FLASHCHORD: REUSING AND REINFORCING DISTILLATION FOR EFFICIENT 4D SCENE GENERATION

Anonymous authors
Paper under double-blind review

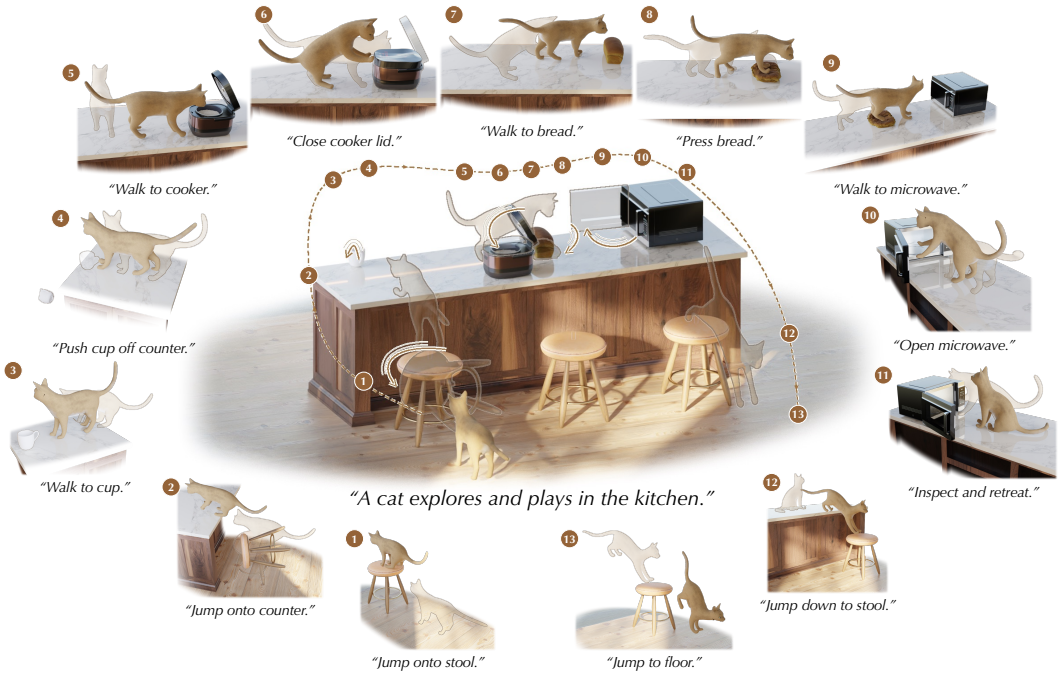

Figure 1: **Given a 3D scene and text prompts, FlashCHORD efficiently generates corresponding scene-level motion.** This efficiency further makes long-horizon generation practical: the sequence shown comprises 13 motion clips, each generated in around 32 minutes on a single NVIDIA H200 GPU. Our method achieves a 24× speedup over the previous state of the art (Lyu et al., 2026a) while improving motion quality.

ABSTRACT

Recent advances in distilling 4D scene motion from video generation models have demonstrated promising results. However, lengthy optimization and motion artifacts limit their practical use. In this paper, we introduce FlashCHORD, a framework that generates more realistic motion with substantially shorter optimization times. At the core of our method is a strategy to retain and reuse supervision signals obtained from expensive video generation model evaluations. Our analysis further shows that optimizing deformation through matching rendered appearance often leads to motion artifacts, as the resulting motion is highly sensitive to the input appearance. We address this with semantics-aware coloring, which recolors scene regions to provide visual cues better suited to motion optimization. Together, these designs yield a framework that outperforms prior distillation-based methods in both runtime and motion quality. Extensive experiments demonstrate its effectiveness across diverse scenes and motion prompts. Video results are available at `https://chordflash.github.io/chordflashpage/`.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



Figure 2: **Diverse 4D scene generation results produced by our method.** Our pipeline is robust and general, producing faithful motion for objects of diverse categories.

1 INTRODUCTION

Generating 4D (3D + time) scene motions — time-varying deformations of 3D objects within a scene — is a long-standing challenge in computer graphics and generative AI. Such motions animate 3D assets into vivid dynamic scenes for video games and robotics simulation, yet they are costly to create and scarce on the Internet. This data scarcity limits the generalization capability of end-to-end learning-based methods (Wu et al., 2025c; Gat et al., 2025; Chen et al., 2026; Wang et al., 2026) in this modality and largely restricts them to object-centric motions.

Recent approaches (Yu et al., 2024; Yuan et al., 2024; Bahmani et al., 2024; Lyu et al., 2026a) attempt to overcome this 4D data scarcity problem by leveraging motion priors from 2D video generative models. Given the abundant motion priors in 2D video generative models, these distillation-based 4D generation approaches lift 2D information to 4D through an analysis-by-synthesis process: maintaining a 4D scene representation and iteratively updating it with gradients from video generative models. Although these approaches yield impressive results, their optimization-based nature leads to lengthy generation times and optimization instability. Notably, CHORD (Lyu et al., 2026a) achieves state-of-the-art 4D scene motion generation quality with a generation time of 12 hours on a single H200 GPU, while still producing artifacts in the generated motion because of local minima during the optimization.

This paper proposes new algorithms to enhance both the efficiency and quality of distillation-based 4D scene generation approaches. Since such methods possess an optimization-based nature, we improve them from the perspective of optimization: how should we accelerate each optimization step, and how should we stabilize the optimization landscape?

We observe that the running time of each optimization step is dominated by querying the large video generative models. To reduce this cost, we propose a caching mechanism that amortizes each expensive query across several optimization steps. Specifically, we maintain a fixed-capacity first-in-first-out (FIFO) cache that stores the feedback signals from recent queries to the video generative models. At each optimization step, instead of issuing a new set of queries, we reuse the cached feedback signals to update the current 4D representation. We further design a set of algorithms that invalidate stale signals and insert new ones into the cache, thus keeping our supervision fresh and effective at an n-fold lower amortized cost.

Furthermore, we show that the stability of the optimization landscape can be significantly improved with a surrogate distillation objective. Existing distillation-based 4D generation approaches directly operate in the RGB space, which can be easily affected by the appearance of the input mesh. For instance, a featureless mesh texture would provide little meaningful gradients for optimizing deformations in an analysis-by-synthesis approach, which makes the optimization fragile. Instead, we optimize motion using surrogate appearances, obtained by recoloring the original meshes with semantics-aware textures. This creates natural correspondence in the optimization process and stabilizes the optimization landscape.

Together, these designs underpin FlashCHORD, a framework that combines actively constructed, reusable supervision with visual cues tailored to motion optimization. Experiments across diverse scenes show that FlashCHORD achieves up to a $24\times$ speedup over CHORD (Lyu et al., 2026a)

(from 12 hours to 32 minutes) while improving the quality of the results, as reflected by gains in user preference by 14.70 and 23.37 percentage points for text alignment and motion realism, respectively.

In summary, our contributions are as follows:

1. A supervision-reuse mechanism that enables each expensive video generation model query to support multiple scene updates, avoiding repeated teacher evaluations at every iteration.
2. A scene-coloring strategy that tailors supervision to motion optimization by assigning distinguishable, spatially coherent colors to scene regions.
3. An integrated framework for 4D scene generation, with experiments across diverse scenes demonstrating improvements in both motion quality and optimization efficiency.

2 Related Work

4D Scene Motion Generation. Automatically generating scene-level 4D motion for characters and objects has long been intractable in computer graphics and generative AI because training data for this modality are scarce (Lyu et al., 2026a). Prior works (Wu et al., 2025c; 2026; Yenphraphai et al., 2025; Zhang et al., 2025) train end-to-end generators directly on the limited data, yet this data scarcity bottlenecks their generalization capability. Video generative models (Wan et al., 2025; Brooks et al., 2024), however, contain abundant motion priors and are therefore widely used by 4D generation approaches to obtain 4D information. We categorize these approaches by how they use video generative models. The first type fine-tunes video generative models by adding camera control to produce multi-view videos from an input video (Bahmani et al., 2025; Bai et al., 2025; He et al., 2024; Wang et al., 2024b; Xu et al., 2024; Zhou et al., 2025; 2026), and then reconstructs the deformable geometry from these videos (Chu et al., 2024; Lei et al., 2025; Wang et al., 2025; Wu et al., 2025b). However, these approaches do not guarantee geometric consistency in the first stage and therefore yield artifacts in the reconstructed geometry.

The second type generates 4D content through analysis-by-synthesis and maintains a persistent 4D representation throughout the optimization (Bahmani et al., 2024; Chen et al., 2025; Hongyuan et al., 2026; Gao et al., 2024; Geng et al., 2025; Jiang et al., 2024; Li et al., 2024; Liang et al., 2024a; Lyu et al., 2026b; Ma et al., 2025; Mou et al., 2025; Liao et al., 2025; Ren et al., 2023; 2024; Sun et al., 2024; 2025; Uzolas et al., 2025; Wang et al., 2024a; Xie et al., 2024; Yang et al., 2024; Yao et al., 2025; Yu et al., 2024; Yuan et al., 2024; Zhao et al., 2023). This design guarantees geometric consistency, yet its optimization is typically slow. Our method belongs to the second category and significantly accelerates its optimization while improving generation quality.

Distillation-based 3D Generation. Popularized by Score Distillation Sampling (SDS) in DreamFusion (Poole et al., 2023), distillation-based 3D generation is a family of methods (Lin et al., 2023; Wang et al., 2023a) that generate 3D/4D content by distilling from 2D generative models rather than by training with direct 3D/4D supervision. Prior works improve the computational efficiency (Tang et al., 2024; Ren et al., 2023), generation diversity (Wang et al., 2023c), and generation quality (Wang et al., 2023c; Katzir et al., 2024; Liang et al., 2024b; Lukoianov et al., 2024; McAllister et al., 2024; Tang et al., 2023; Wang et al., 2023b) of these methods. Our paper focuses specifically on distillation-based 4D motion generation and proposes specialized algorithms to improve both the speed and the quality of distillation.

3 Method

Given a static 3D scene containing multiple objects and a text prompt describing the desired motion, our goal is to generate deformation sequences of the objects in the scene. Fig. 3 illustrates our pipeline. We represent the input scene geometry using 3D Gaussians (Kerbl et al., 2023), converting meshes to this representation when necessary. The optimized deformation is then transferred to the input meshes to produce the final mesh sequences. Following CHORD (Lyu et al., 2026a) and SC4D (Wu et al., 2024), we use a sparse set of control points to drive the Gaussians' deformation. We iteratively optimize their time-dependent transformations with guidance distilled from a pretrained video generative model, using Wan 2.2 (Wan et al., 2025) by default. After optimization,

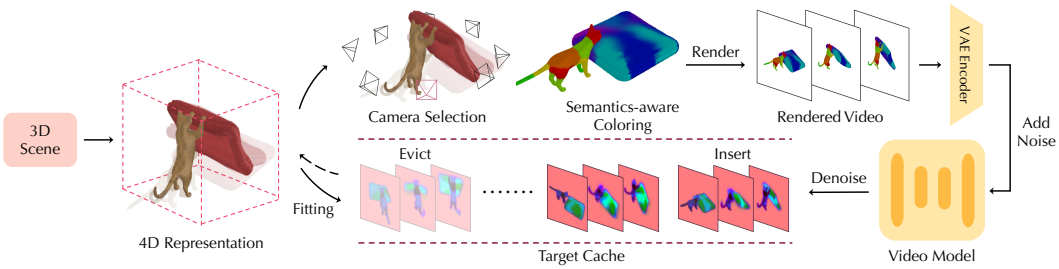

Figure 3: **Overview.** FlashCHORD optimizes a 4D representation iteratively using targets from a fixed-capacity FIFO cache. Only at scheduled refreshes do we query the video model to generate a new pseudo-target, which is inserted into the cache and reused across subsequent optimization steps.

the learned motion is transferred back to drive the input meshes. Representation details are provided in Section A.1.

3.1 Preliminary: Score Distillation for 4D Generation

We employ Score Distillation Sampling (SDS) (Poole et al., 2023) to obtain supervision from video generation models. Originally introduced for 3D generation, SDS optimizes a 3D representation using guidance from pretrained image diffusion models (Ho et al., 2020). At each optimization iteration, an image is rendered and encoded by a VAE (Kingma & Welling, 2014) encoder into a latent $\mathbf{z}$. A noise level τ and Gaussian noise $\boldsymbol{\epsilon}$ are then sampled and used to perturb the latent, producing $\mathbf{z}_\tau$. The diffusion model takes $\mathbf{z}_\tau$ as input and predicts the noise $\hat{\epsilon}$. SDS updates the scene with the following gradient:

$$\nabla_\theta \mathcal{L}_{\text{SDS}}(\theta; \mathbf{z}, \mathbf{y}) = \mathbb{E}_{\tau,\epsilon}\left[w(\tau) \left(\hat{\epsilon}\left(\mathbf{z}_\tau; \tau, \mathbf{y}\right) - \epsilon \right) \frac{\partial \mathbf{z}}{\partial \theta} \right], \tag{1}$$

where θ denotes the parameters of the 3D representation, $\mathbf{y}$ is the conditioning signal, and $w(\tau)$ is a noise-level-dependent weighting function.

Recent methods have extended SDS to 4D scene generation (Yu et al., 2024; Geng et al., 2025; Yuan et al., 2024; Bahmani et al., 2024; Wu et al., 2024; Lyu et al., 2026a). In this setting, $\mathbf{z}$ denotes the encoded latent of a rendered video. CHORD (Lyu et al., 2026a) further introduced rectified-flow supervision for SDS-based 4D scene optimization. Under the rectified-flow formulation, the target velocity is $\boldsymbol{\epsilon} - \mathbf{z}$, yielding the following gradient:

$$\nabla_\theta \mathcal{L}_{\text{W-RFSDS}}(\theta; \mathbf{z}, \mathbf{y}) = \mathbb{E}_{\tau\sim\hat{w}(\tau),\,\boldsymbol{\epsilon}}\left[\left(\hat{\mathbf{v}}(\mathbf{z}_\tau; \tau, \mathbf{y}) - \boldsymbol{\epsilon} + \mathbf{z} \right) \frac{\partial \mathbf{z}}{\partial \theta} \right], \tag{2}$$

where $\hat{\mathbf{v}}$ denotes the velocity predicted by the rectified-flow model and $\hat{w}(\tau)$ denotes the distribution used to sample noise levels.

3.2 Distillation with Reusable Pseudo-Targets

Although the score-distillation approach described above offers a way to generate motion using supervision obtained from a video model, repeatedly evaluating these large models makes optimization computationally expensive, limiting its practical use. To reduce this cost, we propose to retain and reuse the supervision signals across multiple updates rather than discarding it after a single update, as in conventional SDS. To enable effective reuse while keeping supervision aligned with the evolving scene, we design a set of complementary algorithms, as detailed below.

Pseudo-Target Construction. To express the model's supervision in a reusable form, we draw inspiration from the reconstruction-loss formulation of SDS (Zhou & Tulsiani, 2023; Zhu et al., 2024), which interprets score distillation as fitting the scene to a model-generated pseudo-target. For the rectified-flow update in Eq. (2), this target takes the form

$$\mathbf{z}^\star = \text{sg}[\boldsymbol{\epsilon} - \hat{\mathbf{v}}(\mathbf{z}_\tau; \tau, \mathbf{y})], \tag{3}$$

with the corresponding fitting loss

$$\mathcal{L}_{\text{target}} = \frac{1}{2} \left\| \mathbf{z} - \mathbf{z}^\star \right\|_2^2, \tag{4}$$

where sg denotes stop-gradient. Holding the target fixed gives

$$\nabla_{\mathbf{z}_{\theta,\mathbf{c}}} \mathcal{L}_{\text{target}} = \mathbf{z} - \mathbf{z}^\star = \hat{\mathbf{v}} - \boldsymbol{\epsilon} + \mathbf{z}, \tag{5}$$

which matches the residual in Eq. (2). Applying the chain rule therefore recovers the sampled RF-SDS gradient at the scene state used to generate the target. The detached pseudo-target thus expresses the model's supervision in a reusable form. Once stored, the fitting loss and its gradient can be evaluated at updated scene parameters by rendering, encoding, and backpropagating through the current scene, enabling multiple optimization steps without additional video-model evaluations.

Caching algorithms. Throughout optimization, we maintain a fixed-capacity first-in-first-out (FIFO) cache of the targets computed above. Each cache entry $\mathcal{C}_i = (\mathbf{z}_i^\star, \mathbf{c}_i)$ stores both a target and its rendering camera pose. At the u-th of T optimization steps, we issue $\rho(u)$ new queries to the video generative model and use the resulting targets to refresh our cache. We refer to $\rho(u)$ as the *refreshing rate* of our cache.

Over a single optimization run, we fix the total number of video-model queries at $B = \int_0^T \rho(u)\mathrm{d}u$. Instead of distributing these queries uniformly throughout optimization, we design an adaptive refreshing strategy so that the query frequency matches the rate at which the 4D scene changes. Early in optimization, the scene changes rapidly, so we query more frequently. Later, when optimization stabilizes, we query less frequently.

To design such a strategy, we observe that the learning rate $\eta(u)$ already follows this adaptive pattern (Huang et al., 2024; Wu et al., 2024; Lyu et al., 2026a), with larger values at the beginning and smaller values at the end. Therefore, we set $\rho(u)$ proportional to $\eta(u)$ by solving:

$$\arg\min_{\rho>0} \int_0^T \frac{\eta(u)}{\rho(u)}\,\mathrm{d}u \quad \text{subject to} \quad \int_0^T \rho(u)\,\mathrm{d}u = B, \tag{6}$$

which yields $\rho(u) \propto \sqrt{\eta(u)}$, specifying how the available queries are distributed across optimization. Section A.3 provides derivation details.

Active Camera Selection. Our caching mechanism reduces the number of model queries, yet it also reduces the number of camera observations during optimization. Therefore, we need to select the camera viewpoint of each query more carefully so that each new query improves the motion coverage of the cache as a whole. Under this objective, random camera sampling, as adopted in prior works (Lyu et al., 2026b; Poole et al., 2023), ignores the cached observations and may repeatedly expose the same motion while leaving other deformations occluded or poorly constrained. Instead, we select each new camera viewpoint according to the additional motion coverage it provides.

Specifically, at each refresh, we randomly sample a set of candidate cameras $\mathcal{V}$. Let $\mathcal{C}$ denote the cached viewpoints, and let $U(\mathcal{C})$ measure how well the viewpoint set $\mathcal{C}$ collectively observes the scene's motion. We select the candidate that maximizes the increase in U:

$$\mathbf{c}^\star = \arg\max_{\mathbf{c}\in\mathcal{V}} \left[U(\mathcal{C} \cup \{\mathbf{c}\}) - U(\mathcal{C})\right]. \tag{7}$$

To compute $U(\mathcal{C})$, we assess how well the cameras observe the motion associated with each control point over time. Inspired by the log-determinant criterion of D-optimal experimental design (Boyd & Vandenberghe, 2004), we define

$$U(\mathcal{C}) = \sum_{k,t} m_{k,t} \log(1 + n_{\text{eff}}(k, t; \mathcal{C})), \tag{8}$$

where k indexes control points, t indexes video frames, and $m_{k,t}$ denotes the associated motion magnitude. The effective observation count n_{eff} summarizes the observations available for each control point's motion, accounting for visibility, image-plane displacement, and redundancy between nearby views. Motion weighting prioritizes substantial movement, while the logarithm favors additional observations of motion that is not yet well observed. The effective observation count and implementation details are provided in Section A.4.

Efficient Target Replay. To further reduce the cost of fitting the scene to cached targets, we use gradient memory (Defazio et al., 2014; Schmidt et al., 2017) to estimate the full-cache fitting gradient from a small subset of targets at each iteration.

For each cache entry i, we store its most recently computed fitting gradient $\mathbf{h}_i$ with respect to the scene parameters θ, and maintain their mean $\overline{\mathbf{h}}$. At each iteration, we recompute gradients only for a subset $\mathcal{B}$ of cached targets. Their differences from the stored gradients correct the cached mean:

$$\hat{\mathbf{g}}_{\text{replay}} = \overline{\mathbf{h}} + \frac{1}{K} \sum_{i \in \mathcal{B}} \left(\nabla_\theta \mathcal{L}^{(i)}_{\text{target}} - \mathbf{h}_i \right). \tag{9}$$

The scene is updated using this estimated gradient $\hat{\mathbf{g}}_{\text{replay}}$, after which the selected entries' stored gradients and their mean are updated. This design allows the system to evaluate only K targets per iteration while retaining global gradient information from the full cache.

3.3 Semantics-aware Coloring for Robust Motion Optimization

The target cache supplies rich supervision efficiently, but this supervision is expressed through the appearance of rendered videos rather than explicit 3D correspondences. In prior works (Lyu et al., 2026b; Poole et al., 2023), to fit the cached targets, optimization adjusts the scene's deformation parameters to reduce differences between the rendered and target videos in RGB space. The input appearance therefore determines the visual cues that guide these adjustments, even when those cues do not clearly distinguish which scene regions should move toward the target.

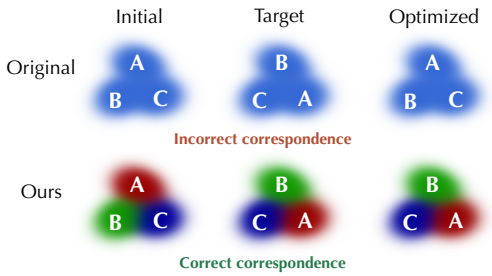

Figure 4: **Input appearance affects motion optimization.**

To further illustrate this problem, we show a toy example in Fig. 4: optimizing the locations of Gaussian blobs using photometric supervision. Each Gaussian represents a distinct region in 3D space, and the goal is to fit the initial scene to the target by optimizing the regions' positions. When the regions have identical colors, different arrangements produce the same rendered appearance. The reconstruction loss is zero despite incorrect region correspondences, providing no gradient to move the regions to their intended positions.

This ambiguity motivates visual cues that distinguish scene regions during target fitting. Since our goal is to optimize deformation, the appearance used for supervision need not match that of the final output. We therefore propose *semantics-aware coloring*, which assigns distinguishable colors to scene regions and uses them consistently during target generation and fitting. These colors make incorrect region substitutions distinguishable under the reconstruction loss.

Specifically, we first decompose each object into semi-rigid parts using a vision-language model agent, and the parts are used as the regions for color assignment. To retain shading cues that support the teacher's part-level motion guidance, we incorporate a compressed luminance field into the region colors: $\tilde{\mathbf{c}}(\mathbf{x}) = [(1-\beta) + \beta Y(\mathbf{x})]\mathbf{c}_{r(\mathbf{x})}$, where $Y(\mathbf{x})$ is the original luminance of Gaussian $\mathbf{x}$ and $r(\mathbf{x})$ denotes its region. This modulation retains spatial brightness structure with reduced local texture contrast that could otherwise hinder optimization.

We select region colors $\mathbf{c}_{r(\mathbf{x})}$ with the objective to maximize the minimum pairwise separation:

$$\max_{\{\mathbf{c}_r\}} \min_{r \neq s} d(\bar{m}_r \mathbf{c}_r, \bar{m}_s \mathbf{c}_s), \tag{10}$$

where $\bar{m}_r$ is the median luminance multiplier within region r, and d measures color differences. We optimize each object's palette using lexicographic refinement to prioritize the least-separated region pairs. We construct multiple color assignments, and cycle through them during target generation to provide complementary color contrasts and reduce ambiguities associated with any single palette. Each cached target retains its assignment for consistent rendering during replay.



Figure 5: **Qualitative comparisons.** We compare our method with baselines for scene-level motion generation. AAM denotes AnimateAnyMesh (Wu et al., 2025c).

4 Experiments

4.1 4D Scene Motion Generation

We evaluate scene-level motion generation on six scenes spanning diverse objects and interactions, each representing a different category of motion generation methods. We select 6 scenes spanning diverse scenarios for comparison. Specifically we compare our method with three baselines: AnimateAnyMesh (Wu et al., 2025c), UniWorld-View (Zhou et al., 2026), CHORD (Lyu et al., 2026a) and Codex (OpenAI, 2025) with GPT-6-Astra (OpenAI, 2026). AnimateAnyMesh is a rectified-flow model trained on mesh deformations and directly samples deformation sequences for given meshes conditioned on text at inference time. UniWorld-View uses estimated depth to guide multi-view video generation from an input video, followed by 4D reconstruction. For comparison, we first generate a video using Wan 2.2 and then apply UniWorld-View. CHORD generates scene-level motion by optimizing a hierarchical 4D representation under guidance distilled from video generation models. We additionally evaluate GPT-6-Astra as a baseline by prompting it to generate motion for the input scene, given its recently demonstrated capability in 3D modeling.

Qualitative Comparisons. Representative qualitative results are shown in Figure 5. Please refer to our provided website for the complete results. AnimateAnyMesh fail to generate motion that

Table 1: **Quantitative comparison with baselines.** We report user preferences, Semantic Adherence (SA) and Physical Commonsense (PC) scores from VideoPhy-2 (Bansal et al., 2025), and runtime.

	User Study		VideoPhy-2		Runtime ↓
	Alignment ↑	Realism ↑	SA ↑	PC ↑	
AnimateAnyMesh	1.47%	1.31%	2.83	**4.67**	**3 min**
UniWorld-View	0.82%	0.16%	3.00	3.67	53 min
GPT-6-Astra	15.03%	12.42%	3.67	3.92	65 min
CHORD	33.99%	31.37%	3.67	3.83	12 h
FlashCHORD	**48.69%**	**54.74%**	**3.75**	4.00	32 min

adheres to the prompt due to the lack of scene-level data during training. UniWorld-View captures the intended overall motion but exhibits severe geometric artifacts, which may arise from errors propagated through depth estimation, multi-view video generation, and 4D reconstruction. GPT-6-Astra generates motion that follows the text prompt with relatively few geometric artifacts. However, the motion is often stiff, and the resulting object poses can appear distorted and unnatural. CHORD also follows the motion prompts, but its reliance on the input appearance for supervision can lead to incorrect deformations and noticeable motion artifacts.

Quantitative Comparisons. We conduct a user study with 102 participants, asking them to indicate their preferences in terms of text alignment and motion realism. We additionally use VideoPhy-2 (Bansal et al., 2025) to evaluate Semantic Adherence (SA) and Physical Commonsense (PC). We also report runtime, with all methods evaluated on a single NVIDIA H200 GPU. The results are reported in Table 1. Our method achieves the highest user preference rates for both text alignment and motion realism. It also obtains the highest Semantic Adherence score and the second-highest Physical Commonsense score, indicating strong prompt alignment and plausible motion. AnimateAnyMesh obtains the highest Physical Commonsense score, but due to its tendency to leave objects largely static, as also noted by CHORD (Lyu et al., 2026a). Such outputs may appear physically plausible without performing the requested motion.

Our method generates scene-level motion in 32 minutes, achieving the highest user preference rates while running faster than UniWorld-View, GPT-6-Astra, and CHORD. Although AnimateAnyMesh is faster, our method offers substantially stronger text alignment and user-perceived motion quality.

4.2 Extensions and Applications

Scanned Asset Motion Generation. Our method also supports motion generation for scanned 3D assets. As shown in Figure 6, it produces coherent motion from real-world scans, extending its applicability beyond synthetic assets.



Figure 6: **Motion generation for scanned assets.**

Long-horizon Motion Generation. Our method supports long-horizon motion generation by freezing the final few frames of each clip and using them as the initial frames of the next, extending the motion across successive clips. Figure 1 presents a sequence of 13 clips. Based on CHORD's per-clip runtime, a sequence of this length would require approximately 6.5 days to generate. Our substantially reduced optimization time makes long-horizon scene generation more practical.

4.3 Ablation Studies

Cache-based Supervision. We ablate target replay, refresh scheduling, and active camera sampling to assess their contributions to effective supervision. As shown in Figure 7, removing any component degrades motion quality. Without target replay, each target is discarded after a single

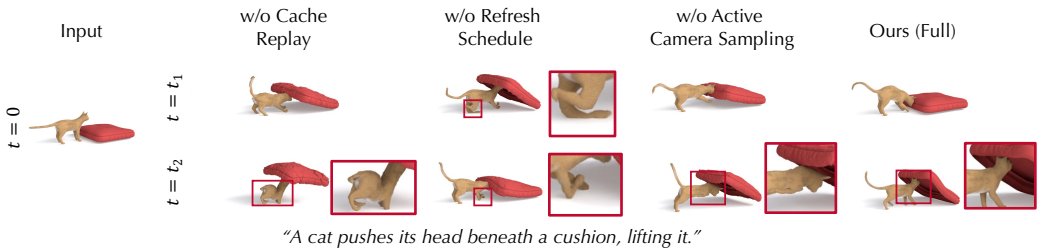

Figure 7: **Ablation of cache-based supervision.** Discarding targets after a single update, uniformly spacing refreshes, or uniformly sampling cameras degrades motion quality.

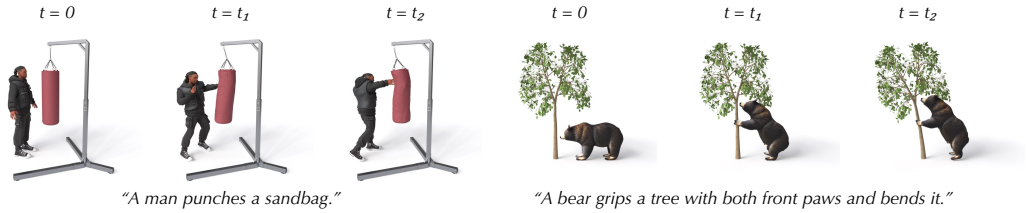

Figure 9: **Results with HunyuanVideo-1.5.** FlashCHORD generates coherent scene-level motion when using a different video model, demonstrating generalization across different base models.

update, leaving the acquired supervision underutilized and resulting in severe distortions. Replacing our refresh schedule with uniformly spaced refreshes under the same query budget also degrades the results, highlighting the importance of allocating supervision according to the progression of optimization. Finally, replacing active camera sampling with uniform sampling introduces noticeable deformation artifacts in the cat. With limited queries, occluded regions can remain insufficiently observed and poorly constrained, whereas active sampling provides complementary views of their motion. These ablations demonstrate the importance of coordinating target reuse, refresh timing, and viewpoint selection to obtain effective supervision from a limited number of teacher evaluations.

Semantics-aware Coloring. We evaluate semantics-aware coloring by removing it from the pipeline and using the original scene appearance for supervision. As shown in Figure 8, similar textures can make distinct regions ambiguous during motion fitting. In this example, the cowboy's hand is incorrectly deformed to fulfill the motion intended for a leg. Semantics-aware coloring distinguishes these regions, helping the intended body parts follow the target motion and removing the resulting artifacts.

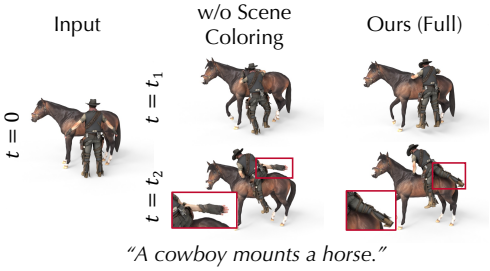

Figure 8: **Ablation for semantics-aware coloring.**

Video Model Ablation. To evaluate compatibility with different video generation models, we replace the default teacher with HunyuanVideo-1.5 (Wu et al., 2025a). As shown in Figure 9, our method continues to generate coherent scene-level motion, demonstrating that it is not restricted to a single video teacher.

5 Conclusion

In this paper, we present FlashCHORD, a pipeline for efficient, high-quality scene-level 4D motion generation. Our approach reduces expensive teacher evaluations by caching and reusing their supervision across multiple scene updates. We further identify a limitation of distillation-based motion generation: its dependence on the input appearance during deformation fitting. Our proposed deformation oriented texturing addresses this limitation by redesigning the supervisory appearance to better guide motion optimization. Limitations and future work are further discussed in Section C.

AI USE STATEMENT

We used generative AI tools to generate synthetic datasets; propose and refine hypotheses; design and provide feedback on research methods and experiments; implement methods; support qualitative and thematic data analysis; and assist with the interpretation of results. We also used generative AI tools to create or modify scientific images and figures, create or edit software code and other artifacts, and summarize or analyze existing literature.

We did not use generative AI tools to develop theoretical models or conceptual frameworks, assist with translation, or clean or reformat datasets. The use of generative AI for formulating mathematical claims, providing critical ingredients for mathematical proofs, or assisting with proof writing is not applicable to this work.

All AI-assisted content has been reviewed by the authors. We take full responsibility for the final content of this work, including all text, claims, code, figures, and other artifacts produced with the assistance of generative AI.

REFERENCES

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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7996–8006, 2024. 2, 3, 4

Sherwin Bahmani, Ivan Skorokhodov, Aliaksandr Siarohin, Willi Menapace, Guocheng Qian, Michael Vasilkovsky, Hsin-Ying Lee, Chaoyang Wang, Jiaxu Zou, Andrea Tagliasacchi, et al. Vd3d: Taming large video diffusion transformers for 3d camera control. In *International Conference on Learning Representations*, volume 2025, pp. 66712–66737, 2025. 3

Jianhong Bai, Menghan Xia, Xintao Wang, Ziyang Yuan, Zuozhu Liu, Haoji Hu, Pengfei Wan, and Di Zhang. Syncammaster: Synchronizing multi-camera video generation from diverse viewpoints. In *International Conference on Learning Representations*, volume 2025, pp. 58038–58060, 2025. 3

Hritik Bansal, Clark Peng, Yonatan Bitton, Roman Goldenberg, Aditya Grover, and Kai-Wei Chang. Videophy-2: A challenging action-centric physical commonsense evaluation in video generation. *arXiv preprint arXiv:2503.06800*, 2025. 8

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*, volume 1. Cambridge university press Cambridge, 2004. 5

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators. 2024. URL `https://openai.com/research/video-generation-models-as-world-simulators`. 3

Hongyuan Chen, Xingyu Chen, Zexiang Xu, and Anpei Chen. Motion 3-to-4: 3d motion reconstruction for 4d synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28947–28958, 2026. 2

Jianqi Chen, Biao Zhang, Xiangjun Tang, and Peter Wonka. V2m4: 4d mesh animation reconstruction from a single monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11643–11653, October 2025. 3

Wen-Hsuan Chu, Lei Ke, and Katerina Fragkiadaki. Dreamscene4d: Dynamic multi-object scene generation from monocular videos. *Advances in Neural Information Processing Systems*, 37: 96181–96206, 2024. 3

Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. *Advances in neural information processing systems*, 27, 2014. 6

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. 3

Inbar Gat, Sigal Raab, Guy Tevet, Yuval Reshef, Amit Haim Bermano, and Daniel Cohen-Or. Anytop: Character animation diffusion with any topology. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pp. 1–10, 2025. 2

Chen Geng, Yunzhi Zhang, Shangzhe Wu, and Jiajun Wu. Birth and death of a rose. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 26102–26113, 2025. 3, 4

Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation. *arXiv preprint arXiv:2404.02101*, 2024. 3

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. URL https://arxiv.org/abs/2006.11239. 4

Chen Hongyuan, Chen Xingyu, Youjia Zhang, Xu Zexiang, and Chen Anpei. Motion 3-to-4: 3d motion reconstruction for 4d synthesis. *arXiv preprint arXiv:2601.14253*, 2026. 3

Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. Scgs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4220–4230, 2024. 5, 15

Yanqin Jiang, Li Zhang, Jin Gao, Weiming Hu, and Yao Yao. Consistent4d: Consistent 360°dynamic object generation from monocular video. In B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun (eds.), *International Conference on Learning Representations*, volume 2024, pp. 51844–51861, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/e33a77969c26e5efdf3989ac46e7a036-Paper-Conference.pdf. 3

Oren Katzir, Or Patashnik, Daniel Cohen-Or, and Dani Lischinski. Noise-free score distillation. In *International conference on learning representations*, volume 2024, pp. 53344–53365, 2024. 3

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 3

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014. 4

Jiahui Lei, Yijia Weng, Adam W Harley, Leonidas Guibas, and Kostas Daniilidis. Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 6165–6177, 2025. 3

Zhiqi Li, Yiming Chen, and Peidong Liu. Dreammesh4d: Video-to-4d generation with sparse-controlled gaussian-mesh hybrid representation. *Advances in Neural Information Processing Systems*, 37:21377–21400, 2024. 3

Hanwen Liang, Yuyang Yin, Dejia Xu, Hanxue Liang, Zhangyang Wang, Konstantinos N Plataniotis, Yao Zhao, and Yunchao Wei. Diffusion4d: Fast spatial-temporal consistent 4d generation via video diffusion models. *arXiv preprint arXiv:2405.16645*, 2024a. 3

Yixun Liang, Xin Yang, Jiantao Lin, Haodong Li, Xiaogang Xu, and Yingcong Chen. Luciddreamer: Towards high-fidelity text-to-3d generation via interval score matching. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6517–6526. IEEE, 2024b. 3

Ting-Hsuan Liao, Haowen Liu, Yiran Xu, Songwei Ge, Gengshan Yang, and Jia-Bin Huang. Pad3r: Pose-aware dynamic 3d reconstruction from casual videos. *SIGGRAPH ASIA*, 2025. 3

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

Artem Lukoianov, Haitz S de Ocáriz Borde, Kristjan Greenewald, Vitor C Guizilini, Timur Bagautdinov, Vincent Sitzmann, and Justin Solomon. Score distillation via reparametrized ddim. *Advances in Neural Information Processing Systems*, 37:26011–26044, 2024. 3

Yanzhe Lyu, Chen Geng, Karthik Dharmarajan, Yunzhi Zhang, Hadi Alzayer, Shangzhe Wu, and Jiajun Wu. Choreographing a world of dynamic objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 32647–32658, June 2026a. 1, 2, 3, 4, 5, 7, 8, 15, 17

Yanzhe Lyu, Chen Geng, Karthik Dharmarajan, Yunzhi Zhang, Hadi Alzayer, Shangzhe Wu, and Jiajun Wu. Choreographing a world of dynamic objects. *arXiv preprint arXiv:2601.04194*, 2026b. 3, 5, 6

Ziqiao Ma, Xuweiyi Chen, Shoubin Yu, Sai Bi, Kai Zhang, Chen Ziwen, Sihan Xu, Jianing Yang, Zexiang Xu, Kalyan Sunkavalli, et al. 4d-lrm: Large space-time reconstruction model from and to any view at any time. *arXiv preprint arXiv:2506.18890*, 2025. 3

David McAllister, Songwei Ge, Jia-Bin Huang, David W Jacobs, Alexei A Efros, Aleksander Holynski, and Angjoo Kanazawa. Rethinking score distillation as a bridge between image distributions. *Advances in Neural Information Processing Systems*, 37:33779–33804, 2024. 3

Linzhan Mou, Jiahui Lei, Chen Wang, Lingjie Liu, and Kostas Daniilidis. Dimo: Diverse 3d motion generation for arbitrary objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14357–14368, 2025. 3

OpenAI. Codex is now generally available, October 2025. URL `https://openai.com/index/codex-now-generally-available/`. Accessed: 2026-09-19. 7

OpenAI. GPT-6 Astra: A New Generation of Intelligence. `https://openai.com/index/gpt-6-astra/`, September 2026. Accessed: 2026-09-19. 7

Björn Ottosson. A perceptual color space for image processing, 2020. URL `https://bottosson.github.io/posts/oklab/`. 17

Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. In *ICLR*. OpenReview.net, 2023. URL `https://openreview.net/forum?id=FjNys5c7VyY`. 3, 4, 5, 6

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. 3

Jiawei Ren, Kevin Xie, Ashkan Mirzaei, Hanxue Liang, Xiaohui Zeng, Karsten Kreis, Ziwei Liu, Antonio Torralba, Sanja Fidler, Seung Wook Kim, and Huan Ling. L4GM: Large 4d gaussian reconstruction model. *Advances in Neural Information Processing Systems*, 37:56828–56858, 2024. 3

Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1):83–112, 2017. 6

Qi Sun, Zhiyang Guo, Ziyu Wan, Jing Nathan Yan, Shengming Yin, Wengang Zhou, Jing Liao, and Houqiang Li. Eg4d: Explicit generation of 4d object without score distillation. *arXiv preprint arXiv:2405.18132*, 2024. 3

Qi Sun, Can Wang, Jiaxiang Shang, Wensen Feng, and Jing Liao. Animus3d: Text-driven 3d animation via motion score distillation. In *Proceedings of the SIGGRAPH Asia 2025 Conference Papers*, pp. 1–11, 2025. 3

Boshi Tang, Jianan Wang, Zhiyong Wu, and Lei Zhang. Stable score distillation for high-quality 3d generation. *arXiv preprint arXiv:2312.09305*, 2023. 3

Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. In *The Twelfth International Conference on Learning Representations(ICLR)*, 2024. 3

Lukas Uzolas, Elmar Eisemann, and Petr Kellnhofer. Motiondreamer: Exploring semantic video diffusion features for zero-shot 3d mesh animation. In *2025 International Conference on 3D Vision (3DV)*, pp. 893–904. IEEE, 2025. 3

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 3

Haochen Wang, Xiaodan Du, Jiahao Li, Raymond A Yeh, and Greg Shakhnarovich. Score jacobian chaining: Lifting pretrained 2d diffusion models for 3d generation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12619–12629. IEEE, 2023a. 3

Miaowei Wang, Qingxuan Yan, Zhi Cao, Yayuan Li, Oisin Mac Aodha, Jason J Corso, and Amir Vaxman. Bimotion: B-spline motion for text-guided dynamic 3d character generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10152–10164, June 2026. 2

Peihao Wang, Zhiwen Fan, Dejia Xu, Dilin Wang, Sreyas Mohan, Forrest Iandola, Rakesh Ranjan, Yilei Li, Qiang Liu, Zhangyang Wang, et al. Steindreamer: Variance reduction for text-to-3d score distillation via stein identity. *arXiv preprint arXiv:2401.00604*, 2023b. 3

Qianqian Wang, Vickie Ye, Hang Gao, Weijia Zeng, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9660–9672, 2025. 3

Yikai Wang, Xinzhou Wang, Zilong Chen, Zhengyi Wang, Fuchun Sun, and Jun Zhu. Vidu4d: Single generated video to high-fidelity 4d reconstruction with dynamic gaussian surfels. *Advances in Neural Information Processing Systems*, 37:131316–131343, 2024a. 3

Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. *Advances in neural information processing systems*, 36:8406–8441, 2023c. 3

Zhouxia Wang, Ziyang Yuan, Xintao Wang, Yaowei Li, Tianshui Chen, Menghan Xia, Ping Luo, and Ying Shan. Motionctrl: A unified and flexible motion controller for video generation. In *ACM SIGGRAPH 2024 Conference Papers*, pp. 1–11, 2024b. 3

Bing Wu, Chang Zou, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Jack Peng, Jianbing Wu, Jiangfeng Xiong, Jie Jiang, et al. Hunyuanvideo 1.5 technical report. *arXiv preprint arXiv:2511.18870*, 2025a. 9

Rundi Wu, Ruiqi Gao, Ben Poole, Alex Trevithick, Changxi Zheng, Jonathan T Barron, and Aleksander Holynski. Cat4d: Create anything in 4d with multi-view video diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 26057–26068, 2025b. 3

Zijie Wu, Chaohui Yu, Yanqin Jiang, Chenjie Cao, Fan Wang, and Xiang Bai. Sc4d: Sparse-controlled video-to-4d generation and motion transfer. In *European Conference on Computer Vision*, pp. 361–379. Springer, 2024. 3, 4, 5, 15

Zijie Wu, Chaohui Yu, Fan Wang, and Xiang Bai. Animateanymesh: A feed-forward 4d foundation model for text-driven universal mesh animation. pp. 13557–13568, 2025c. 2, 3, 7

Zijie Wu, Chaohui Yu, Fan Wang, and Xiang Bai. Animateanymesh++: A flexible 4d foundation model for high-fidelity text-driven mesh animation, 2026. URL https://arxiv.org/abs/2604.26917. 3

Yiming Xie, Chun-Han Yao, Vikram Voleti, Huaizu Jiang, and Varun Jampani. Sv4d: Dynamic 3d content generation with multi-frame and multi-view consistency. *arXiv preprint arXiv:2407.17470*, 2024. 3

Dejia Xu, Weili Nie, Chao Liu, Sifei Liu, Jan Kautz, Zhangyang Wang, and Arash Vahdat. Camco: Camera-controllable 3d-consistent image-to-video generation. *arXiv preprint arXiv:2406.02509*, 2024. 3

Zeyu Yang, Zijie Pan, Chun Gu, and Li Zhang. Diffusion^2: Dynamic 3d content generation via score composition of video and multi-view diffusion models. *arXiv preprint arXiv:2404.02148*, 2024. 3

Chun-Han Yao, Yiming Xie, Vikram Voleti, Huaizu Jiang, and Varun Jampani. Sv4d 2.0: Enhancing spatio-temporal consistency in multi-view video diffusion for high-quality 4d generation. pp. 13248–13258, 2025. 3

Jiraphon Yenphraphai, Ashkan Mirzaei, Jianqi Chen, Jiaxu Zou, Sergey Tulyakov, Raymond A Yeh, Peter Wonka, and Chaoyang Wang. Shapegen4d: Towards high quality 4d shape generation from videos. *arXiv preprint arXiv:2510.06208*, 2025. 3

Heng Yu, Chaoyang Wang, Peiye Zhuang, Willi Menapace, Aliaksandr Siarohin, Junli Cao, László Jeni, Sergey Tulyakov, and Hsin-Ying Lee. 4real: Towards photorealistic 4d scene generation via video diffusion models. *Advances in Neural Information Processing Systems*, 37:45256–45280, 2024. 2, 3, 4

Yu-Jie Yuan, Leif Kobbelt, Jiwen Liu, Yuan Zhang, Pengfei Wan, Yu-Kun Lai, and Lin Gao. 4dynamic: Text-to-4d generation with hybrid priors. *arXiv preprint arXiv:2407.12684*, 2024. 2, 3, 4

Bowen Zhang, Sicheng Xu, Chuxin Wang, Jiaolong Yang, Feng Zhao, Dong Chen, and Baining Guo. Gaussian variation field diffusion for high-fidelity video-to-4d synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12502–12513, 2025. 3

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. 3

Haiyang Zhou, Wangbo Yu, Chaoran Feng, Xunyu Zhou, Yonghong Tian, and Li Yuan. Uniworld-view: Large-baseline view synthesis via video diffusion models. *arXiv preprint arXiv:2608.04701*, 2026. 3, 7

Jensen Zhou, Hang Gao, Vikram Voleti, Aaryaman Vasishta, Chun-Han Yao, Mark Boss, Philip Torr, Christian Rupprecht, and Varun Jampani. Stable virtual camera: Generative view synthesis with diffusion models. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12405–12414. IEEE, 2025. 3

Zhizhuo Zhou and Shubham Tulsiani. Sparsefusion: Distilling view-conditioned diffusion for 3d reconstruction. In *2023 IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp. 12588–12597. IEEE, 2023. 4

Junzhe Zhu, Peiye Zhuang, and Sanmi Koyejo. Hifa: High-fidelity text-to-3d generation with advanced diffusion guidance. In *International Conference on Learning Representations*, volume 2024, pp. 5762–5786, 2024. 4

A METHOD DETAILS

A.1 SCENE REPRESENTATION

Similar to CHORD (Lyu et al., 2026a) and SC4D (Wu et al., 2024), we use a sparse set of control points (Huang et al., 2024) to drive the motion of the 3D Gaussians. To encourage temporally coherent motion while limiting the drift caused by fully cumulative transformations, we parameterize each control point using exponentially accumulated transformation residuals. Specifically, let $\Delta\mathbf{p}_{k,t}$ denote the learnable translation and rotation residual of control point k at frame t. Its effective deformation state is computed recursively as

$$\mathbf{p}_{k,t} = \alpha\mathbf{p}_{k,t-1} + \Delta\mathbf{p}_{k,t}, \qquad \mathbf{p}_{k,0} = \Delta\mathbf{p}_{k,0}, \tag{11}$$

or, equivalently,

$$\mathbf{p}_{k,t} = \sum_{j=0}^{t} \alpha^{t-j}\Delta\mathbf{p}_{k,j}, \tag{12}$$

where $\alpha \in [0, 1]$ controls the temporal persistence of each residual. We use $\alpha = 0.9$ in all experiments. Thus, a residual at one frame influences subsequent frames with exponentially decaying strength, coupling nearby frames while preventing motion errors from accumulating indefinitely.

A.2 PROOF OF THE RECONSTRUCTION-LOSS FORMULATION OF SDS

For the detached pseudo-target $\mathbf{z}^\star = \mathrm{sg}[\boldsymbol{\epsilon} - \hat{\mathbf{v}}(\mathbf{z}_\tau; \tau, \mathbf{y})]$, consider the reconstruction loss

$$\mathcal{L}_{\text{target}} = \frac{1}{2}\left\|\mathbf{z}_{\theta,\mathbf{c}} - \mathbf{z}^\star\right\|_2^2. \tag{13}$$

Since $\mathbf{z}^\star$ is detached, its derivative with respect to θ is zero. Applying the chain rule gives

$$\begin{aligned}\nabla_\theta\mathcal{L}_{\text{target}} &= \left(\frac{\partial\mathbf{z}_{\theta,\mathbf{c}}}{\partial\theta}\right)^{\mathsf{T}}(\mathbf{z}_{\theta,\mathbf{c}} - \mathbf{z}^\star) \\ &= \left(\frac{\partial\mathbf{z}_{\theta,\mathbf{c}}}{\partial\theta}\right)^{\mathsf{T}}(\hat{\mathbf{v}}(\mathbf{z}_\tau; \tau, \mathbf{y}) - \boldsymbol{\epsilon} + \mathbf{z}_{\theta,\mathbf{c}}).\end{aligned} \tag{14}$$

At target construction, this is exactly the sampled rectified-flow SDS gradient. Taking the expectation over $\tau \sim \hat{w}(\tau)$ and $\boldsymbol{\epsilon}$ recovers Eq. (2).

A.3 DERIVATION OF THE REFRESH SCHEDULE

The refresh schedule allocates a fixed teacher-query budget according to the prescribed learning rate. The reuse penalty of a target is modeled as proportional to the learning rate accumulated since its creation, serving as a proxy for the optimization performed while the target remains fixed.

Consider a reuse interval of length d beginning at optimization time b. Assuming the learning rate is approximately constant within the interval, the penalty after elapsed time s is $\eta(b)s$. Integrating over the interval gives

$$C(b, d) \approx \int_0^d \eta(b)s\, ds = \frac{\eta(b)d^2}{2}. \tag{15}$$

Let $J[\rho]$ denote the total reuse penalty over the optimization horizon. At refresh rate $\rho(u)$, each reuse interval has length approximately $1/\rho(u)$ and incurs a cost of $\eta(u)/(2\rho(u)^2)$. Since there are $\rho(u)$ such intervals per unit optimization time, the total cost is

$$J[\rho] \approx \int_0^T \rho(u)\frac{\eta(u)}{2\rho(u)^2}\, du = \frac{1}{2}\int_0^T \frac{\eta(u)}{\rho(u)}\, du. \tag{16}$$

Over an optimization horizon T with continuous refresh budget B, the allocation solves

$$\min_{\rho>0} \frac{1}{2}\int_0^T \frac{\eta(u)}{\rho(u)}\, du \quad \text{subject to} \quad \int_0^T \rho(u)\, du = B. \tag{17}$$

Introducing a Lagrange multiplier λ and differentiating with respect to $\rho(u)$ yields

$$-\frac{\eta(u)}{2\rho(u)^2} + \lambda = 0. \tag{18}$$

Normalizing to satisfy the budget constraint gives

$$\rho^\star(u) = \frac{B\sqrt{\eta(u)}}{\int_0^T \sqrt{\eta(s)}\, ds}. \tag{19}$$

Since $\eta(u)/\rho(u)$ is strictly convex in $\rho(u)$ for $\eta(u) > 0$, this stationary solution minimizes the stated continuous objective. Thus, refresh frequency scales with the square root of the learning rate, allocating more queries to early, higher-learning-rate stages and permitting longer reuse during refinement.

A.4 Details of Active Camera Selection

At each cache refresh, 64 candidate cameras are sampled uniformly in azimuth over $[0^\circ, 360^\circ)$ and elevation over $[-50^\circ, 0^\circ]$ under our camera convention. This viewing domain spans a solid angle of $\Omega \approx 4.81$ steradians. Both candidate and cached viewpoints are evaluated on the current scene deformation, excluding the oldest cache entry if it will be evicted.

To estimate scene motion efficiently, representative points are sampled from the undeformed scene and tracked through the full deformation model at six evenly spaced video frames. For scoring, each point is assigned to its nearest control point within the same object and control-point layer. Let $\mathcal{P}_u$ denote the points assigned to control point u, and $\Delta\mathbf{x}_{p,t}$ the displacement of point p between consecutive sampled frames. The motion magnitude used in the main-text objective is

$$m_{u,t} = \frac{1}{|\mathcal{P}_u|} \sum_{p \in \mathcal{P}_u} \|\Delta\mathbf{x}_{p,t}\|_2. \tag{20}$$

This average measures surface movement, including displacement induced by rotation, without favoring control points merely because they have more associated samples.

While $m_{u,t}$ measures how much the surface moves, the observation strength $q_{\mathbf{c}}(u,t)$ measures how well camera $\mathbf{c}$ observes that motion. Depth buffering identifies points visible at the start of each interval. For these points, displacement is projected onto the plane perpendicular to the viewing ray, excluding motion directly toward or away from the camera. The sum of the projected displacement magnitudes is divided by the total 3D displacement magnitude summed over all associated points, including occluded points. The resulting fraction is multiplied by a clip-level visibility weight, computed by aggregating object visibility according to each object's share of scene motion and taking the product across sampled intervals. This favors views that maintain visibility of the moving objects throughout the clip. No additional teacher inference is required.

To aggregate these observations without counting nearby views as fully independent, the effective observation count is defined as

$$n_{\text{eff}}(u,t;\mathcal{C}) = \frac{\left(\sum_{i\in\mathcal{C}} q_i(u,t)\right)^2}{\sum_{i,j\in\mathcal{C}} q_i(u,t) q_j(u,t) \kappa_{ij}}, \tag{21}$$

where $\kappa_{ii} = 1$ and $\kappa_{ij} = a\exp(-\theta_{ij}^2/(2\lambda^2))$ for distinct entries. Here, θ_{ij} is the angle between viewing directions. Equally strong, independent observations contribute their full count, whereas nearby, correlated views contribute less independent evidence. We use $a = 0.2$ and set $\lambda = \sqrt{\Omega/(\pi N_{\text{cache}})}$, where N_{cache} is the cache capacity. This scale approximates the angular radius of one cache entry's share of the viewing domain, giving $\lambda \approx 17.7^\circ$ for a capacity of 16. The effective count is zero when no observations are available.

Each candidate is scored by the increase it produces in the motion-weighted logarithmic utility defined in the main text. The highest-scoring camera is selected, with uniform sampling as a fallback when no candidate provides a positive gain.

Before appreciable motion emerges, the motion-based criterion is replaced by a first-frame visibility criterion. Each object's visible fraction is normalized by its maximum across the candidate views,

and the camera maximizing the minimum normalized visibility across objects is selected. This initializes the cache with clear observations of the participating objects before switching to motion-based selection.

A.5 Details of Deformation-Oriented Texturing

Part decomposition is performed once during preprocessing. The source mesh is first partitioned into surface patches. A multimodal agent groups these patches into coherent object parts using textured multi-view renders, patch visualizations, and geometric statistics. The resulting assignments are checked through rendered part visualizations and revised where necessary. After aligning the mesh with the Gaussian representation, part labels are transferred to the scene points by nearest-vertex assignment.

Color assignments are constructed in the undeformed scene and remain attached to the corresponding scene points throughout motion optimization. Each object is assigned a separate hue interval, providing a shared color family for its parts while distinguishing them from other objects. Region colors are selected within this interval to introduce part-level contrast without disrupting object-level visual coherence.

For each point, the original RGB appearance is converted to luminance as $Y = 0.299R + 0.587G + 0.114B$. The assigned region color is multiplied by $m = (1-\beta) + \beta Y$, with $\beta = 0.65$. The median multiplier within each region is used when evaluating candidate color separations.

The distance function d is the Euclidean distance between colors after conversion to OKLab (Ottosson, 2020), a color space. The palette solver uses farthest-first initialization followed by coordinate-wise lexicographic refinement. For each region, candidate replacements are compared by sorting the pairwise distances from smallest to largest. A replacement is accepted if it increases the first differing distance, prioritizing the least-separated pairs without sacrificing smaller distances to improve larger ones. Refinement terminates when no improving replacement is found or after 12 sweeps.

Six alternative color assignments are constructed using different initializations and hue-interval placements, then cycled during target generation. Each cached target retains its assignment, which is restored when rendering the current scene for replay. Consequently, target generation and fitting use consistent supervisory appearances even when different cache entries use different assignments. The original appearance is retained separately and used to render the final optimized motion.

A.6 Additional Implementation Details

Our method is implemented on CHORD's codebase (Lyu et al., 2026a) and retains its ARAP, temporal, and ground-penetration regularization losses. We adopt its learning-rate schedule with adjustments for a compressed optimization timeline. Specifically, the first 500 original iterations are compressed into 250 updates, with the scheduled learning rate doubled to compensate for the reduced update count. The remaining iterations is compressed by a factor of five, while retaining the scheduled learning rate without compression compensation. Learning rates and regularization schedules are evaluated at the corresponding original iteration indices rather than the compressed update indices. In total we optimize for 550 iterations.

The FIFO cache holds 16 pseudo-targets, and the target-generation budget is set to 360. Supervision uses videos at 832×480 resolution, beginning with 21 frames and increasing to 41 frames at iteration 100 to further improve efficiency. The cache is cleared and refilled at this transition to maintain compatible target dimensions.

B Additional Experimental Results

B.1 Video Generation from Recolored Scenes

Deformation-oriented texturing introduces appearances that may differ from the video model's training distribution, raising the question of whether the model can still generate the intended motion from recolored scenes. Figure 10 shows videos generated by HunyuanVideo-1.5 and Wan 2.2, conditioned on a recolored scene. Both models generate the required action while retaining the scene's



Figure 10: **Video generation from a recolored scene.** Given the recolored input and the prompt "A man mounts a horse," both HunyuanVideo-1.5 and Wan 2.2 generate the requested interaction despite the modified appearance.

Table 2: **Raw results of the user study on scene-level 4D motion generation.** For each scene and metric, 102 participants each selected the method they considered best. We report vote counts and their averages across six scenes.

Metric	Scene	AnimateAnyMesh	UniWorld	GPT-6	CHORD	Ours	Total
Prompt Alignment	Cat rising on a cushion	2	1	16	48	35	102
	Man sitting on a rock chair	0	2	8	43	49	102
	Two men performing a high five	0	1	3	45	53	102
	Cowboy mounting a horse	1	0	13	26	62	102
	Man sitting on an exercise ball	3	1	21	13	64	102
	Sandbag falling onto a trampoline	3	0	31	33	35	102
	Avg.	1.50	0.83	15.33	34.67	49.67	102
Motion Realism	Cat rising on a cushion	5	0	11	41	45	102
	Man sitting on a rock chair	0	0	3	31	68	102
	Two men performing a high five	1	0	0	47	54	102
	Cowboy mounting a horse	0	1	1	20	80	102
	Man sitting on an exercise ball	0	0	34	19	49	102
	Sandbag falling onto a trampoline	2	0	27	34	39	102
	Avg.	1.33	0.17	12.67	32.00	55.83	102

broad color organization. These examples demonstrate that the modified appearance remains compatible with motion generation across different video models, supporting its use for supervisory target construction.

B.2 Runtime Breakdown

Figure 11 reports the runtime breakdown of our method across target generation, cache replay, regularization, and other operations. The latter includes camera selection, control-point initialization, color preparation, and optimizer updates. Target generation remains the largest component, accounting for 47.0% of the optimization time despite the reduced number of video-model queries. Cache replay accounts for 30.2%, reflecting the cost of repeatedly rendering and fitting the scene to stored targets. Regularization contributes 13.6%, while other operations account for the remaining 9.2%. These results show that target reuse substantially reduces reliance on expensive model inference, although target generation remains the main computational cost.

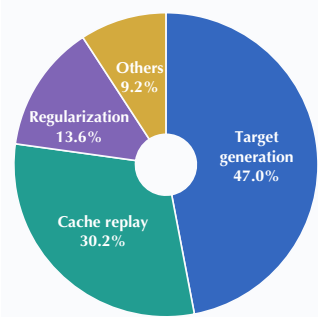

Figure 11: **Runtime breakdown.**

B.3 Runtime across Different GPUs

We evaluate the runtime of our method on additional GPU architectures. Using a single GPU, our method takes 45.79 minutes on an NVIDIA RTX PRO 6000, 68 minutes on an NVIDIA A100, and 85 minutes on an NVIDIA L40S. These results demonstrate that our method can also be deployed on GPUs beyond the H200, with runtime varying according to the available hardware.

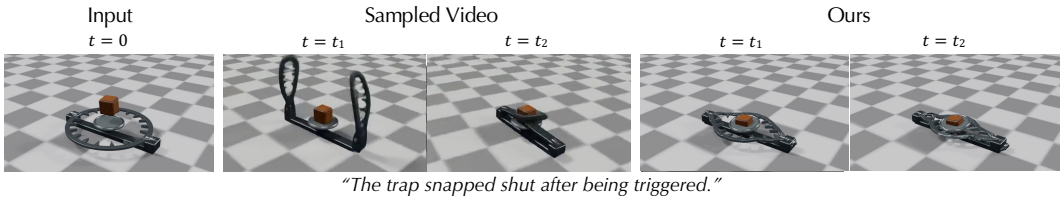

Figure 12: **Failure case.** Given the prompt "The trap snapped shut after being triggered," the video model generates an incorrect closing motion. Our method also fails to recover the intended mechanism, illustrating its dependence on reliable video-model guidance.

B.4 Full User Study Results

Table 2 reports the number of participants who selected each method as the best for each scene and metric. Ours receives the most votes for prompt alignment in five of the six scenes and for motion realism in all six scenes. Averaged across scenes, ours receives 48.69% of prompt-alignment votes and 54.74% of motion-realism votes, exceeding CHORD, the strongest baseline overall, by 14.71 and 23.37 percentage points, respectively.

C Limitations and Future Work

As a method that distills supervision from video generation models, our capabilities are inherently bounded by those of the underlying video model. Consequently, motions or interactions that fall outside the teacher's training distribution and cannot be generated reliably by the teacher also remain challenging for our method, as illustrated in Figure 12.

A second source of failure arises from our 4D representation. Since we model dynamics by deforming the initial scene, the representation cannot naturally handle objects that newly appear over time. For the same reason, it is also poorly suited to fluids, whose motion can involve large topological changes that are difficult to express as deformations of the initial geometry.

Therefore, one promising direction for future work is to incorporate stronger video generation models with broader motion priors, together with more expressive 4D representations that can accommodate topology changes and the appearance or disappearance of geometry over time.