Content-aware Tile Generation using Exterior Boundary Inpainting

SAM SARTOR, College of William & Mary, USA
PIETER PEERS, College of William & Mary, USA

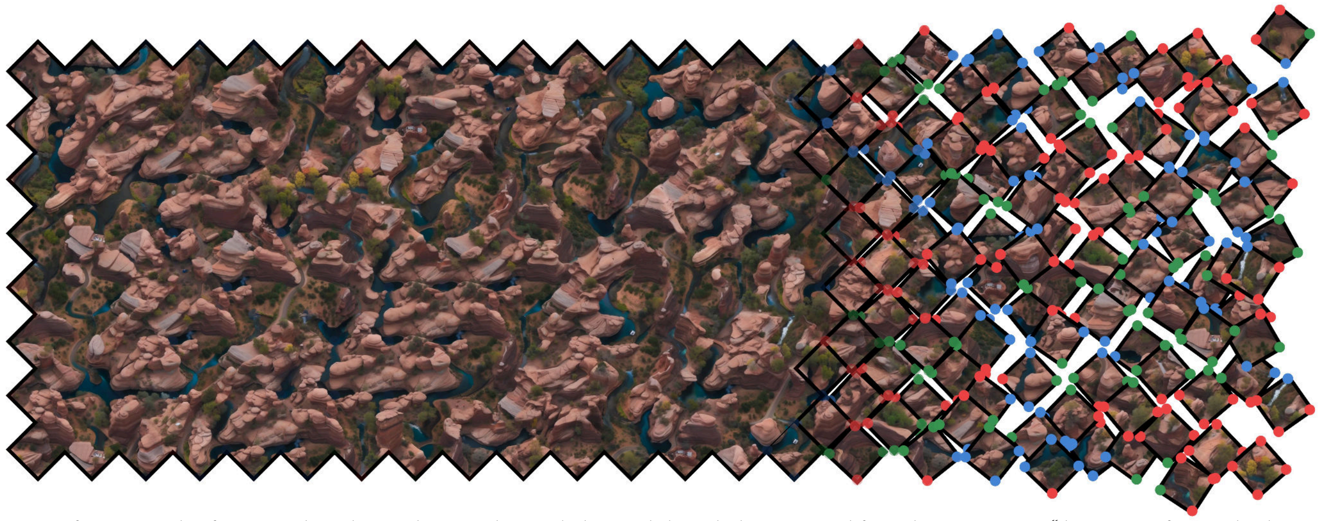


Fig. 1. *Left:* An example of a textured 3-color Dual Wang tiling with diamond shaped tiles generated from the text-prompt *"drone view of canyonlands in utah, rivers, tiny trees"*. *Right:* selected textured tiles used to create the tiling shown on the left.

We present a novel and flexible learning-based method for generating tileable image sets. Our method goes beyond simple self-tiling, supporting sets of mutually tileable images that exhibit a high degree of diversity. To promote diversity we decouple structure from content by foregoing explicit copying of patches from an exemplar image. Instead we leverage the prior knowledge of natural images and textures embedded in large-scale pretrained diffusion models to guide tile generation constrained by exterior boundary conditions and a text prompt to specify the content. By carefully designing and selecting the exterior boundary conditions, we can reformulate the tile generation process as an inpainting problem, allowing us to directly employ existing diffusion-based inpainting models without the need to retrain a model on a custom training set. We demonstrate the flexibility and efficacy of our content-aware tile generation method on different tiling schemes, such as Wang tiles, from only a text prompt. Furthermore, we introduce a novel Dual Wang tiling scheme that provides greater texture continuity and diversity than existing Wang tile variants.

CCS Concepts: • **Computing methodologies** → **Texturing**; **Appearance and texture representations**.

Additional Key Words and Phrases: Wang Tiles, Diffusion, Inpainting, Prompt

Authors' addresses: Sam Sartor, College of William & Mary, Williamsburg, USA, slsartor@wm.edu; Pieter Peers, College of William & Mary, Williamsburg, USA, ppeers@siggraph.org.

ACM Reference Format:
Sam Sartor and Pieter Peers. 2024. Content-aware Tile Generation using Exterior Boundary Inpainting. *ACM Trans. Graph.* 43, 6 (December 2024), 12 pages. https://doi.org/10.1145/3687981

1 INTRODUCTION

Textures are ubiquitous in computer graphics. A rich variety of methods have been introduced to aid in the creation of textures, ranging from specialized texture editors, to exemplar-based texture synthesis methods [Efros and Leung 1999], and to procedural methods [Perlin 1985]. Classic texture synthesis methods, however, generally couple structure and content by verbatim copying parts from an exemplar without understanding the semantics of the content, thereby adversely affecting the diversity of the generated textures.

Advances in machine learning have renewed the interest in texture synthesis. Generative networks, especially prompt-conditioned diffusion models, have good semantic understanding of the image content. The availability of large pretrained diffusion models makes these models an attractive candidate for generating new textures without the need to gather large sets of training textures. However, diffusion models generate whole textures/images at once, and thus these models are limited in the size of the texture they can synthesize. Tileable textures circumvent this problem by synthesizing one or more small textures that can be seamlessly tiled into a larger texture. However, existing learning-based methods [Rodriguez-Pardo et al. 2024; Vecchio et al. 2023] are limited to generating a single self-tiling texture, yielding a tiling that exhibits noticeable repetition.

In this paper we address both tileability as well as diversity in content-aware tile generation. To promote diversity, we forego explicit copying of parts of an exemplar texture, and instead rely on the prior knowledge embedded in prompt-conditioned diffusion models to synthesize a unique content per tile. Our key idea is to condition the synthesis of the image on exterior boundary conditions derived from the exemplar. Conceptually, this is akin to solving for the interior of a domain using partial differential equations (PDE) with boundary conditions at the edge of the domain, except that we leverage a diffusion model for computing the solution corresponding to the content specified by the prompt. For flexibility and simplicity, we avoid designing and training a specialized diffusion model to generate tiles conditioned on prompt and exterior boundary conditions, but instead we leverage existing pre-trained diffusion-based inpainting models by carefully designing and selecting the *exterior* boundary conditions to synthesize each tile's *interior* separately while ensuring tileability.

We demonstrate the flexibility of our content-aware tile generation on a wide variety of tile types: self-tiling tiles, stochastic self-tiling tiles, textured Escher tiles, Wang tiles, and a novel Dual Wang tiling that addresses the difference in diversity between the corners and the interior of Corner Wang tiles [Lagae and Dutré 2005]. To the best of our knowledge, this simple yet flexible method for generating tile sets using exterior boundary inpainting has not been explored in literature, nor have we found any similar implementation on the various public repositories in the diffusion community. To facilitate reproduction of our results and to stimulate further research, a reference implementation of our context-aware tile generation can be found at https://github.com/samsartor/content_aware_tiles.

In summary, our contributions are:

(1) a flexible method for content-aware generation of diverse tile sets by specifying exterior boundary conditions;
(2) that enables tileability for diffusion-based image synthesis beyond self-tiling; and
(3) new tiling schemes such as stochastic self-tiling and textured Escher tiling, as well as a Dual Wang tile formulation that solves the corner versus interior diversity problem with Corner Wang tiles [Lagae and Dutré 2005].

2 RELATED WORK

Texture Synthesis. Classic texture synthesis algorithms can be categorized as either parametric methods [Heeger and Bergen 1995] that optimize a texture to match learned statistics from an exemplar, or as non-parametric techniques that reassemble patches from an exemplar either by copying [Efros and Leung 1999], quilting [Efros 2001], optimization [Kwatra et al. 2005], graph-cuts [Kwatra et al. 2003], or randomized correspondences [Barnes et al. 2009].

Recent successes of neural networks for various image processing tasks also generated a renewed interest in texture synthesis. The vast majority of machine learning based texture synthesis approaches fall in the first category of parametric texture synthesis. Optimization-based approaches replace the manually crafted filters by statistics over activations from a pretrained convolutional neural network [Gatys et al. 2015; Heitz et al. 2021]. However, optimization methods are computationally costly, and various feed-forward networks have been introduced to directly output the resulting texture. These feed-forward network solutions can be categorized on whether they are trained per texture exemplar [Li and Wand 2016; Mardani et al. 2020; Rodriguez-Pardo and Garces 2022; Ulyanov et al. 2016; Zhou et al. 2023, 2018] or per texture category [Bergmann et al. 2017; Guo et al. 2022; Li et al. 2017; Ulyanov et al. 2017; Yu et al. 2019]. The vast majority of these methods can only synthesize a larger texture as a whole, either by repeated doubling [Guo et al. 2022; Li et al. 2017; Ulyanov et al. 2016, 2017; Zhou et al. 2023, 2018], leveraging a fully convolutional architecture [Bergmann et al. 2017; Li and Wand 2016], or by applying a neural Fast Fourier Transform [Mardani et al. 2020]. An alternative approach to feed-forward networks are pointwise evaluation models parameterized as a Multi-Layer Perceptron (MLP) [Henzler et al. 2021, 2020; Portenier et al. 2020] that take a position and a crop from a large noise field as input. While these methods can synthesize an infinitely large texture, they are limited to textures with a high degree of stochasticity.

A final class, to which our method also belongs, are tiling-based synthesis methods which sit in the middle between point-evaluation and feed-forward methods. Tile-based methods precompute small tiles that are merged together at run-time. Classic tileable synthesis methods either explicitely maximize stationarity [Moritz et al. 2017] or extract the largest tileable patch from an exemplar [Rodriguez-Pardo et al. 2019]. However, tiling a single texture often results in a clear visible repetition. Vanhoey et al. [2013] and Kolvar et al. [2016] exchange patches in a self-tiling texture with a small precomputed set of compatible (i.e., seamless) patches from elsewhere in the tile. While more diverse, these methods still rely on verbatim copying from the same tile. Rodriguez-Pardo and Garces [2022] specialize a Generative Adversarial Network (GAN) architecture to synthesize a single tileable texture. Frühstück et al. [2019] tile, based on a guidance map, the activations of an intermediate layer of a GAN trained to synthesize tiles for a particular texture class (e.g., a terrain map). The resulting tiled latent field is then processed by the remainder of the GAN to produce a seam-free final texture. Zhou et al. [2022] condition a GAN on a template to control the tileable structure. All three prior learning-based tiling methods employ a GAN architecture which often needs to be retrained for new texture categories. We circumvent the problem of retraining (and thus gathering a sufficiently large and diverse set of texture exemplars) by leveraging existing pretrained text-to-image diffusion models. Furthermore, using a text-to-image diffusion model also allows the user the specify the texture with a text-prompt instead of providing an appropriate exemplar and/or structure-template. Finally, we revisit Wang tiles in the context of learning-based texture synthesis, which to the best of our knowledge has not yet been explored.

Wang Tiles. Wang tiles [1961] are squares with colored edges that can tile the 2D plane by adjoining tiles with matching colored edges. Cohen et al. [2003] introduced a patch-based method for synthesizing textured tiles that meet the Wang tile matching rules to enable fast synthesis of large textures. Wei [2004] showed that, once the texture tiles are precomputed, the tiling process can be directly implemented on graphics hardware. Furthermore, Wei showed that the Wang tiling can be evaluated on the fly without the need to synthesize the full tiling. Other graphics applications of Wang tiles include blue noise generation [Kopf et al. 2006; Lagae and Dutré 2005], fabrication [Liu et al. 2022], and texturing an arbitrary 3D shape [Fu and Leung 2005]. Texture synthesis with edge-colored

Wang tiles suffers from the aptly named *corner-problem* where the corners of each texture tile are the same for all tiles thereby creating a noticeable repetition in the tiled textures. The corner-problem is inherent to edge-colored Wang tiles because diagonally neighboring tiles are not directly constrained, and therefore each corner must match all other possible corners. Corner Wang tiles [Lagae and Dutré 2005; Ng et al. 2005] overcome this issue by matching colored corners instead of colored edges. We also leverage Wang tiles to support fast synthesis without the need to synthesize the whole 2D plane. However, unlike the above methods, we do not require an exemplar sample, but instead allow the user to specify the texture via a text-prompt and directly synthesize the different Wang tiles. In addition, we introduce a novel Dual Wang tile variant that increases diversity and that does not suffer from the corner-problem. Finally, due to the flexibility of our content-aware tile generation, our tiles show greater diversity than prior graph-cut generated tile textures.

Generative Diffusion Models. Diffusion models formulate the generative process of a signal as an iterative neural denoising process [Karras et al. 2022; Song et al. 2021] outperforming the state-of-the art in image synthesis tasks [Dhariwal and Nichol 2021]. When conditioned on text-prompts [Nichol et al. 2022; Ramesh et al. 2022; Rombach et al. 2022; Saharia et al. 2022], diffusion models enable non-artists to concretize their mental images. Diffusion models have been successfully applied to a wide variety of downstream tasks, including text-based image editing [Kawar et al. 2023; Kim et al. 2022; Liu et al. 2020; Mokady et al. 2023; Tumanyan et al. 2023], sketch and depth-based synthesis [Ham et al. 2023; Voynov et al. 2023; Šubrtová et al. 2023; Ye et al. 2023; Zhang et al. 2023], and appearance capture [Sartor and Peers 2023; Vecchio et al. 2023].

A related class of prior work are methods that leverage diffusion models to directly synthesize textures on 3D shapes [Chen et al. 2023; Liu et al. 2023; Richardson et al. 2023; Xiang et al. 2023; Xu et al. 2023; Zeng et al. 2023]. However, these methods generate a fixed texture of finite size. In contrast, we leverage diffusion models to generate textured tiles suitable for real-time synthesis of, potentially, infinite textures. Similar to prior work, we rely on powerful pretrained text-to-image diffusion models, without fine-tuning or retraining, to sample the space of textures.

Most related to our method are tileable diffusion variants. Vecchio et al. [2023] introduced a method to generate tileable SVBRDFs (i.e., a 10-channel image) using noise-rolling. Noise-rolling *"rolls"* the noise tensor by a random translation each diffusion step, thereby placing the seam at a random location. The diffusion models subsequently attempts to remove this seam as it is not a natural feature. Furthermore, to condition noise-rolling on a non-tileable exemplar, Vecchio et al. introduce a conditional noise rolling variant that masks a small region around the *input* border (1/16 of the image size) allowing the diffusion model to seamlessly 'inpaint' the missing texels. Rodriguez-Pardo et al. [2024] introduce *"TexTile"*, a differentiable tileability metric. TexTile can be leveraged to force a diffusion model (i.e., SinFusion [Nikankin et al. 2023]) to produce self-tiling images by interleaving each diffusion step with a TexTile optimization step. It is unclear how either noise-rolling or TexTile can be extended beyond the generation of a self-tiling image. In contrast, we demonstrate that our content-aware tile generation method is also applicable to more general tiling schemes such as Wang tiles.

3 EXTERIOR BOUNDARY INPAINTING

Our goal is to generate a small set of one or more mutually tileable images from a text-prompt and optionally an exemplar image. When provided, the content of the optional exemplar and text-prompt should match. Alternatively, the exemplar image can also be generated from the text-prompt. In our implementation, we use *Stable-Diffusion-XL* [Stability AI 2022b] to generate the exemplars from a text-prompt and apply (unconditional) noise-rolling [Vecchio et al. 2023] to obtain a more texture-like exemplar; a similar result can be obtained with appropriate prompt engineering.

We introduce our method starting with the simplest tiling configuration (i.e., a self-tiling texture), and then demonstrate our method's flexibility by applying the same methodology to more complex tiling schemes, culminating in a novel Dual Wang tiling.

Self-tiling Texture. We start with the most straightforward tileable texture (a self-tiling texture that does not introduce visible seams when tiled over a 2D plane) as a didactic example to explain the core of our method; similar results can also be obtained with prior self-tiling methods [Rodriguez-Pardo et al. 2024; Vecchio et al. 2023].

To promote diversity, we aim to fully synthesize the interior tile without verbatim copying of parts from the exemplar image. Instead, we will leverage the exemplar to establish exterior boundary conditions that constrain the synthesis of the image to be tileable. To synthesize the self-tiling texture, we employ an existing prompt-conditioned diffusion-based inpainting model (*Stable-Diffusion-2-Inpainting* [Stability AI 2022a]). The role of the prompt is to constrain the content, whereas the boundary conditions define the structure near the tile edges. Inpainting methods require a wide enough strip of example pixels surrounding the target region in order to guarantee continuity and a consistent structure. At the same time, to ensure continuity at matching edges, the combined boundary strips at matching edges need to be contiguous and not exhibit any seams. We can fulfill both goals, by selecting a template patch from the exemplar (with a size similar to the target tile size) for each pair of matching edges. In the case of a self-tiling image, we have two pairs of matching edges: the horizontal and vertical edge pairs. Next, we cut the template patches in half horizontally and vertically respectively, and copy each half template patch to the *outside* of the tile with the cut-edge abutting the tile edges acting as the exterior boundary conditions. By construction, these boundary conditions will be contiguous across matching edges. Finally, we generate the interior of the tile by inpainting. We only retain the synthesized interior as the final tile such that no pixels from the exemplar image end up in the final tile texture. Note, that we also inpaint the corners of the image not covered by the template halves; this helps in creating reasonable content for the corners of the tile. Figure 2 illustrates the process, and Figure 8 (top row, 1st column) shows an example of a tiled texture.

Stochastic Self-tiling Texture. Self-tiling textures often produce visibly repeating patterns, and the resulting texture is visually not very diverse. To enrich the tiling, we leverage that diffusion-based

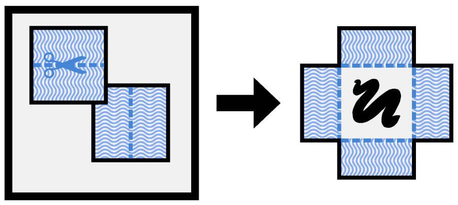


Fig. 2. **Self-tiling Texture Generation:** We establish contiguous horizontal and vertical boundary conditions by selecting two template patches from the exemplar, and cutting them in half horizontally and vertically respectively. Each cut half is placed on the outside of the tile with the cut edge (dashed line) aligned with a tile edge. The interior of the tile (scribbled area) is then inpainted. The final tile is then cropped to only retain the synthesized part.

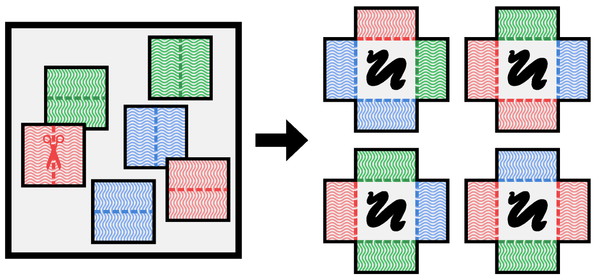


Fig. 3. **Wang Tile Generation:** Given C colors, we select $2 \times C$ template patches from an exemplar image, and cut each template patch in half (a horizontal and a vertical cut per color, marked by the dashed line). For each Wang tile we set the boundary conditions by coping each half template patch to the exterior of the Wang tile with the (dashed) edge matching the corresponding Wang tile edge. Finally, we generate the Wang tile texture by inpainting the interior region (i.e., the scribbled area).

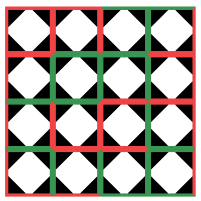
inpainting takes, besides an image and a mask indicating the area to be inpainted, also a seed as input. Changing the seed will result in a slightly different generated texture. Keeping the boundary conditions (i.e., template patches) the same, allows us to generate multiple textured tiles that are mutually tileable. This yields a simple tiling algorithm where we randomly select, during tiling, which texture tile to use, yielding a more diverse tiled texture. Figure 8 (top row, 2nd column) shows an example of such a stochastic self-tiling texture.

Escher Tiles. Salient features that cross a tile edge impose stricter constraints on the tile generation process. While we could select different boundary conditions, it is not always possible to find straight boundary conditions without salient features. However, we observe that diffusion-based inpainting imposes no constraints on the shape of the boundaries. Hence, we can trivially generate non-square *"Escher"* tiles by cutting each template patch along an arbitrary path. Figure 8 (middle row, 1st column) shows an example of such a self-tiling Escher texture.

Wang Tiles. The above stochastic tiles are good for textures with isolated objects and simple boundaries (e.g., shells in sand). However, if the features at the tile boundaries are distinct, the resulting texture will exhibit a clearly visible repetition. Cohen et al. [2003] proposed to use textured Wang tiles to produce tiled textures with greater diversity. We can apply a similar method for generating Wang tiles as the previous tiling variants. However, unlike the self-tiling case, we select two template patches (for horizontal and vertical splitting respectively) per Wang tile edge color (= $2C$ template patches for C colors). For each of the C^4 Wang tiles, we generate the texture by copying the appropriate half template patches (based on the edge color) as an exterior boundary condition, and inpaint the interior region as before (Figure 3). Figure 8 (middle row, 2nd column) shows an example of a titled texture generated with a 3-color textured Wang tile set (= 81 tiles). An advantage of our exterior boundary inpainting strategy is that our textured Wang tiles will exhibit high diversity since each tile only shares the boundary conditions and the interior of each textured tile is inpainted separately. In contrast, the graph-cut based method of Cohen et al. reuses the same diamond templates, and thus the same texture content appears in multiple textured tiles. Similar as in stochastic self-tiling, we can also further increase diversity by generating multiple textures per Wang tile and randomly selecting one at tiling-time (a similar strategy was also proposed by Cohen et al. [2003]).

Dual Wang Tiles. Wang tiles suffer from the so-called *"corner problem"* [Cohen et al. 2003; Lagae and Dutré 2005], i.e., each corner texel is shared between all tiles because the tiles placed diagonally across a corner do not share any edges and thus each corner needs to match to the opposing corner of *any* other tile. Lagae and Dutré [2005] introduced Corner Wang tiles as a solution; please refer to the supplemental document for a description of how to apply content-aware tile generation to Corner Wang tiles. While Corner Wang tiles avoid the corner problem, it also introduces two new practical problems. First, there is a difference in diversity between the corners (copied from the exemplar) and the center (synthesized). Second, as observed by Lagae and Dutré [2005], discontinuities often occur close to the center of each tile edge as these regions are weakly constrained by the surrounding copied patches.

We introduce a new Dual Wang tile variant to solve both problems. The key idea is to make the content of the corner regions of a Wang tile dependent on its neighboring tiles. We start from a regular Wang tiling. However, instead of attaching a square texture to each tile, we attach a diamond-shaped texture (each diamond's corner touches the center of each Wang tile edge). We call these diamond shaped textures the *"interior tiles"* (white diamonds in the inset). Each interior tile is identified by the edge-colors of the surrounding Wang tile, and thus there are C^4 different interior tiles. Tiling a plane with such interior tiles results in a texture with diamond shape holes between each four interior tiles that share a corner. We employ a second set of diamond shaped textures, named *"cross tiles"* (black diamonds in the inset), to seamless fill the resulting hole. Each cross tile is identified by the colors of the edges incident on the Wang tile-corner at the center of the cross tile. Hence, there also exist C^4 possible cross tiles. The combined set of interior and cross tiles form the Dual Wang tile set. Since every interior tile can be combined with different cross tiles (depending on its neighboring tiles), both the interior and corner regions are now equally diverse.



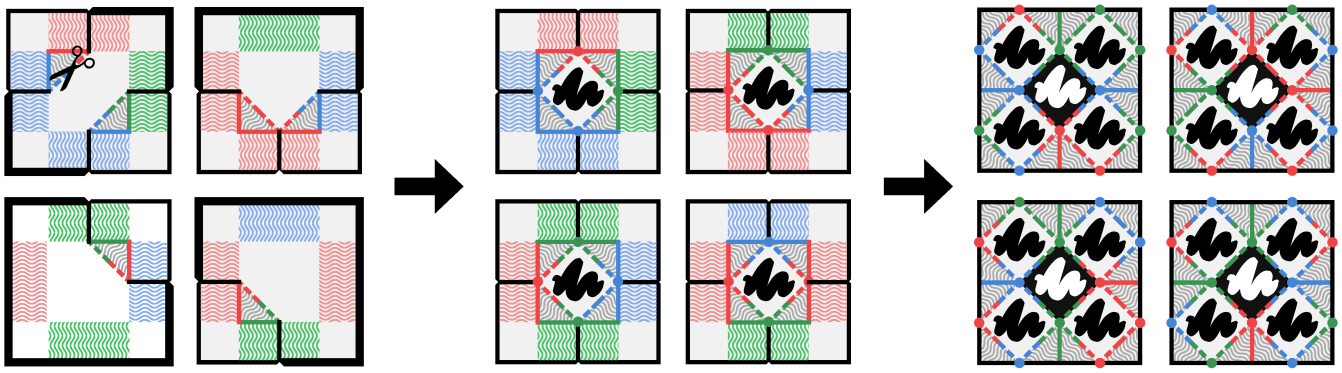


Fig. 4. **Dual Wang Tiles** are created in a three-stage process. Left: we first generate a subset of $4 \times C^2$ regular Wang tiles, and extract the boundary conditions for each interior tile edge (i.e., 4 edges, each identified by 2 colors or C^2 combinations). Middle: we assemble the generated boundary conditions corresponding to each interior tile's color combinations, and inpaint and extract the central interior tile texture (black scribble). Right: the cross texture tiles (white scribble) are generated by inpainting a 2×2 tiling of the plane with the interior tiles as boundary conditions.

Our goal now is to create two sets of textured tiles, one for the interior tiles and one for the cross tiles. Each tile does not need to seamlessly tile (along an edge) with another tile from the same set, but it does need to tile with corresponding tiles from from the other set. Consequently, we can synthesize both sets separately; once we have one set (e.g., interior tile set), we can use it as boundary conditions for generating the second set. Additionally, we want to impose a corner-continuity-constraint at the (diamond) corners in both sets in order to avoid discontinuities as in Corner Wang tiles.

We start with the observation that the boundary conditions used for generating regular Wang tiles already meet the (diamond) corner-continuity-constraint. However, these are not sufficient conditions for seamless Dual Wang tiles, as we further need to constrain each interior/cross tile edge. Since each interior/cross tile edge is identified by two colors, there are C^2 possible conditions per edge (or $4 \times C^2$ in total). Unlike the corner-continuity-constraint, the edge constraints do not need to enforce continuity between multiple sides of tiles from the same set; it only needs to impose the same constraint for corresponding tile edges in the other set. Both constraints cannot be met by simply cutting the boundary constraints from the exemplar. Instead, we first generate a regular Wang tile set, which by construction is contiguous across the edges (with matching colors), and thus also across the diamond-corners. It therefore forms a potential source for extracting the Dual Wang tile boundary conditions. However, each interior/cross edge-color combination occurs multiple times over the Wang tile set, each with potentially different synthesized pixel values along the diamond-tile edges. We therefore select a subset of the Wang tile set such that each combination only occurs once (i.e., a subset of $4 \times C^2$ Wang tiles). From this subset, we can then extract the boundary conditions for generating the interior tiles (Figure 4, left). Note, because we want to retain continuity across the interior/cross tile corners, we also include the original boundary conditions for each Wang tile, yielding a square-shaped exterior boundary condition with a triangular corner (i.e., overlapping with the interior tile) cut out.

To generate the interior tiles, we assemble the Wang-tile-extracted boundary conditions for each interior edge, and inpaint the interior tiles (Figure 4, middle). Once we have generated all interior tiles, we can then synthesize the cross tiles by creating a 2×2 tiling of interior tiles with corresponding colors which we use as boundary conditions for inpainting the cross tiles (Figure 4, right).

The generated Dual Wang tiles solve both of the identified issues with Corner Wang tiles. First, all tiles in both sets are fully generated, and no part of the boundary conditions are included in the final textured tile sets. Therefore, no part of the textured tile set occurs at a different frequency. Second, the Wang-tile-extracted boundary conditions provide boundary conditions across the diamond boundaries, as well as across the diamond-tile corners. Figure 8 (bottom row) shows an example of a Dual Wang tiled texture.

4 RESULTS

We implemented our content-aware tiling in PyTorch [Paszke et al. 2019], using *Stable-Diffusion-XL* [Stability AI 2022b] to generate the exemplar image from a user-provided text-prompt, and *Stable-Diffusion-2-Inpainting* [Stability AI 2022a] for tile generation. Any sufficiently performant combination of text2image and inpainting models could be easily used instead, including future more advanced models, without the need for any additional training or per-model customization. All results in this paper use a tile texture resolution of 256×256 and a tiling of 3 edge colors. We use an Euler sampler with 40 inference steps, and lower the CFG scale to 7.5.

Figure 9 shows four different generated tiled textures using our Dual Wang tile scheme. The tiled textures, as shown, have an effective resolution of 7168×2560 or 28×10 tiles (i.e., each tile appears on average 3.5 times). As can be seen, our tiled textures include fine details as well as a large diversity in the generated tiles. *We refer to the supplemental material for additional generated tiled textures.*

Template Patch Selection. The exterior boundary conditions are determined by selecting patches from the exemplar image. For many

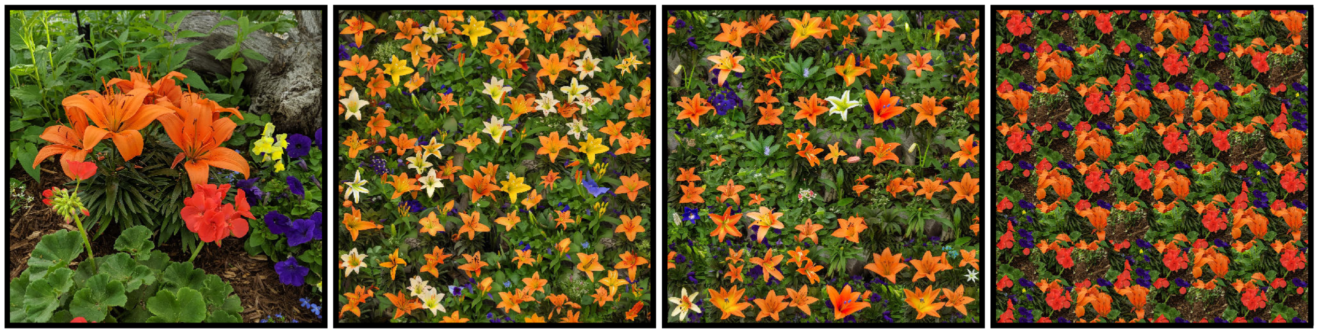


Fig. 5. **Img2Tile:** Our method can also start from template patches selected from a suitable photograph and a descriptive prompt (*"bright orange lily in a flower garden, small blue and white flowers, leaves"*) (left). Please zoom-in on the Dual Wang tiled texture (2nd) and regular Wang tiled texture (3rd) to fully appreciate the generated texture detail. Our tile generation method produces a more diverse set of tiles compared to graph-cut based tile synthesis (right).

exemplar images a simple random selection often works surprisingly well (e.g., the "pebbles in a stream" and "woven basket" in Figure 9). Similar to Cohen et al. [2003] we potentially can also repeatedly select a random set of candidate template patches from which we retain the template patches that produce the highest quality tiles according to CLIP-IQA [Wang et al. 2023]. Alternatively, when the image contains strongly aligned features (e.g., "wooden floor" in the supplemental material), we can exploit that our method only requires that opposing vertical features along the horizontal boundaries are aligned and likewise for opposing horizontal features along the vertical boundaries. Hence, we constrain the selection to first pick a random horizontal/vertical line along which all vertical/horizontal template patches are randomly selected. This constrained selection method is unique to our method as prior graph cut-based methods require that all four quadrants are mutually aligned and thus are unlikely to align structures when the horizontal and vertical (constrained selection) line is randomly selected. Finally, if artistic control is desired, then the user can also manually select the template patches by simply marking the horizontal and vertical boundary lines per patch.

Tile Rejection. Diffusion-based inpainting can introduce unwanted features such as noticeable seams and fake watermarks. Since every tile is generated independently (given a set of boundary conditions), we can simply regenerate the tile that exhibits the undesirable image feature using a different seed. The user can either manually mark the affected tiles or follow a similar selection strategy as for the automatic template patch selection. For each tile, we generate multiple candidates (4 in our implementation), each with a different seed, and retain the candidate with the highest quality. In this case we use SIFID [Shaham et al. 2019], as we found that CLIP-IQA is blind to fake watermarks; the most common artefact produced by *Stable-Diffusion-2-Inpainting*.

Performance. The computational cost of context-aware tile generation depends on the tiling variant and on whether tile rejection is used. We generally require 40ms per U-Net evaluation on an NVidia A5000, times 40 inference steps per inpainting operation, yielding a 1.7s total computation time (including VAE encoding/decoding) per tile. When generating 3-color Wang tiles (i.e., 81 tiles), this corresponds to a total of 140s. When evaluating 4 candidate tiles, including the cost of scoring, this cost increases to 12 minutes. For 3-color Dual Wang tiles, we perform 243 inpainting operations, yielding 7 minutes without rejection, or 36 minutes with 4 candidate rejection. The cost of rejection can be avoided by manually selecting which tiles to regenerate (approximately 2-5 minutes overhead). Newer diffusion models are significantly faster (0.2s instead of 1.7s, or an 8× speedup), but we did not implement such optimizations. Furthermore, our method can trivially be parallelized over multiple GPUs. We believe that, with careful engineering, the computational synthesis cost can be significantly lowered.

5 ADDITIONAL APPLICATIONS

Img2Tile. All results shown so far have been generated starting from a text-prompt. However, we can also skip the initial step and extract the template patches from a suitable photograph; we also also expect the user to provide a text-prompt describing the image content. Once the template patches have been selected, the tile generation proceeds as before. Figure 5 shows an example of a 3-color Dual Wang tiling and a regular Wang tiling with textured tiles generated from a photograph of flowers with manually selected template patches from the input photograph. To ensure that the flowers fit within a tile, we downsample the input photograph by a factor 4. Because our tile generation leverages inpainting, we never directly copy any texels from the input photograph. This results in a greater diversity of salient objects; i.e., different orientations/poses and even semantically correct recoloring (e.g., the exemplar only contains orange Lilies, whereas the tiles also includes white Lilies).

Infinite Stochastic Tiling. Each tiling scheme discussed in Section 3 also supports stochastic tiling; we simply generate additional tiles with the same template patches using different inpainting seeds. We can also postpone the actual inpainting of the tiles to tiling-time. During preprocessing we compute the necessary boundary conditions (i.e., the template patches). When evaluating the tiled texture, we generate the corresponding tile on-the-fly with a random seed. The effective result would be equivalent to a tiling with an infinitely large stochastic tile set. Figure 6 shows an example of such an infinite stochastic Wang tiled texture. While more diverse, it does

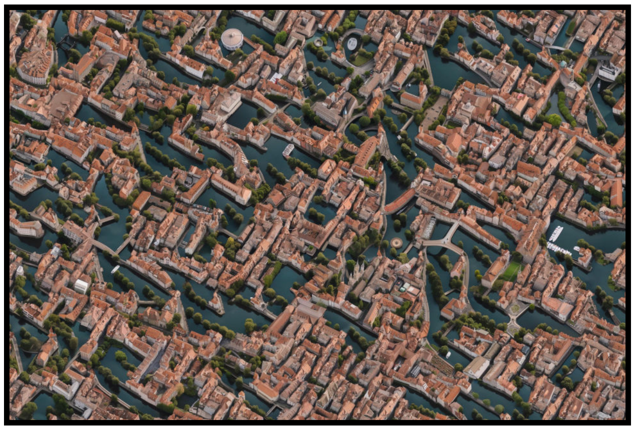


Fig. 6. **Infinite Stochastic Tiling:** An example of on-the-fly generation of Wang tile textures with a random seed to produce a unique texture per tile.

Table 1. Quantitative comparison of context-aware tile generation for different tiling schemes using CLIPScore [Hessel et al. 2021] (for semantic similarity; higher is better), CLIP-IQA [Wang et al. 2023] (for quality; higher is better), and average correlation of the inception features [Salimans et al. 2016] (to measure diversity; lower is better) over the 12 classic textures from SeamlessGAN [Rodriguez-Pardo and Garces 2022], as well as semantically more complex textures shown in this paper. Additionally, we also include the scores for Wang tiles synthesized using the graph-cut based method of Cohen et al. [2003] (3rd row marked in gray). All scores, except single self-tiling, are computed using 81 tiles per scheme.

	Classic Texture			Semantic Textures		
	CLIP Score	CLIP IQA	Incep. Correl.	CLIP Score	CLIP IQA	Incep. Correl.
Single Self	24.587	0.765	⁄	23.370	0.746	⁄
Stoch. Self	**27.018**	0.767	9.368	28.070	0.744	5.144
Wang (cut)	24.467	**0.785**	10.089	26.548	0.739	5.773
Wang Tile	26.002	0.762	9.293	**28.57**	0.756	5.251
Dual Wang	26.327	0.765	**8.331**	28.44	**0.760**	**4.875**

come at a significant computational overhead; future advances in inpainting might make this variant less costly and more practical.

6 EVALUATION

Comparison to Graph-cut based Synthesis. Cohen et al. [2003] propose to generate tile textures using graph-cut quilting of diamond shaped templates cut from an exemplar. Because the same texels from the templates are copied to multiple tile textures, the resulting tiles (Figure 5, right) are visually less diverse compared to our inpainting-based method (Figure 5, 3rd column) that generates a related, but unique, texture per tile.

Quantitative Comparison. To quantitatively assess our content-aware tile generation method we employ two commonly used quality metrics: CLIP-IQA [Wang et al. 2023] to measure the overall quality of the tiles, and CLIPScore [Hessel et al. 2021] to measure the semantic similarity with the prompt. Besides these classic image quality measures, diversity between the different tiles also matters. We express this by computing the average pairwise correlation of the inception features [Salimans et al. 2016] between tiles. The lower the correlation, the more diverse the tile-set. Other common metrics such as SSIM and LPIPS assume aligned images, and are therefore not suited for evaluating the quality of tiles [Rodriguez-Pardo et al. 2024]. SIFID [Shaham et al. 2019] between the tiles and the exemplar is also not suited as it penalizes diversity. We also do not use TexTile [Rodriguez-Pardo et al. 2024] because it is intended for comparing self-tiling images.

We compute the average scores over all the examples included in this paper and the supplemental material. Because our exemplar images are semantically more complex than commonly used stationary and stochastic textures, we also report the scores on 12 test textures (resolution > 256) from SeamlessGAN [Rodriguez-Pardo and Garces 2022] (*see supplemental material for corresponding tilings*). We manually create a text-prompt for each test texture in the SeamlessGAN set, and use it as input to our content-aware tile generation, as well as for evaluating CLIPScore. Table 1 lists the scores for each test set for context-aware generated textures for different tiling schemes. For completeness, we also include the scores for graph-cut based Wang tile synthesis [Cohen et al. 2003]. All tilings except single self-tiling consist of 81 tile textures. The quantitative comparisons show that our method produces higher quality Wang tile textures for typical textures than the graph-cut based tile synthesis. We found that context-aware tile generation generally performs relatively better on semantic textures that exhibit less regular structures than on the classic SeamlessGAN test set. The quantitative comparison also demonstrates that with a large tile set (81 in this case), stochastic self-tiling is a viable alternative especially for inconspicuous boundaries. Finally, Dual Wang tiling performs overall best, and it produces the most diverse tilings for *both* test sets.

Self-tiling Comparison. While the focus of our method is on generating tile sets of multiple mutually tileable images, our method can also be used for generating self-tiling images. While there exist a number of prior learning-based methods for generating self-tiling images, it should be stressed than none can be extended to produce multi-tile sets or stochastic self-tiling images.

For completeness, Figure 7 and Table 2 compare our method against recent self-tiling image generation methods: the feedforward SeamlessGAN [Rodriguez-Pardo and Garces 2022], the optimization based Neural Texture Synthesis [Heitz et al. 2021] with a TexTile loss [Rodriguez-Pardo and Garces 2022] to promote tileability, the SinFusion [Nikankin et al. 2023] single-image diffusion model in which each denoising step is interleaved with an TexTile maximization step, and Conditional Noise Rolling [Vecchio et al. 2023] using the same diffusion models as our method. Because SeamlessGAN is not trained for semantic textures, we only compare on the SeamlessGAN dataset for fairness. In addition to CLIPScore [Hessel et al. 2021] and CLIP-IQA [Wang et al. 2023] used in the multi-tile comparison (Table 1), we also include SIFID [Shaham et al. 2019] to measure the similarity to the exemplar, and TexTile [Rodriguez-Pardo and Garces 2022] to quantify self-tileability. For both Conditional Noise Rolling and our method, we perform tile selection with 4 candidate tiles using SIFID as the selection criterion; for completeness we also include a variant where we replace the selection criterion by TexTile

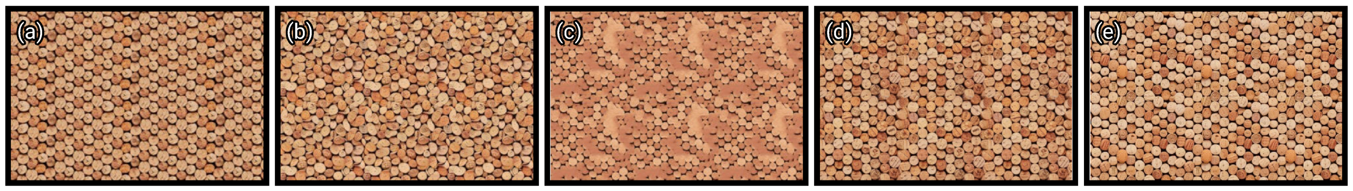


Fig. 7. **Self-tiling Qualitative Comparison** of SeamlessGAN (a), Neural Texture Synthesis with TexTile loss (b), SinFusion with TexTile loss (c), Conditional Noise Rolling (d), and our context-aware self-tiling (e) on an exemplar from the SeamlessGAN test set.

Table 2. Quantitative comparison of different learning-based self-tiling texture generation methods (SeamlessGAN [Rodriguez-Pardo and Garces 2022], Neural Texture Synthesis [Heitz et al. 2021] with a TexTile loss [Rodriguez-Pardo and Garces 2022], SinFusion [Nikankin et al. 2023] interleaved with a TexTile optimization step, and Conditional Noise Rolling [Vecchio et al. 2023]) using CLIPScore [Hessel et al. 2021] (for semantic similarity; higher is better), CLIP-IQA [Wang et al. 2023] (for quality; higher is better), SIFID [Shaham et al. 2019] (for similarity to the exemplar; lower is better), and TexTile score (for self-tileability; higher is better) over the 12 classic textures from SeamlessGAN [Rodriguez-Pardo and Garces 2022]. For both Conditional Noise Rolling and our method, we employ tile selection (using SIFID or TexTile) from 4 candidate tiles.

	CLIP Score	CLIP IQA	SIFID	TexTile
SeamlessGAN	22.528	0.7803	9.215	0.7149
Neural Tex. Synth. (TexTile)	22.350	**0.8012**	8.599	**0.7783**
SinFusion (TexTile)	20.829	0.7918	10.184	0.7407
Cond. Noise Roll. (SIFID)	22.668	0.7447	**6.607**	0.4826
Ours Self Tiling (SIFID)	22.857	0.7717	7.978	0.6950
Cond. Noise Roll. (TexTile)	**23.212**	0.7445	6.926	0.5270
Ours Self Tiling (TexTile)	22.228	0.7737	9.119	0.7494

to favor tileability rather than quality. From Table 2 we see that none of the methods outperforms the others on all error metrics. Conditional Noise Rolling does not score well on tileability (TexTile) due to the very small inpainting region (1/16th of the image size). Our method with SIFID-based candidate selection performs similar to SeamlessGAN while adhering better to the exemplar (SIFID). When using TexTile for tile selection, our method performs similar to the methods that explicitly optimize tileability. Qualitatively (Figure 7) we see that Conditional Noise Rolling (d) and our method with SIFID selection (e) best preserve the characteristics of the exemplar without deforming the corks (e.g., Neural Texture Synthesis (b)) or reducing variation (e.g., SinFusion (c)). Moreover, in contrast to SeamlessGAN (a), our results appear less regular and exhibit less tileability artifacts (e.g., Conditional Noise Rolling). We believe our context-aware tile generation method (with SIFID selection) strikes a good balance between exemplar similarity and tileability.

7 DISCUSSION

Dual Wang Tile Storage Requirements. Dual Wang tile sets contain double the number of tiles compared to a regular Wang tile set. However, it should be noted that the interior and cross tile textures contain just 50% of the texels compared to regular Wang tile textures. Hence, when packed in a single texture map, the dual Wang tiles have exactly the same storage costs.

Wei [2004] packs a complete regular Wang tile set in a single texture using every tile only once and where each edge is matched. Such a packing ensures minimal storage and fast fetching and, importantly, correct texture filtering. However, Wei's Wang tile packing only guarantees that each tile occurs once (i.e., interior Dual Wang tile), but it does not offer such guarantee for each cross tile. Fortunately, a valid minimal Dual Wang tile packing is possible, and we refer to the supplemental document for novel generative algorithms for Dual Wang tile packings with odd and even colors.

Wang Tile Equivalence. It is possible to assemble the Dual Wang tiles into a regular Wang tile set because the underlying tiling is the same. Each texture in each of underlying Wang tiles is determined by the colors of the four edges, as well as the two colors of incident edges (not part of the Wang tile) at each corner. This implies that each texture is determined by 12 edges, and thus the equivalent number of unique Wang tiles textures required equals $C^{12} = (C^3)^4$, or a Wang tiling consisting of C^3 colors. While the number of unique tiles is huge compared to regular Wang tiles (e.g., for $C = 3$, regular Wang tiles consists of 81 textures and dual Wang tiles are equivalent to 531,441 regular Wang tile textures), the resulting Wang tiles are a combination of a smaller set of basis textures.

Limitations. Not all exemplar images are suitable for tile generation. Images with salient features larger than the tile size fail to produce satisfactory results. We address this issue by simply downsampling the image such that the salient features are smaller than the tile size. Furthermore, images with perspective distortions (e.g., finite vanishing points) or images with brightness gradients typically result in noticeable repetitive patterns. However, prior graph-cut based methods also struggle with such textures. In general, our context-aware tile generation effectively 'paints' around localized challenges, but it cannot correct global deviations. When starting from a prompt, diffusion is not aware of this assumption (unless carefully specified in the prompt), hence we apply (unconditional) noise-rolling to impose the assumptions during exemplar synthesis. Note we did not apply noise-rolling as a preprocessing step on the SeamlessGAN dataset used for evaluation in which many exemplar violate the above assumptions, hence the SeamlessGAN results can be seen as a worst case scenario; applying noise-rolling (or any other stationarization method) as a preprocessing step could further improve performance for suboptimal exemplars.

8 CONCLUSION

In this paper we presented an easy to implement, yet flexible, method for generating tileable textures using inpainting. Unlike prior work, we do not reuse pixels or patches from the exemplar, but instead use them to impose exterior boundary conditions on the tile textures. Furthermore, our tile generation method can easily accommodate tiling schemes beyond self-tiling. By appropriate selection of the template patches, we ensure that the boundaries match seamlessly with corresponding boundaries on matching tiles. We demonstrated our method on a variety of tiling schemes, including a novel Dual Wang tile scheme that provides greater tile diversity than prior Wang tile schemes without incurring an additional storage cost. For future work, we would like to speed up and improve the automated methods for selecting the template patches and defective tile rejection. Furthermore, we would like to explore other tiling schemes.

ACKNOWLEDGMENTS

This research was supported in part by NSF grant IIS-1909028.

REFERENCES

Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. 2009. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.* 28, 3 (2009), 24.

Urs Bergmann, Nikolay Jetchev, and Roland Vollgraf. 2017. Learning texture manifolds with the Periodic Spatial GAN. In *ICML*. 469–477.

Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. 2023. Fantasia3D: Disentangling Geometry and Appearance for High-quality Text-to-3D Content Creation. In *ICCV*.

Michael F Cohen, Jonathan Shade, Stefan Hiller, and Oliver Deussen. 2003. Wang tiles for image and texture generation. *ACM Trans. Graph.* 22, 3 (2003), 287–294.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *NeurIPS* 34 (2021), 8780–8794.

Alexei A Efros. 2001. Image Quilting for Texture Synthesis and Transfer. In *Proc. Siggraph 2001*. 341–346.

Alexei A Efros and Thomas K Leung. 1999. Texture synthesis by non-parametric sampling. In *CVPR*, Vol. 2. 1033–1038.

Anna Frühstück, Ibraheem Alhashim, and Peter Wonka. 2019. Tilegan: synthesis of large-scale non-homogeneous textures. *ACM Trans. Graph.* 38, 4 (2019), 1–11.

Chi-Wing Fu and Man-Kang Leung. 2005. Texture Tiling on Arbitrary Topological Surfaces using Wang Tiles.. In *Rendering Techniques*. 99–104.

Leon Gatys, Alexander S Ecker, and Matthias Bethge. 2015. Texture synthesis using convolutional neural networks. *NeurIPS* 28 (2015).

Shouchang Guo, Valentin Deschaintre, Douglas Noll, and Arthur Roullier. 2022. U-attention to textures: hierarchical hourglass vision transformer for universal texture synthesis. In *Proc. of the 19th ACM SIGGRAPH European Conference on Visual Media Production*. 1–10.

Cusuh Ham, James Hays, Jingwan Lu, Krishna Kumar Singh, Zhifei Zhang, and Tobias Hinz. 2023. Modulating Pretrained Diffusion Models for Multimodal Image Synthesis. In *ACM SIGGRAPH 2023 Conference Proceedings*. Article 35, 11 pages.

David J Heeger and James R Bergen. 1995. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. 229–238.

Eric Heitz, Kenneth Vanhoey, Thomas Chambon, and Laurent Belcour. 2021. A sliced wasserstein loss for neural texture synthesis. In *CVPR*. 9412–9420.

Philipp Henzler, Valentin Deschaintre, Niloy J Mitra, and Tobias Ritschel. 2021. Generative modelling of BRDF textures from flash images. *ACM Trans. Graph.* 40, 6 (2021), 1–13.

Philipp Henzler, Niloy J Mitra, and Tobias Ritschel. 2020. Learning a neural 3d texture space from 2d exemplars. In *CVPR*. 8356–8364.

Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. CLIPScore: A Reference-free Evaluation Metric for Image Captioning. In *EMNLP*.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the Design Space of Diffusion-Based Generative Models. In *NeurIPS*.

Bahjat Kawar, Shiran Zada, Oran Lang, Omer Tov, Huiwen Chang, Tali Dekel, Inbar Mosseri, and Michal Irani. 2023. Imagic: Text-based real image editing with diffusion models. In *CVPR*. 6007–6017.

Gwanghyun Kim, Taesung Kwon, and Jong Chul Ye. 2022. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *CVPR*. 2426–2435.

Martin Kolář, Alan Chalmers, and Kurt Debattista. 2016. Repeatable texture sampling with interchangeable patches. *The Visual Computer* 32 (2016), 1263–1272.

Johannes Kopf, Daniel Cohen-Or, Oliver Deussen, and Dani Lischinski. 2006. Recursive Wang tiles for real-time blue noise. Vol. 25. 509–518.

Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. 2005. Texture optimization for example-based synthesis. In *ACM SIGGRAPH 2005 Papers*. 795–802.

Vivek Kwatra, Arno Schödl, Irfan Essa, Greg Turk, and Aaron Bobick. 2003. Graphcut textures: Image and video synthesis using graph cuts. *ACM Trans. Graph.* 22, 3 (2003), 277–286.

Ares Lagae and Philip Dutré. 2005. A procedural object distribution function. *ACM Trans. Graph.* 24, 4 (2005), 1442–1461.

Chuan Li and Michael Wand. 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. In *ECCV*. 702–716.

Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. 2017. Diversified texture synthesis with feed-forward networks. In *CVPR*. 3920–3928.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. 2023. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*. 9298–9309.

Xiaokang Liu, Chenran Li, Lin Lu, Oliver Deussen, and Changhe Tu. 2022. Fabricable Multi-Scale Wang Tiles. In *Comp. Graph. Forum*, Vol. 41. 149–159.

Xihui Liu, Zhe Lin, Jianming Zhang, Handong Zhao, Quan Tran, Xiaogang Wang, and Hongsheng Li. 2020. Open-edit: Open-domain image manipulation with open-vocabulary instructions. In *ECCV*. 89–106.

Morteza Mardani, Guilin Liu, Aysegul Dundar, Shiqiu Liu, Andrew Tao, and Bryan Catanzaro. 2020. Neural ffts for universal texture image synthesis. *NeurIPS* 33 (2020), 14081–14092.

Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2023. Null-text inversion for editing real images using guided diffusion models. In *CVPR*. 6038–6047.

Joep Moritz, Stuart James, Tom S. F. Haines, Tobias Ritschel, and Tim Weyrich. 2017. Texture Stationarization: Turning Photos Into Tileable Textures. *Comp. Graph. Forum* 36, 2 (2017), 177–188.

T-Y Ng, T-S Tan, and Xinyu Zhang. 2005. Generating ω-tile set for texture synthesis. In *IEEE Int. Comp. Graph.* 177–184.

Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. 2022. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models. In *ICML*. 16784–16804.

Yaniv Nikankin, Niv Haim, and Michal Irani. 2023. SinFusion: training diffusion models on a single image or video. In *ICML*. 26199–26214.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS* 32 (2019).

Ken Perlin. 1985. An image synthesizer. *ACM Siggraph Computer Graphics* 19, 3 (1985), 287–296.

Tiziano Portenier, Siavash Arjomand Bigdeli, and Orcun Goksel. 2020. GramGAN: Deep 3d texture synthesis from 2d exemplars. *NeurIPS* 33 (2020), 6994–7004.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical Text-Conditional Image Generation with CLIP Latents. *arXiv preprint arXiv:2204.06125* (2022).

Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. 2023. Texture: Text-guided texturing of 3d shapes. *arXiv preprint arXiv:2302.01721* (2023).

Carlos Rodriguez-Pardo, Dan Casas, Elena Garces, and Jorge Lopez-Moreno. 2024. TexTile: A Differentiable Metric for Texture Tileability. In *CVPR*.

Carlos Rodriguez-Pardo and Elena Garces. 2022. Seamlessgan: Self-supervised synthesis of tileable texture maps. *IEEE Trans. Vis. and Comp. Graph.* (2022).

Carlos Rodriguez-Pardo, Sergio Suja, David Pascual, Jorge Lopez-Moreno, and Elena Garces. 2019. Automatic extraction and synthesis of regular repeatable patterns. *Computers & Graphics* 83 (2019), 33–41.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis With Latent Diffusion Models. In *CVPR*. 10684–10695.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS* 35 (2022), 36479–36494.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. *NeurIPS* 29 (2016).

Sam Sartor and Pieter Peers. 2023. MatFusion: A Generative Diffusion Model for SVBRDF Capture. In *SIGGRAPH Asia 2023 Conference Papers*. 1–10.

Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. 2019. Singan: Learning a generative model from a single natural image. In *CVPR*. 4570–4580.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. 2021. Score-Based Generative Modeling through Stochastic Differential Equations. In *ICLR*.

Stability AI. 2022a. Stable Diffusion V2 - Inpainting. https://huggingface.co/stabilityai/stable-diffusion-2-inpainting.

Stability AI. 2022b. Stable Diffusion XL. https://huggingface.co/docs/diffusers/using-diffusers/sdxl.

Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. 2023. Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation. In *CVPR*. 1921–1930.

Dmitry Ulyanov, Vadim Lebedev, Andrea Vedaldi, and Victor Lempitsky. 2016. Texture networks: feed-forward synthesis of textures and stylized images. In *ICML*. 1349–1357.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. 2017. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVRP*. 6924–6932.

Kenneth Vanhoey, Basile Sauvage, Frédéric Larue, and Jean-Michel Dischler. 2013. On-the-fly multi-scale infinite texturing from example. *ACM Trans. Graph.* 32, 6 (2013).

Giuseppe Vecchio, Rosalie Martin, Arthur Roullier, Adrien Kaiser, Romain Rouffet, Valentin Deschaintre, and Tamy Boubekeur. 2023. ControlMat: A Controlled Generative Approach to Material Capture. *arXiv preprint arXiv:2309.01700* (2023).

Andrey Voynov, Kfir Aberman, and Daniel Cohen-Or. 2023. Sketch-Guided Text-to-Image Diffusion Models. In *ACM SIGGRAPH 2023 Conference Proceedings*. Article 55, 11 pages.

Adéla Šubrtová, Michal Lukáč, Jan Čech, David Futschik, Eli Shechtman, and Daniel Sýkora. 2023. Diffusion Image Analogies. In *ACM SIGGRAPH 2023 Conference Proceedings*. Article 79, 10 pages.

Hao Wang. 1961. Proving theorems by pattern recognition—II. *Bell system technical journal* 40, 1 (1961), 1–41.

Jianyi Wang, Kelvin C.K. Chan, and Chen Change Loy. 2023. Exploring CLIP for assessing the look and feel of images. In *AAAI*. Article 284, 9 pages.

Li-Yi Wei. 2004. Tile-based texture mapping on graphics hardware. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*. 55–63.

Jianfeng Xiang, Jiaolong Yang, Binbin Huang, and Xin Tong. 2023. 3D-aware Image Generation using 2D Diffusion Models. *arXiv preprint arXiv:2303.17905* (2023).

Xudong Xu, Zhaoyang Lyu, Xingang Pan, and Bo Dai. 2023. Matlaber: Material-aware text-to-3d via latent brdf auto-encoder. *arXiv preprint arXiv:2308.09278* (2023).

Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. 2023. IP-Adapter: Text Compatible Image Prompt Adapter for Text-to-Image Diffusion Models. *arXiv preprint arXiv:2308.06721* (2023).

Ning Yu, Connelly Barnes, Eli Shechtman, Sohrab Amirghodsi, and Michal Lukac. 2019. Texture mixer: A network for controllable synthesis and interpolation of texture. In *CVPR*. 12164–12173.

Xianfang Zeng, Xin Chen, Zhongqi Qi, Wen Liu, Zibo Zhao, Zhibin Wang, Bin Fu, Yong Liu, and Gang Yu. 2023. Paint3D: Paint Anything 3D with Lighting-Less Texture Diffusion Models. *arXiv preprint arXiv:2312.13913* (2023).

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. In *CVPR*. 3836–3847.

Xilong Zhou, Milos Hasan, Valentin Deschaintre, Paul Guerrero, Kalyan Sunkavalli, and Nima Khademi Kalantari. 2022. TileGen: Tileable, Controllable Material Generation and Capture. In *SIGGRAPH Asia 2022 Conference Papers*. Article 34, 9 pages.

Yang Zhou, Kaijian Chen, Rongjun Xiao, and Hui Huang. 2023. Neural Texture Synthesis With Guided Correspondence. In *CVPR*. 18095–18104.

Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. 2018. Non-stationary texture synthesis by adversarial expansion. *ACM Trans. Graph.* 37, 4 (2018), 1–13.

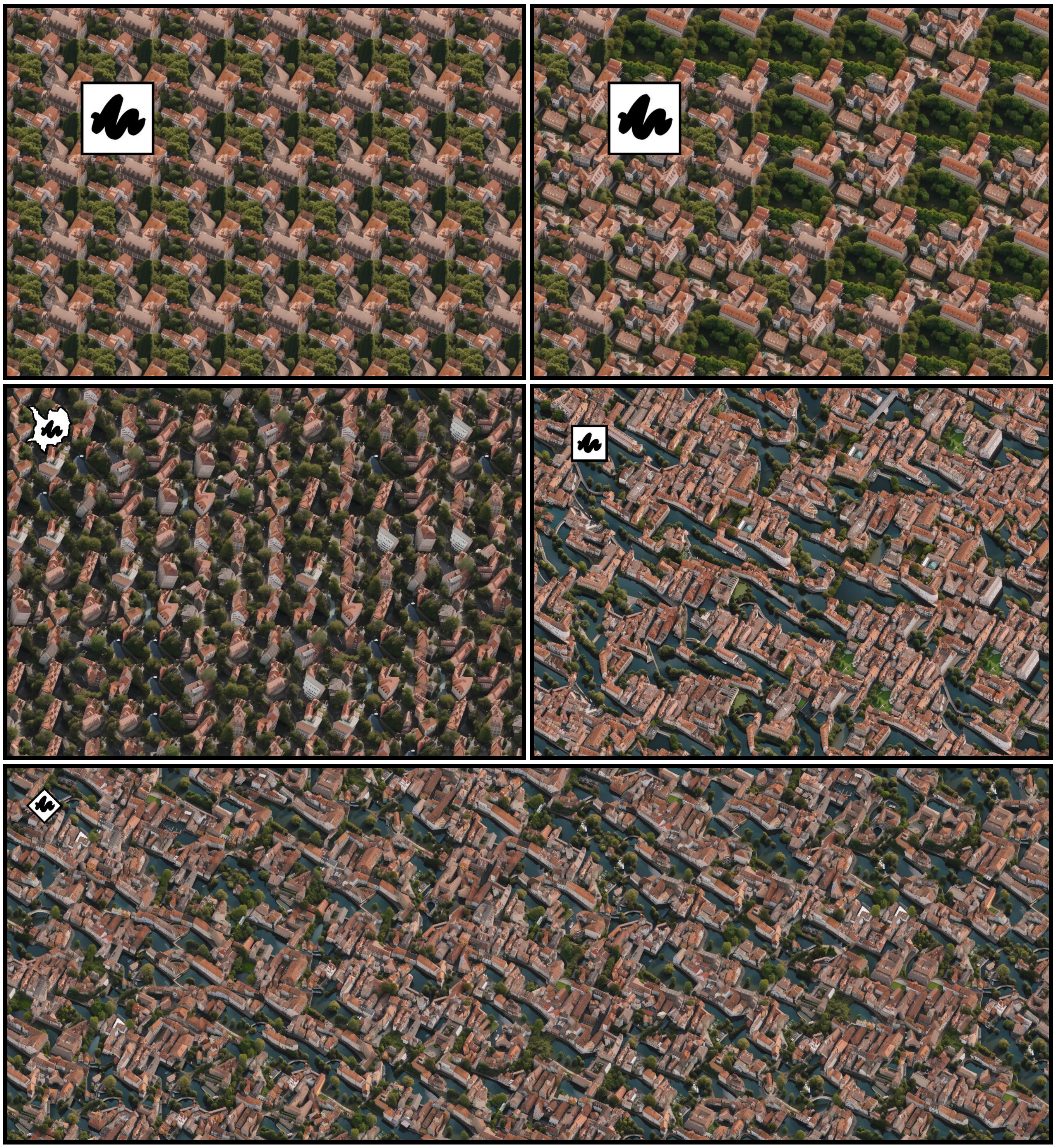


Fig. 8. **Comparison of Tiling Schemes:** A comparison of the different tiling schemes (first row: self-tiling and stochastic self-tiling (with 4 tiles); second row: stochastic Escher self-tiling (with 4 tiles) and regular 3-color Wang tiling (81 tiles); last row: 3-color Dual Wang tiling) demonstrated on texture tiles generated with the prompt "*European city blocks, tile roofs, streams, drone footage*". For each example we also show the tile shape and size with the scribbled overlay.

.

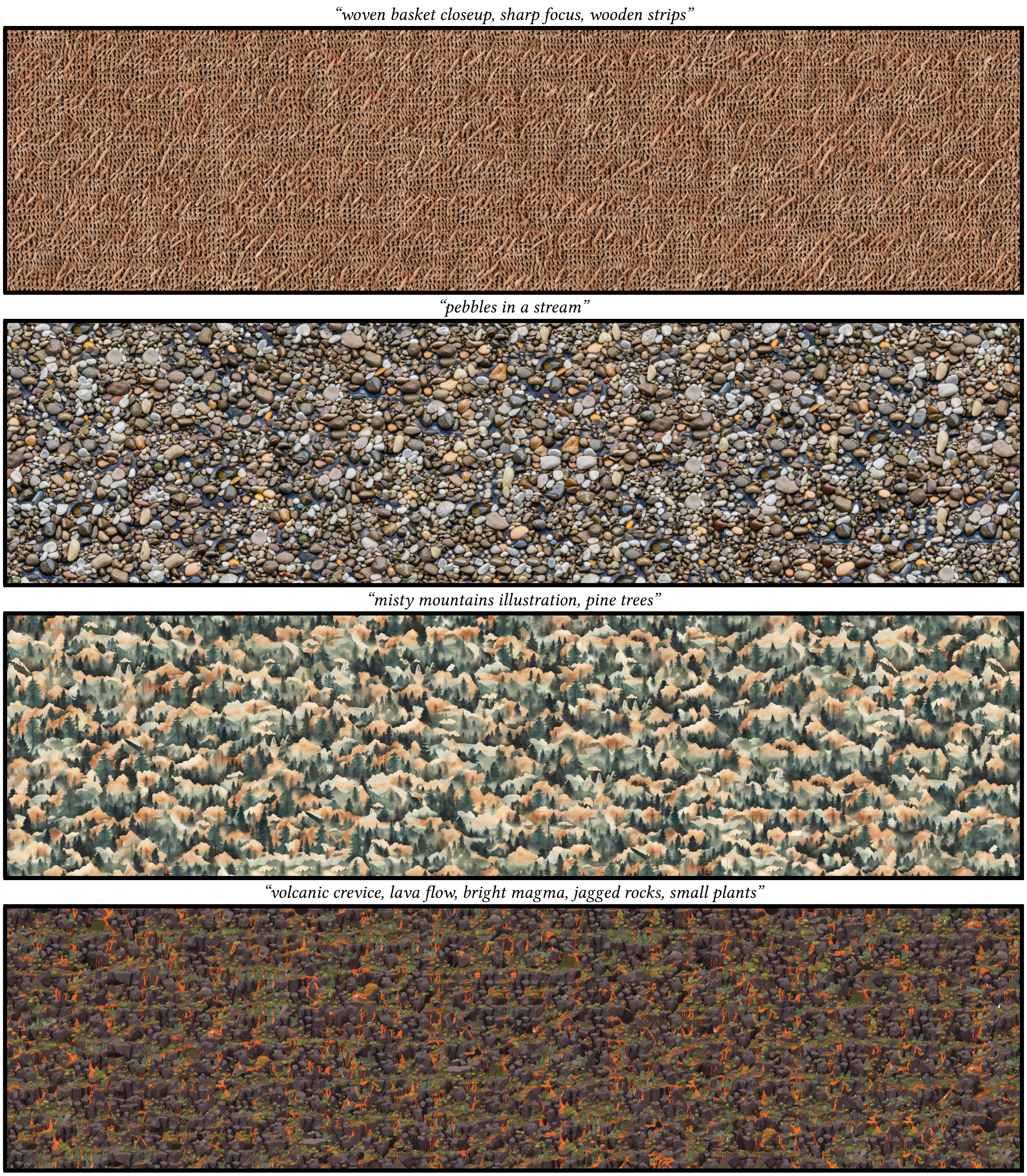

"woven basket closeup, sharp focus, wooden strips"
"pebbles in a stream"
"misty mountains illustration, pine trees"
"volcanic crevice, lava flow, bright magma, jagged rocks, small plants"

Fig. 9. **Dual Wang Tile Results:** High resolution (7168 × 2560) Dual Wang tiled textures generated from a text-prompt (listed above each example). Please zoom-in on the tiled textures to fully appreciate the generated texture detail.