

3D Video Models through Point Tracking, Reconstructing and Forecasting

Wen-Hsuan Chu

CMU-RI-TR-25-79

August, 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:

Katerina Fragkiadaki, *chair*

Kris Kitani

Shubham Tulsiani

Kosta Derpanis (York University)

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Robotics.*

Copyright © 2025 Wen-Hsuan Chu. All rights reserved.

Abstract

From autonomous driving to household robotics and embodied AI, intelligent systems must build internal models of the world: mental representations that allow them to perceive their surroundings, reason about unobserved structure, and plan future interactions. Humans do this effortlessly from vision alone, but enabling machines to do the same from raw video remains a core challenge. Sparse viewpoints, occlusions, and complex dynamics make it difficult to recover mental models of scenes that are both geometrically accurate and physically actionable. This thesis addresses this challenge by developing a unified framework that allows agents to *see* the world, *imagine* its hidden structure and dynamics, and *plan* interactions within it, ultimately building **mental world models** from monocular videos.

To see and imagine the video observation in 4D, we develop DreamScene4D, an optimization-based pipeline that fuses learned object tracking priors and generative image priors with 4D Gaussian splatting to decompose video into object-centric 3D shapes and motion trajectories. By independently optimizing the 3D Gaussians of each object to minimize both the rendering error and the score-distillation sampling loss, and then composing the Gaussians post-optimization, our method yields accurate geometry and precise motion estimates, outperforming prior NeRF-based and Gaussian-splatting-based approaches in complex real-world videos.

Although DreamScene4D’s decomposition enables precise reconstruction and completion of individual objects, its post hoc composition can lead to inconsistencies in spatial layout and depth ordering of the video scene. To address this, we introduce GenMOJO, a joint multi-object optimization framework that unifies scene-level rendering with object-centric imagination. By optimizing all objects together to match full-scene observations, GenMOJO faithfully preserves inter-object spatial relationships and occlusion structure. Simultaneously, it applies score-distillation-sampling in local object frames to hallucinate unseen geometry and motion. This combination of global supervision and localized generative priors yields coherent, high-fidelity 4D representations that outperform per-object methods in both visual quality and trajectory accuracy.

Finally, we present ParticleWorldDiffuser, a transformer-based diffusion

model for forecasting 3D particle trajectories in learned scene representations. The action-conditioned variant predicts object displacements given control sequences, enabling model-predictive control, while the unconditional variant generates both action and object trajectories via score-based sampling for efficient, rollout-free planning. Trained in simulation but evaluated on real-world object reconstructions, our model generalizes across novel objects and scenes, demonstrating effective sim-to-real transfer.

Together, these contributions form a cohesive pipeline for generating a mental world model of the video observation. By grounding generative models in real video and validating their utility for physical prediction, this thesis takes a step toward embodied systems that can perceive and interact with the world through the lens of video, a capability essential for the next generation of intelligent agents.

Acknowledgments

I would like to express my deepest gratitude to everyone who has supported me throughout the journey of this thesis.

I am profoundly grateful to my advisor, Professor Katerina Fragkiadaki, for taking me as a student and for their unwavering guidance and insightful feedback. Their expertise and patience have been invaluable in shaping this research and in helping me grow as a researcher. I have learned a great deal from them over the years about selecting impactful research directions, devising innovative solutions, and presenting my work.

In particular, I also owe a special debt of gratitude to my undergraduate advisor, Professor Yu-Chiang Frank Wang, and my master’s advisor, Professor Kris Kitani. Their mentorship introduced me to rigorous research practices, inspired my curiosity, and laid the foundation for my development as a researcher. Without their early support and guidance, I would not have been in this position in the first place.

Next, I would like to thank the members of my thesis committee, Professor Kris Kitani (double shout-out!), Professor Shubham Tulsiani, and Professor Kosta Derpanis, for their thoughtful comments, constructive critiques, and generous time. Their diverse perspectives significantly improved the rigor and clarity of this work.

During my time at CMU, I have also met an extraordinary number of great people. My thanks go to my colleagues and friends, Adam Harley, Xian Zhou, Nikos Gkanatsios, Gabe Newell, Brian Yang, Ayush Jain, Alexander Swerlow, Tsung-Wei Ke, Lei Ke, Yunchu Zhang, Mihir Prabhudesai, Kashu Yamazaki, Matthew Bronars, Jianren Wang, Yu-Jhe Li, Yunze Man, Shun Iwase, and many others, for many stimulating discussions, collaborative coding sessions, and moral support. I am also grateful to my broader peer community for sharing ideas, data, and jokes that lightened up the research process.

Lastly, I owe my deepest appreciation to parents, Tung-Yuan and Ching-Wen, for their unconditional love, patience, and encouragement. Their belief in me has been a constant source of strength and motivation.

Thank you to everyone who helped make this work possible.

Contents

1	Introduction	11
2	Dynamic Multi-Object Scene Generation from Monocular Videos	17
2.1	Introduction	17
2.2	Related Work	19
2.2.1	Video-to-4D Reconstruction	19
2.2.2	Video-to-4D Generation	20
2.3	Method	20
2.3.1	Background: Generative 3D Gaussian Splatting	21
2.3.2	DreamScene4D	21
2.3.3	Video Amodal Completion	22
2.4	Implementation Details	28
2.4.1	Deformation Network.	28
2.4.2	Learning Rate.	28
2.4.3	Densification and Pruning.	29
2.4.4	Running Time.	29
2.5	Experiments	29
2.5.1	Datasets	29
2.5.2	Evaluation Metrics	30
2.5.3	Video to 4D Scene Generation	30
2.5.4	4D Gaussian Motion Accuracy	31
2.6	Limitations	33
2.7	Conclusion	34
3	Generative 4D Scene Gaussian Splatting with Object View-Synthesis Priors	39
3.1	Introduction	39
3.2	Related work	41
3.3	4D Generation from Monocular Video	44
3.3.1	Preliminaries: Generative Gaussian Splatting	44
3.3.2	GenMOJO	45
3.4	MOSE-PTS Dataset	48
3.4.1	Point Trajectory Analysis.	48
3.4.2	Annotation Procedure	49

3.4.3	Point Track Difficulty Analysis	50
3.5	Implementation Details	51
3.5.1	Static Gaussian Optimization	52
3.5.2	Dynamic Gaussian Optimization	52
3.6	Experiments	53
3.6.1	Video to 4D Scene Generation	53
3.6.2	4D Gaussian Motion Accuracy	56
3.6.3	Ablations	58
3.6.4	Limitations	60
3.7	Conclusion	61
4	Generative 3D Particle World Models	63
4.1	Introduction	63
4.2	Related Work	66
4.2.1	Learning World Dynamics	66
4.2.2	Planning with diffusion models	67
4.3	Generative 3D Particle World Models	67
4.3.1	Preliminaries: Diffusion Models	68
4.3.2	ParticleWorldDiffuser	69
4.4	Curating a Dataset of Diverse Robot-Object Interaction Trajectories in a Physics Engine	73
4.4.1	Data Generation Details for Dynamics Prediction Experiments	75
4.4.2	Data Generation Details for Motion Planning Experiments . .	75
4.5	Experiments	76
4.5.1	Action-Conditioned 3D Particle Trajectory Prediction	76
4.5.2	Planning for Object Manipulation	83
4.5.3	Runtime Analysis	85
4.6	Limitations	85
4.7	Conclusion	86
5	Discussion and Future Directions	89
A	User Study Procedure for DreamScene4D and GenMOJO	93
	Bibliography	97

List of Figures

2.1	DreamScene4D Teaser Figure.	18
2.2	Method overview for DreamScene4D	24
2.3	3D Motion Factorization.	26
2.4	Video to 4D Comparisons.	35
2.5	Motion Comparisons.	36
2.6	Gaussian Motion Visualizations.	37
2.7	Mitigating the parallax effect.	37
2.8	Failure Cases.	38
3.1	GenMOJO Teaser Figure.	40
3.2	Overview of GenMOJO.	43
3.3	Statistical comparison between MOSE-PTS and Tap-Vid-DAVIS. . .	48
3.4	Sample annotation from MOSE-PTS.	49
3.5	Annotation Interface.	51
3.6	Video to 4D Scene Generation Comparisons.	54
3.7	Tracking Comparisons.	58
3.8	Failure cases.	59
4.1	Architecture of ParticleWorldDiffuser.	69
4.2	Simulation Data Generation.	74
4.3	Qualitative Sim2real Results.	77
4.4	MSE Loss between GT and Predicted Actions.	79
4.5	Prediction MSE versus timestep.	80
4.6	Long Video Predictions.	81
4.7	Diversity of Planned Actions.	82
4.8	MPC vs Guided Diffusion for Motion Planning.	84
4.9	Motion Planning via Guidance.	85
A.1	User survey interface.	95

List of Tables

2.1	Video to 4D Scene Generation Comparisons.	32
2.2	Gaussian Motion Accuracy.	33
3.1	Point tracking performance of CoTrackerV2 and CoTrackerV3 on various datasets.	51
3.2	Video to 4D Scene Generation Comparisons.	56
3.3	Point tracking comparison between 4D generation / reconstruction methods.	57
3.4	Ablation Experiments.	60
4.1	Evaluation in Simulation Data.	76
4.2	Model Performance vs. Number of Samples (Best-of-K Selection). . .	81
4.3	Planning Comparisons.	83

Chapter 1

Introduction

Humans live in a complex three-dimensional world. As such, we inherently perceive and interact with the world in three dimensions, a skill that is deeply ingrained in our cognitive processes and sensory experiences throughout evolution. From early childhood, individuals develop an intuitive understanding of spatial relationships, depth, motion, and physics through continuous interaction with their environment [105]. This natural aptitude enables humans to navigate complex spatial layouts, anticipate the movements of objects, and perform intricate tasks with remarkable ease. As a result, our cognitive models and reasoning abilities are fundamentally tuned to process and interpret three-dimensional information.

In recent years, computer vision has experienced rapid growth, significantly driven by advances in deep learning. However, much of this progress has been confined to two-dimensional representations. Deep learning methods, when fueled by large-scale datasets of 2D images and videos, can achieve unprecedented performance in discriminative tasks such as image classification [24, 35, 70], object detection [36, 67, 144], semantic segmentation [55, 88], and visual tracking [89], as well as generative tasks such as image [92] and video synthesis [13]. While these approaches have demonstrated remarkable capabilities, they inherently lack the explicit three-dimensional understanding that underpins human perception and interaction. As a consequence, despite excelling at interpreting large amounts of visual data, these models do not yet possess the human-like spatial reasoning required to reliably interact with the physical world.

1. Introduction

Bridging this gap to three-dimensional understanding is critical, but has progressed more slowly, largely due to the comparative scarcity of large-scale annotated 3D data and the inherent complexity of directly modeling spatial and temporal relationships in three dimensions. Recent advancements in multiview methods have significantly improved the accuracy and realism of 3D reconstruction from multiple 2D images. These methods leverage multiple camera views, whether overlapping [113] or non-overlapping [96], to generate accurate and detailed 3D representations of environments and objects. Furthermore, deep learning methods have begun to revolutionize this space by enabling direct inference of 3D information from limited viewpoints using learned priors, significantly mitigating constraints posed by traditional geometry-based techniques. State-of-the-art neural models can reliably estimate depth maps [50, 129], infer surface normals [40], and even predict complete 3D meshes [65, 108] from single or sparse monocular images. At a higher level of abstraction, recent volumetric and point-cloud-based approaches have notably improved tasks such as 3D object detection [97] and semantic segmentation [68]. Additionally, dynamic 3D understanding from video sequences has shown promise, initially focusing on class-specific domains with well-defined priors, such as articulated pose tracking of human skeletons and hands [30], and more recently extending toward class-agnostic frameworks for 6-DOF rigid object pose estimation [120] and fine-grained point-level 3D motion tracking [124]. However, these models predominantly emphasize isolated components of perception rather than integrating perception with higher-level cognitive functions essential for practical real-world interactions.

Despite impressive progress discussed above, current AI systems understand the world in a fundamentally different way from humans. They excel in 2D perception [13, 88, 92], static 3D reconstruction [52, 77], novel view synthesis [66], or model-based planning [100], but struggle to integrate perception, imagination, and planning in a unified framework. In contrast, in the mind of a human acting in the physical world, the world is understood as a mental model of the external reality and of its possible actions [19]. We *see* the present scene in 3D, *imagine* the unobserved structure consistent with past and future views, as well as the potential behavior of interactions available to the agent, and *plan* actions accordingly that bring about the desired future outcome.

We posit that to create agents and algorithms that can effectively interpret

and interact with their surroundings, we must bridge the gap between human-like spatial cognition and current artificial intelligence. By developing models that can understand 3D structures and dynamic movements from video data, we aim to replicate this natural human capability, enabling machines to perform tasks that require a sophisticated comprehension of spatial geometry and motion dynamics, which are both ambiguous when represented in 2D. This is crucial for a wide range of applications, including autonomous driving [26, 101], embodied agents [28, 42], and robotics [98, 100, 143].

In this thesis, we identify three complementary capabilities to build such mental world models: (i) **See**: accurate reconstruction from monocular video into persistent 4D geometry; (ii) **Imagine**: high-fidelity generative modeling of dynamic multiobject scenes in 4D, and being able to forecast dynamics over time; and (iii) **Plan**: determine optimal control sequences within the reconstructed or imagined world model. Each capability supplies inductive biases or supervisory signals that the others lack: accurate 4D geometry constrains imagination; the understanding of dynamics constrains planning; and planning objectives expose where the geometry must be precise (e.g., contact-rich regions).

To effectively realize these intertwined capabilities in practice, we structure our approach around two interconnected research directions. The first direction, **Generative 4D Scene Modeling**, focuses primarily on addressing the *see* and *imagine* aspects by reconstructing accurate 3D geometries and motion from monocular observations and generating plausible completions for unseen or occluded regions. The resulting complete and coherent 4D scene representations lay the foundation for agents to reason and interact with their environment beyond the immediate visible data. The second direction, **3D World Models for Forecasting and Planning**, leverages 4D scene representations to learn the dynamics of objects and their interactions in a data-driven manner, using particle-based diffusion models. This enables the agent to accurately predict future states conditioned on possible actions (*imagine*) and to determine optimal action sequences that achieve the desired outcomes (*plan*). Together, these two lines of work form a coherent pipeline that integrates perception, imagination, and action planning into a unified, human-like spatial reasoning framework.

In the following chapters, we ground these conceptual ideas into concrete technical

contributions. We introduce and thoroughly evaluate novel methods within our two main research directions, demonstrating how the principles of seeing, imagining, and planning can be realized and integrated into coherent computational frameworks. Each chapter incrementally builds upon the previous work, validating our claims with extensive experiments and ablations, and illustrating the practical implications of our approaches for embodied intelligence and autonomous systems.

Chapter 2 discusses how we can utilize data-driven object and point tracking priors to improve 4D Gaussian Splatting [90, 121], an optimization-based method, to obtain a better 3D and motion representation of a video scene under a monocular view setting. Specifically, we propose DreamScene4D, which decomposes a video scene into multiple objects and estimates the complete 3D geometry and motion of these objects independently. The motion of the objects is factorized into three components to handle objects that undergo fast motion or complex motion: 1) a camera motion, 2) an object-centric deformation, and 3) an object-centric to world frame transformation. This allows us to incorporate tracking priors to estimate the third component and existing camera pose estimation methods to estimate the first component, thereby significantly reducing the complexity of the optimization, as it only needs to handle the remaining object-centric deformation motion.

Chapter 3 extends DreamScene4D into a joint multiobject optimization framework. By optimizing all foreground objects and the background together under rendered mask supervision, score distillation sampling losses, and temporal spherical harmonic color adjustments, GenMOJO captures inter-object interactions and lighting variations, yielding high-fidelity 4D scene representations that outperform per-object methods.

Chapter 4 proposes a memory-efficient transformer-based diffusion model for 3D particle dynamics and planning. We develop two variants, one action-conditioned and one unconditional, that forecast object-particle trajectories from robot action sequences or the scene alone, respectively. The action-conditioned model performs natural dynamics predictions by autoregressively rolling out the model based on an input action sequence. The unconditional model can be used for planning by guiding the reverse diffusion process with score-based gradients. Our model enables both model-predictive control rollouts and fast, rollout-free planning with sim-to-real generalization.

Finally, in Chapter 5, we synthesize lessons from our optimization-based scene generative models and data-driven dynamics models to identify key challenges and chart a path forward.

1. Introduction

Chapter 2

Dynamic Multi-Object Scene Generation from Monocular Videos

2.1 Introduction

Videos are the result of entities moving and interacting in 3D space and over time, captured from a moving camera. Inferring the dynamic 4D scene from video projections in terms of complete 3D object reconstructions and their 3D motions across seen and unseen camera views is a challenging problem in computer vision. It has multiple essential applications, such as 3D object and scene state tracking for robot perception [38, 80], action recognition, visual imitation, digital content creation/simulation, and augmented reality.

Video-to-4D is a highly under-constrained problem since multiple 4D generation hypotheses project to the same video observations. Existing 4D reconstruction works [11, 58, 64, 71, 77, 83] mainly focus on the visible part of the scene contained in the video by learning a differentiable 3D representation that is often a neural field [74] or a set of 3D Gaussians [52] with temporal deformation. *What about the unobserved views of the dynamic 3D scene?* Existing 4D generation works utilize generative models to constrain the appearance of objects in unseen views through score distillation losses. Text-to-4D [4, 5, 63, 103, 126] or image-to-4D [90, 132, 139, 142] setups take a single text prompt or image as input to create a 4D object.

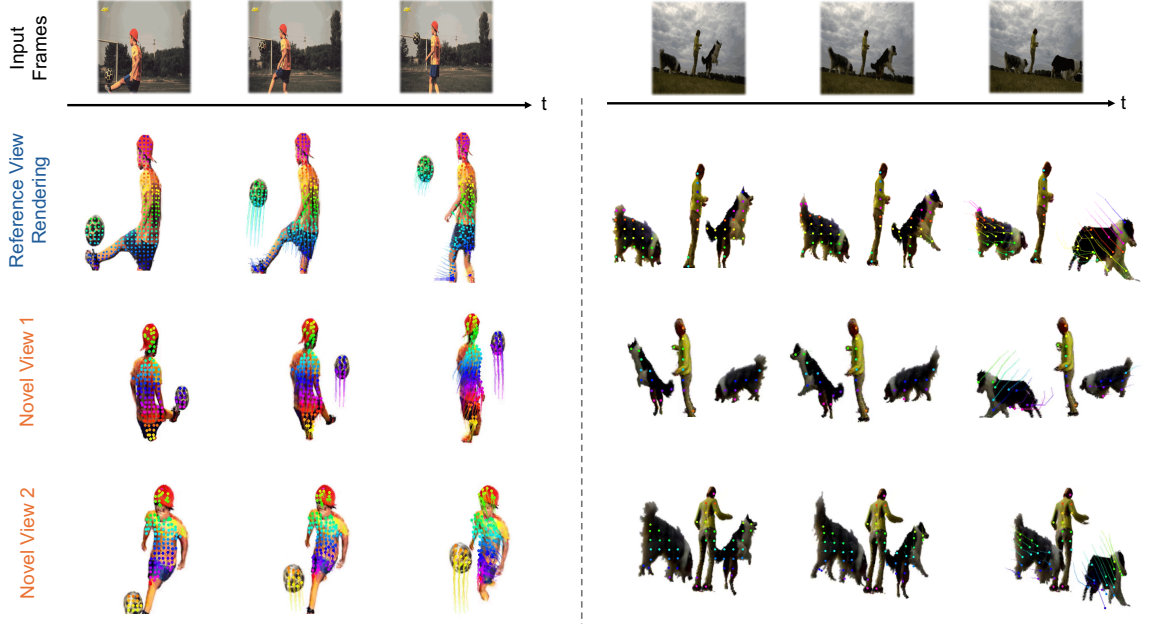


Figure 2.1: **DreamScene4D** extends video-to-4D generation to multi-object videos with fast motion. We present rendered images and the corresponding motions from diverse viewpoints at different timesteps using real-world DAVIS [81] videos with multiple objects and large motions.

Several works [29, 47, 76, 90, 135, 136] explore the video-to-4D setup; however, these methods predominantly focus on videos containing a single object with minor 3D deformations, where the object deforms in place without significant motion in the 3D scene. This focus arises because current generative models perform significantly better at predicting novel views of individual objects [66] than at predicting novel views of multi-object scenes. Consequently, score distillation objectives for 3D object lifting are challenging to apply directly at a scene level. Also, optimization difficulty arises when neural fields or 3D Gaussians are trained to model large temporal deformations directly. This limits their practical real-world usage, where input videos depict complex real-world scenes containing multiple dynamic objects with fast motions, as illustrated in Figure 2.4.

In this chapter, we propose DreamScene4D, the first video-to-4D scene generation approach to produce realistic 4D scene representations from complex multi-object videos featuring significant object motion or deformation. To 360° synthesize novel views for multiple objects of the scene, DreamScene4D proposes a “decompose-

recompose” strategy. A video is first decomposed into objects and the background scene, where each is completed through occlusions and across viewpoints. Then, it is recomposed to estimate relative scales and rigid object-to-world transformations in each frame using monocular depth guidance, so that all objects are placed back in a common coordinate system.

To handle fast-moving objects, DreamScene4D factorizes the 3D motion of the static object Gaussians into three components: 1) camera motion, 2) object-centric deformations, and 3) an object-centric to world frame transformation. This factorization significantly improves the stability of the motion optimization process by using powerful object trackers [15] to handle large motions. It enables view-predictive generative models to receive object-centric inputs that are drawn from the same distribution. The camera motion is estimated by re-rendering the static background Gaussians to match the video frames.

We present the view renderings of DreamScene4D in various time steps and from different viewpoints, using challenging monocular videos from DAVIS [81], as shown in Figure 2.1. DreamScene4D achieves significant improvements compared to the existing SOTA video-to-4D generation approaches [47, 90] on DAVIS, Kubric [31], and our self-captured videos with fast moving objects (Figures 2.4). To evaluate the quality of the learned Gaussian motions, we measure the 2D endpoint error (EPE) of the inferred 3D motion trajectories across occlusions and show that our approach produces accurate and persistent point trajectories in both visible views and synthesized novel views.

2.2 Related Work

2.2.1 Video-to-4D Reconstruction

Dynamic 3D reconstruction extends static 3D reconstruction to dynamic scenes with the goal of 3D lifting the visible parts of the video. Dynamic NeRF-based methods [10, 58, 69, 77, 83] extend NeRF [74] to dynamic scenes, typically using grid or voxel-based representations [11, 64, 71], or learning a deformation field [11, 27] that models the dynamic portions of an object or scene. Dynamic Gaussian Splatting [73] extends 3D Gaussian Splatting [52], where scenes are represented as 4D Gaussians

and show faster convergence than NeRF-based approaches. However, these 4D scene reconstruction works [73, 121, 133] typically take videos where the camera has a large number of multi-view angles, instead of a general monocular video input. This necessitates precise calibration of multiple cameras and constrains their potential real-world applicability. Different from these works [78, 121] on mostly reconstructing the visible regions of the dynamic scene, DreamScene4D can 360° synthesize novel views for multiple objects of the scene, including the unobserved regions in the video.

2.2.2 Video-to-4D Generation

In contrast to 4D reconstruction works, this line of research is most related by attempting to complete and 3D reconstruct a video scene across both visible and unseen (virtual) viewpoints. Existing text to image to 4D generation works [29, 47, 76, 90, 135, 136] typically use score distillation sampling (SDS) [82] to supply constraints in unseen viewpoints in order to synthesize full 4D representations of objects from single text [4, 5, 63, 103], image [132, 142], or a combination of both [139] prompts. They first map the text prompt or image prompt to a synthetic video, then lift the latter using deformable 3D differentiable NeRFs [58] or set of Gaussians [121] representation. Existing video-to-4D generation works [29, 47, 76, 90, 135, 136] usually simplify the input video by assuming a non-occluded and slow-moving object while real-world videos with multiple dynamic objects inevitably contain occlusions. Owing to our proposed scene decoupling and motion factorization schemes, DreamScene4D is the first approach to generate complicated 4D scenes and synthesize their arbitrary novel views by taking real-world videos of multi-object scenes.

2.3 Method

To generate dynamic 4D scenes of multiple objects from a monocular video input, we propose DreamScene4D, which takes Gaussian Splatting [52, 121] as the 4D scene representation and leverages powerful foundation models to generalize to diverse zero-shot settings.

2.3.1 Background: Generative 3D Gaussian Splatting

Gaussian Splatting [52] represents a scene with a set of 3D Gaussians. Each Gaussian is defined by its centroid, scale, rotation, opacity, and color, represented as spherical harmonics (SH) coefficients.

Score Distillation Sampling (SDS) [82] is widely used for text-to-3D or image-to-3D tasks by leveraging a diffusion prior for optimizing 3D Gaussians to synthesize novel views. For 3D object generation, DreamGaussian [109] uses Zero-1-to-3 [66], which takes a reference view and a relative camera pose as input and generates plausible images for the target viewpoint, for single frame 2D-to-3D lifting. The 3D Gaussians of the input reference view I_1 are optimized by a rendering loss and an SDS loss [82]:

$$\nabla_{\phi} \mathcal{L}_{\text{SDS}}^t = \mathbb{E}_{t, \tau, \epsilon, p} \left[w(\tau) \left(\epsilon_{\theta} \left(\hat{I}_t^p; \tau, I_1, p \right) - \epsilon \right) \frac{\partial \hat{I}_t^p}{\partial \phi} \right], \quad (2.1)$$

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

2.3.2 DreamScene4D

We propose a “*decompose-recompose*” principle to handle complex multi-object scenes. As in Figure 2.2, given a monocular video of multiple objects, we first segment and track [15, 16, 51, 91] each 2D object and recover the appearance of the occluded regions (Section 2.3.2). Next, we decompose the scene into multiple amodal objects and use SDS [82] with diffusion priors to obtain a 3D Gaussian representation for each object (Section 2.3.3). To handle large object motions, we optimize the deformation of 3D Gaussians under various constraints and factorize the motion into three components (Figure 2.3): the object-centric motion, an object-centric to world frame transformation, and the camera motion (Section 2.3.3). This greatly improves the stability and quality of the Gaussian optimization and allows view-predictive image generative models to operate under in-distribution object-centric settings. Finally, we compose each individually optimized object to form a complete 4D scene

representation using monocular depth guidance (Section 2.3.3).

Video Scene Decomposition

Instead of taking the video scene as a whole, we first adopt mask trackers [15, 16, 51, 91] to segment and track objects in the monocular video when GT object masks are not provided. From the monocular video and associated object tracks, we amodally complete each object track before lifting it to 3D as in Figure 2.2. To achieve zero-shot object appearance recovery for occluded regions of individual object tracks, we build off of inpainting diffusion models [92] and extend it to videos for amodal video completion.

2.3.3 Video Amodal Completion

We build off SD-Inpaint [92] and adapt it for video amodal completion by making two modifications to the inference process without further fine-tuning.

Spatial-Temporal Self-Attention

A common technique for extending Stable Diffusion-based models for video generation editing inflates the spatial self-attention layers to additionally attend across frames without changing the pre-trained weights [12, 53, 84, 122]. Similar to [12], we inject tokens from adjacent frames during self-attention to enhance inpainting consistency. Specifically, the self-attention operation can be denoted as:

$$Q = W_Q z_t, K = W_K [z_{t-1}, z_t, z_{t+1}], V = W_V [z_{t-1}, z_t, z_{t+1}], \quad (2.2)$$

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

Latent Consistency Guidance

While inflating the self-attention layers allows the diffusion model to attend to and denoise multiple frames simultaneously, it does not ensure that the inpainted video

frames are temporally consistent. To solve this issue, we take inspiration from previous works that perform test-time optimization while denoising for structured image editing [79] and panorama generation [56] and explicitly enforce the latents during denoising to be consistent.

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

$$\hat{z}^\tau = z^\tau - \eta \nabla_z \mathcal{L}_c, \quad (2.3)$$

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

$$z^{\tau-1} = \sqrt{\alpha_{t-1}} \left(\frac{\hat{z}^\tau - \sqrt{1 - \alpha_t} \hat{\epsilon}^\tau}{\sqrt{\alpha_t}} \right) + \sqrt{1 - \alpha_{t-1}} \hat{\epsilon}^\tau, \quad (2.4)$$

where α_t is the noise scaling factor defined in DDIM [104].

Object-Centric 3D Lifting from World Frame

After decomposing the scene into individual object tracks, we use Gaussians Splatting [52] with SDS loss [82, 109] to lift them to 3D. Since novel-view generative models [66] trained on Objaverse [20] are inherently object-centric, we take a different manner to 3D lifting. Instead of directly using the first frame of the original video, where the object areas may be small and not centered, we create a new object-centric frame \tilde{I}_1 by cropping the object using its bounding box and re-scaling it. Then, we optimize the static 3D Gaussians with both the RGB rendering on \tilde{I}_1 and the SDS loss [82] in Eq. 2.1.

2. Dynamic Multi-Object Scene Generation from Monocular Videos

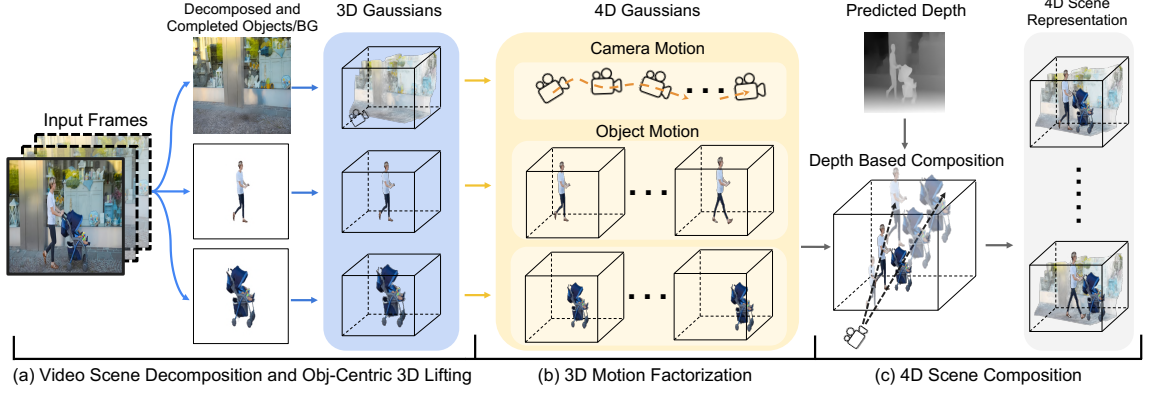


Figure 2.2: **Method overview for DreamScene4D:** (a) We first *decompose* and amodally complete each object and the background in the video sequence and use DreamGaussian [109] to obtain static 3D Gaussian representation. (b) Next, we factorize and optimize the motion of each object track independently, detailed in Figure 2.3. (c) Finally, we use the estimated monocular depth to *recompose* the independently optimized 4D Gaussians into one unified coordinate frame.

Modeling Complex 3D Motions via Motion Factorization

To estimate the motion of the first-frame lifted 3D Gaussians $\{G_1^{obj}\}$, one solution like DreamGaussian4D [90] is to model the object dynamics by optimizing the deformation of the 3D Gaussians directly in the world frame. However, this approach falls short in videos with large object motion, as the rendering loss yields minimal gradients until there is an overlap between the deformed Gaussians in the re-rendered frames and the objects in the video frames. Large motions of thousands of 3D Gaussians also increase the training difficulty of the lightweight deformation network [11, 27].

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

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

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

To compute the 3D object motion in the object-centric frames, we take the cropped and scaled objects in the individual frames I_t , forming a new set of frames \tilde{I}_t^r for each object. Following DreamGaussian4D [90], we adopt a K-plane [27] based deformation network $D_\theta(G_1^{obj}, t)$ to predict the 10-D deformation parameters (μ_t, R_t, s_t) for each object per timestep. We denote the rendered image at timestep t under the camera pose p as \hat{I}_t^p , and optimize D_θ using the SDS loss in Eq. 2.1, as well as the rendering loss between \hat{I}_t^r and \tilde{I}_t^r for each frame under the reference camera pose r .

Since 3D Gaussians can freely move within uniformly-colored regions without penalties, the rendering and SDS loss are often insufficient for capturing accurate motion, especially for regions with near-uniform colors. Thus, we additionally introduce a flow rendering loss \mathcal{L}_{flow} , which is the masked L1 loss between the rendered optical flow of the Gaussians and the flow predicted by an off-the-shelf optical flow estimator [127]. The flow rendering loss only applies to the confident masked regions that pass a simple forward-backward flow consistency check.

Physical Prior on Object-Centric Motion Object motion in the real world follows a set of physics laws, which can be used to constrain the Gaussian deformations further. For example, objects usually maintain a similar size in temporally neighboring frames. Thus, we incorporate a scale regularization loss

$$\mathcal{L}_{scale} = \frac{1}{T} \sum_{t=1}^T \|s_{t+1} - s_t\|_1, \quad (2.5)$$

where we penalize large Gaussian scale changes.

To preserve the local rigidity during deformations, we apply a loss \mathcal{L}_{rigid} to penalize changes to the relative 3D distance and orientation between neighboring Gaussians following [73]. We disallow pruning and densification of the Gaussians when optimizing for deformations like [73, 90].

Object-to-world Frame Transformation We compute the translation $\Delta_t = (\Delta_{x,t}, \Delta_{y,t}, \Delta_{z,t})$ and scaling factor s'_t that warps the Gaussians from the object-centric frame to the world frame. The 2D bounding-box-based cropping and scaling (Sec 2.3.3) from the original frames to the object-centric frames can be represented

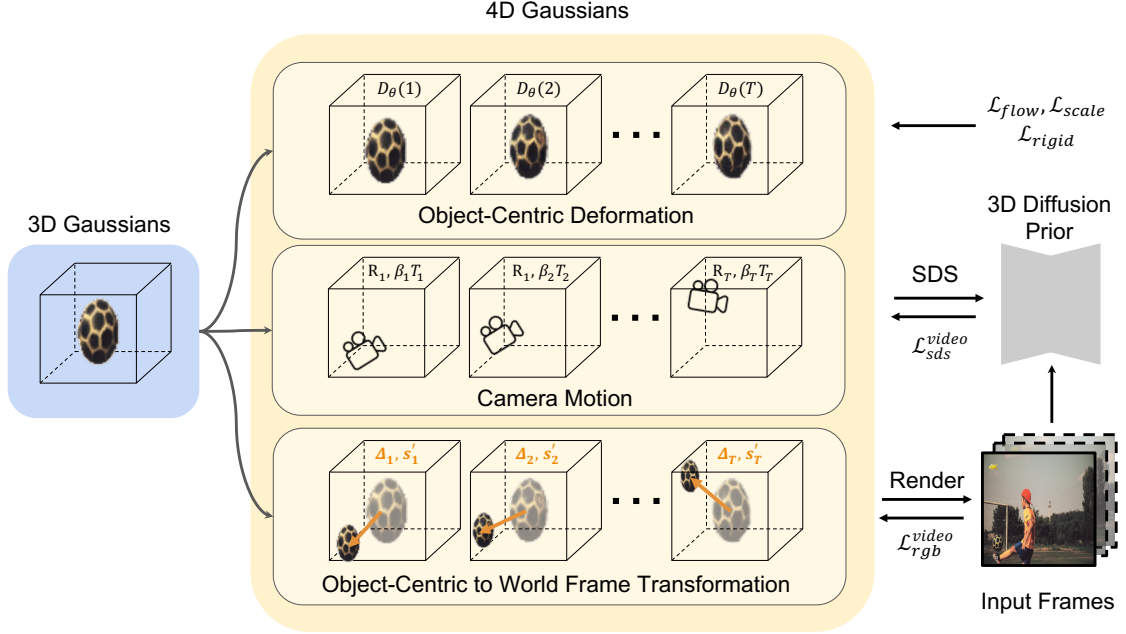


Figure 2.3: **3D Motion Factorization.** The 3D motion is decomposed into 3 components: 1) the object-centric deformation, 2) the camera motion, and 3) the object-centric to-world frame transformation. After optimization, they can be composed to form the original object motion observed in the video.

as an affine warp, which we use to compute and initialize $\Delta_{x,t}$, $\Delta_{y,t}$, and s'_t for each object in each frame. $\Delta_{z,t}$ is initialized to 0. We then adopt the rendering loss on the original frames I_t instead of center-cropped frames \tilde{I}_t to fine-tune Δ_t with a low learning rate.

To further improve the alignment between renderings and the video frames, it is essential to consider the perceptual parallax difference. This arises when altering the object’s 3D position while maintaining a fixed camera perspective, resulting in subtle changes in rendered object parts. Thus, we compose the individually optimized motion components and jointly fine-tune the deformation network D_θ and affine displacement Δ_t using the rendering loss. This refinement process, conducted over a limited number of iterations, helps mitigate the parallax effect as shown in Figure 2.7 of the appendix.

Camera Motion Estimation We leverage differentiable Gaussian Splatting rendering to jointly reconstruct the 3D static video background and estimate camera

motions. Taking multi-frame inpainted background images I_t^{bg} as input, we first use an off-the-shelf algorithm [118] to initialize the background Gaussians and relative camera rotation and translation $\{R_t, T_t\}$ between frame 1 and frame t . However, the camera motion can only be estimated up to an unknown scale [134] as there is no metric depth usage. Therefore, we also estimate a scaling term β for T_t . Concretely, from the background Gaussians G^{bg} and $\{R_t, T_t\}$, we find the β that minimizes the rendering loss of the background in subsequent frames:

$$\mathcal{L}_{\text{bg}} = \frac{1}{T} \sum_{t=1}^T \left\| I_t^{\text{bg}} - \phi(G^{\text{bg}}, R_t, \beta T_t) \right\|_2, \quad (2.6)$$

Empirically, optimizing a separate β_t per frame [7] yields better results by allowing the renderer to compensate for erroneous camera pose predictions.

4D Scene Composition with Monocular Depth Guidance

Given the individually optimized 4D Gaussians, we recompose them into a unified coordinate frame to form a coherent 4D scene. As illustrated in Step (c) of Figure 2.2, this requires determining the depth and scale for each object along camera rays.

Concretely, we use an off-the-shelf depth estimator [129] to compute the depth of each object and the background and exploit the relative depth relationships to guide the composition. We randomly pick an object as the “reference” object and estimate the relative depth scale k between the reference object and all other objects. Then, the original positions μ'_t and scales s'_t of the 3D Gaussians for the objects are scaled along the camera rays given this initialized scaling factor k : $\mu'_t = \mathcal{C}^r - (\mathcal{C}^r - \mu_t) * k$ and $s'_t = s_t * k$, where \mathcal{C}^r represents the position of the camera. Finally, we compose and render the depth map of the reference and scaled object, and minimize the affine-invariant L1 loss [87, 129] between the rendered and predicted depth map to optimize each object’s scaling factor k :

$$\mathcal{L}_{\text{depth}} = \frac{1}{HW} \sum_{i=1}^{HW} \left\| \hat{d}_i^* - \hat{d}_i \right\|_1, \quad \hat{d}_i = \frac{d_i - t(d)}{\sigma(d)}. \quad (2.7)$$

Here, \hat{d}_i^* and \hat{d}_i are the scaled and shifted versions of the rendered depth d_i^* and

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

2.4 Implementation Details

We run our experiments on one 40GB A100 GPU. We crop and scale the individual objects to around 65% of the image size for object lifting. For static 3D Gaussian optimization, we optimize for 1000 iterations with a batch size of 16. For optimizing the dynamic components, we optimize for 100 times the number of frames with a batch size of 10.

2.4.1 Deformation Network.

The deformation network uses a Hexplane [11] backbone representation with a 2-layer MLP head on top to predict the required outputs. In our evaluations, the resolution of the Hexplanes is $[64, 64, 64, 25]$ for (x, y, z, t) to ensure fair comparisons with the baselines. For longer videos (more than 32 frames), we set the resolution to $[64, 64, 64, 0.8T]$ for (x, y, z, t) , where T is the number of frames. We found that the network is generally quite robust to the temporal resolution of the Hexplane grid.

2.4.2 Learning Rate.

Following DreamGaussian [109] and DreamGaussian4D [90], we set different learning rates for different Gaussian parameters. We use the same set of hyperparameters as DreamGaussian and use a learning rate that decays from $1e^{-3}$ to $2e^{-5}$ for the position, a static learning rate of 0.01 for the spherical harmonics, 0.05 for the opacity, and $5e^{-3}$ for the scale and rotation. The learning rate of the Hexplane grid is set to $6.4e^{-4}$ while the learning rate of the MLP prediction heads is set to $6.4e^{-3}$. During joint fine-tuning of the deformation network and the object-centric to world frame

transformations, we set the learning rate to 0.1x the original value. We use the AdamW optimizer for all our optimization processes.

2.4.3 **Densification and Pruning.**

Following [90, 109], the densification in the image-to-3D step is applied for Gaussians with accumulated gradient larger than 0.5 and max scaling smaller than 0.05. Gaussians with an opacity value less than 0.01 or max scaling larger than 0.05 are also pruned. This is done every 100 optimization step. Densification and pruning are both disabled during motion optimization.

2.4.4 **Running Time.**

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

2.5 **Experiments**

2.5.1 **Datasets**

While there exist datasets used in previous video-to-4d generation works [47], they only consist of a small number of single-object synthetic videos with small amounts of motion. Thus, we evaluate the performance of DreamScene4D on more challenging multi-object video datasets, including DAVIS [81], Kubric [31], and some self-captured videos with large object motion. We select a subset of 30 challenging real-world videos from DAVIS [81], consisting of multi-object monocular videos with various amounts of motion. We further incorporate the labeled point trajectories from TAP-Vid-DAVIS [22] to evaluate the accuracy of the learned Gaussian deformations. In addition, we generated 50 multi-object videos from the Kubric [31] simulator, which

provides challenging scenarios where objects can be small or off-center with fast motion.

2.5.2 Evaluation Metrics

The quality of 4D generation can be measured in two aspects: the view rendering quality of the generated 3D geometry of the scene, and the accuracy of the 3D motion. For the former, we follow previous works [47, 90] and report the CLIP [86] and LPIPS [138] scores between 4 novel-view rendered frames and the reference frame, and compute its average score across all views and timesteps per video. We render from the following combination of (elevation, azimuth) angles: (0, 45), (0, -45), (45, 0), (-45, 0). These metrics allow us to assess the semantic similarity between rendered and reference frames. We also conducted a user study to evaluate the 4D generation quality for the DAVIS videos using two-way voting to compare each baseline with our method, where 50% / 50% indicates equal preference.

The accuracy of the estimated motion can be evaluated by measuring the End Point Error (EPE) of the projected 3D trajectories. For Kubric, we report the mean EPE separately for fully visible points and points that undergo occlusion. For DAVIS, we report the mean and median EPE [141], as the annotations only exist for visible points.

2.5.3 Video to 4D Scene Generation

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

- (1) Consistent4D [47], a recent state-of-the-art method for 4D generation from monocular videos that fits dynamic NeRFs per video using rendering losses and score distillation.
- (2) DreamGaussian4D [90], which uses dynamic 3D Gaussian Splatting like us for 4D generation from videos, but does not use any video decomposition or motion factorization as DreamScene4D. This is most related to our method.
- (3) DreamGaussian4D+VSD (Video Scene Decomposition). We augment DreamGaussian4D with VSD, where we segment every object before 4D lifting, and recompose them. The main difference between this stronger variant and our DreamScene4D is the lack of motion factorization.

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

4D Generation Results on DAVIS & Kubric We present the 4D generation quality comparison in Table 2.1, where our proposed Video Scene Decomposition (VSD) and Motion Factorization (MF) schemes greatly improve the CLIP and LPIPS score compared to the input reference images. From the user study, we can also observe that DreamScene4D is generally preferred over each baseline. Compared to the baselines, these significant improvements are mainly due to our proposed motion factorization, which enables the SDS loss to perform in an object-centric manner while reducing the training difficulty for the lightweight Gaussian deformation network in predicting large object motions. We also show qualitative comparisons of 4D generation on multi-object videos and videos with large motion in Figure 2.4, where both variants of DreamGaussian4D [90] and Consistent4D [47] tend to produce distorted 3D geometry, faulty motion, or broken artifacts of objects. This highlights the applicability of DreamScene4D to handle real-world complex videos.

4D Generation Results on Self-Captured Videos We also captured some monocular videos with fast object motion using a smartphone to test the robustness of DreamScene4D, where objects can be off-center and are subject to motion blur. We present qualitative results of the rendered 4D Gaussians in the right half of Figure 2.4. Even under more casual video capturing settings with large motion blur, DreamScene4D can still provide temporally consistent 4D scene generation results while the baselines generate blurry results or contain broken artifacts of the objects.

2.5.4 4D Gaussian Motion Accuracy

Baselines and Ablations Design

To evaluate the accuracy of the 4D Gaussian motion, we consider DreamGaussian4D [90] as the baseline, since extracting motion from NeRF-based methods [47] is highly non-trivial. In addition, we compare against PIPS++ [141] and CoTracker [48], two fully-supervised methods explicitly trained for point-tracking, serving as upper bounds for performance.

Table 2.1: **Video to 4D Scene Generation Comparisons.** We report the CLIP and LPIPS scores in Kubric [31] and DAVIS [81]. For user preference, A% / B% denotes that A% of the users prefer the *baseline* while B% prefer *ours* in two-way voting. We denote methods with Video Scene Decomposition as **VSD** and methods with Motion Factorization as **MF**.

Method	VSD	MF	DAVIS			Kubric	
			CLIP \uparrow	LPIPS \downarrow	User Pref.	CLIP \uparrow	LPIPS \downarrow
Consistent4D [47]	-	-	82.14	0.141	28.3% / 71.7%	80.46	0.117
DreamGaussian4D [90]	\times	\times	77.81	0.181	22.1% / 77.9%	73.45	0.146
DreamGaussian4D w/ VSD	\checkmark	\times	81.39	0.169	30.4% / 69.6%	79.83	0.122
DreamScene4D (Ours)	\checkmark	\checkmark	85.09	0.152	-	85.53	0.112
w/o \mathcal{L}_{flow}	\checkmark	\checkmark	84.94	0.152	-	86.41	0.113
w/o \mathcal{L}_{rigid} and \mathcal{L}_{scale}	\checkmark	\checkmark	83.24	0.153	-	84.07	0.115

4D Motion Accuracy in Video Reference Views

In Table 2.2, we tabulate the motion accuracy comparison, where DreamScene4D achieves significantly lower EPE than the baseline DreamGaussian4D on both the DAVIS and Kubric datasets. We noted that conventional baselines often fail when objects are positioned near the edges of the video frame or undergo large motion. Interestingly, the motion accuracy of DreamScene4D outperforms PIPS++ [141], despite never being trained on point tracking data, as in Figure 2.5. This is due to the strong object priors of DreamScene4D, as the Gaussians adhere to remaining on the same object it generates and their motion is often strongly correlated.

4D Motion Results on Generated Novel Views

An advantage of representing the scene using 4D Gaussians is being able to obtain motion trajectories in arbitrary camera views, which we visualize in Figure 2.1 and Figure 2.6. DreamScene4D can both generate a 4D scene with consistent appearance across views and produce temporally coherent motion trajectories in novel views.

Mitigating Parallax Effects via Joint Optimization

We show an example of the rendered Gaussians before and after performing the joint optimization for the deformation network and the object-centric to world frame transformations in Figure 2.7. We can see that a small amount of joint fine-tuning

Table 2.2: **Gaussian Motion Accuracy**. We report the EPE in Kubric [31] and DAVIS [22, 81]. We denote methods with our Video Scene Decomposition in column **VSD** and methods with 3D Motion Factorization in column **MF**. Note that CoTracker is trained on Kubric.

Method	VSD	MF	DAVIS		Kubric	
			Mean EPE ↓	Median EPE ↓	EPE (vis) ↓	EPE (occ) ↓
<i>(a) Not trained on point tracking data</i>						
DreamGaussian4D [90]	✗	✗	26.65	6.98	101.79	120.95
w/ VSD	✓	✗	20.95	6.72	85.27	92.42
DreamScene4D (Ours)	✓	✓	8.56	4.24	14.30	18.31
w/o \mathcal{L}_{flow}	✓	✓	10.91	3.83	18.54	24.51
w/o \mathcal{L}_{rigid} and \mathcal{L}_{scale}	✓	✓	10.29	4.78	16.21	22.29
<i>(b) Trained on point tracking data</i>						
PIPS++ [141]	-	-	19.61	5.36	16.72	29.65
CoTracker [48]	-	-	7.20	2.08	2.51	6.75

steps helps alleviate the parallax effect and better aligns the rendered Gaussians to the input video frames.

2.6 Limitations

Despite the exciting progress and results presented in the paper, several limitations still exist: **(1)** The SDS prior fails to generalize to videos captured from a camera with steep elevation angles. **(2)** Scene composition may fall into local suboptimas if the rendered depth of the lifted 3D objects is not well aligned with the estimated depth. **(3)** Despite the inpainting, the Gaussians are still under-constrained when heavy occlusions happen, and artifacts may occur. **(4)** Our runtime scales linearly with the number of objects and can be slow for complex videos. Addressing these limitations by pursuing more data-driven ways for video to 4D generation is a direct avenue of our future work.

We additionally show some failure cases corresponding to the limitations documented in the main text in Figure 2.8. Based on our observations, the inpainting is very unstable during heavy occlusions. We believe that instead of solely relying on rendering losses for the occluded regions, incorporating some form of semantic

guidance loss (e.g. CLIP feature loss) might be a promising direction.

2.7 Conclusion

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

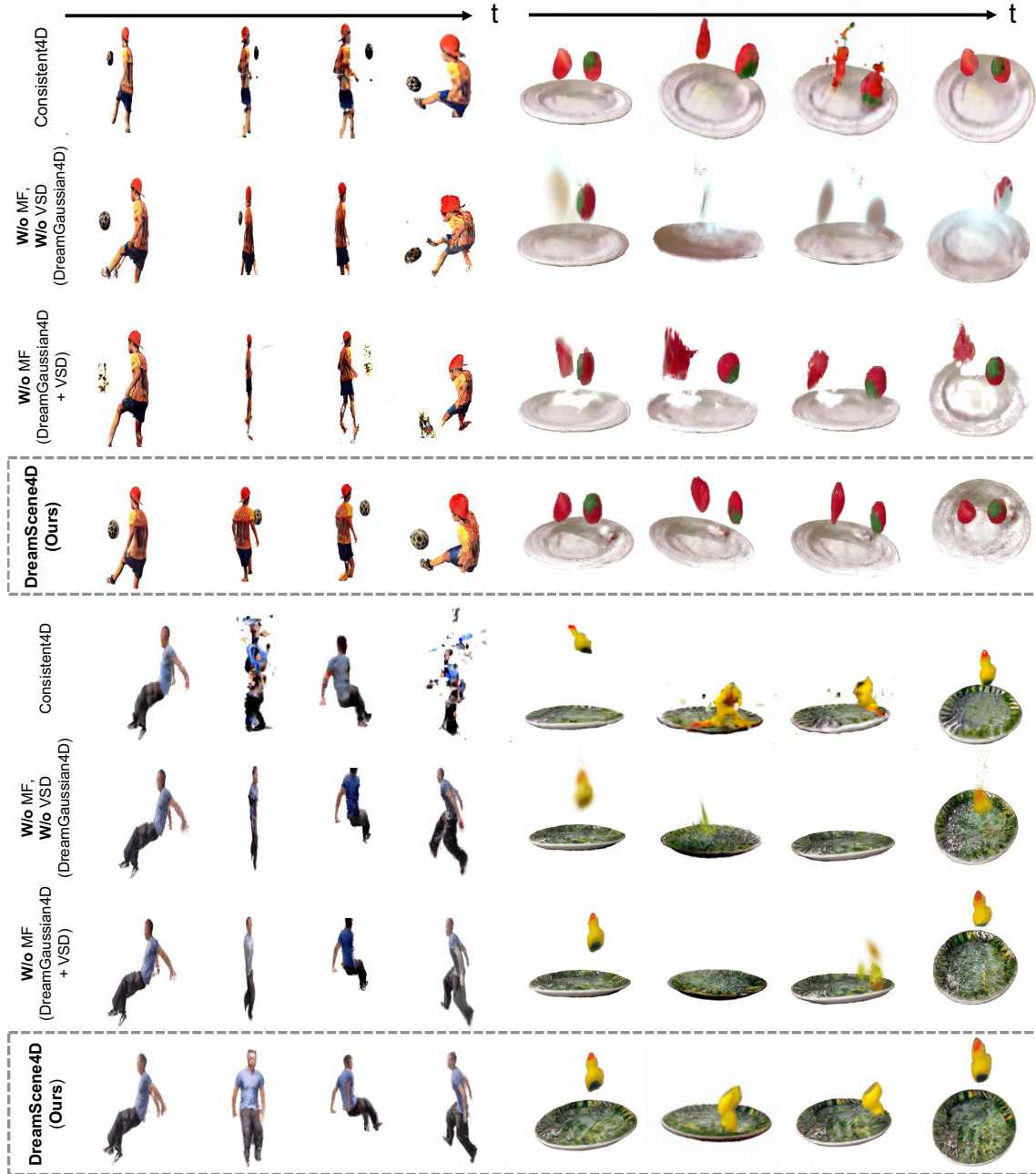


Figure 2.4: **Video to 4D Comparisons.** We render the Gaussians at various timesteps and camera views. We denote Motion Factorization as **MF** and Video Scene Decomposition as **VSD**. Our method produces consistent and faithful renders for fast-moving objects, while DreamGaussian4D [90] (2nd row) and Consistent4D [47] (1st row) produce distorted 3D geometry, blurring, or broken artifacts. Refer to our Supp. Materials for extensive qualitative comparisons.



Figure 2.5: **Motion Comparisons.** The 2D projected motion of Gaussians accurately aligns with dynamic human motion trajectory in the video, where the point trajectories estimated by PIPS++ [141] tend to get “stuck” in the background wall. For CoTracker [48], partial point trajectories are mixed up, where some points in the chest region (yellow/green) ending up in the head area (red).

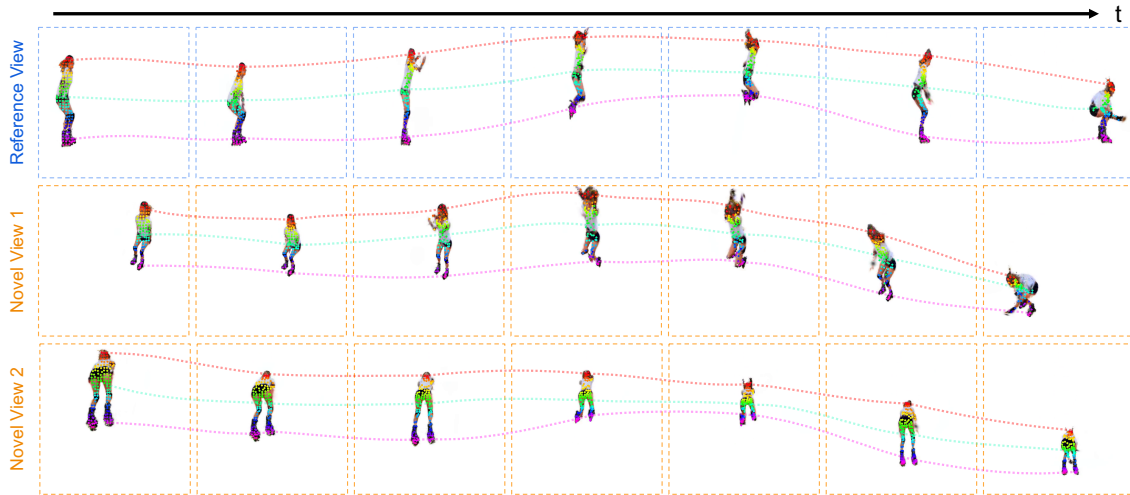


Figure 2.6: **Gaussian Motion Visualizations.** We visualize the Gaussian trajectories in the reference view corresponding to the video as well as in multiple novel views. The rendered Gaussians are sampled *independently* for each view. DreamScene4D can produce accurate motion in different camera poses **w/o** explicit point trajectory supervision.



Figure 2.7: **Mitigating the parallax effect.** A small amount of joint fine-tuning steps can help mitigate the parallax effect and align the rendered Gaussians to the input video frames.

2. Dynamic Multi-Object Scene Generation from Monocular Videos

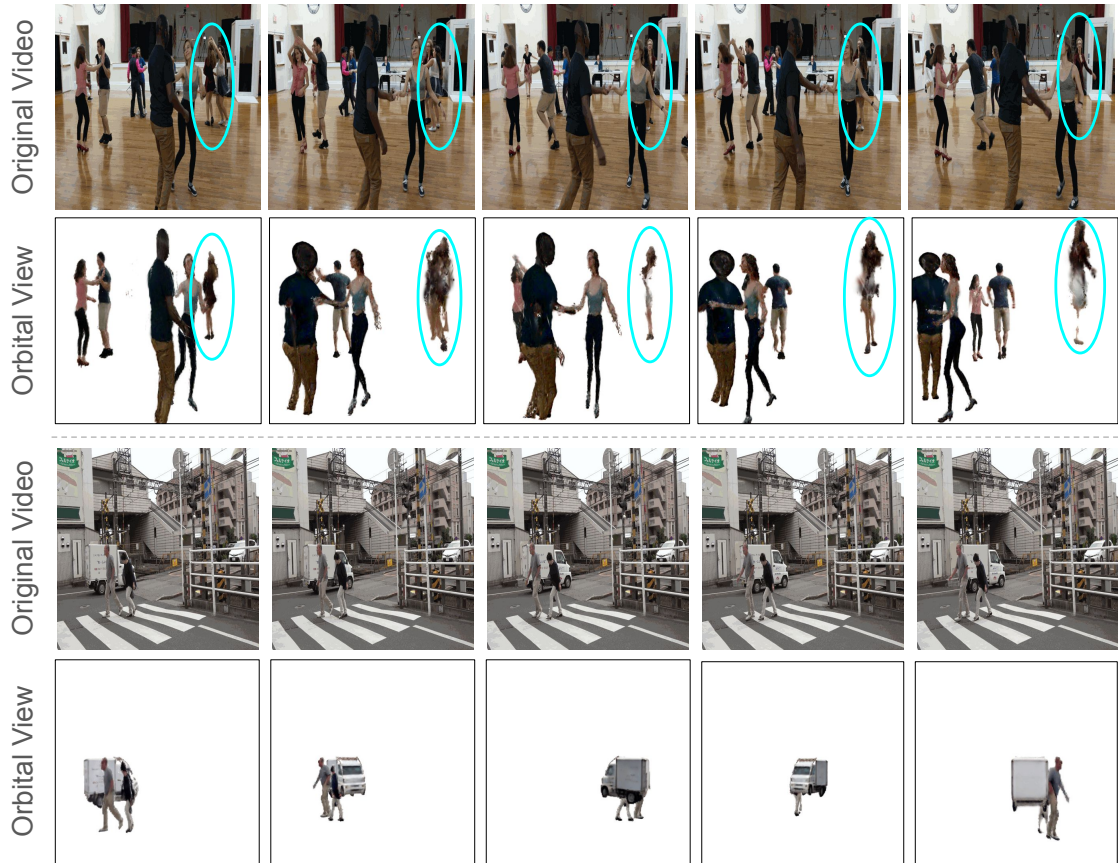


Figure 2.8: **Failure Cases.** We show 2 representative failure cases. The first case (top 2 rows) is due to inpainting failures (circled in blue), where the inpainted frames are not of high quality, leading to flickering objects when rendered. The second case (bottom 2 rows) arises from poor depth predictions, which leads to composition errors. The two humans are placed too close to the truck, making the scale proportions of the objects seem unnatural (i.e. the truck is too small).

Chapter 3

Generative 4D Scene Gaussian Splatting with Object View-Synthesis Priors

3.1 Introduction

Humans effortlessly perceive complete, coherent objects and their motions from videos even though they only observe the pixels on the front surface of a dynamic scene, which constantly changes due to motion, occlusion, and reappearance. Recovering persistent and accurate 4D representations from monocular videos with multiple objects is a highly under-constrained problem. However, it is critical for fine-grained video understanding, visual imitation in robotics [38], and building causal world models of intuitive physics [125] that can simulate the consequences of actions.

Recent view-predictive generative models [66, 94] provide powerful priors for novel view synthesis. However, existing video-to-4D generation works [17, 47, 135, 136] based on these generative models often struggle to maintain temporal coherence and accurate 3D geometry in complex multi-object scenes due to the intricate dynamics involved, such as heavy occlusions and fast motion.

Building on the per-object decompositions and robust motion tracks produced by DreamScene4D, we now turn to modeling the interactions and relationships between



Figure 3.1: **GenMOJO** addresses **video-to-4D** generation for **real-world scenes with many moving objects and heavy occlusions**. It creates complete 4D scene reconstructions, supports 360° novel view synthesis, and accurately tracks how points move over time. We show examples of rendered images from different viewpoints and time steps, highlighting the motion of a single object for clarity.

objects. While DreamScene4D excels at recovering individual object geometry and trajectories—even under large displacements and severe occlusions—it treats each object in isolation. This limits its ability to capture the spatial context and mutual influences that drive real-world dynamics. To overcome this, Chapter 3 introduces GenMOJO, a compositional method that enables **Multi-Object JOint** splatting for 4D scene **GENeration** from monocular videos by combining frame-centric test-time rendering error optimization with object-centric generative priors.

GenMOJO decomposes a scene into individual foreground objects and background, representing each object with a differentiable and deformable set of 3D Gaussians. We optimize the parameters of these Gaussians by:

1. *Jointly* splatting all Gaussians to compute rendering errors in the input video

frames, accounting for occlusions;

2. *Individually* rendering each object’s Gaussians from unobserved viewpoints to optimize a score distillation (SDS) objective.

To bridge frame-centric and object-centric coordinate frames, GenMOJO infers **differentiable transformations** that align the object-centric Gaussians with the frame-centric Gaussians. Unlike previous methods that focused on single-object [47, 90] or simple multi-object videos [17], GenMOJO is designed to handle scenes with heavy occlusions, crowded environments, and intricate dynamics, as those shown in Figure 3.1.

We evaluate GenMOJO on multi-object videos from DAVIS and a subset of MOSE [21], for which we manually annotated accurate point tracks. We compare against state-of-the-art NeRF and Gaussian-based 4D generation methods [17, 47, 116]. Our results show substantial improvements in both 4D rendering quality and motion accuracy. In addition, we conducted human evaluations to assess the perceptual realism of our results, which further support our quantitative findings.

3.2 Related work

Dynamic Scene Reconstruction. Dynamic scene reconstruction aims to lift the visible parts of a video to a dynamic 3D representation for novel view synthesis. These methods often take videos where a large number of camera views exist as input [73, 121, 133], alleviating the issue of partial observability. As a result, many of these methods can not be readily applied to monocular videos, as no constraints exist in unobserved viewpoints. These methods fall into two large categories. Dynamic Neural Radiance Field (NeRF) based methods [10, 58, 69, 77, 83] extend NeRFs [74] to dynamic scenes and model a dynamic scene using grid-based, voxel-based representations [11, 64, 71] or through deformation networks [11, 27]. Recently, Gaussian Splatting [52, 73, 121] has gained great attention in the field of neural rendering. Compared to NeRF-based methods, Gaussian Splatting represents the scene as explicit Gaussians and uses efficient rasterization techniques for rendering, greatly improving the computation efficiency. There have also been efforts to model the visible parts of a video scene from monocular inputs, but they do not attempt to model the unobserved parts [57,

106, 116, 117]. Compared to these works, we adopt dynamic Gaussians to synthesize both the visible and unobserved parts of the video scene.

Dynamic Scene Generation. In contrast to dynamic scene reconstruction, dynamic scene generation is concerned with modeling a complete video scene across both visible and unobserved viewpoints, often operating under monocular or sparse view settings. Existing text-to-4D [4, 5, 63, 103] or image-to-4D [90, 132, 142] works use diffusion models [66] to condition on text or image inputs and generate synthetic videos, then use score distillation sampling (SDS) [82] losses to supply constraints in unobserved viewpoints to synthesize complete 4D representations using dynamic NeRFs or Gaussians. Many existing video-to-4D works simplify this problem by assuming that the input consists of a single object undergoing small movements [29, 47, 76, 90, 135, 136]. More recently, this line of work has been extended to multi-object video by first decomposing the video into multiple object tracks, optimizing them independently, and then using depth models [129] to recompose the optimized 4D objects to re-form the original video scene [17].

The work most closely related to ours is DreamScene4D [17], which also leverages object-centric generative priors to produce 4D scene completions from monocular video. However, their approach inpaints and optimizes the 3D deformable Gaussians for each object independently and only combine them during a final rendering step to infer object depth ordering. This strategy struggles with cross-object occlusions, as rendering objects separately fails to account for occluders. As a result, their method often produces unnatural, inter-penetrating artifacts in scenes with multiple heavily occluded objects, as demonstrated in our experiments. In contrast, our approach unifies image-based and object-centric constraints within a single gradient-based optimization framework. By jointly splatting all object Gaussians, we accurately model cross-object occlusions and interactions, while still imposing strong appearance priors by rendering each object independently.

Point Tracking via 4D Reconstruction. Reconstructing 4D scenes from monocular videos enables point trajectory extraction through test-time optimization, where dynamic radiance fields are fit to observed video frames without requiring explicit point tracking annotations [17, 73, 95, 115, 116]. In contrast, data-driven 2D and 3D

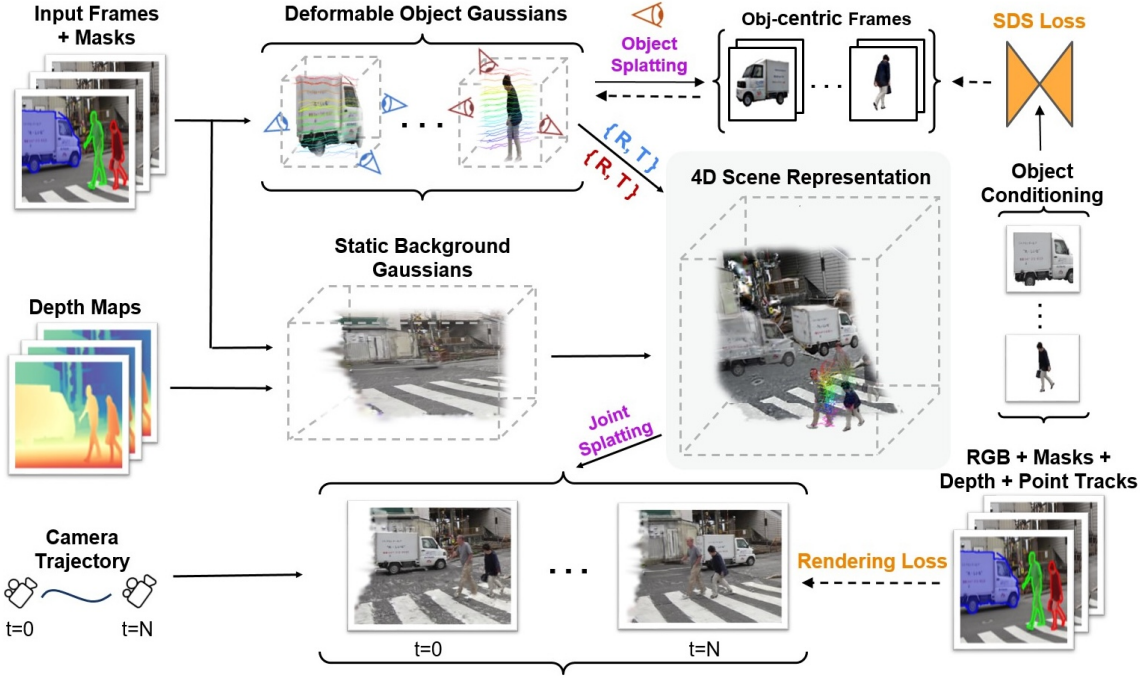


Figure 3.2: **Overview of GenMOJO.** Given a monocular video, its camera pose trajectory, and segmentation tracks for each dynamic object, our model generates a set of deforming 3D Gaussians per object. This is achieved through object-centric Score Distillation Sampling (SDS) and frame-centric joint Gaussian splatting, which accounts for cross-object occlusions by imposing rendering losses on rendered instance masks, point tracks, depth, and RGB appearance. Differentiable transformations are estimated for each object and each timestep to map the object-centric Gaussians into the camera coordinate frame.

point trackers trained on synthetic datasets often outperform these optimization-based methods on standard benchmarks [48].

However, in our experiments, we find that the performance gap narrows significantly in multi-object scenes with heavy occlusions [21]. Feedforward trackers struggle with severe occlusions, whereas our method maintains robust tracking by leveraging generative models to infer the *complete* 4D object geometry, even in unobserved or occluded regions.

3.3 4D Generation from Monocular Video

GenMOJO represents 4D scenes using deformable 3D Gaussians [52, 121], and leverages diffusion-based generative models [66] to enable novel view synthesis. Background on deformable 3D Gaussians and their optimization via score distillation sampling is provided in Section 3.3.1, while the complete model architecture is detailed in Section 3.3.2.

3.3.1 Preliminaries: Generative Gaussian Splatting

3D Gaussian Splatting [52] represents a scene as a set of colored 3D Gaussians. Each Gaussian is defined by its position $\mathbf{p} = \{x, y, z\} \in \mathbb{R}^3$, size $\mathbf{s} \in \mathbb{R}^3$ (given as standard deviations), and orientation $\mathbf{q} \in \mathbb{R}^4$ encoded as a quaternion. For efficient α -blending during rendering, each Gaussian also has an opacity value $\alpha \in \mathbb{R}$ and a color representation \mathbf{c} expressed in spherical harmonics (SH) coefficients.

Generative 3D Gaussian Splatting through SDS. Score Distillation Sampling (SDS) [82] is widely used for optimizing 3D Gaussian representations in text-to-3D and image-to-3D generation by leveraging a diffusion prior to help infer plausible novel views. DreamGaussian [109], for instance, uses Zero-1-to-3 [66], a model that predicts a novel view given a reference image and a relative camera pose, to lift 2D inputs into 3D Gaussian representations from a single frame. The optimization objective combines a differentiable rendering loss and an SDS loss:

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

where t is the timestep indices, $w(\tau)$ is a weighting function for denoising timestep τ , $\phi(\cdot)$ represents the Gaussian rendering function, \hat{I}_t^p is the rendered image under camera pose p , $\epsilon_{\theta}(\cdot)$ is the predicted noise by the diffusion model, and ϵ is the target noise.

Modeling Gaussian Deformations in Videos. To capture motion over time, DreamGaussian4D [90] models the deformation of 3D Gaussians using learnable param-

eters for each frame: **(1)** a 3D displacement for each timestep $\mu_t = (\mu x_t, \mu y_t, \mu z_t)$, **(2)** a 3D rotation for each timestep, represented by a quaternion $\mathcal{R}_t = (qw_t, qx_t, qy_t, qz_t)$, and **(3)** a 3D scale for each timestep $s_t = (sx_t, sy_t, sz_t)$. The color (SH coefficients) and opacity remain fixed across time and are initialized from the first frame. To estimate these per-frame deformations, a deformation network $D_\theta(G_0^{obj}, t)$ is trained for each object. Given the static Gaussians G_0^{obj} from the first frame and a target timestep t , the network outputs the full 10-D deformation vector. The network is trained with both rendering and SDS losses over all video frames, ensuring temporally consistent 4D Gaussians.

3.3.2 GenMOJO

Figure 3.2 illustrates the architecture and workflow of GenMOJO, a test-time 4D Gaussian splatting framework that generates complete dynamic scenes from monocular multi-object videos. GenMOJO jointly optimizes RGB, optical flow, depth, and trajectory-based rendering objectives by combining scene-centric and object-centric modeling.

Given a monocular video, we estimate camera poses and per-frame depth using recent video geometry methods [41, 61], and segment and track objects with state-of-the-art tools [51, 89, 91]. To constrain the scene’s appearance from novel viewpoints, GenMOJO incorporates object-centric view synthesis priors [82]. The scene is represented as a composition of deformable Gaussians for objects and static Gaussians for the background, jointly optimized in a shared scene-centric coordinate frame to accurately capture cross-object occlusions and spatial relationships. To do so, for each object, we fit a sequence of transformations that map its Gaussians from object-centric to scene-centric coordinates in each frame, enabling joint optimization of Gaussian parameters using both object-centric and scene-centric objectives.

Initializing Canonical 3D Gaussians. We initialize 3D Gaussians for each object using Score Distillation Sampling (SDS) and differentiable rendering within object-centric viewing spheres extracted from the first video frame. For the static background, we unproject RGBD data into a 3D point cloud and initialize a separate set of background Gaussians. While the 3D positions of the background Gaussians

remain fixed, they are trainable in other attributes, such as color and opacity, using only RGB-based rendering supervision during optimization.

Transforming Between Object-Centric and Frame-Centric Coordinate Systems. To bridge object-centric and frame-centric spaces, we estimate a set of per-object, per-frame transformations. These allow us to integrate object-centric priors (e.g., SDS) with scene-centric rendering losses during the optimization process.

These transformations are initialized using pre-trained mask trackers [89] and video depth estimators [41]. Specifically, for each object in frame t , we fit a 2D bounding box B_t to its segmentation mask. We then compute an affine warp W_t that maps B_t to a canonical bounding box in a virtual object-centric view. Geometrically, W_t can be interpreted as: (1) a translation that centers the object within a virtual viewing sphere, and (2) a scaling operation that normalizes its size. To resolve 3D translation ambiguity in the depth direction, we leverage temporally consistent depth estimates [41]. We randomly select one object j as the reference and compute the relative depth scaling factor for another object i at each frame as $k_i = \frac{D_i}{D_j}$, where D_i and D_j are the median depth values for objects i and j , respectively. Then, for frame t , we update the 3D position \mathbf{p}_t^i and scale \mathbf{s}_t^i of the Gaussians for object i using: $\mathbf{p}_t^i = \mathcal{C}^r - (\mathcal{C}^r - \mu_t^i) \times k_i$, $\mathbf{s}_t^i = \mathbf{s}_t^i \times k_i$, where μ_t^i is the original 3D position, \mathbf{s}_t^i is the scale, and \mathcal{C}^r denotes the camera position. This depth-aware scaling ensures consistent object placement across timesteps in the 3D video scene.

Multi-Object Joint Gaussian Splatting. We model the deformation of each object-centric Gaussian using K-plane-based deformation networks that predict changes in 3D position, rotation, and scale. These predicted deformations, along with the object-centric to frame-centric transformations, are then used to jointly render and optimize the deformable 3D Gaussians across all objects. To efficiently capture the complex yet structured motion patterns across different object parts, which are often highly correlated and lie in a lower-dimensional subspace, we adopt the concept of motion bases [116]. Specifically, the displacement of each Gaussian is expressed as a linear combination of a shared set of motion basis vectors. To preserve local structural rigidity during deformation, we further regularize changes in relative 3D distance and orientation between neighboring Gaussians, following [73]. Finally, we

transform the deformed Gaussians from a virtual frame-centric coordinate system to the camera coordinate system using the estimated camera pose at each timestep t , and render the scene accordingly. This allows us to disentangle the object motion from the camera motion.

To account for temporal lighting and color variation common in long videos, we allow Gaussian appearance to evolve over time by introducing an additional MLP head on top of the K-plane deformation representation [11] to predict time-varying adjustments to the spherical harmonic (SH) coefficients. The spatial and temporal continuity induced by the K-plane structure encourages nearby Gaussians to share similar color dynamics, providing a strong inductive bias. To avoid overfitting and prevent the network from explaining all appearance changes through color alone, we apply an L1 regularization term \mathcal{L}_{reg} on the SH adjustments.

A combination of losses is minimized to optimize the deformation network, including RGB rendering loss \mathcal{L}_{rgb} [52], optical flow rendering loss $\mathcal{L}_{\text{flow}}$ [17, 116], depth consistency loss, and instance segmentation alignment loss, and the two regularization losses detailed above. For the instance segmentation alignment loss, each Gaussian is assigned a one-hot instance label as a fixed attribute, and we extend the differentiable 3D Gaussian renderer from [52] to render these instance labels, producing a segmentation map \hat{y}_k for each frame. This rendered map is compared with the 2D instance masks y_k from object trackers using a standard negative log-likelihood loss: $\mathcal{L}_{\text{class}} = -\sum_k y_k \log \hat{y}_k$. This prevents Gaussians from drifting between objects during joint optimization.

Implementation Details. All experiments were conducted on a 48GB NVIDIA A6000 GPU. For object-centric lifting, we crop and scale the individual objects to approximately 65% of the image size. Static 3D Gaussian optimization is performed over 1,000 iterations with a batch size of 16, consistent with prior work [17, 90]. For deformation optimization, we run for $40\times$ the number of frames, using a batch size of 8 and gradient accumulation over 2 steps. Additional implementation details, including loss-term weighting, are provided in the supplementary material.

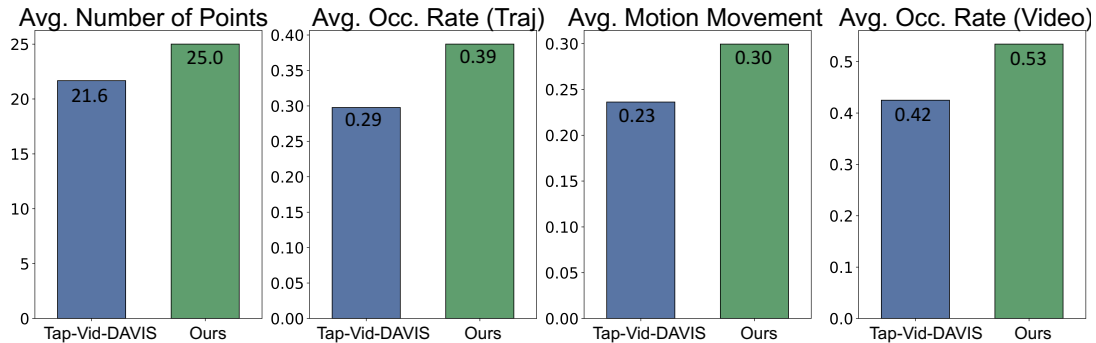


Figure 3.3: **Statistical comparison between MOSE-PTS and Tap-Vid-DAVIS [22].** Avg. number of points: the labeled points in each dataset. Avg. OCC. Rate (Traj / Video): the mean occlusion rate across individual trajectories, capturing the frequency of occlusions within distinct trajectories or videos. Avg. Motion Movement: the mean displacement of points between frames. These metrics show that MOSE-PTS contains more complex point tracks that have varied motion and undergo large occlusions.

3.4 MOSE-PTS Dataset

The proposed MOSE-PTS dataset includes 20 videos selected from MOSE [21], a dataset designed for complex video object segmentation with substantial real-world occlusions. We filter the videos using 2D visible object masks to exclude low-interaction scenes, ensuring meaningful object interactions and tracking challenges. Compared to datasets like TAP-Vid-DAVIS[22], which often feature salient and isolated objects, MOSE-PTS presents more complex, cluttered scenes. Annotators labeled up to five objects per video, with five key points per object, and all annotations were conducted at a high resolution of 1080p to ensure precision and detail. Some annotation samples are visualized in Figure 3.4.

3.4.1 Point Trajectory Analysis.

In Figure 3.3, we compare MOSE-PTS with TAP-Vid-DAVIS [22] based on four metrics: average number of points, occlusion rate per trajectory, motion range, and occlusion rate per video. MOSE-PTS has a higher average number of points and objects, which increases the complexity of the dataset. Additionally, MOSE-PTS has more complex multi-object occlusions and a wider motion range than TAP-Vid-



Figure 3.4: **Sample annotation from MOSE-PTS.** We visualize annotated points that are not occluded, with corresponding point tracks displayed in matching colors.

DAVIS.

3.4.2 Annotation Procedure

We annotate 20 videos selected from MOSE [21], which is a complex and in-the-wild video dataset designed for video object segmentation with substantial real-world occlusions. Inspired by the TAP dataset [22], we aim for generality by allowing annotators to choose any object and any point they consider important, rather than specifying a closed-world set of points to annotate.

Given a video, annotation proceeds in two stages, depicted in Figure 3.5. First, annotators choose objects, especially moving ones, without regard to their difficulty in tracking. Next, they choose points on each selected object and track them. Finally, we review and mark low-quality annotations for correction, repeating this correction procedure as many times as needed. All annotators provided informed consent before completing tasks and were reimbursed for their time.

Stage 1: Object Selection. For object selection, we began with the MOSE [21] video mask annotations, initially identifying objects from these masks that predominantly captured moving elements within the scene. We then refined this selection by asking annotators to exclude overly simple objects, such as small, stationary, or occluded elements that contribute minimally to the scene’s complexity. To enhance the dataset’s challenge for point tracking, we included additional moving objects by human eyes, prioritizing those that experience partial occlusion during movement. This process ensured that the dataset maintained a higher level of complexity, better suited for evaluating robust point-tracking algorithms.

Stage 2: Point Annotation. For each object selected in the initial stage, annotators identified a set of five key points on the first frame, including prominent features such as eyes for animals and key poses for humans. To streamline the annotation process, we used CoTracker2 [48] for point initialization across frames. Annotators then refined these points every alternate frame to ensure precise point tracking consistency with preceding frames, adjusting for minor shifts. In cases of occlusion, annotators marked frames where points were no longer visible, preserving accurate tracking continuity throughout the sequence.

3.4.3 Point Track Difficulty Analysis

To provide some insights into the difficulty of the annotated point tracks in MOSE-PTS, we run CoTrackerV2 [48] and CoTrackerV3 [49] on MOSE-PTS and report the Occlusion Accuracy (OA; accuracy of occlusion prediction), Average Delta (δ , fraction of visible points tracked within 1, 2, 4, 8, and 16 pixels, averaged over the 5 threshold values), and Average Jaccard (AJ, a combination of tracking accuracy and occlusion prediction accuracy) [22]. For the evaluation on TAP-Vid subsets, we report the values reported in the original paper.

We find that in Table 3.1 that both versions of CoTracker see up to 10 points of performance drop on MOSE-PTS across all metrics. Even the most recent CoTrackerV3, which has been trained on extra point annotations extracted from real world data, see a noticeable drop in performance on MOSE-PTS. This shows that MOSE-PTS is even more challenging than the videos presented in TAP-Vid-Kinetics [22].

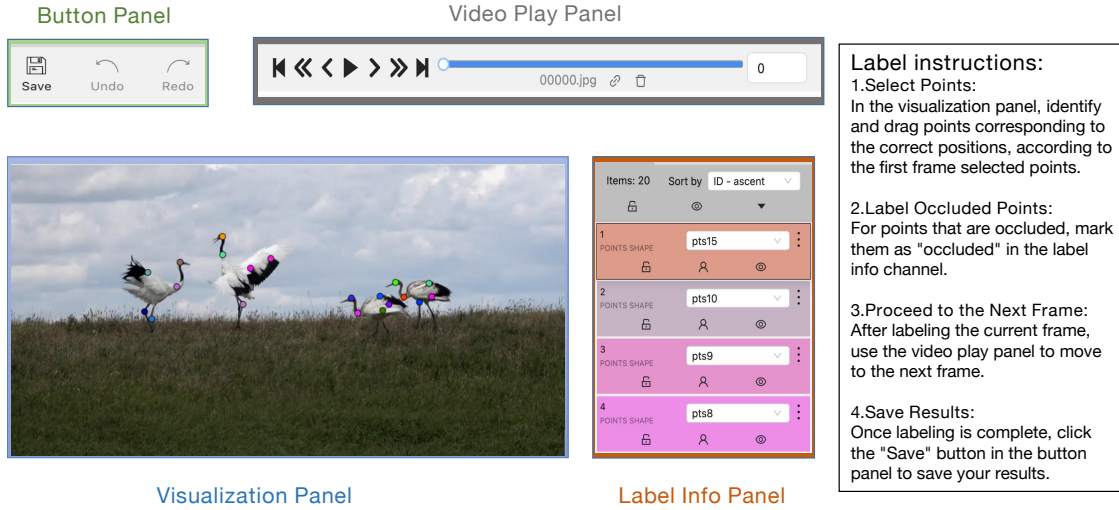


Figure 3.5: The annotation interface comprises four key components: a visualization panel for reviewing and marking points, button panel for interaction, an information panel displaying relevant details, and a video play panel for navigating through frames. The annotation process is structured into four iterative steps, guiding annotators to refine their annotations progressively.

Table 3.1: **Point tracking performance of CoTrackerV2 and CoTrackerV3 on various datasets.** The performance of CoTracker degrades by up to 10 points in MOSE-PTS, suggesting that it is a much harder dataset compared to the subsets present in TAP-Vid [22].

Dataset	CoTrackerV2 [48]			CoTrackerV3 [49]		
	AJ	δ	OA	AJ	δ	OA
RGB-Stacking	67.4	78.9	85.2	71.7	83.6	91.1
DAVIS	61.8	76.1	88.3	63.8	76.3	90.2
Kinetics	49.6	64.3	83.3	55.8	68.5	88.3
MOSE-PTS	40.4	56.1	75.5	46.7	63.6	77.9

3.5 Implementation Details

In this section, we provide some extra details on implementation and optimization, as well as the hyper-parameter choices for our experiments. We use the same set of

hyperparameters for all DAVIS and MOSE videos in our experiments.

3.5.1 Static Gaussian Optimization

Pruning and Densification. Following [17, 90, 109], we prune Gaussians with an opacity smaller than 0.01 or a scale larger than 0.05. The densification is applied for Gaussians with an accumulated gradient larger than 0.5 and max scaling smaller than 0.05. Both pruning and densification are performed every 100 optimization steps if the total number of Gaussians is smaller than 20000. We impose this limit on the number of Gaussians due to memory constraints. Unlike traditional 3D Gaussian Splatting works, we do not reset the opacity values during optimization, which yields better results.

Hyperparameters. For static Gaussian optimization, we use the same set of hyperparameters from previous works [17, 90, 109] and use a learning rate that decays from $1e^{-3}$ to $2e^{-5}$ for the position, a static learning rate of 0.01 for the spherical harmonics, 0.05 for the opacity, and $5e^{-3}$ for the scale and rotation. We optimize for a fixed number of 1000 steps for all objects in the video using a batch size of 16 when calculating the SDS loss.

Running Time. On the A6000 GPU, the static 3D-lifting process takes around 5.63 minutes for the 1000 steps, comparable to what was reported in previous works.

3.5.2 Dynamic Gaussian Optimization

Deformation Network. We follow previous works [17, 90] and use a Hexplane [11] backbone representation with a 2-layer MLP head on top to predict the required outputs. We set the resolution of the Hexplanes to $[64, 64, 64, 0.8T]$ for (x, y, z, t) dimensions, where T is the number of frames in the input video sequence. This is the same setting used in the baselines, and we keep this setting to ensure fair comparisons.

Hyperparameters. The learning rate of the Hexplane is set to $6.4e^{-4}$, and the learning rate of the MLP prediction heads is set to $6.4e^{-3}$ [17, 90]. We optimize for a $35 \cdot T$ steps with a batch size of 8, where T is the number of frames in the video.

Like previous works, we sample 4 novel views per frame in the batch to calculate the SDS loss. We use the AdamW optimizer to optimize the deformation parameters.

Running Time. The running time is heavily dependent on the length of the video, as the number of optimization steps is dependent on the total number of frames, as well as the number of objects. For reference, on the A6000 GPU, a video with 32 frames with 3 objects takes around 74 minutes. DreamScene4D [17] is roughly 10% slower as the optimization process needs to be done independently, but GenMOJO requires 15% more VRAM on a GPU due to the joint splitting procedure requiring Gaussians of all objects to be kept in memory.

3.6 Experiments

We test GenMOJO in 4D lifting from monocular videos. Our experiments aim to address these questions:

- (1) How does GenMOJO compare to other methods [17, 47, 90] in terms of novel view synthesis?
- (2) How does GenMOJO perform in tracking points through occlusions compared to existing state-of-the-art supervised point trackers [48, 49] and test-time optimization [17, 90, 116] methods?
- (3) To what extent do generative priors and object-based video decomposition contribute to improved view synthesis and point tracking?
- (4) Do object-centric generative priors [66] in GenMOJO outperform priors obtained from scene-level view-conditioned generative models [92, 94]?

To evaluate GenMOJO on complex, real-world scenarios, we select a challenging subset of 15 multiobject videos from DAVIS [22, 81] and incorporate the 20 videos from our newly proposed MOSE-PTS dataset. This combined evaluation set spans a wide range of difficult scenes with diverse object motions and interactions.

3.6.1 Video to 4D Scene Generation

Baselines. We consider the following baselines for evaluating 4D scene generation:

3. Generative 4D Scene Gaussian Splatting with Object View-Synthesis Priors



Figure 3.6: **Video to 4D Scene Generation Comparisons.** We render Shape of Motion [116], DreamScene4D [17], and GenMOJO on a DAVIS video (left) and a MOSE video (right) from the reference view (top three rows) and a novel view (bottom two rows). We can see that the baselines produce artifacts or incorrectly model spatial relationships between objects, while our method does not exhibit such errors. We also render novel views for DreamScene4D and our method, where we can similarly observe improvements.

- (1) Consistent4D [47], a method for video-to-4D generation from monocular videos that fits dynamic NeRFs per video using rendering losses and score distillation.
- (2) DreamGaussian4D [90], a 4D video generation method that, like our approach, uses dynamic Gaussian Splatting but does not employ object priors and thus lacks video decomposition.
- (3) DreamScene4D [17], a recent SOTA approach for 4D generation from multi-object

monocular videos using dynamic Gaussian Splatting, making it the most directly comparable to our method. However, unlike GenMOJO, it optimizes each object’s motion independently, whereas GenMOJO jointly optimizes motion across all objects to capture object interactions.

Evaluation Metrics. The quality of 4D generation can be assessed in two key aspects: the view rendering quality of the generated 3D geometry and the accuracy of the 3D motion. This section focuses on view rendering quality. Following previous works [47, 90], we report CLIP [86] and LPIPS [138] scores between four novel-view rendered frames and the input video frames, and calculate the average score per video. These metrics evaluate the semantic similarity between rendered and input video frames. Furthermore, we conducted a user study on MOSE videos, using a two-way voting method to compare GenMOJO with the state-of-the-art method DreamScene4D. Finally, we measure PSNR by comparing the rendered frame from the reference camera view to the input video sequence, providing a metric for the faithfulness of the 4D generation results.

4D Generation Results. The 4D generation results on MOSE and DAVIS videos are presented in Table 3.2. Since GenMOJO directly models objects in world space, it achieves the most faithful scene representation, as indicated by its high PSNR scores on both MOSE and DAVIS. GenMOJO also produces high-quality novel views, supported by its CLIP and LPIPS scores, which measure semantic similarity with the reference view, and by the user preference study. Consistent4D and DreamGaussian4D, originally designed for object-centric inputs, struggle with multi-object scenes like those in MOSE and DAVIS. This limitation often leads to distorted 3D geometry or artifacts due to ineffective motion optimization. While DreamScene4D can handle multi-object videos, it optimizes each object independently, which limits its ability to model inter-object occlusions or interactions in more complex videos.

We show qualitative comparisons on MOSE and DAVIS videos between in Figure 3.6, highlighting instances where DreamScene4D fails to capture accurate relationships between interacting objects, leading to clipping problems. Additionally, we display reference view renderings from Shape of Motion, a recent Gaussian splatting-based method for point tracking. Since Shape of Motion only models visible parts of

the scene, we exclude its results for novel views, and observe noisy renderings even in the reference view. These results demonstrate the robustness of GenMOJO in handling complex real-world videos with frequent occlusions and object interactions.

Table 3.2: Video to 4D Scene Generation Comparisons. We report the PSNR score for the re-rendered video under the reference camera pose, and the CLIP and LPIPS scores in novel views for MOSE [31] and DAVIS [81]. For user preference, we report the preference between DreamScene4D and GenMOJO in a two-way voting. GenMOJO outperforms existing methods for both novel view synthesis and produces faithful re-renderings in the reference camera view.

Method	MOSE				DAVIS		
	CLIP \uparrow	PSNR \uparrow	LPIPS \downarrow	User Pref.	CLIP \uparrow	PSNR \uparrow	LPIPS \downarrow
Consistent4D [47]	77.78	22.59	0.172	-	77.17	22.31	0.146
DreamGaussian4D [90]	81.96	17.82	0.195	-	81.62	17.68	0.183
DreamScene4D [17]	85.16	22.98	0.169	36.8%	84.13	21.73	0.163
GenMOJO (Ours)	85.41	25.56	0.168	63.2%	84.17	23.51	0.163

3.6.2 4D Gaussian Motion Accuracy

Baselines. For evaluating Gaussian motion accuracy, we compare GenMOJO with DreamGaussian4D [90] and DreamScene4D [17], as in the previous section. Consistent4D [47] is not included since extracting accurate point motion from implicit NeRF representations is highly non-trivial and beyond our scope. We also include Shape of Motion [116], a recent optimization-based point-tracking method that employs 4D Gaussian Splatting. However, unlike GenMOJO, Shape of Motion does not leverage object or generative priors in optimizing dynamic Gaussians.

Evaluation Metrics. To evaluate motion accuracy, we use both Average Trajectory Error (ATE) and Median Trajectory Error (MTE) [141], which measure the L2 distance between the estimated tracks and the ground truth tracks in all timesteps. MTE is included because it is less sensitive to extreme failure cases, providing a robust assessment of typical tracking accuracy. We also report the Average End Point Error (A-EPE) and Median End Point Error (M-EPE) [17, 34], which calculates the L2 distance specifically at the final timestep. This provides specific insights into the tracking accuracy over long time horizons. For consistency, the predicted point

Table 3.3: **Point tracking comparison between 4D generation / reconstruction methods.** We report the Average Trajectory Error (ATE), Median Trajectory Error (MTE) [141], Average End Point Error (A-EPE), and Median End Point Error (M-EPE) [17] in MOSE [31] and DAVIS [22, 81]. GenMOJO outperforms existing methods that were not trained on point tracks in MOSE, and achieves competitive performance when compared against learning-based methods.

Method	MOSE				DAVIS			
	ATE ↓	MTE ↓	A-EPE ↓	M-EPE ↓	ATE ↓	MTE ↓	A-EPE ↓	M-EPE ↓
<i>(a) Not trained on point tracking data</i>								
GMRW [102]	24.85	20.87	42.59	36.99	42.17	26.73	75.61	60.14
DreamGaussian4D [90]	27.73	25.71	46.18	43.55	30.27	28.18	61.5	55.69
Shape of Motion [116]	18.39	10.40	28.25	18.30	8.91	3.93	18.94	6.08
DreamScene4D [17]	13.47	10.49	22.48	16.97	9.59	6.50	17.47	10.29
GenMOJO (Ours)	9.91	8.47	15.46	11.11	8.16	6.32	14.14	9.36
<i>(b) Trained on point tracking data</i>								
CoTrackerv2 [48]	15.74	12.15	27.96	20.35	21.90	2.08	26.22	3.49
CoTrackerv3 [49]	11.54	8.33	22.16	16.11	21.24	1.74	24.0	3.11

tracks of all the methods are normalized to a resolution of 256×256 during the evaluation [22, 48].

Point Tracking Results. Table 3.3 presents the point tracking results on MOSE and DAVIS video datasets, including comparisons with SOTA supervised methods [48, 49] trained on point annotations serving as an upper bound for performance. Supervised trackers like CoTrackerV2 and CoTrackerV3 show significantly lower accuracy on MOSE compared to DAVIS, highlighting the greater challenge posed by MOSE videos due to frequent, prolonged occlusions and larger object and point displacements. GenMOJO achieves the lowest tracking error in all semi-supervised and optimization-based methods, excelling in each evaluation metric. It also performs competitively on the simpler DAVIS dataset.

Both DreamScene4D [17] and GenMOJO benefit from object priors that help the Gaussians stay anchored to the same object throughout the video, enhancing tracking accuracy over other baselines, where tracked points often drift and get lost on different objects. This drift is a common issue for GMRW [102], Shape-of-Motion [116], and CoTracker [48], especially during occlusions. However, DreamScene4D, unlike GenMOJO, lacks cross-object interaction modeling, which hinders its performance



Figure 3.7: **Tracking Comparisons.** GenMOJO produces more accurate point tracks compared to other optimization-based methods like Shape of Motion and DreamScene4D. CoTrackerV3 produces very accurate point tracks when it succeeds, but also produces tremendous errors when it fails, as the error can be unbounded.

during occlusions. Without joint motion optimization, DreamScene4D’s Gaussian motion tracks in separate objects struggle to account for observed interactions in the video, causing tracks to drift or flicker within objects during occlusions, ultimately reducing tracking accuracy.

Point Motion Visualizations We visualize some comparisons of the point tracking between Shape of Motion [116], GenMOJO, and CoTrackerV3 [49] in Figure 3.7. We can see the GenMOJO produces more accurate point tracks compared to other optimization-based methods like Shape of Motion and DreamScene4D. CoTrackerV3 produces very accurate point tracks when it succeeds, but also produces tremendous errors when it fails, as the error can be unbounded.

3.6.3 Ablations

We conduct ablations to understand how different components of GenMOJO affect generation quality and motion accuracy, as summarized in Table 3.4 using MOSE videos.



Figure 3.8: **Failure cases.** Top: Erroneous depth predictions can cause entities to jitter along the depth dimension. Bottom: View synthesis diffusion model failures result in degenerate textures in unseen viewpoints.

Joint vs. Independent Object Optimization. We first remove joint splatting and instead optimize each object separately, similar to DreamScene4D. This significantly reduces both motion accuracy and PSNR, showing that joint splatting is essential for handling occlusions and object interactions.

Effect of SDS for Unseen Views. Next, we test what happens if we remove SDS supervision during motion optimization. This leads to a major drop in novel view quality and also hurts motion accuracy. Without SDS, Gaussians in unobserved views become under-constrained and tend to drift.

Scene-level vs. Object-level SDS. We compare object-centric SDS (used in GenMOJO) to using scene-level SDS with a scene-level novel-view synthesis model, ZeroNVS [94]. The object-centric version performs better, confirming that object-specific view synthesis provides more accurate guidance during optimization.

Instance Mask Rendering Loss. Finally, we remove the instance mask loss. Motion accuracy decreases, although the novel view quality stays about the same. This happens because some Gaussians drift between objects, leading to less precise tracking without this constraint.

Depth Model Robustness. We replace DepthCrafter with DepthAnything-V2, which produces less stable depth predictions across frames. The impact on performance

Table 3.4: **Ablation Experiments.** We report motion accuracy and view synthesis metrics in MOSE [31] by removing different proposed components in GenMOJO. We additionally replace object-centric SDS with scene-level SDS [94], and replace DepthCrafter with DepthAnything-V2 [130] for depth initialization.

Method	MOSE			
	A-EPE ↓	M-EPE ↓	CLIP ↑	PSNR ↑
Full Model	15.46	11.11	85.41	25.56
w/o Joint Splatting	23.48	16.97	85.18	22.88
w/o SDS	19.44	14.50	81.44	25.30
w. Scene-Level SDS [94]	20.03	15.49	83.05	25.17
w/o Instance Mask Rendering	16.77	12.49	84.88	25.19
w. DepthAnything-V2 [130]	17.32	12.84	85.29	25.33

is minor, suggesting that GenMOJO is relatively robust to depth estimation noise.

3.6.4 Limitations

Despite the progress demonstrated in this paper, GenMOJO has several limitations: **(1)** Similar to previous methods [17, 90], our approach relies on the generalization capabilities of the view-conditioned generative model, which struggles with videos featuring unconventional camera poses. **(2)** Although our joint motion optimization procedure can correctly determine the depth ordering of objects, jittery depth estimations can still cause objects to “jump” along the depth axis. **(3)** Due to the reliance on test-time optimization, our method is not suitable for online applications. We visualize some failure cases corresponding to the limitations section in the main text in Figure 3.8. We envision that as better-performing foundational models for novel-view synthesis and video depth estimators are released, GenMOJO will also become more robust. Furthermore, we hope to develop novel feedforward models that can enable both real-time inference and provide stronger priors for test-time optimization in future work.

3.7 Conclusion

We introduced GenMOJO, a novel framework for 4D scene reconstruction from monocular videos that addresses the core challenge of recovering persistent object representations in complex, real-world scenarios with multiple moving objects and heavy occlusions. Unlike prior work limited to single-object or lightly occluded scenes, GenMOJO supports accurate 4D reconstruction across space and time in highly cluttered environments. It achieves this through a compositional strategy: jointly rendering all objects for occlusion-aware supervision while independently leveraging diffusion-based priors for object completion from novel viewpoints. Differentiable affine transformations unify object- and frame-centric representations into a coherent generative framework. Through extensive experiments on challenging datasets such as DAVIS and MOSE, we demonstrate that GenMOJO significantly outperforms state-of-the-art approaches in rendering fidelity, point tracking accuracy, and perceptual realism.

Chapter 4

Generative 3D Particle World Models

4.1 Introduction

Having established high-fidelity, interaction-aware 4D scene representations with GenMOJO, we now leverage these rich representations to equip agents with predictive world models. While GenMOJO focuses on accurately capturing how objects occupy and move through space and time, Chapter 4 asks a complementary question. Given such a detailed 4D state, how can we anticipate its future evolution under the influence of actions? To answer this, we introduce ParticleWorldDiffuser, a diffusion-based world model that learns to forecast 3D particle dynamics conditioned on robot inputs, or even jointly infer both agent motions and object responses, thereby closing the loop from perception to planning. To simplify the problem, we first utilize simulation-based data, where such ground truth 4D states can be extracted, to train a diffusion model.

The central premise of world models is to learn predictive models that capture how scenes evolve in response to an agent’s actions, modeling both object dynamics and action consequences [19]. These models enable agents to plan by internally simulating possible futures and selecting actions that achieve a desired scene outcome, such as object placement or manipulation [18]. This is typically done by rolling out the learned model forward in time, conditioned on candidate action sequences, and

choosing the one most closely reaches the specified goal [18].

Earlier approaches framed this as a two-stage pipeline: supervised learning was used to approximate the forward dynamics conditioned on actions, while planning was handled separately by trajectory optimization or discrete search [33, 119]. Learning scene dynamics raises fundamental questions about scene representations and how temporal structure should be modeled. Some methods have approached this by predicting the future directly in pixel space [75], while others have adopted more physics-inspired representations, such as modeling the trajectories of 3D physical points, often called particles. In the former, prediction involves generating new pixels to reflect object and camera motion. In the latter, the model predicts future states by simulating the movement of 3D particles, similar to how physics engines evolve scenes through object-level dynamics [60, 98].

Recent breakthroughs in generative modeling applied in 2D video synthesis have shown that diffusion-based objectives are highly effective for capturing the multi-modality of real-world dynamics [13, 145]. These models, however, are trained purely on visual data. While this allows them to scale using large Internet datasets, it also limits their physical grounding, often resulting in geometric inconsistencies or implausible object interactions in generated scenes [8]. This raises a compelling question: can similar advances in generative modeling be applied to **physics-aware** scene prediction in 3D?

In this work, **we leverage scalable video diffusion techniques to develop 3D particle-based world models that capture object dynamics in point cloud space.** Specifically, we adapt pixel-level diffusion objectives to operate over 3D point clouds, enabling learning of object dynamics conditioned on a sequence of actions, as well as **joint modelling of action and object trajectories**, conditioned on an initial scene configuration. We extend the efficient transformer-based denoiser architecture of Jabri et al. [44] to recurrently process point cloud sequences of arbitrary length, allowing autoregressive rollouts of long-horizon dynamics.

To obtain 3D point trajectory supervision to train our model, we capitalize on recent physics engines to generate large-scale datasets of object interaction sequences across a wide range of object categories, such as rigid, soft, and deformable objects. Despite the inevitable sim-to-real gap, our model generalizes well to real-world scenarios and produces physically plausible dynamics in both domains.

We deploy our learned world model for robot control using guided diffusion and show that it dramatically improves the efficiency of action inference compared to model-predictive control (MPC). **Thanks to joint modelling of actions and 3D object particles within the same diffusion model, a desired object goal (e.g., location, velocity, or trajectory) provides a gradient signal that can steer the denoising process toward action trajectories that achieve the goal.** Compared to random shooting, a standard strategy in MPC, guided diffusion significantly reduces computational cost and improves convergence speed, while maintaining or exceeding task success rates.

We quantitatively evaluate our approach in synthetic benchmark environments, where it exhibits better scalability and lower error drift over long time horizons compared to deterministic dynamics models [100]. We also validate its performance qualitatively in real-world trials, where the model exhibits sim-to-real generalization despite being trained primarily in simulation. Finally, we showcase the potential of 3D point cloud-based world models for model-based control, highlighting the effectiveness of guided diffusion as a mechanism for policy inference and task completion.

To the best of our knowledge, this is the first generalist 3D particle generative model trained across a wide range of object types, and the first method that uses guided diffusion for planning while modelling the interacting object in 3D, instead of simply modelling the robot’s dynamics.

In summary, our contributions are as follows:

- The first scalable framework that integrates diffusion objectives with 3D particle-based dynamics modeling.
- A curated dataset of 3D point trajectories from diverse robot-object interaction sequences, spanning rigid, soft, and deformable objects, collected from both simulation and real-world data.
- A guided diffusion approach for robot-object control, which outperforms traditional model-predictive control in both efficiency and success rate.

We will make our models and code available upon acceptance.

4.2 Related Work

4.2.1 Learning World Dynamics

Learning world dynamics has long been a central objective in robot learning, where the goal is to predict a representation of the future state of the world conditioned on current inputs and agent actions. Dreamer [32] and Daydreamer [123] use recurrent state-space models to learn latent transition dynamics through reinforcement learning with reward supervision. In parallel, models such as Genie [9], UniSim [131], and IRASim [145] aim to build fully learned video simulators that can generate plausible future visual rollouts given a sequence of actions and a specific embodiment. In the autonomous driving domain, large-scale models like GAIA-1 [39] have demonstrated the ability to generate photo-realistic videos conditioned on past frames, text descriptions, and planned actions.

An alternative and more physically grounded approach to modeling scene dynamics lies in **particle-based dynamics models**, which represent the environment as a graph of interacting 3D particles. These models operate directly in 3D space, explicitly encoding motion, geometry, and actions through learned interactions. Prior work has used deterministic graph neural networks to model the evolution of such particle systems via message passing, and applied it in simulating rigid bodies [6, 60], elastic and plastic materials [98, 99], fluids [60, 93], and granular media [93, 111, 119]. These architectures share structural similarities with transformers employing relative positional encodings [112].

ParticleWorldDiffuser advances this line of work in several key ways. First, it replaces deterministic prediction with a generative formulation, enabling the modeling of multi-modal and stochastic behaviors that are common in real-world physical interactions. Second, it introduces a scalable latent-variable diffusion architecture, approximating expensive full all-to-all particle attention operations while maintaining expressiveness. Third, unlike previous particle-based models that often train separate graph networks for each material or object type, ParticleWorldDiffuser is trained across diverse object categories and physical properties, improving generality and cross-domain applicability. Moreover, ParticleWorldDiffuser allows for fast and effective control inference through diffusion guidance.

4.2.2 Planning with diffusion models

A growing body of work has explored planning through guiding diffusion models [2, 14, 23, 25, 45, 59, 62, 128, 131]. These approaches train generative models of state and action trajectories and then guide the generation process using a differentiable reward function, either analytically defined to penalize deviation from a goal state [43, 46], or learned to predict cumulative rewards or value estimates from sampled trajectories. Recent work [72] provides a systematic exploration of the design space for diffusion-based planning. Critically, however, all the aforementioned diffusion-guided planning methods model only the agent’s dynamics, such as a robot manipulator or vehicle [46], while assuming the external environment, if at all existent, is static [43]. In contrast, ParticleWorldDiffuser is, to the best of our knowledge, the first diffusion-based control framework that jointly models both the robot and the external object it interacts with. This joint modeling allows for guided diffusion to be used to achieve specific object arrangements or placements. Our model generalizes across rigid, soft and deformable objects, which means our framework can handle control across all these scenarios.

4.3 Generative 3D Particle World Models

The architecture of ParticleWorldDiffuser is illustrated in Figure 4.1. It is a generative diffusion model designed to synthesize 3D particle trajectories for both an object and its interacting agent. Specifically, the model predicts the 3D displacements of object particles as well as the trajectories of the agent’s fingertips. By abstracting actions as fingertip particle trajectories, ParticleWorldDiffuser can seamlessly model interactions in an embodiment-agnostic manner, as well as support tools, like sticks applying external forces.

A key advantage of ParticleWorldDiffuser lies in its generative formulation, which is crucial for learning from large-scale datasets of robot-object interactions spanning diverse object types including rigid, soft, and deformable bodies. Unlike deterministic models, which often fail to generalize across such multimodal physical dynamics, ParticleWorldDiffuser is able to model the inherent variability of object behavior. This enables generalization beyond the narrow domains or material-specific setups

commonly seen in prior particle-based approaches [93, 111, 119].

We propose two variants of our model:

1. **Action-Conditioned ParticleWorldDiffuser:** Given an initial 3D point cloud and agent fingertip trajectories, this version predicts the resulting object particle trajectories. It is well-suited for use in model-predictive control (MPC) frameworks, where it can simulate the effects of sampled action sequences over time.
2. **Joint Action-Object ParticleWorldDiffuser:** Conditioned only on the initial scene point cloud, this model jointly generates both the agent’s fingertip trajectories and the corresponding object particle displacements. This formulation is critical for control via guided diffusion, where a target object goal (e.g., desired location or configuration) provides a gradient signal that steers the generation process toward action-object trajectories that satisfy the goal. Unlike MPC, which requires repeated rollouts, guided diffusion enables efficient, one-shot action inference, delivering orders-of-magnitude improvements in computational efficiency.

We provide background on diffusion models in Section 4.3.1, present the detailed architecture of ParticleWorldDiffuser in Section 4.3.2, and describe our control method via guided diffusion in Section 4.3.2. Finally, Section 4.4 details the construction of our large-scale dataset of 3D particle trajectories for diverse agent-object interactions.

4.3.1 Preliminaries: Diffusion Models

Diffusion models consist of a forward (noising) process and a reverse (denoising) process. In the forward process, Gaussian noise is gradually added to a clean data sample x_0 over T timesteps, resulting in a sequence of increasingly noisy samples: $q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1 - \bar{\alpha}_t)\mathbf{I})$, where \mathbf{x}_t denotes the noisy version of \mathbf{x}_0 at diffusion step t , and $\bar{\alpha}_t$ is a pre-defined constant determined by a variance schedule. During training, a timestep t is sampled uniformly, and \mathbf{x}_t is generated using:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \mathbf{I}). \quad (4.1)$$

The denoising network ϵ_θ is trained to minimize the mean squared error between

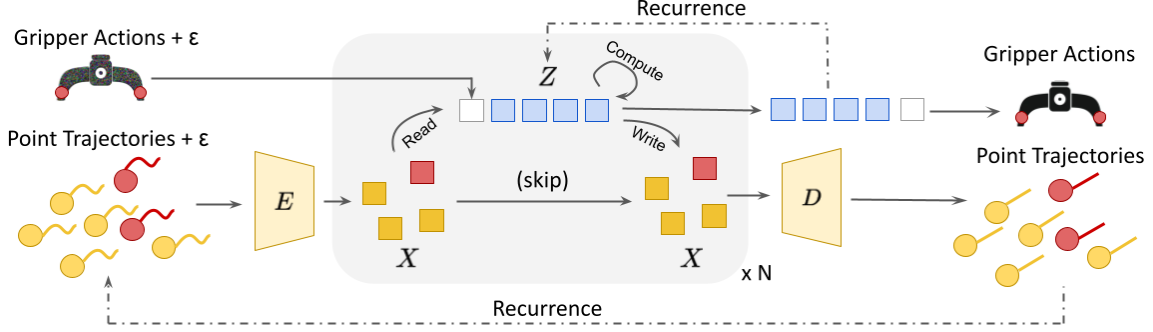


Figure 4.1: **Architecture of ParticleWorldDiffuser.** 3D point trajectories, along with its grouping information (object/gripper, denoted by yellow/red colors) are encoded into tokens \mathcal{X} using an encoder E , denoised using a diffusion model, then decoded using D . The denoising model uses a Read-Compute-Write strategy to keep the bulk of the computation in a lower-dimensional latent space, with the action sequence being injected into the latents \mathcal{Z} to integrate action information.

the predicted and true noise: $\mathcal{L} = |\epsilon_\theta(\mathbf{x}_t, t) - \epsilon_t|^2$. At inference time, the process begins from a Gaussian sample $\mathbf{x}_T \sim \mathcal{N}(0, \mathbf{I})$, and the model iteratively denoises it according to:

$$\mathbf{x}_{t-1} = \frac{\mathbf{x}_t - \sqrt{1 - \alpha_t} \epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{\alpha_t}} \quad (4.2)$$

until $t = 0$ is reached and a clean sample \mathbf{x}_0 is produced. To accelerate sampling, deterministic alternatives such as DDIM [104] can be used, significantly reducing the number of denoising steps required during inference.

4.3.2 ParticleWorldDiffuser

Data Tokenization and Normalization Our input data consists of 3D point cloud trajectories paired with a sequence of agent action pose changes. Each trajectory is represented as a sequence of N points over T time steps, forming a tensor of shape $N \times T \times 3$.

Unlike traditional latent video diffusion models that operate on compact 2D patches, representing 3D data with sufficient spatial fidelity typically requires a much larger number of tokens, often an order of magnitude more in N , leading to significant memory overhead. To address this, we design a **spatio-temporal tokenization strategy** that aggressively reduces the token count while preserving the geometric

and temporal structure of the data. Specifically, we use PointNext layers [85] to downsample the spatial dimension of the point cloud and apply a lightweight MLP to project along the temporal axis. This makes training computationally feasible without noticeably degrading downstream performance.

Since the N points contain object, gripper, and padded points for batching, we introduce a grouping one-hot vector similarly of the shape $N \times 3$ that is used by the model to differentiate between the identity of the points. This grouping vector is then concatenated to the featurized trajectories.

To encode positional information, we adopt **rotary positional embeddings** [107], which encode relative positions in a translation-invariant manner. Compared to absolute encodings such as sinusoidal embeddings, rotary embeddings are better suited for capturing local interactions in 3D space and time.

Unlike 2D video latent diffusion models, where tokenization is handled by a frozen, pre-trained VAE trained on large-scale video datasets, we jointly train our spatio-temporal tokenization layers along with the diffusion model. This joint training is necessary due to the lack of large-scale, pre-trained 3D auto-encoders and the limited generalization of existing models to the diverse object interactions and scene dynamics present in our dataset.

We apply linear scaling independently to each spatial axis using the 1st and 99th percentiles to normalize the inputs. This quantile-based normalization is more robust to outliers than standard min-max normalization, which is particularly important in dynamic scenes where abrupt contacts or large object motions can introduce extreme values.

Latent-Based Attention for Memory-Efficient 3D Diffusion We adopt **RIN** [44] as the backbone of our 3D diffusion model. RIN is a transformer-based architecture that achieves memory efficiency by replacing full self-attention on input tokens \mathcal{X} with a compact set of learned latent embeddings \mathcal{Z} , which mediate the flow of information through a **Read-Compute-Write** process.

- **Read Phase:** Each latent embedding in \mathcal{Z} attends to the input point cloud tokens \mathcal{X} , extracting and encoding relevant scene information into the latent

space:

$$\mathcal{Z}' = \text{Attention}(\mathcal{Q} = \mathcal{Z}, \mathcal{K} = \mathcal{X}, \mathcal{V} = \mathcal{X}) = \text{softmax}\left(\frac{\mathcal{Z} \mathcal{X}^\top}{\sqrt{d}}\right) \mathcal{X}.$$

- **Compute Phase:** The latents \mathcal{Z}' interact with each other via self-attention and feedforward layers, enabling global reasoning and propagation of long-range context:

$$\mathcal{Z}'' = \text{FFN}\left(\text{Attention}(\mathcal{Q} = \mathcal{Z}', \mathcal{K} = \mathcal{Z}', \mathcal{V} = \mathcal{Z}')\right).$$

- **Write Phase:** The updated latents \mathcal{Z}'' attend back to the input tokens \mathcal{X} , modifying \mathcal{X} based on the latent-derived context:

$$\mathcal{X}' = \text{Attention}(\mathcal{Q} = \mathcal{X}, \mathcal{K} = \mathcal{Z}'', \mathcal{V} = \mathcal{Z}'') = \text{softmax}\left(\frac{\mathcal{X} \mathcal{Z}''^\top}{\sqrt{d}}\right) \mathcal{Z}''.$$

Skip-connections are then used to connect the input tokens \mathcal{X} and updated tokens \mathcal{X}' . This latent-centric attention structure significantly reduces memory usage by shifting expensive self-attention operations from the high-dimensional input space to the smaller latent space.

To effectively integrate action information, we embed the action sequence a using a lightweight MLP. These action embeddings are concatenated with the latents \mathcal{Z} before the Read and Compute phases, allowing the model to condition its predictions on action information while maintaining the memory and efficiency benefits of latent-based computation.

Recurrent Training and History Conditioning To enable long-horizon 3D trajectory generation, our diffusion model must recurrently predict future segments by leveraging information from prior states. Inspired by RIN [44], we implement a flexible conditioning mechanism in which the external context is integrated by modifying the latent embeddings \mathcal{Z} of the diffusion model. This simple yet expressive strategy allows for a unified interface to incorporate various conditioning signals, such as gripper actions and previously predicted frames.

At each recurrent step, the model generates a short trajectory segment (e.g., several future timesteps) conditioned on a context window summarizing the past.

4. Generative 3D Particle World Models

During training, we simulate the rollout process by recursively feeding the model’s own predictions back into its input, rather than relying on ground-truth trajectories. This **recurrent training** approach teaches the model to self-correct and improves its robustness at inference time, when ground-truth sequences are unavailable.

For **history conditioning**, we retain the latent embeddings from the most recent \hat{K} predicted frames and allow the current latent set \mathcal{Z} to attend to them before the Read-Compute-Write operations.

$$\mathcal{Z}_t = \text{Attention}(\mathcal{Q} = \mathcal{Z}_t, \mathcal{K} = \mathcal{Z}_{t-1}, \mathcal{V} = \mathcal{Z}_{t-1}) = \text{softmax}\left(\frac{\mathcal{Z}_t \mathcal{Z}_{t-1}^\top}{\sqrt{d}}\right) \mathcal{Z}_{t-1}.$$

This compact representation of the recent movement provides a valuable context for predicting the next trajectory segment. In practice, we find that even a short history window suffices, as the diffusion model is capable of internalizing longer-term dynamics across recurrent steps. We adopt a similar strategy at inference time for long trajectory generation.

Output and Losses We begin by applying PointNext upsampling layers to restore the output to the original input shape of $N \times T \times 3$. Then, a final linear layer is applied to produce the predicted noise $\hat{\epsilon}$, which is compared to the true added noise ϵ using an L2 loss during training.

Planning with Guided Diffusion

Modelling the joint distribution of object particle trajectories and gripper actions within a single diffusion process in our Joint Action-Object ParticleWorldDiffuser variant permits action inference through guided diffusion that steers the generated trajectories towards desired object goal configurations. Specifically, we implement a form of score-based guidance that modifies the reverse diffusion steps using the gradient of a loss function $\mathcal{L}(\mathbf{x}, \mathbf{x}^*)$ that measures deviation of its argument \mathbf{x} from the desired object configuration \mathbf{x}^* (end-location or trajectory for any object particle or subset of them). We adapt the DDIM denoising process as [37]:

$$\mathbf{x}_{0|t} = \mathbf{x}_{0|t} - c_t \nabla_{\mathbf{x}_{0|t}} \mathcal{L}(\mathbf{x}_{0|t}; \mathbf{x}^*), \quad \mathbf{x}_{t-1} = \sqrt{\bar{\alpha}_{t-1}} \mathbf{x}_{0|t} + \sqrt{1 - \bar{\alpha}_{t-1} - \sigma_t^2} \epsilon_\theta(x_t, t) + \sigma_t \epsilon_t \quad (4.3)$$

This process allows us to synthesize action sequences likely to achieve the desired goal, effectively turning the model into a generative policy prior, for *any* goal configuration.

4.4 Curating a Dataset of Diverse Robot-Object Interaction Trajectories in a Physics Engine

Physics simulation engines have become essential tools for generating large-scale interaction data in robotics and reinforcement learning. Compared to real-world data collection, simulators offer three key advantages: (1) access to accurate ground-truth 3D states and action labels, (2) efficient scalability through parallelized data generation, and (3) full control over environment parameters, object properties, and agent behaviors. These benefits make simulation especially valuable for training 3D dynamics models, where rich, diverse, and precisely labeled spatiotemporal data is critical. Nevertheless, simulation data are often limited by the sim-to-real gap, caused by visual discrepancies and simplified physics, which motivates careful design and augmentation strategies.

To generate training data, we build on Genesis [3], a recently introduced simulation engine that supports multiple physics backends to simulate rigid bodies, articulated objects, and deformable materials. This flexibility enables the construction of a wide range of physically grounded scenarios.

To populate our simulation scenes, we begin by downloading 3D assets from Objaverse [20], sorted by their class labels. From this process, we select 3074 meshes from 373 categories. For each selected object, we merge its geometry into a single watertight mesh and normalize it to a consistent scale and orientation to ensure compatibility with our simulation setup. To assign physically plausible material properties, we query GPT-4 [1] with each object’s class label to infer a range of likely values for mass, friction, and elasticity. These ranges reflect semantic priors based on common material characteristics associated with the class. We then randomly sample a specific set of physical parameters within the GPT-inferred range to introduce intra-class diversity while maintaining physical coherence.

Once the object is initialized in the environment, we randomly sample its physical

size and starting location within the scene. We also randomly sample a starting position for the gripper. To simulate interactions, we execute a hand-designed manipulation policy that attempts to either push the object or pick it up and drop it. This policy is intentionally designed to produce a mix of successful and failed outcomes by introducing controlled stochasticity. As a result, it generates rich and varied contact dynamics, enabling the model to learn from both effective and unsuccessful manipulation attempts. Please refer to Figure 4.2 for some examples.

To create evaluation splits, we randomly hold out 10 specific object class labels. During training, we only sample objects that are not part of the held-out classes. At inference time, we evaluate the model on two types of test splits: one using unseen meshes from previously seen classes, and another using meshes from the held-out class labels. This setup allows us to assess the model’s ability to generalize to new object instances and novel semantic categories. However, we note that while the held-out class test split restricts direct exposure to those labels, the model may still encounter semantically similar objects during training. More challenging generalization benchmarks, such as excluding broader semantic categories or using adversarially dissimilar classes, are left for future work.

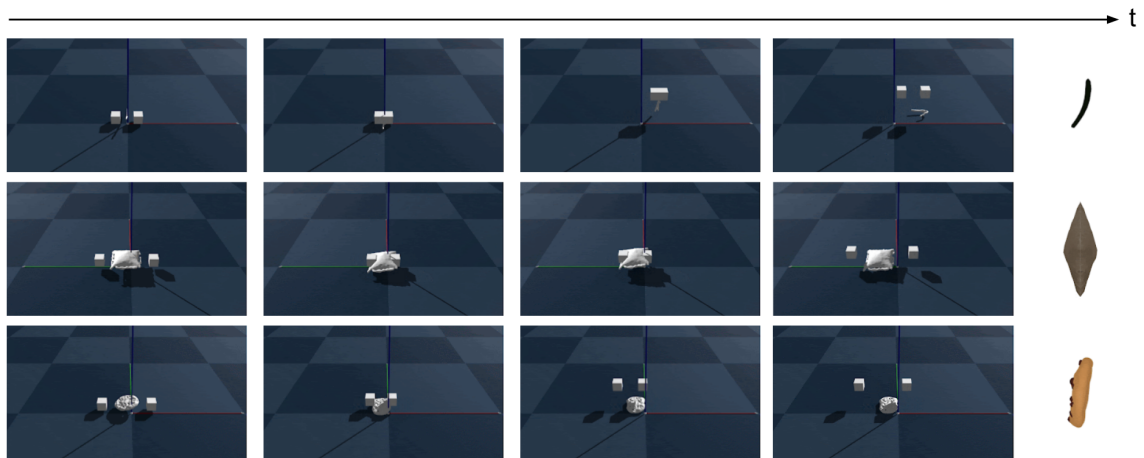


Figure 4.2: **Simulation Data Generation.** We selected some timesteps in different samples for visualization, with the rendered object mesh on the right.

The simulator provides full access to the 3D state over time, allowing us to extract temporally aligned point-cloud trajectories of both the gripper and the manipulated objects. We represent the gripper as two key points, centered at the tips of its fingers,

and define its frame-to-frame pose change as the action signal. These pose changes, paired with the evolving 3D point clouds, form the input-output pairs used to train our 3D particle-based diffusion model.

To better align simulation data with real-world observations, we apply two post-processing steps. First, we retain only surface points, as interior geometry is typically unobservable in real videos. Second, we simulate partial visibility by randomly dropping a subset of visible surface points in a local neighborhood, centered on a randomly selected point on the object. This mimics real-world occlusions and viewpoint limitations, helping to reduce the domain gap and improve the robustness of the learned models in real-world deployments.

4.4.1 Data Generation Details for Dynamics Prediction Experiments

For our training data, we generate 30k videos of 300 to 400 timesteps each. During training, the model is trained to directly predict in sequences of 8, using a context length of 1, and is trained recurrently for 3 steps. This equates to each training sequence being 24 timesteps long. For our baselines, we re-train their model in a similar way for fair comparisons. For evaluation, we randomly generate 150 videos of 90 timesteps for each evaluation setup. The models predict recurrently, conditioned on their previous predictions, until the desired amount of timesteps has been predicted.

4.4.2 Data Generation Details for Motion Planning Experiments

For our training data, we restrict the object type to be a cube, and generate 1.5k videos of 300 to 400 timesteps each. The training procedure is identical to that for dynamics prediction. For inference, we guide the position of the object centroid at the final timestep for 24 timestep-long clips. The results are averaged over 20 such clips.

Table 4.1: **Evaluation in Simulation Data.** We report the Mean Squared Error (MSE), Chamfer Distance (CD), and Earth Mover’s Distance (EMD) between the predictions and ground-truth averaged over time and across scenes. The best results are **bolded** per column.

Method	In-Dist			Out-Dist		
	MSE ↓	CD ↓	EMD ↓	MSE ↓	CD ↓	EMD ↓
GNN [99]	0.0064	0.050	0.066	0.0066	0.053	0.067
ParticleWorldDiffuser (Ours)	0.0027	0.036	0.052	0.0031	0.040	0.055
ParticleWorldDiffuser-pos-grip	0.0091	0.068	0.096	0.0094	0.071	0.098
ParticleWorldDiffuser-no-augs	0.0031	0.039	0.050	0.0038	0.045	0.056
ParticleWorldDiffuser-2-recurr	0.0067	0.052	0.073	0.0089	0.062	0.088

4.5 Experiments

We evaluate ParticleWorldDiffuser on two key capabilities: predicting 3D particle trajectories given robot actions, and inferring action sequences that achieve specified object configurations. Our experiments are designed to address the following questions:

- (1) How does ParticleWorldDiffuser perform compared to existing deterministic particle dynamics models?
- (2) To what extent does ParticleWorldDiffuser generalize to unseen objects, both in simulation and in real-world data?
- (3) How does planning via guided diffusion in ParticleWorldDiffuser compare to traditional model-predictive control (MPC) frameworks in terms of task success?

4.5.1 Action-Conditioned 3D Particle Trajectory Prediction

We evaluate ParticleWorldDiffuser on its ability to predict object dynamics conditioned on input gripper actions. Specifically, given an initial object point cloud and a gripper action trajectory, the model forecasts the resulting sequence of 3D point displacements.

Datasets We test ParticleWorldDiffuser in simulation and in the real world. For evaluation in simulation, we generate two object test sets using the Genesis [3] physics engine and evaluate on clips that are 96 timesteps long, which is 4 times longer

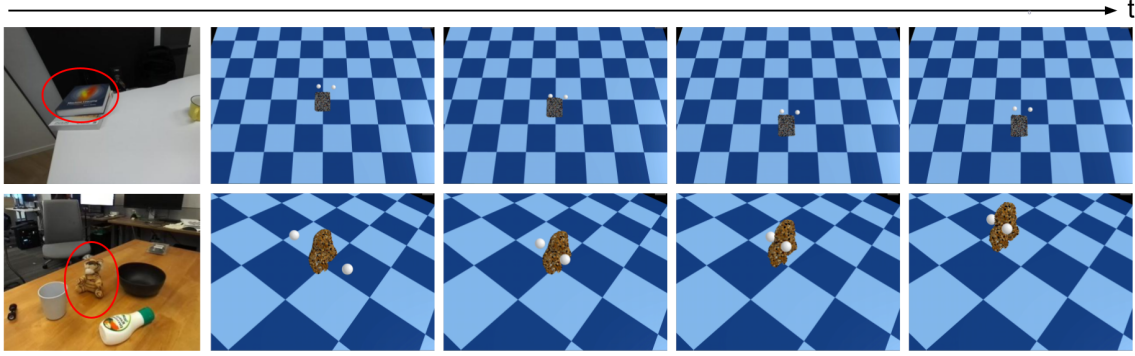


Figure 4.3: **Qualitative Sim2real Results.** We segment objects in DROID [54] videos and reconstruct them into complete 3D meshes using foundation VLM [110] and image-to-3D mesh models [140]. We then sample points on the reconstructed mesh and apply gripper actions to them using ParticleWorldDiffuser. We show a book being pushed in the *top* row and a teddy bear being pinched in the *bottom* row. For more qualitative results, please check our supplementary file.

than the training clip length: **(1) In-distribution set:** This set uses the same object categories as the training data, allowing us to assess the model’s performance under familiar conditions. **(2) Out-of-distribution set:** This set contains entirely new object meshes not seen during training, allowing a test of the model’s ability to generalize to novel shapes and dynamics. Together, these datasets span a diverse range of object categories and physical interactions, providing a comprehensive evaluation for gripper-object dynamics prediction.

For evaluation in the real world, we segment objects captured from videos in the DROID dataset [54] by prompting Gemini 2.5 [110]. Each segmented object is then reconstructed into a high-quality 3D mesh using the publicly available method of Hunyuan-3D [140]. To manipulate the object, we sample surface points on the mesh and sample meaningful actions and visualize the results, as shown in Figure 4.3. Despite being trained exclusively in simulation, ParticleWorldDiffuser predicts coherent and physically plausible object motions on these real-world reconstructions.

These results highlight the model’s robustness to moderate imperfections in geometry, scale, and surface quality. However, we note that sim-to-real adaptation remains an open challenge. In particular, performance can degrade when mesh reconstructions are highly incomplete, overly noisy, or significantly misaligned. Moreover, real-world dynamics may include unmodeled physical factors such as frictional

variation, non-rigid contacts, or unobserved forces. While our model demonstrates encouraging generalization under mild domain shift, stronger sim-to-real transfer could benefit from techniques such as domain randomization, fine-tuning with real data, or incorporating learned uncertainty estimates. We leave these directions for future work.

Evaluation Metrics For evaluation in simulation, we report the Chamfer Distance (CD), Earth Mover’s Distance (EMD), as well as the Mean Squared Error (MSE) between the predicted and ground-truth point clouds in Table 4.1. All metrics are averaged over time and across scenes.

Baselines We compare against a state-of-the-art GNN-based baseline [99]. GNN-based dynamic models [98, 99] represent the scene as a graph, where nodes correspond to objects and edges capture their spatial proximity and relationships. The model learns to predict the temporal evolution of this graph conditioned on agent actions through a graph neural network architecture, enabling it to model object dynamics. We also compare against the following ablative versions of our model:

- (1) **ParticleWorldDiffuser pos-act** replaces the gripper action from the velocity-based control to the absolute gripper position.
- (2) **ParticleWorldDiffuser w/o augs** disables our data augmentation strategies, while keeping the rest of the setup unchanged.
- (3) **ParticleWorldDiffuser-2-recurr** reduces the number of recurrent steps during training from 3 to 2.

ParticleWorldDiffuser performs favorably compared to the state-of-the-art GNN models both on in-distribution and out-of-distribution objects. While the GNN performs well on short horizons like its recurrent training setup, its performance degrades significantly when rolled out over longer timesteps, leading to compounding errors and unstable trajectories. This illustrates ParticleWorldDiffuser’s superior ability to maintain stable, accurate predictions over long horizons and generalize to novel object geometries and interaction dynamics.

Dynamics Prediction Error w.r.t. Rollout Timesteps We evaluated the rollout stability of our model by measuring the prediction error as a function of

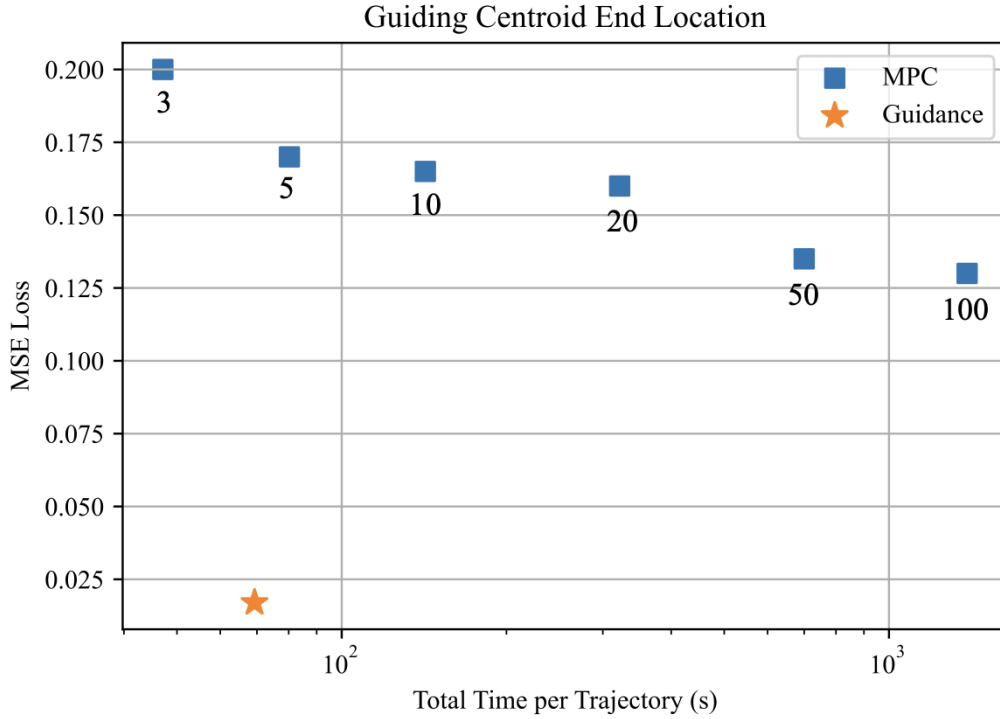


Figure 4.4: **MSE Loss between GT and Predicted Actions.** We vary the amount of random samples using our action-conditioned model+MPC and compare with our guided joint-prediction model. The guided model is both faster and more accurate than MPC.

rollout length, comparing against a GNN baseline trained under identical supervision. As shown in Figure 4.5, the GNN baseline performs competitively in short horizons, even slightly outperforming our diffusion model when $t < 24$, which aligns closely with the supervised training range. However, as the rollout proceeds, the GNN model accumulates compounding errors and gradually drifts away from the true dynamics. This accumulation of errors leads to a distributional shift, where the model inputs become increasingly out of distribution, eventually causing the GNN to produce unstable or implausible predictions.

In contrast, our diffusion-based model exhibits significantly greater robustness over long horizons, as shown in Figure 4.6. Even at $t = 90$, far beyond the training window, it maintains stable predictions and low error. We attribute this to the model’s ability to preform denoising across entire trajectories and reason about temporal consistency

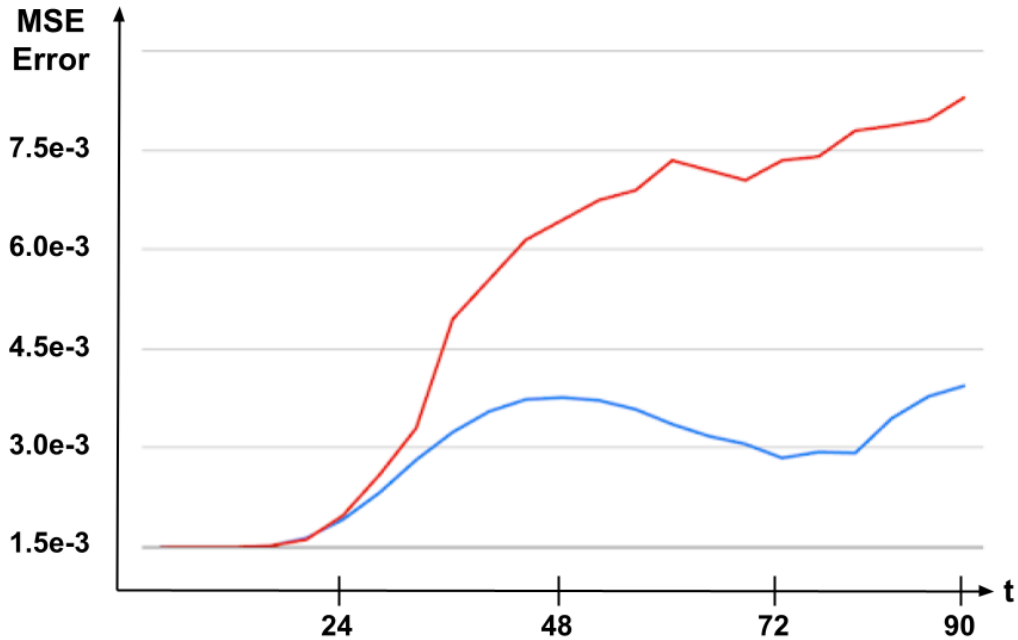


Figure 4.5: **Prediction MSE versus timestep.** ParticleWorldDiffuser (blue) is significantly more accurate compared to GNNs (red) for long term dynamics prediction.

during sampling, rather than relying solely on autoregressive updates. These results highlight the advantage of our approach in long-horizon forecasting scenarios where input drift poses a major challenge for deterministic or stepwise predictors.

Sample Diversity and Variance We analyze the generative diversity of the action-conditioned ParticleWorldDiffuser using best-of-K sampling, selecting the most accurate rollout among multiple stochastic predictions in Table 4.2. Although this evaluation relies on a ground truth oracle and is not directly deployable, it reveals the model’s ability to generate diverse and plausible futures.

As shown in the table, increasing the number of samples improves the best-case MSE, indicating that the model can indeed produce a range of plausible outcomes, among which more accurate ones can be found. This confirms that the model is not simply producing near-identical outputs but is instead generating diverse trajectories. This diversity is especially important for generalization, as it allows downstream components to reason over a richer set of possible futures.

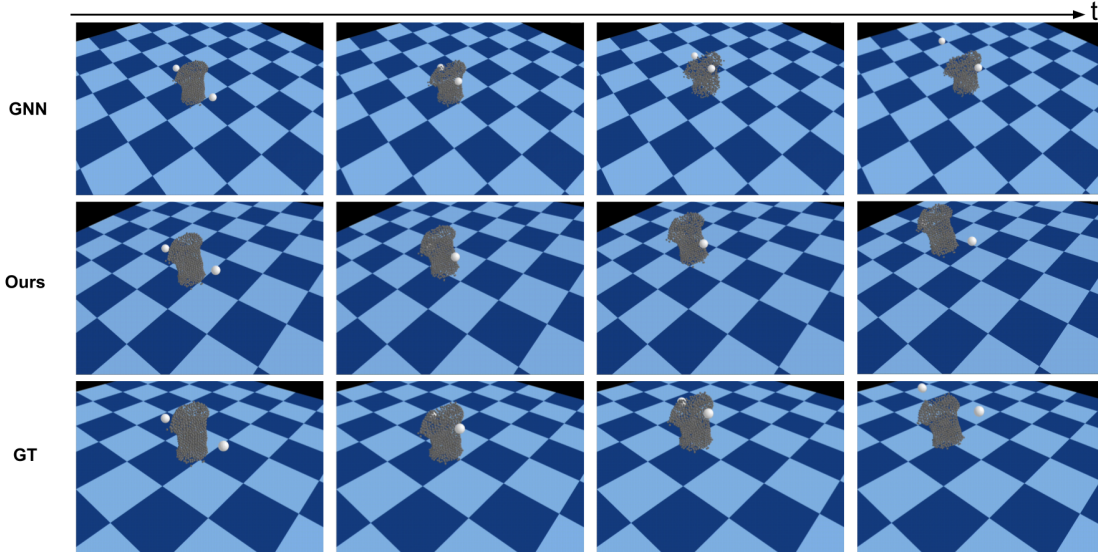


Figure 4.6: **Long Video Predictions.** ParticleWorldDiffuser is significantly more accurate compared to GNNs for long-term dynamics prediction. Object dynamics predicted by GNNs start drifting and eventually break down and stops moving as the number of timesteps increases.

Table 4.2: **Model Performance vs. Number of Samples (Best-of-K Selection).** We report the Mean Squared Error using the best sample and the variance of the MSE obtained from rolling out the model multiple times.

# of Samples	In-Dist		Out-Dist	
	Best MSE	Variance	Best MSE	Variance
1	0.0027	-	0.0031	-
3	0.0023	0.0012	0.0030	0.0016
5	0.0022	0.0011	0.0028	0.0016

In practice, while ground-truth selection is unavailable at test time, it is possible to integrate our model with downstream selection mechanisms, such as learned value functions, goal conditioning, or planning-in-the-loop, that can harness this diversity in a principled way.

For the joint-prediction model, we also observed a similar behavior, where the planned actions can have high diversity. We visualize some examples in Figure 4.7, where we can see that ParticleWorldDiffuser can predict actions from different

modalities.

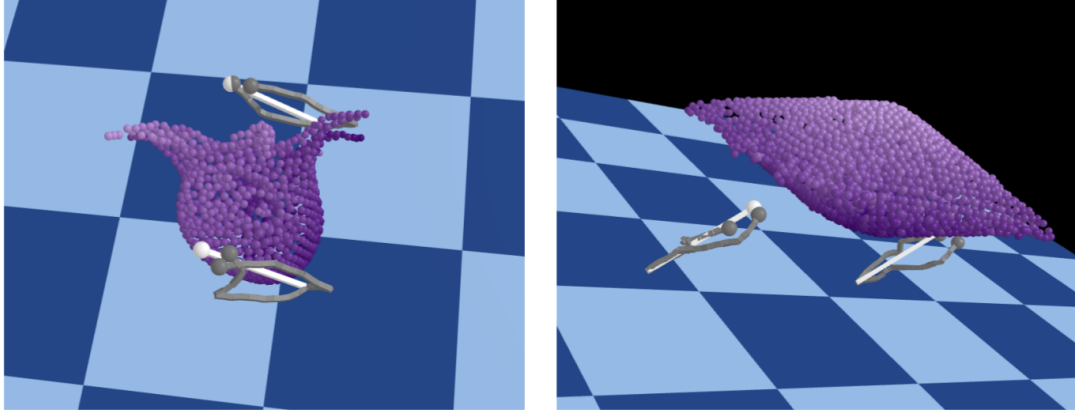


Figure 4.7: **Diversity of Planned Actions.** Due to the stochastic nature of ParticleWorldDiffuser, planned actions can also model different modalities. We show some examples, where the different planned trajectories are visualized in gray, while the GT is visualized in white. Note that the start and endpoints are similar, but the timesteps in the middle has higher variety.

Ablation Studies

As shown in Table 4.1, replacing the gripper action from the velocity-based control to the absolute gripper position results in significantly degraded performance, highlighting that gripper velocities are a more informative and stable conditioning signal for learning object dynamics over time. Disabling augmentations leads to moderate performance drops, especially in the out-of-distribution case, highlighting the role of augmentation in improving generalization. Two recurrent steps during training notably worsens performance, suggesting that multistep temporal modeling is critical for accurate long-horizon predictions. In our experiments, increasing the number of recurrence steps during training beyond 3 does not lead to noticeable improvements. Overall, the full ParticleWorldDiffuser achieves the best performance, demonstrating the importance of each component.

Table 4.3: **Planning Comparisons.** We report the Mean Squared Error (MSE) computed between the predicted and ground-truth states at the final timestep. We also report the time needed to achieve the reported planning performance.

Method	MSE (m) ↓	Planning Time (s) ↓
GNN + MPC [98, 99]	0.053	87.20
ParticleWorldDiffuser + MPC	0.190	140.26
ParticleWorldDiffuser+ Guided Diffusion	0.017	69.33

4.5.2 Planning for Object Manipulation

Next, we evaluate ParticleWorldDiffuser’s utility for planning by reversing the prediction problem: given a desired future object goal, we aim to generate a sequence of gripper actions that achieves the target object goal (trajectory or end location). For this, we use diffusion guidance as outlined in Section 4.3.2 with the variant Joint Action-Object ParticleWorldDiffuser.

Datasets and Evaluation Metrics We use a set of held-out action-object interaction video samples. We randomly sample an initial state for each test instance and define the goal as the ground-truth object position at the final timestep. The model then generates action-conditioned trajectories via guided diffusion, which we compare against the ground truth.

Baselines We consider the following baselines: **(1)** GNN+MPC [98, 99], where we combine the baseline of the graph neural network with MPC for motion planning. **(2)** Action-Conditioned ParticleWorldDiffuser + MPC, where we use the action-conditioned variant of our model with MPC.

The results can be found in Table 4.3 and in Figure 4.4. Guided diffusion dramatically outperforms MPC in both accuracy and efficiency.

We analyze the tradeoff between planning budget and control accuracy for our action-conditioned model in an MPC setting. Specifically, we vary the number of sampled action sequences (3, 5, 10, 20, 50, 100) used for trajectory rollouts within each planning step, and evaluate the corresponding prediction error or task success rate. As shown in Figure 4.4, increasing the planning budget generally leads to

improved performance, as the planner explores a broader set of candidate futures and can better optimize for long-term objectives.

Our joint-prediction model achieves strong performance while being computationally efficient, demonstrating its strong predictive prior and ability to guide effective planning. In contrast, MPC-based methods require significantly more samples to reach satisfactory performance, indicating weaker long-horizon modeling or lower sample efficiency. Since diffusion models are slower at sampling due to the iterative denoising process, we limit this analysis to 100 random samples for our action-conditioned model. We additionally provide some visualizations when we use the same amount of random samples as the GNN baseline in Figure 4.8 and compare with the results using guided diffusion. Under this scenario, the performance is slightly better than that of GNN’s, with an error of 0.052.

We further include some qualitative results of a variant of our joint-prediction modeled trained on the larger dataset of multiple object classes for guidance in Figure 4.9.

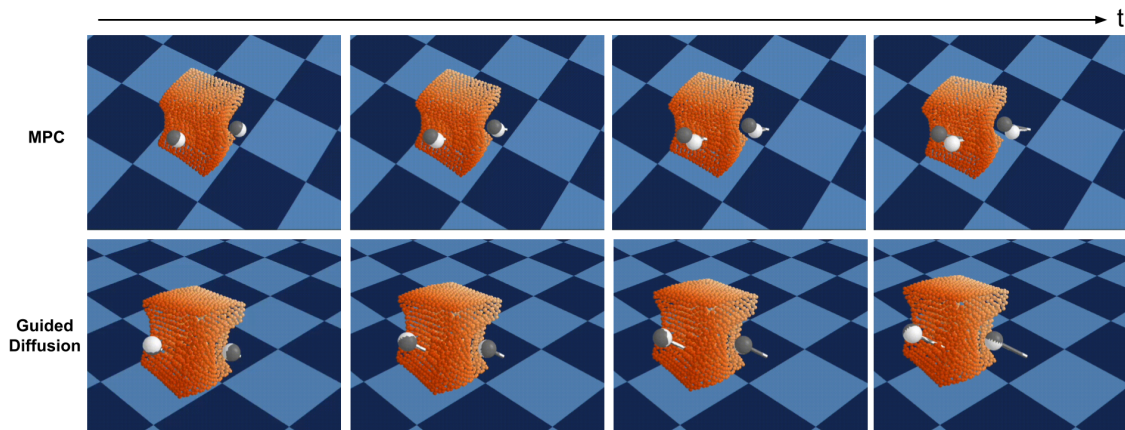


Figure 4.8: **MPC vs Guided Diffusion for Motion Planning.** We show an example of the difference between the GT actions (gray) and predicted actions (white) given the start and goal position of the object. We can observe that the results from the guided diffusion process are a lot more accurate.

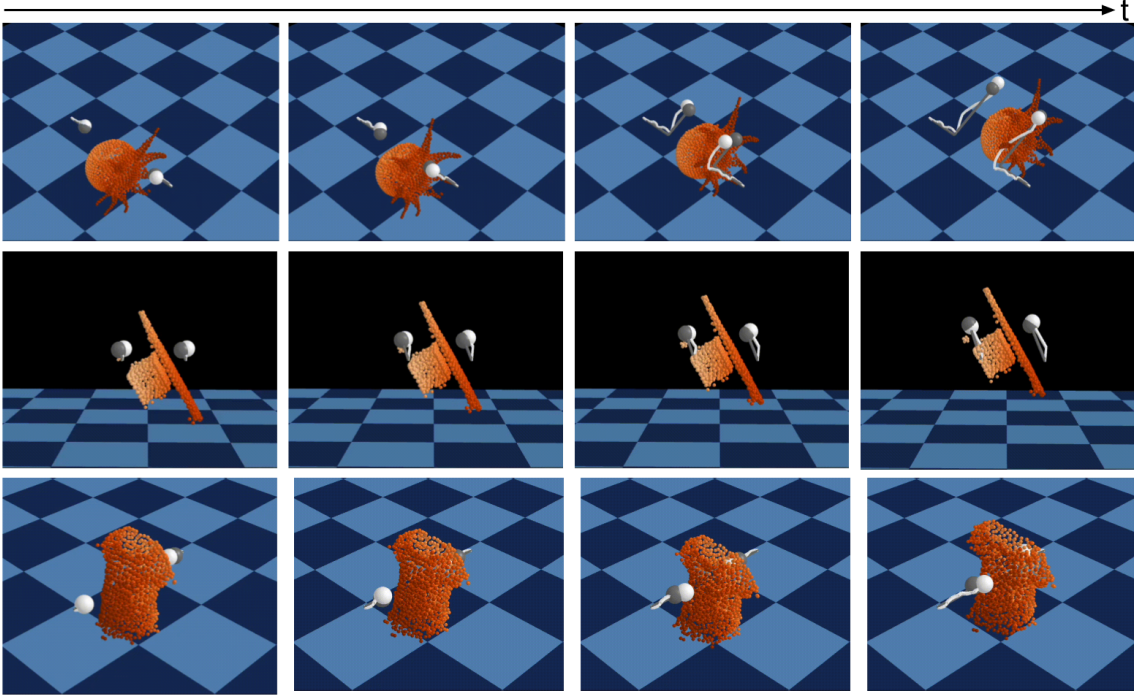


Figure 4.9: **Motion Planning via Guidance.** We use guided diffusion on a joint-prediction model trained with multiple objects. The GT actions are plotted in gray, and the predicted actions are plotted in white. The model is able to do motion planning given the goal position of the object.

4.5.3 Runtime Analysis

We evaluate the runtime performance and memory efficiency of our model on a system equipped with a single NVIDIA A6000. At inference time, generating a trajectory using DDIM/DDPM with 1000 denoising steps takes 92 seconds per sample, measured over different point trajectories of 24 frames each. The runtime scales approximately linearly with the diffusion steps, hence, our approach stands to directly benefit from recent efficient alternatives to diffusion, like Flow Matching.

4.6 Limitations

While our framework is the first to jointly model actions and fine-grained 3D object dynamics across a wide variety of objects, leveraging large-scale training in both

simulation and real-world settings, it still has several limitations:

(1) Lack of Online Adaptation: We have not yet demonstrated the model’s ability to adapt on-the-fly to a specific object as new interactions and observations become available. However, this is a promising direction enabled by recent advances in 3D point tracking [137], which allow for accurate estimation of particle trajectories through occlusions during manipulation. Our model can naturally condition on the availability of such data.

(2) Limited Conditioning Signals: Currently, the model is conditioned only on the initial point cloud configuration and, optionally, on CLIP features extracted from an image. Incorporating richer conditioning in text space, such as textual descriptions of material or physical properties, could help the model predict more accurate dynamics, by narrowing the distribution.

(3) Single-Object Interactions in Training: Our training data currently includes only single-object interaction sequences. Scaling the dataset to include more diverse action trajectories, multi-object interactions, and millions of samples is a key direction for future work.

(4) Towards Model-Based Reinforcement Learning: Extending our framework to support model-based reinforcement learning, where world model learning and action inference iterate over time, is another exciting avenue. Our structured model could enable generalization across object categories by sharing interaction knowledge, paving the way for more data-efficient, goal-directed manipulation.

4.7 Conclusion

We introduced a scalable 3D particle-based world model that combines diffusion objectives with physics-aware dynamics modeling. By jointly modeling actions and object trajectories in point cloud space, our approach enables efficient, goal-conditioned planning through guided denoising. Trained on a diverse dataset of simulated and real-world interactions, the model generalizes well across object types and domains. Compared to traditional model-predictive control, our method offers improved efficiency and long-horizon stability. This work marks the first use of guided diffusion for planning over 3D object dynamics and opens new directions for generalist, physically grounded robot control. We envision that the work will pave

the way to enable more capable and generalizable robotic manipulation systems, benefiting automation and assistive technologies, but it may also raise concerns about job displacement or misuse, causing harm in unregulated settings.

Chapter 5

Discussion and Future Directions

Understanding and modeling the world from video observations remains inherently challenging: Single-view images collapse depth information, occlusions obscure essential geometry, and interactions between multiple moving objects introduce complex and ambiguous dynamics. As introduced in Chapter 1, effectively addressing these challenges requires human-like cognitive capabilities: specifically, the ability to accurately *see* the current visible environment, *imagine* plausible completions of unseen or partially observed geometry and future dynamics, and *plan* actions to achieve desired outcomes based on these imagined scenarios. This thesis presents an integrated framework structured around these three capabilities, demonstrating how their synergistic combination yields robust and coherent mental world models that better reflect human spatial cognition.

We first address the *see* and *imagine* capabilities through **DreamScene4D** (Chapter 2) and **GenMOJO** (Chapter 3), where we attempt to create 4D scene representations from video observations. DreamScene4D reconstructs persistent, object-centric 4D representations from monocular videos, providing accurate geometry and consistent motion trajectories despite complex scene dynamics and challenging occlusions. Using learned 2D tracking priors and generative image-based priors to initialize and constrain optimization, this method effectively decomposes the observed scene into individual objects and systematically factorizes their motion into camera movements, object-centric deformations, and transformations into a global world frame. This structured representation not only achieves robust results but also lays

the critical foundation for subsequent reasoning tasks, closely aligning with how humans naturally perceive their environment.

Extending these perceptual foundations, **GenMOJO** (Chapter 3) improves DreamScene4D through joint multi-object generative modeling. Recognizing that realistic scene understanding requires reasoning beyond directly visible observations, GenMOJO simultaneously optimizes all foreground objects and the background within a single coherent representation. It utilizes score distillation sampling to enforce realistic synthesis of unseen viewpoints, rendered mask supervision to stabilize representations under occlusion, and temporal spherical harmonic color adjustments to model lighting dynamics. Thus, GenMOJO accurately reconstructs the visible and completes the occluded parts of the scene, effectively modeling intricate inter-object relationships and interactions that would otherwise remain ambiguous or incomplete. This enables a more comprehensive and human-like mental model of complex dynamic scenes.

Finally, **ParticleWorldDiffuser** (Chapter 4) extends the passive world model to an interactable world model, completing the link between the *imagining* and *planning* capabilities by focusing on predictive reasoning and actionable decision-making based on 3D scenes. ParticleWorldDiffuser frames the forecasting of 3D scene dynamics as a generative diffusion process over particle states, allowing it to model diverse object interactions and responses to agent actions. Its transformer-based architecture efficiently supports both action-conditioned forecasting, enabling predictions of object trajectories given known robot motions, and unconditional generation, jointly synthesizing plausible future robot and object trajectories. Using these sophisticated generative models, ParticleWorldDiffuser provides critical functionality for proactive planning, allowing embodied agents to evaluate potential actions, predict future outcomes, and select strategies to achieve desired goals effectively and robustly in complex and dynamic environments.

A core insight of this thesis is the synergistic integration of data-driven priors and optimization-based techniques. We observe that data-driven priors provide crucial regularization by incorporating learned constraints and general patterns extracted from extensive datasets. This regularization is essential for guiding optimization toward plausible solutions, particularly in ambiguous or under-constrained scenarios. Conversely, optimization-based methods anchor the solution firmly to the observed data, ensuring the reconstructed or generated models remain grounded and accurate.

This combination allows our methods to balance flexibility and precision, effectively resolving ambiguities, overcoming limitations inherent in each approach individually, and significantly enhancing the quality and reliability of the resulting scene and dynamics models.

An important question to consider is whether, as data-driven methods continue to grow more powerful with increasing amounts of data and computational resources, optimization-based approaches might become obsolete. Will purely data-driven approaches eventually surpass the need for explicit optimization? We argue that the answer is no; instead, optimization-based methods will evolve and adapt to leverage increasingly sophisticated data-driven priors. In this evolving paradigm, optimization techniques will serve primarily as refinement modules, operating on top of powerful learned representations to enhance precision and ensure consistency with observed data. This observation has been consistently reinforced by our experiences with DreamScene4D and GenMOJO, as well as in other contemporary research [114], where purely learning-based methods often outperform purely optimization-based approaches in terms of initial accuracy and generalization. However, explicitly applying optimization as a refinement significantly enhances the final quality beyond initial predictions. Furthermore, this complementary relationship motivates future research directions focused on integrating optimization and learning-based methods more deeply: learning-based approaches serve as foundational priors for optimization-based refinement, while the refinement outcomes help identify limitations and gaps within purely learned approaches, thereby driving iterative improvements and the evolution of even more powerful and precise models. Indeed, our work demonstrates that pushing the boundaries of reconstruction and dynamics modeling often relies on the strategic and clever integration of data-driven priors with carefully designed optimization routines instead of handcrafting extra optimization objectives.

Another open challenge highlighted by this thesis is the missing link between 4D scene generation from video and using these scene representations directly for learning dynamics and planning. Although DreamScene4D and GenMOJO produce high-quality geometric and motion estimates from real-world monocular videos, these reconstructions still fall short of the precision required for robust dynamics learning and closed-loop control. Even small errors, such as drift in low-texture regions, temporal inconsistencies, or inaccuracies under occlusion, can significantly degrade

the quality of motion data used for training. Furthermore, state-of-the-art 2D and 3D trackers [49, 124] are also unable to produce accurate and robust motion trajectories for training. As a result, ParticleWorldDiffuser currently relies on high-quality simulated trajectories with perfect ground truth for supervision. Hybrid strategies that incorporate real-world reconstructions into training, although intuitive, did not yield improvements in practice due to the noise in the extracted trajectories. Bridging this gap will require new research into robust and uncertainty-aware methods for converting imperfect 4D reconstructions into usable supervision for dynamics models. This may involve probabilistic representations, refinement loops, or joint training frameworks that align reconstruction and dynamics objectives. Ultimately, closing this loop, from monocular video to world model to actionable policy, remains an exciting and necessary direction for building truly generalizable, interactive 3D agents.

Looking ahead, the work presented in this thesis offers a hopeful trajectory for the future of spatial intelligence. By embracing the complementary roles of perception, imagination, and planning, and grounding them in unified 3D world models, we move closer to building agents that can understand and interact with their environments in ways that mirror human intuition. While many challenges remain, the methods and insights developed here lay a strong foundation for future systems that see beyond the pixels, imagine possibilities beyond the observed, and act with foresight in the real world.

Appendix A

User Study Procedure for DreamScene4D and GenMOJO

For the user study, we consider the videos in MOSE-PTS. For each method, we render from the reference view, as well as 2 novel views corresponding to the input video. We use Amazon Turk to outsource evaluations and ask the workers to compare the 2 sets of 3 videos with the input video. Each set of videos is reviewed by 30 workers with a HIT rate of over 95% for a total of 900 answers collected. The whole user preference study takes about 89s per question and 22.25 working hours in total.

To filter out poor quality responses, we intentionally included “dummy” questions where one set of renders was replaced by renders from non-converged optimization where the visual quality is noticeably worse, and used these “dummy” questions to filter out workers who submitted responses that did not pass these questions. We additionally filtered out workers who submitted the same answer for all the videos. For every worker whose answers were rejected, we assigned new ones until the desired number of answers had been collected.

We include the full instructions shown to the workers below:

Please read the instructions and check the videos carefully.

Please take a look at the reference video on the top. There are 2 sets of rendered videos (A and B) for this reference video: one rendered from the original viewpoint, and two rendered from other viewpoints. Please

A. User Study Procedure for DreamScene4D and GenMOJO

choose the set of videos that looks more realistic and better represents the original video to you. The options (A) and (B) correspond to the two sets of videos, as denoted in the captions under the videos.

To judge the quality of the videos, consider the following points, listed in order of importance:

- 1. Do the objects in the rendered videos correspond to the reference video?*
- 2. Do the objects clip or penetrate each other in the rendered videos?*
- 3. Does the video look geometrically correct (e.g. not overly flat) when observed from other viewpoints?*
- 4. Are there any visual artifacts (e.g. floaters, weird textures) in the rendered videos?*

Please start from the first criteria to select the better set of renderings. If they are equal, please use the next criteria. If they are equal for all 4 points, please select Equally Preferred.

Please ignore the background in the original video.


A GUI sample of a survey question is also provided in Figure [A.1](#) for reference.

A. User Study Procedure for DreamScene4D and GenMOJO


Instructions | **Shortcuts** | Please take a look at the full instructions and the reference video on the top. There are 2 sets of rendered videos (A and B) for this reference video: one rendered from the original viewpoint, and two rendered from other viewpoints. Please choose the set of videos that better represent.



Reference Video



Rendered Video (original view) (A) Rendered Video (new view 1) (A) Rendered Video (new view 2) (A)



Rendered Video (original view) (B) Rendered Video (new view 1) (B) Rendered Video (new view 2) (B)

Select an option

Prefer (A)	1
Prefer (B)	2
Equally Preferred	3

Submit

Figure A.1: **User survey interface.** A GUI sample of what an Amazon Turk worker sees for the user study.

A. User Study Procedure for DreamScene4D and GenMOJO

Bibliography

- [1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. [4.4](#)
- [2] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022. [4.2.2](#)
- [3] Genesis Authors. Genesis: A universal and generative physics engine for robotics and beyond. URL <https://github.com/Genesis-Embodied-AI/Genesis>, 2024. [4.4](#), [4.5.1](#)
- [4] Sherwin Bahmani, Ivan Skorokhodov, Victor Rong, Gordon Wetzstein, Leonidas Guibas, Peter Wonka, Sergey Tulyakov, Jeong Joon Park, Andrea Tagliasacchi, and David B Lindell. 4d-fy: Text-to-4d generation using hybrid score distillation sampling. *arXiv preprint arXiv:2311.17984*, 2023. [2.1](#), [2.2.2](#), [3.2](#)
- [5] Sherwin Bahmani, Xian Liu, Yifan Wang, Ivan Skorokhodov, Victor Rong, Ziwei Liu, Xihui Liu, Jeong Joon Park, Sergey Tulyakov, Gordon Wetzstein, et al. Tc4d: Trajectory-conditioned text-to-4d generation. *arXiv preprint arXiv:2403.17920*, 2024. [2.1](#), [2.2.2](#), [3.2](#)
- [6] Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in Neural Information Processing Systems*, 2016. [4.2.1](#)
- [7] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. [2.3.3](#)
- [8] Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, et al. Video generation models as world simulators. *OpenAI Blog*, 2024. [4.1](#)
- [9] Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi,

- Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *International Conference on Machine Learning (ICML)*, 2024. [4.2.1](#)
- [10] Marcel Büsching, Josef Bengtson, David Nilsson, and Mårten Björkman. Flowibr: Leveraging pre-training for efficient neural image-based rendering of dynamic scenes. *arXiv preprint arXiv:2309.05418*, 2023. [2.2.1](#), [3.2](#)
- [11] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. [2.1](#), [2.2.1](#), [2.3.3](#), [2.4.1](#), [3.2](#), [3.3.2](#), [3.5.2](#)
- [12] Duygu Ceylan, Chun-Hao P Huang, and Niloy J Mitra. Pix2video: Video editing using image diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [2.3.3](#)
- [13] Wenhao Chai, Xun Guo, Gaoang Wang, and Yan Lu. Stablevideo: Text-driven consistency-aware diffusion video editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [1](#), [4.1](#)
- [14] Chang Chen, Fei Deng, Kenji Kawaguchi, Caglar Gulcehre, and Sungjin Ahn. Simple hierarchical planning with diffusion. *arXiv preprint arXiv:2401.02644*, 2024. [4.2.2](#)
- [15] Ho Kei Cheng and Alexander G Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *European Conference in Computer Vision*, 2022. [2.1](#), [2.3.2](#), [2.3.2](#)
- [16] Wen-Hsuan Chu, Adam W Harley, Pavel Tokmakov, Achal Dave, Leonidas Guibas, and Katerina Fragkiadaki. Zero-shot open-vocabulary tracking with large pre-trained models. *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. [2.3.2](#), [2.3.2](#)
- [17] Wen-Hsuan Chu, Lei Ke, and Katerina Fragkiadaki. Dreamscene4d: Dynamic multi-object scene generation from monocular videos. In *Advances in Neural Information Processing Systems*, 2024. [3.1](#), [3.1](#), [3.2](#), [3.2](#), [3.3.2](#), [3.3.2](#), [3.5.1](#), [3.5.1](#), [3.5.2](#), [3.5.2](#), [3.5.2](#), [3.6](#), [3.6.1](#), [3.2](#), [3.6.2](#), [3.6.2](#), [3.3](#), [3.6.2](#), [3.6.4](#)
- [18] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in Neural Information Processing Systems*, 31, 2018. [4.1](#)
- [19] K. J. W. Craik. *The Nature of Explanation*. University Press, Macmillan, 1943. [1](#), [4.1](#)
- [20] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the*

- IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. [2.3.3](#), [4.4](#)
- [21] Henghui Ding, Chang Liu, Shuting He, Xudong Jiang, Philip HS Torr, and Song Bai. Mose: A new dataset for video object segmentation in complex scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [3.1](#), [3.2](#), [3.4](#), [3.4.2](#), [3.4.2](#)
 - [22] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. In *Advances in Neural Information Processing Systems*, 2022. [2.5.1](#), [2.2](#), [3.3](#), [3.4](#), [3.4.1](#), [3.4.2](#), [3.4.3](#), [3.1](#), [3.6](#), [3.3](#), [3.6.2](#)
 - [23] Zibin Dong, Yifu Yuan, Jianye Hao, Fei Ni, Yao Mu, Yan Zheng, Yujing Hu, Tangjie Lv, Changjie Fan, and Zhipeng Hu. Aligndiff: Aligning diverse human preferences via behavior-customisable diffusion model. *arXiv preprint arXiv:2310.02054*, 2023. [4.2.2](#)
 - [24] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. [1](#)
 - [25] Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 2023. [4.2.2](#)
 - [26] Xin Fei, Wenzhao Zheng, Yueqi Duan, Wei Zhan, Masayoshi Tomizuka, Kurt Keutzer, and Jiwen Lu. Driv3r: Learning dense 4d reconstruction for autonomous driving. *arXiv preprint arXiv:2412.06777*, 2024. [1](#)
 - [27] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. [2.2.1](#), [2.3.3](#), [3.2](#)
 - [28] Pascale Fung, Yoram Bachrach, Asli Celikyilmaz, Kamalika Chaudhuri, DeLong Chen, Willy Chung, Emmanuel Dupoux, Hervé Jégou, Alessandro Lazaric, Arjun Majumdar, et al. Embodied ai agents: Modeling the world. *arXiv preprint arXiv:2506.22355*, 2025. [1](#)
 - [29] 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. [2.1](#), [2.2.2](#), [3.2](#)
 - [30] Shubham Goel, Georgios Pavlakos, Jathushan Rajasegaran, Angjoo Kanazawa, and Jitendra Malik. Humans in 4d: Reconstructing and tracking humans with

- transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [1](#)
- [31] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. [2.1](#), [2.5.1](#), [2.1](#), [2.2](#), [3.2](#), [3.3](#), [3.4](#)
- [32] Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023. [4.2.1](#)
- [33] Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022. [4.1](#)
- [34] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, 2022. [3.6.2](#)
- [35] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [1](#)
- [36] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, 2017. [1](#)
- [37] Yutong He, Naoki Murata, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Dongjun Kim, Wei-Hsiang Liao, Yuki Mitsufuji, J Zico Kolter, Ruslan Salakhutdinov, et al. Manifold preserving guided diffusion. *arXiv preprint arXiv:2311.16424*, 2023. [4.3.2](#)
- [38] Eric Heiden, Ziang Liu, Vibhav Vineet, Erwin Coumans, and Gaurav S Sukhatme. Inferring articulated rigid body dynamics from rgb-d video. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022. [2.1](#), [3.1](#)
- [39] Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023. [4.2.1](#)
- [40] Mu Hu, Wei Yin, Chi Zhang, Zhipeng Cai, Xiaoxiao Long, Hao Chen, Kaixuan Wang, Gang Yu, Chunhua Shen, and Shaojie Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *arXiv preprint arXiv:2404.15506*, 2024. [1](#)
- [41] Wenbo Hu, Xiangjun Gao, Xiaoyu Li, Sijie Zhao, Xiaodong Cun, Yong Zhang, Long Quan, and Ying Shan. Depthcrafter: Generating consistent long depth

- sequences for open-world videos. *arXiv preprint arXiv:2409.02095*, 2024. [3.3.2](#), [3.3.2](#)
- [42] Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3d world. *arXiv preprint arXiv:2311.12871*, 2023. [1](#)
- [43] Xiaoyu Huang, Takara Truong, Yunbo Zhang, Fangzhou Yu, Jean Pierre Sleiman, Jessica Hodgins, Koushil Sreenath, and Farbod Farshidian. Diffuse-cloc: Guided diffusion for physics-based character look-ahead control. *ACM Transactions on Graphics (TOG)*, 2025. [4.2.2](#)
- [44] Allan Jabri, David Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. *arXiv preprint arXiv:2212.11972*, 2022. [4.1](#), [4.3.2](#), [4.3.2](#)
- [45] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022. [4.2.2](#)
- [46] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022. [4.2.2](#)
- [47] Yanqin Jiang, Li Zhang, Jin Gao, Weimin Hu, and Yao Yao. Consistent4d: Consistent 360 $\{\backslash\deg\}$ dynamic object generation from monocular video. In *International Conference on Learning Representations (ICLR)*, 2024. [2.1](#), [2.2.2](#), [2.5.1](#), [2.5.2](#), [2.5.3](#), [2.5.3](#), [2.5.4](#), [2.1](#), [2.4](#), [3.1](#), [3.1](#), [3.2](#), [3.6](#), [3.6.1](#), [3.6.1](#), [3.2](#), [3.6.2](#)
- [48] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023. [2.5.4](#), [2.2](#), [2.5](#), [3.2](#), [3.4.2](#), [3.4.3](#), [3.1](#), [3.6](#), [3.3](#), [3.6.2](#), [3.6.2](#)
- [49] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024. [3.4.3](#), [3.1](#), [3.6](#), [3.3](#), [3.6.2](#), [3.6.2](#), [5](#)
- [50] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [1](#)
- [51] Lei Ke, Mingqiao Ye, Martin Danelljan, Yifan Liu, Yu-Wing Tai, Chi-Keung Tang, and Fisher Yu. Segment anything in high quality. In *Advances in Neural Information Processing Systems*, 2023. [2.3.2](#), [2.3.2](#), [3.3.2](#)

- [52] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 2023. [1](#), [2.1](#), [2.2.1](#), [2.3](#), [2.3.1](#), [2.3.3](#), [3.2](#), [3.3](#), [3.3.1](#), [3.3.2](#)
- [53] Levon Khachatryan, Andranik Movsisyan, Vahram Tadevosyan, Roberto Henschel, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Text2video-zero: Text-to-image diffusion models are zero-shot video generators. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [2.3.3](#)
- [54] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. [4.3](#), [4.5.1](#)
- [55] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [1](#)
- [56] Yuseung Lee, Kunho Kim, Hyunjin Kim, and Minhyuk Sung. Syncdiffusion: Coherent montage via synchronized joint diffusions. *Advances in Neural Information Processing Systems*, 2023. [2.3.3](#)
- [57] Jiahui Lei, Yijia Weng, Adam Harley, Leonidas Guibas, and Kostas Daniilidis. Mosca: Dynamic gaussian fusion from casual videos via 4d motion scaffolds. *arXiv preprint arXiv:2405.17421*, 2024. [3.2](#)
- [58] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. [2.1](#), [2.2.1](#), [2.2.2](#), [3.2](#)
- [59] Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision making. In *International Conference on Machine Learning (ICML)*, 2023. [4.2.2](#)
- [60] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018. [4.1](#), [4.2.1](#)
- [61] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. Megasam: Accurate, fast and robust structure and motion from casual dynamic videos. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. [3.3.2](#)

- [62] Zhixuan Liang, Yao Mu, Mingyu Ding, Fei Ni, Masayoshi Tomizuka, and Ping Luo. Adaptdiffuser: Diffusion models as adaptive self-evolving planners. *arXiv preprint arXiv:2302.01877*, 2023. [4.2.2](#)
- [63] Huan Ling, Seung Wook Kim, Antonio Torralba, Sanja Fidler, and Karsten Kreis. Align your gaussians: Text-to-4d with dynamic 3d gaussians and composed diffusion models. *arXiv preprint arXiv:2312.13763*, 2023. [2.1](#), [2.2.2](#), [3.2](#)
- [64] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 2020. [2.1](#), [2.2.1](#), [3.2](#)
- [65] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems*, 2024. [1](#)
- [66] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [1](#), [2.1](#), [2.3.1](#), [2.3.3](#), [3.1](#), [3.2](#), [3.3](#), [3.3.1](#), [3.6](#)
- [67] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023. [1](#)
- [68] Youquan Liu, Runnan Chen, Xin Li, Lingdong Kong, Yuchen Yang, Zhaoyang Xia, Yeqi Bai, Xinge Zhu, Yuexin Ma, Yikang Li, et al. Uniseg: A unified multi-modal lidar segmentation network and the openpcseg codebase. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [1](#)
- [69] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. [2.2.1](#), [3.2](#)
- [70] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. [1](#)
- [71] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *arXiv preprint arXiv:1906.07751*, 2019. [2.1](#), [2.2.1](#), [3.2](#)
- [72] Haofei Lu, Dongqi Han, Yifei Shen, and Dongsheng Li. What makes a good

- diffusion planner for decision making? *arXiv preprint arXiv:2503.00535*, 2025. [4.2.2](#)
- [73] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. In *International Conference on 3D Vision (3DV)*, 2024. [2.2.1](#), [2.3.3](#), [3.2](#), [3.2](#), [3.3.2](#)
 - [74] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference in Computer Vision*, 2020. [2.1](#), [2.2.1](#), [3.2](#)
 - [75] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. *Advances in Neural Information Processing Systems*, 28, 2015. [4.1](#)
 - [76] Zijie Pan, Zeyu Yang, Xiatian Zhu, and Li Zhang. Fast dynamic 3d object generation from a single-view video. *arXiv preprint arXiv 2401.08742*, 2024. [2.1](#), [2.2.2](#), [3.2](#)
 - [77] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. [1](#), [2.1](#), [2.2.1](#), [3.2](#)
 - [78] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M. Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *ACM Trans. Graph.*, 2021. [2.2.1](#)
 - [79] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *SIGGRAPH*, 2023. [2.3.3](#)
 - [80] Xue Bin Peng, Angjoo Kanazawa, Jitendra Malik, Pieter Abbeel, and Sergey Levine. Sfv: Reinforcement learning of physical skills from videos. *ACM Transactions On Graphics (TOG)*, 2018. [2.1](#)
 - [81] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017. [2.1](#), [2.5.1](#), [2.1](#), [2.2](#), [3.6](#), [3.2](#), [3.3](#)
 - [82] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *International Conference on Learning Representations (ICLR)*, 2023. [2.2.2](#), [2.3.1](#), [2.3.2](#), [2.3.3](#), [3.2](#), [3.3.1](#), [3.3.2](#)
 - [83] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-

- Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. [2.1](#), [2.2.1](#), [3.2](#)
- [84] Chenyang Qi, Xiaodong Cun, Yong Zhang, Chenyang Lei, Xintao Wang, Ying Shan, and Qifeng Chen. Fatezero: Fusing attentions for zero-shot text-based video editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [2.3.3](#)
- [85] Guocheng Qian, Yuchen Li, Houwen Peng, Jinjie Mai, Hasan Hammoud, Mohamed Elhoseiny, and Bernard Ghanem. Pointnext: Revisiting pointnet++ with improved training and scaling strategies. *Advances in Neural Information Processing Systems*, 35:23192–23204, 2022. [4.3.2](#)
- [86] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*, 2021. [2.5.2](#), [3.6.1](#)
- [87] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *TPAMI*, 2022. [2.3.3](#)
- [88] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. [1](#)
- [89] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. [1](#), [3.3.2](#), [3.3.2](#)
- [90] 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. [1](#), [2.1](#), [2.2.2](#), [2.3.3](#), [2.3.3](#), [2.4.2](#), [2.4.3](#), [2.5.2](#), [2.5.3](#), [2.5.3](#), [2.5.4](#), [2.1](#), [2.2](#), [2.4](#), [3.1](#), [3.2](#), [3.3.1](#), [3.3.2](#), [3.5.1](#), [3.5.1](#), [3.5.2](#), [3.5.2](#), [3.6](#), [3.6.1](#), [3.6.1](#), [3.2](#), [3.6.2](#), [3.3](#), [3.6.4](#)
- [91] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kunchang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024. [2.3.2](#), [2.3.2](#), [3.3.2](#)
- [92] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and

- Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. [1](#), [2.3.2](#), [2.3.3](#), [3.6](#)
- [93] Alvaro Sanchez-Gonzalez, Jonathan Godwin, Tobias Pfaff, Rex Ying, Jure Leskovec, and Peter Battaglia. Learning to simulate complex physics with graph networks. In *International Conference on Machine Learning (ICML)*, 2020. [4.2.1](#), [4.3](#)
- [94] Kyle Sargent, Zizhang Li, Tanmay Shah, Charles Herrmann, Hong-Xing Yu, Yunzhi Zhang, Eric Ryan Chan, Dmitry Lagun, Li Fei-Fei, Deqing Sun, et al. Zeronvs: Zero-shot 360-degree view synthesis from a single real image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [3.1](#), [3.6](#), [3.6.3](#), [3.4](#)
- [95] Jenny Seidenschwarz, Qunjie Zhou, Bardienus Duisterhof, Deva Ramanan, and Laura Leal-Taixé. Dynomo: Online point tracking by dynamic online monocular gaussian reconstruction. *arXiv preprint arXiv:2409.02104*, 2024. [3.2](#)
- [96] Yue Shan, Jun Xiao, Lupeng Liu, Yunbiao Wang, Dongbo Yu, and Wenniu Zhang. A coarse-to-fine transformer-based network for 3d reconstruction from non-overlapping multi-view images. *Remote Sensing*, 2024. [1](#)
- [97] Yichao Shen, Zigang Geng, Yuhui Yuan, Yutong Lin, Ze Liu, Chunyu Wang, Han Hu, Nanning Zheng, and Baining Guo. V-detr: Detr with vertex relative position encoding for 3d object detection. *arXiv preprint arXiv:2308.04409*, 2023. [1](#)
- [98] Haochen Shi, Huazhe Xu, Samuel Clarke, Yunzhu Li, and Jiajun Wu. Robocook: Long-horizon elasto-plastic object manipulation with diverse tools. *arXiv preprint arXiv:2306.14447*, 2023. [1](#), [4.1](#), [4.2.1](#), [4.5.1](#), [4.3](#), [4.5.2](#)
- [99] Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects in 3d with graph networks. *The International Journal of Robotics Research*, 2024. [4.2.1](#), [4.1](#), [4.5.1](#), [4.3](#), [4.5.2](#)
- [100] Haochen Shi, Huazhe Xu, Zhiao Huang, Yunzhu Li, and Jiajun Wu. Robocraft: Learning to see, simulate, and shape elasto-plastic objects in 3d with graph networks. *The International Journal of Robotics Research*, 43(4):533–549, 2024. [1](#), [4.1](#)
- [101] Yining Shi, Jiusi Li, Kun Jiang, Ke Wang, Yunlong Wang, Mengmeng Yang, and Diange Yang. Panossc: Exploring monocular panoptic 3d scene reconstruction for autonomous driving. *arXiv preprint arXiv:2406.07037*, 2024. [1](#)
- [102] Ayush Shrivastava and Andrew Owens. Self-supervised any-point tracking by contrastive random walks. *European Conference in Computer Vision*, 2024. [3.3](#),

3.6.2

- [103] Uriel Singer, Shelly Sheynin, Adam Polyak, Oron Ashual, Iurii Makarov, Filippos Kokkinos, Naman Goyal, Andrea Vedaldi, Devi Parikh, Justin Johnson, et al. Text-to-4d dynamic scene generation. *arXiv preprint arXiv:2301.11280*, 2023. [2.1](#), [2.2.2](#), [3.2](#)
- [104] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *International Conference on Learning Representations (ICLR)*, 2021. [2.3.3](#), [4.3.1](#)
- [105] Elizabeth S Spelke. Core knowledge. *American psychologist*, 55(11):1233, 2000. [1](#)
- [106] Colton Stearns, Adam Harley, Mikaela Uy, Florian Dubost, Federico Tombari, Gordon Wetzstein, and Leonidas Guibas. Dynamic gaussian marbles for novel view synthesis of casual monocular videos. *arXiv preprint arXiv:2406.18717*, 2024. [3.2](#)
- [107] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. RoFormer: Enhanced Transformer with Rotary Position Embedding. In *International Conference on Learning Representations (ICLR)*, 2020. [4.3.2](#)
- [108] Jiaxiang Tang, Zhaoxi Chen, Xiaokang Chen, Tengfei Wang, Gang Zeng, and Ziwei Liu. Lgm: Large multi-view gaussian model for high-resolution 3d content creation. *arXiv preprint arXiv:2402.05054*, 2024. [1](#)
- [109] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *International Conference on Learning Representations (ICLR)*, 2024. [2.3.1](#), [2.3.3](#), [2.2](#), [2.4.2](#), [2.4.3](#), [3.3.1](#), [3.5.1](#), [3.5.1](#)
- [110] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. [4.3](#), [4.5.1](#)
- [111] Neea Tuomainen, David Blanco-Mulero, and Ville Kyrki. Manipulation of granular materials by learning particle interactions. *IEEE Robotics and Automation Letters*, 2022. [4.2.1](#), [4.3](#)
- [112] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017. [4.2.1](#)
- [113] Dan Wang, Xinrui Cui, Xun Chen, Zhengxia Zou, Tianyang Shi, Septimiu Salcudean, Z Jane Wang, and Rabab Ward. Multi-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF International Conference on*

- Computer Vision (ICCV)*, 2021. [1](#)
- [114] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vgggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, 2025. [5](#)
 - [115] Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking everything everywhere all at once. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [3.2](#)
 - [116] Qianqian Wang, Vickie Ye, Hang Gao, Jake Austin, Zhengqi Li, and Angjoo Kanazawa. Shape of motion: 4d reconstruction from a single video. *arXiv preprint arXiv:2407.13764*, 2024. [3.1](#), [3.2](#), [3.2](#), [3.3.2](#), [3.6](#), [3.6.2](#), [3.3](#), [3.6.2](#), [3.6.2](#)
 - [117] Shizun Wang, Xingyi Yang, Qiuhong Shen, Zhenxiang Jiang, and Xinchao Wang. Gflow: Recovering 4d world from monocular video. *arXiv preprint arXiv:2405.18426*, 2024. [3.2](#)
 - [118] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [2.3.3](#)
 - [119] Yixuan Wang, Yunzhu Li, Katherine Driggs-Campbell, Li Fei-Fei, and Jiajun Wu. Dynamic-resolution model learning for object pile manipulation. *arXiv preprint arXiv:2306.16700*, 2023. [4.1](#), [4.2.1](#), [4.3](#)
 - [120] Bowen Wen, Wei Yang, Jan Kautz, and Stan Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [1](#)
 - [121] Guanjuan Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [1](#), [2.2.1](#), [2.2.2](#), [2.3](#), [3.2](#), [3.3](#)
 - [122] Jay Zhangjie Wu, Yixiao Ge, Xintao Wang, Stan Weixian Lei, Yuchao Gu, Yufei Shi, Wynne Hsu, Ying Shan, Xiaohu Qie, and Mike Zheng Shou. Tune-a-video: One-shot tuning of image diffusion models for text-to-video generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [2.3.3](#)
 - [123] Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference*

- on *Robot Learning (CORL)*, 2023. [4.2.1](#)
- [124] Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [1](#), [5](#)
 - [125] Tianyi Xie, Zeshun Zong, Yuxing Qiu, Xuan Li, Yutao Feng, Yin Yang, and Chenfanfu Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [3.1](#)
 - [126] Dejia Xu, Hanwen Liang, Neel P Bhatt, Hezhen Hu, Hanxue Liang, Konstantinos N Plataniotis, and Zhangyang Wang. Comp4d: Llm-guided compositional 4d scene generation. *arXiv preprint arXiv:2403.16993*, 2024. [2.1](#)
 - [127] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. [2.3.3](#)
 - [128] Cheng-Fu Yang, Haoyang Xu, Te-Lin Wu, Xiaofeng Gao, Kai-Wei Chang, and Feng Gao. Planning as in-painting: A diffusion-based embodied task planning framework for environments under uncertainty. *arXiv preprint arXiv:2312.01097*, 2023. [4.2.2](#)
 - [129] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. [1](#), [2.3.3](#), [3.2](#)
 - [130] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. *Advances in Neural Information Processing Systems*, 2024. [3.4](#)
 - [131] Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023. [4.2.1](#), [4.2.2](#)
 - [132] Qitong Yang, Mingtao Feng, Zijie Wu, Shijie Sun, Weisheng Dong, Yaonan Wang, and Ajmal Mian. Beyond skeletons: Integrative latent mapping for coherent 4d sequence generation. *arXiv preprint arXiv:2403.13238*, 2024. [2.1](#), [2.2.2](#), [3.2](#)
 - [133] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023. [2.2.1](#), [3.2](#)

- [134] Vickie Ye, Georgios Pavlakos, Jitendra Malik, and Angjoo Kanazawa. Decoupling human and camera motion from videos in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. [2.3.3](#)
- [135] Yuyang Yin, Dejia Xu, Zhangyang Wang, Yao Zhao, and Yunchao Wei. 4dgen: Grounded 4d content generation with spatial-temporal consistency. *arXiv preprint arXiv:2312.17225*, 2023. [2.1](#), [2.2.2](#), [3.1](#), [3.2](#)
- [136] Yifei Zeng, Yanqin Jiang, Siyu Zhu, Yuanxun Lu, Youtian Lin, Hao Zhu, Weiming Hu, Xun Cao, and Yao Yao. Stag4d: Spatial-temporal anchored generative 4d gaussians. *arXiv preprint arXiv:2403.14939*, 2024. [2.1](#), [2.2.2](#), [3.1](#), [3.2](#)
- [137] Bowei Zhang, Lei Ke, Adam W Harley, and Katerina Fragkiadaki. Tapip3d: Tracking any point in persistent 3d geometry. *arXiv preprint arXiv:2504.14717*, 2025. [4.6](#)
- [138] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. [2.5.2](#), [3.6.1](#)
- [139] 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. [2.1](#), [2.2.2](#)
- [140] Zibo Zhao, Zeqiang Lai, Qingxiang Lin, Yunfei Zhao, Haolin Liu, Shuhui Yang, Yifei Feng, Mingxin Yang, Sheng Zhang, Xianghui Yang, et al. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation. *arXiv preprint arXiv:2501.12202*, 2025. [4.3](#), [4.5.1](#)
- [141] Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. [2.5.2](#), [2.5.4](#), [2.5.4](#), [2.2](#), [2.5](#), [3.6.2](#), [3.3](#)
- [142] Yufeng Zheng, Xueting Li, Koki Nagano, Sifei Liu, Otmar Hilliges, and Shalini De Mello. A unified approach for text-and image-guided 4d scene generation. *arXiv preprint arXiv:2311.16854*, 2023. [2.1](#), [2.2.2](#), [3.2](#)
- [143] Siyuan Zhou, Yilun Du, Yuncong Yang, Lei Han, Peihao Chen, Dit-Yan Yeung, and Chuang Gan. Learning 3d persistent embodied world models. *arXiv preprint arXiv:2505.05495*, 2025. [1](#)
- [144] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *European Conference on Computer Vision*, 2022. [1](#)

- [145] Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Irasim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024. [4.1](#), [4.2.1](#)