

# SphereDrag: Spherical Geometry-Aware Panoramic Image Editing

Zhiao Feng<sup>1</sup>, Xuewei Li<sup>1</sup> (✉), Junjie Yang<sup>1</sup>, Jingchao Li<sup>1</sup>, Yuxin Peng<sup>2</sup>, and Xi Li<sup>3</sup>

<sup>1</sup> School of Electronic and Information Engineering, Shanghai DianJi University, Shanghai 201306, China

xuweili@sdju.edu.cn

<sup>2</sup> Department of Sports Science, Zhejiang University, Hangzhou 310058, China

<sup>3</sup> College of Computer Science and Technology, Zhejiang University, Hangzhou 310058, China

**Abstract.** Image editing has made great progress on planar images, but panoramic image editing remains underexplored. Due to their spherical geometry and projection distortions, panoramic images present three key challenges: boundary discontinuity, trajectory deformation and uneven pixel density. To tackle these issues, we propose **SphereDrag**, a novel panoramic editing framework utilizing spherical geometry knowledge for accurate and controllable editing. Specifically, adaptive reprojection (AR) uses adaptive spherical rotation to deal with discontinuity; great-circle trajectory adjustment (GCTA) tracks the movement trajectory more accurate; spherical search region tracking (SSRT) adaptively scales the search range based on spherical location to address uneven pixel density. Also, we construct PanoBench, a panoramic editing benchmark, including complex editing tasks involving multiple objects and diverse styles, which provides a standardized evaluation framework. Experiments show that SphereDrag gains a considerable improvement compared with existing methods in geometric consistency and image quality, achieving up to 10.5% relative improvement.

**Keywords:** Interactive Point-based Editing · Diffusion Models · Panoramic Image.

## 1 Introduction

Image generation is one of the current research hotspots. Many works [18, 27, 29, 9] have made significant progress in controllable high-quality planar image generation, primarily by fine-tuning pre-trained large-scale diffusion models to fit various application scenarios. Numerous studies [24, 4, 16] have also been proposed for image editing, enabling highly controllable modifications to existing images based on new requirements. To gain a more comprehensive understanding of the scene, spherical panoramic images, also known as omnidirectional panoramic images or 360° panoramic images, are widely used in autonomous driving [13, 14], and virtual reality, etc.

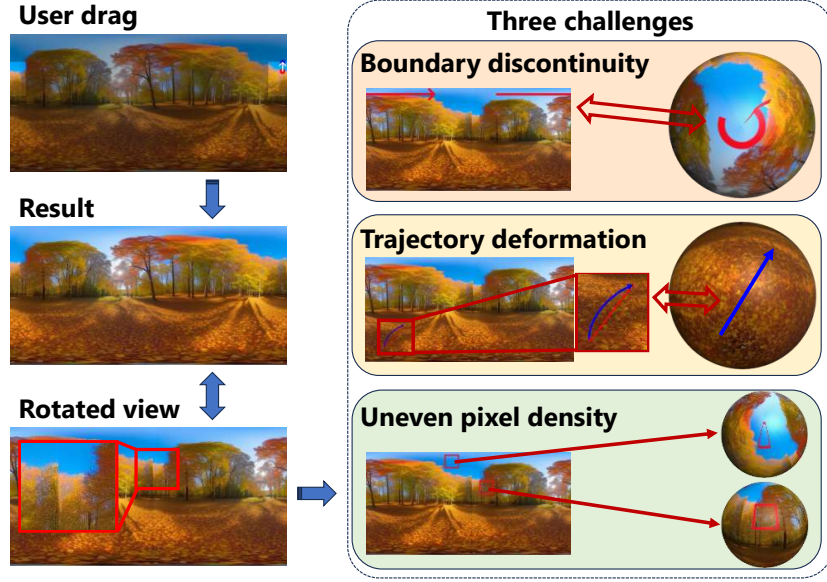


Fig. 1: Illustration of challenges in panoramic image editing. (Upper right) The panoramic image may divide movement trajectory into two parts, located near the left and right boundaries, respectively. (Middle right) Straight lines in the panoramic image do not correspond to great-circle paths on the sphere, leading to trajectory deviations. (Lower right) The same region in the panoramic image corresponds to unequal solid angles at different latitudes, causing non-uniform tracking across the sphere.

However, panoramic image editing is relatively underexplored. Compared with normal plane images, panoramic images have three unique challenges caused by spherical geometry and image distortion that are appeared during projection process (e.g., Equirectangular Projection, ERP). First, boundary discontinuity occurs when the trajectories split at the left and right boundaries of the image. Second, trajectory deformation arises because the straight trajectories on the sphere do not always correspond to straight trajectories in the panoramic images. Third, uneven pixel density results from the latitude-based distortion in panoramic images, which means the equal-size regions in the panoramic image correspond to different sizes on the sphere. These challenges are illustrated in fig. 1 in detail. How can we design the image editing model to deal with these challenges above and utilize the characteristics of spherical images to realize better panoramic image editing?

Faced with these challenges, we propose the SphereDrag framework, which aims to achieve accurate and controllable panoramic point-interactive image editing by leveraging spherical knowledge. We incorporate the characteristics of spherical panoramic images into the editing process. To address boundary discon-

tinuity, we propose adaptive reprojection (AR), reprojecting the input panoramic images, that provides a more intuitive and editable representation. Also, we propose great-circle trajectory adjustment (GCTA), incorporating spherical knowledge to ease trajectory deformation. Additionally, to address uneven pixel density, we propose spherical search region tracking (SSRT), which adaptively adjusts the search range based on spherical location information to mitigate non-uniform pixel density. For a fair and comprehensive image editing evaluation, we introduce a new spherical point-interactive image editing benchmark **PanoBench**, including a lot of scenes with multiple objects and diverse styles. It not only includes basic single-object editing tasks but also supports challenging scenarios where the interplay of multiple objects and stylistic transformations must be handled simultaneously. Experiments show that SphereDrag considerably outperforms existing methods in terms of geometric consistency and image quality. At a 30° Field of View (FOV) evaluation setting, a 10.5% relative improvement in IF is gained over the baseline, demonstrating the high-quality editing of SphereDrag. SphereDrag also achieves considerable reductions in both FID and sFID, indicating its advantages in both geometric accuracy and image quality. Our contributions are summarized as follows:

- We propose SphereDrag, a novel panoramic image editing framework that supports point-interactive editing by leveraging AR to address boundary discontinuity, great-circle trajectory adjustment to tackle GCTA, and SSRT to deal with uneven pixel density.
- We construct a spherical editing benchmark, PanoBench, which includes complex scenes with multiple objects and multiple styles, providing a standardized evaluation for panoramic image editing.
- Experiments show that SphereDrag considerably outperforms existing methods in terms of geometric consistency and image quality.

## 2 Related Work

The two most related fields of our work are image editing and spherical panoramic image generation, which are discussed in section 2.1 and section 2.2, respectively.

### 2.1 Image Editing

Image generation is a popular topic in image generation because of its user-friendly interaction. Early image editing methods are based on Generative Adversarial Networks (GANs) [6, 11], which made great progress in image generation but struggled with training instability and mode collapse [17, 1]. These limitations make researchers explore more advanced generation models. Diffusion models [8] become popular because of its ability to generate high-quality and highly controllable images through iterative denoising, and achieve remarkable improvements in both fidelity and diversity [23, 22].

Recently, many works focus on sophisticated image editing capabilities [12, 2]. Early ones integrate natural language with generation models to enable text-guided editing [21, 5], but these methods suffer from insufficient precision due to inherent textual ambiguity [27], limiting their effectiveness for fine-grained adjustments. To deal with these challenges, researchers develop more precise point-driven editing methods based on interactive drag operations. DragGAN [20] pioneered this approach by enabling intuitive local structure adjustments with direct user control. Building upon this foundation, subsequent works incorporate diffusion framework [30], leading to DragDiffusion [24], which leverages diffusion models’ advantages in maintaining global consistency while enabling local edits. DragNoise [16] further refine this paradigm through innovative noise optimization strategies that better balance local control with global coherence. StableDrag [4] introduces a robust framework for point-based image editing by enhancing DragGAN and DragDiffusion with a discriminative point tracking method and a confidence-based latent enhancement strategy.

The methods listed above all focus on normal plane image editing. To address the special challenges of panoramic image editing, SphereDrag deals with the three special issues, and extends point-driven editing strategy to the panoramic image editing task.

## 2.2 Spherical Panoramic Image Generation

In the contemporary panoramic image generation field, research methodologies are generally categorized into two main paradigms: GAN-based models and diffusion-based models.

In the domain of GAN-based [6, 11] approaches, COCO-GAN [15] introduces a coordinate-conditional framework that generates images in a divided manner, using spatial coordinates as a guiding signal for the generator to progressively synthesize local regions. Its discriminator evaluates global consistency, local appearance, and edge continuity, ensuring seamless output and enabling prediction beyond training dimensions via cylindrical coordinates. PanoGAN [25] utilizes an adversarial feedback mechanism along with a cross-view generation framework, achieving a breakthrough in the semantic matching between aerial perspective and ground-level panoramic images. By using a dual-branch discriminator and repeated optimization, it can ensure the feature matching at the pixel level and maintain the spatial consistency within multimodal data. Similarly, SphericGAN [3] adopts a semi-supervised hyperspherical latent space modeling strategy, aligning real and generated data clusters to capture category-independent variations effectively. Furthermore, CycleGAN [32] uses a cycle-consistent adversarial network, and addresses unpaired training data scenarios, making it possible to perform cross-domain image translation without explicit pixel-wise correspondences.

As diffusion-based methods appears, SphereDiffusion [26] introduces spherical geometry-aware training by enforcing rotational invariance in 360-degree space, ensuring boundary continuity in generated images. In contrast, CubeDiff [10] splits panoramic images into six cube faces, which helps create clearer,

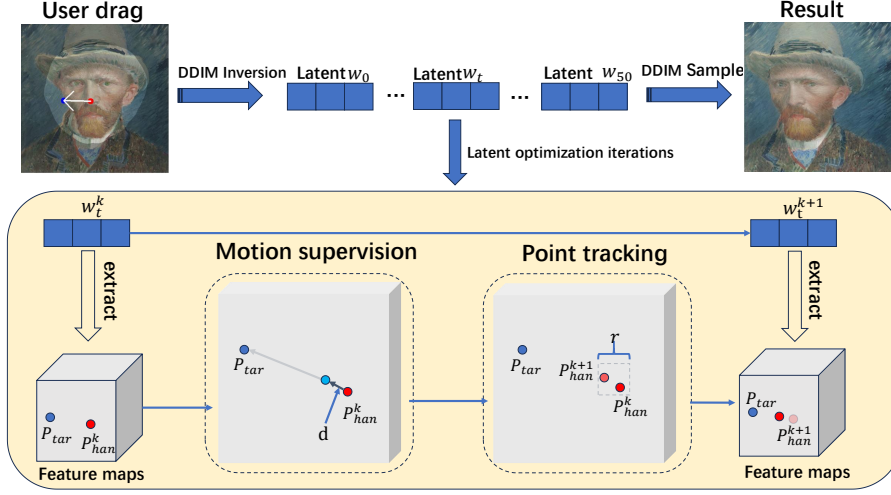


Fig. 2: Classic point-interactive image editing pipeline

distortion-free views compared to traditional equirectangular projections. It uses a lightweight solution with special attention and position encoding to generate high-quality panoramic images without complex cross-view attention. It also allows for precise control over details on different cube faces. Also, it uses tools (e.g., masks, bounding boxes, etc.) to guide content creation and reconstruct the missing region. Furthermore, TwinDiffusion [31] enhances coherence and efficiency in panoramic image generation by employing a training-agnostic optimization phase, maximizing similarity between adjacent cropped regions through Crop Fusion and dynamically adjusting sample selection via Cross Sampling. This framework establishes a new benchmark in balancing quality and computational efficiency for high-resolution panoramic synthesis. These developments highlight a progressive shift from GANs towards diffusion-based architectures, driven by the increasing demand for geometric adaptability, semantic robustness, and high-resolution controllability in panoramic image generation.

However, the above methods primarily focus on generation rather than editing. While these generation models have made significant progress in creating high-quality panoramic images, they often lack the capability for precise editing of existing panoramic images. SphereDrag introduces a new paradigm for panoramic image editing, addressing the unique challenges of spherical content and enabling fine-grained modifications by addressing boundary discontinuity, trajectory deformation, and uneven pixel density.

### 3 Method

In this section, we introduce our SphereDrag in detail. First, we briefly introduce classic image editing algorithm and build up our notations. The notations are summarized in Section A “Notations used in our method” in the Supplementary Material.

Next, we introduce our SphereDrag framework. Faced with boundary discontinuity challenge, we introduce our adaptive reprojection (AR) to get the suitable panoramic representation. Also, we introduce our spherical latent optimization, including great-circle trajectory adjustment (GCTA) and spherical search region tracking (SSRT), to deal with trajectory deformation and uneven pixel density, respectively.

#### 3.1 Background and Notations

As shown in fig. 2, classic point-interactive image editing methods (e.g., DragDiffusion) place a handle point  $P_{han}$  (in red), a target point  $P_{tar}$  (in blue), and a mask  $M$  (denoted by the brighter region of the user drag) on the user drag. The  $P_{han}$  is the starting point for the motion, while the  $P_{tar}$  indicates the desired destination. They define the direction and distance of the movement during editing together. The  $M$  specifies the editable region in the image, while the area outside the mask is the uneditable region. These methods apply DDIM inversion to obtain a sequence of latent codes, where each code corresponds to the intermediate representation of the image at a specific timestep. The latent code at fixed timestep  $t$ , denoted as  $w_t$ , is used to extract its corresponding UNet feature maps, which encodes the detailed features of the image. The feature maps are then iteratively optimized to modify the image according to the editing objective.

During each iteration, the current handle point as  $P_{han}^k$  is moved in the feature maps toward the target point  $P_{tar}$  using **motion supervision**. Afterward, the next handle point as  $P_{han}^{k+1}$  is located on the moved feature maps through **point tracking**. This process is repeated in each iteration while keeping the region outside the mask unchanged.

**Motion Supervision** In motion supervision, we first compute the normalized movement direction from the  $P_{han}^k$  to the  $P_{tar}$  at each iteration  $k$ . The normalized direction ensures that the movement from the handle to the target is consistent and precise, guiding the feature maps to move accordingly in the editing process:

$$\vec{d} = \frac{P_{tar} - P_{han}^k}{\|P_{tar} - P_{han}^k\|_2}, \quad (1)$$

where  $\vec{d}$  represents the direction of movement from the current handle point  $P_{han}^k$  to the target point  $P_{tar}$ , ensuring a controlled and consistent movement.

Finally, combining the normalized movement direction, the loss function is set as follows:

$$\begin{aligned} \mathcal{L}_{\text{ms}}(\hat{w}_t^k) = & \sum_{q \in \mathcal{F}} \left\| F_{q+\vec{d}}(\hat{w}_t^k) - \text{sg}(F_q(\hat{w}_t^k)) \right\|_1 \\ & + \lambda \left\| (\hat{w}_{t-1}^k - \text{sg}(\hat{w}_{t-1}^0)) \odot (1 - M) \right\|_1, \end{aligned} \quad (2)$$

The motion supervision loss  $\mathcal{L}_{\text{ms}}$  guides the update of the latent code  $\hat{w}_t^k$  during movement. **The first term** encourages feature maps extracted at spatial locations  $q \in \mathcal{F}$  to move toward a target position along the motion direction  $\vec{d}$ , where  $F_q(\cdot)$  denotes the feature at location  $q = (x, y)$ , and  $P_{\text{han}}^k$  is the handle point at the  $k$ -th iteration. **The second term** enforces consistency in uneditable regions. A binary mask  $M$  indicates editable ( $M = 1$ ) and uneditable ( $M = 0$ ) areas. It penalizes deviations between the current latent code  $\hat{w}_{t-1}^k$  and its initial version  $\hat{w}_{t-1}^0$  *only in the uneditable regions* ( $M = 0$ ), with the strength controlled by  $\lambda$  and using element-wise multiplication  $\odot$ . The stop-gradient operator  $\text{sg}(\cdot)$  prevents gradients from backpropagating through fixed reference features  $F_q(\hat{w}_t^k)$  and  $\hat{w}_{t-1}^0$ .

**Point Tracking** The point tracking process aims to find the next handle point  $P_{\text{han}}^{k+1}$  within the search region by minimizing the L1 norm of the difference in features. The search region is defined as:

$$\Omega_{\text{search}} = \{(x, y) \mid |x - p_x| \leq r, |y - p_y| \leq r\}, \quad p \in \{P_{\text{han}}^k\}. \quad (3)$$

Point tracking is defined as:

$$P_{\text{han}}^{k+1} = \arg \min_{q \in \Omega_{\text{search}}} \|F_q(\hat{w}_t^{k+1}) - F_{P_{\text{han}}}(w_t^0)\|_1, \quad (4)$$

where  $F_q(\cdot)$  represents the UNet feature maps extracted from the latent code at position  $q$ . The next handle point  $P_{\text{han}}^{k+1}$  is the point in the region  $\Omega_{\text{search}}$  that best matches the feature maps of the handle point  $P_{\text{han}}$ . The base radius  $r$  determines the size of the search region used in point tracking.  $\hat{w}_t^{k+1}$  denotes the latent code at the  $(k+1)$ -th iteration, and  $w_t^0$  denotes the initial latent code at timestep  $t$ .  $F_{P_{\text{han}}}(\cdot)$  represents the feature maps extracted from the latent code at the handle point  $P_{\text{han}}$ . It is a reference for the feature comparison to track the movement of the handle point across iterations.

### 3.2 SphereDrag Framework

When applied to panoramic images, classic point-interactive image editing faces three major panoramic challenges due to its reliance on planar assumptions: boundary discontinuity, trajectory deformation, and uneven pixel density. As illustrated in fig. 3, we introduce three corresponding modules to address these limitations. **AR** is proposed to deal with boundary discontinuity and achieve better image representation. **GCTA** and **SSRT** are designed to tackle trajectory deformation and uneven pixel density in spherical latent optimization, respectively.

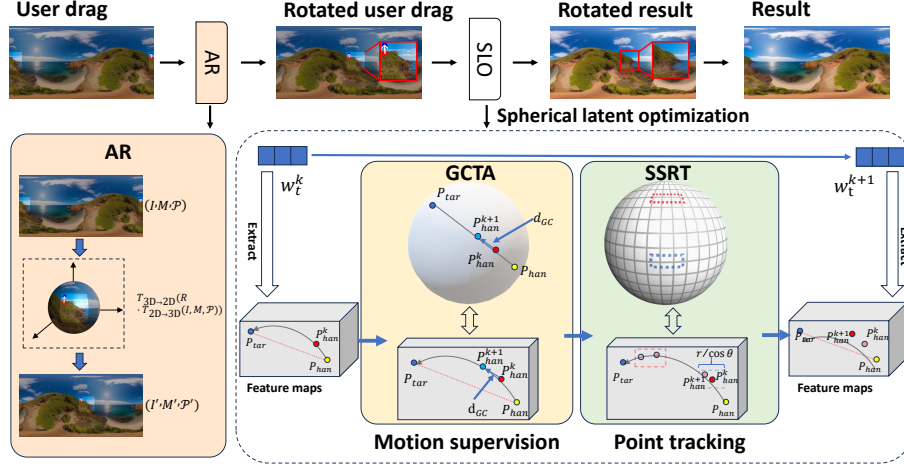


Fig. 3: Overview of SphereDrag. Using DragDiffusion as our baseline, we introduce our three parts: adaptive reprojection (AR), great-circle trajectory adjustment (GCTA), and spherical search region tracking (SSRT). **AR**: It applies spherical rotation to transform input panoramic images into a suitable representation. **GCTA**: It handles the points  $P_{tar}$ ,  $P_k$ , and  $P_{han}$  using the great-circle distance  $d_{gc}$  in a spherical manner. The underlying planar feature maps visualize the corresponding trajectory. **SSRT**: It highlights the current (blue) and future (red) search regions, which have equal sizes on the sphere. However, their projected areas differ on the feature maps due to spherical distortion.

**Adaptive Reprojection (AR)** As shown in fig. 1, trajectory discontinuity poses challenges to image editing: trajectories that cross the left and right image boundaries may appear as two disconnected segments. Inspired by the concept of *spherical symmetry*, the property that any rotation of the original sphere preserves the semantic content of a panoramic image, we introduce a spherical alignment strategy to mitigate these issues.

First, we compute the midpoint  $P_{mid} = (i_m, j_m)$  of the handle point  $P_{han} = (i_1, j_1)$  and the target point  $P_{tar} = (i_2, j_2)$ . This  $P_{mid}$  is projected from 2D coordinates to spherical coordinates  $(\theta, \phi)$  by a spherical projection function  $T_{2D \rightarrow \text{Sphere}}(\cdot)$ .

We then compute the rotation matrix  $R = \mathcal{R}_{\text{align}}(\theta, \phi)$  that aligns the midpoint direction to get a suitable panoramic representation, which is minimally affected by spherical issues (e.g., the center of the panoramic image). This rotation is performed on the sphere for the image  $I$ , the mask  $M$ , and the handle point  $P_{han}$  and target point  $P_{tar}$  within  $\mathcal{P}$ . Then, the rotated results are projected back to 2D using  $T_{\text{Sphere} \rightarrow 2D}(\cdot)$ . The overall spherical alignment process is computed as follows:

$$(I', M', \mathcal{P}') = \mathcal{T}_{\text{align}}(I, M, \mathcal{P}), \quad (5)$$



$$\mathcal{T}_{\text{align}}(I, M, \mathcal{P}) = T_{\text{Sphere} \rightarrow 2\text{D}}(R \cdot T_{2\text{D} \rightarrow \text{Sphere}}(I, M, \mathcal{P})), \quad (6)$$

where the function  $\mathcal{T}_{\text{align}}(\cdot)$  denotes the complete alignment pipeline.  $I'$  denotes the image,  $M'$  denotes the mask, and  $\mathcal{P}'$  denotes the handle point and target point.  $T_{2\text{D} \rightarrow \text{Sphere}}(\cdot)$  is the transformation that projects the input data  $(I, M, \mathcal{P})$  from 2D coordinates to spherical coordinates. The rotation matrix  $R$  is applied to these spherical coordinates to align them with a suitable panoramic representation. Finally,  $T_{\text{Sphere} \rightarrow 2\text{D}}(\cdot)$  projects the rotated data back to 2D coordinates, resulting in the final image, mask, handle point and target point,  $(I', M', \mathcal{P}')$ . As shown in fig. 3, our spherical alignment strategy makes the editing region more stable semantic consistency and trajectory continuity. The further details are provided in Section B “Detail for  $\mathcal{T}_{\text{align}}(\cdot)$  in Adaptive Reprojection (AR)” in the Supplementary Material.

**Great-Circle Trajectory Adjustment (GCTA)** In classic image editing, motion supervision typically computes the straight movement direction between the current handle point and the target point based on 2D planar assumptions. However, when applied to panoramic images, this planar approximation deviates from the correct direction on the sphere, resulting in motion distortions.

To mitigate this challenge, we replace the 2D straight direction with the spherical great-circle direction. Specifically, at the  $k$ -th iteration, given the current handle point  $P_{han}^k$ , the initial handle point  $P_{han}$ , and the target point  $P_{tar}$ , the motion direction is defined as:

$$\vec{d} = \mathcal{G}_{\text{gc}}(P_{han}^k, P_{han}, P_{tar}), \quad (7)$$

where  $\mathcal{G}_{\text{gc}}(\cdot)$  denotes the great-circle direction computation function (see Section C “Detail for Great Circle Direction Calculation” in the Supplementary Material).

Based on this direction, the motion supervision loss is defined as:

$$\begin{aligned} \mathcal{L}_{\text{ms}}(\hat{w}_t^k) = & \sum_{q \in \mathcal{F}} \left\| F_{q+\vec{d}}(\hat{w}_t^k) - \text{sg}(F_q(\hat{w}_t^k)) \right\|_1 \\ & + \lambda \left\| (\hat{w}_{t-1}^k - \text{sg}(\hat{w}_{t-1}^0)) \odot (1 - M) \right\|_1. \end{aligned} \quad (8)$$

By using the great-circle direction  $\vec{d}$ , the original planar motion supervision is naturally adapted to the spherical domain. This ensures that the movement follows the correct direction on the sphere instead of a straight line on the 2D plane, effectively eliminating the deviation introduced by the planar approximation. The other variables follow the formulation in eq. (2).

Compared to conventional planar motion supervision, our GCTA more accurately captures directional behavior on the spherical domain, particularly in high-latitude regions where projection distortion is most severe.

**Spherical Search Region Tracking (SSRT)** In classic image editing, the search region for point tracking is typically defined as a fixed-size square window, based on planar assumptions. However, when applied to panoramic images,

such a design fails to account for the non-uniform distortions introduced by ERP. Consequently, the search region becomes overstretched in the high-latitude area, resulting in a degraded tracking accuracy, which constitutes the challenge officially defined as uneven pixel density.

To address this issue, we propose SSRT, which adaptively adjusts the search region according to the spherical location of the current handle point. Specifically, we project the current handle point  $P_{han}^k$  from 2D planar coordinates to spherical coordinates  $(\theta, \phi)$  using the mapping function  $T_{2D \rightarrow \text{Sphere}}(\cdot)$ , where  $\theta$  and  $\phi$  denote longitude and latitude, respectively.

To preserve consistent pixel density across latitudes, we scale the vertical radius of the tracking region as a function of latitude:

$$r(\phi) = \frac{r}{\cos \phi}, \quad (9)$$

where  $r$  is a fixed base radius. The use of  $\cos \phi$  compensates for the horizontal stretching caused by the ERP in high-latitude areas, improving the geometric consistency of the search region and improving the accuracy of the tracking. An illustration of how  $\cos \phi$  relates to latitude distortion is provided in Section D “Stretching Factor and Latitude Distortion” in the Supplementary Material.

Based on this, we define the spherical-adaptive search region centered at the current handle point  $P_{han}^k$  as:

$$\Omega_{\text{search}} = \{(x, y) \mid |x - p_x| \leq r_0, |y - p_y| \leq r(\phi)\}, \quad p \in P_{han}^k. \quad (10)$$

Within this region  $\Omega_{\text{search}}$ , the updated handle point is determined by minimizing the L1 distance between UNet features:

$$P_{han}^{k+1} = \arg \min_{q \in \Omega_{\text{search}}} \|F_q(\hat{w}_t^{k+1}) - F_{P_{han}}(w_t^0)\|_1, \quad (11)$$

where  $F_q(\cdot)$  denotes the UNet feature extracted from the latent code at position  $q$ , and  $F_{P_{han}}(w_t^0)$  is the reference feature from the initial handle point in the unoptimized latent code.

By incorporating spherical-aware scaling into the search region, our method better aligns with the geometric properties of panoramic images. This adaptive adjustment enhances the robustness and accuracy of handle point tracking, particularly in high-latitude regions where distortions caused by ERP are more pronounced.

## 4 Experiments

In this section, we conduct experiments to evaluate SphereDrag. Firstly, we introduce the benchmark, experimental protocols, and evaluation metrics used in our experiments. Secondly, we compare the performance of SphereDrag with state-of-the-art image editing methods. Third, we carry out ablation experiments to clarify the contribution of each module. Finally, we provide a detailed discussion of the experimental hyperparameters in Section E “Discussion” in the Supplementary Material.

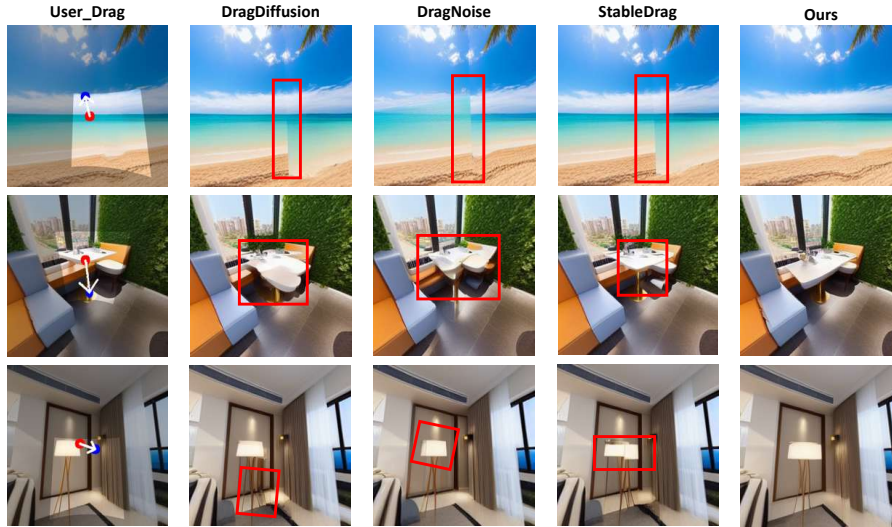


Fig. 4: 90° FOV drag visualization: In the first row, our method effectively handles the seams at the sandy beach with strong winds, while other methods exhibit visible boundary artifacts. In the second row, our method successfully follows the intended dragging path, whereas other methods either fail to complete the drag or produce distorted results. In the third row, our approach accurately interprets the dragging intent. When dragging the lamp, the surrounding elements remain properly arranged, while other methods either fail to move the lamp correctly or result in chaotic outputs.

#### 4.1 Benchmark, Protocols, and Evaluation Metrics

**Benchmark** Classic image editing Dragbench benchmarks utilize planar images, which is not suitable for panoramic image editing evaluation for lack of panoramic characteristics. We construct **PanoBench**, a novel benchmark comprising  $1024 \times 512$  panoramic images. It includes 81 panoramic images of different scenes, along with the corresponding masks, handle points, and target points for drag-based editing evaluation. These images feature a wide variety of objects and scenarios, including windows, sofas, lamps, mountains, coastlines, and more, providing a diverse and realistic benchmark for evaluating panoramic image editing methods.

**Evaluation Metrics** Image Fidelity (IF), Fréchet Inception Distance (FID), and Spatial Fréchet Inception Distance (sFID) are used as evaluation metrics. Detailed definitions and formulations of these metrics are provided in Section F “Evaluation Metrics” in the Supplementary Material.

**Experimental Protocols** All experiments are based on Stable Diffusion v1.5 with a uniform learning rate of 0.01. To enhance the model’s fidelity for panoramic scenes, the training process involves fine-tuning the cross-attention

Table 1: Quantitative results of our method (**Ours**) and state-of-the-art models. All methods are evaluated on images with FOV of 30°, 60°, and 90°. Metrics include IF (higher is better), FID, and sFID (lower is better).

Method	30°			60°			90°		
	IF↑	FID↓	sFID↓	IF↑	FID↓	sFID↓	IF↑	FID↓	sFID↓
Baseline	0.6666	125.6694	409.8609	0.8037	80.4527	277.0527	0.8924	64.0885	182.5420
DragNoise	0.7161	126.3004	391.0360	0.8047	88.8964	302.2573	0.8734	77.1288	226.8186
StableDrag	0.7133	116.4910	390.5301	0.8312	72.2143	253.5878	0.9064	57.7487	168.7382
<b>Ours</b>	<b>0.7364</b>	<b>114.8911</b>	<b>375.9909</b>	<b>0.8495</b>	<b>70.4464</b>	<b>244.3402</b>	<b>0.9191</b>	<b>52.8041</b>	<b>160.2049</b>

layers of its UNet architecture via Low-Rank Adaptation (LoRA). We set the latent variable at fixed timestep  $t = 35$ , the strength of the constraint  $\lambda = 0.1$ , the suitable panoramic representation, we set the longitude to 0, The latitude remains consistent before and after rotation and maintain a resolution of the  $1024 \times 512$  image. The implementation runs on servers equipped with 8 NVIDIA L40S GPUs. The rest of the protocols are the same as DragDiffusion.

## 4.2 Performance Comparison

In this section, we evaluate our SphereDrag method against the latest image editing models, including DragDiffusion [24], DragNoise [16], and StableDrag [4]. For this evaluation, we extract perspective views from panoramic images at FOVs of 30°, 60°, and 90°, enabling comprehensive evaluation of model performance across diverse viewing angles. As shown in table 1, at a 30° FOV, where the focus is concentrated on the dragged region, the performance differences between the models are particularly pronounced. However, as the FOV increases and more undragged areas are incorporated into the generation process, these discrepancies gradually diminish. As shown in fig. 4, the figure visualizes the performance variation in different FOVs.

## 4.3 Ablation Studies

In this section, we analyze the contribution of each module to overall performance on images with FOV angles of 30°, 60°, and 90°. The evaluation is conducted using the IF metric. To provide a more thorough assessment, we also incorporate the sFID metric in an extended experiment.

The results are summarized in table 2. We consider several configurations, including a baseline model (without any our module), a model with adaptive reprojection (AR), and extended versions incorporating great-circle trajectory adjustment (GCTA) and spherical search region tracking (SSRT).

The results show that AR consistently improves performance across all FOV settings. Adding GCTA brings further gains, and combining AR, GCTA, and SSRT achieves the best results. Section G, “Effectiveness of Adaptive Reprojection,” in the Supplementary Material provides visualizations demonstrating

Table 2: Ablation study results comparing different model variants on images with FOV of 30°, 60°, and 90°. AR / GCTA / SSRT denote adaptive reprojection, great-circle trajectory adjustment, and spherical search region tracking, respectively.

AR	GCTA	SSRT	IF↑			sFID↓		
			30°	60°	90°	30°	60°	90°
$\times$	$\times$	$\times$	0.6666	0.8037	0.8924	409.86	277.05	182.54
✓	$\times$	$\times$	0.7060	0.8321	0.9101	398.08	260.21	172.47
✓	✓	$\times$	0.7081	0.8326	0.9103	399.30	259.97	172.21
✓	✓	✓	<b>0.7364</b>	<b>0.8495</b>	<b>0.9191</b>	<b>375.99</b>	<b>244.34</b>	<b>160.20</b>

that AR significantly enhances semantic continuity and geometric structure in the output panoramic images.

## 5 Conclusion

Faced with three major challenges in spherical panoramic image editing: boundary discontinuity, trajectory deformation, and uneven pixel density. We propose SphereDrag, a panoramic image editing framework designed to produce high-quality, easily controllable, and precisely manipulated spherical panoramic images. For boundary discontinuity, we introduce adaptive reprojection, placing the region that needs to be manipulated in the suitable location of the panoramic image. For trajectory deformation and uneven pixel density, we propose our spherical latent optimization composed of great-circle trajectory adjustment and spherical search region tracking to accurately track the movement trajectory of the editing point. Also, we construct a panoramic image editing benchmark, PanoBench, to compare SphereDrag with other methods in spherical panoramic image editing task comprehensively. Experimental results show that SphereDrag achieves the state-of-the-art quantitative and qualitative performance.

## Acknowledgements

This work is supported in part by Natural Science Foundation of Shanghai under Grant 24ZR1425600, “Pioneer” and “Leading Goose” R&D Program of Zhejiang (No.2025C02014), Ningbo Science and Technology Special Projects under Grant No. 2025Z028, the Chenguang Program of Shanghai Education Development Foundation and Shanghai Municipal Education Commission with Grant No. 24CGA73, and the Fundamental Research Funds for the Central Universities.

## Supplementary Material: “SphereDrag: Spherical Geometry-Aware Panoramic Image Editing”

### A Notations used in our method

---

$sg(\cdot)$	Stop gradient operation, preventing gradient backpropagation.
$P_{\text{han}}/P_{\text{tar}}$	The handle point / the target point.
$P_{\text{mid}}$	The midpoint between the handle and target point.
$P_{\text{han}}^k$	The current handle point at the $k$ -th iteration.
$r$	The base radius determining the search region size.
$(\theta, \phi)$	The spherical coordinates: longitude $\theta$ , latitude $\phi$ .
$\mathcal{F}$	The feature maps.
$\Omega_{\text{search}}$	The search region.
$\hat{w}_t^k$	The latent variable at fixed timestep $t$ after the $k$ -th iteration.
$w_t^0$	The initial latent variable at timestep $t$ .
$T_{\text{Sphere} \rightarrow \text{2D}}(\cdot)$	The projection from spherical to 2D coordinates.
$T_{\text{2D} \rightarrow \text{Sphere}}(\cdot)$	The projection from 2D to spherical coordinates.
$M$	A binary mask indicating editable vs. uneditable regions.
$\vec{d}$	The direction vector from the current handle point to the target point.
$\mathcal{L}_{\text{ms}}$	The motion supervision loss.
$F_q(\cdot)$	Feature maps extracted at location $q$ .
$F_q(\hat{w}_t^k)$	Feature maps extracted from $\hat{w}_t^k$ at $q$ , with spherical alignment and resizing.
$\mathcal{R}_{\text{align}}(\theta, \phi)$	Rotation to align $(\theta, \phi)$ with panoramic view.
$\mathcal{G}_{\text{gc}}(\cdot)$	Great-circle direction computation.

---

## B Detail for $\mathcal{T}_{\text{align}}(\cdot)$ in Adaptive Reprojection (AR)

In this section, we describe the details of the  $\mathcal{T}_{\text{align}}(\cdot)$  process. Given an input panoramic image with width  $W$ , height  $H$ , corresponding handle point  $P_{\text{han}}$ , target point  $P_{\text{tar}}$ , and masks  $M$ . The handle point in the image is  $P_{\text{han}} = (i_1, j_1)$ , and the target point is  $P_{\text{tar}} = (i_2, j_2)$ . The midpoint  $P_{\text{mid}} = (i_m, j_m)$  is the average of the handle and target pixel coordinates  $i_m = \frac{i_1+i_2}{2}$ ,  $j_m = \frac{j_1+j_2}{2}$ . Then we converted to spherical coordinates (latitude  $\theta$  and longitude  $\phi$  as follow:

$$\theta = \frac{\pi}{2} - \frac{j}{H} \cdot \pi, \quad \phi = \frac{i}{W} \cdot 2\pi - \pi. \quad (\text{S-1})$$

After calculating the spherical coordinates, we align the image by adjusting the longitude  $\phi$  and latitude  $\theta$  of the midpoint to match a suitable panoramic representation.

**Rotation to align the longitude:** As before, the rotation is used to align the image's midpoint to the desired longitude. This rotation matrix  $R_\phi$  is given by:

$$R_\phi(\Delta\phi) = \begin{bmatrix} \cos(\Delta\phi) & -\sin(\Delta\phi) & 0 \\ \sin(\Delta\phi) & \cos(\Delta\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (\text{S-2})$$

where  $\Delta\phi$  is the angle required to adjust the longitude of the midpoint to the desired longitude.

**Rotation to align the latitude:** Once the longitude is aligned, the rotation is used to align the image's midpoint to the desired latitude. This rotation matrix  $R_\theta$  is given by:

$$R_\theta(\Delta\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\Delta\theta) & -\sin(\Delta\theta) \\ 0 & \sin(\Delta\theta) & \cos(\Delta\theta) \end{bmatrix}, \quad (\text{S-3})$$

where  $\Delta\theta$  is the angle required to adjust the latitude of the midpoint to the desired latitude.

**Combined Rotation Matrix:** The final rotation matrix  $R$  is the combination of both rotations. The combined rotation matrix is given by the matrix multiplication:

$$R(\Delta\theta, \Delta\phi) = R_\theta(\Delta\theta) \cdot R_\phi(\Delta\phi), \quad (\text{S-4})$$

Finally, the rotation matrix  $R$  is applied to each pixel  $i$  in the panoramic image  $I$ , the mask  $M$ , and the point set  $\mathcal{P}$ , which includes the handle and target points. Since the mask and points are defined in the same coordinate system as the image, applying  $R$  to these elements preserves their spatial relationships. The 3D vector  $\mathbf{p}_i = (x_i, y_i, z_i)$  for each pixel is rotated to obtain the new 3D vector  $\mathbf{p}_i^{\text{new}} = R \cdot \mathbf{p}_i$ . These new 3D coordinates are then projected back onto the image plane using inverse spherical mapping, resulting in the transformed image  $I'$ , mask  $M'$ , and points  $\mathcal{P}'$ .

## C Detail for Great Circle Direction Calculation

In this section, we describe the great-circle movement direction computation in detail. This method approximates the optimal direction for moving from a current position toward a target along a great circle on a sphere.

Given the handle point  $P_{\text{han}}$ , the target point  $P_{\text{tar}}$ , and the current handle point  $P_{\text{han}}^k$  in panoramic image 2D coordinates, we first convert these points into spherical coordinates, and then into 3D coordinates to facilitate the great-circle calculations.

The conversion from panoramic image coordinates to spherical coordinates (latitude  $\theta$  and longitude  $\phi$ ) for a pixel at position  $(i, j)$  in panoramic image (width  $W$  and height  $H$ ) coordinates is given by:

$$\theta = \frac{\pi}{2} - \frac{j}{H} \cdot \pi, \quad \phi = \frac{i}{W} \cdot 2\pi - \pi, \quad (\text{S-5})$$

where  $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$ ,  $\phi \in (-\pi, \pi]$ ,  $\theta = 0$  indicating equatorial, and  $\phi = 0$  indicating meridian (the middle of the panoramic image).

These spherical coordinates are then transformed into 3D Cartesian coordinates on the unit sphere using the standard conversion:

$$\begin{aligned} x &= \cos(\theta) \cos(\phi), \\ y &= \cos(\theta) \sin(\phi), \\ z &= \sin(\theta). \end{aligned} \quad (\text{S-6})$$

This yields the 3D unit vectors  $\mathbf{P}_{\text{han}}$ ,  $\mathbf{P}_{\text{tar}}$ , and  $\mathbf{P}_{\text{han}}^k$  corresponding to the handle point, target point, and current handle point, respectively.

To compute the great-circle movement direction between the handle and target points, we first determine the unit normal vector to the plane containing the great circle. This unit normal vector  $\mathbf{n}$  is given by the normalized cross product of the handle and target vectors:

$$\mathbf{n} = \frac{\mathbf{P}_{\text{han}} \times \mathbf{P}_{\text{tar}}}{\|\mathbf{P}_{\text{han}} \times \mathbf{P}_{\text{tar}}\|}. \quad (\text{S-7})$$

Since the current position  $\mathbf{P}_{\text{han}}^k$  may not lie exactly on this great circle, we orthogonally project it onto the plane defined by  $\mathbf{n}$ :

$$\mathbf{P}' = \frac{\mathbf{P}_{\text{han}}^k - (\mathbf{P}_{\text{han}}^k \cdot \mathbf{n})\mathbf{n}}{\|\mathbf{P}_{\text{han}}^k - (\mathbf{P}_{\text{han}}^k \cdot \mathbf{n})\mathbf{n}\|}. \quad (\text{S-8})$$

This projection  $\mathbf{P}'$  lies on the unit sphere and is the closest point on the great-circle path to the current position.

The desired movement direction in 3D space is then computed as the normalized vector from  $\mathbf{P}'$  to the target:

$$\mathbf{v}_{\text{move}} = \frac{\mathbf{P}_{\text{tar}} - \mathbf{P}'}{\|\mathbf{P}_{\text{tar}} - \mathbf{P}'\|}. \quad (\text{S-9})$$



To interpret this 3D direction in a local 2D coordinates, we construct an orthonormal basis on the tangent plane of the sphere at the current position  $\mathbf{P}_{\text{han}}^k$ . This basis consists of two vectors, the zonal (eastward) vector  $\hat{\mathbf{e}}_\phi$  and the meridional (northward) vector  $\hat{\mathbf{e}}_\theta$ , their unit vectors are:

$$\hat{\mathbf{e}}_\phi = \frac{1}{\cos(\theta)} \frac{\partial \mathbf{P}}{\partial \phi} = [-\sin(\phi), \cos(\phi), 0]^T, \quad (\text{S-10})$$

$$\hat{\mathbf{e}}_\theta = \frac{\partial \mathbf{P}}{\partial \theta} = [-\sin(\theta) \cos(\phi), -\sin(\theta) \sin(\phi), \cos(\theta)]^T. \quad (\text{S-11})$$

Project  $\mathbf{v}_{\text{move}}$  onto this tangent plane and make vector decomposition based on the orthonormal basis above:

$$\begin{bmatrix} v_\phi \\ v_\theta \end{bmatrix} = \begin{bmatrix} \mathbf{v}_{\text{move}} \cdot \hat{\mathbf{e}}_\phi \\ \mathbf{v}_{\text{move}} \cdot \hat{\mathbf{e}}_\theta \end{bmatrix}. \quad (\text{S-12})$$

the tangent plane at  $\mathbf{P}_{\text{han}}^k$  does not correspond to the panoramic image plane, but the zonal (eastward) component are scaled by  $1/\cos(\theta)$  due to the ERP:

$$\vec{d} = \frac{\left[ \frac{v_\phi}{\cos(\theta)}, v_\theta \right]}{\left\| \left[ \frac{v_\phi}{\cos(\theta)}, v_\theta \right] \right\|}. \quad (\text{S-13})$$

This normalized vector  $\vec{d}$  represents the final movement direction in the panoramic image coordinate and can be directly interpreted as a direction vector in pixel space.

## D Stretching Factor and Latitude Distortion

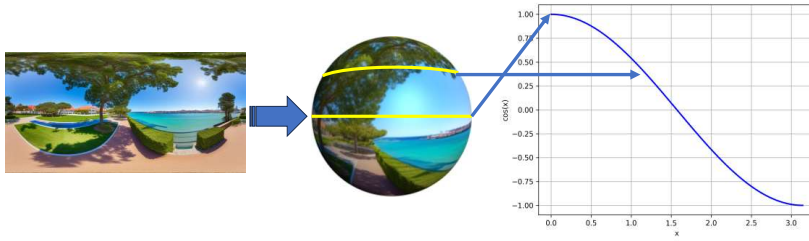


Fig. S-1: In a panoramic image,  $\cos \phi$  acts as the stretching factor, ensuring that the degree of stretching in the latitude direction from the sphere to the plane is consistent with the spherical geometry. The variation of  $\cos \phi$  with latitude  $\phi$  precisely reflects the projection requirements of the parallel lengths on the sphere, thus maintaining the accuracy and consistency of geographic information during the projection process.

## E Evaluation Metrics

- **IF** is a metric that quantifies the visual similarity between the original and edited images. It is calculated by first computing the LPIPS [28] values between the original and edited images, averaging these values, and then subtracting the average LPIPS score from 1:

$$\text{IF} = 1 - \text{avg}(\text{LPIPS}), \quad (\text{S-14})$$

where  $\text{avg}(\text{LPIPS})$  denotes the mean LPIPS value over all image patches or samples. A higher IF value indicates better preservation of the original image’s identity.

- **FID** [7] is a widely used metric for evaluating generative models. It measures the similarity between the feature distributions of real and generated samples in a feature space. The calculation involves extracting features using the Inception-v3 model, computing the mean ( $\mu$ ) and covariance matrix ( $\Sigma$ ) for both real and generated samples, and then computing FID as:

$$\text{FID} = \|\mu_1 - \mu_2\|_2^2 + \text{Tr}\left(\Sigma_1 + \Sigma_2 - 2(\Sigma_1 \Sigma_2)^{1/2}\right), \quad (\text{S-15})$$

where  $\mu_1, \Sigma_1$  are the mean and covariance of real samples, and  $\mu_2, \Sigma_2$  are those of generated samples.

- **sFID** [19] is a variant of FID that uses spatial features instead of global features. Like FID, it compares the feature distributions of real and generated samples, but focuses on spatial information rather than pooled features. The computation formula is identical to FID, applied over spatial features.

IF indicates editing accuracy when both FID and sFID show the structure and semantic preservation capabilities of models. The multimetric evaluation considers both editing quality and panoramic structure maintenance.

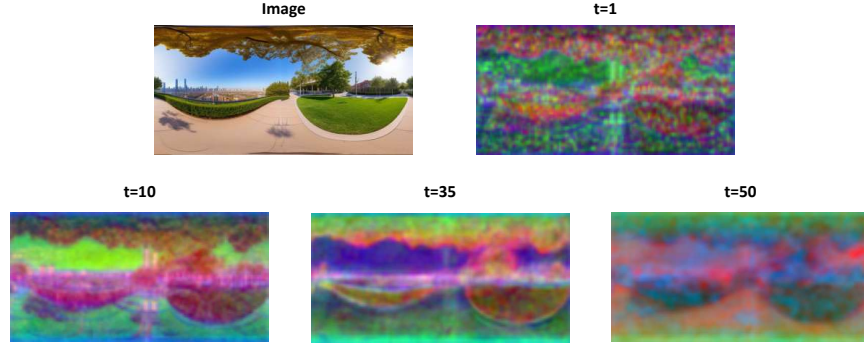


Fig. S-2: PCA visualization results. At smaller timesteps (e.g.,  $t = 1, 10$ ), the latent variable contains redundant details, while at larger timesteps (e.g.,  $t = 50$ ), the spatial information is overly diminished. At  $t = 35$ , a good balance is achieved with fewer noise and richer spatial information, effectively preserving spatial coherence.

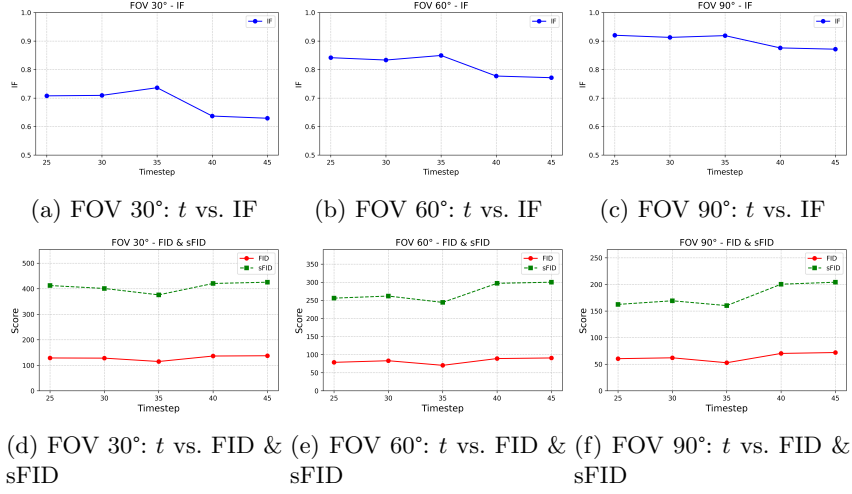


Fig.S-3: Influence of different hyperparameters  $t$  across various FOV settings ( $30^\circ$ ,  $60^\circ$ , and  $90^\circ$ ). The top row (a-c) shows the relationship between  $\lambda$  and the IF value, while the bottom row (d-f) shows  $\lambda$  versus FID and sFID for the corresponding FOV.

## F Discussion

In this section, we discuss the influence of different hyperparameter  $t$  and the  $\lambda$ .

**Hyperparameters  $t$**  As shown in fig. S-3, based on the evaluation in three FOVs ( $30^\circ$ ,  $60^\circ$  and  $90^\circ$ ),  $t = 35$  demonstrates consistently strong performance in IF, FID and sFID, with noticeable improvements in narrower FOVs ( $30^\circ$  and  $60^\circ$ ) and competitive results in wider FOV ( $90^\circ$ ), indicating robust generalization. In comparison, configurations such as  $t = 30$ ,  $t = 40$ , and  $t = 45$  display more variability and certain limitations in different settings. Furthermore, the PCA visualization in fig. S-2 reveals that  $t = 35$  achieves a favorable balance between noise suppression and spatial information retention, offering latent representations that effectively preserve spatial coherence. Overall,  $t = 35$  emerges as a reasonable default choice, offering balanced and reliable performance across a wide range of FOV conditions.

**Hyperparameter  $\lambda$**  controls the proportion of the second term in the motion supervision loss. Specifically, we evaluate  $\lambda$  values of 0, 0.05, 0.1, 0.15, and 0.2. When  $\lambda = 0$ , the second term of the motion supervision loss is removed, leading to a considerable decrease in IF, FID and sFID, as shown in fig. S-4, which highlights the importance of this loss term in improving the model’s performance. When  $\lambda > 0$ , the performance generally improves, and is not sensitive to  $\lambda$ . The performance fluctuations remain small, demonstrating the robustness of our method with respect to  $\lambda$ .

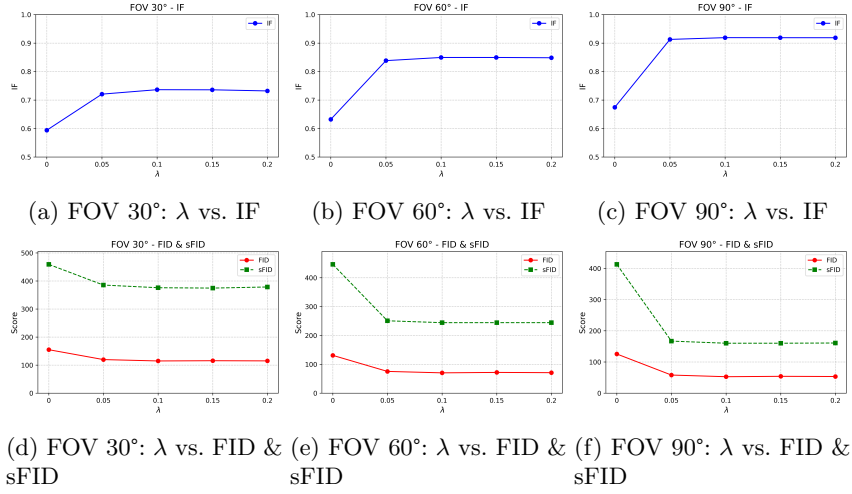


Fig.S-4: Influence of different hyperparameters  $\lambda$  across various FOV settings (30°, 60°, and 90°). The top row (a-c) shows the relationship between  $\lambda$  and the IF value, while the bottom row (d-f) shows  $\lambda$  versus FID and sFID for the corresponding FOV.

By comparing the performance across different FOV settings, we set  $\lambda = 0.1$  finally for its best and most stable performance across all conditions.

## G Effectiveness of Adaptive Reprojection

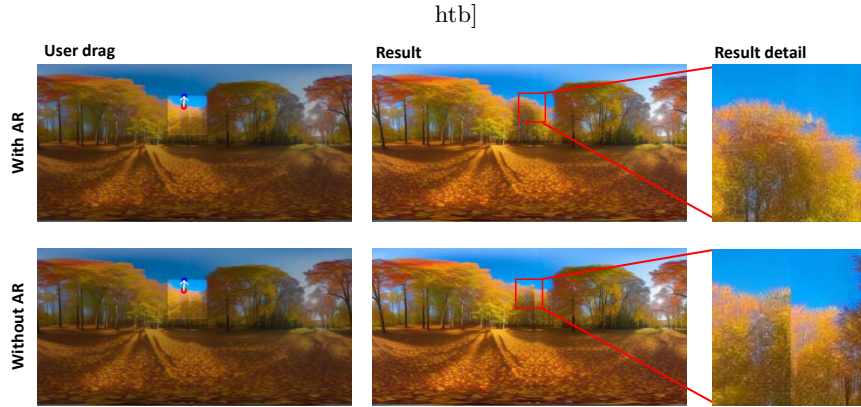


Fig.S-5: Ablation study on adaptive reprojection (AR). The top row shows the result with AR enabled, while the bottom row shows the result without it. The red boxes highlight the seam regions. With AR, the transitions across the seams are smooth and visually coherent. In contrast, without AR, noticeable artifacts and discontinuities appear. These results demonstrate the effectiveness of AR in enhancing geometric alignment and visual consistency in panoramic image editing.

## References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: Proc. ICML. pp. 214–223. PMLR (2017)
2. Brooks, T., Holynski, A., Efros, A.A.: Instructpix2pix: Learning to follow image editing instructions. In: Proc. CVPR. pp. 18392–18402 (2023)
3. Chen, T., Zhang, Y., Huo, X., Wu, S., Xu, Y., Wong, H.S.: Sphericgan: Semi-supervised hyper-spherical generative adversarial networks for fine-grained image synthesis. In: Proc. CVPR. pp. 10001–10010 (2022)
4. Cui, Y., Zhao, X., Zhang, G., Cao, S., Ma, K., Wang, L.: Stabledrag: Stable dragging for point-based image editing. In: Proc. ECCV. pp. 340–356. Springer (2024)
5. Gal, R., Patashnik, O., Maron, H., Bermano, A.H., Chechik, G., Cohen-Or, D.: Stylegan-nada: Clip-guided domain adaptation of image generators. ACM Trans. Graph. **41**(4), 1–13 (2022)
6. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial networks. Commun. ACM **63**(11), 139–144 (2020)
7. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Proc. NeurIPS (2017)
8. Ho, J., Jain, A., Abbeel, P.: Denoising diffusion probabilistic models. Proc. NeurIPS **33**, 6840–6851 (2020)
9. Jiang, R., Fu, X., Zheng, G., Li, T., Yao, T., Li, X.: Energy-guided optimization for personalized image editing with pretrained text-to-image diffusion models. arXiv preprint arXiv:2503.04215 (2025)

10. Kalischek, N., Oechsle, M., Manhardt, F., Henzler, P., Schindler, K., Tombari, F.: Cubediff: Repurposing diffusion-based image models for panorama generation. In: Proc. ICLR. p. to appear (2025)
11. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proc. CVPR. pp. 4401–4410 (2019)
12. Kavar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., Irani, M.: Imagic: Text-based real image editing with diffusion models. In: Proc. CVPR. pp. 6007–6017 (2023)
13. de La Garanderie, G.P., Abarghouei, A.A., Breckon, T.P.: Eliminating the blind spot: Adapting 3d object detection and monocular depth estimation to 360 panoramic imagery. In: Proc. ECCV. pp. 789–807 (2018)
14. Li, X., Wu, T., Qi, Z., Wang, G., Shan, Y., Li, X.: Sgat4pass: spherical geometry-aware transformer for panoramic semantic segmentation. Proc. of IJCAI (2023)
15. Lin, C.H., Chang, C.C., Chen, Y.S., Juan, D.C., Wei, W., Chen, H.T.: Coco-gan: Generation by parts via conditional coordinating. In: Proc. CVPR. pp. 4512–4521 (2019)
16. Liu, H., Xu, C., Yang, Y., Zeng, L., He, S.: Drag your noise: Interactive point-based editing via diffusion semantic propagation. In: Proc. CVPR. pp. 6743–6752 (2024)
17. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: Proc. ICML. pp. 3481–3490. PMLR (2018)
18. Mou, C., Wang, X., Xie, L., Wu, Y., Zhang, J., Qi, Z., Shan, Y.: T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. In: Proc. AAAI. pp. 4296–4304 (2024)
19. Nash, C., Menick, J., Dieleman, S., Battaglia, P.W.: Generating images with sparse representations. arXiv preprint arXiv:2103.03841 (2021)
20. Pan, X., Tewari, A., Leimkühler, T., Liu, L., Meka, A., Theobalt, C.: Drag your gan: Interactive point-based manipulation on the generative image manifold. In: ACM SIGGRAPH Conf. Proc. pp. 1–11 (2023)
21. Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., Lischinski, D.: Styleclip: Text-driven manipulation of stylegan imagery. In: Proc. CVPR. pp. 2085–2094 (2021)
22. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., Chen, M.: Hierarchical text-conditional image generation with clip latents. arXiv preprint arXiv:2204.06125 1(2), 3 (2022)
23. Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E.L., Ghasemipour, K., Gontijo Lopes, R., Karagol Ayan, B., Salimans, T., et al.: Photorealistic text-to-image diffusion models with deep language understanding. Proc. NeurIPS **35**, 36479–36494 (2022)
24. Shi, Y., Xue, C., Liew, J.H., Pan, J., Yan, H., Zhang, W., Tan, V.Y., Bai, S.: Dragdiffusion: Harnessing diffusion models for interactive point-based image editing. In: Proc. CVPR. pp. 8839–8849 (2024)
25. Wu, S., Tang, H., Jing, X.Y., Zhao, H., Qian, J., Sebe, N., Yan, Y.: Cross-view panorama image synthesis. IEEE Trans. Multimedia **25**, 3546–3559 (2022)
26. Wu, T., Li, X., Qi, Z., Hu, D., Wang, X., Shan, Y., Li, X.: Spherediffusion: Spherical geometry-aware distortion resilient diffusion model. In: Proc. AAAI. pp. 6126–6134 (2024)
27. Zhang, L., Rao, A., Agrawala, M.: Adding conditional control to text-to-image diffusion models. In: Proc. ICCV. pp. 3836–3847 (2023)
28. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proc. CVPR. pp. 586–595 (2018)

29. Zheng, G., Li, S., Wang, H., Yao, T., Chen, Y., Ding, S., Li, X.: Entropy-driven sampling and training scheme for conditional diffusion generation. In: Proc. ECCV. pp. 754–769. Springer (2022)
30. Zheng, G., Zhou, X., Li, X., Qi, Z., Shan, Y., Li, X.: Layoutdiffusion: Controllable diffusion model for layout-to-image generation. In: Proc. CVPR. pp. 22490–22499 (2023)
31. Zhou, T., Tang, Y.: Twindiffusion: Enhancing coherence and efficiency in panoramic image generation with diffusion models. In: Proc. ECAI, pp. 386–393. IOS Press (2024)
32. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proc. ICCV. pp. 2223–2232 (2017)