Example-Based Feature Painting on Textures

ANDREI-TIMOTEI ARDELEAN, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
TIM WEYRICH, Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany

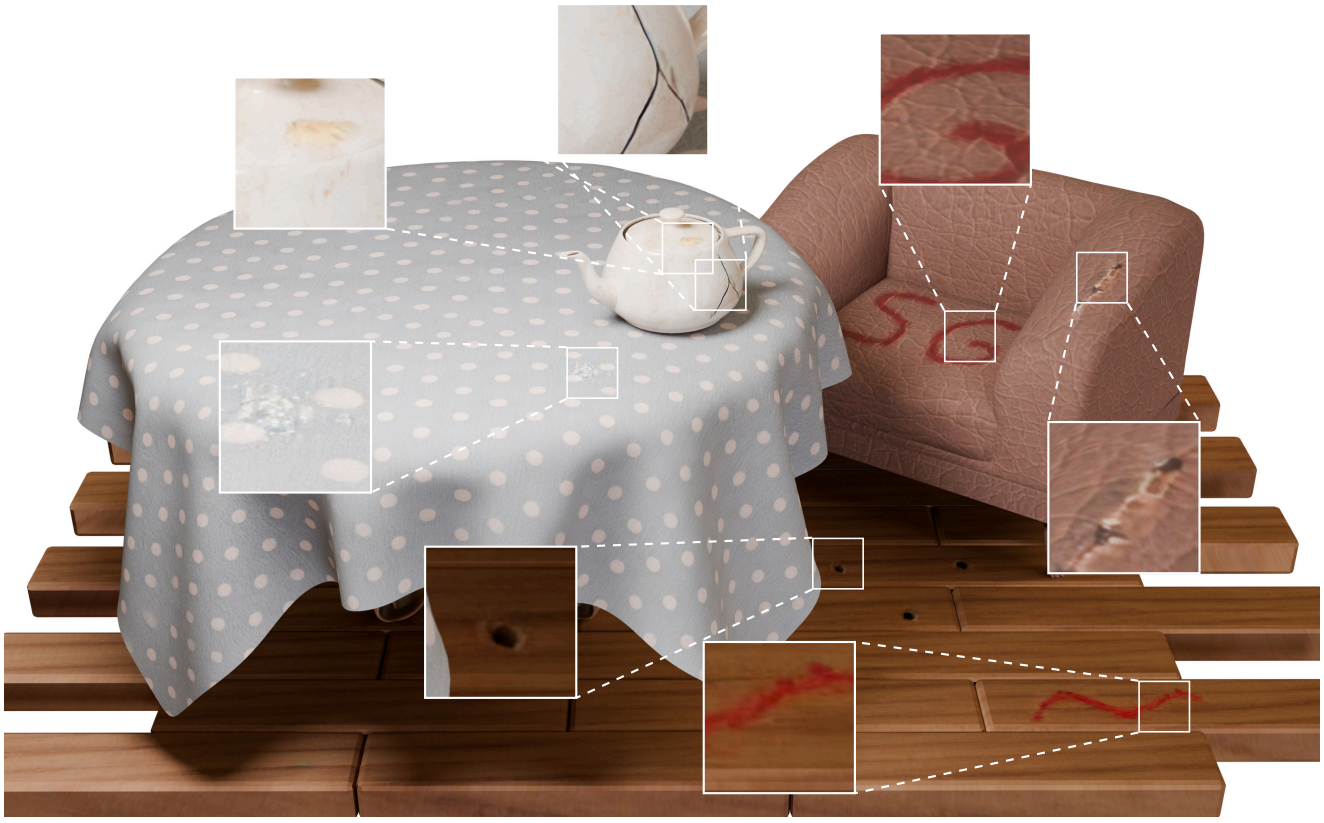

Fig. 1. Our method generates large textures with non-stationary features learned from a small number of images. The type and shape of the painted features can be freely controlled: the figure is the result of an interactive editing session (see supplementary video). The scene uses free 3D models from cgtrader.com.

In this work, we propose a system that covers the complete workflow for achieving controlled authoring and editing of textures that present distinctive local characteristics. These include various effects that change the surface appearance of materials, such as stains, tears, holes, abrasions, discoloration, and more. Such alterations are ubiquitous in nature, and including them in the synthesis process is crucial for generating realistic textures. We introduce a novel approach for creating textures with such blemishes, adopting a learning-based approach that leverages unlabeled examples. Our approach does not require manual annotations by the user; instead, it detects the appearance-altering features through unsupervised anomaly detection. The various textural features are then automatically clustered into semantically coherent groups, which are used to guide the conditional generation of images. Our pipeline as a whole goes from a small image collection to a versatile generative model that enables the user to interactively create and paint features on textures of arbitrary size. Notably, the algorithms we introduce for diffusion-based editing and infinite stationary texture generation are generic and should prove useful in other contexts as well.
Project page: `reality.tf.fau.de/pub/ardelean2025examplebased.html`

Authors' Contact Information: Andrei-Timotei Ardelean, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany, timotei.ardelean@fau.de; Tim Weyrich, Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany, tim.weyrich@fau.de.

ACM 1557-7368/2025/12-ART183
https://doi.org/10.1145/3763301

CCS Concepts: • **Computing methodologies** → **Texturing**; **Anomaly detection**; *Image segmentation*.

Additional Key Words and Phrases: Texture synthesis, Anomaly clustering, Conditioned generation

ACM Reference Format:
Andrei-Timotei Ardelean and Tim Weyrich. 2025. Example-Based Feature Painting on Textures. *ACM Trans. Graph.* 44, 6, Article 183 (December 2025), 24 pages. https://doi.org/10.1145/3763301

1 Introduction

In visual content creation, automatic image generation and texturing methods seek to assist artists, reducing the time and effort required to develop photo-realistic texture assets. Many of the existing tools, however, are biased toward an idealized, pristine appearance, so that texture artists spend significant time to augment their textures with the type of blemishes and imperfections characteristic to real-world surfaces. Our work offers an automated framework to analyze texture samples, separating their characteristic (stationary) statistics from sporadic irregularities, that are equally characteristic, feeding an interactive system to paint such prominent features on top of the underlying pristine texture.

Our system takes as input a small number of (unannotated) images representative of a certain material, which include both normal appearance and irregular features (stains, cracks, holes, and abrasions, etc.). The user then specifies the locations of irregularities, and our method synthesizes arbitrarily large textures that resemble the texture in the input images, with the user-specified features naturally blending into the surrounding texture. We present the first framework that simultaneously holds the following capabilities:

(1) **Automatically** extracts the normal and irregular texture appearance from a small number of images.
(2) Generates textures with **spatial** and **semantic** control, that is also **interactive**.
(3) Facilitates **painting features** on both synthesized textures and real images through **feature transfer**.
(4) Creates **arbitrarily large** textures without distribution drift.

2 Related Work

Painting features on textures is a task traditionally undertaken by artists that design digital assets. The toolbox (e.g., Substance 3D painter [Adobe 2023b]) generally includes a library of carefully crafted materials that covers frequently occurring effects, such as dirt accumulation, or cracks. Alongside the materials, there usually are pre-made alpha-masks that represent a realistic distribution of the desired features. An artist would then select an appropriate combination to overlay on the canvas as needed. This process can be time-consuming and it is generally limited to the available pool of materials and patterns, which may not be easy to customize for a specific application. In order to extend their capabilities, digital creation software (*e.g.*, Photoshop [Adobe 2023a], Gimp [GNU 2024], Substance 3D painter [Adobe 2023b]) include clone stamp brushes; however, the features are simply copied over the canvas, making it difficult to ensure realistic effects and transitions.

2.1 Example-based feature synthesis

Several research projects directly or indirectly target the replication of image features into a new context. In the seminal work of image analogies [Hertzmann et al. 2001], the texture-by-numbers method is used to create a new instance of a texture and paint the prominent features as guided by a layout map. The same can be achieved using single-image generative models and image reshuffling techniques. SinGAN [Shaham et al. 2019] and SinDiffusion [Wang et al. 2022a] train a generative adversarial network (GAN) and a diffusion model [Dhariwal and Nichol 2021] respectively from a single image. The models can then be used to generate similar, realistic patches in a new configuration. GPNN [Granot et al. 2022], Image style transfer [Gatys et al. 2016], Sliced Wasserstein [Heitz et al. 2021], and GPDN [Elnekave and Weiss 2022] are non-parametric generative methods that can be used to paint features by mimicking the patches and statistics from a given source image. Neural Texture Synthesis with Guided Correspondence [Zhou et al. 2023], Non-Stationary Textures using Self-Rectification [Zhou et al. 2024], and Texture Reformer [Wang et al. 2022b] focus specifically on textures with non-stationary regions. These approaches showcase different ways to condition the generation in order to improve the authoring process: orientation and progression control, content image, and a collage of crops. Painting With Texture [Ritter et al. 2006], Painting by Feature [Lukáč et al. 2013], Brushables [Lukáč et al. 2015], and Neural Brushstroke [Shugrina et al. 2022] propose different ways to create brushes from images, enabling painting of the extracted textural features on a canvas. Recently, Diffusion Painting [Hu et al. 2024] displayed remarkable capabilities in terms of the texture complexity controllable by a brush. Using a diffusion model pretrained on a large dataset, the method hallucinates realistic variations and transitions from a single texture image, albeit with limited fidelity to the input mask.

While the above-mentioned approaches share the advantage of being able to work from a single image, it is desirable to incorporate the information from multiple source images of the same texture class if available, as a single image often does not cover the appearance of a feature type in its fullness, or does not contain all transition types that are characteristic for the given material. That being said, in most cases it is not straight forward to extend the methods to accept multiple images in a way that actually improves the generation process. Our method is fine-tuned on multiple images from a single texture class in order to learn from several instantiations of a certain prominent feature type.

A limitation of previous methods is the requirement to manually select and indicate the relevant features from the input images. As we want to capture the entire distribution of a texture and/or the possible prominent features (stains, cuts, etc), several dozen images might be needed. In this case, manual segmentation would put an unreasonable burden on the user. Therefore, in this work we also address a preprocessing part in the asset-creation process, by automatically finding and grouping the relevant features.

2.2 Anomaly Localization and Classification

In order to automatically detect the prominent features, which are non-stationary regions in the texture, we pose the problem as an anomaly detection task. Since the input consists of a mix of (unlabeled) normal and anomalous features, we find ourselves in a fully unsupervised setting. This is more challenging than the one-class classification task employed by most anomaly detection methods, where the normal instances are labeled [Batzner et al. 2024; Deng and Li 2022; Roth et al. 2022].

Fully unsupervised anomaly localization is framed as normality-supervised detection with contamination [Liu et al. 2022; Patel et al.

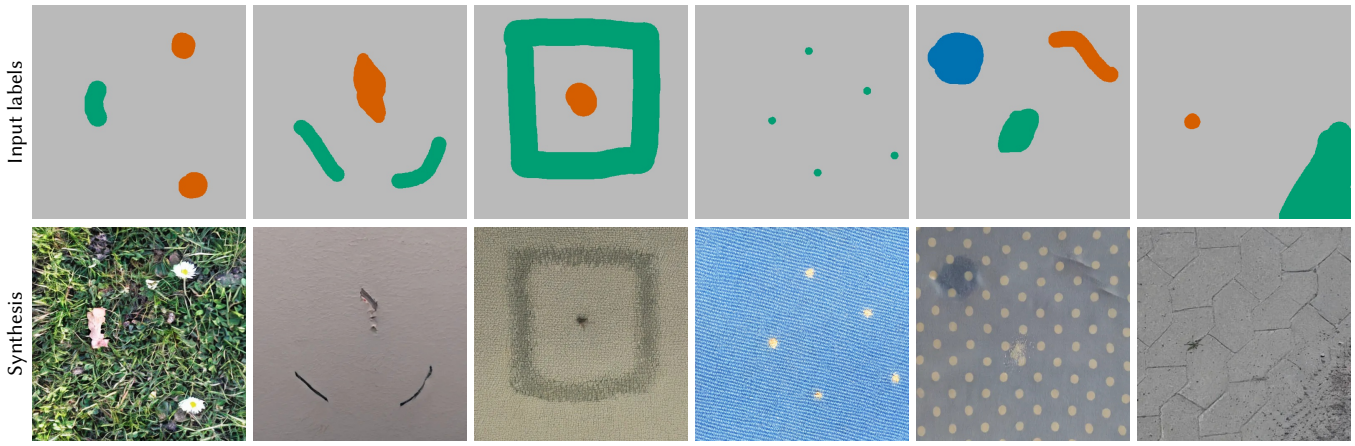

Fig. 2. Feature-conditioned results generated by our model on various textures. The label maps are sketched by the user/artist. Please note the graceful, realistic transitions between the normal class and the painted features; digital zoom recommended.

2023; Yoon et al. 2022; Zhang et al. 2024] or zero-shot anomaly detection with test-time adaptation [Li et al. 2023, 2024]. BlindLCA [Ardelean and Weyrich 2024b] is the current state-of-the-art method designed specifically for textures. We build on this method and adapt it to pixel-level anomaly segmentation. The unsupervised classification of anomalies into semantic categories has been approached by prior work [Ardelean and Weyrich 2024b; Sohn et al. 2023] at the image level. Differently, we perform the semantic segmentation at the level of pixels rather than images and lift the limitation of a single anomaly type per texture instance.

More often than not, the features that are painted on a certain texture are trying to replicate naturally occurring defects that can be attributed to weathering. Therefore, weathering synthesis is targeted by methods related to altering the appearance of textures.

2.3 Weathering synthesis

A traditional approach to weathering is to develop material-specific simulations based on the physical and chemical processes that occur through time and update the appearance accordingly [Bajo et al. 2021; Chen et al. 2005; Dorsey and Hanrahany 2006; Liu et al. 2012]. While there are certain advantages granted by a physically-based model, this class of methods does not generalize across different materials, and it often requires an explicit model of the external factors that influence the weathering.

A different line of work builds time-varying appearance models based on a single image and a few annotations provided by the user. Wang et al. [2006] are the first to develop an appearance manifold in BRDF space and use it to model the time-dependent change from normal to weathered. The manifold is constructed with the help of the user, who selects the least- and most-weathered parts of the image. A similar strategy was successfully applied directly in color space [Bandeira and Walter 2009; Xuey et al. 2008], separating illuminance and reflectance. Iizuka et al. [2016] advance from simple chromatic transitions and produce weathering with more complex appearance. Synthesis is performed using image quilting techniques [Efros and Freeman 2001] based on a weathering exemplar. To improve the coverage of weathered appearance distribution, Du and Song [2023] additionally identify discrete degrees of weathering and create multiple weathering exemplars accordingly. Bellini et al. [2016] propose a time-varying weathering framework that does not require user-annotations. The method predicts anomaly-maps which are used to remove the anomalies or to increase their amount by repeating the existing flaws. While these methods yield impressive results from just one image, they are restricted to a single weathering trajectory (type) and can mostly model relatively simple effects, *e.g.*, decoloration, peeling, moss growth, and oxidation.

Learning-based methods can use multiple images and model various weathering types and effects. Chen et al. [2021a] pose the weathering task as an image-to-image translation problem using a pix2pix [Isola et al. 2017] GAN. The model is trained on a single image, with automatic annotations [Bellini et al. 2016]. While their approach could support multiple images for training, it is limited to a single weathering type. Recently, Hao et al. [2023] introduced an approach for weathered texture synthesis that supports multiple defect types; however, the method is trained on several thousand weathered images prefiltered and grouped by type. The guided texture transition method of Guerrero-Viu et al. [2024] allows users to holistically modify an image with fine control over the degree of weathering, and the concurrent work of Hadadan et al. [2025] aims to add realistic details, such as signs of wear, using text prompts. In contrast, we focus on spatially controllable edits, while preserving the appearance of unweathered regions.

Our approach allows the user to create weathered textures of arbitrary size, and therefore relates to the creation of infinite textures using generative models [Bergmann et al. 2017; Lin et al. 2021; Wang et al. 2024]. Similarly to Wang et al. [2024], we use a diffusion model and averaged denoising scores [Bar-Tal et al. 2023] to progressively generate a large texture. Additionally, we propose a way to reduce distribution drift, preserving global consistency.

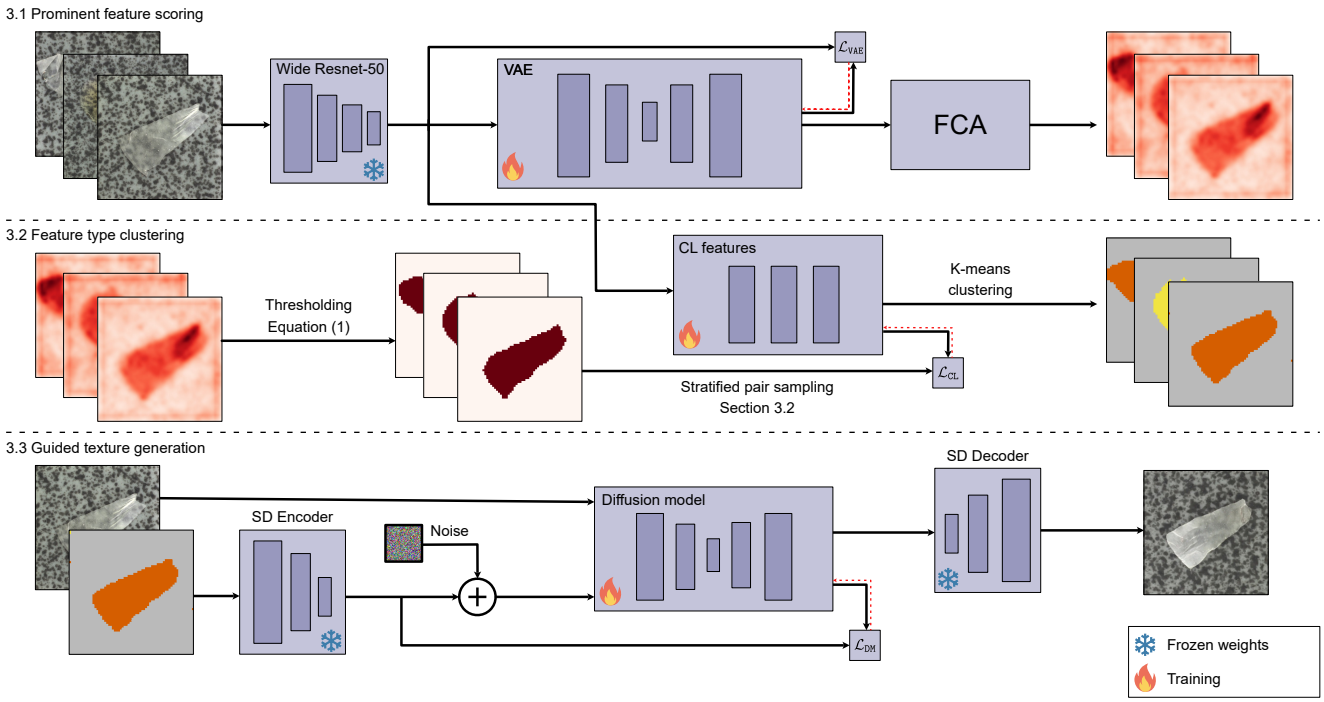

Fig. 3. Our 3-stage pipeline for obtaining a generative model for textures with prominent features.

3 Method

Our pipeline consists of three stages (Fig. 3). The first stage is our approach for automatically identifying regions with prominent features by posing that task as a fully-unsupervised anomaly localization problem. This framing is feasible since regions with irregular features break the stationarity assumption of textures, making these regions deviate from the overall statistics. After identifying such regions, the second stage addresses the separation of prominent features in different groups. Leveraging the anomaly maps, we devise an approach for sampling positive and negative pairs of pixels, which are used to optimize a contrastive learning objective. This produces a disentangled feature space that we cluster using k-means to obtain pixel-level semantic maps where the labels indicate the feature type or the absence of a prominent feature accordingly. The result of the second stage enables us to train a generative model that follows the desired spatial and semantic conditions. Our diffusion model can generate new textures interactively (1 sec for a 512×512 image). Moreover, we demonstrate synthesis of arbitrary size using constant GPU memory, and design a way to avoid distribution drift across the generated texture. Finally, we advance a noise-mixing technique to support texture editing while remaining faithful to the original input. Our editing method even supports transferring blemishes from one texture class to an image of another class.

3.1 Prominent feature scoring

The input of our pipeline consists of a handful of photographs of textures that are largely stationary, yet contain prominent features that we want to disentangle. These irregularities can be considered anomalies in the context of the global textures, which can be detected using unsupervised anomaly localization.

The initial stage of our pipeline leverages an unsupervised anomaly detection approach, as a first step towards removing the need for manual user annotations, required by previous methods [Du and Song 2023; Iizuka et al. 2016]. We employ FCA, the zero-shot anomaly detection method of Ardelean and Weyrich [2024a], and apply it to the residual obtained from a VAE-based reconstruction, similarly to Blind LCA [Ardelean and Weyrich 2024b]. The output of this step is a set of anomaly-score maps $\{A_i\}_{i=1}^N$, one for each input image $\{I_i\}_{i=1}^N$, as shown in the first section of Fig. 3. These methods, however, do not provide a way to threshold the anomaly scores, which have an arbitrary range.

3.2 Feature type clustering

Our pixel-level semantic segmentation procedure requires well-defined (binary) anomaly regions. Therefore, we propose an adaptive thresholding function to obtain a binary segmentation B_i of the areas of interest. Let $B_i^{xy} := \begin{cases} 1 & A_i^{xy} > \mathcal{T}(A_i) \\ 0 & \text{otherwise} \end{cases}$; where $\mathcal{T}$ is a function that adaptively computes the threshold based on the anomaly scores. We observed that global thresholding and simple quantiles do not generalize well across different datasets. Otsu's method [Otsu et al. 1975] yields reasonable results, yet it tends to produce segmentations that are too permissive. To reduce the number of false positives, we skew the distribution using an exponential factor: $\hat{\mathcal{T}}(A_i) = \texttt{Otsu}(A_i^\beta)^{1/\beta}$, where $\beta = 1.5$ in our experiments.

This function is able to find adequate thresholds across different datasets; however, since it always segments a part of the image as

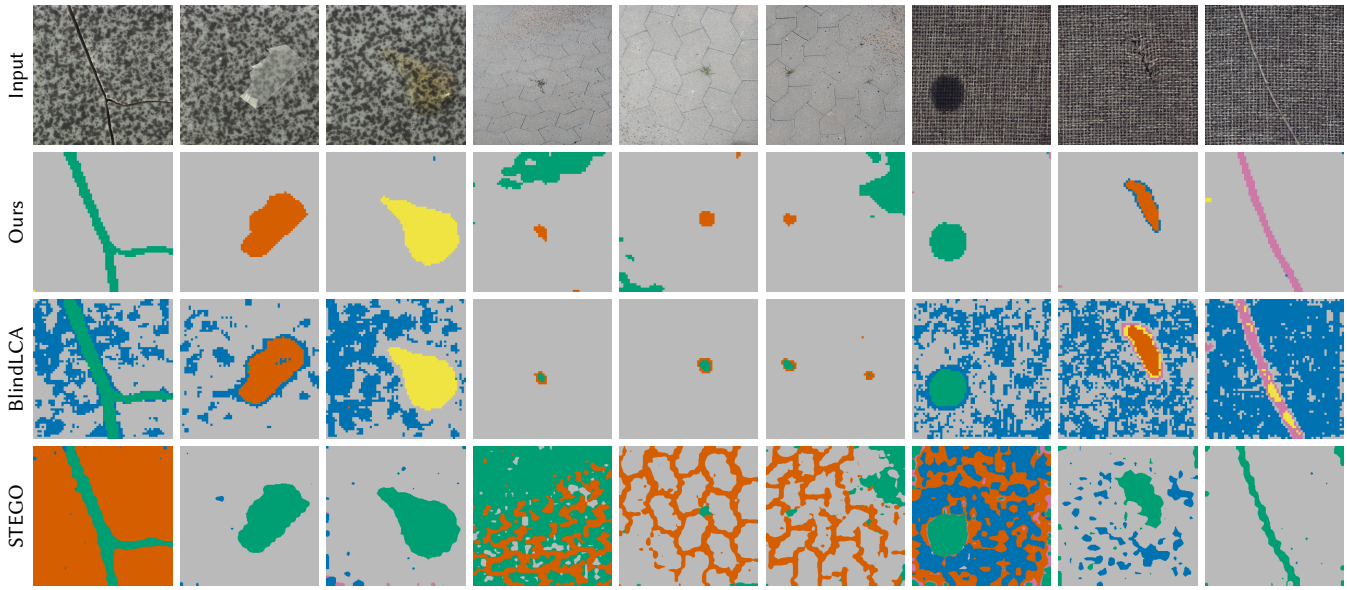

Fig. 4. Results of our pixel-level anomaly segmentation. We seek to have the background pixels mapped to a single class and the different features to distinct labels. The figure compares the results of our approach, BlindLCA [Ardelean and Weyrich 2024b], and STEGO [Hamilton et al. 2022].

anomalous it will inevitably yield false positives on textures that are completely stationary. To overcome this issue, we compute a global threshold and segment each anomaly map based on the maximum between the local and global thresholds:

$$\mathcal{T}(A_i) := \max(\texttt{Otsu}(\{\hat{\mathcal{T}}(A_j)\}_{j=1}^{N}), \hat{\mathcal{T}}(A_i)) \ . \tag{1}$$

We empirically validate our thresholding function in the supplementary material (Sec S8).

To enhance the authoring control of the final generative model, we are interested in grouping the prominent features of the same type into clusters. Based on the binary detection of anomalous regions, we identify the various types of features through contrastive learning (CL) and clustering. As opposed to prior work [Ardelean and Weyrich 2024b], we are interested in pixel-level rather than image-level clustering. Moreover, we do not assume there is only one possible anomaly type per image, making the method more flexible and applicable in practical scenarios. To this end, we divide each image into regions by computing the connected components in the binary masks B_i. For simplicity, it is reasonable to assume that each such region either belongs to the normal class or contains a single type of feature. We then compute region-level descriptors by averaging the pixel-level features extracted by a pretrained WideResnet-50 network, and use the descriptors to find positive pairs as nearest neighbors. Importantly, we observe that naively creating negative pairs for contrastive learning by sampling regions that are far away in feature space yields poor results. As shown in Fig. 26, this is due to the oversampling of the normal class, which limits the effectiveness of contrastive learning to separate the various anomaly types. We alleviate this problem by sampling negative pairs in a stratified manner, *i.e.*, the regions are preclustered using k-means based on the computed feature descriptors, and negative pairs are sampled equally across these clusters. The number of classes used for preclustering is a free hyperparameter of our method; that being said, we observed that using the same number of classes as for the final clustering is a robust choice.

The positive and negative pairs are used to train a 3-layer CNN, with a receptive field of 5×5, optimizing for the InfoNCE [Oord et al. 2018] contrastive objective. After training, the CL-features produced by the neural net are easily separable into clusters: to obtain pixel-level segmentations, we pool the CL-features for all pixels in all input images and cluster them using k-means.

3.3 Guided texture generation

The output of the clustering stage consists of N label maps $\{L\}_{i=1}^{N}$ with values from 0 (normal class) to K, the number of feature types in the dataset. We leverage the label maps as conditioning for a diffusion model [Ho et al. 2020] (DM) trained to generate the same set of images $\{I_i\}_{i=1}^{N}$.

Diffusion models are a class of generative models that synthesize outputs through progressive denoising. We include here a brief explanation of the concept of diffusion models, and refer the reader to a more extensive analysis of these models [Ho et al. 2020; Karras et al. 2022; Song et al. 2021a,b]. Let $p_{\text{data}}(\boldsymbol{x})$ be the distribution of the data to be modeled; in our case, natural textures. The forward diffusion process pushes samples away from the true data distribution by adding i.i.d. (independent identically distributed) Gaussian noise with standard deviation $\sigma(t)$. The noise is increasing with the timestep t, so that for a large enough t_N the resulting distribution $p(\boldsymbol{x}; \sigma(t_N))$ is virtually identical to the normal distribution $\mathcal{N}(\mathbf{0}, \sigma(t_N)^2 \boldsymbol{I})$. Sampling from $p_{\text{data}}(\boldsymbol{x})$ can be achieved by reversing the diffusion process. In practice, this is implemented by a progressively applied, learned denoising function, starting from pure noise

sampled from $\mathcal{N}(\mathbf{0}, \sigma(t_N)^2\boldsymbol{I})$. The denoising is applied at several steps with decreasing standard deviation, essentially modeling the distribution $p(\boldsymbol{x}; \sigma(t_i))$, with a monotonic timestep schedule $\{t_i\}_{i=0}^{N}$, where $\sigma(t_0) = 0$. After iterating from N to 0, the resulting samples should resemble the original distribution $p(\boldsymbol{x}; 0)$. In our implementation, we follow the ODE formulation from Karras et al. [2022] (EDM) including scheduling, preconditioning, and the Heun solver. The direction to the data distribution is approximated by a neural network $\boldsymbol{\epsilon}(\boldsymbol{x}, \sigma(t))$ that can be trained with a simple MSE loss. Similarly to Karras et al. [2024], we perform the diffusion in latent space, using the Variational Autoencoder (VAE) from Stable Diffusion (SD) [Rombach et al. 2022]. Moreover, we add spatial conditioning to enable fine-grained control over the generated images.

Conditioning a generative model through a spatial map has been used for many image-to-image translation tasks, such as layout-based generation, colorization, and inpainting. Our use-case is similar to the generation of images based on semantic maps [Ko et al. 2024; Park et al. 2019; Zhang et al. 2023], where the semantic labels are represented by the K feature types. Our method is designed to work with a very small number of photographs – as low as one. To incorporate this additional input, we modify the U-Net-based [Ronneberger et al. 2015] ADM [Dhariwal and Nichol 2021] architecture used by EDM [Karras et al. 2022]: The label maps are encoded with a small convolutional network and then added to the timestep embeddings to modulate the activations of the U-Net at several intermediate layers. In the original architecture, in each U-Net block, the timestep embedding is processed using a linear layer to predict an affine transformation for each feature channel. We modify the U-Net blocks to accept spatially varying embeddings, which are processed by a convolutional layer to predict a different scale and shift for each spatial position and channel. The additional parameters are trained jointly with the rest of the network. To facilitate training from a few images, we pretrain our diffusion model on the DTD dataset [Cimpoi et al. 2014] (5640 textures). When training the conditional diffusion model $\boldsymbol{\epsilon}(\boldsymbol{x}, \boldsymbol{c}; \sigma(t))$ the label map $\boldsymbol{c}$ is dropped with a probability of 7.5%, enabling classifier-free guidance [Ho and Salimans 2021] during inference. Since the dataset contains different (47) texture classes, we use the class labels instead of semantic masks as conditioning, that is, the class label is spatially expanded to match the shape required by the model. Note that since the number of prominent features (semantic classes) during fine-tuning differs from the number of classes in the DTD dataset used during pretraining, we cannot reuse the first convolutional layer that embeds the class label; therefore, we drop that specific layer and train it from scratch.

We choose EDM as our backbone as it generates high-quality images with low latency, enabling our interactive synthesis goal. Other diffusion models or different generative approaches could potentially be used for this stage (e.g., inpainting [Lugmayr et al. 2022; Suvorov et al. 2022] or Texture Mixer [Yu et al. 2019]). We provide an extended analysis of different choices in the supplementary (S11).

Similarly, various mechanisms for fine-tuning diffusion models have been established in recent years, such as: Dreambooth [Ruiz et al. 2023] for the preservation of visual characteristic of a certain subject, ControlNet [Zhang et al. 2023] for injecting spatial control into a diffusion model trained without such condition, and LoRA [Hu et al. 2022] to make fine-tuning more efficient with respect to the number of training samples, time, and memory footprint. A combination of the techniques above (e.g., ControlLora [Hecong 2024] and CtrLora [Xu et al. 2025]) may enable the generative model of our pipeline to use larger models with massive pretraining. In this work, however, we use a lightweight network and very little pretraining, prioritizing interactive inference time and limiting dependence on large-scale data.

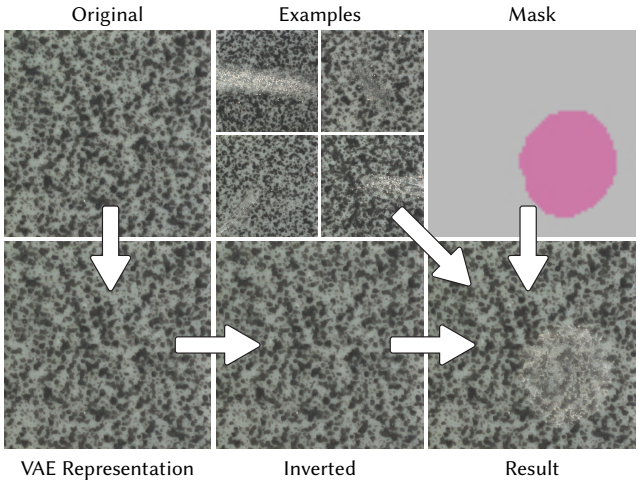

Fig. 5. Image editing visualization, showing: the original image, irregular feature examples from the dataset, desired edit mask, the image after being encoded and decoded by the VAE of SD, the result of our diffusion model for the inverted noise, and the result of our noise-mixing synthesis.

As we show in Fig. 2, this setup allows us to generate new, realistic images, with the desired texture, that follow the conditioning of the label maps. This functionality covers the first two capabilities of our pipeline. In the following, we present how our method can be used for real-image editing and arbitrary-size image generation.

3.4 Editing

The possibility to edit existing images is a crucial capability in the context of generating textures with non-stationary features. Moreover, the edited region should not simply *replace* the previous texture, but remain faithful to the original structure, as highlighted in Fig. 5 and Fig. 6. Most diffusion-based editing frameworks rely on diffusion inversion [Hertz et al. 2023; Wallace et al. 2023; Zhang et al. 2025], *i.e.*, an input image is reproduced by the diffusion model by finding the noise $\boldsymbol{z}_N$ that produces the image when solving the underlying ODE. This is achieved by running the noise estimation model $\boldsymbol{\epsilon}(\boldsymbol{z}_i, \varnothing; \sigma_i)$ with the null conditioning $\varnothing$ for N steps, finding the noise that must be added at each iteration. That is $\boldsymbol{z}_{i+1} = \boldsymbol{z}_i + (\sigma_{i+1} - \sigma_i)\boldsymbol{\epsilon}(\boldsymbol{z}_i, \varnothing, \sigma_i)$, resting on the assumption that $\boldsymbol{\epsilon}(\boldsymbol{z}_i, \varnothing, \sigma_i) \sim \boldsymbol{\epsilon}(\boldsymbol{z}_{i+1}, \varnothing; \sigma_{i+1})$. Note that we write $\sigma_i := \sigma(t_i)$ for brevity.

After obtaining $\boldsymbol{z}_N$, the edited image can be synthesized by running the generative process from the found noise with the new desired conditioning label map. We opt for the fixed-point iteration method [Pan et al. 2023] for diffusion inversion, and implement it

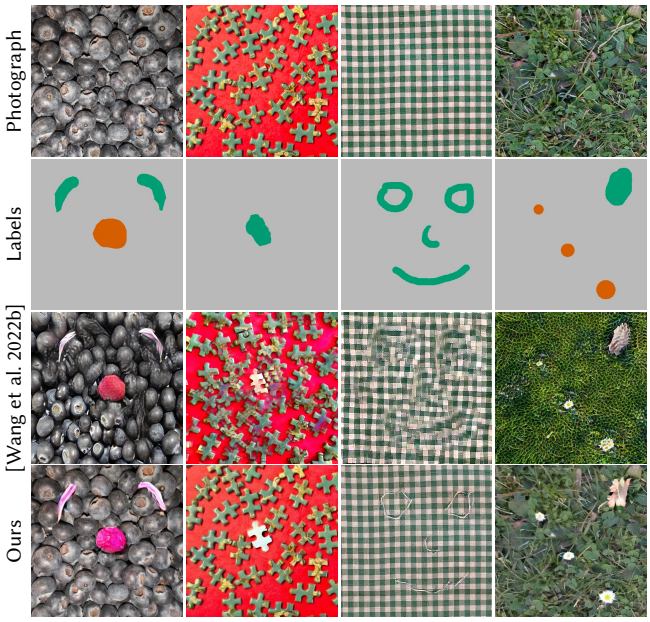

Fig. 6. Editing results with our noise-mixing approach. The rows depict top to bottom: a real photograph, the conditioning label-map, the result obtained with TextureReformer [Wang et al. 2022b], and our edit.

Fig. 7. Comparison between images sampled independently (*top*) and different uniformization strategies. Rows 2 through 4 attempt to make all images follow the style of the first column, using the methods explained in Sec. 3.5.

into the the EDM formulation. However, we found that the inversion is unstable when the second-order Heun solver is used. This is because each denoising step is harder to invert and errors accumulate; therefore, we use the Euler solver in the context of image editing.

3.4.1 Localized updates.

Since the painted features are localized, there is an expectation that the image remains as close as possible to the original outside of the edited region; the painted feature should also be compatible with the initial texture (for example, the sugar spill should not alter the texture below, Fig. 12). We propose to solve this via **noise-mixing**: during inversion, we save the intermediate noise estimates $\boldsymbol{\epsilon}(\boldsymbol{z}_i, \varnothing; \sigma_i)$, for each level σ_i. Then, to generate the new image from the inverted noise $\boldsymbol{z}_N$, we mix the noise estimates saved during inversion with the new noise estimates ($\hat{\boldsymbol{\epsilon}}$) that are conditioned by the label map $\boldsymbol{c}$:

$$\hat{\boldsymbol{z}}_{i-1} = \hat{\boldsymbol{z}}_i + (\sigma_{i-1} - \sigma_i)\texttt{mix}(\hat{\boldsymbol{\epsilon}}(\hat{\boldsymbol{z}}_i, \boldsymbol{c}; \sigma_i), \boldsymbol{\epsilon}(\boldsymbol{z}_i, \varnothing; \sigma_i), i) \quad . \tag{2}$$

Where $\hat{\boldsymbol{\epsilon}}$ denotes the classifier-free guided direction: $\hat{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{c}, \sigma) = \boldsymbol{\epsilon}(\boldsymbol{z}, \varnothing, \sigma) + \gamma(\boldsymbol{\epsilon}(\boldsymbol{z}, \boldsymbol{c}, \sigma) - \boldsymbol{\epsilon}(\boldsymbol{z}, \varnothing, \sigma))$, with guidance $\gamma = 4$.

To strike a balance between preserving the original structure in the edited regions and being faithful to the new condition, we mix the scores adaptively depending on the step in the backward diffusion process:

$$\texttt{mix}(\hat{\boldsymbol{\epsilon}}_i, \boldsymbol{\epsilon}_i, i) := \hat{\boldsymbol{\epsilon}}_i + (\boldsymbol{\epsilon}_i - \hat{\boldsymbol{\epsilon}}_i)\left(\frac{i}{N}\right)^{\alpha}. \tag{3}$$

The influence of the original (unconditional) noise estimates $\boldsymbol{\epsilon}_i$ decreases from 1 to almost 0 over the N steps. α controls how fast the mixing factor decays, and it is set to 0.3 in our experiments. In order to minimize the alteration of the texture outside the editing region, we set the mixing factor to 0 for the background (outside the editing mask).

3.4.2 Feature transfer.

Our noise-mixing editing method lends itself to another important capability: feature transfer. We demonstrate this function in Fig. 10, where we transfer features from their native class to another textures class from MVTec, as well as to a real image of the author's desk. To transfer a feature from a texture class to a target image that is out-of-distribution (OOD), we first invert the image using the trained diffusion model. When the image is very different from the training distribution, the inversion process needs more steps to reconstruct the image with high accuracy; we use 250 in our experiments. We then perform the steps analogous to image editing, mixing the noise estimates from inversion with the predictions obtained under the desired label map $\boldsymbol{c}$. Thanks to the classifier-free guidance mechanism, the direction of the noise estimates incorporates the difference between the feature and the normal texture, which minimizes the contamination of normal appearance from the training texture to the OOD target image.

3.5 Stationary infinite texture generation

Our method supports generating images at arbitrary resolutions and aspect ratios. A naive approach to high-resolution synthesis would be to simply run the denoising model $\hat{\boldsymbol{\epsilon}}(\boldsymbol{z}, \boldsymbol{c}, \sigma)$ starting with a noise tensor $\boldsymbol{z}_N$ significantly larger than the size of the training images. This approach does not work on generic image synthesis [Haji-Ali et al. 2024; He et al. 2023] because it fails to retain the relationship between elements at a global level; however, textures are defined by local structures, which are well reproduced by the diffusion model even at a resolution well-outside the training range.

This trivial solution works well for image-sizes up to 2048×2048, after which the inference requires more than 24GB of VRAM. In general, the memory requirement of this approach scales poorly

Style Large generated texture

Fig. 8. Synthesized 1024 × 8192 textures. The top texture uses pure white noise; the middle image is obtained by copying the LF components; the third image is obtained using our noise-uniformization technique. Note that white noise yields a non-stationary texture that deviates from the style-prototype.

with size, making it difficult to apply to larger textures. Taking inspiration from MultiDiffusion [Bar-Tal et al. 2023], we develop a window-based infinite texture generation process that uses constant VRAM. The idea consists of simultaneously denoising multiple overlapping patches from the larger input z_t. The predicted noise for the overlapping regions is averaged across the multiple function calls, effectively harmonizing the different denoising trajectories. Wang et al. [2024] used a similar method for generating an infinite texture of a given exemplar. A limitation of this approach is that pixels across distant patches are denoised independently, and nothing pushes the texture to be globally consistent.

We note that the high-level directions of variation in a texture class (such as color change or pattern density) are encoded in the low-frequency (LF) components of the initial noise image (also observed by Chung et al. [2024] and Everaert et al. [2024]). We highlight this relationship in Fig. 7 by showing the output of the diffusion model for independently-sampled noise latents and by comparing them to different manipulations of the LF components of the latent maps. By copying the mean of the noise from a prototype texture (first column), it can be observed that the style of the 4 outputs become more consistent. Copying further LF components (by subtracting a blurred version of itself and adding the blurred prototype) yields increasingly consistent patterns. More precisely, for rows 3 and 4 we use a Lanczos low-pass filter with a cutoff frequency f_c of 0.1 and 0.2 respectively. Using a large cutoff for the filter results in very consistent textures; however, the LF components of the generated images also become similar, creating repetitive patterns. This becomes obvious when generating large textures (see Fig. 8).

To alleviate this issue, we make the observation that obtaining consistent textures does not require identical low frequencies, but only a similar spectral distribution. This follows the intuition that diffusion models perform approximate spectral autoregression [Dieleman 2024]. Therefore, our noise-uniformization method consists of replacing the LF components of the noise with a random permutation of the LF of a prototype which controls the style. To be exact, the noise tensor z_N that follows the style of a different noise prototype p_N is computed as $z_N = w - \texttt{blur}(w) + \texttt{upscale}(\texttt{shuffle}(\texttt{downscale}(\texttt{blur}(p_N))))$, where w is pure white noise, and `shuffle` a random permutation of pixels (cf. Fig. 8).

To improve the efficiency of the sliding-window denoising, we set upper and lower bounds on the number of overlapping pixels in adjacent patches. The patches used by the diffusion model randomly shift within these bounds for each denoising step, which avoids the formation of seams at the edges of the patches. By allowing the shifted windows to wrap around the texture we obtain an additional quality: the denoised texture can be seamlessly tiled. Please see Fig. 17 in the supplementary material for tiled texture samples. We note that our randomly-shifted sliding windows bear similarities with the strategies employed by Wang et al. [2024] and Vecchio et al. [2024] (noise rolling). We investigate the synergy between the latter and our noise uniformization in the supplementary (Fig. 16).

4 Experiments

We firstly evaluate our pipeline on the 5 texture classes from the MVTec dataset [Bergmann et al. 2021], designed for anomaly detection. There are approximately 100 images for each texture (around 20 images per anomaly type). Additionally, we stress-test our method with smaller image sets; using a handheld phone camera, we collect 9 different textures ranging from 1 to 20 images. Notably, 3 of the textures are single-image. We include the exact counts and representative samples for each texture in the supplementary (S1).

In Fig. 4, we compare different anomaly-segmentation approaches. The labels obtained using our method satisfy the most important criteria for conditional synthesis: semantically similar prominent features are grouped into the same class; the detected regions are compact, with relatively little (spatial) noise; and the background/normal pixels are mapped to a unique class. On the other hand, the image-level anomaly clustering method BlindLCA produces noisy labels when used for pixel-level segmentation. Generic unsupervised semantic segmentation methods (such as STEGO [Hamilton et al. 2022]) fail to disentangle the anomalies due to their rarity.

Fig. 2 includes images generated by our method. Please note that the label maps support painting multiple features with various shapes and sizes. In Fig. 18 (supplementary material) we show that this holds even for the MVTecAD dataset, where, except for wood, all textures seen during training have a single anomaly-type per image. That is, the generative model is able to gracefully generalize from single features to multi-feature painting on textures.

We provide a qualitative comparison between our approach and various methods that are adapted for our task in Fig. 9. Since our pipeline is the first to offer a complete workflow for painting rare features from an unlabeled collection of textures, that evaluation includes methods that are able to paint features, guided by given input masks. We do not compare to generic prompt-based image editing [Brooks et al. 2023; Lai et al. 2025; Sun et al. 2024], as we are interested in painting features as extracted from specific exemplars. For the baselines designed to use a single image, we select one texture from our training set that contains a feature with a relatively similar shape to the target (see first column) and use the ground-truth mask of the prominent feature. The significant previous work of Hu et al. [2024] was not included here because the approach proved unsuitable for painting small and subtle features; a representative failure case is shown in Fig. 23 of the supplementary.

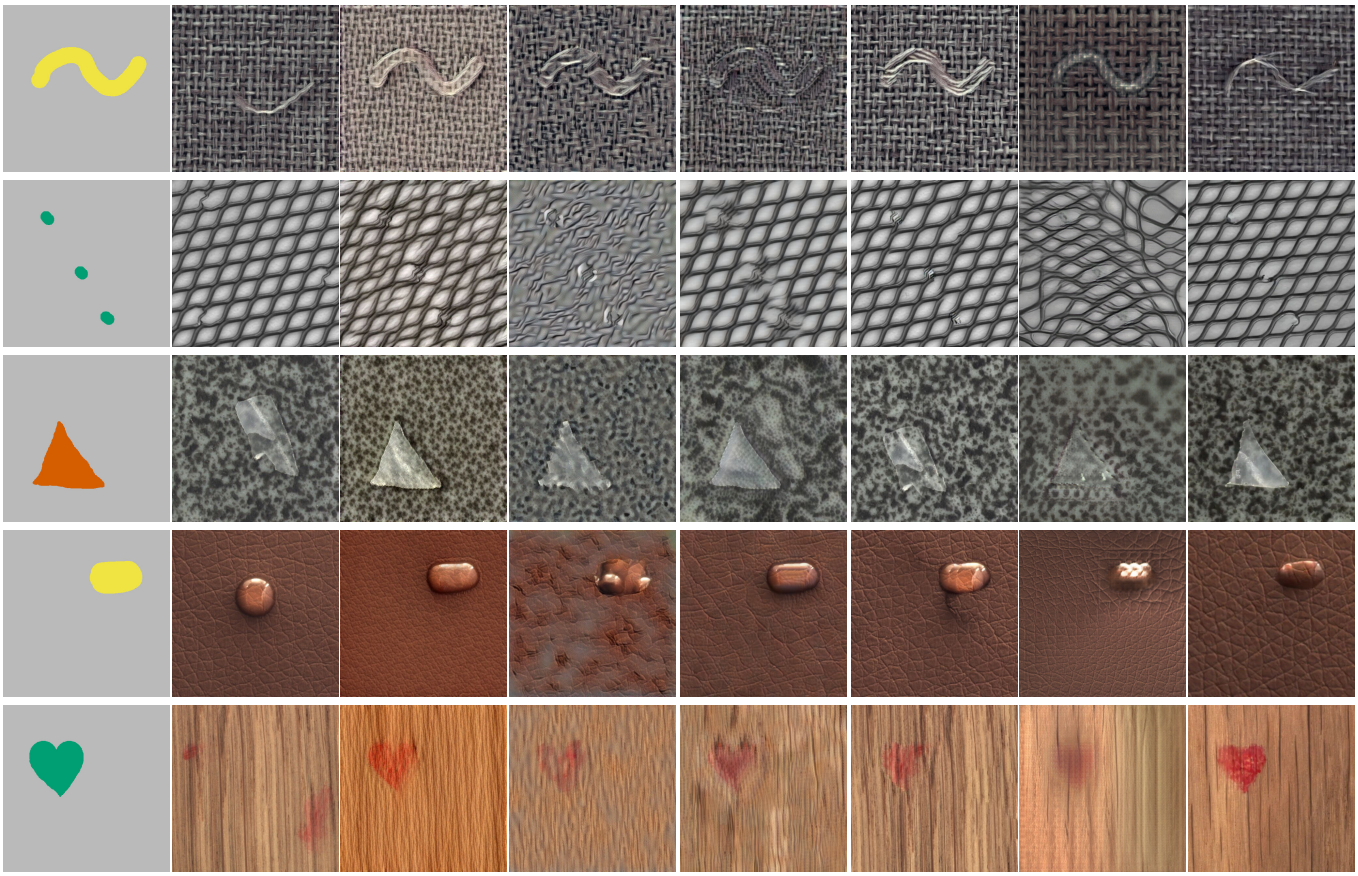

Fig. 9. Qualitative comparison for label-conditioned texture generation.

Fig. 10. Feature transfer to real images of texture classes non-native to each feature. Due to the novel surrounding, cues on scale are missing, so feature scale may vary across transfers, particularly visible with the crack feature.

4.1 Editing results

To demonstrate the editing capabilities of our approach, we use anomaly-free images that were not seen during training - neither for the contrastive learning, nor for the diffusion model. We illustrate firstly the intermediate steps in our texture-editing method in Fig. 5. The resulting image is very close to the original texture outside of the edited region; as shown in the supplementary (Fig. 25), virtually all existing differences are caused by the limited capacity of the pretrained variational autoencoder from Stable Diffusion [Rombach et al. 2022]. Notably, thanks to our noise-mixing method, the edited region is integrated seamlessly in the texture, matching the previous appearance underneath the roughened region and at the boundaries. We include more editing results in Fig. 6.

Thanks to the seamless integration of the edited region, our noise-mixing lends itself to high-resolution editing at interactive rates (see video in the supplementary material). This is achieved by running the diffusion only for the texture patch that contains the feature label to be painted. To this end, we save the diffusion trajectory during generation or inversion and only use the slice of the noise estimates that corresponds to the edited patch. Saving the noise history for the diffusion process makes it possible to forego inversion for each edit,

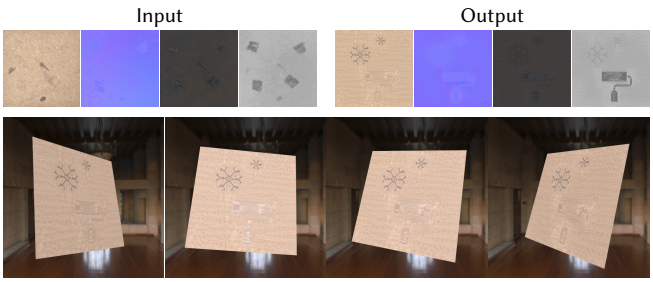

Fig. 11. Synthesizing new material maps using our pipeline from a single SVBRDF. Please see the supplementary GIF for a high-resolution rendering.

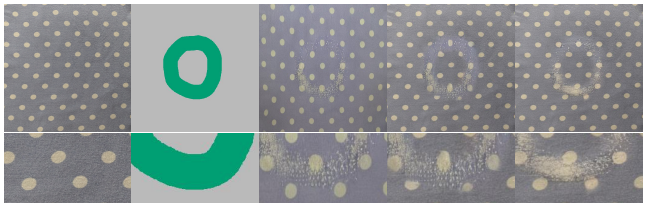

Fig. 12. Ablation of noise-mixing. Note that our edit better preserves the structure of the original texture, while maintaining the realism of the feature.

Table 1. Ablation of our feature detection and clustering components. Metrics are computed under optimal matching to the ground-truth classes.

	Accuracy ↑	IoU ↑	F1-score ↑
Ours w/o equation (1)	0.963	0.320	0.407
Ours w/o stratified negatives	0.794	0.367	0.451
Ours	**0.978**	**0.473**	**0.566**

thus reducing the time to about 1.5 seconds per edit. More details and high-resolution examples are included in the supplementary, S6.

4.2 Synthesizing materials

So far we have described our approach and demonstrated its capabilities in terms of plain RGB textures. All our components, however, easily extend to other modalities, such as material maps for synthesizing SVBRDFs. We include in Fig. 11 a simple example, where the input data consists of a single SVBRDF, captured with a smartphone using MaterialGAN [Guo et al. 2020].

5 Ablations

We ablate our thresholding method, described in Eq. (1), and our stratified selection of negative pairs for contrastive learning. The results of this experiment are summarized in Table 1 and show that the introduced mechanisms are vital for an accurate segmentation.

Our noise-mixing technique is ablated in Fig. 12, by comparing our method with two variants. In the first case, we simply use the inverted noise as input for the diffusion model with the new conditioning map $\boldsymbol{c}$. In the second, we also constrain the latents outside the edit to perfectly match the original image. Both variants unfavorably alter the texture beyond the desired edit. Additionally, we include in the suppl. material (S11) a discussion of our choice of diffusion model used as backbone for the feature painting.

6 Limitations

Naturally, the realism of the generated results depends on the plausibility of the input mask. Fig. 19 in the supplementary shows an exhaustive combination of input masks and generated texture features, demonstrating the versatility, but also limitations of the method. Overall, we believe that it shows that our method still extrapolates commendably beyond the mask shapes observed during training. A different limitation is that our pipeline is not end-to-end trainable, meaning that imperfect segmentations from our automatic feature clustering can cause different features to be mapped to the same label. In practice, this causes the model to *choose* what feature is being painted based on the shape of the conditioning label. Finally, while our method has fast inference times, training the diffusion model for a new texture class can take 6 to 12 hours.

7 Conclusion

Our framework learns and disentangles the stationary characteristics from the prominent features given only a small collection of textures. After training, we are able to generate tileable textures of arbitrary size and to paint features on similar and dissimilar images with realistic transitions. These functionalities are bundled in a single model that can be controlled interactively for iterative authoring. While the main focus of our work is asset creation, the utility of the method extends to various areas, such as, image augmentations, long-tail image generation, and generating fake anomalies for self-supervised anomaly detection. Moreover, our editing method and noise uniformization algorithm can be leveraged in more general contexts, as shown in S2 (supplementary material).

Acknowledgments

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 956585.

References

Adobe. 2023a. Adobe Photoshop clone tool. https://helpx.adobe.com/photoshop/using/tool-techniques/clone-stamp-tool.html.

Adobe. 2023b. Adobe substance 3D painter clone tool. https://helpx.adobe.com/substance-3d-painter/painting/tool-list/clone-tool.html.

Andrei-Timotei Ardelean and Tim Weyrich. 2024a. High-fidelity zero-shot texture anomaly localization using feature correspondence analysis. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 1134–1144.

Andrei-Timotei Ardelean and Tim Weyrich. 2024b. Blind Localization and Clustering of Anomalies in Textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*. 3900–3909.

Juan Miguel Bajo, Claudio Delrieux, and Gustavo Patow. 2021. Physically inspired technique for modeling wet absorbent materials. *The Visual Computer* 37, 8 (2021), 2053–2068.

Djalma Bandeira and Marcelo Walter. 2009. Synthesis and Transfer of Time-Variant Material Appearance on Images. In *XXII Brazilian Symposium on Computer Graphics and Image Processing*. 32–39. https://doi.org/10.1109/SIBGRAPI.2009.38

Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. 2023. MultiDiffusion: fusing diffusion paths for controlled image generation. In *Proceedings of the 40th International Conference on Machine Learning*. Article 74, 16 pages.

Kilian Batzner, Lars Heckler, and Rebecca König. 2024. Efficientad: Accurate visual anomaly detection at millisecond-level latencies. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 128–138.

Rachele Bellini, Yanir Kleiman, and Daniel Cohen-Or. 2016. Time-varying weathering in texture space. *ACM Transactions on Graphics (TOG)* 35 (2016), 1 – 11.

Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. 2021. The MVTec anomaly detection dataset: a comprehensive real-world dataset for unsupervised anomaly detection. *International Journal of Computer Vision* 129, 4 (2021), 1038–1059.

Urs Bergmann, Nikolay Jetchev, and Roland Vollgraf. 2017. Learning texture manifolds with the Periodic Spatial GAN. In *Proceedings of the 34th International Conference on Machine Learning*. 469–477.

Tim Brooks, Aleksander Holynski, and Alexei A. Efros. 2023. InstructPix2Pix: Learning to Follow Image Editing Instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

Li-Yu Chen, I-Chao Shen, and Bing-Yu Chen. 2021a. Guided Image Weathering using Image-to-Image Translation. *SIGGRAPH Asia 2021 Technical Communications* (2021).

Xinlei Chen, Saining Xie, and Kaiming He. 2021b. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*. 9640–9649.

Yanyun Chen, Lin Xia, Tien-Tsin Wong, Xin Tong, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2005. Visual simulation of weathering by y-ton tracing. *ACM Trans. Graph.* 24, 3 (2005), 1127–1133. https://doi.org/10.1145/1073204.1073321

Jiwoo Chung, Sangeek Hyun, and Jae-Pil Heo. 2024. Style injection in diffusion: A training-free approach for adapting large-scale diffusion models for style transfer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8795–8805.

M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. 2014. Describing Textures in the Wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Hanqiu Deng and Xingyu Li. 2022. Anomaly detection via reverse distillation from one-class embedding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 9737–9746.

Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems* 34 (2021), 8780–8794.

Sander Dieleman. 2024. Diffusion is spectral autoregression. https://sander.ai/2024/09/02/spectral-autoregression.html

Julie Dorsey and Pat Hanrahany. 2006. Modeling and rendering of metallic patinas. In *ACM SIGGRAPH 2006 Courses*. https://doi.org/10.1145/1185657.1185722

Shiyin Du and Ying Song. 2023. Multi-exemplar-guided image weathering via texture synthesis. *The Visual Computer* 39, 8 (2023), 3691–3699. https://doi.org/10.1007/s00371-023-02944-5

Alexei A. Efros and William T. Freeman. 2001. Image quilting for texture synthesis and transfer. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*. ACM Press, 341–346. https://doi.org/10.1145/383259.383296

Ariel Elnekave and Yair Weiss. 2022. Generating natural images with direct patch distributions matching. In *European Conference on Computer Vision*. Springer, 544–560.

Martin Nicolas Everaert, Athanasios Fitsios, Marco Bocchio, Sami Arpa, Sabine Süsstrunk, and Radhakrishna Achanta. 2024. Exploiting the signal-leak bias in diffusion models. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 4025–4034.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2414–2423.

GIMP Development Team; Project GNU. 2024. Gimp clone tool. https://docs.gimp.org/2.10/en/gimp-tool-clone.html.

Niv Granot, Ben Feinstein, Assaf Shocher, Shai Bagon, and Michal Irani. 2022. Drop the gan: In defense of patches nearest neighbors as single image generative models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 13460–13469.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. 2020. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems* 33 (2020), 21271–21284.

Julia Guerrero-Viu, Milos Hasan, Arthur Roullier, Midhun Harikumar, Yiwei Hu, Paul Guerrero, Diego Gutierrez, Belen Masia, and Valentin Deschaintre. 2024. Texsliders: Diffusion-based texture editing in clip space. In *ACM SIGGRAPH 2024 Conference Papers*. 1–11.

Yu Guo, Cameron Smith, Miloš Hašan, Kalyan Sunkavalli, and Shuang Zhao. 2020. MaterialGAN: Reflectance Capture using a Generative SVBRDF Model. *ACM Trans. Graph.* 39, 6 (2020), 254:1–254:13.

Saeed Hadadan, Benedikt Bitterli, Tizian Zeltner, Jan Novák, Fabrice Rousselle, Jacob Munkberg, Jon Hasselgren, Bartlomiej Wronski, and Matthias Zwicker. 2025. Generative detail enhancement for physically based materials. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*. 1–11.

Moayed Haji-Ali, Guha Balakrishnan, and Vicente Ordonez. 2024. ElasticDiffusion: Training-free Arbitrary Size Image Generation through Global-Local Content Separation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 6603–6612.

Mark Hamilton, Zhoutong Zhang, Bharath Hariharan, Noah Snavely, and William T. Freeman. 2022. Unsupervised Semantic Segmentation by Distilling Feature Correspondences. In *International Conference on Learning Representations*.

Guoqing Hao, Satoshi Iizuka, Kensho Hara, Hirokatsu Kataoka, and Kazuhiro Fukui. 2023. Natural Image Decay With a Decay Effects Generator. *IEEE Access* 11 (2023), 120402–120418.

Yingqing He, Shaoshu Yang, Haoxin Chen, Xiaodong Cun, Menghan Xia, Yong Zhang, Xintao Wang, Ran He, Qifeng Chen, and Ying Shan. 2023. Scalecrafter: Tuning-free higher-resolution visual generation with diffusion models. In *International Conference on Learning Representations*.

Wu Hecong. 2024. *ControlLoRA Version 3: LoRA Is All You Need to Control the Spatial Information of Stable Diffusion*. https://github.com/HighCWu/control-lora-3

Eric Heitz, Kenneth Vanhoey, Thomas Chambon, and Laurent Belcour. 2021. A sliced wasserstein loss for neural texture synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9412–9420.

Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-or. 2023. Prompt-to-Prompt Image Editing with Cross-Attention Control. In *International Conference on Learning Representations*.

Aaron Hertzmann, Charles E. Jacobs, Nuria Oliver, Brian Curless, and David H. Salesin. 2001. Image analogies. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '01)*. Association for Computing Machinery, New York, NY, USA, 327–340. https://doi.org/10.1145/383259.383295

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems* 30 (2017).

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

Jonathan Ho and Tim Salimans. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.

Anita Hu, Nishkrit Desai, Hassan Abu Alhaija, Seung Wook Kim, and Maria Shugrina. 2024. Diffusion Texture Painting. In *ACM SIGGRAPH 2024 Conference Papers*. 1–12.

Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-Rank Adaptation of Large Language Models. In *International Conference on Learning Representations*.

Satoshi Iizuka, Yuki Endo, Yoshihiro Kanamori, and Jun Mitani. 2016. Single Image Weathering via Exemplar Propagation. *Computer Graphics Forum* 35, 2 (2016), 501–509. https://doi.org/10.1111/cgf.12850

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. 2017. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1125–1134.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. 2022. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems* 35 (2022), 26565–26577.

Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. 2024. Analyzing and improving the training dynamics of diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 24174–24184.

Juyeon Ko, Inho Kong, Dogyun Park, and Hyunwoo J. Kim. 2024. Stochastic Conditional Diffusion Models for Robust Semantic Image Synthesis. In *Proceedings of the 41st*

International Conference on Machine Learning, Vol. 235. 24932–24963.

Zhangyu Lai, Yilin Lu, Xinyang Li, Jianghang Lin, Yansong Qu, Liujuan Cao, Ming Li, and Rongrong Ji. 2025. AnomalyPainter: Vision-Language-Diffusion Synergy for Zero-Shot Realistic and Diverse Industrial Anomaly Synthesis. *arXiv preprint arXiv:2503.07253* (2025).

Aodong Li, Chen Qiu, Marius Kloft, Padhraic Smyth, Maja Rudolph, and Stephan Mandt. 2023. Zero-Shot Anomaly Detection via Batch Normalization. In *Thirty-seventh Conference on Neural Information Processing Systems*.

Xurui Li, Ziming Huang, Feng Xue, and Yu Zhou. 2024. MuSc: Zero-Shot Industrial Anomaly Classification and Segmentation with Mutual Scoring of the Unlabeled Images. In *The Twelfth International Conference on Learning Representations*.

Chieh Hubert Lin, Hsin-Ying Lee, Yen-Chi Cheng, Sergey Tulyakov, and Ming-Hsuan Yang. 2021. InfinityGAN: Towards Infinite-Pixel Image Synthesis. In *International Conference on Learning Representations*.

Hongbo Liu, Kai Li, Xiu Li, and Yulun Zhang. 2022. Unsupervised Anomaly Detection with Self-Training and Knowledge Distillation. In *IEEE International Conference on Image Processing*. 2102–2106. https://doi.org/10.1109/ICIP46576.2022.9897777

Youquan Liu, Yanyun Chen, Wen Wu, Nelson Max, and Enhua Wu. 2012. Physically based object withering simulation. *Computer Animation and Virtual Worlds* 23, 3-4 (2012), 395–406.

Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. 2022. RePaint: Inpainting Using Denoising Diffusion Probabilistic Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 11461–11471.

Michal Lukáč, Jakub Fišer, Paul Asente, Jingwan Lu, Eli Shechtman, and Daniel Sỳkora. 2015. Brushables: Example-based Edge-aware Directional Texture Painting. *Computer Graphics Forum* 34, 7 (2015), 257–267.

Michal Lukáč, Jakub Fišer, Jean-Charles Bazin, Ondřej Jamriška, Alexander Sorkine-Hornung, and Daniel Sỳkora. 2013. Painting by feature: Texture boundaries for example-based image creation. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 1–8.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748* (2018).

Nobuyuki Otsu et al. 1975. A threshold selection method from gray-level histograms. *Automatica* 11, 285-296 (1975), 23–27.

Zhihong Pan, Riccardo Gherardi, Xiufeng Xie, and Stephen Huang. 2023. Effective real image editing with accelerated iterative diffusion inversion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 15912–15921.

Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. Semantic Image Synthesis with Spatially-Adaptive Normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Ashay Patel, Petru-Daniel Tudosiu, Walter HL Pinaya, Mark S Graham, Olusola Adeleke, Gary Cook, Vicky Goh, Sebastien Ourselin, and M Jorge Cardoso. 2023. Self-supervised anomaly detection from anomalous training data via iterative latent token masking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2402–2410.

Lincoln Ritter, Wilmot Li, Brian Curless, Maneesh Agrawala, and David Salesin. 2006. Painting With Texture. In *Rendering Techniques*. 371–376.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. 2015. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI*. Springer, 234–241.

Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. 2022. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 14318–14328.

Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. 2023. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 22500–22510.

Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. 2019. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE/CVF international conference on computer vision*. 4570–4580.

Maria Shugrina, Chin-Ying Li, and Sanja Fidler. 2022. Neural Brushstroke Engine: Learning a Latent Style Space of Interactive Drawing Tools. *ACM Transactions on Graphics (TOG)* 41, 6 (2022).

Kihyuk Sohn, Jinsung Yoon, Chun-Liang Li, Chen-Yu Lee, and Tomas Pfister. 2023. Anomaly clustering: Grouping images into coherent clusters of anomaly types. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 5479–5490.

Jiaming Song, Chenlin Meng, and Stefano Ermon. 2021a. Denoising Diffusion Implicit Models. In *International Conference on Learning Representations*.

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

Han Sun, Yunkang Cao, and Olga Fink. 2024. Cut: A controllable, universal, and training-free visual anomaly generation framework. *arXiv preprint arXiv:2406.01078* (2024).

Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. 2022. Resolution-robust large mask inpainting with fourier convolutions. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2149–2159.

Giuseppe Vecchio, Rosalie Martin, Arthur Roullier, Adrien Kaiser, Romain Rouffet, Valentin Deschaintre, and Tamy Boubekeur. 2024. ControlMat: A Controlled Generative Approach to Material Capture. *ACM Trans. Graph.* 43, 5, Article 164 (sep 2024), 17 pages. https://doi.org/10.1145/3688830

Bram Wallace, Akash Gokul, and Nikhil Naik. 2023. Edict: Exact diffusion inversion via coupled transformations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 22532–22541.

Jiaping Wang, Xin Tong, Stephen Lin, Minghao Pan, Chao Wang, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2006. Appearance manifolds for modeling time-variant appearance of materials. *ACM Transactions on Graphics* 25, 3 (2006), 754–761. https://doi.org/10.1145/1141911.1141951

Weilun Wang, Jianmin Bao, Wengang Zhou, Dongdong Chen, Dong Chen, Lu Yuan, and Houqiang Li. 2022a. Sindiffusion: Learning a diffusion model from a single natural image. *arXiv preprint arXiv:2211.12445* (2022).

Yifan Wang, Aleksander Holynski, Brian L Curless, and Steven M Seitz. 2024. Infinite Texture: Text-guided High Resolution Diffusion Texture Synthesis. *arXiv preprint arXiv:2405.08210* (2024).

Zhizhong Wang, Lei Zhao, Haibo Chen, Ailin Li, Zhiwen Zuo, Wei Xing, and Dongming Lu. 2022b. Texture reformer: Towards fast and universal interactive texture transfer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 36. 2624–2632.

Yifeng Xu, Zhenliang He, Shiguang Shan, and Xilin Chen. 2025. CtrLoRA: An Extensible and Efficient Framework for Controllable Image Generation. In *International Conference on Learning Representations*.

Su Xuey, Jiaping Wang, Xin Tong, Qionghai Dai, and Baining Guo. 2008. Image-based Material Weathering. *Computer Graphics Forum* 27, 2 (2008), 617–626. https://doi.org/10.1111/j.1467-8659.2008.01159.x

Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O Arik, Chen-Yu Lee, and Tomas Pfister. 2022. Self-supervise, Refine, Repeat: Improving Unsupervised Anomaly Detection. *Transactions on Machine Learning Research* (2022).

Ning Yu, Connelly Barnes, Eli Shechtman, Sohrab Amirghodsi, and Michal Lukac. 2019. Texture mixer: A network for controllable synthesis and interpolation of texture. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 12164–12173.

Sergey Zagoruyko and Nikos Komodakis. 2016. Wide Residual Networks. In *British Machine Vision Conference*. Article 87, 12 pages. https://doi.org/10.5244/C.30.87

Guoqiang Zhang, Jonathan P Lewis, and W Bastiaan Kleijn. 2025. Exact diffusion inversion via bidirectional integration approximation. In *European Conference on Computer Vision*. Springer, 19–36.

Jie Zhang, Masanori Suganuma, and Takayuki Okatani. 2024. That's BAD: blind anomaly detection by implicit local feature clustering. *Machine Vision and Applications* 35, 2 (2024), 31.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 3836–3847.

Yang Zhou, Kaijian Chen, Rongjun Xiao, and Hui Huang. 2023. Neural texture synthesis with guided correspondence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18095–18104.

Yang Zhou, Rongjun Xiao, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. 2024. Generating Non-Stationary Textures using Self-Rectification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 7767–7776.