Just-in-Time: Training-Free Spatial Acceleration for Diffusion Transformers

Supplementary Material

A. The detailed derivation of SAG-ODE

This section provides the formal derivation for the SAG-ODE introduced in the main paper. We restate the governing equation for a given stage k, which operates on the active token set Ω_k:

$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \mathbf{\Pi}_k \, \boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{S}_k^\top \mathbf{y}(t), t). \tag{11}$$

Here, $\mathbf{\Pi}_k$ is the augmented lifter operator, and the term $\boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{S}_k^\top \mathbf{y}(t), t)$ represents the velocity computed by the Transformer using only the m_k active tokens. The following derivation shows how this compact form is obtained by starting from an exact orthogonal decomposition of the ideal velocity field and then applying the proposed approximation.

A.1. Derivation via orthogonal decomposition

Let $\boldsymbol{v}^*(\mathbf{y}, t) = \boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{y}, t) \in \mathbb{R}^{Nd}$ denote the ideal, full-dimensional velocity field that acts on all N tokens. Using the projection matrix $\mathbf{P}_k = \mathbf{S}_k \mathbf{S}_k^\top$, we can decompose this velocity field into two orthogonal components: One lying within the subspace spanned by the anchor tokens (active subspace) and the other in its orthogonal complement (inactive subspace). The decomposition is given by

$$\boldsymbol{v}^* = \mathbf{P}_k \boldsymbol{v}^* + (\mathbf{I} - \mathbf{P}_k)\boldsymbol{v}^*. \tag{12}$$

Substituting $\mathbf{P}_k = \mathbf{S}_k \mathbf{S}_k^\top$ into the first term, we obtain

$$\boldsymbol{v}^* = \mathbf{S}_k(\mathbf{S}_k^\top \boldsymbol{v}^*) + (\mathbf{I} - \mathbf{P}_k)\boldsymbol{v}^*, \tag{13}$$

where $\mathbf{S}_k^\top \boldsymbol{v}^* \in \mathbb{R}^{m_k d}$ represents the precise velocity projected onto the m_k anchor tokens.

A.2. Approximation strategy

Computing the full velocity $\boldsymbol{v}^*$ is computationally prohibitive. To accelerate inference, we introduce two coherent approximations corresponding to the two terms in the decomposition:

1. **Sparse input approximation (active component):** Instead of computing the velocity based on the full context $\mathbf{y}$, we approximate the active component by evaluating the network only on the projected anchor tokens $\mathbf{S}_k^\top \mathbf{y} \in \mathbb{R}^{m_k d}$. We define this computable sparse velocity as
$$\tilde{\boldsymbol{u}} := \boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{S}_k^\top \mathbf{y}, t) \in \mathbb{R}^{m_k d}. \tag{14}$$
Thus, the first term in Eq. (12) is approximated by the embedding of this sparse calculation $\mathbf{S}_k \tilde{\boldsymbol{u}} \in \mathbb{R}^{Nd}$.
2. **Subspace extrapolation (inactive component):** For the second term $(\mathbf{I} - \mathbf{P}_k)\boldsymbol{v}^*$, which represents the velocity of the inactive tokens, we avoid explicit neural function evaluations. Instead, we assume that the dynamics of the inactive tokens can be reasonably inferred from the active anchors via a pre-defined interpolation operator $\mathcal{I}_k$ (please refer to Appendix B.2 for specific implementation details). This yields the approximation
$$(\mathbf{I} - \mathbf{P}_k)\boldsymbol{v}^* \approx \mathcal{I}_k(\tilde{\boldsymbol{u}}). \tag{15}$$

A.3. Synthesis and the augmented lifter

Substituting the approximations from the previous section, the approximate velocity field $\boldsymbol{v}$ is formed by combining the exact velocity for the anchor tokens with the interpolated velocity for the inactive ones. We now define the augmented lifter $\mathbf{\Pi}_k$ as the operator that encapsulates this entire process. Its action on the computed sparse velocity $\tilde{\boldsymbol{u}}$ is given by

$$\boldsymbol{v} = \mathbf{\Pi}_k \tilde{\boldsymbol{u}} := \mathbf{S}_k \tilde{\boldsymbol{u}} + \mathcal{I}_k(\tilde{\boldsymbol{u}}). \tag{16}$$

This definition states that the operator $\mathbf{\Pi}_k$ maps the sparse velocity to the full space. By substituting $\tilde{\boldsymbol{u}} = \boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{S}_k^\top \mathbf{y}, t)$, we arrive at the final, compact form of our SAG-ODE:

$$\frac{\mathrm{d}\mathbf{y}(t)}{\mathrm{d}t} = \mathbf{\Pi}_k \boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{S}_k^\top \mathbf{y}(t), t). \tag{17}$$

A.4. Proof of consistency

Applying the projection $\mathbf{S}_k^\top$ to the derived ODE confirms that the dynamics on the activate anchor tokens remain exact relative to the sparse computation:

$$\mathbf{S}_k^\top \frac{\mathrm{d}\mathbf{y}}{\mathrm{d}t} = \mathbf{S}_k^\top (\mathbf{S}_k \tilde{\boldsymbol{u}} + \mathcal{I}_k(\tilde{\boldsymbol{u}})) \tag{18}$$

$$= (\mathbf{S}_k^\top \mathbf{S}_k)\tilde{\boldsymbol{u}} + \mathbf{S}_k^\top \mathcal{I}_k(\tilde{\boldsymbol{u}}) \tag{19}$$

$$= \mathbf{I} \cdot \tilde{\boldsymbol{u}} + \mathbf{0} \tag{20}$$

$$= \boldsymbol{u}_{\boldsymbol{\theta}}(\mathbf{S}_k^\top \mathbf{y}, t). \tag{21}$$

Here, we utilize the property that $\mathbf{S}_k^\top \mathbf{S}_k = \mathbf{I}$ (orthonormality of the selector) and $\mathbf{S}_k^\top \mathcal{I}_k(\cdot) = \mathbf{0}$ (the interpolation operator has no effect on the anchor subspace by design). This concludes the derivation.

B. Implementation details

In this section, we provide further implementation details of our JiT framework, including the specific construction of the selector matrix $\mathbf{S}_k$ for the initial stage, the algorithm for our interpolation operators $\mathcal{I}_k$ and Φ_k, and the detailed hyperparameter configurations for the experimental results reported in the main text.

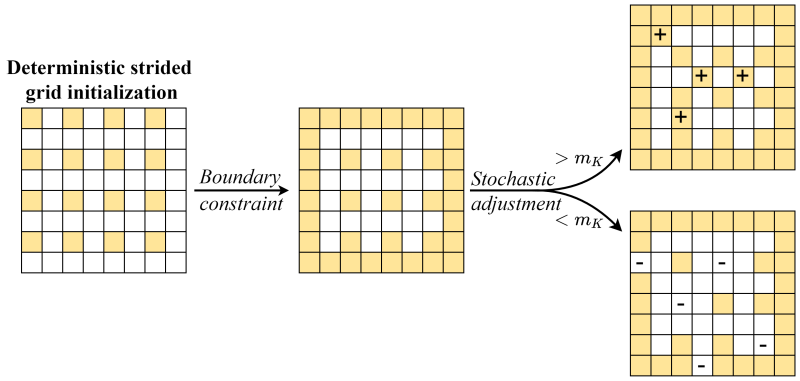

Figure 5. An illustration of the construction process for the initial selector matrix $\mathbf{S}_K$.

B.1. Construction of the initial selector matrix

As described in Sec. 3.3, for all subsequent stages $k < K$, anchor tokens are selected dynamically based on the importance map derived from the velocity field variance. Consequently, we only need to define the construction of the selector matrix $\mathbf{S}_K$ for the initial and most sparse stage.

For this initialization, our primary objective is to capture the global semantic structure while strictly adhering to the sparsity budget. To achieve this, we employ a deterministic strided grid initialization strategy, reinforced by explicit boundary constraints. Specifically, this strategy subsamples the latent grid with a fixed stride of 2 in both the height and width dimensions, which is akin to selecting the top-left token from every non-overlapping 2×2 block. To ensure the integrity of the image frame and prevent boundary artifacts, we augment this set by explicitly including all tokens located at the top, bottom, left, and right boundaries of the latent feature map.

Since the number of tokens resulting from this geometric pattern may not strictly align with the target budget m_K, we apply a final stochastic adjustment step. If the number of geometrically selected tokens falls short of m_K, we randomly sample supplementary indices from the remaining inactive set to fill the deficit. Conversely, if the count exceeds m_K, we randomly drop excess tokens from the generated set to strictly meet the m_K budget. This hybrid approach combines the regularity required for low-frequency structure generation with the flexibility to meet precise computational constraints. We provide a simplified schematic diagram of the above process in Fig. 5.

B.2. Implementation of interpolation operators

It should be clarified that while sharing the same algorithmic implementation, the two interpolation operators serve distinct roles within our framework, the subspace extrapolation operator $\mathcal{I}_k$ acts on the velocity field $\boldsymbol{u}_\theta$, whereas the structural prior operator Φ_k operates on the data latent tokens $\hat{\mathbf{y}}$.

We opt for a combination of nearest-neighbor interpolation and masked Gaussian blur rather than standard linear (e.g., bilinear) interpolation. This choice is driven by the nature of the high-dimensional representations in DiTs. Velocity fields and latent states often exhibit high-frequency spatial variations and complex semantic structures. Linear-based interpolation assumes a smooth gradient between sparse anchors, which can lead to over-smoothed features that drift off the valid feature manifold, particularly when the spatial grid is irregular or highly sparse.

In contrast, nearest-neighbor interpolation strictly preserves the original feature statistics and signal integrity of the anchor tokens. We then address the resulting spatial discontinuities (blockiness) via a controlled Gaussian blur, which smooths the transition boundaries without corrupting the feature values at the anchor positions. Specifically, derived from the token sparsity ρ, we approximate the average anchor gap as $L \approx \rho^{-1/2}$. We dynamically set the blur scale $\sigma = 0.4L$ such that the effective radius of the Gaussian kernel ($3\sigma \approx 1.2L$) slightly exceeds and covers this gap. This geometric constraint systematically suppresses blocky artifacts while strictly preventing over-smoothing.

We present the unified implementation in Algorithm 2.

Algorithm 2 Unified Interpolation Operator ($\mathcal{I}_k$ and Φ_k)

Input: Number of active tokens m, total tokens N, token dimension d, active features $\mathbf{Z_{act}} \in \{\boldsymbol{u}_{\mathbf{act}}, \hat{\mathbf{y}}_{\mathbf{act}}\} \in \mathbb{R}^{md}$, active indices $\mathcal{A}_{\mathbf{act}}$
Output: Interpolated full tensor $\mathbf{Z_{full}} \in \{\boldsymbol{u}_{\mathbf{full}}, \hat{\mathbf{y}}_{\mathbf{full}}\} \in \mathbb{R}^{Nd}$
Step 1: Nearest-Neighbor Interpolation
Generate full coordinate grid $\mathbf{C_{full}}$ and active coordinates $\mathbf{C_{act}}$
for each token $i \in \{1, \dots, N\}$ **do**
    $j^* = \arg\min_{j \in \mathcal{A}_{\mathbf{act}}} \|\mathbf{C_{full}}[i] - \mathbf{C_{act}}[j]\|_2$
    $\mathbf{Z_{NN}}[i] \leftarrow \mathbf{Z_{act}}[j^*]$
end for
Step 2: Controlled Gaussian Blur
Calculate token density $\rho = m/N$ and approximate anchor gap $L \approx \rho^{-1/2}$
Set blur scale $\sigma = 0.4L$ and kernel size $k \approx 3\sigma$ {Ensure $k > L$ for adequate spatial coverage}
$\mathbf{Z_{blur}} \leftarrow \text{GaussianBlur}(\mathbf{Z_{NN}}, \text{kernel} = k, \sigma = \sigma)$
Step 3: Masked Composition
Initialize binary mask $\mathbf{M} \leftarrow \mathbf{0}_N$
$\mathbf{M}[\mathcal{A}_{\mathbf{act}}] \leftarrow 1$ {Identify anchor positions}
$\mathbf{Z_{full}} \leftarrow \mathbf{M} \odot \mathbf{Z_{NN}} + (\mathbf{1} - \mathbf{M}) \odot \mathbf{Z_{blur}}$ {Preserve exact anchor values}
return $\mathbf{Z_{full}}$

B.3. Detailed experimental configurations

As shown in Tab. 4, we provide the detailed hyperparameter settings for our JiT framework under the $4\times$ and $7\times$ acceleration settings reported in the main paper. The configuration includes the total NFEs, the stage definition, and the specific token sparsity ratios used at each stage.

It is important to note that these configurations are not the product of an exhaustive, brute-force search. Instead, they are derived from a principled and lightweight hyperparameter recipe. Specifically, we first determine an initial token sparsity threshold that strictly ensures semantic stability at the generation starting point. To achieve the target acceleration ratios (e.g., $4\times$ or $7\times$), we then strategically allocate the solver steps, assigning fewer steps to the highly sparse early stages to maximize inference speed, and dedicating more steps to the denser late stages to preserve fine-grained generation quality.

B.4. Timestep schedule configuration

Standard uniform timestep sampling typically assumes a constant rate of information generation, which fails to capture the dynamic nature of the diffusion process. Recent spectral analysis in Beta Sampling [16] reveals that significant changes in

Parameter	JiT ($\sim 4\times$)	JiT ($\sim 7\times$)
Total NFEs	18	11
Number of Stages	3	3
Stage 2 (Coarse)		
Steps	7	4
Token Sparsity	35%	32%
Stage 1 (Medium)		
Steps	4	3
Token Sparsity	62%	60%
Stage 0 (Fine)		
Steps	7	4
Token Sparsity	100%	100%

Table 4. Detailed configuration of JiT schedules for different acceleration factors. The sparsity ratio denotes the fraction of tokens activated at each stage relative to the full sequence length N.

image content, specifically the establishment of low-frequency global structures, occur predominantly in the early stages of the denoising process, while high-frequency details are refined later.

To align the computational effort with this non-uniform information flow, we adopt a Beta-distribution-based strategy to define our timestep schedule. The core methodology leverages the probability integral transform (PIT) to warp the standard uniform timestep discretization. Formally, given a linear sequence $s_i \in [0, 1]$ representing the uniform progress of generation, we generate a non-uniform set of timesteps by mapping them through the inverse cumulative distribution function (CDF) of the Beta distribution. The final timesteps t_i corresponding to these warped time points are then calculated as

$$t_i = F^{-1}(s_i; \alpha, \beta), \tag{22}$$

where $F^{-1}(\cdot; \alpha, \beta)$ denotes the inverse CDF of the Beta distribution. This procedure effectively creates a non-uniform sequence of timesteps $\{t_i\}$ for the ODE solver.

In the context of our JiT framework, the robust formation of the global structure in the early phase is a prerequisite for the effectiveness of the subsequent spatial acceleration. Therefore, we prioritize the early denoising stages by allocating a denser distribution of timesteps to this period. We implement this by setting the hyperparameters to $\alpha = 1.4$ and $\beta = 0.42$. This configuration skews the sampling density towards the high-noise region (i.e., early in the denoising process). This ensures that the model dedicates sufficient NFEs to solidly establish the semantic layout and structural foundation before transitioning to the accelerated, spatially-sparse refinement stages.

B.5. Evaluation dataset

To ensure a comprehensive and unbiased evaluation, we constructed our evaluation dataset using the official prompt sets from the GenEval [3] and T2I-CompBench [7] benchmarks.

Specifically, for the 553 prompts from GenEval, we generated four images per prompt, each with a different random seed, yielding a subset of 2,212 images (553 prompts $\times$ 4 seeds). For the 2,400 prompts from T2I-CompBench, we generated one image per prompt.

Therefore, our full evaluation dataset comprises a total of 4,612 images, providing a robust and diverse foundation for the metrics reported in our experiments.

C. Additional ablation study on JiT scheduling

In this section, we provide further qualitative ablation studies on the scheduling hyperparameters of our JiT framework. These experiments are designed to offer insight into how different scheduling choices impact the trade-off between computational cost and generation quality, thereby justifying the configurations used in our main experiments.

C.1. Impact of the number of stages

The number of stages in the JiT schedule plays a significant role in defining the granularity of the coarse-to-fine generation process. To investigate this, we compare our default 3-stage setup against 2-stage and 4-stage variants, keeping the total

NFEs fixed at 18. For fairness, the sparsity ratios and transition timings in these variants are set to be approximately uniform (e.g., 50% sparsity for the first stage in the 2-stage schedule).

As visually demonstrated in Fig. 6, our default 3-stage schedule strikes the most effective balance. (a) The 2-stage schedule (50% $\rightarrow$ 100%) offers limited acceleration and produces a slightly blurred image, a quality degradation we attribute to the large, abrupt incorporation of half the tokens simultaneously. (b) Our default 3-stage schedule avoids both pitfalls, providing substantial acceleration in the early phases while allowing sufficient time for detail refinement in the later, denser stages. (c) Conversely, the 4-stage schedule, while computationally cheaper, suffers from noticeable quality degradation. Its final transition to full resolution occurs too late in the denoising process. By this point, the global structure is largely solidified, and the late introduction of high-frequency information results in persistent noise and artifacts that the model cannot fully remove in the remaining few steps.

Figure 6. Visual ablation on the number of stages for a fixed total NFE of 18. (a) A 2-stage schedule offers limited acceleration. (b) Our default 3-stage schedule yields the best trade-off. (c) A 4-stage schedule introduces persistent noise due to a late transition to full resolution. This validates that a 3-stage approach offers a superior balance between speed and quality.

C.2. Impact of token allocation (sparsity ratios)

Given a fixed 3-stage schedule, the allocation of tokens to the early and middle stages directly controls the computational cost and the quality of the structural prior. A more aggressive schedule (fewer tokens) saves more computation but risks creating a poor initial structure that is difficult to correct later. We explore this trade-off in Fig. 7 by comparing our default schedule (35% $\rightarrow$ 62% $\rightarrow$ 100%) with two variants: An aggressive schedule (20% $\rightarrow$ 50% $\rightarrow$ 100%) and a conservative one (50% $\rightarrow$ 75% $\rightarrow$ 100%).

The results are revealing. (a) The aggressive schedule, despite its significant acceleration, yields an image with compromised semantic integrity and lower overall quality. This suggests that an overly sparse initial set of tokens fails to capture sufficient global information, leading to errors that propagate through the entire generation process. (b) In contrast, our chosen token allocation robustly preserves generation quality while still achieving substantial acceleration. (c) On the other hand, while the conservative schedule also produces a high-quality result comparable to our default, its acceleration advantage is substantially diminished due to the high token count in the initial stages. This three-way comparison validates that our chosen approach provides the most effective compromise.

Figure 7. Visual ablation on token allocation for a fixed 3-stage, 18-NFE schedule. (a) An aggressive schedule (20% $\rightarrow$ 50%) compromises semantic integrity. (b) Our chosen schedule (35% $\rightarrow$ 62%) provides the best balance. (c) A conservative schedule (50% $\rightarrow$ 75%) offers similar quality but with a reduced acceleration advantage.

D. Extensibility to other backbone models

To demonstrate the generalizability of our JiT framework and its applicability beyond the FLUX.1-dev model, we conducted an additional experiment applying JiT to a different, publicly available backbone: Qwen-image [41]. Qwen-image is another powerful text-to-image model based on a DiT architecture, but with a distinct design and training paradigm compared to FLUX. This makes it an excellent test case for the model-agnostic nature of our approach.

By simply defining a multi-stage schedule with dynamic token selection and the corresponding SAG-ODE approximations, we are able to achieve a inference acceleration of approximately 4$\times$ (from 26.95s to 6.51s). As shown in the visual results in Fig. 8, the outcomes are highly compelling. Despite the substantial speedup, the model maintains a high degree of visual fidelity, successfully rendering complex scenes, textures, and semantic concepts presented in the prompts.

Furthermore, to rigorously validate the scalability of our method across diverse modalities, we extend JiT to the spatiotemporal domain using the HunyuanVideo-1.5 [35] backbone. Video diffusion models inherently process 3D latent representations, exhibiting significant redundancy both spatially within a single frame and temporally across continuous frames. Because our spatial acceleration strategy treats latents as a unified sequence of discrete tokens, it seamlessly identifies and bypasses these redundant spatiotemporal computations without requiring architectural modifications.

As illustrated in Fig. 9, we provide qualitative showcases of the generated video frames under both $\sim 4\times$ (from 1830.21s to 423.52s) and $\sim 7\times$ (from 1830.21s to 268.12s) acceleration settings. The visual outcomes confirm that JiT effectively maintains motion coherence and spatial structural fidelity even at extreme speedup ratios. This successful cross-modality application serves as strong evidence that JiT relies on the fundamental coarse-to-fine nature of the diffusion probability flow, rather than over-fitting to the architectural biases of a specific 2D model.

E. Additional qualitative comparisons

To further validate the robustness of our JiT framework, Fig. 10 presents additional qualitative comparisons on challenging prompts. The results consistently demonstrate the superiority of JiT over competing methods at similar computational budgets, particularly in preserving complex compositions and fine-grained details.

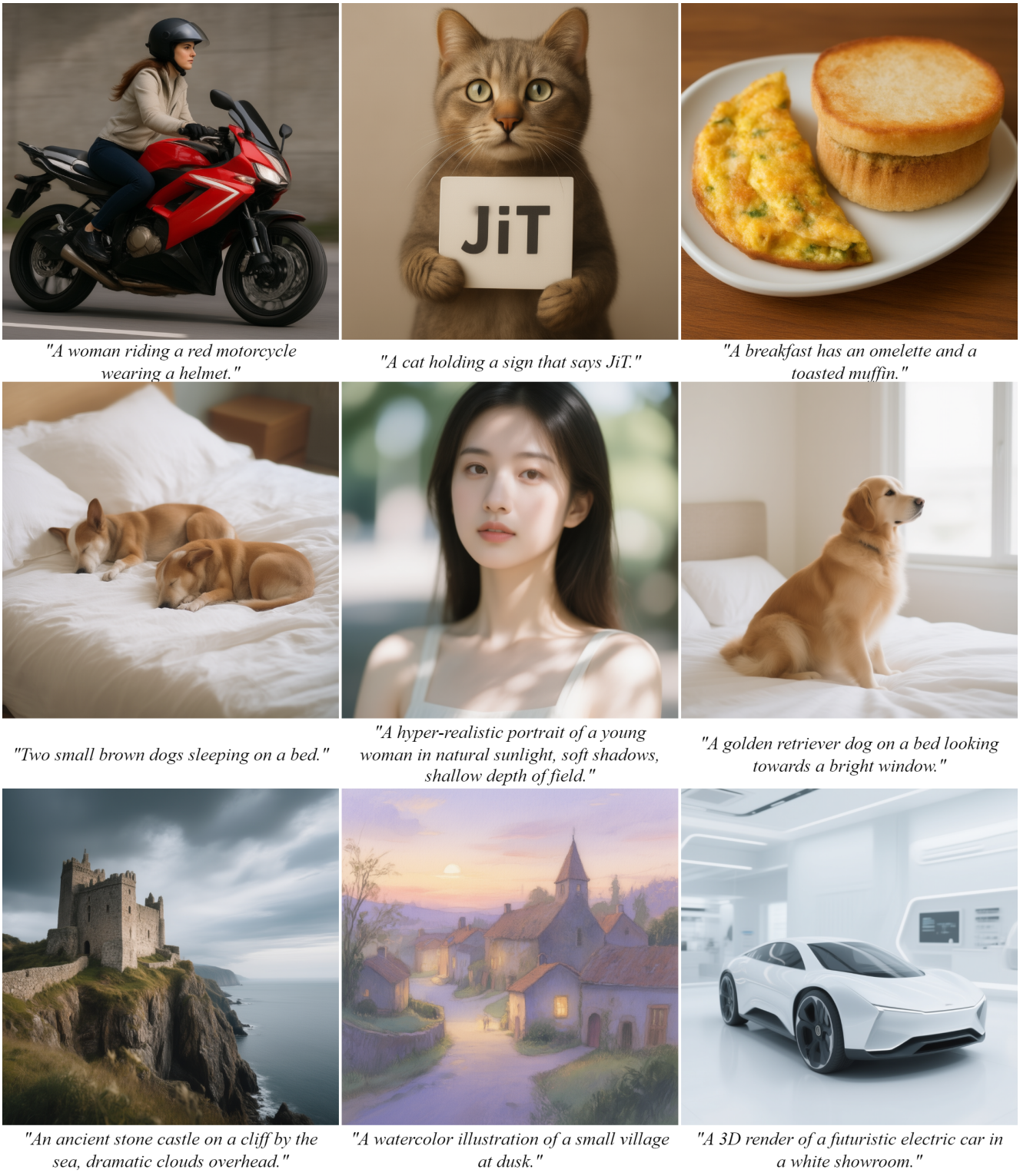

Figure 8. Qualitative results of our JiT framework applied to the Qwen-image model. The images were generated with $\sim 4\times$ acceleration compared to the standard pipeline.

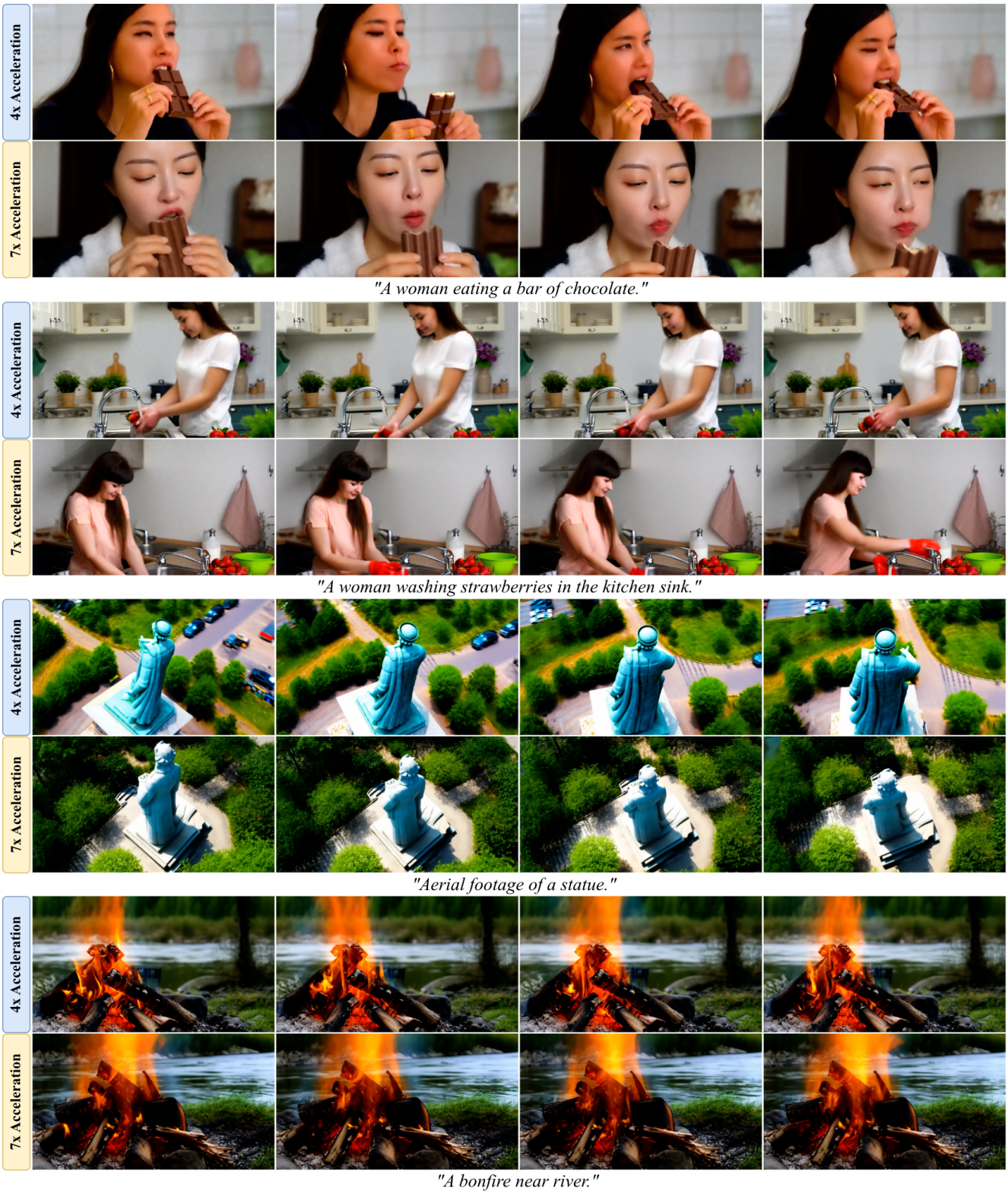

Figure 9. Qualitative results of our JiT framework applied to the HunyuanVideo-1.5 backbone. Our JiT framework demonstrates robust generalizability in the spatiotemporal domain, maintaining semantic consistency and temporal coherence under both $\sim 4\times$ and $\sim 7\times$ acceleration settings.

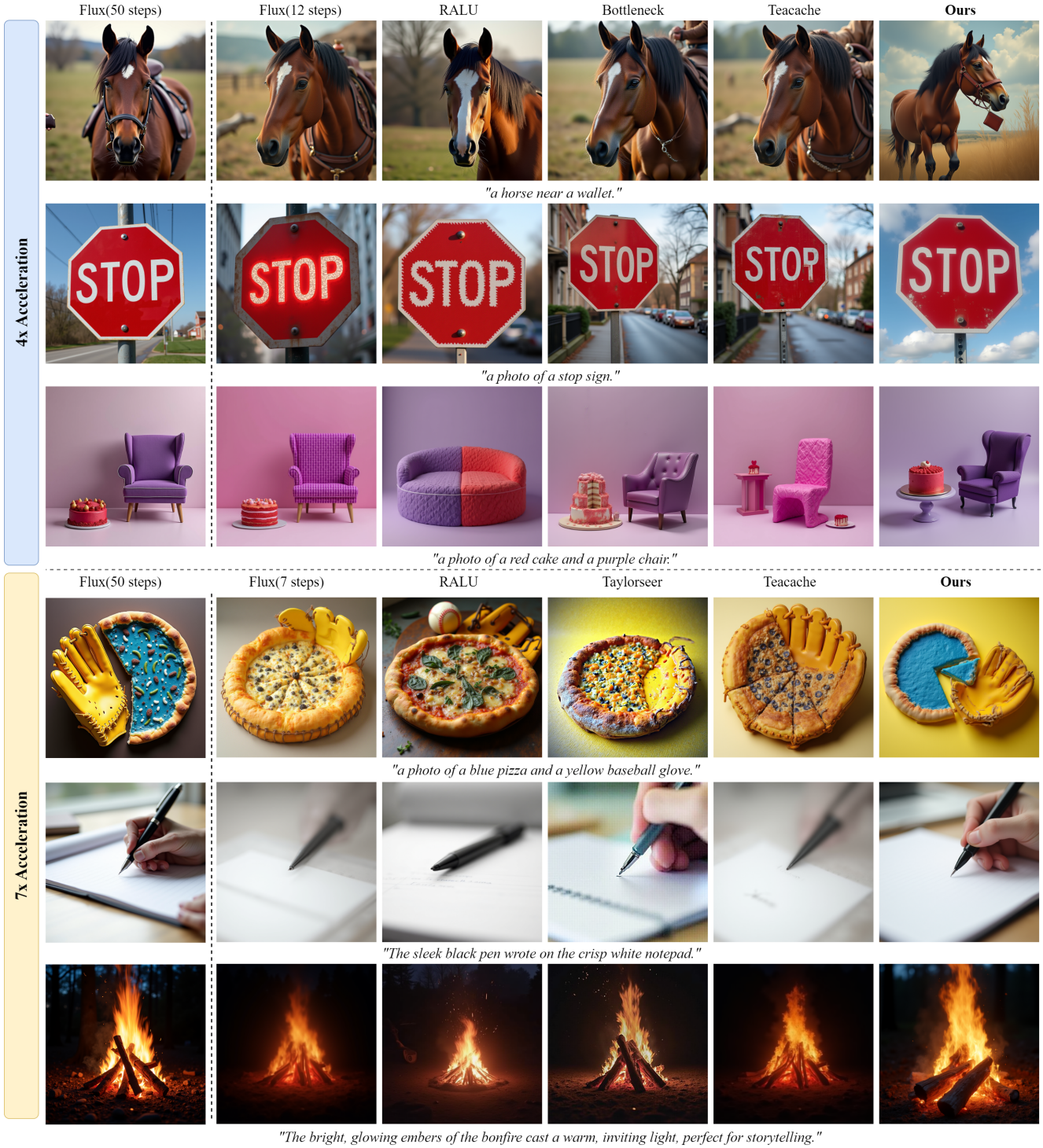

Figure 10. Additional qualitative comparisons of our JiT framework against baseline methods at $\sim 4\times$ and $\sim 7\times$ acceleration. These examples further showcase the superior performance of JiT across a variety of challenging prompts.