

Dynamic World Generation Made Efficient

Fengrui Tian¹, Jinqi Luo¹, Uday Kiran Reddy Tadipatri¹, Hancheng Min², and René Vidal¹

¹ University of Pennsylvania

² Shanghai Jiao Tong University
tianfr@upenn.edu

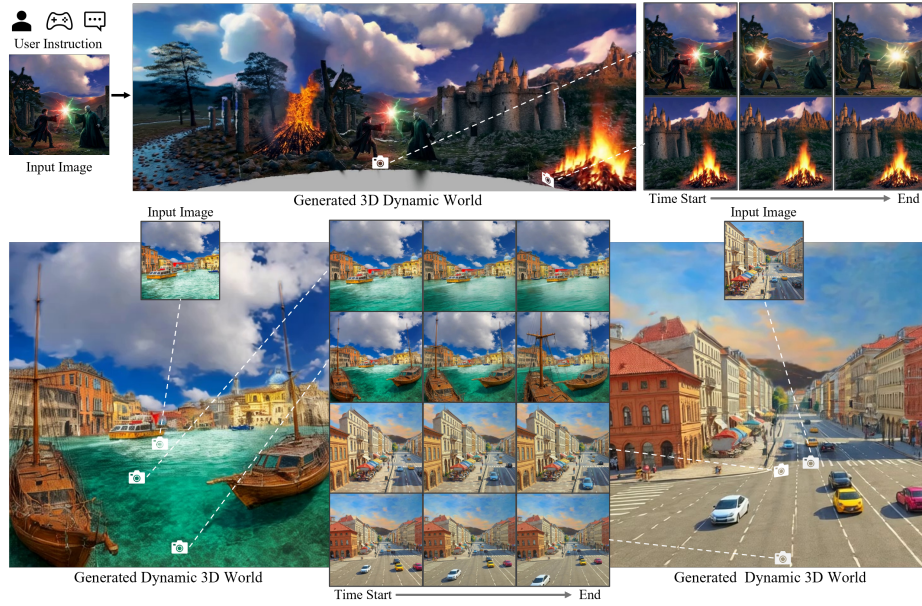


Fig. 1: Given an input image and user instructions (*e.g.*, world descriptions and camera paths), DYNWORLD generates multiple spatially interconnected 3D dynamic scenes including diverse object motions (*e.g.*, wizards casting magic, moving cars, and sailing boats) and texture dynamics (*e.g.*, beating flames and floating clouds) in 15-30 minutes.

Abstract. Generating large dynamic 3D worlds is important for applications such as virtual reality, simulation, and robotics. However, the lack of large-scale 3D dynamic training data prevents training world-scale 3D generative models directly. Recent progress in video generation provides powerful priors for generating 3D dynamic visual content. As a result, current approaches typically generate videos from multiple camera views using image or video diffusion models and fuse them into a 3D representation. However, these models generate each view independently, making

it difficult to maintain motion consistency across viewpoints. Existing solutions address this problem by sequentially generating each view while updating an explicit 3D representation, which introduces a significant computational bottleneck. In this work we propose DYNWORLD, a framework that enables parallel generation of motion-consistent videos across multiple views. Our key insight is that the latent tokens of transformer-based video generators implicitly encode spatial and dynamic information that can be geometrically related across cameras. We thus introduce a *Spatial Token Memory* that stores latent video tokens along with their estimated 3D positions. During generation, tokens from one view are back-projected into 3D and re-projected into other views, allowing multiple video generators to share consistent motion context through attention. This mechanism enforces cross-view motion consistency while allowing all views to be generated simultaneously. Experiments show that our method produces large, diverse, view-consistent dynamic worlds while reducing generation time by an order of magnitude compared to sequential pipelines.

1 Introduction

Generating large dynamic 3D worlds is a long-standing goal in computer vision and graphics with applications in virtual and augmented reality, simulation, and robotics. Such worlds must contain not only complex geometry and appearance, but also consistent dynamics across viewpoints, allowing users to freely navigate the environment while observing coherent object motion and scene evolution. However, training generative models that directly produce dynamic 3D worlds remains challenging due to the scarcity of large-scale 3D dynamic datasets.

As a consequence, the latest approaches to dynamic 3D world generation [4, 7, 18, 50, 60, 64, 67, 68] leverage recent advances in video generation [3, 19, 55, 66] to synthesize multiple videos from different viewpoints and subsequently fuse them into a 3D representation. However, this paradigm introduces the fundamental challenge of *multi-view motion consistency*: modern video generators typically synthesize each video independently, without enforcing geometric correspondence between viewpoints. As a result, the same object may exhibit inconsistent motion or appearance when observed from different cameras. To mitigate this problem, existing world-generation pipelines [4, 7, 50, 67, 68] enforce consistency through sequential generation. They generate a video from one viewpoint, update an explicit 3D representation of the scene (e.g., Gaussian Splattings [26] or point clouds), and then use this representation to guide generation in the next view (e.g., through video outpainting [23, 50, 56]). While effective, this sequential dependency creates a severe computational bottleneck when generating large dynamic environments. For instance, recent methods [50] require several hours to synthesize only a small number of views.

In this work we introduce DYNWORLD, a dynamic world generation framework for parallel generation of multiple consistent views. Our key observation is that latent tokens in transformer-based video generation models implicitly encode spatial and motion information. Thus, by augmenting each token with its

estimated 3D location and establishing geometric correspondences across views, we can generate multiple views in parallel while maintaining cross-view motion consistency. More specifically, given a single image, DYNWORLD first constructs an initial 3D scene from that image and expands the scene to a static 3D world by outpainting unseen regions of the scene. It then samples multiple camera view-points and generates videos for all views in parallel. To enhance view-consistent motion generation, we propose a Spatial Token Memory that stores video tokens with their estimated 3D positions. The memory is updated by back-projecting video tokens from one view with their estimated token depths during video generation, and is further injected into the attention layers of video generator in other views to share view-consistent context. Finally, the generated videos are fused into a unified dynamic 3D representation. Our experiments (see examples in Figure 1) demonstrate that DYNWORLD produces view-consistent dynamic worlds with diverse moving objects (e.g., moving cars, pedestrians, and boats) and detailed dynamic textures (e.g., drifting clouds, butterflies, and crashing waves), while reducing generation time by more than an order of magnitude compared with sequential generation pipelines. In short, the main contributions of this work are:

- We propose a new framework for *parallel generation of view-consistent videos across multiple viewpoints*, addressing the efficiency bottleneck of existing world generation pipelines that rely on sequential view synthesis.
- We introduce a *Spatial Token Memory* that stores video generation tokens with estimated 3D positions. This allows multiple views to exchange view-consistent motion context during generation by injecting re-projected tokens into the attention layers of transformer-based video generators.
- We show that DYNWORLD enables efficient creation of large dynamic worlds from a single image, achieving *an order-of-magnitude speedup* compared to sequential generation methods while maintaining strong motion consistency.

2 Related Work

Creating high-fidelity 3D worlds is a core problem in computer vision and graphics. Early approaches in the 2000s tackled this problem by either employing procedural graphics engines [43], or reconstructing large-scale environments from Internet images [1, 24]. Recently, researchers have begun building 3D worlds by leveraging powerful 2D and video generative models [3, 28, 32, 48, 59]. Such methods can be categorized into two broad families: (1) outpainting-based approaches that incrementally extrapolate novel views from existing ones, and (2) stitching-based techniques that create a large world by aligning overlapping regions across multiple views. Next, we discuss these approaches and their 3D representations.

World generation via outpainting. Recent research in 3D generation has shifted significantly from object-level generation [5, 14, 31, 39, 40, 44, 45, 49, 54, 62, 69] toward scene-level and even world-scale generation [11–13, 50, 60, 61, 63]. Compared to earlier attempts [35, 37] that only built perpetual textures, recent work such as WonderJourney [68] and WonderWorld [67] successfully constructs

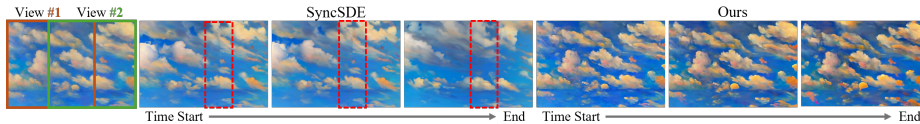


Fig. 2: Visual comparisons of our method against state-of-the-art denoising score stitching method SyncSDE [29] on the task of sky dynamics generation. By employing Spatial Token Memory, DYNWORLD achieves smooth motion consistency across two views, whereas SyncSDE is prone to producing visible seams near the view boundaries.

large static 3D worlds with diverse content. Building upon this progress, the community has begun exploring dynamic world generation [11, 50] by leveraging powerful video generation models [3, 53, 66]. A common paradigm is to use outpainting techniques [23, 56] to generate each view of the world autoregressively: before generating the next view, the system waits for the complete outpainting result of the current view. While such a sequential pipeline is reasonable for static world generation, it introduces a substantial efficiency bottleneck when synthesizing large dynamic worlds. For example, DynamicVoyager [50] requires 3–6 hours to sequentially generate a dynamic world with multiple views, which is impractical for downstream applications in AR/VR and game development. Moreover, outpainting-based models typically rely only on the visual context observed in the current view to enforce multi-view consistency, which often fails to maintain coherent generation across non-overlapping but spatially connected views (Figure 3). In contrast, DYNWORLD efficiently generates dynamic worlds by injecting geometric information into video tokens, enabling parallel generation across multiple views while maintaining cross-view motion consistency.

World generation via stitching. Recent works [27, 30, 46, 65] enforces multi-view consistency by synchronizing diffusion processes across views, typically by coupling denoising scores in overlapping regions. Compared with these methods, DYNWORLD achieves multi-view consistency by employing *spatial video tokens*, which offers two main advantages. First, noise blending can perturb the denoising trajectory [17], which may degrade synthesis quality and cross-view consistency (see Figure 2). In contrast, DYNWORLD blends the video tokens of the current generation process with the spatial tokens from historical views. By leveraging an attention mechanism, this approach ensures consistent generation without quality degradation. Second, noise blending can enforce consistency only in overlapping regions. DYNWORLD instead propagates spatial information through token-level interactions, allowing consistent generation even between views that do not share overlapping image regions (see Figure 3b).

3D representations in world generation. Recent world generation methods [50, 67, 68] rely on explicit 3D representations (*e.g.*, Gaussian Splatting [26], point clouds, or Neural Radiance Fields [42, 51, 52]) to store scene geometry. The motivation behind explicit representations is that they directly encode spatial relationships in the underlying 3D geometry. Meanwhile, recent video genera-

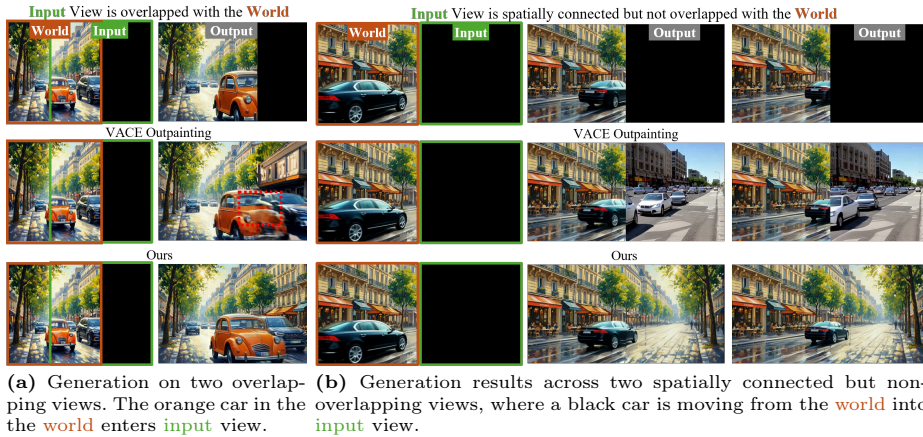


Fig. 3: Comparison with the state-of-the-art outpainting model VACE [23] on motion consistency. DYNWORLD maintains consistent object motions across views (overlapping in Figure 3a and non-overlapping in Figure 3b) by propagating context through the proposed Spatial Token Memory, while VACE often produces inconsistent motion or visible artifacts, especially when the target view lacks direct overlap.

tion models have demonstrated a strong ability to capture spatial relationships among tokens [34, 47, 58], suggesting that similar mechanisms could be used to represent 3D structure. Building on this observation, we investigate how geometric structure can be encoded within implicit token-based representations. Specifically, we augment video tokens with estimated 3D positions and use them to propagate geometric information across views, enabling consistent dynamic generation.

3 DYNWORLD

In this section, we present DYNWORLD, our approach for efficient dynamic world generation. As discussed in §1, our key insight is that the latent tokens of transformer-based video generators implicitly encode spatial and dynamic information that can be geometrically related across cameras. Building on this insight, DYNWORLD generates a large dynamic world through three stages:

- *Static World Construction* (§3.1): Given a single image, a world description, and a camera path, we build an initial 3D world with Gaussian Splatting and iteratively outpaint the scene along the camera path to construct a complete static 3D world.
- *Video Generation with Spatial Token Memory* (§3.2): Given multiple views sampled from the static world, we generate view-consistent videos from each view in parallel using a transformer-based flow-matching model. We maintain a Spatial Token Memory to store key-value video tokens together with their estimated 3D positions. During generation, tokens from one view are projected

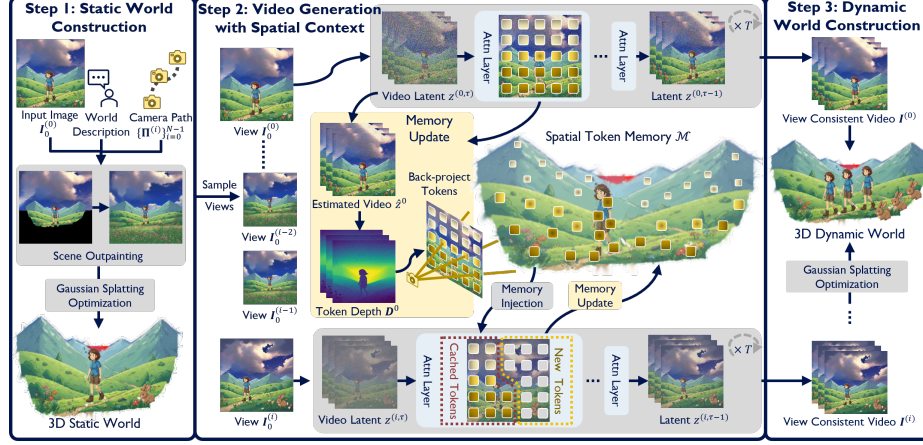


Fig. 4: DYNWORLD overview. We first build a static world from an input image, a world description, and a camera path (§3.1). Then given multiple views sampled from the static world, we generate videos from these views simultaneously. To maintain multi-view motion consistency in these videos, we propose a Spatial Token Memory that stores back-projected video tokens with their 3D positions and injects them into the video generation process (§3.2). We finally build a dynamic world from the generated videos with 3D Gaussian representations (§3.3).

into other views via their 3D positions and injected into the attention layers, enabling different views to share spatial and dynamic context.

- *Dynamic World Construction* (§3.3): We segment the generated videos into static backgrounds and dynamic foregrounds, reconstruct 3D Gaussians for each separately, and aggregate them into a unified dynamic 3D representation.

3.1 Static World Construction

Given an input image, a world description, and a camera path $\{\Pi^{(i)}\}_{i=0}^{N-1}$, where the superscript (i) denotes variables associated with the i^{th} camera pose $\Pi^{(i)}$, for $i = 0, \dots, N-1$, we begin by describing how we initialize the static 3D world.

Given an initial input RGB image $I_0^{(0)} \in \mathbb{R}^{H \times W \times 3}$, where H, W are the height and width of the image, respectively, we estimate a depth map and a normal map using 2D depth and surface normal estimation models [25]. We then construct a 3D Gaussian representation [26] at pose $\Pi^{(0)}$ from the input image and its estimated depth and normal map by instantiating one Gaussian kernel for each pixel on the image. Specifically, each Gaussian kernel is initialized with a set of parameters $G = \{\mathbf{x}, \mathbf{R}, \mathbf{s}, o, \mathbf{c}\}$. Here, $\mathbf{x} \in \mathbb{R}^3$ is the 3D position initialized by back-projecting the 2D pixel coordinates using the estimated depth; $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ represents the orientation, whose third column is initialized as the estimated surface normal, with the first two columns chosen to be orthogonal to it and to each other [67]; $\mathbf{s} = [s_x, s_y, \epsilon]^\top$ is the scale vector and $\epsilon \ll \min(s_x, s_y)$

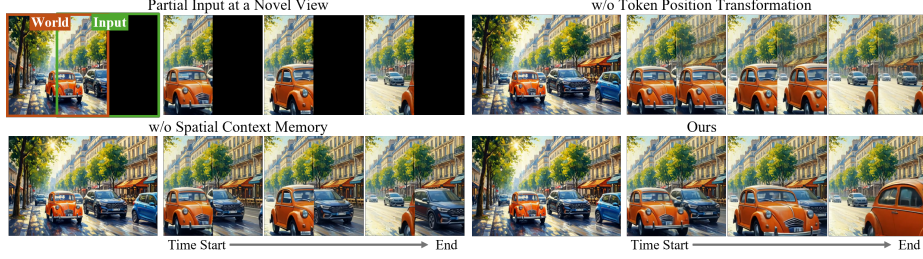


Fig. 5: Ablation study on the proposed Spatial Token Memory. The visualization shows that the proposed memory and token position transformations help maintain the view consistency in DYNWORLD.

is a minimal value that enforces a slight thickness along the z-axis; $\mathbf{c} \in [0, 255]^3$ is the color directly sampled from the image pixel, and $o \in [0, 1]$ denotes the opacity. The collection of all Gaussian kernels yields an initial 3D representation of the static world. This initial representation is then optimized with the input image $\mathbf{I}^{(0)}$ using the typical losses of $\mathcal{L}_1$ and SSIM [57, 67].

To expand the scene, we move to the next camera pose $\Pi^{(1)}$ and rasterize the current 3D Gaussians to render an image. Since the novel view exposes unseen regions, this rendering is partially incomplete. We outpaint the missing regions with an image outpainting model [28] and the world description to obtain a complete image $\mathbf{I}_0^{(1)}$. We then estimate the depth and normal maps for $\mathbf{I}_0^{(1)}$ and subsequently initialize new Gaussian kernels in the outpainted regions following a procedure similar to that for the initial view $\Pi^{(0)}$. We merge the newly initialized Gaussian kernels into the initial Gaussian representation and optimize them in the novel view with the outpainted image by using the $\mathcal{L}_1$ and SSIM losses. We repeat this scene outpainting process to create a static 3D world that is complete along the camera path. During the static world generation, we also obtain multi-view images of the world $\{\mathbf{I}_0^{(i)}\}_{i=0}^{N-1}$ that will be leveraged to animate the world, as described next.

3.2 Video Generation with Spatial Context

Our objective is to animate the constructed static world by generating multi-view consistent videos from the sampled multi-view images $\{\mathbf{I}_0^{(i)}\}_{i=0}^{N-1}$. Given these images with motion prompts, where each prompt is prepended with the phrase “Camera is strictly fixed” to ensure the generated videos are constrained to each fixed viewpoint, we leverage transformer-based flow matching models [48, 70] to simultaneously generate fixed-viewpoint videos at all views.

If these multi-view videos are generated independently, motion inconsistency arises across views, as illustrated in the bottom left of Figure 5. The question is how to coordinate these view-specific video generation processes so that the generated videos are consistent with the same 3D dynamics. As motivated in §1,

our key observation is that the key-value (KV) tokens in the attention layers of transformer-based video generators implicitly encode spatial and dynamic information [20, 71, 72]. If these tokens can be geometrically related across cameras, they can serve as a shared representation of the underlying 3D dynamics. A naive approach is to directly cache KV tokens from one view and inject them into another. However, this produces videos whose dynamics are *identical yet geometrically inconsistent* with those of the source view, as shown in the top right of Figure 5. The reason is that the 2D spatial positions of the cached KV tokens no longer correspond to the correct locations in the novel view due to the change of perspective. To resolve this, we propose a Spatial Token Memory that stores KV tokens together with their estimated 3D positions, enabling us to compute the correct 2D token positions in any novel view via camera projection and inject the tokens at their geometrically transformed locations.

Spatial token memory. We formally define the Spatial Token Memory as $\mathcal{M} = \{(\mathbf{k}^{(\cdot, \tau)}, \mathbf{v}^{(\cdot, \tau)}, \mathbf{p}_{3D}^{(\cdot, \tau)})\}_{\tau=0}^{\Gamma}$, where $\tau \in \{0, 1, \dots, \Gamma\}$ indexes the denoising step of the flow-matching model and Γ is the total number of denoising steps. Here, $\mathbf{k}^{(\cdot, \tau)}$ and $\mathbf{v}^{(\cdot, \tau)} \in \mathbb{R}^{M \times d}$ denote the key and value token matrices, respectively, each with d channels per token and a variable length M that grows as more KV tokens are merged into the memory, while $\mathbf{p}_{3D}^{(\cdot, \tau)} \in \mathbb{R}^{M \times 3}$ represents their corresponding 3D positions. Since these KV tokens and positions are shared across all views, we use a placeholder $(\cdot)$ for the view index. During the generation process of each view, we apply the following strategy to each attention layer: we first use the memory $\mathcal{M}$ to compute the attention output (*cross-view token sharing*), then merge the KV tokens derived from the current layer into the memory (*memory update*). Next, we describe each of these two operations. We omit the attention layer index in the notation $\mathbf{k}^{(\cdot, \tau)}$ and $\mathbf{v}^{(\cdot, \tau)}$ for clarity.

Cross-view token sharing. We inject the constructed Spatial Token Memory $\mathcal{M}$ into the video generation process for all views. Consider view $\mathbf{\Pi}^{(i)}$ as an example, when we calculate a certain attention layer at the denoising step τ , we retrieve the KV tokens and their 3D positions $(\mathbf{k}^{(\cdot, \tau)}, \mathbf{v}^{(\cdot, \tau)}, \mathbf{p}_{3D}^{(\cdot, \tau)})$ from the memory $\mathcal{M}$. Again, we omit the layer index in the notation. We project the 3D token positions into the current view using the camera matrix $\mathbf{\Pi}^{(i)}$,

$$\mathbf{p}_{2D}^{(i, \tau, *)} = \varphi(\mathbf{p}_{3D}^{(\cdot, \tau)}, \mathbf{\Pi}^{(i)}), \quad (1)$$

where φ is the camera projection function and $\mathbf{p}_{2D}^{(i, \tau, *)} \in \mathbb{R}^{M \times 2}$ are the 2D positions of these tokens in view $\mathbf{\Pi}^{(i)}$. Here, the superscript $*$ indicates tokens retrieved from the Spatial Token Memory, and we distinguish them from the original tokens within the attention block. We apply the Rotary Position Embedding (RoPE) [47] to embed the projected 2D positions into the key tokens, while value tokens do not involve positional encoding in standard attention mechanism and are reused directly:

$$\mathbf{k}^{(i, \tau, *)} = \text{RoPE}\left(\mathbf{k}^{(\cdot, \tau)}, \mathbf{p}_{2D}^{(i, \tau, *)}\right), \quad \mathbf{v}^{(i, \tau, *)} = \mathbf{v}^{(\cdot, \tau)}. \quad (2)$$

We concatenate these transformed key and value tokens with the original KV tokens $\mathbf{k}^{(i,\tau)}, \mathbf{v}^{(i,\tau)}$ in the attention layer to compute the output. Because the injected tokens originate from the same underlying world and are reprojected with geometrically correct positions, they guide the video generator toward producing motions that are consistent with those already generated in other views.

Memory update. After computing the attention output, we update the memory $\mathcal{M}$ with all the KV tokens from the current attention layer. In standard transformer-based video generators, key tokens from the linear projection layers are passed through RoPE to encode 2D spatial positions. Since we aim to reuse these tokens in other views via 3D projection, we need to associate them with 3D positions instead. To this end, we store the key tokens $\mathbf{k}^{(\cdot,\tau)}$ *before* applying RoPE, along with the value tokens $\mathbf{v}^{(\cdot,\tau)}$. We use the placeholder notation $(\cdot)$ for the view index to emphasize that these tokens will be stored in the shared memory $\mathcal{M}$. Concretely, we merge all KV tokens produced by the current attention layer into $\mathcal{M}$. Since the camera matrix $\mathbf{\Pi}^{(i)}$ and the 2D spatial coordinates $\mathbf{p}_{2D}^{(i,\tau)}$ of the KV tokens are known, we only need to estimate their scalar depths $D^{(i,\tau)}$ to compute the 3D positions,

$$\mathbf{p}_{3D}^{(\cdot,\tau)} = \phi\left(\mathbf{p}_{2D}^{(i,\tau)}, D^{(i,\tau)}, \mathbf{\Pi}^{(i)}\right), \quad (3)$$

where ϕ is the camera-to-world transformation function and $\mathbf{p}_{3D}^{(\cdot,\tau)} \in \mathbb{R}^3$ denotes the absolute 3D positions of these tokens. In the following, we detail how to estimate token depths $D^{(i,\tau)}$ at the current denoising step τ .

Token depth estimation. Since the KV tokens encode meaningful dynamic contents of the generated video, their depths should correspond to the depth maps estimated from the fully denoised video frames. However, relying on fully denoised results would break the parallel generation across views. Instead, we estimate the denoised video frames directly from the noisy video latent at the current denoising step τ using the posterior mean property of flow-matching models [36]. Specifically, let $\mathbf{z}^{(i,\tau)} \in \mathbb{R}^{t \times h \times w \times c}$ denote the noisy video latent tensor at the denoising step τ . For a flow matching model with linear interpolation path, data schedule α_τ , noise schedule σ_τ , and velocity field $\mathbf{u}_\theta$, the clean latent $\hat{\mathbf{z}}^{(i,0)}$ can be estimated via the posterior mean using the relation (further details can be found in §2 of the *supplementary material*):

$$\hat{\mathbf{z}}^{(i,0)} = \mathbb{E}[\mathbf{z}^{(i,0)} \mid \mathbf{z}^{(i,\tau)}] = \frac{1}{\dot{\sigma}_\tau \alpha_\tau - \sigma_\tau \dot{\alpha}_\tau} \left(\dot{\alpha}_\tau \mathbf{z}^{(i,\tau)} - \alpha_\tau \mathbf{u}_\theta(\mathbf{z}^{(i,\tau)}, \tau) \right). \quad (4)$$

Throughout this section, a hat on a variable (*e.g.*, $\hat{\mathbf{z}}$) denotes an estimated quantity. After obtaining $\hat{\mathbf{z}}^{(i,0)}$, we decode it into video frames and employ a video depth estimation model [6] to extract their depth maps $\hat{\mathbf{D}}^{(i)} \in \mathbb{R}^{F \times H \times W}$, where $\hat{\mathbf{D}}^{(i)} = \{\hat{\mathbf{D}}_f^{(i)}\}_{f=0}^{F-1}$ and $\hat{\mathbf{D}}_f^{(i)}$ is the depth map of the f^{th} frame. Two remarks are in order. First, prior work [38, 41, 48] has observed that video latent tensors evolve smoothly and exhibit minimal variations across adjacent denoising steps. Likewise, we find that our estimated depth maps exhibit strong coherence between adjacent steps. We take advantage of this and explicitly estimate

depth maps at only three anchor timesteps corresponding to the early, intermediate, and late stages of the denoising trajectory, and we cache them for reuse at neighboring denoising steps. Second, since these depth maps are estimated independently per viewpoint, they inherently lack global 3D consistency. We address this by aligning them with the globally consistent geometry of the generated static world.

Token depth alignment. The primary source of inconsistency in these estimated depth maps is scale-and-shift ambiguity. While the relative depth values within each view are accurate, they lack a common global calibration across views. We address this problem by anchoring the depth maps to the 3D geometry of the generated static world. Concretely, when rasterizing multi-view images from the static 3D Gaussian representation, we simultaneously extract the corresponding rasterized depth maps $\{\mathbf{D}^{(i,\text{st})} \in \mathbb{R}^{H \times W}\}_{i=0}^{N-1}$ of the static scene. Since these rendered depth maps $\{\mathbf{D}^{(i,\text{st})}\}_{i=0}^{N-1}$ are inherently multi-view consistent, they serve as robust geometric anchors. We align the video depth maps with the static depth $\mathbf{D}^{(i,\text{st})}$ by computing a single optimal scale $\lambda^{(i)}$ and shift $\mu^{(i)}$ parameters,

$$\lambda^{(i)}, \mu^{(i)} = \underset{\lambda, \mu}{\operatorname{argmin}} \sum_{f=0}^{F-1} \|\lambda \cdot \hat{\mathbf{D}}_f^{(i)} + \mu \cdot \mathbf{1} - \mathbf{D}^{(i,\text{st})}\|_F^2, \quad (5)$$

where $\|\cdot\|_F$ denotes the Frobenius norm and $\mathbf{1} \in \mathbb{R}^{H \times W}$ denotes an all-ones matrix. We then apply the optimal calibration parameters to each estimated video depth map using the formula,

$$\mathbf{D}_f^{(i)} = \lambda^{(i)} \cdot \hat{\mathbf{D}}_f^{(i)} + \mu^{(i)} \cdot \mathbf{1}. \quad (6)$$

Through this global calibration, we obtain multi-view consistent video depth maps $\{\mathbf{D}^{(i)}\}_{i=1}^{N-1}$. We downsample these depth maps from the original $F \times H \times W$ resolution to $t \times h \times w$ to match the resolution of video tokens, where t, h, w denote temporal, height, and width dimensions of the video tokens, respectively. The scalar depth for each token is then directly extracted by indexing with its corresponding timestamp and 2D spatial coordinate. These token-level depth values allow us to employ (3) to compute the 3D positions of KV tokens via (3).

By employing the Spatial Token Memory, we generate multi-view consistent videos $\{\mathbf{I}^{(i)}\}_{i=0}^{N-1}$. We finetune the video generative model to incorporate the proposed memory, with full finetuning details provided in the *supplementary material*. In the following, we introduce how to construct the dynamic world from these consistent videos.

3.3 Dynamic World Construction

We employ 3D Gaussian Splatting (GS) to construct the final dynamic world. Since the foreground objects and the background scene exhibit fundamentally different dynamic behaviors, we decouple the world and reconstruct the foreground Gaussians G^{fg} and background Gaussians G^{bg} separately. We extract



Fig. 6: Dynamic world generation results from the input image and world descriptions. The camera icons sample regions with generated dynamics and motion events.

Table 1: Quantitative world generation results on the WorldScore [9] benchmark.

Methods	Overall Score		Dynamics			Controllability			Quality			
	Dynamics Only	Average	Motion Acc	Motion Mag	Motion Smooth	Camera Ctrl	Object Ctrl	Content Align	3D Consist	Photo Consist	Style Consist	Subjective Qual
LucidDreamer [8]	0.00	49.28	0.00	0.00	0.00	88.93	41.18	75.00	90.37	90.20	48.10	58.99
WonderJourney [68]	0.00	44.63	0.00	0.00	0.00	84.60	37.10	35.54	80.60	79.03	62.82	66.56
InvisibleStitch [10]	0.00	42.78	0.00	0.00	0.00	<u>93.20</u>	36.51	29.53	<u>88.51</u>	<u>89.19</u>	32.37	58.50
WonderWorld [67]	0.00	50.88	0.00	0.00	0.00	92.98	51.76	71.25	86.87	85.56	70.57	49.81
Voyager [21]	0.00	54.53	0.00	0.00	0.00	85.95	66.92	68.92	81.56	85.99	84.89	71.09
T2V-Turbo [33]	27.48	40.20	<u>34.87</u>	40.09	7.48	27.80	30.68	69.14	38.72	34.84	49.65	68.74
4D-fy [2]	<u>41.72</u>	32.10	22.22	22.88	80.06	69.92	55.09	0.85	35.47	1.59	32.04	0.89
DYNWORLD	48.90	66.11	45.41	<u>24.65</u>	<u>76.64</u>	93.27	<u>59.82</u>	70.24	85.36	83.29	70.23	52.22

binary video foreground masks with the foreground segmentation method [22]. This decomposes the videos into foreground videos $\{\mathbf{I}^{(i,\text{fg})}\}_{i=0}^{N-1}$ and background videos $\{\mathbf{I}^{(i,\text{bg})}\}_{i=0}^{N-1}$.

Background Gaussian Construction. As foreground objects may occlude the background, we introduce a foreground removal model [73] to obtain complete background videos. We then estimate background depth maps with our depth estimation pipeline [6] and calibrate them via Equations (5) and (6) for multi-view consistency. We also extract frame-wise background normals with a surface normal model [25]. While the background geometry remains static, its appearance may vary over time (*e.g.*, flowing water, moving clouds, or dynamic lighting). We thus model the background with constant depths and normals but time-varying colors. For each view $\Pi^{(i)}$, we average the depth and normal maps along the temporal dimension. We parameterize a background Gaussian kernel as $G^{\text{bg}} = \{\mathbf{x}^{\text{bg}}, \mathbf{R}^{\text{bg}}, \mathbf{s}^{\text{bg}}, \mathbf{o}^{\text{bg}}, \mathbf{c}^{\text{bg}}\}$. The static geometric properties $(\mathbf{x}^{\text{bg}} \in \mathbb{R}^3,$

Table 2: Quantitative results on the visual quality (VQ), temporal consistency (TC), T2I alignment (TA), and factual consistency (FC) metrics [15] show that DYNWORLD achieves better consistency.

Methods	CLIP (↑)	VQ (↑)	TC (↑)	TA (↑)	FC (↑)
MOTIA [56]	21.84	1.922	1.553	1.956	1.929
VACE [23]	23.80	2.287	1.852	2.654	1.858
DynVoyager [50]	23.78	2.381	1.952	2.656	2.003
DYNWORLD	24.55	2.781	2.447	2.997	2.235

Table 3: Ablation study on our proposed Spatial Token Memory, 3D token geometry, token depth alignment, background completion, and foreground Gaussians.

Methods	CLIP (↑)	VQ (↑)	TC (↑)	TA (↑)	FC (↑)
w/o memory	23.12	2.331	2.112	2.658	2.012
w/o 3D geo	19.98	2.322	1.884	2.672	1.915
w/o depth align	22.04	2.128	1.667	2.542	2.145
w/o bg completion	22.22	2.134	1.855	2.340	2.029
w/o fg GS	21.07	2.021	1.636	2.242	1.627
DYNWORLD	24.55	2.783	2.442	2.997	2.233



Fig. 7: World generation results compared with DynamicVoyager [50]. While DynamicVoyager yields limited world coverage, our DYNWORLD generates a larger world with consistent object motions.

$\mathbf{R}^{\text{bg}} \in \mathbb{R}^{3 \times 3}$, $\mathbf{s}^{\text{bg}} \in \mathbb{R}^3$, and $o^{\text{bg}} \in [0, 1]$) are initialized following §3.1, while time-varying colors $\mathbf{c}^{\text{bg}} \in [0, 255]^{F \times 3}$ are initialized from corresponding pixel colors across F video frames. These parameters of Gaussians are optimized using typical losses of $\mathcal{L}_1$ and SSIM.

Foreground Gaussian Construction. Unlike the background, foreground objects in the video exhibit time-varying geometry and appearance. For each view $\Pi^{(i)}$, we first estimate frame-wise normal maps, and apply the binary foreground masks to the calibrated depth maps $\mathbf{D}^{(i)}$ and normals to obtain foreground depth and normal maps. We parameterize a dynamic foreground Gaussian as $G^{\text{fg}} = \{\mathbf{x}^{\text{fg}}, \mathbf{R}^{\text{fg}}, \mathbf{s}^{\text{fg}}, \mathbf{o}^{\text{fg}}, \mathbf{c}^{\text{fg}}\}$, where every parameter is time-varying: $\mathbf{x}^{\text{fg}} \in \mathbb{R}^{F \times 3}$, $\mathbf{R}^{\text{fg}} \in \mathbb{R}^{F \times 3 \times 3}$, $\mathbf{s}^{\text{fg}} \in \mathbb{R}^{F \times 3}$, $\mathbf{o}^{\text{fg}} \in [0, 1]^F$, and $\mathbf{c}^{\text{fg}} \in [0, 255]^{F \times 3}$. These parameters are initialized frame-by-frame following §3.1. We jointly optimize both the foreground and background Gaussians by minimizing the losses $\mathcal{L}_1$ and SSIM. Through this decoupled global optimization, we effectively lift the multi-view consistent 2D videos into a large-scale, cohesive 3D dynamic world.

4 Experiments

This section first details our experimental setup, followed by a qualitative and quantitative evaluation of our dynamic world generation results against state-of-the-art methods. Finally, we present our efficiency analysis and ablation studies.

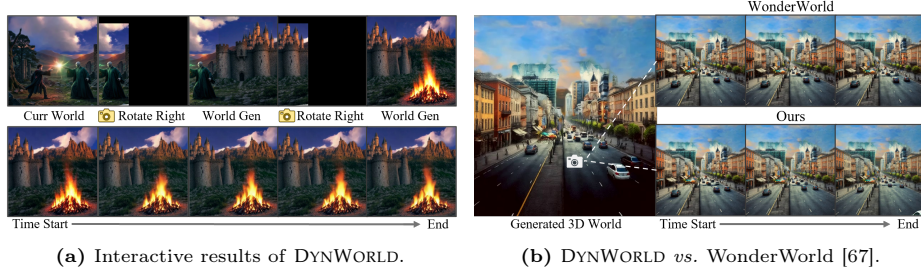


Fig. 8: DYNWORLD enables users to create a dynamic world interactively. Compared to WonderWorld [67], our method generates novel views with dynamic motions.

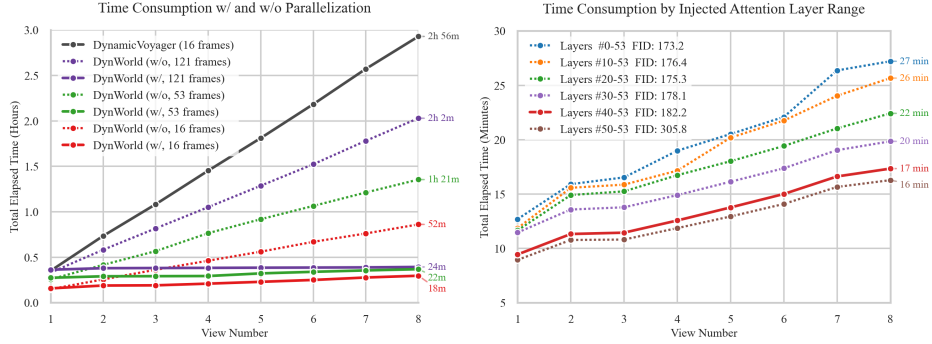


Fig. 9: Time consumption for world generation with different numbers of video frames and injected attention layers. Details are in §3 of the *supplementary material*.

4.1 Experiment Setups

Implementation details. We adopt Hunyuan1.5_480p_i2v_distilled [48] as our video generative model, with finetuning details provided in §3 of the *supplementary material*. All experiments are conducted on 8 NVIDIA A40 GPUs with 48GB memory. The spatial resolution of each view is set to 512×512 (*i.e.*, $H = W = 512$), and the number of video frames is set to $F = 37$. For auxiliary models, we employ Marigold Depth [25] as our image depth estimator, Video-Depth-Anything [6] as our video depth estimator, Marigold Normal [25] as our image normal estimator, and Minimax-Remover [73] as our foreground removal model.

Evaluation metrics. We adopt the WorldScore [9] and VideoScore [15] benchmarks to evaluate world generation from both static and dynamic perspectives. WorldScore assesses world generation across three dimensions with ten metrics in total. Since our work focuses on dynamic world generation, we additionally report the average score on the Dynamics dimension to specifically evaluate the temporal dynamics of the generated worlds. VideoScore is a widely used bench-

mark for video quality assessment, which we employ to evaluate the quality of videos rendered from different viewpoints of the generated world.

4.2 Dynamic 3D World Generation

Qualitative and quantitative results. We present two 3D dynamic world generation results in Figure 6. Each world contains seven views where the input view is in the middle of the world. We also list the motion prompts used to generate the scene dynamics in the figure. It can be seen that our model supports large 3D dynamic world generation with diverse dynamic objects (walking people, floating boats, flying birds, and butterflies) and textures (waving water, moving clouds). We also compare our method with state-of-the-art 3D dynamic world generation method DynamicVoyager [50] in Figure 7. It can be seen that our model generates a larger dynamic world with more consistent motions, whereas DynamicVoyager fuses only a 3D world between two sequential perpetual views with inconsistent motions between two views.

Interactive results. Our method also supports interactive world generation, as shown in Figure 8. Compared with the interactive method WonderWorld [67] that only generates a static world, our model can generate a large 3D dynamic world with diverse dynamic objects and textures.

Comparison with outpainting and stitching methods. Figure 3 compares DYNWORLD with the outpainting baseline VACE [23]. In the overlapping setting (Figure 3a), VACE already shows motion inconsistency and local artifacts around the moving car, while DYNWORLD preserves consistent object motions. For non-overlapping (Figure 3b), VACE fails to outpaint content consistent with the world context, while DYNWORLD successfully maintains consistent object motions across views. Figure 2 further compares DYNWORLD with state-of-the-art denoising score stitching method SyncSDE [29]. DYNWORLD maintains coherent sky motion between views, while SyncSDE introduces visible seams near the view boundary in the red dashed boxes. Table 2 quantitatively demonstrates the superior performance of our method on CLIP and VideoScore metrics.

4.3 Efficiency Study

Figure 9 reports the end-to-end wall-clock time for world generation as the number of views increases. We first study the efficiency with 16, 53, and 121 video frames for each view. Compared to DynamicVoyager [50] that scales close to linearly with the number of views and requires 2 hours and 56 minutes for eight views (16 frames), DYNWORLD reduces the runtime to 18-22 minutes (about $10 \times$ acceleration) with parallel generation. We also study the consistency performance of using memory in different numbers of attention layers in the right figure. It shows that DYNWORLD achieves only a minor FID [16] increase (from 173.2 to 182.2) while reducing generation time by 10 minutes (a 50% decrease) when injecting memory into only the last #40–53 attention layers, making this a practical configuration for memory construction.

4.4 Ablation Study

We conduct an ablation study of the proposed memory in Figure 5 and Table 3. The results show that the proposed geometric transformation of tokens preserves their spatial relationships, while the Spatial Token Memory effectively maintains motion consistency across views. We further report quantitative ablation results in Table 3 for the proposed depth alignment, background (bg) completion, and foreground Gaussian components. The improvements observed in both CLIP score and VideoScore [15] demonstrate the effectiveness of these design choices.

5 Conclusion

In this paper, we presented DYNWORLD, an efficient framework for generating large dynamic 3D worlds. Our approach leverages the proposed Spatial Token Memory to maintain motion consistency across different viewpoints, enabling multiple views of the dynamic world to be generated in parallel. Experimental results demonstrate that our method can synthesize large 3D worlds containing diverse dynamic objects and motions. Compared with sequential world generation pipelines, our approach reduces generation time from 3 hours to 15–30 minutes, achieving approximately a $10\times$ speedup.

Limitations. Since DYNWORLD relies on KV tokens from historical views to maintain the 3D consistency of the generated world, we leave the optimization of KV token lifecycle management for future work. Additionally, our token geometry estimation depends on external depth models, making the system vulnerable to inaccurate depth predictions. We expect this limitation to be naturally mitigated by ongoing advancements in depth estimation.

Acknowledgements

The authors acknowledge the support of the University of Pennsylvania Startup Funds. The authors also would like to sincerely thank Tianjiao Ding, Qiao Feng, Boshu Lei, and Chen Wang for their insightful feedback.

References

1. Agarwal, S., Snavely, N., Simon, I., Seitz, S.M., Szeliski, R.: Building rome in a day. In: ICCV. Kyoto, Japan (2009)
2. Bahmani, S., Skorokhodov, I., Rong, V., Wetzstein, G., Guibas, L., Wonka, P., Tulyakov, S., Park, J.J., Tagliasacchi, A., Lindell, D.B.: 4d-fy: Text-to-4d generation using hybrid score distillation sampling. In: CVPR (2024)
3. Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al.: Stable video diffusion: Scaling latent video diffusion models to large datasets. arXiv preprint arXiv:2311.15127 (2023)

4. Cao, W., Zhang, H., Tian, F., Wu, Y., Li, Y., Wang, S., Yu, N., Liu, Y.: Free-Orbit4D: Training-free arbitrary camera redirection for monocular videos via geometry-complete 4D reconstruction. arXiv preprint arXiv:2601.18993 (2026)
5. Chen, M., Shapovalov, R., Laina, I., Monnier, T., Wang, J., Novotny, D., Vedaldi, A.: Partgen: Part-level 3d generation and reconstruction with multi-view diffusion models. In: CVPR (2025)
6. Chen, S., Guo, H., Zhu, S., Zhang, F., Huang, Z., Feng, J., Kang, B.: Video depth anything: Consistent depth estimation for super-long videos. In: CVPR (2025)
7. Chen, Y., Liang, Y., Wang, J., Chen, T., Cheng, J., Gu, Z., Huang, Y., Jiang, Z., Li, W., Li, T., et al.: Teleworld: Towards dynamic multimodal synthesis with a 4d world model. arXiv preprint arXiv:2601.00051 (2025)
8. Chung, J., Lee, S., Nam, H., Lee, J., Lee, K.M.: Luciddreamer: Domain-free generation of 3d gaussian splatting scenes. IEEE TVCG (2023)
9. Duan, H., Yu, H.X., Chen, S., Fei-Fei, L., Wu, J.: Worldscore: A unified evaluation benchmark for world generation. In: ICCV (2025)
10. Engstler, P., Vedaldi, A., Laina, I., Rupprecht, C.: Invisible stitch: Generating smooth 3d scenes with depth inpainting. arXiv preprint arXiv:2404.19758 (2024)
11. Fang, S., Jiang, H., Bai, Y., Mitra, N.J., Huang, Q.: Worldreel: 4D video generation with consistent geometry and motion modeling. arXiv preprint arXiv:2512.07821 (2025)
12. Fang, S., Yu, C., Wang, F., Huang, Q.: Mvroom: Controllable 3d indoor scene generation with multi-view diffusion models. arXiv preprint arXiv:2512.04248 (2025)
13. Fridman, R., Abecasis, A., Kasten, Y., Dekel, T.: SceneScape: Text-driven consistent scene generation. In: NeurIPS (2023)
14. Gao*, R., Holynski*, A., Henzler, P., Brussee, A., Martin-Brualla, R., Srinivasan, P.P., Barron, J.T., Poole*, B.: CAT3D: Create anything in 3D with multi-view diffusion models. In: NeurIPS (2024)
15. He, X., Jiang, D., Zhang, G., Ku, M., Soni, A., Siu, S., Chen, H., Chandra, A., Jiang, Z., Arulraj, A., Wang, K., Do, Q.D., Ni, Y., Lyu, B., Narsupalli, Y., Fan, R., Lyu, Z., Lin, Y., Chen, W.: VideoScore: Building automatic metrics to simulate fine-grained human feedback for video generation. In: EMNLP (2024)
16. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. In: NeurIPS. vol. 30 (2017)
17. Ho, J., Salimans, T.: Classifier-free diffusion guidance. In: NeurIPS (2021)
18. Höllein, L., Cao, A., Owens, A., Johnson, J., Nießner, M.: Text2Room: Extracting textured 3D meshes from 2D text-to-image models. In: ICCV (2023)
19. Hong, W., Ding, M., Zheng, W., Liu, X., Tang, J.: CogVideo: Large-scale pretraining for text-to-video generation via transformers. In: ICLR (2022)
20. Huang, X., Li, Z., He, G., Zhou, M., Shechtman, E.: Self forcing: Bridging the train-test gap in autoregressive video diffusion. In: NeurIPS (2025)
21. HunyuanWorld, T.: Hy-world 1.5: A systematic framework for interactive world modeling with real-time latency and geometric consistency, http://3d-models.hunyuan.tencent.com/world/world1_5/HYWorld_1.5_Tech_Report.pdf, accessed: 2026-06-30
22. Jain, J., Li, J., Chiu, M.T., Hassani, A., Orlov, N., Shi, H.: OneFormer: One transformer to rule universal image segmentation. In: CVPR (2023)
23. Jiang, Z., Han, Z., Mao, C., Zhang, J., Pan, Y., Liu, Y.: Vace: All-in-one video creation and editing. In: ICCV. pp. 17191–17202 (2025)

24. Kaneva, B., Sivic, J., Torralba, A., Avidan, S., Freeman, W.T.: Infinite images: Creating and exploring a large photorealistic virtual space. *Proceedings of the IEEE* **98**(8), 1391–1407 (2010)
25. Ke, B., Qu, K., Wang, T., Metzger, N., Huang, S., Li, B., Obukhov, A., Schindler, K.: Marigold: Affordable adaptation of diffusion-based image generators for image analysis. *arXiv preprint arXiv:2505.09358* (2025)
26. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3D gaussian splatting for real-time radiance field rendering. In: *TOG* (2023)
27. Kim, J., Koo, J., Yeo, K., Sung, M.: Synctweedies: A general generative framework based on synchronized diffusions. *arXiv preprint arXiv:2403.14370* (2024)
28. Labs, B.F.: Flux. <https://github.com/black-forest-labs/flux> (2024), accessed: 2026-06-30
29. Lee, H., Lee, H., Han, S.: Syncsde: A probabilistic framework for diffusion synchronization. In: *CVPR*. pp. 17508–17517 (2025)
30. Lee, Y., Kim, K., Kim, H., Sung, M.: Syncdiffusion: Coherent montage via synchronized joint diffusions. In: *NeurIPS* (2023)
31. Li, B., Zheng, C., Zhu, W., Mai, J., Zhang, B., Wonka, P., Ghanem, B.: Vivid-zoo: Multi-view video generation with diffusion model. In: *NeurIPS* (2024)
32. Li, B., Mishra, M., Liu, Y.I., Tao, S., Kumar, N., Gray, A.G., Wickramarachchi, R., Francis, J., Scherer, S., Silver, T.: Recover, discover, plan: Learning skills and concepts from robot failures. *arXiv preprint arXiv:2606.18328* (2026)
33. Li, J., Feng, W., Fu, T.J., Wang, X., Basu, S., Chen, W., Wang, W.Y.: T2v-turbo: Breaking the quality bottleneck of video consistency model with mixed reward feedback. In: *NeurIPS* (2024)
34. Li, R., Yi, B., Liu, J., Gao, H., Ma, Y., Kanazawa, A.: Cameras as relative positional encoding. In: *NeurIPS* (2025)
35. Li, Z., Wang, Q., Snavely, N., Kanazawa, A.: Infinitenature-Zero: Learning perpetual view generation of natural scenes from single images. In: *ECCV* (2022)
36. Lipman, Y., Havasi, M., Holderrieth, P., Shaul, N., Le, M., Karrer, B., Chen, R.T., Lopez-Paz, D., Ben-Hamu, H., Gat, I.: Flow matching guide and code. *arXiv preprint arXiv:2412.06264* (2024)
37. Liu, A., Tucker, R., Jampani, V., Makadia, A., Snavely, N., Kanazawa, A.: Infinite Nature: Perpetual view generation of natural scenes from a single image. In: *ICCV* (2021)
38. Liu, F., Zhang, S., Wang, X., Wei, Y., Qiu, H., Zhao, Y., Zhang, Y., Ye, Q., Wan, F.: Timestep embedding tells: It’s time to cache for video diffusion model. In: *CVPR* (2025)
39. Liu, R., Wu, R., Van Hoorick, B., Tokmakov, P., Zakharov, S., Vondrick, C.: Zero-1-to-3: Zero-shot one image to 3d object. In: *ICCV*. pp. 9298–9309 (October 2023)
40. Liu, Y., Lin, C., Zeng, Z., Long, X., Liu, L., Komura, T., Wang, W.: SyncDreamer: Generating multiview-consistent images from a single-view image. In: *ICLR* (2024)
41. Ma, X., Fang, G., Wang, X.: Deepcache: Accelerating diffusion models for free. In: *CVPR* (2024)
42. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF: Representing scenes as neural radiance fields for view synthesis. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) *ECCV* (2020)
43. Parish, Y.I.H., Müller, P.: Procedural modeling of cities. In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. p. 301–308 (2001)
44. Poole, B., Jain, A., Barron, J.T., Mildenhall, B.: Dreamfusion: Text-to-3d using 2d diffusion. In: *ICLR* (2022)

45. Ren, J., Pan, L., Tang, J., Zhang, C., Cao, A., Zeng, G., Liu, Z.: DreamGaussian4D: Generative 4D gaussian splatting. arXiv preprint arXiv:2312.17142 (2023)
46. Song, C., Stary, M., Chen, B., Kopanas, G., Sitzmann, V.: Generative view stitching. arXiv preprint arXiv:2510.24718 (2025)
47. Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., Liu, Y.: Roformer: Enhanced transformer with rotary position embedding. *Neurocomput.* **568**(C) (Feb 2024)
48. Team, T.H.F.M.: Hunyuanvideo 1.5 technical report. arXiv preprint arXiv:2511.18870 (2025)
49. Team, T.H.: Hunyuan3d 2.5: Towards high-fidelity 3d assets generation with ultimate details. arXiv preprint arXiv:2506.16504 (2025)
50. Tian, F., Ding, T., Luo, J., Min, H., Vidal, R.: Voyaging into perpetual dynamic scenes from a single view. In: *ICCV* (2025)
51. Tian, F., Du, S., Duan, Y.: MonoNeRF: Learning a generalizable dynamic radiance field from monocular videos. In: *ICCV* (October 2023)
52. Tian, F., Duan, Y., Wang, A., Guo, J., Du, S.: Semantic Flow: Learning semantic fields of dynamic scenes from monocular videos. In: *ICLR* (May 2024)
53. Tian, F., Fan, J., Yu, X., Du, S., Song, M., Zhao, Y.: TCVM: Temporal contrasting video montage framework for self-supervised video representation learning. In: *ACCV*. pp. 1539–1555 (December 2022)
54. Voleti, V., Yao, C.H., Boss, M., Letts, A., Pankratz, D., Tochilkin, D., Laforte, C., Rombach, R., Jampani, V.: SV3D: Novel multi-view synthesis and 3D generation from a single image using latent video diffusion. In: *ECCV* (2024)
55. Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.W., Chen, D., Yu, F., Zhao, H., Yang, J., Zeng, J., Wang, J., Zhang, J., Zhou, J., Wang, J., Chen, J., Zhu, K., Zhao, K., Yan, K., Huang, L., Feng, M., Zhang, N., Li, P., Wu, P., Chu, R., Feng, R., Zhang, S., Sun, S., Fang, T., Wang, T., Gui, T., Weng, T., Shen, T., Lin, W., Wang, W., Wang, W., Zhou, W., Wang, W., Shen, W., Yu, W., Shi, X., Huang, X., Xu, X., Kou, Y., Lv, Y., Li, Y., Liu, Y., Wang, Y., Zhang, Y., Huang, Y., Li, Y., Wu, Y., Liu, Y., Pan, Y., Zheng, Y., Hong, Y., Shi, Y., Feng, Y., Jiang, Z., Han, Z., Wu, Z.F., Liu, Z.: Wan: Open and advanced large-scale video generative models. arXiv preprint arXiv:2503.20314 (2025)
56. Wang, F.Y., Wu, X., Huang, Z., Shi, X., Shen, D., Song, G., Liu, Y., Li, H.: Be-your-outpainter: Mastering video outpainting through input-specific adaptation. In: *ECCV* (2024)
57. Wang, Q., Ye, V., Gao, H., Zeng, W., Austin, J., Li, Z., Kanazawa, A.: Shape of motion: 4d reconstruction from a single video. In: *ICCV* (2025)
58. Wang, Y.A., Chen, Y.N.: What do position embeddings learn? an empirical study of pre-trained language model positional encoding. In: *EMNLP* (2020)
59. Wu, C., Li, J., Zhou, J., Lin, J., Gao, K., Yan, K., ming Yin, S., Bai, S., Xu, X., Chen, Y., Chen, Y., Tang, Z., Zhang, Z., Wang, Z., Yang, A., Yu, B., Cheng, C., Liu, D., Li, D., Zhang, H., Meng, H., Wei, H., Ni, J., Chen, K., Cao, K., Peng, L., Qu, L., Wu, M., Wang, P., Yu, S., Wen, T., Feng, W., Xu, X., Wang, Y., Zhang, Y., Zhu, Y., Wu, Y., Cai, Y., Liu, Z.: Qwen-image technical report. arXiv preprint arXiv:2508.02324 (2025)
60. Wu, R., Gao, R., Poole, B., Trevithick, A., Zheng, C., Barron, J.T., Holynski, A.: Cat4D: Create anything in 4D with multi-view video diffusion models. arXiv preprint arXiv:2411.18613 (2024)
61. Xia, C., Zhang, S., Liu, F., Liu, C., Hirunyaratsameewong, K., Duan, Y.: Scene-painter: Semantically consistent perpetual 3d scene generation with concept relation alignment. In: *ICCV* (2025)

62. Xiang, J., Chen, X., Xu, S., Wang, R., Lv, Z., Deng, Y., Zhu, H., Dong, Y., Zhao, H., Yuan, N.J., Yang, J.: Native and compact structured latents for 3d generation. arXiv preprint arXiv:2512.14692 (2025)
63. Xie, H., Chen, Z., Hong, F., Liu, Z.: CityDreamer: Compositional generative model of unbounded 3D cities. In: CVPR (2024)
64. Xie, Y., Yao, C.H., Voleti, V., Jiang, H., Jampani, V.: SV4D: Dynamic 3D content generation with multi-frame and multi-view consistency. arXiv preprint arXiv:2407.17470 (2024)
65. Xu, Z., Peng, S., Lin, H., He, G., Sun, J., Shen, Y., Bao, H., Zhou, X.: 4k4D: Real-time 4D view synthesis at 4K resolution. In: CVPR (2024)
66. Yang, Z., Teng, J., Zheng, W., Ding, M., Huang, S., Xu, J., Yang, Y., Hong, W., Zhang, X., Feng, G., et al.: CogVideoX: Text-to-video diffusion models with an expert transformer. In: ICLR (2024)
67. Yu, H.X., Duan, H., Herrmann, C., Freeman, W.T., Wu, J.: WonderWorld: Interactive 3D scene generation from a single image. arXiv preprint arXiv:2406.09394 (2024)
68. Yu, H.X., Duan, H., Hur, J., Sargent, K., Rubinstein, M., Freeman, W.T., Cole, F., Sun, D., Snavely, N., Wu, J., et al.: WonderJourney: Going from anywhere to everywhere. In: CVPR (2024)
69. Zhang, H., Chen, X., Wang, Y., Liu, X., Wang, Y., Qiao, Y.: 4diffusion: Multi-view video diffusion model for 4d generation. In: NeurIPS (2024)
70. Zhang, L., Cai, S., Li, M., Wetzstein, G., Agrawala, M.: Frame context packing and drift prevention in next-frame-prediction video diffusion models. In: NeurIPS (2025)
71. Zhou, D., Sun, Q., Peng, Y., Yan, K., Dong, R., Wang, D., Ge, Z., Duan, N., Zhang, X.: Taming teacher forcing for masked autoregressive video generation. In: CVPR. pp. 7374–7384 (2025)
72. Zhu, H., Zhao, M., He, G., Su, H., Li, C., Zhu, J.: Causal forcing: Autoregressive diffusion distillation done right for high-quality real-time interactive video generation. arXiv preprint arXiv:2602.02214 (2026)
73. Zi, B., Peng, W., Qi, X., Wang, J., Zhao, S., Xiao, R., Wong, K.F.: Minimax-remover: Taming bad noise helps video object removal. In: NeurIPS (2025)