

BulletTime: Decoupled Control of Time and Camera Pose for Video Generation

Supplementary Material

A. Additional Implementation Details

We fine-tune the pretrained CogVideoX-5B-T2V [81] model using the AdamW optimizer with a learning rate of 2×10^{-5} and a weight decay of 10^{-4} . We apply gradient clipping with a maximum norm of 1.0. The learning rate follows a linear decay schedule with 100 warm-up steps. The model is trained for a total of 40K iterations.

Since existing latent-space video diffusion models downsample the temporal resolution via the 3D-VAE and the transformer’s patchify block, directly injecting per-frame world-time signals becomes incompatible with the reduced temporal resolution. To address this, we encode the world-time sequence with learnable 1D convolution layers to map it to the downsampled temporal resolution. For our non-learnable Time-RoPE design, we downsample the world-time sequence via average pooling so that it matches the latent temporal resolution. When computing the RoPE positional encoding, we scale the world-time differences by the FPS to convert them into consistent temporal offsets. With this scaling, Time-RoPE becomes exactly equivalent to the standard RoPE formulation under uniform time sampling, making standard RoPE a special case of our design to better preserve the model’s original generative prior. For camera-conditioning signals, we follow prior work [8, 25] and directly subsample the camera trajectories at fixed intervals to match the latent temporal resolution.

For video distribution-level evaluation, we report FVD and KVD following the protocol of TATS [19]. We use videos from UCF101 [63] as the real distribution and generate camera–time–conditioned outputs for each baseline as the fake distribution. Following TATS, each video is embedded using a pretrained I3D network to obtain a single feature vector, and we compute FVD (Fréchet distance) and KVD (polynomial-kernel MMD) between the real and generated feature distributions. Using UCF101 as the reference distribution introduces several limitations: many of our time-control operations (e.g., slow motion, reversal, temporal pausing) fall outside the temporal statistics of UCF101, potentially inflating divergences unrelated to perceptual quality; and UCF101 primarily contains human-action videos with limited camera motion, whereas our generated videos may include large, user-specified camera trajectories.

B. Additional Results

We provide additional qualitative and quantitative results to complement the main paper and further characterize the capabilities and limitations of our approach.

Result on large camera motions. Our model can handle substantial viewpoint changes, including camera rotations up to 180° on real-world sequences, as illustrated in Fig. B.1. These results suggest that the model can extrapolate beyond the observed trajectory to a certain extent.



Figure B.1. **Results under large camera motions.** Our model can synthesize plausible results under substantial viewpoint changes, including 180° rotations.

Comparison with time-editing baselines. Beyond the RIFE [33] baseline in the main paper we further compare against MoMo [37], a recent diffusion-based frame interpolation method, as a stronger time-editing baseline. The results are summarized in Tab. B.1. These comparisons highlight the benefit of modeling time control within a unified generative framework, rather than relying on a separate interpolation stage. In addition to the reported metrics, our formulation is naturally compatible with autoregressive long-video generation, which is less straightforward for two-stage pipelines.

Table B.1. **Comparison with time-editing baselines.** Our unified 4D framework outperforms two-stage pipelines (e.g., RIFE [33] and MoMo [37] + ReCamMaster [8]) across all metrics, demonstrating the advantages of joint spatio-temporal modeling over disjoint interpolation and synthesis.

Method	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
RIFE + ReCamMaster	21.96	0.5852	0.1846
MoMo + ReCamMaster	22.54	0.6210	0.1656
Ours	24.57	0.6905	0.1265

Comparison with a monocular video-NVS baseline. Novel view synthesis from a single monocular video is inherently ill-posed, especially when large disoccluded regions must be hallucinated. To provide a reference point, we compare our method with *Shape-of-Motion (SOM)* [68] in Fig. B.2. While geometry-based approaches such as *SOM* can be effective when geometry is reliably recovered, they may struggle in heavily occluded or previously unseen re-

gions. In contrast, our method leverages generative priors to produce visually coherent outputs under 4D control, particularly in challenging disoccluded areas.



Figure B.2. **Comparison with Shape-of-Motion (SOM).** Our model produces more coherent content in disoccluded regions by leveraging learned generative priors.

Generation quality after fine-tuning. To examine whether introducing the 4D control module compromises the original generation capability of the base model [81], we evaluate the fine-tuned model on 100 VBench prompts. As shown in Tab. B.2, the overall generation quality remains largely preserved after fine-tuning. We believe this is enabled by dense 4D supervision from synthetic data, which teaches the model to respond to camera and time controls while retaining the underlying content prior of the pre-trained video generator.

Table B.2. **Impact of fine-tuning on generation quality.** Evaluation on the VBench benchmark shows that our fine-tuning process, which introduces 4D control, maintains the aesthetic and imaging quality of the original CogVideoX base model.

Method	Aesthetic Quality↑	Imaging Quality↑
Base Model (CogVideoX)	0.601	0.606
Ours	0.597	0.604

Dense camera–time grid visualization. To further illustrate disentangled control over camera motion and world time, Fig. B.3 presents a dense grid of samples obtained by varying the two axes independently. This visualization shows that the model can traverse continuous time and camera trajectories within the same scene in a structured and consistent manner.

Generalization to real-world videos. Although trained with synthetic 4D supervision, the model generalizes to real-world videos at inference time without requiring explicit camera or time annotations. In particular, it can tolerate input videos with jittery camera motion and generate outputs along smoother, user-specified trajectories. We invite readers to visit our project page at <https://19reborn.github.io/Bullet4D/> for comprehensive visual results.

Failure cases. We illustrate representative failure modes of our method in Fig. B.4. Despite its effectiveness, our method inherits certain constraints from the base pre-trained video diffusion model. It occasionally struggles with fine-grained structures, such as hands under extreme viewpoints, leading to anatomical distortions or motion artifacts. Furthermore, generating high-fidelity details for entirely unobserved regions—particularly real-world backgrounds—remains challenging. We attribute this primarily to the domain gap between our predominantly synthetic training data and the complexity of real-world scenes. Performance also tends to degrade under aggressive camera trajectories that necessitate large-scale synthesis of unseen areas.

C. 4D Controlled Dataset

We have curated our 4D-controlled dataset using PointOdyssey [90] and its released 3D assets, including character models, HDR environment textures, and indoor scene layouts. Specifically, we use 100 human-like characters and animate them using real-world motion capture data. The characters are paired with 3D environments that include both outdoor and indoor scenes. For outdoor scenes, we use 60 HDR environment textures to simulate natural backgrounds, while for indoor scenes, we use 20 unique 3D environments provided by PointOdyssey.

For each character–scene pair, we introduce independent control over camera viewpoints and world-time to generate 4D-controlled sequences. We first define linear world-time as the original motion-capture timeline, where the motion progresses at a constant rate. To enable world-time control, we replace this linear progression with a remapped timeline that changes how the animation advances over time. The character animation is then generated by retargeting the mocap sequence according to this remapped timeline. By accelerating, decelerating, or holding specific portions of the timeline, these remappings allow effects such as slow motion, pausing, and locally accelerated motion segments. Specifically, To realize these behaviors, we construct several temporal variants that introduce diverse world-time dynamics. These include a slow-motion variant, where consecutive output frames map to closely spaced points on the mocap timeline, effectively reducing the motion speed, and a pausing variant, where selected poses are held for extended durations. We also incorporate a reverse variant that plays the timeline backward to create inverted temporal dynamics. In addition, we use a random time-warping variant that maps the linear timeline to an upsampled one while enforcing local speed constraints, and a spline-based variant, where monotonic spline control points are sampled under slope constraints to produce smooth, continuously varying changes in motion speed. Together, these variants provide diverse and natural temporal augmentations and enable the

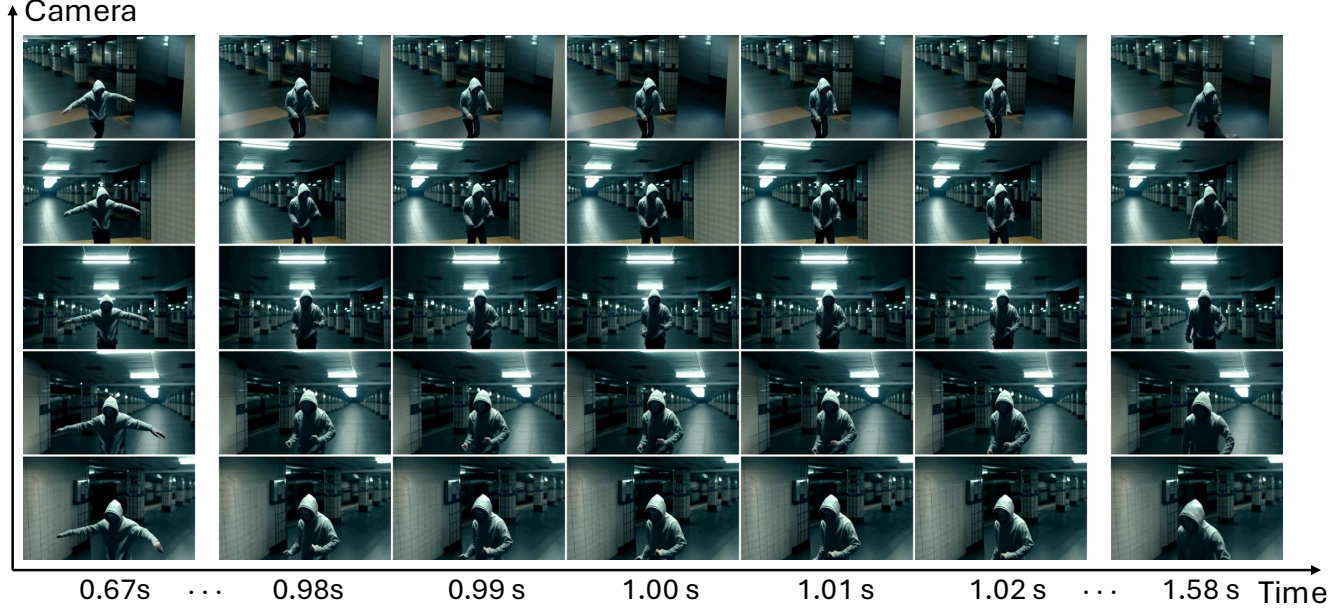


Figure B.3. **Dense camera-time grid visualization.** Our model supports independent control over temporal progression (horizontal axis) and camera motion (vertical axis).

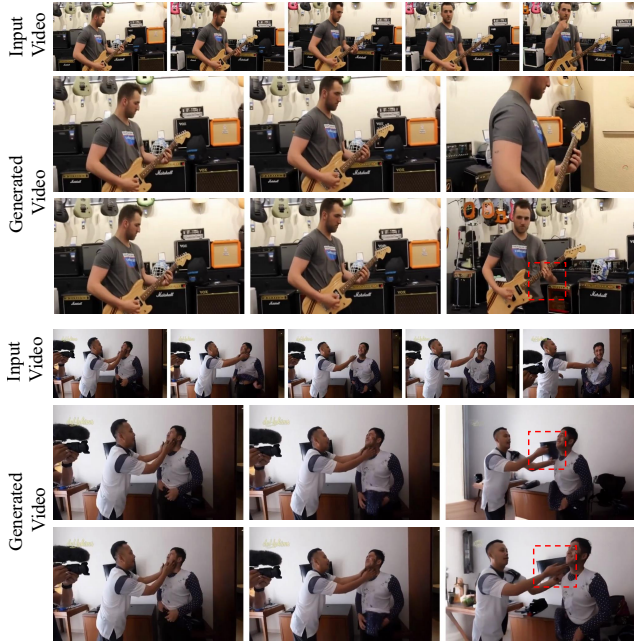


Figure B.4. **Failure Cases.** Our model may struggle with fine-grained hand motion and with generating high-quality background details under certain viewpoints.

model to learn a wide range of world-time controls. For each scene, we curate data using one linear timeline and three additional variants randomly selected from the above set.

We determine each camera position using a look-at center, a radius, and a pair of rotation angles representing azimuth and elevation. A static camera keeps these parameters fixed throughout the sequence. For dynamic trajectories, we sample 2–4 waypoint cameras by perturbing the look-at center, radius, and rotation angles, and then interpolate these waypoints to obtain a smooth camera path. To ensure realistic framing, we enforce specific constraints across all trajectories: the radius is sampled within a range of 4–12 meters, the total azimuth variation is limited to 75° , the elevation variation to 30° , and the look-at center offset to a maximum of 1 meter from the human character’s centroid. For each scene, we generate four distinct views: two trajectories with more than two waypoints, one orbit-style trajectory with a single waypoint, and one static camera. To produce natural, eased motion between waypoints, we randomly select either uniform-speed interpolation or a smoothstep function, defined as $f(t) = 3t^2 - 2t^3$ for $t \in [0, 1]$. We use fixed camera intrinsic settings with a focal length of 30 mm and a sensor width of 50 mm.

D. Baselines

ReCamMaster [8]. We utilize the official codebase and the released checkpoint, which was pre-trained on 122k synthetic videos. To ensure a fair comparison on our synthetic benchmark, we fine-tune ReCamMaster on our dataset of 20k synthetic videos. As demonstrated in Tab. D.1, while fine-tuning yields substantial performance gains, the method still significantly underperforms com-

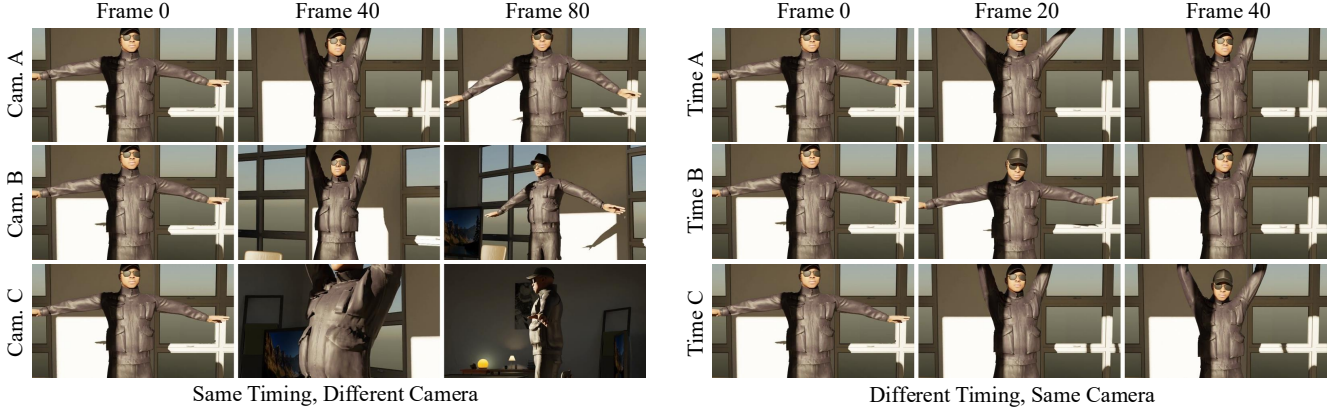


Figure C.1. **4D Controlled Data Sample.** We demonstrate the decoupled curation of spatial control (camera motion) and temporal control (timing). (Left) Three distinct camera views (Cam. A, B, C) show the same synchronized frames (Frame 0, 40, 80), confirming simultaneous camera and action control. (Right) Three videos with different action timings (Time A, B, C) are generated from the same camera trajectory, showing independent control over the temporal dimension.



Figure C.2. **Examples of Our 4D Controlled Data Samples.** We showcase the diversity of our generated dataset, which spans diverse motions, a variety of human-like subjects, both single- and multi-character scenes, and a wide range of indoor and outdoor environments. We will release the dataset and its generation code.

pared to our approach. This highlights the superior effectiveness of our proposed 4D control module.

We further observe that visual artifacts in ReCamMaster results stem primarily from the attempt to enforce time-control effects, as illustrated in Fig. D.1. When time control is disabled (top row), ReCamMaster exhibits significantly fewer artifacts compared to scenarios where time control is active (bottom row). Crucially, these artifacts persist even after fine-tuning on our time-controlled patterns. Given that time manipulation is a prerequisite for full 4D controllability, these results suggest that ReCamMaster’s two-stage pipeline suffers from intrinsic limitations in handling com-

plex temporal dynamics.

Table D.1. **Finetuning ReCamMaster on Our Dataset.**

	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
ReCamMaster	19.67	0.5426	0.2594
+ finetune on our data	21.86	0.5852	0.1846
Ours	24.57	0.6905	0.1265

TrajectoryCrafter [85]. We employ the official code-base and the provided checkpoint, trained on approximately 180k videos. TrajectoryCrafter relies on the reconstruction

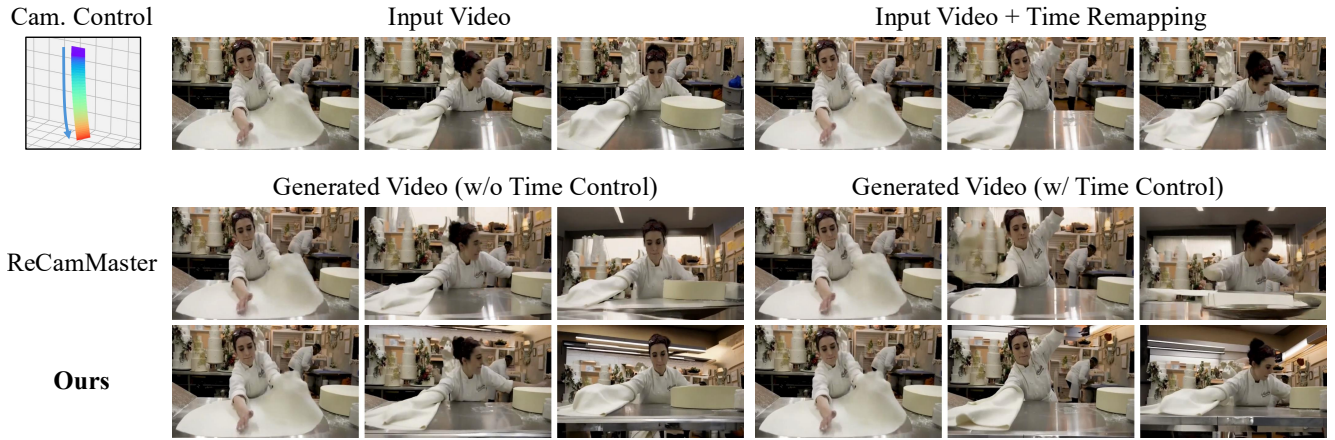


Figure D.1. **ReCamMaster Artifacts under Time Remapping.** In the left three columns we show how ReCamMaster and our method perform under no time remapping. The right three columns show the same generated scene with time remapping. We can observe how ReCamMaster creates significant artifacts after applying time remapping. In comparison our method does not suffer from this.

of dynamic point clouds from monocular videos. However, inaccuracies in these point clouds—often stemming from unreliable depth estimation—frequently lead to geometric artifacts, a limitation also noted in the original paper. Our experimental results show similar artifact cases driven by depth inconsistencies.