NeoVerse: Enhancing 4D World Model with in-the-wild Monocular Videos

Supplementary Material

We provide videos on the project page[1] to vividly present qualitative results for an enhanced view experience.

A. Implementation Details

Reconstruction model. The transformer decoders in the bidirectional motion-encoding branch follow the design of DUSt3R [71], where each decoder block consists of a self-attention layer for intra-frame spatial modeling and a cross-attention layer for inter-frame temporal modeling. Finally, two DPT [56] heads are employed to predict the forward and backward motions, respectively. Here, we define the forward/backward velocities $\{\boldsymbol{v}_i^+, \boldsymbol{v}_i^-\}$ as the 3D displacements from the current frame to the next/previous frame in the camera coordinate.

Generation model. The multiple encoders for multi-modal conditions are implemented with 1) VAE [65] encoder for RGB images and depth maps, 2) convolutional layers with $8\times$ spatial and $4\times$ temporal compression ratio for masks and plüker embeddings. During the generation training stage, only convolutional layers are trainable while the VAE encoder is frozen.

B. Training Details

To ensure compatibility with the patch size of DINOv2 [54] in the reconstruction model ($\times 14$ downsampling) and the VAE in the generation model ($\times 8$ compression), we resize all input videos to have a longest edge of 560 pixels during reconstruction training, and a fixed resolution of 336×560 during generation training.

Reconstruction model. We train the reconstruction model on a combination of static and dynamic 3D datasets. For each training iteration, we sample N key frames (where $2 \leq N \leq 8$) and $N-1$ intermediate target frames between adjacent key frames. While only the N key frames are processed by the reconstruction model to predict Gaussians, the supervision loss is computed on all $2N-1$ frames. We utilize a cosine learning rate schedule with a peak learning rate of 1×10^{-4} and a warmup 5K iterations. To enhance the model's robustness to temporal direction, we apply a random temporal reversal augmentation with a probability of 0.5. The weights for the multi-task loss (Eq. 6 in the main paper) are set as follows: $\lambda_1 = 5.0$ (camera), $\lambda_2 = 1.0$ (depth), $\lambda_3 = 1.0$ (motion), and $\lambda_4 = 0.1$ (regularization).

[1] https://neoverse-4d.github.io

	Dataset	Dynamic	Depth	Pose	Flow	Real	Clip
①	PointOdyssey [93]	✓	✓	✓	✓		131
	DynamicReplica [34]	✓	✓	✓	✓		483
	Kubric [21]	✓	✓	✓	✓		5.7K
	Spring [53]	✓	✓	✓	✓		37
	VKITTI2 [8]	✓	✓	✓	✓		50
	Waymo [63]	✓	✓	✓	✓	✓	798
②	TartanAir [72]	✓	✓	✓			369
	BEDLAM [6]	✓	✓	✓			10.4K
	MVS-Synth [29]	✓	✓	✓			120
	GFIE [27]	✓	✓	✓		✓	81
③	HOI4D [47]	✓	✓			✓	3.0K
	CoP3D [61]	✓		✓		✓	2.8K
④	DL3DV [44]		✓	✓	✓	✓	6.4K
	Scannet++ [85]		✓	✓	✓	✓	853
	ARKitScenes [4]		✓	✓	✓	✓	4.5K
	HyperSim [59]		✓	✓	✓		457
	MapFree [1]		✓	✓	✓	✓	460
⑤	SpatialVID† [67]	✓	✓	✓		✓	371.3K
	Monocular Videos	✓				✓	**1M**

Table S1. **Training Datasets.** We categorize existing datasets into 5 groups based on their data characteristics. Group ①∼④ are used in reconstruction training, while group ⑤ is used in generation training. †: we only use videos for generation training.

Generation model. For the generation model, we use a constant learning rate of 1×10^{-5} and a batch size of 1 per GPU. To enable efficient on-the-fly reconstruction, we randomly sample $11 \sim 21$ keyframes from each video clip to reconstruct the 4DGS representation. Additionally, we employ a mask drop strategy where we randomly set all masks to 0 (indicating all degraded renderings need inpainting) with a probability of 0.2 to improve model robustness.

C. Dataset Details

We summarize the datasets used in our training in Table S1. Our training data is categorized into five groups:

① Dynamic datasets with 3D flow for velocity supervision.
② Dynamic datasets with depth and camera poses.
③ Dynamic datasets with incomplete 3D information (e.g., only camera poses or depth).
④ Static datasets (we assume 3D flow is zero).
⑤ Monocular videos.

We train the reconstruction model on ① to ④, while the generation model is trained on ⑤. Though SpatialVID provides 3D information, we don't use it for reconstruction training due to its unstable depth quality.

D. Evaluation Protocol

Following AnySplat, we perform test-time pose alignment to facilitate fair comparison, without introducing ground-truth poses during inference.

Figure S1. **Effectiveness of degradation simulation.** The model learns to suppress artifacts and hallucinate realistic details in occluded or distorted regions through degradation simulation.

Figure S2. **Failure cases.** Top: Text generation failure. Bottom: Novel view generation on 2D data.

Static reconstruction. We evaluate static reconstruction performance on VRNeRF [77] and Scannet++ [85].

- **VRNeRF:** We select 6 scenes captured with pinhole cameras. For each scene, we randomly sample 16 views as input for reconstruction and 8 novel views for testing.
- **Scannet++:** We evaluate on all 50 scenes in the test set. We utilize 32 input views for reconstruction and evaluate on 16 novel views.

Dynamic reconstruction. For dynamic reconstruction on ADT [55], we follow 4DGT [78] to evaluate the same 4

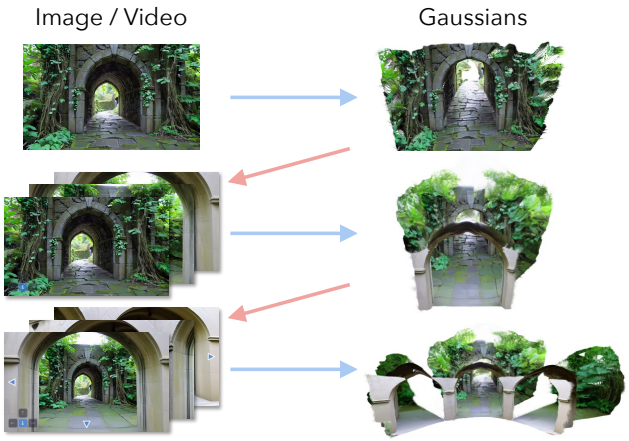

Figure S3. **Image to world.** Starting from a single view, NeoVerse can reconstruct a 3D scene, generate an exploration video, and iteratively expand the visible area.

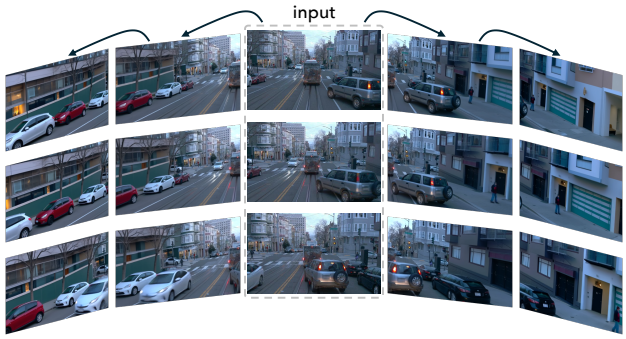

Figure S4. **Single-view to multi-view generation.** Starting from a single front-view video, NeoVerse can generate multi-view consistent videos.

scenes:

- Apartment_release_multiuser_cook_seq141_M1292
- Apartment_release_multiskeleton_party_seq114_M1292
- Apartment_release_meal_skeleton_seq135_M1292
- Apartment_release_work_skeleton_seq137_M1292

For each sequence, we sample a clip of 64 consecutive frames. We use 32 frames (stride 2) as input and the remaining 32 interleaved frames for testing.

For DyCheck [20], we evaluate 5 scenes (apple, block, paper-windmill, spin, teddy). We sample 64 consecutive timestamps for each scene, using 32 frames (stride 2) from a casually-captured video (camera 0) for reconstruction and the complete 64 frames from another fixed-camera video (camera 1) for testing.

E. Impact of Degradation Simulation

As discussed in Sec. 3.2, large camera motions often result in degraded renderings containing flying edge pixels and distortions. Fig. S1 demonstrates the necessity of our online degradation simulation. Without training on simulated degraded samples, the generation model tends to trust the

	Struct. Dist.↓	CLIP Score ↑	NIQE↓	Second Per Frame↓
AnyV2V [38]	0.071	24.89	5.04	6.11
Wan-Edit [39]	**0.013**	26.39	6.54	3.07
VACE [33]	0.015	**26.92**	**4.37**	4.30
Ours	0.018	26.66	5.13	**0.49**

Table S2. **Video editing evaluation on FiVE [39].**

	SpatialTracker [75]	St4RTrack [19]	Ours
APD ($\delta_{3D} = 0.1$m)↑	3.79	2.47	**7.31**
EPE↓	3.35	5.64	**3.10**

Table S3. **3D tracking evaluation on DriveTrack through TAPVid-3D [37].** The prediction of SpatialTracker is offered by TAPVid-3D and St4RTrack is predicted from its official codebase.

geometric artifacts in the condition, leading to "ghosting" effects or blurred outputs. **By incorporating degradation simulation, the model learns to suppress these artifacts and hallucinate realistic details in occluded or distorted regions**.

F. Downstream Task Evaluation

In the main paper, we qualitatively demonstrate several downstream applications of NeoVerse. Here, we provide quantitative evaluations on two representative tasks: video editing and 3D tracking.

Video editing. We evaluate video editing on the FiVE [39] benchmark and compare with AnyV2V [38], Wan-Edit [39], and VACE [33]. As shown in Tab. S2, although NeoVerse is not specifically designed for video editing, it achieves competitive performance while being significantly faster (0.49 seconds per frame vs. 3.07–6.11 seconds per frame for other methods).

3D tracking. We evaluate 3D tracking on the DriveTrack subset of TAPVid-3D [37] and compare with SpatialTracker [75] and St4RTrack [19]. As shown in Tab. S3, it demonstrates that the 3D flow predicted by our reconstruction model provides reliable 3D correspondences.

G. Discussion on Linear Motion Assumption

As described in Eq. (3), our method assumes approximately linear motion between adjacent key frames for Gaussian interpolation. While this is a simplified assumption, it does not negatively affect the generation quality for the following reasons. During training, we reconstruct from sparse key frames and render to all frames. The less-accurate non-keyframe renderings naturally serve as a form of temporal degradation, encouraging the generation model to learn to produce high-quality videos with non-linear motions from degraded renderings. During inference, users can input all frames to ensure reliable renderings when needed. Moreover, linear motion is a common and reasonable assumption adopted by prior works such as 4DGT [78], as real-world motion within short intervals between adjacent frames is generally well-approximated by linear interpolation.

H. Limitations and Failure Cases

Although our method can handle various challenging scenarios, there are some limitations as shown in Fig. S2. Similar to many video diffusion models, our method occasionally ***struggles to render legible and correct text*** (Top two rows). Besides, our method relies on extracting 3D clues from videos. It ***struggles with data lacking 3D geometry, such as 2D cartoons***. For instance, as the camera moves to the right side of a 2D cartoon character (Bottom two rows), the model may fail to generate the correct 3D profile (e.g., revealing the other side of a face), as the input video lacks inherent 3D structure.

I. Additional Qualitative Results

Image to world. Our NeoVerse allows for exploration in a captured image by iteratively generating new views and reconstructing the scene. As illustrated in Fig. S3, given a single starting image, we can generate a spatially coherent video trajectory. This generated video is then used to reconstruct a larger Gaussian Splatting scene, effectively "outpainting" the 3D world.

Single-view to multi-view Fig. S4 demonstrates the capability of generating multi-view consistent videos from a single-view video through **iterative application of NeoVerse**.