

Generating Non-Stationary Textures using Self-Rectification

Yang Zhou¹ Rongjun Xiao¹ Dani Lischinski² Daniel Cohen-Or^{1,3} Hui Huang^{1*}
¹Shenzhen University ²The Hebrew University of Jerusalem ³Tel Aviv University

Abstract

This paper addresses the challenge of example-based non-stationary texture synthesis. We introduce a novel two-step approach wherein users first modify a reference texture using standard image editing tools, yielding an initial rough target for the synthesis. Subsequently, our proposed method, termed “self-rectification”, automatically refines this target into a coherent, seamless texture, while faithfully preserving the distinct visual characteristics of the reference exemplar. Our method leverages a pre-trained diffusion network, and uses self-attention mechanisms, to gradually align the synthesized texture with the reference, ensuring the retention of the structures in the provided target. Through experimental validation, our approach exhibits exceptional proficiency in handling non-stationary textures, demonstrating significant advancements in texture synthesis when compared to existing state-of-the-art techniques. Code is available at <https://github.com/xiaorongjun000/Self-Rectification>

1. Introduction

Example-based texture synthesis aims to generate a texture that faithfully captures all the visual characteristics of a provided reference texture exemplar. The key challenge is to generate a texture that visually mimics the reference, while avoiding exact replication and without producing conspicuous, unnatural artifacts.

Over the past decades, numerous methods have emerged for synthesizing textures from examples, and many have demonstrated impressive results, particularly for homogeneous textures that can be accurately captured by stationary models [45]. Nonetheless, a significant challenge remains when dealing with real-world textures that are inherently inhomogeneous, and non-stationary.

Non-stationary textures exhibit distinctive attributes, such as sprawling irregular large-scale structures or variations in attributes such as color, local orientation, and local scale. Fig. 1 (left) shows two examples. Mimicking such

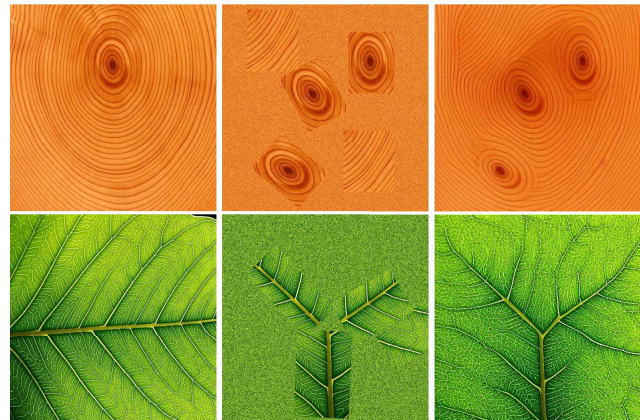


Figure 1. Our method takes as input a reference texture (left), and a crude target texture provided by the user (middle column), which may lack coherence and completeness. Self-rectification is used to transform the target into a visually coherent texture (right) that complies with the structure of the crude target, while exhibiting the visual characteristics of the reference texture.

complex structures and variations through example-based synthesis is a long-standing challenging task [31].

The emergence of neural networks has provided powerful means to deal with non-stationary textures. Zhou *et al.* [51] introduced a method that involves overfitting a GAN, where the encoder extracts structural guidance for the decoder. This approach provides a viable means of spatially extending non-stationary textures while preserving the visual characteristics of the reference. However, this technique requires an extremely long optimization process for a single texture exemplar, and moreover, it fails to provide any controllability or editability.

In this paper, we introduce a two-step lazy-editing technique where the user first edits the given reference texture using conventional image editing tools, to obtain an extremely rough initial result, which may be incomplete and incoherent, as demonstrated in Fig. 1 (middle). Next, our technique automatically rectifies the edited texture into a regularized, coherent and seamless texture that follows the rough target, while retaining the local characteristics of the reference; see, *e.g.*, Fig. 1 (right). We term this regulariza-

*Corresponding author.

tion process *self-rectification*, since the texture is rectified based on the input texture itself.

To self-rectify the crude target, we use a pre-trained diffusion network, and synthesize the final rectified result while utilizing the intermediate byproducts obtained by inverting both the initial target and the reference exemplar. Inspired by editing approaches that leverage self-attention [1, 6, 20], we extract certain self-attention features of the diffusion model during the inversion of the rough target, as well as the reference. These features are then injected into various steps in the inversion and denoising of the generated texture. Such feature injection forms “cross-attention” between the reference and target features [1, 6], and thereby, in the course of the diffusion process, the target texture is progressively refined to be increasingly *locally* similar to the reference, while retaining the global structure prescribed by the provided target. This self-rectification occurs in two passes, first addressing the larger scale structure and subsequently focusing on finer local details.

To enhance the synthesis quality, we augment the reference texture with several transformed copies of it, which increases the diversity of the reference patterns, thereby improving the compatibility between the reference and the synthesized result. The augmented source features are injected into the corresponding target layers as well. Furthermore, we show that our method can also be applied to lazy editing of natural images, using the same means of synthesizing highly non-stationary textures. Experiments show that our method can deal with a large scope of non-stationary textures, with unprecedented flexibility and quality, compared to the state-of-the-art.

2. Related Work

Non-stationary texture synthesis. Early methods in example-based texture synthesis mainly focused on synthesizing textures with stationary characteristics [8, 9, 19, 21, 22, 43, 44]. To cope with non-stationary textures, researchers typically involve additional guidance for the control of distinctive attributes, *e.g.*, using label maps to guide the layout synthesis of composite textures [15, 24, 31], representing the spatial variations of weathering textures with age/progression maps [3, 42], and describing the local orientations of directional textures with vector fields [26, 50]. Although the aforementioned methods have shown success, the guidance required from the user is tedious to provide, yet still limited.

In the deep learning era, neural methods for texture generation have rapidly emerged, either through new texture losses for texture optimization [12, 13, 49] or via training generative networks [4, 5, 10, 18, 23, 29, 32, 35–37, 40, 51]. Among these methods, Sendik and Cohen-Or [35] attempted to preserve the non-local structures of non-stationary textures by regularizing the feature correla-

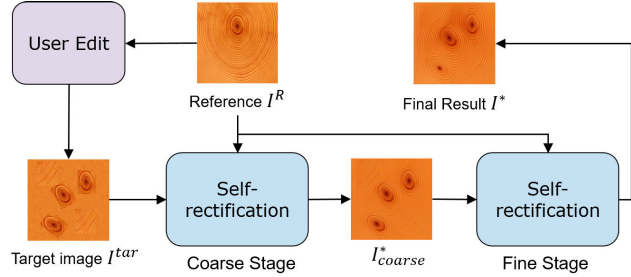


Figure 2. Framework overview. Given a reference texture I^R , we allow the user to quickly build a target image I^{tar} in a lazy-editing manner. A coarse-to-fine synthesis is performed by running self-rectification twice. The coarse stage synthesizes a coarse yet complete overall structure, and the fine stage refines its output I^*_{coarse} with finer and more accurate details, producing the final result I^* .

tion between different locations. Zhou *et al.* [51] proposed overfitting a GAN to learn the expansion from a small texture block to a larger one containing it. Their approach involves the GAN’s encoder extracting the global structure of an input texture, duplicated by bottleneck residual blocks before decoding. Although these trained models can effectively extend non-stationary textures, their overfitting nature severely restricts their generalization and controllability.

Diffusion-based image synthesis. The emergence of large-scale generative diffusion models, such as Stable Diffusion (SD) [30] and DALLE-2 [28], has revolutionized image synthesis due to their unprecedented generation power. To make the pre-trained diffusion models synthesize on image conditions, many solutions are proposed, including optimizing prompt tokens [2, 11], fine-tuning the entire model [33], or training an additional adapter [17, 34, 46–48].

Recently, researchers have investigated the role of the intermediate attention maps and features in the diffuser [7, 14, 27, 39], finding them crucial for layout/structure synthesis. The method we present in this paper, builds upon the mechanism of injecting keys and values from attention layers of one diffusion process into another, as a means for transferring visual features between images [1, 6, 20]. We use such injection to “copy” the local patterns from a source texture to a target one. In addition to the injection we also incorporate coarse-to-fine and data augmentation schemes, forming the self-rectification framework for generating non-stationary textures.

3. Method

Our method employs a two-step process for synthesizing a new texture I^* resembling a provided non-stationary reference texture I^R , while guided by a user-provided rough target I^{tar} , as depicted in Fig. 2. Initially, the user rapidly creates I^{tar} , by assembling patches from the source refer-

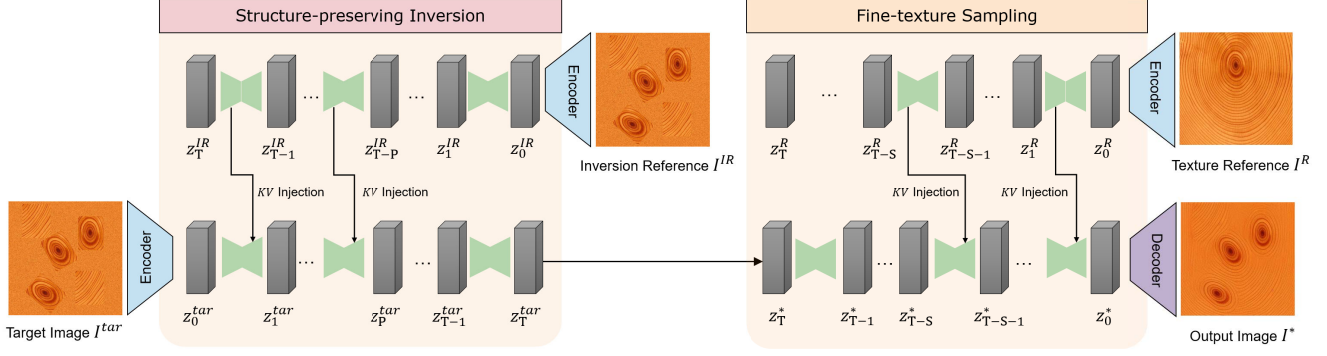


Figure 3. Our self-rectification synthesizes an output texture I^* via structure-preserving inversion from a rough target image I^{tar} and fine texture sampling using the reference I^R . Both processes require the injection of self-attention features (KV) from the DDIM inversion of a corresponding reference. More specifically, for structure-preserving inversion, the reference is the target image itself, denoted as I^R . For fine texture sampling, the input exemplar I^R is used to inject features that help to synthesize a plausible output with fine texture details.

ence texture. Such a lazy-edit forms a basic layout for the output image, I^* . However, this initial sketch may be incoherent or incomplete, hence necessitating rectification. Subsequently, the texture undergoes self-rectification to align with both the user’s coarse layout and the reference image’s detailed texture characteristics. This self-rectification process is depicted in Fig. 2 and is executed in two stages: coarse rectification, followed by a finer one.

Considering that the rough target I^{tar} comprises patches derived from the reference texture I^R , we implement a process of self-rectification, detailed below. In a broad sense, we utilize a pre-trained latent diffusion model to invert both the target image and the reference image into an initial latent noise. This is followed by employing feature injection during the sampling process of I^* . The features from I^{tar} guide the structural synthesis and the features from I^R guide the fine texture details.

Below, we first briefly review diffusion models and their self-attention and the cross-KV-injection mechanisms, and then proceed to present our self-rectification technique.

3.1. DDIM Sampling and Inversion

Denosing diffusion models [16, 38] involve two processes: a noising process that gradually transforms an input image into Gaussian noise, and a denoising/sampling process that generates images from Gaussian samples. Using *DDIM sampling* [38], starting from a noise sample z_T , one can generate a clean sample z_0 via T deterministic steps:

$$z_{t-1} = \sqrt{\bar{\alpha}_{t-1}} f_{\theta}(z_t, t) + \sqrt{1 - \bar{\alpha}_{t-1}} \epsilon_{\theta}(z_t, t), \quad (1)$$

where ϵ_{θ} is a noise prediction network conditioned on the current noisy sample z_t and timestamp t . $\bar{\alpha}_t$ is the noise scaling factor defined in [38], and $f_{\theta}(z_t, t)$ is

$$f_{\theta}(z_t, t) = \frac{z_t - \sqrt{1 - \bar{\alpha}_t} \epsilon_{\theta}(z_t, t)}{\sqrt{\bar{\alpha}_t}}. \quad (2)$$

In the opposite direction, *DDIM inversion* [38] of a given image z_0 is the process of incrementally adding deterministic noise, until obtaining z_T :

$$z_{t+1} = \sqrt{\bar{\alpha}_{t+1}} f_{\theta}(z_t, t) + \sqrt{1 - \bar{\alpha}_{t+1}} \epsilon_{\theta}(z_t, t). \quad (3)$$

Such inversion leads to a nearly faithful reconstruction [38].

3.2. Stable Diffusion and Self-Attention.

We base our texture synthesis framework on Stable Diffusion (SD) [30], which is a pretrained latent diffusion model consisting of an encoder \mathcal{E} that maps an input image into the latent space, and a decoder \mathcal{D} that reconstructs a latent code back into image space. Both DDIM sampling and DDIM inversion are performed in the SD latent space.

The noise predictor ϵ_{θ} of SD is a large-scale U-Net that contains multiple self-attention modules [41]. Each self-attention layer transforms its input intermediate feature map (also called spatial features) into an attended representation by the following equation:

$$\text{Att}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V, \quad (4)$$

where Q , K , and V are the queries, keys, and values, respectively, obtained by learned linear projections of the same input spatial features, having dimension d . The self-attention mechanism uses the similarities between the queries and keys as attention scores to weigh the importance or relevance of the values. Relevant info is thus aggregated as the attended representation.

3.3. KV-Injection

The self-attention features contain rich information about both the large-scale structures and local fine details of an input image [1, 6, 39]. Tumanyan *et al.* [39] demonstrate that by injecting the spatial features and the queries

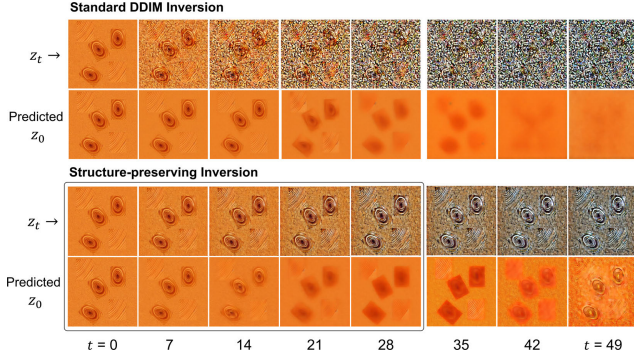


Figure 4. Visualization of the intermediate latent codes in the inversion. For the standard DDIM inversion (top), the U-Net predicts noise to diffuse the distinctive structures so as to transform the input into Gaussian noise. In contrast, our structure-preserving inversion (bottom) reserves the distinctive patterns from user edits along the inversion process. See texts in Sec. 3.4 for more details.

and keys (QK) from the self-attention layers of a source (guidance) image into the corresponding layer of the generated target image during the sampling process, one can preserve the layout of the source while modifying its appearance. Injecting only the KV features of a source reference, instead, transfers its appearance, including textures, to the generated target [1, 6].

In our case, the target image is a rough and incomplete guidance, typically containing only a few source patches rotated and placed freely by the user. A complete overall structure needs to be synthesized reasonably yet conforming to user’s constraints. Locally, however, the resulting texture should resemble the reference. To achieve these goals, we adapt the KV -injection into both the DDIM inversion and sampling, bringing a novel self-rectification operation.

3.4. Self-rectification

As shown in Fig. 3, our self-rectification consists of two parts: structure-preserving inversion and fine texture sampling. Both are performed in the SD latent space. The last latent code of inversion is used as the starting code of the sampling. The core modification to both processes is the KV -injection of self-attention features.

Structure-preserving inversion. The standard DDIM inversion (Eq. (3)) progressively transforms an input image (SD latent code) into pure Gaussian noise. At each time step, the noise to be added is predicted by the U-Net. As visualized in Fig. 4, for a given target image, the noise predicted by the U-Net for early time steps (e.g., $t \leq 20$) is mainly distributed to “diffuse” the prominent structures so that the distinctive patterns from user edits get scattered and random. As the inversion progresses, the magnitude of the noise added in each step becomes smaller and similar for

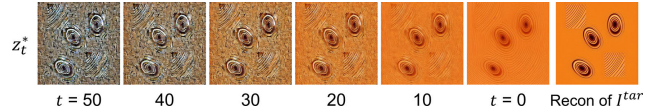


Figure 5. Visualization of the intermediate latent codes in the fine texture sampling. Here, for the first 20 steps (from $t = 50$ to 30), we perform the standard DDIM sampling to reconstruct the target layout. Next, we perform KV -injection in the remaining sampling steps ($t = 30$ to 0), to synthesize fine textures for the output image. The rightmost shows the result produced by simply performing standard DDIM sampling for all steps, i.e., $S = 50$. No additional structure is synthesized to complete the user edits.

all latent pixels, since the “diffusion” is getting close to done. As no text prompt is involved, the noise prediction in each inversion step is dominantly determined by the self-attention mechanism in the U-Net. Our key observation is that if we inject the KV features from a large time step $t_1 (\gg T/2)$ into an early time step $t_2 (\ll T/2)$, the noise predicted at t_2 will be smaller and more spatially uniform. The distinctive patterns of I^{tar} , reflecting the user’s edits, are thus better preserved. Hence, we refer to this process as *structure-preserving inversion*.

Specifically, given a rough target image I^{tar} , we invert it twice. The first inversion is a standard DDIM inversion. The produced self-attention features along the noising steps are regarded as the *inversion reference* (IR) for the second inversion. The self-attention during the second inversion, at time step t , is now given by:

$$\text{Att}(Q_t^{tar}, K_{T-t}^{IR}, V_{T-t}^{IR}) = \text{Softmax}\left(\frac{Q_t^{tar} (K_{T-t}^{IR})^T}{\sqrt{d}}\right) V_{T-t}^{IR}, \quad (5)$$

where T denotes the total number of steps (we use $T = 50$). Here KV features are injected in a reverse order in the second round. We would like to stress that we have also experimented with using an offset, i.e., replacing $T - t$ with $t + \text{offset}$ in the above equation, but found no advantage (see supplementary for more analysis and comparisons).

During the second inversion, we do not perform KV injections for all time steps. We set a parameter P that defines the time step beyond which we continue with standard inversion. Fig. 4 shows the effect of our structure-preserving inversion, where $P = 30$. Since the final latent code after the second inversion still contains rich information about the distinctive structures from I^{tar} , the subsequent sampling that starts from this latent code is guided to synthesize a global structure that complies with I^{tar} .

Fine texture sampling. We start the DDIM sampling from the final structure-preserving latent code. The sampling uses the U-Net to predict the noise to be removed, as defined in Eq. (1). However, simply performing standard

DDIM sampling for all denoising steps results in a nearly reconstructed target image since DDIM sampling is deterministic. Therefore, we set the first S steps (*i.e.*, from time step T to $T - S$) to reconstruct the target layout to a certain extent. For the remaining $T - S$ denoising steps, we synthesize the output image I^* by matching fine textures from the reference via KV -injection from the reference I^R .

To this end, we first DDIM-invert the reference texture I^R . At denoising time step t ($t > T - S$), the KV features extracted during the inversion are injected into the corresponding self-attention layers of the synthesized texture:

$$\text{Att}(Q_t^*, K_t^R, V_t^R) = \text{Softmax}\left(\frac{Q_t^* (K_t^R)^T}{\sqrt{d}}\right) V_t^R. \quad (6)$$

This forms a cross-image attention, where corresponding fine local patterns in the reference are transferred to the output image in a plausible manner. Fig. 5 visualizes the intermediate process of our texture sampling, where $S = 20$.

3.5. Implementation details and data augmentation

In the user editing phase, we fill the background of the target canvas with pixels randomly drawn from the source texture, such that the encoding of the target image would not deviate too far from the source in the SD latent space. As the self-rectification is performed twice, the output of coarse stage I_{coarse}^* , will be used as the input target image of fine stage to produce the final output I^* . In contrast, for the inversion reference I^R involved in structure-preserving inversion of the two self-rectifications, we use the same initial target image that contains the original user edits. See the supplementary for the full algorithm pseudo-code.

Considering the parameter settings: since the coarse stage aims for structure synthesis, relatively large P and S are required in self-rectification, and vice versa in the fine stage. More specifically, let P_1, P_2, S_1 , and S_2 , denote the parameters used in the two rounds of structure-preserving inversion and fine texture sampling. We typically set $P_1 = 20, P_2 = 5, S_1 = 20$, and $S_2 = 5$. Following [6], we choose the KV features from the 10th to 15th self-attention layers of the U-Net decoder part.

To further improve the synthesis quality, especially when dealing with textures containing a dominant directional structure, such as the leaf shown in Fig. 1, we can introduce a few transformed images (flips and rotations) to augment the reference texture. We concatenate the new attention features from the augmentation to the original reference feature. For example, when we have n augmented source textures, the K^R and V^R in Eq. (6) is now given by

$$\begin{cases} K^R &= \text{Concat}\left(K_{(0)}^R, K_{(1)}^R, \dots, K_{(n)}^R\right) \\ V^R &= \text{Concat}\left(V_{(0)}^R, V_{(1)}^R, \dots, V_{(n)}^R\right) \end{cases}, \quad (7)$$

where $K_{(i)}^R$, and $V_{(i)}^R$ are the self-attention features acquired from the DDIM inversion of the augmented references.

4. Experiments

We apply our method with Stable Diffusion with publicly available checkpoints v1.4. All experiments were conducted on a single Quadro P6000 24G GPU. The inference time synthesizing an image of 512×512 pixels takes about three minutes for our coarse-to-fine self-rectification.

4.1. Evaluations and Comparisons

We evaluate our method with non-stationary textures released by [51]. Each example is resized to 512×512 pixels as the reference, and we quickly built several different target images for each in PhotoShop with just a few lazy edits. Fig. 6 shows a gallery of results generated by our method. As can be seen our method faithfully reproduces the delicate textures of the exemplar, their global structure and yet respecting the sparse edits of the target image. More results are included in the supplementary.

To compare with state-of-the-art texture synthesis methods, we fed the target images we sketched to the models trained by adversarial expansion (TexExp) [51]. As can be seen in Fig. 6, TexExp failed to reproduce the fine textures of the reference, as the edited target images are unseen data to its training. We also tested the texture optimization based on a recently proposed textural loss (GCD Loss) [49], where the target images are down-scaled as the initialization in its multi-resolution synthesis. Although plausible local textures are synthesized, the output of this method does not always conform to the user-edited layout (see Fig. 6).

We have also applied our method on nearly homogeneous textures of stationary statistics. Since such textures do not have a prominent global structure, we can simply reshuffle patches of the reference to serve as the target. This, however, might cut some local elements in the example. Alternatively, we can use the reference image as the target and shuffle its inversion code before the sampling, which better preserves the local texture elements. Fig. 7 shows a few examples, demonstrating promising quality.

4.2. Ablation Studies

In this section, we analyze the effect of the key components in our method through ablation studies.

KV-Injection in sampling. As KV -Injection is applied both in the inversion and sampling process, we applied two ablations to study its effect. First, we set $P_1 = P_2 = 0$, and explore the full parameter space for S_1 and S_2 , to study the effect of KV -Injection in texture sampling without being affected by the inversion.

Fig. 8 shows a matrix of results of this parameter study. A smaller value of S (S_1 & S_2) means performing more time

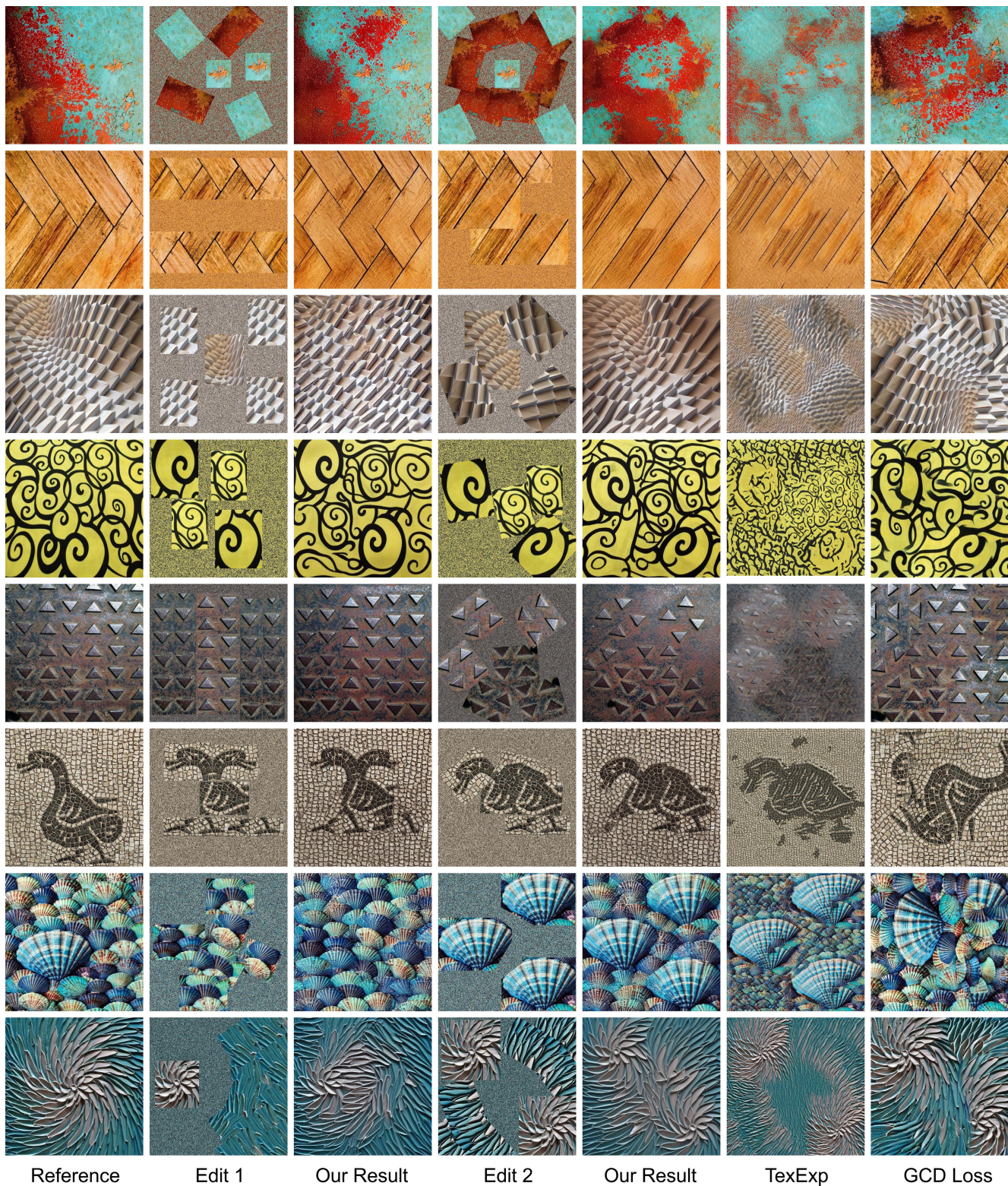


Figure 6. We sketch two targets for each reference, denoted as Edit 1 and Edit 2, respectively. Our self-rectification method generates textures with global structures that faithfully respect the user edits of the target images, while still producing high-quality texture details. In contrast, adversarial expansion (TexExp) [51] does not capture the fine details well. Optimization by GCD Loss [49] reproduces the local textures, but does not always conform to the target image. Note that Edit 2 is used as the input for TexExp, and as the initialization in texture optimization of GCD Loss. More results are included in the supplementary.

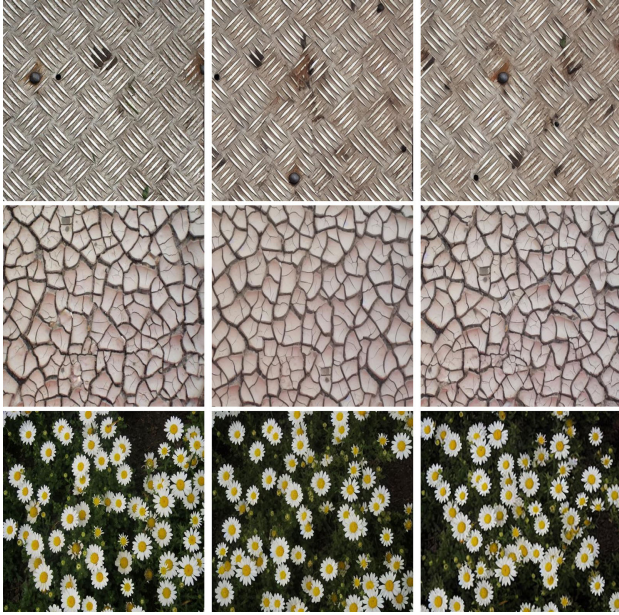


Figure 7. For nearly homogeneous textures (left), we can use patch shuffle of the reference to define a random target layout, where the shuffling could be performed in image space before applying self-rectification (middle), or in latent space after the inversion in the first self-rectification (right).

steps of KV -Injection in sampling, and thus, the target image is rectified to be closer to the reference texture. On the contrary, a larger S reserves more of the target image’s layout, however, at the same time, it introduces more structural errors or conflicts, yielding artifacts. None of these results reaches a good trade-off between synthesizing a reasonable global layout (especially considering the user edits) and reproducing local textures of the reference. Nevertheless, we can find a proper parameter setting for S , which is 10~20 for S_1 , and 5 for S_2 . Hence we set the default values of S_1 and S_2 to be 20 and 5, and use it for all experiments.

KV -Injection in inversion. Next, we investigate the effect of KV -Injection in our structure-preserving inversion. By fixing S_1 and S to default values, we search the parameter space defined by P_1 and P_2 . As shown in Fig. 9, introducing KV -Injection significantly improves the synthesis result. Both the structural errors and local artifacts are drastically reduced at the same time. Another important point is that, we may have a relatively wide range for setting the value of P , which is empirically suggested by the results: 10~30 for P_1 , and 5~15 for P_2 . We usually set P_1 as 20, and P_2 as 5 in production. See supplementary for the full exploration results and more examples of this study.

Data augmentation. In many cases, data augmentation may not be necessary, as the pre-trained Stable Diffusion

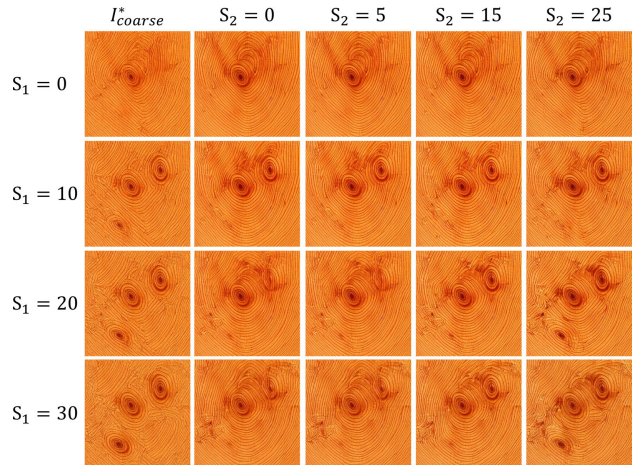


Figure 8. Ablation study of KV -Injection in texture sampling. We explore the parameter space for S_1 and S_2 , with $P_1 = P_2 = 0$, and adopt the wood texture and its edited target shown in Fig. 1 for this test. The coarse self-rectification results are shown in the 1st column. We can see that changing the starting time-step of KV -Injection in the sampling greatly affects the final output. However, none of these settings yields a qualified balance when considering both the target layout and the texture details. Note setting $S = 50$ (*i.e.*, no KV -Injection in sampling) only results in an approximate layout reconstruction without any details synthesized (see Fig. 5).

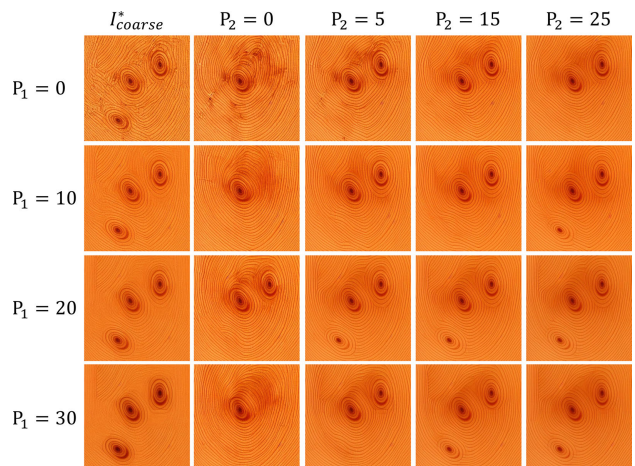


Figure 9. Ablation study of KV -Injection in structure-preserving inversion by exploring the parameter space defined by P_1 and P_2 . Here, we fixed $S_1 = 20$ and $S_2 = 5$ according to the test of Fig. 8. While compared with Fig. 8, the outputs are significantly improved after introducing KV -Injection to the inversion. Many of these results can be considered good synthesis regarding the target image edited and the source texture of the wood.

model already has a certain ability to synthesize rotated texture patterns, except when the reference has dominant directional structures. To allow more free editing for directional textures, augmenting the reference is essential to synthesize

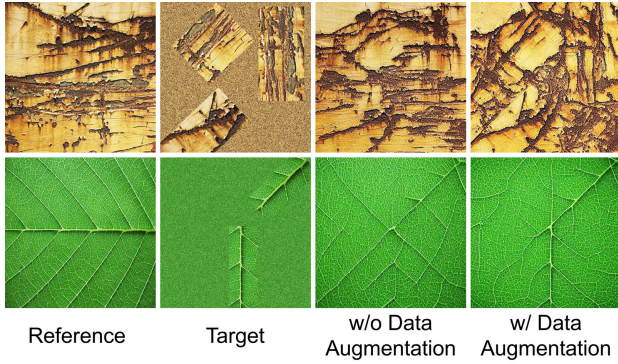


Figure 10. For textures that contain dominant directional structures, data augmentation is essential to allow the output to admit with user edits. Specifically, we augment each source shown above with three images rotated at angles of $\pm 45^\circ$ and 90° , respectively.

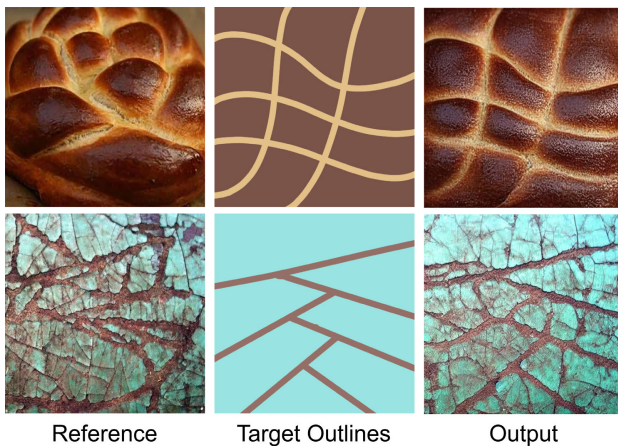


Figure 11. Guided texture synthesis. The user provides a color layout to guide the output structure. The synthesized outputs follow the outlines well and still exhibit the reference texture details.

a more consistent output structure conforming to user edits; see, *e.g.*, Fig. 10 for a comparison on data augmentation, where additional rotated transformations are involved.

4.3. Additional Applications

Our self-rectification framework can extend its utility to seemingly other applications. For instance, as illustrated in Fig. 11, users can input a straightforward target image outlining a layout structure with colors compatible with reference textures. To avoid excessive smoothing, random noise is introduced to enhance the target image. The self-rectification process ensures that the target image incorporates texture details from the reference, while adhering to the user-provided layout structure.

Moreover, our self-rectification method can be employed for image editing beyond textures. Users can edit a given image by employing basic cut-and-paste patch operations,

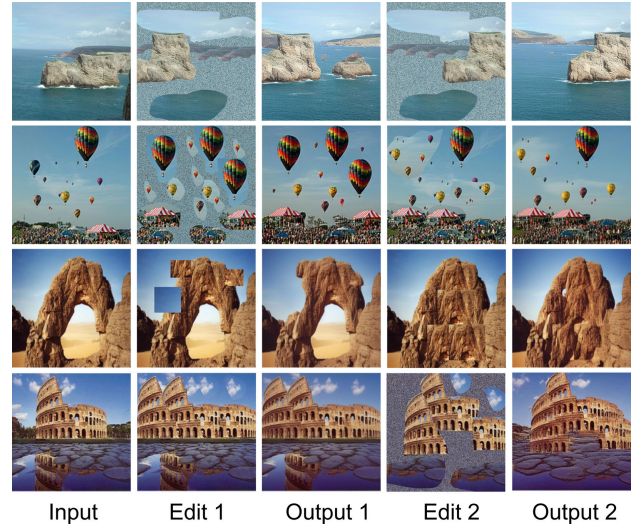


Figure 12. Our method can also be applied to image editing. For creating the target image, the user can either loosely sketch a layout on the target canvas, or directly edit the input image.

and the resulting crude edits undergo self-rectification. Fig. 12 presents several instances of such image editing operations, presenting two edits for each input image.

5. Conclusions

Our work addresses the intricate challenge of synthesizing non-stationary textures, offering a method that empowers users to efficiently design new textures with unprecedented controllability. This stands as a notable improvement over existing methods, providing a user-friendly process, which consists of two steps: Users begin with an initial rough edit using conventional image editing tools, followed by an automated self-rectification process. This process leverages a pre-trained diffusion network and injections of self-attention features, showcasing flexibility in synthesizing a diverse range of challenging non-stationary textures.

In the future, we would like to explore the extension of our approach to synthesize large-scale textures. Additionally, there is a promising avenue to incorporate semantic understanding into the self-rectification process, enhancing alignment with user intent. The integration of texture semantics holds the potential to yield contextually relevant and visually appealing synthesized textures.

Acknowledgements This work was supported in parts by NSFC (62161146005, U21B2023, U2001206), GD NSF (2022A1515010221), DEGP Innovation Team (2022 KCXTD025), Shenzhen Science and Technology Program (KQTD20210811090044003, RCJC20200714114435012), ISF (3441/21), Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ) and Scientific Development Funds of Shenzhen University.

References

- [1] Yuval Alaluf, Daniel Garibi, Or Patashnik, Hadar Averbuch-Elor, and Daniel Cohen-Or. Cross-image attention for zero-shot appearance transfer, 2023. [2](#), [3](#), [4](#)
- [2] Yuval Alaluf, Elad Richardson, Gal Metzger, and Daniel Cohen-Or. A neural space-time representation for text-to-image personalization, 2023. [2](#)
- [3] Rachele Bellini, Yanir Kleiman, and Daniel Cohen-Or. Time-varying weathering in texture space. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 35(4):1–11, 2016. [2](#)
- [4] Sagie Benaim, Ron Mokady, Amit Bermano, and Lior Wolf. Structural analogy from a single image pair. *Computer Graphics Forum*, 40(1):249–265, 2021. [2](#)
- [5] Urs Bergmann, Nikolay Jetchev, and Roland Vollgraf. Learning texture manifolds with the periodic spatial gan. In *Proc. of IEEE Int. Conf. on Machine Learning*, page 469–477, 2017. [2](#)
- [6] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. MasaCtrl: tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proc. of Int. Conf. on Computer Vision*, pages 22560–22570, October 2023. [2](#), [3](#), [4](#), [5](#)
- [7] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models, 2023. [2](#)
- [8] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In *Proc. of SIGGRAPH*, pages 341–346, 2001. [2](#)
- [9] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *Proc. of Int. Conf. on Computer Vision*, 1999. [2](#)
- [10] Anna Frühstück, Ibraheem Alhashim, and Peter Wonka. Tiledgan: synthesis of large-scale non-homogeneous textures. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 38(4):1–11, 2019. [2](#)
- [11] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An image is worth one word: Personalizing text-to-image generation using textual inversion, 2022. [2](#)
- [12] Leon Gatys, Alexander S Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2015. [2](#)
- [13] Eric Heitz, Kenneth Vanhoey, Thomas Chambon, and Laurent Belcour. A sliced wasserstein loss for neural texture synthesis. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*, June 2021. [2](#)
- [14] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-prompt image editing with cross attention control. In *Proc. of International Conference on Learning Representations*, 2023. [2](#)
- [15] Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. Image analogies. In *Proc. of SIGGRAPH*, page 327–340, 2001. [2](#)
- [16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020. [3](#)
- [17] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. [2](#)
- [18] Nikolay Jetchev, Urs Bergmann, and Roland Vollgraf. Texture synthesis with spatial generative adversarial networks. *arXiv preprint arXiv:1611.08207*, 2016. [2](#)
- [19] Alexandre Kaspar, Boris Neubert, Dani Lischinski, Mark Pauly, and Johannes Kopf. Self tuning texture optimization. *Computer Graphics Forum (Proc. of Eurographics)*, 34(2):349–359, may 2015. [2](#)
- [20] Anant Khandelwal. Infusion: Inject and attention fusion for multi concept zero shot text based video editing. *arXiv preprint arXiv:2308.00135*, 2023. [2](#)
- [21] Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. Texture optimization for example-based synthesis. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, August 2005. [2](#)
- [22] Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk, and Aaron Bobick. Graphcut textures: Image and video synthesis using graph cuts. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 22(3):277–286, jul 2003. [2](#)
- [23] Chuan Li and Michael Wand. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *Proc. of Euro. Conf. on Computer Vision*, pages 702–716, 2016. [2](#)
- [24] Yitzhak David Lockerman, Basile Sauvage, Rémi Allègre, Jean-Michel Dischler, Julie Dorsey, and Holly Rushmeier. Multi-scale label-map extraction for texture synthesis. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 35(4):140:1–140:12, July 2016. [2](#)
- [25] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models, 2022.
- [26] M. Lukáč, J. Fišer, P. Ašente, J. Lu, E. Shechtman, and D. Šykora. Brushables: Example-based edge-aware directional texture painting. *Computer Graphics Forum (Proc. of Pacific Conf. on Computer Graphics & Applications)*, 34(7):257–267, oct 2015. [2](#)
- [27] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023. [2](#)
- [28] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022. [2](#)
- [29] Carlos Rodriguez-Pardo and Elena Garces. Seamlessgan: Self-supervised synthesis of tileable texture maps. *IEEE Trans. Visualization & Computer Graphics*, 2022. [2](#)
- [30] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*, pages 10684–10695, 2022. [2](#), [3](#)
- [31] Amir Rosenberger, Daniel Cohen-Or, and Dani Lischinski. Layered shape synthesis: Automatic generation of control maps for non-stationary textures. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 28(5):107:1–9, Dec. 2009. [1](#), [2](#)

- [32] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Proc. of Int. Conf. on Computer Vision*, 2019. 2
- [33] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*, 2023. 2
- [34] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Wei Wei, Tingbo Hou, Yael Pritch, Neal Wadhwa, Michael Rubinstein, and Kfir Aberman. Hyperdreambooth: Hypernetworks for fast personalization of text-to-image models, 2023. 2
- [35] Omry Sendik and Daniel Cohen-Or. Deep correlations for texture synthesis. *ACM Trans. on Graphics*, 36(5):161:1–161:15, July 2017. 2
- [36] Wu Shi and Yu Qiao. Fast texture synthesis via pseudo optimizer. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*, pages 5498–5507, 2020. 2
- [37] Assaf Shocher, Shai Bagon, Phillip Isola, and Michal Irani. Ingan: Capturing and retargeting the "dna" of a natural image. In *Proc. of Int. Conf. on Computer Vision*, 2019. 2
- [38] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020. 3
- [39] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*, pages 1921–1930, June 2023. 2, 3
- [40] Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *Proc. of IEEE Int. Conf. on Machine Learning*, page 1349–1357, 2016. 2
- [41] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017. 3
- [42] Jiaping Wang, Xin Tong, Stephen Lin, Minghao Pan, Chao Wang, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Appearance manifolds for modeling time-variant appearance of materials. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 25(3):754–761, 2006. 2
- [43] Li-Yi Wei and Marc Levoy. Fast texture synthesis using tree-structured vector quantization. In *Proc. of SIGGRAPH*, pages 479–488, 2000. 2
- [44] Yonatan Wexler, Eli Shechtman, and Michal Irani. Space-time completion of video. *IEEE Trans. Pattern Analysis & Machine Intelligence*, 29(3):463–476, 2007. 2
- [45] Li-Yi Wie, Sylvain Lefebvre, Vivek Kwatra, and Greg Turk. State of the Art in Example-based Texture Synthesis. In M. Pauly and G. Greiner, editors, *Eurographics 2009 - State of the Art Reports*. The Eurographics Association, 2009. 1
- [46] Binxin Yang, Shuyang Gu, Bo Zhang, Ting Zhang, Xuejin Chen, Xiaoyan Sun, Dong Chen, and Fang Wen. Paint by example: Exemplar-based image editing with diffusion models. *arXiv preprint arXiv:2211.13227*, 2022. 2
- [47] Hu Ye, Jun Zhang, Sibio Liu, Xiao Han, and Wei Yang. Ip-adapt: Text compatible image prompt adapter for text-to-image diffusion models. In *arXiv preprint arxiv:2308.06721*, 2023. 2
- [48] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023. 2
- [49] Yang Zhou, Kaijian Chen, Rongjun Xiao, and Hui Huang. Neural texture synthesis with guided correspondence. In *Proc. of IEEE Conf. on Computer Vision & Pattern Recognition*, pages 18095–18104, 2023. 2, 5, 6
- [50] Yang Zhou, Huajie Shi, Dani Lischinski, Minglun Gong, Johannes Kopf, and Hui Huang. Analysis and controlled synthesis of inhomogeneous textures. *Computer Graphics Forum (Proc. of Eurographics)*, 36(2):199–212, 2017. 2
- [51] Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Non-stationary texture synthesis by adversarial expansion. *ACM Trans. on Graphics (Proc. of SIGGRAPH)*, 37(4):49:1–49:13, 2018. 1, 2, 5, 6