Computers & Graphics 89 (2020) 167–177

Contents lists available at ScienceDirect

Computers & Graphics

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

Special Section on SMI 2020

Proxy-driven free-form deformation by topology-adjustable control lattice

Yuzhe Zhang[a], Jianmin Zheng[a,*], Yiyu Cai[b]

[a] School of Computer Science and Engineering, Nanyang Technological University, Singapore
[b] School of Mechanical and Aerospace Engineering, Nanyang Technological University, Singapore

ARTICLE INFO

Article history:
Received 4 May 2020
Accepted 5 May 2020
Available online 16 May 2020

Keywords:
Free-form deformation
interface
proxy
topology
T-spline volume
Interactive editing

ABSTRACT

This paper presents a flexible free-form deformation (FFD) method that overcomes two limitations of many FFD methods: the cumbersome interface and the fixed topology of the control lattice. The novelty of the method mainly lies in two aspects: (1) a specifically-tailored T-spline volume with its local refinement is introduced to define the deformation space and construct the deformation tool, and (2) a proxy represented as a three-dimensional mesh close to the object to be deformed is constructed from the control lattice of the T-spline volume and is used to manipulate and drive the deformation. While the method inherits all the virtues of FFD such as C^2 continuous local and global modifications, the proxy provides a much concise and intuitive interface for the user to edit the deformation in a way similar to cage-based deformations. Moreover, the method allows local topology changes of the proxy after parameterization, which is not available for most FFD methods or even cage-based deformations. These features enable more complicated free-form deformation to be achieved in a flexible and intuitive manner, as demonstrated by many experimental examples.

1. Introduction

Free-form deformation (FFD) is a well-known spatial deformation technique that deforms an object by warping its ambient space through moving lattice control points [1]. It usually proceeds in four steps: creation of the volumetric lattice of control points (i.e., the deformation tool) that also implicitly defines a deformation space, parameterization that embeds the object to be deformed in the deformation space, deformation triggered by moving the control lattice, and transfer of the deformation to the object. Due to the ability of offering smooth deformation, the simplicity of its mathematical formalism and no restrictions on the representation of the object to be deformed, FFD is a very popular tool in practice. Many modeling and animation software packages such as Autodesk® 3DS Max® and Autodesk® Maya® support FFD or its variants.

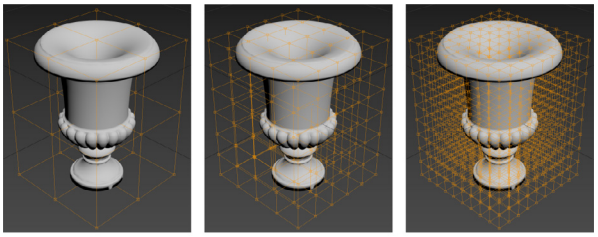
However, FFD has two major limitations. The first one is that if a control lattice with a large number of control grids is set, the interface for the control of deformation looks very crowded or even messy (see Fig. 1(b) and (c)). This not only blocks the view of the object to be deformed but also causes inconvenience for interactive editing of the control lattice points. The second limitation is that after the initial creation of the control lattice, the paramterization of the object with respect to the deformation space is determined and then the topology of the control lattice is fixed during the subsequent deformation process. That is, the only freedom that FFD offers after the setup of the initial control lattice is the relocation of the control lattice points. This seriously affects the capability of continuous, interactive deformation processes.

In practice the dimension of the lattice is usually very small. For example, the default dimensions in Autodesk® 3DS Max® range from $2\times2\times2$ to $4\times4\times4$, which limits the local control of deformation in fine detail (see Fig. 1(a)). To overcome this limitation, 3DS Max® offers configurable FFD modifiers. To conform the lattice to an object, the "Set Volume" function lets the user set the initial lattice, as shown in Fig. 2(a). However, this process could be done only at the initial stage and is also tedious. The "Conform to Shape" function is provided to fit the control points to the shape of the object automatically, which moves each control point along the line connecting the current location and the object's center towards the object by a certain distance (see Figs. 2(b,c)). However, this function may not work well for objects with complicated shape (see Fig. 2(c)).

Besides FFD, most of FFD's extensions such as [2–5] also suffer from the two limitations. This paper presents a solution to overcoming these limitations. The basic idea is to construct a surface

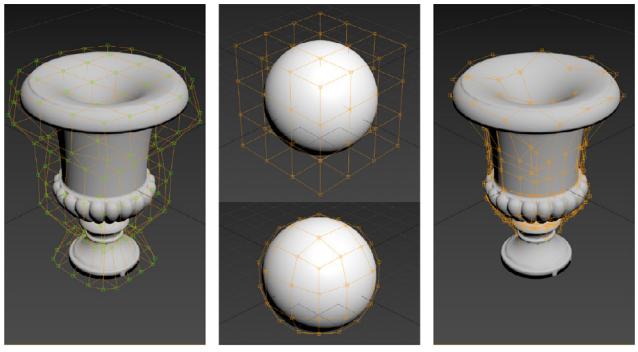
* Corresponding author.
E-mail address: asjmzheng@ntu.edu.sg (J. Zheng).

https://doi.org/10.1016/j.cag.2020.05.013
0097-8493/



(a) (b) (c)

Fig. 1. FFDs in Autodesk® 3DS Max®. (a): A box control lattice of size $3 \times 3 \times 3$; (b): A box control lattice of size $5 \times 6 \times 7$; (c): A box control lattice of size $10 \times 10 \times 10$. The control lattices in (b) and (c) look crowded.



(a) (b) (c)

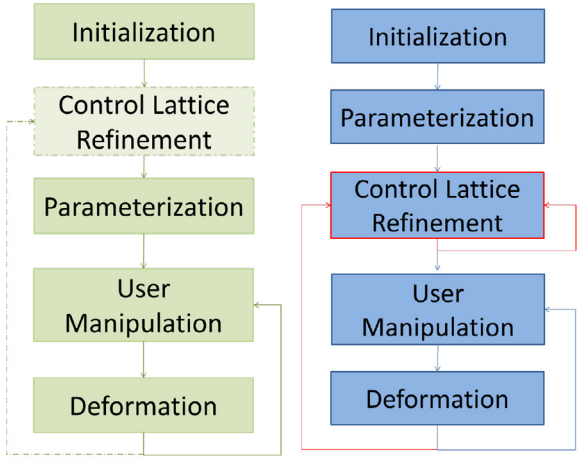
Fig. 2. Configurable FFDs in Autodesk® 3DS Max®. (a): Manually set a control lattice by "Set Volume"; (b): Automatically adapt a box control lattice to a sphere object by "Conform to Shape"; (c): Automatically adapt a box control lattice, which actually does not fit the shape (the bottom part) well.

mesh conforming to the shape of the object, which is called a *proxy* serving as the interface or handle for deformation. We construct the proxy from the volumetric lattice such that the proxy encloses the object and is close to the surface of the object as well. The proxy contains much fewer control points than the volumetric lattice, thus providing clearer visualization of the object. More importantly, the proxy is used to manipulate and drive the deformation, like the cage in cage-based deformations [6,7].

The implementation of the proxy concept is made possible by a specifically-tailored T-spline volume with its local refinement, and an extension of as-rigid-as-possible surface deformation to volumetric meshes. The T-spline volume and its local refinement algorithm define a solid as the deformation space, permit the generation of a flexible control lattice and hence a proxy that conforms well to the shape of the object, and enable the topology of the control lattice and the proxy to be changed after parameterization, thus allowing continuous deformations in different levels of detail. The extension of surface deformation to volumetric meshes connects the proxy to the volumetric lattice and makes the lattice deform according to the change of the proxy, thus triggering the deformation of the ambient space.

1.1. Technical Novelty

The paper presents a proxy-driven free-form deformation technique with topology-adjustable control lattice. While inheriting all the virtues of FFD such as C^2 continuous global and local modifications, the proposed deformation provides a novel paradigm for free-form deformation, which matches several perspectives for good spatial deformation including locality, ease of use and topology alteration [8,9]. Particularly, the user works with the proxy that provides a much concise and intuitive interface. The technique allows local topology changes of the proxy after parameterization,



(a) *Conventional FFD* **(b)** *Proposed deformation*

Fig. 3. Comparison of the worksflows: conventional FFD vs proposed deformation.

which is not available for most FFD methods or even cage-based deformations. This actually changes the workflow of the conventional FFD. As shown in Fig. 3, every time when the control lattice performs refinement (i.e., change of topology), the conventional FFD needs to re-calculate parameterization, which implies starting a new FFD. By contrast, our method permits adjustment of the control lattice both geometrically and topologically without the need of re-parameterization.

The FFD method that is closest to our work is w-TFFD [10]. However, there are two significant differences between our method and w-TFFD. First, [10] introduced weighted T-spline volumes and used weighted T-splines as parametric volumes for free-form deformation. As a result, the calculation of parameterization in w-TFFD is generally very complicated and any change on the topology of the control lattice requires re-calculation of parameterization. Our method uses a tailored T-spline volume and makes full use of advantages of T-splines: supporting non-tensor-product structure and allowing truly local refinement. Therefore the parameterization in our method is trivial and there is no need to perform reparameterization. Second, w-TFFD generates multi-resolution lattices for weighted T-spline volumes. While multi-resolution provides convenience to handle different levels of detail, it also introduces extra overheads such as managing the levels and may cause the interface messy. Our method does not need to introduce multiple levels and just performs local refinement at the region that needs more degrees of freedom for fine detail control. Moreover, our method introduces the concept "proxy" to simplify the interface. The two limitations mentioned earlier are still there for w-TFFD.

It is also worth pointing out that this paper does not aim to improve cage-based methods though they do not allow topology change of the cage either. Our goal is to improve FFD techniques so that those software packages using FFD can have an enhanced FFD version.

2. Related work

Geometric deformation has been extensively studied in computer graphics. There is rich literature on this topic. Roughly geometric deformation can be classified into surface deformation and spatial deformation [11]. Surface deformation deforms the surface using differential coordinates that represent the local shape of the surface. In the last decade, many mesh deformation methods such as-rigid-as-possible surface deformation were developed [12–16]. Spatial deformation is an indirect approach that deforms a space and thus any object contained in the space is deformed accord-

ingly. Our work basically falls in the category of spatial deformation. This section briefly reviews spatial deformation, especially free-form deformation. An excellent survey on spatial deformation from a user-centered perspective is given in [9].

Spatial deformation was initially introduced in 1984 [17], which uses a set of transformations such as stretching, bending, twisting and tapering to deform a solid object. Later Sederberg and Parry introduced free-form deformation that creates an initial 3D lattice by regularly subdividing a parallelepiped and defines the deformation space as a trivariate Bézier volume by the lattice points [1]. The objects are embedded in the space. The parallelepiped form of the lattice allows a linear parameterization for any point of the objects. When the lattice is deformed, the deformation can be easily transferred to the objects by simply substituting their parameters into the equation of the trivariate Bézier volume. Due to its simplicity, the method is widely used. Since then, extensions have been done in several directions :

- Local Control
  The original FFD used trivariate Bernstein polynomials, which lacks local control. To support local control, the basis functions with finite support can be employed to replace Bernstein polynomials. For example, uniform B-spline basis functions have been used in building FFD [18]. To achieve linear precision for parameterization, the control points of the lattice should be set up properly. Non-uniform rational B-spline (NURBS) volumes have also been proposed to define FFD, which provide local control and more degrees of freedom for manipulating the deformation [3]. With general setting of NURBS, the parameterization of the deformation space is nonlinear and thus a complicated inverse point problem is required to be solved. Recently, truncated hierarchical B-splines were used to provide a hierarchical and local refinable FFD [19], which is even able to check the injectivity of the deformation in real time.
- Complex Lattice Topology
  The lattice in the original FFD has a regular structure which limits the shape of the deformable space. To support a more general lattice structure, Extended FFD was introduced in [2], in which the control lattices could be shaped without the constraint of the parallelepiped. A continuous FFD (CFFD) based on piecewise Bézier tetrahedra was proposed in [20], which allows to build lattices with complicated shape. By developing a volume version of Catmull-Clark subdivision [4], allows the user to define a lattice of arbitrary topology, which provides great flexibility in specifying the lattice. However, the subdivision based FFD is computationally costly and memory intensive. To overcome the tartan-like pattern of cells in FFD, a weighted T-spline volume was introduced to FFD, giving the w-TFFD scheme [10]. The lattice was constructed by adaptive octree subdivision of the bounding volume of the object and the deformation space was defined by the weighted T-spline. The w-TFFD can create a lattice with T-junctions, which approximates the boundary of the object well and improves the viewing of the object. However, the parameterization for w-TFFD is computationally expensive and w-TFFD does not allow topology changes after parameterization.
- Direct Manipulation
  It is well-known that Bézier or B-spline control points generally do not lie on the shape. The direct manipulation of one and multiple constraints on the object to be deformed was proposed in [21], which provides more direct control of deformation and matches two criteria “easy to use” and “precise control” required in spatial deformation [9]. A set of tools such as hammer, crimp and reset tools were developed for users to directly interact with the object in NURBS-FFD [8]. An explicit solution to deforming an object to pass through target point(s) in NURBS-FFD was derived in [22]. Our work does not focus on direct manipulation, which however can be implemented in our framework.
- Cage-based Deformation
  Inspired by the concept of the original spatial deformation, cage-based deformation was proposed, which constructs a coarse control cage to enclose the object to be deformed tightly and then deforms the cage to trigger the deformation of the space bounded by the cage. The key component here is the parameterization that assigns local coordinates to any point in the bounded space according to the combination of the vertices of the cage. For this purpose, various coordinates have been developed. Mean value coordinates (MVCs) were first introduced by Floater [23] and generalized by Ju et al. [6], and Hormann and Floater [24]. MVCs allow a closed triangular surface as the cage to deform its interior space. Harmonic coordinates of [7] and positive MVCs of [25] were proposed to avoid negative coordinates. To have good preservation of shape and details, Green coordinates were introduced [26], which give quasi-conformal mappings in 3D space. Huang et al. introduced a modified barycentric interpolation by adding a local transformation at each control point to minimize the first-order discontinuity [27]. To further improve the quality of deformed shapes, Ben-Chen et al. [28] proposed to find a harmonic mapping of the domain, which satisfies the specified deformation constraints, through a global variational procedure. Compared to FFD, cage-based deformations have more intuitive interface and more flexibility in controlling the deformation handles (i.e., cage). However, they have the similar limitation that the connectivity of the cage cannot be altered after parameterization.
- Other deformation methods
  Besides FFD and cage-based deformation, many other techniques have also been developed for spatial deformation. For example, the linear blending skinning is a widely used technique in character animation [29], where the deformation or animation is defined and driven by rigging, skinning and manipulation of bones. A scalar-field based deformation was proposed [30], in which scalar fields defined by implicit functions are used as the embedding spaces and each point on the object to be deformed is relocated to remain on the same level set once the fields change. The scalar-field based deformation provides a mesh-free technique. It was further combined with physics-based editing to manipulate point set surfaces [31]. While most cage-based deformation is affected by both topology and geometric relations among the control vertices, Jacobson et al. introduced bounded biharmonic weights that can be used for cages of arbitrary topology and also for points and bones to produce smooth deformation [32]. Cerveró et al. proposed two cage-free methods based on so-called T-Celestial Coordinates and T-Celestial Coordinates, which depend only on the positions of the control vertices [33]. Different from these methods, our work in this paper still follows the conventional FFD and requires the mesh structure.

3. Proposed deformation method

This section presents the proposed deformation method, which falls in the category of FFD. Following the conventional FFD, the method involves the following four steps. First, we need to set up the control lattice using a tailored T-spline volume and then construct the proxy using T-spline refinement algorithm (see Section 3.1). Second, we parameterize each point of the object with respect to the deformation space defined by the T-spline volume. Due to the special construction of the proxy, the parameterization is linear, which can be explicitly written out. Third, the user edits the control point of the proxy to deform the proxy. The change of

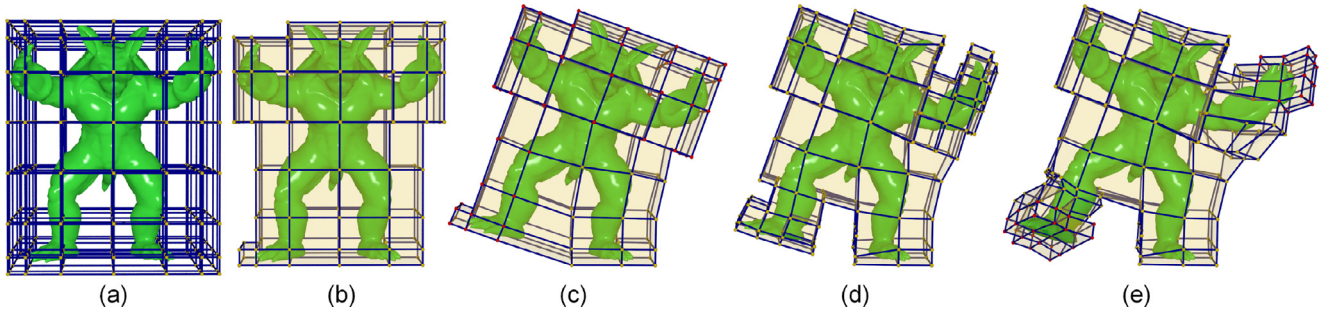


Fig. 4. Illustration of the proposed deformation: (a) initial dense B-spline volume control lattice, (b) concise proxy for deformation, (c) a deformation driven by the proxy points in red color, (d) topology refined at the lower-left and upper-right corners, and (e) further deformation driven by the refined control grids in red color. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

the proxy will drive the control lattice to deform. This is achieved by a surface-driven as-rigid-as-possible volumetric mesh deformation as explained in Section 3.4. The approach avoids manually manipulating many control lattice points and simplifies the deformation process. Fourth, the deformed lattice defines the deformation of the space, which is passed to the object through the Eq. (1) of the T-spline volume with new control lattice points. Fig. 4 illustrates the deformation process.

3.1. T-spline volume

While a T-spline surface is defined by a T-mesh that is topologically a rectangular mesh with T-junctions [34,35], we can similarly define a T-spline volume by a T-lattice that is a parallelepiped possibly with T-junctions. T-junctions are the end points of partially terminated grid lines. T-spline volumes have been used in FFD, isogeometric analysis and others [10,36–39]. Note that general T-spline volumes could have very flexible T-lattices and local refinement. This is good for shape representation and modeling, but could lead to very complicated lattice that is not suitable for FFD. Thus a simple extension of FFD just by replacing Bézier or B-spline volume by T-spline volume may not work well. Therefore we introduce a class of degree $3 \times 3 \times 3$ T-spline volume that is tailored for FFD deformation.

The T-lattice for the tailored T-spline volume initially comes from a parallelepiped. Each edge of the parallelepiped is assigned to a non-negative number called a knot interval. There is a rule for the assignment of knot intervals: the knot intervals for all the edges in the same cell that are parallel are equal. A face of the lattice is defined by a minimal connected loop of vertices. A cell of the lattice is defined to be the region of space bounded by a set of faces. The parallelepiped could be refined by inserting new points locally, which may create T-junctions. While T-spline volume could have very general local refinement, to make the adjustment of connectivity of the control lattice simple and intuitive, we only allow three types of insertion: (1) a point is inserted to an existing edge, which splits the edge into two edges and divides the knot interval proportionally into two, each of which is assigned to the corresponding new edges; (2) A point is inserted to the center of an existing face and splits the face into four sub-faces; (3) A point is inserted into an existing cell to split the cell into eight sub-cells.

The three directions of the parallelepiped are labeled by parameters s, t and u. If one vertex on the T-lattice is specified as the local origin with $(s, t, u) = (0, 0, 0)$, all the other vertices on the T-lattice have a unique parameter coordinate, which can be computed by traveling along the edges of the T-lattice. For any point P_i on the T-lattice whose parameter coordinates are (s_{i0}, t_{i0}, u_{i0}), we pass a ray along the positive s-direction. We record the s-coordinates $s_{i,1}$, $s_{i,2}$ for its first two hits with an existing point, edge or face. Similarly, we can find the s-coordinates $s_{i,-1}, s_{i,-2}$ along the negative s-directions. Thus we obtain a local knot vector $\mathbf{s}_i = \{s_{i,-2}, s_{i,-1}, s_{i,0}, s_{i,1}, s_{i,2}\}$ that can define a cubic B-spline basis function $N[\mathbf{s}_i](s)$. In the same way, we can define another two B-spline basis functions $N[\mathbf{t}_i](t)$, $N[\mathbf{u}_i](u)$ along the t and u directions. Thus the equation of the T-spline volume is given by

$$\mathcal{T}(s,t,u) = \frac{\sum_{i=1}^{n} P_i w_i N[\mathbf{s}_i](s) N[\mathbf{t}_i](t) N[\mathbf{u}_i](u)}{\sum_{i=1}^{n} w_i N[\mathbf{s}_i](s) N[\mathbf{t}_i](t) N[\mathbf{u}_i](u)} \tag{1}$$

where P_i are the control points on the T-lattice, n is the total number of the control points, and w_i are the weights associated with control points P_i.

3.2. Construction of control lattice and proxy

Now we discuss how to construct the control lattice and proxy for deformation. Given an object to be deformed, we find a bounding box. Assume that the two opposite corners of the bounding box have coordinates $(min.x, \min.y, \min.z)$ and $(max.x, \max.y, \max.z)$. We first generate a parallelepiped for our initial control lattice. If the number of control points in $x(s)$ direction is n_s, the number of knots in x/s direction will be $n_s + 4$. To make the exterior faces of the parallelepiped be close to the object, we set the knots to satisfy the Bézier end conditions. That is, we choose multiple knots at the two ends and uniform knots at the other places: $\mathbf{s} = (s_{-2}, s_{-1}, s_0, \cdots, s_i, \cdots, s_{n_s-1}, s_{n_s}, s_{n_s+1}) = (0, 0, 0, 0, k, 2k, \cdots, (i-1)k, \cdots, 1, 1, 1, 1)$, where $k = \frac{1}{n_s-3}$. The knots of $y(t)$ and $z(u)$ directions can be set in the same way. Consequently, the parameter domain for the deformation space is $[0, 1] \times [0, 1] \times [0, 1]$.

As for geometry of the control point P_{ijk}, its Cartesian coordinates (x_i, y_i, z_i) are set at their Greville abscissae in order to achieve the property of linear precision:

$$\begin{cases} \bar{x}_{ijk} = min.x + \frac{s_{i-1}+s_i+s_{i+1}}{3} * (max.x - min.x) \\ \bar{y}_{ijk} = min.y + \frac{t_{j-1}+t_j+t_{j+1}}{3} * (max.y - min.y) \\ \bar{z}_{ijk} = min.z + \frac{u_{k-1}+u_k+u_{k+1}}{3} * (max.z - min.z) \end{cases} \tag{2}$$

The task of parameterization is to calculate the parametric coordinates of the object which represent the link of the object to the control lattice. Due to the special set-up of the control points, the parameterization has the linear precision property. Thus the parameter coordinates of any point $P = (x, y, z)$ contained in the deformation space can be explicitly obtained:

$$s = \frac{x - min.x}{max.x - min.x}, t = \frac{y - min.y}{max.y - min.y}, u = \frac{z - min.z}{max.z - min.z}.$$

Once all the control points P_{ijk} are computed, they define a B-spline volume that is actually a special case of a T-spline volume of (1) with all the weights being 1. We then refine the control lattice by inserting additional control points locally, or perform octree

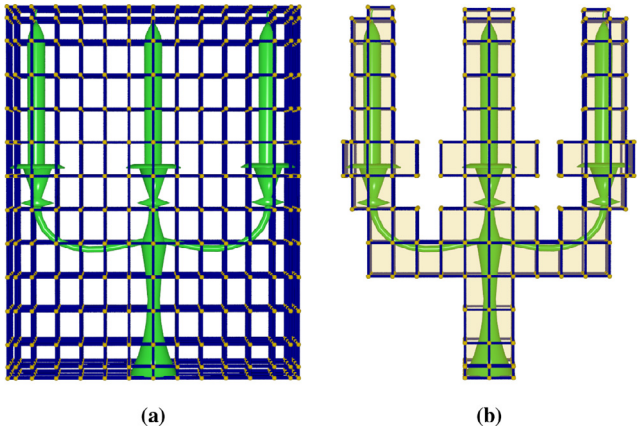


Fig. 5. (a) control lattice vs (b) proxy.

partition to make the control lattice fit the shape of the embedded object. The refinement is performed using the T-spline volume local refinement algorithm, which assures no change of the deformation space and its parameterization. Since the control lattice contains too many control points that are hard to manipulate, we choose to leave the corners of those cells intersecting with the object, which are closer to the object and have more influence on deforming the object and hide the rest of control points. The corner points form a mesh which we call a proxy. Fig. 5 shows a comparison between the whole control lattice and the proxy. The proxy looks much cleaner.

3.3. User interface

Once the control lattice and the proxy are set up, the proxy is displayed by default and this significantly simplifies the UI. The user can switch back and forth between the control lattice and the proxy if needed. Tools are also developed for the user to select control points, edges and cells for freezing, moving or rotating.

The proxy can be refined by locally adding points for fine control of the shape, which results in local topology changes of the proxy. The user can select one or more edges, faces, or cells to subdivide. The edge subdivision just adds a new point in the middle of the edge. The face subdivision splits the face into 4 sub-faces by inserting 4 points in the middle of the four boundary edges and 1 point in the center of the face. The cell subdivision is an octree partition that creates 8 sub-cells. To keep the parameterization of the deformable space unchanged, the refinement is done by T-spline volume local refinement algorithm. Note that the refinement may cause insertion of some extra points or move some existing control points to new locations. In addition, the refinement or deformation may cause the change of cells containing the boundary of the object. Thus the proxy will be updated by removing the cells containing no point on the boundary of the object and adding the cells containing point(s) on the boundary of the object. Fig. 6(a) shows the octree partition of all cells contained in the current proxy. The updated versions of the proxy after the refinements are shown in Figs. 6(b)–(d). Another example is given in Fig. 7 shows another example of octree partition.

Moreover, the proxy serves as the deformation handle. The user directly edits the proxy both geometrically and topologically. As illustrated in Fig. 8, the user could refine the proxy (Fig. 8(a)) by inserting several control points at top as shown in Fig. 8(b). Through relocating the positions of control points, the geometry of the embedded object could be deformed as shown in Fig. 8(c). Since there is no re-parameterization, the manipulation of topology and geometry can be done alternately as demonstrated in Fig. 8(d).

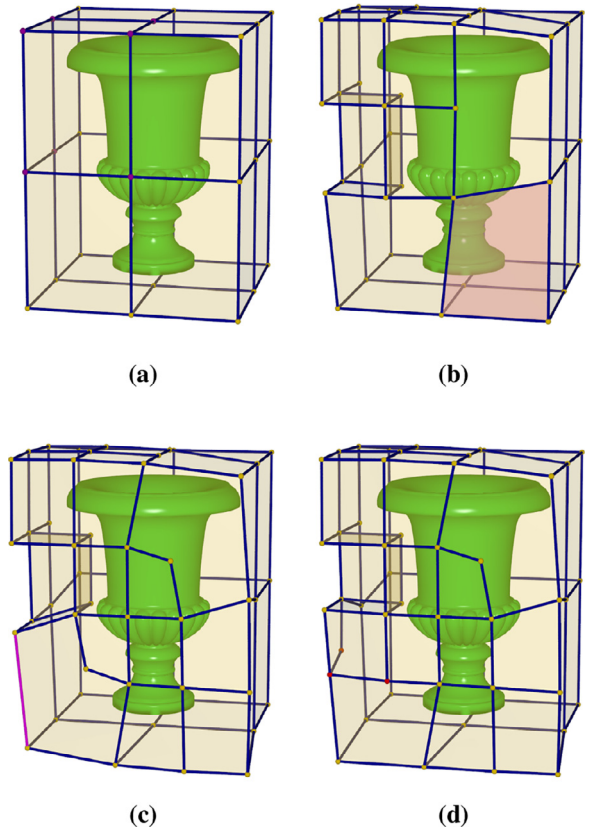


Fig. 6. Topology refinement. (a):Original lattice of size $2 \times 2 \times 2$; (b)–(d): Refinements by Octree partition.

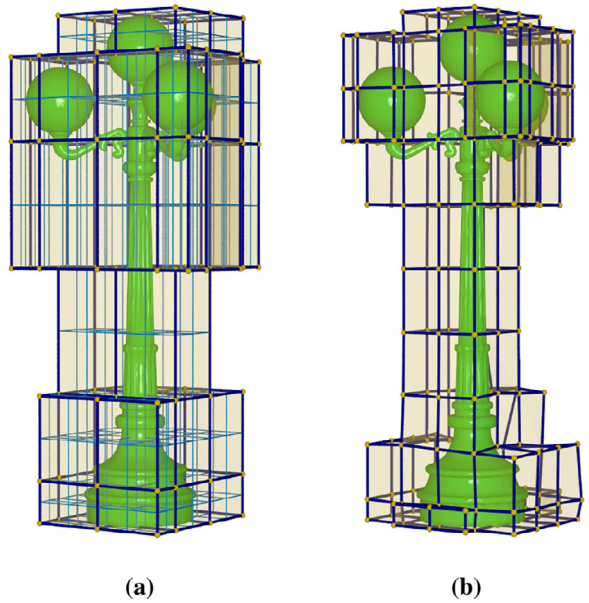


Fig. 7. Octree partition. Left: topological division of the current proxy; Right: updated proxy after the refinement.

3.4. Surface-driven volumetric deformation

To introduce a deformation, the user moves one or a few control points on the proxy to new locations. There are a lot of control points on the proxy or in the lattice that are not specified. They should move in an appropriate manner so as to respond to the user's input intelligently. Our idea is to use an as-rigid-as-possible

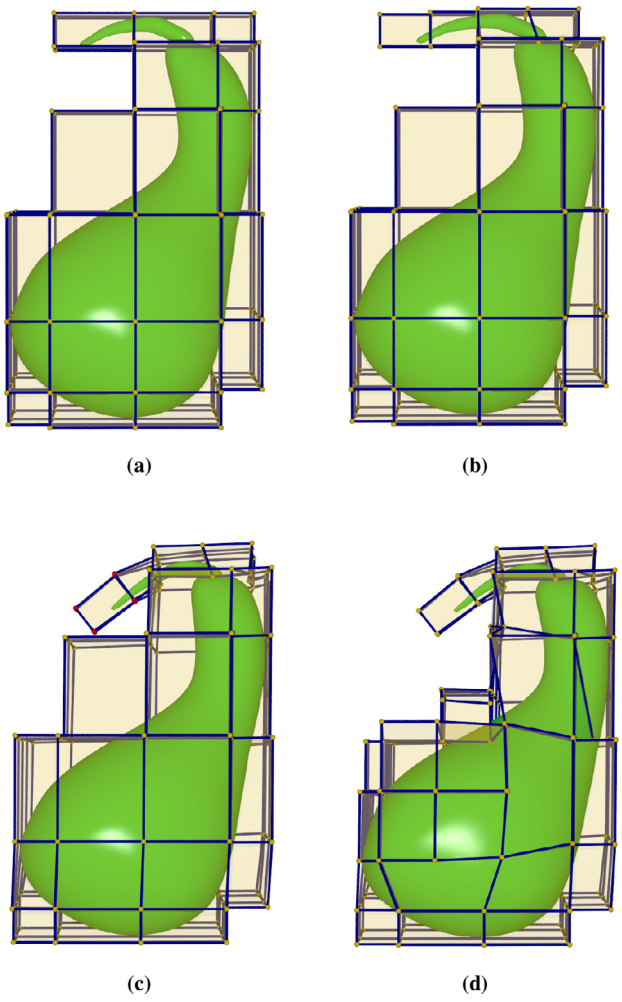


Fig. 8. Editing the topology and geometry of the proxy.

approach to displace those points to preserve the surface and volumetric details. Specifically, the new locations of the control points P_i are found as a solution to the following minimization problem:

$$\min_{\{P_i\}} \; w_{1F}E_{1F} + w_M E_M + w_V E_V + w_C E_C \tag{3}$$

where w_{1F}, w_M, w_V and w_C are the tradeoff factors, and E_C, E_M, E_V and E_{1F} are four terms accounting for target deformation, proxy shape, volume shape, and volumetric rigidity constraints, respectively. Their formulations are explained below.

(1) Target deformation term
Let C denote the set of the control points on the proxy moved by the user. We introduce the target deformation term

$$E_C = \sum_{P_i \in C} \left\| P_i' - Q_i \right\|^2 \tag{4}$$

where Q_i is the target position of P_i and P_i' represents the updated P_i. Minimizing E_C enables the deformation to move towards the target.

(2) Surface Laplacian term
For any point P_i on the proxy M, the Laplace-Beltrami operator at P_i is defined as

$$L_M(P_i) = P_i - \sum_{P_j \in N_s(i)} w_{ij} P_j$$

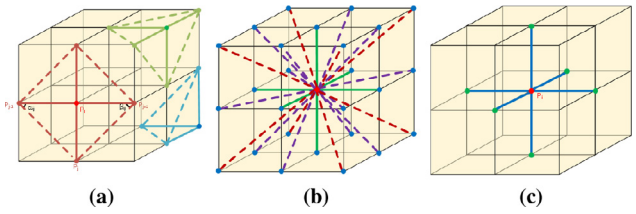


Fig. 9. Neighborhoods for (a) Laplace-Beltrami operator, (b) Volumetric Laplacian, and (c) One-form.

where $N_s(i)$ denotes the set of points in the neighborhood of point P_i over the proxy M and w_{ij} is the common cotangent weight (see Fig. 9(a)). To preserve the detail of the proxy as a mesh, we do not want the Laplace–Beltrami operator of the proxy at the vertices of the proxy which do not belong to C to be changed too much. Without causing ambiguity, we also let M represent the set of the vertices of the proxy. We define the surface Laplacian term as follows:

$$E_M = \sum_{P_i \in M \setminus C} \left\| L_M(P_i') - L_M(P_i) \right\|^2. \tag{5}$$

(3) Volumetric Laplacian term
Analogous to the Laplace-Beltrami operator on manifolds, we can introduce the volumetric Laplace operator for volumetric meshes [40]. In particular, we construct a volumetric graph by introducing virtual edges as the diagonals on each facet of a cube and inside of the cube. Then the volumetric Laplacian operator at point P_i in the volume lattice can be defined as the difference between P_i and a linear combination of its neighboring points in the volumetric graph:

$$L_V(P_i) = P_i - \sum_{P_j \in N_v(i)} w_{ij} P_j$$

where w_{ij} is set to be the normalized reciprocal of edge length so that $\sum_{j \in N_v(i)} w_{ij} = 1$, and $N_v(i)$ is the set of neighboring points for control point P_i over the volumetric graph, which includes at most 26 neighbors (see Fig. 9(b)).
To preserve the shape of lattice in the volumetric way, we define volumetric Laplacian term:

$$E_V = \sum_{P_i \in S} \left\| L_V(P_i') - L_V(P_i) \right\|^2 \tag{6}$$

where S stands for the control lattice.

(4) One-form term
To enhance the consistency of local rigid transformations, [41] introduced a face-based local cell and kept the deformation in each local cell as rigid as possible. In our work, we introduce 1-form for each point in a mesh, which is a local structure formed by all the vectors of its incident edges (see Fig. 9(c)). Preserving the 1-form will keep the local structure and also the edge lengths, giving a rigid deformation. Thus we define the one-form term as follows:

$$E_{1F} = \sum_{P_i \in S} \sum_{P_j \in N_L(i)} \left\| P_i' P_j' - P_i P j \right\|^2 \tag{7}$$

where $N_L(i)$ denotes the set of points in the neighboring of point P_i over the control lattice S.

The minimization problem of (3) can be converted to solving a system of linear equations

$$[CM]\left[P'\right] = [B] \tag{8}$$

where $[CM]$ represents the coefficient matrix and $[B]$ is a column vector. To propagate the local transformation, we follow the

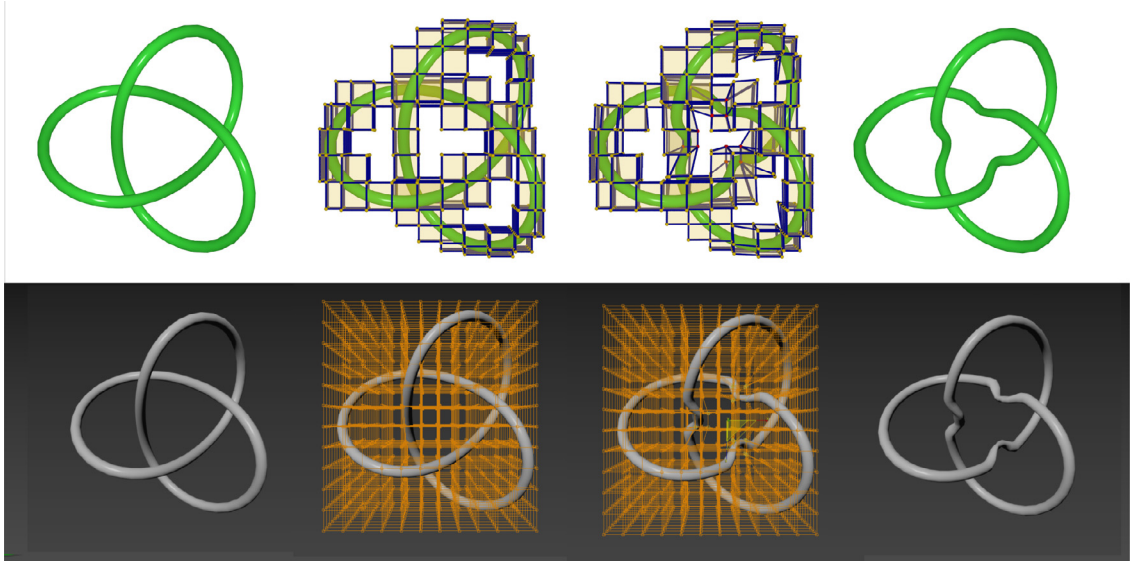


Fig. 10. Comparison of the proxy-driven deformation (top) and Autodesk® 3DS Max®'s FFD (bottom).

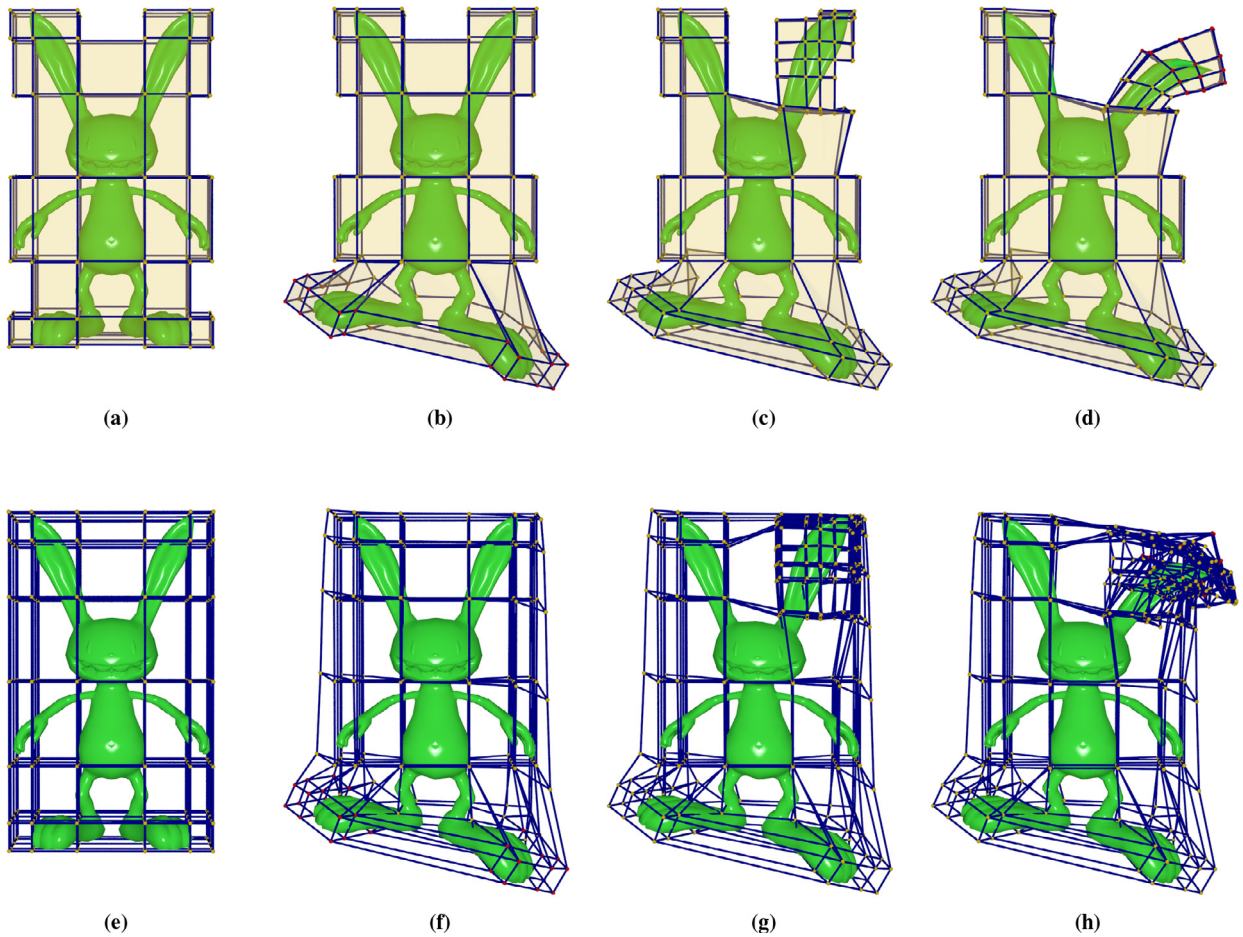


Fig. 11. Deformation of a " Rabbit" model: the proxy (top) vs the underlying T-spline control lattice (bottom).

approach of [16]. During each iteration, P_i' is obtained by solving Eq. (8). A rotation matrix R_i is then computed for each control point by minimizing $\sum_{j\in N(i)} \|(P_i' - P_j') - R_i(P_i - P_j)\|^2$. The rotation matrix is plugged into the righthand side of Eq. (8). The process repeats several iterations until convergence.

4. Experiments

This section presents examples to show the capacity and efficiency of the proposed deformation method. The algorithm was implemented using C++ under Windows 10. The experiments were

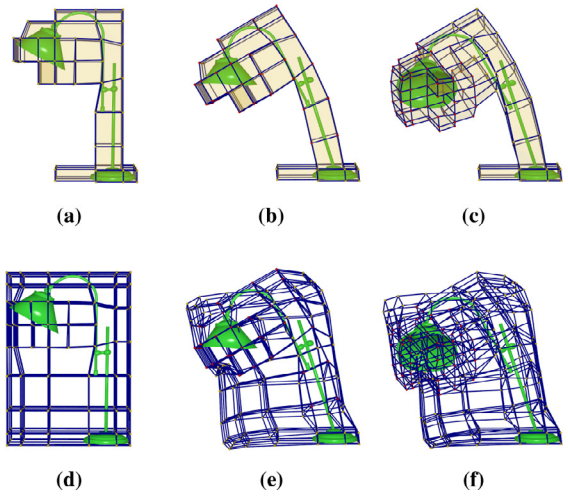


Fig. 12. Deformation of a "Lamp" model: the proxy (top) vs the underlying T-spline control lattice.

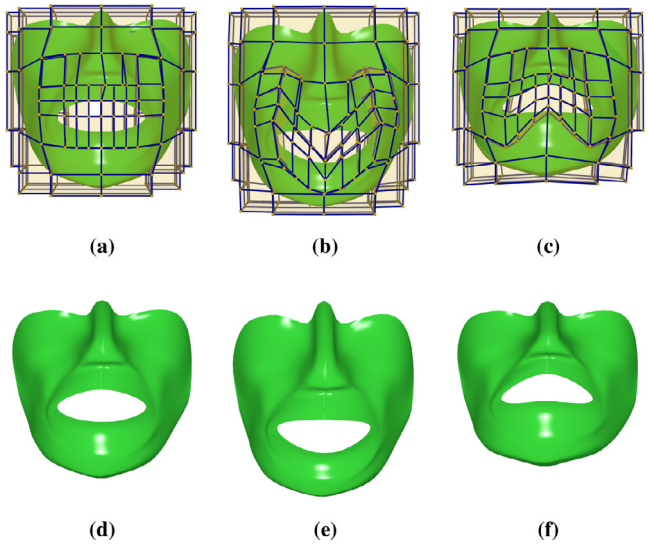


Fig. 13. "Face" model: local deformation.

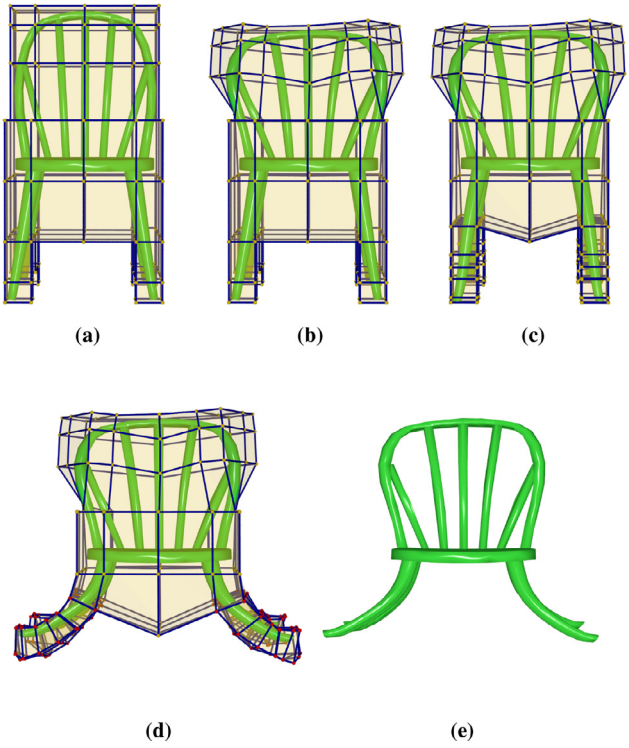


Fig. 14. "Chair" model: global and local deformations.

run on an x64-based PC with Intel Core i9-9900K CPU 3.6 GHz and 64G memory. Our minimization model has a few parameters such as tradeoff parameters and the iteration number. In our experiments, we heuristically set iteration $= 5$ and $w_{1F} = 1.5w_M = 2w_V = w_C$. An accompanying video is also submitted to show how the proposed proxy-driven deformation works.

4.1. Proxy-driven deformation

An example is given in Fig. 10, where we compare the proposed proxy-driven deformation with Autodesk® 3DS Max®'s FFD. The initial setup for both methods is a control lattice of dimensions $12 \times 12 \times 12$, which give 1728 control points. With the proposed method, the proxy is formed by 628 points. It can be seen that the proxy gives much clean UI and thus facilitates the specification of the deformation.

Figs. 11 and 12 compare the proxy with its underlying T-spline control lattice when the FFDs are applied to the "Rabbit" and "Lamp" models. It is apparent that the proxy provides a much clear view than the volumetric lattice. With the proxy, selecting and moving control points are also made simpler.

4.2. Local and global deformation

The T-spline volume naturally support both local and global deformations. Fig. 13 demonstrates local deformation, in which some parts of the object are deformed by manipulating nearby control points and other parts remain unchanged.

The example in Fig. 14 exhibits both local and global deformations using the proxy with both geometry and topology manipulations.

More examples are provided in Figs. 15–17 where both geometric and topological editing is applied to the proxy, which drives the free-form deformation. It can be seen that with the proxy, various deformation processes can be easily and flexibly done. For instance, for the "bottle" model in Fig. 16, the body part could be deformed into a round shape globally and its pump part could be elongated and pressed down by inserting more control points in the local area and then deforming them.

4.3. Performance analysis

The initialization, parameterization and geometric manipulation of our method are linear problems and can be done quickly. The relatively time-consuming step is local refinement and octree partitioning refinement after parameterization. Hence the performance of our topological refinement is related to the number of additional control points to be inserted, which relies on the complexity of T-volume's topology and is independent of the complexity of the embedded object. The overall running time for all the examples in the paper is roughly $0.39 \sim 1.6$ seconds.

In addition, the proxy uses fewer control points than the control lattice, which substantially cleans the interface. Table 1 reports the statistics of the models used in the experiments, the corresponding numbers of control points and faces on the proxy in the deformation, and the running time (in seconds).

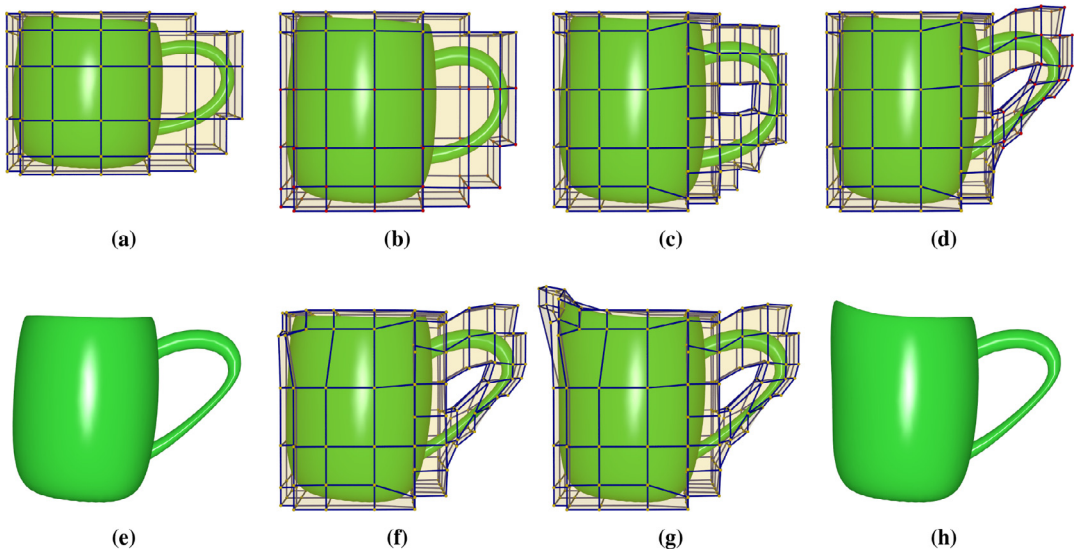


Fig. 15. Various deformations on the "Cup" model.

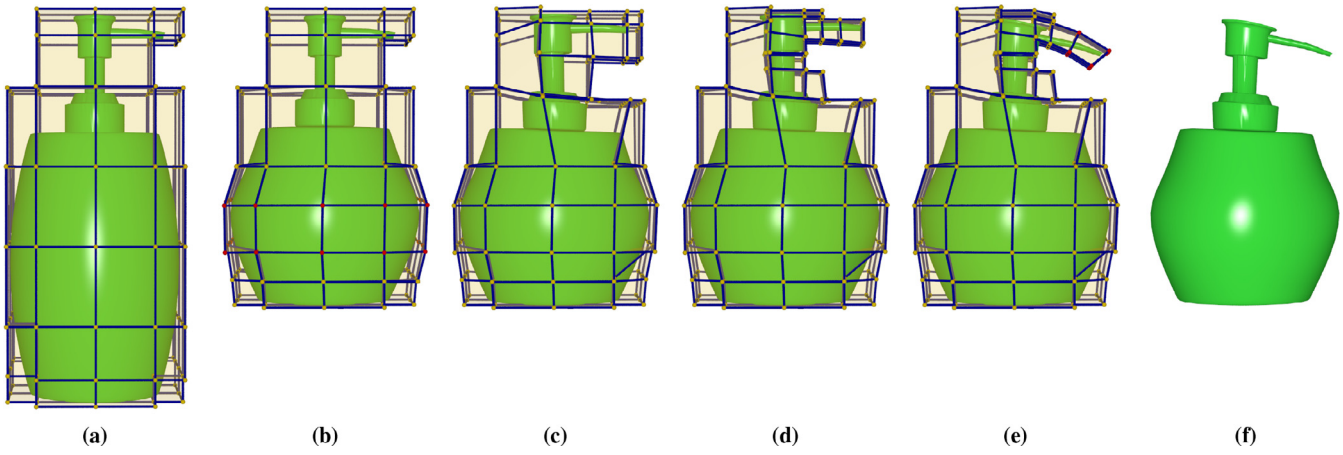


Fig. 16. Various deformations on the "Bottle" model.

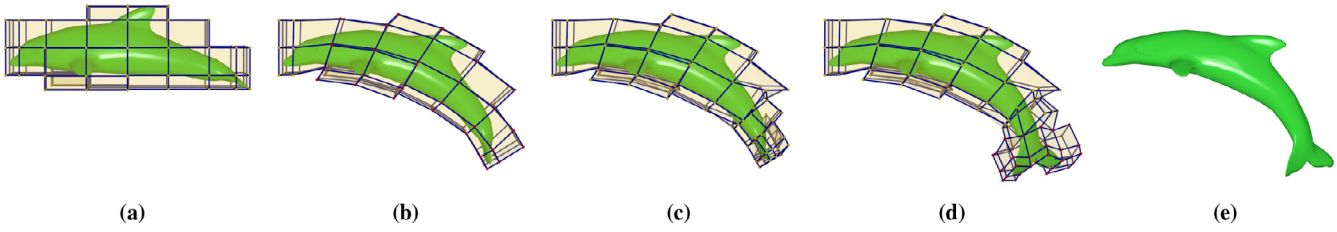


Fig. 17. Various deformations on the "Dolpin" model.

Table 1
Statistics of the examples in experiments and running time (in seconds) (# v: number of vertices on object or proxy; # f: number of faces on proxy; # cp: number of control points on lattices.).

Model (# v)	Initial Lattices (# cp)	Proxy			Refined Lattices (# cp)	Updated Proxy			Running time
		Fig.#	# v	# f		Fig.#	# v	# f	
Rabbit (5510)	$6\times7\times5$ (210)	Fig 11a	138	136	465	Fig 11c	244	266	1.376
Lamp (4347)	$5\times8\times6$ (240)	Fig 12a	118	116	326	Fig 12c	164	168	0.988
Face (2975)	$7\times5\times6$ (210)	Fig 13a	146	146	449	Fig 13c	306	345	1.571
Chair (2528)	$5\times8\times7$ (280)	Fig 14a	170	168	336	Fig 14c	200	202	0.49
Cup (4270)	$8\times6\times6$ (288)	Fig 15a	200	200	540	Fig 15c	252	256	1.134
Bottle (2635)	$7\times8\times6$ (336)	Fig 16a	180	178	579	Fig 16d	182	190	0.392
Dolphin (2069)	$5\times5\times9$ (225)	Fig 17a	112	110	292	Fig 17d	151	152	0.616

5. Conclusions

We have described a flexible free-form deformation technique. It has several nice features: (1) it provides a cage-like interface for the user to perform free-form deformation, which avoids manipulating too many control points and achieves ease to use; (2) it allows topology change of the control lattice during the deformation process, which is not possible for most FFD methods or cage-base deformation as well; and (3) it supports continuous, interactive deformation processes. These features overcome the two major limitations of the conventional FFD methods.

The proposed method also has limitations. First, since the method is intrinsically a volumetric lattice based approach, it is less flexible to achieve large deformation than cage-based deformation though the user can handle the cage-like proxy mesh for deformation. Second, if we start with a very coarse mesh, it will take time to perform T-spline volume local refinement algorithm to create a dense proxy. Third, our method does not provide limits on the deformation or guarantees on the as-rigid-as-possible minimization procedure to make sure that the computed deformation is valid. One way is to post-check whether the deformation degenerates or not once it is computed. A better way is to develop schemes to pre-compute the limits for a sound deformation, which warrants further investigation.

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.

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests.

CRediT authorship contribution statement

Yuzhe Zhang: Methodology, Software, Validation, Writing - original draft. **Jianmin Zheng:** Conceptualization, Methodology, Supervision, Writing - review & editing. **Yiyu Cai:** Investigation, Resources, Writing - review & editing.

Acknowledgments

This work was supported by the Ministry of Education, Singapore, under its MoE Tier-2 Grant (MoE 2017-T2-1- 076).

References

[1] Sederberg TW, Parry SR. Free-form deformation of solid geometric models. SIGGRAPH Comput Graph 1986;20:151–60. doi:10.1145/15886.15903.
[2] Coquillart S. Extended free-form deformation: a sculpturing tool for 3d geometric modeling. SIGGRAPH Comput Graph 1990;24:187–96. doi:10.1145/97880.97900.
[3] Lamousin HJ, Waggenspack Jr WN. NURBS-based free-form deformations. IEEE Comput Graph Appl 1994;14(6):59–65. doi:10.1145/237170.237247.
[4] MacCracken R, Joy KI. Free-form deformations with lattices of arbitrary topology. In: Proceedings of the 23rd annual conference on computer graphics and interactive techniques. New York, NY, USA: ACM; 1996. p. 181–8. ISBN 0-89791-746-4. doi:10.1145/237170.237247.
[5] Ono Y, Chen B-Y, Nishita T, Feng J. Free-form deformation with automatically generated multiresolution lattices. In: Proceedings of the first international symposium on cyberWorlds (CW'02). Washington, DC, USA: IEEE Computer Society; 2002.
[6] Ju T, Schaefer S, Warren J. Mean value coordinates for closed triangular meshes. ACM Trans Graph 2005;24(3):561–6. doi:10.1145/1073204.1073229.
[7] Joshi P, Meyer M, DeRose T, Green B, Sanocki T. Harmonic coordinates for character articulation. ACM Trans Graph 2007;26(3). doi:10.1145/1276377.1276466.
[8] Noble RA, Clapworthy GJ. Direct manipulation of surfaces using Nurbs-based free-form deformations. In: International conference on information visualisation. Los Alamitos, CA, USA: IEEE Computer Society; 1999. http://doi.ieeecomputersociety.org/10.1109/IV.1999.781565
[9] Gain J, Bechmann D. A survey of spatial deformation from a user-centered perspective. ACM Trans Graph 2008;27(4):107:1–107:21.
[10] Song W, Yang X. Free-form deformation with weighted t-spline. Vis Comput 2005;21(3):139–51.
[11] Cohen-Or D. Space deformations, surface deformations and the opportunities in-between. J Comput Sci Technol 2009;24(1):2–5.
[12] Lipman Y, Sorkine O, Cohen-Or D, Levin D, Rossi C, Seidel HP. Differential coordinates for interactive mesh editing. In: Proceedings of shape modeling applications; 2004. p. 181–90.
[13] Sorkine O, Cohen-Or D, Lipman Y, Alexa M, Rössl C, Seidel H-P. Laplacian surface editing. In: Proceedings of the 2004 eurographics/ACM SIGGRAPH symposium on geometry processing. New York, NY, USA: ACM; 2004. p. 175–84. http://doi.acm.org/10.1145/1057432.1057456. ISBN 3-905673-13-4.
[14] Yu Y, Zhou K, Xu D, Shi X, Bao H, Guo B, et al. Mesh editing with poisson-based gradient field manipulation. ACM Trans Graph 2004;23(3):644–51. doi:10.1145/1015706.1015774.
[15] Lipman Y, Sorkine O, Levin D, Cohen-Or D. Linear rotation-invariant coordinates for meshes. ACM Trans Graph 2005;24(3):479–87. doi:10.1145/1073204.1073217.
[16] Sorkine O, Alexa M. As-rigid-as-possible surface modeling. In: Proceedings of the Fifth eurographics symposium on geometry processing. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association; 2007. p. 109–16. http://dl.acm.org/citation.cfm?id=1281991.1282006. ISBN 978-3-905673-46-3.
[17] Barr A. Global and local deformations of solid primitives. SIGGRAPH Comput Graph 1984;18(3):21–30.
[18] Griessmair J, Purgathofer W. Deformation of solids with trivariate b-splines. In: FRA Hopgood WS, editor. Proceedings of eurographics; 1989.
[19] Reis J, Kosinka J. Injective hierarchical free-form deformations using THB-splines. Comput-Aided Des 2018;100:30–8. URL http://www.sciencedirect.com/science/article/pii/S0010448518300782.
[20] Bechmann D, Bertrand Y, Thery S. Continuous free form deformation. Comput Netw ISDN Syst 1997;29(14):1715–25.
[21] Hsu WM, Hughes JF, Kaufman H. Direct manipulation of free-form deformations. SIGGRAPH Comput Graph 1992;26(2):177–84. doi:10.1145/142920.134036.
[22] Hu S-M, Zhang H, Tai C-L, Sun J-G. Direct manipulation of FFD: efficient explicit solutions and decomposible multiple point constraints.. Vis Comput 2001;17(6):370–9.
[23] Floater MS. Mean value coordinates. Comput Aided Geom Des 2003;20(1):19–27. doi:10.1016/S0167-8396(02)00002-5.
[24] Hormann K, Floater MS. Mean value coordinates for arbitrary planar polygons. ACM Trans Graph 2006;25(4):1424–41. doi:10.1145/1183287.1183295.
[25] Lipman Y, Kopf J, Cohen-Or D, Levin D. GPU-assisted positive mean value coordinates for mesh deformations. In: Proceedings of the fifth eurographics symposium on geometry processing. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association; 2007. p. 117–23. http://dl.acm.org/citation.cfm?id=1281991.1282007. ISBN 978-3-905673-46-3.
[26] Lipman Y, Levin D, Cohen-Or D. Green coordinates. ACM Trans Graph 2008;27(3):78:1–78:10. doi:10.1145/1360612.1360677.
[27] Huang J, Chen L, Liu X, Bao H. Efficient mesh deformation using tetrahedron control mesh. In: Proceedings of the 2008 ACM symposium on solid and physical modeling. New York, NY, USA: ACM; 2008. p. 241–7. http://doi.acm.org/10.1145/1364901.1364935. ISBN 978-1-60558-106-4.
[28] Ben-Chen M, Weber O, Gotsman C. Variational harmonic maps for space deformation. ACM Trans Graph 2009;28(3).
[29] Hopkins K. Example-Based parameterization of linear blend skinning for skinning decomposition. Michigan State University. Computer Science; 2017. ISBN 9780355405200. URL https://books.google.com.sg/books?id=AaXEtQEACAAJ.
[30] Hua J, Qin H. Free-form deformations via sketching and manipulating scalar fields. In: Proceedings of the Eighth ACM Symposium on Solid Modeling and Applications; 2003. p. 328–33.
[31] Guo X, Hua J, Qin H. Scalar-function-driven editing on point set surfaces. IEEE Comput Graph Appl 2004;24(4):43–52.
[32] Jacobson A, Baran I, Popović J, Sorkine O. Bounded biharmonic weights for real-time deformation. ACM Trans Graph 2011;30(4):78:1–78:8. doi:10.1145/2010324.1964973.
[33] Cerveró M, Vinacua À, Brunet P. 3D model deformations with arbitrary control points. Comput Graph 2016;57:92–101. doi:10.1016/j.cag.2016.03.010.
[34] Sederberg TW, Zheng J, Bakenov A, Nasri A. T-splines and T-NURCCs. ACM Trans Graph 2003;22:477–84. doi:10.1145/882262.882295.
[35] Sederberg TW, Cardon DL, Finnigan GT, North NS, Zheng J, Lyche T. T-spline simplification and local refinement. ACM Trans Graph 2004;23:276–83. doi:10.1145/1015706.1015715.
[36] Escobar J, Cascn J, Rodrguez E, Montenegro R. A new approach to solid modeling with trivariate T-splines based on mesh optimization. Comput Methods Appl Mech Eng 2011;200(45–46):3210–22. URL http://www.sciencedirect.com/science/article/pii/S0045782511002386.
[37] Zhang Y, Wang W, Hughes TJ. Solid T-spline construction from boundary representations for genus-zero geometry. Comput Methods Appl Mech Eng 2012;249–252:185–97. Higher Order Finite Element and Isogeometric Methods. URL http://www.sciencedirect.com/science/article/pii/S0045782512000254.

[38] Wang W, Zhang Y, Liu L, Hughes TJR. Trivariate solid T-spline construction from boundary triangulations with arbitrary genus topology. Comput Aided Des 2013;45(2):351–60. doi:10.1016/j.cad.2012.10.018.
[39] Zhang Y, Wang W, Hughes TJR. Conformal solid t-spline construction from boundary T-spline representations. Comput Mech 2013;51(6):1051–9. doi:10.1007/s00466-012-0787-6.
[40] Zhou K, Huang J, Snyder J, Liu X, Bao H, Guo B, et al. Large mesh deformation using the volumetric graph Laplacian. ACM Trans Graph 2005;24(3):496–503.
[41] Qin X, Wu T, Liu Y. A surface deformation method based on stiffness control. J Adv Mech Des SystManuf 2020;14(1).