Generative Panoramic Image Stitching

Supplementary Material

Contents

S1. Supplementary Implementation Details

S1.1. Architecture Details.

S1.1.1. Inpainting Model Architecture and Inputs

The architecture backbone is based on the Stable Diffusion 2.1 inpainting model [4], which uses an encoder–decoder UNet architecture with embedded self-attention and cross-attention layers for multi-scale reasoning. The denoising process follows the DDPM framework [22], where a noisy latent representation is progressively refined to reconstruct the image.

As described in the main paper, we adapt the pre-trained inpainting diffusion model $\Psi(\mathbf{z}^{(i)}_{\text{crop},t}, t, \mathcal{C})$ where

$$\mathcal{C} = \{\mathbf{m}, (1-\mathbf{m}) \odot \mathbf{z}^{(i)}_{\text{crop}}, \mathbf{c}_{\text{ctx}}\}, \tag{S1}$$

Noisy Latent Input. During training, the input noisy latent input $\mathbf{z}^{(i)}_{\text{crop},t}$ is generated by corrupting the encoded latent of a random crop from the input panorama set $\{\mathbf{x}^{(i)}_{\text{pano}}\}_{i=1}^{N}$ with noise, as

$$\mathbf{z}^{(i)}_{\text{crop}} = \mathcal{E}_{\text{SD}}(\textsc{RandCrop}(\mathbf{x}^{(i)}_{\text{pano}})),$$

where the noisy latent is then

$$\mathbf{z}^{(i)}_{\text{crop},t} = \sqrt{\alpha_t}\mathbf{z}^{(i)}_{\text{crop}} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}$$

Conditioned Input and Mask. Following Stable Diffusion 2.1, the inpainting mask $\mathbf{m} \in \{0,1\}^{B\times 1\times 64\times 64}$ is a downsampled representation of the region to be inpainted. It is constructed by combining random shapes with warped boundary regions to simulate occlusion patterns. The conditioning input, $\mathbf{z}^{(i)}_{\text{crop}}$, is masked by $(1-\mathbf{m})$ to simulate occlusion and then concatenated with the mask itself and the noised latent $\mathbf{z}_t$ as input to the UNet.

Context Embeddings. The embeddings $\mathbf{c}_{\text{ctx}} \in \mathbb{R}^{B\times 77\times 1024}$ are derived from the same crop dimensions used for $\mathbf{z}^{(i)}_{\text{crop}}$, applied to the positional encoding map $\mathbf{x}_\gamma$ and passed through the context encoder $\mathcal{E}_{\text{ctx}}$:

$$\mathbf{c}_{\text{ctx}} = \mathcal{E}_{\text{ctx}}(\textsc{RandCrop}(\mathbf{x}_\gamma)).$$

These embeddings replace text conditioning and are fed into the cross-attention blocks of the UNet (see Section S1.1.4), enabling spatially-aware denoising. This conditioning scheme enables the model to perform both inpainting and outpainting with spatial coherence.

S1.1.2. Data Preparation

We use reference images capturing a scene from multiple viewpoints to build a panorama. Images are aligned via homography-based registration and warped onto a panorama of size $H_{\text{pano}} \times W_{\text{pano}}$ using homography matrices $\mathbf{H}_i$, computed with feature matching (we use SIFT [8, 38]).

A global positional encoding $\mathbf{x}_\gamma \in \mathbb{R}^{H_{\text{pano}}\times W_{\text{pano}}\times C}$, with $C \in \{4, 8, 12\}$, encodes spatial patterns. Coordinates $x \in [0, W_{\text{pano}}]$, $y \in [0, H_{\text{pano}}]$ are normalized:

$$p_x = \frac{2x - W_{\text{pano}}}{W_{\text{pano}}}, \quad p_y = \frac{2y - H_{\text{pano}}}{H_{\text{pano}}}.$$

Frequencies are:

$$f_i = \exp\left(\log f_{\min} + \tfrac{i}{F}\log\frac{f_{\max}}{f_{\min}}\right), \quad i = 0, \ldots, F-1.$$

The encoding is:

$$\mathbf{x}_\gamma(y,x,c) = \begin{cases} \sin(\pi p_x f_{c/2}), & c = 2i, \\ \cos(\pi p_x f_{c/2}), & c = 2i+1, \\ \sin(\pi p_y f_{(c-2F)/2}), & c = 2F+2i, \\ \cos(\pi p_y f_{(c-2F)/2}), & c = 2F+2i+1, \end{cases}$$

where $F = C/4$ is the number of frequencies per dimension, $i = 0, \ldots, F-1$, and c denotes the channel index. In our proposed method, we use $f_{\min} = 1$, $f_{\max} = 50$ and $C = 12$.

S1.1.3. Positional Encoding Processing

A crop of the global positional encoding $\mathbf{x}_\gamma[\mathbf{b}] \in \mathbb{R}^{H_{\text{pano}} \times W_{\text{pano}} \times C}$, where $\mathbf{b} = \{x, y, W = 512, H = 512\}$, is transformed into context embeddings $\mathbf{c}_{\text{ctx}} \in \mathbb{R}^{B \times 77 \times 1024}$ (where B is the batch size) through a convolutional processing module. The transformation proceeds as follows:

$$
\begin{aligned}
F_1 &= \text{GELU}(\text{Conv2D}(\mathbf{x}_\gamma[\mathbf{b}]; k{=}4,\ s{=}2,\ p{=}1,\ out{=}128)), \\
F_2 &= \text{GELU}(\text{Conv2D}(F_1; k{=}4,\ s{=}2,\ p{=}1,\ out{=}128)), \\
F_3 &= \text{AdaptiveAvgPool2D}(F_2,\ (7, 11)),
\end{aligned}
$$

reducing the spatial dimensions from 512×512 to 7×11. The feature map is then reshaped and projected to a higher-dimensional space:

$$
\begin{aligned}
F_4 &= \text{Reshape}(F_3,\ (B,\ 77,\ 128)), \\
F_5 &= \text{Linear}(F_4,\ \text{out} = 1024),
\end{aligned}
$$

yielding $F_5 \in \mathbb{R}^{B \times 77 \times 1024}$.

A token-level positional encoding inspired by transformer-based language models [61] is added to F_5. This encoding is precomputed for all 77 token positions and uses sinusoidal functions to encode each token's position in the sequence. For each token index $p \in \{0, 1, \ldots, 76\}$, its corresponding embedding $PE(p) \in \mathbb{R}^{1024}$ is defined by:

$$
\begin{aligned}
PE(p, 2i) &= \sin\left(p/10000^{\,2i/d}\right), \\
PE(p, 2i+1) &= \cos\left(p/10000^{\,2i/d}\right).
\end{aligned}
$$

for $i \in \{0, 1, \ldots, d/2 - 1\}$, where $d = 1024$ is the embedding dimension. The token positional encoding is added to F_5, i.e., $F_6 = F_5 + PE$, where $PE \in \mathbb{R}^{77 \times 1024}$ is broadcast across the batch dimension.

Finally, a layer normalization is applied:

$$
\mathbf{c}_{\text{ctx}} = \text{LayerNorm}(F_6),
$$

producing the final context embeddings $\mathbf{c}_{\text{ctx}} \in \mathbb{R}^{B \times 77 \times 1024}$, which are used as input to the pre-trained inpainting diffusion model's cross-attention layers. The entire positional encoding processor network is trained from scratch, allowing it to optimally learn the transformation from spatial positional encodings to meaningful context embeddings for the panorama inpainting and outpainting task.

S1.1.4. Attention Mechanisms

The model employs self-attention and cross-attention to integrate internal features across multiple views and enforce spatial consistency, respectively.

Self-Attention Self-attention operates on the latent feature map $\mathbf{z} \in \mathbb{R}^{B \times C \times H \times W}$ (e.g., $C = 320, H = W = 64$), flattened to $\mathbf{z}_{\text{flat}} \in \mathbb{R}^{B \times N \times C}$ where $N = H \times W$. We use multi-head attention, with h heads, each processing a subspace of dimension $d_{\text{head}} = C/h$:

$$
Q_i = \mathbf{z}_{\text{flat}} W_{q,i}, \quad K_i = \mathbf{z}_{\text{flat}} W_{k,i}, \quad V_i = \mathbf{z}_{\text{flat}} W_{v,i},
$$

where $W_{q,i}, W_{k,i}, W_{v,i} \in \mathbb{R}^{C \times d_{\text{head}}}$. Attention scores are computed as:

$$
A_{\text{head},i} = \frac{Q_i K_i^\top}{\sqrt{d_{\text{head}}}} \in \mathbb{R}^{B \times N \times N},
$$

with outputs aggregated across heads:

$$
\text{Attention}_{\text{self}} = \text{Concat}(\text{Attention}_{\text{head},1}, \ldots, \text{Attention}_{\text{head},h}) W_o,
$$

where $W_o \in \mathbb{R}^{C \times C}$. This enables global spatial reasoning, crucial for maintaining coherence across different views in the final panoramic image.

Cross-Attention Cross-attention integrates the context embeddings $\mathbf{c}_{\text{ctx}} \in \mathbb{R}^{B \times 77 \times 1024}$ with $\mathbf{z}_{\text{flat}}$ as

$$
Q_i = \mathbf{z}_{\text{flat}} W_{q,i}^{\text{cross}}, \quad K_i = \mathbf{c}_{\text{ctx}} W_{k,i}^{\text{cross}}, \quad V_i = \mathbf{c}_{\text{ctx}} W_{v,i}^{\text{cross}},
$$

where $W_{q,i}^{\text{cross}} \in \mathbb{R}^{C \times d_{\text{head}}}$, and $W_{k,i}^{\text{cross}}, W_{v,i}^{\text{cross}} \in \mathbb{R}^{1024 \times d_{\text{head}}}$. The output is

$$
\begin{aligned}
&\text{Attention}_{\text{cross}} = \\
&\quad \text{Concat}\left(\text{Attention}_{\text{cross},1}, \ldots, \text{Attention}_{\text{cross},h}\right) W_o^{\text{cross}},
\end{aligned}
$$

ensuring inpainted regions align with the spatial context, preserving consistent features like textures and lighting across multiple input views.

S1.2. Inference Procedure Details

We provide pseudocode for our panorama generation procedure in Algorithm 1. Our approach progressively generates the full panoramic canvas by sequentially denoising tiles. For each tile, the model performs T denoising steps, leveraging the concatenated input and cross-attention-guided context embeddings $\mathbf{c}_{\text{ctx}}$ to produce the tile output. We used $T = 50$. This procedure is repeated for all tiles, and the final panoramic image $\mathbf{x}_{\text{pano}}^{\text{gen}}$ is assembled sequentially, tile by tile. Tiles are sorted by the distance to the centroid of the starting reference image, in increasing distance, in a breadth-first-search manner. This allows the denoising of tiles with overlap of the starting reference image first, and subsequently outpainting tiles with overlap from previous generations. An example is shown in Figure S1.

Correspondence-based seed selection. Due to stochasticity in the inference process, the generation quality varies between random seeds. This is amplified by the numerous tiles required to denoise a full panorama, and artifacts

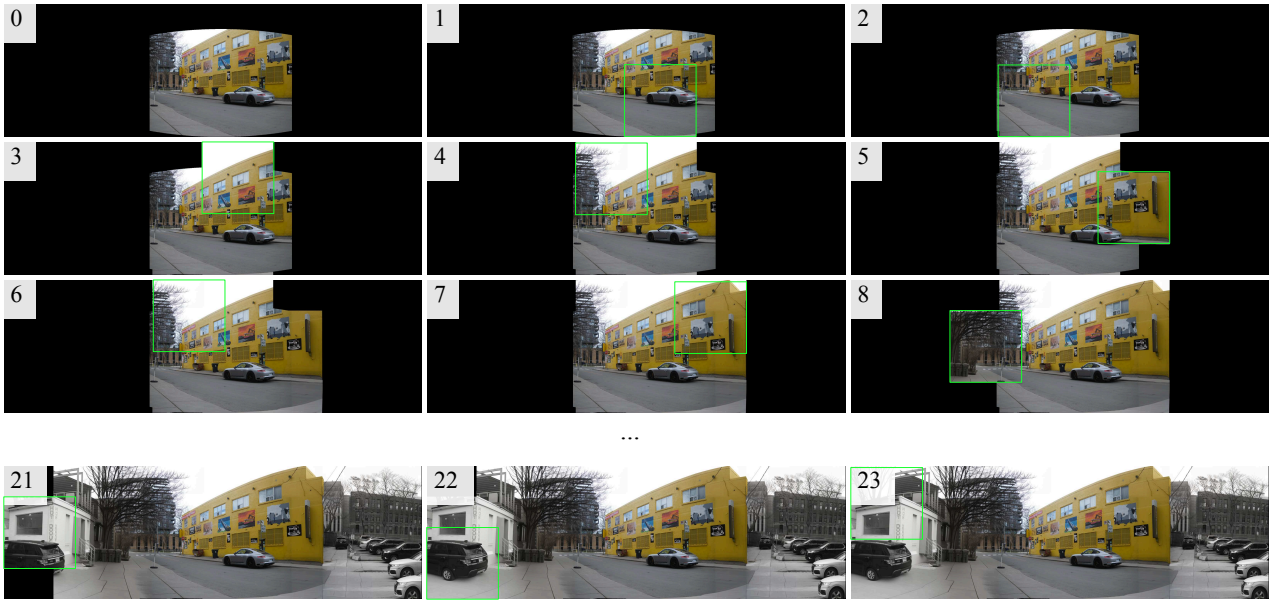

Figure S1. Example of the tiling strategy to generate the full panoramic canvas. Tiles closest to the reference image are denoised first, with subsequent tiles denoised in a breadth-first-search manner.

Figure S2. Example of the correspondence-based seed selection strategy to generate the full panoramic canvas. We generate 10 panoramas with different seeds and select the one with most feature matches. The selected panorama has the least artifacts in this example and is the most seamless and similar to the reference.

early-on may propagate throughout the canvas. We employ a correspondence-based seed selection process [60] to mitigate this problem, identifying generated panoramas whose layout matches the result of feature-based image registration [8]. An example of various seed generations is shown in Figure S2. We generate ten panoramas with different random seeds and take our output to be the panorama with the most feature matches (computed with LoFTR [56]) compared to the output of AutoStich [8] on the casually-captured dataset. Final metrics would be calculated by comparing the reference panorama from the tripod-captured dataset. This process could be further enhanced with more seeds, depending on desired computation budget (e.g. RealFill [60] generates 64 outputs).

Algorithm 1: Panorama Generation

Input: $\mathbf{x}_{\text{pano}}^{(0)}$ ◂ Input sparse panorama
$\mathbf{x}_\gamma$ ◂ Positional encoding map
$\{\mathbf{b}^{(i)}\}_{i=1}^{B}$ ◂ Tiles covering panorama

Output: Panorama $\mathbf{x}_{\text{pano}}$ with all tiles denoised

```
// Order tiles based on distance
```
$[\mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \ldots, \mathbf{b}^{(B)}] \leftarrow \textsc{OrderByDist}(\{\mathbf{b}^{(i)}\}_{i=1}^{B})$

for $i = 1$ **to** B **do**

```
    // Initialize noisy latent by adding
       noise to reference latent
```
    $\mathbf{z}_{\mathbf{b}^{(i)},T} \leftarrow \mathcal{E}_{\text{SD}}(\mathbf{x}_{\text{pano}}^{(0)}[\mathbf{b}^{(i)}]) + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I)$
    $\mathbf{c}_{\text{ctx}} \leftarrow \mathcal{E}_{\text{ctx}}(\mathbf{x}_\gamma[\mathbf{b}^{(i)}])$
    $\mathbf{m} \leftarrow \textsc{GetRegionToInpaint}(\mathbf{z}_{\mathbf{b}^{(i)},T})$
    for $t = T$ **to** 1 **do**

```
        // Denoise tile
```
        $\mathbf{z}_{\mathbf{b}^{(i)},t-1} \leftarrow$ $\textsc{Denoise}(\mathbf{z}_{\mathbf{b}^{(i)},t}, \mathbf{m}, (1-\mathbf{m}) \odot \mathbf{z}_{\mathbf{b}^{(i)},t-1}, \mathbf{c}_{\text{ctx}})$

```
    // Update output with denoised tile
       using latent decoder D_SD
```
    $\mathbf{x}_{\text{pano}}[\mathbf{b}^{(i)}] \leftarrow \textsc{Composite}(\mathbf{x}_{\text{pano}}[\mathbf{b}^{(i)}], \mathcal{D}_{\text{SD}}(\mathbf{z}_{\mathbf{b}^{(i)},0}))$

return $\mathbf{x}_{\text{pano}}$

S1.3. Training

Training begins by sampling a latent representation $\mathbf{z}_{\text{crop}}^{(i)}$ and the corresponding positional embedding $\mathbf{c}_{\text{ctx}}$ as explained in S1.1. The diffusion model $\Psi(\mathbf{z}_{\text{crop},t}^{(i)}, t, \mathcal{C})$ is trained to predict the noise added to the latent during the forward diffusion [22].

Optimization Strategy. The model parameters are optimized using the AdamW optimizer. To preserve the generative power of the pre-trained Stable Diffusion model, we adopt a selective fine-tuning approach:

- **LoRA (Low-Rank Adaptation)** [25] is applied to the self-attention layers of the UNet to enable efficient adaptation with fewer trainable parameters. We use a learning rate of 1×10^{-4}.
- **Cross-attention layers** are fully fine-tuned to allow better integration of positional context via $\mathbf{c}_{\text{ctx}}$. We use a learning rate of 3×10^{-4}.
- The **VAE encoder/decoder** and other layers of the UNet remain frozen to retain the fidelity of the original image reconstruction.
- The **context encoder** $\mathcal{E}_{\text{ctx}}$, which produces $\mathbf{c}_{\text{ctx}}$, is trained from scratch using standard initialization. We use a learning rate of 8×10^{-4}.

S1.4. Dataset

UDIS dataset. We pull two scenes from the UDIS dataset, defined by images `005737.jpg`, `005738.jpg`, `005740.jpg`, `005744.jpg`, `005832.jpg`, `005845.jpg`, `005846.jpg`, `005848.jpg`, `005849.jpg`, `005891.jpg`, `005904.jpg` and images `000083.jpg`, `000086.jpg`, `000093.jpg`, `000098.jpg`, `000104.jpg`, `000110.jpg`, `000117.jpg`, `000121.jpg`, `000127.jpg` (scene 2) from the training set (folder `input1`. Example images and the final panorama are shown in Figure S3.

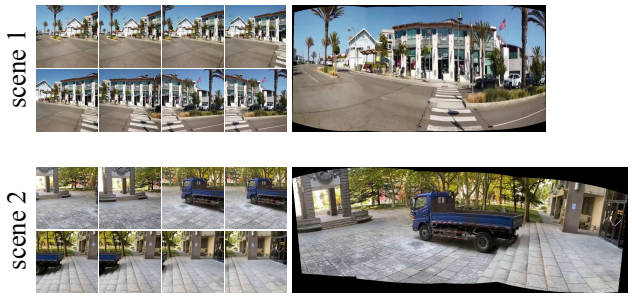

Figure S3. UDIS scenes, including example reference images from the original dataset and final stitched panorama using AutoStitch.

Tripod-captured and casually captured dataset. Figure S4 and Figure S5 show the reference images for the tripod-captured and casually-captured datasets, respectively. Both include the same eight scenes, captured (for the most part) at the exact same time: variations in lighting or time of day are captured in the casually captured set. The tripod-captured dataset attempts to capture a complete coverage of the scene, while the casually-captured dataset includes variations that make conventional stitching methods like AutoStitch [8] difficult. Specifically, we outline the variations for each scene as follows:

"Backyard" scene

Lighting/time-of-day (night vs. day), small parallax

"Bedroom" scene

Lighting (lights on vs. off), small parallax

"College" scene

White balance, image color filtering

"Donuts" scene

Image color filtering, strong parallax

"Livingroom" scene

Lighting variation (lights on vs. off), strong parallax, missing objects (foosball table removed), image orientation (landscape, portrait)

"Street" scene

Seasonal variation (winter vs. summer), small parallax, different objects (cars), image orientation (landscape, portrait)

"Subway" scene

Image color filtering (sepia), image orientation (landscape, portrait)

"Waterfront" scene

Strong parallax, white balance, image color filtering, image orientation (landscape, portrait)

S1.5. Baseline Implementation Details

RealFill [60]. We follow the default implementation guidelines and code for RealFill. We use the default prompt "a photo of sks" and guidance scale of 0.99 (as used in the original work) during inference. RealFill employs a simple prompt fine-tuning strategy as proposed in [53], where each reference image is randomly cropped during training and fine-tuned with the same input prompt. RealFill uses random masking during training and samples each reference image with equal probability.

SD2 Inpainting. We use ChatGPT to generate the following text prompts to inpaint scenes in our dataset using the Stable Diffusion 2 inpainting [4] baseline. We fed it the reference images and asked it to describe the scene for an in/outpainting task. We use the default guidance scale of 7.5.

"Backyard" scene prompt

A cozy backyard garden patio on a sunny spring afternoon, with light wooden fencing arranged in a chevron pattern enclosing the space. The ground is a mix of wooden decking and brick pavers. There is a modern white outdoor dining table surrounded by white molded chairs with wooden legs. Raised garden beds line the perimeter, filled with lush green ferns, hostas, and flowering tulips. A small Japanese maple tree with red leaves adds a vibrant accent. Overhead, string lights hang between tall trees. Suburban townhouses and fire escapes are visible beyond the fence. Continue the garden with more raised beds, greenery, and cozy shaded seating areas.

"Bedroom" scene prompt

A cozy bedroom with warm lighting and natural wood trim. A soft grey bedspread covers a modern dark wood bed, with a large plush gnome toy sitting upright at the head of the bed. The gnome has a fluffy white beard, red hat, and blue outfit. To the side, there's a pair of bright yellow slippers on the bed. The walls are painted beige, with framed artwork and a tall wooden door. Hardwood floors reflect the warm overhead light. The room is neat but lived-in, with a cardboard box on the floor, a dresser topped with a camera and board games, and open shelves filled with books and gadgets. Extend the scene naturally with matching lighting, wood finishes, and layout.

"College" scene prompt

A serene university courtyard on a crisp early spring morning, lined with tall leafless trees casting long shadows across the stone-paved paths. Old gothic stone buildings and vintage street lamps border the green lawn, while scattered wooden benches sit empty under the bare branches. A soft blue sky with gentle morning sunlight peeks through the trees, illuminating the historic campus in a calm, peaceful atmosphere. Natural lighting, detailed textures, realistic architecture, high-resolution photo.

Figure S4. Reference images for the the tripod-captured dataset. Eight scenes were captured, showing a range of contexts.

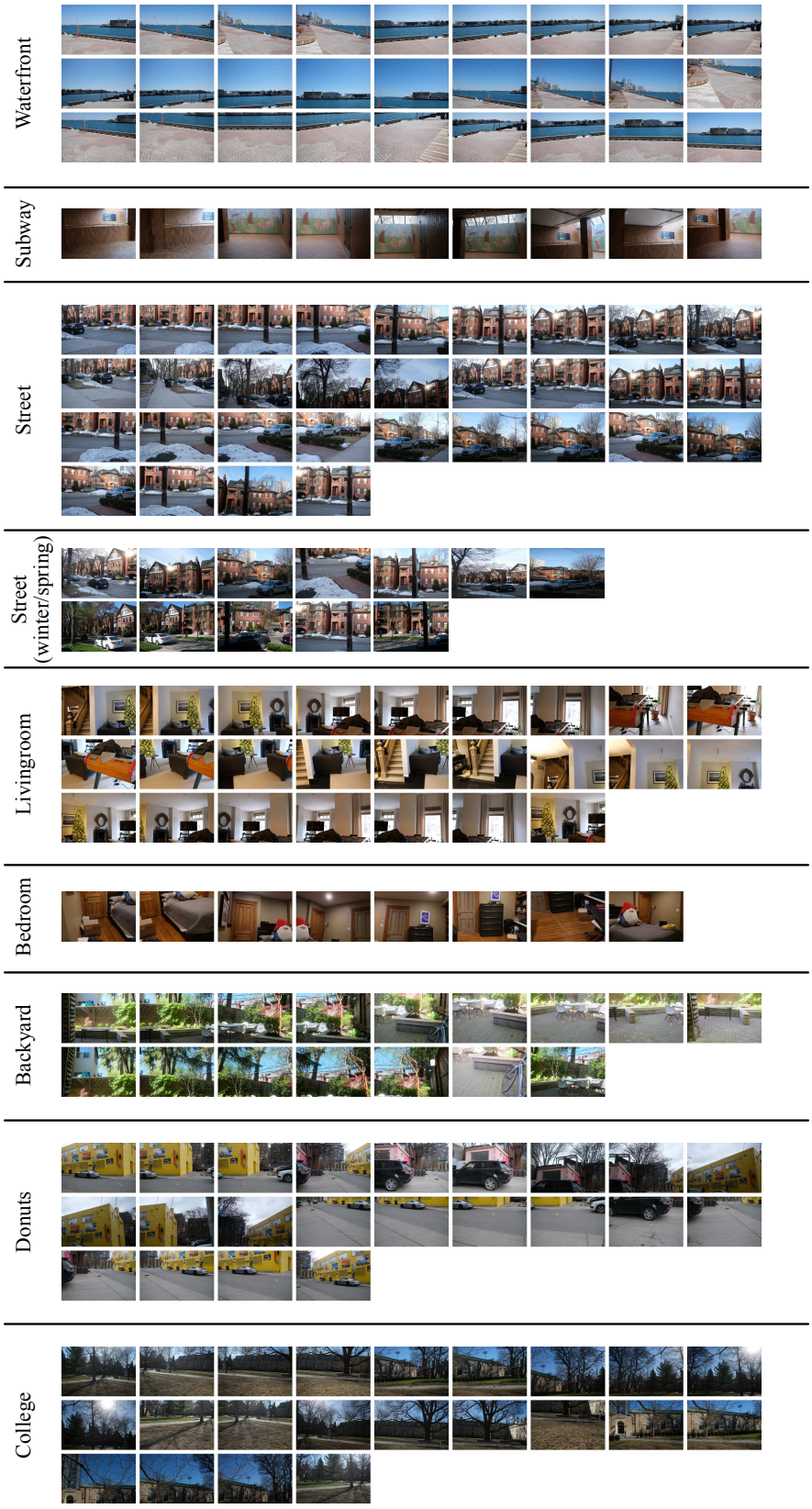

Figure S5. Reference images for the casually captured dataset. Eight scenes were captured, mirroring the tripod-captured dataset. This dataset shows challenging scenarios, where the input images have strong parallax effects (“Waterfront”, “Donuts”), variations in style (“Waterfront”, “Subway”), illumination (“Bedroom”, “Backyard”), color palette (“Waterfront”, “Donuts”, “College”), or seasonal changes (“Street”).

"Donuts" scene prompt

Urban back alley beside a bright yellow building with murals depicting industrial and artistic scenes, a red 'RECEIVING' sign, and barred windows. A silver sports car is parked on the street beside a city parking meter. Adjacent to it is a pastel pink storefront with bold 'HOT DOG' signage, garbage bins, and leafless winter trees. In the distance, high-rise glass apartments and brick institutional buildings frame the cityscape. Overcast sky, soft urban lighting, clean and calm street scene.

"Livingroom" scene prompt

Extend the cozy living room with warm lighting and rustic cabinetry, holiday decor continuing throughout the space.

"Street" scene prompt

A quiet urban residential street in winter, lined with large Victorian red-brick houses with gabled roofs and stone foundations. Bare trees stand on small front lawns covered in patches of snow. Parked cars line the street, including compact sedans and hatchbacks. The sky is clear and blue, with soft late afternoon sunlight casting long shadows. A mix of brick textures, wooden porches, and balconies add architectural charm. Extend the street with similar architecture, snow-covered sidewalks, more houses in perspective, and consistent lighting and color tone.

"Subway" scene prompt

A vintage subway concourse with brown ceramic tiles and steel railings, illuminated by ceiling spotlights. A surreal mural on the wall shows whimsical floating objects, vintage figures, a red car, a lion on a bed, and abstract staircases, all set against a bright grassy hill and a blue sky with clouds. Large glass windows above let in natural daylight and reveal leafless tree branches outside. The overall atmosphere is clean, quiet, and dreamlike, blending realism with surrealism.

"Waterfront" scene prompt

Toronto waterfront promenade on a clear sunny day, with a patterned brick walkway, stone barriers, life buoys on silver poles, modern industrial dock buildings across the lake, calm blue water, Porter Airlines ferry terminal, and distant city skyline with high-rise towers, boats on the lake, realistic urban scenery, vibrant shadows and natural lighting

3D Reconstruction + NeRF. We use https://www.agisoft.com/, a commercial photogrammetry software, to estimate camera poses for both the tripod-captured and casually captured image sets, as COLMAP failed to register many of our scenes due to limited overlap and strong parallax. We apply pose estimation to the combined collection of tripod-captured images and casually captured images for a given scene to obtain camera poses in the same coordinate system. We then train a NeRF model was using only the casually captured images, using NeRF Studio [59] and default settings. We rendered novel views from the poses corresponding to the tripod-captured images and stitched the resulting renders using AutoStitch to produce the final panoramas.

RDIStitcher [67]. Since the original RDIStitcher implementation is limited to pairwise stitching, we extended it to support multi-image panorama generation. We began by aligning all input images to a global cylindrical coordinate frame using AutoStitch to obtain pairwise homographies and global layout. To determine the stitching sequence, we preserved the stitch order produced by AutoStitch, which ranks image pairs by the number of feature correspondences. Because RDIStitcher accepts fixed-size 512×512 inputs and lacks an automated mechanism for region selection, we manually defined crop regions corresponding to overlapping areas across the panorama. Each cropped image pair was then processed using the pretrained RDIStitcher diffusion pipeline with LoRA weights provided in the official repository, employing 50 denoising steps and a classifier-free guidance scale of 7.5, consistent with the original inference settings. The fused result from each pass was composited back into the global cylindrical panorama at the corresponding spatial coordinates. This iterative procedure was repeated until all warped views were integrated into the final stitched panorama.

S1.6. Metrics

To evaluate the quality of generated panoramic images, we employ a comprehensive set of metrics that assess standard image quality, high-level image structure, and preservation of scene layout. The latter two use learning-based metrics and feature-matching-based metrics, respectively. We evaluate generated panoramas $\mathbf{x}_{\text{pano}}^{\text{gen}} \in \mathbb{R}_{+}^{H_{\text{pano}} \times W_{\text{pano}} \times 3}$ against the reference panorama produced using AutoStitch [8] on the tripod-captured dataset $\mathbf{x}_{\text{pano}}^{\text{ref}} \in \mathbb{R}_{+}^{H_{\text{pano}} \times W_{\text{pano}} \times 3}$. All metrics except DreamSim, CLIP, and DINO omit the region of the sparse panorama provided as input during inference (defined by a binary mask $\mathbf{m}_{\text{input}} \in \{0,1\}^{H_{\text{pano}} \times W_{\text{pano}}}$) to focus on inpainted areas. Below, we describe each metric, its implementation details, and its significance.

PSNR (Peak Signal-to-Noise Ratio). Measures pixel-level similarity [24], defined as

$$\texttt{PSNR} = 10 \cdot \log_{10} \left(\frac{\texttt{MAX}_{\mathbf{x}}^2}{\texttt{MSE}(\mathbf{x}_{\text{pano}}^{\text{ref}}, \mathbf{x}_{\text{pano}}^{\text{gen}})} \right),$$

where $\texttt{MAX}_{\mathbf{x}} = 255$ for 8-bit RGB images, and $\texttt{MSE}(\cdot)$ is the mean squared error between valid pixels (i.e., where $\mathbf{m}_{\text{input}} = 0$) of $\mathbf{x}_{\text{pano}}^{\text{ref}}$ and $\mathbf{x}_{\text{pano}}^{\text{gen}}$. Higher values indicate better pixel fidelity.

SSIM (Structural Similarity Index). Assesses structural and perceptual similarity by comparing luminance, contrast, and structure in grayscale images [64]:

$$\texttt{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)},$$

where μ_x, μ_y are means, σ_x, σ_y are variances, σ_{xy} is covariance, and c_1, c_2 are constants. We compute SSIM on grayscale images with masked regions ($\mathbf{m}_{\text{input}} = 1$) set to zero. Higher values indicate better structural consistency.

LPIPS (Learned Perceptual Image Patch Similarity). Measures perceptual similarity using a pre-trained AlexNet [27], as proposed by Zhang et al. [76]. For permuted image tensors $\mathbf{x}_{\text{pano}}^{\text{ref}}, \mathbf{x}_{\text{pano}}^{\text{gen}} \in \mathbb{R}_+^{B\times 3\times H_{\text{pano}}\times W_{\text{pano}}}$, normalized to $[-1, 1]$, LPIPS computes feature distances as

$$\texttt{LPIPS} = \frac{1}{H'W'} \sum_{h,w} \texttt{loss}_{\texttt{Alex}}(\mathbf{x}_{\text{pano}}^{\text{ref}}, \mathbf{x}_{\text{pano}}^{\text{gen}}) \cdot (1 - \mathbf{m}'_{\text{input}}),$$

where $\texttt{loss}_{\texttt{Alex}}$ is the weighted L2 distance between feature activations from AlexNet layers, $\mathbf{m}'_{\text{input}}$ is the resized mask, and H', W' match the feature map size. Lower values indicate better perceptual similarity.

DreamSim [19]. Evaluates high-level perceptual similarity using the DreamSim model trained on human perceptual judgments. The metric is:

$$\texttt{DreamSim} = \texttt{dreamsim\_model}(\mathbf{x}_{\text{pano}}^{\text{ref}'}, \mathbf{x}_{\text{pano}}^{\text{gen}'}),$$

where $\mathbf{x}_{\text{pano}}^{\text{ref}'}, \mathbf{x}_{\text{pano}}^{\text{gen}'}$ are images resized to 224×224 to match the model's input requirements. The DreamSim model, based on a vision transformer, predicts perceptual similarity by comparing feature embeddings. Lower scores indicate closer perceptual alignment.

DINO. Measures semantic similarity using the DINOv2-base model [46] as the cosine distance between features extracted from the last hidden state:

$$\texttt{DINO} = \frac{\cos(\texttt{feat}^{\text{ref}}, \texttt{feat}^{\text{gen}}) + 1}{2},$$

where $\texttt{feat}^{\text{ref}}, \texttt{feat}^{\text{gen}}$ are mean-pooled features from the last hidden state of DINOv2, and $\cos(\cdot)$ is the cosine similarity. Higher values indicate better semantic alignment.

CLIP. Assesses semantic similarity using the CLIP ViT-B/32 model [49]. The CLIP score is defined as the cosine similarity between normalized CLIP image embeddings:

$$\texttt{CLIP} = \cos(\mathbf{z}_{\texttt{clip}}^{\text{ref}}, \mathbf{z}_{\texttt{clip}}^{\text{gen}}),$$

where $\mathbf{z}_{\texttt{clip}}^{\text{ref}}, \mathbf{z}_{\texttt{clip}}^{\text{gen}}$ are image embeddings from the forward pass of the CLIP ViT-B/32 model. Higher values indicate better semantic consistency.

Figure S6. Example of LoFTR feature matching between reference and generated panoramas. White circles mark selected keypoints in the reference, with corresponding points in the generated panorama color-coded by L2 pixel distance from their reference positions.

LoFTR Metrics. Evaluates feature correspondence using LoFTR [56]. Images are resized to 512×512 and converted to grayscale and then processed. We report both the L2 distance between the pixel coordinates of matching features and the number of matched features divided by the total number of features in the reference image. Specifically,

$$\texttt{LoFTR\_L2\_Distance} = \frac{1}{N} \sum_{i=1}^{N} \sqrt{\sum (\texttt{mkpts}_i^{\text{ref}} - \texttt{mkpts}_i^{\text{gen}})^2}, \tag{S2}$$

$$\texttt{LoFTR\_Match\_Proportion} = \frac{N}{\texttt{total\_features}},$$

where $\texttt{mkpts}^{\text{ref}}, \texttt{mkpts}^{\text{gen}}$ are matched keypoints outside $\mathbf{m}_{\text{input}}$, N is the number of valid matches between the reference and the generated panorama identified by

LoFTR, and `total_features` is the number of reference keypoints identified by LoFTR in the reference panorama. Lower `LoFTR_L2_Distance` and higher `LoFTR_Match_Proportion` indicate better correspondence. An example of matches identified by LoFTR for an image crop is visualized in Figure S6.

S2. Supplementary Results

S2.1. UDIS Qualitative Dataset

We evaluate no-reference quality metrics on two scenes from UDIS (see Figure S3). Qualitative results are shown in Figure S7. Our method handles the scenes well, while RDISTitcher exhibits consistent banding and poor fidelity.

Figure S7. Qualitative comparison of the proposed method against AutoStitch and RDIStitcher on two scenes from the UDIS dataset [42].

S2.2. Tripod-Captured Dataset

We show the additional seven scenes for the tripod-captured dataset in Figure S8. Similar to before, we observe that the Stable Diffusion inpainting model [4] produces image content that is locally plausible, but fails to adhere to the layout and content of the actual scene. Similar to previous scenes, RealFill [60] improves on this result, but tends to repeat scene content and ignores scene layout. Our approach provides a much closer match to the layout provided by the reference panorama.

S2.3. Casually Captured Dataset

We show the additional four scenes for the casually captured dataset in Figure S9. Similar to other scenes, AutoStitch [8], fails to convincingly blend between the different image regions, resulting in ghosting and other artifacts. RealFill exhibits similar artifacts as in the tripod-captured dataset, and we find that our approach produces seamless results that are more consistent with the layout and content of the scene.

S2.4. Supplementary Ablation Studies

Quantitative results. We conduct an ablation study on the casually captured dataset (see Table S1). We evaluate (1) the effects of parameter choices in positional encoding frequencies (number of channels, max frequency, and omitting token positional encodings), (2) inference strategies, omitting the reference image during inference and denoising tiles row-by-row, with rows sorted by distance to the starting image in the y-axis, and tiles sorted by the distance to the centroid in the x-axis, (3) various guidance scales, (4) various overlap ratios, and (5) training without positional encoding and only using warped reference images. Each ablation uses correspondence-based seed selection to eliminate concerns over seed selection.

Significantly lower max frequency (10Hz), smaller number of channels (4 channels), and no token positional encoding show improvements in some image quality metrics, but a fall in the feature-matching-based metrics. The higher frequencies of the proposed method (12-channels, 50Hz) allow for finer details and better reconstruction of features from the reference images.

Removing the reference image shows a drop across the board in performance, showing the necessity of a starting reference image, as expected. Performance still outperforms prior baselines (see Table 2).

Guidance scales between 1.5 and 2.00 and overlap ratios between 0.1 and 0.2 show the best performance in class. We chose a guidance scale of 1.5 with an overlap ratio of 0.2. Generating panoramas using a row-by-row sorting shows marginal improvements in some metrics, however, we found qualitatively that more artifacts are produced. These artifacts are more evident to users, and therefore we opted not to use this strategy.

RealFill with warped reference images suffers from similar repetitive content and a lack of adhesion to the reference layout, demonstrating the need for our proposed positional encoding conditioning.

Qualitative results. We show qualitative results for the various guidance scales in Figure S13. Lower guidance scales (< 1.5) maintain scene quality but fail to properly blend through artifacts (e.g. building remains grayscale). Guidance scales between $1.5 - 2$ show how scene cohesion can be maintained while also resolving artifacts found in the reference images (building is well blended). Higher guidances begin to exhibit "cartoonish" effects and the scene loses cohesion (obvious seams between regions in the panorama).

We show qualitative results for the effect of perturbing the locations of sparse images in the panorama Figure S11. Without perturbation, the model is less robust to misalignments in the initial layout estimation.

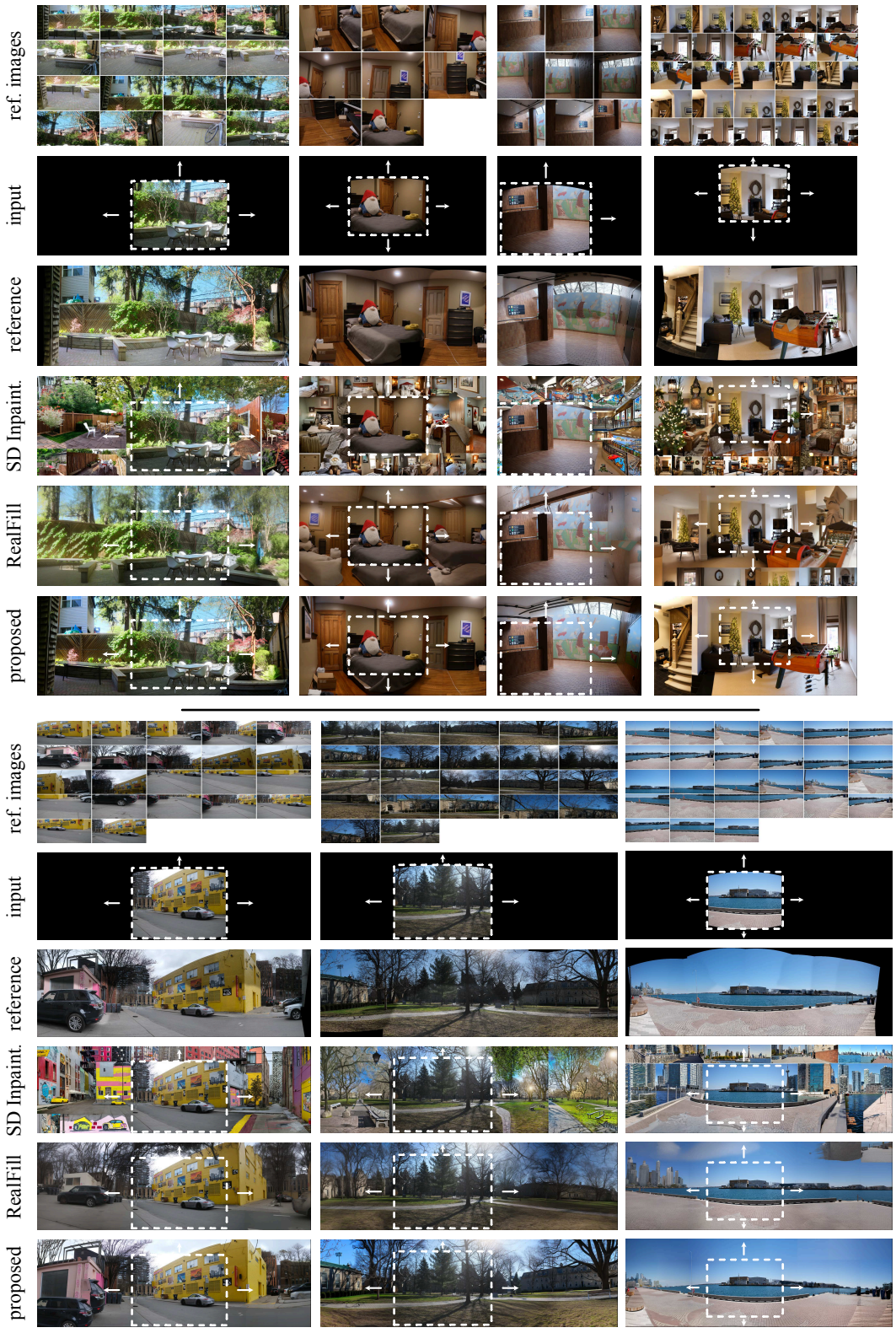

Figure S8. Qualitative results on the additional scenes from the tripod-captured dataset. We find that our approach produces panoramas that are more consistent with the layout and content of the reference panorama than baseline approaches based on inpainting/outpainting.

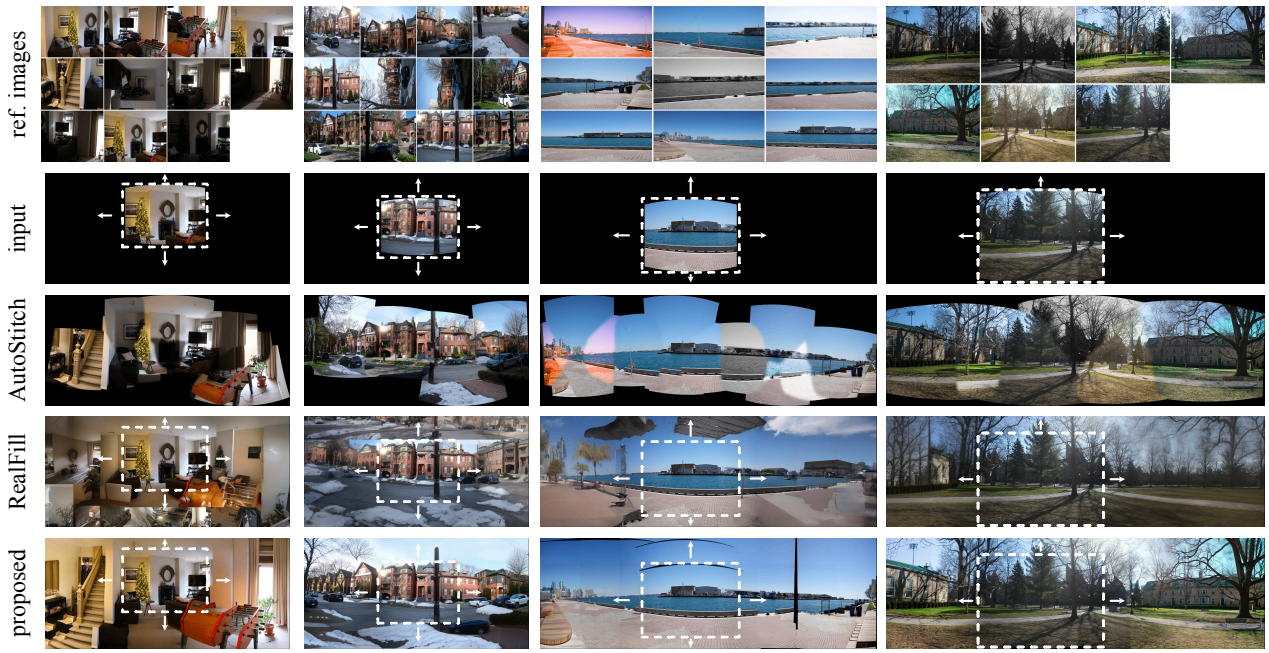

Figure S9. Qualitative results of the additional four scenes on the casually captured dataset. Even in this challenging scenario, where the input images have strong parallax effects and variations in style, illumination, color palette, or camera capture settings, our approach reconstructs seamless panoramas that preserve the content and layout of the reference.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	DreamSim ↓	DINO ↑	CLIP ↑	LoFTR (L2 Distance) ↓	LoFTR (Matching) ↑
proposed (10Hz)	11.09 (2.03)	0.414 (0.100)	0.532 (0.065)	0.137 (0.024)	0.971 (0.011)	0.917 (0.034)	21.65 (4.48)	0.119 (0.037)
proposed (4 channels)	11.12 (1.96)	0.416 (0.102)	0.543 (0.065)	0.134 (0.013)	0.973 (0.011)	0.914 (0.042)	21.95 (5.54)	0.109 (0.039)
proposed (w/o token pos enc)	10.99 (2.29)	0.413 (0.104)	0.544 (0.068)	0.149 (0.033)	0.975 (0.009)	0.913 (0.037)	19.89 (3.82)	0.114 (0.044)
proposed (no ref)	10.20 (2.04)	0.311 (0.138)	0.666 (0.067)	0.235 (0.065)	0.943 (0.011)	0.866 (0.047)	26.89 (6.25)	0.107 (0.042)
proposed (guidance=0.99)	12.22 (1.92)	0.395 (0.140)	0.508 (0.061)	0.154 (0.032)	0.972 (0.011)	0.922 (0.043)	18.13 (4.57)	0.118 (0.057)
proposed (guidance=1.00)	12.22 (1.92)	0.395 (0.140)	0.508 (0.061)	0.154 (0.032)	0.972 (0.011)	0.922 (0.043)	18.13 (4.57)	0.118 (0.057)
proposed (guidance=2.00)	11.07 (2.25)	0.363 (0.135)	0.514 (0.073)	0.145 (0.044)	0.972 (0.012)	0.922 (0.038)	17.20 (4.32)	0.131 (0.056)
proposed (guidance=3.00)	10.43 (2.28)	0.348 (0.129)	0.539 (0.073)	0.165 (0.042)	0.968 (0.013)	0.919 (0.032)	16.91 (5.64)	0.122 (0.058)
proposed (guidance=5.00)	9.57 (2.14)	0.330 (0.125)	0.593 (0.066)	0.188 (0.041)	0.962 (0.018)	0.917 (0.035)	17.79 (6.15)	0.109 (0.055)
proposed (guidance=7.50)	8.80 (1.50)	0.310 (0.107)	0.648 (0.049)	0.251 (0.065)	0.940 (0.034)	0.897 (0.040)	24.03 (15.30)	0.097 (0.056)
proposed (overlap=0.00)	11.37 (2.18)	0.373 (0.144)	0.514 (0.079)	0.136 (0.031)	0.974 (0.009)	0.919 (0.035)	17.82 (5.25)	0.124 (0.052)
proposed (overlap=0.10)	11.34 (1.89)	0.372 (0.137)	0.507 (0.077)	0.140 (0.033)	0.970 (0.012)	0.920 (0.030)	16.52 (5.02)	0.128 (0.053)
proposed (overlap=0.50)	11.56 (2.11)	0.377 (0.140)	0.501 (0.082)	0.135 (0.040)	0.970 (0.010)	0.927 (0.029)	19.20 (7.96)	0.128 (0.055)
proposed (overlap=0.75)	11.71 (2.06)	0.378 (0.133)	0.499 (0.084)	0.135 (0.038)	0.971 (0.013)	0.932 (0.022)	18.43 (9.01)	0.119 (0.062)
proposed (row-by-row)	11.54 (2.16)	0.379 (0.142)	0.507 (0.070)	0.131 (0.026)	0.974 (0.011)	0.911 (0.045)	17.75 (4.98)	0.129 (0.051)
RealFill (warped ref)	10.39 (1.64)	0.309 (0.132)	0.667 (0.080)	0.248 (0.040)	0.932 (0.020)	0.884 (0.024)	60.56 (26.77)	0.016 (0.002)
proposed	11.35 (2.15)	0.374 (0.143)	0.508 (0.076)	0.137 (0.033)	0.971 (0.013)	0.917 (0.035)	17.97 (5.14)	0.130 (0.056)

Table S1. Ablation study. We evaluate the effects of (1) using a max frequency of 10Hz for the positional encoding (10Hz), (2) using 4 channels for the positional encoding (4 channels), (3) omitting the token positional encoding in the positional encoding (w/o token pos enc), (4) omitting the reference image during inference (no ref), (5) various guidance scales (guidance), (6) various overlap ratios (overlap), (7) tiling strategy, denoising row-by-row (row-by-row), and (8) RealFill trained with the warped reference images. We compare the generated results to a reference panorama produced using AutoStitch [8] on the tripod-captured dataset as in the main paper.

S2.5. Qualitative Evaluation on Tourist Spots

To demonstrate the applicability of our method to real-world scenarios with diverse image sources, we tested it on a set of images of the Trevi Fountain (Rome) sourced from online photo collections. These images exhibit variations in lighting, quality, and capture conditions, reflecting the challenging settings typical of internet-sourced data. The resulting panorama, generated using our proposed method, maintains high visual quality and coherence, seamlessly integrating the content from multiple input images. Figure S14 shows the input images and the resulting stitched panorama, highlighting the method's ability to handle diverse capture conditions

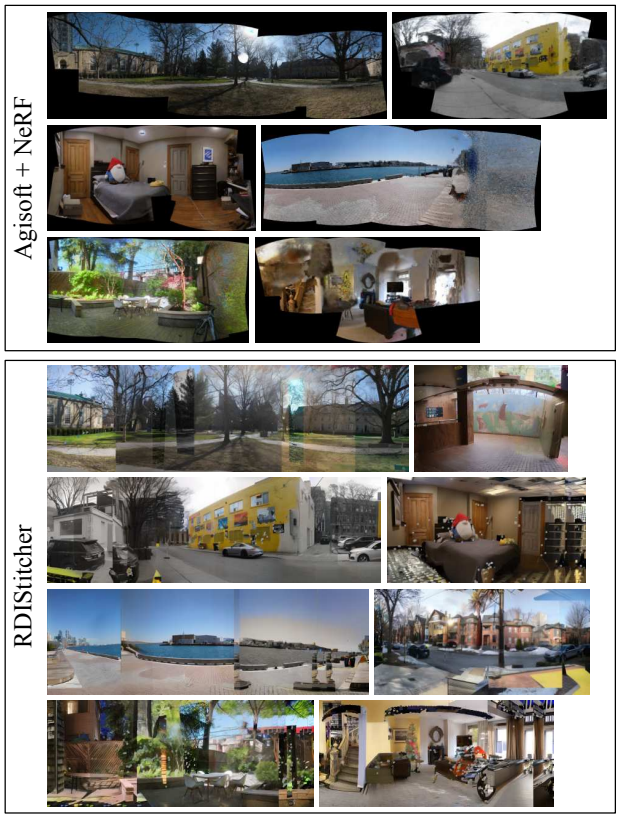

Figure S10. Qualitative results of Agisoft + NeRF and RDIStitcher on the casually captured dataset. RDIStitcher exhibits considerable banding and artifacts, and Agisoft + NeRF suffers from grainy renders and missing scene content. Note that for Agisoft + NeRF, two scenes ("Subway" and "Street") failed for all seeds due to AutoStitch failing.

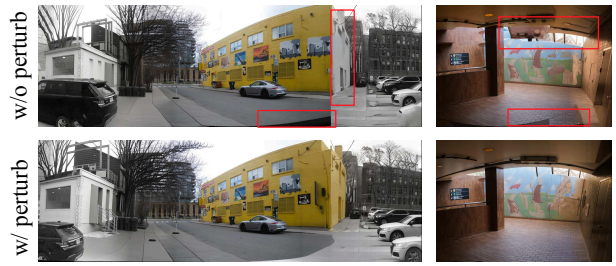

Figure S11. Qualitative evaluation of the ablation omitting the similarity transform used to perturb the location of the warped images in the sparse panoramas.

while producing a visually consistent output.

S2.6. Robustness to Layout Errors

To evaluate the robustness of our method to errors in the initial homography-based layout estimation, we conducted an experiment on the "Backyard" scene from our casually captured dataset. We randomly perturbed the horizontal and vertical positions of the images within the initial layout by 0, 100, 200, 300, and 400 pixels in each direction. For each perturbation level, we fine-tuned our model and evaluated its performance using multiple metrics, reporting the average and standard deviation over 10 inference seeds. The results, shown in Table **??**, indicate that the model remains relatively robust to moderate perturbations, with performance degrading as perturbations increase significantly. This suggests that our positional encoding mechanism helps mitigate errors in the initial layout estimation.

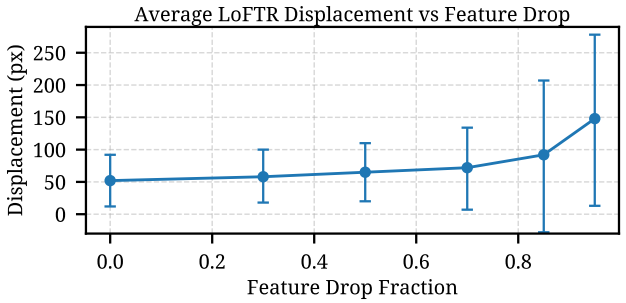

Figure S12. Alignment output robustness: misalignments are typically within 100 px, even when dropping 80% of correspondences.

S2.7. Performance vs. Panorama Size

We assessed the performance of our method as a function of output panorama size on the casually captured dataset, comparing it to baselines (RealFill, SD2) across panorama widths of 1000, 1500, 2000, 2500, and 3000 pixels. The results, presented in Table **??**, show that our method achieves superior performance across all panorama sizes and metrics. While photometric accuracy (e.g., PSNR) decreases with increasing panorama size, perceptual (DreamSim) and layout-based (LoFTR) metrics remain relatively stable, indicating that our method effectively preserves scene content and structure even for larger panoramas.

S2.8. Robustness to Misalignment

We evaluate robustness to misalignment by estimating realistic misalignment values by running AutoStitch [8] while randomly dropping alignment features. See Fig. S12: the resulting perturbations are below 100px up until a large feature dropout of >0.8. Hence, while at 100 px perturbation, LoFTR-based metrics degrade, this is level of misalignment is uncommon.

S2.9. Additional Experiments on Omni^2

Omni^2 [71] is a generative inpainting/outpainting model for omni-directional images. We apply the same iterative outpainting process as our approach for both model types, similar to the implementation for SD2. Results are shown in Figure S3. We note that Omni^2 fails in this task, showing large gaps in performance to our proposed method.

Figure S13. Qualitative comparison of various guidance scales on the casually-captured dataset. We find that increasing guidance leads to more "cartoonish" outputs and more seams, with a guidance scale around 1.5 showing best results.

Figure S14. Qualitative result of stitching images of the Trevi Fountain (Rome) sourced from online photo collections. The input images (left) exhibit variations in lighting, quality, and capture conditions. The resulting panorama (right) demonstrates seamless integration and high visual coherence.

Metric	P0	P100	P200	P300	P400
PSNR (↑)	10.83 ± 0.27	10.50 ± 0.16	10.26 ± 0.31	10.50 ± 0.17	9.83 ± 0.14
SSIM (↑)	0.211 ± 0.003	0.197 ± 0.004	0.201 ± 0.004	0.207 ± 0.003	0.193 ± 0.003
LPIPS (↓)	0.479 ± 0.013	0.519 ± 0.007	0.531 ± 0.010	0.544 ± 0.006	0.590 ± 0.007
DreamSim (↓)	0.143 ± 0.006	0.158 ± 0.011	0.165 ± 0.012	0.193 ± 0.015	0.223 ± 0.007
DINO (↑)	0.983 ± 0.002	0.981 ± 0.004	0.975 ± 0.003	0.966 ± 0.003	0.944 ± 0.006
CLIP (↑)	0.938 ± 0.019	0.961 ± 0.005	0.960 ± 0.011	0.954 ± 0.007	0.947 ± 0.005
LoFTR L2 Dist. (↓)	9.69 ± 1.46	26.49 ± 2.59	30.39 ± 4.29	38.65 ± 8.09	65.94 ± 6.88
LoFTR Match Prop. (↑)	0.209 ± 0.007	0.165 ± 0.010	0.129 ± 0.008	0.080 ± 0.009	0.059 ± 0.004

Table S2. Robustness to layout errors on the "Backyard" scene with varying perturbation levels. Metrics are averaged over 10 inference seeds, with standard deviations reported.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	DreamSim ↓	DINO ↑	CLIP ↑	LoFTR (L2 Distance) ↓	LoFTR (Matching) ↑
Omni^2 Inpainting	10.12 (1.24)	0.302 (0.042)	0.808 (0.044)	0.355 (0.040)	0.802 (0.021)	0.809 (0.023)	68.06 (22.95)	0.062 (0.027)
Omni^2 Outpainting	8.87 (0.64)	0.264 (0.013)	0.848 (0.018)	0.360 (0.021)	0.829 (0.013)	0.829 (0.018)	84.49 (12.28)	0.006 (0.000)
proposed	11.35 (2.15)	0.374 (0.143)	0.508 (0.076)	0.137 (0.033)	0.971 (0.013)	0.917 (0.035)	17.97 (5.14)	0.130 (0.056)

Table S3. Additional quantitative assessment with Omni^2 [71] of generative panoramic image stitching from casually captured images. We compare the generated results to a reference panorama using AutoStitch [8] on the tripod-captured dataset. Our approach generates panoramas that are close to the reference despite operating on images with parallax and variations in style or lighting.

Metric	Method	Width 1000	Width 1500	Width 2000	Width 2500	Width 3000
PSNR (↑)	Proposed	14.0132	13.1722	12.8197	12.0585	11.5414
	RealFill	13.7579	13.1322	12.5492	12.2542	11.5709
	SD2	12.2434	11.6118	11.2623	10.5282	10.1696
SSIM (↑)	Proposed	0.4985	0.4694	0.4324	0.4009	0.3788
	RealFill	0.4783	0.4575	0.4120	0.3867	0.3592
	SD2	0.4315	0.3768	0.3391	0.3035	0.2745
LPIPS (↓)	Proposed	0.3222	0.3826	0.4370	0.4758	0.4974
	RealFill	0.3454	0.4084	0.4899	0.5331	0.5804
	SD2	0.3997	0.4875	0.5564	0.6054	0.6353
DreamSim (↓)	Proposed	0.1059	0.1179	0.1273	0.1329	0.1307
	RealFill	0.1102	0.1231	0.1815	0.1843	0.2104
	SD2	0.1697	0.2401	0.2581	0.3137	0.2933
DINO (↑)	Proposed	0.9760	0.9697	0.9752	0.9737	0.9714
	RealFill	0.9714	0.9539	0.9513	0.9549	0.9509
	SD2	0.9583	0.9177	0.9143	0.9137	0.9127
CLIP (↑)	Proposed	0.9424	0.9337	0.9241	0.9227	0.9139
	RealFill	0.9434	0.9173	0.9106	0.9130	0.9204
	SD2	0.9215	0.8489	0.8479	0.8468	0.8512
LoFTR L2 Distance (↓)	Proposed	17.7942	22.6325	20.5730	18.7149	17.6509
	RealFill	22.9863	21.8171	27.4882	27.7174	36.9792
	SD2	37.0040	40.0898	45.0119	57.2499	63.2730
LoFTR Matching Proportion (↑)	Proposed	0.0222	0.0468	0.0736	0.1146	0.1326
	RealFill	0.0137	0.0242	0.0264	0.0280	0.0257
	SD2	0.0061	0.0098	0.0114	0.0121	0.0123