# TexTile: A Differentiable Metric for Texture Tileability

Carlos Rodriguez-Pardo[1] Dan Casas[1] Elena Garces[1,2] Jorge Lopez-Moreno[1,2]
[1]Universidad Rey Juan Carlos, Spain [2]SEDDI, Spain

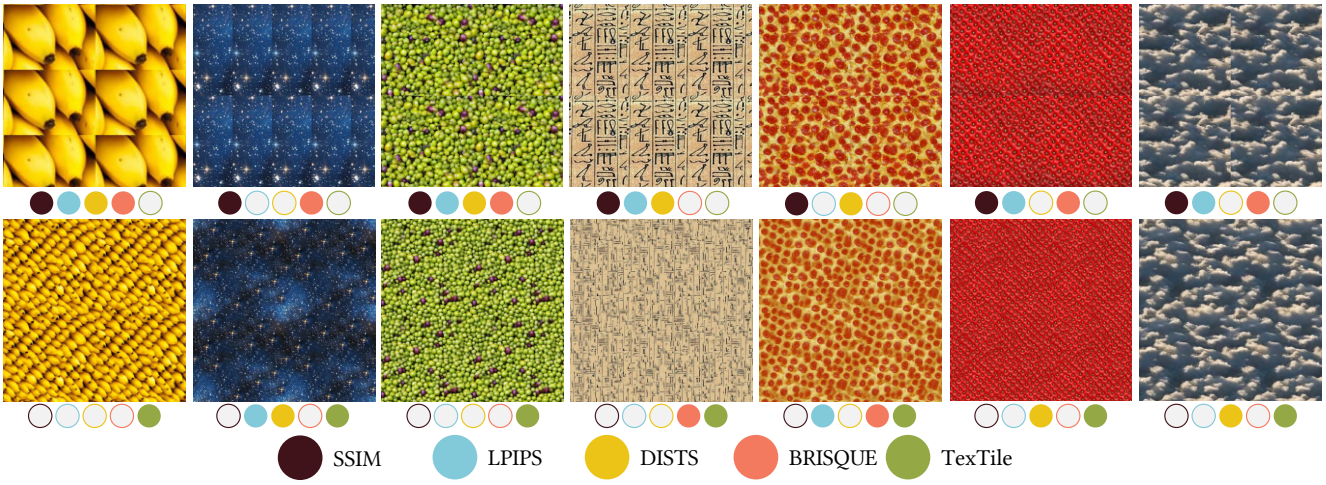


Figure 1. Existing perceptual metrics, commonly used to evaluate texture synthesis algorithms, typically fail to account for tileability. Such weakness is depicted in this figure where, for each column, we show tiled versions of textures with (top) and without (bottom) tiling artifacts. For each column, we highlight using saturated color dots the preferred image (*i.e.*, higher score) according to different metrics. It can be seen that there is no correlation across existing methods (*i.e.*, saturated dots distributed over top and bottom rows), while our method TexTile consistently prefers seamless tiled textures (*i.e.,* saturated green dots on the bottom for all columns).

## Abstract

*We introduce TexTile, a novel differentiable metric to quantify the degree upon which a texture image can be concatenated with itself without introducing repeating artifacts (i.e., the tileability). Existing methods for tileable texture synthesis focus on general texture quality, but lack explicit analysis of the intrinsic repeatability properties of a texture. In contrast, our TexTile metric effectively evaluates the tileable properties of a texture, opening the door to more informed synthesis and analysis of tileable textures. Under the hood, TexTile is formulated as a binary classifier carefully built from a large dataset of textures of different styles, semantics, regularities, and human annotations. Key to our method is a set of architectural modifications to baseline pre-train image classifiers to overcome their shortcomings at measuring tileability, along with a custom data augmentation and training regime aimed at increasing robustness and accuracy. We demonstrate that TexTile can be plugged into different state-of-the-art texture synthesis methods, including diffusion-based strategies, and generate tileable textures while keeping or even improving the overall texture quality. Furthermore, we show that TexTile can objectively evaluate any tileable texture synthesis method, whereas the current mix of existing metrics produces uncorrelated scores which heavily hinders progress in the field.*

## 1. Introduction

The appearance of 3D digital objects plays a fundamental role in the overall realism of a virtual environment. To create realistic textures, many strategies have been widely explored, including procedural algorithms [19, 28], scanning [22, 45] and, more recently, text-to-image generative pipelines [7–9]. Among the different properties that we wish for the synthesized textures (*e.g.*, photorealism, variety in detail, high resolution), the ability to seamlessly repeat or tile itself without noticeable artifacts —its tileability— is especially important in the frequent case of applying a texture to a large surface. For example, when texturing the facade of a building or a field of grass.

Many methods exist that focus on the specific case of tileable texture synthesis [1, 11, 36, 45, 48, 53, 54]. This has been achieved, for example, by manipulating image borders [36], maximizing stationary image properties [45], or conditioning generative models on structured patterns [74]. How-

ever, despite such significantly diverse methodologies used in existing methods, they typically rely on evaluation using common metrics based on general texture quality which, unfortunately, do not explicitly account for the intrinsic repeatability properties of a texture.

To address this shortcoming, we introduce TexTile, a novel metric for texture tileability. TexTile is a data-driven metric that brings two key novel functionalities into texture synthesis methods: first, it computes a human-friendly score that captures the intrinsic repeatability of any texture; and second, it provides a differentiable metric that can be used as an additional data term in any learning-based or test-time optimization method for tileable texture synthesis. We demonstrate that TexTile enables a factual analysis of state-of-the-art methods for tileable texture synthesis, while previous metrics often result in uncorrelated evaluations (*i.e.*, a good tileable texture might have low SSIM [66] score, or a poor tileable texture might have a high SSIM), as illustrated in Figure 1. Furthermore, we demonstrate that TexTile can be used off-the-shelf as an additional loss term in state-of-the-art methods for texture synthesis, including diffusion-based models, to output tileable textures while preserving or even improving the overall image quality.

Under the hood, we formulate TexTile as a binary classifier built using a carefully designed architecture with an attention-enhanced convolutional network. The convolutional filters can detect local discontinuities –which are common in borders that are not seamlessly tileable– and can deal with images of arbitrary sizes, but they struggle with global understanding to detect artifacts and repeating patterns. Therefore, we introduce Self-Attention layers into our architecture, which is known to capture a global understanding of the input. This, combined with a custom data augmentation policy designed for tileability detection, enables us to train a novel neural classifier to unleash a new functionality for the state-of-the-art texture synthesis methods.

In summary, we introduce the following contributions:

- A novel learning-based metric for texture analysis that accurately quantifies tileability.
- An attention-enhanced convolutional classifier, and a training configuration aimed at maximizing robustness and accuracy.
- A differentiable loss function which can be plugged into texture synthesis algorithms to generate tileable textures.
- Open-source code and trained weights for our metric. We believe this will open the door to quantitative benchmarks on tileable textures, which is currently not possible due to the lack of a specific metric for such task.

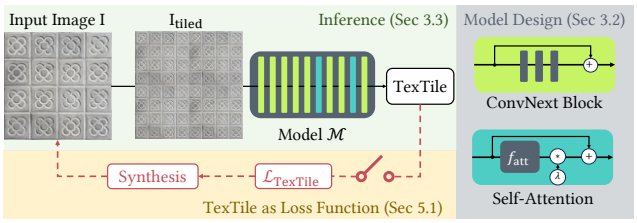


Figure 2. Our model takes as input a texture image I, which we tile to form $I_{tiled}$, and returns an estimation of its tileability. This metric can be used as a loss function $\mathcal{L}_{TexTile}$ to allow synthesis algorithms to generate tileable textures. Our model, $\mathcal{M}$ architecture is comprised of ConvNext [39] and residual self-attention blocks.

## 2. Related Work

### 2.1. Image Quality Assessment

Image Quality Assessment (IQA) algorithms can be categorized into three different groups. **Reference-based** IQA methods compare an input and a reference image, which is the most widely studied strategy for IQA. These methods have traditionally leveraged pixel-wise differences (*e.g.,* PSNR, $\ell_1$ or $\ell_2$ distances) or image statistics (*e.g.*, SSIM [66] or FSIM [69]) to compute the similarity between the images. Neural reference-based IQA, which leverage the stronger correlation of deep neural networks with human perception [71], have been also proposed. These strategies either leverage features from untrained [3] or pre-trained convolutional neural networks [20], or use direct supervision from human judgments, as in LPIPS [71], PIE-APP [51], DISTS [14], Si-FID [56], or DreamSIM [18]. These methods introduce powerful and differentiable metrics, however, they require a reference image and are thus not suitable for measuring global properties, like tileability.

Instead of comparing an input and a reference image, **distribution-based** IQA methods compare statistics of two sets of reference images and generated images. These methods are commonly used to evaluate the perceptual quality of generative models, typically using metrics based on neural networks activations [6, 27, 57, 67], nearest neighbors [33], spectral [61], or geometric distances [31]. While these methods are useful for evaluating the performance of generative models, they struggle as loss functions [57], and also require reference images.

Finally, **no-reference** IQA methods compute the overall quality score of an input image without requiring an explicit reference. These methods rely on image statistics, as in BIQI [44] amd BRISQUE [43]; or on training deep neural networks on human judgments of image quality, like HyperIQA [59], MANIQA [68], VCRNet [49], or CLIP-IQA [64]. Despite competitive results in image quality assessment, these methods do not incorporate tileabilty analysis. SeamlessGAN [53] leverages the discriminator of a single-image generative model to find artifacts in the borders of generated textures. However, the discriminator only

measures seamlessness, ignoring other factors that influence tileability and, most importantly, it cannot generalize to any image outside of its single-image dataset.

Our metric is most closely related to no-reference IQA methods, as it takes a single image as input. However, in contrast to existing methods, we assess the image quality based on tilebility instead of general image quality. We demonstrate that, when combined with existing IQA metrics, our metric successfully captures overall quality *and* tileability. To the best of our knowledge, our metric is the first no-reference tileability metric.

### 2.2. Tileable Texture Synthesis

**Non-parametric** tileable texture synthesis methods generate new tileable images by maximizing image *stationarity* [45], by manipulating the images with border transformations using Graphcuts [36], or by patch-based synthesis with histogram-preserving blending [11]. **Parametric** alternatives typically leverage deep neural networks in diverse ways. Rodriguez-Pardo *et al*. [54] look for repeating patterns in images using deep features in pretrained CNNs, then synthesize tileable images by blending the borders. Tileability can be also achieved with specific image parameterizations and neural network design, as in Neural Cellular Automata [48], or in the Periodic Spatial GAN [5]. By manipulating latent spaces in pre-trained GANs, SeamlessGAN [53] achieve tileability without specific model modifications. Tileable image generation has also been explored with the goal of material capture, by means of models conditioned on structured patterns [74], or through *rolled diffusion* [62]. Specialized methods have been proposed for tileable vector image generation [1]. For a review on texture synthesis, we refer the reader to the survey in [2].

These methods typically provide quantitative evaluation using reference-based IQA. As mentioned in Section 2.1, these metrics measure the perceptual similarity between generated and input images, however, they do not account for tiling. To the best of our knowledge, there is no available metric that can be used to compare these methods in terms of tileability, hindering the progress of the field and limiting evaluation to qualitative analyses. Our metric aims to solve this gap. Being fully differentiable, it can be leveraged as a loss function to enable existing synthesis algorithms to produce seamlessly tileable outputs.

## 3. TexTile

### 3.1. Introduction

Our goal is to develop a differentiable no-reference image metric that measures texture tileability, that is, a single-image metric that does not require a second image for comparison purposes. To achieve this, we leverage a convolutional neural network as our differentiable function, which


Figure 3. From a tileable texture (left), our data augmentation can generate tileable (top row) and non-tileable (bottom) variations.

we train on a dataset of textures on a binary classification task. Our model learns to classify between tileable and non-tileable textures, by means of a comprehensive data augmentation policy and custom architecture design choices. We explain our model design choices in Sec. 3.2, validate them using ablation studies in Sec. 4.1, and explain the model predictions in Sec. 4.3. We show examples of results of TexTile as a loss function (Sec. 5.1), as a means of benchmarking image synthesis algorithms (Sec. 5.2), and applications for alignment and repeating pattern detection (Sec. 5.3).

### 3.2. Model Design and Training

#### Network Design

A model that can measure texture tileability should have at least three properties. First, it must be able to detect local discontinuities, which happen when borders are not seamlessly tileable. Second, the model should be able to handle images of any dimension or aspect ratios. Finally, it should have a global understanding of the image, in order to detect artifacts and repeating patterns. The first two properties can be achieved by fully-convolutional architectures, which are strongly biased towards textures [26]. However, problems that require global understanding of images are typically tackled using attention-based Vision Transformers [23], which are more biased towards shape [46] and are less flexible in terms of input dimensionality.

We propose an architecture that can benefit from the properties of both convolutional and attention-based models. Because of the limited size of our training dataset, we also want to leverage ImageNet pretraining [12]. We thus use a state-of-the-art pretrained ConvNext [39] fully-convolutional model. We further introduce Linear Self-Attention modules [65] to allow it to learn global patterns in the images, while keeping a limited computational cost. We design them as residual layers $x \leftarrow x + \lambda f_{\text{att}}(x)$, multiplied by a learnable parameter $\lambda$, which we initialize at $1e-6$. We illustrate this module in Figure 2. This way, we can modify the internal model architecture without disrupting the performance of the pre-trained backbone during early training iterations. We only use two of such modules, which we place on the deeper layers of the ConvNext model. Previous work on texture estimation and synthesis also benefit from adding attention to fully-convolutional backbones [21, 55].

**Data Augmentation for Tileable Textures**

We leverage a comprehensive data augmentation policy, to train the model to detect repeating artifacts in images.

Our policy contains several operations which we divide into: **Tileability-preserving** policies, which generate variations of textures without reducing their tileability. These include global color and gamma changes, random flipping (horizontal and vertical), translations, equalization, blurs or noise, or rescaling the images with different scale factors across each dimensions. We also introduce an operation named *UnFold*, which mirrors the input image horizontally and vertically. **Tileability-breaking** policies include rotations, shears, random cropping, or thin-plate spline warping [4]. We apply tileability-preserving policies to tileable examples, and both kinds of operations to non-tileable examples. Finally, our last operation is random tiling, by which we tile the textures a random number of times (1-5), followed by a random rescaling, and random cropping. With this policy, we allow the model to detect tileability regardless on the number of repetitions in the input images.

We introduce two additional data augmentation policies. With $AUG_{T\to F}$, we generate non-tileable textures from tileable textures. We do this by applying a tileability-breaking operation to a tileable texture. With $AUG_{F\to F'}$, we create repetition-free texture examples by applying every data augmentation operation except random tiling to non-tileable examples, and assign these textures a positive tileability label. These two policies are needed to reduce the distribution shift between tileable and non-tileable training datasets, as they come from different sources and their semantics, feature scales, and contents may differ. With $AUG_{T\to F}$ and $AUG_{F\to F'}$, we force the model to learn patterns exclusively related to tileability, ignoring other texture properties. We illustrate some of these policies in Figure 3, including example of *UnFolding* on the top right image.

### 3.3. Model Inference

We illustrate the inference process in Figure 2. We tile the input image I once across each spatial dimension $\mathrm{I}_{\text{tiled}} \leftarrow tile(\mathrm{I}, (2, 2))$. We observe that this limited number of repetitions is enough to accurately detect tileability. We use this process both in texture evaluation and when we use TexTile as loss function. $\mathrm{I}_{\text{tiled}}$ is fed to our model $\mathcal{M}$, which outputs an unbounded prediction, which can be transformed into the desired $(0, 1)$ range with a Logistic function. However, doing this typically leads to predictions very close to 0 or 1, making our metric less useful for comparing different textures. We mitigate this issue by introducing $\lambda$, which controls how far the predicted values are from the boundaries:

$$\text{TexTile} = \frac{1}{1 + \exp\left(-\lambda \cdot \mathcal{M}(\mathrm{I}_{\text{tiled}})\right)} \tag{1}$$

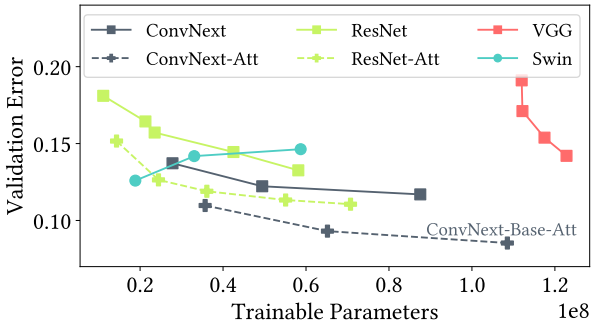


Figure 4. Influence on the neural architecture type and size on its quantitative performance (Cross-entropy error on the validation dataset). Convolutional architectures are marked with ■, attention-based models with ●, and our versions of convolutional networks with embedded attention with ✚.

We set $\lambda = 0.25$, which preserves discrimination between clearly tileable and non-tileable samples while ambiguous cases sit closer to 0.5. This is a strictly monotonic function, so relative tileability orders are maintained.

### 3.4. Dataset

Our goal is to make our model robust to textures of any semantics, regularity, stochasticity, and homogeneity. We also want our model to work with both natural and synthetic texture maps. To achieve this, we collect a dataset of tileable and non-tileable textures from a variety of sources.

We gather a novel dataset of 4276 tileable textures, comprised of high-resolution photographs of materials, turned into tileable textures by artist labor. We extend this data with publicly-available tileable textures, including 285 images from [63], 186 *CC-BY* images from *Julio Sillet*, including both albedo and surface normal maps, and 290 royalty-free textures from *ManyTextures*. These images, being curated by human experts, ensure the *tileability* property.

We also create an additional dataset of non-tileable textures, for which examples are comparably easier to obtain. To this end, we gather 3922 synthetic high-quality from [13], 1187 photographs of a wide variety of texture types from [10], 506 facade photographs from [29], 7265 images taken under different illumination conditions from [72], 760 from [73], and 93 from [45]. We extend this dataset with 8675 images from [16, 32, 34, 41], and 86 non-stationary textures from [75]. While we can create negative examples from tileable textures using $AUG_{T\to F}$, this dataset is important for generalization. We use 504 tileable images and 504 non-tileable examples for validation, and the rest is left for training. We have approximately 5 times more non-tileable examples than tileable images, which may hinder the task of learning an unbiased classifier. We overcome this with our training and data augmentation policies.

In Section 4 we evaluate the performance of our model, measuring average binary cross-entropy error on our test set,

| | Configuration | Error ↓ | Accuracy ↑ | $F_1$ ↑ | AUC ↑ |
|---|---|---|---|---|---|
| Training | W/out Pretraining | 0.459 | 0.808 | 0.804 | 0.886 |
| | W/out NAdam [15] | 0.096 | 0.965 | 0.966 | 0.992 |
| | W/out Look-Ahead [70] | 0.082 | 0.971 | 0.971 | 0.992 |
| Data Augmentation | W/out Negative Samples | 0.319 | 0.905 | 0.909 | 0.943 |
| | W/out Color Aug. | 0.119 | 0.956 | 0.957 | 0.992 |
| | W/out Rescales | 0.105 | 0.960 | 0.968 | 0.990 |
| | W/out Flips | 0.087 | 0.972 | 0.972 | 0.993 |
| | W/out Distortions | 0.094 | 0.966 | 0.966 | 0.987 |
| | W/out $AUG_{T\to F}$ | 0.121 | 0.960 | 0.961 | 0.990 |
| | W/out $AUG_{F\to F'}$ | 0.087 | 0.969 | 0.966 | 0.991 |
| | W/out UnFold | 0.082 | 0.970 | 0.971 | 0.992 |
| | **Final Model** | **0.064** | **0.982** | **0.983** | **0.997** |

Table 1. Ablation study on different configurations of model training configurations and data augmentation policies.

as well as classification metrics, like accuracy, $F_1$-Score and Area Under the Curve (AUC). We provide more details of our datasets in the supplementary material.

### 3.5. Implementation Details

We train our models for 100 epochs using NAdam [15] Lookahead [70], Automatic Gradient Scaling and Mixed Precision Training [42], with an initial learning rate of 0.002, halved every 33 epochs. We use PyTorch [50] for training, and Kornia [52] for data augmentation. We use batch sizes of 24 samples, composed of balanced number of positive and negative examples. This process takes approximately 6 hours on a single Nvidia RTX 3060 GPU. We use images of $(384, 384)$ pixels for training and $(512, 512)$ for inference. Hyperparameters are tuned using Bayesian optimization on a validation dataset containing 1000 images.

## 4. Evaluation

### 4.1. Ablation Study

**Network Architecture Design** In Figure 4, we show the impact of the neural network architecture design on generalization performance. We evaluate different fully-convolutional backbones, including ResNet [24], VGG [58] and ConvNext [39]; as well as transformer-based Swin V2 [38], on different model sizes. We also show the results of our variations of the fully-convolutional backbones, where we introduce linear Self-Attention [65] modules in the last layers of the models. Every network is pre-trained on ImageNet [12], then fine tuned in our task, and evaluated on a validation dataset. As shown, larger networks typically perform better, with the exception of Swin, which interestingly benefits from fewer parameters. Further, ConvNexts strongly outperform ResNets and VGGs. By introducing self-attention into these fully-convolutional models, we significantly improve their generalization capabilities. In the rest of our experiments, we will use our custom *ConvNext-Base-Att* model, as it achieves the lowest error overall.

| Distance | Error ↓ | Accuracy ↑ | $F_1$ ↑ | AUC ↑ |
|---|---|---|---|---|
| FID [27] | 0.568 | 0.703 | 0.699 | 0.764 |
| GS [31] | 0.694 | 0.507 | 0.533 | 0.510 |
| MSID [61] | 0.797 | 0.582 | 0.544 | 0.503 |
| **TexTile** | **0.064** | **0.982** | **0.983** | **0.997** |

Table 2. Comparison of our metric with distribution-based distances, on a downstream classification task.

**Data Augmentation and Training Configuration**

In Table 1, we show an ablation study of our data augmentation and optimization setups. From our final model configuration, we remove different components to its training policy to study their impact on generalization. We report different classification metrics measured on our test set, which contains a balanced number of positive and negative examples.

First, we show that the model strongly benefits from pretraining on ImageNet. The model performance can be enhanced by introducing *Nesterov momentum* [15] into the optimizer, and further with Lookahead training [70]. We observe that these results are consistent across architectures, and that other optimizers, such as RAdam [37] or AdamW [40], performed worse than NAdam or Adam.

Regarding data augmentation, we first tested a model trained without negative examples, relying on synthetic non-tileable textures from tileable ground truths. However, this yielded limited generalization. Traditional augmentations (color, geometry, noise, blurs, elastic transformations) provided incremental improvements. Our custom policies, which generate negative samples from tileable images and vice versa, further enhanced model performance. By introducing random *unfolding*, we achieve small gains. This comprehensive data augmentation policy mitigates model and dataset limitations and makes TexTile more robust to important factors like color variations, sharpness, or scales.

### 4.2. Comparison with Distribution-Based Metrics

In this experiment, we compare our metric with off-the-shelf distribution-based metrics. To achieve this, we compute the latent features of both our tileable and non-tileable training datasets, using 2x2 tilings as in our setup. Then, for randomly selected subsets from our test set, we compare their latent features to those of both training datasets. Finally, we classify the set according to the which of both distributions is closest. We show the results on Table 2, where we find that these metrics fail to capture important characteristics that influence the tileability of textures. Directly supervising for tileability is unsurprisingly more effective. Interestingly, the GS [31] and MSID [61] metrics perform only marginally better than random, whereas FID [27] better captures the differences between the distributions.

### 4.3. Qualitative Evaluation

We leverage Axion-Based Class-Activation Mappings [17] to visualize which features are most relevant to our model.

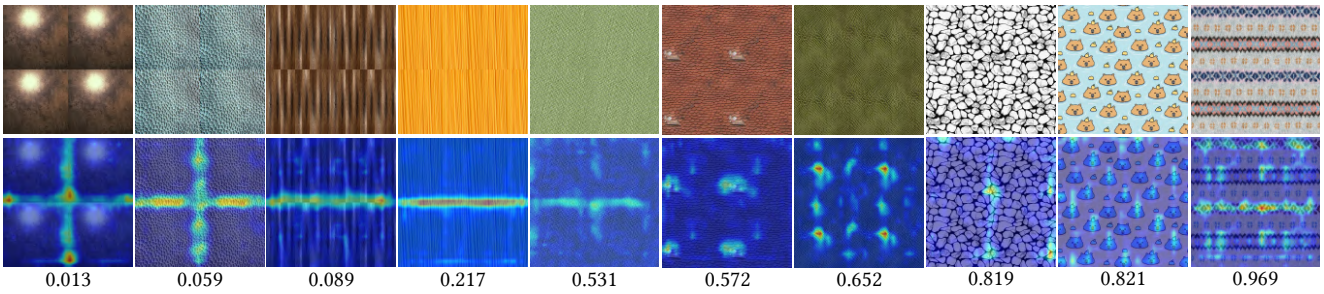


Figure 5. On top, textures samples (tiled $2 \times 2$) with increasing predicted tileability. Below them, model saliency maps and TexTile values.

In Figure 5, we show saliency maps and predicted TexTile values for a few representative textures with increasing degrees of tileability. As shown on the first two examples, our model predicts low tileability values for textures without seamless borders. In the next three examples, which are seamless only on one of their axes, the model outputs higher tileability values. On the last five examples, which are all seamlessly tileable, the model is leveraging other features, like uneven shadings, unusual artifacts, or repeating objects. These results show that our model can exploit patterns other than border discontinuity for its prediction, and that it can integrate distant information for finding repeating elements.

# 5. Results

## 5.1. TexTile as a Loss Function

Because TexTile is a fully-differentiable metric, it can be leveraged as a loss function for synthesizing tileable textures. We explore this on two different types of algorithms.

First, we extend an optimization-based **neural texture synthesis** algorithm [25] to generate tileable textures. To do so, we simply optimize a joint loss function $\mathcal{L} = \lambda_{\text{style}}\mathcal{L}_{\text{style}} + \lambda_{\text{TexTile}}\mathcal{L}_{\text{TexTile}}$, where $\lambda_{\text{style}}$ and $\lambda_{\text{TexTile}}$ control the weight of the each component of the loss and are selected empirically to $\lambda_{\text{style}} = \lambda_{\text{TexTile}} = 1$. We observed little sensitivity to these weightings as long as they are on the same order of magnitude. We show end-to-end synthesis results in Figure 8. We can also generate tileable textures in this fashion by optimizing only the texture borders using outpainting, as we illustrate in Figure 6.

Relatedly, we can also extend **single image diffusion models** so they can generate tileable textures. While the goal of these methods is more general image or video synthesis, we leverage them as powerful texture synthesis algorithms. To do so, we use SinFusion models [47] without any modifications in the training process. During inference, after each diffusion step, we perform a single optimization step to the noisy image on the direction that maximizes TexTile. We show qualitative results in Figure 8.

With these simple modifications, we can transform these methods –and potentially any texture generative model– into tileable texture synthesis algorithms, without significant loss in the perceptual quality of the generated textures.

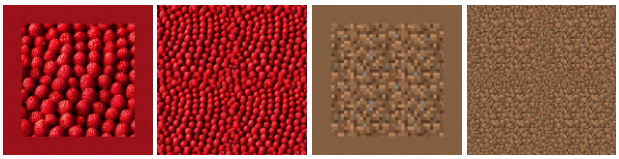

Figure 6. Image outpainting for tileable texture synthesis. On the left, *non-tileable* input images with the area to be outpainted in a solid color; on their sides, outpainted results, obtained by maximizing tileability, shown in a 2x2 tile composition.

Importantly, this can be achieved without re-training the generative models. Further implementation details and results are included in the supplementary material.

## 5.2. Benchmarking Texture Synthesis Algorithms

In Table 3, we show a quantitative comparison between different texture synthesis algorithms and generative models, on the 14-texture dataset used in [53], using reference and no-reference metrics. For [25, 47], we show results of their baseline methods and our modifications that generate tileable textures. The qualitative results on the complete dataset is present in the supplementary material.

Besides, we observe that powerful generative models like [47, 53] do not consistently beat non-parametric alternatives like [11, 36] across either reference or no-reference metrics. In terms of tileability, introducing Textile as a loss function to [25, 47] not only significantly improves the tileability of their outputs but it does it without reducing their perceptual quality. Unsurprisingly, methods that perform border blending [11, 54] achieve lower tileability values across different metrics, than methods that synthesize the textures in a more holistic way, like [5, 45, 48, 53]. These results confirm that there are more constituent factors in texture tileability than simply seamless borders.

A correlation matrix between these metrics is shown in Figure 7. Learned reference metrics (LPIPS, DISTS and PieAPP) correlate strongly between each other but poorly with other measures. Si-FiD and SSIM are not closely related with any other metric, while TexTile is slightly correlated with BRISQUE and CLIP-IQA. As we also illustrate in Figure 1, there is no consensus across metrics on which algorithm is outperforming the others, showing that perceptual evaluation for texture synthesis remains a challenge.

| | Reference-Based Metrics | | | | | No Reference | | |
|---|---|---|---|---|---|---|---|---|
| | SSIM [66] ↑ | Si-FID [56] ↓ | LPIPS [71] ↓ | DISTS [14] ↓ | PieAPP [51] ↓ | BRISQUE [43] ↓ | CLIP-IQA [64] ↑ | **TexTile** ↑ |
| Deloit *et al*. [11] | 0.138 | 1.707 | 0.595 | 0.372 | 2.289 | 49.43 | 0.407 | 0.639 |
| Li *et al*. [36] | 0.149 | 0.981 | **0.559** | 0.341 | 1.672 | 45.84 | 0.437 | 0.707 |
| Rodriguez-Pardo *et al*. [54] | 0.171 | 0.969 | 0.594 | 0.361 | 1.890 | **43.14** | **0.661** | 0.403 |
| Bergmann *et al*. [5] | 0.148 | 0.981 | 0.579 | 0.359 | 1.789 | **44.13** | **0.638** | 0.675 |
| Niklasson *et al*. [48] | 0.146 | **0.819** | 0.623 | 0.446 | 2.292 | 53.99 | 0.435 | 0.731 |
| Rodriguez-Pardo *et al*. [53] | 0.166 | **0.694** | **0.540** | 0.336 | **1.542** | 51.10 | 0.452 | 0.729 |
| Heitz *et al*. [25] w/out TexTile | 0.140 | 1.741 | 0.575 | **0.329** | 1.687 | 49.05 | 0.376 | 0.431 |
| Heitz *et al*. [25] with TexTile | 0.152 | 1.764 | 0.555 | **0.327** | **1.653** | 48.49 | 0.392 | **0.781** |
| Nikankin *et al*. [47] w/out TexTile | **0.172** | 1.314 | 0.591 | 0.386 | 1.963 | 55.96 | 0.408 | 0.388 |
| Nikankin *et al*. [47] with TexTile | **0.189** | 1.415 | 0.569 | 0.387 | 1.926 | 56.99 | 0.396 | **0.798** |

Table 3. Quantitative evaluation between different tileable texture synthesis algorithms across a variety of metrics, including reference and no-reference measures, and TexTile. Best two results for each columns are marked in **bold**.

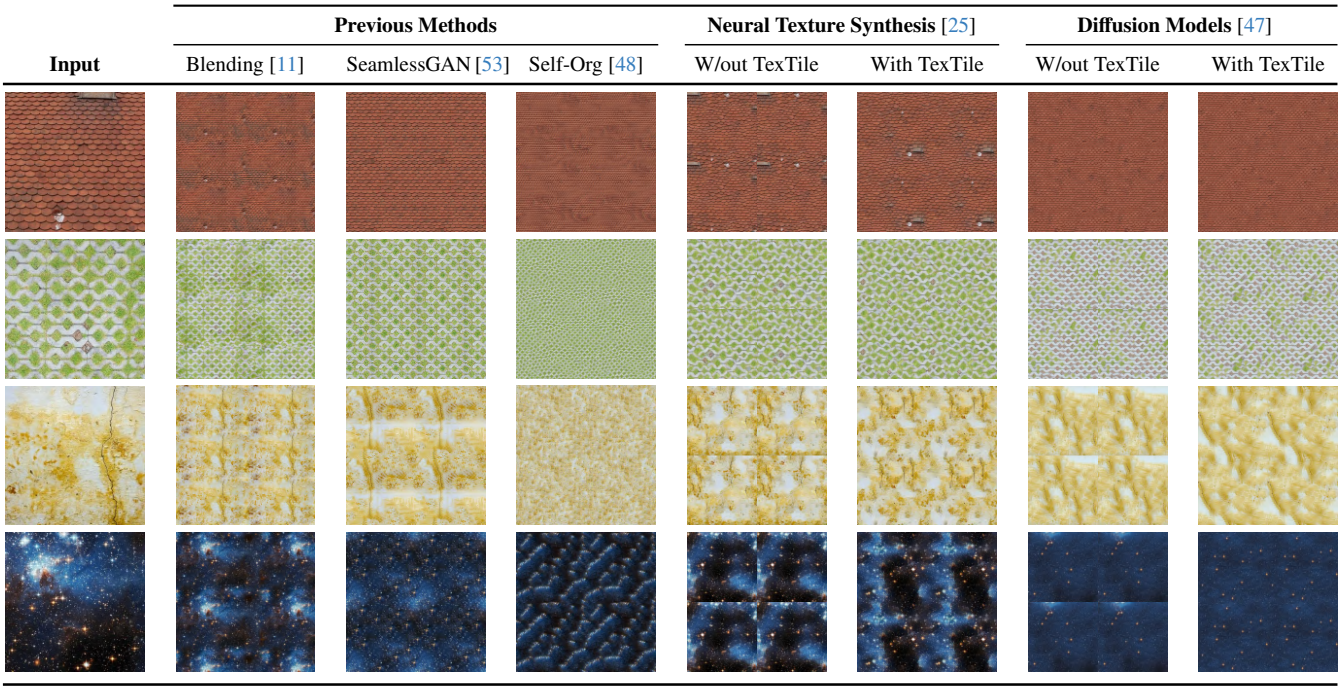


Figure 8. Comparisons of different texture synthesis algorithms. On the leftmost column, we show the input texture, on the right, 2x2 tilings of the outputs of different methods. For [25] and [47], we show the original versions our the modifications for tileable texture synthesis.

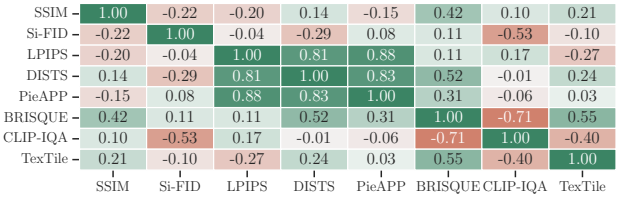


Figure 7. Pearson correlation matrix between different metrics.

## 5.3. Alignment and Repeating Pattern Detection

Besides benchmarking and enabling tileable texture synthesis, our metric enables additional applications.

Previous work on texture analysis used the Radon Transform [30, 54] for automatically aligning images with the $xy$-axes. Similarly, we can find the **optimal rotation angle** $\theta$ for an image I by $\arg\max_\theta$ TexTile(Rotate(I, $\theta$)), maximizing the tileability of the input image. In Figure 9, we show the scores of our metric, for the same image, to which we apply different rotation angles. Our metric provides high scores for rotation angles which provide seamless borders, and lower values for misalignments. Interestingly, there is a small peak at $\pm 35°$, in which the lines in the image connect with each other, but their colors do not match in the borders. These results indicate that TexTile not only measures seamlessness, but also color continuity, and axis alignment.

We can also leverage TexTile to compute the **size of the repeating pattern** in images. Given an axis-aligned image I, we can find the size $h, w$ of the repeating pattern in the image by finding the crop that maximizes its tileability:

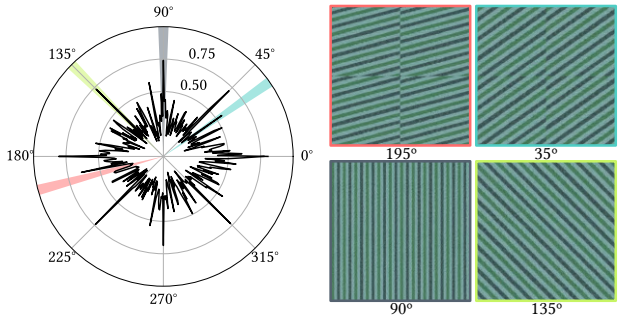


Figure 9. On the left, TexTile under different rotation angles. On the right, samples of rotated images on different local peaks. Scores below 0.5, as the red peak, depict non-tileable textures with highly-noticeable artifacts. A local maxima for a non-tileable texture is found at ±35°, in blue, while highly tileable textures are found at the gray and green insets.

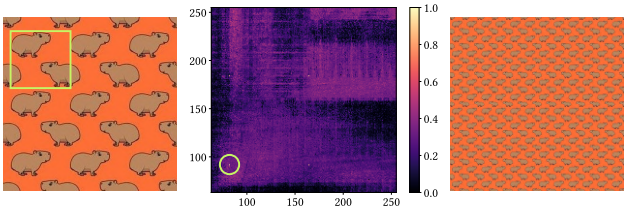


Figure 10. On the left, input image. In the middle, TexTile values for different crop sizes. The optimal crop is highlighted on both images with a green inset. On the right, the optimal crop, tiled many times for visualization.

$\arg\max_{h,w}$ TexTile(Crop(I, $(h, w)$)). We show a result in Figure 10, where this algorithm finds the optimal crop size at the minimum repeatable pattern, as well as three lower peaks at different discrete scale factors of this crop size. Note that we limit the crops to $h, w \geq 64$ and perform the crop at the center of the image. Previous work [35, 54] found these repeating patterns in the activations of pre-trained CNNs. Because internal neural activations operate at lower resolutions that those of the original image, these methods are limited in precision. Our method, in contrast, operates at the resolution of the original image and may thus be more precise. More results on these two applications are present in the supplementary material.

### 5.4. Failure Cases

As shown in Table 1, our model predictions are accurate, however, some errors occur. We show some examples of misclassified textures in Figure 11. On the left, we show a texture labeled as non-tileable in our test dataset, that our model predicts as tileable despite it showing discontinuities in the borders. On the right, we show a texture that is labeled as tileable, which our model classifies as non-tileable. Both examples are edge cases and highlight the ambiguity in what constitutes a tileable texture. Besides, when used as a loss for synthesis models, TexTile cannot compensate for the limitations of the generative backbone. If a model cannot

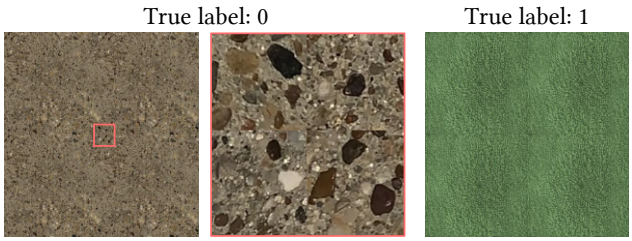


Figure 11. Examples of misclassifications done by our model. On the top, ground truth labels, on the bottom, the predicted TexTile value. For the example on the left, we show an inset of the central crop of the texture, to highlight that it is not seamlessly tileable.

adequately synthesize textures that match the appearance of the input, adding TexTile helps reduce discontinuities in the borders but will not improve its perceptual quality, as can be seen in the textures generated with [25] in Figure 8.

## 6. Conclusions

We have presented *TexTile*, the first differentiable metric for texture tileability. While it is trained on a simple classification task, we design custom data augmentation, training regimes, and neural architectures, all specifically tailored to accurately measure tileability. We validated our design choices with comprehensive ablation studies, and leveraged saliency maps for model understanding. We showed different applications of our differentiable metric, including benchmarking texture synthesis algorithms, detecting repetitions and misalignment in images, and transforming image generative models into tileable texture synthesis algorithms. We will provide code and model weights upon acceptance.

**Limitations and Future Work** We could extend our work in several ways. While our method accurately measures tileability in textures, it is does not quantify their perceptual quality, limiting its scope. Combining perceptual metrics with TexTile, to measure both perceptual quality and tileability, is an important research avenue for a more integrated analysis of texture quality. Besides, our model is pre-trained on a ImageNet, then fine-tuned on a manually-curated dataset, and thus may inherit biases. While we believe the datasets we used were comprehensive, it is possible that some type of texture is underrepresented and the model may not perform accurately. Using synthetic data [18] may help alleviate this issue. Finally, there is no solid understanding of human perception of texture repetitiveness [60], and, while human perceptual validation is out of the scope of this work, its apparent higher correlation with TexTile, might add new insights to the elements identified so far.

**Acknowledgments** This publication is part of the project TaiLOR, CPP2021-008842 funded by MCIN/AEI/10.13039/501100011033 and the NextGenerationEU / PRTR programs. Elena Garces was partially supported by a Juan de la Cierva - Incorporacion Fellowship (IJC2020-044192-I).

## References


[1] Noam Aigerman and Thibault Groueix. Generative escher meshes. *arXiv preprint arXiv:2309.14564*, 2023. 1, 3

[2] Adib Akl, Charles Yaacoub, Marc Donias, Jean-Pierre Da Costa, and Christian Germain. A survey of exemplar-based texture synthesis methods. *Computer Vision and Image Understanding*, 172:12–24, 2018. 3

[3] Dan Amir and Yair Weiss. Understanding and simplifying perceptual distances. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12226–12235, 2021. 2

[4] Adrien Bartoli, Mathieu Perriollat, and Sylvie Chambon. Generalized thin-plate spline warps. *International Journal of Computer Vision*, 88:85–110, 2010. 4

[5] Urs Bergmann, Nikolay Jetchev, and Roland Vollgraf. Learning texture manifolds with the periodic spatial gan. pages 469–477, 2017. 3, 6, 7

[6] Mikołaj Bińkowski, Danica J Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. In *International Conference on Learning Representations*, 2018. 2

[7] Tianshi Cao, Karsten Kreis, Sanja Fidler, Nicholas Sharp, and Kangxue Yin. Texfusion: Synthesizing 3d textures with text-guided image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4169–4181, 2023. 1

[8] Dan Casas and Marc Comino-Trinidad. SMPLitex: A Generative Model and Dataset for 3D Human Texture Estimation from Single Image. In *British Machine Vision Conference (BMVC)*, 2023.

[9] Dave Zhenyu Chen, Yawar Siddiqui, Hsin-Ying Lee, Sergey Tulyakov, and Matthias Nießner. Text2Tex: Text-driven Texture Synthesis via Diffusion Models. 2023. 1

[10] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2014. 4

[11] Thomas Deliot and Eric Heitz. Procedural stochastic textures by tiling and blending. *GPU Zen*, 2, 2019. 1, 3, 6, 7

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 248–255. Ieee, 2009. 3, 5

[13] Valentin Deschaintre, Miika Aittala, Fredo Durand, George Drettakis, and Adrien Bousseau. Single-image svbrdf capture with a rendering-aware deep network. *ACM Transactions on Graphics (TOG)*, 37(4):1–15, 2018. 4

[14] Keyan Ding, Kede Ma, Shiqi Wang, and Eero P Simoncelli. Image quality assessment: Unifying structure and texture similarity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2567–2581, 2020. 2, 7

[15] Timothy Dozat. Incorporating nesterov momentum into adam. 2016. 5

[16] Mario Fritz, E. Hayman, B. Caputo, and J. Eklundh. The kth-tips database. 2004. 4

[17] Ruigang Fu, Qingyong Hu, Xiaohu Dong, Yulan Guo, Yinghui Gao, and Biao Li. Axiom-based grad-cam: Towards accurate visualization and explanation of cnns. *arXiv preprint arXiv:2008.02312*, 2020. 5

[18] Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. Dreamsim: Learning new dimensions of human visual similarity using synthetic data. *arXiv preprint arXiv:2306.09344*, 2023. 2, 8

[19] Bruno Galerne, Ares Lagae, Sylvain Lefebvre, and George Drettakis. Gabor noise by example. *ACM Transactions on Graphics (TOG)*, 31(4):73:1–73:9, 2012. 1

[20] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015. 2

[21] Shouchang Guo, Valentin Deschaintre, Douglas Noll, and Arthur Roullier. U-attention to textures: hierarchical hourglass vision transformer for universal texture synthesis. In *Proceedings of the ACM SIGGRAPH European Conference on Visual Media Production*, pages 1–10, 2022. 3

[22] Yu Guo, Cameron Smith, Miloš Hašan, Kalyan Sunkavalli, and Shuang Zhao. Materialgan: reflectance capture using a generative svbrdf model. *ACM Transactions on Graphics (TOG)*, 39(6):1–13, 2020. 1

[23] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(1):87–110, 2022. 3

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 770–778, 2016. 5

[25] Eric Heitz, Kenneth Vanhoey, Thomas Chambon, and Laurent Belcour. A sliced wasserstein loss for neural texture synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9412–9420, 2021. 6, 7, 8

[26] Katherine Hermann, Ting Chen, and Simon Kornblith. The origins and prevalence of texture bias in convolutional neural networks. *Advances in Neural Information Processing Systems*, 33:19000–19015, 2020. 3

[27] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in Neural Information Processing Systems*, 30, 2017. 2, 5

[28] Yiwei Hu, Julie Dorsey, and Holly Rushmeier. A novel framework for inverse procedural texture modeling. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019. 1

[29] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1125–1134, 2017. 4

[30] Kourosh Jafari-Khouzani and Hamid Soltanian-Zadeh. Radon transform orientation estimation for rotation invariant texture analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(6):1004–1008, 2005. 7

[31] Valentin Khrulkov and Ivan Oseledets. Geometry score: A method for comparing generative adversarial networks. In *International Conference on Machine Learning*, pages 2621–2629. PMLR, 2018. 2, 5

[32] Gustaf Kylberg. The kylberg texture dataset v. 1.0. External report (Blue series) 35, Centre for Image Analysis, Swedish University of Agricultural Sciences and Uppsala University, Uppsala, Sweden, 2011. 4

[33] Tuomas Kynkäänniemi, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Improved precision and recall metric for assessing generative models. *Advances in Neural Information Processing Systems*, 32, 2019. 2

[34] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. A sparse texture representation using local affine regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(8):1265–1278, 2005. 4

[35] Louis Lettry, Michal Perdoch, Kenneth Vanhoey, and Luc Van Gool. Repeated pattern detection using cnn activations. In *IEEE Winter Conference on Applications of Computer Vision*, pages 47–55. IEEE, 2017. 8

[36] Zhengqin Li, Mohammad Shafiei, Ravi Ramamoorthi, Kalyan Sunkavalli, and Manmohan Chandraker. Inverse rendering for complex indoor scenes: Shape, spatially-varying lighting and svbrdf from a single image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2475–2484, 2020. 1, 3, 6, 7

[37] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. *arXiv preprint arXiv:1908.03265*, 2019. 5

[38] Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12009–12019, 2022. 5

[39] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11976–11986, 2022. 2, 3, 5

[40] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 5

[41] P. Mallikarjuna, Alireza Tavakoli Targhi, Mario Fritz, E. Hayman, B. Caputo, and J. Eklundh. The kth-tips 2 database. 2006. 4

[42] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *International Conference on Learning Representations*, 2018. 5

[43] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *IEEE Transactions on Image Processing*, 21(12): 4695–4708, 2012. 2, 7

[44] Anush Krishna Moorthy and Alan Conrad Bovik. A two-step framework for constructing blind image quality indices. *IEEE Signal Processing Letters*, 17(5):513–516, 2010. 2

[45] Joep Moritz, Stuart James, Tom SF Haines, Tobias Ritschel, and Tim Weyrich. Texture stationarization: Turning photos into tileable textures. In *Computer Graphics Forum*, pages 177–188. Wiley Online Library, 2017. 1, 3, 4, 6

[46] Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. *Advances in Neural Information Processing Systems*, 34: 23296–23308, 2021. 3

[47] Yaniv Nikankin, Niv Haim, and Michal Irani. Sinfusion: Training diffusion models on a single image or video. In *International Conference on Machine Learning*. PMLR, 2023. 6, 7

[48] Eyvind Niklasson, Alexander Mordvintsev, Ettore Randazzo, and Michael Levin. Self-organising textures. *Distill*, 2021. https://distill.pub/selforg/2021/textures. 1, 3, 6, 7

[49] Zhaoqing Pan, Feng Yuan, Jianjun Lei, Yuming Fang, Xiao Shao, and Sam Kwong. Vcrnet: Visual compensation restoration network for no-reference image quality assessment. *IEEE Transactions on Image Processing*, 31:1613–1627, 2022. 2

[50] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 5

[51] Ekta Prashnani, Hong Cai, Yasamin Mostofi, and Pradeep Sen. Pieapp: Perceptual image-error assessment through pairwise preference. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1808–1817, 2018. 2, 7

[52] Edgar Riba, Dmytro Mishkin, Daniel Ponsa, Ethan Rublee, and Gary Bradski. Kornia: an open source differentiable computer vision library for pytorch. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 3674–3683, 2020. 5

[53] Carlos Rodriguez-Pardo and Elena Garces. Seamlessgan: Self-supervised synthesis of tileable texture maps. *IEEE Transactions on Visualization and Computer Graphics*, 2022. 1, 2, 3, 6, 7

[54] Carlos Rodriguez-Pardo, Sergio Suja, David Pascual, Jorge Lopez-Moreno, and Elena Garces. Automatic extraction and synthesis of regular repeatable patterns. *Computers & Graphics*, 83:33–41, 2019. 1, 3, 6, 7, 8

[55] Carlos Rodriguez-Pardo, Henar Dominguez-Elvira, David Pascual-Hernandez, and Elena Garces. Umat: Uncertainty-aware single image high resolution material capture. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. 3

[56] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. SinGAN: Learning a Generative Model from a Single Natural Image. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019. 2, 7

[57] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in Neural Information Processing Systems*, 29, 2016. 2

[58] K Simonyan and A Zisserman. Very deep convolutional networks for large-scale image recognition. In *International*

*Conference on Learning Representations*. Computational and Biological Learning Society, 2015. 5

[59] Shaolin Su, Qingsen Yan, Yu Zhu, Cheng Zhang, Xin Ge, Jinqiu Sun, and Yanning Zhang. Blindly assess image quality in the wild guided by a self-adaptive hyper network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3667–3676, 2020. 2

[60] Hua-Chun Sun, David St-Amand, Curtis L Baker Jr, and Frederick AA Kingdom. Visual perception of texture regularity: Conjoint measurements and a wavelet response-distribution model. *PLoS Computational Biology*, 17(10): e1008802, 2021. 8

[61] Anton Tsitsulin, Marina Munkhoeva, Davide Mottin, Panagiotis Karras, Alex Bronstein, Ivan Oseledets, and Emmanuel Mueller. The shape of data: Intrinsic distance for data distributions. In *International Conference on Learning Representations*, 2019. 2, 5

[62] Giuseppe Vecchio, Rosalie Martin, Arthur Roullier, Adrien Kaiser, Romain Rouffet, Valentin Deschaintre, and Tamy Boubekeur. Controlmat: A controlled generative approach to material capture. *arXiv preprint arXiv:2309.01700*, 2023. 3

[63] Madhawa Vidanapathirana, Qirui Wu, Yasutaka Furukawa, Angel X. Chang, and Manolis Savva. Plan2scene: Converting floorplans to 3d scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10733–10742, 2021. 4

[64] Jianyi Wang, Kelvin CK Chan, and Chen Change Loy. Exploring clip for assessing the look and feel of images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2555–2563, 2023. 2, 7

[65] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. *arXiv preprint arXiv:2006.04768*, 2020. 3, 5

[66] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 2, 7

[67] Xudong Xie, Zijie Wu, Zhiliang Xu, and Zhen Zhu. Learning in a single domain for non-stationary multi-texture synthesis. *arXiv preprint arXiv:2305.06200*, 2023. 2

[68] Sidi Yang, Tianhe Wu, Shuwei Shi, Shanshan Lao, Yuan Gong, Mingdeng Cao, Jiahao Wang, and Yujiu Yang. Maniqa: Multi-dimension attention network for no-reference image quality assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1191–1200, 2022. 2

[69] Lin Zhang, Lei Zhang, Xuanqin Mou, and David Zhang. Fsim: A feature similarity index for image quality assessment. *IEEE Transactions on Image Processing*, 20(8):2378–2386, 2011. 2

[70] Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. Lookahead optimizer: k steps forward, 1 step back. *Advances in neural information processing systems*, 32, 2019. 5

[71] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 586–595, 2018. 2, 7

[72] Xilong Zhou and Nima Khademi Kalantari. Adversarial single-image svbrdf estimation with hybrid training. *Computer Graphics Forum*, 2021. 4

[73] Xilong Zhou and Nima Khademi Kalantari. Look-ahead training with learned reflectance loss for single-image svbrdf estimation. *ACM Transactions on Graphics (TOG)*, 41(6), 2022. 4

[74] Xilong Zhou, Milos Hasan, Valentin Deschaintre, Paul Guerrero, Kalyan Sunkavalli, and Nima Khademi Kalantari. Tilegen: Tileable, controllable material generation and capture. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–9, 2022. 1, 3

[75] Yang Zhou, Zhen Zhu, Xiang Bai, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. Non-stationary texture synthesis by adversarial expansion. *ACM Transactions on Graphics (TOG)*, 37(4):1–13, 2018. 4