4Real-Video: Learning Generalizable Photo-Realistic 4D Video Diffusion

Supplementary Material

1. Additional Implementation Details

1.1. Base video model training

The base video model is pretrained on a dataset comprising 1.2 million images and 238k hours of video, each paired with a corresponding text caption. This dataset includes a subset of 3D videos with a distribution similar to popular open-source datasets such as RealEstate and MvImageNet. During training, pseudo-generated videos are created on the fly from all videos in the dataset, excluding the 3D videos. We use the AdamW optimizer with a cosine learning rate scheduler. The batch size for training the base model is 1,152, while for the 4D model, it is 48. Due to limited compute resources, our current model is small. For masked video model training, we keep the first K frames of the video, where K is uniformly sampled from $[0, 4]$.

1.2. Implementation details of ablation study

We compared different baseline variants to analyze our approach. Below, we provide details for each method corresponding to the columns in Fig 7.

Sequential w/o training. We sequentially interleave cross-view and cross-time attention, as described in Equation 5. All parameters in the attention layers are directly inherited from the base video model without additional training. We observe that this variant produces noisy outputs lacking meaningful structure.

Parallel w/o training, hard sync. We perform inference using the proposed architecture without training the synchronization layers. For hard synchronization, we average the token updates, *i.e*.

$$\mathbf{x}_{l+1} = \frac{1}{2}(\mathbf{y}_l^{\mathrm{v}} + \mathbf{y}_l^{\mathrm{t}}). \tag{1}$$

This version generates some content with elements from the input video, but it remains highly noisy.

Parallel w/o training, soft sync. The soft synchronization is implemented as weighted averaging,

$$\begin{aligned} \mathbf{x}_{l+1}^{\mathrm{v}} &= (1 - w_l)\mathbf{y}_l^{\mathrm{v}} + w_l\mathbf{y}_l^{\mathrm{t}} \\ \mathbf{x}_{l+1}^{\mathrm{t}} &= (1 - w_l)\mathbf{y}_l^{\mathrm{t}} + w_l\mathbf{y}_l^{\mathrm{v}} \end{aligned} \tag{2}$$

Here, w_l represents the weight, which gradually increases with the layer depth, specifically defined as $w_l = 0.1 + \frac{l}{L} \cdot 0.4$. This approach produces results with more discernible content compared to hard synchronization.

Sequential trained. The sequential architecture is trained following the same procedure as our proposed approach. We experimented with two variants: finetuning only the cross-time attention and finetuning only the temporal attention. Our findings indicate that finetuning temporal attention results in more stable outcomes. Therefore, for brevity, we report results only for the version where cross-time attention is finetuned.

Parallel hard sync. The variant of our proposed method employing hard synchronization.

Parallel soft sync w/o Objaverse. A variant of our proposed method with soft synchronization, without finetuning on animated 4D Objaverse data.

2. Additional Analysis

Ablation of importance of different synchronization layers. In the paper, we included Figure 3, which analyzes the strength of synchronization at different layers during inference. We observed that the update strength increases in deeper layers of the network. Additionally, we conducted an ablation study (see Table 1) where we trained the 4D model after removing either the first eight (33%) or the last 33% of synchronization layers, using the same training setup as the original model. According to the Dust3R-confidence metric, the model adapts to removing these layers, with both scenarios yielding similar results. However, the overall quality is worse compared to the full model.

How does mixed data training affect the results? The 4D video model can be trained alternatively with a mix of pseudo-4D and Objaverse data, instead of sequentially pretraining on pseudo-4D and finetuning on Objaverse. However, empirically, we were unable to find a suitable combination that led to improvements. As shown in Table 1, finetuning the pretrained model with a 1:2 mix of pseudo-4D to Objaverse led to a continuous decline in Dust3R-confidence and slightly worse overall metrics compared to fine-tuning with Objaverse alone. By inspecting the generated videos, we believe this is because the pseudo data distracted the model, causing it to focus on 2D rather than true perspective transformations, reducing quality. To scale up, we suggest improving the base video model and incorporating larger synthetic or real 4D datasets.

Generate long duration videos. Our model is *autoregressive* and can generate videos longer than 2s. In the supplementary material, we provided samples with 120 frames (5s), reaching the typical duration of videos generated by mainstream models. However, since our model processes only 8×8 frames at a time, it has limitations in maintaining long-term consistency. As shown below, the model "forgets" the side appearance of the pillow toy from the first frame, leading to errors in later frames. We believe ad-

	$\tau = 2.0 \uparrow$	$\tau = 2.5 \uparrow$	$\tau = 3.0 \uparrow$
Pseudo 4D pretrain; Objaverse ft for 3k iters	**41.0**	**33.4**	**25.7**
Drop first 33% sync. layers	40.2	32.6	24.7
Drop last 33% sync. layers	40.3	32.8	25.1
Pseudo 4D pretrain; Objaverse ft for 10k iters	40.9	33.2	25.5
Pseudo 4D + Objaverse ft for 1k iters	40.7	33.1	25.3
Pseudo 4D + Objaverse ft for 5k iters	40.5	33.0	25.2
Pseudo 4D + Objaverse ft for 10k iters	40.4	32.8	25.1

Table 1. Additional Ablation Studies: Due to space constraints, we report only the Dust3R-confidence metric. Even though the numeric difference in the ablations is small, we observed significant quality degradation in a few tested samples.

vancements in long-context conditioning for video generation could address this issue, though we did not focus on it, as it is somewhat orthogonal to this work.

Figure 1. Failure case of maintaining long-term consistency.

3. Deformable 3D GS Reconstruction Details

Using generated 4D videos with multi-view frame grids, we apply a reconstruction method to produce an explicit 3D representation, *i.e.*, deformable 3D geometric structures (GS).

Canonical 3D representation. We use 3D Gaussian Splats [1] to represent the canonical shape of the dynamic scene. This representation consists of a set of 3D Gaussian points defined by their 3D position, orientation, scale, opacity, and RGB color. The 3D Gaussian Splats are rendered by projecting the Gaussian points onto the image plane and aggregating pixel values using a NeRF-like volumetric rendering equation. In our implementation, we find that constraining the Gaussians to be isotropic effectively reduces artifacts when viewing the 3D representation from viewpoints distinct from the training perspectives.

Deformation field. To model a 4D scene, we use a deformation field to represent the offsets of the 3DGS. This deformation field is implemented as an MLP, which takes the 3D position of a point and time as input and outputs a 3D displacement offset.

Initialize canonical 3D GS with 3D dense tracking. While the input freeze-time video may appear visually plausible, it is not truly geometrically accurate, particularly in the background regions. Directly optimizing 3D GS using these frames as ground truth results in significant artifacts. The most noticeable issue is the noisy reconstruction of background regions, which fail to separate cleanly from the foreground. In our preliminary exploration, we tested state-of-the-art feedforward reconstruction methods, including Dust3R [4] and Splatt3R [3]. However, in most cases, only the foreground regions could be reliably reconstructed, while the background remained noisy and entangled with the foreground. We attribute this limitation to the quality of the video model used to generate the inputs. In the long term, this issue could potentially be addressed by employing a higher-quality video model. At this stage, we instead use a recent 3D dense tracking method [2], which performs pixel-wise tracking to aggregate 3D points from various keyframes of the freeze-time video. These points are aligned towards a central frame, whose coordinates are treated as the canonical frame.

The advantage of switching to 3D tracking is that it does not require the scene to be static, allowing it to handle multi-view inconsistencies in the generated videos by treating them as non-rigid deformations. Furthermore, 3D tracking leverages monocular depth estimation as input, preserving the clean foreground/background separation provided by the estimated depth map. This results in a visually more coherent and appealing outcome.

Removing boundary floaters. 3D tracking often produces outlier points along depth boundaries, a common artifact in monocular depth estimation. To eliminate these 'floaters,' we apply a rendering loss to optimize the opacity of each point, effectively pruning points that cause visual artifacts.

Specifically, given a set of aggregated 3D points from dense tracking, we know each point's 3D position in the frame coordinates of each frame of the input freeze-time video. This allows us to use the differentiable 3DGS renderer to re-render each input frame and compute the loss. Furthermore, since the points are modeled as isotropic Gaussians without orientation and are already in the frame coordinate system, we avoid the need to estimate camera extrinsics at this stage. This approach enhances robustness against multi-view inconsistencies in the input video.

Temporal deformation with view-dependent compensation. The next step involves fitting a temporal deformation field to animate the canonical 3DGS to follow the motion in the input 4D video. However, due to imperfections in the multi-view consistency of the 4D video—an issue inherited from the input freeze-time video—directly optimizing the temporal deformation field would lead to noisy reconstructions, mirroring the challenges previously discussed.

To address this issue, we augment the temporal deformation with additional view-dependent deformation to compensate for inconsistencies in the generated frames across different views. Specifically, to re-render a point on the input frame $\mathcal{I}_{ij}$ of the input frame grid, where i and j represent the indices of view and time respectively, the deformation offset $\Delta\mathbf{p}_{ij}$ for each point $\mathbf{p}$ in canonical space is now

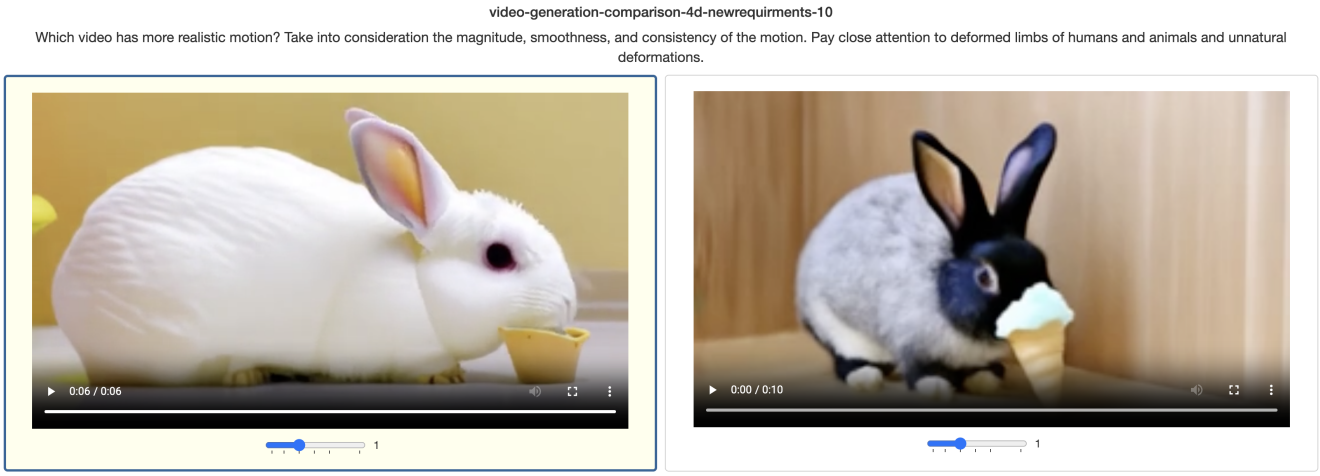

Figure 2. A screenshot of the interface for user study.

computed as:

$$\Delta \mathbf{p}_{ij} = \Delta \mathbf{p}_i^{\mathrm{v}} + \Delta \mathbf{p}_j^{\mathrm{t}}, \tag{3}$$

where $\Delta \mathbf{p}_j^{\mathrm{t}}$ represents the temporal deformation computed by an MLP, and $\Delta \mathbf{p}_i^{\mathrm{v}}$ is the view-dependent deformation estimated via dense 3D tracking during the canonical 3DGS reconstruction stage. It is worth noting that view-dependent deformation has also been employed in 4Real [5]; however, in our approach, the view-dependent deformation is predicted from dense 3D tracking rather than optimized using rendering loss, making it more robust.

4. User study details

The user study shown in Fig. 2 is conducted with 10 evaluators per video pair. During each session, evaluators were presented with two anonymized videos with an interface as shown in Fig. 2. The evaluators were given the following instructions:

> You are shown a description of a video and two different 3D videos generated by AI based on this description. Your task is to answer 7 questions regarding the quality of these videos. Please pay close attention to instructions and answer as thoughtfully as you can. The video shows several consecutive views of the same dynamic object.
>
> 1. Which video has more realistic motion? Take into consideration the magnitude, smoothness, and consistency of the motion. Pay close attention to the deformed limbs of humans and animals and unnatural deformations.
> 2. Which video has the highest quality foreground?
> 3. Which video has the highest quality background?
> 4. Which video has an object of a better, more realist shape? That is the video in which the main object has the most natural shape, again paying attention to deformed limbs of humans and animals and unnatural deformations.
> 5. In general which video looks higher quality?
> 6. Which video is most dynamic? The video that contains the most motion. Please keep in mind that these is several views of the same dynamic video, played one after the other, so ignore all camera movement and focus solely on object movement. Please exclude from consideration any random limb deformations.
> 7. Which video is better following the text description? That is which video reflects all the aspects included in the text description
>
> Finally, if there is no significant difference in your opinion send the video to junk.

References

[1] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2

[2] Tuan Duc Ngo, Peiye Zhuang, Chuang Gan, Evangelos Kalogerakis, Sergey Tulyakov, Hsin-Ying Lee, and Chaoyang Wang. Delta: Dense efficient long-range 3d tracking for any video, 2024. 2

[3] Brandon Smart, Chuanxia Zheng, Iro Laina, and Victor Adrian Prisacariu. Splatt3r: Zero-shot gaussian splatting from uncalibrated image pairs. 2024. 2

[4] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, 2024. 2

[5] Heng Yu, Chaoyang Wang, Peiye Zhuang, Willi Menapace, Aliaksandr Siarohin, Junli Cao, Laszlo A Jeni, Sergey Tulyakov, and Hsin-Ying Lee. 4real: Towards photorealistic 4d scene generation via video diffusion models. In *NeurIPS*, 2024. 3