2015 IEEE International Conference on
Automation Science and Engineering (CASE)
Aug 24-28, 2015. Gothenburg, Sweden

Direct Computation of Minimal Rotation for Support Slimming

Kailun Hu, Xiaoting Zhang and Charlie C. L. Wang[†]

***Abstract*— To reduce the usage of supporting structure in additive manufacturing, an orientation-driven shape optimizer was developed in our prior work [Hu et al. 2015] which employs a volumetric mesh enclosing the input 3D models as the domain of computation. The orientation of a model is computed indirectly by surface of the volumetric mesh. In this paper, we extend our indirect computation to an approach that the computation is directly based on the information of input models by an algorithm using incremental linear programming and K-means clustering. The performance of this approach is decoupled from the shape similarity between the volumetric mesh and the input model. As a result, representations obtained from simpler volumetric decomposition such as voxels can be adopted as the domain of computation.**

I. INTRODUCTION

Additive manufacturing (AM) technology, also called 3D printing, has emerged as one of the most important approaches for realizing fast fabrication of freeform solids. *Stereolithography Apparatus* (SLA) and *Fused Deposition Modelling* (FDM) are two widely used approaches in AM as they achieve very good balance between the cost and the quality. Both SLA and FDM fabricate models in a layer-upon-layer manner (see Fig.1 for an example of SLA), where supporting structures (also simply called *support*) need to be added during the printing process. Specifically, the printing material cannot be deposited on a layer there is insufficient material on the previous layer. Overhangs with large hanged area can easily collapse under gravity. The problem is solved by adding supports to the originally designed models. When the support is fabricated by a dissolvable material that is different from the one used to fabricate the designed model, the support can be removed by a post-processing step [1]. However, there are many machines such as SLA and low-cost FDM that are only able to fabricate models with single material. The accessorial supporting structure brings in many problems, including the waste of material, the difficulty of removal, the damage of surface at connected points.

In practice, engineers always modify the parts to be fabricated into a shape be more 'self-supported'. A shape optimizer was developed in our prior work [2] to automate this step to improve the manufacturability by an iterative deformation procedure. As will be briefed in Section II, the deformation-based optimization is formulated as repeatedly applied 1) a *local* step of orientation correction to drive the deformation followed by 2) a *global* step for blending the

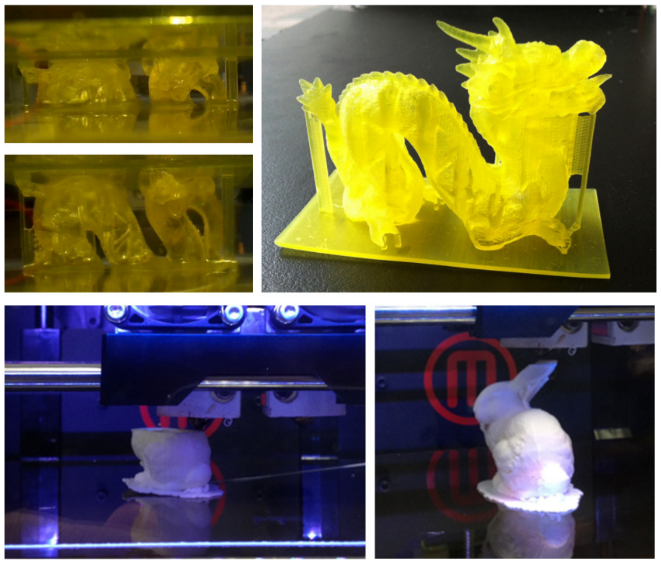
Fig. 1. Layer-based manufacturing by SLA (top) and FDM (bottom): supports must be added to avoid the collapse of overhangs.

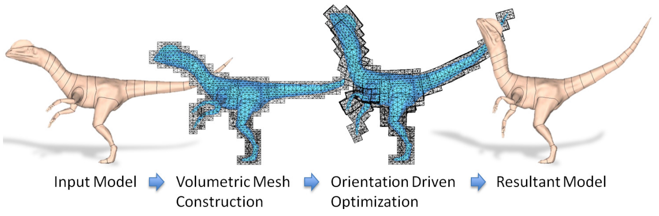

Fig. 2. Flowchart overview of our shape optimization algorithm.

oriented computation units (i.e., tetrahedra) to enforce the continuity of a deformed model. For a given mesh model $\mathcal{M}$, the local rotation applied in the local step of [2] is computed by the normal faces of tetrahedra that enclose the faces of $\mathcal{M}$. As a result, shape and quality of the volumetric mesh $\mathcal{T}$ have a direct influence on the results of optimization. When using a tetrahedral model as shown in Fig.2 that is simply obtained from the voxelization of $\mathcal{M}$, the approach presented in [2] cannot give a correct result. To overcome this limitation, we develop a new local step in this paper to re-orientate tetrahedra according to the orientations of faces on $\mathcal{M}$ – i.e., direct computation of minimal rotation.

A. Problem Statement

Given a printing direction $\hat{\mathbf{d}}_p$ and the self-supporting coefficient τ in accordance with the maximal self-supported angle $\alpha_{\max}$ as $\tau = \sin(\alpha_{\max})$, a face on $\mathcal{M}$ whose normal $\hat{\mathbf{n}}$ satisfy

*This work was partially supported by HKSAR Innovation and Technology Fund: ITS/247/11 and ITS/060/13.

All authors are with the Department of Mechanical and Automation Engineering, The Chinese University of Hong Kong, Shatin, NT, Hong Kong
[†]Corresponding Author (Email: cwang@mae.cuhk.edu.hk)

$$\hat{\mathbf{n}} \cdot \hat{\mathbf{d}}_p + \tau < 0$$

is named as *risky* face. Other faces with $\hat{\mathbf{n}} \cdot \hat{\mathbf{d}}_p + \tau \geq 0$ are considered as *safe* faces. Each of the tetrahedra containing faces of $\mathcal{M}$ are detected to see if there is any risky faces inside. If so, the tetrahedron is called risky tetrahedron; otherwise, it is safe.

What enclosed in a risky tetrahedron is a polygonal mesh $\mathcal{P} \subset \mathcal{M}$. For a tetrahedron with $\mathcal{P} = \{f_i\}$, our problem to be solved as the local step in shape optimizer is to find a minimal rotation applied to this tetrahedron to make all the risky faces safe.

$$\begin{aligned} &\arg\min_{\hat{\mathbf{r}},\theta} E(\mathbf{R}(\hat{\mathbf{r}},\theta),\mathbf{I}) \\ &s.t.,\ \forall f_i \in \mathcal{P},\ (\mathbf{R}(\hat{\mathbf{r}},\theta)\hat{\mathbf{n}}_i) \cdot \hat{\mathbf{d}}_p \geq -\tau \end{aligned} \tag{1}$$

where $\hat{\mathbf{n}}_i$ is the normal of face f_i, θ is the minimal rotation angle and $\hat{\mathbf{r}}$ is its corresponding rotation axis. Here, $E(\mathbf{R}(\hat{\mathbf{r}},\theta),\mathbf{I})$ measures how significantly $\mathbf{R}(\hat{\mathbf{r}},\theta)$ deviates from $\mathbf{I}$ in terms of rotation. This is an extension of our prior work published in [2], in which the minimal rotation computation only considered one or two faces. The problem becomes more challenge when the minimal rotation is computed by the orientations of faces in $\mathcal{P}$.

B. Contribution

The major technical contribution of this paper is a a direct minimal rotation computation approach that is based on the original mesh encoded in each volumetric element. By this direct approach, the results of orientation-driven shape optimizer are not sensitive to the shape of the volumetric computation domain. We even can use the simple tetrahedral meshes generated from a voxel-set as the domain of computation. The direct computation of minimal rotation consists of two steps:

- A novel incremental LP solver to find the minimal rotation based on the original mesh enclosed in each tetrahedron.
- An area-weighted K-means clustering method to find the proxies of original mesh enclosed in each tetrahedron.

In the first step, the more information from the original mesh we use, the more precise resultant minimal rotation we can get from the incremental LP solver. Ideally, we wish can use the normal vectors of all faces enclosed in a tetrahedron. In such scenario, if a feasible solution can be found, the result minimal rotation is exact. However, in many cases, we cannot find such an exact solution as there are too many faces encoded in one volumetric element. For these cases, an area-weighted K-means clustering method on the Gauss sphere is employed on these parts of mesh to find the optimal approximation. Our method tries to find the largest number of the representative proxies on a Gauss sphere by K-means clustering, where the minimal rotation can be find by the incremental LP solver based on these proxies.

C. Related Work

This review does not aim for completeness, but rather provides an overview of the scope of techniques that our work is related to.

Recently, shape and topology optimization techniques have been used in AM applications. Topology optimization is conducted in [3], [4] to compute interior structure of a given model to have a better mechanical property. Shape optimization is employed in [5] and [6] to make a given model have a balance and an optimal moment of inertia respectively. Moreover, an optimal printing direction that will maximise the mechanical strength of a printed model is computed in [7] by cross-sectional structural analysis. However, there is few approach available in literature considering how to optimise the shape to reduce the supporting structure. This paper presents an extension of our prior work [2] by directly computing minimal rotations based on the faces of an input model.

Following [2] , the local/global shape optimization strategy is conducted in this paper to compute an optimal shape that is more self-supported. Starting from [8], [9], many deformation approaches have been developed in the literature of computer graphics to obtain a fast converge of non-linear optimization (e.g., [10]–[12]). The novelty here is to use a new re-orientation scheme to drive deformation in the optimization framework.

The rest of this paper is organized as follows. Section II briefly introduces framework of deformation-based optimization for support slimming. The details about how to compute a minimal rotation based on the faces on the input model are presented in Section III. Experimental results and the method to preserve global features are discussed in Section IV. Lastly, our paper ends with the conclusion section.

II. DEFORMATION BASED OPTIMIZATION

The shape optimization is taken in a local/global deformation framework. The deformation of a volumetric mesh $\mathcal{T}$ enclosing the input model $\mathcal{M}$ is driven by a local rigid transformation $\mathbf{L}_t$ applied to each tetrahedron t. Without loss of generality, four vertices on a tetrahedron can be used to construct a local tensor as $\mathbf{V}_t = [\mathbf{v}_1 - \mathbf{v}_4\ \mathbf{v}_2 - \mathbf{v}_4\ \mathbf{v}_3 - \mathbf{v}_4]$. Then, the *as-rigid-as-possible* (ARAP) energy on the whole volume mesh can be defined as

$$E_{ARAP} = \sum_t w_t \|\mathbf{V}_t^{new} - \mathbf{L}_t \mathbf{V}_t^0\|_F^2, \tag{2}$$

where $\|\cdot\|_F$ is the Frobenius norm, $\mathbf{V}_t^0$ is based on the positions of vertices on in the input volumetric mesh, and w_t is the volume of t serving as the weight. Minimizing the ARAP energy defined in Eq.(2) can be converted into a least-square problem, where our formulation makes its computation's factorization re-usable. After updating the vertices of $\mathcal{T}$ in each iteration, the shape of $\mathcal{M}$ can be deformed accordingly by applying the barycentric coordinate. Details can be found in [2].

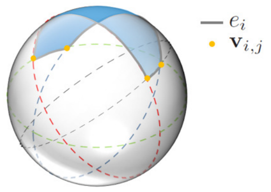

Fig. 3. An illustration for the feasible region $\mathcal{H}_t$ (see the blue region on the Gauss sphere), which is commonly determined by a set of half-spaces. Edges and corners used in the incremental LP are also shown.

The rigid transformation, $\mathbf{L}_t$, applied in our optimization framework is a cascade of the transformation $\mathbf{M}$, which simulates an elasticity to deform a tetrahedron t back into its original shape $\mathbf{V}_t^0$, and the minimal rotation $\mathbf{R}(\hat{\mathbf{r}}, \theta)$ determined by Eq.(1). The transformation $\mathbf{M}$ can be obtained by first applying a *singular value decomposition* on $\mathbf{Q} = \mathbf{V}_t^c(\mathbf{V}_t^0)^{-1}$ to obtain $\mathbf{Q} = \mathbf{U}\Sigma\mathbf{W}^T$ and then $\mathbf{M} = \mathbf{U}\mathbf{W}^T$.

The optimization to minimize the energy defined in Eq.(2) is in a least-square form $\min \|\mathbf{Ax}-\mathbf{b}\|^2$. Also, the coefficient matrix of the global least-square system, $\mathbf{A}$, is derived from $\mathbf{V}_t$ and keeps invariant during the iterations. The strategy of re-used decomposition can be employed to speed-up the computation. Specifically, the least-square problem can be determined by solving $\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{b}$. When LU-decomposition is applied, the decomposition only need to be conducted once in the first iteration. Computation in the later iterations can use the same lower and upper triangular matrices as $\mathbf{A}^T\mathbf{A}$ is invariant among all iterations.

III. COMPUTATION OF MINIMAL ROTATION

This section presents the details of our method to compute the minimal rotation of a tetrahedron t by using the faces on the input mesh model inside t – that is $\mathcal{P}$.

A. Algorithm

For each face f_i on $\mathcal{P}$, the set of possible printing directions that make f_i safe can be described as a half-space

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

For a *risky* tetrahedron t with n faces inside $\mathcal{P}$, the intersection of all $\mathcal{H}_i$s , which is denoted by $\mathcal{H}_t = \mathcal{H}_1 \cap \cdots \cap \mathcal{H}_n$, actually defines a convex polygon on the Gauss sphere. The problem defined in Eq.(1) can then be interpreted as rotating $\mathcal{H}_t$ to contain $\hat{\mathbf{d}}_p$ with a minimal rotation on the Gauss sphere if it exists. Specifically, we first quickly check whether $\mathcal{H}_t = \emptyset$. If $\mathcal{H}_t \neq \emptyset$, a minimal rotation is found to let $\hat{\mathbf{d}}_p$ be located on the boundary of $\mathcal{H}_t$, denoted by $\partial\mathcal{H}_t$. When $\mathcal{H}_t = \emptyset$, an approximation of the polygonal patch $\mathcal{P}$ inside t should be generated. The minimal rotation is computed on this approximation.

The minimal rotation can be interpreted as the closest distance from $\mathcal{H}_t$ to point $\hat{\mathbf{d}}_p$ on the Gauss sphere. Thus, we propose to use an incremental LP solver to quickly find out whether the convex polygon $\mathcal{H}_t$ exists on the Gauss sphere, meanwhile the closest point $\hat{\mathbf{c}} \in \partial\mathcal{H}_t$ to $\hat{\mathbf{d}}_p$ is reported. Note that, after obtaining the point $\hat{\mathbf{c}}$, the minimal rotation is defined as a rotation around the axis $\mathbf{r} = \hat{\mathbf{d}}_p \times \hat{\mathbf{c}}$ that makes $\hat{\mathbf{d}}_p$ be consistent with $\hat{\mathbf{c}}$. For those case that $\mathcal{H}_t = \emptyset$, we conduct an area-weighted K-means clustering on the Gauss sphere to find an approximation of $\mathcal{P}$, $\bar{\mathcal{P}}$. Then, the incremental LP is applied to determine the minimal rotation by the half-spaces defined by $\bar{\mathcal{P}}$. Suppose there are m clusters in $\bar{\mathcal{P}}$, the greater m is the more accurate approximation is obtained. Moreover, the new common region $\bar{\mathcal{H}_t}$ defined by $\bar{\mathcal{P}}$ must be feasible (i.e., $\bar{\mathcal{H}_t} \neq \emptyset$).

B. Incremental Linear Programming

Given a polygon $\mathcal{P}$ with n, its corresponding $\mathcal{H}_t$ on the Gauss sphere is a convex polygon that consists of boundary edges $\{e_i\}$ and the corner points $\{\mathbf{v}_{i,j}\}$, where a corner point $\mathbf{v}_{i,j}$ is formed by the intersections of two edges e_i and e_j (see also the illustration in Fig.3). Specifically, we can determine the corner points of $\mathcal{H}_t$ by

- computing the intersection points for any pair of two half-spaces $\mathcal{H}_i$ and $\mathcal{H}_j$ on the Gauss sphere;
- checking if the point $\mathbf{v}_{i,j}$ is in the feasible region by

$$\mathbf{v}_{i,j} \cdot \hat{\mathbf{n}}_k + \tau \geq 0 \quad (\forall f_k \in \mathcal{P}). \tag{3}$$

Only the feasible intersection points satisfying the above condition for all faces are kept as the corner points.

If no intersection point satisfies the constraints in Eq.(3), it means there is no solution for the minimal rotation problem defined in Section I-A (i.e., $\mathcal{H}_t = \emptyset$). The surface patch $\mathcal{P}$ must be approximated by m proxies with $m < n$ to compute an approximate solution for minimal rotation, which will be detailed in the section below.

When $\mathcal{H}_t \neq \emptyset$, the Geodesic distances between $\hat{\mathbf{d}}_p$ and all the corner points are computed. The nearest corner point $\mathbf{v}_c$ of $\hat{\mathbf{d}}_p$ is found. To further obtain a more accurate nearest point on $\partial\mathcal{H}_t$, we compute the closest points on the two edges forming $\mathbf{v}_c$. The real nearest point $\hat{\mathbf{c}}$ can then be determined among these three points.

Geodesic Distance: Using Euclidean distance to evaluate the distance between two points on the Gauss sphere is inappropriate. Geodesic distance is employed in our computation. For two points $\hat{\mathbf{n}}_a$ and $\hat{\mathbf{n}}_b$ on the Gauss sphere, the distance can be calculated by

$$D_g(\hat{\mathbf{n}}_a, \hat{\mathbf{n}}_b) = \arccos(\hat{\mathbf{n}}_a \cdot \hat{\mathbf{n}}_b). \tag{4}$$

C. Area-Weighted K-means Clustering

For a tetrahedron t containing many faces, $\mathcal{H}_t = \emptyset$ is often found. An area-weighted K-means clustering is then applied to approximate the normals of n faces in t by fewer number of proxies. As illustrated in Fig.4, the normals of 32 faces enclosed in a tetrahedron can be clustered into 3 regions with $\mathbf{c}_1$, $\mathbf{c}_2$ and $\mathbf{c}_3$ as centers of these regions. Here, we define the center point of a region as the region's proxy as it minimizes the approximation energy between the points in this region and the proxy. These proxies can then be employed as the input of our incremental LP to find the approximate minimal rotation.

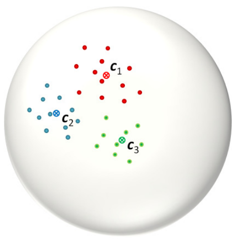

Fig. 4. An illustration of K-means clustering with 3 clusters on the Gauss Sphere: $\mathbf{c}_i$s are the proxies of the normals on the Gauss Sphere.

Area-Weighted Centroid: For a given set of points on the Gauss sphere (denoted by $\mathcal{C}$), the centroid of points in $\mathcal{C}$ is

$$\mathbf{c}_{\mathcal{C}} = \arg\min_{i \in \mathcal{C}} D_g^2(\hat{\mathbf{n}}_i, \mathbf{c}_{\mathcal{C}}).$$

To simplify the computation, an area-weighted average is employed in our implementation as

$$\bar{\mathbf{c}}_{\mathcal{C}} = \sum_{i \in \mathcal{C}} w_i \hat{\mathbf{n}}_i / \sum_{j \in \mathcal{C}} w_j \tag{5}$$

where w_i is the area of a face f_i serving as the weight. Note that, the approximate centroid computed by Eq.(5) in general has to be further normalized to serve as a proxy of $\mathcal{C}$ on the Gauss sphere.

Clustering Algorithm: For a tetrahedron encoding n faces of the original model (their normals are $\mathcal{N} = \{\hat{\mathbf{n}}_1, \hat{\mathbf{n}}_2, ..., \hat{\mathbf{n}}_n\}$ and the respective weights are $w_1, w_2, ..., w_n$), the area-weighted K-means algorithm classifies the given faces into k clusters as follows.

- Randomly select k points from $\mathcal{N}$ to serve as the seeds for clustering (i.e., the k-the random point is temporarily used as the centroid $\mathbf{c}_k$ for the cluster $\mathcal{C}_k$).
- For each point $\hat{\mathbf{n}}_i$, compute its distances to n seeds (i.e., $\mathbf{c}_k$) and assign it to the cluster of its nearest seed.
- Update the centroid of each cluster by the classification generated in the above step.
- Go back to step 2) until the terminal condition has met.

We observe a very fast converge on this K-means clustering algorithm. Moreover, a small k is usually used because we incrementally try to enlarge k until an infeasible $\mathcal{H}_t$ is found.

Termination Condition: The termination criterion is the same as what is employed in the traditional K-means cluster: i) the maximal allowed number of iterations (e.g. 100 is employed in our implementation) and ii) there is trivial change on the centroids' positions.

IV. RESULTS AND DISCUSSION

We have implemented this approach by C++ together with the Eigen library [13] as the numerical solver. All the tests are computed on a PC with Intel Core i7-3770 3.49GHz GPU and 8GB RAM. Only single thread is employed in our current implementation. Computation has been taken on volumetric meshes with up to $10k$ vertices and it can be completed within one minute on all examples.

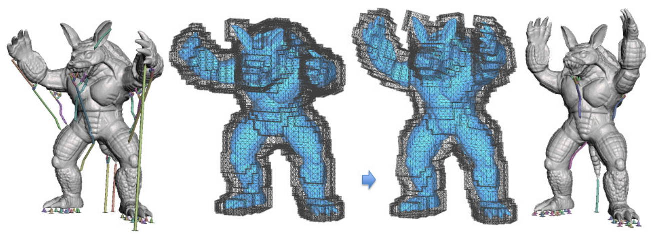

Fig. 5. Orientation-driven optimization taken on the Armadillo model to reduce the need of supporting structures: (left) before optimization (AP: 74) and (right) after optimization (AP: 42). Supporting structures generated by MeshMixer are displayed in colorful trusses.

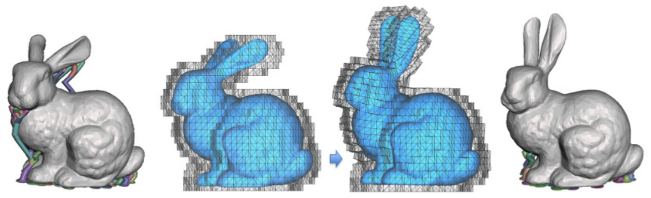

Fig. 6. Orientation-driven shape optimization of the Bunny model: (left) the input model needs to add supports with 267 APs and (right) the optimized model and its supporting structure with 189 APs.

A. Experiments

A few examples have been tested in our experiments. We have tried different support generation algorithms (ref. [14], [15]) to verify the resultant models generated by our method. Similar number of anchor points – the contact points connecting the original model and the supporting structure – can be observed. To be fair in the verification, we count the number of AP generated by the commercial software, Autodesk® MeshMixer™, as an indicator to quantify the improvement brought by using our approach.

- **Armadillo:** The Armadillo model shown in Fig.5 is optimized in the computational domain generated from voxelization. The self-supporting angle, $\alpha_{\max}$, is set as *zero* in our computation. Comparing to our prior work [2], the model generated by direct minimal rotation (i.e., the method presented in this paper) is less deformed. Local details on the input model are well-preserved by our shape optimization. The number of AP is reduced from 74 to 42 (i.e., 43.3% reduction of surface artifacts can be found).
- **Bunny:** The Bunny model shown in Fig.6 is optimized with $\alpha_{\max} = 0°$ again in the voxelized computation domain. In this example, the ears of Bunny become upright after the shape optimization, which proves the effectiveness of our direct approach. 267 APs are generated by MeshMixer on the original model, which can be reduced into 189 that is 29.2% reduction.
- **Dinosaurus:** The original and the optimized Dinosaurus models can be found in Fig.2. This model is optimized with $\alpha_{\max} = 15°$. The Dinosaurus 'rises up' its head and tail meanwhile putting down its arms after shape optimization. Geometric details can be well-preserved on the resultant model as our optimization is taken

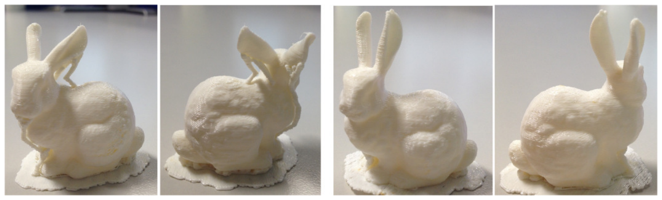

Fig. 7. Physical models fabricated by FDM using MakerBot Replicator 3D printer: (left-two) an original model and (right-two) a model after applying our shape optimization. The supporting structures are generated by MeshMixer.

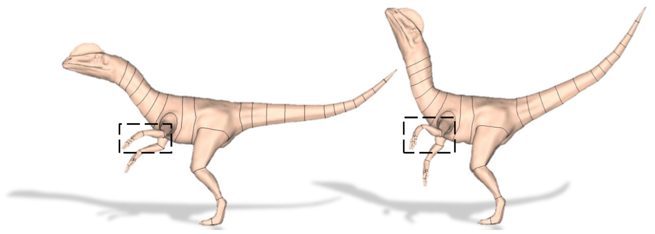

Fig. 8. Inconsistency rotation taken on neighboring tetrahedra leads to an unnatural optimization result – the left paw and the forearm of the Dinasaurus model are rotated into different directions in the local step of shape optimization: (left) the input and (right) the result.

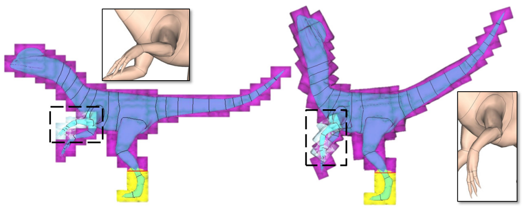

Fig. 9. The global feature can be simply selected and preserved during the shape optimization.

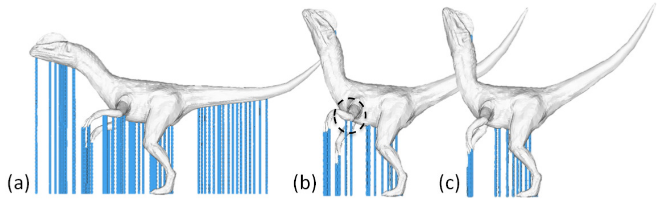

Fig. 10. The global feature can be specified and preserved in our framework: (a) the input Dinosaurus model, (b) the result without specifying global feature and (c) a better result with the whole shape of left-arm preserved. When the left-arm is bended unnaturally, more support needs to be added (see the circled region in (b)).

on the volumetric mesh and the resultant deformation is transferred to the input mesh $\mathcal{M}$ with the help of barycentric coordinates as what we did in [12].

The physical tests of fabrication using single material is taken on the MakerBot Replicator 2 FDM machine. As shown in Fig.7, the shape optimized by our method needs much less supporting structures to make the model manufacturable.

B. Feature Preservation

Our direct minimal rotation computation is based on the faces of an input model. When a very dense volumetric mesh surface is employed, this may introduce a problem of inconsistency that some neighboring tetrahedra may be driven to opposite direction by the faces inside. An unnatural result can be obtained from the optimizer (see Fig.8). This happens because the minimal rotations are computed separately on each tetrahedron without considering the coherence between neighboring regions. To overcome this problem, we introduce a tool to interactively specify and therefore automatically preserve global features on an input model.

Interactive Tool: In our framework, users are allowed to select some parts of the volumetric mesh to be computed as a whole. The minimal rotations of all tetrahedra in this selected region are enforced to be the same so that avoids the aforementioned problem of inconsistency. Specifically, a common $\mathbf{R}(\hat{\mathbf{r}}, \theta)$ is computed from all the faces on $\mathcal{M}$ falling in this region. Incremental LP is first applied. If there is no solution, we will use the K-means clustering method to find an approximate solution to serve as the common rotation.

With the help of user's input, the global feature of left-arm on the Dinosaurus model can be well preserved in the orientation-driven shape optimization. As demonstrated in Fig.9, the shape of left-arm can be preserved after selecting the related region as a global feature. Comparing to the result shown in Fig.8, the unnatural bending has been successfully removed. More than that, additional supporting structures need also to be added at the forearm (see Fig.10(b)) comparing to a better result (see Fig.10(c)) with only support at the paw. The supports are generated by the method presented in [15], which is a variant of [16]. To further verify the result, we also use MeshMixer [14] to generate supporting structures. As a result, the number of APs produced by MeshMixer is reduced from 97 on the input model to 56 on the optimized model (i.e., having 42.3% reduction) – see Fig.11.

C. About Shape Approximation Error

To quantify the accuracy of the shape approximation error produced by the area-weighted K-means clustering, a metric needs to be defined. We adopt the ratio of area that cannot be represented by our shape proxy within a certain error. Specifically, for a face with normal $\hat{\mathbf{n}}_i$, the face is defined as *well-approximated* when $\hat{\mathbf{n}}_i \cdot \hat{\mathbf{c}}_k \leq \cos(\alpha_{\max})$ with $\hat{\mathbf{c}}_k$ denoting its proxy. Otherwise, it is considered as having significant-variation from the proxy. Here, the threshold for classification is defined as value relating to the self-supporting angle as the facing-down angle within such angle can be neglected during the fabrication. The metric of shape approximation error (AE) is defined as the percentage of the summed area of faces having significant-variation in terms of the total surface area on the input model.

Keep it in mind that if we can directly get a solution from the incremental LP solver in the first step, it means all the normals are their own proxy – no approximation error is introduced. Fig.12 shows the AE of other two examples – Dinosaurus (left) and Armadillo (right) during

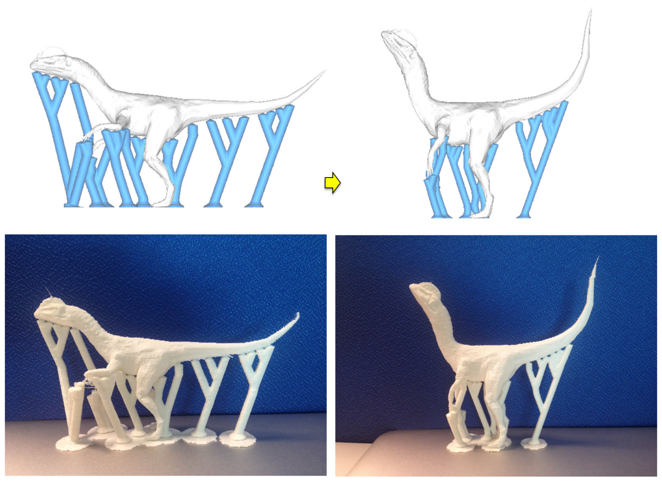

Fig. 11. Result of shape optimization on the Dinosaurus model: (left) the input model needs to add supports with 29 APs and (right) the optimized model and its supporting structure with 19 APs – that is 34.5% reduction.

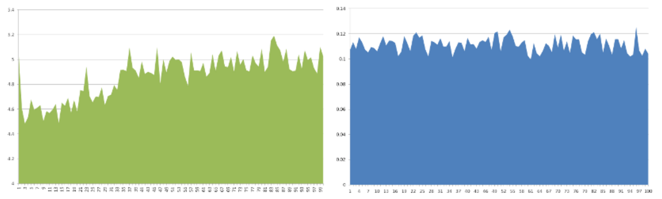

Fig. 12. Charts to show the approximation error during the iterations of shape optimization – the percentage of total area that cannot be represented by their proxies: (left) the Dinosaurus example and (right) the Armadillo example.

iterations. It can easily observed that AE is bounded during the optimization. This verifies the effectiveness of our direct computation approach. Moreover, we can also conclude that the simpler a given model is and the denser the volumetric mesh (as computation domain) is, the less approximation error can be observed. For example, shape of the Bunny model in Fig.6 is simple and the computation domain is very dense, so that the approximation error is nearly during the iterations.

V. CONCLUSION

In this paper, we present an approach based on incremental LP and K-means clustering to compute minimal rotation for a tetrahedron to reduce the need of adding support for polygonal faces inside this tetrahedron. With the help of this algorithm, the shape optimization for slimming support structure in AM can be directly driven by the orientations of faces on the input model $\mathcal{M}$. As a result, the effectiveness of optimization is less effected by the similarity of the volumetric mesh to $\mathcal{M}$. The volumetric mesh for computation can be more easily obtained – e.g., by directly splitting the voxel-set. This extension of our prior work [2] provides a very useful tool for designers to automate the optimization for manufacturability at the early stage of their design. In short, designers can use this tool to adjust the shape of his design during the whole process. This avoids the scenario that he suddenly finds at the end stage of design that his final design can only be fabricated after adding many supporting structures at the regions having high requirement of the surface quality.

ACKNOWLEDGMENT

This work is partially supported by Hong Kong RGC General Research Fund (CUHK/14207414) and Natural Science Foundation of China (Ref. No.: 61432003).

REFERENCES

[1] K. G. Swift and J. D. Booker, *Manufacturing Process Selection Handbook*. Elsevier Ltd., 2013.
[2] K. Hu, S. Jin, and C. C. L. Wang, "Support slimming for single material based additive manufacturing," *Computer-Aided Design*, vol. 65, pp. 1–10, 2015.
[3] L. Lu, A. Sharf, H. Zhao, Y. Wei, Q. Fan, X. Chen, Y. Savoye, C. Tu, D. Cohen-Or, and B. Chen, "Build-to-last: Strength to weight 3d printed objects," *ACM Trans. Graph.*, vol. 33, no. 4, 2014.
[4] A. N. Christiansen, J. A. Brentzen, M. Nobel-Jrgensen, N. Aage, and O. Sigmund, "Combined shape and topology optimization of 3D structures," *Computers & Graphics*, vol. 46, pp. 25–35, 2015.
[5] R. Prévost, E. Whiting, S. Lefebvre, and O. Sorkine-Hornung, "Make it stand: Balancing shapes for 3d fabrication," *ACM Trans. Graph.*, vol. 32, no. 4, pp. 81:1–81:10, 2013.
[6] M. Bacher, E. Whiting, B. Bickel, and O. Sorkine-Hornung, "Spin-it: Optimizing moment of inertia for spinnable objects," *ACM Trans. Graph.*, vol. 33, no. 4, 2014.
[7] N. Umetani and R. Schmidt, "Cross-sectional structural analysis for 3d printing optimization," in *SIGGRAPH Asia 2013 Technical Briefs*, 2013, pp. 5:1–5:4.
[8] T. Igarashi, T. Moscovich, and J. F. Hughes, "As-rigid-as-possible shape manipulation," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1134–1141, 2005.
[9] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Proceedings of the Fifth Eurographics Symposium on Geometry Processing (SGP'07)*, 2007, pp. 109–116.
[10] S. Bouaziz, M. Deuss, Y. Schwartzburg, T. Weise, and M. Pauly, "Shape-up: Shaping discrete geometry with projections," *Comp. Graph. Forum*, vol. 31, no. 5, pp. 1657–1667, 2012.
[11] R. Brouet, A. Sheffer, L. Boissieux, and M.-P. Cani, "Design preserving garment transfer," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 36:1–36:11, 2012.
[12] T.-H. Kwok and C. Wang, "Shape optimization for human-centric products with standardized component," *Computer-Aided Design*, vol. 52, pp. 51–63, 2014.
[13] G. Guennebaud and B. Jacob, "Eigen 3.2," http://eigen.tuxfamily.org, 2013.
[14] Autodesk, "MeshMixer 2.7," http://www.meshmixer.com, 2014.
[15] P. Huang, C. C. L. Wang, and Y. Chen, "Algorithms for layered manufacturing in image space," in *ASME Advances in Computers and Information in Engineering Research*, 2014.
[16] Y. Chen, K. Li, and X. Qian, "Direct geometry processing for telefabrication," *ASME Journal of Computing and Information Science in Engineering*, vol. 13, p. 041002, 2013.