

# Support Slimming for Single Material Based Additive Manufacturing

Kailun Hu, Shuo Jin, and Charlie C.L. Wang\*

Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, P. R. China

\*Corresponding Author. Tel: (+852)-39438052. Fax: (+852)-26036002. Email: cwang@mae.cuhk.edu.hk

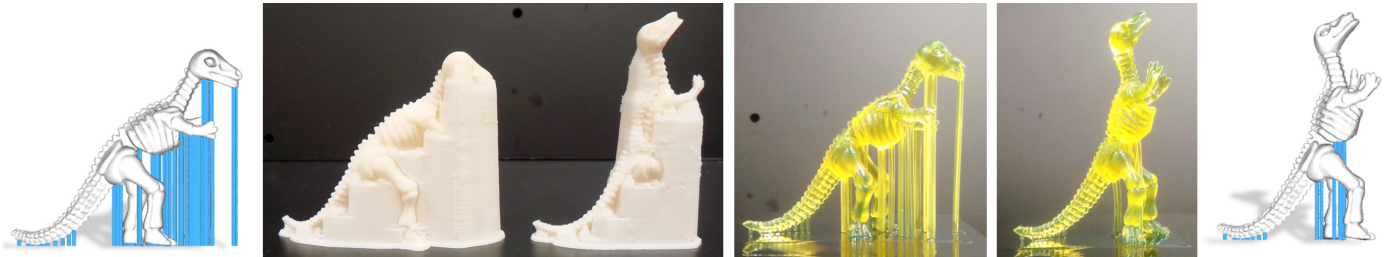


Figure 1: An orientation-driven shape optimization approach is presented in this paper to show how a given model can be deformed so that the usage of the support structure is significantly reduced in AM with a single material. Layered fabrication based on the FDM method (middle left) and on the SLA method (middle right) can both benefit from this work to improve the efficiency of the manufacturing and the quality of the finished models.

## Abstract

In layer-based additive manufacturing (AM), supporting structures need to be inserted to support the overhanging regions. The adding of supporting structures slows down the speed of fabrication and introduces artifacts onto the finished surface. We present an orientation-driven shape optimizer to slim down the supporting structures used in single material-based AM. The optimizer can be employed as a tool to help designers to optimize the original model to achieve a more self-supported shape, which can be used as a reference for their further design. The model to be optimized is first enclosed in a volumetric mesh, which is employed as the domain of computation. The optimizer is driven by the operations of reorientation taken on tetrahedra with ‘facing-down’ surface facets. We formulate the demand on minimizing shape variation as global rigidity energy. The local optimization problem for determining a minimal rotation is analyzed on the Gauss sphere, which leads to a closed-form solution. Moreover, we also extend our approach to create the functions of controlling the deformation and searching for optimal printing directions.

**Keywords:** geometric algorithm, shape optimization, design tool, additive manufacturing

## 1. Introduction

AM has emerged as one of the most important methods for realizing the fast fabrication of freeform solids. *Stereolithography Apparatus* (SLA) and *Fused Deposition Modeling* (FDM) are two widely used approaches in AM because they achieve a very good balance between the cost and the quality. Both SLA and FDM fabricate models in a layer-by-layer manner, where supporting structures (also simply called *support*) need to be added during the manufacturing process. Specifically, the manufacturing material cannot be deposited on a layer where there is insufficient material on the previous layer. For example, the overhangs with a large hanging area can easily collapse under gravity. The problem is solved by adding supports to the originally designed models (Fig.1). Recent developments in AM allow us to generate the supporting structures automatically (e.g., [1, 2]).

### 1.1. Problems Caused by Support

When the support is fabricated by a dissolvable material which is different from the one used to print the designed

model, the support can be removed automatically by a post-process (ref. [4]). However, for those manufacturing techniques with a single material (e.g., SLA and the low-cost FDM machines), the supporting structure poses many problems to users. Firstly, the volume of the support could be large compared to the designed model, which leads to a significant waste of materials, energy and time - our study shows that up to 63.6% of the manufacturing time in FDM could be spent on the fabrication of the support. Although the increase of fabrication time in the *Mask-Image Projection based Stereolithography Apparatus* (MIP-SLA) [5] is not significant, because it is proportional only to the number of layers, another problem caused by the supports in single-material AM is common. This is the difficulty in removing the supports automatically. When the support is fabricated in the same material as the design model, it is linked to the model by many thin columns. After fabrication, the support is separated from the main object by being torn away at the top of the columns. This step is always performed manually. More seriously, the surface of the main object is easily damaged by the visual artifacts which are left on its surface (see Figure 22 for an example). Note that although the process of drop-on-powder

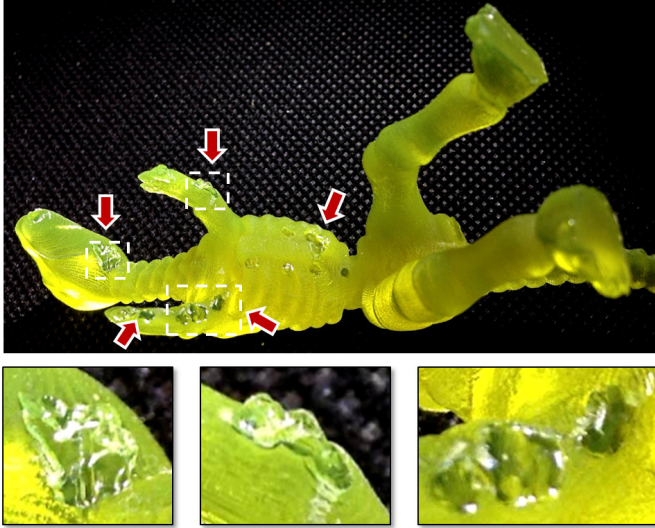


Figure 2: Artifacts are left on the surface of finished models after removing the support structures – the photograph is of a model fabricated by MIP-SLA. Examples of single-material FDM can be found in [3].

based AM and *Selective Laser Sintering* (SLS) is self-supported by the powders, a large group of machines using FDM and SLA suffer from the problem of support.

### 1.2. Tool for Design Pipeline - Motivation of This Work

In the literature of design-for-manufacturing, many applications allow the shape of a model be adjusted before finalizing the design so that a better manufacturability can be achieved. After designing a shape with the help of modern geometric modeling techniques, designers start to take the ‘self-supportness’ into consideration when they wish to fabricate the physical model by single-material AM. In many cases, designers manually change the shape of a design and then verify the self-supportness by applying the support generation tools. Such a trial-and-error process is tedious and can take up much time and effort. Little research attention has been paid in the literature to the automation of this procedure. In this work, we provide a support slimming optimizer in the design pipeline to help designers to generate better self-supported intermediate models as references for finalizing their designs. A new design pipeline after integrating our shape optimizer is shown in Fig.3.

### 1.3. Main Results

We propose a novel shape optimization approach to optimize the shape of a designed model  $\mathcal{M}$  into a ‘self-supported’ state for AM (see Fig.1). The optimizer is formulated on a volumetric mesh  $\mathcal{T}$  enclosing  $\mathcal{M}$  (Section 3.1). The global shape of  $\mathcal{M}$  is preserved by minimizing the energy of rigidity defined on  $\mathcal{T}$  (Section 3.2). The benefit of taking the computation on  $\mathcal{T}$  is twofold. Firstly, the computation is more efficient and easier to converge when the mesh density of  $\mathcal{T}$  is coarser than  $\mathcal{M}$ . Secondly, geometric details on  $\mathcal{M}$  can be preserved during the

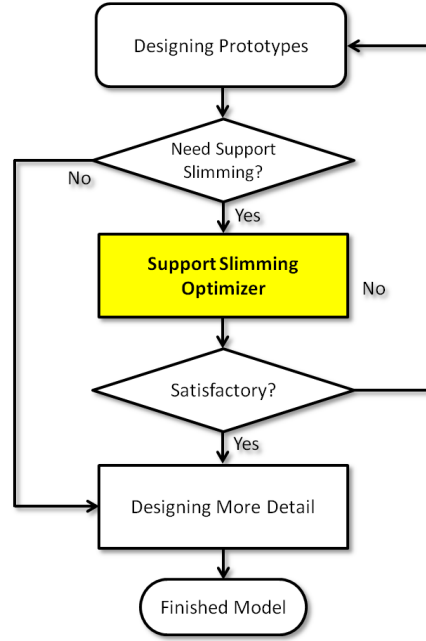


Figure 3: Our supporting slimming optimizer (the stage shown in yellow) can be incorporated into the common design pipeline as a tool. This flowchart shows how to use our optimizer in AM-based model design.

optimization. The procedure of optimization is driven by reorienting the tetrahedra with ‘facing-down’ surface facets. Being a major technical contribution of this work, a closed-form solution is derived from the analysis on the Gauss sphere to determine the minimal rotations (Section 4). This work provides a tool (with error control) for manipulating the shape of a designed model to reduce the usage of supporting structures. Our optimizer can be integrated into the design pipeline to relieve the designers from carrying out tedious work (Fig.3). To the best of our knowledge, this paper is the first one which tackles the shape optimization problem so as to reduce the usage of supporting structures in AM.

## 2. Related Works

Shape and topology optimization techniques have been widely employed in a variety of engineering applications (ref. [6, 7]). Recently, these techniques have been used in AM applications. For instance, the issue of a model balance for AM has been tackled by Prévost et al.[8] by carving and shape deformation. This approach is followed by a work in [9] to optimize the moment of inertia for designing a spinnable object. Telea and Jalba [10] investigated the printability of a model by voxel representation, where regions, bridges, spike and holes which are too thin can be automatically detected. A physically based stress-relief algorithm has been presented in [11] to automatically detect and modify the areas with high stress by hollowing, thickening and inserting struts to reinforce the strength of the models. Wang et al. [12] tried to generate a skin-frame structure to minimize the volume of manufacturing

material and the number of struts. Zhou et al. [13] have developed a method for determining the worst-load distribution for a shape that will cause high local stress or large deformations so that the shape can be enhanced at those weak regions. Umetani and Schmidt [14] have proposed a method to determine an optimal printing direction to maximize the mechanical strength of a model fabricated by AM. Lu et al. recently solved a strength-to-weight problem in [15] to relieve the interior stress inside a printed model by introducing optimal honeycomb-cells structures. Although many optimization issues relating to AM have been investigated, how to optimize the shape to reduce the supporting structures is still an open problem which needs to be studied.

The determination of the parting direction for a mold is a typical orientation optimization problem, and it is one that has been studied for decades. Chen [16] presented an algorithm to determine the parting direction by finding the minimum volume bounding box through fuzzy representation selection. Priyadarshi and Gupta proposed an algorithm in [17] to find the feasible parting directions through the visibility of facets by graphics hardware acceleration. The graphics accelerated hardware was also employed in [18] to help speed up the search of the feasible mold parting directions. Li et al. proposed a parting line generation algorithm for mesh models in [19]. All these approaches can be regarded as ‘rigid’ orientation optimization techniques that do not change the shape of the models. Our work deforms both the shape and the pose of an input model to slim down its supporting structure for single-material AM.

In the literature, many geometric modeling and processing problems are formulated under the framework of optimization. A good survey can be found in the book by Botsch et al. [20]. Because the shape optimization taken in this paper is based on a local/global deformation strategy, we review only the related techniques below. Some recent approaches solve the deformation problem based on the *as-rigid-as-possible* (ARAP) consideration [21, 22], where they preserve the rigidity of every element in the local step and then globally blend the elements together by solving a least-square problem. The position constraints of handles can be enforced by fixing the corresponding variables in the global step. The local/global optimization is also employed in [23] for solving the mesh parameterization problem. Recently, Kwok and Wang [24] presented an optimization framework for the design automation of human-centric products by solving a mixed-integer ARAP optimization problem. The procedure of local/global ARAP deformation employed in this paper is driven by a novel reorientation scheme to solve the problem of support slimming.

### 3. Optimization by Deformation

The mission of our optimizer is to deform the input model  $\mathcal{M}$  to a shape  $\mathcal{M}'$  by which the supporting structures are reduced and local details are preserved. To achieve this goal, a volumetric mesh  $\mathcal{T}$  is employed as the computation domain so that the geometry details can be preserved. The demands placed on the slimming down supports are formulated as reducing the

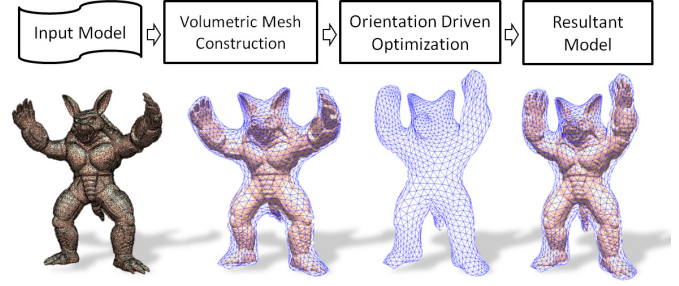


Figure 4: Flowchart of our shape optimization algorithm.

‘facing-down’ surface facets on  $\mathcal{T}$  while minimizing its deformation (see the flowchart shown in Fig.4).

#### 3.1. Computation Domain and Notations

To preserve the geometry details on the given model  $\mathcal{M}$ , we constructed a tetrahedral mesh  $\mathcal{T}$  enclosing  $\mathcal{M}$  as the computation domain. Every vertex on  $\mathcal{M}$  is encoded with the positions of the tetrahedral vertices on  $\mathcal{T}$  by the barycentric coordinate. As a result, when  $\mathcal{T}$  is deformed into a new shape, a deformed shape of  $\mathcal{M}$  (i.e.,  $\mathcal{M}'$ ) can be obtained by applying the barycentric coordinate. When the volumetric mesh  $\mathcal{T}$  is coarser than  $\mathcal{M}$ , the geometry details on  $\mathcal{M}$  can be preserved in the deformation. There are many methods described in the literature for generating a surface mesh to enclose an input surface mesh (e.g., [25, 26]). After that, the tetrahedral mesh  $\mathcal{T}$  can be generated from the enclosing surface mesh [27].

For a tetrahedral mesh  $\mathcal{C}_{\mathcal{T}} = (\mathcal{V}; \mathcal{E}; \mathcal{F}; \mathcal{T})$ ,  $\mathcal{V} = \{v_i\}$ ,  $\mathcal{E} = \{e_{ij}\}$ ,  $\mathcal{F} = \{f_{ijk}\}$ ,  $\mathcal{T} = \{t_{ijkl}\}$  are used to denote the sets of vertices, edges, facets and tetrahedrals, respectively.  $\mathbf{v}_i \in \mathbb{R}^3$  gives the position of each vertex  $v_i$ , and  $\hat{\mathbf{n}}_j$  denotes the normal of a face  $f_j$ . The faces that are located at the boundary of  $\mathcal{T}$  are called *surface faces*, and the tetrahedra that have at least one surface face are named as *surface tetrahedra*. The sets of surface faces and tetrahedra are represented by  $\mathcal{F}^s$  and  $\mathcal{T}^s$ . Moreover, all values from the original model are represented with a superscript ‘o’, and those computed from the current status are denoted with a superscript ‘c’.

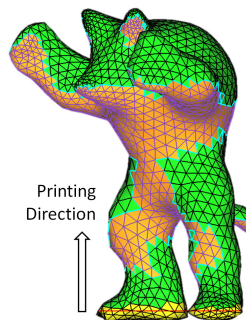


Figure 5: Faces are classified according to the printing direction.

**Printing Direction:** It is the fabrication direction along which the model is printed layer by layer - denoted by  $\hat{\mathbf{d}}_p$ . Without the loss of generality, in the rest of this paper we assume that  $\hat{\mathbf{d}}_p$  is a unit vector.

**Maximal Self-Supported Angle:** An overhanging region that can be printed without adding support is called self-supported. The angle between the region’s tangent plane and the print-

ing direction is called the self-supported angle  $\alpha$ . When different manufacturing materials are used, the AM processes are equipped with different maximal self-supported angles. We define a self-supporting coefficient  $\tau$  in our formulation in accordance with the maximal self-supported angle  $\alpha_{\max}$  as  $\tau = \sin(\alpha_{\max})$ .

**Fixed Region:** During the shape optimization, a region of the model is selected to be fixed to serve as the boundary conditions (see the yellow region in Fig.5 as an example). Tetrahedral vertices (both the boundary and the interior ones) in this region are fixed. Users are free to define the parts to be fixed in our approach. Usually, the bottom of a model is selected as the fixed region.

**Safe Face:** The surface face on  $C_{\mathcal{T}}$  whose normal  $\hat{\mathbf{n}}$  satisfies

$$\hat{\mathbf{n}} \cdot \hat{\mathbf{d}}_p \geq -\tau. \quad (1)$$

See the green faces in Figure 55 for an example. These faces are termed the self-supported faces, and the above inequality on the normals of faces is termed the *self-supporting condition*.

**Risky Face:** The surface face with  $\hat{\mathbf{n}} \cdot \hat{\mathbf{d}}_p < -\tau$ . For example, the orange regions in Figure 55 are formed by risky faces.

Note that a surface tetrahedron containing no risky face is called a *safe tetrahedron*. On the other aspect, if one of the surface tetrahedron's faces is risky, the tetrahedron is also defined as a *risky tetrahedron*.

### 3.2. Optimization Framework

The goal of slimming down the support in AM can be interpreted as deforming the given model into a new shape so that it has fewer risky faces. Because the volumetric mesh  $C_{\mathcal{T}}$  is used as the domain of computation, we convert the problem into the requirement of turning risky faces into safe ones while minimizing the deformation.

**Deformation Energy:** For each tetrahedron  $t \in \mathcal{T}$  with four vertices  $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$  and  $\mathbf{v}_4$ , we construct a local frame in  $t$  as

$$\mathbf{V}_t = [\mathbf{v}_1 - \mathbf{v}_4 \quad \mathbf{v}_2 - \mathbf{v}_4 \quad \mathbf{v}_3 - \mathbf{v}_4] \quad (2)$$

with  $\mathbf{v}_4$  being an interior vertex on the volumetric mesh. This selection of local origin is based on the heuristic that an interior vertex is connected with more tetrahedra. Making it static (as origin) in a local rotation will have less influence on the neighboring tetrahedra. With the help of  $\mathbf{V}_t$ , the *as-rigid-as-possible* (ARAP) energy of a deformed tetrahedral mesh can be defined as

$$E(C_{\mathcal{T}}^{new}, C_{\mathcal{T}}^o) = \sum_{t \in \mathcal{T}} w_t \|\mathbf{V}_t^{new} - \mathbf{L}_t \mathbf{V}_t^o\|_F^2, \quad (3)$$

where  $\|\cdot\|_F$  is the Frobenius norm,  $\mathbf{L}_t$  is a rigid transformation matrix of the tetrahedron  $t$ , and  $w_t$  is  $t$ 's volume serves as the weight. In the prior work of ARAP deformation, position handles are applied to move some vertices so that an updated shape can be obtained by minimizing the above ARAP energy. The optimization is actually in a least-square form when  $\mathbf{L}_t$ s are the

constant matrices determined in the local step. Here, we formulate the ARAP energy as Eq.(3) instead of  $\sum w_t \|\mathbf{L}_t \mathbf{V}_t^{new} - \mathbf{V}_t^o\|_F^2$ . As a result, the factorization of a linear system can be reused in the subsequent iterations of optimization.

**Rigid Transformation:** The local step for obtaining  $\mathbf{L}_t$  plays a very important role in our formulation.

- In the prior ARAP work, the rigid transformation from the original shape  $\mathbf{V}_t^o$  of a tetrahedron  $t$  to its current shape  $\mathbf{V}_t^c$  is determined by using a ‘‘signed version’’ of the *singular value decomposition* (SVD) on the affine transformation matrix  $\mathbf{Q} = \mathbf{V}_t^c (\mathbf{V}_t^o)^{-1}$  (ref. [22]). That is,  $\mathbf{Q} = \mathbf{U} \mathbf{\Sigma} \mathbf{W}^T$ , where  $\mathbf{\Sigma}$  is a diagonal matrix with all elements being positive.  $\mathbf{\Sigma}$  defines the scaling factors in three orthogonal directions in  $\mathbf{Q}$ . As a result,  $\mathbf{Q}$  can be turned into a rigid transformation  $\mathbf{M} = \mathbf{U} \mathbf{W}^T$ .
- Applying  $\mathbf{M}$  to a tetrahedron  $t$  will transform it to an orientation that best approximates to the current shape of  $t$ . A transformed surface tetrahedron may contain some risky faces, and these need to be turned into safe ones by applying an additional rotation  $\mathbf{R}$ . Basically, an ideal rotation turns all risky faces on a tetrahedron into safe ones while conducting a rotation with a minimal rotational angle. The method for solving this problem will be presented in Section 4. Specifically, after determining a minimal rotation angle  $\theta$  and its corresponding rotation axis  $\hat{\mathbf{r}}$ ,  $\mathbf{R}(\hat{\mathbf{r}}, \theta)$  can be obtained [28]. For those safe tetrahedra and the interior tetrahedra,  $\mathbf{R}(\hat{\mathbf{r}}, \theta) = \mathbf{I}$ .

The rigid transformation applied in our optimization framework (i.e., Eq.(3)) is a cascade of these two transformations because  $\mathbf{L}_t = \mathbf{R}(\hat{\mathbf{r}}, \theta) \mathbf{M}$ .

**Procedure of Optimization:** We iteratively apply the local and then the global steps on the volumetric mesh  $C_{\mathcal{T}}$  to reduce the number of risky faces.

- Step 1): For each tetrahedron, the rigid transformation  $\mathbf{M}$  between its current position/shape and the given position/shape is computed.
- Step 2): For each risky tetrahedron, a minimal rotation  $\mathbf{R}(\hat{\mathbf{r}}, \theta)$  is computed to convert it into a safe one (see Section 4 below). As a result,  $\mathbf{L}_t$  is obtained for all tetrahedra.
- Step 3): A least-square solution for the new position/shape of the volumetric mesh is obtained by minimizing the deformation energy (Eq.(3)).
- Step 4): Go back to step 1) until the terminal condition is reached.

A hybrid terminal condition is employed in the iteration: (a) the maximal iteration steps (e.g., 100 in our implementation) and (b) the number of risky faces do not decrease in the successive five steps of the iteration. More sophisticated terminal conditions will be introduced in Section 5.2 to control the geometry approximation error on the optimized (deformed) model.

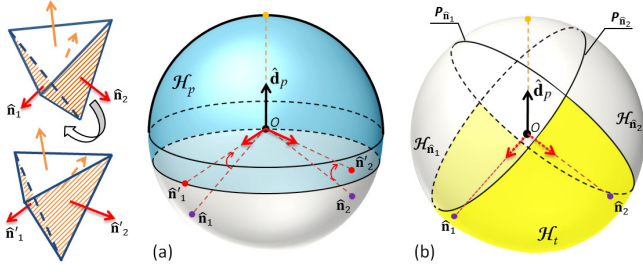


Figure 6: Rotation analysis on the Gauss sphere to enforce the condition of self-supporting: (a) a rotation to move  $\hat{\mathbf{n}}_1$  and  $\hat{\mathbf{n}}_2$  into the feasible region  $\mathcal{H}_p$  (in cyan) w.r.t. the printing direction  $\hat{\mathbf{d}}_p$ , and (b) dual-feasible-regions of  $\hat{\mathbf{n}}_1$  and  $\hat{\mathbf{n}}_2$ ,  $\mathcal{H}_{\hat{\mathbf{n}}_1}$  and  $\mathcal{H}_{\hat{\mathbf{n}}_2}$ , forms a target region as  $\mathcal{H}_t = \mathcal{H}_{\hat{\mathbf{n}}_1} \cap \mathcal{H}_{\hat{\mathbf{n}}_2}$  to inversely rotate  $\hat{\mathbf{d}}_p$  into.

#### 4. Computation of Minimal Rotation

Served as the core of our shape optimization approach, the computation of minimal rotation is formulated in a closed-form in this section.

**Problem Definition:** Given a tetrahedron  $t \in \mathcal{T}^s$  with four faces  $\{f_i\}$  ( $i = 1, \dots, 4$ ) and their corresponding normals as  $\{\hat{\mathbf{n}}_i\}$ , the minimal rotation is define as

$$\begin{aligned} & \arg \min_{\hat{\mathbf{r}}, \theta} \|\mathbf{R}(\hat{\mathbf{r}}, \theta) - \mathbf{I}\|_g^2 \\ & \text{s.t.}, \quad \forall f_i \in \mathcal{F}^s, (\mathbf{R}(\hat{\mathbf{r}}, \theta)\hat{\mathbf{n}}_i) \cdot \hat{\mathbf{d}}_p \geq -\tau \end{aligned} \quad (4)$$

where  $\|\cdot\|_g$  is a *general* matrix norm to measure how significantly  $\mathbf{R}(\hat{\mathbf{r}}, \theta)$  deviates from  $\mathbf{I}$ .

Directly solving this problem by constrained nonlinear optimization is tedious and time-consuming. The problem is a pure rotation issue. Therefore, we analyze it on the Gauss sphere to find a compact solution.

##### 4.1. Analysis on Gauss Sphere

For a risky tetrahedron in  $\mathbb{R}^3$ , the unit normal vectors of its four faces can be considered as four points on the Gauss sphere  $\mathbb{S}^2$ . Meanwhile, the unit vector  $\hat{\mathbf{d}}_p$  of the printing direction is also mapped to a point on  $\mathbb{S}^2$ . Without a loss of generality,  $\hat{\mathbf{d}}_p$  is assumed to be the top point of the Gauss sphere (i.e.,  $\hat{\mathbf{d}}_p = (0, 0, 1)$ ). Now the condition of self-supporting for a surface face (Eq.1) can be interpreted as that the normal  $\hat{\mathbf{n}}$  must be located at the half-space  $\mathcal{H}_p$  above/on the plane  $z = -\tau$ . As illustrated in Fig.6(a), a risky tetrahedron may have a few surface faces that are not self-supported. When this situation occurs, we can apply a rotation on the normal vectors to move them into the safe region (i.e., inside  $\mathcal{H}_p$ ). The rotation is actually  $\mathbf{R}(\hat{\mathbf{r}}, \theta)$  used in Eq.(4), where a minimal rotation angle  $\theta$  is demanded.

When there is more than one risky face on a tetrahedron, determining a minimal rotation to move all points (corresponding to the normals on risky faces) into the feasible region  $\mathcal{H}_p$  is not straightforward – see Fig.6(a). To solve this problem, a dual representation of  $\mathcal{H}_p$  is defined as follows.

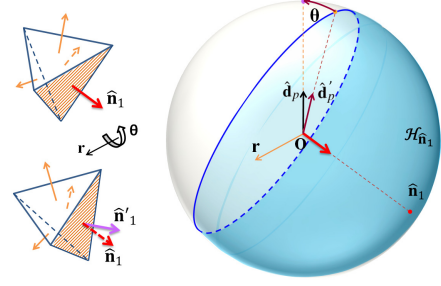


Figure 7: Case I – there is only one surface face on a risky tetrahedron.

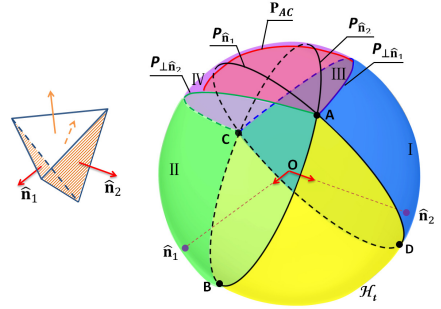


Figure 8: Case II – there are two surface faces on a risky tetrahedron. The Gauss sphere is decomposed into five regions corresponding to the five different configurations of minimal rotation.

**Dual-Feasible-Region (DFR):** The self-supporting condition for a surface face with normal  $\hat{\mathbf{n}}$  (i.e., Eq.(1)) can be considered as requesting  $\hat{\mathbf{d}}_p$  to be located at the same side of  $\hat{\mathbf{n}}$  with reference to the plane:  $x\hat{\mathbf{n}}_x + y\hat{\mathbf{n}}_y + z\hat{\mathbf{n}}_z + \tau = 0$ . Therefore, a DFR of  $\hat{\mathbf{d}}_p$  for  $\hat{\mathbf{n}}$  can be defined by the half-space

$$\mathcal{H}_{\hat{\mathbf{n}}} = \{\mathbf{p} \mid \forall \mathbf{p} \in \mathbb{S}^2, \mathbf{p} \cdot \hat{\mathbf{n}} + \tau \geq 0\}. \quad (5)$$

Thus, the self-supporting condition becomes  $\hat{\mathbf{d}}_p \in \mathcal{H}_{\hat{\mathbf{n}}}$ .

**Inverse Rotation:** Rotating  $\hat{\mathbf{n}}$  by  $\theta$  around the axis  $\hat{\mathbf{r}}$  is equivalent to inversely rotating  $\hat{\mathbf{d}}_p$  with the angle  $(-\theta)$  around the same axis. As a result, the minimal angle can be determined by rotating  $\hat{\mathbf{d}}_p$  into the DFR of  $\hat{\mathbf{n}}$ . Specifically, for a surface tetrahedron  $t$  with  $m$  surface faces ( $m > 1$ ), we can inversely rotating  $\hat{\mathbf{d}}_p$  into the common area defined by the DFRs of these faces – that is  $\mathcal{H}_t = \mathcal{H}_{\hat{\mathbf{n}}_1} \cap \dots \cap \mathcal{H}_{\hat{\mathbf{n}}_m}$  (see Fig.6(b)).

Methods for determining the minimal angle (by inverse rotation) in different configurations are presented below. Cases of tetrahedra with one and two surface faces are discussed. We found only a very few cases with three surface faces in our tests. For such cases, the configuration space on the Gauss sphere is divided into eight regions. This makes the closed-form formulation of minimal rotation very tedious. To have an easy-to-implement approach, a tetrahedron  $t$  with more than two surface faces is split into four tetrahedra by inserting a new vertex at the middle of  $t$ , and this will not affect  $t$ 's neighboring tetrahedra.

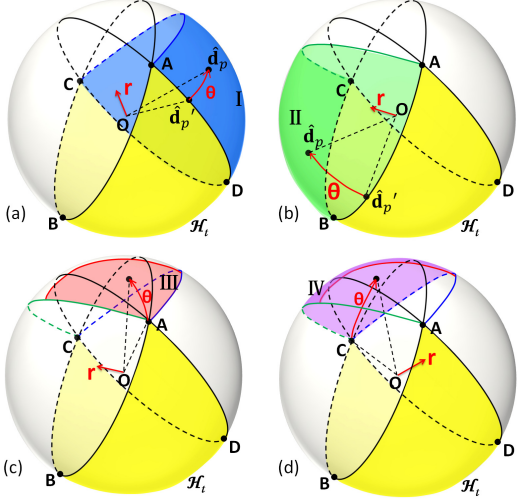


Figure 9: When  $\hat{\mathbf{d}}_p$  falls in different regions, it has different targets for the minimal rotation: (a)-(d) the configurations for regions I-IV, respectively.

#### 4.2. Case I: One Surface Face

For the case of a tetrahedron  $t$  with only one surface face  $f$  whose normal is  $\hat{\mathbf{n}}$ , if the self-supporting condition (Eq.(1)) is not satisfied on  $f$ , the point  $\hat{\mathbf{d}}_p$  must fall outside the feasible region  $\mathcal{H}_{\hat{\mathbf{n}}}$ . The minimal inverse rotation that moves  $\hat{\mathbf{d}}_p$  back into  $\mathcal{H}_{\hat{\mathbf{n}}}$  can be determined by projecting  $\hat{\mathbf{d}}_p$  onto the plane of  $\mathcal{H}_{\hat{\mathbf{n}}}$  and then normalizing it to the unit vector  $\hat{\mathbf{d}}'_p$ . Here  $\hat{\mathbf{d}}'_p$  is the closest point to  $\hat{\mathbf{d}}_p$  in  $\mathcal{H}_{\hat{\mathbf{n}}}$ . Thus, the minimal rotation axis is  $\mathbf{r} = \hat{\mathbf{d}}'_p \times \hat{\mathbf{d}}_p$  and the rotation angle  $\theta = \arccos(\hat{\mathbf{d}}'_p \cdot \hat{\mathbf{d}}_p)$ .

#### 4.3. Case II: Two Surface Faces

For the case that a tetrahedron  $t$  has two surface faces  $f_1$  and  $f_2$  (with  $\hat{\mathbf{n}}_1$  and  $\hat{\mathbf{n}}_2$  as their normals respectively), we present a closed-form solution of the minimal rotation below. Again, the analysis is performed with the help of the DFRs and the inverse rotation. Specifically, the Gauss sphere is decomposed into five regions with respect to  $\mathcal{H}_{\hat{\mathbf{n}}_1}$  and  $\mathcal{H}_{\hat{\mathbf{n}}_2}$ , where different regions lead to different configurations of minimal rotation. The yellow part is the *feasible* region  $\mathcal{H}_t$ . The boundaries of the configuration regions are formed by five planes (see the illustration in Fig.8). Two planes,  $\mathbf{P}_{\perp\hat{\mathbf{n}}_1}$  and  $\mathbf{P}_{\perp\hat{\mathbf{n}}_2}$  passing through the corner points of  $\mathcal{H}_t$  are perpendicular to  $\mathbf{P}_{\hat{\mathbf{n}}_1}$  and  $\mathbf{P}_{\hat{\mathbf{n}}_2}$  respectively. The plane  $\mathbf{P}_{AC}$  passing through the origin is perpendicular to the vector  $\hat{\mathbf{c}}_a$ , which splits the region into two halves. Equations of the five planes help to define five half-spaces, as follows:

$$\begin{aligned}
\mathcal{H}_{\hat{\mathbf{n}}_1} &= \{\mathbf{p} \mid \forall \mathbf{p} \in \mathbb{S}^2, \hat{\mathbf{n}}_1 \cdot \mathbf{p} + \tau \geq 0\} \\
\mathcal{H}_{\hat{\mathbf{n}}_2} &= \{\mathbf{p} \mid \forall \mathbf{p} \in \mathbb{S}^2, \hat{\mathbf{n}}_2 \cdot \mathbf{p} + \tau \geq 0\} \\
\mathcal{H}_{AC} &= \{\mathbf{p} \mid \forall \mathbf{p} \in \mathbb{S}^2, \hat{\mathbf{c}}_a \cdot \mathbf{p} \geq 0\} \\
\mathcal{H}_{\perp\hat{\mathbf{n}}_1} &= \{\mathbf{p} \mid \forall \mathbf{p} \in \mathbb{S}^2, (\hat{\mathbf{c}}_a \times \hat{\mathbf{n}}_1) \cdot \mathbf{p} + d \geq 0\} \\
\mathcal{H}_{\perp\hat{\mathbf{n}}_2} &= \{\mathbf{p} \mid \forall \mathbf{p} \in \mathbb{S}^2, (\hat{\mathbf{n}}_2 \times \hat{\mathbf{c}}_a) \cdot \mathbf{p} + d \geq 0\}
\end{aligned} \tag{6}$$

where  $\hat{\mathbf{c}}_a = (\hat{\mathbf{n}}_1 \times \hat{\mathbf{n}}_2) / \|\hat{\mathbf{n}}_1 \times \hat{\mathbf{n}}_2\|$  and  $d = \tau \|\hat{\mathbf{n}}_1 \times \hat{\mathbf{n}}_2\| / (1 + \hat{\mathbf{n}}_1 \cdot \hat{\mathbf{n}}_2)$ .

**Configuration I:** When  $\hat{\mathbf{d}}_p \in \overline{\mathcal{H}_{\hat{\mathbf{n}}_1}} \cap \mathcal{H}_{\perp\hat{\mathbf{n}}_1}$ , the minimal inverse rotation moves  $\hat{\mathbf{d}}_p$  onto the arc  $\widehat{ADC}$  (see the blue region in

Fig.9(a)). Here,  $\overline{\mathcal{H}}$  denotes a complementary set of  $\mathcal{H}$ . To realize that, we project  $\hat{\mathbf{d}}_p$  onto the plane  $\mathbf{P}_{\hat{\mathbf{n}}_1}$  and normalize it to a unit vector  $\hat{\mathbf{d}}'_p$ . As a result, the rotation axis for turning risky faces safe is  $\mathbf{r} = \hat{\mathbf{d}}'_p \times \hat{\mathbf{d}}_p$  with the angle  $\theta = \arccos(\hat{\mathbf{d}}'_p \cdot \hat{\mathbf{d}}_p)$ . Note that, in all the configurations below,  $\mathbf{r}$  and  $\theta$  can be determined in the same way after obtaining  $\hat{\mathbf{d}}'_p$ .

**Configuration II:** This configuration occurs when  $\hat{\mathbf{d}}_p \in \overline{\mathcal{H}_{\hat{\mathbf{n}}_2}} \cap \mathcal{H}_{\perp\hat{\mathbf{n}}_2}$  (i.e., falling in the green region shown in Fig.9(b)). We then project  $\hat{\mathbf{d}}_p$  onto the plane  $\mathbf{P}_{\hat{\mathbf{n}}_2}$  and normalize it to  $\hat{\mathbf{d}}'_p$ , which actually moves  $\hat{\mathbf{d}}_p$  onto the arc  $\widehat{ABC}$ .

**Configuration III:** The respective region of this configuration is  $\mathcal{H}_{AC} \cap \overline{\mathcal{H}_{\perp\hat{\mathbf{n}}_1}} \cap \overline{\mathcal{H}_{\perp\hat{\mathbf{n}}_2}}$  (the red region shown in Fig.9(c)). If  $\hat{\mathbf{d}}_p$  falls in this configuration, the closest point in the feasible region  $\mathcal{H}_t$  is the point A. The position of A can be computed by the planes. We then assign the position of A to  $\hat{\mathbf{d}}'_p$ .

**Configuration IV:** This is  $\hat{\mathbf{d}}_p \in \mathcal{H}_{AC} \cap \overline{\mathcal{H}_{\perp\hat{\mathbf{n}}_1}} \cap \overline{\mathcal{H}_{\perp\hat{\mathbf{n}}_2}}$  (the purple region in Fig.9(d)). In  $\mathcal{H}_t$ , the closest point to  $\hat{\mathbf{d}}_p$  is point C. Thus, its position is assigned as  $\hat{\mathbf{d}}'_p$  to compute the minimal rotation along the axis  $\mathbf{r} = \hat{\mathbf{d}}'_p \times \hat{\mathbf{d}}_p$  with the angle  $\theta = \arccos(\hat{\mathbf{d}}'_p \cdot \hat{\mathbf{d}}_p)$ .

By the configuration analysis above, we solve the problem of minimal rotation in a closed-form instead of non-linear optimization. Meanwhile, the global step is a pre-factorized least-square problem. As a result, each iteration of the optimization procedure can be evaluated very efficiently.

## 5. Results and Discussion

The approach proposed in this paper has been implemented by C++ together with the Eigen library [29] as the numerical solver. All the tests below are taken on an Intel Core i7-3770 3.40GHz computer with 8GB RAM. Our current implementation does not use the multi-core acceleration. The approach has been tested with several models, and all the results are encouraging.

### 5.1. Experimental Results

Our first example is the Dino model which was shown in Fig.1. In this optimization, we set the maximal self-supported angle  $\alpha_{\max} = 0$ . As a result of shape optimization, the Dino ‘raises up’ its head and arms. Local details of the bone structures are well-preserved. The original and the optimized models have been tested on two different types of AM machines – the MIP-SLA and the Fortus 360mc FDM – where the supporting structures for MIP-SLA are generated by the region subtraction method [2] and the support for FDM is generated by the software delivered together with the Fortus machine. The second example is the Armadillo model. Figure 10 shows the result of shape optimization on this example, where Armadillo ‘raises up’ its arms and ‘pulls down’ its tail. Quantitative measurements of these two examples are given in Table 1 for the purpose of comparison.

In both MIP-SLA and FDM, the material usage for the support has been greatly reduced. Because the fabrication time

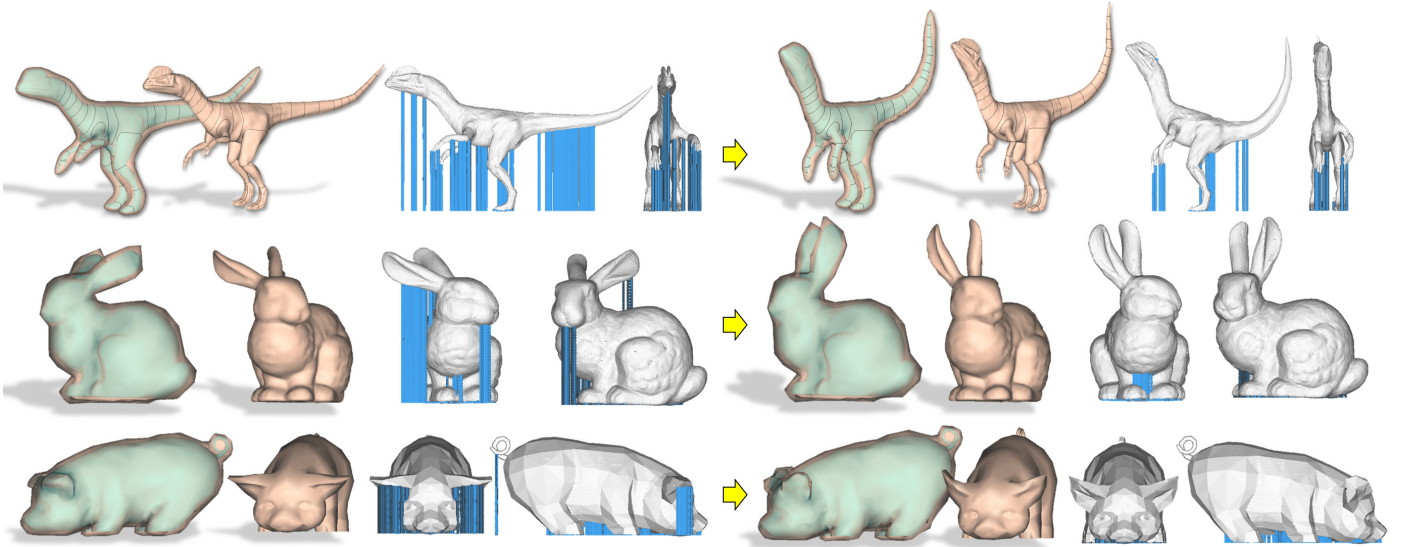


Figure 11: Examples of support slimming on more models: Dinosaur (top), Bunny (middle) and Pig (bottom). The supporting structure can be effectively reduced by applying our shape optimization approach to all of these models.

Table 1: Comparison of Fabrication Cost

	Printer	Material Usage <sup>†</sup>		Layer Num	Fab <sup>‡</sup> Time
		Main	Support		
Dino – Fig.1					
Input	MIP-SLA	11.2 g	3.2 g	717	127 m
Optm	MIP-SLA	10.9 g	0.5 g	865	153 m
Input	FDM	1.30 in <sup>3</sup>	2.12 in <sup>3</sup>	408	428 m
Optm	FDM	1.21 in <sup>3</sup>	1.07 in <sup>3</sup>	491	281 m
Armadillo – Fig.10					
Input	MIP-SLA	26.4 g	2.9 g	692	117 m
Optm	MIP-SLA	25.6 g	0.2 g	785	133 m
Input	FDM	1.47 in <sup>3</sup>	2.57 in <sup>3</sup>	389	483 m
Optm	FDM	1.38 in <sup>3</sup>	1.23 in <sup>3</sup>	440	322 m

<sup>†</sup>The material used in MIP-SLA is measured by the weight of resin and in FDM is measured by the volume of filaments.

<sup>‡</sup>The time of fabrication is reported in the unit of minutes.

Table 2: Statistics of Computational Performance

Model	Fig.	Volumetric Mesh		$\alpha_{\max}$	Time*
		Ver. #	Tet. #		
Dino	1	974	3,299	0°	4.151 s
Armadillo	10	2,373	8,189	0°	10.717 s
Dinosaur	11	4,345	16,803	10°	22.525 s
Bunny	11	439	1,536	0°	1.899 s
Pig	11	1,046	4,045	10°	5.133 s

\*The time reported in this column is the total time of 100 iterations in the unit of seconds. We assume different materials are used in the fabrication – therefore, different maximal self-supported angles,  $\alpha_{\max}$ , are used.

of FDM is proportional to the total materials usage, slimming down the support will significantly improve the efficiency of the fabrication as well. Note that this will benefit not only the FDM printing with single material but also the advanced FDM using dissolvable material in supports. The time of fabrication in MIP-SLA depends on the number of total layers of a model. Therefore, slimming down the support will not speed up the MIP-SLA printing. However, another important aspect of MIP-SLA benefited by this work is the surface quality of finished model. The supporting structures and the main model are linked by a few anchor points (AP). After removing the supports in a post-process, in place of the APs some artifacts usually appear (Fig.2). The number of APs on the optimized Dino is reduced from 61 to 18, and the number of APs on the optimized Armadillo model is reduced to 1/16 of the original input with 32 APs. Three more examples are shown in Fig.11.

We also study the performance and the convergency of the shape optimization algorithm. The statistics about the computation of different models are given in Table 2, where the total time of 100 iterations is reported. Note that the majority of the time is spent on the local steps for computing SVD and minimal rotation. The equation system of the global blending step needs only to be factorized in the first step, and the factorization can be reused in the subsequent steps where the substitution can be computed very efficiently. To further investigate the convergency of our shape optimization algorithm, we track the number of risky faces during the computation. As shown in Fig.12, the number of risky faces keeps dropping in all the examples (although this may not be monotonically).

It is obvious that having more vertices on the volumetric mesh leads to longer computing time. On the other hand, a volumetric mesh with too coarse a resolution will give a large shape approximation error. Therefore, we usually employ a mesh with less than 5k vertices, and this results in a good trade-off between speed and quality.

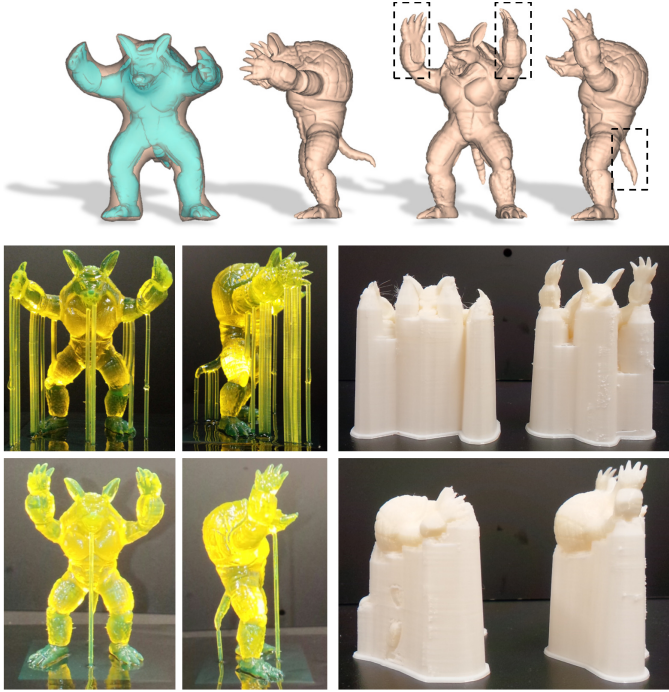


Figure 10: Example of support slimming on the Armadillo model and the comparison of AM results before vs. after the shape optimization in both MIP-SLA (bottom-left) and FDM (bottom-right). The optimization ‘raises up’ the arms and ‘pulls down’ the tail. In the pictures of MIP-SLA, the bottom row shows the optimized result; in the pictures of FDM, the right model has been optimized.

## 5.2. Extensions

Our optimization approach provides a basic tool for realizing the more interesting applications. Two examples, (1) support slimming with shape control and (2) searching for the optimal printing direction, are presented in this section.

### Shape Control in Optimization

In many cases, the designers may not wish to change the shape of a model too much during the optimization for support slimming. This can be realized by adding a penalty factor  $\beta$ , which is an angle between  $0^\circ$  and  $90^\circ$ . Specifically,  $\mathbf{L}_t = \mathbf{R}(\hat{\mathbf{f}}, \theta')\mathbf{M}$  is employed in the shape optimization as the transformation, where

$$\theta' = \max(\theta - \beta, 0). \quad (7)$$

As a result, when a large  $\beta$  is employed, many of the risky faces will not be rotated (in the cases that  $\theta$  determined by the local minimal rotation is less than  $\beta$ ). As shown in Fig.13, different values of  $\beta$  result in different degrees of deformations in the shape optimization for support slimming. In practice, the penalty factor is used together with two metrics for shape deviation:

- *Maximal Node Deviation Ratio (MNDR)*: This metric evaluates the maximal displacement at the vertices of the input mesh model  $\mathcal{M}$  with respect to the diagonal length of  $\mathcal{M}$ ’s bounding box.

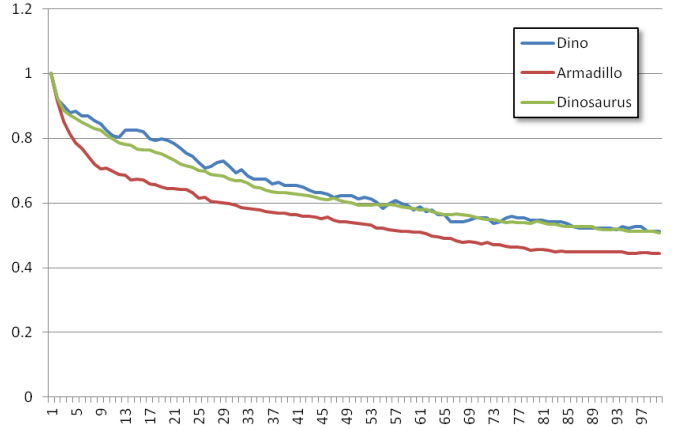


Figure 12: A chart to show the percentage of risky faces compared to the number of risky faces on the input models.

- *Average Node Deviation Ratio (ANDR)*: This measures the average displacement at all the vertices of  $\mathcal{M}$  with respect to the diagonal length of  $\mathcal{M}$ ’s bounding box.

We start from the optimization with  $\beta = 0$  and evaluate the MNDR and ANDR on the resultant model. If they are not satisfied, the value of  $\beta$  is incrementally changed until a result with a satisfactory shape deviation is obtained. In practice, designers usually require that ANDR is less than 5% or MNDR is less than 15% to control the deformation. For the dinosaur model shown in Fig.13, the desirable  $\beta$  is less than  $30^\circ$ .

### Searching for Optimal Printing Direction

The printing direction is usually chosen empirically by users in accordance with the features on an input model or simply assigned in the bottom-up direction. However, for some models such as the running Armadillo model shown in Fig.14, it is not easy to find a heuristic printing direction. Our approach can be employed in the inner loop as a tool to search for an optimal printing direction. Here the optimal printing direction is defined as a direction that leads to the minimal number of anchor points (APs) linking supports to the main model (ref. [2]). That means we wish to introduce minimal damage during the post-process to remove the supports. We iteratively rotate the input model along the  $x$ - and  $y$ -axes with a constant angle. For each new orientation, we apply the optimization approach followed by the support generation algorithm, on the result of which the number of APs is counted. Among all the discretely spanned orientations, the one with the minimal AP is finally chosen as an optimal printing direction (see Fig.14 for an example).

### 5.3. Limitation and Discussion

Although our approach can successfully optimize the shape of given models to slim down the supporting structure, it also has some limitations. First of all, our optimizer is based on the assumption that the shape of the original model can be changed. For some applications in which the original model cannot be deformed, our optimizer is not able to help. Secondly, because we set the goal of optimization only at slimming down the supporting structure, the model cannot be converted into a fully



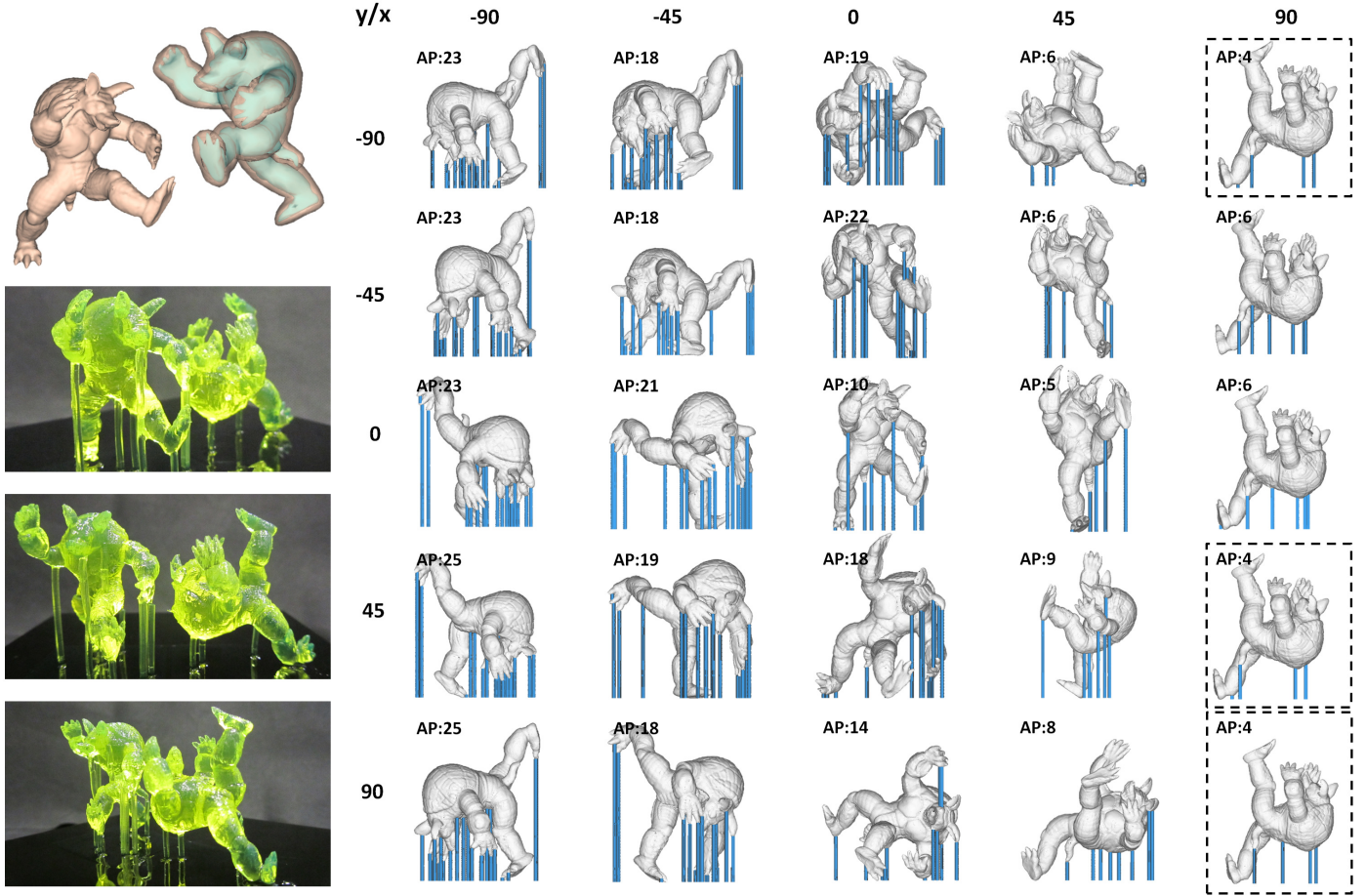


Figure 14: Our approach can be computed efficiently on freeform models. As a result, it can be used as a tool to compute the optimal shape for each printing direction. The number of APs linking supports to the main model can be used as a criterion for selecting the printing direction. Smaller AP leads to less surface damage when removing the supports in the post-process of printing. We have used the MIP-SLA to print the original model and the model with optimal direction circled by the dashed lines – see the left column of the fabricated models. Note that, in each orientation, optimization is also taken to further deform the model into an optimal shape. This is different from what can be obtained by searching with rigid transformations.

self-supported shape. On the other aspect, self-supportiveness usually needs to be considered at the very beginning of the design process. One of the related works can be found in [30]. Thirdly, our support slimming work does not consider the surface damaged on the safe faces by adding supports for the risky faces exactly above them. Lastly, the volumetric meshes which are coarser than the input mesh models are used as the computational domain in our approach to preserve the geometric details. On the downside, this also neglects the overhangs inside a safe tetrahedron. We will consider these factors by investigating a method for computing the minimal rotation in more complex configurations. Specifically, multiple faces on the input mesh  $\mathcal{M}$  inside a tetrahedron will be employed to construct the DFR on the Gauss sphere. We will study this in our future work.

The success of shape optimization on a coarser volumetric mesh  $C_{\mathcal{T}}$  relies on the error between mesh surface of the given model and the boundary of  $C_{\mathcal{T}}$ ; the smaller the better. When a very coarse mesh is employed, the ‘optimized’ shape computed from the volumetric mesh may have more risky faces before

computation. We prevent such cases by checking the number of risky faces after each iteration, and always store the best result among the past steps of iteration. How to generate a volumetric mesh bounding the input model more tightly has beyond the scope of this paper. We consider it to be a work which needs to be taken in the future.

Because the computational domain of our optimization framework is a coarse volumetric mesh, local/global self-intersection may be introduced into the volumetric mesh. A post-processing step needs to be applied to remove self-intersections in such cases (e.g., [31]).

In our current formulation, one interior vertex (i.e.,  $\mathbf{v}_4$ ) is chosen as the origin of local rotation based on a heuristic consideration. The effectiveness of selecting different origins of local rotation can be released by using the mean-subtraction strategy in global blending (ref. [32]). However, in all of our tests, the current implementation converges well when we use one of the interior vertices as the origin of local rotation.

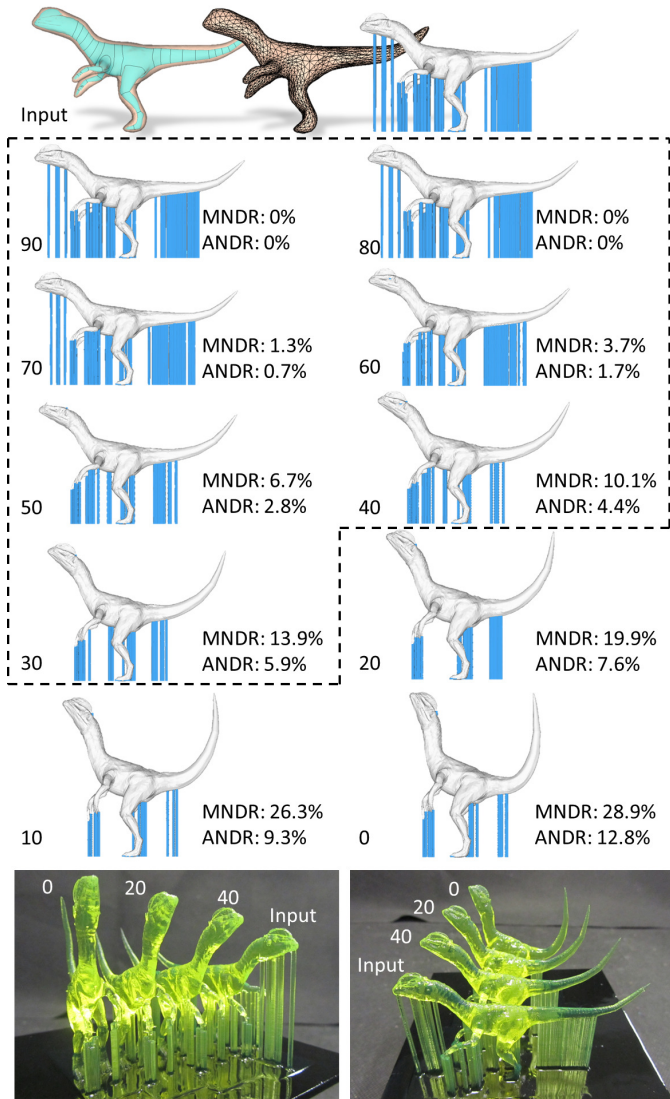


Figure 13: When different values of the penalty factor  $\beta$  are employed in the shape optimization, results with different *maximal node deviation ratio* (MNDR) and *average node deviation ratio* (ANDR) can be obtained. If less deformation is allowed in the optimization, the resultant model may need to add more supporting structures.

## 6. Conclusion

This paper presents a method for optimizing the shape of a designed model to be fabricated by layer-based AM. When only a single material can be used during the fabrication, two widely used types of AM machines, FDM and SLA, both suffer from the problems caused by the additionally inserted supporting structures. We develop an optimization approach based on a local minimal rotation and a global blending. A closed-form formulation is derived in the local step to determine the minimal rotations in different configurations. The computation of global blending can be offloaded to a factorization of the normal equation system that can be reused in the subsequent iterations. As a result, the optimization can be conducted very efficiently.

Experimental tests shown at the end of the paper have verified the effectiveness of this approach. The optimized models can serve as useful references for designers when self-supportness is an important factor to be considered.

## Acknowledgement

The research work reported in this paper was supported by the Hong Kong Innovative Technology Fund (ITS/247/11 and ITS/060/13). Ka-chun Chan built the machine for the mask-image-projection based SLA, Yuen-shan Leung implemented the support generation algorithm, and the cage-generation program was provided by Chuhua Xian. The authors would like to acknowledge their contribution in this work.

## References

- [1] P. Huang, C. C. L. Wang, Y. Chen, Algorithms for layered manufacturing in image space, in: ASME Advances in Computers and Information in Engineering Research, 2014.
- [2] Y. Chen, K. Li, X. Qian, Direct geometry processing for tele-fabrication, ASME Journal of Computing and Information Science in Engineering 13 (2013) 041002.
- [3] L. Chilson, Printing with Support - Extreme Overhangs (2012).
- [4] K. G. Swift, J. D. Booker, Manufacturing Process Selection Handbook, Elsevier Ltd., 2013.
- [5] Y. Pan, C. Zhou, Y. Chen, Rapid manufacturing in minutes: The development of a mask projection stereolithography process for high-speed fabrication, in: Proceedings of ASME 2012 International Manufacturing Science and Engineering Conference, 2012, pp. 405–414.
- [6] M. P. Bendsoe, O. Sigmund, Topology Optimization: Theory, Methods and Applications, Springer, 2003.
- [7] J. Haslinger, R. A. E. Mäkinen, Introduction to Shape Optimization: Theory, Approximation, and Computation, SIAM, 2003.
- [8] R. Prévost, E. Whiting, S. Lefebvre, O. Sorkine-Hornung, Make it stand: Balancing shapes for 3d fabrication, ACM Trans. Graph. 32 (4) (2013) 81:1–81:10.
- [9] M. Bacher, E. Whiting, B. Bickel, O. Sorkine-Hornung, Spin-it: Optimizing moment of inertia for spinnable objects, ACM Trans. Graph. 33 (4).
- [10] A. Telea, A. Jalba, Voxel-based assessment of printability of 3d shapes, in: Proceedings of the 10th International Conference on Mathematical Morphology and Its Applications to Image and Signal Processing, Springer-Verlag, 2011, pp. 393–404.
- [11] O. Stava, J. Vanek, B. Benes, N. Carr, R. Měch, Stress relief: Improving structural strength of 3d printable objects, ACM Trans. Graph. 31 (4) (2012) 48:1–48:11.
- [12] W. Wang, T. Y. Wang, Z. Yang, L. Liu, X. Tong, W. Tong, J. Deng, F. Chen, X. Liu, Cost-effective printing of 3d objects with skin-frame structures, ACM Trans. Graph. 32 (6) (2013) 177:1–177:10.
- [13] Q. Zhou, J. Panetta, D. Zorin, Worst-case structural analysis, ACM Trans. Graph. 32 (4) (2013) 137:1–137:12.
- [14] N. Umetani, R. Schmidt, Cross-sectional structural analysis for 3d printing optimization, in: SIGGRAPH Asia 2013 Technical Briefs, 2013, pp. 5:1–5:4.
- [15] L. Lu, A. Sharf, H. Zhao, Y. Wei, Q. Fan, X. Chen, Y. Savoye, C. Tu, D. Cohen-Or, B. Chen, Build-to-last: Strength to weight 3d printed objects, ACM Trans. Graph. 33 (4).
- [16] Y. Chen, Determining parting direction based on minimum bounding box and fuzzy logics, International Journal of Machine Tools and Manufacture 37 (9) (1997) 1189–1199.
- [17] A. K. Priyadarshi, S. K. Gupta, Finding mold-piece regions using computer graphics hardware, in: Geometric Modeling and Processing, 2006, Lecture Notes in Computer Science, 2006, pp. 655–662.
- [18] R. Khardekar, G. Burton, S. McMains, Finding feasible mold parting directions using graphics hardware, Comput. Aided Des. 38 (4) (2006) 327–341.

- [19] W. Li, R. Martin, F. Langbein, Molds for meshes: Computing smooth parting lines and undercut removal, *IEEE Transactions on Automation Science and Engineering* 6 (3) (2009) 423–432.
- [20] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Levy, *Polygon Mesh Processing*, AK Peters, 2010.
- [21] T. Igarashi, T. Moscovich, J. F. Hughes, As-rigid-as-possible shape manipulation, *ACM Trans. Graph.* 24 (3) (2005) 1134–1141.
- [22] O. Sorkine, M. Alexa, As-rigid-as-possible surface modeling, in: *Proceedings of the Fifth Eurographics Symposium on Geometry Processing (SGP'07)*, 2007, pp. 109–116.
- [23] L. Liu, L. Zhang, Y. Xu, C. Gotsman, S. J. Gortler, A local/global approach to mesh parameterization, *Computer Graphics Forum (Proc. Eurographics Symposium on Geometry Processing (SGP))* 27 (5) (2008) 1495–1504.
- [24] T.-H. Kwok, C. Wang, Shape optimization for human-centric products with standardized component, *Computer-Aided Design* 52 (2014) 51–63.
- [25] C. Xian, H. Lin, S. Gao, Automatic cage generation by improved obbs for mesh deformation., *The Visual Computer* 28 (1) (2012) 21–33.
- [26] L. Markosian, J. M. Cohen, T. Crulli, J. Hughes, Skin: A constructive approach to modeling free-form shapes, in: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '99*, 1999, pp. 393–400.
- [27] H. Si, Tetgen 1.5.0, <http://www.tetgen.org> (2013).
- [28] C. J. Taylor, C. J. Taylor, D. J. Kriegman, D. J. Kriegman, Minimization on the lie group  $so(3)$  and related manifolds, Tech. rep., Yale University (1994).
- [29] G. Guennebaud, B. Jacob, et al., Eigen 3.2, <http://eigen.tuxfamily.org> (2013).
- [30] E. Vouga, M. Höbinger, J. Wallner, H. Pottmann, Design of self-supporting surfaces, *ACM Trans. Graph.* 31 (4) (2012) 87:1–87:11.
- [31] C. C. L. Wang, D. Manocha, Efficient boundary extraction of BSP solids based on clipping operations, *IEEE Transactions on Visualization and Computer Graphics* 19 (1) (2013) 16–29.
- [32] S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, M. Pauly, Shape-up: Shaping discrete geometry with projections, *Comp. Graph. Forum* 31 (5) (2012) 1657–1667.