By Example Synthesis of Three-Dimensional Porous Materials

Hui Zhang[a], Weikai Chen[a], Bin Wang[b], Wenping Wang[a,*]

[a]*Department of Computer Science, The University of Hong Kong, Hong Kong*
[b]*School of Software, Tsinghua University, Beijing, China*

Abstract

Porous materials are ubiquitous in nature and are used for many applications. However, there is still a lack of computational methods for generating and modeling complex porous structures. While conventional texture synthesis methods succeed in synthesizing solid texture based on a 2D input, to generate a 3D structure that visually matches a given 3D exemplar remains an open question. We present the first framework that can synthesize porous material that is structurally consistent to input 3D exemplar. In our framework, the 2D texture optimization method is extended built upon 3D neighborhood. An adaptive weighted mechanism method is proposed to reduce blurring and accelerate the convergence speed. Moreover, a connectivity pruning algorithm is performed as post-processing to prune spurious branches. Experimental results demonstrate that our method can preserve both the structural continuity and material descriptors of input exemplar while maintain visual similarity with input structure.

Keywords: texture synthesis, porous material, solid texture, digital fabrication

1. Introduction

Porous structures and materials are widely employed in various applications, from artistic design to mechanical fabrication and biomedical engineering [1]. Due to its large compressive strain and low relative density, porous structures is considered suitable for many material substitutes [2]. Recent research advances have witnessed the keen interest in porous material as it can undertake special tasks that general solid materials fail to accomplish, for example bone replacement and tooth implant.

Application-wise, exemplar-driven synthesis of porous materials is demanding as features of input exemplar (i.e. interior geometry and material statistic properties) are expected to be preserved in many application scenarios. For instance, in bone replacement, the inner structure of healthy bone, which is of porous form, needs to be replicated inside artificial bone so as to be implanted into human body [3]. Due to the intricate nature of such porous structures, manual design remains extremely tedious while conventional methods fail to accomplish this task. Over the last two decades, various methods have been put forward to model porous structures. However, they are mainly limited to model regular and repetitive structures [4]. To address these issues, we leverage the advantages of texture synthesis and propose a novel method to synthesize porous materials directly from a given 3D exemplar.

Texture synthesis [5, 6, 7, 8] is a well-established method in computer graphics. It usually uses 2D digital data of physical textures as input to generate large scale textures with similar appearance. However, to synthesize a 3D structure that is visually similar to a 3D exemplar remains a challenging problem. 2D textures are not competent to encode the structural complexity of a 3D sample as it cannot capture the information in its perpendicular dimension. Our experiments have shown that the conventional solid texture synthesis methods based on 2D inputs are not able to synthesize 3D porous material well by examples. In

*Corresponding author
Email address: wenping@cs.hku.hk (Wenping Wang)

this paper, we present the first 3D porous structure synthesis framework that directly uses the 3D exemplar as input and could preserve well both the structural connectivity and material property of input exemplar.

Specifically, we extend the 2D texture optimization method to 3D. The synthesized 3D texture *becomes* structure that resembles the input. Compared to 2D texture synthesis, blurring is more detrimental in our case as it introduces discontinuity in synthesized structures. To overcome the blurring issue and accelerate the synthesis speed, we propose a novel adaptive weighting mechanism in the energy optimization phase. To improve the connectivity of synthesis result, we apply a pruning step as post-processing which can either remove protruding branches or enhance internal connections according to different scenarios. Experimental results demonstrate that our method can preserve both the structural continuity and material descriptors of input exemplar while maintain visual similarity with input structure. To further validate our method, we fabricate the synthesized results with 3D printers.

The remainder of the present article is organized as follows. Section 2 reviews the methods used in porous structure modeling and 3D solid texture synthesis. Section 3 gives an overview of our 3D porous material synthesis method and more details are presented in Section 4. Results and discussions are shown in section 5, and the conclusion is given in Section 6.

2. Previous Work

2.1. Porous Structure Modeling

Most three dimensional porous modeling techniques apply the reconstruction methodology to mimic the full or partial scale of existing porous structures. Generally these techniques can be described as a process that applies statistical or stochastic models for reconstructing 3D porous media from 2D images or 3D voxel data. With scanning electron microscopy, digitized serial sections of samples can be obtained as the source of reconstruction [9]. Later the use of computed tomography (CT) and X-ray (μCT) give significant insight into the pore geometry, which makes reconstruction from high-resolution voxel data possible [10, 11, 12].

The creation and design of porous structures are mainly limited to regular and periodic shapes [13, 14, 15, 16]. In modeling of irregular porous structures, Wyvill et al. [17] firstly proposed a method to construct 2D pore inscribed within Voronoi cells with implicit signed distance function. Kou et al. [18, 19] resorted to stochastic Voronoi diagram and B-spline representation to create 3D porous structures. Schroeder et al. [20] present a representation of porous artifacts based on stochastic geometry method. Researchers from geophysics try to take advantage of the fact that the pore structures are often shaped in physical processes. For instance, the construction of an analog sandstone can be produced by stochastically modeling the results of different sandstone forming processes [21].

2.2. Example based Texture Synthesis

Example based texture synthesis has been intensively studied in the past twenty years due to its wide applications. Many elegant methods have been proposed, including the pixel based methods [22, 5], the patch based methods [6, 23, 7] and the optimization based methods [8, 24, 25], and so on. We only give a brief review to those relevant to our work. Kwatra et al. [8] formulated the texture synthesis problem as minimization of an energy function. This energy function is optimized in an iterative way similar to the Expectation Maximization algorithm. Compared to most example-based algorithms, Kwatra et al.'s optimization method is more suitable to control the scale and direction of the synthesized results. This pioneer article inspired many subsequent researches.

Kopf et al. [25] first extended the 2D texture optimization idea into solid textures synthesis. By integrating the color histogram matching into the optimization approach, the convergence process is sped up and the quality of result is improved. Zhou et al. [26] deals with texture synthesis with connected components along a surface. Lefebvre and Hoppe [27] proposed a method of using appearance vectors for coherent anisometric and surface texture synthesis. Dong et al. [28] proposed a fast "lazy" texture synthesis around the surface. Chen and Wang [29] proposed a method integrating the position and index histogram matching into the optimization phase. These two new matching methods force the the synthesized solid textures and the exemplars share the same global statistics resulting in high quality output. Takayama et al. [30] deals

with tileable solid textures of different types. Zhang et al. [31] proposed a sketch guided method to synthesize solid texture. A tensor filed is derived from the sketch curves to guide the direction of the textures. Du et al. [32] introduced a semiregular solid texturing method by analyzing the shape and distribution of the particles from the image. Shu et al. [33] proposed a method combining procedural and example based method to generate particles which are stored as points in cellular textures. Liu and Shapiro [34] applied texture synthesis to synthesize random heterogeneous materials from 2D image and firstly demonstrated the effectiveness of texture synthesis method in preserving the material descriptors of input exemplar. However, all of these methods are based on 2D example. Straightforwardly employing these methods to synthesize porous materials usually produces very poor results. Our method also belongs to the optimization based methods, however, we take the 3D exemplar as input and develop several novel techniques to generate results with good statistic material properties.

3. Overview

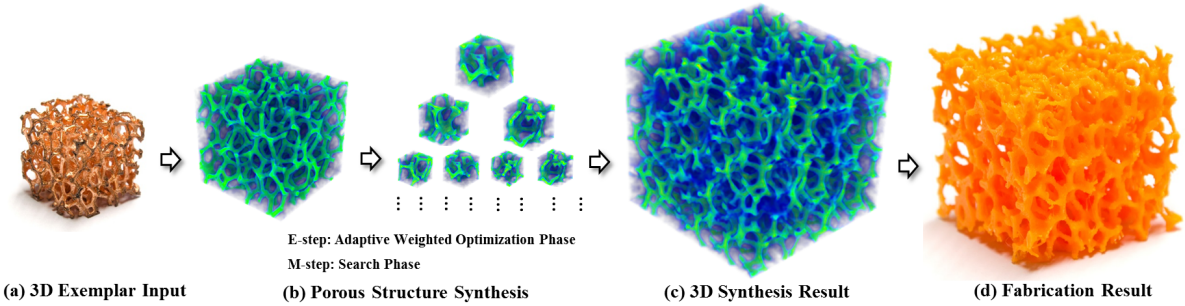


Figure 1: Flowchart of our proposed approach.

Given a 3D exemplar of input porous structure (Figure 1a), we firstly digitalize the exemplar via CT scanning. The resulting grayscale data (Figure 1b) is an assembly of voxels with intensity value ranging from 0 to 255. In preprocessing, the input structure is captured via extracting an iso-surface from the scanned volumetric data. In particular, the regions with intensity value higher than the iso-value will be treated as solid; otherwise we mark it as void. The iso-value is selected using the automatically segment method [35] which can choose the most suitable threshold to extract the iso-surface. This iso-value can best preserve the structure of input exemplar and will be used to extract the synthesized structure in the final result.

The synthesis of porous structure was then formulated as a texture optimization problem. To well capture the 3D geometry of input exemplar, we propose the first cubic-neighborhood based texture optimization method. Although the basic framework is extended based on its 2D counterpart (Kwatra et al. [8]), the challenging nature of complex structure synthesis demands a careful tuning of parameters. In our framework, we aim to minimize a global energy function which measures the mismatch between the synthesized result and the input exemplar. To preserve the structural continuity during synthesis, we design an energy function including not only a least square term, that measures mismatches of input and output neighborhoods, but also a gradient energy term which can maintain the structure information.

We solve this energy minimization problem via an iterative Expectation Maximization algorithm. In the search phase(Maximization), the output $\mathcal{S}$ is fixed and the best matched neighborhood cubes are found in input exemplar via kd-tree search. In the optimization phase(Expectation), the set of closest neighborhoods remains fixed while the output voxels are solved and updated. Blurring is a common issue in 2D texture synthesis. The issue is even more complicated in our scenario as cubic neighborhood would lead to more conflicting regions for adjacent voxels. To resolve this issue, we propose an adaptive weighted mechanism method for solving the output $\mathcal{S}$. Similar to the robust optimization of Kwatra et al. [8], adaptive weighted mechanism will avoid the immoderate influence of the found nearest neighborhood if it is not very close to

its corresponding cube in the output. Moreover, the adaptive weight can further accelerate the convergence speed. In contrast to solid textures, volumetric porous material is very sensitive to the discontinuity of structures. To further improve the structural continuity of resulting structure, we perform a connectivity pruning step as post processing. The whole pipeline is summarized in Figure 1.

4. Porous Structure Synthesis

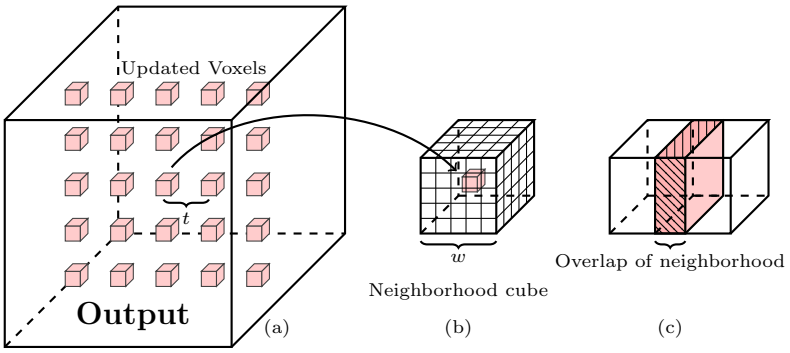


Figure 2: Illustration of neighborhood in our method. For simplicity, we only demonstrate the sampled voxels of the sparse grid on a bisector plane in the left figure.

4.1. Energy Function

Let $\mathcal{S}$ denote the set of output voxels. For a neighborhood with width w, we define $\mathcal{N}_p$ as the 3D cubic neighborhood in $\mathcal{S}$ centered at voxel p (as shown in Figure 2b). Via concatenating all the intensity value of voxels inside $\mathcal{N}_p$, we denote s_p as the vectorized version of $\mathcal{N}_p$. The closest neighborhood of s_p in input exemplar $\mathcal{E}$ is denoted by e_p. Here, e_p has the lowest l_2 norm difference with s_p compared to other voxels belonging to $\mathcal{E}$.

We then define the global energy function over $\mathcal{S}$ as following:

$$E(\mathcal{S},\mathcal{E}) = \sum_{p\in\mathcal{S}'} w_p \left\|s_p - e_p\right\|^2 + \mu \sum_{p\in\mathcal{S}'} \left\|G(s_p) - G(e_p)\right\|^2 \tag{1}$$

In Equation 1, we only consider neighborhood of voxels belonging to a subset $\mathcal{S}' \in \mathcal{S}$. $\mathcal{S}'$ is uniformly sampled from $\mathcal{S}$ with step size t (as shown in Figure 2a). The reason to do sampling in $\mathcal{S}$ is because it would be redundant and computationally expensive to consider the neighborhood matching over all voxels in the output. Instead, we allow the 3D neighborhood of each sampled voxel to have a proper amount of overlap ($w > t$)(Figure 2c). Computing the energy over a sparse grid will also reduce blurring as less voxels are overlapped in the conflicting regions. We will detail the selection of t and w in Section 5.1.

The first term of the energy function penalizes the intensity difference between the exemplar and the synthesized result. Here, w_p is the adaptive weight controlling the impact of closest neighborhood with respect to its similarity with s_p. We will detail w_p in Section 4.3. The second term of Equation 1 seeks to minimize the difference of gradients between the matching pair of neighborhood. Let a denotes a single voxel in the vector s_p or e_p. (a, b) denotes the adjacent pair at a, where b is the adjacent voxel of a including the up, down, left, right, front and back six voxels. Thus the gradient term can be represented as $\left\|G(s_p) - G(e_p)\right\|^2 = \sum_{(a,b)} \left\|s_{p,a} - s_{p,b} - (e_{p,a} - e_{p,b})\right\|^2$ in the energy function.

In our experiment, we tend to use a smaller value of μ if the input exemplar is filled with thin features. Introducing the gradient term helps enhancing the smoothness of the solid/void structure in the output and in turn preserving the structural continuity in the synthesized result. We will demonstrate the effectiveness of our framework in the experimental result in Section 5. To minimize the energy function, we iterate

between the *search* phase and the *optimization* phase. In the search phase, we find the best matching neighborhood e_p for each s_p in the output $\mathcal{S}$ while s_p is solved and updated using adaptive weighted method in the optimization phase. To accelerate the synthesis, we optimize the output in a multi-level and multi-resolution fashion. The detailed parameters are discussed in Section 5.1.

4.2. Search Phase

The search phase, which corresponds to the M-step of our algorithm, seeks to minimize $E(\mathcal{S}, \mathcal{E})$ via finding nearest neighbors $\{e_p\}$ from the input for a fixed $\mathcal{S}$. In 2D texture synthesis, the closest neighborhood searching usually dominates the running time of entire optimization. The computational cost is even higher in our case due to the higher dimension of searching space. Hence we accelerate the neighborhood searching in multiple ways.

We firstly reduce the dimension of neighborhood vectors via PCA projection [36]. Specifically, we only keep the eigenvalues that add up to around 95% of the total sum. For a typical neighborhood size 10, which leads to 1000-dimensional neighborhood vector, the resulting vector drops to 30-40 dimensions after PCA projection. The compression of data is even more dramatically with larger size of cubic neighborhood. We then build a kd-tree for all the dimension-reduced vectors based on l_2 norm metric. In practice, neighborhood searching is likely to get stuck in local minimum as the closest neighborhood is pointing to the same cube in the exemplar. Therefore, we employ the index histogram matching in Chen and Wang [29] to restrict the selection of nearest neighborhood. In particular, we select k (k is set to 10 in our implementation) nearest neighbors from kd-tree search. Then the k candidates are sorted again by a modified distance $d = w_d \|s_p - e_p\|^2$, where $w_d = 1 + \max[0, H(i(e_p)) - \theta]$. Here $H(i(e_p))$ refers to the histogram value of the index of closest neighborhood e_p; and θ equals to $\frac{1}{|\mathcal{E}|}$, which is a constant reflecting the possibility of picking a neighborhood vector in exemplar under equal-probability distribution. If $H(i(e_p))$ is larger than θ, d will be increased, making e_p harder to be selected.

4.3. Optimization Phase

In the optimization phase, we minimize $E(\mathcal{S}, \mathcal{E})$ with respect to s_p. As we mentioned in Section 3, overlapping of neighborhoods would incur blurring in conflicting regions. The blurring issue is more complicated in 3D as volumetric neighborhoods usually result in more overlapping regions for adjacent voxels. Conventional least square estimation fails to distinguish the impact of e_p and thus may lead to immoderate influence on s_p when the mismatch between e_p and s_p is large. However, on the other hand, we expect s_p not to be changed when e_p is very close to it. Leveraging the idea of robust optimization in Kwatra et al. [8], we propose an adaptive weighted mechanism to further improve the performance of 3D texture synthesis. Specifically, the adaptive weight w_p is formulated as follows:

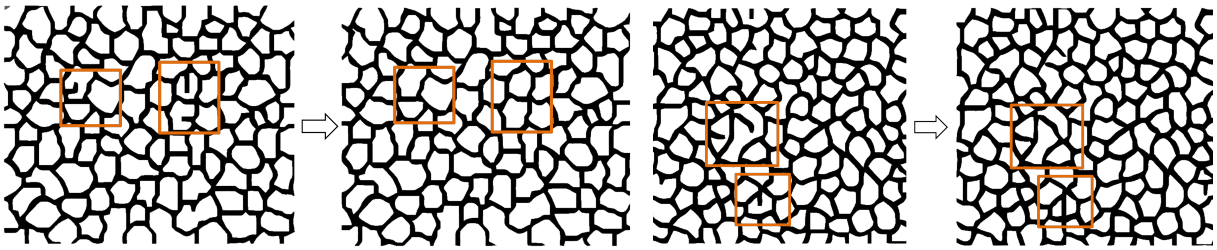
$$w_p = \sum_{p \in \mathcal{S}'} \lambda_{k+1} \left\| s_p - e_p \right\|^{r-2}, \quad \lambda_{k+1} := \begin{cases} (1+\varepsilon_i)\lambda_k, & E(\mathcal{S},\mathcal{E})_{k+1} < E(\mathcal{S},\mathcal{E})_k \\ (1-\varepsilon_d)\lambda_k, & E(\mathcal{S},\mathcal{E})_{k+1} > E(\mathcal{S},\mathcal{E})_k \\ \lambda_k, & E(\mathcal{S},\mathcal{E})_{k+1} = E(\mathcal{S},\mathcal{E})_k \end{cases} \tag{2}$$

where $r \in [0, 2]$ and $E(\mathcal{S}, \mathcal{E})_{k+1}$ is the global energy at $(k+1)$p-th iteration. λ is the adaptive parameter that controls the scaling of weight in different scenarios while ε_i and ε_d are two positive constants. In practice of texture energy minimization, the energy curve would be oscillating which leads to more iterations to converge. The proposed adaptive weighted method introduces feedback control mechanism that can speedup the convergence. Specifically, Equation 2 would provide a positive feedback to increase weights of least square difference if the global mismatch error is decreased compared to previous iteration ($E(\mathcal{S}, \mathcal{E})_{k+1} < E(\mathcal{S}, \mathcal{E})_k$). The larger weight would lead to a faster minimization of global energy as the decrease of $E(\mathcal{S}, \mathcal{E})$ has indicated a correct configuration of weights. However, if the global energy is increasing ($E(\mathcal{S}, \mathcal{E})_{k+1} > E(\mathcal{S}, \mathcal{E})_k$), a negative feedback will take effect to hamper the weights according to Equation 2, prohibiting the increasing momentum of $E(\mathcal{S}, \mathcal{E})$. The combined effect of such feedback control mechanism contributes to a faster and more robust minimization of the objective function. Figure 5 compares the number of iterations with and without the proposed adaptive weighted mechanism. As seen from result, our method (orange curve) converges in 10 less iterations than the counterpart without using adaptive weights. In experiments, we

usually set $r = 0.8$, $\lambda_0 = 1$ (initialized as 1 at the first iteration) while ε_i and ε_d equal to 0.05 and 0.3, respectively.

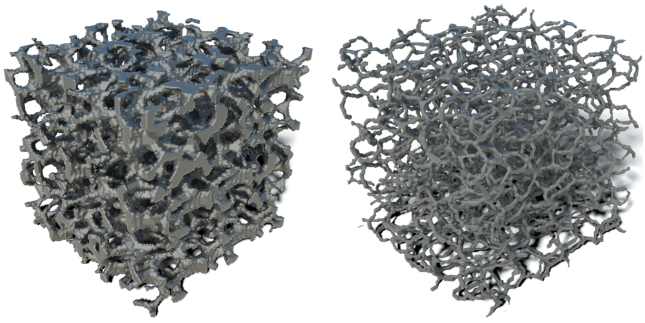
To minimize the global energy $E(\mathcal{S}, \mathcal{E})$, which includes both a weighted least square and a gradient energy term, we resort to a hybrid optimization method that alternates between discrete Poisson solver [37] and iteratively reweighted least square (IRLS) solver [38]. For each iteration, we firstly compute w_p, given a fixed set of nearest neighborhoods $\{e_p\}$. Then the voxel values of $\mathcal{S}$ are solved using the discrete Poisson solver. At the end of each iteration, the weights are adjusted according to Equation 2 and the iterations are repeated until the global energy variation is smaller than a threshold or the maximum number of iterations is reached.

4.4. Connectivity Pruning



(a) Remove protruding branches (b) Add new connections

Figure 3: Illustration of connectivity pruning in 2D.



(a) Volumetric data (b) 3D voxel medial axis

Figure 4: 3D Voxel thinning

Thanks to texture optimization framework, the porous structures can be faithfully replicated in the synthesized result. However, synthesis artifacts are difficult to be completely avoided. For instance, as shown in results in the left column of Figure 3(a) and (b), some spurious branches (highlighted in orange rectangles) which do not appear in the input are introduced during synthesis. In many applications of tissue engineering, such protruding branches will be detrimental to the growth of implanted tissues. Aesthetics-wise, the spurious branches are not visually pleasant. In addition, for 3D digital fabrication, we, therefore, present a post-processing step to prune the connectivity of resulting structures.

To spot the artifacts, we firstly extract binary topology of synthesized result using previous exemplar threshold and compute voxel medial axis (Figure 4). Here we use voxel thinning [39, 40] to iteratively removes voxels on the boundary until a skeleton of single-voxel width is obtained. We treat the voxel skeleton as an undirected graph in which each voxel acts a node and the immediate neighborhood between two adjacent

voxels is treated as an edge. Following the identification of joint nodes (nodes that have more than two immediate neighbors) on the voxel skeleton, the protruding parts are easy to be spotted via analyzing loops and branches in the graph. We considers two scenarios in connectivity pruning step (as shown in the 2D illustration in Figure 3). (1) If the spurious branch b is short compared to the perimeter L of its enclosing cell $\mathcal{C}$ (Figure 3(a) left), we directly remove it. (2) On the other hand, we will elongate b to form new connections to its adjacent cell, if its length is relatively large. The elongation is performed in an inconspicuous ways via copying similar branch structure in the exemplar. In our implementation, a threshold θ (θ equals to 0.11 in our implementation) is introduced so that if the length of b is smaller than θL, we remove b; otherwise, new connections are added.

5. Results Comparison and Discussion

5.1. 3D Volumetric Synthesis

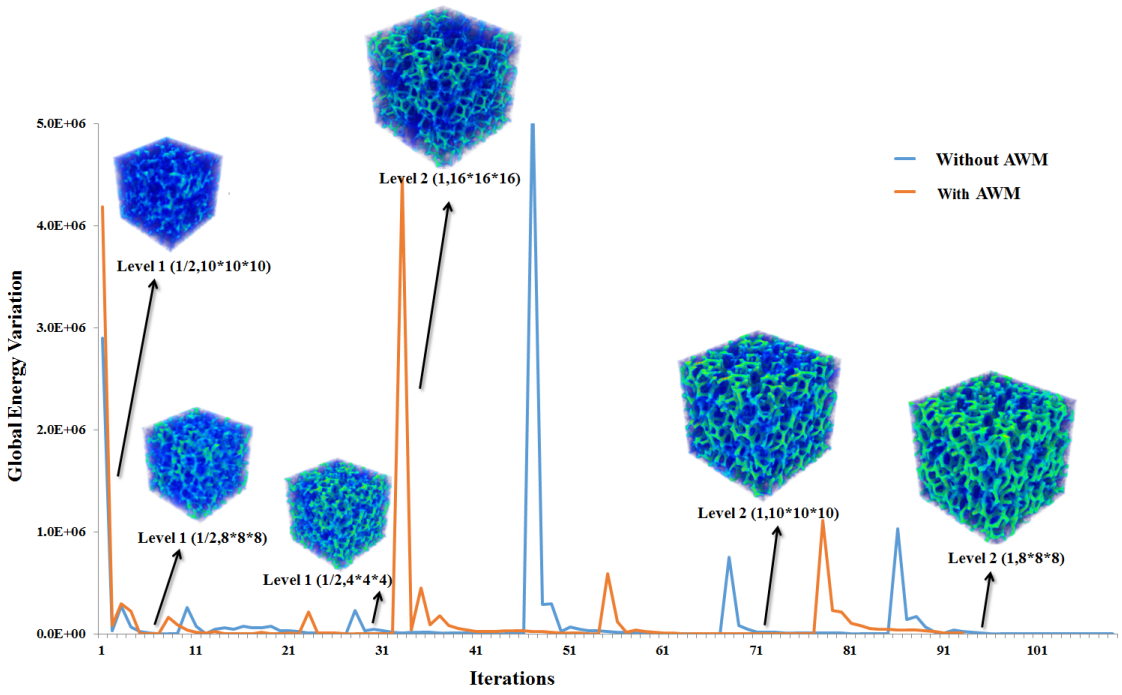


Figure 5: The global energy variation comparison with and without adaptive weighted mechanism(AWM). The results of multi-level and multi-resolution synthesis are also demonstrated. Level 1($1/2$, $8*8*8$) shows the result with $1/2$ resolution using $8*8*8$ neighborhood size. The other synthesis results are denoted in a similar way.

For all of our experiments, we use two-level resolutions and three different neighborhood sizes within each level (as demonstrated in Figure 5). At the first level, we synthesize the result with half resolution with coarse-to-fine neighborhood sizes (from $10*10*10$ to $4*4*4$). After level 1 is accomplished, a full-resolution synthesis is deployed in level 2. We initialize level 2 by upsampling (trilinear interpolation) the output of level 1. The synthesis is then proceeded following a similar multi-resolution manner. The synthesis result at each stage can be found in Figure 5.

In the experiments, we consider the rotation and symmetries in building the input neighborhood sets. Instead of adding rotation or symmetry items to energy function, the input exemplar is rotated and applied

symmetry in order to build more potential neighborhoods. We only consider the internal voxels which neighborhood is inside the boundary of exemplar and choose the voxel that can generate complete neighborhood. If voxels lie close to the boundary of input example, we do not generate its neighborhood. We initialize the output domain by randomly copying 3D neighborhoods from exemplar to the output domain. In the searching step, applying modified distance to all candidates will result in a huge increase of runtime. We found by experiments that only testing the k candidates with modified distance can speed up the computation and will not affect the performance in the meanwhile. As shown in the results at different levels and resolutions in Figure 5, the output is progressively refined. The intensity value are color-coded in order to visualize the inner structures. As shown in Figure 6, regardless of entirely different input exemplars, our method is capable to generate plausible results that are visually similar to the input. Figure 7 demonstrates the synthesis result in free-form shapes using the foam Cu as input exemplar. Thanks to the connectivity pruning, the bunny ear is enhanced and therefore connected to other parts (as demonstrated in the closeup on the right).

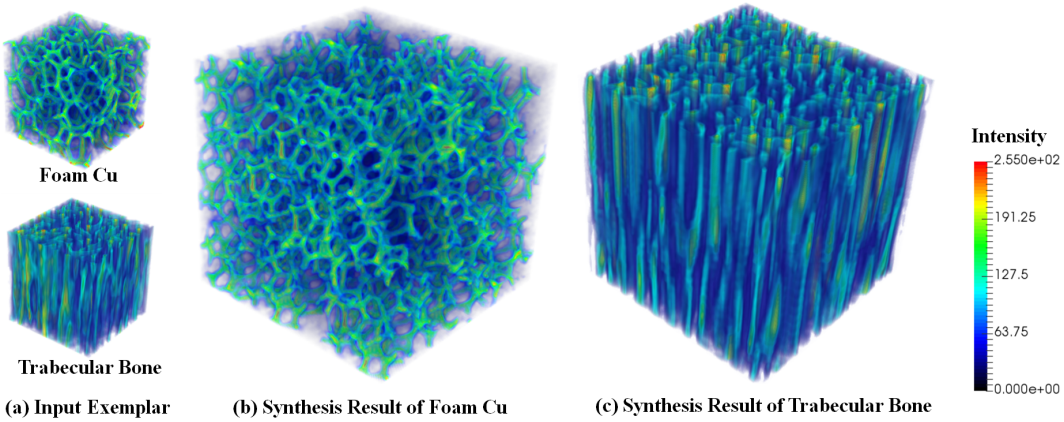


Figure 6: Synthesis results using two different input exemplars.

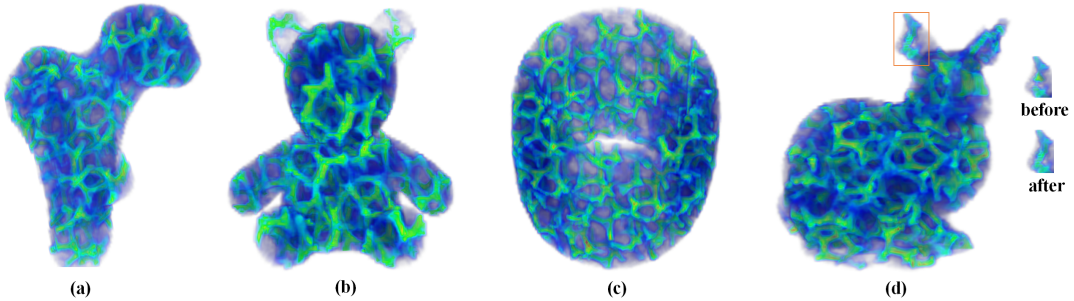


Figure 7: Synthesis results in free form shapes.

There exists blurring problem during synthesis which greatly influence appearance of synthesized results (as shown in the highlighted areas of Figure 8). The blurring is introduced by the average of the overlapping neighborhoods in optimization phase when the two neighborhoods don't match well. As the synthesized structure is extracted from the voxels with the same intensity value, blurring of voxels tends to eliminate features of output structure, leading to small isolated parts that are not continuous (Figure 8(b)). Figure 8 shows the effect with and without AWM. As seen from the results, the proposed adaptive weighted mechanism can effectively remove blurs and thus strongly enhance the integrity of the synthesized structure.

In order to test the stability of the synthesize result, we performed an experiment to iteratively synthesize a larger piece of structure from a part of previous results. In particular, we randomly extract 80% of the preceding result as the input for the next iteration. Figure 9 shows the results after repeating the iterations several times. Note that Figure 9(a) is the original input reconstructed from CT images. The normalized difference of correlation function between output and input for iteration 1 to 4 are 0.1243, 0.1514, 0.1191 and 0.1511 respectively. We will explain correlation function in Section 5.3.1.

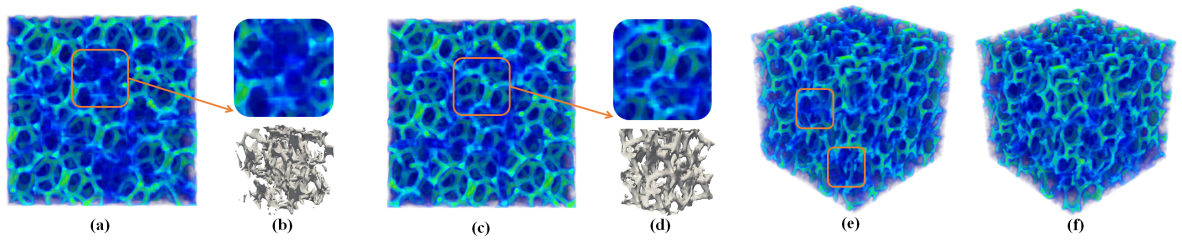


Figure 8: Synthesis results without and with adaptive weighted mechanism (AWM).(a) and (e) shows the result without AWM while (c) and (f) corresponds to the one with AWM. The blurring is highlighted in rectangular boxes.

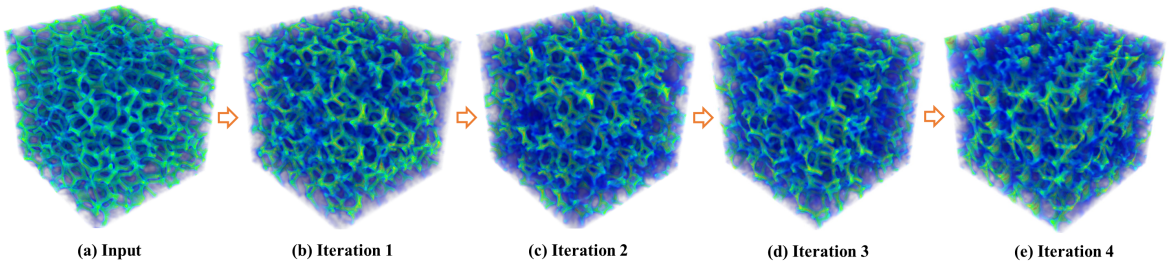


Figure 9: Synthesis results of iterations that use part of previous result as input.

Timing. Our method is implemented using C++. All the results are generated on a computer with a 3.16GHz CPU and 8GB RAM. For the cubic result in Figure 5, the output has the size of 120^3 voxels, which is synthesized from an input of 60^3 voxels. For optimization method with adaptive weighted mechanism, it takes 20 minutes to complete while the method without AWM costs 30 minutes. The total computation time cost by the connectivity pruning operations is usually trivial (around a few seconds) compared to the searching and optimization phase during synthesis.

Parameters. The maximum number of iterations for each resolution of each level is 30. However, our algorithm never reaches the maximum iteration in all of our experiments (usually converged within 20 iterations). There are only a few connectivity pruning operations performed in our results. In particular, we performed 5 and 2 pruning operations for Figure 6(b) and (c), respectively; 7, 9, 3 and 6 times for Figure 7(a), (b), (c) and (d), respectively.

5.2. Comparisons

Figure 10 shows comparisons with previous techniques. As shown in Figure 10(a), from left to right, the input exemplars are foam Ni, foam Cu and trabecular bone respectively. Figure 10(b) shows our results and Figure 10(c) shows the results of Chen and Wang's method [29]. The results in Figure 10(d) are synthesized by Kopf et al.'s method [25]. It is easy to observe that, in comparison with the other two methods, our results share the most visual similarity with the input exemplars, not only from the total synthesis error of

measuring the Euclidean distance of the intensity value, but also from the direct sense of human's eyes. For the trabecular bone exemplar, the results of the two other methods look reasonable. However, for the foam Ni and foam Cu exemplars, their results suffer a lot from the structure discontinuity.

Note that the two methods in [25] and [29] can only take 2D images as input, we have tried both using one exemplar slice and three orthogonal slices as input of their algorithms. The results shown in Figure 10 are the best results we experimented for the two methods. One possible explanation for this discontinuity phenomenon is that a few texture slices cannot capture the whole structure information in 3D spaces.

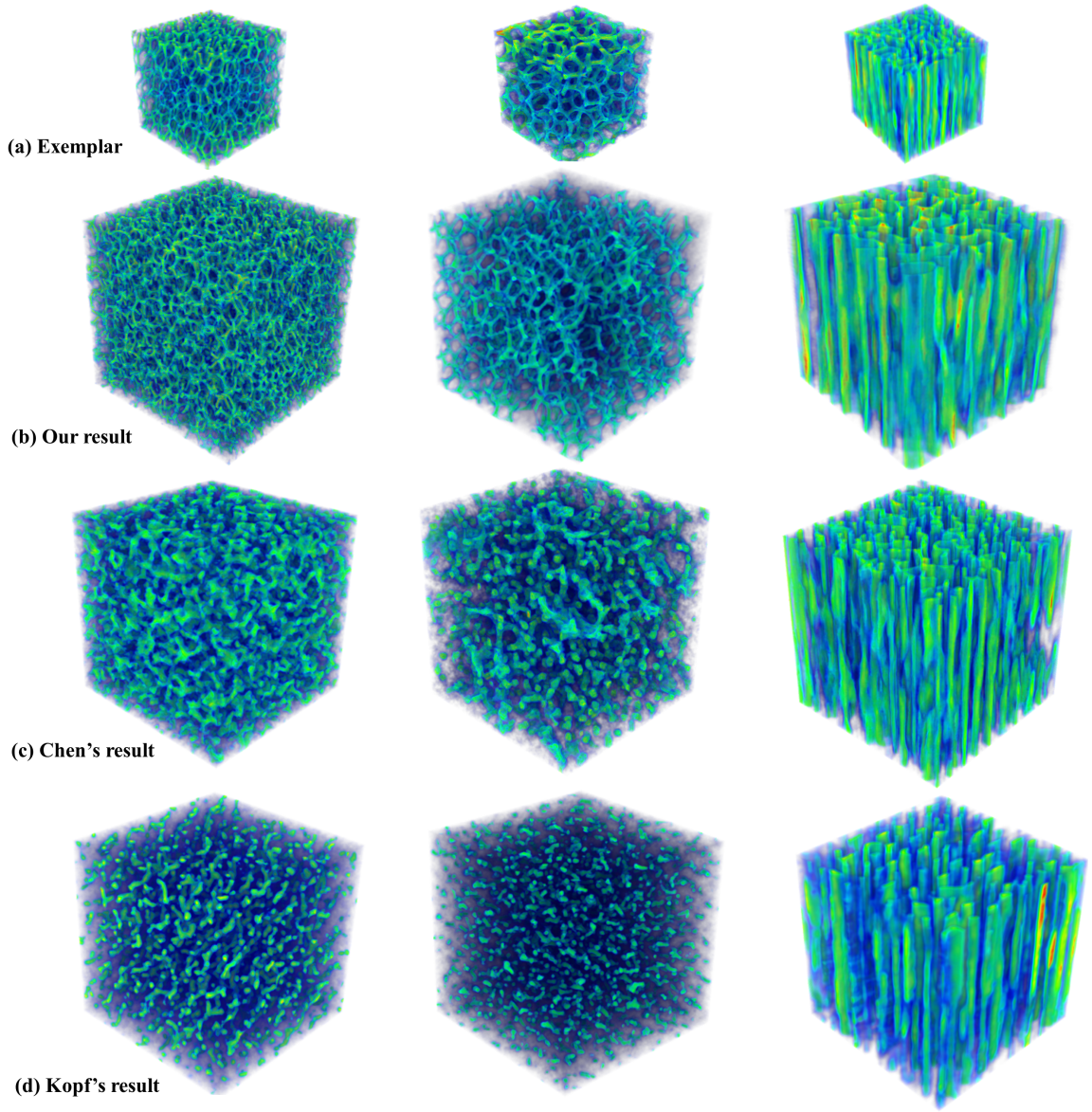


Figure 10: Comparisons of porous structure synthesis. (a) Input exemplar; (b) Synthesis result of our method; (c) Result of Chen's method; (d) Result of Kopf's method.

5.3. Preservation of Statistic Material Properties

Our method is capable to preserve the material properties of input exemplar. We validate the effectiveness of our framework in terms of correlation function and intensity histogram in this section.

5.3.1. Correlation function

Correlation functions is used widely in stochastic geometry and stereology to quantitatively measure the distribution of distances between different phases. An n-point correlation function can be defined as the probability that all vertices of a random n-vertex polyhedron falls into the same phase of a material microstructure. In our validation, we employ two-point correlation function, in which a line segment is used for phase query. Figure 11 compares the correlation function curves of our synthesized result with the other methods using different input exemplars. As seen from the result, the correlation curve of our methods has best approximation to that of input exemplar no matter in term of trabecular bone or foam material.

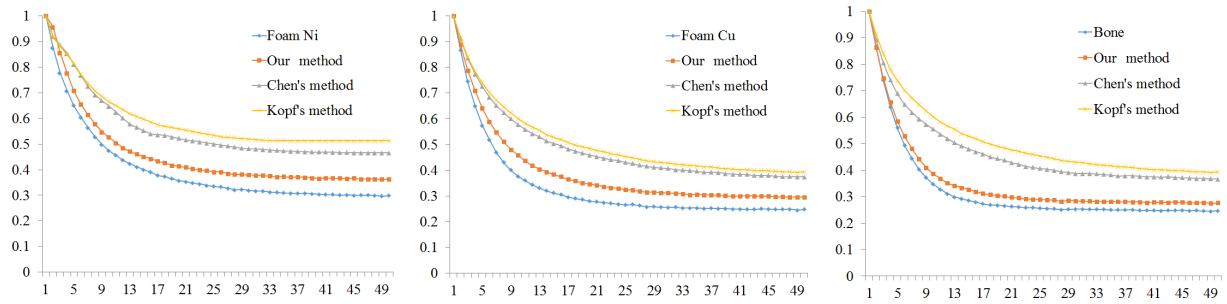


Figure 11: Comparisons of correlation function curves of different method synthesized result with respect to input exemplars (from left to right: Foam Ni, Cu and Bone). The horizontal axis is the length of a query line segment while the vertical axis is the probability of that segment falling into same phase.

Material \ Method	Our Method	Kopf's Method	Chen's Method
Foam Ni	0.1367	0.3691	0.4488
Foam Cu	0.1598	0.4195	0.4747
Bone	0.0897	0.3239	0.4339

Table 1: Normalized difference between correlation function of synthesized results and exemplar

Table 1 compares the normalized l_2 norm difference between correlation function of synthesized results and that of input exemplar. It is obvious that our method outperforms the state-of-the-art solid texture synthesis methods (Kopf et al. [25] and Chen and Wang [29]) in terms of the correlation function errors.

5.3.2. Intensity Histogram

Intensity histogram is another important descriptor for measuring the statistic material property. Figure 12 shows the comparison of intensity histogram obtained from different methods. The intensity value of voxels ranges from 0 to 255. To visualize the result better, we cluster the voxels into 10 bins shown in the horizontal axis. Each bin contains 25 intensity values, while the 10th-bin contains the values ranges from 225-255. The vertical axis is the appearance frequency of the intensity value. As shown from Figure 12, the histogram of our method is closest to the input exemplar while both Kopf et al. [25] and Chen and Wang [29] deviate obviously from that of exemplar.

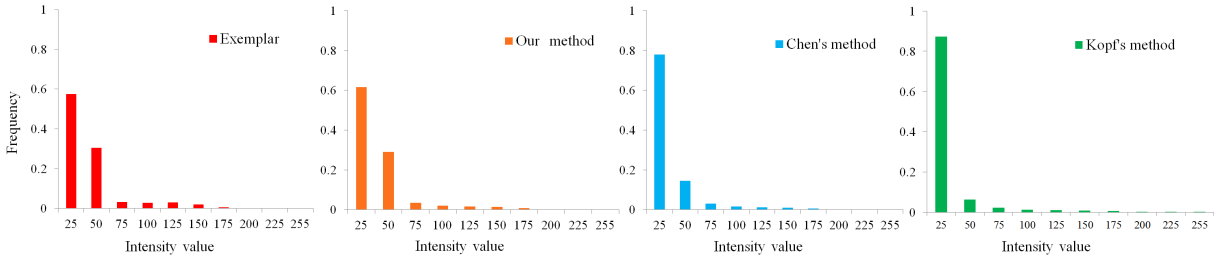
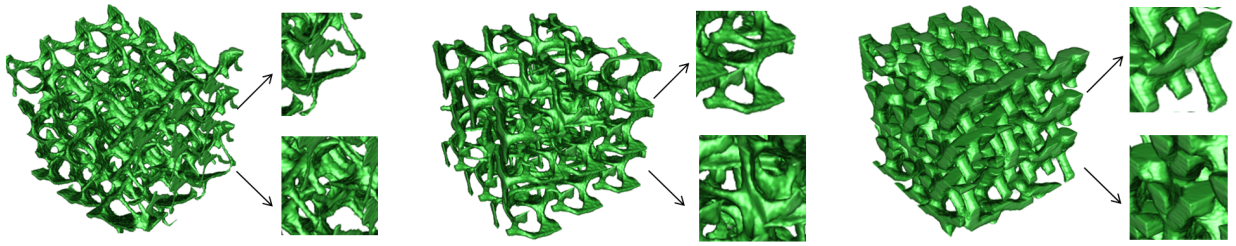


Figure 12: Comparison of histogram between input exemplar and synthesis result.

5.4. Digital Fabrication

To further validate the robustness of the results, we fabricate them with 3D printer. We have demonstrated that our synthesized results are visually similar to the input exemplars, sharing the same global statistics with the input exemplars and are structurally continuous over the 3D space after connectivity pruning on the binary result. After pruning the final result is a connected structure without isolated parts which can be fabricated.



(a) Input structure (b) Gaussian smoothing (c) Dilation

Figure 13: Illustration of enhancing the porous structure via morphology operators.

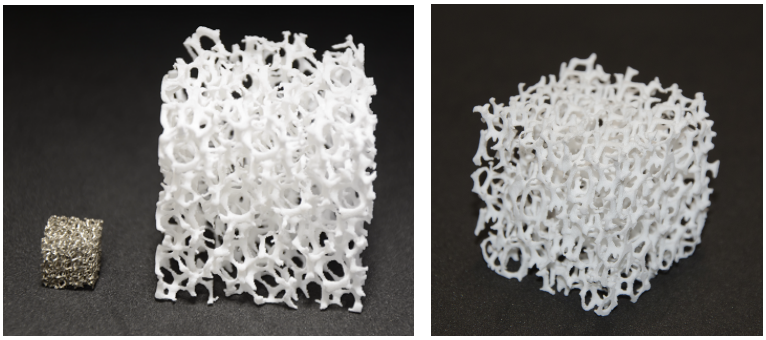


Figure 14: 3D printed results using SLS printer.

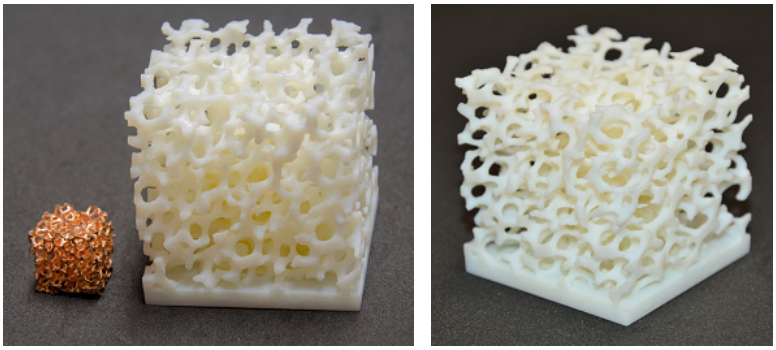


Figure 15: 3D printed results using SLA printer.

We use a threshold to extract the iso-surface from results. In theory, the extracted iso-surface could be directly used as input for 3D printer. However, we find that in some cases, if the synthesis result is too thin and fragile, such as the new added voxels in connectivity pruning, it will not be printable due to its weak structures (shown in Figure 13(a)). In such cases, we enhance the printability of the porous structure via two morphological operations, Gaussian smoothing and dilation. Figure 13 shows the morphology process of our result. The structure in Figure 13(b) and (c) is more robust than the original one and is ready for digital fabrication. Note that the user could easily control the thickness of resulting structure in our pipeline.

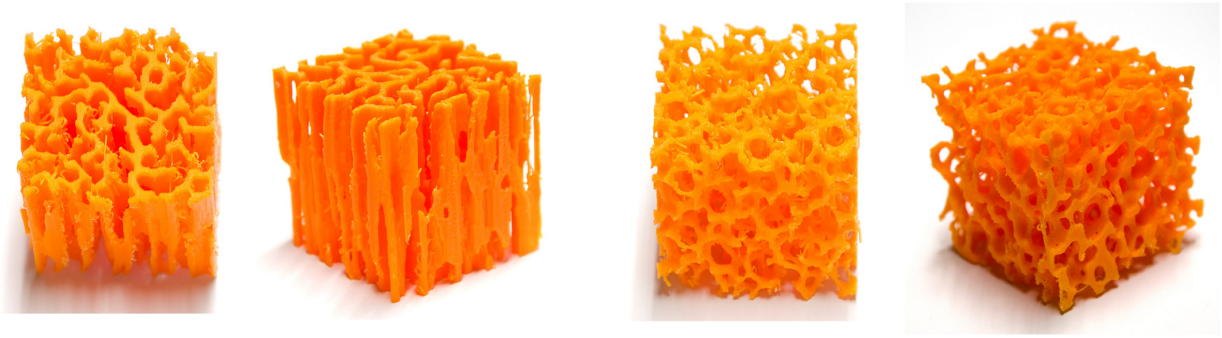


Figure 16: 3D printed results using the low-end FDM printer.

We test our models using different types of 3D printers. Figure 14, Figure 15 and 16 demonstrate the fabrication results using SLS, SLA and FDM printers, respectively. Note that, as shown in Figure 16, our synthesized result can even be fabricated using low-precision FDM 3D printers which proves that our method can well preserve the structural continuity of input exemplar.

6. Conclusion

In this paper, we have presented a new method for synthesizing volumetric porous material from 3D exemplars. While the conventional solid texture synthesis methods cannot synthesize 3D porous structures well from 2D exemplars, we have successfully accomplished this difficult task by extending the texture optimization framework to 3D with our novel adaptive weighted controlling algorithm and connectivity pruning post processing. Experiments show that our method is capable of synthesizing high quality volumetric porous material that can well preserve both the structural continuity and the global statistics of input exemplar. We further validate the results by fabricating them with 3D printer. In the future, we would like

to explore the techniques to further improve the synthesis speed and quality. We believe that our method has the potential of being an easy-to-use volumetric porous structure modeling tool by example.

7. Acknowledgement

We would like to thank the anonymous reviewers for their comments. This work is supported by NSFC (61272019, 61572021) and Hong Kong GRF(HKU 717813E). Thanks Microsoft Research Asia for their support.

References

[1] F. A. Dullien, Porous media: fluid transport and pore structure, Academic press, 2012.
[2] S. J. Hollister, Porous scaffold design for tissue engineering, Nature materials 4 (7) (2005) 518–524.
[3] D. A. Garzon-Alvarado, M. A. Velasco, C. A. Narvaez-Tovar, Self-assembled scaffolds using reaction–diffusion systems: a hypothesis for bone regeneration, Journal of Mechanics in Medicine and Biology 11 (01) (2011) 231–272.
[4] J. Pikunic, C. Clinard, N. Cohaut, K. E. Gubbins, J.-M. Guet, R. J.-M. Pellenq, I. Rannou, J.-N. Rouzaud, Structural modeling of porous carbons: constrained reverse monte carlo method, Langmuir 19 (20) (2003) 8565–8582.
[5] L.-Y. Wei, M. Levoy, Fast texture synthesis using tree-structured vector quantization, in: Proceedings of the 27th annual conference on Computer graphics and interactive techniques, ACM, 2000, pp. 479–488.
[6] A. A. Efros, W. T. Freeman, Image quilting for texture synthesis and transfer, in: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM, 2001, pp. 341–346.
[7] V. Kwatra, A. Schödl, I. Essa, G. Turk, A. Bobick, Graphcut textures: image and video synthesis using graph cuts, ACM Transactions on Graphics (ToG) 22 (3) (2003) 277–286.
[8] V. Kwatra, I. Essa, A. Bobick, N. Kwatra, Texture optimization for example-based synthesis, ACM Transactions on Graphics (ToG) 24 (3) (2005) 795–802.
[9] M. Kwiecien, I. Macdonald, F. Dullien, Three dimensional reconstruction of porous media from serial section data, Journal of Microscopy 159 (3) (1990) 343–359.
[10] J. H. Dunsmuir, S. Ferguson, K. D'Amico, J. Stokes, et al., X-ray microtomography: A new tool for the characterization of porous media, in: SPE annual technical conference and exhibition, Society of Petroleum Engineers, 1991.
[11] J. Pouech, J. M. Mazin, P. Tafforeau, High quality 3d imaging of vertebrate microremains using x-ray synchrotron phase contrast microtomography, Comptes Rendus Palevol 9 (6-7) (2010) 389–395.
[12] X. Zhu, S. Ai, X. Lu, K. Cheng, X. Ling, L. Zhu, B. Liu, Collapse models of aluminum foam sandwiches under static three-point bending based on 3d geometrical reconstruction, Computational Materials Science 85 (0) (2014) 38–45.
[13] S. Cai, J. Xi, A control approach for pore size distribution in the bone scaffold based on the hexahedral mesh refinement, Computer-Aided Design 40 (10) (2008) 1040–1050.
[14] I. Zein, D. W. Hutmacher, K. C. Tan, S. H. Teoh, Fused deposition modeling of novel scaffold architectures for tissue engineering applications, Biomaterials 23 (4) (2002) 1169–1185.
[15] D. J. Yoo, Porous scaffold design using the distance field and triply periodic minimal surface models, Biomaterials 32 (31) (2011) 7741–7754.
[16] D. Yoo, New paradigms in internal architecture design and freeform fabrication of tissue engineering porous scaffolds, Medical Engineering and Physics 34 (6) (2012) 762–776.
[17] B. Wyvill, P. G. Kry, R. Seidel, D. Mould, Determining an aesthetic inscribed curve, in: Proceedings of the Eighth Annual Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging, Eurographics Association, 2012, pp. 63–70.
[18] X. Y. Kou, Modeling functionally graded porous structures with stochastic voronoi diagram and b-spline representations, in: S. T. Tan (Ed.), International Conference on Manufacturing Automation, IEEE, 2010, pp. 99–106.
[19] X. Y. Kou, S. T. Tan, Microstructural modelling of functionally graded materials using stochastic voronoi diagram and b-spline representations, International Journal of Computer Integrated Manufacturing 25 (2) (2012) 177–188.
[20] C. Schroeder, W. C. Regli, A. Shokoufandeh, W. Sun, Computer-aided design of porous artifacts, Computer-Aided Design 37 (3) (2005) 339–353.
[21] P. E. Ren, S. Bakke, Process based reconstruction of sandstones and prediction of transport properties, Transport in Porous Media 46 (2-3) (2002) 311–343.
[22] A. A. Efros, T. K. Leung, Texture synthesis by non-parametric sampling, in: Proceedings of the 7th IEEE International Conference on Computer Vision, Vol. 2, IEEE, 1999, pp. 1033–1038.
[23] L. Liang, C. Liu, Y.-Q. Xu, B. Guo, H.-Y. Shum, Real-time texture synthesis by patch-based sampling, ACM Transactions on Graphics (ToG) 20 (3) (2001) 127–150.
[24] J. Han, K. Zhou, L.-Y. Wei, M. Gong, H. Bao, X. Zhang, B. Guo, Fast example-based surface texture synthesis via discrete optimization, The Visual Computer 22 (9-11) (2006) 918–925.
[25] J. Kopf, C.-W. Fu, D. Cohen-Or, O. Deussen, D. Lischinski, T.-T. Wong, Solid texture synthesis from 2d exemplars, ACM Transactions on Graphics (TOG) 26 (3) (2007) 2.
[26] K. Zhou, X. Huang, X. Wang, Y. Tong, M. Desbrun, B. Guo, H.-Y. Shum, Mesh quilting for geometric texture synthesis, in: ACM Transactions on Graphics (TOG), Vol. 25, ACM, 2006, pp. 690–697.

[27] S. Lefebvre, H. Hoppe, Appearance-space texture synthesis, ACM Transactions on Graphics (TOG) 25 (3) (2006) 541–548.
[28] Y. Dong, S. Lefebvre, X. Tong, G. Drettakis, Lazy solid texture synthesis, Computer Graphics Forum 27 (4) (2008) 1165–1174.
[29] J. Chen, B. Wang, High quality solid texture synthesis using position and index histogram matching, The Visual Computer 26 (4) (2010) 253–262.
[30] K. Takayama, M. Okabe, T. Ijiri, T. Igarashi, Lapped solid textures: filling a model with anisotropic textures, in: ACM Transactions on Graphics (TOG), Vol. 27, ACM, 2008, p. 53.
[31] G.-X. Zhang, S.-P. Du, Y.-K. Lai, T. Ni, S.-M. Hu, Sketch guided solid texturing, Graphical Models 73 (3) (2011) 59–73.
[32] S.-P. Du, S.-M. Hu, R. R. Martin, Semiregular solid texturing from 2d image exemplars, IEEE Transactions on Visualization and Computer Graphics 19 (3) (2013) 460–469.
[33] Y. Shu, Y. Qian, H. Sun, Y. Chen, Efficient texture synthesis of aggregate solid material, The Visual Computer 30 (6-8) (2014) 877–887.
[34] X. Liu, V. Shapiro, Random heterogeneous materials via texture synthesis, Computational Materials Science 99 (2015) 177–189.
[35] H. R. Buie, G. M. Campbell, R. J. Klinck, J. A. MacNeil, S. K. Boyd, Automatic segmentation of cortical and trabecular compartments based on a dual threshold technique for in vivo micro-ct bone analysis, Bone 41 (4) (2007) 505–515.
[36] A. Hertzmann, C. E. Jacobs, N. Oliver, B. Curless, D. H. Salesin, Image analogies, in: Proceedings of the 28th annual conference on Computer graphics and interactive techniques, ACM, 2001, pp. 327–340.
[37] P. Pérez, M. Gangnet, A. Blake, Poisson image editing, ACM Transactions on Graphics (TOG) 22 (3) (2003) 313–318.
[38] D. Coleman, P. Holland, N. Kaden, V. Klema, S. C. Peters, A system of subroutines for iteratively reweighted least squares computations, ACM Transactions on Mathematical Software (TOMS) 6 (3) (1980) 327–336.
[39] T.-C. Lee, R. L. Kashyap, C.-N. Chu, Building skeleton models via 3-d medial surface axis thinning algorithms, CVGIP: Graphical Models and Image Processing 56 (6) (1994) 462–478.
[40] D. Legland, I. Arganda-Carreras, P. Andrey, Morpholibj: integrated library and plugins for mathematical morphology with imagej, Bioinformatics 32 (22) (2016) 3532–3534.