

Diffusion-based Image Mosaics

Lars Doyle
Carleton University
Ottawa, Canada
larsdoyle@email.carleton.ca

David Mould
Carleton University
Ottawa, Canada
mould@scs.carleton.ca



Figure 1: Diffusion-based photomosaics are created by generating target tiles with a diffusion model. In our work, we demonstrate how a pre-trained diffusion model can be adapted to this purpose.

Abstract

Image mosaics have traditionally been automated by employing a search method to match source images to locations in a target layout. However, search can be replaced by a generative process that generates custom tiles. The challenge is twofold: the generated images must match a target color layout, while maintaining the image quality of unconditional generation. Here, we use a diffusion model within a training-free, optimization-based framework. Users select a theme that comprises a dataset of text prompts, which guide the generation of each mosaic tile. Rather than fine-tuning, we introduce a custom loss function that balances two competing objectives: ensuring that the local color of the generated tile matches that of the target, while preserving sufficient visual diversity and detail. Our approach requires no model training or large image databases, making it broadly accessible and adaptable to arbitrary themes and target images.

CCS Concepts

• **Computing methodologies** → **Non-photorealistic rendering; Image processing.**

Keywords

Image Mosaics, Photomosaics, Image Generation, Diffusion Models

ACM Reference Format:

Lars Doyle and David Mould. 2026. Diffusion-based Image Mosaics. In *Proceedings of Graphics Interface (GI '26)*. ACM, New York, NY, USA, 10 pages. <https://doi.org/XXXXXXX.XXXXXXX>

1 Introduction

Image mosaics have origins as early as the Renaissance, when Giuseppe Arcimboldo composed paintings where a subject is depicted by a collection of unrelated objects such as fruits and vegetables or fish. Viewers can engage with the artwork on two levels, perceiving both the larger composition and its individual elements.

The modern photomosaic is characterized by a rectangular grid of small photographs that form a larger composition. When viewed from a distance, the individual photographs blend into a larger composition. Traditional photomosaic algorithms [4, 17, 31] rely on searching through image databases to find photographs matching each target region. Here, we take advantage of diffusion models that generate images from text prompts, allowing us to create each sub-image on demand.

Gathering image datasets for photomosaics presents significant practical challenges. Typically, a designer envisions a thematic relationship between the input image and its constituent sub-images. In most cases, specialized datasets must be manually constructed, a time-consuming process that must be repeated for each new concept. Furthermore, certain imagined themes may be difficult or impossible to source. Since the quality of a search-based photomosaic depends critically on dataset size and diversity, these practical limitations are constraining. Generative approaches enable designers to explore arbitrary themes without the burden of dataset construction. Figure 2 shows a mosaic created on the theme of ‘mythological beasts’. It would be difficult to source a large, varied dataset on this subject while maintaining style consistency across images.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

GI '26, Kitchener-Waterloo, ON

© 2026 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/2026/06
<https://doi.org/XXXXXXX.XXXXXXX>



Figure 2: An example of an image mosaic generated using the ‘mythological beasts’ theme.

Previous authors [31, 44] have discussed an accuracy/discernibility tradeoff when generating a photomosaic. Orchard and Kaplan [31] provide the definition: “Accuracy seeks to reproduce the target image as closely as possible; discernibility seeks to ensure that each of the source tiles is legible.” Enhanced accuracy is possible through altering the original sub-image colors but makes them less discernible. In diffusion mosaics, images are directly generated to align with the desired color layout.

Our contributions are as follows:

- We propose an optimization-based method for guiding pre-trained diffusion models to generate images that fit a target color layout.
- We design a pipeline that creates image mosaics from user-selected themes. This pipeline automatically selects from a collection of LLM-generated prompts, based on characteristics in the underlying region of the target image, that are then used to guide individual mosaic tile generation.

2 Related Work

While other mosaic types exist, such as crystallization mosaics [14, 16, 19], ancient mosaics [13, 15, 20], and puzzle image mosaics [25, 35, 43], we are primarily concerned with photomosaics. Battiato et al. provide a survey of other mosaic types [2].

2.1 Photomosaics

Creating a photomosaic [38] has been treated as a search problem: given a large dataset of candidate images, find the images that best match corresponding tiles in the underlying target image using some feature representation. (It is common to refer to both the small source images, and their corresponding sub-region of the target image, as tiles.) Using the pixel colors directly [31, 41] can produce reasonable accuracy. However, some authors have accelerated search with more compact comparisons: Finkelstein and Range [17] use a wavelet signature; Pavic et al. [32] use polynomial

descriptors; Di Blasi [4] subdivide tiles into 3×3 grids and obtain a 27-dimensional vector from the RGB color averages.

The bottleneck in photomosaic creation is the image search used at each tile. A linear search over the dataset can be improved with a tree-based nearest neighbor search strategy. Once the data structure is built for a particular dataset, it can be reused for subsequent mosaics. An example is seen in the work of Di Blasi [4] who employs an Antipole tree [6] to accelerate image search. Orchard and Kaplan [31] introduce searching over all possible shifts within a source tile. This additional search is made tractable by performing a convolution in the frequency domain.

The image search problem can also be extended to include source-tile color modifications [17, 31] to improve accuracy. However, the integrity of the source images may be lost if color-shifting is applied too aggressively. In our diffusion-based image mosaics, color shifting is unnecessary since images are generated to naturally align with target colors. As a final note, we also mention that a uniform tile grid is not mandatory. For example, Pavic et al. [32] demonstrate adaptive tiling, using larger tiles in uniform regions and smaller tiles where greater detail is needed.

2.2 Diffusion-based image generation and editing

Recently, diffusion models [23, 39] are able to generate high-quality images closely resembling those in their training distributions. Diffusion models were then extended to conditional models where inputs, such as text and external images, are used to guide the generation process, with applications including style transfer [9, 10], texture synthesis [47], and image editing [7, 21, 42].

For diffusion-based image mosaics, we must produce images (tiles) conforming to a target structure. One approach involves training adapters that condition existing diffusion models on edge maps, color schemes, and other guidance signals. ControlNet [46] pioneered this approach by training additional neural network modules to condition the diffusion process on structural inputs such as Canny edges or depth maps. T2I-Adapter [29] similarly extracts guidance features from input images to modify the UNet backbone, including a spatial color-palette conditioning mechanism among its various guidance types. IP-Adapter [45] introduced a decoupled cross-attention mechanism that operates in parallel with text-based cross-attention in the UNet, enabling image-prompt conditioning. While these methods require training, Shum et al. [37] propose a zero-shot approach that follows an input color palette.

3 Diffusion preliminaries

Diffusion models are designed to create images from noise, consisting of both forward and reverse processes. In the forward process, noise is gradually added to an image x_0 over a series of timesteps $t \in [0, T]$ until only Gaussian noise is present in the terminal state x_T . Of greater interest, this gradual noising can be reversed by means of a sampling process and a trained noise approximator ϵ_θ . In particular, DDIM [40] sampling predicts an image through T sampling steps:

$$x_{t-1} = \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0 + \sqrt{1 - \bar{\alpha}_{t-1}}\epsilon_\theta(x_t, t), \quad (1)$$

where $\bar{\alpha}_t$ is based on a predefined noise schedule. At each denoising step, $\hat{x}_0$ is the current approximation to the clean image x_0 and is computed as

$$\hat{x}_0 = \frac{x_t - \sqrt{1 - \bar{\alpha}_t} \epsilon_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t}}. \quad (2)$$

Latent diffusion models such as Stable Diffusion [34], perform iterative denoising in a lower-dimension latent space to improve performance. This dimensionality reduction requires an encoder $\mathcal{E}$ and a decoder $\mathcal{D}$ to transform the image x_0 into its latent representation $z_0 = \mathcal{E}(x_0)$ or back into image space $x_0 = \mathcal{D}(z_0)$.

Conditional controls extend diffusion models by allowing users to specify a condition y that controls the generated output. Classifier-free guidance [24], used in *text-to-image* diffusion models, gradually aligns the generated image with a text prompt in a shared encoding space. Classifier guidance [12], by contrast, employs a trained, time-dependent classifier to steer generation toward class-conditional outputs via the classifier gradient. While classifier guidance requires training a classifier for each task (in order to operate on noise-corrupted images), using the clean intermediate predictions from Equation 2 unlocks a variety of training-free loss functions [36]. In the case of latent diffusion, the intermediate result $\hat{z}_0$ requires that it also be passed through the decoder to obtain $\hat{x}_0 = \mathcal{D}(\hat{z}_0)$ before standard image-space losses can be applied. We illustrate the development of the intermediate $\hat{x}_0$ predictions in Figure 3 as they are extracted from the Stable Diffusion model using $T = 50$ timesteps. Notice that even after a single denoising step, an approximate color layout of the final generated image is obtained.

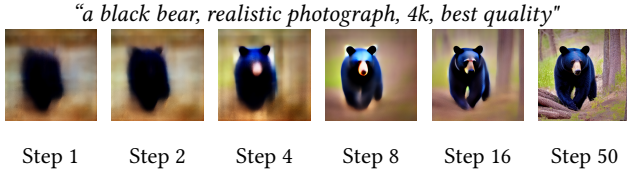


Figure 3: Intermediate predictions for $\hat{x}_0 = \mathcal{D}(\hat{z}_0)$ extracted from the Stable Diffusion [34] model, using the prompt specified at the top. Denoising steps are indicated below each corresponding image.

4 Method

Our method builds upon a frozen *text-to-image* diffusion model. Being training-free and independent of any particular model architecture, our approach allows users to select any diffusion model based on their specific requirements, such as prioritizing output resolution, or a particular image domain such as faces. The final mosaic is assembled from independently generated tiles. While per-tile generation is computationally expensive, current diffusion models on consumer-level GPUs are unable to produce images at sufficient resolution for large-format printing; at 300 DPI, a typical diffusion model output covers only a few square inches, making a tile-based approach necessary.

We propose two main extensions to existing diffusion processes. First, we describe an automated prompt selection process to determine suitable prompts for tile generation (Section 4.1). This

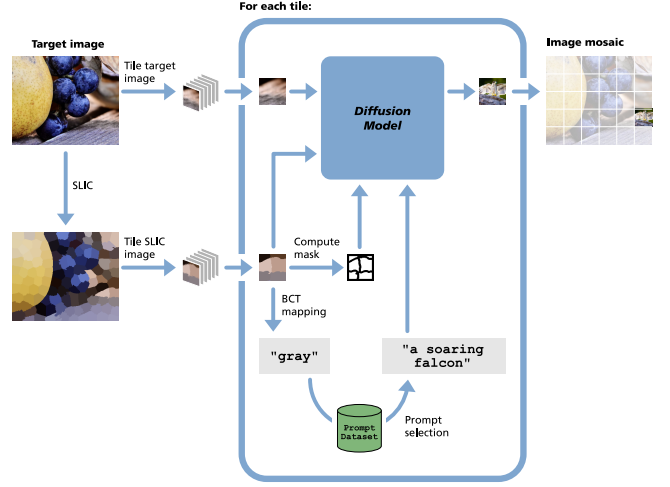


Figure 4: Pipeline of our method.

determination is informed by the underlying pixel values in the target image, combined with a common human color lexicon. Second, we adapt the diffusion model itself for tile generation (Section 4.2). These adaptations guide the denoising process to align each generated tile with the underlying target-image content. Since our method is training-free, it can adjust parameters on a per-tile basis, shifting the accuracy/discernibility tradeoff according to a semantic map.

A high-level overview of our method is the following:

- (1) Compute a tiling of the target image I
- (2) for each tile I_i in I :
 - (a) Generate a prompt p
 - (b) Generate a tile G_i , conditioned on p and the color layout of I_i
- (3) assemble the image mosaic G from all generated tiles G_i

The pipeline diagram in Figure 4, illustrates these steps, with the addition of details that will be explained the following sections.

To maximize image quality, we generate tiles at the diffusion model's native resolution (512×512 in our Stable Diffusion implementation) even when input tiles are smaller. This approach provides a secondary benefit: the output G is also a super-resolution version of I , enabling viewers to zoom in digitally or, for a print reproduction, to step closer and examine each sub-image in detail. Still higher resolutions can be obtained by using a larger diffusion model such as SDXL [33].

4.1 Prompt selection

Generating tiles for an image mosaic requires a prompt for each image. While a variety of images could be generated from a single prompt using different seeds, our experiments suggest that image quality declines without specific, targeted prompts. Furthermore, we expect that users will want a variety of subjects across the mosaic and some degree of control over the tile content. We therefore assign a unique prompt to each tile; however, manually writing a huge number of prompts is impractical. Instead, we organize pre-written prompts into thematic datasets (e.g., birds, flowers), allowing users

to simply select a theme. This provides access to a diverse set of theme-specific prompts without requiring manual input for each tile. In a later refinement step, a user can exercise more control and specify customized prompts on targeted tiles.

We must also consider how well each prompt matches the target tile’s visual content. Color is the most critical attribute; the goal of an image mosaic is for each source tile to resemble the underlying target image at a reasonable viewing distance. We therefore generate tiles containing subjects whose natural colors align with the target tile, achieved by precomputing a mapping from colors to prompts. For instance, a tile with predominantly white regions would be matched with prompts like “a polar bear” or “snow-covered mountains”, helping the generated image align with the target color distribution. This dataset organization has other advantages. Subjects can be reused under multiple color keys (e.g., *bear* associates with white, brown, and black). Our prompts need not specify color attributes, which are imprecise, risk *attention leakage* [8], and might yield contradictory or semantically dubious descriptions (e.g., blue cardinal, green flower). Figure 5 illustrates these issues on a yellowish input tile. Simply prompting “a yellow animal” yields an ambiguous subject, while “a yellow roaring lion” introduces attention leakage; the color attribute attaches to the teeth rather than the lion. The final two results demonstrate the color optimization method from Section 4.2, showing improved color adherence but revealing that including “yellow” in the prompt still produces unnatural hues on the lion’s teeth. The last result addresses this by omitting color from the prompt entirely, delegating the target hue to the optimization process.



Figure 5: Variations on prompt selection are shown with (w), and without (w/o), the color optimization that will be discussed in Section 4.2. From left to right: a target color layout, “a yellow animal” (w/o), “a yellow roaring lion” (w/o), “a yellow roaring lion” (w), “a roaring lion” (w).

Given a set of colors $C = \{c_i\}_{i=1}^N$, we construct a mapping where each color c_i maps to a disjoint set of prompts $f(c_i) = \{p_{i,j}\}_{j=1}^M$. These prompts will be randomly selected, without replacement, such that neighboring tiles do not have matching prompts. We employ ChatGPT to generate this dataset of prompts through a two-step process: first querying it to generate a list of M subjects associated with each color by asking:

Using these colors: <list of N colors>, generate a Python dictionary mapping each color to <M> single-word subjects naturally associated with it. All subjects should follow the theme: <theme>.

To obtain the final prompts, the initial request is followed up with:

Using this dictionary, generate a 3-word prompt for each subject (matching the <theme> theme). Return a Python dictionary with colors as keys. Prompts must suit a text-to-image diffusion model and must not mention the color.

The above process can be completed in seconds. Optionally, the user can further refine and personalize some prompts afterward. Finally, prompts are uniformly appended with “high quality <theme> themed photograph”.

We draw on work by Berlin and Kay [3] to select $N = 11$ basic color terms (BCT) that are named consistently across cultures: *black, white, red, yellow, green, blue, brown, pink, purple, orange, and gray*. These will comprise the color set C .

With the color-to-prompt mapping established, we require an additional mapping from tile pixel values to colors in C . Lindsey and Brown [27] conducted a study in which participants applied labels to a set of color swatches. We use the high-consensus labeled swatches to map tile colors to color terms, computing the Euclidean distance in CIELAB color space and selecting the c_i of the nearest swatch.

Even when using small tile sizes, substantial color variation can occur within a single tile, making the tile mean an unreliable representative color. Instead, we segment the input image I using the SLIC [1] superpixel clustering algorithm, which groups pixels in a 5-dimensional feature space combining color and spatial proximity. Formally:

$$D = \sqrt{\left(\frac{d_c}{m}\right)^2 + \left(\frac{d_s}{g}\right)^2}, \quad (3)$$

where $2g$ is the width of a search grid around a super-pixel centre, and m controls the tradeoff between spatial and color distance. Our implementation employs the adaptive SLICO variant, which sets m automatically per cluster. From the superpixel result S , we compute the mean color of each region in CIELAB space and map each region to a color term c_i . Then, for each tile, we select the c_i associated with the largest number of pixels within that tile. Secondary colors within the tile are not considered; we instead rely on the diffusion model to generate appropriate background content.

4.2 Tile generation

For each tile, our generation process takes as input a target color layout I_t , a corresponding binary mask Ω_t which indicates areas to be optimized with respect to the colors in I_t , a SLIC segmentation map S_t , and a text prompt p . (Typical inputs are illustrated in Figure 6.) As noted in Section 3, each step of the inference process provides an estimate of the final latent variable $\hat{z}_0$. This latent estimate can be passed through the decoder to obtain $\hat{x}_0 = \mathcal{D}(\hat{z}_0)$ in image space. Throughout the denoising process, these intermediate $\hat{x}_0$ predictions are used to steer the latent variable z_t so that it decodes to an image with a similar target-color layout.

We follow established practice [8, 11] and update z_t between denoising steps according to the gradient of a loss $\mathcal{L}$ with respect to z_t . At each update iteration, a modified z'_t is computed as follows:

$$z'_t = z_t - \eta_t \nabla_{z_t} \mathcal{L}, \quad (4)$$



Figure 6: Inputs to our tile generation method include (left to right): a target color layout I_i , a SLIC segmentation map S_i , a binary mask Ω_i , and a text prompt. The resulting output tile is shown on the far right.

using a time-dependent step size η_t following Loizou et al. [28]:

$$\eta_t = \frac{\mathcal{L}}{\|\nabla_{z_t} \mathcal{L}\|^2}. \quad (5)$$

This design is motivated by the observation that gradients will be large in early steps when z_t is mostly noise; the step size should be small in these early stages, to avoid potential instability and excessive oscillation. Conversely, later in the inference process, when gradients are small, a larger step size speeds convergence.

4.2.1 Accuracy and discernibility motivated guidance. If we only valued accuracy, then *mean-squared-error* (MSE) would be an ideal loss function. However, strictly reproducing the target image is not the goal: discernibility necessitates some deviation from the target image. We promote discernibility in two ways: *pixel masking* removes selected pixels from the loss function entirely, and *adaptive quantile-weighted MSE loss* weights pixels in the loss function according to an error map. These techniques are discussed below.

Pixel masking. We define a mask Ω_i by removing edge and boundary pixels from the loss. First, we apply morphological dilation to the boundaries between SLIC regions and exclude the resulting pixels. Since believable generated images rarely align with target image edges, we focus the loss on color regions between SLIC boundaries rather than enforcing strict edge correspondence. Second, we remove a band of pixels around each tile’s outer boundary, preserving the mosaic aesthetic by keeping tiles visibly distinct. A typical mask is shown in Figure 6.

Adaptive quantile-weighted MSE loss. Our loss function is motivated by accommodating, rather than constraining, the generative process. During denoising, some pixels naturally align with target colors as a prompt-consistent image emerges, while others remain poorly matched. Rather than forcing all pixels to contribute equally—which can introduce distortions by overly restricting poorly-aligned color areas—we reduce the loss weight for pixels with large color discrepancies. We control pixel weights by first computing the squared error map E , then partitioning E into P quantiles, each weighted independently in the loss. Importantly, this quantile partitioning is performed within each SLIC region independently, ensuring that all regions contribute to the final loss. In our experiments, we set P to 4. Algorithm 1 details the method for adaptively weighting pixel contributions to the loss.

4.2.2 Hierarchical Image Optimization. In Figure 3, the intermediate $\hat{x}_0$ predictions establish a basic color layout within a few steps, which is then refined to produce more accurate shapes before the remaining process synthesizes fine details and textures. To align

Algorithm 1 Compute loss

Require: Set of SLIC regions S_i , target tile I_i , current prediction $\hat{x}_0$, number of quantiles P , weights w_p
Initialize $[Q_0, \dots, Q_{P-1}] \leftarrow$ False map for all $p \in [0, P-1]$
 $E \leftarrow (I_i - \hat{x}_0)^2$
for $s \in S_i$ **do** ▷ s indexes the pixel set of SLIC region
 $[q_0, \dots, q_{P-1}] \leftarrow \text{quantile_masks}(E[s], P)$
▷ Boolean masks for P quantile bins of E within region s
for $p \in [0, P-1]$ **do**
 $Q_p \leftarrow Q_p \vee q_p$ ▷ Accumulate active pixels
end for
end for
 $\mathcal{L} \leftarrow \sum_{p=0}^{P-1} w_p \text{MSE}(I_i[Q_p], \hat{x}_0[Q_p])$
return $\mathcal{L}$

with this observed coarse-to-fine image formation, we apply a corresponding optimization schedule rather than enforcing a detailed layout throughout denoising. We construct a Gaussian pyramid from the target tile I_i and optimize $\hat{x}_0$, with respect to rescaled targets I_i , progressively increasing in resolution as denoising proceeds.

To speed inference, we apply guidance at selected timesteps, rather than uniformly at each timestep. Gradient descent iterations proceed until either a target loss is reached, or to a maximum number of iterations. Further details will be presented in Section 5.

4.2.3 Semantic Focus. The accuracy/discernibility tradeoff does not need to be uniform across the entire image. Certain foreground regions, with important details, demand that we skew towards accuracy as viewers will likely be more critical when these details are poorly reproduced in the generated tile. We mark tiles for further refinement by combining foreground/background segmentation [18] with a Canny edge detector [5] applied at a reduced resolution. Tiles that contain both foreground and edge pixels are then treated for higher accuracy. We apply a stricter target loss and increase the maximum number of gradient descent iterations. In Figure 7 we note that the background edges (trees) are not marked for increased focus, nor are the foreground regions without prominent edges.

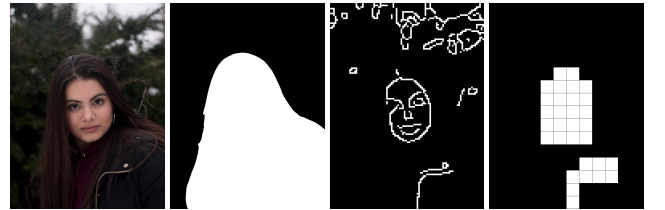


Figure 7: Selection of tiles for greater accuracy. From left to right: input image, foreground segmentation, edge map, selected tiles.

5 Experiments

5.1 Implementation Details

We implement our method with Stable Diffusion 1.5 with $T = 50$ denoising steps, using the diffusers library. All our experiments are run on a single NVIDIA RTX 4090 GPU.

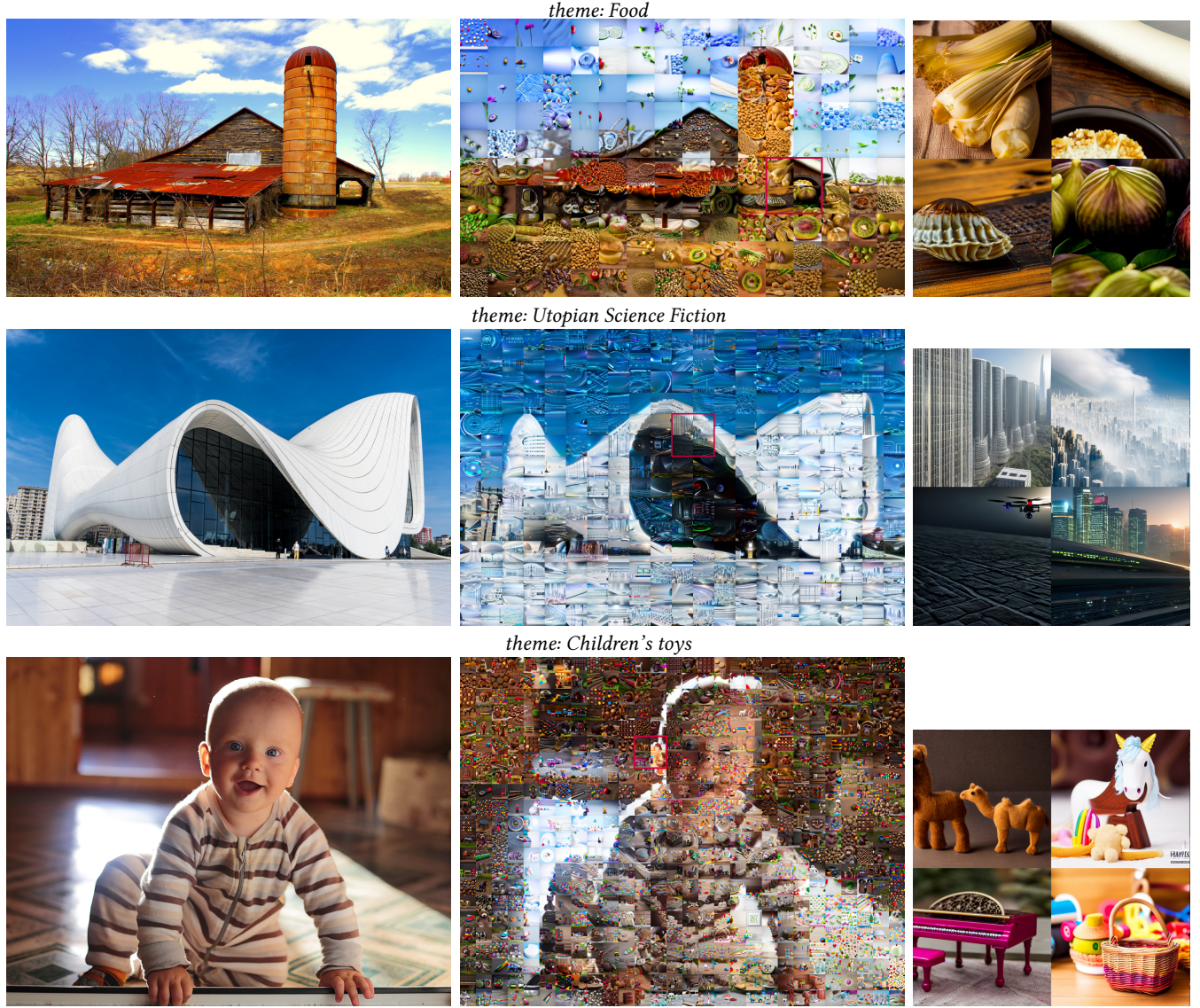


Figure 8: Image Mosaic results from our method.

5.1.1 Parameter settings. Several user-specified parameters in our method are set to default values. While a full ablation study is beyond the scope of this work, empirical testing has guided these defaults toward settings that reliably produce quality results with Stable Diffusion 1.5. Other models may differ. We discuss the parameter space below.

Target loss and iterations. Focus and non-focus tiles are configured independently, reflecting a speed-accuracy tradeoff. For non-focus tiles, faster generation with relaxed loss targets is acceptable, reducing overall generation time across the mosaic. We use a target loss of 0.004 with a maximum of 8 optimization iterations. In contrast, focus tiles are expected to more faithfully conform to the local structure of the input image. Here we use a target loss of 0.001 and up to 16 iterations.

Quantiles and loss weights. The number of quantiles used in the loss function, and their relative weights, determine the extent to which a tile may spatially deviate from its target. Too few quantiles limits the spatial resolution of the loss, while too many increases computational cost. We selected 4 quantiles with a declining weight scheme [0.4, 0.3, 0.2, 0.1] corresponding to increasing error in E , prioritizing regions that already conform closely to the target.

Pyramid levels. The number of pyramid levels is determined by the smallest image size that can be reliably optimized. Since quantiles are computed within independent SLIC regions, we found that 8×8 is the smallest patch size providing sufficient pixel count for quantile computation. For a typical 64×64 input tile, this gives 4 pyramid levels. For smaller 40×40 input tiles, 3 pyramid levels are used to achieve comparable spatial resolution.

Optimization steps. We select a stopping timestep based on the point at which image structure is determined and remaining iterations primarily refine fine detail. As shown in Figure 3, step 16 serves as a reasonable cutoff. To reduce computational overhead, guidance is applied at every second denoising step. For a standard 64×64 tile, optimization is therefore applied at 8 steps: 1, 3, ..., 13, 15 with each of the 4 pyramid level used twice.

5.2 Qualitative Results

Figure 8 shows some results from our method. We used 64×64 pixel tiles to reconstruct images in the first two rows, showing that we can produce an image mosaic even with a coarse tiling. Search-based methods typically use a high tile count to achieve better accuracy. The final row demonstrates a smaller tile size of 40×40 . Generative methods have a clear advantage in aligning sub-tile details with image content. In a search-based method, a large dataset must be assembled to be able to reasonably source images with accurate details beyond a simple color average. Even when source image transformations are used, accuracy is directly tied to dataset size, and large datasets take significant effort to assemble.



Figure 9: Comparison of our quantile loss (centre) score with MSE (right). Quantile loss uses weights [0.4, 0.3, 0.2, 0.1] corresponding to increasing error in E .

We demonstrate the effect of our quantile-based loss in Figure 9. While greater accuracy can be obtained with a uniform MSE loss, there is a greater chance of image distortion. Strict enforcement of color alignment will not produce natural-looking images. By allowing the diffusion process to fill in image structures with weak

constraints, the prevalence of obvious distortions is reduced. For example, the main sandpiper in the first row becomes smaller, with a bluish tint, when uniform weights are applied in the rightmost image. The smaller submerged bird also becomes distorted by the blue background. In the second row, more varied background detail emerges when using the quantile-based loss. Both results in the bottom row are distortion-free; however, allowing some divergence from the target colors enables the typical blue head of the fairywren to appear in the output.

The optimization process can be costly, as it effectively adds additional steps to the denoising process for each tile. Therefore, budgeting the diffusion steps can be a useful strategy. We achieve this by lowering the target loss score, and maximum optimization steps, when generating tiles away from the regions of interest. The process for selecting tiles for increased attention was communicated in Section 4.2.3. In Figure 10, we demonstrate how decreasing the target loss score produces a tile more closely aligned to the target.

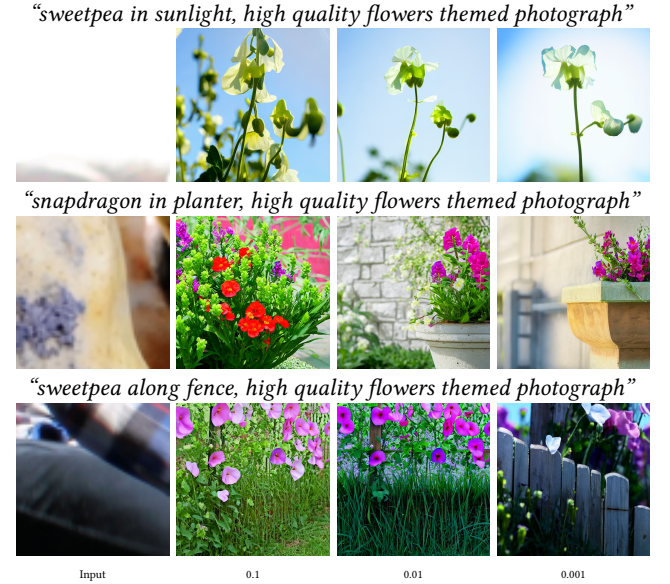


Figure 10: Increasing the target loss score at each iteration. The 3 images on the right are optimized against the leftmost images with target losses of 0.1, 0.01, and 0.001.

5.3 Comparisons

Figure 11 compares our method with baselines: Stable Diffusion’s image-to-image pipeline [34], IP Adapter [45], and T2I Adapter [29]. The baselines attempt to match the color input from all spatial locations equally, while we allow non-uniform guidance with our quantile-based loss. With Stable Diffusion *image-to-image*, we add noise up to 85% and then denoise. While color matching is reasonable here, image quality suffers in some instances. Notably, the hedgehog (row 3) and the cricket (row 4) are pushed to the bottom of the image and the ducks (row 5) are shrunk to a minuscule size. This pattern manifested in our preliminary results as well, under spatially uniform optimization. After adding the quantile-based loss, our output images adequately depict the prompt subject.

IP Adapter and T2I Adapter both used 8×8 approximations of the input color layout, as described by the authors. Both frequently struggle to match the input color scheme. For example, the circus theme results (row 2) depart from the input quite significantly. Also note the IP Adapter result in row 1: the colors are matched, but the layout is reversed, suggesting that image-based cross-attention lacks the spatial precision necessary for image mosaics.

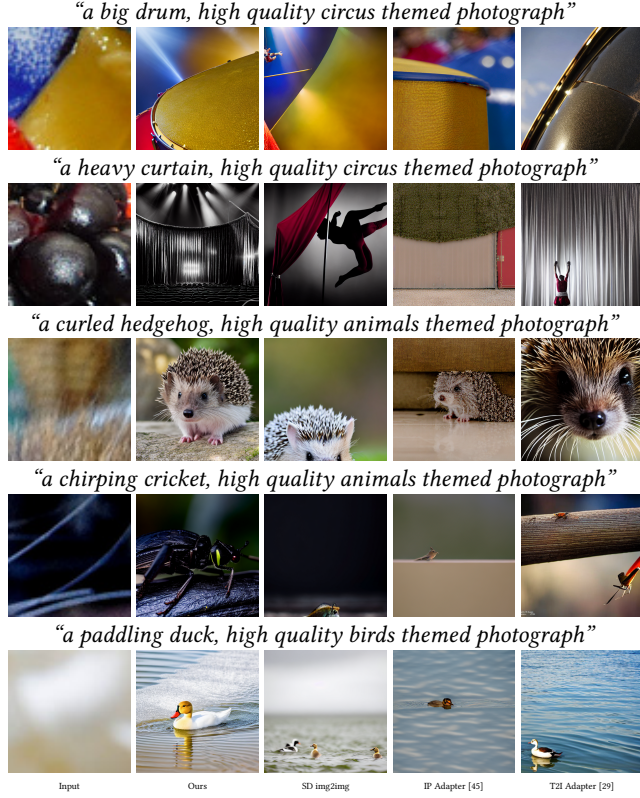


Figure 11: Comparison of our method with Stable Diffusion’s *image-to-image* pipeline [34], IP Adapter [45], and T2I Adapter [29].

We also compare our method against the search-based photomosaic methods of Orchard and Kaplan [31] and Di Blasi et al. [4] in Figure 12 using square tiles of 40×40 pixels. Orchard and Kaplan’s method can better preserve the input image, even without color correction, in comparison with Di Blasi et al. This is primarily due to the ability to search sub-images. They report a RMSE (root mean squared error) of 50.1, while Di Blasi reports 67.1 using pixel values in $[0, 255]$. One consequence of sub-image search is that unusual cropping can occur and reduce quality of the individual tiles in the output. This phenomenon is evident in the group of images at the bottom-right of the figure: the primary subject is squeezed out of the left side of the image. In our generative approach, undesirable crops are less likely to occur, since diffusion models are typically trained on intact images. Our quantile-based loss function also enables images to be generated without universally adhering to the background color layout. We report a RMSE of 46.5 in our example on the complete mosaic shown in the left group of images.

5.4 Quantitative Results

Table 1 shows comparisons of our method, and others, against the Stable Diffusion (SD) baseline. We chose two no-reference metrics: CLIP Score [22] measures alignment between image and text prompt, while Pick score [26] estimates image quality. We compute MSE by blurring and sub-sampling the 512×512 output tiles to 64×64 , then comparing against the target tiles at their original 64×64 resolution. Each score is presented with its corresponding difference relative to the SD baseline. Ideal output should have CLIP and Pick Scores consistent with the SD baseline, plus low MSE.

Evaluation is conducted on the 20 images of the NPR benchmark [30]. Each image is divided into 64×64 pixel tiles forming a target for image synthesis. Prompts are selected following Section 4.1. We compare our method against Stable Diffusion *image-to-image* using a 0.85 noise strength, T2I Adapter, and IP Adapter. The baseline Stable Diffusion model scores best on CLIP and Pick Score metrics. This is expected because the model output can align itself to the text prompt without additional color constraints. Notably, the Stable Diffusion *image-to-image* pipeline scores slightly better than ours on MSE and slightly worse on prompt alignment (Clip) and image quality (Pick). Referring to the qualitative results in Figure 11 we can see that the numerical trends are confirmed; however, it should be apparent that our method improves on image quality more than these numbers suggest. When comparing focus and non-focus tiles, we can see that image quality and prompt alignment score similarly. However, the extra optimization steps that are performed in the focus tiles improve MSE. Finally, both IP Adapter and T2I Adapter show inferior results, indicating that these methods are unsuited to the image mosaic task.

Method	CLIP Score		Pick Score		MSE Score
	Score	Diff	Score	Diff	
SD	0.304	–	0.197	–	–
Ours (non-focus)	0.296	-0.008	0.196	-0.001	0.035
Ours (focus)	<i>0.294</i>	-0.010	<i>0.194</i>	-0.003	<i>0.028</i>
SD img2img	0.291	-0.013	0.191	-0.006	0.024
IP Adapter	0.267	-0.037	0.185	-0.012	0.098
T2I Adapter	0.281	-0.023	0.191	-0.006	0.112

Table 1: Comparison of methods against Stable Diffusion (SD) [34] baseline. Bold and italic text indicates best and second-best scores.

5.5 Limitations

While our optimization-based method achieves superior color-guided generation, it reduces efficiency by requiring additional denoising iterations and decoder evaluations at each guidance step. We add a maximum of $8 \times 16 = 128$ additional steps on focus tiles and $8 \times 8 = 64$ additional steps on non-focus tiles. This can take up to 20 seconds per tile on a RTX 4090 GPU using 64×64 pixel input tiles that are synthesized at 512×512 . This will take around 85 minutes for a 1 megapixel image that outputs at 64 megapixels.

Although diffusion models are known to occasionally produce distorted images, we find a poor prompt/layout combination can increase this tendency. For example, Figure 13 shows how the prompt

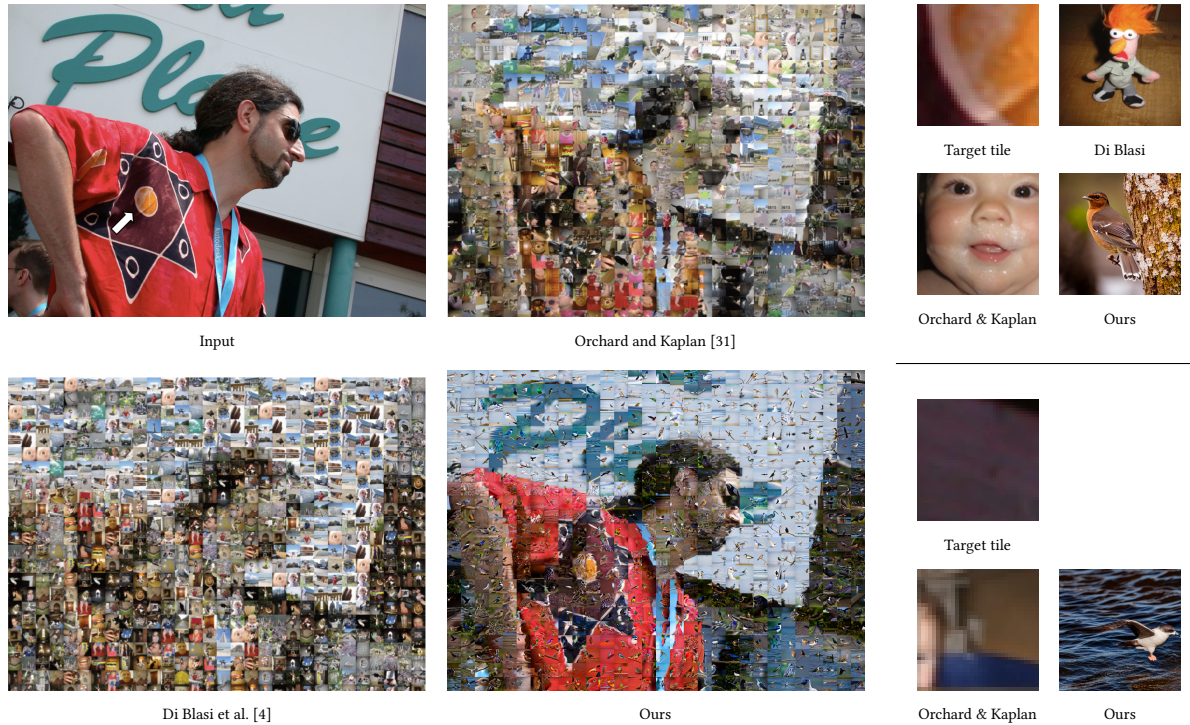


Figure 12: Comparison with Orchard and Kaplan [31] and Di Blasi et al. [4].

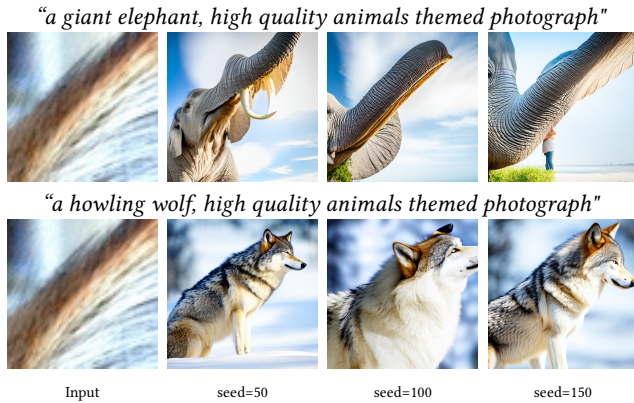


Figure 13: Our method sometimes struggles with some prompt/layout combinations. The elephant prompt (above) produced distorted images in each of 3 random initializations. The wolf prompt (below) produced better results.

“a giant elephant, high quality animals themed photograph” failed to guide the generation of a believable image across three random initializations. Because prompts are chosen randomly from a dataset, these distortions can happen occasionally and the only solution is regeneration of a tile with a more suitable prompt. The wolf in the bottom row shows improved results.

6 Conclusions

This paper presents a training-free method for adapting diffusion models to image mosaic generation. By generating tiles directly from text prompts, our approach eliminates the need to assemble large image datasets. Instead, diverse mosaics can be created by organizing AI-generated prompts into thematic collections that are quick to construct and offer numerous creative possibilities. We demonstrate that an optimization-based framework delivers enhanced controllability and improves results compared to off-the-shelf color guided methods [29, 45].

In the future, we would like to explore acceleration options. While users can experiment with diverse themes to transform a given image, faster processing times are required to make diffusion-based image mosaics an interactive experience. Beyond static square tiles, many other options are possible, including video mosaics.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments. We also thank members of the Graphics, Imaging and Games Lab for productive comments and discussions. Funding for this work was provided by NSERC (RGPIN-2018-06298), OGS, and Carleton University. Images were sourced from Flickr and Unsplash under a Creative Commons license. We thank the photographers who contributed material: evakovacs787 (pear), zhre-fch (baby), Merenda Mattia (portrait), MrClean1982 (barn)—Flickr; Zulfugar Karimov (architecture), Matt Benson (knight)—Unsplash.

References

- [1] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. 2012. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34, 11 (2012), 2274–2282. doi:10.1109/TPAMI.2012.120
- [2] S. Battiato, G. Di Blasi, G.M. Farinella, and G. Gallo. 2006. A Survey of Digital Mosaic Techniques. In *4th Eurographics Italian Chapter Conference*. The Eurographics Association. doi:10.2312/LocalChapterEvents/ItalianChapConf2006/129-135
- [3] Brent Berlin and Paul Kay. 1969. *Basic color terms; their universality and evolution*. University of California Press, Berkeley, CA.
- [4] G. Di Blasi, G. Gallo, and M.P. Petralia. 2006. Smart Ideas for Photomosaic Rendering. In *4th Eurographics Italian Chapter Conference*. The Eurographics Association. doi:10.2312/LocalChapterEvents/ItalianChapConf2006/267-271
- [5] John Canny. 1986. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8, 6 (1986), 679–698. doi:10.1109/TPAMI.1986.4767851
- [6] D. Cantone, A. Ferro, A. Pulvirenti, D.R. Recupero, and D. Shasha. 2005. Antipole tree indexing to support range search and k-nearest neighbor search in metric spaces. *IEEE Transactions on Knowledge and Data Engineering* 17, 4 (2005), 535–550. doi:10.1109/TKDE.2005.53
- [7] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. 2023. MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 22503–22513. doi:10.1109/ICCV51070.2023.02062
- [8] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. 2023. Attend-and-Excite: Attention-Based Semantic Guidance for Text-to-Image Diffusion Models. *ACM Trans. Graph.* 42, 4, Article 148 (July 2023), 10 pages. doi:10.1145/3592116
- [9] Zichong Chen, Shijin Wang, and Yang Zhou. 2025. StyleBlend: Enhancing Style-Specific Content Creation in Text-to-Image Diffusion Models. *Computer Graphics Forum* (2025). doi:10.1111/cgf.70034
- [10] 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 *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 8795–8805. doi:10.1109/CVPR52733.2024.00840
- [11] Omer Dahary, Or Patashnik, Kfir Aberman, and Daniel Cohen-Or. 2025. Be Yourself: Bounded Attention for Multi-subject Text-to-Image Generation. In *Computer Vision – ECCV 2024*. Springer Nature Switzerland, Cham, 432–448. doi:10.1007/978-3-031-72630-9_25
- [12] Prafulla Dhariwal and Alexander Nichol. 2021. Diffusion Models Beat GANs on Image Synthesis. In *Advances in Neural Information Processing Systems*, Vol. 34. Curran Associates, Inc., 8780–8794. https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf
- [13] Gianpiero Di Blasi and Giovanni Gallo. 2005. Artificial mosaics. *The Visual Computer* 21, 6 (2005), 373–383. doi:10.1007/s00371-005-0292-4
- [14] Yoshinori Dobashi, Toshiyuki Haga, Henry Johan, and Tomoyuki Nishita. 2002. A Method for Creating Mosaic Images Using Voronoi Diagrams. In *Eurographics 2002 - Short Presentations*. Eurographics Association. doi:10.2312/egs.20021036
- [15] Gershon Elber and George Wolberg. 2003. Rendering traditional mosaics. *The Visual Computer* 19, 1 (2003), 67–78. doi:10.1007/s00371-002-0175-x
- [16] G.M. Faustino and L.H. de Figueiredo. 2005. Simple Adaptive Mosaic Effects. In *XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'05)*. 315–322. doi:10.1109/SIBGRAPI.2005.46
- [17] Adam Finkelstein and Marisa Range. 1998. Image mosaics. In *International conference on raster imaging and digital typography*. Springer, 11–22. doi:10.1007/BFb0053259
- [18] Daniel Gatis. 2020. *rembg*. <https://github.com/danielgatis/rembg>
- [19] Paul Haeberli. 1990. Paint by numbers: Abstract image representations. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*. 207–214. doi:10.1145/97879.97902
- [20] Alejo Hausner. 2001. Simulating decorative mosaics. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. Association for Computing Machinery, New York, NY, USA, 573–580. doi:10.1145/383259.383327
- [21] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. 2022. *Prompt-to-Prompt Image Editing with Cross Attention Control*. arXiv:2208.01626 doi:10.48550/arXiv.2208.01626
- [22] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. 2021. *CLIPScore: A Reference-free Evaluation Metric for Image Captioning*. arXiv:2104.08718 doi:10.48550/arXiv.2104.08718
- [23] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, Vol. 33. Curran Associates, Inc., 6840–6851. https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf
- [24] Jonathan Ho and Tim Salimans. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*.
- [25] Junhwan Kim, Fabio Pellacini, et al. 2002. Jigsaw image mosaics. *ACM Transactions on Graphics* 21, 3 (2002), 657–664. doi:10.1145/566654.566633
- [26] Yuval Kirstain, Adam Polyak, Uriel Singer, Shabbuland Matiana, Joe Penna, and Omer Levy. 2023. Pick-a-Pic: An Open Dataset of User Preferences for Text-to-Image Generation. In *Advances in Neural Information Processing Systems*, Vol. 36. Curran Associates, Inc., 36652–36663.
- [27] Delwin T. Lindsey and Angela M. Brown. 2014. The color lexicon of American English. *Journal of Vision* 14, 2 (02 2014), 17–17. doi:10.1167/14.2.17
- [28] Nicolas Loizou, Sharan Vaswani, Issam Hadj Laradji, and Simon Lacoste-Julien. 2021. Stochastic Polyak Step-size for SGD: An Adaptive Learning Rate for Fast Convergence. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 130)*. PMLR, 1306–1314. <https://proceedings.mlr.press/v130/loizou21a.html>
- [29] Chong Mou, Xintao Wang, Liangbin Xie, Yanze Wu, Jian Zhang, Zhongang Qi, and Ying Shan. 2024. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 38. 4296–4304. doi:10.1609/aaai.v38i5.28226
- [30] David Mould and Paul L. Rosin. 2016. A benchmark image set for evaluating stylization.. In *Expressive*. 11–20.
- [31] Jeff Orchard and Craig S Kaplan. 2008. Cut-out image mosaics. In *Proceedings of the 6th international symposium on Non-photorealistic animation and rendering*. 79–87. doi:10.1145/1377980.1377997
- [32] D Pavić, Ulf Ceumern, and Leif Kobbelt. 2009. GLZMOs: Genuine image mosaics with adaptive tiling. In *Computer Graphics Forum*, Vol. 28. Wiley Online Library, 2244–2254. doi:10.1111/j.1467-8659.2009.01437.x
- [33] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. 2024. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. In *International Conference on Learning Representations*, Vol. 2024. 1862–1874. https://proceedings.iclr.cc/paper_files/paper/2024/file/081b08068e4733ae3e7ad019fe8d172f-Paper-Conference.pdf
- [34] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis with Latent Diffusion Models. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10674–10685. doi:10.1109/CVPR52688.2022.01042
- [35] Reza Adhitya Saputra, Craig S Kaplan, Paul Asente, and RADOMIR Mech. 2017. FLOWPAK: Flow-based Ornamental Element Packing.. In *Graphics Interface*. 8–15.
- [36] Yifei Shen, Xinyang Jiang, Yifan Yang, Yezhen Wang, Dongqi Han, and Dongsheng Li. 2024. Understanding and Improving Training-free Loss-based Diffusion Guidance. In *Advances in Neural Information Processing Systems*, Vol. 37. Curran Associates, Inc., 108974–109002. doi:10.52202/079017-3460
- [37] Ka Chun Shum, Binh-Son Hua, Duc Thanh Nguyen, and Sai-Kit Yeung. 2025. Color Alignment in Diffusion. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 28446–28455. doi:10.1109/CVPR52734.2025.02649
- [38] Robert Silvers. 1997. *Photomosaics*. Henry Holt and Co., Inc.
- [39] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep Unsupervised Learning using Nonequilibrium Thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 37)*. PMLR, Lille, France, 2256–2265. <https://proceedings.mlr.press/v37/sohl-dickstein15.html>
- [40] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. *Denoising Diffusion Implicit Models*. arXiv:2010.02502 doi:10.48550/arXiv.2010.02502
- [41] Nicholas Tran. 1999. Generating photomosaics: an empirical study. In *Proceedings of the 1999 ACM Symposium on Applied Computing* (San Antonio, Texas, USA) (SAC '99). Association for Computing Machinery, New York, NY, USA, 105–109. doi:10.1145/298151.298213
- [42] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. 2023. Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1921–1930. doi:10.1109/CVPR52729.2023.00191
- [43] Zhenyu Wang and Min Lu. 2025. Image-Space Collage and Packing with Differentiable Rendering. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Papers (SIGGRAPH Conference Papers '25)*. Association for Computing Machinery, New York, NY, USA, Article 172, 11 pages. doi:10.1145/3721238.3730690
- [44] Pengfei Xu, Jianqiang Ding, Hao Zhang, and Hui Huang. 2019. Discernible image mosaic with edge-aware adaptive tiles. *Computational Visual Media* 5, 1 (2019), 45–58. doi:10.1007/s41095-019-0130-7
- [45] Hu Ye, Jun Zhang, Sibio Liu, Xiao Han, and Wei Yang. 2023. *IP-Adapter: Text Compatible Image Prompt Adapter for Text-to-Image Diffusion Models*. arXiv:2308.06721 doi:10.48550/arXiv.2308.06721
- [46] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. 2023. Adding Conditional Control to Text-to-Image Diffusion Models. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 3813–3824. doi:10.1109/ICCV51070.2023.00355
- [47] Yang Zhou, Rongjun Xiao, Dani Lischinski, Daniel Cohen-Or, and Hui Huang. 2024. Generating Non-Stationary Textures Using Self-Rectification. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7767–7776. doi:10.1109/CVPR52733.2024.00742