Supplymentary Materials for

SpaceTimePilot: Generative Rendering of Dynamic Scenes Across Space and Time

Zhening Huang[1,2] Hyeonho Jeong[2] Xuelin Chen[2] Yulia Gryaditskaya[2]
Tuanfeng Y. Wang[2] Joan Lasenby[1] Chun-Hao Huang[2]

[1]University of Cambridge [2]Adobe Research

https://zheninghuang.github.io/Space-Time-Pilot/

Overview of Supplementary Materials

The supplementary material consists of this document and the accompanying website. **Please refer to the webpage for videos.** In this document, we provide the following sections:

A. **Network Architecture.**
A more comprehensive description of the architecture, implementation details, and design choices not covered in the main paper.

B. **Longer Video Segments with Disentangled Controls.**
An extended discussion and additional results for autoregressive segment-based video generation with disentangled controls, complementing the "Support for Longer Video Segments" section of the main paper.

C. **Additional Details on the Proposed Cam×Time Dataset.**
An extended discussion of the construction of the proposed Cam×Time dataset, with visualizations of example sequences.

D. **Additional Ablation Studies.**
Ablations on various model settings with analysis of their impact on performance.

E. **Additional Qualitative Visualizations.**
Additional qualitative results and their discussion.

A. Network Architecture

The network architecture of SpaceTimePilot is depicted in Fig. 1. The newly introduced animation-time embedder $\mathcal{E}_{\text{ani}}$ encodes the source and target animation times, $\mathbf{t}_{\text{src}}$ and $\mathbf{t}_{\text{trg}}$, into tensors matching the shapes of x_{src} and x_{trg}, which are then added to them respectively. During training, we train only the camera embedder $\mathcal{E}_{\text{cam}}$, the animation-time embedder $\mathcal{E}_{\text{ani}}$, the self-attention (full-3D attention), and the projector layers before the cross-attention.

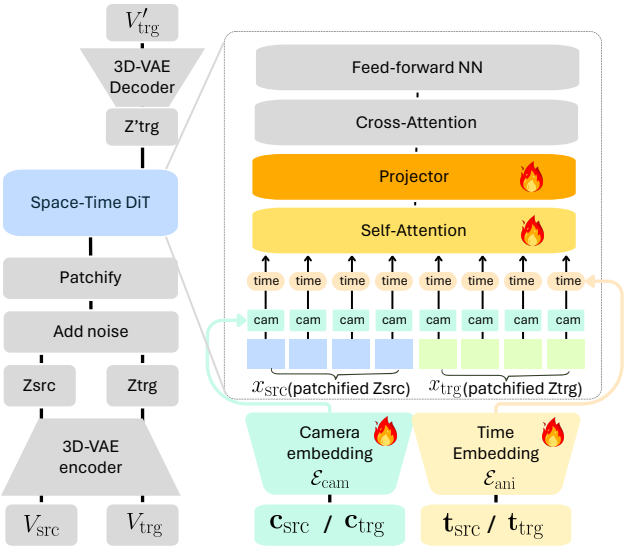


Figure 1. **Architecture of SpaceTimePilot.** Our model jointly conditions on camera trajectories and temporal control signals via space–time attention, enabling non-monotonic motion generation such as reversals, repeats, accelerations, and zigzag time.

B. Longer Space-Time Exploration Video with Disentangled Controls

One of the central advantages of SpaceTimePilot is its ability to freely navigate both spatial and temporal dimensions, with arbitrary starting points in each dimension and fully customizable trajectories through them. Although each individual generation is limited to an 81-frame window, we show that SpaceTimePilot can effectively extend this window indefinitely through a multi-turn autoregressive inference scheme, enabling continuous and controllable space–time exploration from a single input video. The

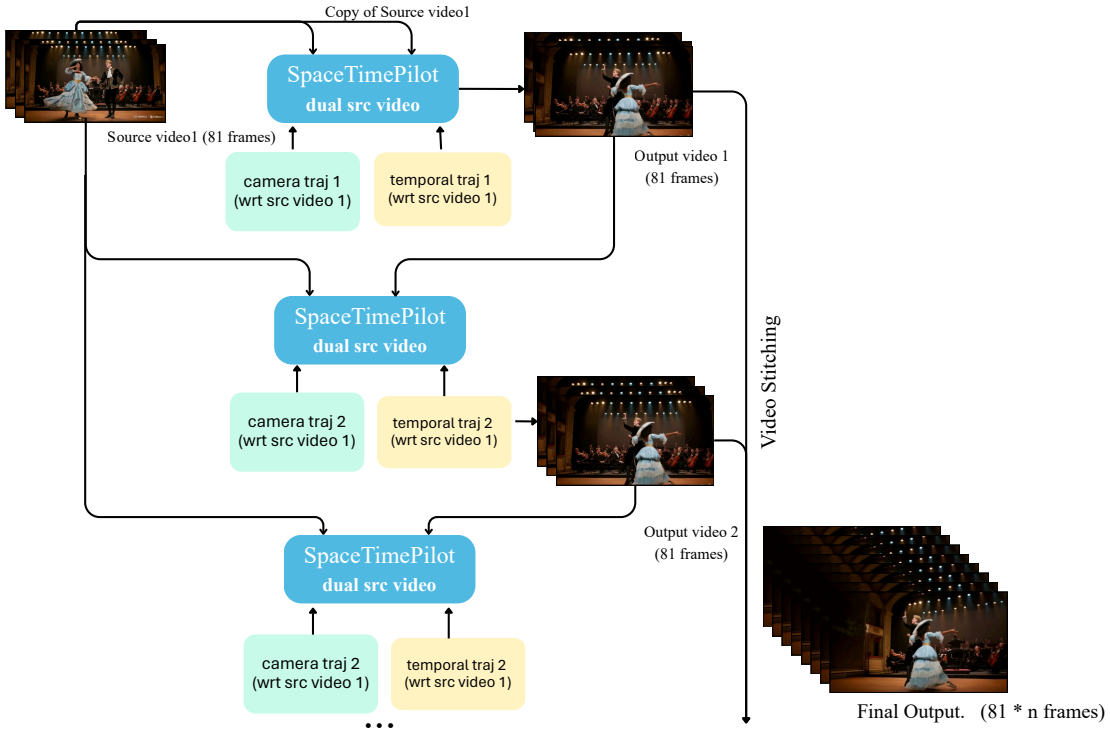


Figure 2. **Overview of the multi-turn autoregressive inference scheme.** The model first generates an 81-frame segment conditioned on the source video and a chosen space–time trajectory. The resulting output is then reused as a secondary source video for subsequent iterations, each with its own camera and temporal trajectory. By chaining these iterations and stitching the outputs, SpaceTimePilot produces a long, coherent video that follows an arbitrary space–time path.

overall pipeline is illustrated in Fig. 2.

The core idea is to generate the final video in autoregressive segments that connect seamlessly. For example, given a source video of 81 frames, we may first generate a 0.5× slow-motion sequence covering frames 0–40 with a new camera trajectory. Then, continuing both the visual context and the generated camera trajectory, we can produce the next segment starting from the final camera pose of the previous output, while temporally traversing the remaining frames 40–81. This yields an autoregressive chain of viewpoint-controlled video segments that together create a continuous long-range space–time trajectory.

A key property that enables this behavior is that our model can generate video segments whose camera poses do *not* need to start at the first frame. This allows precise control over the starting point, both in time and viewpoint, for every generated chunk, ensuring smooth, consistent motion over extended sequences.

To maintain contextual coherence across iterations, we introduce a lightweight memory mechanism. During training, the model is conditioned on **a pair of source videos**, which enables consistent chaining during inference. Specifically:

- At iteration $i = 1$, the model is conditioned only on the original source video.
- At iteration $i = 2$, it is conditioned on both the source video and the previously generated 81-frame segment.
- This process repeats, with each iteration conditioning on the source video as well as the most recent generated segment.

This simple yet effective strategy allows SpaceTimePilot to generate arbitrarily long, smoothly connected sequences with continuous and precise control over both temporal manipulation and camera motion.

Here, we showcase how this can be used to conduct large viewpoint changes, as demonstrated in Fig. 3.

C. Additional Details on the Proposed Cam×Time Dataset.

The `Cam×Time` dataset is built using high-quality, commercially licensed 3D environments that include both realistic indoor and outdoor scenes. For each environment, we populate the space with multiple animated human characters. The character assets are sourced from Mixamo [3] and HUMOTO [5], and each character is manually textured and refined to ensure realistic geometry, appearance, and material quality. The animations span a diverse range of human

Figure 3. **Multi-turn autoregressive generation with SpaceTimePilot.** Top row: source video frames. Rows 2–4: Turn-1, Turn-2, and Turn-3 generations. At each turn, SpaceTimePilot jointly conditions on (1) the original source video and (2) the previously generated chunk, ensuring temporal continuity, stable motion progression, and consistent camera geometry. This dual-conditioning design enables viewpoint changes far beyond the input video—such as rotating to the rear of the tiger or transitioning from a low-angle shot to a high bird's-eye view—while preserving visual and motion coherence. Please refer to section "AR Demos" in the website for videos.

motions, including locomotion, gestures, and human-object interactions. Examples of scenes are shown in Fig. 4. Please refer to the complementary website for the video examples.

To capture rich spatial coverage, we generate four distinct camera trajectories for every scene. Camera paths include rotational orbits, linear tracking motions, and smoothly curved arcs. A dedicated validity module ensures that each trajectory: (1) begins at a collision-free location with clear visibility of the main character, (2) maintains non-intersecting movement with the environment throughout the path, and (3) preserves full subject visibility across all viewpoints.

Each trajectory is rendered into a 120-frame sequence at a resolution of 1080×1080 pixels, providing dense temporal sampling with high visual fidelity. This yields three multi-view video sequences per scene, each covering the full motion duration with consistent lighting, textures, and geometry. Overall, we rendered 3000 videos from 750 animations, each with 120 videos full grid rendering, leading to 360,000 videos.

For temporal-control training, we could sample any time variants from these sequences, including slow motion, reverse playback, bullet-time around arbitrary frames, and non-monotonic time patterns such as forward–backward oscillation. These augmented temporal signals are illustrated in Fig. 5.

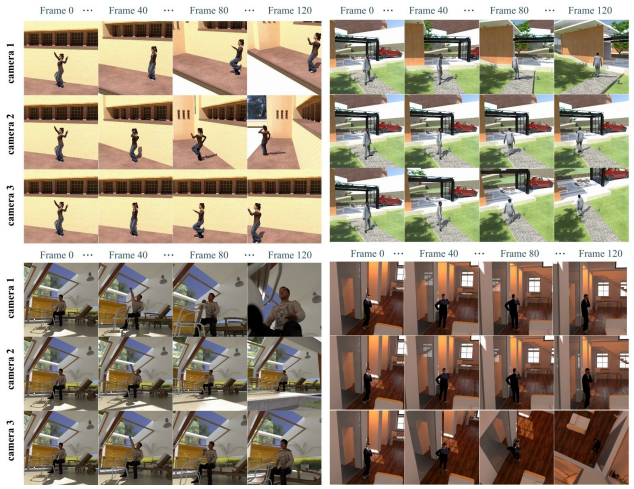


Figure 4. **Example of Cam×Time**. Multi-view, densely sampled sequences from the Cam×Time dataset. Each row shows frames from one camera trajectory, and each column samples different timesteps (0–120). The dataset provides diverse environments, human motions, and four camera paths per scene with full 120-frame temporal coverage.

D. Additional Ablation Studies

D.1. Temporal Warping Augmentation

Using [1, 2] as our default datasets, we compare training jointly with static-scene datasets [4, 6] with applying only temporal warping (TW) augmentation on the default datasets (Sec. 3.2.2 in the main paper). Although static-scene datasets naturally support bullet-time effects, they do not provide enough diversity of temporal control configurations for models to reliably learn time locking on their own, as shown in Fig. 7 (top). Please refer to section "Effective Temporal Warping" in the website for more videos.

In Fig. 7 (bottom), we further show that freezing temporal warping (3rd row) produces better results than training without freezing it. Please refer to section "Freeze Warping Ablations" in the website for more videos.

D.2. Significance of `Cam×Time` Dataset

Besides the quantitative results in the main paper (Table 5), in Fig. 8 (top), we provide visual comparisons demonstrating the effectiveness of the proposed `Cam×Time` dataset. Clear artifacts appear in baselines trained without additional data or with only static-scene augmentation (highlighted in red boxes), whereas incorporating `Cam×Time` removes

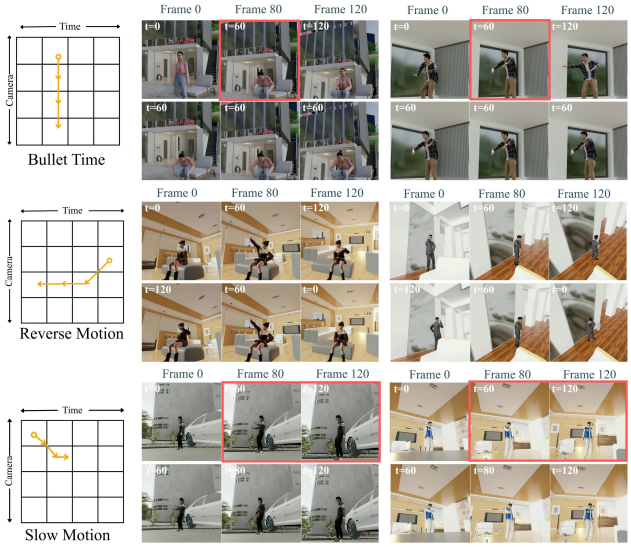


Figure 5. **Sampling from Cam×Time.** By sampling from the Cam×Time dataset, we can extract frames corresponding to arbitrary combinations of camera viewpoints and temporal positions, forming source-target pairs with rich camera and temporal control signals.

these artifacts, demonstrating its significance. Please refer to section "Dataset Ablations" in the website for more videos.

D.3. Time Embedding Ablation

As promised in Sec. 3.2.1 in the main paper, we compare several time-embedding strategies. RoPE(f') can freeze the scene dynamics at $\mathbf{t}$=40, but it also undesirably locks the camera motion. Using MLP, by contrast, fails to lock the temporal state at all (red boxes). Conditioning on the latent frame f' (with uniform sampling) introduces noticeable artifacts. In comparison, the proposed 1D-Conv embedding enables SpaceTimePilot to preserve the intended scene dynamics while still generating accurate camera motion. Adding `Cam×Time` to training further enhances the results. Please refer to section "Time-Embedding Method Ablation" in the website for more examples.

E. Additional Qualitative Visualizations

We show more qualitative results of SpaceTimePilot in Fig. 6. Our model provides fully disentangled control over camera motion and temporal dynamics. Each row presents a different pairing of temporal control inputs (top-left icon) and camera trajectories. SpaceTimePilot reliably generates coherent videos under diverse conditions, including normal and reverse playback, bullet-time, slow motion, replay motion, and complex camera movements such as panning, tilting, zooming, and vertical translation. Please refer to section "Video Demonstrations" in the website for more examples.

References

[1] Jianhong Bai, Menghan Xia, Xiao Fu, Xintao Wang, Lianrui Mu, Jinwen Cao, Zuozhu Liu, Haoji Hu, Xiang Bai, Pengfei Wan, and Di Zhang. ReCamMaster: Camera-controlled generative rendering from a single video. In *ICCV*, 2025. 3, 6, 7

[2] Jianhong Bai, Menghan Xia, Xintao Wang, Ziyang Yuan, Xiao Fu, Zuozhu Liu, Haoji Hu, Pengfei Wan, and Di Zhang. SynCamMaster: Synchronizing multi-camera video generation from diverse viewpoints. In *ICLR*, 2025. 3, 6, 7

[3] Adobe Systems Inc. Mixamo, 2018. Accessed: 2025-03-07. 2

[4] Zhengqi Li, Tali Dekel, Forrester Cole, Richard Tucker, Noah Snavely, Ce Liu, and William T Freeman. Learning the depths of moving people by watching frozen people. In *CVPR*, 2019. 3, 6

[5] Jiaxin Lu, Chun-Hao Paul Huang, Uttaran Bhattacharya, Qixing Huang, and Yi Zhou. Humoto: A 4d dataset of mocap human object interactions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10886–10897, 2025. 2

[6] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018. 3, 6

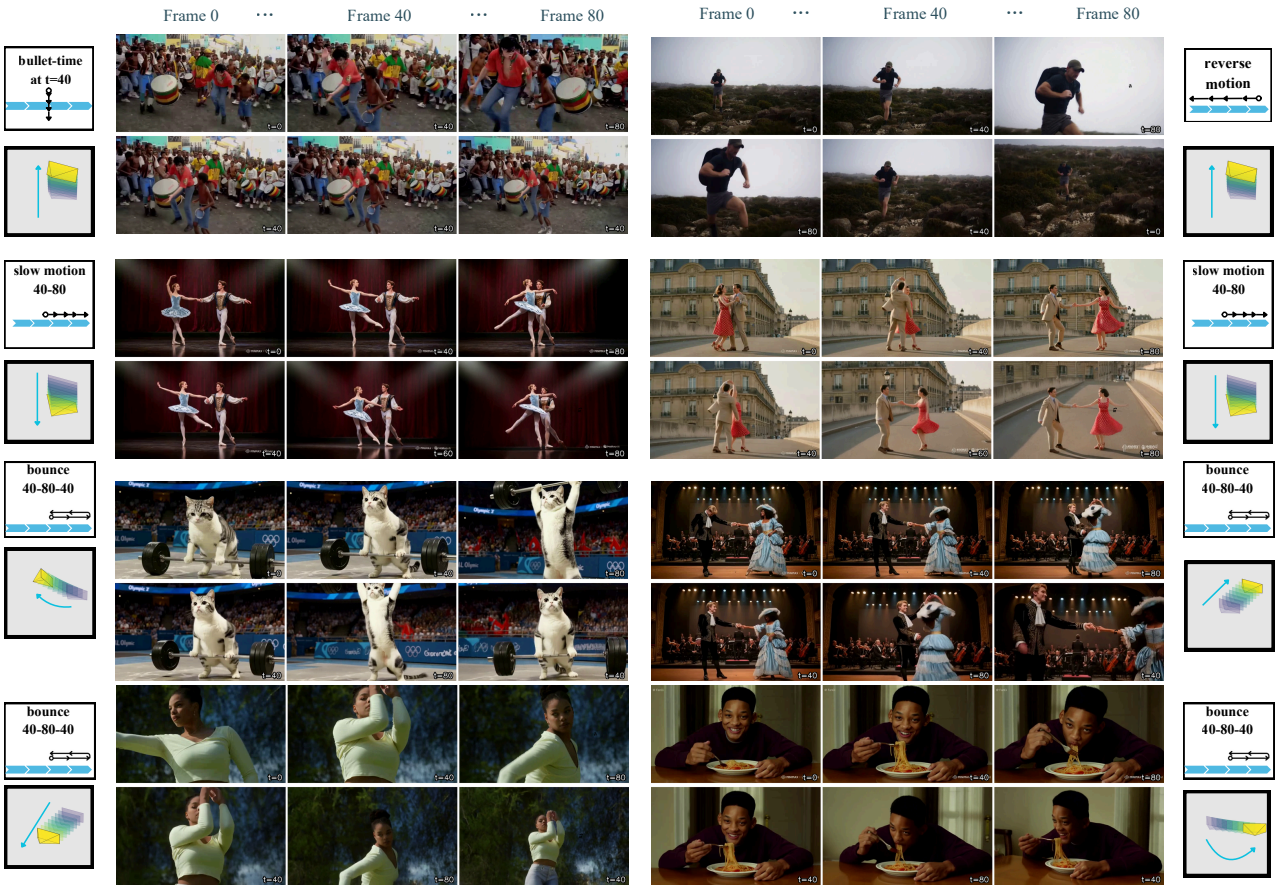
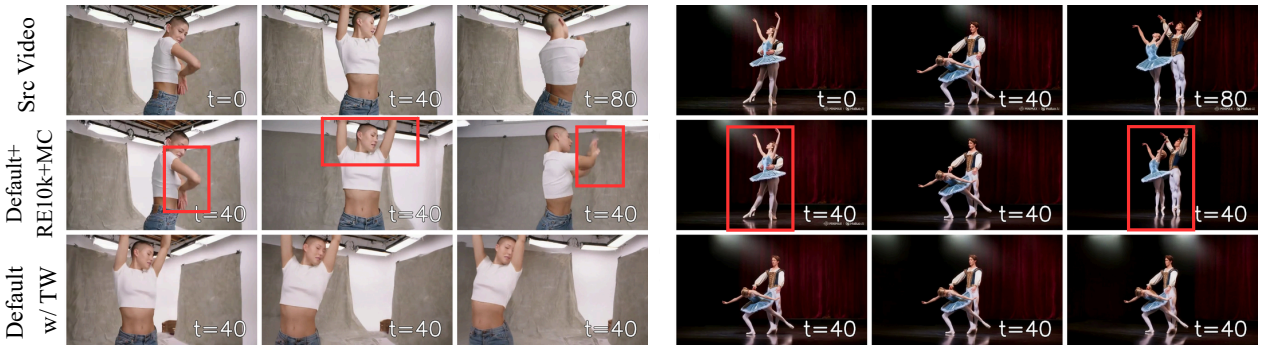
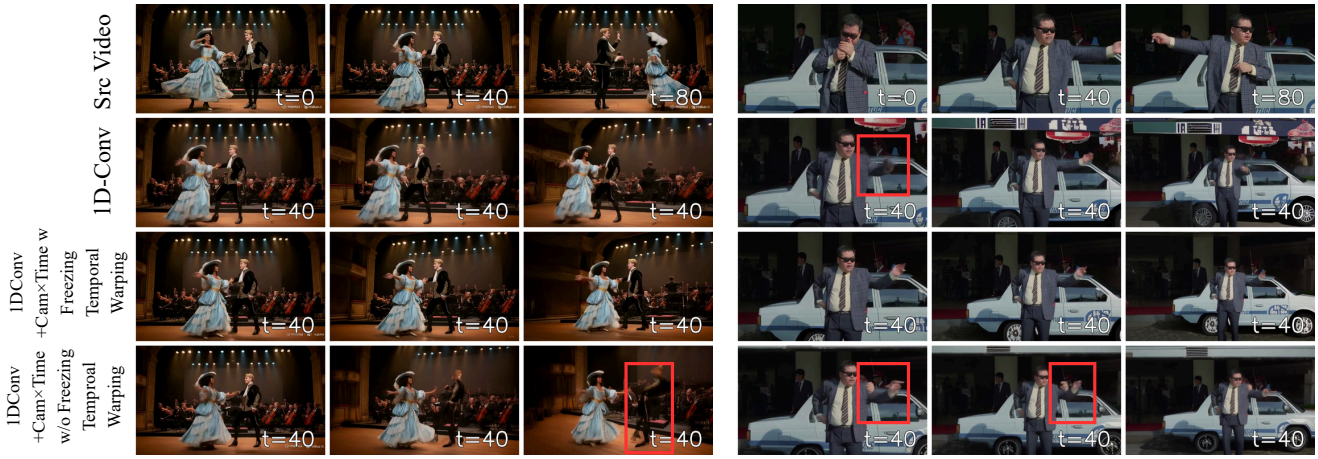


Figure 6. **More Qualitative results.** Our model provides fully disentangled control over camera motion and temporal dynamics. Each row illustrates a different combination of temporal control inputs (top-left icon) and camera trajectories. SpaceTimePilot consistently produces coherent videos across a wide range of controls, including normal and reverse playback, bullet-time, slow motion, replay motion, and complex camera paths such as panning, tilting, zooming, and vertical motion.



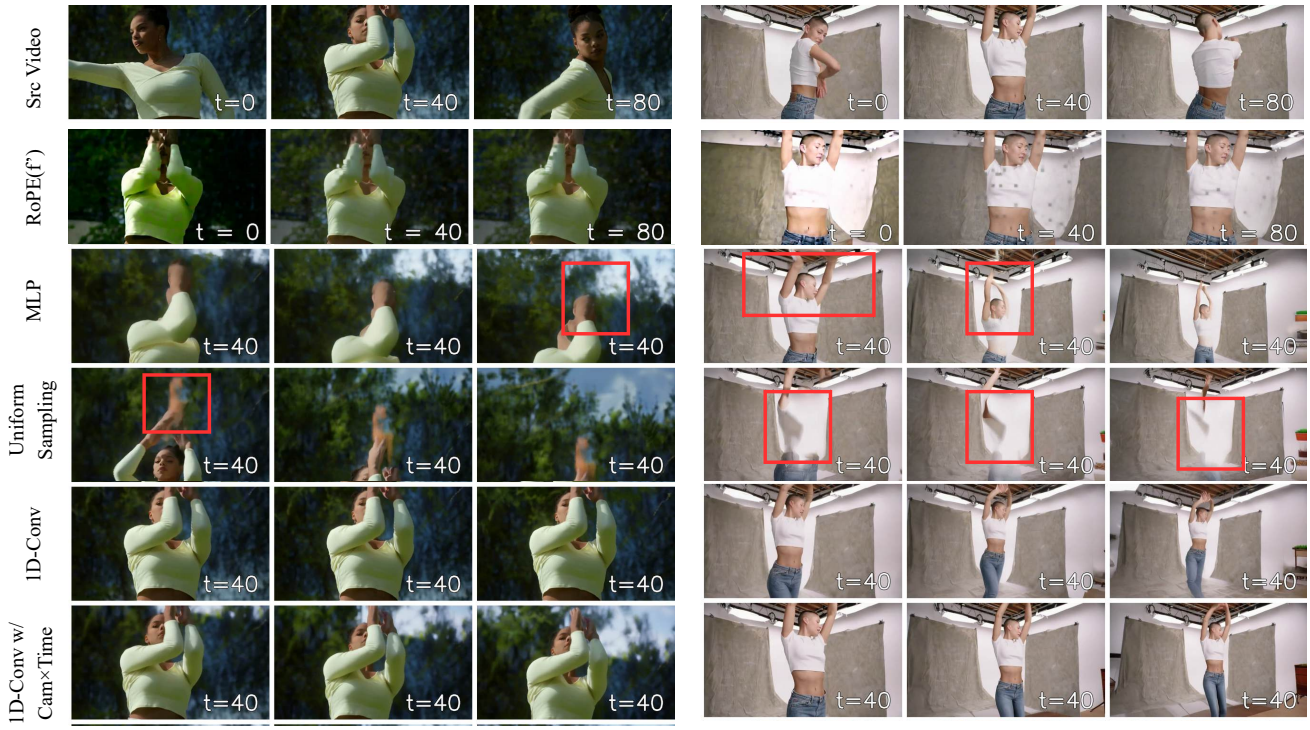
Ablation 1: Temporal warping (TW) vs. Joint dataset training on bullet-time effect (**t**=40)



Ablation 2: Varied temporal warping configuration on bullet-time effect (**t**=40)

Figure 7. **Ablation study.** (Top) Using [1, 2] as default datasets, we compare the influence of adding static-scene datasets [4, 6] vs. just doing temporal warping (TW) augmentation (Sec. 3.2.2 in the main paper). Temporal warping definitely provide more variety of time control signals, allowing models to learn better camera-time disentanglement. (Bottom) We further compare different configurations of warping, where we show freezing temporal warping (3rd row) leads to better results than those trained without freezing temporal warping.

Ablation 3: Dataset ablation: Cam×Time vs. static-scene datasets on bullet-time effect (**t**=40)



Ablation 4: Different time embedding schemes on bullet-time effect (**t**=40)

Figure 8. **Ablation study.** (Top) We verify the efficacy of the proposed Cam×Time dataset. Considering [1, 2] as default datasets, we compare the impact of different datasets on the generated videos. One can clearly see artifacts in baselines without any extra data or augmented with static-scene data, whereas training additionally with Cam×Time leads to no artifacts, confirming the usefulness of our dataset. (Bottom) We compare several time-embedding strategies. The MLP fails to lock the temporal state (red boxes), while RoPE(f') correctly freezes the scene dynamics at **t**=40 but unintentionally locks the camera motion too. Conditioning on the latent frame f' (with uniform sampling) introduces noticeable artifacts. In contrast, the proposed 1D-Conv embedding allows SpaceTimePilot to both freeze the scene dynamics at **t**=40 and produce intended camera motion. Incorporating Cam×Time during training further improves performance.