Generative Rendering: Controllable 4D-Guided Video Generation with 2D Diffusion Models

Shengqu Cai[1,2,△] Duygu Ceylan[2*] Matheus Gadelha[2*] Chun-Hao Paul Huang[2]
Tuanfeng Yang Wang[2] Gordon Wetzstein[1]
[1]Stanford University [2]Adobe Research

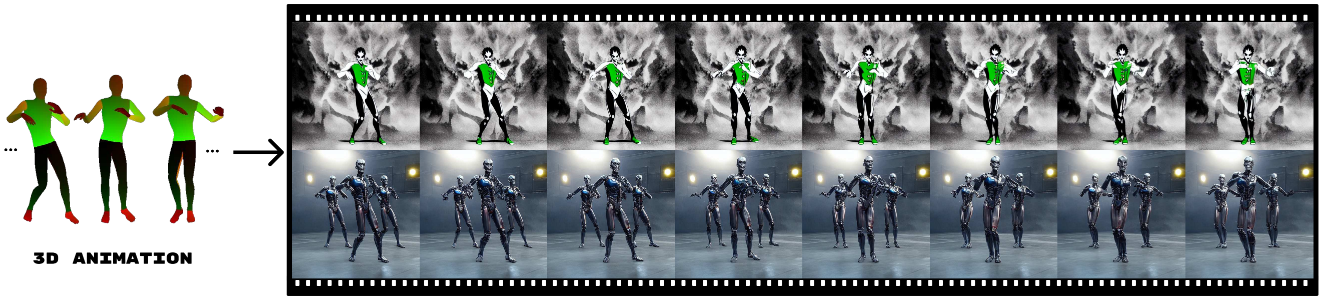


Figure 1. **Generative Rendering** is a novel diffusion-based approach to render 3D animated but untextured scenes (left) directly to a stylized animation (right), with styles specified by text prompts. Our approach offers new levels of user control to image generation models.

Abstract

Traditional 3D content creation tools empower users to bring their imagination to life by giving them direct control over a scene's geometry, appearance, motion, and camera path. Creating computer-generated videos, however, is a tedious manual process, which can be automated by emerging text-to-video diffusion models. Despite great promise, video diffusion models are difficult to control, hindering a user to apply their own creativity rather than amplifying it. To address this challenge, we present a novel approach that combines the controllability of dynamic 3D meshes with the expressivity and editability of emerging diffusion models. For this purpose, our approach takes an animated, low-fidelity rendered mesh as input and injects the ground truth correspondence information obtained from the dynamic mesh into various stages of a pre-trained text-to-image generation model to output high-quality and temporally consistent frames. We demonstrate our approach on various examples where motion can be obtained by animating rigged assets or changing the camera path. Project page: primecai.github.io/generative_rendering.

1. Introduction

Artists, designers, architects, and other creators rely on traditional 3D content creation tools to render computer-generated videos. Unfortunately, existing 3D workflows are laborious, time consuming, and require expertise. Emerging generative artificial intelligence tools, such as text-to-image (T2I) and text-to-video (T2V) models, solve these issues by automating many of the manual steps of traditional workflows. Video generation, however, is difficult to control in that it is not easily possible to specify scene layout and motion in a temporally consistent manner.

Recent approaches have attempted to control diffusion models. For example, ControlNet [41] uses a pre-trained T2I diffusion model and finetunes an adapter network that is conditioned on depth, pose, or edge images to control the layout. This strategy is successful for generating individual frames, but results in flicker for video generation. Other approaches aim at learning the complex types of motions encountered in natural videos directly [3,9,11,12,20,32,35,39,40,42]. While successful in generating smooth motions, these approaches are not easily controllable. Finally, video-to-video diffusion models [6,10,19,37] enable video editing and stylization, but they require a high-fidelity video as input, which is not always available.

In this paper, we aim to combine the power of 3D workflows with T2I models for generating 4D-guided stylized

[△]: Part of this work was done during an internship at Adobe Research.
[*] : Equal contribution.

animations. Specifically, we leverage 3D tools to rapidly prototype proxy geometry and motion of a scene (e.g., camera paths, physically based simulation, or character animation) while utilizing the T2I generation model as a *renderer* to output the final stylized animations (see Fig. 1). At the core of our method is the ability to leverage the ground truth 4D spatio-temporal correspondences that can be obtained from an animated 3D scene to guide the image generation. First, we leverage the correspondence information to effectively perform noise initialization which is critical for temporal consistency. Many recent video-to-video stylization methods resort to inverting the source video to obtain consistent noise initialization [6, 10]. However, this is not possible in our setup since we do not have an initial source video and inversion does not work well on untextured renderings of the 3D scene. Instead, we propose to utilize the canonical representation (i.e., UV space) of the 3D scene to initialize random noise which is then projected to each frame of the animation. Next, we enrich the self-attention layers of the image diffusion model which are known to contain spatial appearance information vital for obtaining consistent renderings [6, 37]. Specifically, we propagate both the input and output of self-attention layers across the frames via known correspondences to enforce more consistent results. Finally, similar to previous work [41], we use the depth cues rendered from the 3D scene in conjunction with control models to provide structure guidance. In summary, our framework treats the pre-trained T2I diffusion model as a multi-frame renderer that is able to capture the correspondences while maintaining high-fidelity and consistent generations.

We evaluate our method on a variety of scenes and demonstrate its advantages with comparisons to state-of-the-art methods. In summary, our contributions include:

- We present a novel framework for 4D-guided animation synthesis utilizing the pre-trained T2I generation models as a multi-frame renderer.
- We enhance the self-attention layers of the image generation model by performing correspondence-aware blending of both input and output features to enforce consistent appearance synthesis.
- We introduce a UV-space noise initialization mechanism. Combined with the correspondence-aware attention mechanism, this enables better consistency across frame generations.

2. Related Work

In this section, we discuss the methods most closely related to ours. A comprehensive review of diffusion models for visual computing can be found in [25].

Video Generation. T2V diffusion models extend T2I models by generating dynamic scenes. The majority of works in this area focus on extending pre-trained T2I diffusion models and finetuning on video data [3, 9, 12, 14]. A typical method of lifting image-based models to videos is to add a time-aware module. This can include 3D convolutions [3], or pseudo-3D convolutions factorized over spatial and temporal dimensions to reduce computational costs [7, 7, 9, 11, 32]. Recently, [11] has shown very compelling video generation results by training a motion module with such temporal layers on top a frozen StableDiffusion model [31]. All of these approaches, however, lack fine-grained structural control of the video generation process in a temporally stable manner. Moreover, training a high-quality T2V model is computationally demanding and challenging, in part due to the limited sizes of available captioned high-quality video datasets. This is part of the reason why existing T2V models are limited to outputting only a few seconds of animation. For these reasons, we build on T2I diffusion models and directly leverage ground truth correspondences of the input meshes to generate highly expressive and temporally consistent animations of arbitrary length.

Video Editing and Stylization. Another line of work focuses on video editing and stylization. Given an input video, video editing/stylization aims to change the input video according to a user-defined prompt, while keeping other components harmonious. Earlier work, such as [2], decomposes a video to atlases and performs editing directly on these atlases. This approach obtains perfect consistency, but it is only applicable to certain types of scenes and the decomposition is very slow to compute. Exemplar works based on diffusion models [6, 7, 19, 27, 37] along this path generally rely heavily on extracting temporal information from the input video, e.g., via DDIM inversion [16], then inflating the self-attention modules to attend to multiple frames, and propagating the edited contents across different frames. These techniques typically overfit to video inputs and are not directly applicable to our setting. For this reason, we adopt these baselines ensuring a fair comparison to their core methods, and demonstrate that our approach outperforms them. In addition, as the temporal prior is either extracted from the input video or learned from video training data, these works generally lack 3D awareness and may not generate 3D-aware animations. A recent attempt designed a 4D representation that employs a canonical field and deformation field to each frame, such that one can easily edit the video by changing contents on the canonical field which propagates naturally to the other frames [23]. However, this representation still has a relatively weak generalization ability and tends to only work well on very short video clips.

In summary, while related, none of these video editing approaches operate in the same setting as our approach, which takes a low-fidelity, untextured rendering of a scene as input to generate a video.

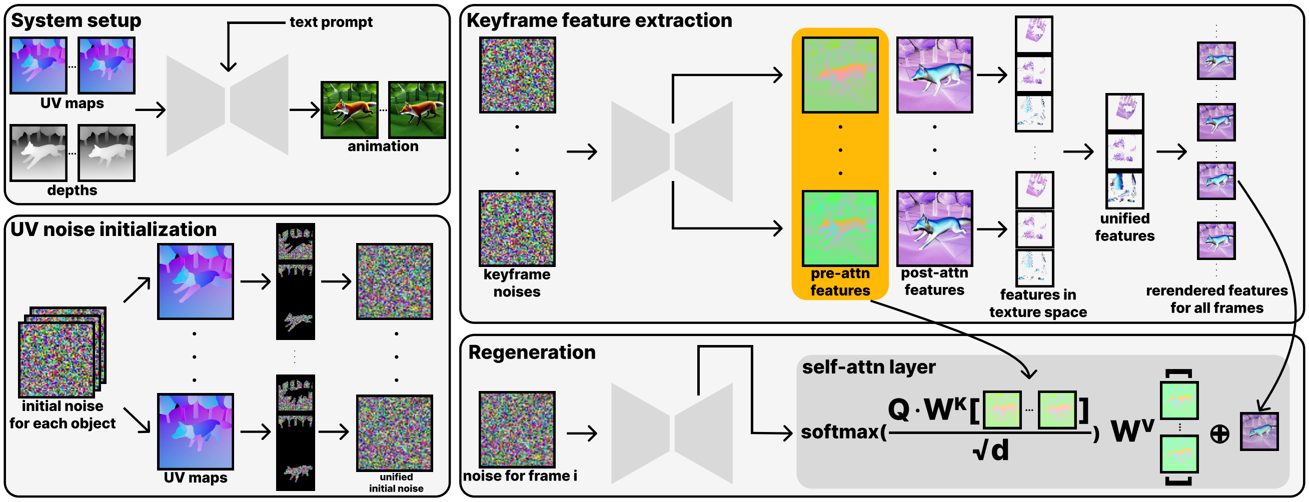


Figure 2. **Overview of our pipeline**. Our system takes as input a set of UV and depths maps rendered from an animated 3D scene. We use a depth-conditioned ControlNet [41] to generate corresponding frames while using the UV correspondences to preserve the consistency. We initialize the noise in the UV space of each object which we then render into each image (Sec. 3.3). For each diffusion step, we first use extended attention for a set of keyframes and extract their pre- and post-attention features (Sec. 3.1). The post-attention features are projected to the UV space and unified. Finally, all frames are generated using a weighted combination of the outputs of the extended attention with the pre-attention features of the keyframe, and the UV-composed post-attention features from the keyframes (Sec. 3.2).

3D Generation. This class of methods is able to achieve near-perfect 3D consistency by using underlying 3D representations. One line of work [4, 8, 15] focuses on scene generation using diffusion models. These works typically predict or estimate the scene geometry on the fly using monocular predictors such as MiDaS [29] and use that information to constitute individual generations into a scene. This approach is applicable only in specific domains such as indoor scenes or nature scenes, with limited motion control ability due to depth–scale ambiguity. The closest form of 3D generation to our task is mesh texturing. Given an explicit representation such as a mesh, prior works [5, 30, 36] demonstrate convincing asset mesh texturization using the powerful inpainting function of Stable Diffusion [31]. Other works [24, 26] use Score Distillation [26] to directly optimize a neural radiance field [22], but they suffer from quality degradation potentially due to heavy classifier-free guidance [13]. Follow-up works [21] attempt to combine the strength of score distillation and mesh texturization. Since these methods generate a static texture for a given mesh, they are not able to capture appearance interactions (e.g., reflections or shadows) in case of scene-level content and motions.

3. Method

The goal of generative rendering is to transfer creator-defined scene-level proxy meshes (without appearance) and motions directly to a convincing animation. Given a 3D scene composed of animated meshes, we render guiding channels that consist of depth maps $\boldsymbol{\mathcal{D}} = [\boldsymbol{D}^1, ..., \boldsymbol{D}^N]$ and UV coordinate maps $\boldsymbol{\mathcal{UV}} = [\boldsymbol{UV}^1, ..., \boldsymbol{UV}^N]$ where N is the length of the sequence and each UV map contains the 2D texture coordinates as well as an object ID in its RGB channels. Using a depth-conditioned 2D image generation model such as StableDiffusion [31] with depth ControlNet [41], we aim to transfer the sequence $(\boldsymbol{D}^i, \boldsymbol{UV}^i)$ into a set of generated images that depict a stylized animation. To achieve this goal, we perform feature blending over the input and output features of the self-attention modules of the diffusion model (Sec. 3.2) and also perform UV-guided noise initialization (Sec. 3.3) to improve the temporal consistency. We provide an overview of our framework in Fig. 2 and our overall algorithm in Alg. 1. Next, we provide some background on the self-attention features of a diffusion model before discussing the different parts of our method in detail.

3.1. Self-attention feature injection

Recent work has demonstrated that self-attention features encode spatial self-similarity [34] and they are effective in terms of maintaining temporal consistency when they are used to perform cross-frame attention [6, 37]. A typical approach is to extend the diffusion network to process multiple frames jointly where they can attend to each other's features. Furthermore, it is possible to manipulate both the input and output features of a self-attention module to further enforce consistency. We denote these operations as *pre-attention* and *post-attention* feature injections respectively and provide a description in the following.

Pre-attention Feature Injection. An attention module is defined as:

$$\boldsymbol{F} = \texttt{Attn}(\boldsymbol{Q};\boldsymbol{K};\boldsymbol{V}) = \texttt{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\cdot \boldsymbol{V}, \quad (1)$$

where $\boldsymbol{Q}$, $\boldsymbol{K}$, and $\boldsymbol{V}$ denote the query, key, and value features respectively. In case of a self-attention module, $\boldsymbol{Q}$, $\boldsymbol{K}$ and $\boldsymbol{V}$ are computed by projecting the same features $\boldsymbol{f}^{(i,l)}$, via the corresponding attention matrices, i.e.:

$$\boldsymbol{Q}^{(i,l)} = \mathbf{W}^{Q}\boldsymbol{f}^{(i,l)}, \boldsymbol{K}^{(i,l)} = \mathbf{W}^{K}\boldsymbol{f}^{(i,l)}, \boldsymbol{V}^{(i,l)} = \mathbf{W}^{V}\boldsymbol{f}^{(i,l)} \quad (2)$$

where i denotes the frame index and l denotes the diffusion step and $\mathbf{W}^{Q/K/V}$ are weight matrices for query, key, and values, respectively. In other words, for each query feature an attention weight is computed based on the similarity between the query and each key feature. Then, the value features are combined based on this attention weight.

One can manipulate the input features provided to the self-attention module, specifically the key and value features, which we denote as *pre-attention feature injection*. As discussed by [10], in case of generating N frames, one naive option is to perform *extended attention* where the features $\boldsymbol{f}^{(i,l)}$ of all the frames are concatenated and utilized as key and value pairs:

$$\begin{aligned}\boldsymbol{K}^{(i,l)} &= \mathbf{W}^{K}[\boldsymbol{f}^{(1,l)}, ..., \boldsymbol{f}^{(N,l)}],\\ \boldsymbol{V}^{(i,l)} &= \mathbf{W}^{V}[\boldsymbol{f}^{(1,l)}, ..., \boldsymbol{f}^{(N,l)}].\end{aligned} \quad (3)$$

Extended attention is a synchronized operation, where all frames attend to features of each other resulting in a more consistent generation. However, generating multiple frames together incurs a high computational cost, which in practice limits the number of frames that can be processed in a sequence. For this reason, we utilize extended attention only for a subset of keyframes and propose mechanisms to propagate the keyframe features to the rest of the sequence, as discussed in the next section.

Post-attention Feature Injection. While it is possible to manipulate the input of the self-attention layer, another option is to directly manipulate the output, which we denote as *post-attention feature injection*. In particular, given known correspondences between two frames i and j, one can reproject $\boldsymbol{F}^{(i,l)}$, the output of the self-attention module of frame i at diffusion step l, to obtain $\boldsymbol{F}^{(j,l)}$:

$$\boldsymbol{F}^{(j,l)} = \pi_{i,j}(\boldsymbol{F}^{(i,l)}), \quad (4)$$

where $\pi_{i,j}$ denotes the re-projection operation, which is enabled by the ground truth pixel-level correspondences between frames given by our UV maps.

While post-attention feature injection is effective in achieving consistency, it tends to create blurry results and also avoids generating of certain interactions between objects (e.g., shadows, reflections) since it overwrites the output the self attention. Hence, as described next, our method combines pre- and post-attention feature injection in an effective way to generate consistent but also high-quality frames.

3.2. UV-space feature injection

Given that pre- and post-attention features of the self-attention module can be manipulated to enhance consistency, our main goal is to perform this manipulation guided by the ground truth correspondences we have from the 3D scene. Specifically, we utilize the canonical UV space to bring the features from the different frames into correspondence which we denote as *UV-space feature blending*. Given a set of feature maps for each frame i, we project them to a canonical UV space. In particular, for each texel in the UV space, we find its closest point correspondence in each frame. Having such correspondences from multiple frames, a straightforward approach is to take the (weighted) average to blend the features [10]. However, we find this consistently leads to sub-optimal blurry results. Therefore, to combat over-smoothing of the features, we additionally perform blending sequentially and fill a certain texel with the features of its corresponding pixel in a frame only if it has not been filled before. The final unified texture is then the mean of the inpainted texture and the average texture. After features from all frames are blended, we obtain a UV-space feature map which we denote as $\boldsymbol{T}^{(i,l)}$. Once such a unified feature map is obtained, we can project it back to each frame to obtain $\bar{f}^{(i,l)}$.

Given the UV-space feature blending mechanism, we next describe how we utilize it during the image generation process. For each diffusion step, we first sample a random batch of m keyframes and perform extended attention on these keyframes, extracting both their pre-attention features $\boldsymbol{f}_{KF}^{(i,l)}$ and post-attention features $\boldsymbol{F}_{KF}^{(i,l)}$. We further composite a UV space feature map, $\bar{\boldsymbol{T}}_{KF}^{l}$, by blending $\boldsymbol{F}_{KF}^{(i,l)}$ as described previously. We then perform the diffusion step on all the frames (in case of keyframes, the diffusion step is simply repeated) in a sequential manner. For each frame, we compose the pre-attention features by concatenating the pre-attention features of the keyframes, $\mathbf{f}_{KF}^{l} = [\boldsymbol{f}_{KF}^{(1,l)}, ..., \boldsymbol{f}_{KF}^{(m,l)}]$, with the pre-attention features of the current frame $\boldsymbol{f}^{(i,l)}$ resulting in:

$$\begin{aligned}\boldsymbol{K}^{(i,l)} &= \mathbf{W}^{K}[\boldsymbol{f}^{(i,l)}, \mathbf{f}_{KF}^{l}],\\ \boldsymbol{V}^{(i,l)} &= \mathbf{W}^{V}[\boldsymbol{f}^{(i,l)}, \mathbf{f}_{KF}^{l}].\end{aligned} \quad (5)$$

After executing the self-attention module, we blend the output, i.e., post-attention features of the current frame $\hat{\boldsymbol{F}}^{(i,l)}$, with the projection of the blended post-attention features to the current frame $\bar{\boldsymbol{F}}^{(i,l)}$. Hence, the updated output of the module becomes:

$$\boldsymbol{F}_{out}^{(i,l)} = \alpha \cdot \bar{\boldsymbol{F}}^{(i,l)} + (1-\alpha)\cdot \hat{\boldsymbol{F}}^{(i,l)} \quad (6)$$

where $\hat{\boldsymbol{F}}^{(i,l)}$ is the output of the inflated self-attention block computed using keys and values in Eq. 5.

3.3. UV noise initialization

Prior video editing works [6, 10] generally rely on inversion methods, such as DDIM inversion [16], to invert each source frame to the noise space which are then used as initialization for editing. Since source frames in the input video are coherent, this inversion strategy yields relatively coherent noise patterns that in turn helps with consistent editing. However, in our case, since we have an untextured scene, inverting rendered frames that lack appearance does not lead to coherent noisy latents. Another option is to keep the noise fixed across all frames. However, this results in severe texture-sticking issues as we demonstrate in the supplementary material. We seek an alternative to capture temporal correlations in initial noise patterns. Inspired by previous work [9, 18], we leverage the fact that we have a canonical UV space by initializing the noise in this space by creating a Gaussian noise texture. We then project this noise to each frame by utilizing the frame–UV correspondences. As we illustrate in our evaluations, this significantly boosts the coherency of the generated outputs.

3.4. Latent normalization

We notice a color flickering issue, where the overall color distribution and contrast change between frames, even if the motion is limited. Therefore, we apply an AdaIN operation on the predicted latents to make every frame's overall distribution similar to the first frame. The simple AdaIN operation solves this issue for some samples, but we find that calculating the distribution over all pixels is very sensitive to the motions. Therefore, inspired by [1], we calculate the statistics only on the background pixels, which are easy to separate in our setup.

4. Experiments

Datasets. We evaluate our method on scenes with various complexity demonstrating three main types of animation sequences: **1)** Camera rotations: given a single static object we create a camera path with 360-rotation around the object; **2)** Physical simulation: we prepare a motion sequence by utilizing physically based simulation, e.g., bouncing balls where we use rigid body dynamics for collision handling; **3)** Character animation: we create scenes with rigged and animated characters obtained from Mixamo and render them with various dance motions.

Baselines. To our knowledge, no published work directly focuses on the same task our method does. Therefore, we adapt two recent methods to our setting as baselines. 1) we use the pre-attention feature propagation mechanism of Pix2Video [6] (i.e., each frame attends to an anchor and previous frame) together with depth conditioning, and 2) we apply the extended attention mechanism and token propagation from TokenFlow [10] using nearest neighbor matching. We note that the original TokenFlow method computes correspondences by matching features obtained via inverting original frames. Since this is not meaningful in our case, we instead provide the ground truth correspondences we obtain from 3D. Both Pix2Video and TokenFlow rely on inversion for noise initialization. Again, this is not meaningful in our setting, so we instead use the same noise to initialize each frame.

Algorithm 1 Generative Rendering

```
Input: UV maps $\boldsymbol{\mathcal{UV}} = [\boldsymbol{UV}^1, ..., \boldsymbol{UV}^N]$; depth maps $\boldsymbol{\mathcal{D}} = [\boldsymbol{D}^1, ..., \boldsymbol{D}^N]$; text prompt $\boldsymbol{c}$; diffusion model $\Phi$; projection from UV to image $\pi_{\boldsymbol{\mathcal{UV}}}$; $T$ diffusion steps; $k$ keyframes

$z_{\boldsymbol{\mathcal{UV}}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$    // Sample noise in UV space
$\mathbf{z} = \mathbf{z}_T = \pi_{\boldsymbol{\mathcal{UV}}}(z_{\boldsymbol{\mathcal{UV}}})$
For $t = T, \ldots, 1$ do
    $KF = \text{shuffle}(\{i_1, \ldots, i_k\})$    // Sample keyframes
    $\boldsymbol{f}_{KF}, \boldsymbol{F}_{KF} \leftarrow \Phi(\mathbf{z}_{KF}, \boldsymbol{\mathcal{D}}_{KF}, c)$    // Grab features from $\Phi$
    $\bar{\boldsymbol{T}}_{KF} = \text{blend}(\pi^{-1}_{\boldsymbol{\mathcal{UV}}}(\boldsymbol{F}_{KF}))$    // Blends features in UV space
    $\mathbf{z}_{t-1} = [\,]$
    For $z^i$ in $\mathbf{z}$ do
        $\bar{\boldsymbol{F}}^i = \pi_{UV^i}(\bar{\boldsymbol{T}}_{KF})$
        $z^i_{t-1} = \Phi(z^i, \boldsymbol{\mathcal{D}}_{KF}, c, \boldsymbol{f}_{KF}, \bar{\boldsymbol{F}}^i)$
        $\mathbf{z}_{t-1}.\text{append}(z^i_{t-1})$
        $\mathbf{z} = \mathbf{z}_{t-1}$
Output: $\mathbf{z}$
```

Evaluation metrics. Following prior video editing evaluation protocols [3, 6, 7, 10, 37], we report two metrics based on CLIP [28]. To measure *frame consistency*, we compute CLIP image embeddings for each frame of the output video and compute the average cosine similarity between all pairs of consecutive video frames. To measure *prompt fidelity*, we compute the mean CLIP embedding similarity score between each frame of the output video and the corresponding input prompt.

Implementation details. We use the StableDiffusion [31] v1.5 model and a depth-conditioned ControlNet [41] as our image generator. We generate all results at 512 × 512 resolution using Karras scheduler [17] with 50 denoising steps.

https://www.mixamo.com/

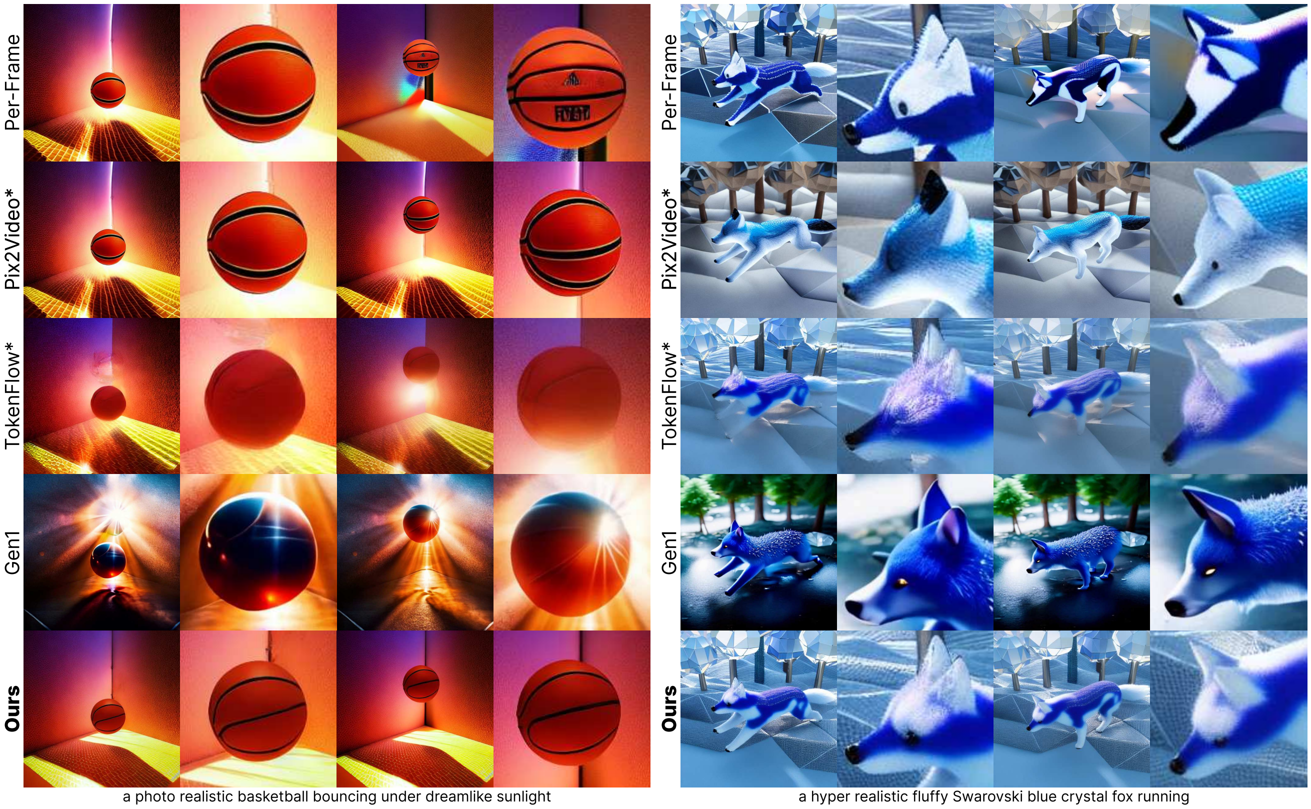

Per-Frame
Pix2Video*
TokenFlow*
Gen1
Ours
a photo realistic basketball bouncing under dreamlike sunlight
Per-Frame
Pix2Video*
TokenFlow*
Gen1
Ours
a hyper realistic fluffy Swarovski blue crystal fox running

Figure 3. **Qualitative comparisons.** We adapt the feature blending mechanisms proposed by Pix2Video [6] and TokenFlow [10] to our setting. While Pix2Video produces reasonable results, it lacks a mechanism to maintain frame-level consistency. TokenFlow uses a token propagation & interpolation mechanism between nearby keyframes, but this results in blurred outputs in our setting.

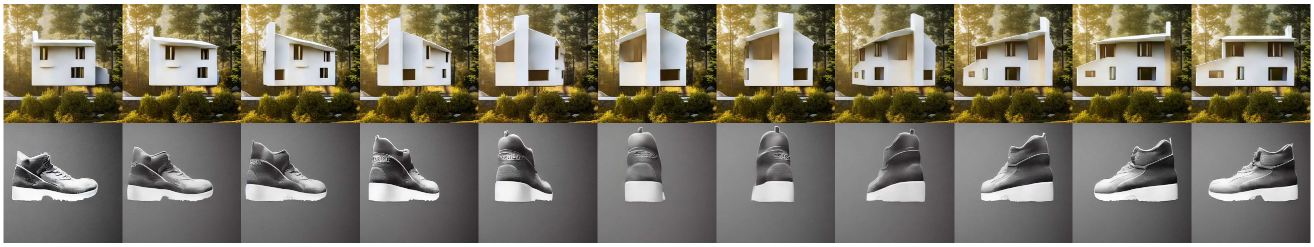
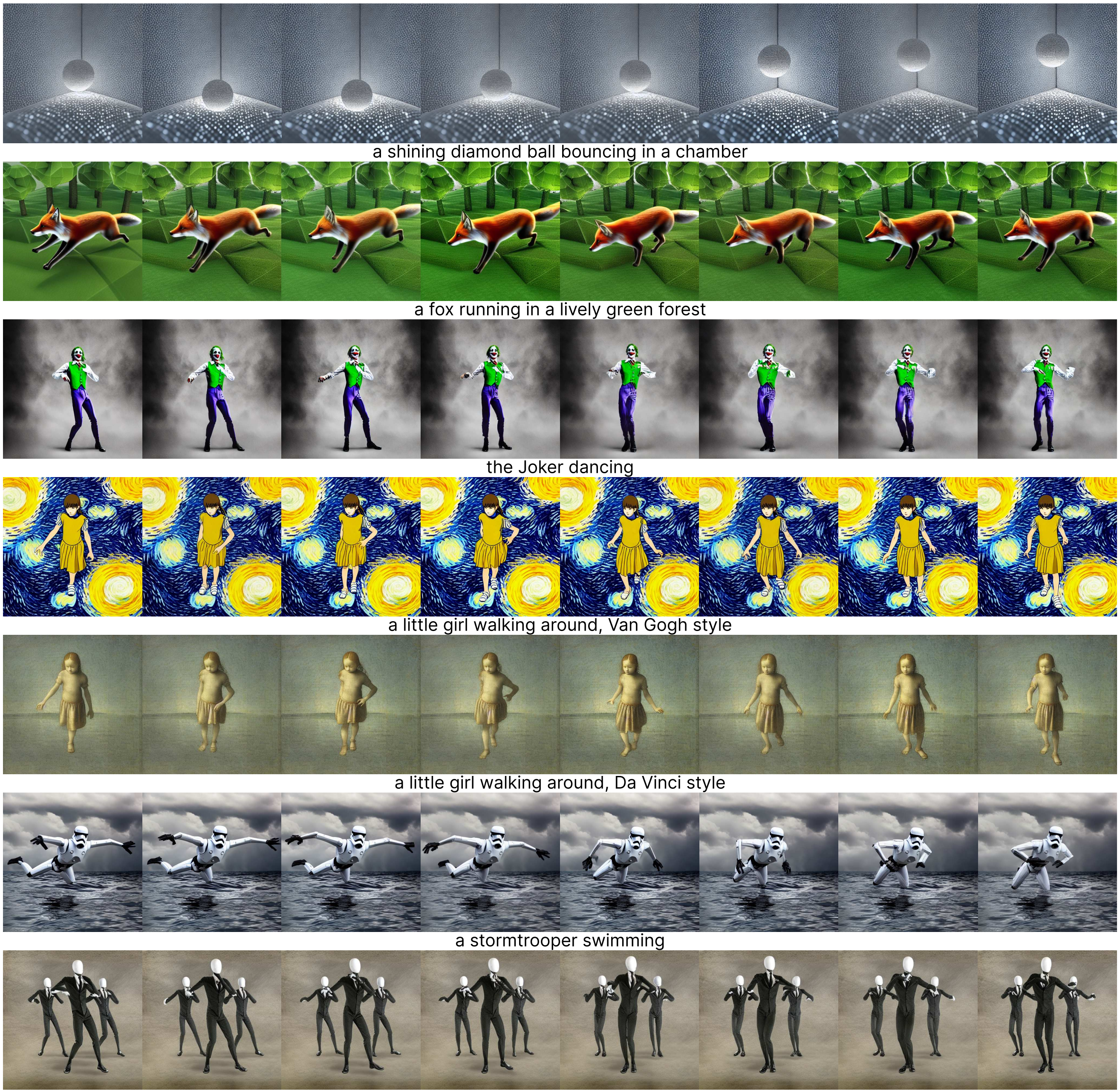


Figure 4. **Camera and object rotation.** Our algorithm supports camera and object rotations for static scenarios.

4.1. Results

Qualitative results. Qualitative comparisons to the baselines are shown in Fig. 3. While Pix2Video [6] produces high-quality frames it shows limited spatio-temporal consistency, because it purely relies on the cross-frame attention mechanism to attend to corresponding features. TokenFlow [10] can produce smoother results, but we find that it consistently produces blurry textures. This is in part due to the lack of DDIM inversion [16], which provides a decent level of feature similarities across frames. In our setting, this DDIM inversion is not meaningful and skipping the self-attention module to directly blend the features of keyframes means averaging less similar features, resulting in blurry appearance. Without accurate DDIM inversion and correct nearest neighbor matching, the blending oversmoothes the features and lead to blurry outputs of low quality. In addition, unlike nearest neighbor correspondences computed using DDIM inversion, UV correspondences are not guaranteed to be an exact match in the low-dimensional feature space, which is only 64 × 64 pixels in our model, especially around object edges. Consequently, a straight blending of the post-attention features also creates blurry results.

Our method can handle camera rotation with reasonable quality in static object-level scenarios, as shown in Fig. 4. In Fig. 5, we provide additional results on the same input



a shining diamond ball bouncing in a chamber

a fox running in a lively green forest

the Joker dancing

a little girl walking around, Van Gogh style

a little girl walking around, Da Vinci style

a stormtrooper swimming

three slendermen dancing

Figure 5. **Qualitative results.** Our method generalizes to a diverse set of input meshes, styles, and prompts.

meshes but with different prompts. Our method shows robustness across a diverse set of prompts and styles. Our inflated attention modules can capture motion-specific traits, such as shadows (Row 3, Row 7), lighting changes (Row 1), and reasonable deformations that are not modeled by the input low-fidelity meshes (Row 1 ground regions).

Quantitative results. We show quantitative evaluations for frame consistency and prompt fidelity in Tab. 1. We observe that our frame consistency is better than all baselines using T2I models. As expected, Pix2Video [6] suffers more from temporal changes due to the lack of a correspondence injection module. Even though TensorFlow's token propagation encourages smoothness over fidelity, our method still shows better consistency. Only the T2V model Gen1 has a better frame consistency, which is expected because it is trained on video data. All current T2V models, including Gen1, can only generate a few frames and are limited in other ways. Our approach is an attempt to overcome this deficiency using

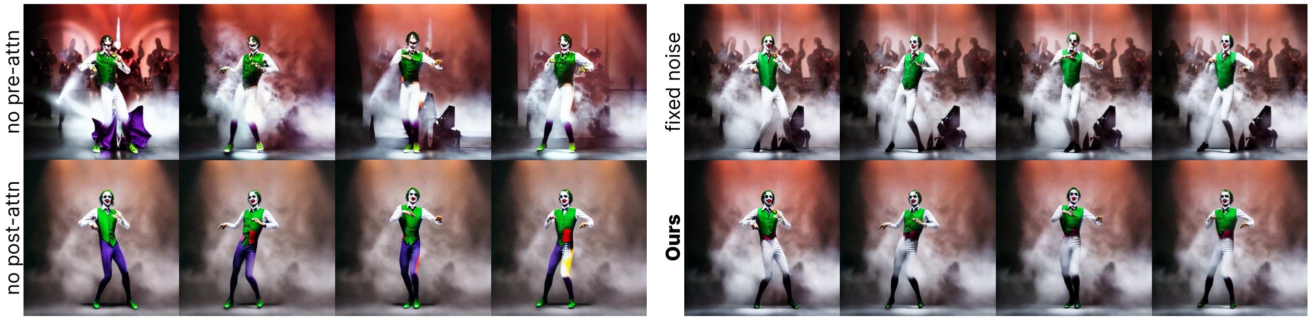


Figure 6. **Ablation study.** We disabled several key building blocks and observe clear artifacts or inconsistencies in all ablations.

Method	Frame Consistency ↑	Prompt Fidelity ↑
Per-Frame	0.9547	**0.3233**
Pix2Video*	0.9630	0.2983
TokenFlow*	0.9822	0.2737
Gen1	**0.9907**	0.3029
Ours	0.9845	0.3227

Table 1. **Quantitative evaluation**. Our approach offers the best frame consistency among approaches using T2I models. As expected, only the proprietary T2V model Gen1 observes better temporal consistency. Our method also shows a higher prompt fidelity than relevant baselines that offer a reasonable amount of temporal consistency. *baseline adapted to our setting.

2D T2I models exclusively. Fidelity-wise, we are slightly better than Pix2Video [6], because it reuses the set of pre-attention features from previous frames, while we only reuse a portion of the pre-attention features to generate each frame. Since TokenFlow [10]'s token propagation mechanism introduces over-smoothing effects, its results are blurry for many of the samples, hence achieving lower fidelity scores. Our generative rendering utilizes the benefits of pre-attention inflation to gain overall global structural and semantic consistency, while also combining with post-attention feature warping as local guidance. Combined with our UV noise initialization, we outperform or are on par with all relevant baselines in the proposed setting.

Ablation studies. We perform a thorough ablation study to verify our design choices, shown in Fig. 6. We study three aspects of our pipeline: 1) comparing with pre-attention injection only, 2) comparing with post-attention injection only, and 3) the effectiveness of UV noise initialization. We observe that directly unifying and projecting the post-attention features without passing through the attention module produces chaotic results (severe artifacts in "no pre-attn", the joker's face and clothes regions). This observation is in line with our hypothesis for the poor performance of the adapted TokenFlow mechanism. When the motion is relatively large, geometrically matched features may not be good matches in the feature space. Therefore, directly warping a feature from one frame to another based on the geometrical correspondence has a high chance of not producing reasonable outputs. However, when we combine post-attention feature warping with pre-attention feature injection, the post-attention features can act as guidance for the next network layer. Without post-attention feature warping, purely pre-attention feature injection does not guarantee pixel-wise consistency since the attention operation merely computes a correspondence based on feature similarities (in "no post-attn", the joker's clothes change over frames). Our UV noise initialization is well-aligned with our setup. If we use fixed noise instead, the blending of the features may cause unwanted artifacts (in "fixed noise", the joker's face has blending artifacts and is not recognizable in the third image). Our UV noise initialization serves generally and can work beyond our setup. Additional details on ablations are included in the supplementary.

5. Discussion

In this paper, we introduce *Generative Rendering*, a novel zero-shot pipeline based on 2D diffusion models for 4D-conditioned animation generation. Our method can animate creator-defined low-fidelity meshes and motion sequences, thereby bypassing steps requiring significant manual labor such as detailing, texturing, physical simulation, etc. The key idea of generative rendering is to utilize the prior within a depth-conditioned 2D diffusion model to provide the basic structure and physical fidelity to make convincing animations. This is accomplished by injecting the correspondences into the diffusion model using a combination of pre- and post-attention feature injection while unifying these features in the UV space. Our model demonstrates better frame consistency and prompt fidelity than relevant baselines.

Limitation and future work. Generative rendering cannot achieve real-time animations due to the multi-step inference of current diffusion models. However, accelerating this inference stage has been an active research area and advances in this area, such as consistency models [33], can be directly applied to speed up our method. Furthermore, generative rendering is not yet able to guarantee perfect consistency and preservation of details. This is mainly because our method works purely in the low-dimensional latent space, which is only 64×64 pixels in our model resulting in imprecise UV correspondences. For the same reason, generative rendering may suffer from unwanted misalignment and artifacts. We believe that applying some of findings to pre-trained video diffusion models to augment them with 3D controllability is an exciting future direction.

Conclusion. Diffusion models have emerged as state-of-the-art generative methods across multiple domains, such as 2D, 3D, and video generation and editing. generative rendering adds unprecedented levels of control to video generation with T2I diffusion models by bridging the gap between the traditional rendering pipeline and diffusion models.

Acknowledgement. G.W. was in part supported by Google, Samsung, and Stanford HAI.

References

[1] Yuval Alaluf, Daniel Garibi, Or Patashnik, Hadar Averbuch-Elor, and Daniel Cohen-Or. Cross-image attention for zero-shot appearance transfer. In *arXiv*, 2023. 5

[2] Omer Bar-Tal, Dolev Ofri-Amar, Rafail Fridman, Yoni Kasten, and Tali Dekel. Text2live: Text-driven layered image and video editing. In *ECCV*, 2022. 2

[3] 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 *CVPR*, 2023. 1, 2, 5

[4] Shengqu Cai, Eric Ryan Chan, Songyou Peng, Mohamad Shahbazi, Anton Obukhov, Luc Van Gool, and Gordon Wetzstein. Diffdreamer: Towards consistent unsupervised single-view scene extrapolation with conditional diffusion models. In *ICCV*, 2023. 3

[5] Tianshi Cao, Karsten Kreis, Sanja Fidler, Nicholas Sharp, and Kangxue Yin. Texfusion: Synthesizing 3d textures with text-guided image diffusion models. In *ICCV*, 2023. 3

[6] Duygu Ceylan, Chun-Hao Huang, and Niloy J. Mitra. Pix2video: Video editing using image diffusion. In *ICCV*, 2023. 1, 2, 3, 5, 6, 7, 8, 11

[7] Patrick Esser, Johnathan Chiu, Parmida Atighehchian, Jonathan Granskog, and Anastasis Germanidis. Structure and content-guided video synthesis with diffusion models. In *ICCV*, 2023. 2, 5, 11

[8] Rafail Fridman, Amit Abecasis, Yoni Kasten, and Tali Dekel. Scenescape: Text-driven consistent scene generation. In *arXiv*, 2023. 3

[9] 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 *ICCV*, 2023. 1, 2, 5

[10] Michal Geyer, Omer Bar-Tal, Shai Bagon, and Tali Dekel. Tokenflow: Consistent diffusion features for consistent video editing. In *arXiv*, 2023. 1, 2, 4, 5, 6, 8, 11

[11] 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. In *arXiv*, 2023. 1, 2

[12] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models. In *arXiv*, 2022. 1, 2

[13] Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *arXiv*, 2022. 3

[14] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. In *arXiv*, 2022. 2

[15] Lukas Höllein, Ang Cao, Andrew Owens, Justin Johnson, and Matthias Nießner. Text2room: Extracting textured 3d meshes from 2d text-to-image models. In *ICCV*, 2023. 3

[16] Inbar Huberman-Spiegelglas, Vladimir Kulikov, and Tomer Michaeli. An edit friendly ddpm noise space: Inversion and manipulations. In *arXiv*, 2023. 2, 5, 6, 11

[17] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *NeurIPS*, 2022. 5

[18] Michael Kass and Davide Pesare. Coherent noise for non-photorealistic rendering. In *TOG*, 2011. 5

[19] Levon Khachatryan, Andranik Movsisyan, Vahram Tadevosyan, Roberto Henschel, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Text2video-zero: Text-to-image diffusion models are zero-shot video generators. In *arXiv*, 2023. 1, 2

[20] Xin Li, Wenqing Chu, Ye Wu, Weihang Yuan, Fanglong Liu, Qi Zhang, Fu Li, Haocheng Feng, Errui Ding, and Jingdong Wang. Videogen: A reference-guided latent diffusion approach for high definition text-to-video generation. In *arXiv*, 2023. 1

[21] 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 *CVPR*, 2023. 3

[22] 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. In *ECCV*, 2020. 3

[23] Hao Ouyang, Qiuyu Wang, Yuxi Xiao, Qingyan Bai, Juntao Zhang, Kecheng Zheng, Xiaowei Zhou, Qifeng Chen, and Yujun Shen. Codef: Content deformation fields for temporally consistent video processing. In *arXiv*, 2023. 2

[24] Ryan Po and Gordon Wetzstein. Compositional 3d scene generation using locally conditioned diffusion. In *arXiv*, 2023. 3

[25] Ryan Po, Wang Yifan, Vladislav Golyanik, Kfir Aberman, Jonathan T Barron, Amit H Bermano, Eric Ryan Chan, Tali Dekel, Aleksander Holynski, Angjoo Kanazawa, et al. State of the art on diffusion models for visual computing. In *arXiv*, 2023. 2

[26] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*, 2023. 3

[27] Chenyang Qi, Xiaodong Cun, Yong Zhang, Chenyang Lei, Xintao Wang, Ying Shan, and Qifeng Chen. Fatezero: Fusing attentions for zero-shot text-based video editing. In *ICCV*, 2023. 2

[28] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *CoRR*, 2021. 5, 11

[29] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. In *TPAMI*, 2022. 3

[30] Elad Richardson, Gal Metzer, Yuval Alaluf, Raja Giryes, and Daniel Cohen-Or. Texture: Text-guided texturing of 3d shapes. In *SIGGRAPH*, 2023. 3

[31] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 2, 3, 5, 11

[32] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-a-video: Text-to-video generation without text-video data. In *arXiv*, 2022. 1, 2

[33] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *arXiv*, 2023. 8

[34] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *CVPR*, 2023. 3

[35] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. Modelscope text-to-video technical report. In *arXiv*, 2023. 1

[36] Tianfu Wang, Menelaos Kanakis, Konrad Schindler, Luc Van Gool, and Anton Obukhov. Breathing new life into 3d assets with generative repainting. In *BMVC*, 2023. 3

[37] Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Wynne Hsu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In *arXiv*, 2022. 1, 2, 3, 5

[38] Shuai Yang, Yifan Zhou, Ziwei Liu, , and Chen Change Loy. Rerender a video: Zero-shot text-guided video-to-video translation. In *SIGGRAPH Asia*, 2023. 11

[39] Sihyun Yu, Kihyuk Sohn, Subin Kim, and Jinwoo Shin. Latent video diffusion models for high-fidelity video generation with arbitrary lengths. In *arXiv*, 2022. 1

[40] Sihyun Yu, Kihyuk Sohn, Subin Kim, and Jinwoo Shin. Video probabilistic diffusion models in projected latent space. In *CVPR*, 2023. 1

[41] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *ICCV*, 2023. 1, 2, 3, 5

[42] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. Magicvideo: Efficient video generation with latent diffusion models. In *arXiv*, 2022. 1

Generative Rendering: Controllable 4D-Guided Video Generation with 2D Diffusion Models

Supplementary Material

	Pairwise Frame Interval				
Method	1	5	10	15	20
Per-Frame	0.9547	0.9173	0.9109	0.9145	0.9062
Pix2Video*	0.9630	0.9503	0.9494	0.9415	0.9471
TokenFlow*	0.9822	0.9754	0.9728	0.9706	0.9712
Gen1	**0.9907**	0.9715	0.9582	0.9624	0.9601
Ours	0.9845	**0.9815**	**0.9738**	**0.9749**	**0.9727**

Table 2. **Frame consistency with different intervals.** Our approach offers better or competitive consistencies at different intervals. *: baseline adapted to our setting.

A. Additional qualitative results

We refer to the supplemental webpage for various qualitative results obtained by using different 3D scenes, motions, and target prompts.

B. Additional quantitative results

We report CLIP [28] embedding similarities for a pair of frames separated by different frame intervals in Tab. 2. As shown, semantic consistency achieved by our method is stable across different intervals. This is potentially because of our texture-based feature aggregation. This provides a canonical representation for the objects that links frames that are temporally further apart even in long sequences. Even though video-based methods such as Gen1 [7] achieve temporally smoother results, we speculate that it is not easy for them to capture such long range interactions.

C. Robustness to video length

Our method can be scaled up to handle long input sequences. To generate a long output video, our method requires the sampling of a sparse set of keyframes, and the unified features in the texture space can be propagated. The only GPU memory bottleneck in our method is the parallel processing of the keyframes using inflated attention. Our algorithm is robust enough to generate long videos with >200 frames with <12GB GPU memory.

D. Additional ablations

Effectiveness of UV noise initialization. We find that the performance of many components in our pipeline, e.g. inflated attention, are tightly related to the initial noise. Video editing works [6, 10, 38] typically require accurate DDIM inversion [16] as noise initialization. Using DDIM inverted noise encourages temporal coherence and similarities to the input video. However, since our input is textureless UV maps and depths, DDIM inversion constantly fails and will not provide useful information. We instead propose to utilize the canonical UV space to initialize the noise in each frame of the sequence. In our supplemental page, we show the effectiveness of this initialization strategy.

Effectiveness of latent normalization. Our latent normalization can address a large portion of the color flickering issue, where the overall color distribution shifts randomly for different frames. We notice that small color shifts in the latent space will be inflated by the VAE decoder used in Stable Diffusion [31], causing the generation process very sensitive to small differences in the latent space. The video comparison is included in our supplemental page.

E. Limitation and failure cases

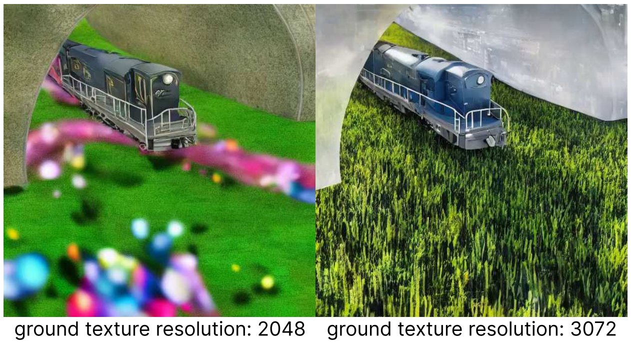

ground texture resolution: 2048
ground texture resolution: 3072

Figure 7. **Effect of different texture resolution.** If the ground's texture resolution is too low (2048×2048), artifacts appear at ground regions in the final rendered image compared with using a higher texture resolution (3072×3072). Prompt: a train running in a field of green grass.

Our method still suffers from several aspects. A large portion of our inconsistency comes from the VAE decoder. As mentioned in D, small inconsistencies in the latent space will be inflated by the VAE decoder while transiting into RGB frames. This phenomenon scales to image details, where small differences in the latent space will be inflated after decoding with the VAE. Additionally, since our noise initialization and feature fusion both work in the UV space, it could be tricky to set the texture resolutions. Setting the texture resolution too low will create corrupted regions while too high will cause the UV coordinates to be too far away and no blending effect will take place. An example is shown in Fig. 7. Finally, our work does not yet generalize to large environmental changes and dramatic perspective changes. This is because of the highly overlapping pixel-wise matches and bad feature projections. We believe finetuning or adding a video module to be an exciting future direction to improve the performances on these more challenging scenes.