S^3-Slicer: A General Slicing Framework for Multi-Axis 3D Printing

TIANYU ZHANG*, The University of Manchester, United Kingdom
GUOXIN FANG*, The University of Manchester, United Kingdom
YUMING HUANG, The University of Manchester, United Kingdom
NEELOTPAL DUTTA, The University of Manchester, United Kingdom
SYLVAIN LEFEBVRE, Université de Lorraine, CNRS, INRIA, France
ZEKAI MURAT KILIC, The University of Manchester, United Kingdom
CHARLIE C. L. WANG†, The University of Manchester, United Kingdom

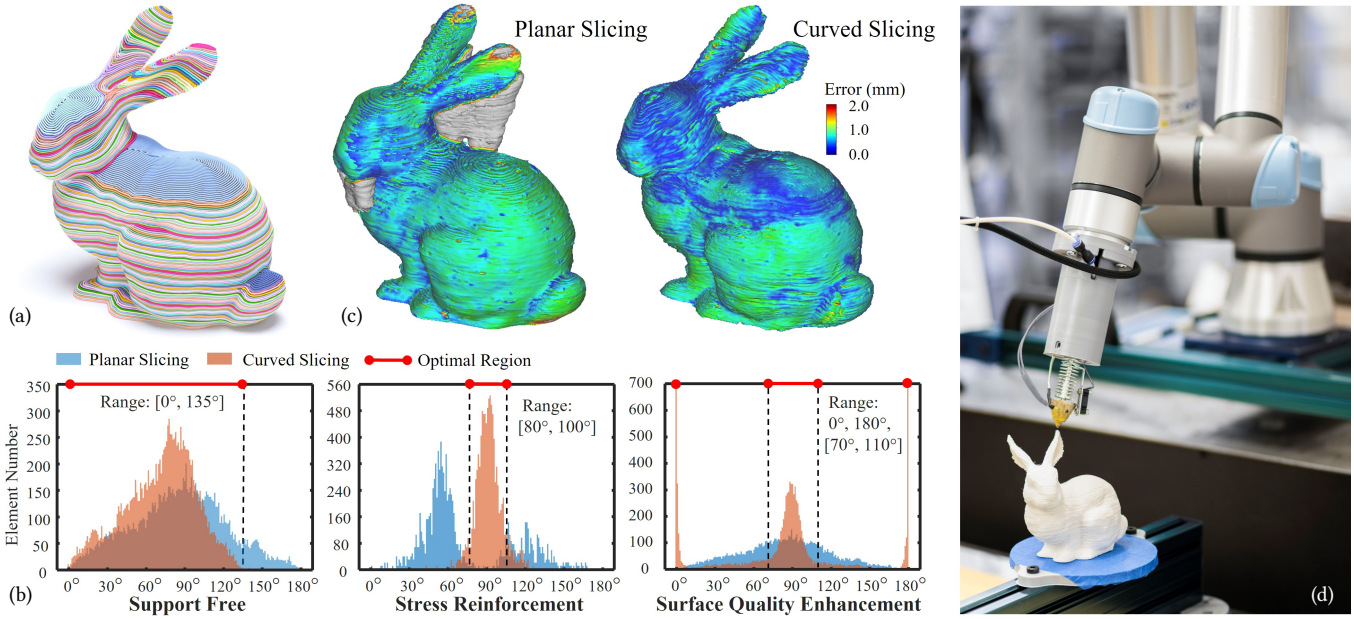


Fig. 1. We present a slicing framework to generate curved layers for multi-axis 3D printing. (a) Toolpaths of a Bunny model that simultaneously satisfy the fabrication objectives of *support free* (SF), *strength reinforcement* (SR) and *surface quality* (SQ), which makes the name of our approach as S^3-slicer. (b) All the objectives of fabrication are formulated as the requirements of local printing directions (LPDs) – the ranges displayed in red color, and our quaternion-based deformation optimization can effectively generate curved layers satisfying these requirements. (c) Comparison of the models 3D printed by conventional planar layers (left) vs. our optimized layers in curved surfaces (right), where the surface errors evaluated by a structured-light based 3D scanner are shown as color maps. (d) Multi-axis 3D printing of curved layers is realized on the hardware equipped with a UR5e robotic arm.

*Joint first authors.
†Corresponding author: changling.wang@manchester.ac.uk (Charlie C. L. Wang)

Authors' addresses: Tianyu Zhang, The University of Manchester, Manchester, United Kingdom, tianyu.zhang-10@postgrad.manchester.ac.uk; Guoxin Fang, The University of Manchester, Manchester, United Kingdom, guoxin.fang@manchester.ac.uk; Yuming Huang, The University of Manchester, Manchester, United Kingdom, yuming.huang@postgrad.manchester.ac.uk; Neelotpal Dutta, The University of Manchester, Manchester, United Kingdom, neelotpal.dutta@postgrad.manchester.ac.uk; Zekai Murat Kilic, Manchester, The University of Manchester, United Kingdom, zekaimurat.kilic@manchester.ac.uk; Sylvain Lefebvre, Université de Lorraine, CNRS, INRIA, France, sylvain.lefebvre@inria.fr; Charlie C. L. Wang, The University of Manchester, Manchester, United Kingdom, changling.wang@manchester.ac.uk.

Multi-axis motion introduces more degrees of freedom into the process of 3D printing to enable different objectives of fabrication by accumulating materials layers upon curved layers. An existing challenge is how to effectively generate the curved layers satisfying multiple objectives simultaneously. This paper presents a general slicing framework for achieving multiple fabrication objectives including support free, strength reinforcement and surface quality. These objectives are formulated as local printing directions varied in the volume of a solid, which are achieved by computing the rotation-driven deformation for the input model. The height field of a deformed model is mapped into a scalar field on its original shape, the isosurfaces of which give the curved layers of multi-axis 3D printing. The deformation can be effectively optimized with the help of quaternion fields to achieve the fabrication objectives. The effectiveness of our method has been verified on a variety of models.

CCS Concepts: • **Computing methodologies** → **Shape modeling**; **Mesh geometry models**.

Additional Key Words and Phrases: support-free, strength-reinforcement, surface-quality, multi-axis motion, 3D printing

ACM Reference Format:
Tianyu Zhang, Guoxin Fang, Yuming Huang, Neelotpal Dutta, Sylvain Lefebvre, Zekai Murat Kilic, and Charlie C. L. Wang. 2022. S^3-Slicer: A General Slicing Framework for Multi-Axis 3D Printing. *ACM Trans. Graph.* 41, 6, Article 277 (December 2022), 15 pages. https://doi.org/10.1145/3550454.3555516

1 INTRODUCTION

Multi-axis 3D printing has caught a lot of attention in recent years. The additional degrees-of-freedom can lead to a variety of advantages compared with the planar layer based 3D printing, including the reduction of supporting structures [Mitropoulou et al. 2022], the enhanced mechanical strength [Tam and Mueller 2017], and the improved surface quality [Ahlers et al. 2019]. Although approaches have been developed to decompose the volume of a given model into curved layers for optimizing different objectives, curved slicing to achieve multiple objectives is still an unsolved problem. Existing approaches [Dai et al. 2018; Etienne et al. 2019; Fang et al. 2020] are specifically designed for one particular objective and they are difficult to be modified to achieve other objectives.

This paper introduces a general slicing framework to tackle this challenge. We consider the three objectives including *support free* (SF), *strength reinforcement* (SR) and *surface quality* (SQ). These can be satisfied on the same model independently or simultaneously. Moreover, we have also considered the manufacturing constraints including the local collision (i.e., gouging) and the layer thickness in the same framework. An example is shown in Fig.1, where the Bunny model can be fabricated in a support free manner on a multi-axis 3D printing hardware equipped with a UR5e robotic arm. The breaking force of the Bunny model fabricated by our curved layers is 25.3% higher than that of the model fabricated by planar layers. The average surface error is significantly reduced 28.2% from 0.632mm to 0.454mm. We name our approach S^3-slicer since all three objectives SF, SQ and SR can be achieved simultaneously.

1.1 Method and Contributions

We argue that all the SF, SR and SQ requirements can be formulated as objectives in terms of *local printing directions* (LPDs), where the LPD indicates the ideal local orientation for material accumulation in the printing process. Every model to be fabricated is represented as a volumetric mesh in our framework. Ideally, we need to compute the working surfaces as layers satisfying the requirements of LPDs in all elements. Optimizing for LPDs under multiple objectives is difficult as angular constraints may disagree with each other. In addition, directly optimizing orientation by vectors is not stable as blending constraints-disagreed vectors may result in vanished vectors. A better approach is needed.

We proposed a rotation-driven deformation framework to solve this issue. All the relevant elements are rotated into orientations satisfying the fabrication objectives when printing layers along the height direction (i.e., $\mathbf{z}$-axis as LPD). After that, the height field of a deformed model is mapped into a scalar field on the input model. We use the isosurfaces of the scalar field as the curved layers of multi-axis 3D printing. We decouple the deformation problem into the sub-problems of 1) computing optimized and compatible quaternions as rotations on all elements and 2) computing a scale-controlled deformation driven by the optimized rotations – both can be efficiently obtained by a local/global solver. Rotations on elements are determined as an optimized quaternion field. The SF, SR and SQ objectives as well as the constraint to avoid local collision are all defined as constraints of quaternions. The manufacturing constraint of layer thickness is defined as scales to be controlled in the rotation-driven deformation. This newly proposed framework can simultaneously achieve SF, SR and SQ objectives and effectively overcomes the problem of vanishing LPDs (i.e., vectors). As a byproduct, optimizing quaternions on elements gives a linear system with the dimension only 1/3 of that of the deformation computed on vertices – therefore, it can be computed more efficiently.

Our technical contributions are summarized as follows:

- A slicing framework to generate curved layers for multi-axis 3D printing that can simultaneously satisfy multiple fabrication objectives.
- A quaternion-based formulation to effectively achieve the SF, SR and SQ objectives while preventing the local collision in multi-axis 3D printing.
- A rotation-deformation decoupled paradigm to efficiently determine the rotation-driven deformation.

To verify the performance of curved layers generated by our slicing framework, we physically fabricated prototypes on a robotic setup of multi-axis 3D printing. The surface quality is evaluated with the help of a structured-light based 3D scanner, and the mechanical strength is tested on a universal testing machine.

1.2 Related Work

1.2.1 Multi-axis 3D printing. Traditional additive manufacturing devices are limited to three-axis movement with most slicers producing 2.5D toolpaths. While this simplification results in an easy-to-implement software solution, it causes the problems of 3D printed parts in both the weak mechanical strength [Ahn et al. 2002] and the poor surface quality [Chakraborty et al. 2008]. Moreover, supporting structures are needed below the large overhangs, which leads to the problems of hard-to-remove, surface damage, and material waste [Zhang et al. 2015].

Previous research has shown that multi-axis 3D printing can help to achieve support-less or even support-free material accumulation [Dai et al. 2018; Huang et al. 2016; Mitropoulou et al. 2020; Wu et al. 2016], enhance the mechanical strength of printed parts [Fang et al. 2020; Tam and Mueller 2017], and reduce the staircase effect [Etienne et al. 2019], where many works were physically realized on robot-assisted 3D printing hardware (ref. [Bhatt et al. 2021; Li et al. 2022]). All these existing methods only work for a specific objective. For example, Dai et al. [2018] decomposed the volume into curved layers with support-free consideration, where the algorithm is based on local progressive optimization but not a global one. Etienne et al. [2019] proposed deformation-based slightly curved printing for 3-axis machines, which however does not exploit additional degrees of freedom of multi-axis machines. Fang et al. [2020] provided a pipeline to take advantage of the anisotropy of mechanical properties introduced by the process of fused filament fabrication.

However, there is ambiguity in the vector field calculation of their method requiring special treatments to ensure conformity. Existing approaches only considered a single objective for the generation of curved layers and toolpaths. There is no direct way to change existing approaches' formulation to achieve other objectives – e.g., making a support-free approach to preserve surface quality. It is more challenging for them to meet multi-objectives simultaneously, which can be realized by our framework.

1.2.2 Field generation / optimization. As discussed above, the problem to achieve all the SF, SR and SQ objectives can be formulated as requirements defined on LPDs, which are actually vectors. One may consider to extend the existing techniques that process vector fields on triangular mesh surfaces [de Goes et al. 2016] to volumetric meshes. However, as discussed in [Fang et al. 2020], directly smoothing vectors pointing to significantly disagreed directions is not an effective method to obtain a compatible vector field. Another option is to formulate the problem into frame-based field optimization (e.g., [Huang et al. 2011; Li et al. 2012; Ray et al. 2016]). Arora et al. [2019] proposed an approach to generate volumetric Michell trusses by non-linear optimization of frames, which however is very time-consuming. Differently, we propose a simple and efficient method to achieve the SF, SR and SQ objectives in different combinations.

1.2.3 Deformation. Our framework employs a strategy of rotation-driven deformation to obtain the deformed shape of a model to satisfy the LPD requirements, which borrows the ideas of local/global optimization [Bouaziz et al. 2012; Liu et al. 2008] and *as-rigid-as-possible* (ARAP) deformation [Sorkine and Alexa 2007; Sorkine et al. 2004]. The final results of deformation are obtained by first conducting rotations separately on each element in a local project step and then stitching all the elements together in a global blending step, where the optimization problem can be solved by computing least squares solution in iterations. The computation of local/global solver converges very fast when the rotations applied to neighboring elements are nearly compatible – i.e., with small difference. However, this is not naturally satisfied in the problem to be solved in this paper. Therefore, a quaternion-field optimization step is introduced as the inner loop of our framework to obtain nearly compatible rotations, which is also solved by using the local/global optimization strategy.

1.2.4 Volume parameterization. In our approach, the height field on the deformed model is mapped to a scalar field defined on the input model. This shares some similarity to the problem of volume parameterization [Patane et al. 2013], which are widely used for all-hex mesh generation [Livesu et al. 2013], isogeometric analysis [Aigner et al. 2009] and biomedical application [Xu et al. 2013]. The problem to compute a deformed model for SF, SR and SQ objectives can be considered as a special volume parameterization problem with orientation constraints. We need to achieve multiple objectives related to the LPDs while considering the manufacturing constraints such as local collision and layer thickness. A new formulation is proposed in this paper.

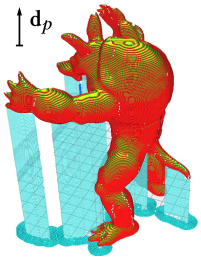
Our work focuses on computing deformation optimized for multiple fabrication objectives, which leads to a general curved slicing framework for multi-axis 3D printing. The existing techniques for optimizing volume parameterization such as resolving foldovers to ensure the local injectivity (ref. [Du et al. 2020; Fu et al. 2015; Garanzha et al. 2021; Kovalsky et al. 2015; Liao et al. 2021; Schüller et al. 2013; Su et al. 2019])) can be applied to improve the quality of the mapping determined by our approach. For example, the injectivity was achieved by further optimizing the ARAP-based deformation in [Rabinovich et al. 2017]. A comprehensive survey of inversion-free mapping can be found in [Fu et al. 2021].

2 OVERVIEW

Our framework represents every input model as a tetrahedral mesh $\mathcal{M}$, where the boundary surface $\partial\mathcal{M}$ is formed by a set of triangle faces and the boundary region $\mathcal{B}$ is defined as a set of tetrahedral elements that have faces on $\partial\mathcal{M}$. This section first presents the objectives (Sec. 2.1) and the manufacturing constraints (Sec. 2.2) of multi-axis 3D printing that need to be considered when generating curved layers for an input model. After that, the overall algorithm is introduced in Sec. 2.3.

2.1 Objectives for Multi-Axis 3D Printing

Three fabrication objectives are considered in our framework – all defined according to a LPD (denoted by $\mathbf{d}_p$). Instead of directly computing a distribution of LPDs in the given model $\mathcal{M}$, we compute a deformed model $\mathcal{M}^d$ by rotating the elements in $\mathcal{M}$ to satisfy the fabrication objectives in the deformed space with a fixed $\mathbf{d}_p = (0, 0, 1)$. Equivalently, the mapping from $\mathcal{M}^d$ to $\mathcal{M}$ converts $\mathbf{d}_p = (0, 0, 1)$ into spatially changed LPDs on $\mathcal{M}$.



2.1.1 Support free (SF). As shown in the wrapped figure on the right, supporting structures (shown in blue) are required in planar layer based 3D printing to enable the fabrication of overhangs. The support free criterion is defined on every face $f \in \partial\mathcal{M}$ by its surface normal vector $\mathbf{n}_f$ as (ref. [Hu et al. 2015])

$$\mathbf{n}_f \cdot \mathbf{d}_p + \sin(\alpha) \geq 0, \tag{1}$$

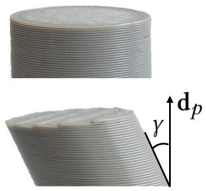
where α is self-supporting angle that depends on the material properties and $\mathbf{d}_p$ is the vector of printing direction. Both $\mathbf{n}_f$ and $\mathbf{d}_p$ are unit vectors. When being applied to multi-axis 3D printing, the printing direction $\mathbf{d}_p$ is allowed to change in different regions. This SF objective is realized in our framework by rotating every boundary element $e \in \mathcal{B}$ into an orientation such that the above condition is satisfied on the deformed model $\mathcal{M}^d$ of $\mathcal{M}$ when $\mathbf{d}_p = (0, 0, 1)$.

2.1.2 Strength reinforcement (SR). Anisotropic mechanical behaviour is observed on models fabricated by filament-based 3D printing due to the weak adhesion between deposited filaments. Under a given load scenario, the mechanical strength of printed models can be significantly improved when the filament is aligned with the stress field direction [Riddick et al. 2016; Tam and Mueller 2017]. In other words, the 3D printed model can have a reinforced strength when

the LPD is nearly perpendicular to the direction of maximum principal stress $\tau_{\max}(e)$ in an element e. The SR criterion regarding the printing direction $\mathbf{d}_p$ can be defined as

$$|\mathbf{d}_p \cdot \tau_{max}(e)| \leq \sin(\beta), \tag{2}$$

where $\tau_{\max}(e)$ is a unit vector, β is a parameter to allow a certain level of tolerance, and $\beta = 10^\circ$ is employed in our experimental tests. This SR objective is realized in our framework by rotating every element e in critical regions[1] into an orientation so that Eq.(2) is satisfied on the deformed model $\mathcal{M}^d$ when $\mathbf{d}_p = (0, 0, 1)$.



2.1.3 Surface quality (SQ). The staircase effect is the main reason for the surface quality issues in 3D printing. Different from the approaches using slices with varied thicknesses (e.g., [Mao et al. 2019; Wang et al. 2015]), multi-axis 3D printing research has been conducted to 1) use a boundary conformal layer to realize a large smooth region [Ahlers et al. 2019], or 2) generate curved layers by forming boundary surfaces nearly 'vertical' to the printing direction [Etienne et al. 2019] – see the wrapped figure above for an illustration. These observations lead to two criteria for achieving the SQ objective defined on a boundary face $f \in \partial\mathcal{M}$ as:

$$\mathbf{n}_f = \pm\mathbf{d}_p \tag{3}$$

$$|\mathbf{n}_f \cdot \mathbf{d}_p| \leq \sin(\gamma) \tag{4}$$

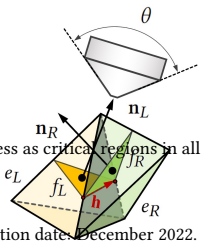
where γ controls the maximally allowed 'cliff-angle' for surface quality. A smaller value of γ will lead to better surface quality with fewer stair-cases. The criterion defined in Eq.(3) is to impose boundary 'conformal' layers (denoted by SQ-C) and the one in Eq.(4) is for enabling boundary 'vertical' layers (denoted by SQ-V).

2.2 Manufacturing Constraints

In addition to the user-defined objectives (SF, SQ, SR), manufacturing constraints such as collision prevention and layer-thickness bounds have to be enforced.

2.2.1 Local collision-free. As more degrees-of-freedom are available to drive the motion of the 3D printing process, collisions can happen locally if the curved layers as working surfaces are not properly designed. Specifically, the geometry of a printer head forms a cone with the apex angle θ. When using this cone to contact a point on the working surface, any concave region with a dihedral angle less than θ leads to an unavoidable local collision between the working surface and the printer head. This definition is adapted from the multi-axis CNC machining in which gouging (i.e., local collision) is a commonly encountered issue (e.g., [Bartoň et al. 2021]).

The local collision-free constraint is defined by the dihedral angle between two neighboring faces on a curved layer, which is an isosurface extracted from the tetrahedral mesh $\mathcal{M}$ as a piece-wise linear polygonal mesh. For two neighboring elements e_L and e_R containing the polygonal faces f_L and f_R as isosurface, the local collision is unavoidable when



$$\begin{cases} -\sin(\theta) \leq (\mathbf{n}_L \times \mathbf{n}_R) \cdot \mathbf{h} < 0 & (\theta < \frac{\pi}{2}) \\ (\mathbf{n}_L \times \mathbf{n}_R) \cdot \mathbf{h} \leq -\sin(\theta) & (\theta \geq \frac{\pi}{2}) \end{cases} \tag{5}$$

with $\mathbf{n}_L$ and $\mathbf{n}_R$ being the unit normal of f_L and f_R and $\mathbf{h}$ being a unit vector for the edge shared by f_L (as its left) and f_R (as its right). Local collision is prevented in our framework by controlling the rotations applied to face-neighbored elements (details can be found in Sec. 3.3.2). Global collision is checked and resolved during the motion planning process of the physical realization (ref. [Ezair et al. 2018; Zhang et al. 2021a]), which is not the focus of this paper.

2.2.2 Layer thickness. Because of hardware limitations (e.g., the speed of extrusion and the material properties during the phase transition between solid and liquid), the layer thickness must be controlled within a certain range $[d_{\min}, d_{\max}]$ which are commonly determined by the hardware design and the materials employed in 3D printing. In our curved slicing framework, the requirement of layer thickness is formulated as a soft constraint to be imposed by computing a field of compatible rotations (Sec. 3.2) and a scale-harmonic deformation (Sec. 4.1). After that, the requirement on layer thickness can be strictly preserved by using the adaptive (partial) slicing algorithm (Sec. 4.2) although only one example needs this post-processing step among all the examples tested in our work.

2.3 Optimized Deformation for Slicing

To achieve multiple fabrication objectives in multi-axis 3D printing, we introduce a deformation-based computational paradigm which generates curved layers efficiently. As illustrated in Fig. 2, an input model is represented as a tetrahedral mesh $\mathcal{M}$ and the curved layers are generated in the elements $e \in \mathcal{M}$. In order to compute strength-reinforced toolpaths, the stress tensors according to specified loading need to be generated by *Finite Element Analysis* (FEA). The stress tensors work as input to our framework where the maximum principal stress $\tau_{\max}$ is defined in every element.

The main idea of our framework is to compute a deformed model of $\mathcal{M}$ as $\mathcal{M}^d$ that optimizes (multi-)objectives with LDP being $\mathbf{d}_p = (0, 0, 1)$ for $\mathcal{M}^d$. This deformation is driven by computing compatible rotations (represented by a field of quaternions $\mathbf{q}_e$ defined on $e \in \mathcal{M}$) so that the (multi-)objectives are satisfied locally when pure rotations are applied to elements separately (see Fig. 2(b) for an illustration). Computing the optimized quaternion field $\mathcal{Q} = \{\mathbf{q}_e\}$ is the inner-loop optimization of our framework, and the outer-loop optimization is to blend the rotations $\mathbf{R}(\mathbf{q}_e)$ on all elements as a scale-controlled deformation. The overall algorithm of our framework is as follows:

(1) The input model $\mathcal{M}$ of our algorithm is a tetrahedral mesh (see Fig. 2(a)), which is also used as the current model $\mathcal{M}^d$ in the first iteration of computation.

[1] We employ the region with top 30% of the tensile stress as critical regions in all our examples. Users can define different regions as critical.

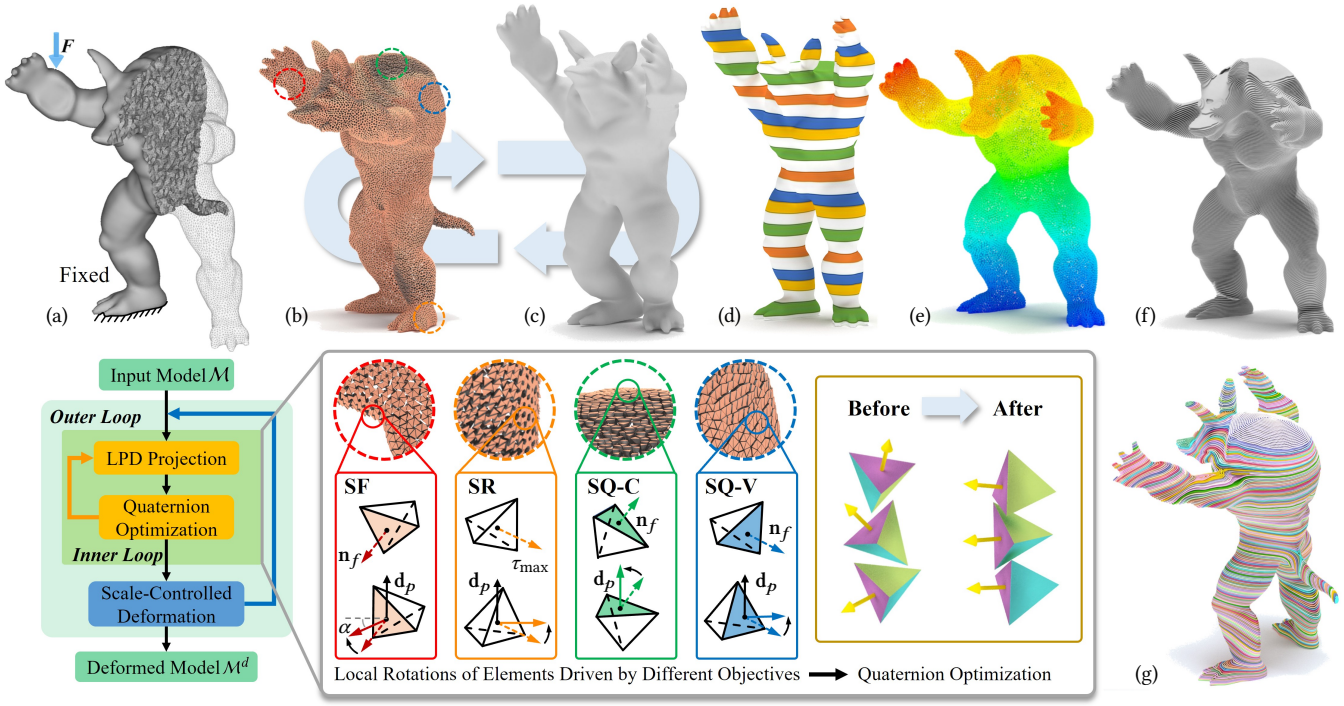


Fig. 2. An overview of our slicing framework for generating curved layers that satisfy multiple objectives of multi-axis 3D printing. (a) The input armadillo model is represented by a tetrahedral mesh $\mathcal{M}$ with principal stresses obtained from FEA. (b, c) A rotation-driven optimization is computed by first rotating elements to locally satisfy the SF, SQ and SR objectives and then globally assembling the rotated elements to obtain a deformed model. These *local / global* steps are run in iteration to obtain the deformed model $\mathcal{M}^d$, where the compatible rotations applied to elements are computed by optimizing a quaternion-field $\mathcal{Q}$ as the inner loop of optimization. (d, e) The height field of $\mathcal{M}^d$ is mapped to a scalar field $G(\cdot)$ defined on $\mathcal{M}$. (f) The isosurfaces of $G(\cdot)$ are extracted to work as curved layers. (g) Toolpaths are generated on curved layers for 3D printing.

(2) We compute the rotation applied to every element e by deforming from $\mathcal{M}$ to $\mathcal{M}^d$ and representing the rotations by the quaternions which form a quaternion field $\mathcal{Q} = \{\mathbf{q}_e\}$.
(3) An inner-loop optimization problem is solved to determine a new quaternion-field $\mathcal{Q}^* = \{\mathbf{q}_e^*\}$ that a) minimizes the difference between quaternions in neighboring elements, and b) satisfies the specific (multi-)objectives locally (i.e., SF, SQ-C, SQ-V and SR). A local/global solver is employed to efficiently compute the optimized field $\mathcal{Q}^*$ (see Fig. 2(b) and the illustration in the second row of Fig. 2). Details are presented in Sec. 3.
(4) The rotations $\{\mathbf{R}(\mathbf{q}_e^*)\}$ determined for all element $e \in \mathcal{M}$ are assembled together to form a global shape $\mathcal{M}^d$ by minimizing an objective function that can effectively control the scale and therefore the variation of layer thickness on the slicing results (Sec. 4.1). An example of this deformation for assembly can be found in Fig. 2(c).
(5) The height field of $\mathcal{M}^d$ defined as piecewise linear functions in tetrahedral elements is mapped into a scalar field $G(\cdot)$ defined on $\mathcal{M}$ (see Fig. 2(d, e) for an example).
(6) The algorithm goes back to (2) or stops iterating when the fabrication objectives are achieved on G with the terminal condition defined by using the gradient ∇G as LPDs (Sec. 4.1).
(7) The curved layers are generated by extracting the isosurfaces of $G(\cdot)$. We check the distance between layers and apply the adaptive slicing algorithm when needed (Sec. 4.2).
(8) The hybrid strategy of contour-parallel and stress-reinforced directional-parallel is employed to generate the toolpaths on each curved layer (see Fig. 2(f, g)).

Considering G is a piecewise linear function with field values defined on the vertices of $\mathcal{M}$, the gradient ∇G gives the surface normal (i.e., LPD) of curved layers which are constant inside each element $e \in \mathcal{M}$. Therefore, we define the terminal condition by using ∇G.

3 QUATERNION-BASED ROTATION OPTIMIZATION

We formulate the inner loop of our framework as a constrained optimization problem to determine the compatible rotations on all elements of $\mathcal{M}$ as a quaternion field $\mathcal{Q} = \{\mathbf{q}_e\}$. The SF, SQ and SR objectives are defined as constraints denoted by C_{SF}, C_{SQ} and C_{SR}, where C_{SF}, C_{SQ} are defined on the elements in the boundary region $\mathcal{B}$, and C_{SR} is defined on a set of critical elements $\mathcal{S} \subset \mathcal{M}$. The

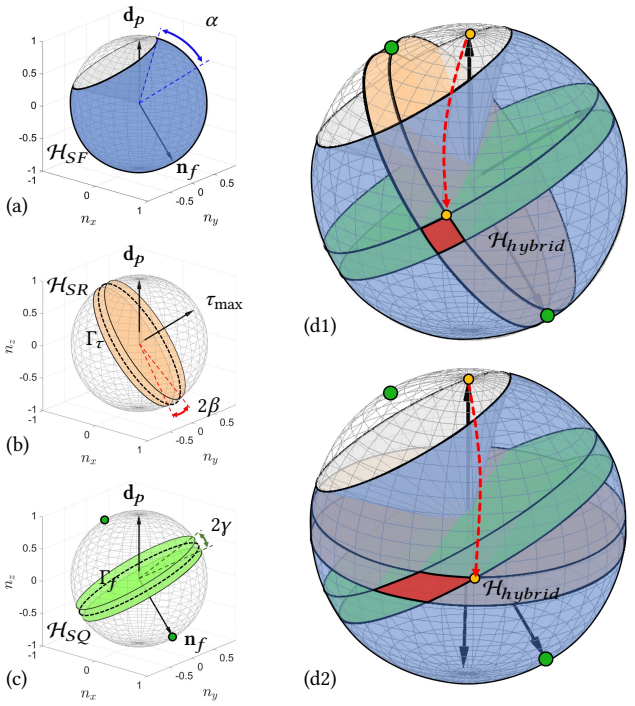


Fig. 3. Illustrations of feasible regions defined for (a) support-free $\mathcal{H}_{SF}$, (b) strength-reinforcement $\mathcal{H}_{SR}$ and (c) surface-quality $\mathcal{H}_{SQ}$. (d1, d2) Feasible regions for multi-objectives are defined as $\mathcal{H}_{hybrid} = \mathcal{H}_{SF} \cap \mathcal{H}_{SR} \cap \mathcal{H}_{SQ}$. The minimal rotation for moving $\mathbf{d}_p$ into a feasible region can be determined by searching its closest point on the boundary of feasible regions – indicated by the red dashed arrows.

optimization problem is formulated as

$$\begin{aligned} \arg\min_{\{\mathbf{q}_e\}} & \sum_{(e_i,e_j)\in\mathcal{N}_F} w_s(e_i,e_j)\|\mathbf{q}_{e_i}-\mathbf{q}_{e_j}\|^2 \\ s.t.\ & C_{SR}(\mathbf{q}_e) \qquad\qquad (\forall e\in\mathcal{S}) \\ & C_{SF}(\mathbf{q}_e),\ C_{SQ}(\mathbf{q}_e) \quad (\forall e\in\mathcal{B}) \end{aligned} \tag{6}$$

where $\mathcal{N}_F$ is the set of pairs of elements that are face-neighbors. One of the major difficulties to solve this constrained optimization problem is that the initial guesses of $\mathcal{Q} = \{\mathbf{q}_e\}$ usually do not fall in the feasible region defined by the constraints. A local/global solver is employed to compute the optimized $\mathcal{Q}$ effectively (Sec. 3.2). The constraints of quaternions are defined by analyzing the feasible regions of LPDs (Sec. 3.1), and the weights $w_s(\cdot)$ will be employed to control the concavity to prevent local collision (Sec. 3.3.2).

3.1 Objectives as Feasible Regions of Quaternions

The analysis of feasible regions for LPDs (thus also quaternions) is conducted on the Gauss sphere. As we will discuss later, a common feasible region always exists. According to the formulas given in Sec. 2.1, every fabrication objective indeed defines a feasible region of LPDs for each element e on the Gauss sphere. Specifically, we need to figure out a rotation $\mathbf{q}_e$ applied to e to ensure that the LPD of e on the deformed model (i.e., $\mathbf{d}_p = (0,0,1)$) falls in every feasible region. This can also be formulated as a requirement on the inversely rotated $\mathbf{d}_p$ as $\mathbf{d} = \mathbf{R}^{-1}(\mathbf{q}_e)\mathbf{d}_p$ be within the following half-spaces:

$$\begin{aligned} \mathcal{H}_{SF} &= \{\mathbf{d} \mid \forall \mathbf{d}\in\mathbb{S}^2,\ \mathbf{d}\cdot\mathbf{n}_f + \sin(\alpha) \geq 0\} \\ \mathcal{H}_{SR} &= \{\mathbf{d} \mid \forall \mathbf{d}\in\mathbb{S}^2,\ |\mathbf{d}\cdot\tau_{\max}| \leq \sin(\beta)\} \\ \mathcal{H}_{SQ} &= \{\mathbf{d} \mid \forall \mathbf{d}\in\mathbb{S}^2,\ \mathbf{d} = \pm\mathbf{n}_f \text{ or } |\mathbf{d}\cdot\mathbf{n}_f| \leq \sin(\gamma)\} \end{aligned} \tag{7}$$

Given $\mathbf{d}_p$ on the deformed model $\mathcal{M}^d$, the feasible regions of C_{SF}, C_{SR} and C_{SQ} should be the sets of $\mathbf{q}_e$s that let

$$\mathbf{R}^{-1}(\mathbf{q}_e)\mathbf{d}_p \in \mathcal{H}_{SF}, \mathbf{R}^{-1}(\mathbf{q}_e)\mathbf{d}_p \in \mathcal{H}_{SR}, \mathbf{R}^{-1}(\mathbf{q}_e)\mathbf{d}_p \in \mathcal{H}_{SQ}$$

be satisfied respectively. Note that the constraints of $\mathbf{q}_e$ are nonlinear as $\mathbf{R}^{-1}(\cdot)$ is involved (ref. [Voight 2021]).

The feasible regions $\mathcal{H}_{SF}$, $\mathcal{H}_{SQ}$ and $\mathcal{H}_{SR}$ defined on Gauss sphere are illustrated in Fig.3. For an element applied with multiple objectives, the feasible region is the intersection of three half-spaces as $\mathcal{H}_{hybrid} = \mathcal{H}_{SF} \cap \mathcal{H}_{SR} \cap \mathcal{H}_{SQ}$ that is the red region in Fig. 3(d). Two different configurations are given. It is worthy to mention that the relative position between $\mathcal{H}_{SF}$ and $\mathcal{H}_{SQ}$ is invariant as they are both defined on the normal of boundary face $\mathbf{n}_f$. Defining the big circle on the Gauss sphere perpendicular to $\mathbf{n}_f$ as Γ_f (see Fig.3(c)), we can have $\Gamma_f \subset \mathcal{H}_{SF} \cap \mathcal{H}_{SQ}$ ($\because \alpha, \gamma > 0$). Similarly, we can define the big circle perpendicular to $\tau_{\max}$ as Γ_τ (see Fig.3(b)) and have $\Gamma_\tau \subset \mathcal{H}_{SR}$ ($\because \beta > 0$). It is easy to see that $\Gamma_f \cap \Gamma_\tau \neq \emptyset$. Therefore, we can conclude that $\mathcal{H}_{hybrid} \neq \emptyset$, which guarantees that the feasible solution can always be found for a single element e.

3.2 Constrained Optimization for Quaternion Field

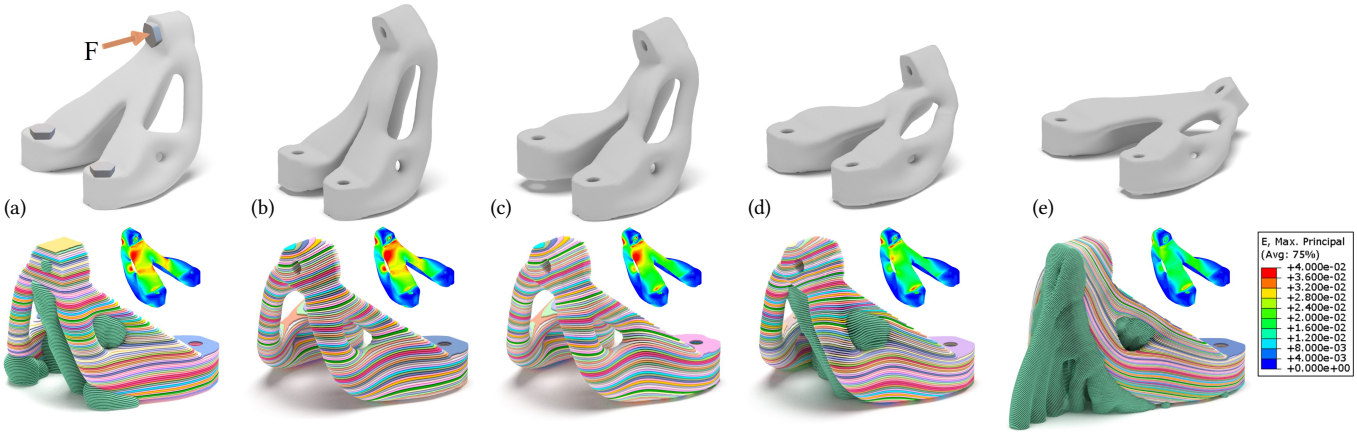
As an optimization problem with nonlinear constraints, Eq.(6) is not easy to be solved effectively – especially when the initial guesses of $\{\mathbf{q}_e\}$ are not in the feasible regions. The problem becomes more challenging when multiple objectives are applied; it would become very difficult to naturally obtain feasible quaternions that are also *compatible* to the neighbors – i.e., we seek for a smooth quaternion field. We employ a local/global solver to compute the compatible quaternions that satisfy the fabrication objectives.

For the local projection step, we need to determine a minimal rotation that moves $\mathbf{d}_p$ into $\mathcal{H}_{hybrid}$. This can be solved by searching $\mathbf{d}_p$'s closest point $\mathbf{d}_c$ on the boundary of $\mathcal{H}_{hybrid}$ (see the red dashed arrows in Fig. 3(d)). A feasible quaternion $\mathbf{q}_e^t$ according to this minimal rotation is determined by $\mathbf{d}_p = \mathbf{R}(\mathbf{q}_e^t)\mathbf{d}_c$.

Once feasible quaternions $\mathbf{q}_e^t$ are found locally for all elements in $\mathcal{S}$ and $\mathcal{B}$, a global blending step is applied to compute the updated quaternion field $\mathcal{G} = \{\mathbf{q}_e\}$ that is 1) smooth (i.e., compatible between neighboring quaternions) and 2) not varied too much from the locally determined feasible quaternions. This can be achieved by solving the following problem:

$$\arg\min_{\{\mathbf{q}_e\}} \sum_{(e_i,e_j)\in\mathcal{N}_F} w_s(e_i,e_j)\|\mathbf{q}_{e_i}-\mathbf{q}_{e_j}\|^2 + \sum_{e\in\mathcal{S}\cup\mathcal{B}} w_k(e)\|\mathbf{q}_e-\mathbf{q}_e^t\|^2, \tag{8}$$

where the first term is to impose the objective of compatible quaternions and the second term is employed to preserve the feasible results that are obtained from the local projection step. Thanks to its least squares form Eq.(8) can be efficiently solved. Note that the hard constraints presented in Eq.(6) are actually converted into soft constraints in Eq.(8) when the local/global solver is employed.



User Preference		(a) Planar Slicing	(b) $w_{SF} : w_{SR} = 1.0 : 0.0$	(c) $w_{SF} : w_{SR} = 0.7 : 0.3$	(d) $w_{SF} : w_{SR} = 0.3 : 0.7$	(e) $w_{SF} : w_{SR} = 0.0 : 1.0$
Strain	Average	3.06e-3	2.94e-3	2.53e-3	2.16e-3	2.01e-3
	Maximal	3.52e-2	3.76e-2	3.50e-2	3.01e-2	2.28e-2
Support Volume (%)		72.4%	0.0% (No Support)	0.0% (No Support)	61.6%	173.9%

Fig. 4. Comparison of results with different SF and SR preferences by adjusting the weights w_{SF} and w_{SR} for achieving multiple objectives. $w_{SQ} = 0.0$ is used for all these tests, and the volume of the supporting structure is given as the percentage of the given model's volume. An interesting observation is that only considering SR objective will lead to a slicing result that needs more supporting structures for fabrication.

3.3 Weight Adaptation

The strategy for changing the weights in our optimization framework is discussed below, where the weight $w_k(e)$ is employed to reflect the user's preference on different objectives. The concavity-aware weighting strategy is introduced to $w_s(e)$ for preventing local collision.

3.3.1 Objective preference. As already discussed in Sec. 3.1, the feasible solution satisfying all constraints exists locally for all elements $e \in \mathcal{S} \cap \mathcal{B}$. However, the feasible solution in neighboring elements may still lead to incompatible quaternions. Two quaternions are considered incompatible if the L^2-norm of their difference is larger than a threshold (e.g., 0.12 is used in our tests). A weighting scheme is introduced here to reflect the user's preference for the different objectives by changing the values of $w_k(e)$. Specifically, we define

$$w_k(e) = \max(w_{SF}U_{SF}(e), w_{SQ}U_{SQ}(e), w_{SR}U_{SR}(e)) \tag{9}$$

with the following switch functions defined on the deformed model $\mathcal{M}^d$. The values of these switch functions will be updated after each step of the global deformation in the outer loop of optimization.

$$U_{SF}(e) = \begin{cases} 1, & e \in \mathcal{B},\ \mathbf{n}_f^d(e) \cdot \mathbf{d}_p + \sin(\alpha) < 0 \\ 0, & \text{otherwise} \end{cases}$$

$$U_{SR}(e) = \begin{cases} 1, & e \in \mathcal{S} \\ 0, & \text{otherwise} \end{cases}$$

$$U_{SQ}(e) = \begin{cases} 1, & e \in \mathcal{B},\ \mathbf{n}_f^d(e) \neq \pm\mathbf{d}_p \ \&\ |\mathbf{n}_f^d(e) \cdot \mathbf{d}_p| > \sin(\gamma) \\ 0, & \text{otherwise} \end{cases}$$

Here $\mathbf{d}_p = (0, 0, 1)$ is fixed and $\mathbf{n}_f^d(e)$ gives the normal of the boundary face $f \in \partial\mathcal{M}^d$ on the tetrahedral element e.

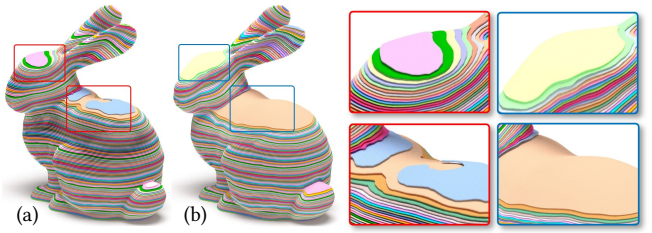


Fig. 5. Different results of the bunny model for SQ-C objective by using different values for w_k in the last step of iteration: (a) $w_k = 1.0$ and (b) $w_k = 10.0$. The boundary conformal layer is enhanced in (b). Note that $w_k = 1.0$ is employed in other steps of the iteration.

The values of w_{SL}, w_{SR} and w_{SQ} are in the $[0, 1]$ range and are defined by users. An example of how w_k influences the slicing results of curved layers is demonstrated in Fig. 4. The hybrid objectives of SF and SR are required for this model that is generated by topology optimization. As can be seen, the curved layers generated by our framework can be 3D printed in a completely support-free way manner when only the SF constraint is applied (e.g., $w_{SF} : w_{SR} = 1.0 : 0.0$). When the weights are set as $w_{SF} : w_{SR} = 0.7 : 0.3$, we can achieve a result with a good balance between the SF and SR objectives – i.e., the printed model shows an enhanced strength compared to the result with the SF objective only (see data in the table of Fig. 4). The SQ objective is left out of these tests by setting $w_{SQ} = 0.0$.

The feasible region of the SQ objective, $\mathcal{H}_{SQ}$, contains two points (for SQ-C) and a band region (for SQ-V). Our optimization pipeline will automatically project $\mathbf{d}_p$ to the closet solution in a feasible region. Therefore, whether it will be an SQ-C or SQ-V case is selected

automatically. For those elements defined as SQ-C in the last step of iteration, we assign a very large w_k to ensure $\mathbf{q}_e = \mathbf{q}_e^t$. This is very important for preserving the surface quality of a boundary conformal layer in a large area (see the Bunny model in Fig. 5 for an example).

3.3.2 Local collision and concavity control. A uniform weight as $w_s(e) = 1.0$ is assigned to all elements in the first iteration of our optimization framework for generally controlling the compatibility (i.e., smoothness) of neighboring quaternions. The values of $w_s(e)$ on different elements are changed in the later steps of iteration to prevent local collisions (i.e., gouging) between the printer head and the curved layers as working surfaces (see Fig.6 for an example). The concavity-aware weighting scheme applied here is based on the analysis of local collision conducted in Sec. 2.2.

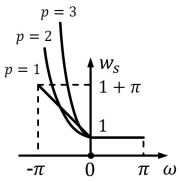
After obtaining a deformed model $\mathcal{M}^d$ and mapping its height field into a scalar-field $G(\cdot)$ on $\mathcal{M}$, we can evaluate the local concavity of curved layers in two neighboring tetrahedral elements e_L and e_R. Using the isovalue g_c as the average at the center of the face shared by e_L and e_R, we obtain an isosurface $G(\mathbf{x}) = g_c$ in these two elements as two polygons with normal vectors $\mathbf{n}_L$ and $\mathbf{n}_R$. At the same time, we can obtain the edge vector $\mathbf{h}$ on the face shared by e_L and e_R. Whether the dihedral angle formed as the edge $\mathbf{h}$ is concave can be evaluated by the sign of $(\mathbf{n}_L \times \mathbf{n}_R) \cdot \mathbf{h}$ – i.e., '+' for convex and '-' for concave edge. We define the signed angle between $\mathbf{n}_L$ and $\mathbf{n}_R$ as

$$\omega(e_L, e_R) = \text{sign}\left((\mathbf{n}_L \times \mathbf{n}_R) \cdot \mathbf{h}\right) \arccos(\mathbf{n}_L \cdot \mathbf{n}_R) \tag{10}$$

In our framework, we check if there are cases of local collisions by using the condition defined in Eq.(5). When a collision happens, the concavity-aware adaptive weights defined below are employed to 'flatten' concave regions by applying a large weight for smoothness.

$$w_s(e_L, e_R) = \begin{cases} 1.0 + |\omega(e_L, e_R)|^p & (\omega(e_L, e_R) < 0) \\ 1.0 & (\omega(e_L, e_R) \geq 0) \end{cases} \tag{11}$$

with $p = 1$. The curves of $w_s(e_L, e_R)$ by using different p are shown in the right.



As shown in Fig.6, this adaptive weighting scheme can effectively remove local collisions. Note that this will slightly modify the rotation directly determined by fabrication objectives because a larger smoothness term is imposed in highly concave regions. When cases of local collisions are still found after applying the weights defined in Eq.(11), we further 'flatten' the local concave region by incrementally enlarging the value of p – although it has never happened in our experiments. As shown by the curves of w_s in the wrapped image, using a larger p will impose stronger smoothness requirement in the highly concave regions.

4 CURVED LAYERS GENERATION

This section introduces the scale-controlled deformation scheme which forms the outer loop of our framework. After that, the adaptive slicing algorithm that ensures the required range of layer thicknesses is presented.

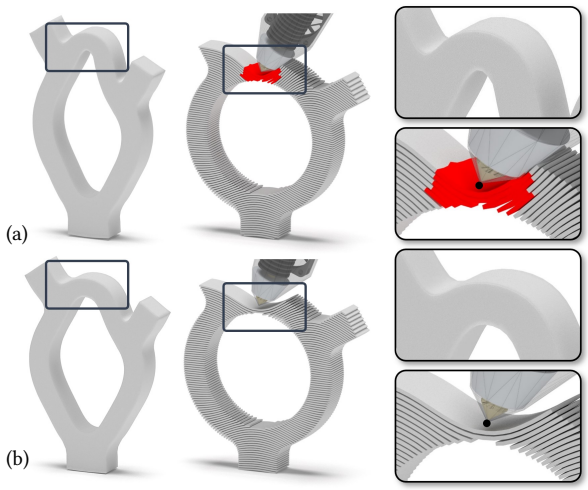


Fig. 6. The adaptive scheme to assign the values of w_s can effectively remove local collisions generated by using uniform weights – see the result shown in (a). Collision-free results (b) are obtained by the concavity-aware adaptive weights.

4.1 Scale-Controlled Deformation

After computing the quaternion field $\mathcal{G}(\cdot) = \{\mathbf{q}_e\}$ on the input model $\mathcal{M}$, we realize the desired rotation on every element e as $\mathbf{R}(\mathbf{q}_e)(\mathbf{N}\mathbf{V}_e)^{\mathrm{T}}$ with $\mathbf{V}_e$ being a position matrix formed by the coordinates of e's vertices on $\mathcal{M}$. $\mathbf{N}$ is used to transfer an element's centre to the origin as $\mathbf{N} = \mathbf{I}_{4\times4} - \frac{1}{4}\mathbf{1}_{4\times4}$. The rotated elements need to be deformed and stitched together to compute the deformed mesh $\mathcal{M}^d$.

When using the conventional as-rigid-as-possible (ARAP) deformation for this task of stitching, the computation can end up being stuck in a local optimum because of the rigidity requirement. To solve this problem, scaling variables are introduced for each element as $\mathbf{S}_e = \text{diag}(s_e^x, s_e^y, s_e^z)$ – this gives a locally scaled and rotated element as $\mathbf{R}(\mathbf{q}_e)\mathbf{S}_e(\mathbf{N}\mathbf{V}_e)^{\mathrm{T}}$. After controlling the rigidity by the difference between $\mathbf{S}_e$ and $\mathbf{I}$ and controlling the compatibility of scales between neighboring elements, we compute the deformed model $\mathcal{M}^d$ as

$$\begin{aligned} \arg\min_{\mathcal{M}^d} \sum_{e\in\mathcal{M}} \underbrace{\|(\mathbf{N}\mathbf{V}_e^d)^{\mathrm{T}} - \mathbf{R}_e\mathbf{S}_e(\mathbf{N}\mathbf{V}_e)^{\mathrm{T}}\|_F^2}_{\text{Position-Compatibility}} + \\ w_r \sum_{e\in\mathcal{M}} \underbrace{\|\mathbf{S}_e - \mathbf{I}\|_F^2}_{\text{Rigidity}} + w_c \sum_{(e_i,e_j)\in\mathcal{N}_F} \underbrace{\|\mathbf{S}_{e_i} - \mathbf{S}_{e_j}\|_F^2}_{\text{Scale-Compatibility}} \end{aligned} \tag{12}$$

where $\|\cdot\|_F$ is the Frobenius norm, and the positions of vertices in $\mathcal{M}^d$ to be determined are kept in the position matrix $\mathbf{V}_e^d$.

The optimization problem defined in Eq.(12) is highly non-linear, therefore a local/global solution strategy is applied here to compute it efficiently. The quaternion-based optimization (Sec. 3) is employed to determine the rotations as an inner loop step of scaled-deformation. After that, we only consider the scales of elements and the positions of vertices as variables to compute the deformed model $\mathcal{M}^d$. Therefore, the problem in Eq.(12) keeps a least squares form

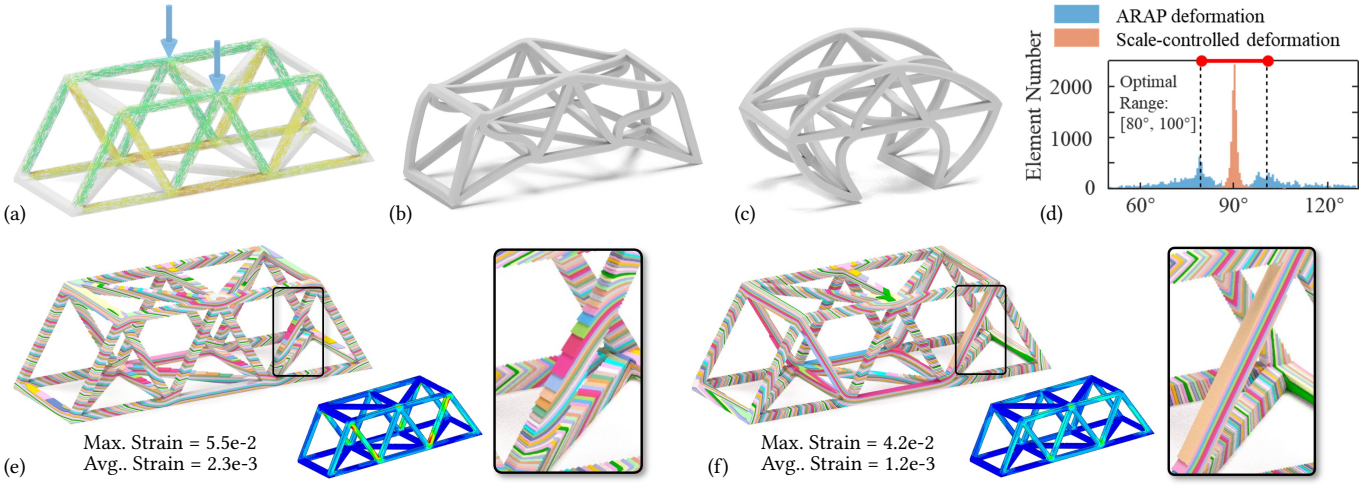


Fig. 7. The comparison of ARAP and our scale-controlled deformation: (a) the stress field under given forces that are shown by arrows, (b, e) the deformed shape obtained by ARAP and its resultant curved layers, (c, f) the shape of $\mathcal{M}^d$ determined by our method ($w_r = 1.0$ and $w_c = 6.0$) and the corresponding curved layers, and (d) the histogram to visualize the results of different deformations in terms of angles between LPDs and maximal principal stresses. It shows that the layers generated by our scale-controlled deformation can meet the requirements of SR w.r.t. the stress field better. FEA is conducted to verify the mechanical strength – i.e., smaller strains are observed on the result of scale-controlled deformation when applying the same load.

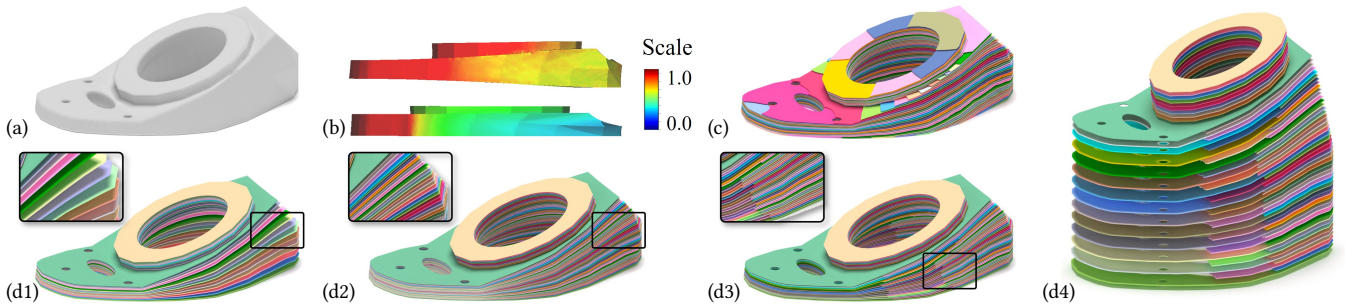


Fig. 8. Comparison of ARAP deformation and our scale-controlled deformation: (a) the input Ankle model, (b) the deformed models after applying ARAP deformation (top) and our scale-controlled deformation (bottom with $w_r = 0.1$, $w_c = 6.0$) where the distribution of scales are visualized by colors, and (c) the slicing result based on ARAP deformation. The process of adaptive slicing are: (d1) initial slicing by considering the required minimal layer thickness $d_{\min}$, (d2) inserting new layers by considering the required maximal layer thickness $d_{\max}$, (d3) partially removing newly inserted layers in the regions where the layer thickness is less than $d_{\min}$, and (d4) the exploded view of the resultant layers.

and can be solved efficiently. Note that this is different from existing as-similar-as-possible approaches which determine the scales in the steps of local projection (ref. [Jiang et al. 2017; Liu et al. 2008]).

4.1.1 Flexibility and control of scales. Adding scales as variables brings in more flexibility for the optimization process to converge. A comparison is given in Fig.7, where our scale-controlled deformation allows for smoother deformation. As a result, curved layers of the bridge model are better aligned with the directions of principal stresses (see the histograms of the directional analysis in Fig.7(d). This also leads to a stronger mechanical strength as can be observed from the results of the FEA.

It is also very important to control the scale-compatibility among neighboring regions because that a radical change in a local region may lead to a large variation in layer thicknesses at the end. The scale-compatibility term plays a very important role in this purpose. A balance is needed between the rigidity term and scale-compatibility term. In most examples conducted in this paper, we choose $w_r = 1.0$ and $w_c = 6.0$ which we determined experimentally. There is only one exceptional case – the anchor example shown in Fig.8, where the rigidity control is released to achieve a complete conformal layer at the top (i.e., $w_r = 0.1$ and $w_c = 6.0$ are employed). The layer thickness issue is then solved by the adaptive slicing algorithm (Sec. 4.2), which generates additional partial layers.

4.1.2 Iteration of deformation. As presented in Sec. 2.3, the scale-controlled deformation needs to run iteratively until the terminal condition is reached. This is mainly because stitching locally rotated elements together and allowing scale change in the deformation step may introduce slight distortion from the rotations determined

in the inner loop. An improved result can be obtained by repeatedly running the outer loop of scale-controlled deformation.

4.1.3 Terminal condition. The following metric is employed to define the terminal condition for the outer loop of our computational framework. Without loss of generality, the feasible region of an element e is denoted as $\mathcal{H}_e$ which can be defined by a single objective (i.e., $\mathcal{H}_{SF}$, $\mathcal{H}_{SQ}$ or $\mathcal{H}_{SR}$) or a combination of them. Given the resultant scalar field $G(\cdot)$ defined on the input model $\mathcal{M}$, the resultant local printing direction is in fact the gradient of $\nabla G(\cdot)$ in each element that is a constant vector. Given a function $D(\mathbf{d}, \mathcal{H})$ that returns the *geodesic* distance between $\mathbf{d}$ and $\mathcal{H}$ on the Gauss sphere, the metric below defines how the objectives are achieved (the smaller the better):

$$\Pi = \sum_{e \in \mathcal{S} \cup \mathcal{B}} D\left(\nabla G(e)/\|\nabla G(e)\|, \mathcal{H}_e\right) \tag{13}$$

The terminal condition for the scale-controlled deformation (i.e., the outer loop of optimization) is chosen as $\frac{|\Pi_k - \Pi_{k-1}|}{\Pi_{k-1}} < 5\%$, where Π_k and Π_{k-1} denote the metric values in the current step and the previous step, respectively.

4.2 Adaptive Slicing

After obtaining the deformed model $\mathcal{M}^d$ by the optimization above, we can obtain the scalar-field $G(\cdot)$ on $\mathcal{M}$ by assigning every node's field-value by its height value on $\mathcal{M}^d$. In each element e, a piecewise linear function is employed to define the field value $G(\mathbf{x})$. Isosurfaces are extracted from $G(\mathbf{x})$ to work as curved layers.

An adaptive slicing algorithm is introduced to ensure the distances between neighboring layers in the range of $[d_{\min}, d_{\max}]$.

- Step 1): Starting from the first layer L_0 as $G(\mathbf{x}) = d_{\min}$, we extract isosurfaces incrementally as L_i by ensuring that the minimal distance between two neighboring layers L_i and L_{i-1} is larger than $d_{\min}$.
- Step 2): For any two neighboring isosurfaces $G(\mathbf{x}) = d_i$ and $G(\mathbf{x}) = d_{i+1}$, we insert a new layer at $G(\mathbf{x}) = \frac{1}{2}(d_i + d_{i+1})$ when the maximal distance between them is larger than $d_{\max}$. This refinement is repeated until the maximal distance bound is guaranteed between all layers.
- Step 3): For a layer L_i as a complete isosurface $G(\mathbf{x}) = d_i$ defined in $\mathcal{M}$, we remove a polygon from L_i if the distance between the polygon and all previous layers L_j $(\forall j < i)$ is less than $d_{\min}$. Again, this checking and removal is only conducted on the layers newly inserted in the second step.

See also Fig.8(d1)-(d4) for an example. In many cases the ranges of thickness $[d_{\min}, d_{\max}]$ are satisfied without having to produce partial layers. Step 2) and 3) are only applied if there are layers with thickness larger than $d_{\max}$. In order to leave more space for optimization, we do not explicitly control the layer thickness in our optimization framework. Therefore, the adaptive slicing algorithm is essential to obtain manufacturable layers. In our implementation, the layer thickness is controlled in the range $[0.4D, 1.0D]$ with D being the diameter of the printer head's nozzle. All layers within this range of thickness can be fabricated by the method of extrusion control in [Etienne et al. 2019].

5 RESULTS AND DISCUSSION

5.1 Implementation details

We implemented our computational framework in C++. Source code of this work will be released after the publication of this paper. The numerical library Eigen [Guennebaud et al. 2010] is employed as the solver of linear equations, and the MKL library [Wang et al. 2014] developed by Intel is used for accelerating the sparse matrix calculation. The PQP library [Gottschalk et al. 1996] is applied to compute the point-to-surface distances for layer thickness evaluation.

The focus of this paper is the generation of curved layers. After that, toolpaths are generated on each curved layers to govern the material deposition of filaments. For the layers passing through the critical regions, the hybrid strategy [Fang et al. 2020] is adopted where the contour-parallel toolpaths and the directional-parallel toolpaths are used for the boundary and the interior regions, respectively. For other layers, the contour-parallel toolpaths are employed. Contour-parallel toolpaths can be generated from the boundary distance field of each curved layer. The directional-parallel toolpaths are generated from the vector-field, which is obtained by projecting the maximal principal stresses onto the surface of each layer. Etienne et al [2019] generated toolpaths on the planar layers and then deformed them back into the original space. A different strategy is conducted in our framework. We directly generate toolpaths on the curved layers as it is easier to control the distances between toolpaths and the distances between sample points on a toolpath.

Each toolpath is sampled into a sequence of waypoints by controlling the distance between neighboring waypoints and using the surface normals as the orientations of the printer head. After that, each waypoint is converted into feasible poses of the robotic arm by a solver of inverse kinematics. The final trajectory of robot motion is computed on the directed graph that uses the poses as the graph's nodes and the collision-free motions between the poses of two neighboring waypoints as the graph's edges. By weighting every edge with the total joint-angle variations between two poses linked by this edge, a smooth trajectory can be determined by searching the shortest path on the graph [Dai et al. 2018; Zhang et al. 2021a]. The feedrate of the extruder is changed according to the layer thickness by using the strategy employed in [Etienne et al. 2019].

5.2 Computational Results

All the computational experiments are conducted on a laptop with an Intel(R) Core(TM) i7-10875H CPU (8 cores @ 2.3GHz) + 32GB RAM, running Windows 10. Models employed in our tests have a variety of topology and the numbers of elements range from 25k to 250k.

5.2.1 Examples and computing time. We tested our approach on a variety of models. The first example is the Bunny model with 196.8k tetrahedral elements that has been shown in Fig. 1. All the SF, SR and SQ objectives are required on this model. Specifically, the surface quality at the feet, the back and the head of bunny is well preserved. The ear region can be 3D printed without supporting structures, and the strength has been improved by 25.3% in the compression test. Besides the physical evaluation, the results of SQ optimization are further verified by a digital evaluation. Each toolpath is converted

Table 1. Computational statistics for the pipeline of S^3-slicer.

Model	Objective	Fig.	Solid #tets	Computing Time of S^3-Slicer (sec.)					Toolpath Gen. (sec.)	Total Time (sec.)
				LDP Proj.	Quaternion Opt.	Scale-controlled Def.	Total of Def.	Slicing		
Bunny	(SF, SR & SQ)	1,9	196.8k	0.7	42.7	12.8	56.2	8.6	10.9	75.7
Armadillo	(SF, SR & SQ)	2	245.4k	1.0	44.5	15.4	60.9	29.9	9.3	100.1
Topo-Opt	(SF & SR)	4	70.5k	0.3	7.4	2.3	10.0	3.3	11.2	24.5
Teapot	(SF & SQ)	11	201.8k	0.8	51.1	15.2	67.1	16.6	31.3	115.0
Ankle	(SF & SQ)	8,12	97.8k	0.4	11.9	5.3	17.6	97.6†	2.9	118.1
Ring	(SF)	6,13	24.8k	0.1	2.8	1.5	4.4	2.4	1.1	7.9
Bridge	(SR)	7	100.4k	0.5	9.8	4.9	15.2	7.1	1.5	23.8

† The adaptive slicing is only applied to a model when necessary. The reported time of Ankle model includes the time for adaptive slicing.

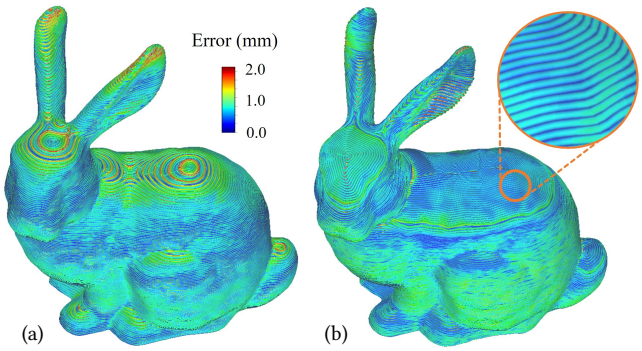


Fig. 9. Digital evaluation of surface quality for the Bunny model fabricated by (a) planar layers ($E_{\text{avg}} = 0.051$; $E_{\max} = 2.01$) vs. (b) curved layers generated by our method ($E_{\text{avg}} = 0.042$; $E_{\max} = 1.39$), where E_{avg} and $E_{\max}$ denote the average and the maximal surface distance errors respectively.

into a tube where the cylindrical cross-section's radius is determined according to the extrusion rate. The surface distances between the input model and the tubes are computed on sample points. Both the average and the maximal distance-errors (denoted by E_{avg} and $E_{\max}$) are reported in Fig.9. The results generated by this digital evaluation are consistent with those obtained from physical experiments.

The second example with all three objectives is the Armadillo model (245.4k tetrahedral elements), as shown in Figs. 2 and 10. In the third example, SF and SR objectives are applied to produce a model that was initially generated by topology optimization – i.e., the Topo-Opt model in Fig. 4. We also tested other two models with SQ and SF objectives: the Teapot in Fig.11 and the Ankle in Fig.12. Lastly, the Ring model (Fig.6) for SF objective and the Bridge model for SR objective (Fig.7) are tested. Computational statistics on these models are given in Table 1. It can be seen that the computation of S^3-slicer can be finished in less than 2 minutes on all examples.

5.2.2 Statistics in LPDs. Histograms are employed to visualize the level of compliance for LPDs generated by our method compared to the required objectives. Specifically, as already shown in Fig.1, the angle between the surface normal ($\mathbf{n}_f$) and the LPD (as $\nabla G(e)$) on boundary elements are visualized through the histograms for SF and SQ objectives. The angles between LPDs (as $\nabla G(e)$) and the directions of maximal principal stress (as $\tau_{\max}$) on all elements in the critical region $\mathcal{S}$ are displayed in the histogram for SR objective. Optimal regions are all indicated on all histograms. We also applied

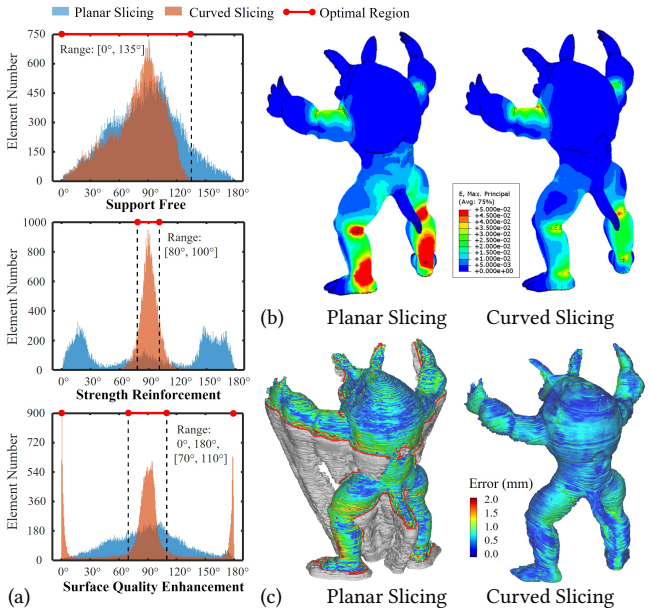


Fig. 10. The result analysis of the Armadillo model according to the toolpaths given in Fig.2: (a) histograms of LPDs, (b) the strain distribution simulated by FEA, and (c) the surface quality analysis visualized by color maps of shape approximation errors (the average errors are 0.738mm and 0.534mm – reduced by 27.6%).

all the SF, SR and SQ objectives to the Armadillo model. As can be observed from the histograms of LPDs shown in Fig.10(a), curved layers are significantly optimized for the SF, SR and SQ objectives. The histogram of SR objective for the Bridge model has been given in Fig.7(d). More histograms of other models can be found in Figs.11-13.

5.2.3 Verification by FEA. To verify the results of SR objectives, we conducted FEA simulation with anisotropic material properties by assigning different Young's moduli along different directions at the element level. Specifically, $Y_1 = 3.5$ GPa is used as the strongest modulus and is assigned to the toolpath's tangential direction. The weakest modulus is assigned both to the surface normal direction of each layer and to the third orthogonal direction as $Y_2 = Y_3 = 1.2$ GPa.

The simulation result of the Bunny model (Fig.14) shows that the curved layers can reduce the maximal strain and the average strain by 27.81% and 18.53%, which is similar to the results obtained by the

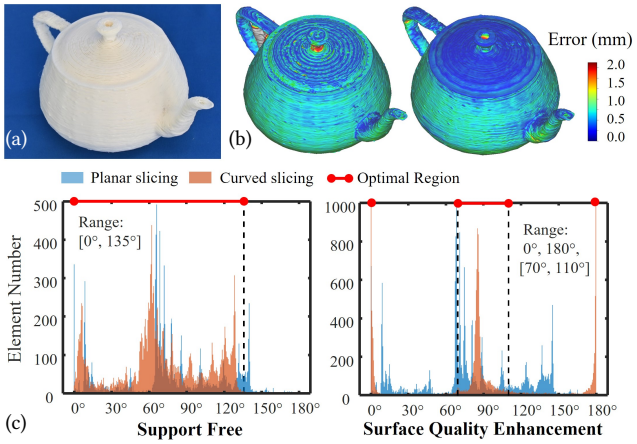


Fig. 11. The result of the Teapot model: (a) The fabrication result, (b) the surface quality analysis visualized by color maps of shape approximation errors (the average errors are 0.537mm and 0.364mm – reduced by 32.2%), and (c) the histograms of LPDs.

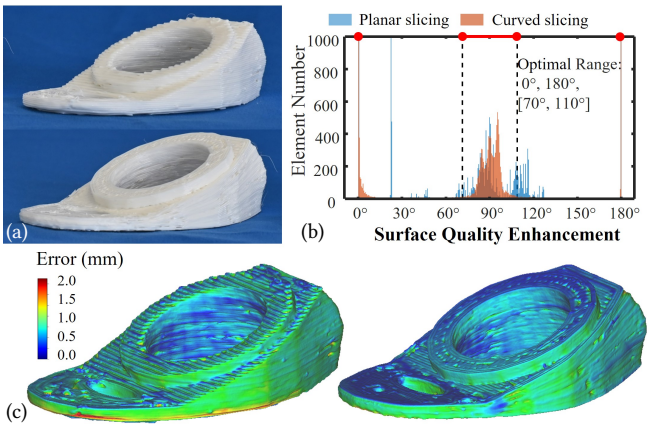


Fig. 12. The result of the Ankle model: (a) The fabrication result, (b) the histogram of LPDs, and (c) the surface quality analysis by the color maps of shape approximation errors (the average errors are 0.580mm and 0.432mm – reduced by 25.5%).

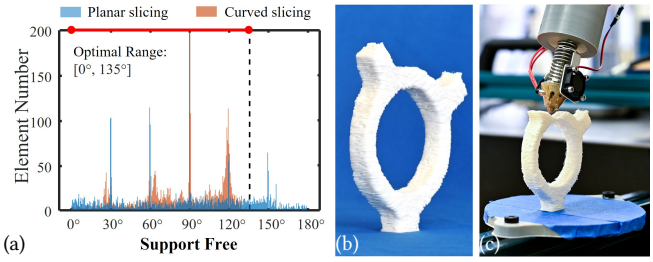


Fig. 13. The result of the Ring model: (a) The histogram of LPDs, (b) the fabrication result, and (c) collision-free fabrication by robotic arm.

physical test (Sec. 5.3.2). The FEA results of the Armadillo model are given in Fig.10(b), where the model with curved layers has reduced 19.4% in the maximal strain and reduced 26.2% in the average strain compared to the model with planar layers. In the third experiment

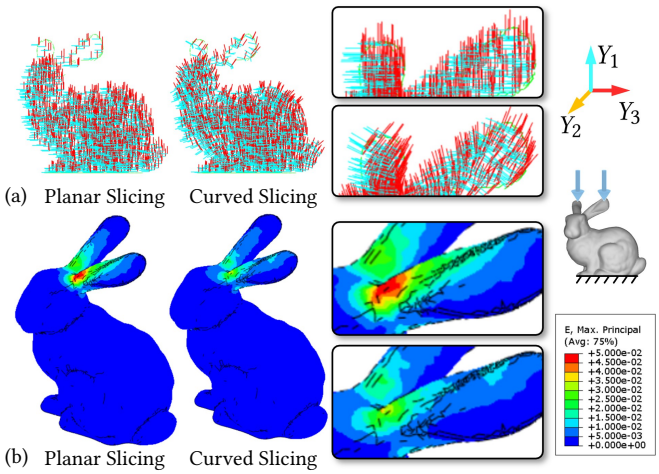


Fig. 14. The simulation results of Bunny model: (a) the frames for anisotropic FEA and (b) the color maps for strain distribution. The curved layers can reduce the maximal strain and the average strain by 27.81% and 18.53%.

Fig. 15. The results of physical fabrication using curved layers generated by our framework.

of simulation, we evaluate the maximal and the average strains for the Topo-Opt model when different user preferences are applied (see Fig.4). It is interesting to find that using a completely support-free slicing (i.e., $w_{SF} : w_{SR} = 1.0 : 0.0$) can make the maximal strain slightly larger than the result of planar slicing when applying the same loading condition. However, the situation can be improved after adding a small weight for SR – e.g., both the average and the maximal strains have been reduced by using $w_{SF} : w_{SR} = 0.7 : 0.3$. Lastly, the FEA results of the Bridge model with different curved layers have been given in Fig.7.

5.3 Physical Experiments

5.3.1 Hardware. We have tested the curved layers and the corresponding toolpaths generated by our framework on a multi-axis 3D printing hardware realized by a UR5e robotic arm, which has the repeatability of ±0.03mm. The extruder control is realized by a Duet3D board. Our printer head has the 1.0mm diameter nozzle. Polylactic acid (PLA) filament with 1.75mm diameter is used in our physical fabrication. As a result, layers with thickness in the range of [0.4mm, 1.0mm] can be reliably produced in our system.

Table 2. Statistics of physical fabrication.

Model	Curved Layers†			Planar Layers†		
	Layer #	Weight	Time	Layer #	Weight‡	Time
Bunny	150	450g	21.3h	151	483g	23.6h
Armadillo	205	183g	19.1h	187	275g	24.4h
Topo-Opt	99	337g	14.8h	100	352g	15.5h
Teapot	150	114g	9.3h	132	127g	8.9h
Ankle	59	52g	2.3h	42	54g	1.8h
Ring	100	56g	2.4h	-	-	-

† The thicknesses for curved layers are in the range of [0.4mm, 1.0mm], and the thickness for planar layers is 0.8mm.
‡ The weight of a model fabricated by planar layers includes the weight of its supporting structures.

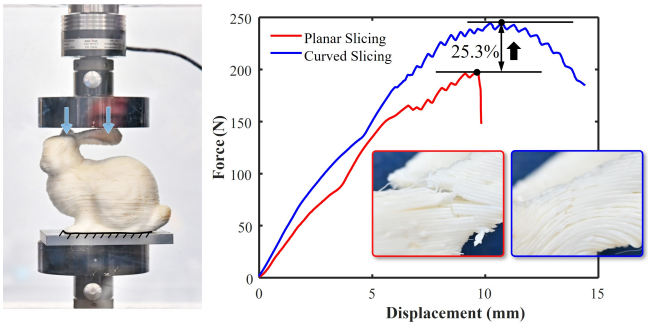


Fig. 16. Results of the Bunny model under compression test, where the pushing forces are applied on the ears and the bottom of Bunny is fixed. Force-displacement curves are generated to study the mechanical strength of models 3D printed by planar layers (red) vs. our curved layers (blue).

5.3.2 Fabrication and mechanical test. All the models fabricated by our system are shown in Fig.15. The statistics of physical fabrication are given in Table 2. Note that because of the additional supporting structures the models fabricated by planar layers are heavier and took longer time. We conduct mechanical compression tests on the Bunny model. Two specimens are employed – one by curved layers generated by our S^3-slicer and the other by conventional planar layers. The force-displacement curves are generated by using a Zwick / Roell tensile machine AGX-X (see Fig.16). From the force-displacement curves of compression tests, we can observe 25.3% improvement in the breaking force, which is consistent with the simulation results as given in Fig.14.

5.3.3 Surface quality study. We have also studied the surface quality on 3D printed models. Point clouds are obtained by a structured-light based scanner EinScan Pro 2X with accuracy of 0.04mm. The *iterative closest point* (ICP) based registration is applied to align the input model with the scanned model, where the distances between these models are generated by the PQP library [Gottschalk et al. 1996] to evaluate the surface approximation error. As already shown in Fig.1, the surface errors on the model fabricated from our curved layers can be significantly reduced. This does not only happen in the region covered by conformal layers (e.g., the top of the Bunny's head and the back) but also other regions achieved by the 'vertical' layers (e.g., the Bunny's mouth / nose regions and the feet region). Similar results can also be observed on the Armadillo model (Fig.10), the Teapot model (Fig.11) and the Ankle model (Fig.12).

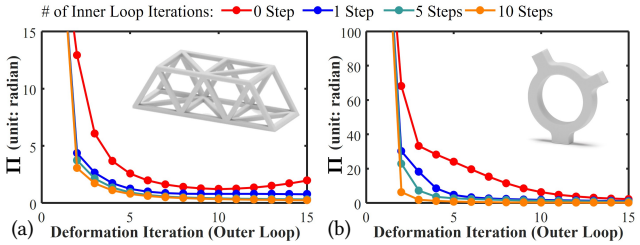
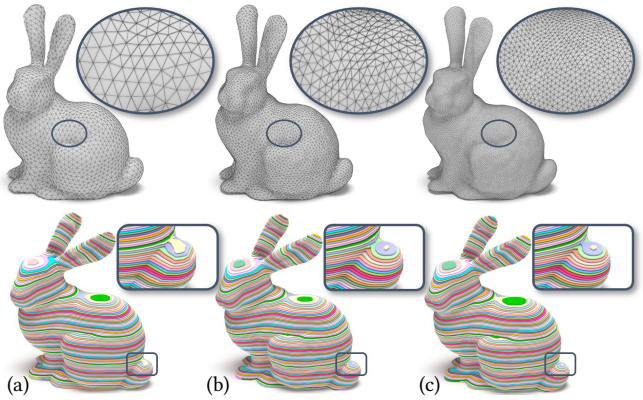


Fig. 17. Tests to study the importance of compatible quaternions on the Bridge and the Ring models, where the metric Π defined in Eq.(13) is evaluated. When different iterations are applied in the quaternion optimization (i.e., the inner loop), different speeds of convergence can be observed for the deformation iteration (i.e., the outer loop).



	(a) Coarse mesh	(b) Original	(c) Dense Mesh
Tetrahedra #	63,787	143,111	390,212
Edge Length (avg.)	4.61	3.15	2.24
Value of Metric Π	0.052	0.048	0.042
Comp. Time (sec.)	8.5	35.6	158.3

Fig. 18. Results of our approach using volumetric meshes of different resolutions. Only the SF objective is imposed in these tests (i.e., $w_{SF} = 1.0$), where the same values of weights $w_r = 1.0$ and $w_c = 6.0$ are applied.

5.4 Discussion

5.4.1 Importance of compatible quaternions. The computation of scale-controlled deformation in our framework can converge very fast when the input rotations are nearly compatible among neighbors. It is interesting to study the performance of this outer loop when feeding incompatible rotations to the deformation step. Specifically, we applied 0 (no iteration), 1, 5 and 10 iterations in the quaternion optimization step as the inner loop. The tests are conducted on the Bridge model and the Ring model (as shown in Fig.17). When no iteration is applied in the quaternion optimization (i.e., only LPD projections are applied to each element to obtain the rotations), the computation does not converge on the Bridge model and needs a large number of deformation steps to converge on the Ring model. The speed of convergence can be immediately improved after applying 1 iteration in the inner loop – i.e., more compatible quaternions (rotations) are fed into the scale-controlled deformation. By testing examples presented in this paper, we found that the computation

of deformation converges quickly when having $5-10$ iterations in the inner loop of quaternion optimization.

Better initial guesses of LPDs can also help to further improve the speed of convergence in our computational framework. Therefore, the heat kernel method [Crane et al. 2017] is extended to support volume meshes to determine the initial guess for SR objective. By using the plane that contacts ground as the heat source, the gradients of a heat field can be employed as the initial guess of LPDs.

5.4.2 Influence of mesh density. Here we study the influence of mesh density on the results when using the same values for the weights w_r and w_c. As shown in Fig. 18, similar results are generated by our approach regardless of the resolution as long as the tetrahedral mesh is fine enough to capture the geometric details of an input model. Specifically, the computations converge to the similar values of the metric Π (as defined in Eq.(13)) that indicates the level of agreement with the objectives. Much longer computing time is consumed on the input model with a dense mesh although the resultant metric value is slightly smaller.

5.4.3 User preference on objectives. Two different types of weights are conducted in our approach: 1) the algorithm related weights that are either computed automatically (e.g., w_s determined by Eq.(11)) or fixed (such as $w_r = 1.0$ and $w_c = 6.0$), and 2) the user-defined parameters (w_{SL}, w_{SR}, w_{SQ}). In our current implementation, these parameters are employed to indicate the preference among multi-objectives. Without loss of generality, we usually let $w_{SL} + w_{SR} + w_{SQ} = 1.0$. When the larger value is given to the weight of an objective, the preference is given to that objective. Different combinations of the weights and their corresponding results are demonstrated in Fig. 19. In the future, a high-level interface can be considered to determine the values of these weights by machine learning.

5.4.4 Limitations. In our current implementation, the foldover elements are not restrictively prohibited. Having such elements in the deformed model will lead to self-intersected layers, which are not possible to be fabricated. Adding new constraints into our framework to prevent foldover may result in a highly nonlinear problem to be solved. Practically, we find that the scale control terms used in Eq.(12) can effectively avoid most of such cases. Only 9 foldover elements were found in the Ring model (which has $40k$ elements in total), and no case was found in any other model. Therefore, we resolved this problem in the post-processing step by applying the method presented in [Su et al. 2019].

Another limitation of our approach is that the setup orientation of a model is not included as a variable to be optimized. For SQ or SF objectives, we choose the setup orientation by the method presented in [Zhang et al. 2015]. For SR objectives, we heuristically determine the printing orientation by searching the orientation that gives the minimal average strain when being fabricated by using planar layers [Ulu et al. 2015]. An alternative could be to use the upright orientation determined by [Fu et al. 2008].

Lastly, only the LPD related fabrication objectives can be integrated into our computational framework now. The other requirements such as global collision avoidance, feedrate control of extrusion and kinematics of hardware are all handled in the post-processing steps. We plan to investigate how to incorporate these objectives in the future work. Other objectives of optimization based on LPDs will also be explored (e.g., the 3D printed magnetic materials for advanced sensors [Zhang et al. 2021b]).

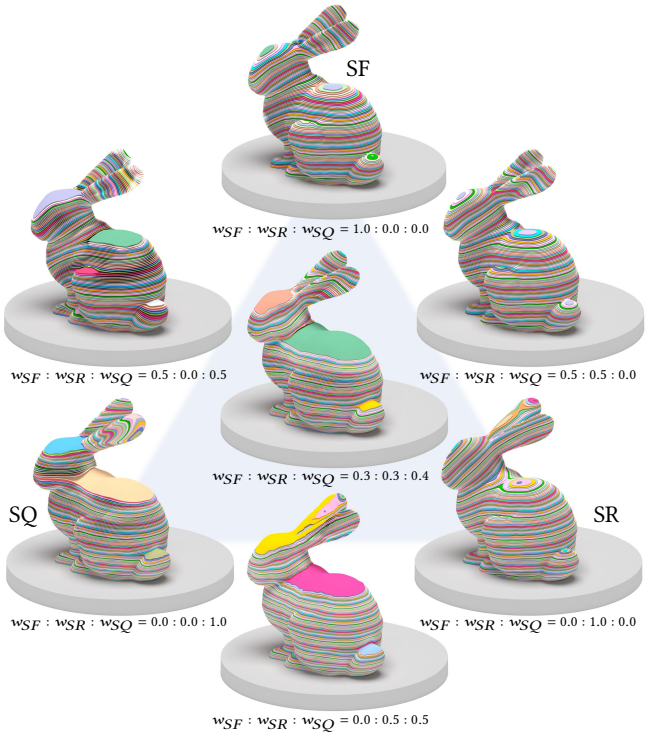


Fig. 19. Users can indicate their preference among the multi-objectives by changing the weights of w_{SF}, w_{SQ} and w_{SR}. Different results can be obtained – e.g., when letting $w_{SQ} = 0.0$ such as the examples located on the right edge of the triangle, only objectives SF and SR are considered.

6 CONCLUSION

This paper presents a new general slicing framework for multi-axis 3D printing, where curved layers for material accumulation are generated to achieve the combined objectives of support-free (SF), strength reinforcement (SR) and enhanced surface quality (SQ). To the best of our knowledge, this is the first time that all these objectives are achieved simultaneously on the same model. The whole framework consists of an inner loop of quaternion-field optimization and an outer loop of scale-controlled deformation, which can be efficiently computed by local/global solvers. The results of our experimental tests are very encouraging. Multiple objectives of fabrication can be successfully achieved. Physical models have been 3D printed and scanned to verify the effectiveness of our approach.

ACKNOWLEDGMENTS

The project is partially supported by the chair professorship fund of the University of Manchester and 5AXISWORKS Ltd via the Innovate UK Smart Grant.

REFERENCES

Daniel Ahlers, Florens Wasserfall, Norman Hendrich, and Jianwei Zhang. 2019. 3D Printing of Nonplanar Layers for Smooth Surface Generation. In *Proceedings of*

the 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE). 1737–1743.

Sung-Hoon Ahn, Shad And, Paul Wright, Michael Montero, Dan Odell, and Shad Roundy. 2002. Anisotropic material properties of fused deposition modeling ABS. *Rapid Prototyping Journal* 8, 4 (2002), 248–257.

M. Aigner, C. Heinrich, B. Jüttler, E. Pilgerstorfer, B. Simeon, and A. V. Vuong. 2009. Swept Volume Parameterization for Isogeometric Analysis. In *Proceedings of the Mathematics of Surfaces XIII*. 19–44.

Rahul Arora, Alec Jacobson, Timothy R. Langlois, Yijiang Huang, Caitlin Mueller, Wojciech Matusik, Ariel Shamir, Karan Singh, and David I.W. Levin. 2019. Volumetric Michell Trusses for Parametric Design & Fabrication. In *Proceedings of the 3rd ACM Symposium on Computation Fabrication (SCF '19)*. 1–13.

Michael Bartoň, Michal Bizzarri, Florian Rist, Oleksii Sliusarenko, and Helmut Pottmann. 2021. Geometry and Tool Motion Planning for Curvature Adapted CNC Machining. *ACM Transactions on Graphics* 40, 4 (2021), 1–16.

Prahar M. Bhatt, Ashish Kulkarni, Rishi K. Malhan, Brual C. Shah, Yeo Jung Yoon, and Satyandra K. Gupta. 2021. Automated Planning for Robotic Multi-Resolution Additive Manufacturing. *Journal of Computing and Information Science in Engineering* 22, 2 (2021), 14 pages. 021006.

Sofien Bouaziz, Mario Deuss, Yuliy Schwartzburg, Thibaut Weise, and Mark Pauly. 2012. Shape-Up: Shaping Discrete Geometry with Projections. *Computer Graphics Forum* 31, 5 (2012), 1657–1667.

Debapriya Chakraborty, B. Aneesh Reddy, and A. Roy Choudhury. 2008. Extruder Path Generation for Curved Layer Fused Deposition Modeling. *Computer-Aided Design* 40, 2 (2008), 235–243.

Keenan Crane, Clarisse Weischedel, and Max Wardetzky. 2017. The Heat Method for Distance Computation. *Commun. ACM* 60, 11 (2017), 90–99.

Chengkai Dai, Charlie C. L. Wang, Chenming Wu, Sylvain Lefebvre, Guoxin Fang, and Yong-Jin Liu. 2018. Support-Free Volume Printing by Multi-Axis Motion. *ACM Transactions on Graphics* 37, 4 (2018), 1–14.

Fernando de Goes, Mathieu Desbrun, and Yiying Tong. 2016. Vector Field Processing on Triangle Meshes. In *ACM SIGGRAPH 2016 Courses (SIGGRAPH '16)*. 1–49.

Xingyi Du, Noam Aigerman, Qingnan Zhou, Shahar Z. Kovalsky, Yajie Yan, Danny M. Kaufman, and Tao Ju. 2020. Lifting Simplices to Find Injectivity. *ACM Transactions on Graphics* 39, 4 (2020), 1–17.

Jimmy Etienne, Nicolas Ray, Daniele Panozzo, Samuel Hornus, Charlie C. L. Wang, Jonàs Martínez, Sara McMains, Marc Alexa, Brian Wyvill, and Sylvain Lefebvre. 2019. CurviSlicer: Slightly Curved Slicing for 3-Axis Printers. *ACM Transactions on Graphics* 38, 4 (2019), 1–11.

Ben Ezair, Saul Fuhrmann, and Gershon Elber. 2018. Volumetric covering print-paths for additive manufacturing of 3D models. *Computer-Aided Design* 100 (2018), 1–13.

Guoxin Fang, Tianyu Zhang, Sikai Zhong, Xiangjia Chen, Zichun Zhong, and Charlie C. L. Wang. 2020. Reinforced FDM: Multi-Axis Filament Alignment with Controlled Anisotropic Strength. *ACM Transactions on Graphics* 39, 6 (2020), 1–15.

Hongbo Fu, Daniel Cohen-Or, Gideon Dror, and Alla Sheffer. 2008. Upright Orientation of Man-Made Objects. In *ACM SIGGRAPH 2008 Papers (SIGGRAPH '08)*. 1–7.

Xiao-Ming Fu, Yang Liu, and Baining Guo. 2015. Computing Locally Injective Mappings by Advanced MIPS. *ACM Transactions on Graphics* 34, 4 (2015), 1–12.

Xiao-Ming Fu, Jian-Ping Su, Zheng-Yu Zhao, Qing Fang, Chunyang Ye, and Ligang Liu. 2021. Inversion-free geometric mapping construction: A survey. *Computational Visual Media* 7, 3 (2021), 289–318.

Vladimir Garanzha, Igor Kaporin, Liudmila Kudryavtseva, François Protais, Nicolas Ray, and Dmitry Sokolov. 2021. Foldover-Free Maps in 50 Lines of Code. *ACM Transactions on Graphics* 40, 4 (2021), 1–16.

S. Gottschalk, M. C. Lin, and D. Manocha. 1996. OBBTree: A Hierarchical Structure for Rapid Interference Detection. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '96)*. 171–180.

Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org.

Kailun Hu, Shuo Jin, and Charlie C. L. Wang. 2015. Support slimming for single material based additive manufacturing. *Computer-Aided Design* 65 (2015), 1–10.

Jin Huang, Yiying Tong, Hongyu Wei, and Hujun Bao. 2011. Boundary Aligned Smooth 3D Cross-Frame Field. *ACM Transactions on Graphics* 30, 6 (2011), 1–8.

Yijiang Huang, Juyong Zhang, Xin Hu, Guoxian Song, Zhongyuan Liu, Lei Yu, and Ligang Liu. 2016. FrameFab: Robotic Fabrication of Frame Shapes. *ACM Transactions on Graphics* 35, 6 (2016), 1–11.

Tao Jiang, Kun Qian, Shuang Liu, Jing Wang, Xiaosong Yang, and Jianjun Zhang. 2017. Consistent as-similar-as-possible non-isometric surface registration. *The Visual Computer* 33, 6 (2017), 891–901.

Shahar Z. Kovalsky, Noam Aigerman, Ronen Basri, and Yaron Lipman. 2015. Large-Scale Bounded Distortion Mappings. *ACM Transactions on Graphics* 34, 6 (2015), 1–10.

Yamin Li, Dong He, Shangqin Yuan, Kai Tang, and Jihong Zhu. 2022. Vector field-based curved layer slicing and path planning for multi-axis printing. *Robotics and Computer-Integrated Manufacturing* 77 (2022), 102362.

Yufei Li, Yang Liu, Weiwei Xu, Wenping Wang, and Baining Guo. 2012. All-Hex Meshing Using Singularity-Restricted Field. *ACM Transactions on Graphics* 31, 6 (2012), 1–11.

Wentao Liao, Renjie Chen, Yuchen Hua, Ligang Liu, and Ofir Weber. 2021. Real-Time Locally Injective Volumetric Deformation. *ACM Transactions on Graphics* 40, 4 (2021), 1–16.

Ligang Liu, Lei Zhang, Yin Xu, Craig Gotsman, and Steven J Gortler. 2008. A local/global approach to mesh parameterization. *Computer Graphics Forum* 27, 5 (2008), 1495–1504.

Marco Livesu, Nicholas Vining, Alla Sheffer, James Gregson, and Riccardo Scateni. 2013. PolyCut: Monotone Graph-Cuts for PolyCube Base-Complex Construction. *ACM Transactions on Graphics* 32, 6 (2013), 1–12.

Huachao Mao, Tsz-Ho Kwok, Yong Chen, and Charlie C. L. Wang. 2019. Adaptive slicing based on efficient profile analysis for additive manufacturing. *Computer-Aided Design* 107 (2019), 89–101.

Ioanna Mitropoulou, Mathias Bernhard, and Benjamin Dillenburger. 2020. Print Paths Key-Framing: Design for Non-Planar Layered Robotic FDM Printing. In *Proceedings of the Symposium on Computational Fabrication (SCF '20)*. 1–10.

Ioanna Mitropoulou, Mathias Bernhard, and Benjamin Dillenburger. 2022. Nonplanar 3D Printing of Bifurcating Forms. *3D Printing and Additive Manufacturing* 9, 3 (2022), 189–202.

Giuseppe Patane, Xin Shane Li, and David Xianfeng Gu. 2013. Surface- and Volume-Based Techniques for Shape Modeling and Analysis. In *SIGGRAPH Asia 2013 Courses (SA '13)*. 1–65.

Michael Rabinovich, Roi Poranne, Daniele Panozzo, and Olga Sorkine-Hornung. 2017. Scalable locally injective mappings. *ACM Transactions on Graphics* 36, 2 (2017), 1–16.

Nicolas Ray, Dmitry Sokolov, and Bruno Lévy. 2016. Practical 3D Frame Field Generation. *ACM Transactions on Graphics* 35, 6 (2016), 1–9.

Jaret C. Riddick, Mulugeta A. Haile, Ray Von Wahlde, Daniel P. Cole, Oluwakayode Bamiduro, and Terrence E. Johnson. 2016. Fractographic analysis of tensile failure of acrylonitrile-butadiene-styrene fabricated by fused deposition modeling. *Additive Manufacturing* 11 (2016), 49 – 59.

Christian Schüller, Ladislav Kavan, Daniele Panozzo, and Olga Sorkine-Hornung. 2013. Locally injective mappings. *Computer Graphics Forum* 32, 5 (2013), 125–135.

Olga Sorkine and Marc Alexa. 2007. As-Rigid-as-Possible Surface Modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing (SGP '07)*. 109–116.

O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, and H.-P. Seidel. 2004. Laplacian Surface Editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP '04)*. 175–184.

Jian-Ping Su, Xiao-Ming Fu, and Ligang Liu. 2019. Practical Foldover-Free Volumetric Mapping Construction. *Computer Graphics Forum* 38, 7 (2019), 287–297.

Kam-Ming Mark Tam and Caitlin T. Mueller. 2017. Additive manufacturing along principal stress lines. *3D Printing and Additive Manufacturing* 4 (2017), 63–81.

Erva Ulu, Emrullah Korkmaz, Kubilay Yay, O. Burak Ozdoganlar, and Levent Burak Kara. 2015. Enhancing the Structural Performance of Additively Manufactured Objects Through Build Orientation Optimization. *Journal of Mechanical Design* 137, 11 (2015), 111410.

John Voight. 2021. *Quaternion algebras*. Springer Nature.

Endong Wang, Qing Zhang, Bo Shen, Guangyong Zhang, Xiaowei Lu, Qing Wu, and Yajuan Wang. 2014. *Intel math kernel library*. Springer. 167–188 pages.

Weiming Wang, Haiyuan Chao, Jing Tong, Zhouwang Yang, Xin Tong, Hang Li, Xiuping Liu, and Ligang Liu. 2015. Saliency-preserving slicing optimization for effective 3d printing. *Computer Graphics Forum* 34, 6 (2015), 148–160.

Rundong Wu, Huaishu Peng, François Guimbretière, and Steve Marschner. 2016. Printing Arbitrary Meshes with a 5DOF Wireframe Printer. *ACM Transactions on Graphics* 35, 4 (2016), 1–9.

Huanhuan Xu, Wuyi Yu, Shiyuan Gu, and Xin Li. 2013. Biharmonic Volumetric Mapping Using Fundamental Solutions. *IEEE Transactions on Visualization & Computer Graphics* 19, 05 (2013), 787–798.

Chengqian Zhang, Xiangjia Li, Laiming Jiang, Daofan Tang, Han Xu, Peng Zhao, Jianzhong Fu, Qifa Zhou, and Yong Chen. 2021b. 3D printing of functional magnetic materials: From design to applications. *Advanced Functional Materials* 31, 34 (2021), 2102777.

Tianyu Zhang, Xiangjia Chen, Guoxin Fang, Yingjun Tian, and Charlie CL Wang. 2021a. Singularity-aware motion planning for multi-axis additive manufacturing. *IEEE Robotics and Automation Letters* 6, 4 (2021), 6172–6179.

Xiaoting Zhang, Xinyi Le, Athina Panotopoulou, Emily Whiting, and Charlie C. L. Wang. 2015. Perceptual Models of Preference in 3D Printing Direction. *ACM Transactions on Graphics* 34, 6 (2015), 1–12.