Modeling and Fabrication with Specified Discrete Equivalence Classes

ZHONG-YUAN LIU, ZHAN ZHANG, DI ZHANG, CHUNYANG YE, LIGANG LIU, and XIAO-MING FU*, University of Science and Technology of China, China

Fig. 1. By applying our method, seven models are modeled and fabricated using nine specified discrete equivalence classes of triangles (shown in the top right corner) . They are a modern chair, the David's head, a table, a decorative lampshade, the Venus, the bunny, and the kitten model (from left to right). A real black 14-inch laptop is put on the table as a size reference.

We propose a novel method to model and fabricate shapes using a small set of specified discrete equivalence classes of triangles. The core of our modeling technique is a fabrication-error-driven remeshing algorithm. Given a triangle and a template triangle, which are coplanar and have one-to-one corresponding vertices, we define their similarity error from a manufacturing point of view as follows: the minimizer of the maximum of the three distances between the corresponding pair of vertices concerning a rigid transformation. To compute the similarity error, we convert it into an easy-to-compute form. Then, a greedy remeshing method is developed to optimize the topology and geometry of the input mesh to minimize the fabrication error defined as the maximum similarity error of all triangles. Besides, constraints are enforced to ensure the similarity between input and output shapes and the smoothness of the resulting shapes. Since the fabrication error has been considered during the modeling process, the fabrication process is easy to proceed. To assist users in performing fabrication using common materials and tools manually, we present a straightforward manufacturing solution. The feasibility and practicability of our method are demonstrated over various examples, including seven physical manufacturing models with only nine template triangles.

CCS Concepts: • **Computing methodologies → Shape modeling**.

Additional Key Words and Phrases: specified discrete equivalence classes, fabrication error, constrained remeshing

ACM Reference Format:
Zhong-Yuan Liu, Zhan Zhang, Di Zhang, Chunyang Ye, Ligang Liu, and Xiao-Ming Fu. 2021. Modeling and Fabrication with Specified Discrete Equivalence Classes. *ACM Trans. Graph.* 40, 4, Article 41 (August 2021), 12 pages. https://doi.org/10.1145/3450626.3459843

1 INTRODUCTION

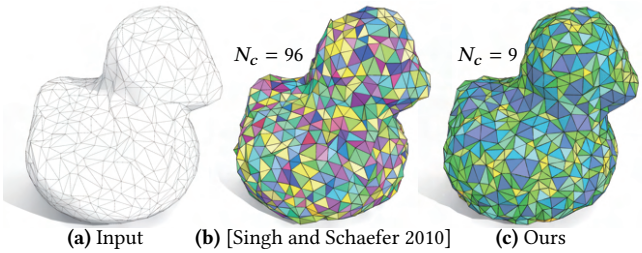
Geometric modeling is a fundamental task for many computer graphics, engineering, architecture applications. In manufacturing-related applications, modeling a shape with a small set of discrete equivalence classes of polygons offers an opportunity for reusing the templates to reduce the fabrication, storage, and construction costs. This modular modeling technique has great potential in many applications, such as freeform architecture modeling, 3D puzzles, and maintenance of surfaces of easily damaged objects.

Several methods have been proposed to tackle the problem [Eigensatz et al. 2010; Fu et al. 2010; Singh and Schaefer 2010]; however, they focus on optimizing vertex positions of the input mesh, while keeping the topology (connectivity) fixed, and automatically determining the discrete equivalence class of polygons. Since these

*The corresponding author

Authors' address: Zhong-Yuan Liu, zyliu28@mail.ustc.edu.cn; Zhan Zhang, whirlwind@mail.ustc.edu.cn; Di Zhang, zhd9702@mail.ustc.edu.cn; Chunyang Ye, yechyang@mail.ustc.edu.cn; Ligang Liu, lgliu@ustc.edu.cn; Xiao-Ming Fu, fuxm@ustc.edu.cn, University of Science and Technology of China, China.

0730-0301/2021/8-ART41 $15.00
https://doi.org/10.1145/3450626.3459843



(a) Input **(b)** [Singh and Schaefer 2010] **(c)** Ours

Fig. 2. Given a nonuniform input (a), the method of [Singh and Schaefer 2010] (b) uses much more discrete equivalence classes than our method (c) to achieve a similar fabrication error (defined in (6)). Different classes are encoded in different colors, and N_c is the number of classes. Each triangle is shown as the template after performing the best rigid transformation.

methods compute a set of templates for a model, different models need different templates. Hence before fabricating a model, template optimization, fabrication, and transportation costs are required. Moreover, they heavily rely on the initial topology and geometry; thus, it may produce a large number of discrete equivalence classes (Fig. 2), resulting in a sharp increase in fabrication cost. Nine prescribed edges and a predefined node in the Zometool system are used for modeling and fabricating shapes [Zimmer and Kobbelt 2014; Zimmer et al. 2014]. Although the Zometool-based methods implicitly guarantee triangles and convex, planar quads, they may produce a large number of template polygons.

Instead of optimizing a collection of template triangles for a single model, we study a different problem of modeling and fabricating various shapes using a given set of specified discrete equivalence classes of triangles. Accordingly, we are able to pre-manufacture those templates for various models, thereby further reducing the template optimization, fabrication, and transportation costs (Fig. 17 and 18). To effectively solve our problem, we optimize the vertex positions of the input mesh as well as alter its topology simultaneously, different from [Singh and Schaefer 2010].

Given a triangle and a template triangle, which are in a common plane and have one-to-one corresponding vertices, the similarity error is obtained by minimizing the maximum distance between the corresponding pair of vertices with respect to a rigid transformation. This definition is based on the physical manufacturing. Then, the fabrication error of a triangle mesh is the maximum similarity error of its all triangles. Consequently, we seek to modify the mesh connectivity and geometry to reduce the fabrication error to facilitate the manufacturing process. Besides, the shape similarity and shape smoothness should be ensured during the optimization process.

However, this modeling problem is very challenging. The reasons are twofold. First, it is not easy to effectively and efficiently find a predefined triangle or update the vertex positions to compute the similarity error for one output triangle. Second, adjustment of the mesh connectivity, as a combinatorial problem, may result in a tremendous searching space. Besides, as the fabrication error is the maximum value in a set of the maximum distances, it is non-trivial to optimize it during the remeshing process.

In this paper, we propose a novel algorithm to use specified discrete equivalence classes of triangles to model various shapes. Central to our modeling algorithm is a fabrication-error-driven remeshing process to reduce the fabrication error greedily. To overcome the first challenge, we convert the similarity error computation process to an easy-to-compute procedure. Then, local topological and geometric optimization can be easily used to reduce the similarity error. Consequently, our remeshing process iteratively performs local operations, including edge collapse, edge flip, and vertex relocation, to optimize the fabrication error, thereby resolving the second challenge. In practice, these operations are carefully filtered to guarantee a high degree of shape similarity. Moreover, we propose an empirical constraint to remove the local operations that cause poor smoothness.

We fabricate models using common materials and tools, including stainless steel triangle plates, stainless steel hinges, nylon cable ties with self-locking, and pliers (Fig. 8). To assemble two adjacent triangles that have a common edge, there are two steps. First, we use the plier to bend the stainless steel hinge so that its opening angle approaches the dihedral angle of the two triangles. Second, nylon cable ties are used to fix the triangle plate and hinge. This simple assembly process is iteratively performed to manufacture the entire model.

To the best of our knowledge, our system is the first to directly and explicitly use a set of specified discrete equivalence classes of triangles to model and fabricate shapes. We apply the developed modeling tool to various shapes with a set of predefined template triangles. Compared to state-of-the-art methods, our approach uses much fewer template triangles. We use nine specified types of triangles to demonstrate the feasibility and practicability of our technique on seven physical manufacturing examples.

2 RELATED WORK

Modeling using discrete equivalence classes. Modeling methods rely on different types of discrete equivalence classes, such as triangles [Huard et al. 2015; Singh and Schaefer 2010], quads [Eigensatz et al. 2010; Fu et al. 2010], triangle-based point-folding structures [Zimmer et al. 2012]. Since these methods do not modify the topology of the input mesh, the resulting discrete equivalence classes heavily depend on the shapes of the input polygons (Fig. 2). Different from them, we explicitly and directly remesh the input meshes using a set of specified template triangles. Our novel remeshing algorithm can be widely used for various models with predefined templates instead of optimizing different templates for different input models, e.g., [Singh and Schaefer 2010].

Instead of using specified triangles, the Zometool-based methods [Zimmer and Kobbelt 2014; Zimmer et al. 2014] perform remeshing using a predefined, fixed set of elements that consists of nine different strut types and a single node type. Although a set of polygonal faces are not explicitly provided by the Zometool system, the methods implicitly use the triangles and convex, planar quads. However, since they are defined as the simple closed loops of edges that are predefined, fixed in the Zometool system, their shapes and numbers cannot change. Besides, the total number of template triangles and quads is large (Fig. 22). Although our used triangles are also predefined, we can use different shapes and numbers for one model. The modular structures, such as universal building blocks [Chen et al. 2018] and Zometool construction set [Shen et al. 2020], are also used for infill fabrication to achieve cost-effective fabrication.

Fabrication error. For manufacturing, the maximum distance between two triangles should be small. To this end, minimizing the two-sided Hausdorff distance concerning a rigid transformation is an appropriate choice. However, optimization is non-trivial. Given two triangles with corresponding vertices, a ℓ_2-norm error is defined as the minimizer of the summation of squares of the distances between the corresponding vertices concerning a rigid transformation [Fu et al. 2010; Singh and Schaefer 2010]. The method in [Arun et al. 1987] is used to compute the best rigid transformation analytically. This error reduces the distances between vertices evenly. We define the fabrication error by optimizing the maximum distance between the corresponding vertices with respect to a rigid transformation. We propose a novel method to compute the best rigid transformation. Although the ℓ_2-norm error is minimized by [Singh and Schaefer 2010], the termination condition is that our defined error is less than a specified threshold. It is inconsistent with their optimization goal.

Triangular remeshing. Many triangular surface remeshing techniques have been proposed in the last two decades (cf. the surveys in [Alliez et al. 2008; Botsch et al. 2010]). Different applications require different properties. There are many requirements, such as generating fairly regular triangles [Jakob et al. 2015; Yan et al. 2009], maximizing the minimum angle [Hu et al. 2017; Wang et al. 2018], bounding the approximation errors [Cheng et al. 2019; Wang et al. 2020], and matching the input Riemannian metric [Fu et al. 2014; Zhong et al. 2013]. The fabrication error drives our remeshing process. Since the fabrication error appears on a certain triangle, the local remeshing operations [Botsch and Kobbelt 2004; Hoppe et al. 1993] are performed on that triangle to reduce it effectively.

3 METHOD

Problem. Given a triangular mesh $\mathcal{M}$ and K specified types of 2D triangles $\mathcal{T} = \{\mathbf{t}_1, \cdots, \mathbf{t}_K\}$, our goal is to generate a triangular mesh $\mathcal{R} = (\mathcal{V},\ \mathcal{F})$ satisfying three requirements:

- *Closeness requirement*: each triangle in $\mathcal{F} = \{\mathbf{f}_i\}$ is as close as possible to one triangle in $\mathcal{T}$.
- *Approximation requirement*: $\mathcal{R}$ lies in the envelope of thickness $\epsilon_{\text{envelope}}$ built around $\mathcal{M}$, where $\epsilon_{\text{envelope}}$ is a user-defined threshold.
- *Smoothness requirement*: there are no flipped or zigzagged triangles in $\mathcal{R}$.

Formulation. The generation of $\mathcal{R}$ is a remeshing process of $\mathcal{M}$, including topological and geometric modifications. It can be formulated as a constrained optimization problem:

$$\begin{aligned} \min_{\mathcal{V},\,\mathcal{F}} \quad & d_{\text{fab}}(\mathcal{R}, \mathcal{T}) \\ s.t. \quad & h(\mathcal{R}, \mathcal{M}) \le \epsilon_{\text{envelope}}, \\ & g_{\text{smooth}}(\mathcal{R}) < 0, \end{aligned} \tag{1}$$

where $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ measures the closeness from the triangles in $\mathcal{R}$ to the template triangles, $h(\mathcal{R}, \mathcal{M})$ denotes the one-sided Hausdorff distance from $\mathcal{R}$ to $\mathcal{M}$, and the constraint $g_{\text{smooth}}(\mathcal{R}) < 0$ indicates that there are no flipped or zigzagged triangles in $\mathcal{R}$.

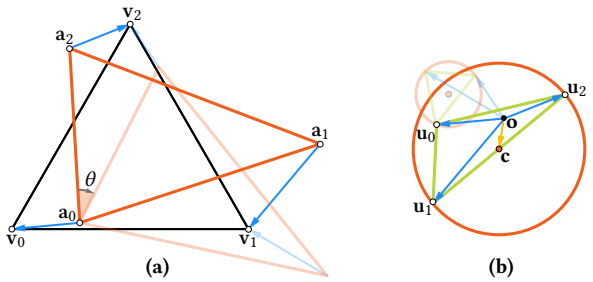


Fig. 3. Best rigid transformation to compute $d_{\max}(\mathbf{f}, \mathbf{t})$. (a) $\triangle\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ is a triangle of $\mathcal{R}$ and $\triangle\mathbf{a}_0\mathbf{a}_1\mathbf{a}_2$ indicates the template triangle $\triangle\mathbf{p}_0\mathbf{p}_1\mathbf{p}_2$ after a rigid transformation $R, \mathbf{b}$. (b) Move the starting points of the three vectors $(\mathbf{v}_0 - \mathbf{a}_0, \mathbf{v}_1 - \mathbf{a}_1, \mathbf{v}_2 - \mathbf{a}_2)$ to the origin $\mathbf{o}$. The orange circle is the MCC of $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ and $\mathbf{c}$ is its center. Here $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ is an obtuse triangle and $\mathbf{c}$ is the midpoint of its longest edge. The transparent figures show another case with a different rotation, where $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ is an acute triangle and the MCC is its circumcircle.

3.1 Fabrication error

Similarity metric. To facilitate the definition of the closeness metric from an output triangle to the predefined templates set $\mathcal{T}$, we first define the similarity metric between two triangles from the perspective of physical manufacturing. Given an output triangle $\mathbf{f} = \triangle\mathbf{v}_0\mathbf{v}_1\mathbf{v}_2$ and a specified triangle $\mathbf{t} = \triangle\mathbf{p}_0\mathbf{p}_1\mathbf{p}_2$, we assume the two triangles are coplanar and $\mathbf{v}_i$ corresponds to $\mathbf{p}_i$, for simplicity. Then, without loss of generality, $\mathbf{f}$ and $\mathbf{t}$ can be considered to be in the 2D plane. The similarity error for $\mathbf{f}$ and $\mathbf{t}$ is defined by minimizing the maximum distance between the corresponding vertices:

$$d_{\max}(\mathbf{f}, \mathbf{t}) = \min_{R,\,\mathbf{b}} \max_{i \in \{0,1,2\}} \{\|R\mathbf{p}_i + \mathbf{b} - \mathbf{v}_i\|_2\}, \tag{2}$$

where $R, \mathbf{b}$ represent a 2D rigid transformation.

3.1.1 Best rigid transformation. To compute the best rigid transformation to obtain $d_{\max}(\mathbf{f}, \mathbf{t})$, we integrate the three position deviations into a vector $\xi := (\|\mathbf{v}_0 - \mathbf{a}_0\|_2, \|\mathbf{v}_1 - \mathbf{a}_1\|_2, \|\mathbf{v}_2 - \mathbf{a}_2\|_2)$, where $\mathbf{a}_i = R\mathbf{p}_i + \mathbf{b}$. Then the optimization problem can be converted to the following formulation:

$$\{R,\ \mathbf{b}\} = \arg\min_{R,\,\mathbf{b}} \|\xi\|_\infty, \tag{3}$$

Computing $\mathbf{b}$. Move the starting points of the three vectors, i.e., $(\mathbf{v}_0 - \mathbf{a}_0, \mathbf{v}_1 - \mathbf{a}_1, \mathbf{v}_2 - \mathbf{a}_2)$ to the origin $\mathbf{o}$. Let $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ be the triangle formed by tail endpoints of the three vectors, and C be the minimum covering circle (MCC) of $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$.

Proposition 1. *The shape of $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ is independent of the translation $\mathbf{b}$, and so do C. Given any R, the best $\mathbf{b}$ minimizing $\|\xi(R)\|_\infty$ is the vector from the origin $\mathbf{o}$ to the center $\mathbf{c}$ of C, and the minimum $\|\xi(R)\|_\infty$ is the radius r_c of C.*

We prove Prop. 1 in the supplementary material. Based on Prop. 1, the optimization problem of $\min_{R,\,\mathbf{b}} \|\xi\|_\infty$ is equivalent to $\min_R r_c$. After solving $\min_R r_c$, we compute $\mathbf{b} = \mathbf{c} - \mathbf{o}$.

Computing r_c. The methods to compute r_c are different for acute triangles and obtuse triangles:

- *Acute case*: MCC is the circumcircle of $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$, thus we have: $r_c = \frac{\|\mathbf{u}_1 - \mathbf{u}_0\|_2 \cdot \|\mathbf{u}_2 - \mathbf{u}_1\|_2 \cdot \|\mathbf{u}_0 - \mathbf{u}_2\|_2}{4\text{Area}(\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2)}$.
- *Obtuse case*: r_c is the half of the longest side of $\triangle\mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$.

Acute case. To solve $\min_R r_c$ for the acute case, r_c can be reformulated as a function $f(\theta)$ of θ, where θ is the rotation angle of R. We solve $\min_\theta f(\theta)$ using the following two steps: (1) compute the roots of a tenth degree polynomial and (2) compare the function value at the ten roots to find the minimum (see details in the supplementary material).

Obtuse case. To solve $\min_R r_c$ for the obtuse case, we propose the following proposition to simplify the calculation process:

Proposition 2. *The longest edge of $\triangle \mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ corresponds to an edge of $\mathbf{f}$ (denoted as $\mathbf{e}_f$) and an edge of $\mathbf{t}$ (denoted as $\mathbf{e}_t$), respectively. Then, if $\triangle \mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ is an obtuse triangle when $\|\xi\|_\infty$ reaches the minimum, then $\mathbf{e}_f$ and $\mathbf{e}_t$ coincide and their midpoints coincide.*

We provide the proof of Prop. 2 in the supplementary material. To make the corresponding edges of $\mathbf{t}$ and $\mathbf{f}$ coincide and their midpoints coincide, there are three rigid transformations. We denote Γ_{con} as a set to include these three rigid transformations. Let Γ_{obtuse} be a rigid transformation set, each of which makes $\triangle \mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ be an obtuse triangle. For a rigid transformation $R, \mathbf{b}$ in Γ_{obtuse}, we define $i_{R,\mathbf{b}} \neq j_{R,\mathbf{b}} \in \{0,1,2\}$ as the indices of the longest edge $\mathbf{u}_{i_{R,\mathbf{b}}}\mathbf{u}_{j_{R,\mathbf{b}}}$ of the obtuse triangle $\triangle \mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$. By Prop. 2, if $\triangle \mathbf{u}_0\mathbf{u}_1\mathbf{u}_2$ is an obtuse triangle, and no corresponding edges of $\mathbf{t}$ and $\mathbf{f}$ coincide and their midpoints coincide, then $\|\xi\|_\infty$ does not reach the minimum. Thus, the elements in $\Gamma_{\text{con}} \cap \Gamma_{\text{obtuse}}$ are the potential rigid transformations to achieve the minimum r_c.

Computing R by solving $\min_R r_c$. Based on the above analyses, the optimization problem $\min_R r_c$ is converted to the following problem:

$$\min\{\min_\theta f(\theta), \min_{R,\mathbf{b}\in\Gamma_{\text{con}}\cap\Gamma_{\text{obtuse}}} \frac{1}{2}\|\mathbf{u}_{i_{(R,\mathbf{b})}} - \mathbf{u}_{j_{(R,\mathbf{b})}}\|_2\}. \tag{4}$$

In practice, we first solve $\min_\theta f(\theta)$ and $\min_{R,\mathbf{b}\in\Gamma_{\text{con}}\cap\Gamma_{\text{obtuse}}} \frac{1}{2}\|\mathbf{u}_{i_{(R,\mathbf{b})}} - \mathbf{u}_{j_{(R,\mathbf{b})}}\|_2$ separately, and then compute the smaller one to obtain the resulting rotation R.

3.1.2 Definition of fabrication error.

Matching error. If the correspondence between vertices of $\mathbf{f}$ and $\mathbf{t}$ are not specified, we should enumerate all correspondences. Similar to [Singh and Schaefer 2010], there are six correspondences between the vertices, denoted as $\{\phi_1, \cdots, \phi_6\}$. If the template triangle $\mathbf{t}$ is not determined, then all of the specified templates should be enumerated. Combining these two situations, the matching error for an output triangle $\mathbf{f}$ is defined as:

$$d_{\text{match}}(\mathbf{f}, \mathcal{T}) = \min_{\substack{\mathbf{t}\in\mathcal{T} \\ j\in\{1,\cdots,6\}}} d_{\max}(\mathbf{f}, \mathbf{t}^{\phi_j}), \tag{5}$$

where $\mathbf{t}^{\phi_j}$ possesses the same shape as $\mathbf{t}$, and the vertex correspondences between $\mathbf{f}$ and $\mathbf{t}^{\phi_j}$ are specified by the index map ϕ_j. If $\mathbf{f}$ is in 3D space, we isometrically flatten $\mathbf{f}$ onto the 2D plane before computing $d_{\text{match}}(\mathbf{f}, \mathcal{T})$, and without the loss of generality, the flattened triangle is still denoted as $\mathbf{f}$.

Fabrication error. When performing real fabrication, the triangle with the largest matching error is the worst case. Thus, the fabrication error for the output mesh $\mathcal{R}$ is defined as:

$$d_{\text{fab}}(\mathcal{R}, \mathcal{T}) = \max_{\mathbf{f}\in\mathcal{F}} d_{\text{match}}(\mathbf{f}, \mathcal{T}). \tag{6}$$

ALGORITHM 1: Modeling with Specified Discrete Equivalence Classes

Input : 3D triangular mesh $\mathcal{M}$, K predefined template triangles, and the distance bound $\epsilon_{\text{envelope}}$
Output : A remeshed mesh $\mathcal{R}$

```
R ← PreProcess(M);
Initialize k = 0; converged = false;
//Q is a priority queue of triangles sorted by matching errors;
for each face f in F do
    compute the matching error d_match(f, T) via Eq.(5);
    Q ← f with d_match(f, T);
end
while !converged do
    lr_state ← true;
    while lr_state do
        c_state ← true; f_state ← true;
        while c_state or f_state do
            f_t ← POPTriWithMaximumError(Q);
            c_state ← Collapse(f_t);
            if !c_state then
                f_state ← Flip(f_t);
            end
            UpdatePriorityQueue(Q);
        end
        lr_state ← LocalPerturbation(f_t);
        UpdatePriorityQueue(Q);
    end
    GlobalRelocation(R);
    UpdatePriorityQueue(Q);
    converged ← (topology fixed and Σ_{v_i∈V} ||v_i^{k+1} − v_i^k||_2 ≤ 10^{-4});
    k ← k + 1;
end
```

3.2 Challenges and methodology

Challenges. There are two challenges to solve the problem (6). First, since the topology of $\mathcal{R}$ is a combinatorial variable, effectively changing the topology to reduce $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ is difficult. Second, $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ is numerically difficult to optimize. To compute $d_{\text{match}}(\mathbf{f}, \mathcal{T})$, it is necessary to use enumeration and use numerical methods to solve (4). It indicates that it is easy to evaluate $d_{\text{match}}(\mathbf{f}, \mathcal{T})$ and $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ and the optimization of $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ is difficult. As a consequence, it is non-trivial to update geometry and topology to minimize $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$.

Key idea. Our key idea is to keep reducing the fabrication error $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ using local topological and geometric operations, instead of global optimization of geometry and topology. We observe that since the fabrication error $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ is the worst case on a certain triangle, the local optimization operations performed on that triangle are effective to reduce $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$. The local operations include edge collapse and edge flip for topological modification, and vertex relocation for geometric update.

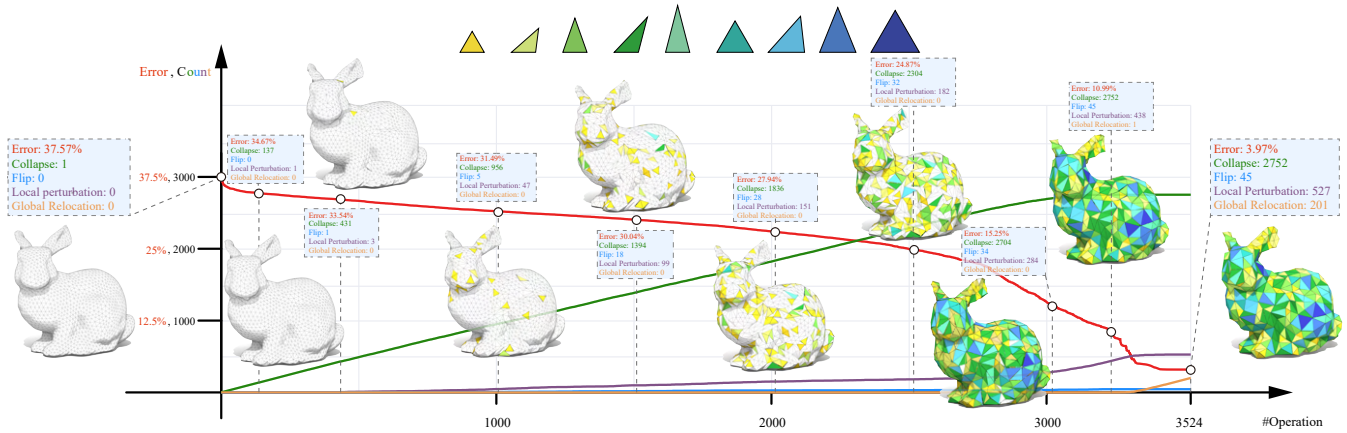


Fig. 4. The graph plots the fabrication error and each operation count vs. the number of all operations. The red, green, blue, purple and orange curves denote the fabrication error, collapse, flip, local perturbation and global relocation, respectively. The fabrication error is the percentage of the minimum length of the template edges (denoted as l_{min}). Ten progressive remeshing results are shown, and their fabrication error and operation counts are reported in the boxes. Nine template triangles are shown in the middle with different colors. If the assembly error is larger than $5\%l_{\text{min}}$ for a triangle in a remeshing result, we show it in white; otherwise, we show it using its closest template triangle.

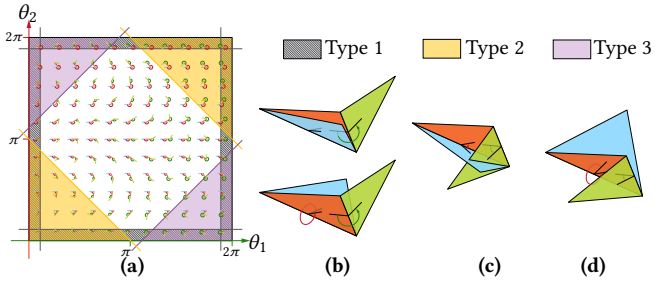


Fig. 5. Smoothness constraints. The white region is feasible (a). We show the typical examples for the first (b), second (c), and third constraints (d).

Workflow. According to our key observation, we propose a practical algorithm to greedily reduce the fabrication error $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ while not violating the approximation and smoothness constraints (see the pseudocode in Alg. 1). Since the input mesh quality and resolution may vary greatly, we first resample the input to a mesh whose edges lengths are less than half of the shortest edge length of the specified templates (Sec. 3.3.1). Then, the local topological operations, including edge collapse and edge flip, are used to change the topology to decrease the fabrication error $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ (Sec. 3.3.2). Since the solution space explored by the topological operations is limited, we further propose local perturbation and global relocation for optimization (Sec. 3.3.3).

3.3 Optimization details

Approximation constraints. The method of [Wang et al. 2020] is used to check whether the approximation constraint $h(\mathcal{R}, \mathcal{M}) \leq \epsilon_{\text{envelope}}$ is satisfied. If the approximation constraint is satisfied after performing one local operation, this operation is accepted; otherwise, we reject the operation.

Smoothness constraints. The dihedral angles between neighbor triangles are used to formulate the smoothness constraints. In detail, for a triangle strip with three triangles, we enforce a set of constraints over the two dihedral angles θ_1, θ_2 (Fig. 15 (a)). The constraint set contains three types:

(1) To avoid two adjacent triangles with nearly opposite normals, we constrain $\theta_{\text{low}} < \theta_i < \theta_{\text{high}}, i = \{1, 2\}$ (Fig. 15 (b)).
(2) Two triangles that share the same neighbour triangle are not allowed to intersect each other, $\theta^c_{\text{low}} < \theta_1 + \theta_2 < \theta^c_{\text{high}}$ is enforced (Fig. 15 (c)).
(3) A zigzag formed by the triangle strip should be prohibited, thus we set $|\theta_1 - \theta_2| < \theta^z$ (Fig. 15 (d)).

For a triangle, its any two different adjacent triangles and itself form a triangle strip. We traverse all triangles to find all strips of this type to form a strip set. The smoothness constraint $g_{\text{smooth}}(\mathcal{R}) < 0$ includes the smoothness constraints on all strips in that set. In practice, we set $\theta_{\text{low}} = \pi/18$, $\theta_{\text{high}} = 35\pi/18$, $\theta^c_{\text{low}} = \pi$, $\theta^c_{\text{high}} = 3\pi$, and $\theta^z = 10\pi/9$.

3.3.1 Initialization. Since the input triangles may deviate significantly from the prescribed template triangles, we split the edges until each edge length is shorter than half of the templates' shortest edge length. The approximation constraint is not violated after conducting the split operations. If the input mesh violates the smoothness constraint, we smooth it as input via Laplacian smoothing.

3.3.2 Topological optimization. Each time we locally adjust the topology around the triangle with the maximum matching error.

Edge collapse. Given the triangle $\mathbf{f}_t$ with maximum matching error, i.e., $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$, there are six candidates for collapse operation, i.e., two opposite collapse directions for an edge. We use the following steps to implement the collapse operation:

(1) Filter out the candidate collapse operations that lead to a non-manifold mesh or cause $\mathcal{R}$ to violate the smoothness constraints or to not meet the approximation constraint.
(2) Virtually perform each remaining candidate and record the fabrication error for the new mesh $\widehat{\mathcal{R}}$.
(3) Find the minimum value (denoted as $\widehat{d}_{\text{fab}}(\widehat{\mathcal{R}}, \mathcal{T})$) of the recorded errors and record the corresponding collapse operation.

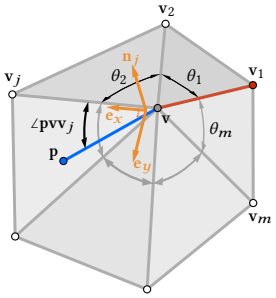
(4) If $\widehat{d}_{\text{fab}}(\widehat{\mathcal{R}}, \mathcal{T}) > d_{\text{fab}}(\mathcal{R}, \mathcal{T})$, no collapse operation is performed and set c_state = false; otherwise, the corresponding collapse operation is conducted and set c_state = true.

Edge flip. Given the triangle $\mathbf{f}_t$ with the maximum matching error, its each edge has a candidate flip operation. The steps to perform the flip operation are similar to the collapse operation. We set f_state = true if the flip operation succeeds to reduce the fabrication error, and the smoothness constraint and the approximation constraint are still satisfied; otherwise, f_state = false.

3.3.3 Geometric optimization.

Local perturbation. Since the space explored by the topological operations is only a finite discrete set, it easily gets stuck in the local minimum. Thus, when the topological operations stops to reduce the fabrication error, we enlarge the solution space by perturbing vertices locally to further reduce the fabrication error.

A probability-guided perturbation method is used to update the geometry of the triangle $\mathbf{f}_t$ with the maximum matching error. For each vertex $\mathbf{v}$ of $\mathbf{f}_t$, we denote the one-ring vertices of vertex $\mathbf{v}$ as $\{\mathbf{v}_1, \cdots, \mathbf{v}_m\}$ and the angles of the incident triangles at $\mathbf{v}$ as $\{\theta_1, \cdots, \theta_m\}$. Then, we parameterize a point $\mathbf{p}$ in the one-ring of $\mathbf{v}$ as follows:



$$\mathbf{p} = \mathbf{v} + r\cos(\theta - \sum_{i=0}^{j-1}\theta_i)\mathbf{e}_x + r\sin(\theta - \sum_{i=0}^{j-1}\theta_i)\mathbf{e}_y, \tag{7}$$

where $\theta = \sum_{i=0}^{j-1}\theta_i + \angle\mathbf{pvv}_j \in [\sum_{i=0}^{j-1}\theta_i, \sum_{i=0}^{j}\theta_i)$, $r = \|\mathbf{p}-\mathbf{v}\|$, $\mathbf{e}_x = \frac{\mathbf{v}_j-\mathbf{v}}{\|\mathbf{v}_j-\mathbf{v}\|}$, and $\mathbf{e}_y = \mathbf{n}_j \times \mathbf{e}_x$. Here $\theta_0 = 0$, $\mathbf{n}_j$ is the normal of the j^{th} triangle, and $j \in \{1, \cdots, m\}$. We sample a point as follows:

(1) Sample an angle θ uniformly in $[0, \sum_{i=0}^{m}\theta_i)$ and a radius r using a Gaussian distribution $\mathcal{N}(0, \sigma^2)$, where σ is set as $\frac{1}{7}$ times the average length of the one-ring edges of $\mathbf{v}$.
(2) Use the angle θ and the radius r to obtain a 3D point $\mathbf{x}$ via (7).
(3) Project $\mathbf{x}$ back to the input mesh $\mathcal{M}$. Denote the normal of the triangle, where the projection point is located, as $\mathbf{n}$.
(4) Sample a value η evenly between -1/2 and 1/2 and obtain the final sampling point as $\mathbf{x} + \eta\epsilon_{\text{envelope}}\mathbf{n}$.

In practice, we sample N_{sample} new triangles for $\mathbf{f}_t$ and filter out triangles that violate the smoothness constraint or the approximation constraint. Then, we compute the fabrication error for each remaining triangle and then find the minimum fabrication error. If the minimum fabrication error is less than the fabrication error before the local perturbation, we conduct the corresponding local perturbation and set lr_state = true. Otherwise, we set lr_state = false and keep $\mathbf{f}_t$ unchanged. In our experiments, $N_{\text{sample}} = 2000$.

Global relocation. For each triangle in $\mathcal{R}$, we compute the template triangle $\mathbf{t}$ and the best rigid transformation $R, \mathbf{b}$ by solving (5) before updating the geometry of $\mathcal{R}$. Then, each vertex $\mathbf{v}_i$ of $\mathcal{R}$ is updated by performing the following steps (Fig. 6):

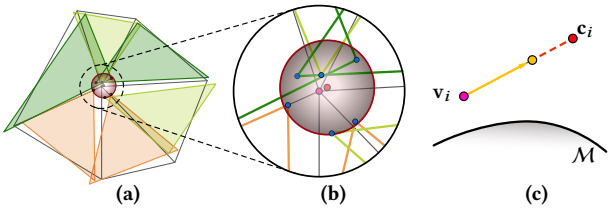


Fig. 6. Global relocation for the vertex $\mathbf{v}_i$. (a) The one-ring triangles of the vertex $\mathbf{v}_i$ and their corresponding templates. Color encodes the type of template. (b) The point set $S(\mathbf{v}_i)$ consists of the blue points, which are the corresponding vertices of the vertex $\mathbf{v}_i$ (magenta) on templates. The grey sphere indicates the minimum bounding sphere of $S(\mathbf{v}_i)$ and the red point $\mathbf{c}_i$ is the center of the sphere. (c) A backtracking line search is used to relocate $\mathbf{v}_i$ while not violating the approximation and smoothness constraints.

(1) Collect a point set $S(\mathbf{v}_i)$ that consists of the templates' vertices corresponding to $\mathbf{v}_i$;
(2) Compute a moving direction $\mathbf{d} = \mathbf{c}_i - \mathbf{v}_i$, where $\mathbf{c}_i$ is the center of the minimum bounding sphere of $S(\mathbf{v}_i)$;
(3) Update $\mathbf{v}_i$ to $\mathbf{v}_i + \alpha\mathbf{d}$, where $0 \le \alpha \le 1$ is determined by a backtracking line search procedure to guarantee that the approximation and smoothness constraints are satisfied.
(4) Solve the template triangle and the best rigid transformation for each triangle in the one-ring of $\mathbf{v}_i$.

In practice, we update each vertex only once and use the updated positions of the previously processed vertices when treating a new point. We propose the following proposition to prove that the fabrication error is guaranteed to decrease after the global relocation.

Proposition 3. *For each vertex $\mathbf{v}_i$, the maximum assembly error on the one-ring triangles (denoted as Ω_i) of $\mathbf{v}_i$ is:*

$$d(\alpha_i) = \max_{\mathbf{f}\in\Omega_i} d_{assembly}(\mathbf{f}) = \max_{\mathbf{f}\in\Omega_i} \min_{\substack{\mathbf{t}\in\mathcal{T} \\ j\in\{1,\cdots,6\}}} d_{max}(\mathbf{f}, \mathbf{t}^{\phi_j}) \tag{8}$$

where $0 \le \alpha_i \le 1$ is the step size. Then, $d(\alpha_i)$ monotonically decreases with respect to α_i.

The proof is provided in the supplementary material.

Complementarity. The two relocation methods are not in conflict. The local perturbation is to enlarge the solution space and prevent our algorithm from falling into an early trap (Fig. 16). After performing many local relocation steps, the algorithm is trapped by a local minimum, and the local perturbation cannot effectively reduce the fabrication error (see Fig. 4 and the supplementary video). Then, the global relocation is applied to reduce the fabrication error further. They complement each other in our pipeline.

4 FABRICATION

Physical assembly. After the optimization process terminates, the template triangle for each output triangle is determined. Then, we assemble the determined template triangles to form the resulting shape during the real manufacturing process. The core challenge of physical fabrication is to fix adjacent template triangles. We use hinges as connectors between adjacent triangles and resolve the challenge as follows (Fig.7 (c)):

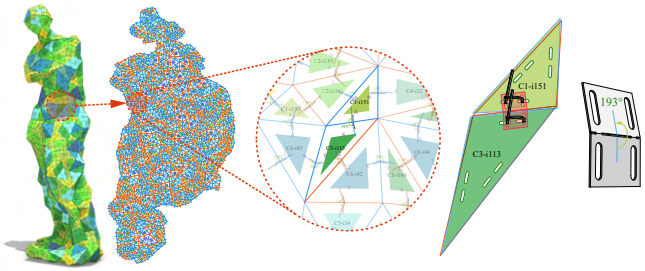


Fig. 7. (a) The digital assembly model is generated via mapping the templates to the corresponding output triangles using the best rigid transformations. The color encodes the type of templates. (b) The parameterization of the cut $\mathcal{R}$ attached with auxiliary information is used to assist the real fabrication. The zoom-in view shows the extra information, including the orientation (blue for convex and orange for concave), the dihedral angles, and the type of templates. (c) The detailed assembly process of two adjacent triangles and the bent hinge.

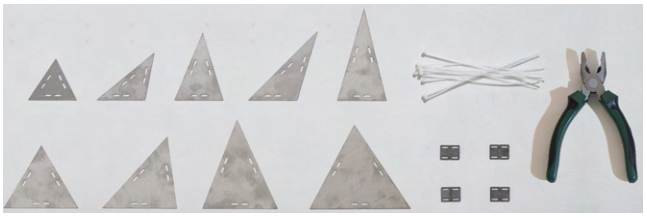


Fig. 8. Real materials and tools used in the real fabrication, including nine stainless steel triangle plates, nylon cable ties with self-locking, stainless steel hinges, and pliers.

(1) Adjust the opening angle of the hinge to match the dihedral angle of the adjacent triangles.
(2) Use ties to bind the hinges and triangles together.

Then, we assemble the templates one by one based on this strategy. To assist the fabrication, we print the parameterization with additional auxiliary information (Fig.7), including the placement and type of templates, the orientation, and the dihedral angles. The cut is generated by [Zhu et al. 2020] and the parameterization is computed by [Su et al. 2020]. See more details in the supplementary video.

Real materials and tools. To support adaptive adjustment of the opening angle of the hinge, we use stainless steel hinges and modify the opening angle by pliers. The template triangles are stainless steel triangle plates in practice. We use nylon cable ties with self-locking for binding. The real materials and tools are shown in Fig. 8. We should punch holes in the hinge and the plate to bind the hinge and plate together with a tie. In practice, two assembly holes are sufficient to fix them.

4.1 Computing holes

Holes. We can not precisely assemble all hinges and plates by only two points due to the fabrication error. Thus, we use two rounded rectangular holes for self-adaptation. The sizes and positions of the holes are related to the fabrication error. To ensure reusability, hinges have the same holes, and so do each template plate.

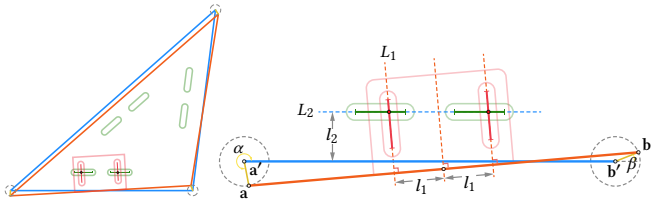


Fig. 9. The design of the rectangular holes on templates and hinges. Left: An output triangle (orange) and its corresponding template (blue). Right: Illustration for the design of holes. $\mathbf{e}' = \overline{\mathbf{a}'\mathbf{b}'}$ (blue) and $\mathbf{e} = \overline{\mathbf{a}\mathbf{b}}$ (orange) are two corresponding edges on the template and the output triangle, respectively. The radius of the dash circle is the maximum allowable fabrication error. $\mathbf{a}$, $\mathbf{b}$ lie on the circle. L_2 is parallel to $\overline{\mathbf{a}'\mathbf{b}'}$ and the distance between them is l_2. L_1 is parallel to the perpendicular bisector of $\overline{\mathbf{a}\mathbf{b}}$ and the distance between them is l_1. Let α, β vary in $[0, 2\pi]$, the trajectories of the intersections of L_1 and L_2 are shown as the green and red solid lines, respectively.

Notations. For an edge $\mathbf{e} = \overline{\mathbf{a}\mathbf{b}}$ of $\mathcal{R}$, we denote its one adjacent triangle as $\mathbf{f}$ (see the notations in Fig. 9). The template triangle for $\mathbf{f}$ is denoted as $\mathbf{t}$, and the corresponding edge of $\mathbf{e}$ in $\mathbf{t}$ is denoted as $\mathbf{e}' = \overline{\mathbf{a}'\mathbf{b}'}$. In our experiments, the center of hinge coincides with the midpoint of $\mathbf{e}$, and the middle axis of the hinge aligns with $\mathbf{e}$. Due to symmetry, we can ignore another adjacent triangle of $\mathbf{e}$ for determining the holes. For convenience, we assume that the holes as line segments, the holes on hinge are perpendicular to $\mathbf{e}$, and the holes on template are parallel to $\mathbf{e}'$. We denote the distance from hinge holes to the perpendicular bisector of $\mathbf{e}$ as l_1 and denote the distance from template holes to $\mathbf{e}'$ as l_2.

Intersections. The intersections between hinge holes and template holes are different for different pairs of $\mathbf{e}$ and $\mathbf{e}'$. The range of the intersections determines the hole locations. We denote the maximum allowable fabrication error as d_b. To compute the intersection range, we take the extreme case, where $\|\mathbf{a} - \mathbf{a}'\|_2 = d_b$ and $\|\mathbf{b} - \mathbf{b}'\|_2 = d_b$. It indicates that $\mathbf{a}$ (or $\mathbf{b}$) lies on the circle with $\mathbf{a}'$ (or $\mathbf{b}'$) as the center. Suppose that the polar angles of $\mathbf{a}$ and $\mathbf{b}$ are α and β, respectively. Then, we have

$$\mathbf{a} = \mathbf{a}' + d_b \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix}, \quad \mathbf{b} = \mathbf{b}' + d_b \begin{pmatrix} \cos\beta \\ \sin\beta \end{pmatrix}. \tag{9}$$

The two linear equations for hinge holes are:

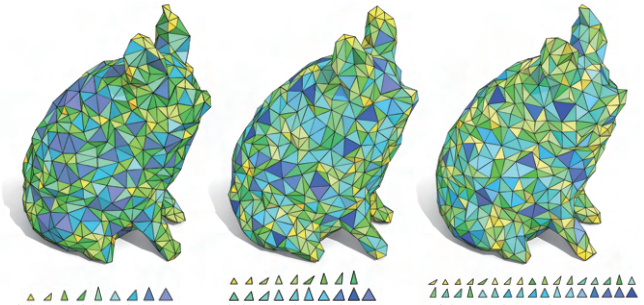
$$(\mathbf{a} - \mathbf{b})^T \left(\mathbf{x} - \left(\frac{\mathbf{a} + \mathbf{b}}{2} \pm l_1 \frac{\mathbf{a} - \mathbf{b}}{\|\mathbf{a} - \mathbf{b}\|_2} \right) \right) = 0. \tag{10}$$

The linear equation for template holes is:

$$\left((\mathbf{a}' - \mathbf{b}')^{\perp}\right)^T \left(\mathbf{x} - \left(\frac{\mathbf{a}' + \mathbf{b}'}{2} + l_2 \frac{(\mathbf{b}' - \mathbf{a}')^{\perp}}{\|(\mathbf{b}' - \mathbf{a}')^{\perp}\|} \right) \right) = 0, \tag{11}$$

where $\perp$ denotes a counterclockwise rotation by $\pi/2$. Given $\mathbf{a}'$, $\mathbf{b}'$, l_1, l_2, and d_b, the intersections are the functions of α, β. When α, β vary in $[0, 2\pi]$, we use the Nelder-Mead method [Nelder and Mead 1965] to solve the range of the intersections for hinge holes (denoted as H_{hinge}) and template holes (denoted as H_{template}).

Final holes. We use the template holes with the same locations and sizes for the edges of template triangles with the same length. Since all hinges have the same holes, we set the union of all H_{hinge}s



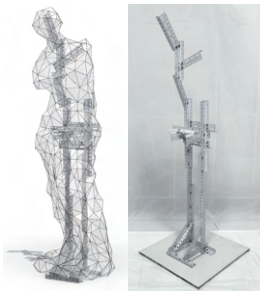
(a) $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 4.15\%$ **(b)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 2.70\%$ **(c)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 2.10\%$

Fig. 10. Different number of classes: 9 classes in (a), 19 classes in (a), and 34 classes in (c). The template triangles are shown below the model.

as the resulting holes, denoted as $H^{\text{u}}_{\text{hinge}}$. The width of the holes is set slightly larger than that of the nylon cable ties.

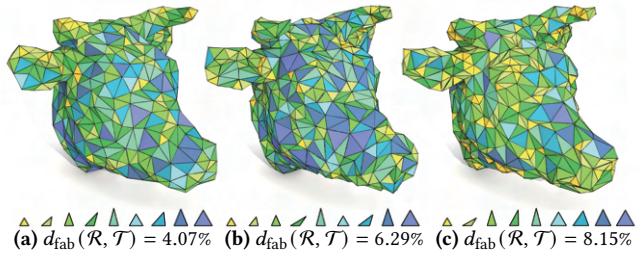
Physical sizes and collision avoidance. Collisions between hinges on a template triangle may appear, thereby making manufacturing infeasible. In our real fabrication, nine template triangles from three different edges (6cm, 9cm, and 12cm) are used. For the real fabrication example, the maximum fabrication error is 0.267cm; thus, d_b is set as 0.3cm. In practice, we take $l_1 = l_2 = 0.6$cm. Consequently, no collisions happen. The template hole sizes are 0.612cm for the template edge of 6cm, 0.605cm for the template edge of 9cm, 0.603cm for the template edge of 12cm. Besides, $H^{\text{u}}_{\text{hinge}} = 0.600$cm.

Supporting structures. Since the mechanical properties are not considered, those physical manufacturing examples usually cannot maintain their shapes and cannot stand independently. We empirically handcraft an additional supporting structure for each model in practice. The right inset shows the supporting structure for the Venus model.



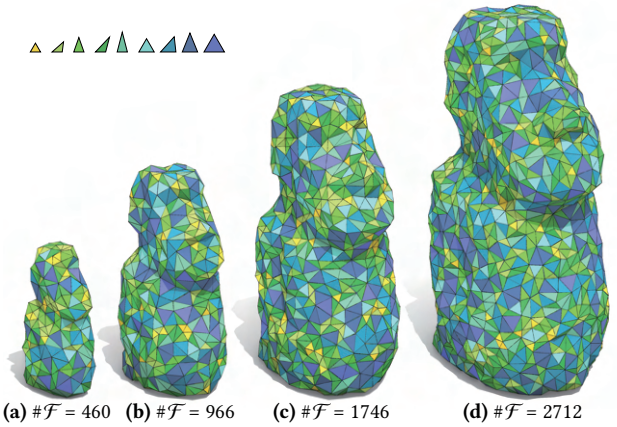
5 EXPERIMENTS

We have tested our modeling algorithm with specified discrete equivalence classes of triangles on various models to evaluate its performance. We select the topology-fixed method [Singh and Schaefer 2010] and the Zometool-based method [Zimmer et al. 2014] as the competitors. Our method was implemented in C++, and all of our experiments were executed on a desktop PC with a 3.7 GHz Intel Core i7-8700K and 32GB of memory. Without additional specification, the fabrication error is measured as a percentage of the shortest edge of the templates and the one-sided Hausdorff distance $d_H(\mathcal{R},\mathcal{M})$ from $\mathcal{R}$ to $\mathcal{M}$ is measured as a percentage of the diagonal length of the bounding box of $\mathcal{M}$. We visualize results by mapping the templates to the corresponding output triangles using the best rigid transformations. In our experiments, we set $\epsilon_{\text{envelope}} = 3\%$ by default. Table 1 summarizes the statistics of our results.



(a) $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 4.07\%$ **(b)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 6.29\%$ **(c)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 8.15\%$

Fig. 11. Three templates with different shapes but the same number of classes. The template triangles are shown below the model.



(a) $\#\mathcal{F} = 460$ **(b)** $\#\mathcal{F} = 966$ **(c)** $\#\mathcal{F} = 1746$ **(d)** $\#\mathcal{F} = 2712$

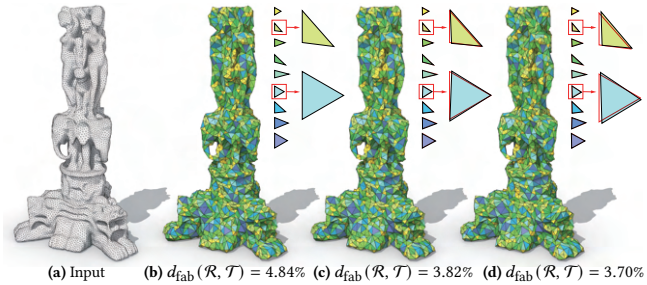
Fig. 12. Various sizes. From left to right, the fabrication errors are 3.93%, 3.93%, 4.53%, and 4.62%, respectively. $\#\mathcal{F}$ indicates the number of triangles in the output $\mathcal{R}$.

5.1 Evaluations

Specifying template triangles. To make the template triangles as consistent as possible on each edge of the output mesh, we specify the template triangles using edge lengths. Given a set of edge lengths $\mathcal{L} = \{l_1, \cdots, l_n\}$, the unique and valid triangles with three edge lengths (that can be the same) in $\mathcal{L}$ forms $\mathcal{T}$. Here the term "valid" means that the three edge lengths satisfy the triangle inequality and the word "unique" indicates that no two triangles are congruent.

There are two main factors that affect the representation ability of the template: (1) the number of classes and (2) the template shapes. In Fig. 10, we tested 9 templates with $\mathcal{L} = \{2, 3, 4\}$, 19 templates with $\mathcal{L} = \{2, 8/3, 10/3, 4\}$, and 34 templates with $\mathcal{L} = \{2, 2.5, 3, 3.5, 4\}$ on the cow model. From the results, the more classes, the smaller the fabrication error. This is in line with common sense. In Fig.11, we test our method with three different $\mathcal{L}$s on the cow model: $\{2, 3, 4\}$ for (a), $\{2, 2.5, 4\}$ for (b), and $\{2, 3.5, 4\}$ for (c). The template triangles in Fig.11 (b) and (c) change greatly, thereby leading to a large fabrication error. We use $\{2, 3, 4\}$ by default.

Shape sizes. If a set of large templates is used to model and fabricate a much smaller model, our algorithm fails to reduce the fabrication error to a small level. At this time, we need to adjust the size of the template or the model. In our experiments, we fix the size of the template and modify the input model size. We provide a practical way to estimate the minimum allowable size. First, an isotropic remeshing algorithm, which bounds the one-sided Hausdorff distance and tries to minimize the number of



(a) Input **(b)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 4.84\%$ **(c)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 3.82\%$ **(d)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 3.70\%$

Fig. 13. Optimizing template triangles. (b) Our algorithm. (c) The first strategy. (d) The second strategy. The red triangles in (c) and (d) are the templates used by (b).

triangles, is performed. Second, compute the average edge length of the isotropically remeshed result (denoted as $l_{\text{avg}}^{\text{iso}}$) and the average edge length of the template triangles (denoted as $l_{\text{avg}}^{\text{temp}}$). Third, scale the model $l_{\text{avg}}^{\text{temp}}/l_{\text{avg}}^{\text{iso}}$ times. In practice, we use the remeshing algorithm of [Yang et al. 2020] by setting the target edge length field as a constant value function and bounding the one-sided Hausdorff distance, whose threshold is set as our defaulting $\epsilon_{\text{envelope}}$. In Fig. 12, we show an example with four different sizes. Our method successfully achieves small fabrication errors for all cases.

Optimizing template triangles. We can modify our pipeline to automatically optimize the templates to reduce the fabrication error with the given template number. The method to optimize the templates without changing the number is as follows:

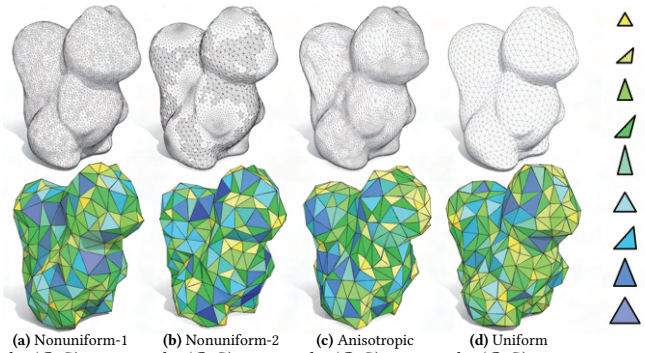
(1) For each triangle $\mathbf{f}$ in $\mathcal{R}$, we first compute the matching error (5) to find the template that has the smallest similarity metric and achieve the best rigid transformation between $\mathbf{f}$ and the obtained template. Then, the best rigid transformation is used to map $\mathbf{f}$ onto the plane.
(2) To update a vertex $\mathbf{v}$ of each template $\mathbf{t}$, we first collect the transformed vertices, which correspond $\mathbf{v}$, into a point set and then move $\mathbf{v}$ to the center of the minimum bounding circle of the point set.
(3) If the maximum movement of the template vertices is less than 10^{-6} times the shortest edge length of our default template, stop the optimization process; otherwise, go to step 1.

We propose two strategies to modify our pipelines:

- *The first strategy*: We alternately run the default algorithm and optimize the templates without changing the number. The alternating process is stopped when the template optimization procedure does not change the templates.
- *The second strategy*: We add the template optimization before the global relocation step to modify the default pipeline. The termination condition is the same as the default condition.

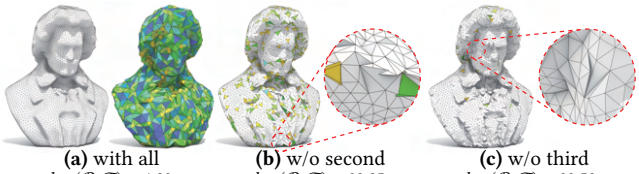
We show a comparison in Fig. 13. Both strategies achieve smaller fabrication errors than our default algorithm.

Various initializations. In Fig.14, four types of tessellations representing one surface are tested. The results generated by our method have similar fabrication errors $d_{\text{fab}}(\mathcal{R},\mathcal{T})$, which demonstrates the robustness of our algorithm against irregular inputs. The fabrication errors are all at a low level.



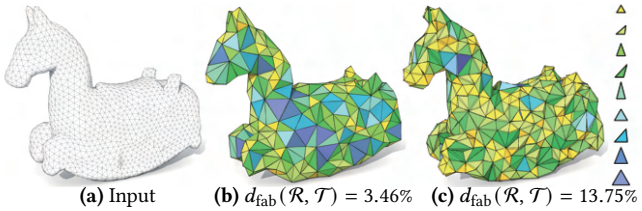
(a) Nonuniform-1 $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 3.10\%$ **(b)** Nonuniform-2 $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 3.45\%$ **(c)** Anisotropic $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 3.88\%$ **(d)** Uniform $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 3.99\%$

Fig. 14. Different tessellations of $\mathcal{M}$. Top: The input meshes. Bottom: The remeshing results.



(a) with all $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 4.39\%$ **(b)** w/o second $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 28.35\%$ **(c)** w/o third $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 28.58\%$

Fig. 15. Necessities of the second and third smoothness constraints.



(a) Input **(b)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 3.46\%$ **(c)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 13.75\%$

Fig. 16. Replacing the local perturbation with the vertex relocation method of the global relocation strategy. (b) With local perturbation. (c) Using the vertex relocation method.

Smoothness constraints. The second and third smoothness constraints are required. The second constraint gives a precaution against intersections, and the third constraint avoids zigzags. Without them, the algorithm is stuck since the intersections of triangles or sharp jagged shapes appear in the result (see the zoom-in views in Fig. 15). For the second constraint, the intersection of triangles also depends on the edge length. However, if the dihedral angles satisfy the second smoothness constraint, there will be no intersection no matter how long the edges are.

Local perturbation. The local perturbation can be replaced by the vertex relocation method of the global relocation strategy. Fig. 16 shows the comparison between the local perturbation and this strategy. The local perturbation is able to make the algorithm produce a much lower fabrication error. The vertex relocation method can be treated as the exact position optimization, whereas the local perturbation can be regarded as a kind of inexact sampling. The inexact sampling enlarges the search space, thereby avoiding early entrapment by local minimum.

Different models with a set of template triangles. We test our method on 71 complex models using a set of discrete equivalence classes of triangles, i.e., $\mathcal{T}$ based on $\mathcal{L} = \{2, 3, 4\}$. The resulting

Fig. 17. Gallery of our results for 13 selected models.

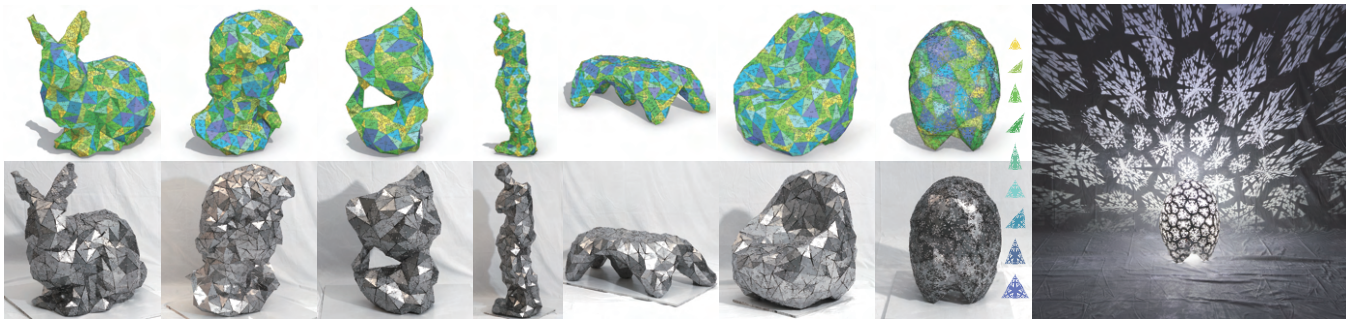


Fig. 18. Gallery of our digital results (top) and real fabrications (bottom) for seven models. The lampshade uses the templates with a specially designed hollow structure. The actual lighting effect is shown on the far right.

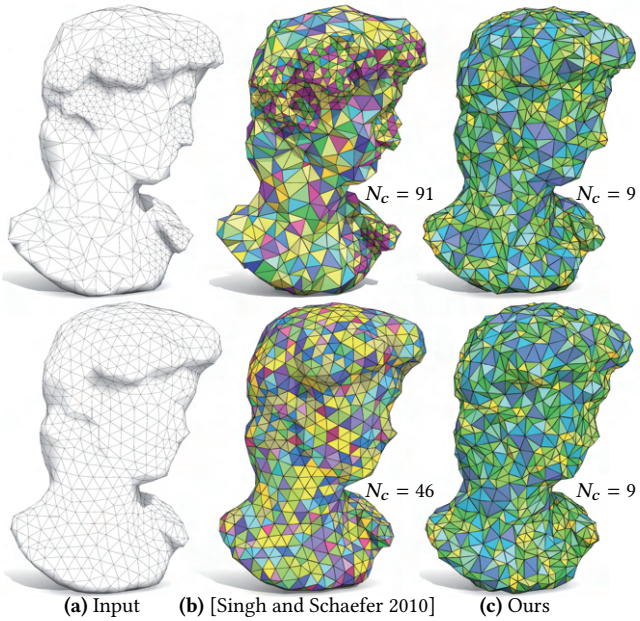
fabrication errors are all lower than 5%. Fig. 17 shows 13 selected models. The seven physically fabricated models are shown in Fig. 18.

5.2 Comparisons

Comparison with [Singh and Schaefer 2010]. Our motivation is different from [Singh and Schaefer 2010]. Our method explores whether the algorithm can be widely used in various models with a set of predefined templates while they optimize different templates for different input models. As a significant benefit, our method can use pre-manufactured templates for different models, thus there is no need to customize templates for each model. It further reduces fabrication costs.

In Fig. 2 and 19, we compare with [Singh and Schaefer 2010]. The method of [Singh and Schaefer 2010] terminates when the maximum distance between vertices of the optimized mesh and the transformed templates is less than the user-specified threshold. Their termination condition uses our $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$ for judgment. Thus, for fair comparison, we first run our algorithm to achieve $d_{\text{fab}}^{\text{ours}}(\mathcal{R}, \mathcal{T})$, then set $d_{\text{fab}}^{\text{ours}}(\mathcal{R}, \mathcal{T})$ as the user-specified threshold for [Singh and Schaefer 2010], and finally compare the number of classes for two methods. Since the method of [Singh and Schaefer 2010] does not optimize the mesh topology, many classes are required for the irregular meshes (Fig. 2 and Fig. 19 - top). Then, we test their algorithm on an isotropic mesh (Fig. 19 - bottom). However, it still generates more classes than our method. Note that both methods produce the comparable $d_{\text{fab}}(\mathcal{R}, \mathcal{T})$; but the method of [Singh and Schaefer 2010] gets smaller $d_{\text{match}}(\mathbf{f}, \mathcal{T})/N_f$, where N_f is the number of triangles in the output mesh.

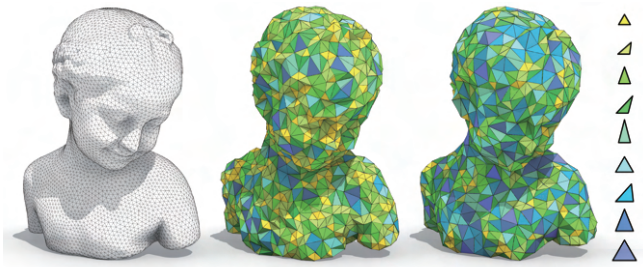
The similarity metric between two triangles in [Singh and Schaefer 2010] is: $d_{\text{sum}}(\mathbf{f}, \mathbf{t}) = \min_{R,\mathbf{b}} \sum_{i\in\{0,1,2\}} \{\|R\mathbf{p}_i + \mathbf{b} - \mathbf{v}_i\|_2^2\}$. The best rigid transformation for $d_{\text{sum}}(\mathbf{f}, \mathbf{t})$ can be analytically obtained



(a) Input **(b)** [Singh and Schaefer 2010] **(c)** Ours

Fig. 19. Comparisons with [Singh and Schaefer 2010] on David's head using nonuniform and isotropic tessellations. N_c is the number of classes.

by [Arun et al. 1987]. $d_{\text{sum}}(\mathbf{f}, \mathbf{t})$ can be used to replace $d_{\max}(\mathbf{f}, \mathbf{t})$ in our algorithm. Fig. 20 shows the comparison on the Bimba model. In Fig. 21, we show the detailed difference on a pair of triangles. From these comparisons, optimizing $d_{\text{sum}}(\mathbf{f}, \mathbf{t})$ may still lead to a large $d_{\max}(\mathbf{f}, \mathbf{t})$. Therefore, $d_{\text{sum}}(\mathbf{f}, \mathbf{t})$ is not directly used for physical fabrication. The minimizer of the two-sided Hausdorff distance concerning a rigid transformation is a suitable metric to measure



(a) Input **(b)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 9.22\%$ **(c)** $d_{\text{fab}}(\mathcal{R},\mathcal{T}) = 4.36\%$

Fig. 20. Replacing $d_{\max}(\mathbf{f},\mathbf{t})$ with $d_{\text{sum}}(\mathbf{f},\mathbf{t})$. Given an input model (a), our algorithm considers $d_{\text{sum}}(\mathbf{f},\mathbf{t})$ (b) and $d_{\max}(\mathbf{f},\mathbf{t})$ (c), respectively.

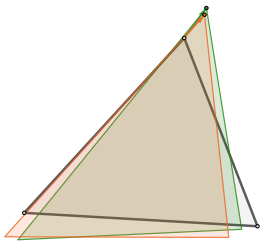
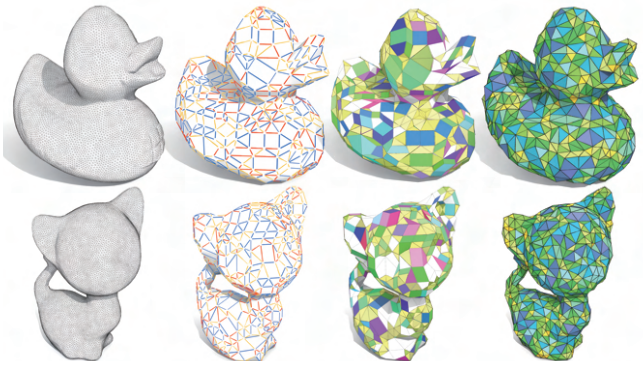


Fig. 21. Difference between the optimization of $d_{\max}(\mathbf{f},\mathbf{t})$ and $d_{\text{sum}}(\mathbf{f},\mathbf{t})$. Given two triangles and we fix one of them (grey), the orange and green triangles are the other triangle after best rigid transformations by optimizing $d_{\max}(\mathbf{f},\mathbf{t})$ and $d_{\text{sum}}(\mathbf{f},\mathbf{t})$, respectively. The arrows indicate the corresponding vertex pair with maximum distance.



(a) Input **(b)** [Zimmer et al. 2014] **(c)** Ours

Fig. 22. Comparison to [Zimmer et al. 2014] on two models. The results of [Zimmer et al. 2014] are shown using their specified edges (left) and the template polygons computed by us (right). The white quads are not planar.

the difference between two triangles. However, it is challenging to numerically optimize the two-sided Hausdorff distance. Based on the following proposition, our similarity metric optimizes the tighter upper bound of the two-sided Hausdorff distance than the similarity metric in [Singh and Schaefer 2010].

Proposition 4. *Given two triangles* $\triangle\mathbf{a}_o\mathbf{a}_1\mathbf{a}_2$ *and* $\triangle\mathbf{b}_0\mathbf{b}_1\mathbf{b}_2$*, let* $d_H(\triangle\mathbf{a}_0\mathbf{a}_1\mathbf{a}_2,\ \triangle\mathbf{b}_0\mathbf{b}_1\mathbf{b}_2)$ *be the two-sided Hausdorff distance between two triangles and* $\xi = (\|\mathbf{b}_0-\mathbf{a}_0\|_2, \|\mathbf{b}_1-\mathbf{a}_1\|_2, \|\mathbf{b}_2-\mathbf{a}_2\|_2)$*. Then,*

$$d_H(\triangle\mathbf{a}_0\mathbf{a}_1\mathbf{a}_2,\ \triangle\mathbf{b}_0\mathbf{b}_1\mathbf{b}_2) \le \|\xi\|_\infty \le \|\xi\|_2. \tag{12}$$

We prove it in the supplementary material.

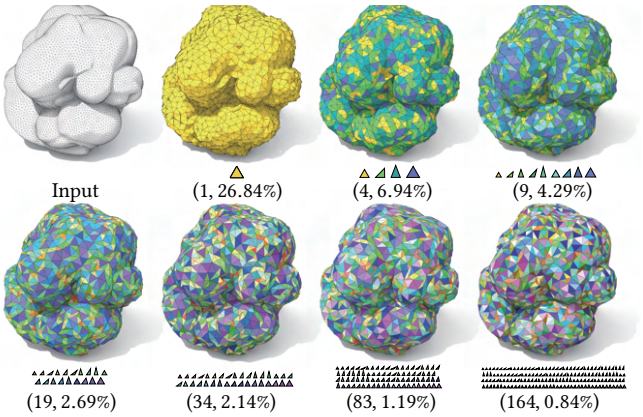


Fig. 23. Limited curvature and bumpy surfaces. The text below each model indicates the number of templates and the fabrication error, respectively.

Comparison to [Zimmer et al. 2014]. We compare the Zometool-based mehtod [Zimmer et al. 2014] one two models in Fig. 22. Our methods uses 9 template triangles. The results of [Zimmer et al. 2014] are kindly provided by the authors. The Zometool system uses nine prescribed edges and one predefined node to model the shapes. The surface panels formed by the Zometool system implicitly consists of the 29 unique triangles and 118 different convex, planar quads [Zimmer and Kobbelt 2014]. We determine the template polygons and count the number of templates of their results. We observe that: (1) there are non-planar quads in their results and (2) their number of templates is much greater than ours.

6 CONCLUSION

We present a novel fabrication-error-driven remeshing algorithm to model and fabricate shapes using a set of prescribed triangles. Specifically, we propose a new similarity metric for measuring the matching error between two triangles from a manufacturing perspective. Then, we present an easy-to-compute solution to compute the new metric. Our greedy remeshing approach effectively minimizes the fabrication error defined as the maximum matching error of all triangles. We have demonstrated the feasibility and practicability of our method using only nine template triangles for manufacturing seven models physically.

Fabrication error. Although our algorithm can not explicitly control the fabrication error to be less than a user-specified threshold, the maximum $d_{\text{fab}}(\mathcal{R},\mathcal{T})$ of our all examples in our experiment is lower than 5%. According to our real manufacturing experience, this error almost does not affect the manufacturing process. To explicitly bound the fabrication error, a trivial solution is to add more discrete equivalence classes during the optimization process automatically.

Bumpy surfaces. Due to the limited number of templates, the limitation of curvature is inevitable. Thus, when the number of template triangles is small, the resulting surfaces are slightly bumpy (Fig. 23 - upper row). Besides, our algorithm fails to generate a low fabrication error when the template number is extremely small. To ameliorate the effects of the limited curvature, we apply the geometric distance error and add smoothness constraints that force the optimized surfaces to be not too rugged. Another practical solution

Table 1. Statistics and timings. We report the number of triangles in the output $\mathcal{R}$ ($\#\mathcal{F}$), the percentage of each class (N^c_{stat}), the fabrication error measured as a percentage of the shortest edge of the templates (d_{fab}), the one-sided Hausdorff distance from $\mathcal{R}$ to $\mathcal{M}$ measured as a percentage of the bounding box diagonal of the input $\mathcal{M}$ (d_H), and the computational time in minutes.

Model	$\#\mathcal{F}$	$N^c_{\text{stat}}(\%)$	$d_{\text{fab}}(\%)$	$d_H(\%)$	Time (m)
Fig.2	1770	4.2/15.3/15.4/23.1/9.3/5.4/14.0/10.7/2.5	4.23	1.11	93.3
Fig.10	1292	6.7/17.3/15.4/24.0/9.4/5.0/11.0/8.3/3.1	4.15	1.54	84.1
Fig.11	1014	8.2/22.3/17.6/21.3/8.2/5.0/9.5/6.4/1.6	4.07	1.78	113.5
Fig.12 (a)	460	7.6/18.5/16.7/23.3/8.9/6.1/11.5/4.8/2.6	3.93	2.99	35.9
Fig.12 (b)	966	5.3/15.4/13.9/22.2/10.2/5.8/14.3/10.7/2.3	3.92	2.17	41
Fig.12 (c)	1746	5.6/15.7/15.9/24.1/7.6/5.6/13.6/8.8/3.2	4.52	2.72	65.2
Fig.12 (d)	2712	4.8/13.4/17.2/23.4/8.6/5.2/14.5/10.3/2.7	4.62	1.11	72.6
Fig.13 (b)	5086	15.5/26.4/18.8/19.6/5.2/4.0/6.4/3.6/0.5	4.84	0.95	123.5
Fig.14 (a)	502	5.4/19.7/19.3/23.5/7.6/6.0/9.2/7.8/1.6	3.10	2.72	56.3
Fig.14 (b)	518	8.3/20.5/20.1/21.2/6.8/5.0/11.2/5.0/1.9	3.45	2.98	66.8
Fig.14 (c)	508	7.9/23/16.9/19.7/7.9/4.9/11.6/6.7/1.4	3.88	2.30	66.6
Fig.14 (d)	564	10.8/26.6/19.3/20.2/5/5.5/8.2/3.7/0.7	3.99	1.86	63.5
Fig.15 (a)	1984	5.7/18.6/16.1/23.5/7.5/6/12.1/8.2/2.3	4.39	1.57	88.7
Fig.16 (b)	668	17.7/28.3/18.1/17.2/5.7/3.7/5.7/2.5/1.0	3.46	2.36	53.2
Fig.17 Doraemon	758	7.8/18.2/21.4/20.3/6.5/7.7/8.3/7.9/2.0	4.57	2.38	61.3
Fig.17 Dog	2738	10.8/26.2/20.4/20.7/4.9/4.8/7.6/4.2/0.4	4.69	1.04	80.1
Fig.17 Lion_recon	2202	9.8/20.6/18.0/21.8/6.2/4.5/10.9/6.9/1.4	4.37	1.19	125.1
Fig.17 Eros	3054	6.3/17.1/17.2/22.2/8.3/5.7/12.5/8.2/2.6	4.42	1.36	117.3
Fig.17 Sculpture	3846	5.2/14.7/16.3/23.3/8.2/6.4/12.9/10.2/2.8	4.37	1.06	115.3
Fig.17 Bulldog	1460	7.1/18.4/15.2/22.3/7.3/6.2/11.7/9.1/2.6	4.42	1.73	45.6
Fig.17 VaseLion	3664	9.0/23.0/18.5/21.3/7.4/4.9/8.8/5.6/1.4	4.57	1.27	115.4
Fig.17 Ramses	1762	8.6/20.6/18.2/22.9/6.8/4.3/9.6/6.9/2.1	4.36	1.70	113.6
Fig.17 Mickey	3078	7.2/18.5/16.7/23.1/8.5/5.6/11.1/7.6/1.7	4.04	1.12	119.6
Fig.17 Buddha	3538	6.3/16.2/15.5/22.2/8.9/5.6/11.8/10.7/2.9	4.43	1.61	122.5
Fig.17 Gargoyle	3008	7.5/18/16.3/22.7/7.7/5.4/11.6/8.4/2.5	4.17	1.39	143.4
Fig.17 SantaClaus	1472	8.4/19.6/17.1/23.4/8.3/4.5/10.2/6.8/1.9	4.02	1.94	87.5
Fig.17 Rocket	944	8.9/20.6/16.1/21.8/9.1/4.3/9.6/8.2/1.4	3.47	1.93	72.4
Fig.18 Bunny	912	14.1/27.1/18.4/19.1/5.4/4.3/7.6/3.3/0.8	3.97	2.25	76.6
Fig.18 David	1186	12.1/27.6/16.9/20.8/5.6/4.6/8.3/3.2/0.9	3.77	2.14	77.5
Fig.18 Kitty	530	13.2/25.5/17.9/20.0/5.1/5.8/7.7/3.8/0.9	4.21	2.21	30.1
Fig.18 Venus	860	14.0/25.1/20.7/18.1/4.0/6.5/7.9/3.3/0.5	4.00	1.99	66.3
Fig.18 Table	1012	6.7/20.8/17.8/20.8/7.4/5.3/11.5/7.4/2.3	4.10	1.62	85.6
Fig.18 Sofa	836	3.6/15.9/21.5/20.8/8.4/5.9/12.4/8.9/2.6	4.06	2.47	62.4
Fig.18 Lampshade	310	11.3/21.0/21.9/17.7/4.5/5.5/11/6.1/1.0	3.76	2.74	24.3
Fig.19 (top)	2116	7.0/18.4/16.2/23.3/7.6/6.1/11.9/7.7/1.9	4.16	1.28	101.2
Fig.19 (bottom)	2074	7.1/18.0/18.0/22.7/8.5/5.2/10.8/7.9/1.7	4.21	1.31	98.3
Fig.20 (c)	2060	7.7/20.5/16.8/21.7/7.1/6.5/10.6/7.1/1.8	4.36	1.22	81.3
Fig.22 (top)	1386	6.8/17.0/16.0/23.6/8.9/4.5/14.3/6.6/2.4	4.21	1.69	85.3
Fig.22 (bottom)	1466	7.4/18.8/17.7/20.7/7.2/5.3/12.4/8.5/2.0	4.40	1.33	91.2
Fig.23 (9 templates)	3216	5.3/15/16.6/22.1/8.4/5.4/13.3/10.9/3.1	4.28	1.26	54.3

to avoid bumpy surfaces is to increase the number of templates. As shown in the bottom row of Fig. 23, a large number of templates help to generate a smooth surface and reduce the fabrication error to a very low level.

Physical considerations. Only the mesh topology and geometry are considered in our algorithm. The mechanical equilibrium and structural stability are not considered during optimization. Thus, we made support structures inside the models for real fabrications. In the future, FEM is required to obtain optimal mechanical performance. Besides, our real fabrication method does not precisely match the computed positions of templates, including the errors caused by the mechanical deformation and gravity. Thus, the real fabrication error is different from the prediction but is still under control in our experiments. In the future, advanced manufacturing methods, such as screwing screws with a robotic arm, can precisely locate the positions of the templates to match the fabrication error and the prediction.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their constructive suggestions and comments. This work is supported by the National Natural Science Foundation of China (61802359, 62025207), Zhejiang Lab (NO. 2019NB0AB03), and USTC Research Funds of the Double First-Class Initiative (YD0010002003).

REFERENCES

Pierre Alliez, Giuliana Ucelli, Craig Gotsman, and Marco Attene. 2008. Recent advances in remeshing of surfaces. In *Shape analysis and structuring*. 53–82.

K. S. Arun, T. S. Huang, and S. D. Blostein. 1987. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9, 5 (1987), 698–700.

Mario Botsch and Leif Kobbelt. 2004. A Remeshing Approach to Multiresolution Modeling. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP '04)*. 185–192.

Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. 2010. *Polygon mesh processing*. AK Peters/CRC Press.

Xuelin Chen, Honghua Li, Chi-Wing Fu, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2018. 3D Fabrication with Universal Building Blocks and Pyramidal Shells. *ACM Trans. Graph.* 37, 6 (2018).

Xiao-Xiang Cheng, Xiao-Ming Fu, Chi Zhang, and Shuangming Chai. 2019. Practical Error-Bounded Remeshing by Adaptive Refinement. *Computers & Graphics* 82 (2019), 163–173.

Michael Eigensatz, Martin Kilian, Alexander Schiftner, Niloy J. Mitra, Helmut Pottmann, and Mark Pauly. 2010. Paneling Architectural Freeform Surfaces. *ACM Trans. Graph.* 29, 4 (2010).

Chi-Wing Fu, Chi-Fu Lai, Ying He, and Daniel Cohen-Or. 2010. K-Set Tilable Surfaces. *ACM Trans. Graph.* 29, 4 (2010).

Xiao-Ming Fu, Yang Liu, John Snyder, and Baining Guo. 2014. Anisotropic simplicial meshing using local convex functions. *ACM Transactions on Graphics (SIGGRAPH Asia)* 33, 6 (2014).

Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. 1993. Mesh Optimization. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '93)*. 19–26.

K. Hu, D. Yan, D. Bommes, P. Alliez, and B. Benes. 2017. Error-Bounded and Feature Preserving Surface Remeshing with Minimal Angle Improvement. *IEEE. T. Vis. Comput. Gr.* 23, 12 (2017), 2560–2573.

Mathieu Huard, Michael Eigensatz, and Philippe Bompas. 2015. Planar Panelization with Extreme Repetition. In *Advances in Architectural Geometry 2014*. 259–279.

Wenzel Jakob, Marco Tarini, Daniele Panozzo, and Olga Sorkine-Hornung. 2015. Instant Field-aligned Meshes. *ACM Trans. Graph.* 34, 6 (2015), 189:1–189:15.

J. A. Nelder and R. Mead. 1965. A Simplex Method for Function Minimization. *Comput. J.* 7, 4 (1965), 308–313.

I-Chao Shen, Ming-Shiuan Chen, Chun-Kai Huang, and Bing-Yu Chen. 2020. ZomeFab: Cost-Effective Hybrid Fabrication with Zometools. *Comput. Graph. Forum* 39, 1 (2020), 322–332.

Mayank Singh and Scott Schaefer. 2010. Triangle Surfaces with Discrete Equivalence Classes. *ACM Trans. Graph.* 29, 4 (2010).

Jian-Ping Su, Chunyang Ye, Ligang Liu, and Xiao-Ming Fu. 2020. Efficient Bijective Parameterizations. *ACM Trans. Graph.* 39, 4 (2020).

Bolun Wang, Teseo Schneider, Yixin Hu, Marco Attene, and Daniele Panozzo. 2020. Exact and Efficient Polyhedral Envelope Containment Check. *ACM Trans. Graph.* 39, 4 (2020).

Yiqun Wang, Dong-Ming Yan, Xiaohan Liu, Chengcheng Tang, Jianwei Guo, Xiaopeng Zhang, and Peter Wonka. 2018. Isotropic Surface Remeshing without Large and Small Angles. *IEEE. T. Vis. Comput. Gr.* (2018).

Dong-Ming Yan, Bruno Lévy, Yang Liu, Feng Sun, and Wenping Wang. 2009. Isotropic remeshing with fast and exact computation of restricted Voronoi diagram. *Computer graphics forum* 28, 5 (2009), 1445–1454.

Yang Yang, Wen-Xiang Zhang, Yuan Liu, Ligang Liu, and Xiao-Ming Fu. 2020. Error-Bounded Compatible Remeshing. *ACM Trans. Graph.* 39, 4 (2020).

Zichun Zhong, Xiaohu Guo, Wenping Wang, Bruno Lévy, Feng Sun, Yang Liu, and Weihua Mao. 2013. Particle-based Anisotropic Surface Meshing. *ACM Trans. Graph.* 32, 4 (2013), 99:1–99:14.

Tianyu Zhu, Chunyang Ye, Xiao-Ming Fu, and Shuangming Chai. 2020. Greedy Cut Construction for Parameterizations. *Computer Graphics Forum* 39, 2 (2020), X.

Henrik Zimmer, Marcel Campen, David Bommes, and Leif Kobbelt. 2012. Rationalization of Triangle-Based Point-Folding Structures. *Comput. Graph. Forum* 31, 2pt3 (2012), 611–620.

H. Zimmer and L. Kobbelt. 2014. Zometool Rationalization of Freeform Surfaces. *IEEE Transactions on Visualization and Computer Graphics* 20, 10 (2014), 1461–1473.

Henrik Zimmer, Florent Lafarge, Pierre Alliez, and Leif Kobbelt. 2014. Zometool shape approximation. *Graphical Models* 76, 5 (2014), 390 – 401.