DreamScene4D: Dynamic Multi-Object Scene Generation from Monocular Videos

Wen-Hsuan Chu[†], **Lei Ke**[†], **Katerina Fragkiadaki**
Carnegie Mellon University
{wenhsuac,leik,katef}@cs.cmu.edu

https://dreamscene4d.github.io/

Abstract

View-predictive generative models provide strong priors for lifting object-centric images and videos into 3D and 4D through rendering and score distillation objectives. A question then remains: what about lifting complete multi-object dynamic scenes? There are two challenges in this direction: First, rendering error gradients are often insufficient to recover fast object motion, and second, view predictive generative models work much better for objects than whole scenes, so, score distillation objectives cannot currently be applied at the scene level directly. We present DreamScene4D, the first approach to generate 3D dynamic scenes of multiple objects from monocular videos via 360° novel view synthesis. Our key insight is a "*decompose-recompose*" approach that factorizes the video scene into the background and object tracks, while also factorizing object motion into 3 components: object-centric deformation, object-to-world-frame transformation, and camera motion. Such decomposition permits rendering error gradients and object view-predictive models to recover object 3D completions and deformations while bounding box tracks guide the large object movements in the scene. We show extensive results on challenging DAVIS, Kubric, and self-captured videos with quantitative comparisons and a user preference study. Besides 4D scene generation, DreamScene4D obtains accurate 2D persistent point track by projecting the inferred 3D trajectories to 2D. We will release our code and hope our work will stimulate more research on fine-grained 4D understanding from videos.

1 Introduction

Videos are the result of entities moving and interacting in 3D space and over time, captured from a moving camera. Inferring the dynamic 4D scene from video projections in terms of complete 3D object reconstructions and their 3D motions across seen and unseen camera views is a challenging problem in computer vision. It has multiple important applications, such as 3D object and scene state tracking for robot perception [16, 36], action recognition, visual imitation, digital content creation/simulation, and augmented reality.

Video-to-4D is a highly under-constrained problem since multiple 4D generation hypotheses project to the same video observations. Existing 4D reconstruction works [39, 33, 23, 28, 6, 25] mainly focus on the visible part of the scene contained in the video by learning a differentiable 3D representation that is often a neural field [31] or a set of 3D Gaussians [20] with temporal deformation. *What about the unobserved views of the dynamic 3D scene*? Existing 4D generation works utilize generative models to constrain the appearance of objects in unseen views through score distillation losses. Text-to-4D [46, 2, 53, 24, 1] or image-to-4D [64, 43, 58, 66] setups take a single text prompt or image as input to create a 4D object. Several works [17, 14, 43, 32, 61, 62] explore the video-to-4D

[†]Equal contribution

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

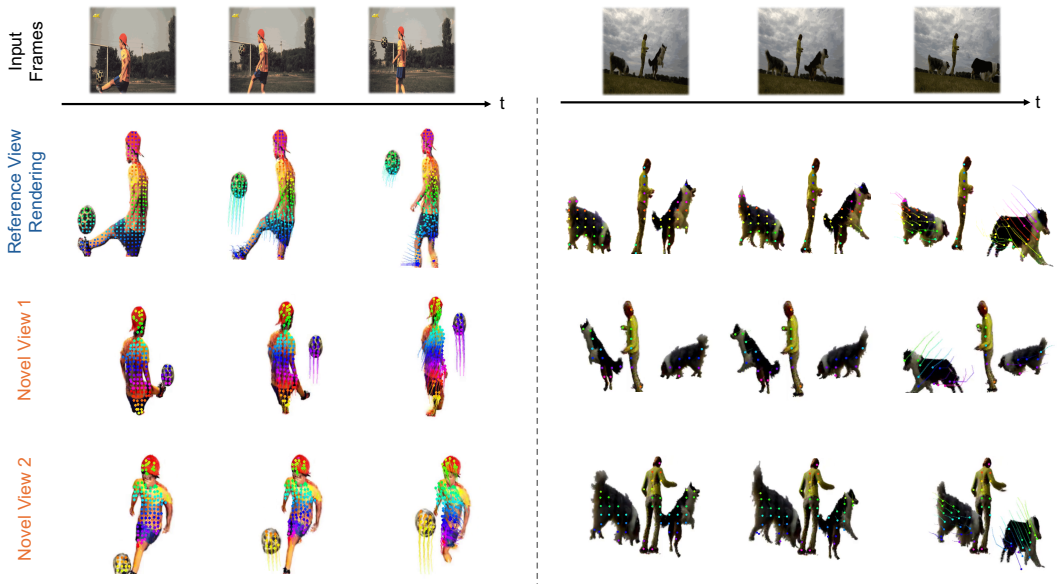

Figure 1: **DreamScene4D** extends video-to-4D generation to multi-object videos with fast motion. We present rendered images and the corresponding motions from diverse viewpoints at different timesteps using real-world DAVIS [37] videos with multiple objects and large motions.

setup, but these methods predominantly focus on videos containing a single object with minor 3D deformations, where the object deforms in place without large motion in the 3D scene. This focus arises because current generative models perform significantly better at predicting novel views of individual objects [26] than multi-object scenes. Consequently, score distillation objectives for 3D object lifting are difficult to apply directly at a scene level. Also, optimization difficulty arises when neural fields or 3D Gaussians are trained to model large temporal deformations directly. This limits their practical real-world usage where input videos depicting real-world complex scenes containing multiple dynamic objects with fast motions, as illustrated in Figure 4.

In this paper, we propose DreamScene4D, the first video-to-4D scene generation approach to produce realistic 4D scene representation from a complex multi-object video with large object motion or deformation. To 360° synthesize novel views for multiple objects of the scene, DreamScene4D proposes a "decompose-recompose" strategy. A video is first decomposed into objects and the background scene, where each is completed across occlusions and viewpoints, then recomposed to estimate relative scales and rigid object-to-world transformations in each frame using monocular depth guidance, so all objects are placed back in a common coordinate system.

To handle fast-moving objects, DreamScene4D factorizes the 3D motion of the static object Gaussians into 3 components: 1) camera motion, 2) object-centric deformations, and 3) an object-centric to world frame transformation. This factorization greatly improves the stability of the motion optimization process by leveraging powerful object trackers [9] to handle large motions and allowing view-predictive generative models to receive object-centric inputs that are in distribution. The camera motion is estimated by re-rendering the static background Gaussians to match the video frames.

We show the view renderings at various timesteps and diverse viewpoints of DreamScene4D using challenging monocular videos from DAVIS [37] in Figure 1. DreamScene4D achieves significant improvements compared to the existing SOTA video-to-4D generation approaches [43, 17] on DAVIS, Kubric [15], and our self-captured videos with fast moving objects (Figures 4). To evaluate the quality of the learned Gaussian motions, we measure the 2D endpoint error (EPE) of the inferred 3D motion trajectories across occlusions and show that our approach produces accurate and persistent point trajectories in both visible views and synthesized novel views.

2 Related Work

Video-to-4D Reconstruction Dynamic 3D reconstruction extends static 3D reconstruction to dynamic scenes with the goal of 3D lifting the visible parts of the video. Dynamic NeRF-based

methods [39, 33, 23, 27, 5] extend NeRF [31] to dynamic scenes, typically using grid or voxel-based representations [28, 6, 25], or learning a deformation field [6, 13] that models the dynamic portions of an object or scene. Dynamic Gaussian Splatting [30] extends 3D Gaussian Splatting [20], where scenes are represented as 4D Gaussians and show faster convergence than NeRF-based approaches. However, these 4D scene reconstruction works [30, 51, 59] typically take videos where the camera has a large number of multi-view angles, instead of a general monocular video input. This necessitates precise calibration of multiple cameras and constrains their potential real-world applicability. Different from these works [34, 51] on mostly reconstructing the visible regions of the dynamic scene, DreamScene4D can 360° synthesize novel views for multiple objects of the scene, including the unobserved regions in the video.

Video-to-4D Generation In contrast to 4D reconstruction works, this line of research is most related by attempting to complete and 3D reconstruct a video scene across both visible and unseen (virtual) viewpoints. Existing text to image to 4D generation works [43, 17, 14, 32, 61, 62] typically use score distillation sampling (SDS) [38] to supply constraints in unseen viewpoints in order to synthesize full 4D representations of objects from single text [46, 2, 24, 1], image [58, 66], or a combination of both [64] prompts. They first map the text prompt or image prompt to a synthetic video, then lift the latter using deformable 3D differentiable NeRFs [23] or set of Gaussians [51] representation. Existing video-to-4D generation works [43, 17, 14, 32, 61, 62] usually simplify the input video by assuming a non-occluded and slow-moving object while real-world videos with multiple dynamic objects inevitably contain occlusions. Owing to our proposed scene decoupling and motion factorization schemes, DreamScene4D is the first approach to generate complicated 4D scenes and synthesize their arbitrary novel views by taking real-world videos of multi-object scenes.

3 Approach

To generate dynamic 4D scenes of multiple objects from a monocular video input, we propose DreamScene4D, which takes Gaussian Splatting [20, 51] as the 4D scene representation and leverages powerful foundation models to generalize to diverse zero-shot settings.

3.1 Background: Generative 3D Gaussian Splatting

Gaussian Splatting [20] represents a scene with a set of 3D Gaussians. Each Gaussian is defined by its centroid, scale, rotation, opacity, and color, represented as spherical harmonics (SH) coefficients.

Generative 3D Gaussian Splatting via Score Distillation Sampling Score Distillation Sampling (SDS) [38] is widely used for text-to-3D or image-to-3D tasks by leveraging a diffusion prior for optimizing 3D Gaussians to synthesize novel views. For 3D object generation, DreamGaussian [49] uses Zero-1-to-3 [26], which takes a reference view and a relative camera pose as input and generates plausible images for the target viewpoint, for single frame 2D-to-3D lifting. The 3D Gaussians of the input reference view I_1 are optimized by a rendering loss and an SDS loss [38]:

$$\nabla_\phi \mathcal{L}^{\mathrm{t}}_{\mathrm{SDS}} = \mathbb{E}_{t,\tau,\epsilon,p}\left[w(\tau)\left(\epsilon_\theta\left(\hat{I}^p_t;\tau,I_1,p\right)-\epsilon\right)\frac{\partial \hat{I}^p_t}{\partial \phi}\right], \tag{1}$$

where t is the timestep indices, $w(\tau)$ is a weighting function for denoising timestep τ, $\phi(\cdot)$ represents the Gaussian rendering function, $\hat{I}^p_t$ is the rendered image, $\epsilon_\theta(\cdot)$ is the predicted noise from Zero-1-to-3, and ϵ is the added noise. We take the superscript p to represent an arbitrary camera pose.

3.2 DreamScene4D

We propose a "*decompose-recompose*" principle to handle complex multi-object scenes. As in Figure 2, given a monocular video of multiple objects, we first segment and track [44, 19, 9, 10] each 2D object and recover the appearance of the occluded regions (Section 3.2.1). Next, we decompose the scene into multiple amodal objects and use SDS [38] with diffusion priors to obtain a 3D Gaussian representation for each object (Section 3.2.2). To handle large object motions, we optimize the deformation of 3D Gaussians under various constraints and factorize the motion into three components (Figure 3): the object-centric motion, an object-centric to world frame transformation, and the camera motion (Section 3.2.3). This greatly improves the stability and quality of the Gaussian

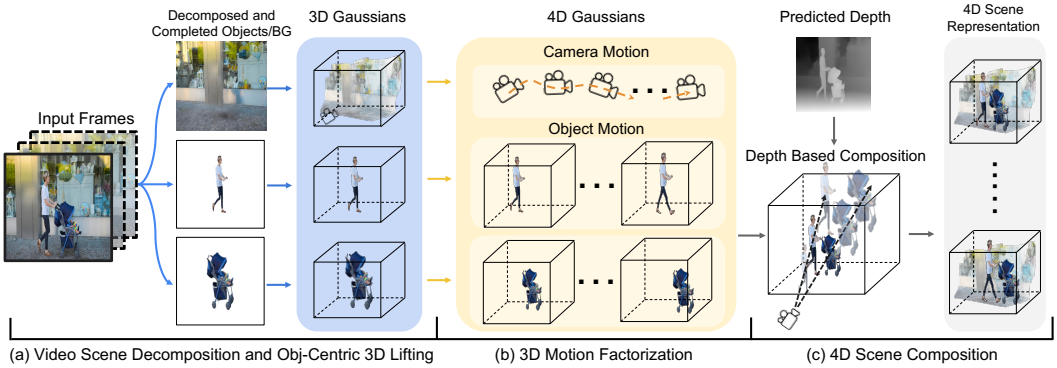

Figure 2: **Method overview for DreamScene4D**: **(a)** We first *decompose* and amodally complete each object and the background in the video sequence and use DreamGaussian [49] to obtain static 3D Gaussian representation. **(b)** Next, we factorize and optimize the motion of each object track independently, detailed in Figure 3. **(c)** Finally, we use the estimated monocular depth to *recompose* the independently optimized 4D Gaussians into one unified coordinate frame.

optimization and allows view-predictive image generative models to operate under in-distribution object-centric settings. Finally, we compose each individually optimized object to form a complete 4D scene representation using monocular depth guidance (Section 3.2.4).

3.2.1 Video Scene Decomposition

Instead of taking the video scene as a whole, we first adopt mask trackers [44, 19, 9, 10] to segment and track objects in the monocular video when GT object masks are not provided. From the monocular video and associated object tracks, we amodally complete each object track before lifting it to 3D as in Figure 2. To achieve zero-shot object appearance recovery for occluded regions of individual object tracks, we build off of inpainting diffusion models [45] and extend it to videos for amodal video completion. We provide the details of amodal video completion in the appendix.

3.2.2 Object-Centric 3D Lifting from World Frame

After decomposing the scene into individual object tracks, we use Gaussians Splatting [20] with SDS loss [38, 49] to lift them to 3D. Since novel-view generative models [26] trained on Objaverse [11] are inherently object-centric, we take a different manner to 3D lifting. Instead of directly using the first frame of the original video, where the object areas may be small and not centered, we create a new object-centric frame $\tilde{I}_1$ by cropping the object using its bounding box and re-scaling it. Then, we optimize the static 3D Gaussians with both the RGB rendering on $\tilde{I}_1$ and the SDS loss [38] in Eq. 1.

3.2.3 Modeling Complex 3D Motions via Motion Factorization

To estimate the motion of the first-frame lifted 3D Gaussians $\{G_1^{obj}\}$, one solution like DreamGaussian4D [43] is to model the object dynamics by optimizing the deformation of the 3D Gaussians directly in the world frame. However, this approach falls short in videos with large object motion, as the rendering loss yields minimal gradients until there is an overlap between the deformed Gaussians in the re-rendered frames and the objects in the video frames. Large motions of thousands of 3D Gaussians also increase the training difficulty of the lightweight deformation network [13, 6].

Thus, we propose to decompose the motion into three components and independently model them: **1)** object-centric motion, modeled using a learnable deformation network; **2)** the object-centric to world frame transformation, represented by a set of 3D displacements vectors and scaling factors; and **3)** camera motion, represented by a set of camera pose changes. Once optimized, the three components can be composed to form the object motion observed in the video.

Object-Centric Motion Optimization The deformation of the 3D Gaussians includes a set of learnable parameters for each Gaussian: **1)** a 3D position for each timestep $\mu_t = (\mu x_t, \mu y_t, \mu z_t)$, **2)** a 3D rotation for each timestep, represented by a quaternion $\mathcal{R}_t = (qw_t, qx_t, qy_t, qz_t)$, and **3)** a

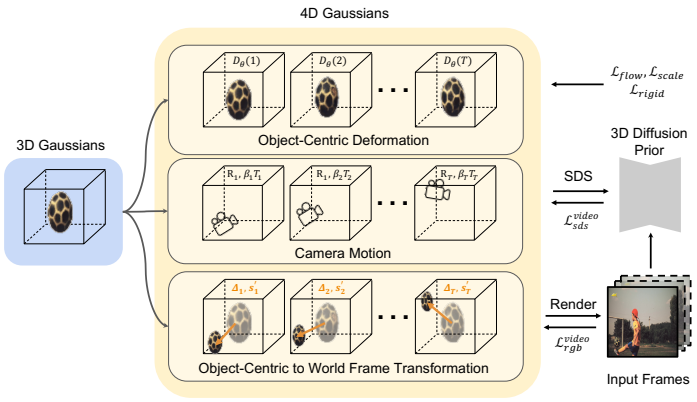

Figure 3: **3D Motion Factorization.** The 3D motion is decomposed into 3 components: 1) the object-centric deformation, 2) the camera motion, and 3) the object-centric to-world frame transformation. After optimization, they can be composed to form the original object motion observed in the video.

3D scale for each timestep $s_t = (sx_t, sy_t, sz_t)$. The RGB (spherical harmonics) and opacity of the Gaussians are shared across all timesteps and copied from the first-frame 3D Gaussians.

To compute the 3D object motion in the object-centric frames, we take the cropped and scaled objects in the individual frames I_t, forming a new set of frames $\tilde{I}_t^r$ for each object. Following DreamGaussian4D [43], we adopt a K-plane [13] based deformation network $D_\theta(G_1^{obj}, t)$ to predict the 10-D deformation parameters (μ_t, R_t, s_t) for each object per timestep. We denote the rendered image at timestep t under the camera pose p as $\hat{I}_t^p$, and optimize D_θ using the SDS loss in Eq. 1, as well as the rendering loss between $\hat{I}_t^r$ and $\tilde{I}_t^r$ for each frame under the reference camera pose r.

Since 3D Gaussians can freely move within uniformly-colored regions without penalties, the rendering and SDS loss are often insufficient for capturing accurate motion, especially for regions with near-uniform colors. Thus, we additionally introduce a flow rendering loss $\mathcal{L}_{flow}$, which is the masked L1 loss between the rendered optical flow of the Gaussians and the flow predicted by an off-the-shelf optical flow estimator [54]. The flow rendering loss only applies to the confident masked regions that pass a simple forward-backward flow consistency check.

Physical Prior on Object-Centric Motion Object motion in the real world follows a set of physics laws, which can be used to constrain the Gaussian deformations further. For example, objects usually maintain a similar size in temporally neighboring frames. Thus, we incorporate a scale regularization loss $\mathcal{L}_{\text{scale}} = \frac{1}{T}\sum_{t=1}^{T} \|s_{t+1} - s_t\|_1$, where we penalize large Gaussian scale changes.

To preserve the local rigidity during deformations, we apply a loss $\mathcal{L}_{\text{rigid}}$ to penalize changes to the relative 3D distance and orientation between neighboring Gaussians following [30]. We disallow pruning and densification of the Gaussians when optimizing for deformations like [43, 30].

Object-to-world Frame Transformation We compute the translation $\Delta_t = (\Delta_{x,t}, \Delta_{y,t}, \Delta_{z,t})$ and scaling factor s'_t that warps the Gaussians from the object-centric frame to the world frame. The 2D bounding-box-based cropping and scaling (Sec 3.2.2) from the original frames to the object-centric frames can be represented as an affine warp, which we use to compute and initialize $\Delta_{x,t}$, $\Delta_{y,t}$, and s'_t for each object in each frame. $\Delta_{z,t}$ is initialized to 0. We then adopt the rendering loss on the original frames I_t instead of center-cropped frames $\tilde{I}_t$ to fine-tune Δ_t with a low learning rate.

To further improve the alignment between renderings and the video frames, it is essential to consider the perceptual parallax difference. This arises when altering the object's 3D position while maintaining a fixed camera perspective, resulting in subtle changes in rendered object parts. Thus, we compose the individually optimized motion components and jointly fine-tune the deformation network D_θ and affine displacement Δ_t using the rendering loss. This refinement process, conducted over a limited number of iterations, helps mitigate the parallax effect as shown in Figure 11 of the appendix.

Camera Motion Estimation We leverage differentiable Gaussian Splatting rendering to jointly reconstruct the 3D static video background and estimate camera motions. Taking multi-frame inpainted background images I_t^{bg} as input, we first use an off-the-shelf algorithm [50] to initialize the background Gaussians and relative camera rotation and translation $\{R_t, T_t\}$ between frame 1 and frame t. However, the camera motion can only be estimated up to an unknown scale [60] as there is no metric depth usage. Therefore, we also estimate a scaling term β for T_t. Concretely, from

the background Gaussians G^{bg} and $\{R_t, T_t\}$, we find the β that minimizes the rendering loss of the background in subsequent frames:

$$\mathcal{L}_{\text{bg}} = \frac{1}{T}\sum_{t=1}^{T}\left\| I_t^{\text{bg}} - \phi\left(G^{bg},\, R_t,\, \beta_t T_t\right)\right\|_2, \tag{2}$$

Empirically, optimizing a separate β_t per frame [3] yields better results by allowing the renderer to compensate for erroneous camera pose predictions.

3.2.4 4D Scene Composition with Monocular Depth Guidance

Given the individually optimized 4D Gaussians, we recompose them into a unified coordinate frame to form a coherent 4D scene. As illustrated in Step (c) of Figure 2, this requires determining the depth and scale for each object along camera rays.

Concretely, we use an off-the-shelf depth estimator [56] to compute the depth of each object and the background and exploit the relative depth relationships to guide the composition. We randomly pick an object as the "reference" object and estimate the relative depth scale k between the reference object and all other objects. Then, the original positions μ_t' and scales s_t' of the 3D Gaussians for the objects are scaled along the camera rays given this initialized scaling factor k: $\mu_t' = \mathcal{C}^r - (\mathcal{C}^r - \mu_t) * k$ and $s_t' = s_t * k$, where $\mathcal{C}^r$ represents the position of the camera. Finally, we compose and render the depth map of the reference and scaled object, and minimize the affine-invariant L1 loss [56, 42] between the rendered and predicted depth map to optimize each object's scaling factor k:

$$\mathcal{L}_{depth} = \frac{1}{HW}\sum_{i=1}^{HW}\left\|\hat{d}_i^* - \hat{d}_i\right\|_1,\ \hat{d}_i = \frac{d_i - t(d)}{\sigma(d)}. \tag{3}$$

Here, $\hat{d}_i^*$ and $\hat{d}_i$ are the scaled and shifted versions of the rendered depth d_i^* and predicted depth d_i. $t(d)$ is defined as the reference object's median depth and $\sigma(d)$ is defined as the difference between the 90% and 10% quantile of the reference object. The two depth maps are normalized separately using their own $t(d)$ and $\sigma(d)$. Once we obtain the scaling factor k for each object, we can easily place and re-compose the individual objects in a common coordinate frame. The Gaussians can then be rendered jointly to form a scene-level 4D representation, as shown in Figure 5.

4 Experiments

Datasets While there exist datasets used in previous video-to-4d generation works [17], they only consist of a small number of single-object synthetic videos with small amounts of motion. Thus, we evaluate the performance of DreamScene4D on more challenging multi-object video datasets, including DAVIS [37], Kubric [15], and some self-captured videos with large object motion. We select a subset of 30 challenging real-world videos from DAVIS [37], consisting of multi-object monocular videos with various amounts of motion. We further incorporate the labeled point trajectories from TAP-Vid-DAVIS [12] to evaluate the accuracy of the learned Gaussian deformations. In addition, we generated 50 multi-object videos from the Kubric [15] simulator, which provides challenging scenarios where objects can be small or off-center with fast motion.

Evaluation Metrics The quality of 4D generation can be measured in two aspects: the view rendering quality of the generated 3D geometry of the scene, and the accuracy of the 3D motion. For the former, we follow previous works [17, 43] and report the CLIP [41] and LPIPS [63] scores between 4 novel-view rendered frames and the reference frame, and compute its average score per video. These metrics allow us to assess the semantic similarity between rendered and reference frames. We also conducted a user study to evaluate the 4D generation quality for the DAVIS videos using two-way voting to compare each baseline with our method, where 50% / 50% indicates equal preference.

The accuracy of the estimated motion can be evaluated by measuring the End Point Error (EPE) of the projected 3D trajectories. For Kubric, we report the mean EPE separately for fully visible points and points that undergo occlusion. For DAVIS, we report the mean and median EPE [65], as the annotations only exist for visible points.

Implementation Details We run our experiments on one 40GB A100 GPU. We crop and scale the individual objects to around 65% of the image size for object lifting. For static 3D Gaussian

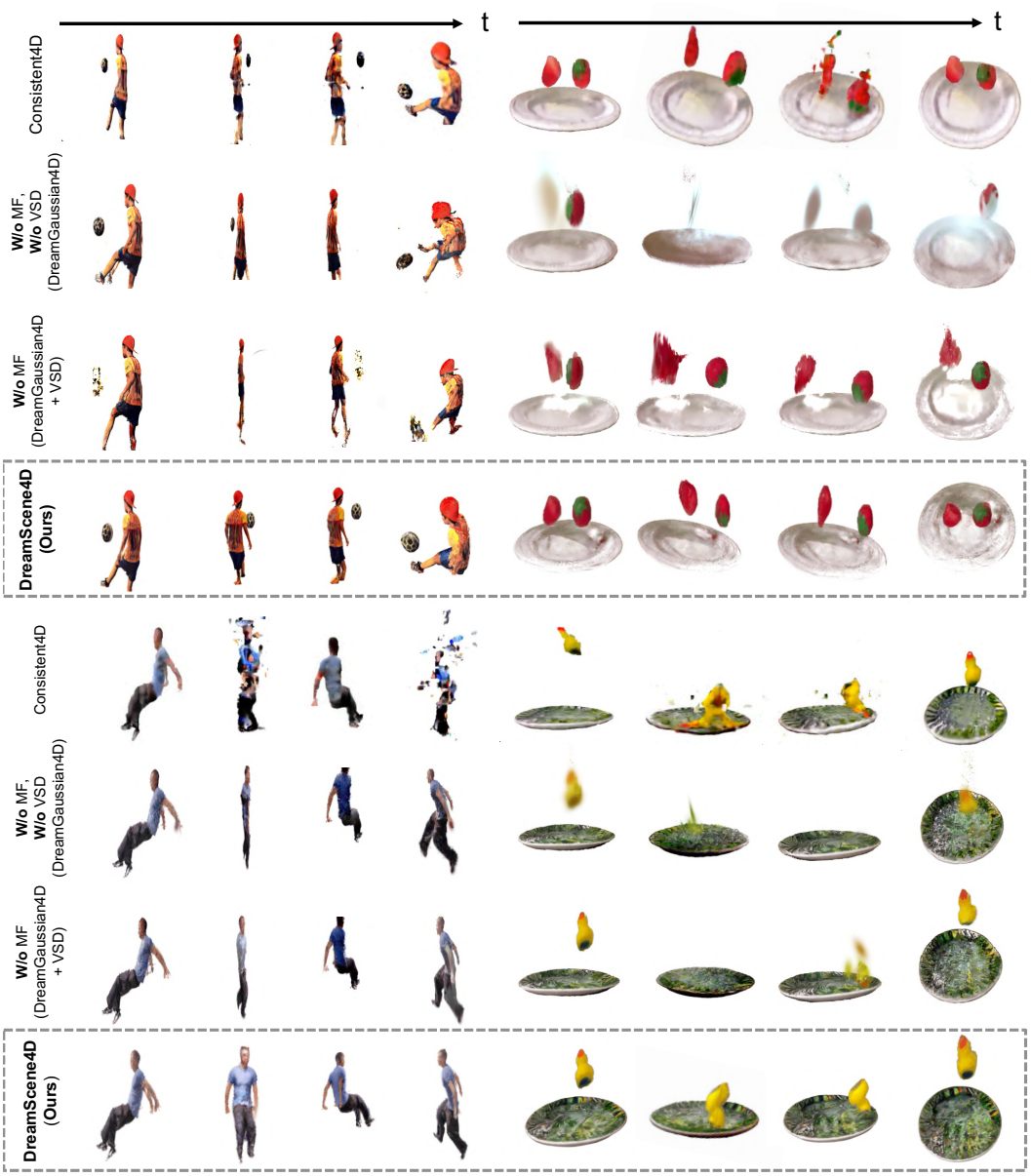

Figure 4: **Video to 4D Comparisons**. We render the Gaussians at various timesteps and camera views. We denote Motion Factorization as **MF** and Video Scene Decomposition as **VSD**. Our method produces consistent and faithful renders for fast-moving objects, while DreamGaussian4D [43] (2nd row) and Consistent4D [17] (1st row) produce distorted 3D geometry, blurring, or broken artifacts. Refer to our Supp. Materials for extensive qualitative comparisons.

optimization, we optimize for 1000 iterations with a batch size of 16. For optimizing the dynamic components, we optimize for 100 times the number of frames with a batch size of 10. More implementation and running time details are provided in the appendix.

4.1 Video to 4D Scene Generation

Baselines We consider the following baselines and ablated versions of our model:

(1) Consistent4D [17], a recent state-of-the-art method for 4D generation from monocular videos that fits dynamic NeRFs per video using rendering losses and score distillation.

(2) DreamGaussian4D [43], which uses dynamic 3D Gaussian Splatting like us for 4D generation from videos, but does not use any video decomposition or motion factorization as DreamScene4D. This is most related to our method.

Table 1: **Video to 4D Scene Generation Comparisons**. We report the CLIP and LPIPS scores in Kubric [15] and DAVIS [37]. For user preference, A% / B% denotes that A% of the users prefer the *baseline* while B% prefer *ours* in two-way voting. We denote methods with Video Scene Decomposition as **VSD** and methods with Motion Factorization as **MF**.

Method	VSD	MF	DAVIS			Kubric	
			CLIP ↑	LPIPS ↓	User Pref.	CLIP ↑	LPIPS ↓
Consistent4D [17]	-	-	82.14	**0.141**	28.3% / **71.7%**	80.46	0.117
DreamGaussian4D [43]	✗	✗	77.81	0.181	22.1% / **77.9%**	73.45	0.146
DreamGaussian4D **w/** VSD	✓	✗	81.39	0.169	30.4% / **69.6%**	79.83	0.122
DreamScene4D (**Ours**)	✓	✓	**85.09**	0.152	-	85.53	**0.112**
w/o $\mathcal{L}_{flow}$	✓	✓	84.94	0.152	-	**86.41**	0.113
w/o $\mathcal{L}_{rigid}$ and $\mathcal{L}_{scale}$	✓	✓	83.24	0.153	-	84.07	0.115

(3) DreamGaussian4D+VSD (Video Scene Decomposition). We augment DreamGaussian4D with VSD, where we segment every object before 4D lifting, and recompose them. The main difference between this stronger variant and our DreamScene4D is the lack of motion factorization.

(4) DreamScene4D ablations on losses. We also ablate without flow losses and regularization losses.

4D Generation Results on DAVIS & Kubric We present the 4D generation quality comparison in Table 1, where our proposed Video Scene Decomposition (VSD) and Motion Factorization (MF) schemes greatly improve the CLIP and LPIPS score compared to the input reference images. From the user study, we can also observe that DreamScene4D is generally preferred over each baseline. Compared to the baselines, these significant improvements are mainly due to our proposed motion factorization, which enables the SDS loss to perform in an object-centric manner while reducing the training difficulty for the lightweight Gaussian deformation network in predicting large object motions. We also show qualitative comparisons of 4D generation on multi-object videos and videos with large motion in Figure 4, where both variants of DreamGaussian4D [43] and Consistent4D [17] tend to produce distorted 3D geometry, faulty motion, or broken artifacts of objects. This highlights the applicability of DreamScene4D to handle real-world complex videos.

4D Generation Results on Self-Captured Videos We also captured some monocular videos with fast object motion using a smartphone to test the robustness of DreamScene4D, where objects can be off-center and are subject to motion blur. We present qualitative results of the rendered 4D Gaussians in the right half of Figure 4. Even under more casual video capturing settings with large motion blur, DreamScene4D can still provide temporally consistent 4D scene generation results while the baselines generate blurry results or contain broken artifacts of the objects.

4.2 4D Gaussian Motion Accuracy

Baselines and Ablations Design To evaluate the accuracy of the 4D Gaussian motion, we consider DreamGaussian4D [43] as the baseline, since extracting motion from NeRF-based methods [17] is highly non-trivial. In addition, we compare against PIPS++ [65] and CoTracker [18], two fully-supervised methods explicitly trained for point-tracking, serving as upper bounds for performance.

4D Motion Accuracy in Video Reference Views In Table 2, we tabulate the motion accuracy comparison, where DreamScene4D achieves significantly lower EPE than the baseline DreamGaussian4D on both the DAVIS and Kubric datasets. We noted that conventional baselines often fail when objects are positioned near the edges of the video frame or undergo large motion. Interestingly, the motion accuracy of DreamScene4D outperforms PIPS++ [65], despite never being trained on point tracking data, as in Figure 6. This is due to the strong object priors of DreamScene4D, as the Gaussians adhere to remaining on the same object it generates and their motion is often strongly correlated.

4D Motion Results on Generated Novel Views An advantage of representing the scene using 4D Gaussians is being able to obtain motion trajectories in arbitrary camera views, which we visualize in Figure 1 and Figure 8 in the appendix. DreamScene4D can both generate a 4D scene with consistent appearance across views and produce temporally coherent motion trajectories in novel views.

4.3 Reliance on External Depth Estimation

DreamScene4D uses the estimated monocular depth to infer the relative depth relationships/scales of each independently optimized 4D Gaussians and recompose them into one unified coordinate frame. To provide more insights into the robustness of depth estimation errors, we replaced the

Table 2: **Gaussian Motion Accuracy**. We report the EPE in Kubric [15] and DAVIS [37, 12]. We denote methods with our Video Scene Decomposition in column **VSD** and methods with 3D Motion Factorization in column **MF**. Note that CoTracker is trained on Kubric.

Method	VSD	MF	DAVIS		Kubric	
			Mean EPE ↓	Median EPE ↓	EPE (vis) ↓	EPE (occ) ↓
(a) ***Not trained on point tracking data***						
Baseline: DreamGaussian4D [43]	✗	✗	26.65	6.98	101.79	120.95
w/ VSD	✓	✗	20.95	6.72	85.27	92.42
DreamScene4D (**Ours**)	✓	✓	**8.56**	4.24	**14.30**	**18.31**
w/o $\mathcal{L}_{flow}$	✓	✓	10.91	**3.83**	18.54	24.51
w/o $\mathcal{L}_{rigid}$ and $\mathcal{L}_{scale}$	✓	✓	10.29	4.78	16.21	22.29
(b) ***Trained on point tracking data***						
PIPS++ [65]	-	-	19.61	5.36	16.72	29.65
CoTracker [18]	-	-	7.20	2.08	2.51	6.75

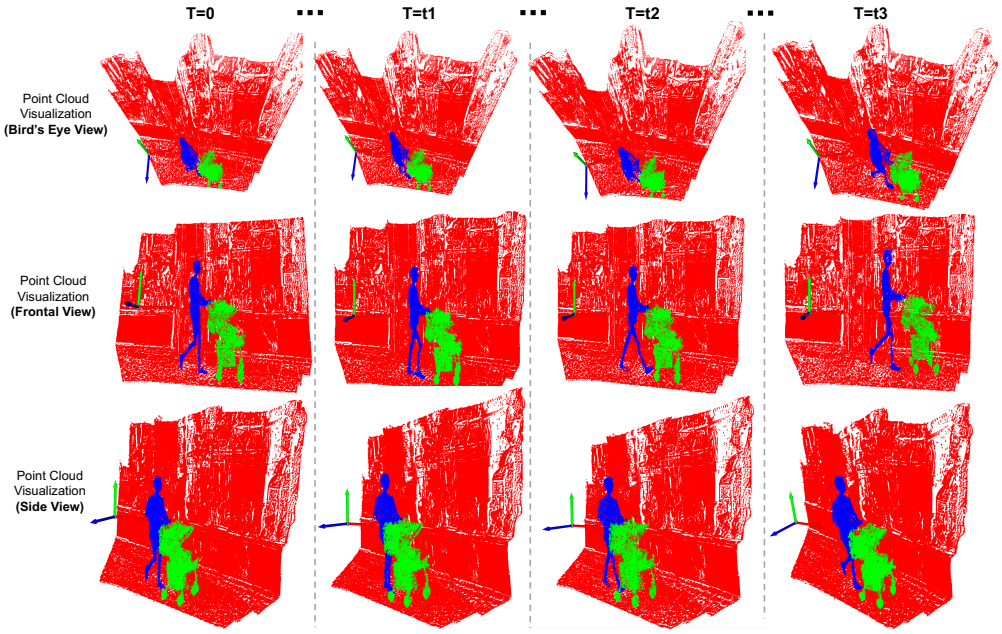

Figure 5: **Grouping visualization of Gaussians.** The grouping of the point cloud is visualized as colored point clouds from different camera views. The spatial relationships between objects are preserved after the composition.

Depth-Anything model with MiDAS [4], a weaker depth prediction model, as well as the newly released Depth-Anything v2 [57]. Furthermore, we added random noise to perturb the outputs of Depth-Anything v1 in various magnitudes.

Since depth estimation is only used for scene composition, we conducted experiments on multi-object DAVIS videos only to emphasize the differences and summarize the results in Table 3. While different depth predictions result in objects being placed in slightly different scene locations, we note that existing SOTA depth prediction models (such as Depth Anything series) meet the requirements in most cases, and the rendered quality of the 4D scene will not deteriorate much as long as the relative depth ordering of the objects is correct, which even holds when we add a small amount of noise to the predicted depth (second row) or use a weaker model like MiDAS (fourth row).

4.4 Limitations

Despite the exciting progress and results presented in the paper, several limitations still exist: **(1)** The SDS prior fails to generalize to videos captured from a camera with steep elevation angles. **(2)** Scene composition may fall into local suboptimas if the rendered depth of the lifted 3D objects is not well

Table 3: **Generation Results Using Different Depth Estimators.** We report the PSNR and LPIPS on multi-object DAVIS videos using different depth estimators.

Method	PSNR ↑	LPIPS ↓
Depth-Anything v1 (original)	83.71	0.169
Depth-Anything v1 + Noise (10%)	83.67	0.171
Depth-Anything v1 + Noise (25%)	83.48	0.174
MiDAS v3.1	83.34	0.176
Depth-Anything v2	83.76	0.169

Figure 6: **Motion Comparisons.** The 2D projected motion of Gaussians accurately aligns with dynamic human motion trajectory in the video, where the point trajectories estimated by PIPS++ [65] tend to get "stuck" in the background wall. For CoTracker [18], partial point trajectories are mixed up, where some points in the chest region (yellow/green) ending up in the head area (red).

aligned with the estimated depth. **(3)** Despite the inpainting, the Gaussians are still under-constrained when heavy occlusions happen, and artifacts may occur. **(4)** Our runtime scales linearly with the number of objects and can be slow for complex videos. Addressing these limitations by pursuing more data-driven ways for video to 4D generation is a direct avenue of our future work.

5 Conclusion

We presented DreamScene4D, the first video-to-4D scene generation work to generate dynamic 3D scenes across occlusions, large object motions, and unseen viewpoints with both temporal and spatial consistency from multi-object monocular videos. DreamScene4D relies on decomposing the video scene into the background and individual object trajectories, and factorizes object motion to facilitate its estimation through pixel and motion rendering, even under large object displacements. We tested DreamScene4D on popular video datasets like DAVIS, Kubric, and challenging self-captured videos. DreamScene4D infers not only accurate 3D point motion in the visible reference view but also provides robust motion tracks in synthesized novel views.

Acknowledgments and Disclosure of Funding

This research was supported by Toyota Research Institute.

References

[1] Sherwin Bahmani, Xian Liu, Yifan Wang, Ivan Skorokhodov, Victor Rong, Ziwei Liu, Xihui Liu, Jeong Joon Park, Sergey Tulyakov, Gordon Wetzstein, et al. Tc4d: Trajectory-conditioned text-to-4d generation. *arXiv preprint arXiv:2403.17920*, 2024.

[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.

[3] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *CVPR*, 2023.

[4] Reiner Birkl, Diana Wofk, and Matthias Müller. Midas v3. 1–a model zoo for robust monocular relative depth estimation. *arXiv preprint arXiv:2307.14460*, 2023.

[5] Marcel Büsching, Josef Bengtson, David Nilsson, and Mårten Björkman. Flowibr: Leveraging pre-training for efficient neural image-based rendering of dynamic scenes. *arXiv preprint arXiv:2309.05418*, 2023.

[6] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *CVPR*, 2023.

[7] Duygu Ceylan, Chun-Hao P Huang, and Niloy J Mitra. Pix2video: Video editing using image diffusion. In *ICCV*, 2023.

[8] Ya-Liang Chang, Zhe Yu Liu, Kuan-Ying Lee, and Winston Hsu. Learnable gated temporal shift module for deep video inpainting. *BMVC*, 2019.

[9] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *ECCV*, 2022.

[10] Wen-Hsuan Chu, Adam W Harley, Pavel Tokmakov, Achal Dave, Leonidas Guibas, and Katerina Fragkiadaki. Zero-shot open-vocabulary tracking with large pre-trained models. *ICRA*, 2024.

[11] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *CVPR*, 2023.

[12] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. In *NeurIPS*, 2022.

[13] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *CVPR*, 2023.

[14] Quankai Gao, Qiangeng Xu, Zhe Cao, Ben Mildenhall, Wenchao Ma, Le Chen, Danhang Tang, and Ulrich Neumann. Gaussianflow: Splatting gaussian dynamics for 4d content creation. *arXiv preprint arXiv:2403.12365*, 2024.

[15] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *CVPR*, 2022.

[16] Eric Heiden, Ziang Liu, Vibhav Vineet, Erwin Coumans, and Gaurav S Sukhatme. Inferring articulated rigid body dynamics from rgbd video. In *IROS*, 2022.

[17] Yanqin Jiang, Li Zhang, Jin Gao, Weimin Hu, and Yao Yao. Consistent4d: Consistent 360 {\deg} dynamic object generation from monocular video. In *ICLR*, 2024.

[18] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023.

[19] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. In *NeurIPS*, 2023.

[20] 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.

[21] 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 *ICCV*, 2023.

[22] Yuseung Lee, Kunho Kim, Hyunjin Kim, and Minhyuk Sung. Syncdiffusion: Coherent montage via synchronized joint diffusions. *NeurIPS*, 2023.

[23] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *CVPR*, 2022.

[24] Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. *arXiv preprint arXiv:2312.13763*, 2023.

[25] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020.

[26] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *ICCV*, 2023.

[27] 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 *CVPR*, 2023.

[28] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019.

[29] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *CVPR*, 2022.

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

[31] 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.

[32] Zijie Pan, Zeyu Yang, Xiatian Zhu, and Li Zhang. Fast dynamic 3d object generation from a single-view video. *arXiv preprint arXiv 2401.08742*, 2024.

[33] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *ICCV*, 2021.

[34] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 40(6), dec 2021.

[35] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *SIGGRAPH*, 2023.

[36] Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. Sfv: Reinforcement learning of physical skills from videos. *ACM Transactions On Graphics (TOG)*, 37(6):1–14, 2018.

[37] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017.

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

[39] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *CVPR*, 2021.

[40] 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.

[41] 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 *ICML*, 2021.

[42] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 44(3), 2022.

[43] 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.

[44] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.

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

[46] Uriel Singer, Shelly Sheynin, Adam Polyak, Oron Ashual, Iurii Makarov, Filippos Kokkinos, Naman Goyal, Andrea Vedaldi, Devi Parikh, Justin Johnson, et al. Text-to-4d dynamic scene generation. *arXiv preprint arXiv:2301.11280*, 2023.

[47] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ICLR*, 2021.

[48] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust large mask inpainting with fourier convolutions. In *WACV*, 2022.

[49] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *ICLR*, 2024.

[50] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024.

[51] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *CVPR*, 2024.

[52] Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Yufei Shi, 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 *ICCV*, 2023.

[53] Dejia Xu, Hanwen Liang, Neel P Bhatt, Hezhen Hu, Hanxue Liang, Konstantinos N Plataniotis, and Zhangyang Wang. Comp4d: Llm-guided compositional 4d scene generation. *arXiv preprint arXiv:2403.16993*, 2024.

[54] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *CVPR*, 2022.

[55] Ning Xu, Linjie Yang, Yuchen Fan, Jianchao Yang, Dingcheng Yue, Yuchen Liang, Brian Price, Scott Cohen, and Thomas Huang. Youtube-vos: Sequence-to-sequence video object segmentation. In *ECCV*, 2018.

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

[57] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *arXiv:2406.09414*, 2024.

[58] Qitong Yang, Mingtao Feng, Zijie Wu, Shijie Sun, Weisheng Dong, Yaonan Wang, and Ajmal Mian. Beyond skeletons: Integrative latent mapping for coherent 4d sequence generation. *arXiv preprint arXiv:2403.13238*, 2024.

[59] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023.

[60] Vickie Ye, Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. Decoupling human and camera motion from videos in the wild. In *CVPR*, 2023.

[61] Yuyang Yin, Dejia Xu, Zhangyang Wang, Yao Zhao, and Yunchao Wei. 4dgen: Grounded 4d content generation with spatial-temporal consistency. *arXiv preprint arXiv:2312.17225*, 2023.

[62] 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.

[63] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

[64] Yuyang Zhao, Zhiwen Yan, Enze Xie, Lanqing Hong, Zhenguo Li, and Gim Hee Lee. Animate124: Animating one image to 4d dynamic scene. *arXiv preprint arXiv:2311.14603*, 2023.

[65] Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *ICCV*, 2023.

[66] 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.

A Appendix / Supplemental Material

In the supplementary materials, we provide the details for video amodal completion, more implementation details of our DreamScene4D, and some qualitative and quantitative evaluations of the amodal completion. For more qualitative video-to-4D generation evaluations, we suggest looking at the videos in the website.

A.1 Video Amodal Completion

We build off SD-Inpaint [45] and adapt it for video amodal completion by making two modifications to the inference process without further fine-tuning.

Spatial-Temporal Self-Attention A common technique for extending Stable Diffusion-based models for video generation editing inflates the spatial self-attention layers to additionally attend across frames without changing the pre-trained weights [52, 21, 7, 40]. Similar to [7], we inject tokens from adjacent frames during self-attention to enhance inpainting consistency. Specifically, the self-attention operation can be denoted as:

$$Q = W_Q z_t,\ K = W_K \left[z_{t-1}, z_t, z_{t+1}\right],\ V = W_V \left[z_{t-1}, z_t, z_{t+1}\right], \tag{4}$$

where $[\cdot]$ represents concatenation, z_t is the latent representation of frame t, and W_Q, W_K, and W_V denote the (frozen) projection matrices that project inputs to queries, keys, and values.

Latent Consistency Guidance While inflating the self-attention layers allows the diffusion model to attend to and denoise multiple frames simultaneously, it does not ensure that the inpainted video frames are temporally consistent. To solve this issue, we take inspiration from previous works that perform test-time optimization while denoising for structured image editing [35] and panorama generation [22] and explicitly enforce the latents during denoising to be consistent.

Concretely, we follow a two-step process for each denoising step for noisy latent z^τ at denoising timestep τ to latent $z^{\tau-1}$. For each noisy latent z_t^τ at frame t, we compute the fully denoised latent z_t^0 and its corresponding image $\hat{I}_t$ directly in one step. To encourage the latents of multiple frames to become semantically similar, we freeze the network and only update z^τ:

$$\hat{z}^\tau = z^\tau - \eta \nabla_z \mathcal{L}_c, \tag{5}$$

where η determines the size of the gradient step and $\mathcal{L}_c$ is a similarity loss, i.e., CLIP feature loss or the SSIM between pairs of $\hat{I}_t$. After this latent optimization step, we take $\hat{z}^\tau$ and predict the added noise $\hat{\epsilon}^\tau$ using the diffusion model to compute $z^{\tau-1}$ as:

$$z^{\tau-1} = \sqrt{\alpha_{t-1}} \left(\frac{\hat{z}^\tau - \sqrt{1-\alpha_t}\hat{\epsilon}^\tau}{\sqrt{\alpha_t}} \right) + \sqrt{1-\alpha_{t-1}}\hat{\epsilon}^\tau, \tag{6}$$

where α_t is the noise scaling factor defined in DDIM [47].

A.2 More Implementation Details

Deformation Network. The deformation network uses a Hexplane [6] backbone representation with a 2-layer MLP head on top to predict the required outputs. In our evaluations, the resolution of the Hexplanes is $[64, 64, 64, 25]$ for (x, y, z, t) to ensure fair comparisons with the baselines. For longer videos (more than 32 frames), we set the resolution to $[64, 64, 64, 0.8T]$ for (x, y, z, t), where T is the number of frames. We found that the network is generally quite robust to the temporal resolution of the Hexplane grid.

Learning Rate. Following DreamGaussian [49] and DreamGaussian4D [43], we set different learning rates for different Gaussian parameters. We use the same set of hyperparameters as DreamGaussian and use a learning rate that decays from $1e^{-3}$ to $2e^{-5}$ for the position, a static learning rate of 0.01 for the spherical harmonics, 0.05 for the opacity, ad $5e^{-3}$ for the scale and rotation. The learning rate of the Hexplane grid is set to $6.4e^{-4}$ while the learning rate of the MLP prediction heads is set to $6.4e^{-3}$. During joint fine-tuning of the deformation network and the object-centric to world frame transformations, we set the learning rate to 0.1x the original value. We use the AdamW optimizer for all our optimization processes.

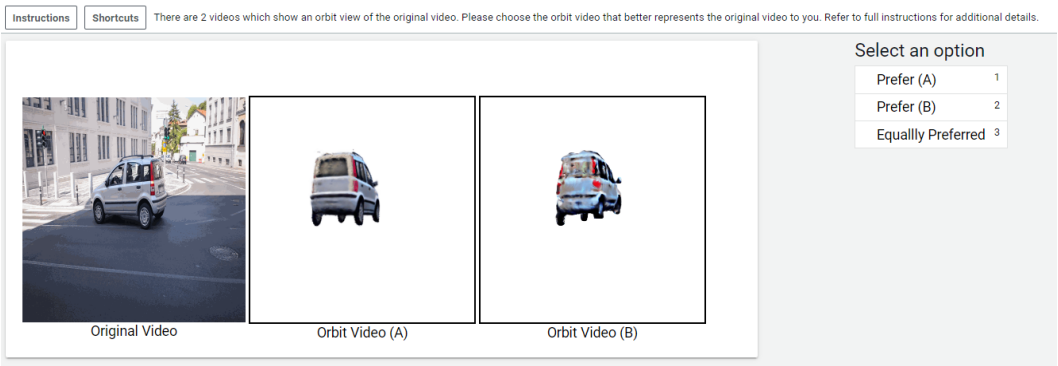

Figure 7: **User survey interface.** A GUI example of what an Amazon Turk worker would see as part of the user preference study.

Densification and Pruning. Following [49, 43], the densification in the image-to-3D step is applied for Gaussians with accumulated gradient larger than 0.5 and max scaling smaller than 0.05. Gaussians with an opacity value less than 0.01 or max scaling larger than 0.05 are also pruned. This is done every 100 optimization step. Densification and pruning are both disabled during motion optimization.

Running Time. As mentioned in the main text, we perform 1000 optimization steps for the static 3D Gaussian splatting process, while the deformation optimization takes $100 \cdot T$ optimization steps, where T is the number of frames. The joint fine-tuning process is conducted over 100 steps. While many videos converge faster, we found that videos with more complex objects and motion require more optimization steps. On a 40GB A100 GPU, the static 3D lifting process takes around 5.5 minutes, and the 4D lifting process takes around 17 minutes for a video of 16 frames per object. For comparisons with the baselines, please refer to Table 4.

Table 4: **Running time comparisons.** Since the running time of DreamScene4D scales w.r.t. the number of objects in the video, we present the results separately for videos with 1, 2, and 3 objects from DAVIS, denoted by the 3 entries in each field (1 obj/2 objs/3 objs).

Method	CLIP	LPIPS	Time (GPU hrs)	FPS (Hz)	Memory (GB)
Consistent4D	82.14	0.141	0.81hr	4.9	26.8
DreamGaussian4D	77.81	0.181	0.44hr	76.7	22.8
DreamScene4D	85.09	0.152	0.27hr/0.53hr/0.81hr	76.1/72.4/68.7	24.7

Evaluation Settings. In our video-to-4D evaluations, we render from the following combination of (elevation, azimuth) angles: (0, 45), (0, -45), (45, 0), (-45, 0). These novel view renders are then compared with the reference view at each timestep to obtain the CLIP and LPIPS scores. The scores are then averaged across all views and timesteps for the final score.

User Preference Study. For the user study, we take the 30 DAVIS videos and produce a smooth orbital render video by varying the azimuth angle while rendering the deforming object(s). We use Amazon Turk to outsource evaluations on the 30 DAVIS videos for each baseline, including DreamGaussian4D [43], DreamGaussian4D [43] + Video Scene Decomposition (VSD), Consistent4D [17] and our DreamScene4D. Each set of videos is reviewed by 30 workers with a HIT rate of over 95% for a total of 2700 answers collected. The whole user preference study takes about 97s per question and 72.8h working hours in total. We manually filtered out workers who submitted the same answer for all the videos and assigned new ones during the collection process until the desired number of answers had been collected.

The full instruction given is as follows:

> *Please read the instructions and check the videos carefully.*
>
> *There are 2 videos that show an orbit view of the original video. Please choose the orbiting video that looks more realistic and better represents the original video to*

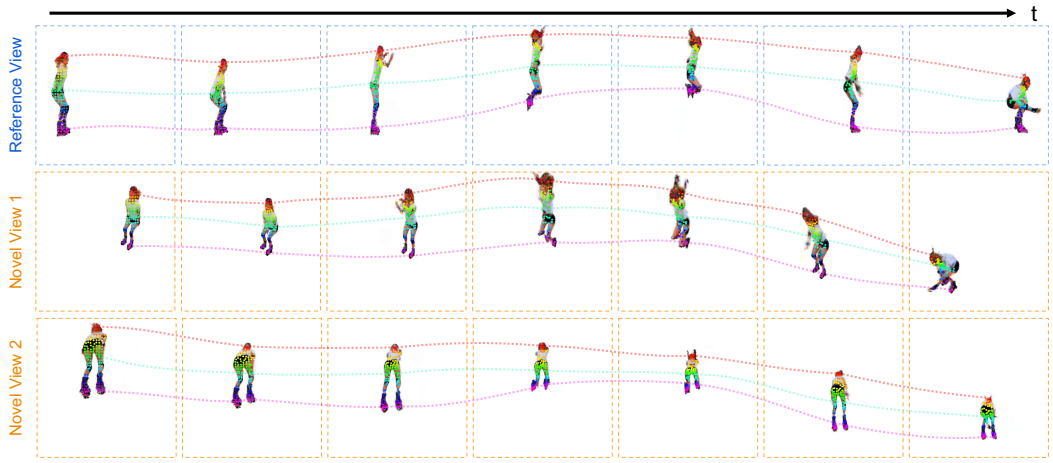

Figure 8: **Gaussian Motion Visualizations.** We visualize the Gaussian trajectories in the reference view corresponding to the video as well as in multiple novel views. The rendered Gaussians are sampled *independently* for each view. DreamScene4D can produce accurate motion in different camera poses **w/o** explicit point trajectory supervision.

> *you. The options (A) and (B) correspond to the two given orbit videos. If you think that both are of the same quality, please select Equally Preferred.*
>
> *To judge the quality of the videos, consider the following points:*
>
> *1. Do the objects in the orbit video correspond to the original video?*
> *2. Does the video look geometrically correct (e.g. not overly flat) when the camera is orbiting?*
> *3. Are there any visual artifacts (e.g. floaters, weird textures) during the orbit?*
>
> ***Please ignore the background in the original video.***

A GUI sample of a survey question is also provided in Figure 7 for reference.

A.3 Additional Results

4D Motion Visualizations in Novel Views Since DreamScene4D represents the scene using 4D Gaussians, it is able to obtain motion trajectories in arbitrary camera views, as in Figure 8. DreamScene4D can both generate a 4D scene with consistent appearance across views and produce temporally coherent motion trajectories.

Video Amodal Completion To ablate our extensions to SD-Inpaint for video amodal completion, we randomly select 120 videos from YoutubeVOS [55] and generate random occlusion masks in the video [48, 8]. We compare against Repaint [29] and SD-Inpaint [45] for video amodal completion. Both baseline methods are based on Stable Diffusion [45]. Repaint alters the reverse diffusion iterations by sampling the unmasked regions of the image. SD-Inpaint, on the other hand, finetunes Stable Diffusion for free-form inpainting. We also ablate the performance of our proposed amodal completion approach without the inflated spatiotemporal self-attention (denoted as STSA) and consistency guidance. We summarize the results in Table 5 and show some visual comparisons in Figure 9. Our modification achieves more consistent and accurate video completion than image inpainting approaches by leveraging temporal information during the denoising process. Note that these techniques complement other video completion approaches since DreamScene4D mainly focuses on video-to-4D scene generation.

Mitigating Parallax Effects via Joint Optimization We show an example of the rendered Gaussians before and after performing the joint optimization for the deformation network and the object-centric to world frame transformations in Figure 11. We can see that a small amount of joint fine-tuning steps helps alleviate the parallax effect and better aligns the rendered Gaussians to the input video frames.

Figure 9: **Video Amodal Completion Comparisons.** Spatiotemporal self-attention and Consistency Guidance both help to preserve the identity consistency of the inpainted objects.

Table 5: **Video Amodal Completion Evaluations.** We report the PSNR, LPIPS, and Temporal Consistency (TC) measured using CLIP similarity in randomly masked YoutubeVOS [55] videos.

Method	PSNR ↑	PSNR ↑ (masked)	LPIPS ↓	TC ↑
Repaint [29]	20.76	14.04	0.23	91.18
SD-Inpaint [45]	21.07	14.35	0.23	91.72
DreamScene4D (**Ours**)	**22.27**	**16.09**	**0.22**	**93.40**
w/o STSA	21.56	15.31	0.23	92.58
w/o Guidance	21.71	15.20	0.23	92.91

A.4 Failure Cases

We additionally show some failure cases corresponding to the limitations documented in the main text in Figure 10. Based on our observations, the inpainting is very unstable during heavy occlusions. We believe that instead of solely relying on rendering losses for the occluded regions, incorporating some form of semantic guidance loss (e.g. CLIP feature loss) might be a promising direction.

A.5 Broader Impact

Our approach is deeply connected to VR/AR applications and can potentially provide 3D meshes and dense 3D trajectories for robot manipulation. While our method does not generate or modify the original video, it is still possible for users to use generative models with malicious intent, and then apply our approach for video-to-4D lifting. The potential negative impact can be avoided by applying preventative measures in generative models and rejecting the video input if violations are found.

A.6 DAVIS Split

We list the DAVIS video names that were used to perform evaluations:

```
bear,blackswan,bmx-bumps,boxing-fisheye,car-shadow,cows,crossing,
```

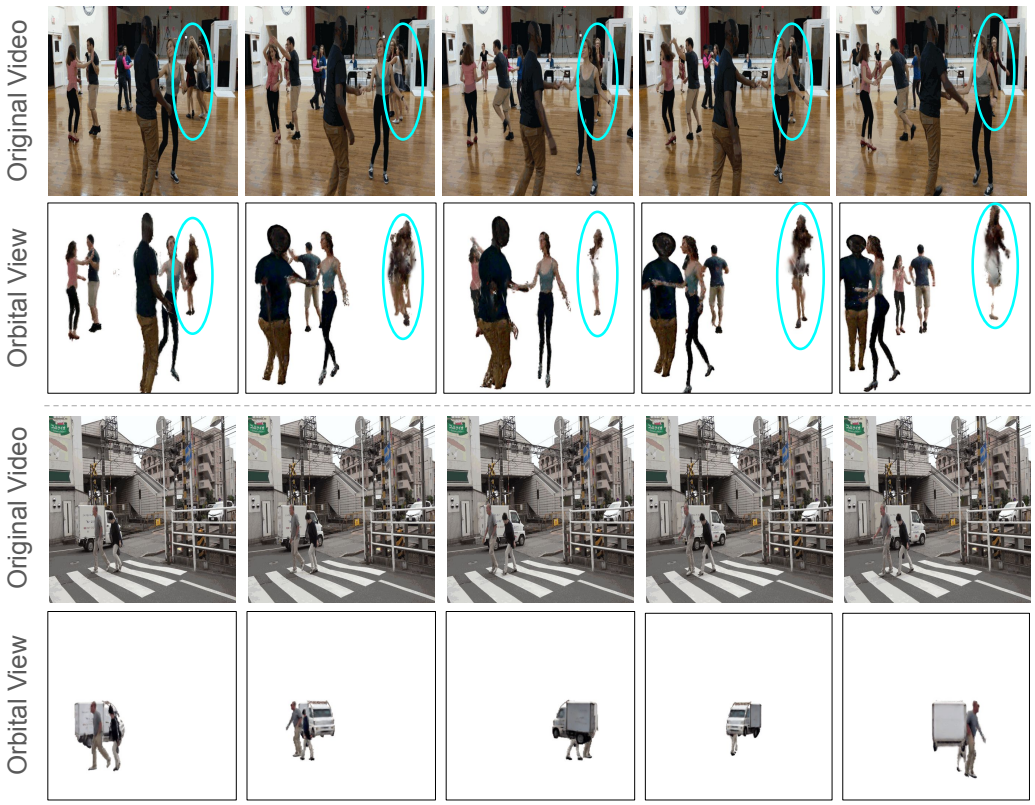

Figure 10: **Failure Cases.** We show 2 representative failure cases. The first case (top 2 rows) is due to inpainting failures (circled in blue), where the inpainted frames are not of high quality, leading to flickering objects when rendered. The second case (bottom 2 rows) arises from poor depth predictions, which leads to composition errors. The two humans are placed too close to the truck, making the scale proportions of the objects seem unnatural (i.e. the truck is too small).

Figure 11: **Mitigating the parallax effect.** A small amount of joint fine-tuning steps can help mitigate the parallax effect and align the rendered Gaussians to the input video frames.

`dance-twirl,dancing,dog-gooses,dogs-jump,gold-fish,hike,hockey,kid-football,`
`lab-coat,lindy-hop,longboard,lucia,night-race,parkour,pigs,rallye,rhino,`
`rollerblade,schoolgirls,scooter-black,scooter-gray,snowboard,stroller,train`

For bmx-bumps, longboard, scooter-black, and scooter-gray, we merge the mask of the human and the other objects into one as they move together for the entire video (e.g. person riding a bike or a scooter).

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our abstract and introduction clearly document the scope and contributions of our paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Please refer to the limitations section in the main text. We also show some failure cases corresponding to these limitations in the supplementary materials.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification: We do not claim or present theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We discuss the details of the method and evaluation in the experiments section in the main text and the supplementary materials. The code will also be made public in the future.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We do not include our code in the submission. However, it will be made public upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Please refer to the experiments section and the supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [NA]

   Justification: Does not apply to our evaluations.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Please refer to the implementation details subsection in the main text and supplementary materials.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have carefully reviewed the Code of Ethics and confirm that the research conforms to it.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Please refer to the supplementary materials.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Does not apply, since our paper does not generate new things, but instead converts existing videos to 4D representations.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Our code is written by ourselves, while the data comes from existing datasets or simulators, which are cited in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not introduce new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [Yes]

    Justification: We include the details of our human evaluations in the supplementary materials.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: Does not apply to the paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.