VividDream: Generating 3D Scene with Ambient Dynamics

https://vivid-dream-4d.github.io

Figure 1. **Generating 3D Scene with Ambient Dynamics.** Given an input image or text prompt, our proposed method, VividDream, generates a large, explorable 4D scene with ambient scene motion.

Abstract

We introduce VividDream, a method for generating explorable 4D scenes with ambient dynamics from a single input image or text prompt. VividDream first expands an input image into a static 3D point cloud through iterative inpainting and geometry merging. An ensemble of animated videos is then generated using video diffusion models with quality refinement techniques and conditioned on renderings of the static 3D scene from the sampled camera trajectories. We then optimize a canonical 4D scene representation using an animated video ensemble, with per-video motion embeddings and visibility masks to mitigate inconsistencies. The resulting 4D scene enables free-view exploration of a 3D scene with plausible ambient scene dynamics. Experiments demonstrate that VividDream can provide human viewers with compelling 4D experiences generated based on diverse real images and text prompts.

1. Introduction

Recent advancements in text-to-image generation [9, 40, 46] have revolutionized the field of computer vision, producing highly realistic and contextually accurate images from textual descriptions. This progress has paved the way for extending generative models beyond static images to higher-dimensional outputs, including video generation, 3D object generation, and 3D scene generation.

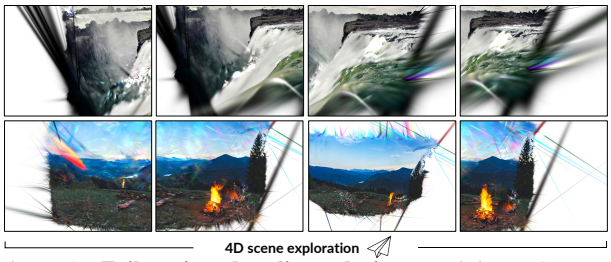

Figure 2. **Failure in a baseline solution.** Training a 4D scene representation (e.g., 4DGS [54]) using a single video generated by SVD [3] poses several challenges. First, the weak or absent camera motion in the animated video leads to many unseen areas and limited novel view synthesis. Second, existing video depth and pose estimation algorithms [22, 61] have difficulty in handling scenes with ambient motion due to their reliance on accurate correspondence estimation (e.g., optical flow [50]). These limitations motivate our multi-video approach to 4D scene generation.

However, in the realm of 4D space, current research predominantly focuses on generating or reconstructing individual 4D objects [2, 25, 45, 58, 60, 62]. While these efforts have led to significant breakthroughs in capturing objects' dynamics over time, there remains a notable gap in the generation of comprehensive 4D scenes. A 4D scene encompasses not just the temporal evolution of the scene but also the spatial scale of the 3D environment, enabling immersive view exploration. Although the recent DreamScene4D [7] generates a 4D scene from an input video, it focuses on reconstructing and generating the dynamic foreground objects but not expanding the 3D scene from the input views for larger view exploration.

In this paper, we aim to generate 4D scenes, enabling users to explore the 3D scene with dynamic ambient motion, such as fluid motions, trees, and grass swaying in the wind. To achieve this, a straightforward baseline might involve using a powerful video generator to create a sequence and then performing 3D reconstruction [22, 61] on this video to form a 4D scene [23, 28, 30, 54]. However, this approach falls short in several ways. No matter how detailed, a single video is insufficient to capture the full complexity and navigability required for an explorable 4D scene (Fig. 2). Such a scene necessitates multiple perspectives and the ability to transition between different views and moments seamlessly.

To address these limitations, we propose a novel pipeline for generating explorable 4D scenes from a single image or text prompt. In Stage 1, we expand the initial 3D point cloud obtained from the input through an iterative view extrapolation and inpainting process. Stage 2 focuses on generating ambient scene motion by animating multiple view-extrapolation videos rendered from the 3D scene. Finally, in Stage 3, we train a 4D scene representation using the animated multi-view videos while handling inconsistencies through visibility masking and per-video motion embeddings.

Our key contributions include:

- A holistic approach to generating explorable 4D scenes with ambient dynamics from a single image or text prompt.
- A multi-video animation framework to generate scene motion while respecting specified camera trajectories.
- Techniques to mitigate inconsistencies when generating a 4D scene from multiple independently animated videos.

We demonstrate the effectiveness of our approach in a variety of scenes. Experimental results showcase our method's ability to generate compelling 4D scene experiences with plausible ambient dynamics. We will release source code to facilitate future research on this problem.

2. Related Work

Text-to-3D generation Generative models have achieved promising results in generating realistic 3D objects [6, 29, 38, 42, 49, 53] and 3D scenes.By leveraging the power of 2D diffusion models to generate high-quality 2D images, certain approaches [13, 19, 57] address text-to-3D scene generation as an inpainting problem. These methods take text prompts as input and utilize 2D diffusion models [1, 31, 46] to generate images and inpaint the parts not seen in previous images. Another trend seeks higher controllability over the generated content by pre-defining the allocation of objects within the scene [8, 59, 63]. However, these methods generate static 3D scenes, lacking dynamic ambient scene motion, essential for better immersion. VividDream aims to generate dynamic scenes with ambient dynamics to provide a more immersive and engaging experience.

Video diffusion models Recent advancements in video diffusion models have demonstrated significant progress in generating high-quality video content [3, 4, 15, 33, 55]. Text-guided video diffusion models [4, 15, 18, 48, 51] extend the capabilities of the powerful text-to-image generators [46, 47] to the domain of video generation. To enhance the context guidance, some methods condition the video generation with images [3, 5, 16, 55]. However, the text and/or image conditions can still be insufficient to fully control the generated motions and separate camera motion from scene motion. To improve the controllability of motions, MotionCtrl [52] and CameraCtrl [17] fine-tune pre-trained video diffusion models [3, 16] with additional conditioning modules to take the user-specified camera or scene motion guidance. On the other hand, Time-Reversal [12] leverages Stable Video Diffusion (SVD) [3] in a zero-shot manner to guide video generation with an end-view condition and can achieve view interpolation and video looping. We build on the camera controllability introduced by Time-Reversal and

focus on generating ambient scene motion with conditioned camera poses.

Single-image animation. Cinemagraph generation [10, 20, 34] animates a single image into a video with looping motions. Text2-Cinemagraph [35] enables text-guided animation of the generated images. Li *et al*. [27] predicts the ambient scene motion and provides the interactivity on the generated dynamics. Li *et al*. [26] lifts the 2D cinemagraph into 3D space to allow small 3D viewpoint changes. Nevertheless, these methods are still limited in the input image view without expanding the scene. In contrast, we expand a 4D scene from an input image to allow exploration with significant viewpoint change.

3. Preliminary

3.1. SVD for image-to-video generation

SVD [3] is a publicly available state-of-the-art video generator that builds upon Stable Diffusion [46], producing 14 or 25 frames at a resolution of 1024× 768. It takes in an image and generates a video sequence by a latent-based diffusion model, where a 3D-UNet, Φ, denoises a video latent $\mathbf{z}_{T_d} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ into a clean $\mathbf{z}_0$ over T_d steps. At each denoising step t_d, the 3D-UNet Φ denoises the video latent $\mathbf{z}_{t_d}$ by $\mathbf{z}_{t_d-1} = \Phi(\mathbf{z}_{t_d} \oplus \mathbf{z}^{\text{img}}, t_d, \mathbf{c}^{\text{img}}, \mathbf{c}^{\text{scalar}})$, where $\oplus$ denotes the concatenating operation, $\mathbf{z}^{\text{img}}$ is the input image condition encoded by VAE. Additionally, $\mathbf{c}^{\text{img}}$ and $\mathbf{c}^{\text{scalar}}$ are CLIP [43]-encoded image condition and motion conditioning scalars for the 3D-UNet attention modules. After the iterative denoising steps, the clean video latent $\mathbf{z}_0$ is decoded into an RGB video by the VAE decoder. Although SVD can generate videos of promising quality, it lacks controllability over both scene and camera motion, restricting its use for scenarios such as view interpolation.

3.2. Time-Reversal for video view interpolation

To control the camera motion generation, Time-Reversal [12] leverages a crucial property of SVD: the input conditioning image serves as the start frame of the generated video. Therefore, at each denoising step t_d, besides a denoising pass on the video latent $\mathbf{z}_{t_d}$ to get deonised $\mathbf{z}^{\text{fwd}}_{t_d-1}$ with start-view image condition $\mathcal{I}^{\text{start}}$, Time-Reversal temporally reverses $\mathbf{z}_{t_d}$ as $\overline{\mathbf{z}}_{t_d}$ and denoises with another end-view image condition, $\mathcal{I}^{\text{end}}$, to obtain the denoised $\overline{\mathbf{z}}^{\text{end}}_{t_d-1}$. Subsequently, the two denoised $\mathbf{z}^{\text{start}}_{t_d-1}$ and $\overline{\mathbf{z}}^{\text{end}}_{t_d-1}$ are fused into $\mathbf{z}_{t_d-z}$ for the next denoising step $t_d - 1$: $\mathbf{z}_{t_d-1} = \gamma \mathbf{z}^{\text{start}}_{t_d-1} + (1-\gamma)\overline{(\overline{\mathbf{z}}^{\text{end}}_{t_d-1})}$, where γ is the fusing weight and $\overline{(\overline{\mathbf{z}}^{\text{end}}_{t_d-1})}$ is the reverse of $\overline{\mathbf{z}}^{\text{end}}_{t_d-1}$, back to the original video time space.

By manipulating additional end-view image condition, $\mathcal{I}^{\text{end}}$, Time-Reversal can perform video looping when the end view $\mathcal{I}^{\text{end}}$ is identical to the start view $\mathcal{I}^{\text{start}}$, and view interpolation between two different views of $\mathcal{I}^{\text{end}}$ and $\mathcal{I}^{\text{start}}$. Nevertheless, a single view-interpolation video generation is insufficient for expanding an explorable 4D scene, especially SVD-generated video with 25 frames. Therefore, we incorporate multiple passes of Time-Reversal to acquire the scene motions in each part of a generated 3D scene (Sec 4.2).

3.3. 4D Gaussian Splatting for 4D-scene view synthesis

3D Gaussian Splatting (3DGS) [21] recently has shown decent rendering quality and fast training and rendering speeds with an explicit representation, compared to NeRF [39] with neural implicit representation. The geometry of each Gaussian splat (GS) is represented by position $\mu \in \mathbb{R}^3$ and an anisotropic covariance matrix Σ, where the covariance is decomposed by $\Sigma = \mathbf{RSS}^T\mathbf{R}^T$ with the rotation $\mathbf{R} \in \mathbb{R}^4$ and scaling $\mathbf{S} = \text{Diag(s)}, s \in \mathbb{R}^3$. On the other hand, the appearance of a GS is parameterized by the color, c, encoded by spherical harmonics coefficients, and an opacity, α. To render the color $\mathcal{C}$ of a given pixel at a view is rendered by depth-based sorting on the GS set $\{G_i\}_{i=1}^N$ and α-blending with 2D-projected opacity:

$$\mathcal{C} = \sum_{i=1}^{N} c_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j). \tag{1}$$

To extend the 3DGS for dynamic scenes, 4DGS [54] (4DGS) introduces a deformation module to deform each GS for dynamic motion. Given the 3D position $\mu = (x, y, z)$ and the query time t, the deformation module obtains the feature f using a spatiotemporal Hexplane encoder [14], f. Then, the feature $\mathcal{H}(x, y, z, t)$ is fed into decoding MLPs Θ to acquire the deformation position $(\Delta x, \Delta y, \Delta z)$, rotation Δr, and scaling Δs for the time t. With all GS $\{G_i\}_{i=1}^N$ deformed, the view of a dynamic scene is rendered by Eq. 1.

However, 4DGS is sensitive to noisy training views due to the reliance on accurate camera poses and wide coverage of viewing directions. In our setting, which is based on multiple videos generated by SVD and Time-Reversal, the appearance and motion inconsistencies among videos can be harmful. To address this issue, we introduce per-video motion embedding and visibility mask to mitigate the blurriness caused by multi-video inconsistencies (Sec. 4.3).

4. Method

Our three-stage pipeline for generating an explorable 4D scene with ambient motion is illustrated in Fig.3. In the following subsections, we provide detailed information about the three stages.

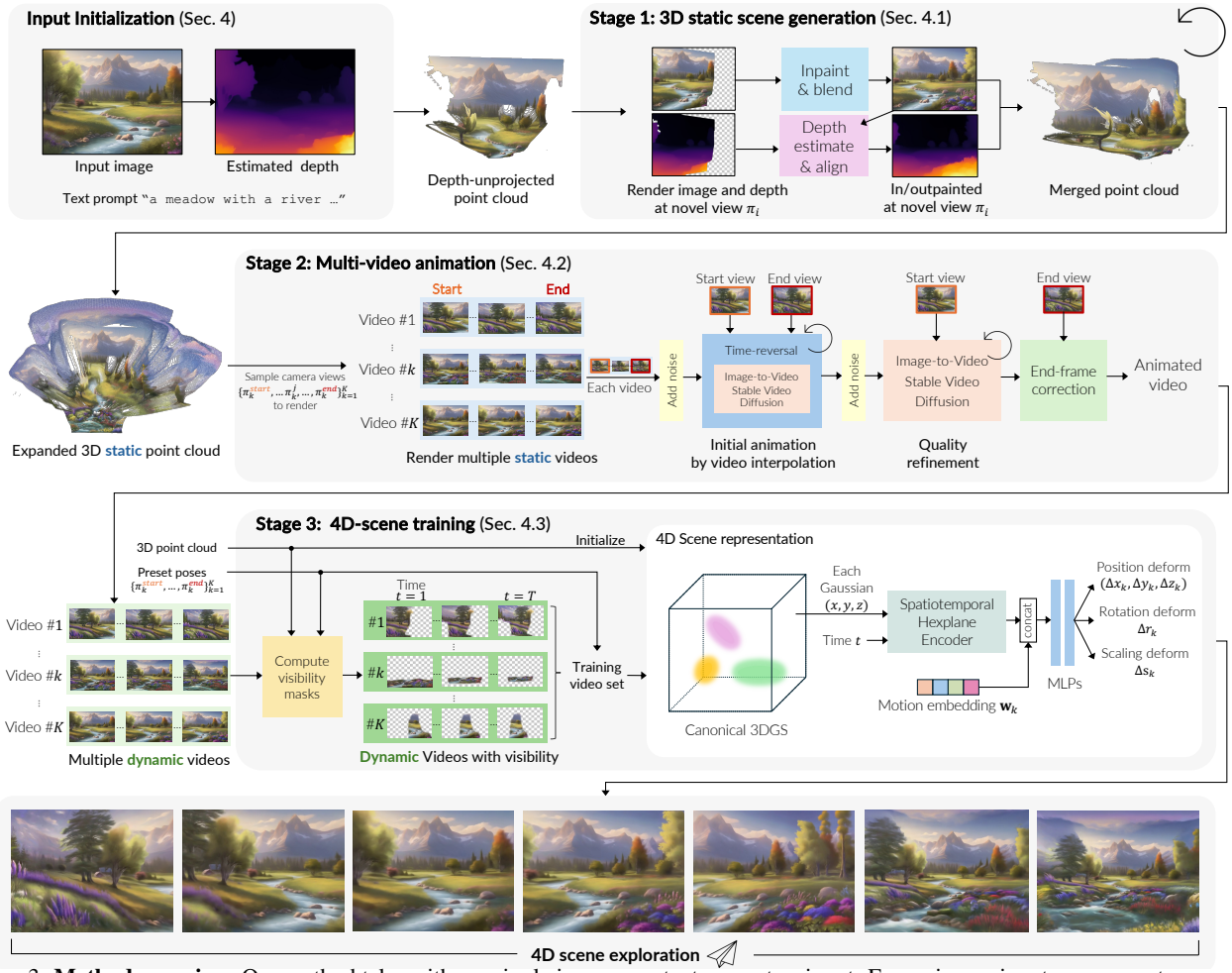

Figure 3. **Method overview.** Our method takes either a single image or a text prompt as input. For an image input, we generate a caption using [24] and estimate its depth using DepthAnything [56]. For a text prompt input, we generate the corresponding image using a text-to-image generator [46]. The image, text prompt, and estimated depth form the initialization for the subsequent stages. In Stage 1 (Sec. 4.1), we expand the initial 3D point cloud through an iterative process of novel view inpainting and point cloud merging using aligned depth estimates. Stage 2 (Sec. 4.2) focuses on generating the scene motions by rendering K static view-extrapolation videos covering the entire 3D scene. Each video is first animated using Time-Reversal [12] with static renderings as conditions. To improve the video quality, we refine the animated videos using SVD [3]. However, this refinement may cause the camera motion to deviate from the desired trajectory. We mitigate this issue by applying a smooth transition on the last few frames using FILM [44] to match the conditioning end view. Finally, in Stage 3 (Sec. 4.3), we train the 4D scene model, 4DGS [54], using the animated videos from Stage 2. To handle appearance and motion inconsistencies among the multi-view videos, we apply visibility masks with soft blending weights and introduce a per-video motion embedding inspired by NeRF-W [36]. The resulting 4D scene model enables consistent motion and immersive 4D scene exploration.

4.1. 3D scene generation

Given a single-view point cloud $\mathcal{P}_0$ unprojected by the input image and the estimated depth, we expand the 3D point cloud through an iterative process of view extrapolations, consisting of novel-view inpainting and point-cloud merging. At each iteration i, we select a novel-view pose, π_i, to render the image $\mathcal{I}_i$ and depth $\mathcal{D}_i$ from the point cloud $\mathcal{P}_{i-1}$, with a mask M_i indicating the known regions. The unseen regions in the rendered image are then filled by the inpainting model [46] conditioned on image $\mathcal{I}_i$, inpainting mask $(1 - M_i)$, and text p. We observe that the VAE encoding and decoding process produces color inconsistencies between the inpainted image $\mathcal{I}_i^{\text{inpaint}}$ and the known regions in $\mathcal{I}_i$. To tackle this, we apply Poisson blending [41] on $\mathcal{I}_i^{\text{inpaint}}$ with $\mathcal{I}_i$ along the mask borders to enhance color consistency (Fig. 4a).

To unproject the inpainted image at novel view π_i, the depth $\mathcal{D}_i^{\text{inpaint}}$ is estimated again by [56]. Since the inconsistency between the new depth $\mathcal{D}_i^{\text{inpaint}}$ and the known regions of existing depth $\mathcal{D}_i$ creates misalignment in 3D space (Fig. 4b), we follow SceneScape [13] and align the depths by fine-tuning the last layer of the depth estimation

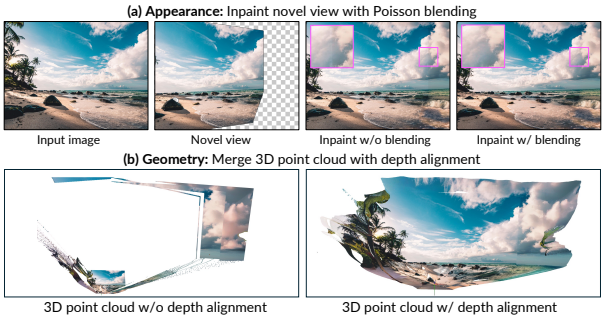

Figure 4. **Handling appearance and geometry inconsistencies in 3D point cloud.** (a) The inpainting model [46] may introduce color shifts caused by VAE, creating inconsistencies with the existing scene (zoomed-in). We mitigate these boundary seams using Poisson blending [41]. (b) After merging the inpainted region with the existing point cloud, the newly estimated depth may be inconsistent with the existing geometry (left). To address this, we fine-tune the depth estimation model following SceneScape [13] to align the new depth with the known view depth. These steps ensure appearance and geometry consistency in each iteration of 3D point cloud generation.

model [56] and minimizing the disparity between the two depths, $\mathcal{D}_i^{\text{inpaint}}$ and $\mathcal{D}_i$, in the known region M_i:

$$\mathcal{L}_{\text{depth}}^{\text{inpaint}} = \sum \left| \frac{\mathcal{D}_i^{\text{inpaint}\,-1} - \mathcal{D}_i{}^{-1}}{\mathcal{D}_i^{\text{inpaint}\,-1} + \mathcal{D}_i{}^{-1}} \right|_2^2 \odot M_i, \qquad (2)$$

where $\odot$ is the element-wise multiplication. We then use the aligned depth $\mathcal{D}_i^{*\text{inpaint}}$ to lift the inpainted region $(1 - M_i)$ of image $\mathcal{I}_i^{\text{inpaint}}$ into 3D to form $\mathcal{P}_i^{\text{inpaint}}$, and merge it with the existing one $\mathcal{P}_{i-1}$ into $\mathcal{P}_i = \mathcal{P}_{i-1} \cup \mathcal{P}_i^{\text{inpaint}}$. By iterating this process, an expanded 3D static point cloud $\mathcal{P}$ is used for view exploration.

4.2. Multi-video animation

In Stage 2, we utilize the image-to-video generator [3] to generate ambient motion using the static 3D scene $\mathcal{P}$ from Sec. 4.2. To bake the motion into a 4D scene representation for view exploration, controlling the camera motions of the generated videos is essential for the 4D scene optimization in Stage 3 (Sec. 4.3). Therefore, instead of a single-image condition in the original SVD, we condition the video generation process with a static-scene video rendered by a preset camera trajectory from the point cloud $\mathcal{P}$. Since a single video cannot encompass the entire 3D scene, we acquire a set of videos $\{\mathcal{V}_k^{\text{static}}\}_{k=1}^K$ to ensure complete view coverage by rendering K videos with camera trajectories $\{\pi_k^{\text{start}}, ..., \pi_k^{\text{end}}\}_{k=1}^K$, where each video $\mathcal{V}_k^{\text{static}} = \{\mathcal{I}_k^{\text{start}}, ..., \mathcal{I}_k^{\text{end}}\}$ captures a part of the scene. We omit k here for simplicity. In practice, we take the **view-extrapolation paths** from Stage 1 since the extrapolation paths for 3D scene generation will cover the entire scene. For each path, we interpolate between the known view π_{i-1} and the novel view π_i to form a camera trajectory of length T, where T is the temporal resolution of the SVD model. This static-scene extrapolation video serves as the condition of the video generation (Fig. 5a) to obtain the ambient scene motion while retaining its camera trajectory.

However, the original SVD cannot control camera motion in the generated videos (Fig. 5b). Although some methods [17, 52] extend the pretrained video diffusion models by inserting additional motion conditioning modules, the generated videos may still not follow the camera condition [52] or contain insufficient ambient scene motion [17]. Alternatively, Time-Reversal [12] builds upon SVD with two end-view image conditions, $\mathcal{I}^{\text{start}}, \mathcal{I}^{\text{end}}$, to allow better control over camera motion and scene motion for video looping and view interpolation (Sec. 3.2). However, the two end-view conditions do not guarantee the camera motion in the middle frames, so it may still deviate from the specified trajectory (Fig. 5c).

To improve camera control, we follow SDEdit [37] and generate the ambient motion conditioned on the static-scene rendering $\mathcal{V}^{\text{static}} = \{\mathcal{I}^{\text{start}}, ..., \mathcal{I}^{\text{end}}\}$. Concretely, we encode the static-scene video into the latent $\mathbf{z}_0^{\text{static}}$ and then add noise to a diffusion step τ_{tr} in the diffusion noise schedule as a perturbed latent $\mathbf{z}_{\tau_{\text{tr}}}^{\text{static}}$. Therefore, the denoising process of Time-Reversal is performed on the perturbed $\mathbf{z}_{\tau_{\text{tr}}}^{\text{static}}$ from the step τ_{tr} to get a clean video latent $\mathbf{z}_0^{\text{dyn}}$ with dynamic motion. The VAE-decoded RGB video $\mathcal{V}^{\text{dyn}} = \{\mathcal{I}_1, ..., \mathcal{I}_T\}$ then could have scene motion at the temporal resolution T and better camera motion consistency with the specified trajectory (Fig. 5d).

Nonetheless, we found that Time-Reversal still yields low visual quality due to the noise-averaging steps. To restore visual quality, we apply a second round of SDEdit to the initial animated video from the denoising step τ_{refine} with SVD only. Although the details can be enhanced, the appearance and camera motion in the later frames may slightly deviate from the specified static-rendering video without the end-view condition from Time Reversal (Fig. 5e). To address this, we adopt FILM [44], to replace the last n frames of the video, $\{\mathcal{I}_{T-n+1}, ..., \mathcal{I}_T\}$, with $\{\mathcal{I}_{T-n+1}^{\text{FILM}}, ..., \mathcal{I}_{T-1}^{\text{FILM}}, \mathcal{I}^{\text{end}}\}$, creating a smooth transition to the conditioned end-view $\mathcal{I}^{\text{end}}$, where $\{\mathcal{I}_{T-n+j}^{\text{FILM}}\}_{j=1}^{n-1}$ is the frame interpolation output with two fixed view $\mathcal{I}_{T-n}^{\text{FILM}}$ and $\mathcal{I}^{\text{end}}$. Consequently, we can obtain a set of animated videos, $\{\mathcal{V}_k^{\text{dyn}}\}_{k=1}^K$, with specified camera trajectories and desired scene motion.

4.3. 4D-scene training

With the expanded 3D point cloud $\mathcal{P}$ and multiple animated videos $\{\mathcal{V}_k^{dyn}\}_{k=1}^K$ covering the 3D scene, we now fit the 4D scene. The canonical 3DGS is initialized by the point cloud $\mathcal{P}$. For each video $\mathcal{V}_k^{\text{dyn}}$, we assign timestamps $t = \{1, 2, ..., T\}$ for the T frames.

Figure 5. **Motion generation with controllable camera trajectory.** We aim to animate ambient scene dynamics while maintaining control over the camera trajectory. (a) We render a static scene video using the 3D point cloud with a smooth camera motion, serving as a condition for the animated videos. (b) Naively applying SVD with only the start view results in uncontrollable camera poses. (c) Time-Reversal ensures start and end view consistency but suffers from blurriness and camera trajectory deviations. (d) Using the static scene video as a condition for SVD with Time-Reversal and SDEdit [37] encourages following the desired trajectory but yields low-quality results. (e) Applying SDEdit again with only the start view improves quality but causes camera pose deviations. (f) We correct this by applying a smooth transition to the last frames using FILM [44] to match the end view. Our approach generates animated videos with ambient dynamics while respecting the specified camera trajectory.

The 4DGS optimization is sensitive to the quality of training data. In contrast to the original 4DGS [54] trained on single or multiple synchronized videos, the multi-view videos, in our case, contain significant inconsistencies in appearance and motion by separate passes of video generations. The inconsistencies could harm the optimization of 4DGS, leading to noticeable blurriness and floaters. To address this, we introduce **per-video motion embeddings** and **visibility masks** to the training in both implicit and explicit approaches to handle the multi-video inconsistencies.

Inspired by NeRF-W [36] for handling inconsistent samples in an embedding space, we employ a *per-video* motion embedding, $\mathbf{w}_k{}_{k=1}^{K} \in \mathbb{R}^W$, instead of per-image embedding, to ensure consistent motion within each video while managing inconsistencies across videos. Specifically, given a 3D position $\mu = (x, y, z)$ for each Gaussian splat (GS) and the query time t, the deformation module first obtains the spatiotemporal feature, $\mathcal{H}(x, y, z, t)$, using a Hexplane encoder $\mathcal{H}$ (as detailed in Sec. 3.3). We concatenate this feature with the motion embedding $\mathbf{w}_k$ and feed

Figure 6. **Mitigating inconsistencies among multiple training videos.** Training a 4D scene with independently animated videos inevitably leads to low-quality renderings. We introduce a per-video motion embedding into the deformation module to handle the inconsistency in the embedding space. The motion embedding results in a significant improvement in the quality (*e.g.* the fluid motion). Moreover, we apply visibility masks to the videos, ensuring that only one video observes each scene part, thereby mitigating the blurriness caused by multi-video inconsistencies.

into MLP decoder Θ to obtain video-dependent deformation terms for training: $\{\Delta x_k, \Delta y_k, \Delta z_k, \Delta r_k, \Delta s_k\} = \Theta(\mathcal{H}(x, y, z, t) \oplus \mathbf{w}_k)$, where $\oplus$ denotes concatenation. After training, all embeddings are averaged into a global motion embedding to render global consistent motion. To regularize the embeddings, we constrain the embedding norm in a 1-norm space [11]. In practice, we set the embedding dimension $W = 16$.

Implicitly handling inconsistencies in the motion embedding space may still be insufficient, so we further apply an explicit visibility mask $\{\mathcal{M}_k\}_{k=1}^{K}$ to each video. Concretely, the visibility masks ensure that each part of the scene is observed by only one video (as shown in Fig. 3). We jointly consider camera poses and the 3D static point cloud to determine the visibility masks. The length of a viewing ray, $\|\mathbf{v}_k^j\|$, can be obtained by the depth of the 3D point cloud. The view-angle score $\vec{\mathbf{v}}_k^j$ of the viewing ray $\mathbf{v}_k^j$ w.r.t to the camera-looking direction, $\mathbf{l}_k^j$, is computed by $\mathbf{v}_k^j \cdot \mathbf{l}_k^j / \|\mathbf{v}_k^j\| \|\mathbf{l}_k^j\|$. The overall viewing length $\|\mathbf{v}_k\|$ and view-angle score $\vec{\mathbf{v}}_k$ of a video $\mathcal{V}_k$ is the average over all the frames.

We then use the two metrics to find the best video that views the scene part with close and forward-facing views: $k^* = \arg\max_k \frac{1}{\|\mathbf{v}_k\|} + \beta \vec{\mathbf{v}}_k$, where $\beta (= 1)$ is a scalar to balance the two terms. The visibility masks for all videos are then acquired by the k^* for each scene part. The hard selection ensures each part is only trained by one video to avoid blurriness, but we also expand the visibility mask by a small width with a soft decreasing weight to encourage a smooth spatial transition among the training videos. Note that the visibility masking may result in separate motions because different videos train different areas. While we focus on generating a scene with ambient motion, the separate motion may be negligible. Then, the visibility masks are used as weighting maps to compute the RGB and depth loss during training: $\mathcal{L}_{\text{rgb}} = \sum \|\tilde{\mathcal{I}}_k - \mathcal{I}_k\|_1 \odot \mathcal{M}_k$, and $\mathcal{L}_{\text{depth}} = \sum |\tilde{\mathcal{D}}_k^{-1} - \mathcal{D}_k^{-1}|_1 \odot \mathcal{M}_k$, where the depth loss is slightly different from Eq. 2. Here, we apply more penalty to near scenes and more tolerance to the error in far scenes since the point cloud depth alignment may still exist misalignment in the far scenes. Notably, the 4D scene depth is trained by the depth of the 3D static point cloud. It may not be suitable when the scene contains outstanding moving objects, such as humans and animals. Since we are generating a 4D scene with ambient motions, we found it sufficient to supervise learning 3D scene structure. During 4DGS fitting, to prevent quality degradation from training with the VAE-decoded videos via SVD, we also use the point-cloud-rendered images in higher quality to supervise the canonical 3DGS model. We adopt the rigidity loss [32] $\mathcal{L}_{\text{rigidity}}$ to encourage neighboring Gaussian splats to have a similar deformation for propagating the high-quality canonical appearance to 4D rendering. The total is then computed by $\mathcal{L} = \mathcal{L}_{\text{rgb}} + \lambda_{\text{depth}}\mathcal{L}_{\text{depth}} + \lambda_{\text{rigidity}}\mathcal{L}_{\text{rigidity}}$, where λ_{depth} and $\lambda_{\text{rigidity}}$ balance the loss terms. Finally, the 4D scene optimization learns the motion from multi-video with proper handling of multi-video inconsistencies, which results in a 4D explorable scene with ambient motion and sharp details (Fig. 6).

5. Experimental Results

Implementation details. In Stage 1, for 3D scene generation, we typically expand the scene ten times, taking about 15 minutes. In Stage 2, for multi-video generation, we create 10 extrapolation videos ($K = 10$). The animation process, including quality refinement and end-frame correction, takes roughly 7 minutes per video with $\tau_{\text{tr}} = 16$ for Time-Reversal and $\tau_{\text{refine}} = 9$ for SVD refinement. We re-implement Time-Reversal [12] by ourselves. In Stage 3, the canonical 3DGS is trained for 3,000 iterations with a maximum spherical harmonics order of 3. Then, the 4DGS is further trained for 15,000 iterations, with $\lambda_{\text{depth}} = 1$ and $\lambda_{\text{rigidity}} = 1$, taking around 1.5 hours. Overall, the 4D scene generation process takes about 3.5 hours. Notably, the baseline method, which involves reconstructing a single SVD video using depth and pose estimation[61] and then training with 4DGS, requires more than 4 hours.

5.1. Qualitative results

As 4D scene generation is a new problem without an existing benchmark, we showcase a free-view exploration of the generated 4D scenes. In Fig. 7, our method expands the real input image into a larger scene with ambient motions. In Fig. 8, we demonstrate diverse text-guided 4D scene generation scenarios with various scene ambient motion. Furthermore, our method can also turn a 360 panorama image into a 4D immersive scene (Fig. 9), which can be further applied to VR applications.

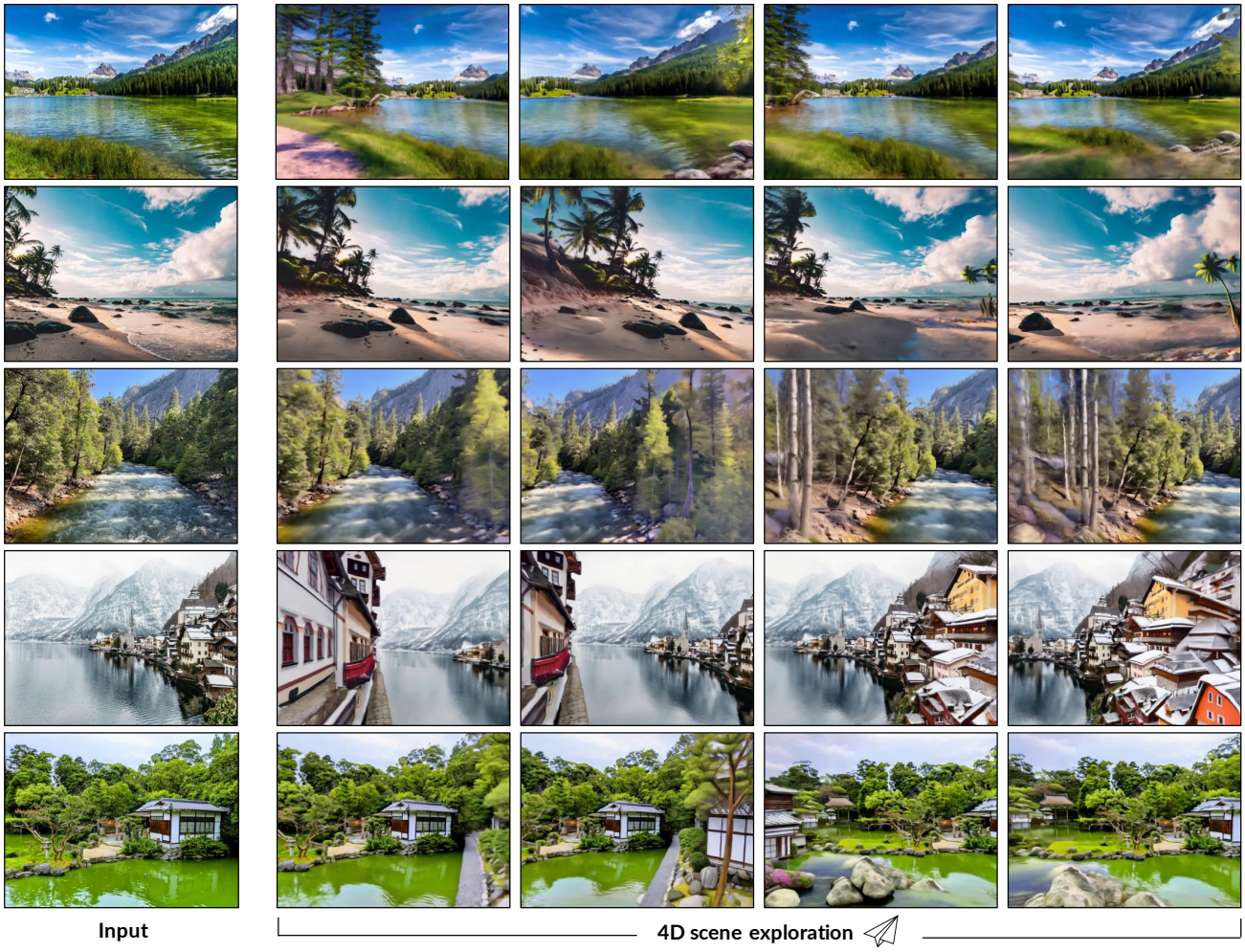

Figure 7. **Qualitative results of real photo inputs. Our method turns a real static image into an alive 4D scene with ambient scene motions.**

5.2. Human perceptual evaluation

The primary application of our work is for creative and entertainment purposes. It's difficult to evaluate such generated results with standard reference-based metrics. We thus employ human perceptual evaluation on the generated 4D-scene videos, focusing on detail quality, visual appeal, and view explorability. We compare our method against a baseline and conduct an ablation study on each component. Participants answered binary choice questions like "Which video has better overall visual appeal?" and we collected feedback from 32 users. As shown in Table 1, our method surpasses the baseline in visual appeal and view explorability. Our lower preference ratio for detail quality is due to inconsistencies from multi-video training, while baseline is trained with single-video which captures more consistent details yet with limited viewpoints. For the ablation study, we compare against our ablated methods without SDEdit quality refinement in the video generation stage, per-video motion embedding, and visibility masking in the training stage to handle the multi-video inconsistencies. The results indicate a general preference for our full method over the ablated versions. The improvement from visibility masking was less significant, likely because detail improvements were not easily noticeable in videos with rapid view changes. However, quality improvements are evident in the image comparison shown in Fig. 6.

Table 1. Human perceptual user study for comparison with baseline and ablation study on details, visual appeal, and explorability.

Questions.	Detail.	Appeal.	Explore.
Ours over baseline.	61.5%	82.3%	91.7%
Ours over w/o SDEdit quality refine.	88.2%	80.6%	-
Ours over w/o motion embed.	59.8%	65.9%	-
Ours over w/o mask.	50.5%	52.2%	-

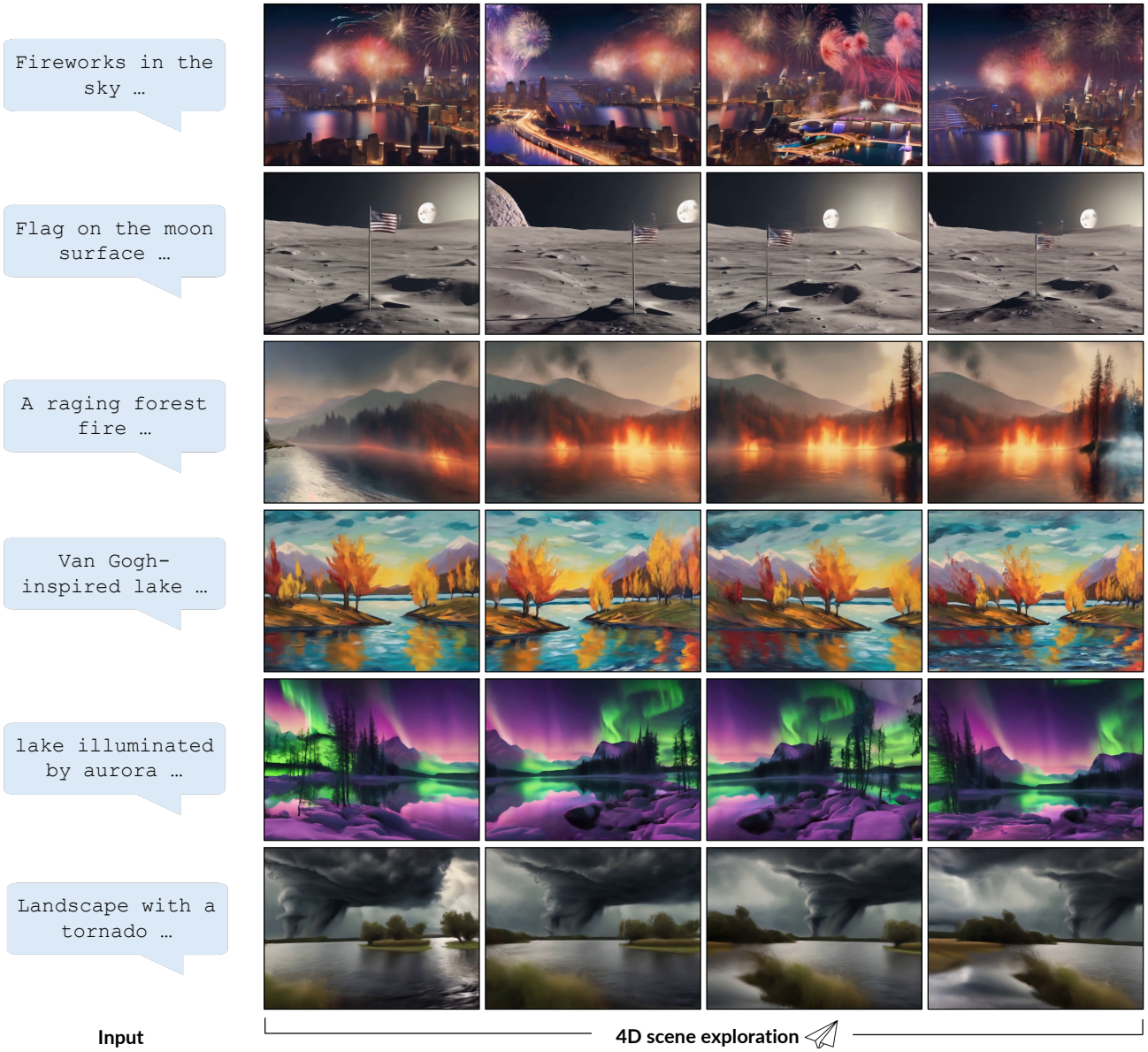

Figure 8. **Qualitative results of text inputs.** Our method can generate diverse ambient scene motion in various text-guided generated scenarios.

Figure 9. **Generating immersive 4D scene from 360 panorama image.** Our method can also take a 360 panorama image to generate an explorable 4D scene with ambient motion.

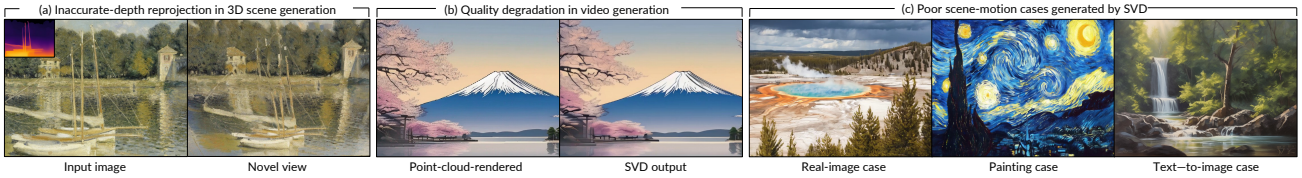

Figure 10. **Limitations.** Our method relies on a series of successes in 3D scene generation and video generation. (a) In 3D scene generation, imperfect depth estimation/reprojection often occurs in thin structures, leading to noisy results. (b) In video generation, the quality may degrade through the encoding and decoding process of SVD VAE, causing a blurry 4D scene. (c) The scene motion generation is difficult to control. SVD often fails to generate plausible motion for paintings and text-conditioned generated images. We expect more advanced video generation models may achieve better quality and controllability, but most of these are not available to the public.

6. Limitation and Discussion

We acknowledge that the quality of our method is far from perfect. Our method is complex and time-consuming with multiple stages, and relies on a series of successful processes to generate a 4D scene with plausible ambient motion. In the 3D scene generation stage, inaccurate depth estimation can be revealed in a distant novel view with a large viewpoint change, restricting the 3D scene expansion. In addition, the depth estimation model often fails to estimate thin structures, leading to distorted results in the novel view (Fig. 10a). Furthermore, the noisy reprojection will also lead to undesirable video generation in Stage 2. In the multi-video generation stage, SVD significantly degrades the input image quality (Fig. 10b), causing a blurry 4D scene reconstruction in Stage 3. Besides, SVD still lacks scene motion control and often fails in non-real image cases, such as paintings, and diffusion-generated images (Fig. 10c). As mentioned by Time-Reversal [12], the SVD only takes a few motion scalars as animation conditions, which are ambiguous and often require careful tuning on the new input. Despite these major limitations, our method is the first to enable the generation of explorable 4D scenes with ambient motions. When applied to a more reliable video generator in the future, we believe that our proposed pipeline can significantly enhance 4D scene generation, leading to improved quality and robustness.

References

[1] Hadi Alzayer, Zhihao Xia, Xuaner Zhang, Eli Shechtman, Jia-Bin Huang, and Michael Gharbi. Magic fixup: Streamlining photo editing by watching dynamic videos. 2024. 2

[2] Sherwin Bahmani, Ivan Skorokhodov, Victor Rong, Gordon Wetzstein, Leonidas Guibas, Peter Wonka, Sergey Tulyakov, Jeong Joon Park, Andrea Tagliasacchi, and David B Lindell. 4d-fy: Text-to-4d generation using hybrid score distillation sampling. *arXiv preprint arXiv:2311.17984*, 2023. 2

[3] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 2, 3, 4, 5

[4] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023. 2

[5] Haoxin Chen, Menghan Xia, Yingqing He, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Jinbo Xing, Yaofang Liu, Qifeng Chen, Xintao Wang, et al. Videocrafter1: Open diffusion models for high-quality video generation. *arXiv preprint arXiv:2310.19512*, 2023. 2

[6] Rui Chen, Yongwei Chen, Ningxin Jiao, and Kui Jia. Fantasia3d: Disentangling geometry and appearance for high-quality text-to-3d content creation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22246–22256, 2023. 2

[7] Wen-Hsuan Chu, Lei Ke, and Katerina Fragkiadaki. Dreamscene4d: Dynamic multi-object scene generation from monocular videos. *arXiv preprint arXiv:2405.02280*, 2024. 2

[8] Dana Cohen-Bar, Elad Richardson, Gal Metzer, Raja Giryes, and Daniel Cohen-Or. Set-the-scene: Global-local training for generating controllable nerf scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2920–2929, 2023. 2

[9] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021. 1

[10] Siming Fan, Jingtan Piao, Chen Qian, Hongsheng Li, and Kwan-Yee Lin. Simulating fluids in real-world still images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15922–15931, 2023. 3

[11] Brandon Yushan Feng, Susmija Jabbireddy, and Amitabh Varshney. Viinter: View interpolation with implicit neural representations of images. In *SIGGRAPH Asia 2022 Conference Papers*, 2022. 7

[12] Haiwen Feng, Zheng Ding, Zhihao Xia, Simon Niklaus, Victoria Abrevaya, Michael J Black, and Xuaner Zhang. Explorative inbetweening of time and space. *arXiv preprint arXiv:2403.14611*, 2024. 2, 3, 4, 5, 7, 10

[13] Rafail Fridman, Amit Abecasis, Yoni Kasten, and Tali Dekel. Scenescape: Text-driven consistent scene generation. *Advances in Neural Information Processing Systems*, 36, 2024. 2, 4, 5

[14] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12479–12488, 2023. 3

[15] Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs, Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for video diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 22930–22941, 2023. 2

[16] Yuwei Guo, Ceyuan Yang, Anyi Rao, Yaohui Wang, Yu Qiao, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023. 2

[17] Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation, 2024. 2, 5

[18] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 2

[19] Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7909–7920, 2023. 2

[20] Aleksander Holynski, Brian L Curless, Steven M Seitz, and Richard Szeliski. Animating pictures with eulerian motion fields. In *CVPR*, 2021. 3

[21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4):1–14, 2023. 3

[22] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 2

[23] Yao-Chih Lee, Zhoutong Zhang, Kevin Blackburn-Matzen, Simon Niklaus, Jianming Zhang, Jia-Bin Huang, and Feng Liu. Fast view synthesis of casual videos. *arXiv preprint arXiv:2312.02135*, 2023. 2

[24] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023. 4

[25] Jiyang Li, Lechao Cheng, Zhangye Wang, Tingting Mu, and Jingxuan He. Loopgaussian: Creating 3d cinemagraph with multi-view images via eulerian motion field. *arXiv preprint arXiv:2404.08966*, 2024. 2

[26] Xingyi Li, Zhiguo Cao, Huiqiang Sun, Jianming Zhang, Ke Xian, and Guosheng Lin. 3d cinemagraphy from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4595–4605, 2023. 3

[27] Zhengqi Li, Richard Tucker, Noah Snavely, and Aleksander Holynski. Generative image dynamics. *arXiv preprint arXiv:2309.07906*, 2023. 3

[28] Zhengqi Li, Qianqian Wang, Forrester Cole, Richard Tucker, and Noah Snavely. Dynibar: Neural dynamic image-based rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4273–4284, 2023. 2

[29] Chen-Hsuan Lin, Jun Gao, Luming Tang, Towaki Takikawa, Xiaohui Zeng, Xun Huang, Karsten Kreis, Sanja Fidler, Ming-Yu Liu, and Tsung-Yi Lin. Magic3d: High-resolution text-to-3d content creation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 300–309, 2023. 2

[30] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13–23, 2023. 2

[31] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11461–11471, 2022. 2

[32] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *3DV*, 2024. 7

[33] Zhengxiong Luo, Dayou Chen, Yingya Zhang, Yan Huang, Liang Wang, Yujun Shen, Deli Zhao, Jingren Zhou, and Tieniu Tan. Videofusion: Decomposed diffusion models for high-quality video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10209–10218, 2023. 2

[34] Aniruddha Mahapatra and Kuldeep Kulkarni. Controllable animation of fluid elements in still images. In *CVPR*, 2022. 3

[35] Aniruddha Mahapatra, Aliaksandr Siarohin, Hsin-Ying Lee, Sergey Tulyakov, and Jun-Yan Zhu. Text-guided synthesis of eulerian cinemagraphs. *ACM Transactions on Graphics (TOG)*, 42(6):1–13, 2023. 3

[36] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 4, 6

[37] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. 5, 6

[38] Gal Metzer, Elad Richardson, Or Patashnik, Raja Giryes, and Daniel Cohen-Or. Latent-nerf for shape-guided generation of 3d shapes and textures. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12663–12673, 2023. 2

[39] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf:

Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 3

[40] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, pages 8162–8171. PMLR, 2021. 1

[41] Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. *ACM SIGGRAPH 2003 Papers*, 2003. 4, 5

[42] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 2

[43] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 3

[44] Fitsum Reda, Janne Kontkanen, Eric Tabellion, Deqing Sun, Caroline Pantofaru, and Brian Curless. Film: Frame interpolation for large motion. In *European Conference on Computer Vision (ECCV)*, 2022. 4, 5, 6

[45] Jiawei Ren, Liang Pan, Jiaxiang Tang, Chi Zhang, Ang Cao, Gang Zeng, and Ziwei Liu. Dreamgaussian4d: Generative 4d gaussian splatting. *arXiv preprint arXiv:2312.17142*, 2023. 2

[46] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1, 2, 3, 4, 5

[47] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022. 2

[48] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 2

[49] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *arXiv preprint arXiv:2309.16653*, 2023. 2

[50] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, 2020. 2

[51] Yaohui Wang, Xinyuan Chen, Xin Ma, Shangchen Zhou, Ziqi Huang, Yi Wang, Ceyuan Yang, Yinan He, Jiashuo Yu, Peiqing Yang, et al. Lavie: High-quality video generation with cascaded latent diffusion models. *arXiv preprint arXiv:2309.15103*, 2023. 2

[52] Zhouxia Wang, Ziyang Yuan, Xintao Wang, Tianshui Chen, Menghan Xia, Ping Luo, and Yin Shan. Motionctrl: A unified and flexible motion controller for video generation. 2023. 2, 5

[53] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[54] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Wang Xinggang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023. 2, 3, 4, 6

[55] Jinbo Xing, Menghan Xia, Yong Zhang, Haoxin Chen, Xintao Wang, Tien-Tsin Wong, and Ying Shan. Dynamicrafter: Animating open-domain images with video diffusion priors. *arXiv preprint arXiv:2310.12190*, 2023. 2

[56] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. *arXiv preprint arXiv:2401.10891*, 2024. 4, 5

[57] Hong-Xing Yu, Haoyi Duan, Junhwa Hur, Kyle Sargent, Michael Rubinstein, William T Freeman, Forrester Cole, Deqing Sun, Noah Snavely, Jiajun Wu, et al. Wonderjourney: Going from anywhere to everywhere. *arXiv preprint arXiv:2312.03884*, 2023. 2

[58] Yifei Zeng, Yanqin Jiang, Siyu Zhu, Yuanxun Lu, Youtian Lin, Hao Zhu, Weiming Hu, Xun Cao, and Yao Yao. Stag4d: Spatial-temporal anchored generative 4d gaussians. *arXiv preprint arXiv:2403.14939*, 2024. 2

[59] Qihang Zhang, Chaoyang Wang, Aliaksandr Siarohin, Peiye Zhuang, Yinghao Xu, Ceyuan Yang, Dahua Lin, Bo Dai, Bolei Zhou, Sergey Tulyakov, and Hsin-Ying Lee. SceneWiz3D: Towards text-guided 3D scene composition. In *arXiv*, 2023. 2

[60] Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T Freeman. Physdreamer: Physics-based interaction with 3d objects via video generation. *arXiv preprint arXiv:2404.13026*, 2024. 2

[61] Zhoutong Zhang, Forrester Cole, Zhengqi Li, Michael Rubinstein, Noah Snavely, and William T. Freeman. Structure and motion from casual videos. In *ECCV*, 2022. 2, 7

[62] Yufeng Zheng, Xueting Li, Koki Nagano, Sifei Liu, Otmar Hilliges, and Shalini De Mello. A unified approach for text-and image-guided 4d scene generation. *arXiv preprint arXiv:2311.16854*, 2023. 2

[63] Xiaoyu Zhou, Xingjian Ran, Yajiao Xiong, Jinlin He, Zhiwei Lin, Yongtao Wang, Deqing Sun, and Ming-Hsuan Yang. Gala3d: Towards text-to-3d complex scene generation via layout-guided generative gaussian splatting. *arXiv preprint arXiv:2402.07207*, 2024. 2