使用龍圖兒做為內部結構以支持大型 3D 模型列印

陳明軒　　黃群凱　　沈奕超　　陳炳宇

國立台灣大學

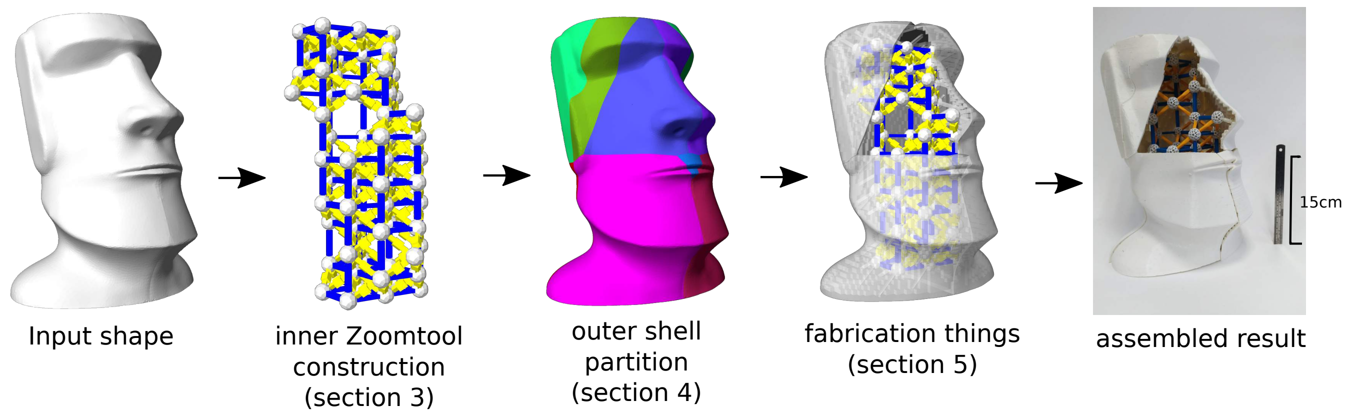

Figure 1: *The overview of our method. Given input shape, we first optimize the inner Zometool structure (Section 3). Guided by optimized Zometool structure, we partition the outer shell (Section 4) and generate connectors for assembling both structures (Section 5). The final fabricated result are obtained by assemble together both assembled Zometool structure and printed outer shell.*

Abstract

近年來，3D列印技術越來越接近消費者階級，因此擁有個人3D列印機的使用者也隨之上升，然而一般購買之3D列印機往往會因為冗長的列印時間以及較小、被受限的列印空間無法進行大型物體之原型設計，為了去達到這樣的目標，使用另外的材料來連接以列印出來的物件是一種常見的做法。在這篇論文中，我們提出一個使用3D列印機以及龍圖兒這種具有強大結構性之玩具來達到具有成本效益且可產出大型物件之製造技術，我們核心方法是使用龍圖兒來當作內部結構和使用3D列印機列印外表皮，此研究最大的挑戰是解決各種棘手的問題例如是否能列印、是否有省耗材以及內部龍圖兒結構的組裝複雜度，而我們不但解決這些核心問題，並運用此系統來製造各式各樣實體的3D模型。

Keywords: Geometric Algorithms, Fabrication, Curve, surface, solid, and object representations

1 Introduction

The production cost of 3D printer has been declining and it's availability for everyone leads to the emergence of large amount of applications. However, the consumer-level 3D printers have several drawbacks: the long printing time, the limited output size and relatively high cost of materials. There are several approaches and researches aim to solve these drawbacks. For time and materials saving, the modern 3D printer have fill-rate setting to using less material and research like [Lu et al. 2014] can save materials effectively. The size of objects that 3D printing can produce are typically limited by 3d printers. For building a large-scale fabrication, lots of researches [Medelln et al. 2007][Hao et al. 2011] [Luo et al. 2012] [Hu et al. 2014] [Vanek et al. 2014] focus on this problem. They all have same characteristic that partition the object to fit the size then assembling printed parts.

The requirement of a large-scale fabrication construction will enlarge the above drawbacks. It turns out that even employ state-of-the-art methods. Time and material spending are still too high. To extremely decrease these cost, our novel idea is to combine 3D printing with another material. That material have several advantages like generality which is easy to gather, reusability which is good for decreasing cost, and robustness which is simple to build and reliable. Base on our requirement, the popular modeling system, i.e. *Zometool* [Zimmer and Kobbelt 2014a], is suitable for coarse fabrication. Including the above advantages, Zometool still has several good chararistics: (i) structural properties, such as stability, expandability and lightness satisfying the requirements for large-scale fabrication. (ii) independent structure and modularity can parallelize the construction to speed up the building process. Hence, Zometool is a good candidate to replace solid 3D printing materials as the inner structure of targeted large-scale structure. Therefore, the goal of this work is to develop a computational method that combine Zometool structure and 3D printing, in order to reduce the time and material costs while large-scale fabrication.

In this paper, we present a novel method, called ZomeFab, automatically generate both outer shell and inner Zometool structures approximating a given 3D input model. We first performs mesh segmentation to split the complex 3D model into different shape parts. For each part, we fit the smallest cube of Zometool structure as initial inner structure. From the initial Zometool structure, we use Simulated Annealing algorithm to effectively explore the huge structure space. Then, with the optimized Zometool structure, we hollowed the shape to obtain the outer shell, and partition the outer shell with respect to several criteria including simplicity and printability. We formulate these criteria in a single MRF problem and solved it using graph cut algorithm. We also design a special types of connectors and optimize their positions for assemble the inner Zometool structure and outer shell.

There are two primary contributions in this paper:

1. The proposed method formulate an optimization to compute the inner Zometool structure. Filled by Zometool structure instead of solid printed is greatly reduce the printing cost including time and materials.

2. We design and print a special connector and optimize their layout for better combine both inner Zometool structure and outer printed shell.

The rest of this paper is organized as follows. In Section 2, we surveyed several previous methods for computational fabrication and applications of modeling system Zometool. Section 3, Section 4 and Section 5 describe the three main stages of our method: *Zometool construction*, *surface partition* and *Fabrication*, respectively. In Section 6, we demonstrate results of our method. Finally, Section 7 concludes this paper.

2 Related Work

Computational Fabrication In recent years, computational fabrication has attracted many attentions in the computer graphics and human computer interaction research fields [Shamir et al. 2016]. Numerous works are proposed to fabricate shapes (i) with different objectives, e.g. balance [Prévost et al. 2013; Bächer et al. 2014], size [Luo et al. 2012], structure soundness [Zhou et al. 2013] and sounds [Umetani et al. 2016], and (ii) using different materials or building blocks, e.g. Lego [Luo et al. 2015], planar slices [Cignoni et al. 2014], and interlocking puzzles [Song et al. 2012].
Although the fast development of assist tools and algorithms, 3D printers still suffer from long production time, excessive material usage, and limited output size. To reduce the usage of print materials, Huang *et al.* [2016] and Wu *et al.* [2016] design devices and algorithms to print shapes in wireframe. Meanwhile, different internal structures are developed, the skin-frame structure by Wang *et al.* [2013], the honeycomb-like structure by Lu *et al.* [2014], and 2D laser cutting shape proxy by Song *et al.* [2016]. To enable the large shape to be printed using 3D printers, Luo *et al.* [2012] developed an iterative planar-cut method, aiming to fit decomposed parts in the 3D printing volume while considering factors such as assemblability and aesthetics. Yao *et al.* [2015] proposed a level-set framework for 3D shape partition and packing. Compared to these works, our method construct a shape with Zometool structure and 3D-printed parts, so that we can reduce the fabrication time and cost, with the reusability of Zometool structures.
Zometool Design and Modeling Zometool is a mathematically-precise plastic construction set for building a myriad of geometric structures, from simple polygons to visualize and model various natural sciences, e.g. DNA molecules. It's history dates back to to the 1960s where it started out as a simple construction system inspired by Buckminster Fulleresque geodesic domes, and it evolved from simple toy to versatile modeling tools through years. Although we can use it to construct complex structure, it's not intuitive for naive users to use and meanwhile time-consuming.
Tools are developed to help users to design the Zometool structures, e.g. vZome [S.Vorthmann.] and ZomeCAD [E.Schlapp.]. These systems provide different ways to grow the structure. However, the difficulties remain when it comes to build a complex structure because it lacks the ability to provide useful suggestions about what kinds of structure to use next in order to build the target shape.
This motivates works toward automatic construction through computational method. Zimmer *et al.* [2014; 2014b] approximate and realize freeform surface automatically using Zometool. Zimmer *et al.* [2014b] adopt an incremental panels growing strategy to approximate the surface without self-collisions. On the other hand, Zimmer *et al.* [2014] first build up a rough initial Zometool structure and explore the modification space using local operations. The final Zometool structure is obtained by a stochastic optimization framework.

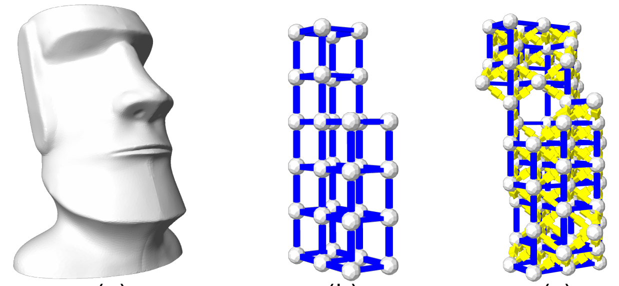

Figure 2: *Given input shape (a), we initialize the Zometool structure with cubes (b), and obtain the optimized result (c) using Simulated Annealing as described in Section 3.*

3 Zometool construction

Our method is aiming for devising an algorithm to construct an object composed of zometool structures and 3D-printed parts with the following objectives:

- *material-effecitiveness* We should aim to minimize the overall fabrication cost and time. Since Zometool is substantially faster to build and reusable, we should maximize it's usage to reduce 3D printing material in the fabrication.
- *easy-to-assemble* We should reduce the difficulties of assemble both Zometool structure and outer printed shell. In terms of Zometool, we should minimize the usage of nodes and rods, so to reduce the assemble time.

3.1 Initialization

We first partition S into m segments ($S = \{s_1, \cdots, s_m\}$) using Shape Diameter Function (SDF) [Shapira et al. 2008] and clustering implemented in CGAL [CGA]. First we want to fill the inner volume of each segment s_i. Although we can use some existing works [Zimmer and Kobbelt 2014b] to generate the inner zometool structure, we intend to use structure that is composed of simple primitives because it's simplicity and easy-assembility. Follow Zimmer *et al.* [2014], we choose to use cube as the basic primitive to initialize the filling. The initialized Zometool structure is denoted as $\bar{\mathcal{Z}} = \{\mathcal{Z}_0, ..., \mathcal{Z}_n\}$, where n is the number of segments with embedded zome tools (see Figure 2(b) for sample initialization).

3.2 Problem Formulation

We measure the quality of the zometools model with an energy E composed of 4 terms accounting to different quality measurements:

$$E(\mathcal{Z}) = w_{\text{dist}} \cdot E_{\text{dist}}(\mathcal{Z}) + w_{\text{reg}} \cdot E_{\text{reg}}(\mathcal{Z}) + w_{\text{val}} \cdot E_{\text{val}}(\mathcal{Z}) + w_{\text{sim}} \cdot E_{\text{sim}}(\mathcal{Z}), \quad (1)$$

3.2.1 Distance

The distance from $\mathcal{Z}$ to S is integrated over all the nodes :

$$E_{\text{dist}}(\mathcal{Z}) = \frac{1}{P \cdot d_{\text{norm}}^2} \sum_{i=1}^{P} \|p_i - \pi(p_i)\|^2 \cdot (1 + F(p_i)), \quad (2)$$

where P is the number of nodes and the normalization factor d_{norm} is used to relate distance to the fixed length of the structs. We follow [Zimmer et al. 2014] and define the term $F(p)$ forbidden zone, which to penalize node points lying too far away from surface. Please see [Zimmer et al. 2014] for more details.

3.2.2 Regularity

In order to obtain a regular Zometool structure for simple assembling, we intend to to regularize the angles between struts to be exact 90° (see Figure 3).

$$E_{\text{reg}}(\mathcal{Z}) = \frac{1}{|\mathcal{N}_i|} \sum_{p_j \in \mathcal{N}_i} (\min(\theta_{ij}) - \frac{\pi}{2}), \tag{3}$$

3.2.3 Valence

We regularize the optimized Zometool structure to have a good valence for simple structure (see Figure 4). We set the target valence as 6 from the initial cube structure.

$$E_{\text{val}}(\mathcal{Z}) = \sum_{i=1}^{P} \frac{(V_p - 6)^2}{6}, \tag{4}$$

where V_p is the valence of each node.

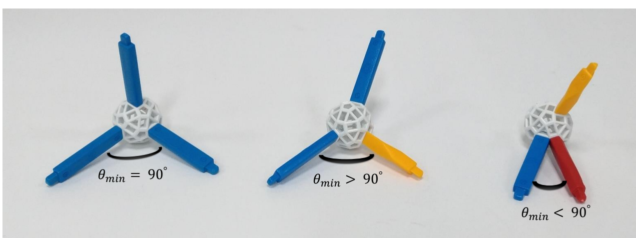

Figure 3: ***Regularity.*** *We penalize the configuration with minimum angle between struts less or greater than* 90°.

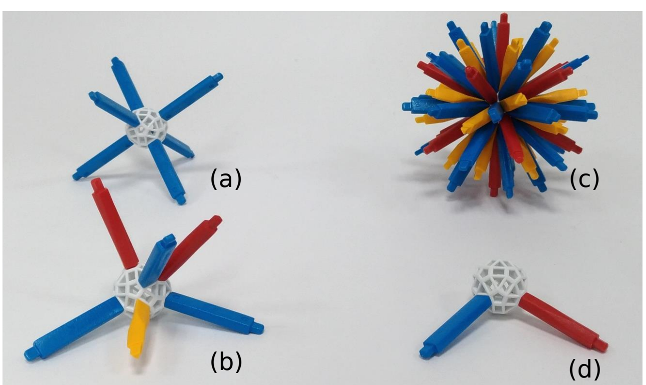

Figure 4: ***Valence.*** *We encourage the valence of each Zometool node to be 6 (as in configuration (a) and (b)). We penalize the valence that is not 6 (configuration (c) and (d)).*

3.2.4 Simplicity

Let N be the total number of both nodes and struts, and N_{target} be the target complexity. The simplicity term is simply encoded as the quadratic differences from the target complexity:

$$E_{\text{sim}}(\mathcal{Z}) = \frac{1}{N_{\text{target}}} (N - N_{\text{target}})^2, \tag{5}$$

3.3 Exploration Mechanism

Searching for the Zometool structure to minimize the energy $E(\mathcal{Z})$ (Eq. 1) is a non trivial optimization problem since $E(\mathcal{Z})$ is non convex and contains global terms. Exhaustive search is impractical and thus we adopt a more scalable strategy based on the Metropolis-Hastings algorithm [Hastings 1970]. In a nutshell, this algorithm makes a random exploration of the solution space by iteratively perturbing the current solution with a certain probability depending on the energy variation between the two solutions and a relaxation parameter T. Following, we describe our local perturbation operators and relaxation scheme. Algorithm 1 details the major steps of our optimization algorith

Algorithm 1 Exploration mechanism

Input: Initialized Zometools $\bar{\mathcal{Z}}$,
relaxation paraemeter $T = T_{init}$
Output: Optimized Zometoos $\mathcal{Z}$
procedure EXPLORATION($\mathcal{Z}$)
    repeat
        generate $\mathcal{Z}'$ from $\mathcal{Z}$ with a random local operation.
        draw a random value $p \in [0, 1]$
        if $p < \exp(\frac{E(\mathcal{Z}) - E(\mathcal{Z}')}{T})$ and CollisionFree($\mathcal{Z}$) **then**
            update $\mathcal{Z}' \leftarrow \mathcal{Z}$
        end if
        Update $T \leftarrow C \times T$ ▷Update temperature.
    until $T < T_{end}$
end procedure

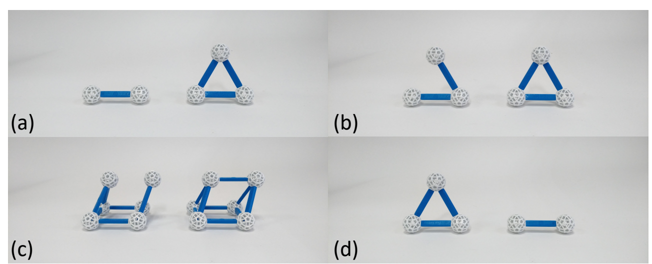

Figure 5: *We use four local operations during structure perturbation. (a) Split, (b) Merge, (c) Bridge, and (d) Kill.*

3.3.1 Local Perturbation Operation

During the exploration, we proposed four local perturbation operations (Figure 5) to construct the Zometool structure by minimizing Eq. 1.

Split This operator insert a new node and two rods to split the original rod.
Merge This operator insert a new rod to merge two disconnected nodes (two node can travel by two edges).
Bridge This operator insert a new rod to merge two disconnected nodes (two node can't travel by two edges).
Kill This operator delete a node and two rods.

3.3.2 Cooling schedule

The relaxation parameter T, referred as temperature, controls both the speed and the quality of exploration. Start from initial temperature T_{init}, we decrease the temperature close to zero as iteration tends to infinity. The decreasing process is referred as cooling, and different cooling schedules are exists for experiment. Although the global minimum convergence is guaranteed using logarithmic cooling schedule [Salamon et al. 2002], we rely on geometric cooling schedule [Henderson et al. 2003]. In our experiment, we set the initial termperature $T_{\text{init}} = 1$, and the decrease rate $C = 0.99$ after every 100 iterations.

4 Surface Partition

We aim to decompose the surface into different partitions for 3D printing, and to cover the optimized Zometool structure $\mathcal{Z}$ from Section 3. Naively, we can simply compute the distance from each triangle t to all the nodes in $\mathcal{Z}$, and assign t to the nearest node as it's label. However, inconsistency may arise among adjacent triangles leading to unsatisfactory visual effects (Figure **??**) and assembly complexities (numerous small partitions might exist). To address this issue, we formulate the problem as a multi-label graph cut minimization. As each triangle t can potentially correspond to different zometool node, it gets assigned data cost for different corresponding nodes. Given n elements, k labels and $n \cdot k$ costs, finding the minimum assignment is a combinatorial problem and typicall NP-hard. We employ Boykov [Boykov and Kolmogorov 2004] to solve it.

4.1 Optimization energy

We compute the assignment function f that assign label to each triangle t, where $t \in T$, such that the labeling f minimize the following eneryg $E(f)$:

$$E(f) = \sum_{t \in T} D(t, f_t) + \sum_{t,s \in N} S(t, s, f_t, f_s), \tag{6}$$

and we optimize this function using multi-label graph-cut algorithm proposed by Boykov *et al.* [Boykov and Kolmogorov 2004]. In our setting, the entire outer nodes of $\mathcal{Z}$ are complete possible label set L. This $E(f)$ consists of three separate terms, i.e. data, smoothness and label costs. Next, we will explain each term in more detail.

4.2 Data cost

Data cost measures how well a triangle t covers a node $p \in P$. This cost is simply defined as the distance of the nearest node to the triangle.

$$D(t, f_t) = -\omega \log(\mathcal{P}(p|t)), \tag{7}$$

where $\mathcal{P}(p|t)$ is the probability of that triangle t belong to the node label p, and ω is a constant that regulates the influence of the data term in the total energy. Here, we simply define $\mathcal{P}(p|t)$ as $1/d(t,p)$, where $d(t,p)$ is the distance of the node to the triangle.

4.3 Smoothness cost

Smoothness term measures the spatial consistency of neighboring elements.

$$S(t, s, f_t, f_s) = \begin{cases} 0, & \text{if } l_t = l_s, \\ -\log(\theta_{t,s}/\pi)\varphi_{t,s}, & \text{otherwise} \end{cases} \tag{8}$$

where $\theta_{p,q}$ and $\varphi_{p,q}$ are the dihedral angle and distance between triangle p and q, respectively. With the smooth term, two adjacent triangles are likely to have consistent labels.

5 Fabrication

We partitioned the surface of the input shape as pieces for fabrication in Section 4. However, the input mesh just have the outer surface, the valid pieces will have both outer and inner surface for 3D printer. After we have the inner surface, the pieces have to generate connection to the zometool structure. After the process we can get all pieces can fabricate, and then we assembled them by connecting to the optimized Zometool structure. In order to generate fabricatable shape, it is practically leave a minimum thickness of the outer shell, and this thickness differs from printer to printer. Thus, we start to prepare the shape to be fabricated by generating a inner surface with a predefined thickness, and the remaining operations are performed within this inner surface.

5.1 Inner surface

There are many potential ways to generate inner surface. The simplest method is shrink the mesh along the vertex normals. Although it is simple, it solely sometimes leads to the triangle flip over the surface. In order to prevent this problem, we use the voxelized mesh to handle it. First, we shrink the mesh along vertex normals with one radius of zometool ball. Second, voxelize the shrinked mesh. Finally, choose the outer surface of the voxelized result and combine the original mesh then we get the new mesh have outer and inner surface. The inner surface generation process is illustrated in Figure 6.

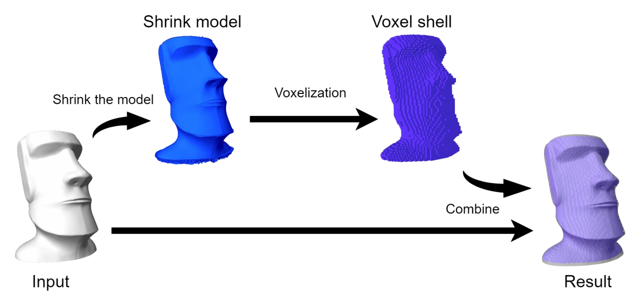

Figure 6: *Inner surface generation method.*

5.2 Generate connector

After get the inner surface, we are able to use the partitioned results from graph cut to cut the surface into pieces. Still, we need to connect the inner Zometool surface with these cutted pieces. Two potential methods for building these connectors are : (i) we can dig holes on the surface and use the Zometool sticks to connect both inner and outer structure (Figure 7(a)), (ii) we can grow Zometool tenons on the surface instead (Figure 7(b)).

5.2.1 Dig holes

If we want to dig some connecting holes on surface, we can't directly dig it because it might break the outer surface. To deal with it, we can put a new ball we called "virtual ball" on the surface. Then we can dig the hole on the virtual ball and thus the outer shell will not be broken. The downside of this approach is that shapes are usually printed with support materials, and the digged hole are usually filled with these materials Due to the low precision issues for most of the consumer 3D printers, it is impractical to generate clean holes for Zometool tenon to plug in. Hence, we design an alternative approcah to generate the connectors.

5.2.2 Grow tenons on surface

Zometool's tenon is a very small object, it fit perfectly on the zometool ball and make the structure be very strong. However, the size of tenon is a strong challenge for the 3D printer due to it's low precision. Beside, same object will be printed differently under different orientations because the way of support printing. In order to verify our printer (Ultimaker 3 [1]) is able to print the tenons, we design an exhausive experiment as follow: we use Ultimaker 3 to print three different tenon of zometool (rectangle, pentagon, triangle), each print in twenty-seven directions (Figure 8). As a result, we found out that even under the lowest precision ("fast print" mode in Ultimaker 3), the printed tenons can still perfectly fit into the slots on the Zometool balls. Compared to dig holes on surface, it is also easier to clean the printed support materials on the printed tenons. We search the nearest node on inner Zometool structure on split piece by graph cut for each triangle. There are sixty-two slots on the Zometool ball, it means we can get sixty-two directions to grow the tenons. We choose the direction which can grow most tenons on the surface to make a strong connection on the inner Zometool structure.

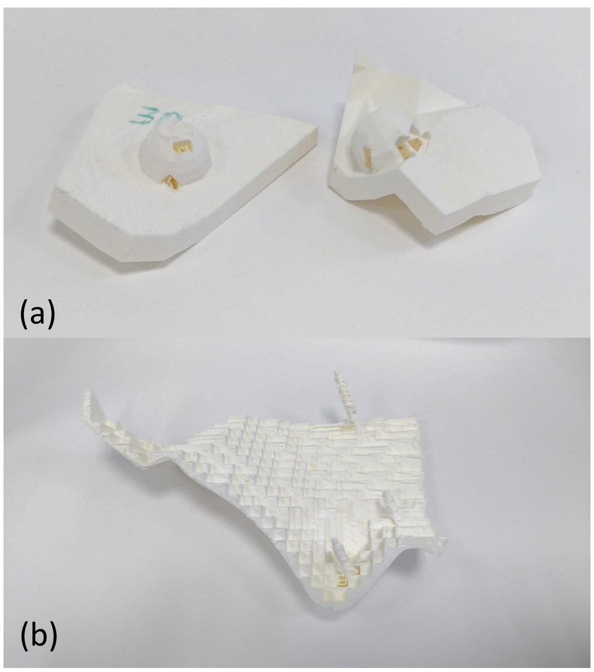

Figure 7: *Experiment: (a) dig hole (b) grow tenon on surface.*

6 Results

Model	Zometool										Printing pieces
	Blue			Red			Yellow			Ball	
	S	M	L	S	M	L	S	M	L		
Maoi	112	0	0	0	0	0	140	0	0	73	11
Squirrel	127	17	1	12	4	0	119	3	0	82	17

Table 1: *Material usage for each result. Including Zometool and printing pieces.*

[1] https://ultimaker.com/en/products/ultimaker-3

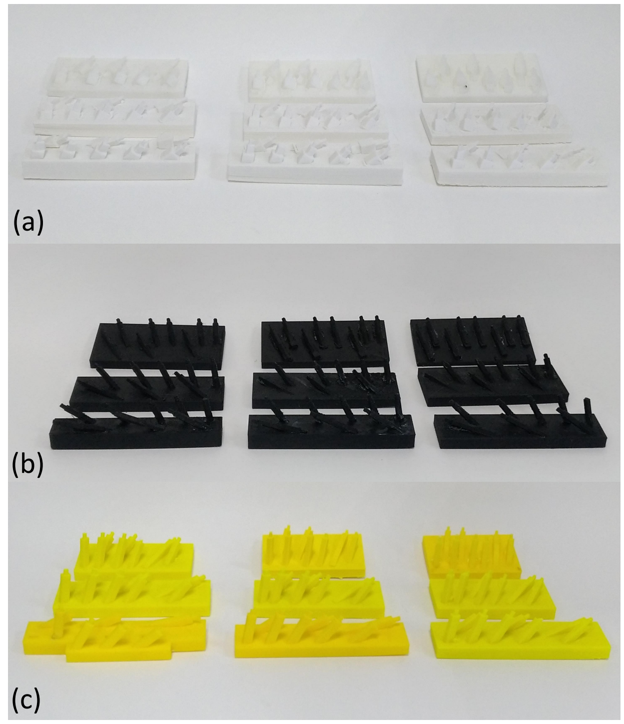

Figure 8: *Experiment: print different types of zometool struts: (a) rectangle (b) pentagon (c) triangle. We designed an experiment to make sure the 3D printer can print tiny object such as the Zometool tenon in all possible configurations. And the same size tenon won't be deformed by different printing route.*

6.1 Experiment environment

We implement ZomeFab in C++ and execute it on desktop PC with 3.4GHz CPU and 16GB memory. The assemble process are showed in Figure 9 and Figure 10. The inner structure is build by Zometool which can construct a strong structure easily. The outer surface are printed by Ultimaker3, a low-cost FDM printer with 0.2m x 0.2m x 0.2m printing volume and PLA material. Table 1 shows the using amount of Zometool and printing pieces.

6.2 Evaluation

We evaluation the material cost and fabrication time between Zomefab and solid mesh with simple partition. We use CURA (version 2.3.1), a slicer software for 3D printing, help us evaluate it. The software will evaluate the fabrication time and amount of material if the volume of input mesh is printable. In Table 1, we demonstrate four experiments for each model, print hollowed with 20% infill and print solid by Zomefab and baseline that partitions a solid object into 3D printable parts. All of experiments show that our method cost less material than baseline. However, the fabrication time will more than baseline in 20% infill rate. We propose that our mesh's complexity is higher than baseline testing mesh. It will cause the printing route be complicated. On the other hand, if the user want to get a exquisite result, our system can get lower cost and fabrication time.

Model	Infill Method	Fabrication Method	Material Cost (US$)			Fabrication Time (hours)			Efficiency (Saved)	
			3D printing	Zometool	Overall (sum)	3D printing	Zometool	Overall (sum)	Material	Time
Maoi	Hollow	Zomefab	83.71	70.57	154.28	134.25	2.5	136.75	16.02%	-23.83%
		Baseline	183.89		183.89	110.43		110.43		
	Solid	Zomefab	95.15	70.57	165.72	169.57	2.5	180.07	76.12%	71.82%
		Baseline	693.96		693.96	639		639		
Squirrel	Hollow	Zomefab	144.88	77.55	222.43	200.63	3.0	203.63	11.66%	-51.02%
		Baseline	251.78		251.78	134.83		134.83		
	Solid	Zomefab	375.67	77.55	453.22	438.48	3.0	441.48	54.68%	34.71%
		Baseline	1000		1000	676.21		676.21		

Table 2: *ZomeFab's performance on saving material and time as compared to a baseline method.*

Figure 9: *Result fabricated and assembly : Maoi*

7 Conclusion

ZomeFab is system which combine Zometool and 3D printing to fabricate a large-scale 3D objects. In this approach, we aimed to represent an input 3D model by a inner structure and pieces of outer surfaces. The inner structure is greatly saving cost of 3D printing compare to directly print partitioned input model. The outer surface still remain the fine geometric characteristic of 3D printer. Thanks to the Zometool reusability, the long term cost of fabrication quietly decreased. However, Zometool is a complex geometric system which can build thousands of structure. Our method generate a Zometool result which can simply build and also well fit outer surface. User can get the pretty well result by using Zomefab.

7.1 Limitation and Future work

Zomefab relied on the input mesh which have the large inner volume and using Zometool as inner structure. The smallest Zometool cube (4.7cm x 4.7cm x 4.7cm) is limited. Therefore, in order to get the initial structure have to make sure input mesh's inner volume must larger than one unit cube. In the future, we can change the unit cube into different and smaller shape to get better structure fitting inside the mesh.

8 致謝

本論文感謝科技部經費補助，計畫編號：MOST103-2221-E-002-158-MY3 與 MOST105-2218-E-002-011。

References

BÄCHER, M., WHITING, E., BICKEL, B., AND SORKINE-HORNUNG, O. 2014. Spin-It: Optimizing moment of inertia for spinnable objects. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH) 33*, 4, 96:1–96:10.

BOYKOV, Y., AND KOLMOGOROV, V. 2004. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence 26*, 9, 1124–1137.

CGAL, Computational Geometry Algorithms Library. http://www.cgal.org.

CIGNONI, P., PIETRONI, N., MALOMO, L., AND SCOPIGNO, R. 2014. Field-aligned mesh joinery. *ACM Trans. Graph. 33*, 1, 11:1–11:12.

E.SCHLAPP. ZomeCAD. [online].

HAO, J., FANG, L., AND WILLIAMS, R. E. 2011. An efficient curvaturebased partitioning of largescale stl models. *Rapid Prototyping Journal 17*, 2, 116–127.

HASTINGS, W. K. 1970. Monte carlo sampling methods using markov chains and their applications. *Biometrika 57*, 1, 97–109.

HENDERSON, D., JACOBSON, S. H., AND JOHNSON, A. W. 2003. *The Theory and Practice of Simulated Annealing*. Springer US, Boston, MA.

HU, R., LI, H., ZHANG, H., AND COHEN-OR, D. 2014. Approximate pyramidal shape decomposition. *ACM Trans. Graph. 33*, 6 (Nov.), 213:1–213:12.

HUANG, Y., ZHANG, J., HU, X., SONG, G., LIU, Z., YU, L., AND LIU, L. 2016. Framefab: Robotic fabrication of frame shapes. *ACM Trans. Graph. 35*, 6 (Nov.), 224:1–224:11.

LU, L., SHARF, A., ZHAO, H., WEI, Y., FAN, Q., CHEN, X., SAVOYE, Y., TU, C., COHEN-OR, D., AND CHEN, B. 2014. Build-to-last: Strength to weight 3d printed objects. *ACM Trans. Graph. 33*, 4 (July), 97:1–97:10.

LUO, L., BARAN, I., RUSINKIEWICZ, S., AND MATUSIK, W. 2012. Chopper: Partitioning models into 3D-printable parts.

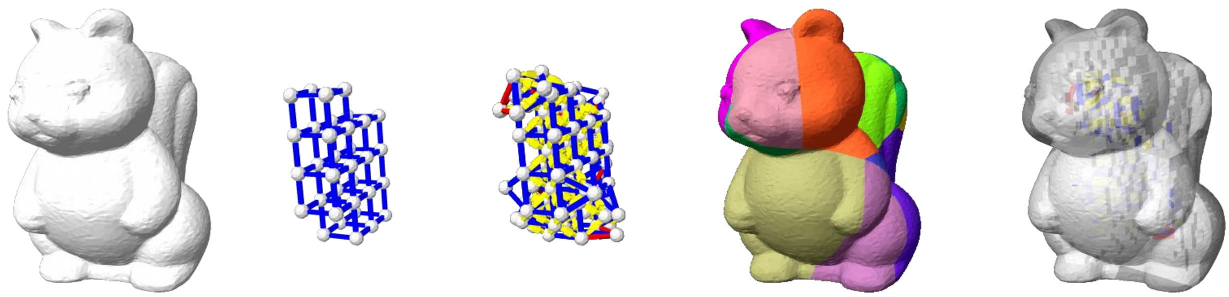

Figure 10: *Result fabricated and assembly : Squirrel*

ACM Transactions on Graphics (Proc. SIGGRAPH Asia) 31, 6 (Dec.).

LUO, S.-J., YUE, Y., HUANG, C.-K., CHUNG, Y.-H., IMAI, S., NISHITA, T., AND CHEN, B.-Y. 2015. Legolization: Optimizing lego designs. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 2015) 34*, 6, 222:1–222:12.

MEDELLN, H., LIM, T., CORNEY, J., RITCHIE, J., AND DAVIES, J. 2007. Automatic subdivision and refinement of large components for rapid prototyping production. *Journal of Computing and Information Science in Engineering 7*, 3 (9), 249–258.

PRÉVOST, R., WHITING, E., LEFEBVRE, S., AND SORKINE-HORNUNG, O. 2013. Make It Stand: Balancing shapes for 3D fabrication. *ACM Transactions on Graphics (proceedings of ACM SIGGRAPH) 32*, 4, 81:1–81:10.

SALAMON, P., FROST, R., AND SIBANI, P. 2002. *Facts, conjectures, and improvements for simulated annealing*. Society for Industrial and Applied Mathematics.

SHAMIR, A., BICKEL, B., AND MATUSIK, W. 2016. Computational tools for 3d printing. In *ACM SIGGRAPH 2016 Courses*, ACM, New York, NY, USA, SIGGRAPH '16, 9:1–9:34.

SHAPIRA, L., SHAMIR, A., AND COHEN-OR, D. 2008. Consistent mesh partitioning and skeletonisation using the shape diameter function. *The Visual Computer 24*, 4, 249.

SONG, P., FU, C.-W., AND COHEN-OR, D. 2012. Recursive interlocking puzzles. *ACM Transactions on Graphics (SIGGRAPH Asia 2012) 31*, 6 (December), 128:1–128:10.

SONG, P., DENG, B., WANG, Z., DONG, Z., LI, W., FU, C.-W., AND LIU, L. 2016. CofiFab: Coarse-to-fine fabrication of large 3d objects. *ACM Transactions on Graphics (SIGGRAPH 2016) 35*, 4. Article 45.

S.VORTHMANN. vZome [online].

UMETANI, N., PANOTOPOULOU, A., SCHMIDT, R., AND WHITING, E. 2016. Printone: Interactive resonance simulation for free-form print-wind instrument design. *ACM Trans. Graph. 35*, 6 (Nov.), 184:1–184:14.

VANEK, J., GALICIA, J. A. G., BENES, B., MECH, R., CARR, N. A., STAVA, O., AND MILLER, G. S. P. 2014. Packmerger: A 3d print volume optimizer. *Comput. Graph. Forum 33*, 6, 322–332.

WANG, W., WANG, T. Y., YANG, Z., LIU, L., TONG, X., TONG, W., DENG, J., CHEN, F., AND LIU, X. 2013. Cost-effective printing of 3D objects with skin-frame structures. *ACM Trans. Graph. 32*, 6, 177:1–177:10.

WU, R., PENG, H., GUIMBRETIÈRE, F., AND MARSCHNER, S. 2016. Printing arbitrary meshes with a 5dof wireframe printer. *ACM Trans. Graph. 35*, 4 (July), 101:1–101:9.

YAO, M., CHEN, Z., LUO, L., WANG, R., AND WANG, H. 2015. Level-set-based partitioning and packing optimization of a printable model. *ACM Trans. Graph. 34*, 6 (Oct.), 214:1–214:11.

ZHOU, Q., PANETTA, J., AND ZORIN, D. 2013. Worst-case structural analysis. *ACM Trans. Graph. 32*, 4 (July), 137:1–137:12.

ZIMMER, H., AND KOBBELT, L. 2014. Zometool rationalization of freeform surfaces. *IEEE Trans. Vis. Comput. Graph. 20*, 10, 1461–1473.

ZIMMER, H., AND KOBBELT, L. 2014. Zometool rationalization of freeform surfaces. *IEEE Transactions on Visualization and Computer Graphics 20*, 10 (Oct), 1461–1473.

ZIMMER, H., LAFARGE, F., ALLIEZ, P., AND KOBBELT, L. 2014. Zometool shape approximation. *Graphical Models 76*, 5, 390–401.