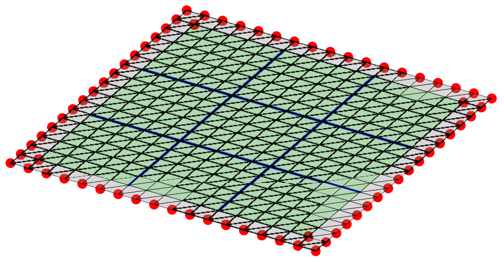


Fig. 14. Shape without filter of Example 2.


(a)

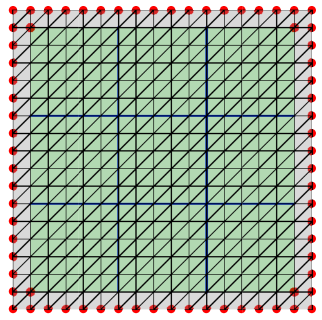

(b)

Fig. 15. Initial shape of Example 4; (a) diagonal view, (b) plan view.

can be successfully generated using the proposed method. Convergence of the energy functional is confirmed in Fig. 17(a). The energy functional rapidly decreases in the first 20 steps, and the line search terminates due to the limit of number of trials rather than satisfying the error bound after 27 steps of iterative steps. Although the error of Gaussian curvature oscillates between 30 and 50 steps as shown in Fig. 17(b), it finally decreases to a small value exhibiting a linear convergence.

Fig. 18(a) shows the shape obtained by assigning different target Gaussian curvatures in the regions. The values of $\bar{K}_i$ are 4.0 in the center region and 1.0 for the remaining eight regions. The strategy S2 is used with $s = 0.8$. Although the target curvatures should be theoretically the same at all vertices, a piecewise smooth surface with different Gaussian curvatures can be obtained using the proposed method.

The surface shape obtained using strategy S1 with $s = 0.5$, S2 with $e = 2$, and S2 with $e = 5$ are shown in Figs. 18(b), (c), and (d), respectively, for the case $\bar{K}_i = 4.0$ in the center region. We can confirm from these figures that similar shapes can be obtained using S1 and S2. Values of s or e should be appropriately assigned so that the desired internal boundary shape is generated.

5.5. Example 5: Developable surface

A developable surface is generated by assigning $\bar{K}_i = 0.0$ as the target Gaussian curvature. The parameters are the same as

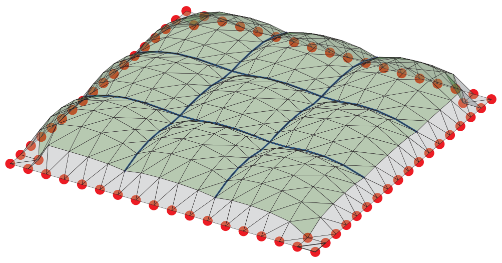
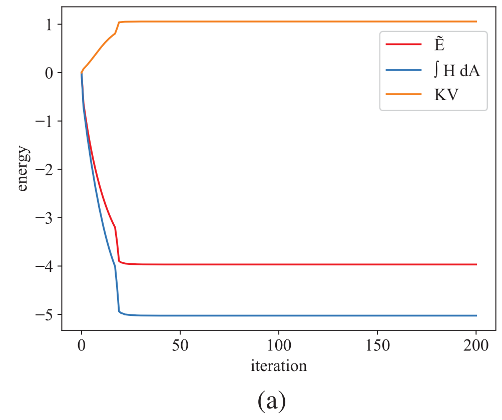


Fig. 16. Final shape of Example 4 (S1: $s = 0.8$).



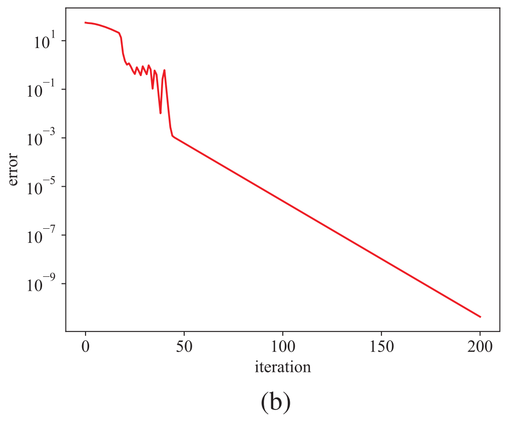


Fig. 17. Iteration history of Example 4; (a) objective function (energy functional) (b) error norm from the target value.

the previous examples except $\bar{d} = 1.5$ and $\mu_0 = 1.0$. The number of iterations is increased to 500. The initial shape is shown in Fig. 19, where the vertices on the top and bottom rings are fixed, and dummy meshes exist outside both of the two rings.

The final shape at the 500th step is shown in Fig. 20. We can see from the figure that a cone-shaped developable surface has been successfully generated. The convergence properties of the energy functional and the curvature error are shown in Fig. 21.

The error of the Gaussian curvature from the target value 0.0 is plotted in Fig. 22. The red and blue colors indicate positive and negative errors, and white dot means that the error is negligibly small. The boundary node is indicated by gray, because the error need not be evaluated.

5.6. Example 6: Surface with complex boundary

Applicability of the proposed method has been demonstrated in Examples 2–5 to generate CGC surfaces. In this section, we consider a surface with more complex boundary as shown in Fig. 23,

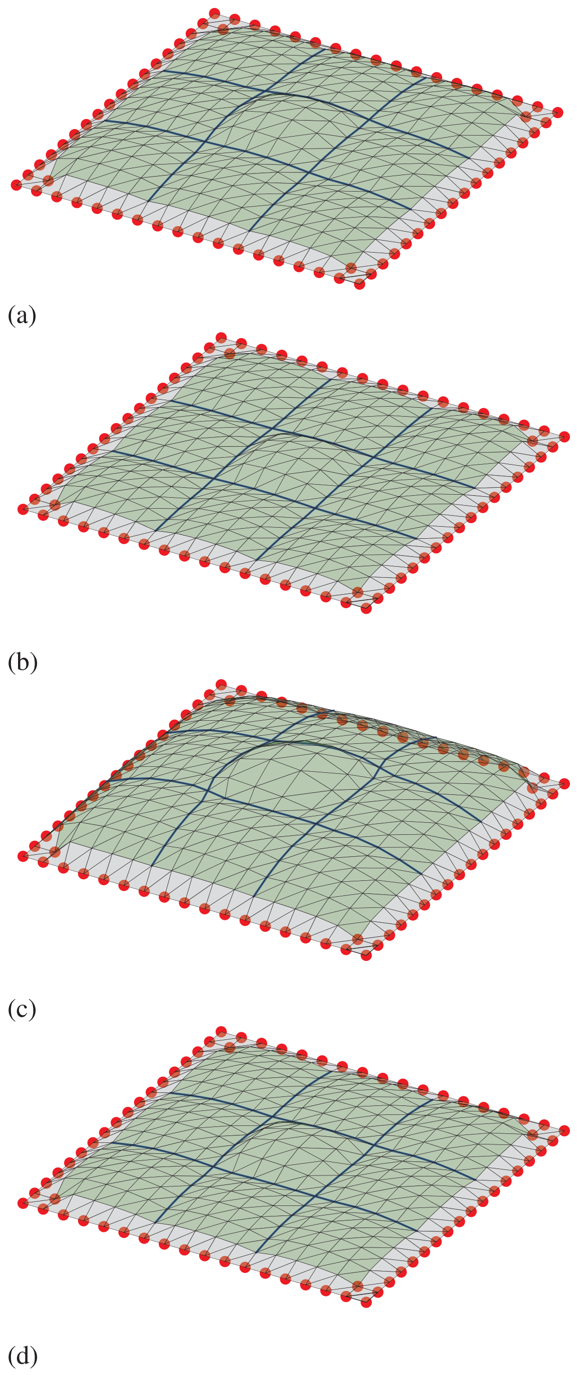


Fig. 18. Shape with different parameter values of Example 4 with $\bar{K}_i = 4.0$ in the center region and $\bar{K}_i = 1.0$ in the other regions; (a) strategy S1 with $s = 0.8$, (b) strategy S1 with $s = 0.5$, (c) strategy S2 with $e = 2$, (d) strategy S2 with $e = 5$.

which has 648 vertices and 1200 triangular faces. The ranges of x and y coordinates are $[-5.32, 5.94]$ and $[-5.34, 5.40]$, respectively, and the edge lengths distribute in the range $[0.215, 0.660]$. The blank circles with coordinate values in Fig. 23(b) are the control points of a closed quadratic Bézier curve for defining the boundary shape. No internal boundary is assigned in this example. The target Gaussian curvature is $\bar{K}_i = 0.03$. The parameters are the same as the previous examples except $\bar{d} = 1.0$ and $\mu_0 = 2.0$. The number of iterations is increased to 500.

The final shape at the 500th step is shown in Fig. 24. We can see from the figure that a convex surface has been successfully

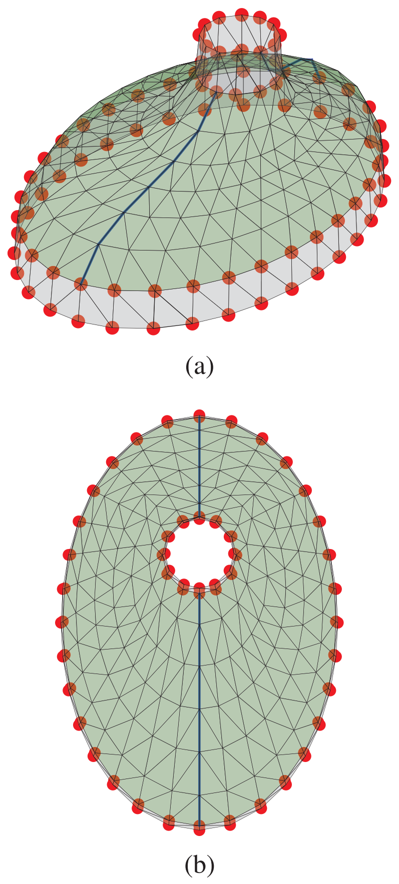


Fig. 19. Initial shape of Example 5; (a) diagonal view, (b) plan view.

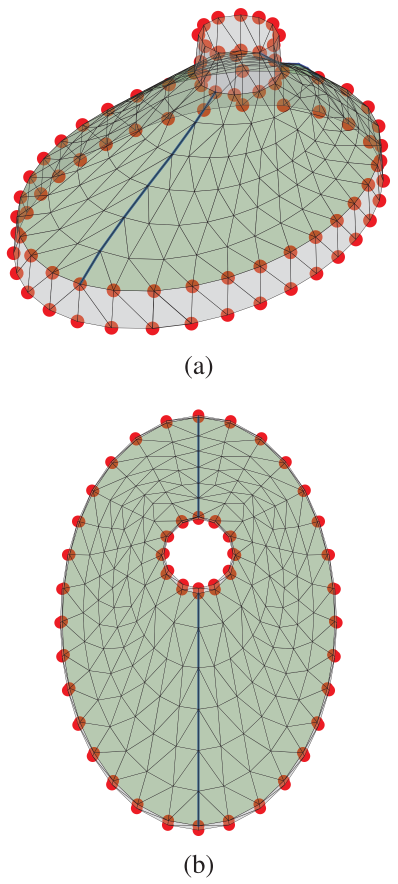


Fig. 20. Final shape of Example 5; (a) diagonal view, (b) plan view.

generated. The convergence properties of the energy functional and the curvature error are shown in Fig. 25. Although the energy functional converges to a vary small value, the error norm of the Gaussian curvature oscillates after 150 iteration steps.

The ratio of the error of Gaussian curvature to the target value is plotted in Fig. 26. The maximum absolute value is 6.031×10^{-4}, which may be small enough for a practical application.

5.7. Example 7: Surface with complex internal boundary

In this section, we consider a surface with more complex internal boundary as shown in Fig. 27, which has 389 vertices and 712 triangular faces.

The 1/8 region of the mesh is generated inside three curves: a line between (0,0,0) and (0,7,0); a circular arc passing (0,7,0), (3,7,0), and (5,5,0); and a quadratic Bézier curve whose control points are (5,5,0), (2.5,2.5,2.5), and (0,0,0). This sub-mesh is copied by mirroring with respect to the planes containing the lines $x = y$, $y = 0$ and $x = 0$ to generate a full mesh. The edges on the four planes of symmetry $x = y$, $x = -y$, $y = 0$, and $x = 0$ are regarded as the internal boundaries which are indicated by thick lines in Fig. 27. The dummy mesh is finally generated around the full mesh.

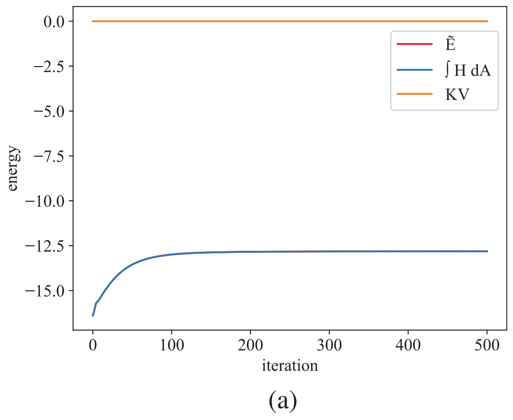
Two different cases of $\bar{K}_i = 0.05$ and 0.0 are considered. The maximum errors of the Gaussian curvature of the initial geometry are 7.349×10^{-2} and 2.349×10^{-2} for $\bar{K}_i = 0.05$ and 0.0, respectively. The parameters are the same as the previous examples except $\bar{d} = 5.0$ and $\mu_0 = 10.0$. The number of iterations is increased to 500. The strategy S1 is used with $s = 0.8$.

The final shapes at the 500th step for $\bar{K}_i = 0.05$ and 0.0 are shown in Figs. 28(a) and (b), respectively. We have good convergence property also for this example as shown in Figs. 29 and 30, and the maximum errors of the Gaussian curvature at the final step are 2.071×10^{-7} and 3.268×10^{-3} for $\bar{K}_i = 0.05$ and 0.0, respectively.

5.8. Discussions on parameter assignment and computational cost

Although there are many parameters in the proposed method, they can be assigned appropriately based on the following properties:

- Parameters $\xi = 0.01$, $\tau = 0.5$, $p = 3$, $q = 5$, and $\zeta = 1.1$ are used for all examples. The results are not sensitive to the values of these parameters; therefore, these values can be used for other examples. Careful tuning is required for the parameters μ_0, $\bar{d}$, e, and s.
- The parameter μ_0 depends on the mesh size. The update of the shape becomes very small and many steps are required before convergence, if μ_0 is too small. By contrast, the solution diverges if μ_0 is too large. The divergence may be prevented by assigning a large value to q, e.g., $q = 20$, to allow a small variation of vertex coordinates in the process of line search, which leads to an increase of computational cost for the line search.
- The parameter $\bar{d}$ is related to the size and density of the mesh. If $\bar{d}$ is too small, the filtering is not effective and small fluctuations may appear in the surface shape, which may lead to divergence of the solution. By contrast, if $\bar{d}$ is too large, distribution of the Gaussian curvature flow becomes uniform and many iteration steps will be needed before convergence. If an appropriate value of $\bar{d}$ is found for a



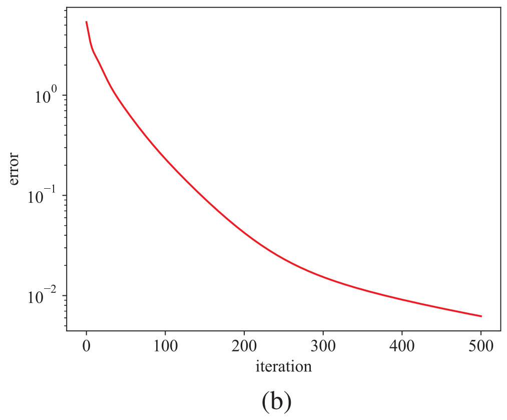


Fig. 21. Iteration history of Example 5; (a) objective function (energy functional) (b) error norm from the target value.

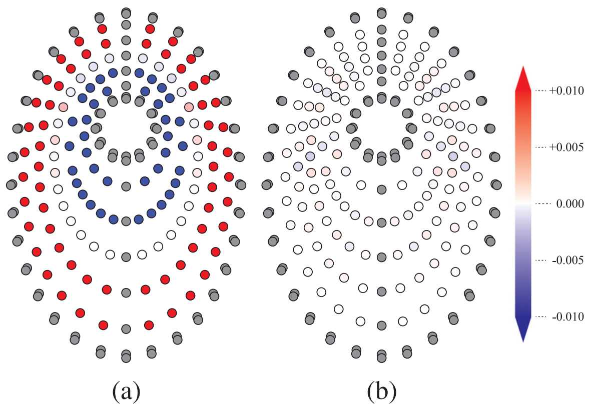


Fig. 22. Error of Gaussian curvature of Example 5; (a) initial shape, (b) final shape. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

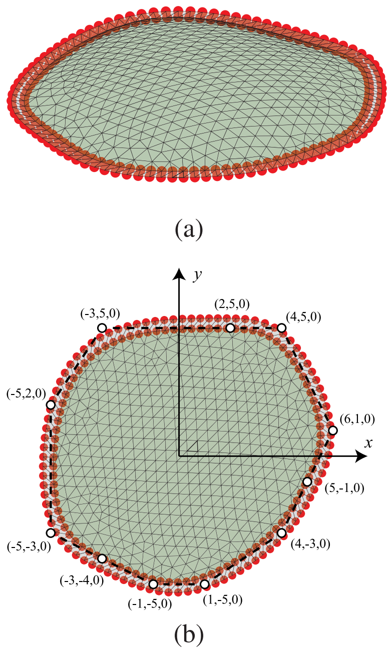


Fig. 23. Initial shape of Example 6; (a) diagonal view, (b) plan view.

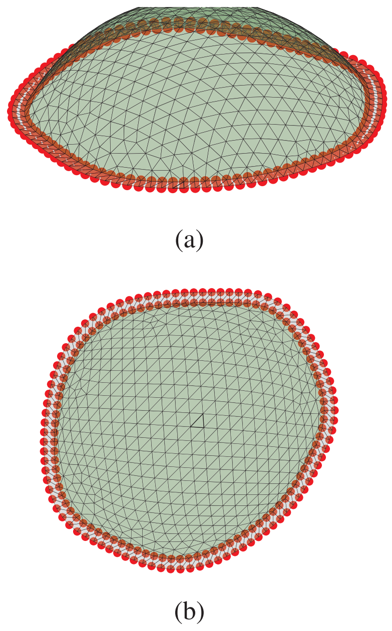


Fig. 24. Final shape of Example 6; (a) diagonal view, (b) plan view.

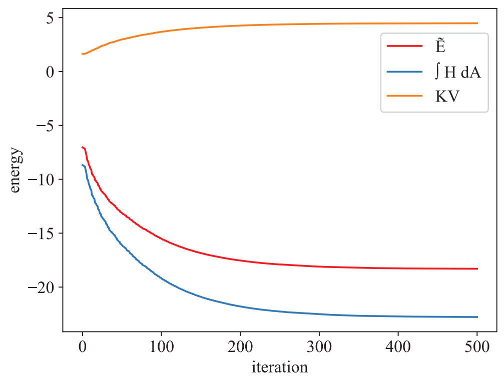
certain mesh size, its value is to be scaled in proportion to the mesh size to obtain the appropriate value for the different mesh size.

- The parameters e and s for generating the internal boundary should be modified after generating the surface for a trial parameter value. A smaller e or a larger s should be assigned if more variation is desired for the internal boundary. By contrast, a large e or a small s should be assigned to have the internal boundary close to its initial shape.

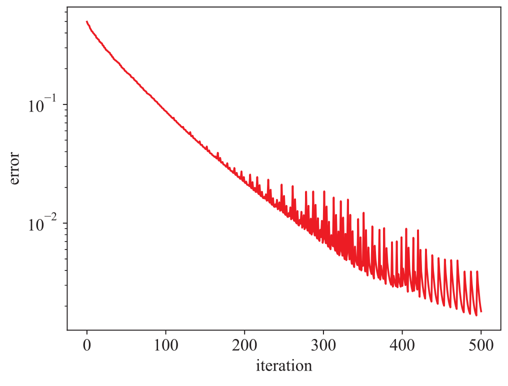
Table 1
Comparison of computational cost with respect to problem size; n, n_f: numbers of vertices and faces including dummy mesh, n_i: number of iterations.

Ex.	n	n_f	n_i	CPU time t	$(t/n/n_\mathrm{i}) \times 1000$
2	144	242	200	117.8	4.09
3	144	242	200	111.5	3.87
4	324	578	200	273.0	4.21
5	226	408	500	488.5	4.32
6	648	1200	500	963.8	2.97
7	389	712	500	870.0	4.58

Table 1 shows the computational cost in relation to the size of the model, where n and n_f are the numbers of vertices and faces including those of dummy mesh, and n_i is the number of iterations. A PC with Intel Core i7-8850H CPU 2.60 GHz is used for computation. The values of CPU time t of Examples 4 and 7



(a)



(b)

Fig. 25. Iteration history of Example 6; (a) objective function (energy functional) (b) error norm from the target value.

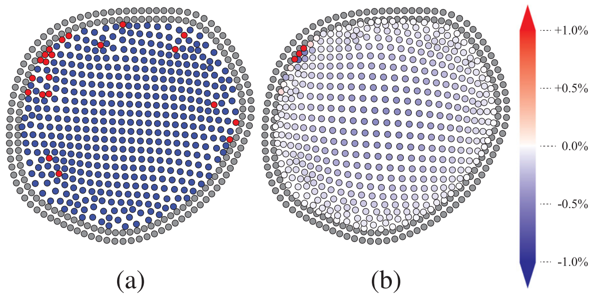
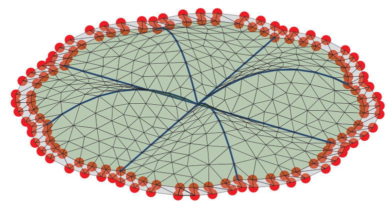
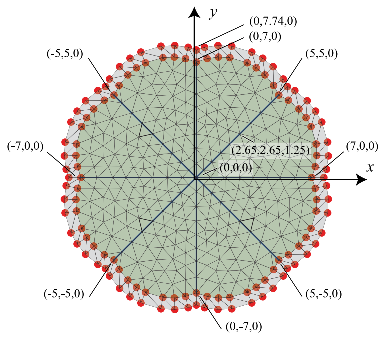


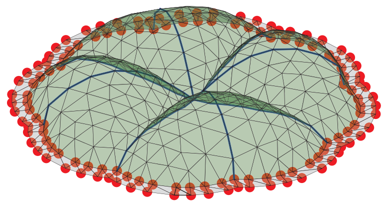
Fig. 26. Error of Gaussian curvature of Example 6; (a) initial shape, (b) final shape.


(a)



(b)

Fig. 27. Initial shape of Example 7; (a) diagonal view, (b) plan view.


(a)

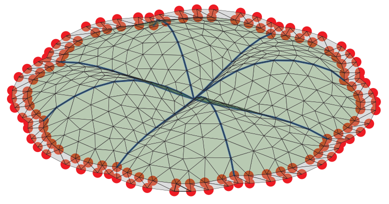

(b)

Fig. 28. Final shape of Example 7; (a) $\bar{K}_i = 0.05$, (b) $\bar{K}_i = 0.0$.

Table 2
Relation between mesh size and error in Gaussian curvature.

Grid	n	n_f	CPU tile t	$\max_{i\in\mathcal{N}_I} \lvert K_i - \bar{K}_i\rvert$
5 × 5	64	98	40.8	7.15×10^{-8}
10 × 10	169	288	130.6	2.15×10^{-9}
15 × 15	324	578	251.7	3.80×10^{-7}
20 × 20	529	968	417.5	4.11×10^{-4}

correspond to the results in Fig. 18(a) and Fig. 28(a), respectively. It may be observed from the table that the CPU time for a single step is almost proportional to the number of vertices, although a larger computational cost is needed for a larger model, if the required number of iterations increases as the number of vertices is increased.

Table 2 shows the CPU time and accuracy of Example 2 with different mesh sizes. The accuracy is measured by $\max_{i\in\mathcal{N}_I}|K_i - \bar{K}_i|$, which is the maximum absolute value of the error of the Gaussian curvature from the target value at the internal vertices. The number of iterations is 200, and the parameters are $\mu_0 = 100.0$ and $\bar{d} = 5.0$. The obtained shapes are not shown, because they are almost the same irrespective of the mesh size. It is seen from the table that the CPU time is almost proportional to n. Although the error in the Gaussian curvature gradually increases as n is increased, the error value for the 20 × 20 grid is still small enough compared with the target Gaussian curvature, which is equal to 4.0. The error may be reduced as the number of iterations is increased.

6. Conclusion

A method has been proposed for generating piecewise CGC surfaces based on direct minimization of an energy functional that is a simplified form of the energy functional, for which the stationary point is the LW surface. The stationary conditions of

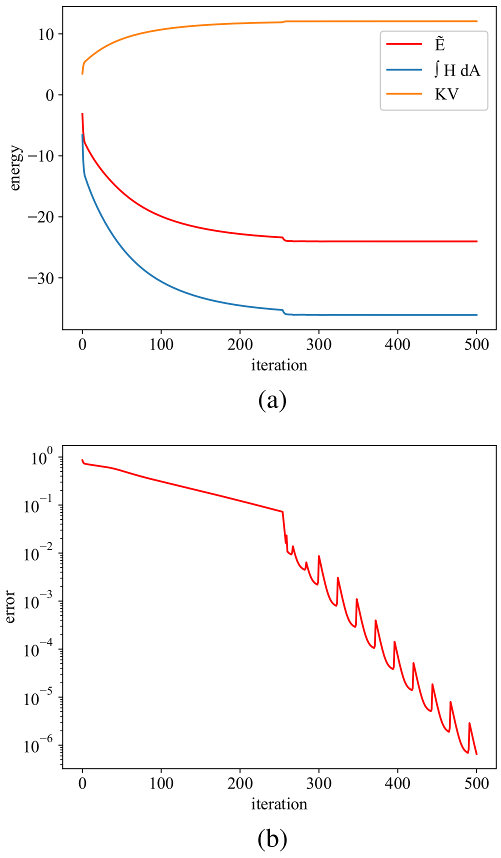


Fig. 29. Iteration history of Example 7 ($\bar{K}_i = 0.05$); (a) objective function (energy functional) (b) error norm from the target value.

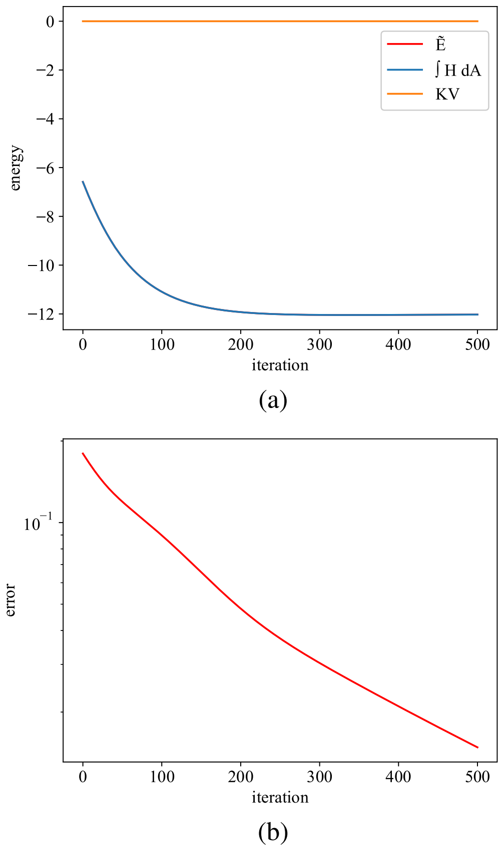


Fig. 30. Iteration history of Example 7 ($\bar{K}_i = 0.0$); (a) objective function (energy functional) (b) error norm from the target value.

the energy functional directly lead to the conditions of CGC, and the gradients of the functional are computed as the first variation of the functional. A filtering technique is proposed for smoothing the Gaussian curvature flow obtained from the gradient of the energy functional. The CGC surface is found by the steepest descent method with line search rather than minimizing the error of the Gaussian curvature. Therefore, the method is fully consistent with the original optimization problem for minimizing the energy functional, and is computationally efficient because it is not necessary to analytically differentiate the governing equations and solve a set of linear equations to obtain the gradients of the objective function.

Two methods have been proposed for generating piecewise CGC surface using the Gaussian curvature flow. The values of gradients at the vertices along the internal boundary are computed from the weighted sum of the gradients at the adjacent internal vertices. Discontinuity of the Gaussian curvature along the internal boundary is naturally generated by scaling the gradients at the vertices or delaying the update of vertex locations along the internal boundaries. This way, the Gaussian or geodesic curvature along the internal boundary need not be specified *a priori*. The various shapes of internal boundary can be obtained by assigning different parameter values for scaling or delaying the update of the vertices along the internal boundaries. Although the target curvatures should be theoretically the same at all vertices, a piecewise smooth surface with different Gaussian curvatures can be obtained using the proposed method. Effectiveness of adding the dummy meshes has also been demonstrated through the examples with variable boundary shape. Good convergence properties have been observed in all examples including the case with complex boundary shape. The computational cost for single step of updating the vertex locations is almost proportional to the number of vertices, although the number of steps may be increased for a complex surface with many vertices.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study is partially supported by JST CREST Grant No. JPMJCR1911, Japan. Valuable comments from Profs. Miyuki Koiso and Shizuo Kaji at Institute of Mathematics for Industry, Kyushu University are appreciated.

References

[1] Ramm E, Bletzinger KU, Reitinger R. Shape optimization of shell structures. Bull Int Assoc Shell Spat Struct 1993;34(2):103–21.
[2] Farin G. Curves and surfaces for CAGD: A practical guide. 5th ed.. Morgan Kaufmann; 2001.
[3] Nagata K, Honma T. Multi-objective optimization for free surface shell structure using particle intelligence with manipulation for decent solution search. J Struct Constr Eng 2013;78(690):1429–37, (in Japanese).
[4] Bandara K, Cirak F. Isogeometric shape optimisation of shell structures using multiresolution subdivision surfaces. Comput Aided Des 2018;95:62–71.
[5] Mesnil R, Douthe C, Baverel O, Léger B. Marionette meshes: modelling free-form architecture with planar facets. Int J Space Struct 2017;32(3–4):184–98.
[6] Pinkall U, Polthier K. Computing discrete minimal surfaces and their conjugates. Experiment Math 1993;2:15–36.

[7] Bobenko AL. Advances in discrete differential geometry. Springer; 2016.
[8] Rando T, Roulier JA. Designing faired parametric surfaces. Comput-Aided Des 1991;23:492–7.
[9] Fu J, Joshi SB, Simpson TW. Shape differentiation of freeform surfaces using a similarity measure based on an integral of Gaussian curvature. Comput Aided Des 2008;40:311–23.
[10] Fujita S, Ohsaki M. Shape optimization of free-form shells using invariants of parametric surface. Int J Space Struct 2010;25:143–57.
[11] M Ohsaki, M Hayashi. Fairness metrics for shape optimization of ribbed shells. J Int Assoc Shell Spat Struct 2000;41(1):31–9.
[12] Nakamura K, Ohsaki M, Cui J. Shape optimization of free-form shells consisting of developable surfaces. In *Proc. IASS symposium 2016, tokyo, int. assoc. shell and spatial struct. paper no. CS2E-1057*, 2016.
[13] Reilly RC. Variational properties of functions of the mean curvatures for hypersurfaces in space forms. J Differential Geom 1973;8:465–77.
[14] Barbosa JL, do Carmo CM. Stability of hypersurfaces with constant mean curvature. Math Z 1984;185:339–54.
[15] Gálvez JA, Martínez A, Milán F. Linear weingarten surfaces in R^3. Monatshefte Math 2003;138:133–44.
[16] Xu G, Zhanga Q. G^2 Surface modeling using minimal mean-curvature-variation flow. Comput Aided Des 2007;39:342–51.
[17] Zhao H, Xu G. Triangular surface mesh fairing via Gaussian curvature flow. J Comput Appl Math 2006;195:300–11.
[18] Rogers C, Schief WK. On the equilibrium of shell membranes under normal loading: Hidden integrability. Proc R Soc Ser A 2003;459:2449–62.
[19] Schief WK. Integrable structure in discrete shell membrane theory. Proc R Soc Ser A 2013;470:2013757.
[20] Xu Z, Xu G. Discrete schemes for Gaussian curvature and their convergence. Comput Math Appl 2009;57:1187–95.
[21] Burstall F, Hertrich-Jeromin U, Rossman W. Discrete linear weingarten surfaces. Nagoya Math J 2018;231:55–88.
[22] Tellier X, Douthe C, Baverel O, L. Hauswirth. Discrete CMC surfaces for doubly-curved building envelopes. In: Hesselgren L, et al., editors. Advances in architectural geometry 2018. Klein Publishing GmbH; 2018, p. 166–93.
[23] Tellier X, Douthe C, Hauswirth L, Baverel O. Linear Weingarten surfaces for conceptual design of double-curvature envelopes. In *Proc. fib conceptual design symposium 2019*, Madrid 2019.
[24] Aumann G. A simple algorithm for designing developable Bézier surfaces. Comput-Aided Geom Des 2003;20:601–19.
[25] Chu CH, Sequin CH. Developable Bézier patches: Properties and design. Comput Aided Des 2002;34:511–27.
[26] Braibant V, Fleury C. Shape optimal design using B-splines. Comput Methods Appl Mech Engrg 1984;44247–67.
[27] Bendsøe MP, Kikuchi N. Generating optimal topologies in structural design using a homogenization method. Comput Methods Appl Mech Engrg 1988;71(2):197–224.
[28] Bruns TE. A reevaluation of the SIMP method with filtering and an alternative formulation for solid–void topology optimization. Struct Multidiscip Optim 2005;30:428–36.
[29] Bendsøe MP, Sigmund O. Topology optimization: Theory, methods and applications. Springer; 2003.
[30] Sigmund O. On the design of compliant mechanisms using topology optimization. Mech Struct Mach 1997;25(4):493–524.
[31] Paris S, Hasinoff SW, Kautz J. Local Laplacian filters: Edge-aware image processing with a Laplacian pyramid. Commun ACM 2015;58(3): 81–91.
[32] Polthier K, Rossman W. Discrete constant mean curvature surfaces and its index. J Reine Angew Math 2002;54947–77.
[33] Pinl M, Trapp HW. Stationä re krü mmungsdichten auf hyperflächen des euklidischen Rn+1. Math Ann 1968;176:257–72, (in German).
[34] Chen BY. On a variational problem on hypersurfaces. J Lond Math Soc 1973;2(6):321–5.
[35] Hopf H. Differential geometry in the large. In: Seminar lectures new york university 1946 and stanford university 1956. Lecture notes in mathematics, Springer; 1989.
[36] Reuter M, Biasotti S, Giorgi D, Patané G, Spagnuolo M. Discrete Laplace-beltrami operators for shape analysis and segmentation. Comput Graph 2009;33:381–90.
[37] Meyer M, Desbrun M, Schröder, Barr AH. Discrete differential-geometry operators for triangulated 2-manifolds. In: Visualization and mathematics III. Mathematics and visualization, Springer; 2003.
[38] Bouzidi R, Van A. Numerical solution of hyperelastic membrane by energy minimization. Comp Struct 2004;82:1961–9.
[39] Tso K. Deforming a hypersurface by its Gauss-Kronecker curvature. Comm Pure Appl Math 1985;38:867–82.