Int J Adv Manuf Technol
DOI 10.1007/s00170-016-9800-0

ORIGINAL ARTICLE

Adaptive direct slicing of volumetric attribute data represented by trivariate B-spline functions

Yuhi Sasaki[1] · Masahito Takezawa[1] · Seungki Kim[1] · Hiroshi Kawaharada[1] · Takashi Maekawa[1]

Received: 23 June 2016 / Accepted: 21 November 2016

Abstract We introduce a framework for modeling of heterogeneous objects in terms of trivariate B-spline functions and a method for slicing them directly for additive manufacturing. We first fit volumetric attribute data associated with the geometry in terms of trivariate B-spline functions under the assumption that the geometric volume is already defined by the trivariate B-spline functions. Then, the B-spline volume and the associated attribute data are directly sliced without converting them to stereo-lithography format, resulting in a tool path with fewer errors. Furthermore, adaptive ray shooting is introduced in the slicing plane so that the zigzag tool path passes through all the tangential intersection points of the heterogeneous objects to represent all the feature points in the fabricated model. Complex examples illustrate the effectiveness of our method.

Keywords Additive manufacturing · Direct slicing · Trivariate B-spline function · Heterogeneous object

1 Introduction

The stereo-lithography (STL) format, which defines a tessellated CAD model in terms of Cartesian coordinates of the vertices of triangles and their unit normals, is the most commonly used file format in additive manufacturing (AM) [35]. Since the representation is linear, computation is easy and fast; however, it fails to accurately describe the nonlinear freeform geometry.

Furthermore, the conversion process from the B-spline CAD model to the tessellated model may induce unnecessary additional artifacts and may result in a large number of triangles that represent the small features in the original model [35]. Therefore, it is inadequate to approximate CAD models that contain freeform surfaces by triangular mesh, as the conversion process not only induces approximation errors, but also sacrifices the geometric and topological features that the original data possess. Additional errors are induced when the tessellated CAD models are sliced by planes to generate the tool path for AM machines. To overcome these defects of STL data, one needs to introduce direct slicing of the input CAD model without converting it to the tessellated model [12].

There is a new AM file format, additive manufacturing file (AMF) format, which can express heterogeneous object representation for AM processes. According to the document by ASTM [1], AMF format is able to represent smooth geometry using curved triangular patches. However, Paul and Anand [30] say that while generating the slices for manufacturing the part, the curved triangles used in AMF format are recursively subdivided back to planar triangles, and hence may lead to the same approximation errors presented in the STL file format.

Nevertheless, research on direct slicing of heterogeneous objects has not been conducted in the past. We consider two types of heterogeneous object design. The first type is a functionally gradient material (FGM) object in which different portions of the material do not have a clear boundary [10, 26, 32]. In FGM design, the variation in the composition and structure of the material inside the volume is determined using specific functions to change the material

✉ Takashi Maekawa
maekawa@ynu.ac.jp

1 Department of Mechanical Engineering, Yokohama National University, Yokohama, Japan

properties. The second is a multi-material (MM) object that is constructed by pieces of different materials with a clear boundary between them [10, 26, 32].

In this paper, we consider the abovementioned two types of heterogeneous object design when the specific function is a distance function from the boundary of the B-spline geometric volume. We introduce a novel method to model heterogeneous data by fitting volumetric attribute data by trivariate B-spline functions under the assumption that the geometric volume is already defined in terms of trivariate B-spline functions. We then slice the heterogeneous B-spline volume directly by ray shooting without converting it to a tessellated model to generate the adaptive zigzag tool path for AM machines.

The main contributions of this study are summarized as follows:

- A fast volumetric attribute data fitting in terms of trivariate B-spline functions based on iterative fitting method is introduced to construct a heterogeneous model.
- An adaptive direct slicing method is introduced so that the tool path passes through all the feature points of heterogeneous objects within the slicing plane.
- Most importantly, it brings together all these techniques into a robust procedure for the automatic generation of a tool path for heterogeneous objects.

This paper is organized as follows: In Section 2, we define the B-spline volume for geometries as well for attribute data. In Section 3, a volume approximation method based on iterative fitting algorithm is developed. In Section 4, adaptive direct slicing is introduced so that the tool path passes through all the feature points within the slice of heterogeneous objects. In Section 5, we demonstrate the effectiveness of the algorithms using some complex examples. Finally, we conclude the paper in Section 6.

2 Notation of B-spline volumes

In representing curves, surfaces, and volumes, B-spline /nonuniform rational B-splines (NURBS) have become the de facto industry standard for representing complex geometric information in the CAD/CAM/CAE field [31]. In this section, we define the B-spline volume for the geometry in Section 2.1, whereas that of the attribute data is defined in Section 2.2.

2.1 B-spline volumetric geometry model

Let us first introduce the notation used in the remainder of the paper. An order $\bar{K}$ B-spline is formed by joining several pieces of polynomials of degree $\bar{K}-1$ with at most $C^{\bar{K}-2}$ continuity at the breakpoints [29, 31]. A set of non-descending breakpoints $t_0 \leq t_1 \leq \ldots \leq t_r$ defines a *knot vector*

$$\mathbf{T} = (t_0, t_1, \ldots, t_r) \ , \tag{1}$$

which determines the parameterization of the basis functions.

Given a knot vector $\mathbf{T}$, the associated B-spline basis functions $N_{i,\bar{K}}(t)$ are defined as

$$N_{i,1}(t) = \begin{cases} 1 \ \text{for} \ t_i \leq t < t_{i+1} \\ 0 \ \text{otherwise} \ , \end{cases} \tag{2}$$

for $\bar{K} = 1$, and

$$N_{i,\bar{K}}(t) = \frac{t - t_i}{t_{i+\bar{K}-1} - t_i} N_{i,\bar{K}-1}(t) + \frac{t_{i+\bar{K}} - t}{t_{i+\bar{K}} - t_{i+1}} N_{i+1,\bar{K}-1}(t) \ , \tag{3}$$

for $\bar{K} > 1$ and $i = 0, 1, \ldots, r - \bar{K}$.

An order $(\bar{K}, \bar{L}, \bar{M})$ B-spline volume is a tensor product volume defined by a topologically rectangular parallelepiped set of control points ($\mathbf{P}_{ijk} \in \mathbf{R}^3$, $i = 0, \ldots, n_u$, $j = 0, \ldots, n_v$, and $k = 0, \ldots, n_w$) and three knot vectors ($\bar{\mathbf{u}} = (\bar{u}_0, \bar{u}_1, \ldots \bar{u}_{n_u+\bar{K}})$, $\bar{\mathbf{v}} = (\bar{v}_0, \bar{v}_1, \ldots, \bar{v}_{n_v+\bar{L}})$, and $\bar{\mathbf{w}} = (\bar{w}_0, \bar{w}_1, \ldots, \bar{w}_{n_w+\bar{M}})$) associated with each parameter u, v, and w, respectively. The corresponding integral B-spline volumetric geometry model is given by

$$\mathbf{V}(u, v, w) = \sum_{i,j,k=0}^{n_u,n_v,n_w} \mathbf{P}_{ijk} N_{i,\bar{K}}(u) N_{j,\bar{L}}(v) N_{k,\bar{M}}(w) \ . \tag{4}$$

2.2 B-spline volumetric attribute model

We assume that we are given the geometry of the volume and the attribute data $(\mathbf{x}_l, \mathbf{a}_l)$, where $\mathbf{x}_l \in \mathbf{R}^3, l = 0, \ldots, N$ are points within the volume and $\mathbf{a}_l \in \mathbf{R}^n$ are attribute data associated with $\mathbf{x}_l$, as shown in Fig. 1a. First, we find the parameter values (u_l, v_l, w_l) corresponding to $\mathbf{x}_l$, as shown in Fig. 1b, by solving the simultaneous 3×3 nonlinear equations:

$$\mathbf{V}(u_l, v_l, w_l) = \mathbf{x}_l \ , \tag{5}$$

which are based on Eq. 4 using Newton's method. The attribute model is a graph function, where the parameters (u_l, v_l, w_l) are the abscissae and $\mathbf{a}_l$ are the ordinates. Therefore, the orders and knot vectors of the attribute model can be different from those of the geometry model, as long as they share the same parameter values within the volume. In other words, a simple geometry with complicated attributes, and vice versa, can be modeled with the desired

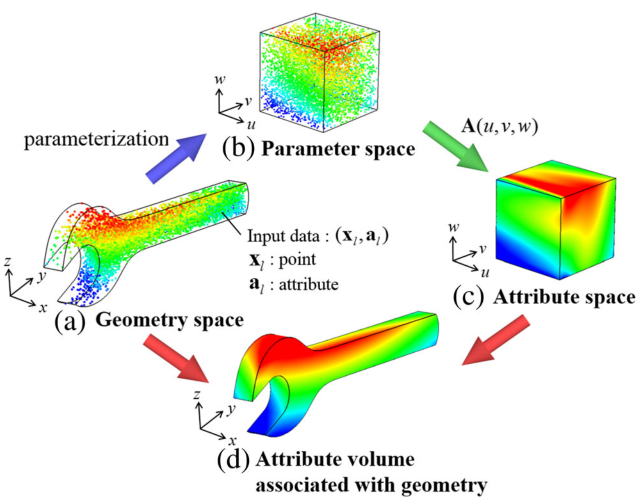
parameterization
(b) Parameter space
$\mathbf{A}(u,v,w)$
Input data : $(\mathbf{x}_l, \mathbf{a}_l)$
$\mathbf{x}_l$: point
$\mathbf{a}_l$: attribute
(a) Geometry space
(c) Attribute space
(d) Attribute volume associated with geometry

Fig. 1 B-spline volume for the attribute model. **a** Input data $(\mathbf{x}_l, \mathbf{a}_l)$ within the geometry. **b** Parameterization of input data $\mathbf{x}_l$. **c** Approximated attribute data by the trivariate B-spline function. **d** Attribute B-spline volume mapped to the geometry model

accuracy. An order (K, L, M) trivariate B-spline attribute model $\mathbf{A}(u, v, w)$ can be expressed as

$$\begin{aligned}\mathbf{A}(u, v, w) &= \sum_{i,j,k=0}^{m_u,m_v,m_w} \mathbf{A}_{ijk} N_{i,K}(u) N_{j,L}(v) N_{k,M}(w), \\ &= (A_1(u, v, w), \dots, A_n(u, v, w)) \,, \qquad (6)\end{aligned}$$

where $\mathbf{A}_{ijk} \in \mathbf{R}^n$, $i = 0, \dots, m_u$, $j = 0, \dots, m_v$, $k = 0, \dots, m_w$ are the B-spline ordinates and $\mathbf{u} = (u_0, u_1, \dots, u_{m_u+K})$, $\mathbf{v} = (v_0, v_1, \dots, v_{m_v+L})$, and $\mathbf{w} = (w_0, w_1, \dots, w_{m_w+M})$ are the three knot vectors associated with each parameter u, v, and w, respectively. We emphasize here that $\mathbf{A}_{ijk}$ are not the control points but the B-spline ordinates. Finally, we fit $(u_l, v_l, w_l, \mathbf{a}_l)$ using (6), as illustrated in Fig. 1c. Once the data are fit, we map them onto the geometry space (see Fig. 1d).

3 Fitting of volumetric attribute data

In this section, we study the construction of a B-spline volume from scientific volume data associated with the geometry using modeling and iterative fitting techniques. We distinguish between two types of volume fitting: geometry model fitting and attribute model fitting [25, 28]. Geometry fitting first associates suitable parameter values (u_l, v_l, w_l) for each input point $\mathbf{x}_l \in \mathbf{R}^3$ within the volume, known as parameterization of data, and outputs control points that represent the geometry, whereas attribute data fitting accepts the attribute data $\mathbf{a}_l \in \mathbf{R}^n$ associated with the geometric points, i.e., $(\mathbf{x}_l, \mathbf{a}_l)$, and generates B-spline ordinates of a graph function whose abscissae are (u_l, v_l, w_l). Note that the attribute model shares the same parameter values with the geometry model, as shown in Fig. 1.

3.1 Related work

Volume fitting is an important problem in many fields, including additive manufacturing of functionally gradient materials [14, 19], isogeometric analysis (IGA) [11], and visualization of measured or computed scientific data [28]. There are many advantages associated with attribute volume fitting, namely, storage reduction, fast execution, noise reduction, and robust visualization [24]. Standard B-spline surface fitting algorithms first conduct a parameterization of data and form a linear system with control points as unknowns. Recently, in contrast to the standard surface fitting methods, iterative fitting methods that do not require the solution of a linear system have received attention [2, 6, 15, 16, 18, 22, 37]. These methods employ a surprisingly simple geometric-based algorithm that iteratively updates the control points in a global manner based on a local parameter distance [18, 37] or a point surface distance [16, 22]. In contrast to research on curve/surface fitting by univariate/bivariate B-spline functions, research on volumetric data fitting by trivariate B-spline functions is not yet so widespread, despite its necessity.

Because our focus is on research on volumetric attribute data fitting, we first briefly review the research on volumetric geometry fitting and continue with a review of volumetric attribute data fitting. Martin et al. [23] proposed a method to fit a single trivariate B-spline on the basis of discrete volumetric harmonic functions to parameterize a volumetric model from input genus-0 triangle meshes. The method guarantees that the slices defining the B-spline do not overlap and only have degeneracies along the skeleton.

Given six boundary B-spline surfaces, Wang and Qian [36] presented a method to automatically construct a trivariate tensor-product B-spline volume via a gradient-based optimization approach. The internal control points are determined such that the resulting trivariate B-spline solid is valid in the sense that the minimal Jacobian of the solid is positive. The above two papers also provide many examples of a diverse set of applications in IGA. The readers should also refer to the references therein.

In the following, we review the research on volumetric attribute fitting. Park and Lee [28] conducted a pioneering work on a flow visualization model based on a trivariate NURBS representation. However, they used the same knot vectors for geometries and attributes, which restricts the representation of attributes.

On the other hand, Martin and Cohen [25] introduced a mathematical framework for representing the geometries and attributes independently by using different trivariate volumes. In other words, the order, the dimensions of the control mesh, and the knot vectors are independent, except that they share the same parametric domain. However, no examples were provided to demonstrate the effectiveness of the mathematical framework.

Liu et al. [19] presented a parametric and feature-based methodology for the design of solids with local composition control (LCC). The identified LCC features are those based on volume, transition, pattern, and (user-defined) surface features. The material composition functions include functions parameterized with respect to the distance or the distances to user-defined geometric features and functions that use Laplace's equation to smoothly blend various boundary conditions including the values and gradients of the material composition on the boundaries.

Yang and Qian [38] introduced a modeling method termed heterogeneous lofting, where heterogeneous profile surfaces are lofted to generate a heterogeneous volume. In their work, they used the same order and knot vectors for the geometry and attribute; accordingly, the attribute representation is limited. Moreover, the lofting technique is simple to formulate, but it is quite difficult to implement a robust and usable lofting capability and often requires a tedious and iterative process [31].

Kineri et al. [15] introduced an iterative geometric approximation method for fitting a point cloud in terms of B-spline surfaces by iteratively repositioning the B-spline's control points on the basis of simple geometrical rules. Lin et al. [17] developed the volumetric geometry approximation. Given a tetrahedral mesh model, Lin et al. [17] partitioned the model surface into six boundary triangular mesh patches that intersect at twelve boundary curves and eight corners. On the basis of this partition, they developed a discrete volume parameterization method and an iterative algorithm for fitting a tetrahedral mesh model with a B-spline solid. The iterative fitting algorithm starts with an initial B-spline base solid, and then it inserts the knots iteratively if the error norm does not become small enough in some regions.

We extend the method by [17] to the fitting of volumetric attribute data, which provides a fast and robust procedure for the automatic generation of trivariate B-spline attribute volumes.

3.2 Generation of accurate data for fitting

First, we compute the discrete offset surfaces from the object boundary toward the inside of the volume using the fast marching method (FMM) [34, 39] to bypass the local and global self-intersection problems [21]. The FMM generates offsets via wavefront propagation on a 3D grid with a resolution chosen according to the desired accuracy, as illustrated in Fig. 2a, b using a 2D model. In the FMM, the moving front is restricted to movement in the same direction by using only the upwind values.

Second, we determine the exact locations of the offset points with a constant distance d from the boundary, as the accuracy of the FMM is dependent on the voxel size. As shown in Fig. 2c∼e, we compute the closest point $\mathbf{R}^{(0)}$ on the boundary surface from the centroid of the offset voxel $\mathbf{Q}^{(0)}$ using Newton's method and adjust the length of the vector $\mathbf{Q}^{(1)} - \mathbf{R}^{(0)}$ so that it will become d. When $\mathbf{Q}^{(0)}$ is close to a self-intersection point/curve, the closest point $\mathbf{R}^{(1)}$ computed from $\mathbf{Q}^{(1)}$ may not coincide with $\mathbf{R}^{(0)}$. In such cases, we repeat these processes until the closest point does not move. If the point does not converge, we simply remove the point from approximation. Third, we assign the attribute data to the points with the constant d_i for i=1, …,h layers (see Fig. 2f). In this way, we can design an accurate distribution of the attribute data within the volume, which will be fit in the downstream process.

3.3 Iterative fitting method

The volumetric attribute data approximation algorithm takes a set of attribute data $\mathbf{a}_l$ associated with data points $\mathbf{x}_l$, $l = 0, \ldots, N$ as an input and generates a B-spline volume that approximates the attribute data. The base volume can start from a Bézier volume, i.e., a B-spline volume without internal knots; however, one can start from several internal knots placed at places with large changes, depending on the complexity of the attribute data. One might expect that starting with a relatively large number of initial internal knots yields a faster convergence, but this often leads to unnecessary degrees of freedom that may not contribute to the overall minimization of the error.

The error vector is defined as the difference vector between the attribute data and the corresponding ordinates evaluated at (u_l, v_l, w_l) for each data point as follows:

$$\mathbf{e}_l^{(\alpha)} = \mathbf{a}_l - \mathbf{A}^{(\alpha)}(u_l, v_l, w_l) \ , \tag{7}$$

where the superscript (α) denotes the αth iteration and

$$\mathbf{A}^{(\alpha)}(u, v, w) = \sum_{i,j,k=0}^{m_u,m_v,m_w} \mathbf{A}_{ijk}^{(\alpha)} N_{i,K}(u) N_{j,L}(v) N_{k,M}(w) \ . \tag{8}$$

We note here that attribute data fitting, unlike geometric surface fitting [15], does not require the computation of the closest point on the volume, leading to faster computation. At the parameter values $(u_l, v_l, w_l)(u_p \leq u_l < u_{p+1}, v_q \leq v_l < v_{q+1}, w_r \leq w_l < w_{r+1})$ corresponding to the input point $\mathbf{x}_l$, there are $K \times L \times M$ nonzero multiples

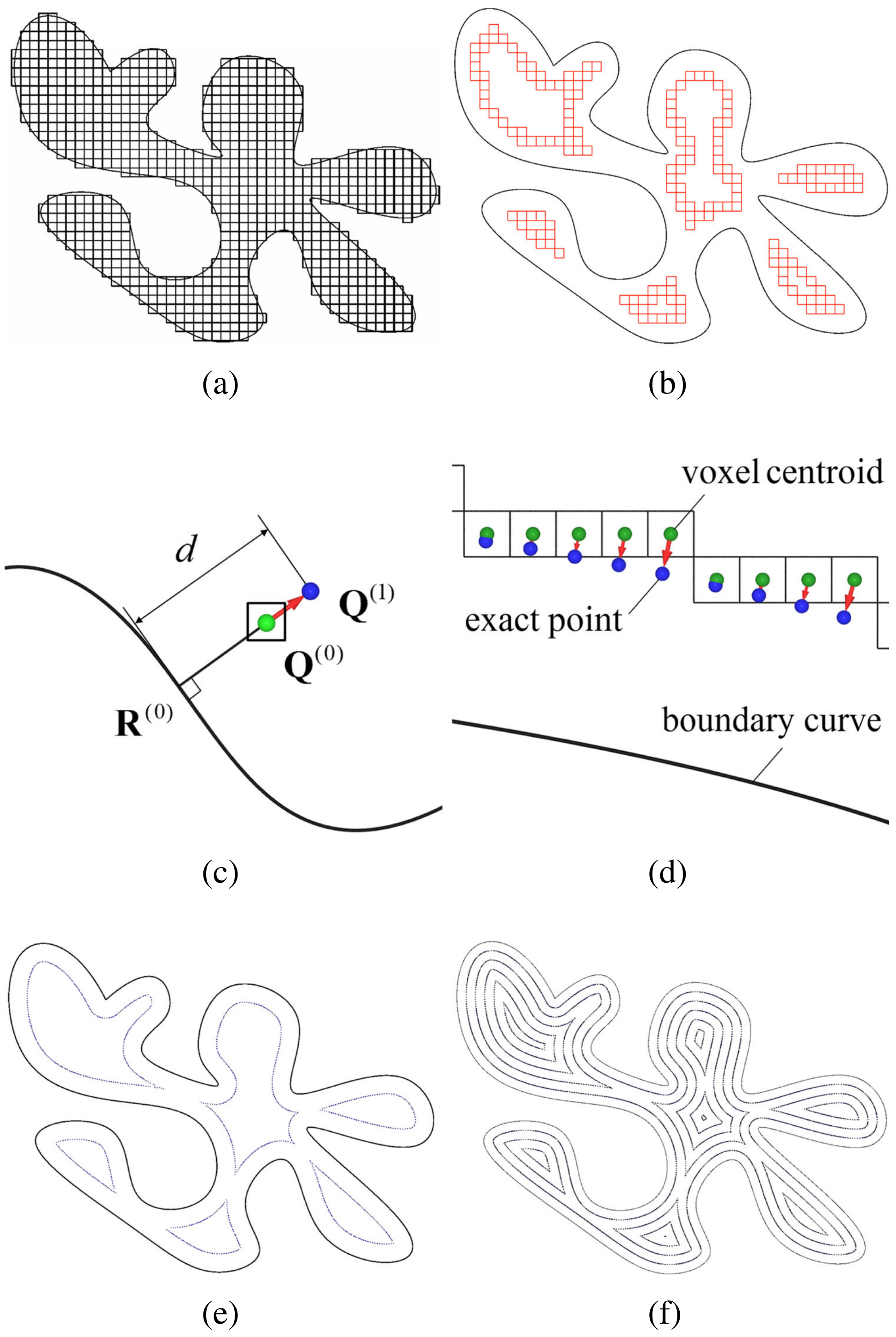

Fig. 2 Attribute data generation. **a** Two-dimensional voxel generation. **b** Two-dimensional voxels at a distance d from the boundary curve. **c** Relocation of the offset points. **d** Relocation of the offset points. **e** Corrected data points. **f** Final results for all distances

of the basis functions $N_{i,K}(u_l)N_{j,L}(v_l)\ N_{k,M}(w_l)$, $i = p - K + 1, \ldots, p$, $j = q - L + 1, \ldots, q$, $k = r - M + 1, \ldots, r$. As B-spline volumes are defined as a linear combination of B-spline ordinates and B-spline basis functions (see (8)), the multiple of the basis functions $N_{i,K}(u)N_{j,L}(v)N_{k,K}(w)$ is associated with the B-spline ordinates $\mathbf{A}_{ijk}^{(\alpha)}$. Accordingly, we can distribute the error vector $\mathbf{e}_l^{(\alpha)}$ to the B-spline ordinates $\mathbf{A}_{ijk}^{(\alpha)}$, $i = p - K + 1, \ldots, p$, $j = q - L + 1, \ldots, q$, $k = r - M + 1, \ldots, r$ with weights $N_{i,K}(u)N_{j,L}(v)N_{k,M}(w)$.

The overall algorithm for repositioning the B-spline ordinates for the αth iteration is described in Algorithm 1 in

Appendix 1. The $(\alpha + 1)$th B-spline volumetric attribute model is given by

$$\mathbf{A}^{(\alpha+1)}(u, v, w) = \sum_{i,j,k=0}^{m_u,m_v,m_w} (\mathbf{A}_{ijk}^{(\alpha)} + \Delta_{ijk}^{(\alpha)}) N_{i,K}(u) N_{j,L}(v) N_{k,M}(w) \; , \quad (9)$$

where $\Delta_{ijk}^{(\alpha)}$ are the repositioning vectors in the αth iteration given by

$$\Delta_{ijk}^{(\alpha)} = \sum_{l \in I_{ijk}} \omega_l^{ijk} \mathbf{e}_l^{(\alpha)} \; , \quad (10)$$

where I_{ijk} denotes the set of ordinates $\mathbf{a}_l$ that contribute to the repositioning vector $\Delta_{ijk}^{(\alpha)}$, and the weight ω_l^{ijk} is defined by

$$\omega_l^{ijk} = \frac{N_{i,K}(u_l) N_{j,L}(v_l) N_{k,M}(w_l)}{W_{den}[i][j][k]} \; , \quad (11)$$

which is similar to [15, 17]. The denominator of Eq. 11, $W_{den}[i][j][k]$, is defined in Algorithm 1.

We repeat steps 1 to 17 of Algorithm 1 until the average and the maximum of the norm of the error vector $\mathbf{e}_l^{(\alpha)}$, $l = 0, ..., N$ become smaller than the prescribed average and maximum tolerances $\varepsilon_{avg}\%$ and $\varepsilon_{max}\%$, respectively, with respect to the difference between the maximum and the minimum values of the input attribute data. Note that, in Algorithm 1, the superscripts (α) of the parameter values (u_l, v_l, w_l) are omitted for simplicity. If the error norm does not become small enough in some regions, a knot is inserted in these locations in all of the u, v, and w directions. The increase in the number of B-spline ordinates provides an increase in the degrees of freedom of the volume, resulting in the reduction of local errors. During the error norm check, row l_u, column l_v, and layer l_w of the spans that have the greatest sum of errors are determined. A knot is then inserted in the middle of the span (l_u, l_v, l_w) in all of the u, v, and w directions. In most of our examples, a knot is inserted in all $\mathbf{u}$, $\mathbf{v}$, and $\mathbf{w}$ after every $5 \sim 10$ iterations. To limit the total number of internal knots, one can define an upper limit for the number of knots in each direction. The proof of convergence is given in Appendix 2. Figure 3 shows the geometry model, the color-coded input data based on the distance function, and the fitted result of the tooth model, respectively.

Table 1 Approximation of the volumetric attribute data of the tooth model by the standard and iterative fitting methods with $(\varepsilon_{avg}, \varepsilon_{max}) = (0.3\ \%, 3.0\ \%)$

Method	# of ctrl pts	Iteration	Time (s)
Standard	19 × 19 × 19	13	37.536
Iterative fitting	19 × 19 × 19	241	13.809

3.4 Comparison with the standard fitting method

We compared our iterative fitting algorithm with the standard fitting method [31] with an example that was computed on a PC with an Intel Core i7-3770 (3.40 GHz) processor and 8.00 GB of RAM. The linear system of the standard fitting methods is solved by CSparse [5].

We first generated the attribute volume data on the basis of the distance function from the boundary of the tooth model as a specific function for trivariate B-spline fitting, as described in Section 3.2. We fit the data within the tooth model with 170,366 points by an order (4, 4, 4) trivariate B-spline function starting with the knot vectors $\mathbf{u} = \mathbf{v} = \mathbf{w} = (0,0,0,0,0.25,0.5,0.75,1,1,1,1)$ by the standard fitting algorithm as well as by the iterative fitting method.

Two pairs of tolerances, $(\varepsilon_{avg}, \varepsilon_{max}) = (0.3\ \%, 3.0\ \%)$ and $(0.2\ \%, 2.0\ \%)$, respectively, were used to terminate the computation. The computational results of the volume approximation by the standard fitting method and the iterative fitting method are shown in Tables 1 and 2, respectively. We can see that the number of control points increased by 2.87 times more as the prescribed tolerances became slightly tighter. In

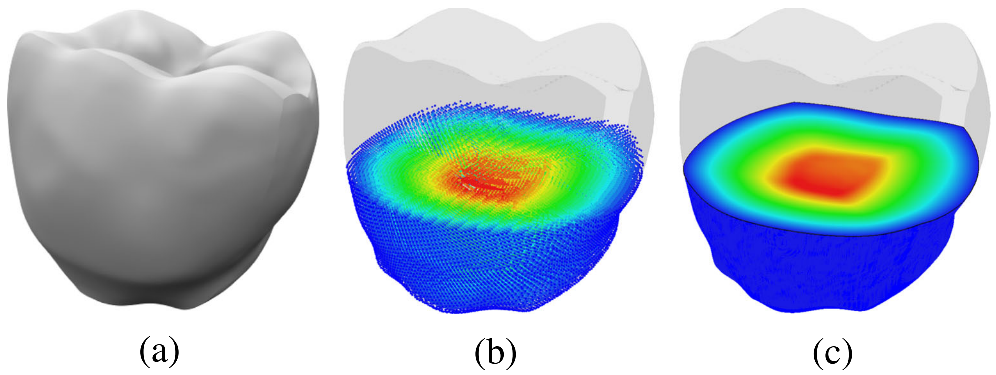

Fig. 3 Fitting of the volumetric attribute data of a tooth model. **a** Geometry model. **b** Color-coded input data based on the distance function. **c** Fitted result

Table 2 Approximation of the volumetric attribute data of the tooth model by the standard and iterative fitting methods with $(\varepsilon_{avg}, \varepsilon_{max}) = (0.2\ \%, 2.0\ \%)$

Method	# of ctrl pts	Iteration	Time (s)
Standard	$27 \times 27 \times 27$	21	237.676
Iterative fitting	$27 \times 27 \times 27$	408	22.645

the following, we explain why it took so many control points to fit the data. Figure 4 visualizes the hypersurface of the fitted attribute data of the tooth model near the maximum error point (yellow point) at $(u, v, w) = (0.644, 0.585, 0.588)$ with $\varepsilon_{max} = 1.995\ \%$) for the tolerance $(0.2\ \%, 2.0\ \%)$ through the isoparametric surface obtained by fixing the parameter $v = 0.585$. Furthermore, we employed normal mapping, where the x, y, z components of the surface normal correspond to the RGB components in Fig. 4 to emphasize the ridges on the isoparametric surface. The maximum error point was found on the ridge curve where B-spline functions require many control points in representing ridges. Therefore, there is a tradeoff between accuracy and storage reduction when sharp features exist in the data.

Figure 5 shows the computational time versus the maximum error for the tooth model. It can be clearly seen that the iterative fitting quickly reached a coarse fit, and we can progressively obtain a finer fit by performing more iterations. In general, our method is faster than the standard method, which can be explained by the following reasons. Whenever knots are inserted, the standard method requires the solution of a linear system, which takes between one and two orders of magnitude more than the computation of the repositioning vectors. Although the number of iterations for the standard method is an order of magnitude smaller than that for our method, in total, the computational time becomes slower.

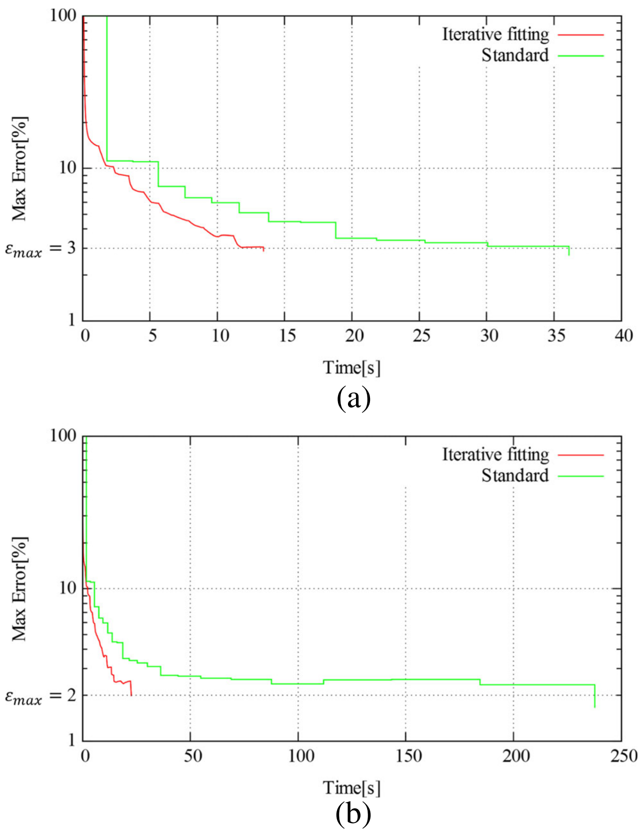

Fig. 5 Computational time versus maximum error for the iterative fitting and standard fitting methods. **a** $\varepsilon_{\max} = 3.0\ \%$. **b** $\varepsilon_{\max} = 2.0\ \%$

4 Adaptive direct slicing

4.1 Related work

To overcome the defects of STL data, Jamieson and Hacker [12] introduced a method to directly slice the input CAD model by consecutively calling the function of Parasolid,

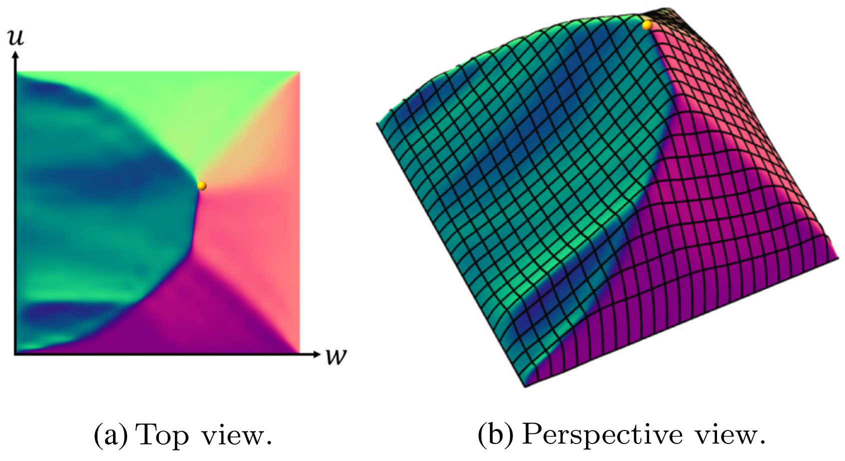

Fig. 4 Visualization of the fitted attribute data of the tooth model near the maximum error point (*yellow point*). The hypersurface is visualized through the isoparametric surface by fixing the parameter v. Normal mapping is used to emphasize the ridges on the surface

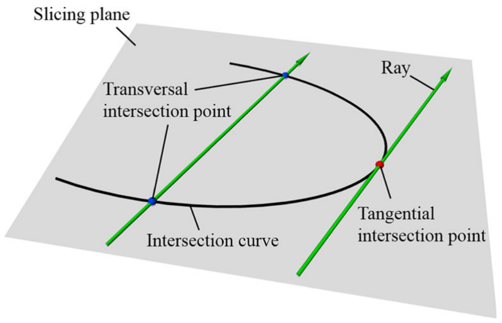

Fig. 6 Ray-object intersection

which is the solid modeling kernel of Unigraphics, for each slice. Ma et al. [20] employed a NURBS-based adaptive slicing strategy so that a variable layer thickness based on the local surface geometry is generated in a Unigraphics software environment. Starly [35] employed a direct slicing method to scaffolds of complex shape used in tissue engineering applications where direct slicing is more advantageous than the traditional indirect slicing method. The direct slice is obtained by casting parallel rays, which lie in the slicing plane, into the object. The intersection points are obtained using the bisection iteration routine, which is a local solution method and is a relatively slow root-finding method.

Martin et al. [24] presented an isosurface visualization technique for attribute data defined on (un)structured (curvi)linear hexahedral grids by combining subdivision and numerical root finding.

As in Starly et al. [35], we employed a ray-tracing method to obtain the tool path. Once the slicing plane is determined, a ray on the plane is cast one by one into the object to find the transversal as well as the tangential intersection points with the outer geometry and the inner attribute data as illustrated in Fig. 6. The interval between rays are adaptively determined to reflect the feature points of the attribute data robustly, as shown in Fig. 9.

4.2 AM techniques for FGM and MM objects

Up to now, we have focused our discussions on the construction of FGM objects by the iterative fitting method; however, we have not yet discussed any AM techniques for this research. Recently, Gao et al. [9] provided an excellent review of AM techniques, and readers should refer to the references therein.

AM techniques such as stereolithography and inkjet binder printing are able to print FGM objects continuously by adjusting the laser power [33] and density of the binder [19], respectively. On the other hand, AM techniques such as fused deposition modeling (FDM) cannot print FGM objects continuously. In such cases, the FGM objects are discretized into a set of slowly varying n_M constant materials with a clear boundary between them, which we call discrete FGM (DFGM) objects as illustrated in Fig. 7. DFGM objects can be printed by FDM machines equipped with a multitool station of a CNC machine [27]. Figure 8 illustrates the multi-tool station of ejecting nozzles where each nozzle ejects different material. Printing of MM objects is similar to printing of DFGM objects except that the material properties can be very different across the boundaries.

Our algorithm is independent of AM techniques, and we hope that the readers of this paper can customize our algorithm to their own AM techniques. In this paper, we apply our algorithm to FGM, DFGM, and MM objects.

4.3 Ray object intersection

In this section and in the following Section 4.4, we examine how to determine the tool path on the slicing plane. Although there exists a more sophisticated layer-based path planning design, which uses a bi-layer pattern of radial and spiral layers consecutively to generate functionally gradient porosity [14], we employ zigzag tool path pattern for simplicity as it is not the focus of this paper. As illustrated in Fig. 9b, rays must pass through the feature points of the outer geometry of the object as well as through those of the

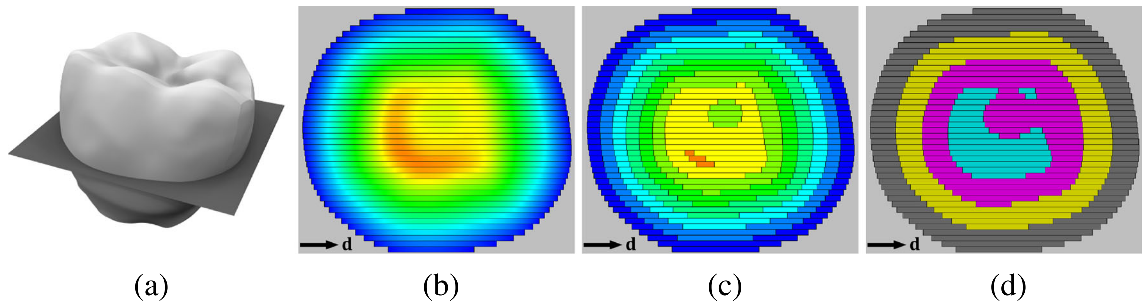

Fig. 7 Direct slicing of the tooth model. **a** Slicing plane. **b** FGM object-based tool path. **c** DFGM object-based tool path. **d** MM object-based tool path

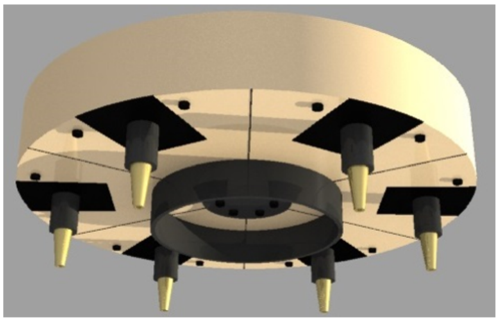

Fig. 8 Illustration of multi-tool station of ejecting nozzles

attribute data of the DFGM and MM objects tangentially to reflect all the features robustly onto the printed model.

To determine the tool path for FGM objects, we first compute the ray-object tangential intersection points and then adjust the horizontal path intervals so that the feature points are reflected on the tool path. Once the path intervals are determined, the ray-object transversal intersection points are computed to determine the entry and exit points. The attribute data values can be easily evaluated by linearly interpolating the coordinate values of the entry and exit points, and then finding the corresponding parameter values using Eq. 5, and substituting them into Eq. 6.

In addition to the above computations, ray-(attribute data) tangential and transversal intersection points are computed for DFGM and MM objects, which will be studied in Section 4.4.

4.3.1 Tangential intersection points

Without loss of generality, we assume that the orientation of the object is already determined and that the slicing direction is perpendicular to the z-axis, and, hence, the xy-plane is parallel to the slicing plane. Accordingly, the equation of the slicing plane is z = constant. We further assume that a variable layer thickness of the input model is already determined. Since the ray-object tangential intersection points lie on the slicing plane, we have

$$\mathbf{N}_1 \cdot (\mathbf{S}_\xi(u,v) - \mathbf{O}) = 0, \tag{12}$$

where $\mathbf{N}_1 = (0,0,1)$, $\mathbf{O}$ is an arbitrary point on the slicing plane and $\mathbf{S}_\xi(u,v)$, $(\xi = 1, \cdots, 6)$ is one of the six boundary B-spline surfaces of the B-spline volume defined by

$$\mathbf{S}_\xi(u,v) = \sum_{i,j=0}^{n_u,n_v} \mathbf{P}_{\xi,ij} N_{i,\bar{K}}(u) N_{j,\bar{L}}(v) \ . \tag{13}$$

Another equation comes from the orthogonality of the ray direction $\mathbf{d} = (d^x, d^y, 0)$ and the surface normal of the boundary surface $\mathbf{S}_\xi(u,v)$ yielding

$$\mathbf{S}_{\xi,u}(u,v) \times \mathbf{S}_{\xi,v}(u,v) \cdot \mathbf{d} = 0 \ , \tag{14}$$

where $\times$ denotes a vector cross product and $\cdot$ denotes a vector dot product. Note that the z-component of the direction vector $\mathbf{d}$ is zero, as the ray is parallel to the xy-plane. We decompose B-spline surfaces into (Bézier) polynomial segments by knot refinement [31]. Then, for each Bézier segment, we formulate (14) using arithmetic operations between polynomials in Bernstein form [8], which has better numerical stability under perturbation of its coefficients than the power basis as follows:

$$\sum_{i,j=0}^{\bar{K}-1,\bar{L}-1} \mathbf{N}_1 \cdot (\mathbf{P}_{\xi,ij,l} - \mathbf{O}_R) B_{i,\bar{K}}(u) B_{j,\bar{L}}(v) = 0 \ , \tag{15}$$

$$\left(\sum_{i,j=0}^{\bar{K}-2,\bar{L}-1} \mathbf{Q}_{\xi,ij,l} B_{i,\bar{K}-1}(u) B_{j,\bar{L}}(v) \times \sum_{i,j=0}^{\bar{K}-1,\bar{L}-2} \mathbf{R}_{\xi,ij,l} B_{i,\bar{K}}(u) B_{j,\bar{L}-1}(v) \right) \cdot \mathbf{d} = 0 \ , \tag{16}$$

where $B_{i,\bar{K}}(u)$ and $B_{j,\bar{L}}(v)$ are the Bernstein polynomials of degree $\bar{K}-1$ and $\bar{L}-1$, $\mathbf{Q}_{\xi,ij,l} = (\bar{K}-1)(\mathbf{P}_{\xi,(i+1)j,l} -$

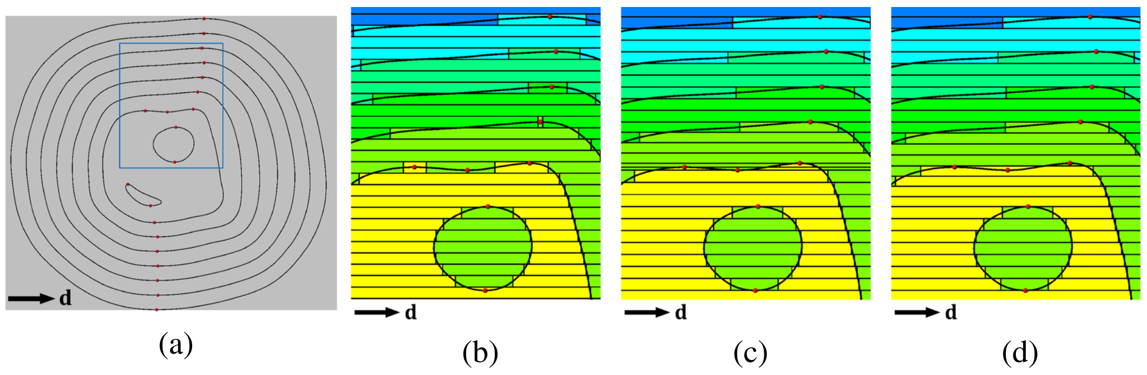

Fig. 9 Adaptive direct slicing of the tooth model. **a** Tangential intersection points located on the slicing plane. **b** Constant interval tool paths. **c** Adaptive tool paths where some of the path intervals are smaller than $\Delta_{\min}$. **d** Path intervals smaller than $\Delta_{\min}$ are iteratively merged so that they are larger than $\Delta_{\min}$

$\mathbf{P}_{\xi,ij,l}$), and $\mathbf{R}_{\xi,ij,l} = (\bar{L}-1)(\mathbf{P}_{\xi,i(j+1),l} - \mathbf{P}_{\xi,ij,l})$. The coupled polynomial Eqs. 15 and 16 are solved by a polynomial solver called the projected polyhedron (PP) method [29]. The PP algorithm is a global solution method designed to compute all roots in some area of interest based on subdivision.

4.3.2 Transversal intersection points

The equation of the ray can be expressed as

$$\mathbf{r}(t) = \mathbf{O}_R + t\mathbf{d}\ , \tag{17}$$

where $\mathbf{O}_R = (O_R^x, O_R^y, O_R^z)$ is a starting point and t is the parameter of the ray. The ray can also be expressed by the intersection line of the following two planes [13]:

$$\mathbf{N}_1 \cdot (\mathbf{P} - \mathbf{O}_R) = 0\ , \tag{18}$$
$$\mathbf{N}_2 \cdot (\mathbf{P} - \mathbf{O}_R) = 0\ , \tag{19}$$

where $\mathbf{N}_2 = \mathbf{N}_1 \times \mathbf{d}$ are the normal vectors of the slicing plane and of the plane perpendicular to the slicing plane, respectively, and $\mathbf{P}$ is a point on the intersection line. The intersection between the ray and the boundary surfaces $\mathbf{S}_\xi(u, v)$ can be formulated by replacing $\mathbf{P}$ in Eqs. 18 and 19 by $\mathbf{S}_\xi(u, v)$ yielding

$$\mathbf{N}_1 \cdot (\mathbf{S}_\xi(u, v) - \mathbf{O}_R) = 0\ , \tag{20}$$
$$\mathbf{N}_2 \cdot (\mathbf{S}_\xi(u, v) - \mathbf{O}_R) = 0\ . \tag{21}$$

We decompose B-spline surfaces into (Bézier) polynomial segments by knot refinement [31]. Then, for each Bézier segment, we formulate (20) and (21) using arithmetic operations between polynomials in Bernstein form [8]:

$$\sum_{i,j=0}^{\bar{K}-1,\bar{L}-1} \mathbf{N}_1 \cdot (\mathbf{P}_{\xi,ij,l} - \mathbf{O}_R) B_{i,\bar{K}}(u) B_{j,\bar{L}}(v) = 0\ , \tag{22}$$

$$\sum_{i,j=0}^{\bar{K}-1,\bar{L}-1} \mathbf{N}_2 \cdot (\mathbf{P}_{\xi,ij,l} - \mathbf{O}_R) B_{i,\bar{K}}(u) B_{j,\bar{L}}(v) = 0\ , \tag{23}$$

where $l = 0, ..., (n_u - \bar{K} + 1)(n_v - \bar{L} + 1)$ are the indexes for the Bézier segments. A set of coupled nonlinear polynomial equations can be effectively solved by the PP method [29]. The resulting transversal intersection points are categorized into entry points and exit points.

4.4 Ray attribute data intersection

4.4.1 Tangential intersection points

To generate an adaptive tool path for DFGM and MM objects, we need to compute the ray-attribute-data tangential intersection points. Since the ray-attribute-data tangential intersection points lie on the slicing plane, we thus have

$$\mathbf{N}_1 \cdot (\mathbf{V}(u, v, w) - \mathbf{O}) = 0\ , \tag{24}$$

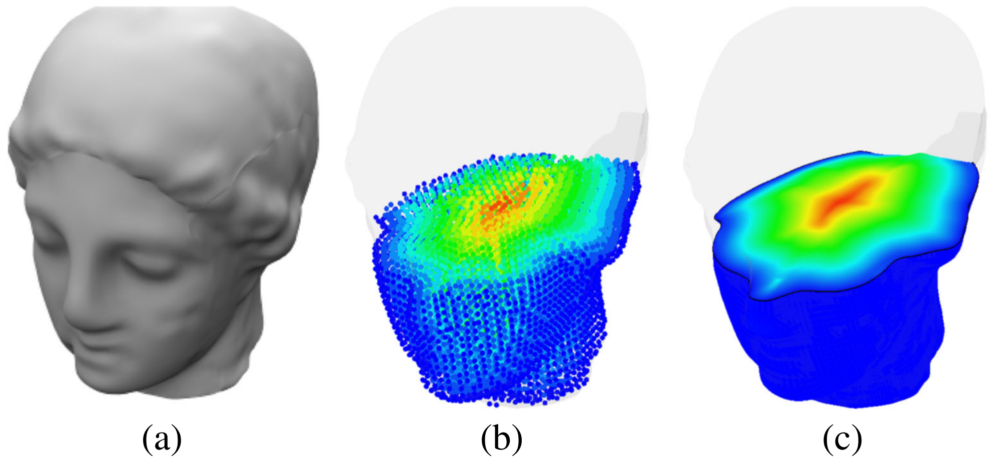

Fig. 10 Attribute volume fitting of the Venus head model. **a** Original model. **b** Color-coded input data based on the distance function. **c** Fitted result

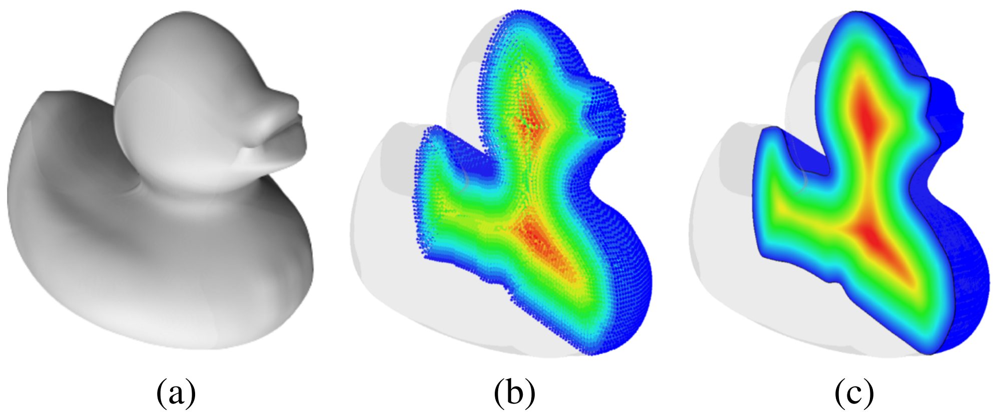
Fig. 11 Attribute volume fitting of the duck model. **a** Original model. **b** Color-coded input data based on the distance function. **c** Fitted result

and the ηth ($1\leq \eta \leq n$) attribute data having the value of T_ζ ($1\leq \zeta \leq n_M$) can be expressed as

$$F_{\eta,\zeta}(u, v, w) = A_\eta(u, v, w) - T_\zeta = 0 \,. \tag{25}$$

The third equation comes from the orthogonality of the normal vector of the isosurface $F_{\eta,\zeta}(u, v, w) = 0$ and $\mathbf{d}$ yielding

$$\nabla F_{\eta,\zeta}(u, v, w) \cdot \mathbf{d} = 0 \,, \tag{26}$$

where

$$\nabla F_{\eta,\zeta}(u, v, w) = (A_{\eta,x}, A_{\eta,y}, A_{\eta,z})^{\mathrm{T}} \,. \tag{27}$$

The partial derivatives of A_η with respect to x, y, and z can be evaluated as follows:

$$\begin{pmatrix} A_{\eta,x} \\ A_{\eta,y} \\ A_{\eta,z} \end{pmatrix} = \begin{pmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{pmatrix} \begin{pmatrix} A_{\eta,u} \\ A_{\eta,v} \\ A_{\eta,w} \end{pmatrix} \,, \tag{28}$$

where $A_{\eta,u}$, $A_{\eta,v}$, and $A_{\eta,w}$ are the partial derivatives of A_η with respect to u, v, and w, respectively, and u_x, u_y, u_z, v_x, v_y, v_z, w_x, w_y, and w_z are the partial derivatives of u, v, w with respect to x, y, z, respectively, which can be obtained using the inverse function theorem [7]:

$$\begin{pmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{pmatrix} = \begin{pmatrix} x_u & y_u & z_u \\ x_v & y_v & z_v \\ x_w & y_w & z_w \end{pmatrix}^{-1} \,. \tag{29}$$

The partial derivatives of x, y, and z with respect to u, v, and w can be obtained easily from Eq. 4. By decomposing the simultaneous Eqs. 24, 25, and 26 into polynomial segments, the system can be solved by the PP method [29].

To generate a feature-sensitive tool path, we adjust the path interval such that the feature points, which are the tangential intersection points with respect to the ray, are reflected in the tool path.

There are cases where the adjacent rays are very close to each other owing to the distribution of the feature points yielding narrow intervals. In such cases, we can iteratively merge the two intervals into a single interval at the middle of the two intervals until each interval is wider than the prescribed tolerance $\Delta_{\min}$, as illustrated in Fig. 9. We also define the maximum allowable path interval $\Delta_{\max}$ so that the interval does not become too wide.

4.4.2 Transversal intersection points

We discretize the ηth attribute data A_η with the values T_ζ, ζ=1,..., n_M. The ray equations similar to Eqs. 20 and 21 can be written as

$$\mathbf{N}_1 \cdot (\mathbf{V}(u, v, w) - \mathbf{O}_R) = 0 \,, \tag{30}$$
$$\mathbf{N}_2 \cdot (\mathbf{V}(u, v, w) - \mathbf{O}_R) = 0 \,, \tag{31}$$

and the ηth attribute data A_η having the value T_ζ can be expressed as

$$A_\eta(u, v, w) - T_\zeta = 0 \,. \tag{32}$$

In general, for FGM objects, the geometry and attribute data have different knot vectors. However, for DFGM and MM objects, simultaneous Eqs. 30, 31, and 32 must be decomposed into (Bézier) polynomial segments to be solved by polynomial solvers [29], and, hence, the geometry $\mathbf{V}(u, v, w)$ and the attribute data $A_\eta(u, v, w)$ must

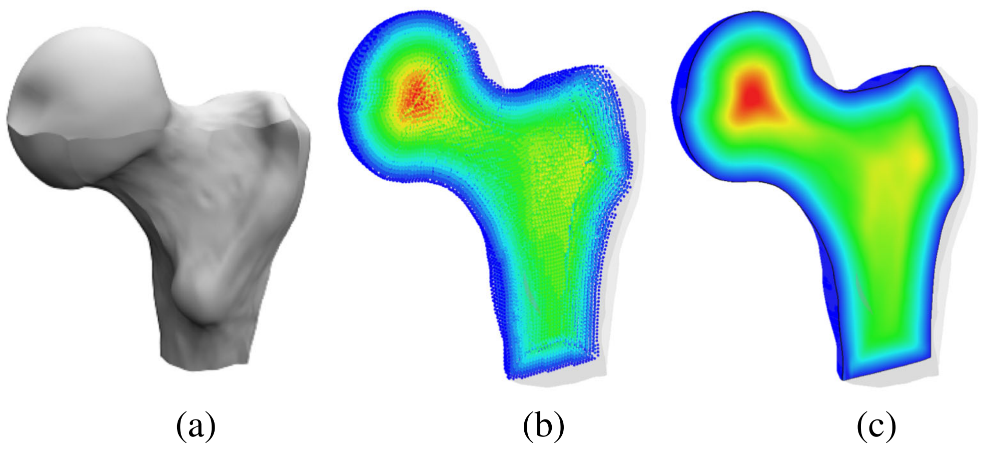

Fig. 12 Attribute volume fitting of the ball joint model. **a** Original model. **b** Color-coded input data based on the distance function. **c** Fitted result

be refined before the computation to have the same knot vectors. Then, the B-spline volumes are decomposed into (Bézier) polynomial segments by knot refinement [31] similar to Eqs. 22 and 23 as

$$\sum_{i,j,k=0}^{\bar{K}-1,\bar{L}-1,\bar{M}-1} \mathbf{N}_1 \cdot (\mathbf{P}_{ijk,l} - \mathbf{O}_R) B_{i,\bar{K}}(u) B_{j,\bar{L}}(v) B_{k,\bar{M}}(w) = 0\ , \tag{33}$$

$$\sum_{i,j,k=0}^{\bar{K}-1,\bar{L}-1,\bar{M}-1} \mathbf{N}_2 \cdot (\mathbf{P}_{ijk,l} - \mathbf{O}_R) B_{i,\bar{K}}(u) B_{j,\bar{L}}(v) B_{k,\bar{M}}(w) = 0\ , \tag{34}$$

$$\sum_{i,j,k=0}^{K-1,L-1,M-1} (A_{\eta,ijk,l} - T_\zeta) B_{i,K}(u) B_{j,L}(v) B_{k,M}(w) = 0\ , \tag{35}$$

where $l = 0, ..., (n_u - \bar{K} + 1)(n_v - \bar{L} + 1)(n_w - \bar{M} + 1)$. The coupled polynomial Eqs. 33, 34 and 35 are effectively solved by the PP method.

4.5 Bucketing technique

Since it is inefficient to check all the possible ray-Bézier volume intersection problems, we introduce a bucketing technique [3, 4]. If we assume that the ray is parallel to the x-axis, then the axis-aligned bounding box of the object can be used for constructing buckets by uniformly dividing it into $n_x \times n_y \times n_z$ voxels, where we used a size of $10\times10\times10$. Then, all the Bézier volumes that are completely or partially inside each bucket are checked for the ray-Bézier volume intersection problems. Once the intersecting Bézier volume is located, we apply the polynomial solver to find the intersection points.

5 Examples

In this section, we demonstrate the effectiveness of our algorithms by applying them to the complex models used in Lin et al. [17], namely Venus head, duck, and ball joint models shown in Figs. 10a, 11a, and 12a, respectively. All the computations were performed on a PC with an Intel Core i7-3770 (3.40 GHz) processor and 8.00 GB of RAM.

5.1 Generation of the volumetric attribute data

We first generated the volumetric attribute data based on the distance function from the boundary of the three models as a specific function for trivariate B-spline fitting, which is described in Section 3.2. Figures 10b, 11b, and 12b show the color-coded data based on the distance from the boundary. We fit the data within the models by an order (4, 4, 4) trivariate B-spline function starting with the knot vectors **u** = **v** = **w** = (0,0,0,0,0.25,0.5,0.75,1,1,1,1) by iterative fitting.

The computational results of the fitting of the volumetric attribute data are shown in Table 3, and the fitted attribute data are shown in Figs. 10c, 11c and 12c. Computational results of the tooth model are given in Section 3.4. The computations were terminated when the average and

Table 3 Approximation of the volumetric attribute data of the Venus head, duck, and ball joint models by the iterative fitting method

Model	# of data pts	# of ctrl pts	Iteration	Time (s)
Venus head	51,953	34 × 31 × 30	1084	18.488
Duck	145,570	28 × 28 × 28	425	20.233
Ball joint	100,149	27 × 22 × 27	410	13.447

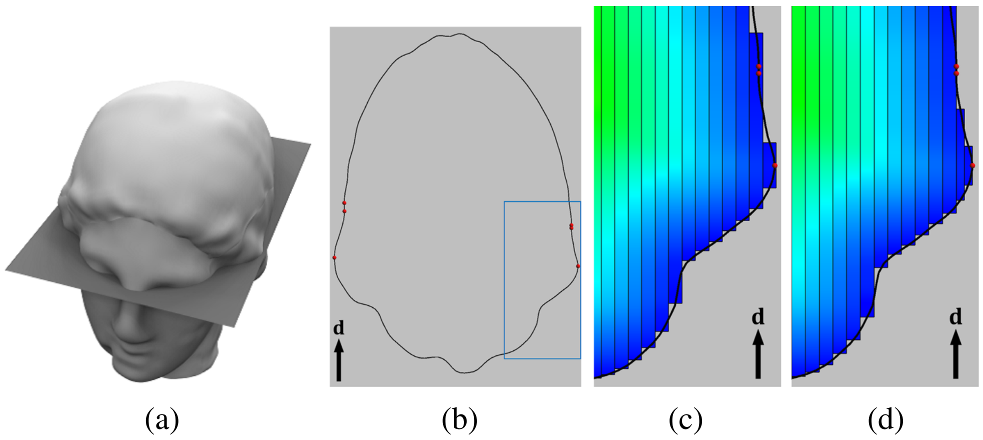

Fig. 13 FGM object-based tool path for the Venus head model. Adaptive direct slicing of the Venus head model. **a** Slicing plane. **b** Tangential intersection points located on the slicing plane. **c** Constant interval tool path. **d** Adaptive tool path

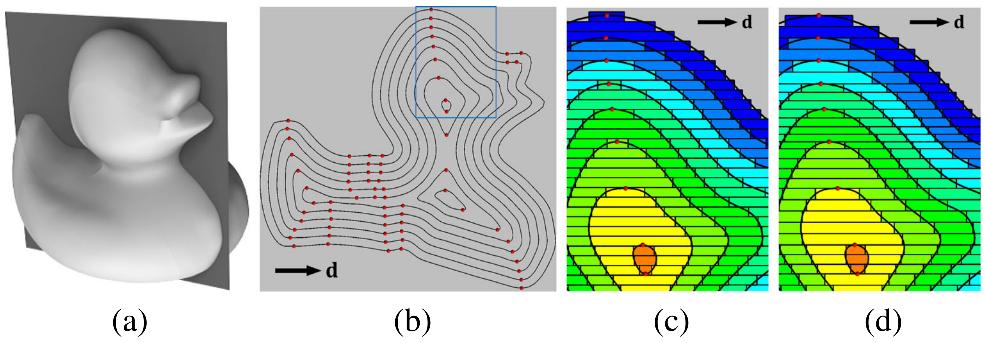

Fig. 14 DFGM object-based tool path for the duck model: Adaptive direct slicing of the duck model. **a** Slicing plane. **b** Tangential intersection points located on the slicing plane. **c** Constant interval tool path. **d** Adaptive tool path

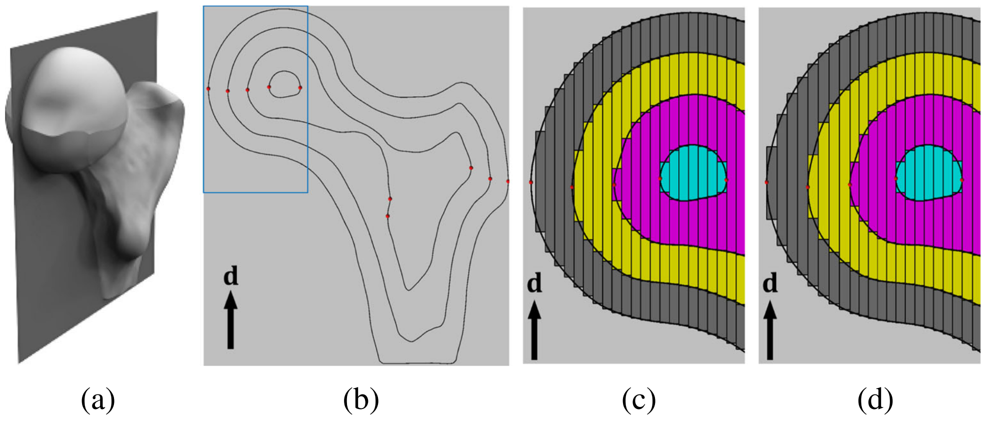

Fig. 15 MM object-based tool path for the ball joint model: Adaptive direct slicing of the ball joint model. **a** Slicing plane. **b** Tangential intersection points located on the slicing plane. **c** Constant interval tool path. **d** Adaptive tool path

Table 4 Computational time for finding tangential and transverse intersection points for determining the horizontal ray path intervals of the models in Figs. 13, 14 and 15

Model	Time (s) tangential intersection points	Time (s) transversal intersection points	# of paths
Venus head	0.062	0.016	44
Duck	16.81	55.59	72
Ball joint	7.733	9.485	63

maximum errors became smaller than 0.3 and 3.0 % with respect to the difference between the maximum and the minimum values of the input data for the models. The accuracy between the fitted results and the original model is discussed in Section 3.4. The same explanation applies to the abovementioned three models.

5.2 Adaptive direct slicing

We applied the adaptive direct slicing method to the three B-spline volumetric geometry models filled with the attribute data, namely the Venus head, duck, and ball joint models. The applications of an FGM object-based tool path for the Venus head model, a DFGM object-based tool path for the duck model, and an MM object-based tool path for the ball joint model are studied in this section. We assumed that the orientation of the object was already determined and that the slicing direction was parallel to the xy-plane as shown in Figs. 13a, 14a and 15a. Without loss of generality, we further assumed that the ray direction was parallel to the x-axis. Next, we determined the horizontal ray path intervals such that the feature points were reflected in the path generation, as shown Fig. 13b, 14b and 15b. In this research, we set $\Delta_{\min}$ and $\Delta_{\max}$ to 1/200 and 1/100 of the diagonal of the bounding box of each model.

Figures 13c, 14c and 15c show examples of the constant interval tool path, whereas Figs. 13d, 14d and 15d show the corresponding adaptive tool path. We can clearly see that the adaptive tool path captured all the tangential intersection points; however, there were cases wherein some of the path intervals were smaller than the prescribed tolerance $\Delta_{\min}$. These tool paths were iteratively merged so that they were larger than $\Delta_{\min}$.

As shown in Table 4, the computational speed for generating a tool path for a single layer is slow owing to the nonlinear computation; however, it takes less storage space and does not require to run check and repair routines compared with the STL-based processing. The computational time for the Venus head model was much faster than those of the other two models, as the computation for the ray-attribute-data intersection was not necessary in generating a tool path for continuous FGM objects.

6 Conclusions

In this paper, we introduced a framework for modeling and adaptive direct slicing of heterogeneous objects in terms of trivariate B-spline functions. The B-spline volume and the associated attribute data are directly sliced without converting them to STL format, keeping the geometric and topological robustness of the original data. Furthermore, adaptive slicing is introduced so that the zigzag tool path passes through all the tangential intersection points of the heterogeneous objects in the scan direction so that all the feature points are represented in the fabricated model.

Although our adaptive direct slicing algorithm is already highly effective, it still has a number of limitations that require further research:

- The PP method becomes extremely slow when the ray overlaps with the boundary curve.
- The B-spline volumetric attribute data fitting requires many control points when the model has sharp features.

Our algorithm is independent of AM techniques, and we hope that the readers of this paper can customize our algorithm to their own AM techniques. There are several possibilities for the extension of our algorithm, a few of which are listed as follows:

- We want to develop machine-specific interpreters for translating our tool path into NC codes and printing the model that is sliced using our algorithm.
- It is expected that AM machines for heterogeneous objects may suffer from many stop-and-start motions. We would like to develop the path planning method for heterogeneous objects so that the stop-and-start motions can be minimized.

Acknowledgments This work is supported by the Cabinet Office, Government of Japan, Cross-Ministerial Strategic Innovation Promotion Program P14030. The tooth, duck, Venus head, and ball joint models are courtesy of Hongwei Lin. We would like to thank Taketoshi Suzuki for his assistance.

Appendix 1

Algorithm 1 Repositioning of the B-spline ordinates.

```
1:  for all $l$ such that $0 \le l \le N$ do
2:    for $i = p$-$K$+1 to $p$ do
3:      for $j = q$-$L$+1 to $q$ do
4:        for $k = r$-$M$+1 to $r$ do
5:          $\mathbf{W}_{num}[i][j][k] \leftarrow \mathbf{W}_{num}[i][j][k] + N_{i,K}(u_l)N_{j,L}(v_l)N_{k,M}(w_l)(\mathbf{a}_l - \mathbf{A}^{(\alpha)}(u_l, v_l, w_l))$
6:          $W_{den}[i][j][k] \leftarrow W_{den}[i][j][k] + N_{i,K}(u_l)N_{j,L}(v_l)N_{k,M}(w_l)$
7:        end for
8:      end for
9:    end for
10: end for
11: for all $i$ such that $0 \le i \le m_u$ do
12:   for all $j$ such that $0 \le j \le m_v$ do
13:     for all $k$ such that $0 \le k \le m_w$ do
14:       $\mathbf{A}_{ijk}^{(\alpha+1)} \leftarrow \mathbf{A}_{ijk}^{(\alpha)} + \frac{\mathbf{W}_{num}[i][j][k]}{W_{den}[i][j][k]}$
15:     end for
16:   end for
17: end for
```

Appendix 2

Proof of convergence

Let us denote the weight for the error vector $\mathbf{e}_l$ as

$$\omega_l^{ijk} = \frac{N_{i,K}(u_l)N_{j,L}(v_l)N_{k,M}(w_l)}{W_{den}[i][j][k]}, \tag{36}$$

where $W_{den}[i][j][k]$ is defined in Algorithm 1. The $(\alpha+1)$th B-spline volume is defined as

$$\mathbf{A}^{(\alpha+1)}(u,v,w) = \sum_{i,j,k=0}^{m_u,m_v,m_w} (\mathbf{A}_{ijk}^{(\alpha)} + \Delta_{ijk}^{(\alpha)}) N_{i,K}(u)N_{j,L}(v)N_{k,M}(w), \tag{37}$$

where $\mathbf{A}_{ijk}^{(\alpha)}$ are the B-spline ordinates in the αth iteration, and $\Delta_{ijk}^{(\alpha)}$ are the repositioning vectors for the corresponding B-spline ordinates.

Therefore, the repositioning vector for the B-spline ordinate with index (i, j, k) is

$$\begin{aligned}
\Delta_{ijk}^{(\alpha+1)} &= \sum_{l\in I_{ijk}} \omega_l^{ijk} \mathbf{e}_l^{(\alpha+1)} \\
&= \sum_{l\in I_{ijk}} \omega_l^{ijk} \left(\mathbf{a}_l - \mathbf{A}^{(\alpha+1)}(u_l, v_l, w_l)\right) \\
&= \sum_{l\in I_{ijk}} \omega_l^{ijk} \left(\mathbf{a}_l - \sum_{i,j,k=0}^{m_u,m_v,m_w} \left(\mathbf{A}_{ijk}^{(\alpha)} + \Delta_{ijk}^{(\alpha)}\right) N_{i,K}(u_l)N_{j,L}(v_l)N_{k,M}(w_l)\right) \\
&= \Delta_{ijk}^{(\alpha)} - \sum_{f,g,h=0}^{m_u,m_v,m_w} \Delta_{fgh}^{(\alpha)} \sum_{l\in I_{ijk}} \omega_l^{ijk} N_{f,K}(u_l)N_{g,L}(v_l)N_{h,M}(w_l),
\end{aligned} \tag{38}$$

where I_{ijk} denotes the set of ordinates $\mathbf{a}_l$ that contribute to the repositioning vector Δ_{ijk}.

By arranging the repositioning vectors for the B-spline ordinates in a one-dimensional array, i.e.,

$$\begin{aligned}
\Delta^{(\beta)} &= \left[\Delta_{000}^{(\beta)}, \Delta_{001}^{(\beta)}, \cdots, \Delta_{00m_u}^{(\beta)}, \Delta_{010}^{(\beta)}, \cdots,\right. \\
&\qquad \left.\Delta_{01m_w}^{(\beta)}, \cdots, \Delta_{0m_vm_w}^{(\beta)}, \Delta_{100}^{(\beta)}, \cdots, \Delta_{m_um_vm_w}^{(\beta)}\right]^T, \\
\beta &= \alpha, \alpha+1,
\end{aligned} \tag{39}$$

we obtain the iterative form

$$\Delta^{(\alpha+1)} = (\mathbf{I} - \mathbf{C})\Delta^{(\alpha)}, \tag{40}$$

where $\mathbf{I}$ is a $(m_u+1)(m_v+1)(m_w+1) \times (m_u+1)(m_v+1)(m_w+1)$ identity matrix, and $\mathbf{C}$ is given by

$$\mathbf{C} = \begin{bmatrix} \sum_{l\in I_{000}} \omega_l^{000} N_{0,K}(u_l)N_{0,L}(v_l)N_{0,M}(w_l) & \cdots & \sum_{l\in I_{000}} \omega_l^{000} N_{m_u,K}(u_l)N_{m_v,L}(v_l)N_{m_w,M}(w_l) \\ \vdots & \ddots & \vdots \\ \sum_{l\in I_{m_um_vm_w}} \omega_l^{m_um_vm_w} N_{0,K}(u_l)N_{0,L}(v_l)N_{0,M}(w_l) & \cdots & \sum_{l\in I_{m_um_vm_w}} \omega_l^{m_um_vm_w} N_{m_u,K}(u_l)N_{m_v,L}(v_l)N_{m_w,M}(w_l) \end{bmatrix}. \tag{41}$$

Furthermore, we have the following result:

Proposition 1 *If the matrix* $\mathbf{C}x$ *(41) is nonsingular, the iterative format (40) is convergent.*

Proof To prove the proposition above, we first construct an $(N+1) \times (m_u+1)(m_v+1)(m_w+1)$ matrix, $\mathbf{B}$, which is given by

$$\mathbf{B} = \begin{bmatrix} N_{0,K}(u_0)N_{0,L}(v_0)N_{0,M}(w_0) & \cdots & N_{m_u,K}(u_0)N_{m_v,L}(v_0)N_{m_w,M}(w_0) \\ \vdots & \ddots & \vdots \\ N_{0,K}(u_N)N_{0,L}(v_N)N_{0,M}(w_N) & \cdots & N_{m_u,K}(u_N)N_{m_v,L}(v_N)N_{m_w,M}(w_N) \end{bmatrix}. \tag{42}$$

In fact, denoting

$$\Lambda = diag\left(\frac{1}{W_{den}[0][0][0]}, \frac{1}{W_{den}[0][0][1]}, \dots, \frac{1}{W_{den}[m_u][m_v][m_w]}\right), \tag{43}$$

the matrix $\mathbf{C}$ (41) can be represented as

$$\mathbf{C} = \Lambda\mathbf{B}^T\mathbf{B}. \tag{44}$$

Therefore, if the matrix $\mathbf{C}$ is nonsingular, $\mathbf{B}^T\mathbf{B}$ is a positive definite matrix, and all of its eigenvalues are positive as well as the eigenvalues of $\mathbf{C}$. On the other hand, all of the eigenvalues of $\mathbf{C}$ are less than or equal to one because $||\mathbf{C}||_\infty = 1$. In conclusion, the eigenvalues of $\mathbf{C}$ satisfy

$$0 < \lambda(\mathbf{C}) \leq 1. \tag{45}$$

Therefore, the eigenvalues of $\mathbf{I} - \mathbf{C}$ fulfill

$$0 < \lambda(\mathbf{I} - \mathbf{C}) = 1 - \lambda(\mathbf{C}) < 1. \tag{46}$$

This means that the iterative form in Eq. 40 is convergent. □

References

1. (2012). Standard specification for additive manufacturing file format (AMF). ASTM standard F2915-12 1.1st edition
2. Böhm W, Farin G, Kahmann J (1984) A survey of curve and surface methods in cagd. Comput Aided Geom Des 1(1):1–60
3. Cho W, Maekawa T, Patrikalakis N (1996) Topologically reliable approximation of composite Bézier curves. Comput Aided Geom Des 13(6):497–520
4. Cormen TH, Leiserson CE, Rivest RL (1990) Introduction to Algorithms. Cambridge, Press, MA
5. Davis TA (2006) Direct methods for sparse linear systems. SIAM philadelphia
6. Deng C, Lin H (2014) Progressive and iterative approximation for least squares b-spline curve and surface fitting. Comput Aided Des 47:32–44
7. do Carmo PM (1976) Differential Geometry of Curves and Surfaces. Prentice-Hall, Inc., Englewood Cliffs
8. Farouki RT, Rajan VT (1988) Algorithms for polynomials in Bernstein form. Comput Aided Geom Des 5(1):1–26
9. Gao W, Zhang Y, Ramanujan D, Ramani K, Chen Y, Williams CB, Wang CCL, Shin YC, Zhang S, Zavattieri PD (2015) The status, challenges, and future of additive manufacturing in engineering. Comput Aided Des 69:65–89
10. Modeling and fabrication of heterogeneous three-dimensional objects based on additive manufacturing (2013)
11. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput Methods Appl Mech Eng 194(39-41):4135–4195
12. Jamieson R, Hacker H (1995) Direct slicing of CAD models for rapid prototyping. Rapid Prototyp J 1(2):4–12
13. Kajiya JT (1982) Ray tracing parametric patches. Comput Graphics (SIGGRAPH '82 Proceedings) 16(3):245–254
14. Khoda AKM, Ozbolat IT, Koc B (2013) Designing heterogeneous porous tissue scaffolds for additive manufacturing processes. Comput Aided Des 39(12):1507–1523
15. Kineri Y, Wang M, Lin H, Maekawa T (2012) B-spline surface fitting by iterative geometric interpolation/approximation algorithms. Comput Aided Des 44(7):697–708
16. Lin H (2010) The convergence of the geometric interpolation algorithm. Comput Aided Des 42(6):505–508
17. Lin H, Jin S, Hu Q, Liu Z (2015) Constructing b-spline solids from tetrahedral meshes for isogeometric analysis. Comput Aided Geom Des 35-36:109–120
18. Lin H, Wang G, Dong C (2004) Constructing iterative non-uniform B-spline curve and surface to fit data points. Sci in China 47(3):315–331
19. Liu H, Maekawa T, Patrikalakis N, Sachs E, Cho W (2004) Methods for feature-based design of heterogeneous solids. Comput Aided Des 36(12):1141–1159
20. Ma W, But WC, He P (2004) NURBS-Based adaptive slicing for efficient rapid prototyping. Comput Aided Des 36(13):1309–1325
21. Maekawa T (1999) An overview of offset curves and surfaces. Comput Aided Des 31(3):165–173
22. Maekawa T, Matsumoto Y, Namiki K (2007) Interpolation by geometric algorithm. Comput Aided Des 39(4):313–323
23. Martin T, Cohen E, Kirby R (2009) Volumetric parameterization and trivariate B-spline fitting using harmonic functions. Comput Aided Geom Des 26(6):648–664
24. Martin T, Cohen E, Kirby R (2012) Direct isosurface visualization of hex-based high-order geometry and attribute representations. IEEE Trans Vis Comput Graph 18(5):753–766
25. Martin W, Cohen E (2001) Representation and extraction of volumetric attributes using trivariate splines: a mathematical framework. In: Proceedings of the sixth ACM symposium on Solid modeling and applications, pp 234–240
26. Ozbolat IT, Koc B (2011) Multi-directional blending for heterogeneous objects. Comput Aided Des 43(8):863–875
27. Pan Y, Zhou C, Chen Y, Partanen J (2014) Multitool and multi-axis computer numerically controlled accumulation for fabricating conformal features on curved surfaces. ASME J Manufacturing Sci and Eng 136(3)
28. Park S, Lee K (1997) High-dimensional trivariate nurbs representation for analyzing and visualizing fluid flow data. Comput Graph 21(4):473–482
29. Patrikalakis N, Maekawa T (2002) Shape interrogation for computer aided design and manufacturing. Springer-Verlag, Heidelberg
30. Paul R, Anand S (2015) A new steiner patch based file format for additive manufacturing processes. Comput Aided Des 63:86–100
31. Piegl L, Tiller W (1997) The NURBS book 2nd Ed Springer-Verlag New York Inc.
32. Samanta K, Koc B (2005) Feature-based design and material blending for free-form heterogeneous object modeling. Comput Aided Des 37(3):287–305
33. Schwahn ES (2015) Using controlled curing in a custom stereolithography-based 3D printing machine to obtain graded property variations. Master thesis, University of Nebraska-Lincoln
34. Sethian JA (1999) Level Set Methods and Fast Marching Methods: Evolving Interfaces in Computational Geome- try, Fluid Mechanics, Computer Vision, and Materials Science. Cambridge University Press
35. Starly B, Lau A, Sun W, Lau W, Bradbury T (2005) Direct slicing of STEP based NURBS models for layered manufacturing. Comput Aided Des 37:387–397

36. Wang X, Qian X (2014) An optimization approach for constructing trivariate B-spline solids. Comput Aided Des 46:179–191
37. Yamaguchi F (1977) A method of designing free form surfaces by computer display (1st report)(in Japanese). Precision Machinery 43(2):168–173
38. Yang P, Qian X (2007) A B-spline-based approach to heterogeneous objects design and analysis. Comput Aided Des 39(2):95–111
39. Yoshihara H, Yoshii T, Shibutani T, Maekawa T (2012) Topologically robust b-spline surface reconstruction from point clouds using level set methods and iterative geometric fitting algorithms. Comput Aided Geom Des 29:422–434