

Less is More: Data-Efficient Adaptation for Controllable Text-to-Video Generation

Supplementary Material

9. Dataset Details

Pyramid Sampling Strategy. We first describe our efficient sampling strategy, which balances coverage of the control space with the cost of generating synthetic scenes (see also Algorithm 4 in Section 10 for the corresponding pseudocode). Enumerating all combinations of control scalars and randomized scene settings would lead to a combinatorial explosion, so we adopt a *pyramid generation strategy* that varies the number of sampled scalars per layer while keeping the sampling distribution uniform.

Specifically, we divide the control range $[-1, 1]$ into five layers and sample $9 + 7 + 5 + 3 + 1 = 25$ scalar values using jittered sampling, ensuring that *every point in the interval has equal probability of being selected*. For each layer, we generate six scenes using our randomized scene-generation procedure (described later in this section), and each scene is rendered across all scalar values assigned to its layer.

This results in $6 \times 25 = 150$ total training samples. Compared to exhaustive sampling ($30 \text{ scenes} \times 150 \text{ conditions} = 4,500$ combinations), our pyramid strategy achieves a **30 \times reduction** in data size while preserving uniform coverage of the control range and diversity across the randomized context space.

Shutter Speed. Given a sampled scalar $c \in [-1, 1]$ and a randomized scene, we synthesize videos using a simple 2D physics-based renderer. Each scene contains 1–3 colored geometric primitives (circle, square, triangle, star) moving with constant velocities on a uniform background inside a 512×512 canvas, with elastic boundary reflections. We map c to an effective frame rate $\text{FPS}(c)$ via a centered log mapping, and set the exposure duration to $1/\text{FPS}(c)$. For each output frame, we then integrate over the exposure window by temporally averaging 32 analytically predicted sub-frames, advancing the shapes according to their motion model at each sub-step. Crucially, the underlying shape trajectories and backgrounds are shared across all shutter values for a given scene, so the only changing factor along the control axis is the strength of the motion blur.

Aperture. For aperture control, we build a 3D depth-of-field image dataset in Blender. Each scene contains 2–4 rigid objects (cubes, spheres, cylinders, cones, pyramids) placed on a ground plane with a vertical back wall, both sharing a neutral or softly colored material. Objects are assigned distinct colors and arranged at different depths between d_{\min} and d_{\max} , forming a clear foreground–midground–background ordering. We always se-

lect one foreground object as the focus target and attach the camera’s DoF focus to this object.

Given a sampled scalar $c \in [-1, 1]$, we map it to a physical f -number using the same centered log mapping used for shutter speed, i.e., $c \mapsto \text{fstop}(c) \in [\text{fstop}_{\text{lo}}, \text{fstop}_{\text{hi}}]$. For each scene, we then render multiple conditions by varying only the aperture $\text{fstop}(c)$ while keeping camera pose, object geometry, materials, and lighting fixed. Lighting consists of a gray world background plus an optional point light placed on a circle around the scene (either per-scene or per-render), which introduces mild but controlled highlights without changing depth layering. This produces aligned image sets in which the foreground remains in focus while mid- and background objects exhibit depth-dependent defocus changes in accordance with the aperture control scalar.

Temperature. We start from sharp 2D scenes and apply a global white-balance transformation. Each base scene consists of 2–4 colored shapes (circles, squares, triangles, stars) randomly placed on a 512×512 canvas with a uniform background color. We first render a single reference image per scene without blur or noise. Then, for each sampled scalar $c \in [-1, 1]$, we map c to a target color temperature $K(c) \in [K_{\text{lo}}, K_{\text{hi}}]$ using a perceptually uniform Mired-based mapping, and apply a Kelvin-to-RGB white-balance adjustment to the entire image. The transform is defined relative to a reference neutral temperature (e.g., $K_{\text{ref}} = 6500 \text{ K}$) and optionally re-normalized to preserve average luminance. This yields condition-aligned image sets in which geometry, layout, and background remain fixed, while only the global color cast smoothly shifts from warm to cool as a function of the temperature control scalar.

10. Algorithmic Details

We provide high-level pseudocode for the pyramid sampling strategy used in dataset construction, our model’s core attention block, and our data-free evaluation methodologies.

10.1. Jointly Trained Attention Block

Algorithm 1 outlines the forward pass of a `WanAttentionBlock` that has been adapted with our joint training strategy. It details how the base LoRA update is integrated into the backbone’s weights and how the conditional signal from the `FPSCrossAttentionAdapter` is computed and combined with the text-based content signal.

Algorithm 1: Forward Pass of a Jointly Trained Attention Block

Input : x : Input video latent features
 c_{text} : Text context embedding
 c_{cond} : Scalar physical condition (e.g., in $[-1, 1]$)
 W_q, W_k, W_v, W_o : Pre-trained cross-attention weights
 ΔW_{loa} : Learned base LoRA update
Adapter_{cond}: Trained conditional adapter module
Output: x_{out} : Updated video latent features

```

/* 1. Form the final adapted backbone weights */
 $W'_q \leftarrow W_q + \Delta W_{\text{loa},q}$ 
 $W'_k \leftarrow W_k + \Delta W_{\text{loa},k}$ 
 $W'_v \leftarrow W_v + \Delta W_{\text{loa},v}$ 
 $W'_o \leftarrow W_o + \Delta W_{\text{loa},o}$ 

/* 2. Compute Query, Text Keys, and Text Values */
 $q \leftarrow \text{norm}(x \cdot (W'_q)^T)$ 
 $K_{\text{text}} \leftarrow \text{norm}(c_{\text{text}} \cdot (W'_k)^T)$ 
 $V_{\text{text}} \leftarrow c_{\text{text}} \cdot (W'_v)^T$ 

/* 3. Compute the text-based content signal */
 $y_{\text{text}} \leftarrow \text{Attention}(q, K_{\text{text}}, V_{\text{text}})$ 

/* 4. Compute Conditional Keys and Values via Adapter */
 $e_{\text{cond}} \leftarrow \text{MLP}_{\text{cond}}(c_{\text{cond}})$ 
 $K_{\text{cond}}, V_{\text{cond}} \leftarrow \text{Adapter}_{\text{cond}}(e_{\text{cond}})$ 

/* 5. Compute the conditional signal */
 $y_{\text{cond}} \leftarrow \text{Attention}(q, K_{\text{cond}}, V_{\text{cond}})$ 

/* 6. Combine signals with a learned gate  $g$  */
 $y_{\text{combined}} \leftarrow y_{\text{text}} + g \cdot y_{\text{cond}}$ 

/* 7. Apply final output projection */
 $x_{\text{out}} \leftarrow x + y_{\text{combined}} \cdot (W'_o)^T$ 

```

10.2. Backbone Spectral Drift Analysis

To support the examination of Hypothesis 1 in Section 7, we evaluate how fine-tuning modifies the backbone’s attention projections. Algorithm 2 describes the “Backbone Intruder Dimension Check.” This data-free method quantifies catastrophic forgetting by measuring the spectral similarity between the original pre-trained backbone weights and the final adapted backbone weights after joint training.

Algorithm 2: Backbone Intruder Dimension Check

Input : W_{pre} : A pre-trained backbone weight matrix
 W_{loa} : The final adapted backbone weight matrix ($W_{\text{pre}} + \Delta W_{\text{loa}}$)
 k : Number of top singular vectors to check (e.g., 64)
 ϵ : Intruder similarity threshold (e.g., 0.5)
Output: $N_{\text{intruders}}$: Count of detected intruder dimensions

```

/* 1. Compute SVD for both weight matrices */
 $U_{\text{pre}}, \sigma, V_{\text{pre}} \leftarrow \text{SVD}(W_{\text{pre}})$ 
 $U_{\text{loa}}, \sigma', V_{\text{loa}} \leftarrow \text{SVD}(W_{\text{loa}})$ 
 $N_{\text{intruders}} \leftarrow 0$ 

/* 2. Check top-k vectors of the adapted backbone */
for  $j \leftarrow 1$  to  $k$  do
     $u_{\text{loa},j} \leftarrow j\text{-th column of } U_{\text{loa}}$ 
    /* 3. Find max similarity to any pre-trained vector */
     $S_{\text{max}} \leftarrow 0$ 
    for  $i \leftarrow 1$  to number of columns in  $U_{\text{pre}}$  do
         $u_{\text{pre},i} \leftarrow i\text{-th column of } U_{\text{pre}}$ 
         $s \leftarrow \text{abs}(\text{cosine\_similarity}(u_{\text{loa},j}, u_{\text{pre},i}))$ 
         $S_{\text{max}} \leftarrow \max(S_{\text{max}}, s)$ 
    /* 4. Count as intruder if similarity is below threshold */
    if  $S_{\text{max}} < \epsilon$  then
         $N_{\text{intruders}} \leftarrow N_{\text{intruders}} + 1$ 

return  $N_{\text{intruders}}$ 

```

10.3. Data-Free Spectral Evaluation of the Conditional Adapter

To complement the comparative spectral analysis of Hypothesis 2 in Section 7, we employ a data-free procedure (“Principal Component Showdown”) summarized in Algorithm 3. This method constructs a diagnostic testbench directly from the model’s weights by extracting the dominant principal directions of the backbone’s attention projections. We then evaluate the conditional adapter’s response to these canonical content directions and compute the singular value spectrum of the resulting signal.

10.4. Pyramid Sampling Strategy

Algorithm 4 formalizes the pyramid generation strategy introduced in Section 9. The key idea is to partition the control range $[-1, 1]$ into L layers with a decreasing number of

Algorithm 3: Data-Free Spectral Rank Analysis
 (“Principal Component Showdown”)

Input : W'_q, W'_k, W'_v : final adapted backbone projection weights
 Adapter_{cond}: trained conditional adapter module
 c_{strong} : a strong condition value (e.g., $c = 1.0$)
 N : number of principal components to test (e.g., $N = 64$)
Output: $S_{\text{text}}, S_{\text{cond}}$: normalized singular value spectra
 $\mathcal{R}_{\text{text}}, \mathcal{R}_{\text{cond}}$: effective ranks of backbone and conditional signals

```

/* 1. Extract principal content primitives from the backbone */
 $U_q, -, - \leftarrow \text{SVD}((W'_q)^T)$ 
 $U_k, -, - \leftarrow \text{SVD}((W'_k)^T)$ 
 $U_v, -, - \leftarrow \text{SVD}((W'_v)^T)$ 
 $q_{\text{test}} \leftarrow \text{TopN}(U_q, N)$ 
 $k_{\text{text\_test}} \leftarrow \text{TopN}(U_k, N)$ 
 $v_{\text{text\_test}} \leftarrow \text{TopN}(U_v, N)$ 

/* 2. Simulate backbone response to principal content directions */
 $y_{\text{text\_principal}} \leftarrow \text{Attention}(q_{\text{test}}, k_{\text{text\_test}}, v_{\text{text\_test}})$ 

/* 3. Generate and simulate conditional response */
 $K_{\text{cond}}, V_{\text{cond}} \leftarrow \text{Adapter}_{\text{cond}}(c_{\text{strong}})$ 
 $y_{\text{cond\_principal}} \leftarrow \text{Attention}(q_{\text{test}}, K_{\text{cond}}, V_{\text{cond}})$ 

/* 4. Compute singular value spectra */
 $-, S_{\text{text}}, - \leftarrow \text{SVD}(\text{flatten}(y_{\text{text\_principal}}))$ 
 $-, S_{\text{cond}}, - \leftarrow \text{SVD}(\text{flatten}(y_{\text{cond\_principal}}))$ 
Normalize  $S_{\text{text}}, S_{\text{cond}}$  by their leading singular values

/* 5. Estimate effective ranks */
 $\mathcal{R}_{\text{text}} \leftarrow \text{EffectiveRank}(S_{\text{text}})$ 
 $\mathcal{R}_{\text{cond}} \leftarrow \text{EffectiveRank}(S_{\text{cond}})$ 

return  $S_{\text{text}}, S_{\text{cond}}, \mathcal{R}_{\text{text}}, \mathcal{R}_{\text{cond}}$ 

```

scalar values per layer, then assign each layer a fixed number of independently randomized scenes. This ensures uniform coverage of the full control range while keeping the total dataset size tractable.

11. Training Details

All models are trained on NVIDIA A100 80GB (A100E) GPUs using Wan2.1-T2V-14B as the backbone with bfloat16 precision. Unless noted otherwise, we follow the

Algorithm 4: Pyramid Sampling Strategy for Dataset Construction

Input : L : number of pyramid layers (e.g., $L = 5$)
 $\mathbf{n} = [9, 7, 5, 3, 1]$: number of scalars per layer
 S : number of scenes per layer (e.g., $S = 6$)
Output: \mathcal{D} : training dataset of (scene, scalar, rendered clip) triples

```

 $\mathcal{D} \leftarrow \emptyset$ 

/* 1. Iterate over pyramid layers */
for  $l \leftarrow 1$  to  $L$  do
    /* 2. Jittered scalar sampling: divide  $[-1, 1]$  into  $n_l$  equal bins, sample one value per bin */
     $\mathcal{C}_l \leftarrow \text{JitteredSample}([-1, 1], n_l)$ 
    /* 3. Generate  $S$  randomized scenes for this layer */
    for  $s \leftarrow 1$  to  $S$  do
         $\text{scene}_s \leftarrow \text{GenerateScene}()$ 
        /* 4. Render each scene at all scalars assigned to this layer */
        for  $c \in \mathcal{C}_l$  do
             $\text{clip} \leftarrow \text{RenderScene}(\text{scene}_s, c)$ 
             $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\text{scene}_s, c, \text{clip})\}$ 

return  $\mathcal{D}$ 

```

same configuration across all experiments. Each run trains for up to 1000 epochs with a micro-batch size of 1 per GPU, gradient accumulation of 4, and a single pipeline stage. This corresponds to a global batch size of 4 and yields roughly 18,000 training steps for our 30-shot synthetic dataset. Under this setup, each pyramid training stage completes in approximately one day on two A100E.

We use four condition-adapter tokens, an adapter embedding dimension of 256, and LoRA rank 32 for both the condition adapters and the backbone LoRA, with the adapter gate fixed at 0.5 throughout training. A learning rate of 2×10^{-5} with a 100-step warmup is applied. Optimization is performed using the AdamW optimizer [20] with betas (0.9, 0.99), weight decay 0.01, and $\varepsilon = 10^{-8}$. All LoRA and adapter parameters are trained in bfloat16 with activation checkpointing enabled for memory efficiency.

12. Experiment Details

In this section, we provide additional details for the experimental comparisons introduced in Section 6.

12.1. Group 1: Data Complexity (Syn. vs. Real)

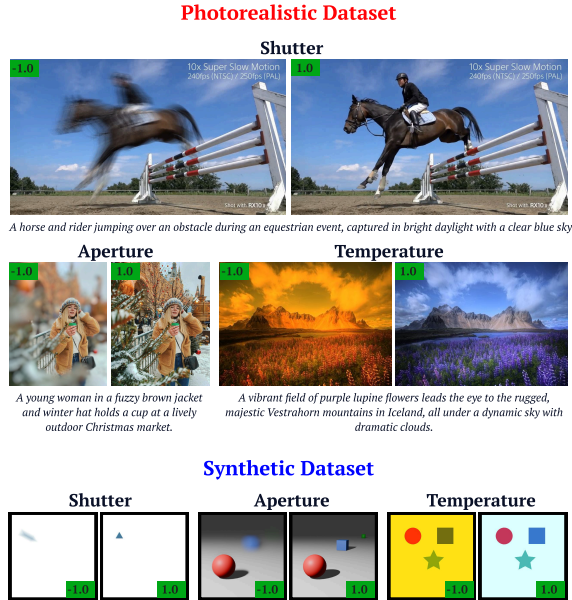


Figure 7. Comparison of the photorealistic and synthetic one-shot datasets used in Group 1. For each physical effect (shutter speed, aperture, temperature), we show the two extreme control values ($c = -1$ and $c = 1$). Although only the endpoints are visualized, each dataset contains seven aligned conditions in total within the normalized range $c \in [-1, 1]$.

Dataset Construction. For the Group 1 comparison, we adopt a controlled one-shot setting in which each physical effect (shutter speed, aperture, temperature) is represented by a single scene containing seven scalar conditions that are aligned across the synthetic and photorealistic datasets (see Figure 7). All conditions are normalized to the shared range $c \in [-1, 1]$, ensuring that both datasets express identical control values despite differences in rendering styles.

Evaluation Prompt Details. Following the prompt construction strategy used in VBench—which we also rely on for the FEP metric—we generate a diverse set of 96 motion-centric T2V prompts using an LLM (Gemini 2.5). The prompts span eight categories (animals, architecture, food, humans, lifestyle, plants, scenery, and vehicles), with 12 prompts per category. We use this set to evaluate SVP under a broad range of motion patterns and scene types.

Quantitative Results Recap. As shown in Figure 4, models fine-tuned on our simple synthetic dataset exhibit substantially smaller drift from the pretrained backbone when monitored using the FEP metrics (Section 5.1). In contrast, training on the photorealistic dataset leads to noticeably larger and faster deviation throughout training. This difference in drift is reflected in the final SVP results (Sec-

tion 5.2): the model trained on synthetic data maintains near-baseline semantic fidelity, while the model trained on photorealistic data shows clear backbone degradation. These findings highlight that, for learning physical controls, low-complexity synthetic data provides a more stable and robust supervision signal.

Visualization Evaluation. Beyond the quantitative trends, the backbone degradation becomes most apparent when inspecting the model’s inference outputs during training (see Figure 8). When trained on our synthetic dataset, the model continues to follow the input prompt and preserves the overall appearance of the generated scene. In contrast, the model trained on the photorealistic dataset begins to visually drift toward the training images themselves: details, textures, and global color tones increasingly resemble the underlying photorealistic scene rather than the prompted content. This “copying” behavior appears early in training and progressively intensifies, manifesting as semantic drift, texture collapse, and color hallucination. These visual failures provide an intuitive counterpart to the backbone corruption measured by our FEP and SVP metrics.

12.2. Group 2: Inference Strategy (Decoupled vs. Full-LoRA)

While Group 1 highlights the stability advantage of synthetic data during training, here we analyze how inference strategy further affects the final output quality. Specifically, we compare **Full-LoRA Inference**, which retains all backbone LoRA weights at test time, against our proposed **Decoupled Inference**, which prunes the shallow DiT blocks to remove residual content drift before decoding.

Quantitative Results Recap. Similar to the quantitative trends observed in Group 1, the metrics in Group 2 also reveal a clear advantage for Decoupled Inference. As summarized in Table 1 and the corresponding FEP/SVP curves and charts in Figure 4, Decoupled Inference consistently yields lower drift and higher semantic fidelity compared to Full-LoRA Inference across all physical controls. Even when the model is trained on the synthetic dataset—where backbone corruption is already minimal—Decoupled Inference further reduces the residual deviations and restores the model’s output closer to the original WAN 2.1 behavior. These quantitative results motivate a closer look at the visual differences between the two inference modes.

Visual Evidence. The visual comparisons further reinforce the quantitative trends. As shown in the last column of Figure 8, applying Decoupled Inference to a corrupted checkpoint trained on photorealistic data dramatically restores prompt fidelity and global appearance: the model no longer copies textures or colors from the training scene, and semantic drift is substantially reduced. This demonstrates that much of the degradation observed during training is

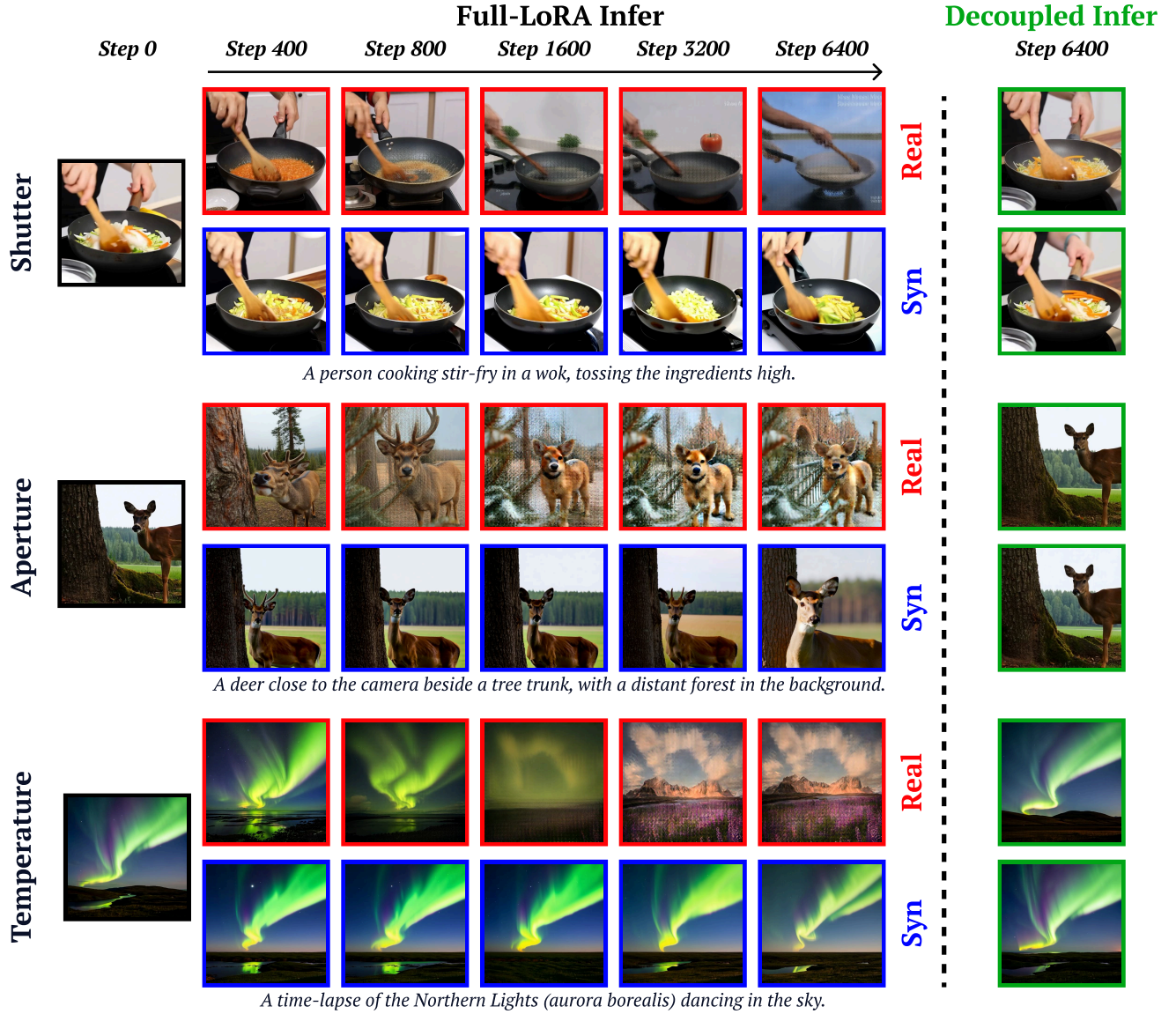


Figure 8. Visual comparison of backbone corruption during training. The synthetic-trained model remains stable and continues to follow the input prompt, whereas the model trained on photorealistic data rapidly drifts toward copying its training scene (see Fig. 7). This manifests as semantic drift, texture collapse, and global color shifts that intensify over training.

not permanently baked into the weights but instead arises from residual content leakage that Decoupled Inference effectively removes.

A similar pattern appears even when the model is trained on the synthetic dataset. Although backbone corruption is minimal in this setting, and the model is capable of yielding high-quality output as shown in Figure 3, Full-LoRA Inference still injects subtle synthetic-style bias—such as flattened textures or blurred shading—into the final output. As shown in Figure 9, Decoupled Inference suppresses these artifacts and produces videos that closely match the

visual style and sharpness of the original WAN 2.1 backbone across different control values.

Limitations of Real Data. Although Decoupled Inference substantially reduces the corruption introduced by real-data training, it does not fully eliminate it—especially under extreme control values. As shown in Figure 10, the model trained on the real shutter speed dataset still exhibits noise and structural artifacts when the control approaches the boundaries (e.g., $c = -1.0$), even after applying Decoupled Inference. This occurs because the real sequence contains complex high-entropy motion: articulated limbs, nonlinear

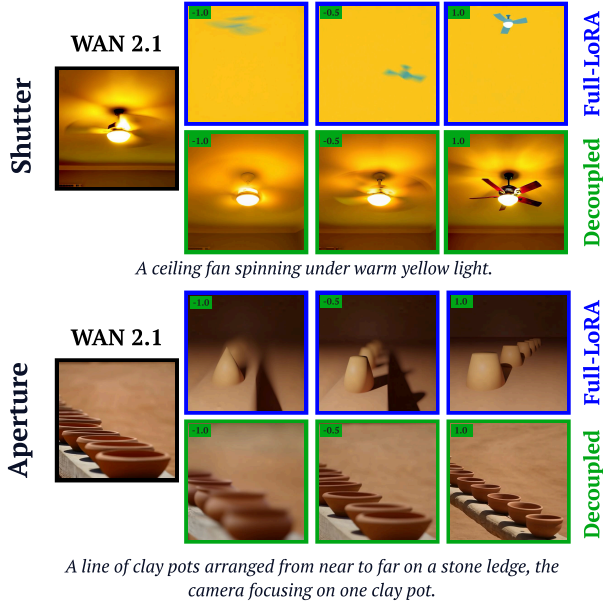


Figure 9. Effect of Decoupled vs. Full-LoRA Inference on a model trained with the synthetic pyramid dataset. Although synthetic training induces only mild backbone drift, Full-LoRA Inference still injects subtle synthetic-style biases (e.g., flattened shading and softened textures) into the outputs. Applying Decoupled Inference removes these artifacts and restores the visual style of the original WAN 2.1 model across different control values.



Figure 10. Residual artifacts under Decoupled Inference at extreme control values ($c = -1.0$, shutter speed). Even after pruning the shallow LoRA blocks, the model trained on real data (left) exhibits visible noise and structural degradation in high-motion regions (red inset), owing to the complex, high-entropy content of the real training sequence. The model trained on synthetic data (right) produces a clean motion-blur effect under the same condition (blue inset), consistent with its simpler, more disentangled supervision signal.

trajectories, and strong background parallax. In contrast, the synthetic shutter speed dataset is constructed from simple rigid motion with clean parametric trajectories, providing a much more disentangled supervision signal.

These observations reinforce our *Less is More* thesis: low-complexity synthetic data is inherently easier for the model to factorize and scales reliably across the full control range, whereas real data introduces irreducible entan-

glement that Decoupled Inference can mitigate but not fully correct. Thus, synthetic data remains the preferred choice for learning stable, well-behaved physical controls.

13. Quantitative Controllability Analysis

To quantitatively validate the precision and monotonicity of our learned controls, we perform a post-hoc analysis on existing 5-point sweeps $c \in \{-1, -0.5, 0, 0.5, 1\}$ across N diverse prompts per control (no additional training required). For each frame, we compute a lightweight image-level proxy capturing the intended physical effect: **shutter** uses Laplacian-variance sharpness (inversely proportional to blur strength); **aperture** uses background-region sharpness (inversely proportional to defocus); **temperature** uses a global warm-cool color-cast statistic. We then measure the Spearman rank correlation ρ between the control scalar c and the proxy across the 5-point sweep for each prompt.

Results are reported in Table 2. The median $|\rho|$ is 1.00 across all three controls, with 100% correct direction in every case, confirming that the learned controls produce consistent, monotonic responses to the conditioning scalar. The slightly lower mean $|\rho|$ for aperture (0.912) reflects occasional non-perfect adjacent steps attributable to proxy sensitivity at fine-grained defocus levels or scene-content variation; the corresponding sweep videos in Section 16 confirm that the visual effect remains well-behaved throughout.

Table 2. **Quantitative Monotonicity Analysis.** Spearman rank correlation $|\rho|$ between the control scalar c and a per-effect image proxy, computed over 5-point sweeps ($c \in \{-1, -0.5, 0, 0.5, 1\}$). N is the number of prompts evaluated per control. Direction accuracy is 100% for all controls.

Control	N	Median $ \rho $	Mean $ \rho $
Shutter Speed	15	1.000	0.987
Aperture	17	1.000	0.912
Temperature	14	1.000	1.000

14. Out-of-Range Inference

A natural question is how the model behaves when the conditioning scalar c is pushed beyond the training range $[-1, 1]$. Figure 11 shows qualitative results for $c \in \{-1.5, -1.0, 0.0, 1.0, 1.5\}$ across all three controls, where $c = \pm 1.5$ falls outside the training distribution.

The model exhibits **plausible extrapolation**: the intended physical effect continues to strengthen smoothly beyond the training boundary without degrading prompt fidelity or visual coherence. For shutter speed, $c = -1.5$ produces stronger motion blur while scene content remains coherent; for aperture, $c = -1.5$ yields more extreme bokeh with the subject still well-rendered; for color tem-

perature, the warm-cool gradient extends naturally at both extremes. This behavior is consistent with successful low-dimensional disentanglement: the learned conditioning operates on a compact, continuous manifold that generalizes modestly beyond its training support. Probing the limits of this extrapolation—and characterizing where prompt guidance eventually degrades at more extreme values—is an interesting direction for future work.

15. Analysis Details

15.1. Hypothesis 1: The Spectrum of Backbone Adaptation

Expanding on Hypothesis 1 in Section 7, we analyze how the complexity of fine-tuning data influences the backbone’s spectral drift under conditional adaptation. Unless otherwise noted, all measurements are performed on the value projection W_v , though we observe qualitatively similar trends in W_q , W_k , and W_o . Figure 12 reports the results for the shutter speed condition using both real and synthetic training data as discussed in Sec. 12. Other physical controls exhibit the same overall behavior: simple synthetic data preserves the pretrained spectral structure, whereas complex real data drives substantial high-rank intrusions into the backbone.

Clean Adaptation (Single Synthetic Scene). As shown in Fig. 12a, synthetic shutter speed conditioning results in minimal drift throughout the network. The similarity heatmap remains uniformly low (green), indicating that the adapter modifies W_v through small non-destructive adjustments. This corresponds to a clean and stable form of adaptation: the model adjusts to the new condition while largely preserving the pretrained spectral structure. However, the specialization remains narrow, as the synthetic data contains only one simple scene.

Catastrophic Forgetting (Single Complex Scene). In contrast, Fig. 12b reveals that real shutter conditioning induces substantial spectral drift in W_v , particularly in shallow and mid-depth blocks. A large number of high-similarity intruders (cosine similarity > 0.5) appear as early as blocks 0–12 and accumulate into dense clusters in deeper layers. This is the hallmark of catastrophic forgetting: the model learns strong, scene-specific features that overwrite general-purpose representations. These intruder directions propagate into the conditioned path during inference, producing the “ghosting” artifacts seen in the qualitative comparisons. Similar behavior appears under other real-data conditions.

Overall, this data-free analysis corroborates our quantitative findings and highlights that deeper blocks inherently

provide cleaner, more stable channels for condition modulation.

15.2. Hypothesis 2: Full Backbone LoRA Is Essential for Context-Disentangled Conditioning

In Hypothesis 2 from Section 7, our spectral analysis demonstrated that the jointly trained model produces a low-rank, effect-specific conditional signal, whereas an adapter-only model produces a high-rank signal that conflates the physical effect with training-scene content. Here we extend this comparison to four training configurations, each evaluated on the same temperature-conditioned tiger scene, to provide a comprehensive visual validation of why Joint Training with a full backbone LoRA constitutes the correct architectural choice. The objective is not to assess the photorealism of the temperature effect in isolation, but to determine whether each configuration can express the physical control cleanly without corrupting scene content or degrading the backbone’s generative priors.

Joint Training, Full Pyramid Dataset (Proposed). As shown in the second row of Figure 13, our proposed method satisfies both objectives. At $c = 0$, the output is visually indistinguishable from the backbone reference (first row), providing direct visual evidence that joint training is non-destructive and preserves the original generative priors. As c varies, the temperature shift is applied in a content-stable manner, consistent with the compact, low-rank conditional signal identified in Figure 6.

Partial Backbone Coverage: 1/3 LoRA + Adapter. The third row restricts the backbone LoRA to the deepest third of the DiT blocks—mirroring the block coverage used at inference time, now applied during training. While the temperature effect is successfully induced, an unnatural warm color bias emerges at positive control values. This reveals that the full backbone LoRA is necessary during training to absorb the synthetic domain shift across all network depths; without it, the unabsorbed distributional bias propagates into the conditioning pathway and manifests as a persistent, condition-correlated color offset.

Data Efficiency: One-Shot Joint Training. The fourth row applies the full joint training configuration to a single training scene. Backbone fidelity is well preserved, confirming that the Joint Training architecture remains non-destructive even under minimal data. The conditioning effect is slightly weaker compared to training on more scenes, suggesting that a more diverse training set tends to produce more robust physical conditioning; a systematic study of data scaling is left to future work.

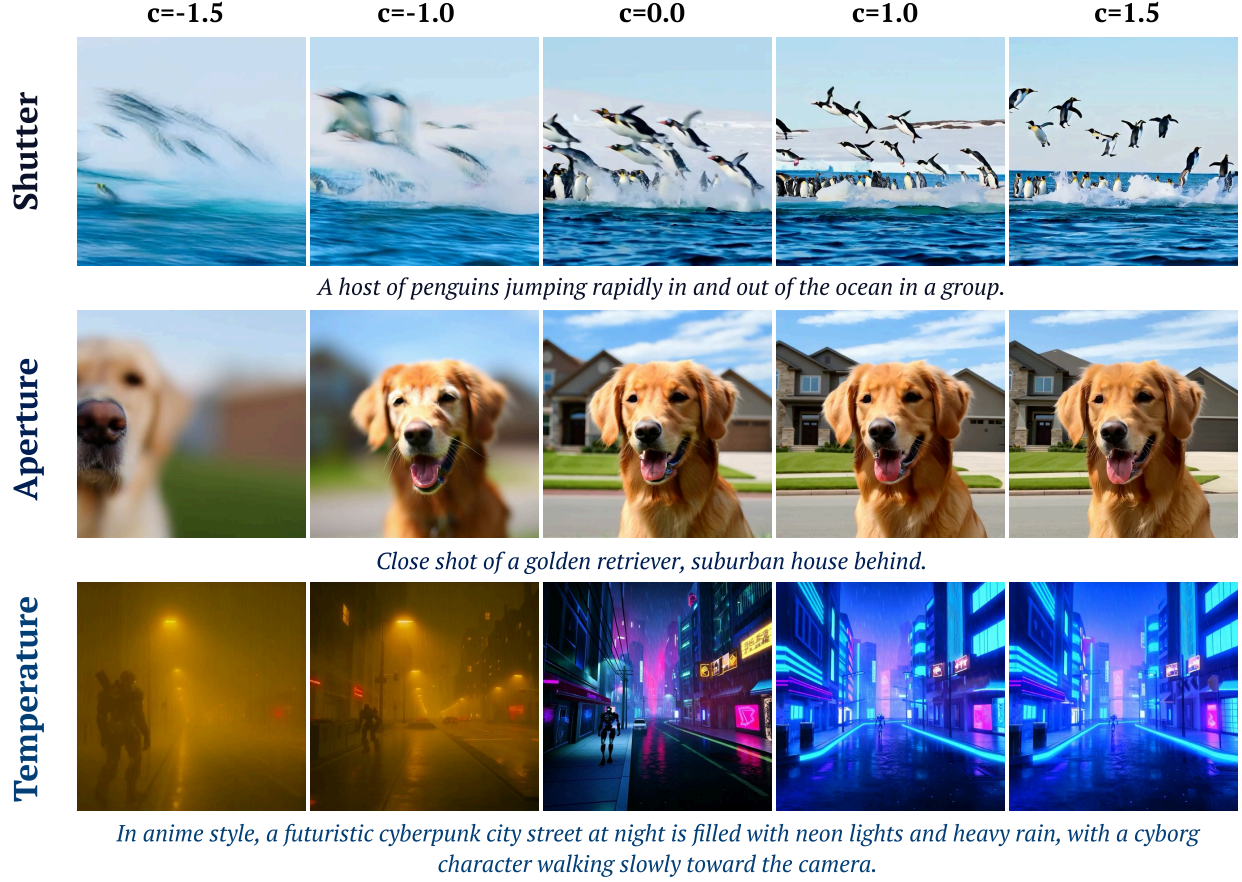


Figure 11. **Out-of-range inference.** Qualitative sweep over $c \in \{-1.5, -1.0, 0.0, 1.0, 1.5\}$ for shutter speed, aperture, and color temperature. Values $c = \pm 1.5$ lie outside the training range $[-1, 1]$. The model extrapolates plausibly in all three cases, with the physical effect strengthening monotonically and prompt fidelity preserved throughout.

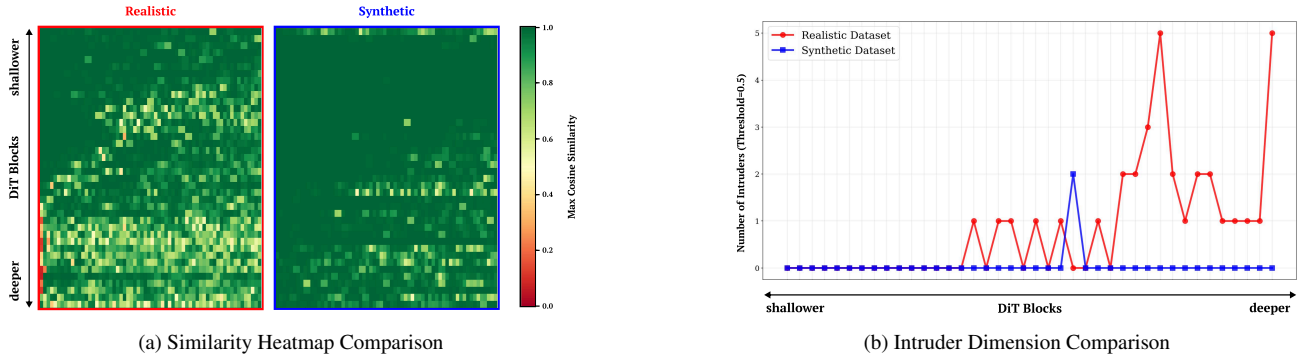


Figure 12. Backbone content drift across depth for the shutter speed condition. Analysis performed on the value projection W_v . (a) Max cosine similarity between content directions and FPS-conditioned features. (b) Number of content intruders (cosine > 0.5).

Adapter-Only Training and the Bulldozer Effect. The fifth row trains only the conditional adapter without a backbone LoRA. In the absence of a dedicated pathway for absorbing domain shift, the adapter is forced to encode both the physical effect and the training-scene context

within its conditional pathway. As described in Section 7, this produces the “Bulldozer Effect”: the high-rank, high-magnitude conditional signal suppresses the text-based content signal, causing the model’s output to be dominated by training-scene features irrespective of the conditioning

WAN 2.1



A tiger jumps over a water stream.

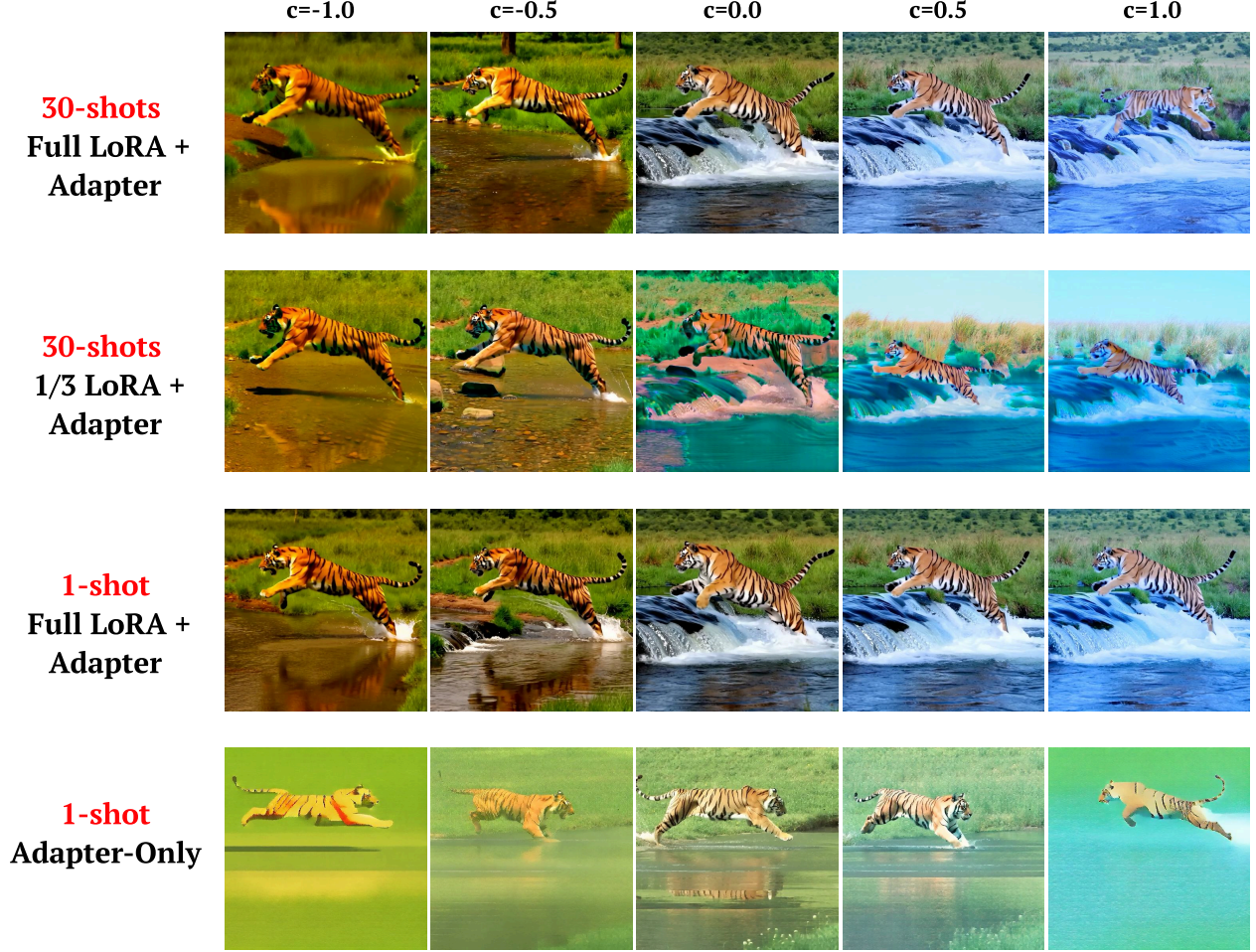


Figure 13. **Qualitative comparison of training configurations on the temperature control (tiger scene).** The first row shows the pre-trained Wan2.1 backbone output without conditioning, serving as the visual reference for generative fidelity. The second row (our proposed Joint Training on the full pyramid synthetic dataset) demonstrates non-destructive adaptation: at $c = 0$ the output is visually indistinguishable from the backbone reference, while varying c produces clean, isolated temperature changes without scene content drift. The third row (1/3 LoRA + Adapter) captures the conditioning effect but introduces an unnatural warm bias at positive control values, indicating that without a full backbone LoRA to absorb the synthetic domain shift, a distributional bias contaminates the conditioning pathway. The fourth row (one-shot Joint Training) preserves backbone fidelity comparably to the second row, but exhibits a slightly weaker conditioning effect. The fifth row (Adapter-Only training) exhibits the “Bulldozer Effect”: scene content is dominated by training-data features regardless of the condition value.

scalar.

Taken together, these four configurations confirm a consistent finding: Joint Training with a full backbone LoRA

is the only configuration that simultaneously achieves accurate physical conditioning and non-destructive backbone adaptation, while increasing training scene diversity further

strengthens the robustness of the learned control.

16. Example Gallery

Figure 14 presents a diverse set of examples generated with our shutter-, aperture-, and temperature-controlled T2V framework. Each physical control is parameterized within a unified range $c \in [-1, 1]$, enabling smooth and interpretable adjustments to motion blur, depth of field, and color tone. Across a wide variety of scenes and prompts, the outputs change in a consistent and approximately monotonic manner as c varies, demonstrating the reliability and generality of our learned control space.

To further assess this generalization, Figures 15, 16, and 18 evaluate each control dimension in more challenging scenarios. Shutter control extends to moving-camera settings and scenes with multiple independently moving objects, while maintaining predictable blur variation. Aperture control remains stable across diverse depth layouts and can shift focus to locations beyond the foreground, producing smooth bokeh shifts as depth varies (see Figure 17). Temperature control remains consistent across indoor environments and highly stylized domains such as anime and pixel art. Together, these results show that our learned controls generalize far beyond the simple one-shot synthetic scenes used for training.

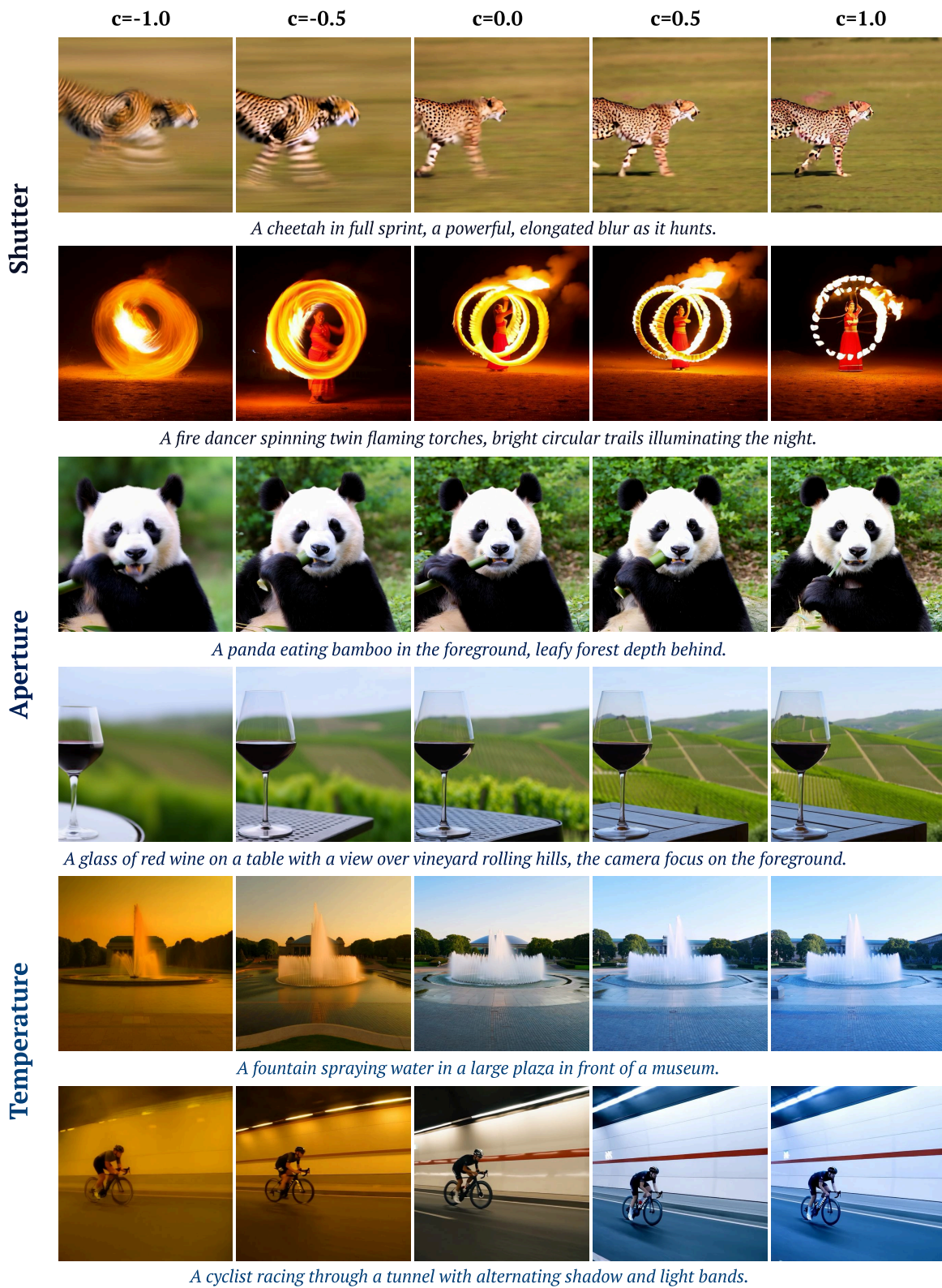


Figure 14. **Qualitative results of our controllable generation.** Our model demonstrates precise and continuous control over shutter speed (Rows 1–2, motion blur), aperture (Rows 3–4, bokeh), and color temperature (Rows 5–6) by varying the conditional input c from -1.0 to 1.0 across diverse, high-fidelity video prompts.

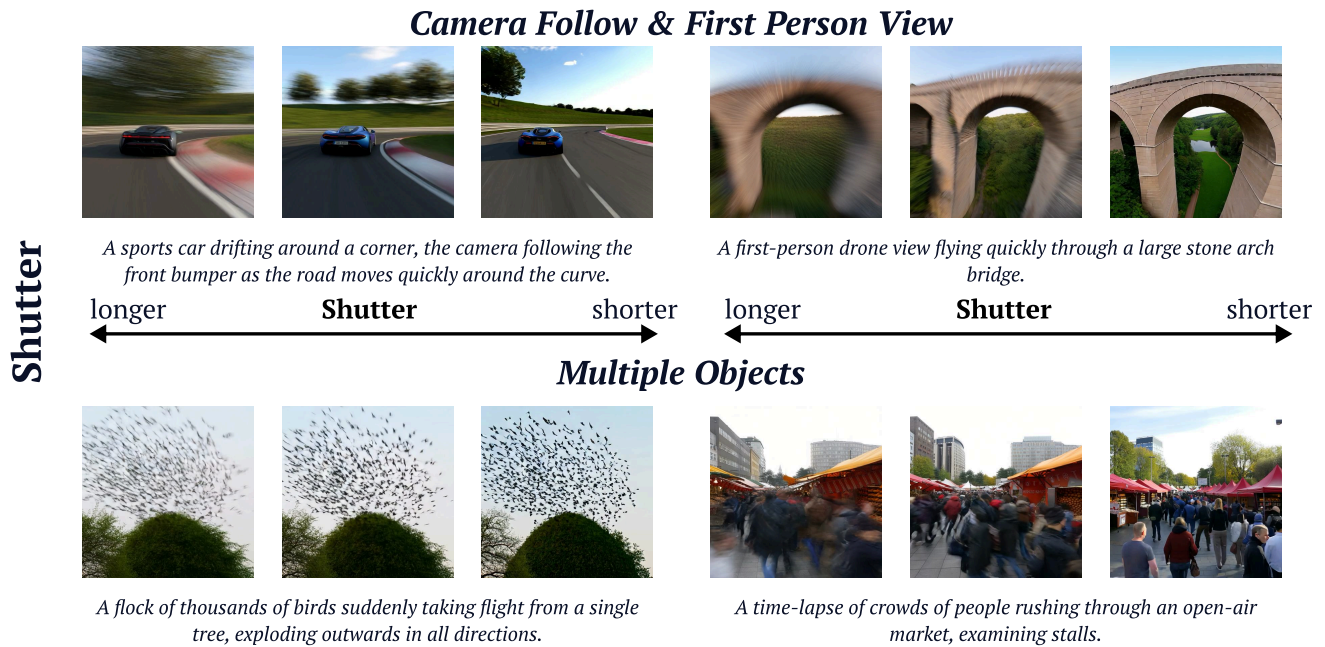


Figure 15. **Generalization of shutter control to scenes with complex motion.** The model responds reliably to the shutter scalar in settings involving moving cameras (e.g., camera-follow and first-person views) and scenes with multiple independently moving objects.

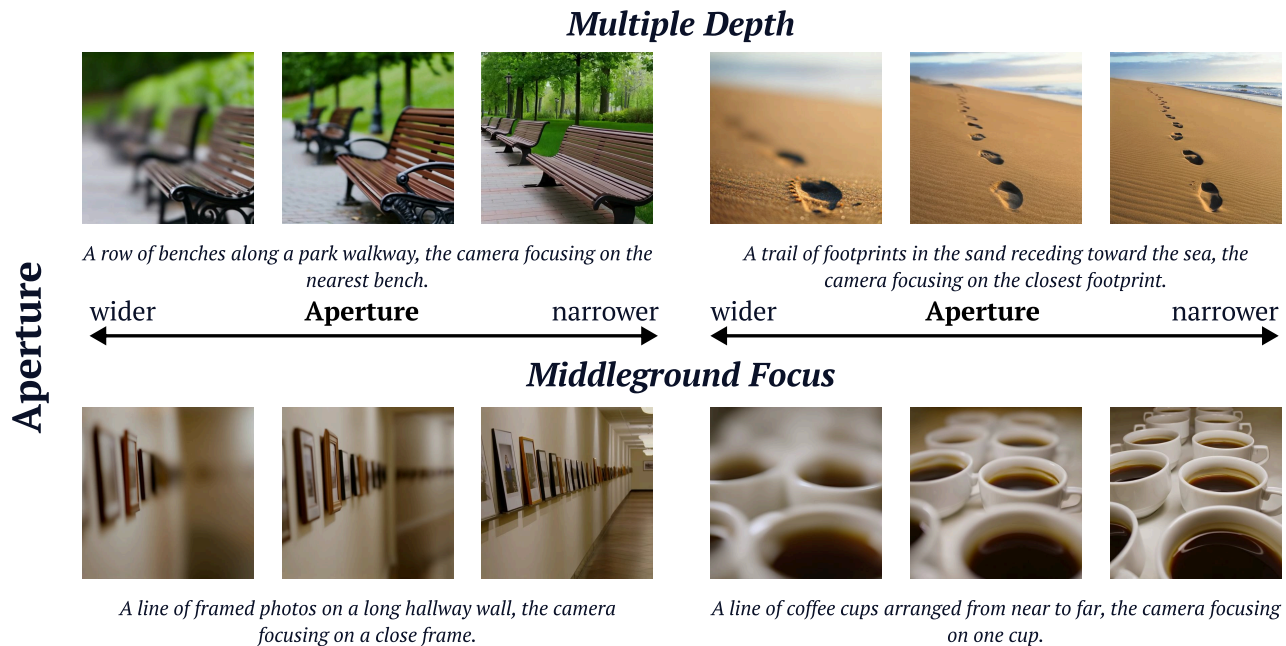


Figure 16. **Generalization of aperture control across diverse depth layouts and focal targets.** The model handles scenes with multiple depth layers and varied spatial arrangements, and can focus on locations beyond the foreground (e.g., mid- or background planes).

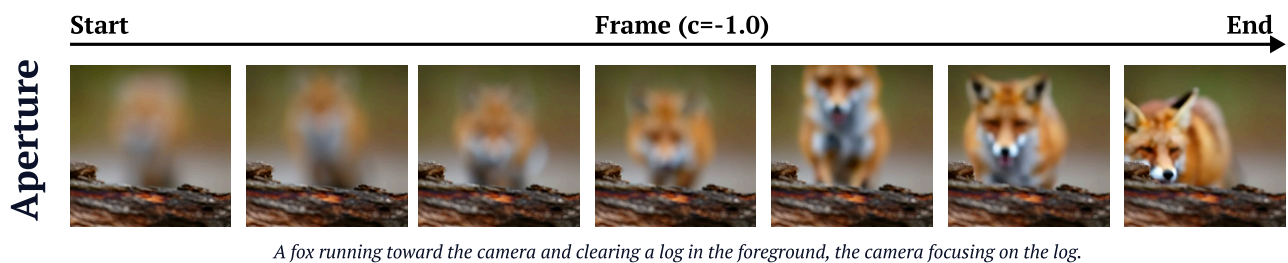


Figure 17. Despite being trained only on images, the model renders smooth bokeh variation as depth changes, enabled by the backbone’s strong prior.

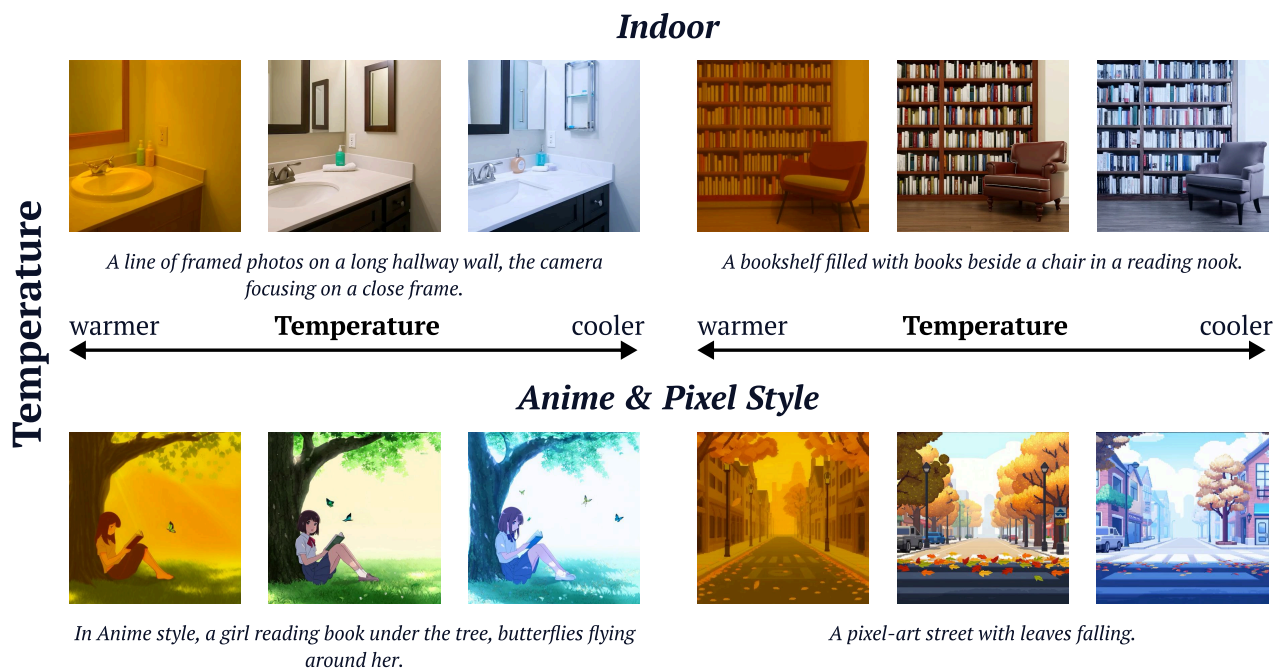


Figure 18. **Generalization of temperature control across a wide range of scene types.** The model produces stable cooler-to-warmer transitions in indoor environments as well as highly stylized domains such as anime and pixel art.