

STS-GAN: Can We Synthesize Solid Texture with High Fidelity from Arbitrary 2D Exemplar?

Xin Zhao¹, Jifeng Guo⁴, Lin Wang^{2,3,*}, Fanqi Li^{2,6}, Jiahao Li², Junteng Zheng⁵, Bo Yang^{3,2,*}

¹ Shandong Provincial Key Laboratory of Preparation and Measurement of Building Materials, University of Jinan, Jinan 250022, China

² Shandong Provincial Key Laboratory of Network Based Intelligent Computing, University of Jinan, Jinan 250022, China

³ Quan Cheng Laboratory, Jinan 250100, China

⁴ School of Computer Science and Engineering, South China University of Technology, Guangzhou 510641, China

⁵ David R. Cheriton School of Computer Science, University of Waterloo, Waterloo, ON N2L 3G1, Canada

⁶ Shandong Qiuqi Analysis Instrument Co., Ltd., Jinan 250024, China

Abstract

Solid texture synthesis (STS), an effective way to extend a 2D exemplar to a 3D solid volume, exhibits advantages in computational photography. However, existing methods generally fail to accurately learn arbitrary textures, which may result in the failure to synthesize solid textures with high fidelity. In this paper, we propose a novel generative adversarial nets-based framework (STS-GAN) to extend the given 2D exemplar to arbitrary 3D solid textures. In STS-GAN, multi-scale 2D texture discriminators evaluate the similarity between the given 2D exemplar and slices from the generated 3D texture, promoting the 3D texture generator synthesizing realistic solid textures. Finally, experiments demonstrate that the proposed method can generate high-fidelity solid textures with similar visual characteristics to the 2D exemplar.

1 Introduction

Solid texture synthesis, a technique for mapping 2D textural information to 3D solid, has been applied widely in computer graphics and vision. In particular, the synthesis process relies only on a few 2D exemplars, generally one 2D exemplar.

Solid texture synthesis, in general, extends the visual characteristics of a 2D exemplar into an object whose voxels belong to a volumetric domain $\mathcal{D} \subset \mathbb{R}^3$, sharing a similar internal appearance with the 2D exemplar. During the synthesis process, the color of each voxel in the generated solids is gradually modified by matching exemplars, describing specific appearance properties. Eventually, the overall appearance of the synthesized solid is expected to be similar to the given 2D texture exemplar.

During the last several decades, solid texture synthesis attracted a lot of attention in 3D visualization and volume rendering [Fayolle *et al.*, 2021; Laursen *et al.*, 2011; Iwasaki *et al.*, 2017]. It has also received numerous successful stories in real-world applications, such as material science [Ashton *et al.*, 2020], medical analysis [Wang *et al.*, 2018], and game development [Mark *et al.*, 2015].

1.1 Motivation

Textures usually refer to the visual or tactile experience composed of repeating similar patterns, formally defined as locally stationary, ergodic, stochastic processes [Wei, 2002; Georgiadis *et al.*, 2013]. The textures in the real world typically have three characteristics: *Local Markov property*, *Multiscality*, and *Diversity*. (1) *Local Markov property* means the spatial coherence is highly localized in the neighbourhood. (2) *Multiscality* suggests the spatial coherence could exist at different scales in different ways. (3) *Diversity* means the style domain of patterns in the real world could be vast. Thus, an STS method needs to map the appearance of a 2D exemplar into a 3D solid texture satisfying the three characteristics simultaneously.

The traditional STS methods, like statistical feature matching methods [Heeger and Bergen, 1995; Ghazanfarpour and Dischler, 1995; Jagnow *et al.*, 2004] or Markov random field-based methods [Wei, 2002; Kopf *et al.*, 2007; Chen and Wang, 2010], have received credits in many fields [Mariethoz and Lefebvre, 2014; Turner and Kalidindi, 2016]. Despite some successful stories, these methods fail in accurately projecting 2D appearance into a 3D solid on account of the low expressive power of the model and complexity of real-world applications.

In 2020, [Gutierrez *et al.*, 2020] introduced neural networks into solid texture synthesis, which may herald a fruitful direction. They proposed a convolutional neural network (CNN)-based method to synthesize solid textures, taking full advantage of its hierarchical expressive capability and ex-

*Corresponding authors: Lin Wang (wangplanet@gmail.com), Bo Yang (yangbo@ujn.edu.cn)

tracting features using the VGG [Simonyan and Zisserman, 2015] feature maps. The visual effects of synthesized volumes are at least comparable to the state-of-the-art methods. Similarly, using VGG statistical features, [Henzler *et al.*, 2020] also provided another point operation-based neural network solution, which can efficiently synthesize 3D textures.

These neural network-based STS methods are trained to match features, such as VGG statistics. However, the *diversity* of textures makes it challenging to fit different appearances with fixed features *a priori*. It is almost impossible to capture an infinite number of textural appearances with a limited number of features.

The Generative Adversarial Nets (GANs) [Goodfellow *et al.*, 2014] have been proven to be an effective universal distribution learner, generating diverse images [Bergmann *et al.*, 2017; Shaham *et al.*, 2019]. Following *point operation* strategy from [Henzler *et al.*, 2020], the GramGAN [Portenier *et al.*, 2020] enables synthesizing solid textures without matching fixed features with generative adversarial nets. Despite its adaptability to diverse textures, the adopted point operation in GramGAN, simply providing spatial information as network inputs, often leads to difficulty learning complex spatial coherence. It is hard to capture the *local Markov property* and *multiscality* of textures, failing in generating structured textures or complicated stochastic structures [Portenier *et al.*, 2020].

Question *can we synthesize high-fidelity solid texture, capturing all three characteristics, to faithfully reflect the true 3D appearance of arbitrary 2D exemplars?*

Yes, we can. Aiming to address this issue, we propose a novel GAN-based framework for solid textures synthesis, STS-GAN.

1.2 Contribution

We summarize the main contributions of this paper:

- STS-GAN extends CNN-based STS to enable learning arbitrary texture distribution and to adapt to textural *diversity*.
- Aiming at the *multiscality* of solid texture, a multi-scale strategy is adopted to encourage learning textures hierarchically.
- Experiments exhibit our method generates solid textures with higher fidelity than the other state-of-the-art ones.

2 Related Works

Solid texture synthesis has attracted considerable research interest in the field of computer graphics and vision since it was proposed by [Perlin, 1985] and [Peachey, 1985]. In this field, the procedural methods were the earliest family with the advantage of low computational cost. They synthesize textures using a function of pixel coordinates and a set of manually tuning parameters. As perhaps the most famous example, the Perlin Noise [Perlin, 1985] is a smooth gradient noise function that is used to create pseudo-random patterns by perturbing mathematical equations.

Nevertheless, determining a suitable set of parameters for the desired texture necessitates tedious trial-and-error. Furthermore, the semantic gap inhibits people from linking notions such as marble or gravel with accurate parameters.

By contrast, exemplar-based STS methods can generate a new 3D texture from a given 2D exemplar without relying on the artificially accurate texture description. The following is a brief review of various families of these methods.

2.1 Statistical Feature-Matching Methods

These methods use a set of statistical features extracted from a given texture and apply it to solid textures. The pyramid matching [Heeger and Bergen, 1995] method pioneered the work on solid texture synthesis from 2D exemplars, using an image pyramid to capture the characteristics of textures at various resolutions. It is useful to create stochastic textures. [Ghazanfarpour and Dischler, 1995] presented a solid texture generation method based on the spectral analysis of a 2D texture in various types. [Jagnow *et al.*, 2004] proposed a solid texture synthesis method using stereoscopic techniques, which effectively preserve the structure of texture.

Textures, in general, are diverse and complicated. Statistical feature-based methods tend to synthesize specific textures based on the certain image feature but fail to work on a broad set of textures.

2.2 Markov Random Field-based Methods

These methods model texture as a Markov Random Field (MRF), where each pixel in a texture image depends only on the pixels in its surrounding neighborhood. Based on the non-parametric MRF model [Efros and Leung, 1999], Wei *et al.* applied the nearest neighborhood matching strategy coupled with an image pyramid technique to synthesize solid textures [Wei, 2002; Wei and Levoy, 2000].

[Kopf *et al.*, 2007] synthesized 3D solid textures by adopting MRF as a similarity metric. In this method, the color histogram matching forces the global color statistics of the synthesized solid to match those of exemplars. [Chen and Wang, 2010] integrated position and index histogram matching into the MRF optimization framework using the k-coherence search [Tong *et al.*, 2002], effectively improving the quality of synthetic solids. In general, while these MRF-based techniques may capture hierarchical texture features and generate outstanding results, the conflict between texture diversity and the difficulty of learning a non-parametric MRF model precludes them from producing high-quality solid textures.

2.3 Neural Nets-based Methods

Recently, neural networks have been used to synthesize solid textures because of their capability to approximate any non-linear functions.

To synthesize realistic volumetric textures, a CNN-based method [Gutierrez *et al.*, 2020] has been introduced, taking advantage of CNN's powerful expressive capability for spatial autocorrelation data. It takes part of VGG-19 as an image descriptor to conceptualize features extracted from an exemplar. The results prove that it can generate a solid texture of arbitrary size while reconstructing the conceptualized visual features of an exemplar along with some directions. In 2020,

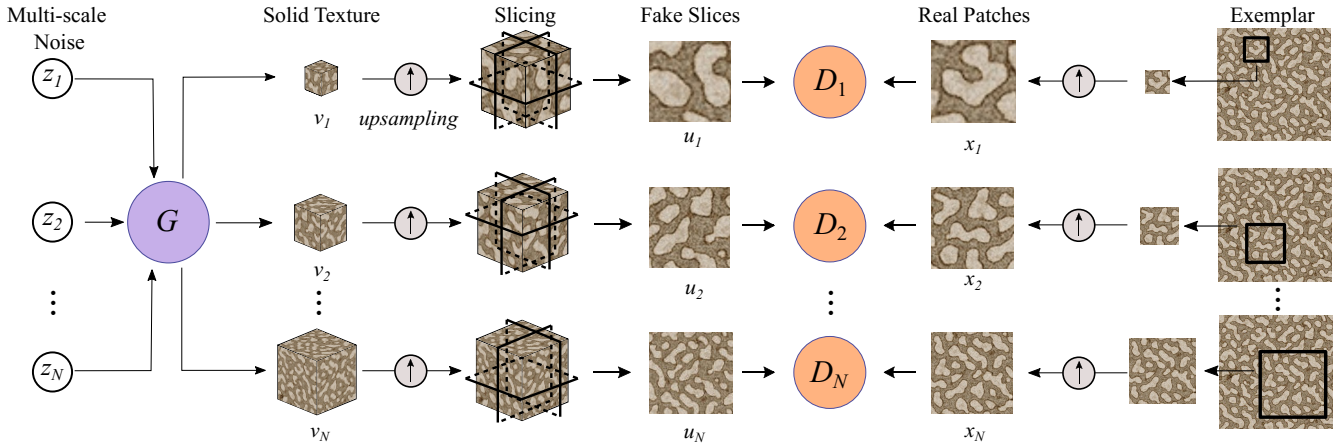


Figure 1: The framework of STS-GAN. It is a hierarchical framework containing N learning scales. The 3D solid texture generator G synthesizes solid textures $\{v_1, \dots, v_N\}$ by processing multi-scale noises $\{z_1, \dots, z_N\}$. At each learning scale n , synthesized solid textures v_n and patches x_n which randomly cropped from a given exemplar are upsampled to a same resolution. The fake slices u_n in the synthetic solid v_n are selected at random from the orthogonal directions. Finally, the 2D slice texture discriminators D_n distinguishes between the slices u_n and the real patches x_n .

[Henzler *et al.*, 2020] also provided another point operation-based neural network solution, which is a generative model of natural textures. The model feeds multiple transformed random 2D or 3D fields into a multi-layer perceptron that can be sampled over infinite domains.

Nevertheless, the diversity of textures makes it hard to use a limited number of VGG features to fit an infinite number of appearances. Thus, these methods may not accurately capture and extend arbitrary exemplars’ texture properties.

As perhaps the pioneer of the GAN-based STS method, GramGAN combined ideas from style transfer and generative adversarial nets to generate realistic 3D textures. Following the idea of point operation strategy, it takes spatial position information as the input to the generative model.

2.4 Challenge

Statistical feature-matching methods rely on features, introducing strong prior and limiting their diversity of applicable textures. Although MRF-based methods, taking advantage of their local Markov property, potentially can generate diversified 3D textures, their inferior learning capability makes it difficult to accurately estimate the conditional probabilities for the 3D neighborhood from a 2D exemplar.

In terms of neural net-based methods, they provide impressive efficacy due to their powerful expressive power. Nevertheless, the CNN-based method and work of Henzler *et al.* cannot always be applicable to diverse textures, as they also try to match features such as VGG statistics. Although GramGAN exhibits its adaptability to diverse textures, the adopted point operation strategy hard to capture local Markov property and multiscale, as it simply provides spatial information as network inputs. Thus, GramGAN fails to generate structured textures or complicated stochastic structures [Portenier *et al.*, 2020].

Therefore, an STS-method, which can synthesize high-fidelity solid textures from arbitrary exemplars, satisfying lo-

cal Markov property, multiscale, and diversity, is highly desired.

3 Methodology

3.1 Cross Dimensional Appearance Association

In the beginning, we need to answer the most fundamental question of solid texture synthesizer: how to associate the appearance of a 3D solid with a given 2D exemplar?

The solid textural appearance can be described by a joint distribution in the volumetric domain \mathcal{D} . This joint distribution can be further decomposed of distributions along distinct directions. Each of these distributions describes the specific appearance in that direction.

Given that we are interested in the texture along a certain direction, the appearance of cross-sections belonging to this direction is drawn from the same distribution, describing the common textural properties. As a result, we can learn a joint distribution, in which subdistribution along a corresponding direction reflects the appearance of the given 2D exemplar. If we can collect representative exemplar from different directions, we can thus map the textural appearance in the 2D exemplars into 3D domain \mathcal{D} by learning this joint distribution. Particularly, for an isotropic solid texture, subdistributions of all directions should be the same. In addition, it should be noted that the textural appearance usually exhibits different properties at different scales, implying the difference of distributions between scales.

In order to learn the joint distribution, we need to learn the subdistribution along different directions. Thus, a slicing strategy is adopted to associate 3D solid with the given 2D texture, playing the role of the cross-dimension junction. In this strategy, each candidate synthesized solid texture is randomly sliced along different directions to obtain "fake" cross-sections, which can be used to compare their appearance with a given exemplar directly in the desired directions. Intuitively, if a randomly sliced cross-section shares the same appearance

with the 2D exemplar, the synthesized solid texture and the corresponding "real" solid texture of the 2D exemplar draw from the same joint distribution.

Normally, for anisotropic solid textures, the slicing strategy is operated along given directions. However, if the target solid texture is isotropic, it is sliced orthogonally, as the spatial autocorrelation in 3D space may result in the redundancy of information between non-orthogonal directions.

3.2 STS-GAN Framework

To improve the *diversity* of applicable textures, this work designs a GAN-based framework for synthesizing arbitrary 3D textural appearances by learning texture distribution in the given 2D exemplar. The framework of STS-GAN is described in Figure 1, consisting of 3D *solid texture generator* (STG), 2D *slice texture discriminators* (STDs), and the slicing strategy as junction.

We first define a synthesizing resolutions set $S = \{S_1, S_2, \dots, S_N\}$, where $|S| = N$, with the intention of learning textural information at various scales. Here, N represents the number of learning scales, n represents the n^{th} scale ($n \in [1, N]$), and the resolution at scale n is S_n . The concept of scale is generally related to the hierarchical levels of detail. Since an image may exhibit different appearances at different scales, we use scaling operation to unify the resolution to observe a texture at different scales.

As a universal distribution learner, GAN is adopted to learn the distribution of solid texture at 3D domain \mathcal{D} . In the framework of STS-GAN, the STG plays the role of 3D textures synthesizer, while STDs play the role of critic for discriminating the fidelity of 2D cross-sections in 3D textures. The objective of STG G is to synthesize a "fake" solid whose slice u_n is similar to the real patch x_n , which is randomly cropped from the given exemplar at the corresponding scale n . It processes a group of input noises z_n to synthesize a solid texture v_n at scale n ,

$$v_n = G(z_n), \tag{1}$$

Meanwhile, as STG's opponents, the STDs consist of a collection of multi-scale discriminators $\{D_1, \dots, D_N\}$. D_n learns to differentiate randomly sliced cross-section u_n of solid texture v_n , from the real patch x_n at scale n . To observe the appearance of texture at different scales, we upsample those patches with different resolutions to the same resolution. Moreover, the synthesized 3D solids are upsampled to the same resolution as the corresponding patches to ensure learning texture distribution at the same scale.

In general, the STG generates realistic 3D textures whose sections appear indistinguishable from the given exemplar. By contrast, each STD attempts to distinguish cross-sections of generated 3D texture (fake sample) from the 2D exemplar (the real one) at the corresponding scale. For STG and STDs, we will describe them in the later sections, respectively.

3.3 Adversarial Learning

Let us focus on how to train this framework now. Like other GANs, this framework takes adversarial learning, promoting the STG synthesizing more realistic solid textures to fool STDs. When the adversarial learning is achieved, the STG

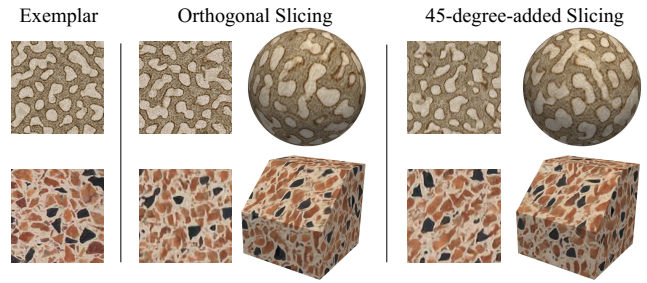


Figure 2: The model's performance using different slicing strategies in training. The middle section presents the outcome of the orthogonal slicing strategy, and the results with the 45 degree-added slicing strategy are shown on the right. In particular, we offer the carved solid and the 2D slice at 45 degree selected randomly from the volume respectively.

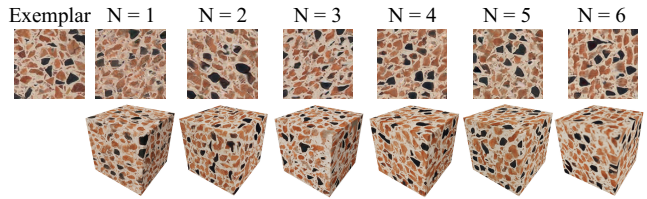


Figure 3: The model's performance with different number of learning scales. The generated solids of each model are shown on bottom, with randomly picked slices from them on the top.

can synthesize such realistic solid textures that the STDs cannot distinguish cross-sections of solid texture from the given exemplar.

It is worth noting that all discriminators at different scales are trained once in a single iteration, whereas the generator is trained only once at a randomly chosen scale, ensuring the learning is balanced.

This framework adopts the Wasserstein GAN with gradient penalty [Gulrajani *et al.*, 2017] loss during optimization to improve the stability of training. When the generator is optimized, the loss is minimized using equation:

$$\mathcal{L}_G = -\mathbb{E}[D(u_n)], \tag{2}$$

where u_n is a 2D slice taken at random from the 3D solid generated by STG. Meanwhile, each discriminator is optimized by minimizing its loss function, denoted by equation:

$$\mathcal{L}_{D_n} = \mathbb{E}[D(u_n)] - \mathbb{E}[D(x_n)] + \lambda \mathbb{E}[(\|\nabla_{r_n} D(r_n)\|_2 - 1)^2], \tag{3}$$

where x_n is a random cropped patch from the exemplar, and r_n is a uniformly sampled data point between u_n and x_n .

3.4 3D Solid Texture Generator

We adopt a fully convolutional network structure to carry out a solid texture generator, utilizing spatial locality by enforcing a local connectivity pattern between neurons of adjacent layers. The locality of pixel dependencies in fully convolutional operation can help STG to enforce the *local Markov property* for generated solid textures.

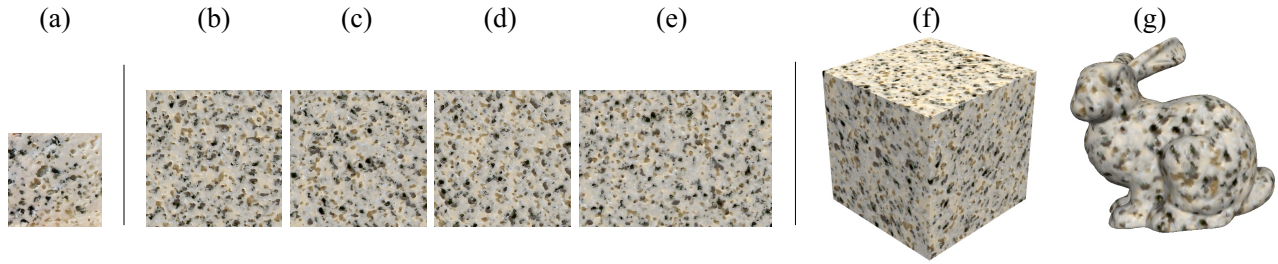


Figure 4: The performance of STS-GAN on isotropic exemplars. (a) texture exemplar, (b) - (d) the slices of the generated solid across the three orthogonal directions, (e) the 45-degree slice, (f) the synthetic solid texture, and (g) texture mapping. Source: the 3D mesh models are from Stanford 3D Scanning Repository. The resolution of the exemplar is 256, and the resolution of the generated slices and solid in the experiment is 1.5 times that of the original exemplar (i.e., 384).

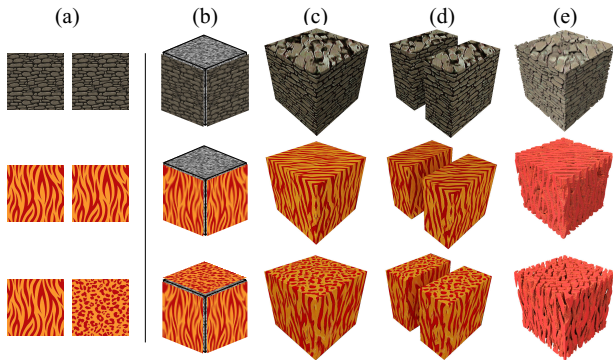


Figure 5: Results of the STS-GAN on different anisotropic exemplars. (a) exemplars, (b) learning configurations, (c) synthetic solids, (d) the cut solids, and (e) the eroded solids.

In the STG, the trick of *multi-scale inputs* [Gutierrez *et al.*, 2020; Ulyanov *et al.*, 2016] is adopted, enabling the generator to learn textural details at different scales and capture the *multiscality* of the texture. In addition, the *multi-scale inputs* influences the solid textural appearance at each scale to increase the diversity of texture and improving the stability of the adversarial learning. Thus, at scale n , the input noise group z_n for G contains K 3D noises $\{z_{n,1}, \dots, z_{n,K}\}$ with different size.

$$v_n = G(\{z_{n,1}, \dots, z_{n,K}\}). \tag{4}$$

3.5 2D Slice Texture Discriminators

The slice texture discriminator plays a critical role in guiding STG to produce realistic solid textures whose cross-section is largely indistinguishable from the given exemplar in the slicing direction. However, the scale of salient features could differ from each other in different directions. Furthermore, a single cross-section image may exhibit different spatial coherence at various scales. Although STG could generate textures with multiple resolutions, the *multiscality* of texture requires an STS models to recognize features at multiple scales. Nevertheless, considering the complexity of textures, it is hard to train a discriminator to assemble multi-scale knowledge into a single model for differentiating multi-scale cross-sections.

Inspired by SinGAN [Shaham *et al.*, 2019], a set of slice texture discriminators are used to differentiate fake slices

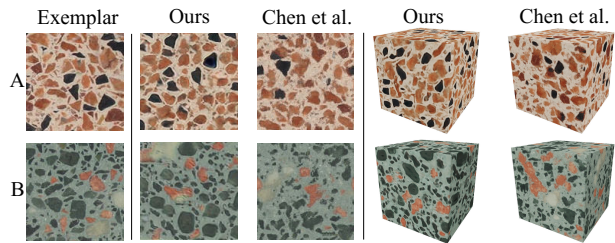


Figure 6: Comparison with the traditional non-neural method [Chen and Wang, 2010]. The synthetic slices are shown in the middle. The generated solids are shown on the right.

from given 2D exemplar at multi-scales. In particular, unlike SinGAN’s training mode from coarse-scale to fine-scale, STS-GAN trains the model on multiple scales simultaneously. Thus, the possible error accumulation from scale-by-scale training can be avoided. The STDs consist of N discriminators sharing the same fully convolutional structure but operating at different image scales. At the n scale, D_n takes the random slicing strategy to slice 2D cross-sections from corresponding synthesized 3D texture as “fake” texture. In order to improve the diversity of real samples, we randomly crop textural patches at multiple predefined sizes from the exemplar and then resize these patches to the same resolution, providing the STDs with multi-scale “real” textures.

4 Experiment

This section examines the efficacy of our hierarchical architecture’s multi-scale and orthogonal slicing components. Isotropic and anisotropic experiments demonstrate the performance of our method. Furthermore, we compare our method to the state-of-the-art STS methods, particularly those based on neural networks.

This framework runs on PyTorch and is optimized using the Adam optimizer [Kingma and Ba, 2015]. The learning rates of STG and STD are usually set to 0.0005 and 0.0003, respectively. The implementation uses Nvidia GeForce TITAN RTX GPU for acceleration.

4.1 Ablation Experiments

Slicing Direction In the training process, to reduce computational overhead, we slice cross-sections in three orthogonal

ExemplarA		ExemplarB		ExemplarC			ExemplarD			ExemplarE			ExemplarF		
Ours	Chen	Ours	Chen	Ours	GramGAN	CNN	Ours	GramGAN	CNN	Ours	GramGAN	CNN	Ours	GramGAN	CNN
0.559	0.564	0.337	0.368	1.87	1.91	1.96	2.11	2.29	2.52	2.76	2.65	2.92	2.14	2.06	2.10

Table 1: Comparison of FDMSE ($\times 10^{-4}$) on different exemplars. CNN means [Gutierrez *et al.*, 2020]. We calculate the FDMSE between the sections of the generated solids and the given exemplars. 2D exemplars are shown in Figure 6 and Figure 7.

	Method	Rank from Study
Non-neural Networks	Ours	1.14 ± 0.35
	Chen and Wang	1.86 ± 0.35
Neural Networks	Ours	1.47 ± 0.74
	GramGAN	2.00 ± 0.68
	Gutierrez <i>et al.</i>	2.53 ± 0.66

Table 2: User ranking for the similarity of the textures generated by the different methods to the exemplar. We report $\mu \pm \sigma$ (mean and standard deviation) for user study ranking (lower is better).

directions in STS-GAN. For comparison, we also slice additional 45-degree-angle cross-sections into the fake patch set to evaluate the sufficiency of the orthogonal slicing strategy.

Figure 2 shows the 3D textures obtained by the orthogonal slicing strategy and the 45-degree-added slicing strategy. There is no significant difference between these two slicing strategies, demonstrating that slicing along the orthogonal plane is sufficient to create high-fidelity solid textures.

4.2 Parameter Analysis

Multi-scale Learning As previously stated, STS-GAN learns textural information on multiple scales. To confirm the efficacy of the multi-scale learning strategy, we explore its effects by varying the number of learning scales, and analyze model’s performance with various parameters (i.e., N).

Figure 3 shows that as the number of learning scales increases, the model provides clearer solids, and the overall structure and the local details gradually resemble the given exemplar. Experiment proves that a model with multiple learning scales can capture the multiscality of textures and generate realistic solid textures at a sufficient learning scales (usually, $N = 5$).

4.3 Dependency of Direction

Isotropic Exemplar In experiments, the STS-GAN is evaluated with isotropic exemplars. Figure 4 depicts the generated solid texture and several slices of solid. It is apparent that the created 3D solid closely resembles the given 2D exemplar and the textural visual characteristics of the slices from the 3D solid at different angles are similar to the exemplar.

Furthermore, the generated solid is used in surface texture mapping. Based on the spatial coordinate information, the synthesized solid distributes color to the surface pixels of the 3D mesh model. The outcome exhibits similar visual qualities to the exemplar, as shown in Figure 4(g).

These experiments show that STS-GAN can learn texture distribution of the given exemplar and synthesize high-fidelity solid textures.

Anisotropic Exemplar Since the STS-GAN learns the texture distribution from a 2D exemplar, it can also generate solid textures with anisotropic properties by capturing different exemplars. In experiments, the STS-GAN learns

anisotropic exemplars in multiple orthogonal orientations based on different configurations.

As shown in Figure 5, the generated solid textures maintain the same texture properties in each orthogonal direction as the corresponding exemplar, and we present the synthetic solids differently. Experiments suggest that STS-GAN is able to learn anisotropic texture and extend it to 3D solids.

4.4 Performance Comparison

In this section, the performance of the STS-GAN is compared against three state-of-the-art methods, including a non-neural method [Chen and Wang, 2010] and two neural networks-based methods [Gutierrez *et al.*, 2020; Portenier *et al.*, 2020].

Qualitative Evaluation Figure 6 compares our method with Chen *et al.*’s method. Although Chen *et al.*’s approach produces solid textures similar to exemplars, they still have failed attempts with variances between the generated solid and the exemplar. It can be observed that the solid texture produced by STS-GAN is more similar to the exemplar. Furthermore, the STS-GAN provides clearer borders and textures consistent in color, shape, and distribution with the exemplars. Due to the powerful learning ability of neural networks, STS-GAN has a more remarkable ability to learn textural properties than non-neural methods. Thus, our method can capture complex textures and generate realistic 3D textures.

Figure 7 exhibits the comparison between STS-GAN and two neural networks-based methods. It can be observed that the solid textures and slices generated by STS-GAN are more visually similar to the exemplar. In most cases, Gutierrez *et al.*’ method focuses solely on the generalized textural styles. For textures with diverse pattern styles, their approach fails to adapt to the *diversity* of textures. In contrast, STS-GAN can learn arbitrary diverse texture patterns and extend them to solid textures benefiting from the ability of GAN to approximate arbitrary distributions. As shown in Figure 7, solid textures generated by GramGAN are blurred, and the color and shape of textures differ from the exemplar. These situations are due to the difficulty of GramGAN in learning the *local Markov property* and *multiscality* of textures. In STS-GAN, the CNNs-based network structure ensures that STS-GAN can capture the *local Markov property* of textures. At the same time, the multi-scale strategy helps STS-GAN to learn textural *multiscality*. Thus, our method produces high-fidelity solid textures compared to the other two methods.

Frequency Domain Mean Square Error In this section, we quantitatively compare STS-GAN with other competitors. In the experiments, we use the frequency domain mean square error (FDMSE) as metric to evaluate synthesis effect. The frequency domain of an image contains information of the texture [Tadi Bani and Fekri-Ershad, 2019; Di Lillo *et al.*, 2007]. Specifically, during the computational

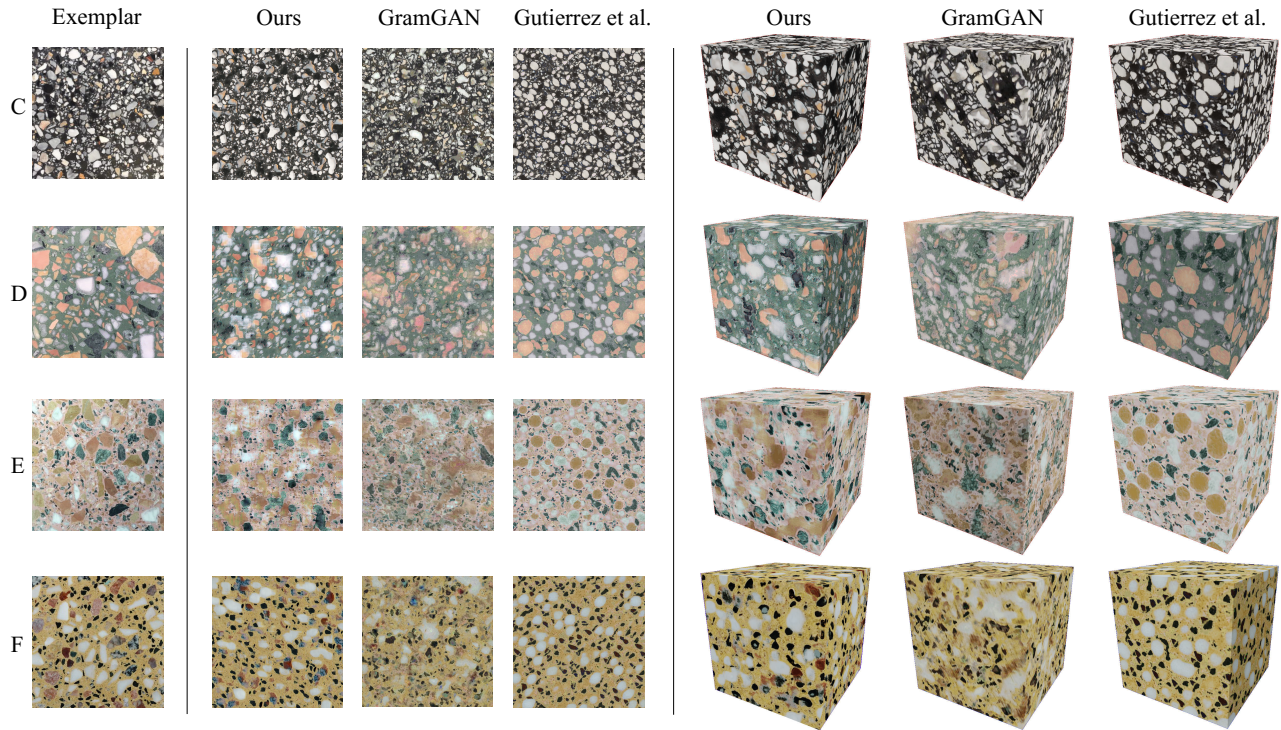


Figure 7: The comparison of STS-GAN’s outcomes with those of neural networks-based methods. The generated solids are shown on the right. The slices are shown in the middle.

process, we first calculate the magnitude based on the image frequency domain, and then proceed with the calculation of mean squared error. Table 1 shows the performance on the this metric. It can be observed that the neural network methods significantly outperform the traditional ones on FDMSE. Particularly, STS-GAN and GramGAN are superior to CNN in multiple exemplars, as they adopt GAN, enabling capturing diverse textural information, generating visually similar solid textures.

Compared to image numerical metrics, the user study provides a closer approximation to the visual quality. Thus, to evaluate different methods more intuitively and effectively, we conducted a user study experiment as described below.

User Study¹ We also conducted a single-blind formal user study to compare the visual effect between approaches. A group of 26 volunteers was given texture exemplars and their corresponding slices from synthesized solid textures by different competitors. They were asked to rank the slices based on their similarity to the corresponding reference exemplar. Apart from their corresponding exemplars, the questionnaire contains 20 groups of slices from non-neural network synthesizers and 20 groups from neural network ones. Table 2 exhibits the average ranking for each approach. It can be observed that the average ranking of our method is significantly higher than the other methods, and users are more accepting of our results. Experimental results prove our method can generate more realistic solid textures.

¹All volunteers have signed an informed consent form, guaranteeing to make independent and objective choices.

5 Conclusion

This research proposes a novel approach to synthesize solid texture, named as STS-GAN, which learns arbitrary 3D solid texture from few 2D exemplars. It can successfully capture textural *local Markov property*, *multiscality*, and *diversity*, synthesizing high-fidelity solid textures. In experiments, STS-GAN generates more realistic solid textures than the other state-of-the-art methods.

There are, however, still some limitations to this method. The high computational cost of STS-GAN learning process is a significant burden. In the future, the simplification method should be further studied to accelerate learning. Furthermore, the generating process must be hastened to meet the efficiency requirements in real-world applications. Finally, there is a future research direction in exploring the relationship between texture and the human visual perception, as well as establishing precise numerical metrics for describing textures.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant No. 61872419, No. 62072213, No. 61873324, No. 61903156. Shandong Provincial Natural Science Foundation No. ZR2022JQ30, No. ZR2022ZD01, No. ZR2020KF006. Taishan Scholars Program of Shandong Province, China, under Grant No. tsqn201812077. “New 20 Rules for University” Program of Jinan City under Grant No. 2021GXRC077. Project of Talent Introduction and Multiplication of Jinan City.

References

- [Ashton *et al.*, 2020] Tristan N Ashton, Donna Post Guillen, and William H Harris. An algorithm to generate synthetic 3d microstructures from 2d exemplars. *JOM*, 72(1):65–74, 2020.
- [Bergmann *et al.*, 2017] Urs Bergmann, Nikolay Jetchev, and Roland Vollgraf. Learning texture manifolds with the periodic spatial gan. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 469–477, 2017.
- [Chen and Wang, 2010] Jiating Chen and Bin Wang. High quality solid texture synthesis using position and index histogram matching. *The Visual Computer*, 26(4):253–262, 2010.
- [Di Lillo *et al.*, 2007] Antonella Di Lillo, Giovanni Motta, and James A Storer. Texture classification based on discriminative features extracted in the frequency domain. In *2007 IEEE International Conference on Image Processing*, volume 2, pages II – 53–II – 56, 2007.
- [Efros and Leung, 1999] Alexei A Efros and Thomas K Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the seventh IEEE international conference on computer vision*, volume 2, pages 1033–1038, 1999.
- [Fayolle *et al.*, 2021] PA Fayolle, Leigh McLoughlin, M Sanchez, G Pasko, and A Pasko. Modeling and visualization of multi-material volumes. *Scientific Visualization*, 13(2):117–148, 2021.
- [Georgiadis *et al.*, 2013] Georgios Georgiadis, Alessandro Chiuso, and Stefano Soatto. Texture compression. In *2013 Data Compression Conference*, pages 221–230, 2013.
- [Ghazanfarpour and Dischler, 1995] Djamchid Ghazanfarpour and JeanMichel Dischler. Spectral analysis for automatic 3-d texture generation. *Computers & Graphics*, 19(3):413–422, 1995.
- [Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27, pages 2672–2680, 2014.
- [Gulrajani *et al.*, 2017] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, volume 30, 2017.
- [Gutierrez *et al.*, 2020] Jorge Gutierrez, Julien Rabin, Bruno Galerne, and Thomas Hurtut. On demand solid texture synthesis using deep 3d networks. *Computer Graphics Forum*, 39(1):511–530, 2020.
- [Heeger and Bergen, 1995] David J Heeger and James R Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 229–238, 1995.
- [Henzler *et al.*, 2020] Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. Learning a neural 3d texture space from 2d exemplars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8356–8364, 2020.
- [Iwasaki *et al.*, 2017] Kei Iwasaki, Yoshinori Dobashi, and Makoto Okabe. Example-based synthesis of three-dimensional clouds from photographs. In *Proceedings of the Computer Graphics International Conference*, pages 1–6, 2017.
- [Jagnow *et al.*, 2004] Robert Jagnow, Julie Dorsey, and Holly Rushmeier. Stereological techniques for solid textures. *ACM Transactions on Graphics (TOG)*, 23(3):329–335, 2004.
- [Kingma and Ba, 2015] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, San Diego, CA, USA*, 2015.
- [Kopf *et al.*, 2007] Johannes Kopf, Chi-Wing Fu, Daniel Cohen-Or, Oliver Deussen, Dani Lischinski, and Tien-Tsin Wong. Solid texture synthesis from 2d exemplars. In *ACM SIGGRAPH 2007 Papers*, page 2–es, 2007.
- [Laursen *et al.*, 2011] Lasse Farnung Laursen, Bjarne Kjær Ersbøll, and Jakob Andreas Bærentzen. Anisotropic 3d texture synthesis with application to volume rendering. In *WSCG’ 2011 Communication Papers Proceedings*, pages 49–57, 2011.
- [Mariethoz and Lefebvre, 2014] Gregoire Mariethoz and Sylvain Lefebvre. Bridges between multiple-point geostatistics and texture synthesis: Review and guidelines for future research. *Computers & Geosciences*, 66:66–80, 2014.
- [Mark *et al.*, 2015] Benjamin Mark, Tudor Berechet, Tobias Mahlmann, and Julian Togelius. Procedural generation of 3d caves for games on the gpu. In *Proceedings of the 10th International Conference on the Foundations of Digital Games, Pacific Grove, CA, USA*, 2015.
- [Peachey, 1985] Darwyn R Peachey. Solid texturing of complex surfaces. In *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, pages 279–286, 1985.
- [Perlin, 1985] Ken Perlin. An image synthesizer. *ACM Siggraph Computer Graphics*, 19(3):287–296, 1985.
- [Portenier *et al.*, 2020] Tiziano Portenier, Siavash Arjomand Bigdeli, and Orcun Goksel. Gramgan: Deep 3d texture synthesis from 2d exemplars. In *Advances in Neural Information Processing Systems*, volume 33, pages 6994–7004, 2020.
- [Shaham *et al.*, 2019] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4570–4580, 2019.
- [Simonyan and Zisserman, 2015] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for

- large-scale image recognition. In *3rd International Conference on Learning Representations, San Diego, CA, USA, 2015*.
- [Tadi Bani and Fekri-Ershad, 2019] Neda Tadi Bani and Shervan Fekri-Ershad. Content-based image retrieval based on combination of texture and colour information extracted in spatial and frequency domains. *The electronic library*, 37(4):650–666, 2019.
- [Tong *et al.*, 2002] Xin Tong, Jingdan Zhang, Ligang Liu, Xi Wang, Baining Guo, and Heung-Yeung Shum. Synthesis of bidirectional texture functions on arbitrary surfaces. *ACM Transactions on Graphics*, 21(3):665–672, 2002.
- [Turner and Kalidindi, 2016] David M. Turner and Surya R. Kalidindi. Statistical construction of 3-d microstructures from 2-d exemplars collected on oblique sections. *Acta Materialia*, 102:136–148, 2016.
- [Ulyanov *et al.*, 2016] D Ulyanov, V Lebedev, A Vedaldi, and V Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, volume 48, pages 1349–1357, 2016.
- [Wang *et al.*, 2018] Na Wang, Guodong Chen, and Huiyan Zeng. An optimization algorithm of space anisotropic hepatic artery solid texture synthesis. *Current Medical Imaging*, 14(4):609–616, 2018.
- [Wei and Levoy, 2000] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques, SIGGRAPH '00*, page 479–488, 2000.
- [Wei, 2002] Li-Yi Wei. *Texture synthesis by fixed neighborhood searching*. PhD thesis, Stanford University, 2002.