DDiT: Dynamic Patch Scheduling for Efficient Diffusion Transformers

Supplementary Material

Contents

6. Additional Qualitative Results

6.1. Text-to-Image Generation

In this section, we provide additional qualitative results for text-to-image generation. Fig. 11, 12 shows qualitative results on DrawBench [90] and Fig. 13, 14, 15, 16, 17 are results on PartiPrompts [126].

6.2. Text-to-Video Generation

We include additional qualitative results for text-to-video generation in the attached .mp4 file.

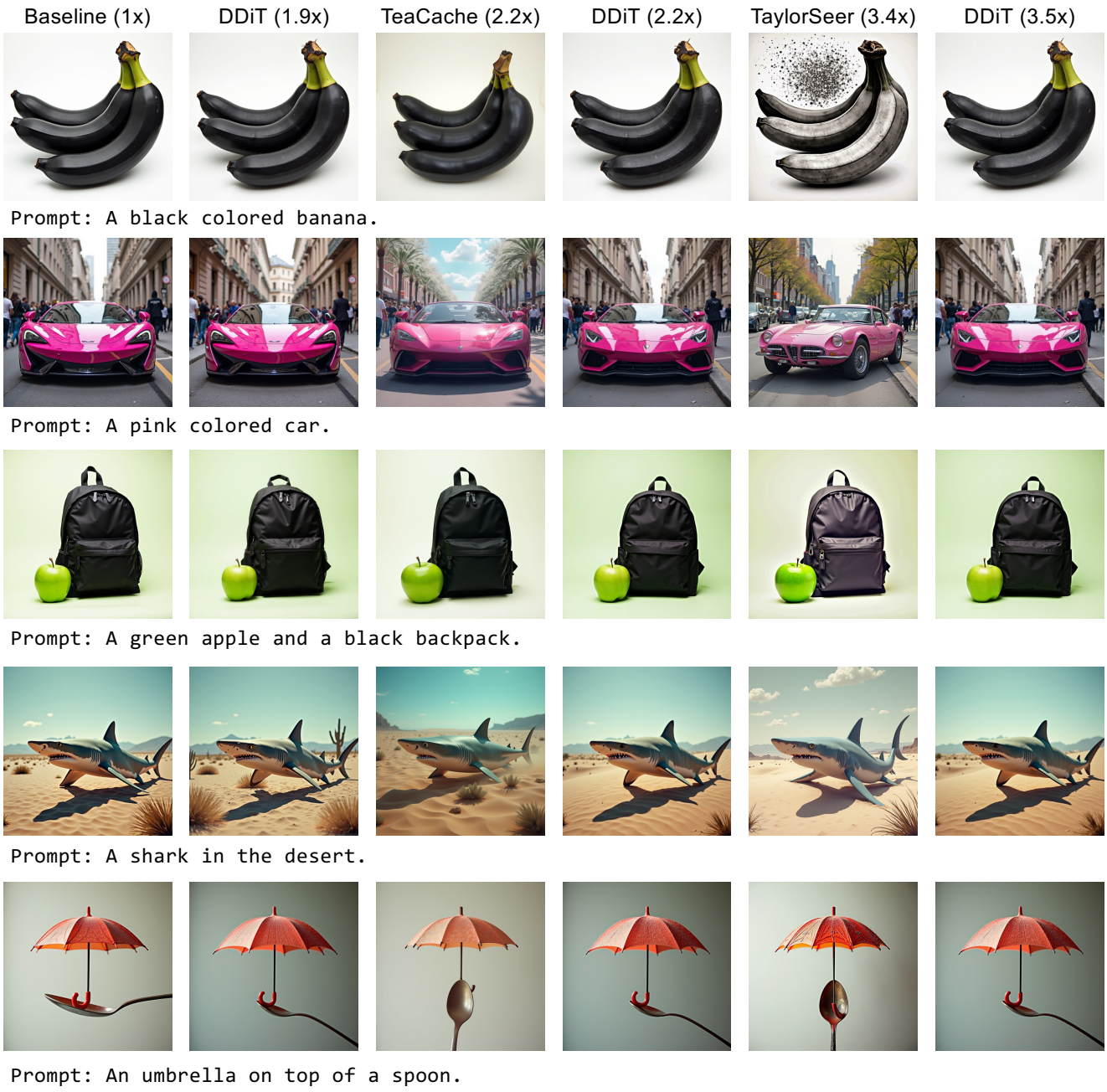

Figure 11. Qualitative comparisons with the base model [56], TeaCache [63], TaylorSeer [64], and DDiT on DrawBench [90] (1/2). The number next to the method name indicates the amount of computational speedup. Notice how for the same speedup, e.g., 2.2×, TeaCache [63] fails to preserve the relative position between objects in the prompt *An umbrella on top of a spoon.* Similar observations hold for TaylorSeer [64], where the details of the banana are not preserved in the prompt *A black colored banana.*

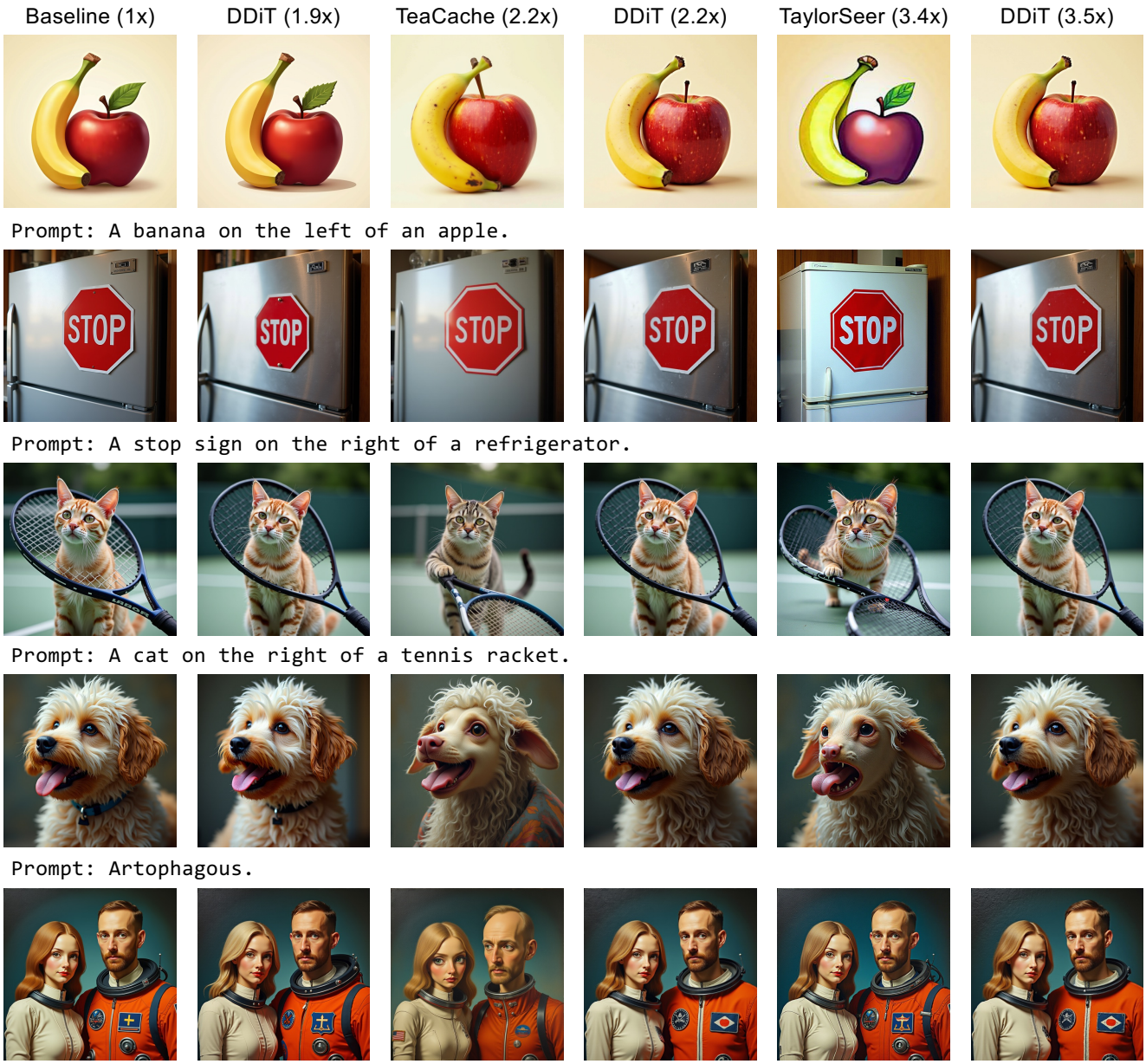

Figure 12. Qualitative comparisons on DrawBench (2/2). The number next to the method name indicates the amount of computational speedup. Notice how for the same speedup, e.g., 2.2×, TeaCache [63] loses fine-grained texture of the prompt *Artophagous* or the identity of the person in the last prompt of the astronaut couple. Similar observations hold true for TaylorSeer [64] where the position of the refrigerator is not preserved in the prompt, *A stop sign on the right of a refrigerator.*

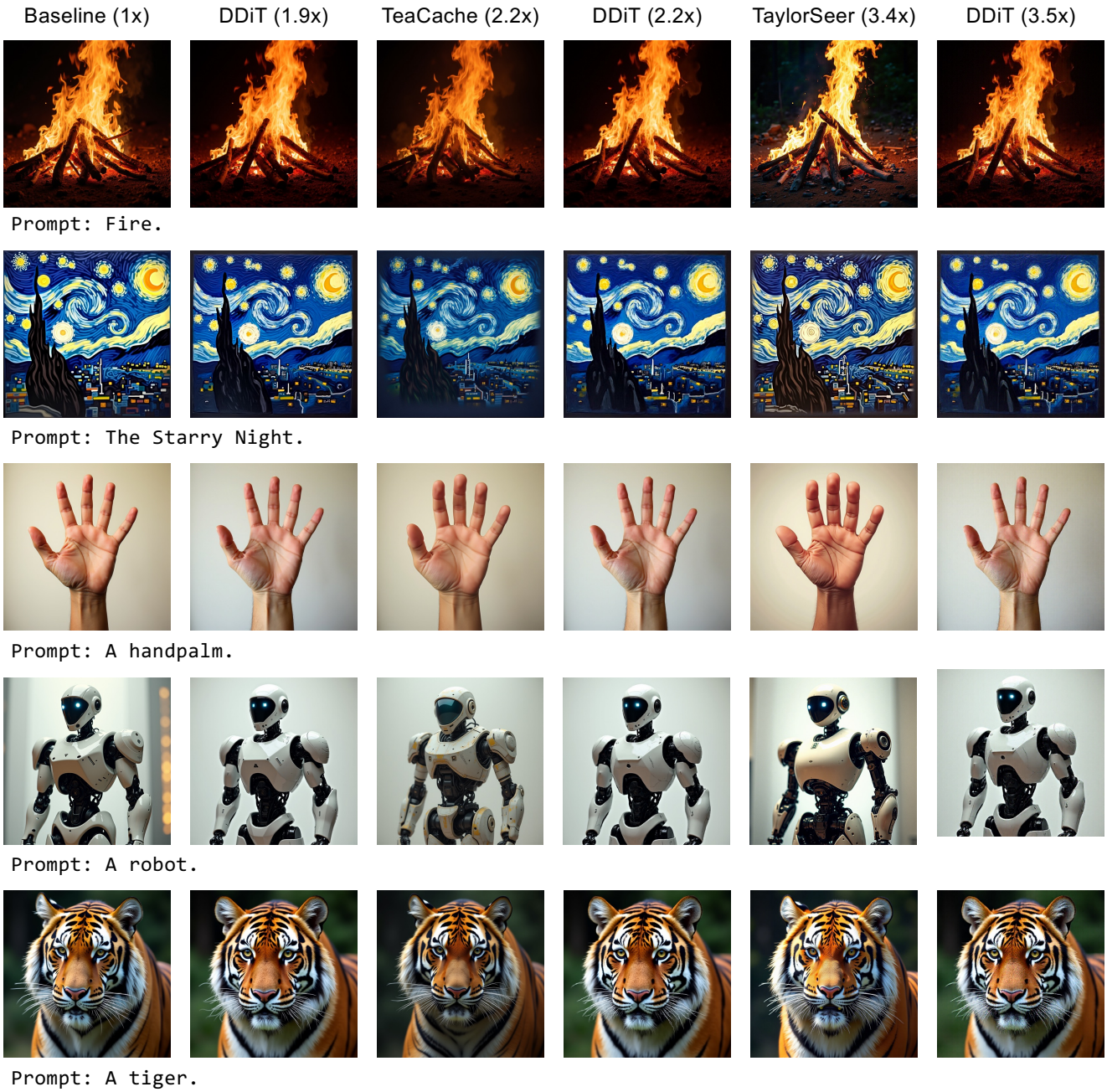

Figure 13. Qualitative comparisons with the base model [56], TeaCache [63], TaylorSeer [64], and DDiT on PartiPrompts [126]. **DDiT** effectively preserves fine-grained details, pose, spatial layout, and overall color distribution of the generated images (1/5). The number next to the method name indicates the amount of computational speedup. Notice how for the same speedup, e.g., 2.2×, TeaCache [63] loses fine-grained texture and color distribution in the prompt *The Starry Night*, or produces missing eyes in the prompt *A robot*. Similar observations hold true for TaylorSeer [64], where the identity and color of the robot are not preserved in the prompt *a robot*.

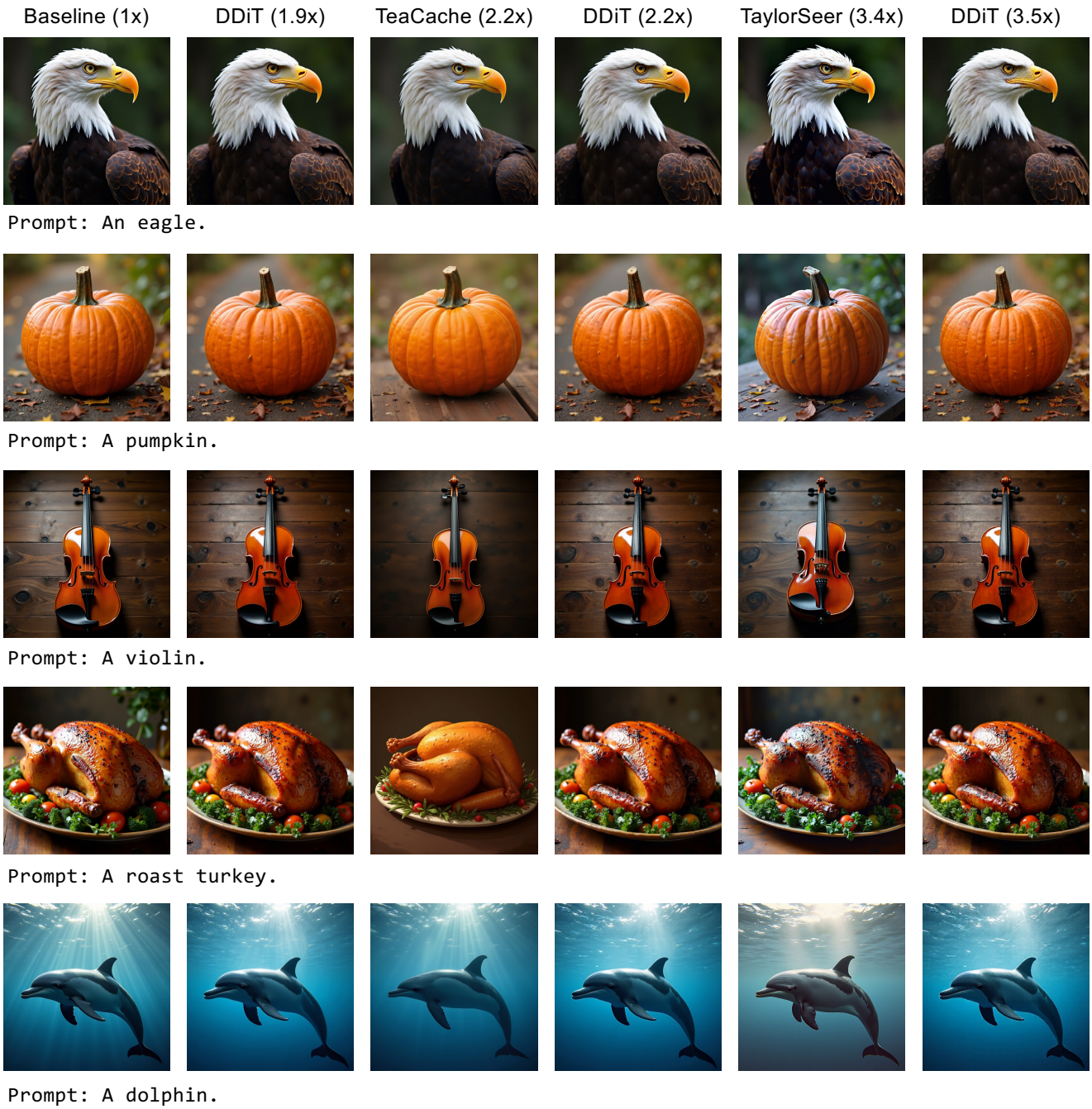

Figure 14. Qualitative comparisons on PartiPrompts (2/5). The number next to the method name indicates the amount of computational speedup. Notice how for the same speedup, e.g., 2.2×, TeaCache [63] loses the fine-grained texture in the prompt *A roast turkey*. Similar observations hold true for TaylorSeer [64], where the overall color distribution and the background are not preserved in the prompts *A pumpkin* and *a dolphin*.

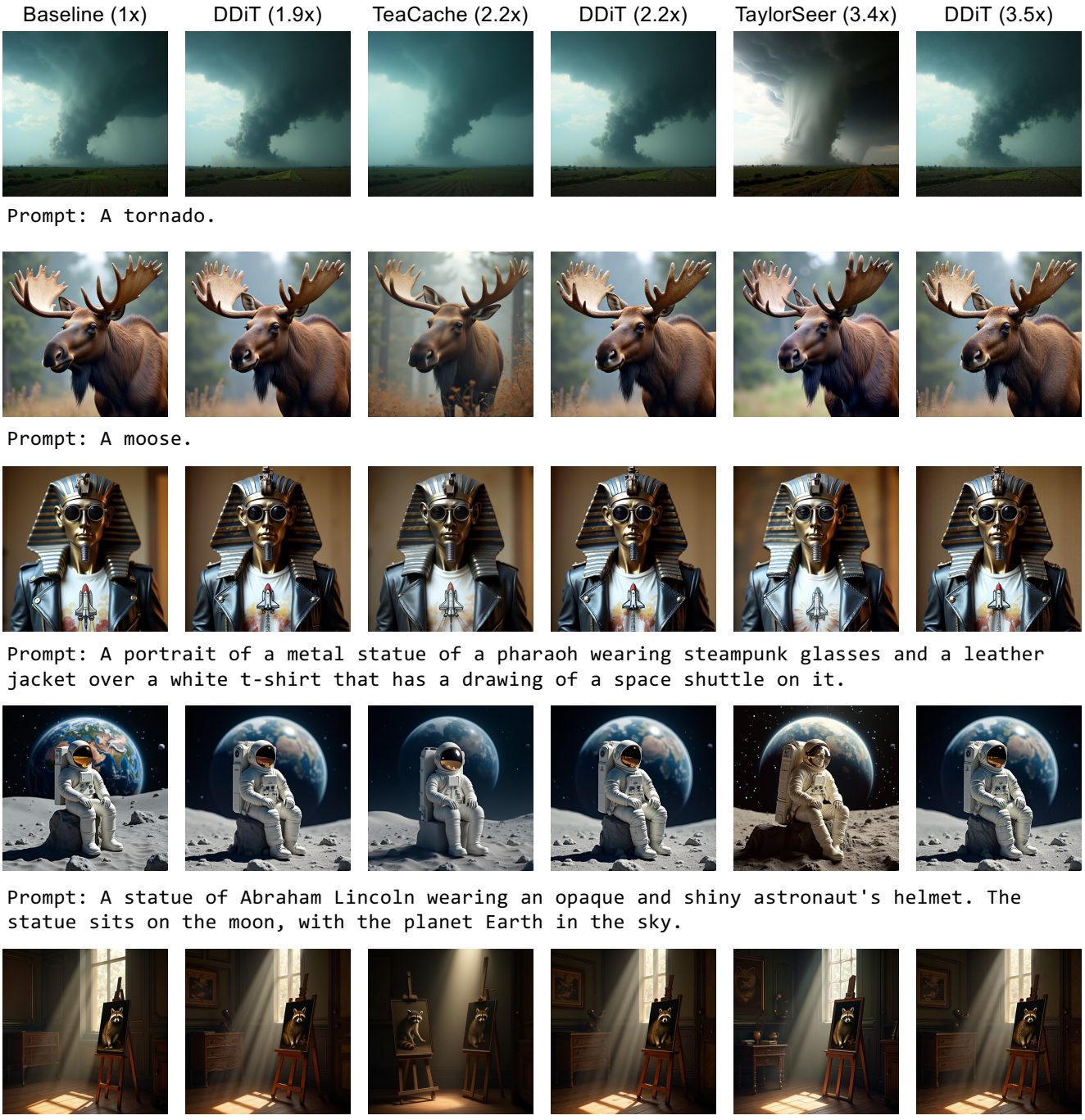

Figure 15. Qualitative comparisons on PartiPrompts (3/5). The number next to the method name indicates the amount of computational speedup. Notice how for the same speedup, e.g., 2.2×, TeaCache [63] fails to preserve the position of the moose in the prompt *A moose*, or produces missing and duplicated objects (the window and the raccoon, respectively) in the last prompt describing a single room with a raccoon painting on an easel. Similar observations hold true for TaylorSeer [64], where the overall color distribution and the fine-grained texture details are not preserved in the prompt *A tornado*.

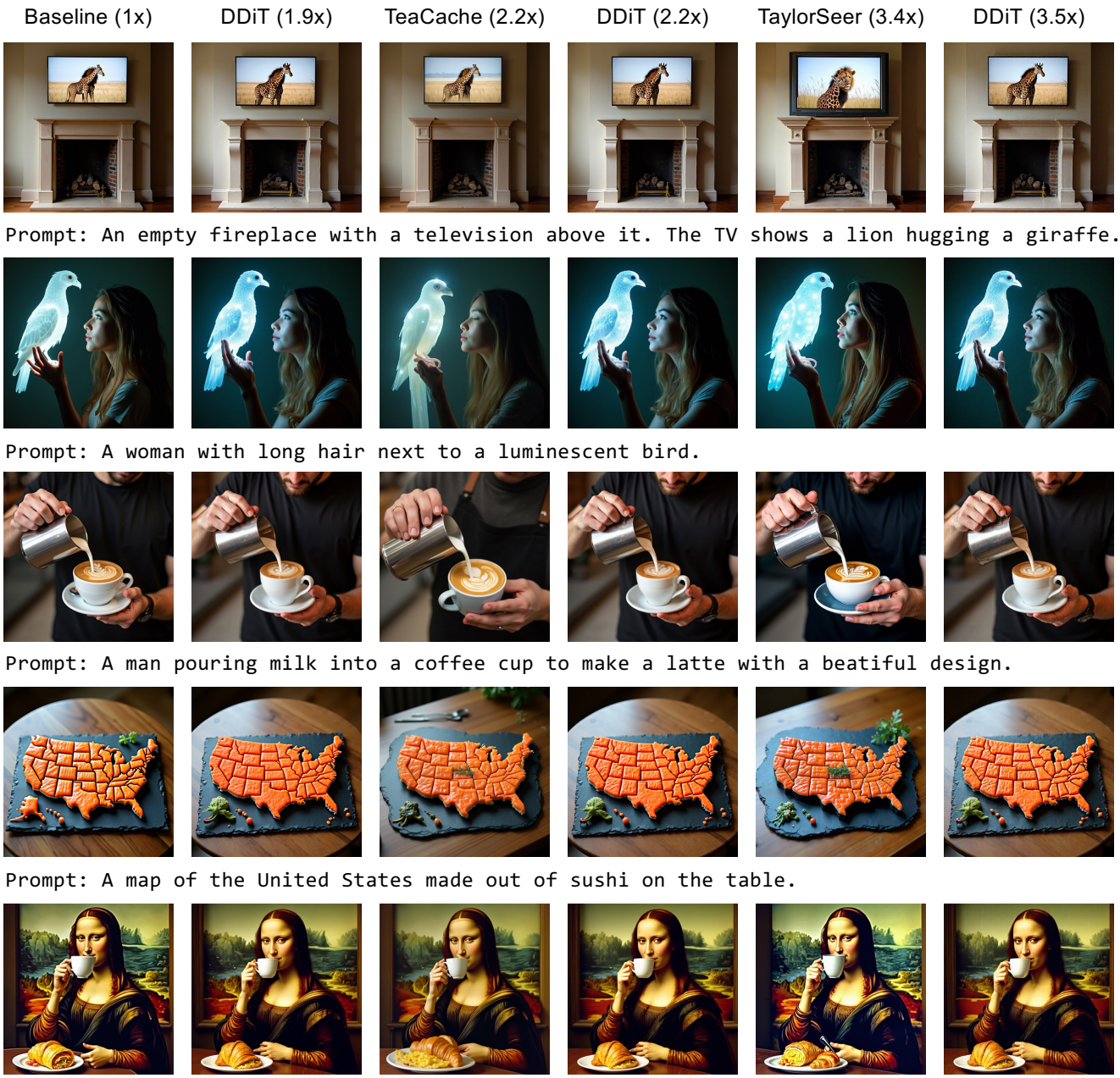

Figure 16. Qualitative comparisons on PartiPrompts (4/5). The number next to the method name indicates the amount of computational speedup. Notice how for the same speedup, e.g., 2.2×, TeaCache [63] loses the hand details and the facial expression of the girl in the prompt *A woman with long hair next to a luminescent bird*, or the cup saucer is missing in the prompt *A man pouring milk into a coffee cup to make a latte with a beautiful design*. Similar observations hold true for TaylorSeer [64], where the position of the giraffe and the lion is not preserved in the prompt *An empty fireplace with a television above it. The TV shows a lion hugging a giraffe.*

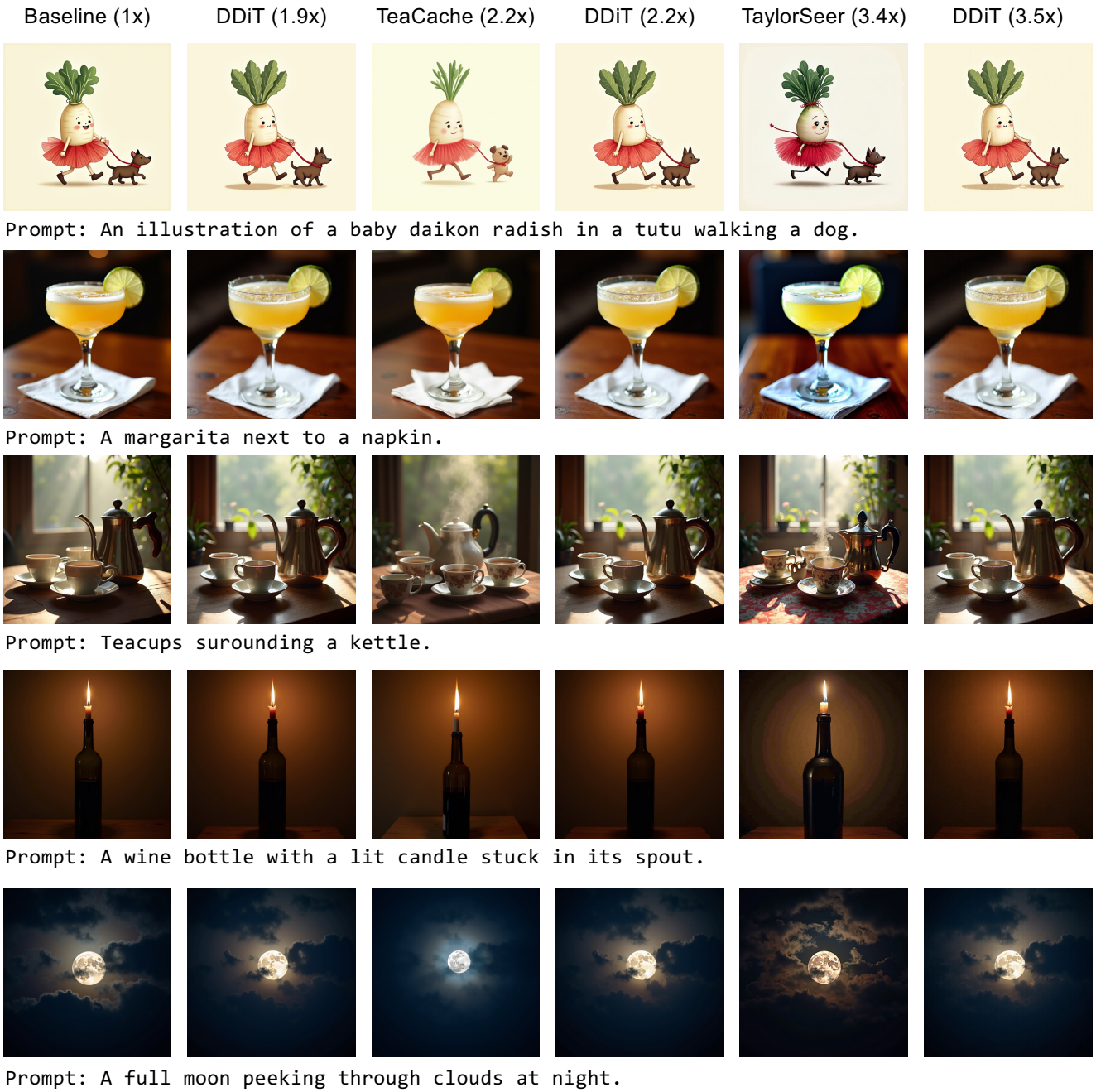

Figure 17. Qualitative comparisons on PartiPrompts (5/5). The number next to the method name indicates the amount of computational speedup. Notice how for the same speedup, e.g., 2.2×, TeaCache [63] and TaylorSeer [64] lose the identity of the dog or the radish character in the prompt *An illustration of a baby daikon radish in a tutu walking a dog*.

7. Additional Analysis

In this section, we provide additional analysis on DDiT.

7.1. Effect of Dynamic Path Scheduling.

Table 5. **Effect of DDiT's scheduling strategy on generation quality.** Using a manually defined schedule leads to a noticeable decrease in image quality.

Method	FID↓	CLIP↑	ImageReward↑
Manual schedule	35.71	0.2817	0.9683
DDiT	**33.42**	**0.3136**	**1.0284**

We further evaluate the effectiveness of our dynamic scheduling strategy for selecting patch sizes during generation. Motivated by prior studies showing that early timesteps generate coarse structures while later ones refine fine details [9, 52, 75, 82, 105, 108, 113], we design a fixed patch-size schedule as a controlled baseline. Specifically, we begin with a coarse patch size ($4p$) for T_{4p} steps, then switch to a medium size ($2p$) for T_{2p} steps, and finally use the finest patch size (p) for the remaining $T - (T_{4p} + T_{2p})$ steps. We compare this manual schedule against DDiT (τ=0.001, 2.18× speedup). For a fair comparison, we set T_{4p} and T_{2p} to achieve the similar overall speedup, with T_{4p}=16, T_{2p}=16, and $T = 50$. As shown in Table 5, dynamically determined schedule yields consistently better performance than the manually defined one across all evaluation metrics, including FID [35], CLIP [34, 85], and ImageReward [121] scores. Figure 18 shows the corresponding generations of 5 prompts randomly selected. Notice that the manual schedule introduces substantial degradation in visual fidelity relative to the baseline. It frequently fails to preserve fine-grained details, exhibits weaker prompt alignment, and produces several notable errors, *e.g.*, generating fewer than five dogs, misinterpreting the description of a boat, or incorrectly capturing spatial relationships between objects. In contrast, the dynamic schedule maintains high-quality generation throughout. This demonstrates that **adaptively adjusting the patch size at each timestep not only enables the model to allocate computational resources more effectively, but also leads to higher visual fidelity and improved text–image alignment.** Overall, these results confirm the importance of our scheduling mechanism in maintaining generation quality while achieving significant efficiency gains.

7.2. Effect of the Threshold τ

In Sec. 4.4, we quantitatively analyzed the effect of varying the threshold τ in our patch scheduling mechanism, which determines when to transition between coarse and fine patch sizes during denoising. In addition to this, we qualitatively studied the impact of the threshold τ. As shown in Fig. 19, increasing τ leads to a slightly lower visual quality. This degradation occurs because higher thresholds make the scheduler less sensitive to temporally local variations in the latent manifold, causing premature selection of coarse patches and consequently suppressing fine-grained detail. To balance visual quality with computational efficiency, we use $\tau = 0.001$ in all experiments.

7.3. Effect of the Quantile ρ

We study the effect of varying the quantile parameter $\rho \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$, as well as replacing ρ with mean aggregation of $\boldsymbol{\sigma}_{t-1}^{p_i}$. As shown in Fig. 20, we find that $\rho = 0.4$ consistently yields the best generation quality across different prompts. Values that are too small (*e.g.*, $\rho = 0.1$) tend to be overly sensitive to outliers in the spatial variance distribution, while larger values (*e.g.*, $\rho = 0.7$) over-smooth the signal, leading to suboptimal patch size decisions. Based on these findings, we set $\rho = 0.4$ in all experiments throughout the paper.

7.4. Failure Cases

Figure 21 shows failure cases of our method. The model struggles with highly intricate scenarios such as contradictory prompts or complex object counting, which are also difficult for the base model.

7.5. Effect of Residual Block

We investigate the role of the residual connection in our architecture. As described in Sec. 3.2 and illustrated in Fig. 3, let *Blocks* denote the transformer blocks in DiTs, $\mathcal{F}_{\text{res}}$ denote the single-layer residual block, and $\mathbf{z}_t$ the latent representation at

Figure 18. **Qualitative comparison with the manual schedule with DDiT.** DDiT preserves structural similarity to the baseline, maintaining pose, spatial layout, and overall color distribution of the generated images.

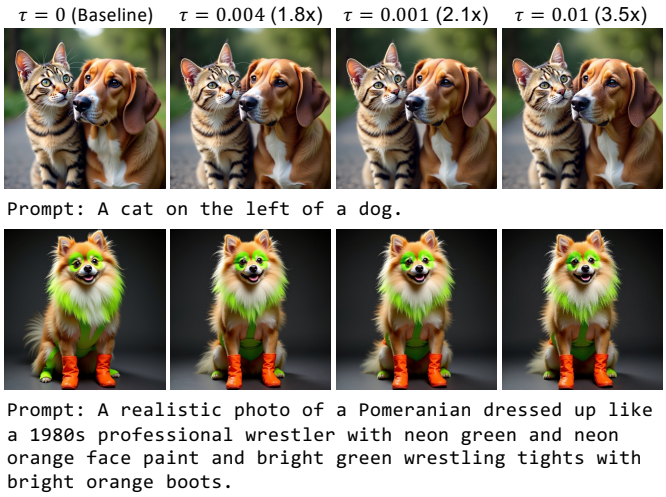

Figure 19. **Effect of different threshold values τ in DDiT.** As τ increases, image quality degrades slightly, while inference speed improves.

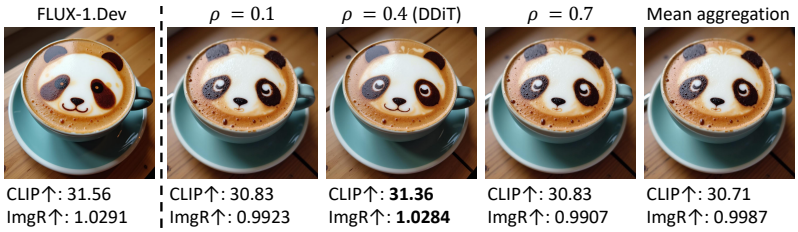

Figure 20. **Ablation of ρ in DDiT.** Using $\rho = 0.4$ consistently yields better generation quality for diverse prompts compared to other aggregation methods. We report CLIP [34, 85], and ImageReward [121] scores.

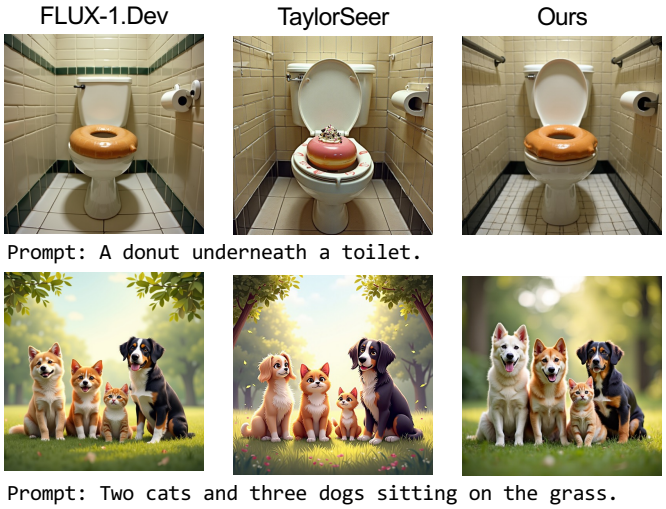

Figure 21. **Failure cases.** Our method struggles in scenarios where the base model also fails, such as contradictory prompts and complex object counting.

timestep t. The residual formulation is given by:

$$\mathbf{z}_{t-1} = \mathit{Blocks}(\mathbf{z}_t) + \mathcal{F}_{\text{res}}(\mathbf{z}_t). \tag{6}$$

To assess the impact of the residual block, we train the network using patch embedding, de-embedding layers, and LoRA,

Figure 22. **Effect of residual block in DDiT.** Removing the residual block results in noticeable image degradation and loss of detail.

Table 6. **Quantitative comparison of T2I generation performance.** We report FID on COCO, and CLIP score and ImageReward on DrawBench.

Method	Steps	FLOPs(T)	Speed↑	FID↓	CLIP↑	ImgR↑
FLUX-1.Dev	50	2990.96	1.0×	33.07	0.3156	1.0291
FLUX-1.Dev	28	1681.44	1.78×	33.35	0.3140	1.0107
FLUX-1.Dev	15	907.63	3.3×	34.02	0.3121	0.9865
RALU [46]	(4×)	723.69	4.13×	35.17	0.2994	0.9523
DDiT	50	1358.73	2.2×	**33.42**	**0.3136**	**1.0284**
DDiT	28	877.11	3.41×	33.68	0.3120	1.0177
DDiT	15	530.04	5.64×	34.29	0.3085	0.9937

Table 7. **Quantitative comparison** with FLUX.1-lite and DyFLUX.

Method	Steps	FLOPs(T)	Speed↑	FID↓	CLIP↑	ImgR↑
FLUX.1-lite [18]	28	1359.85	1.0×	34.12	0.3129	1.0162
DyFLUX [135]	28	869.49	1.56×	35.38	0.2961	0.9728
DDiT	28	715.71	1.9×	**34.72**	**0.3101**	**1.0141**

but without the residual block:

$$\mathbf{z}_{t-1} = \mathit{Blocks}(\mathbf{z}_t). \tag{7}$$

As shown in Fig. 22, removing the residual block results in clear degradation of image quality, including the loss of important fine-grained details. Based on these observations, we retain the residual block in DDiT.

7.6. Comparison with Additional Baselines

In the main paper, we primarily compared DDiT against state-of-the-art caching-based acceleration methods. Here, we additionally compare with RALU [46], a recent spatial acceleration method designed for transformer-based diffusion models that performs mixed-resolution sampling. As shown in Table 6, DDiT achieves consistently better performance across all evaluation metrics. We also evaluate DDiT under reduced inference steps to assess its robustness. DDiT continues to remain robust and more efficient than FLUX-1.Dev even with fewer sampling steps, confirming that our dynamic patch scheduling generalizes well across different inference budgets.

We further compare DDiT with DyFLUX [135], a recent method that leverages knowledge distillation for efficient generation of diffusion transformers. As shown in Table 7, DDiT achieves faster inference speed on FLUX.1-lite [18] while maintaining higher visual quality and better prompt alignment, demonstrating the effectiveness of our dynamic patch scheduling strategy.

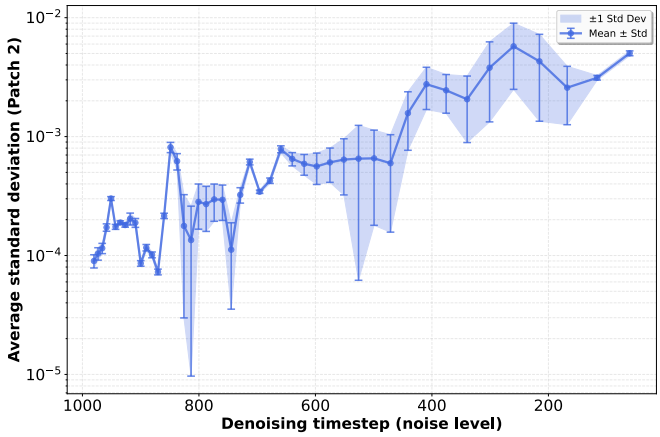

Figure 23. **Average and standard deviation (shown as error bars) of $\boldsymbol{\sigma}_{t-1}^{2p}$ (Eqn. 5) across 200 DrawBench prompts (log scale).** The error bars represent variation across prompts. In general, the standard deviation increases at later timesteps. However, the fluctuations across timesteps and the wide spread across prompts suggest that each prompt exhibits a distinct level of detail and complexity. This observation motivates the use of different acceleration rates $\sigma_{t-1}^{p_i,(\rho)}$.

8. Details on Spatial Variance Estimation

As illustrated in Fig. 5, we compute patch-wise standard deviations $\boldsymbol{\sigma}_{t-1}^{p_i}$ using the third-order latent difference. Let $\Omega_1, \Omega_2, \ldots, \Omega_{N_{new}}$ denote the set of non-overlapping spatial regions, where each Ω_n corresponds to a patch of size $p_i \times p_i$. For each candidate patch size $p_i \in p, p_{new}$, we calculate the standard deviation of the third-order latent difference $\Delta^{(3)}\mathbf{z}_{t-1} \in \mathbb{R}^{H\times W\times C}$ within the patch region of size $p_i \times p_i$:

$$\boldsymbol{\sigma}_{t-1}^{p_i} = \mathrm{Std}_{m\in\Omega_n}\left(\Delta^{(3)}\mathbf{z}_{t-1}(m)\right), \quad n = 1, \ldots, N_{new}, \tag{8}$$

where m indexes spatial positions inside each patch (*i.e.*, the (x, y) pixel coordinates within the region), and $\boldsymbol{\sigma}_{t-1}^{p_i} \in \mathbb{R}^{H/p\times W/p\times C}$ contains the per-patch standard deviations. Intuitively, $\boldsymbol{\sigma}_{t-1}^{p_i}$ measures how strongly the denoising trajectory varies within each spatial region. Higher variance indicates greater local change along the trajectory, signaling the need for smaller patches to capture fine-grained details; lower variance suggests that coarser patches are sufficient. As shown in Fig. 23, prompts with different levels of granularity exhibit distinct variance profiles across timesteps. This observation motivates using different acceleration rates, $\sigma_{t-1}^{p_i,(\rho)}$.

9. Implementation Details

In this section, we provide additional implementation details.

Training details. The T2I model is finetuned on the T2I-2M dataset [45], a synthetic dataset generated using the base model, and the T2V model is trained on synthetic videos generated by the base model using prompts from the Vchitect-T2V-Dataverse [24]. Empirically, we find that training on synthetic data generated by the same base model significantly stabilizes distillation, reduces training time, and leads to higher visual fidelity. We hypothesize that this is because the predictions made on such synthetic data are more accurate and consistent with the base model's learned behavior compared to using real datasets or synthetic data from other sources. Ideally, using the exact dataset on which the base model was originally trained would be most effective for preserving its learned distribution; however, since this data is not publicly available, we find that generating synthetic data from the base model itself serves as a strong proxy, leading to more effective knowledge transfer during distillation. We train both the T2I and T2V tasks for 100,000 steps using $bf16$ mixed-precision.

Patch scheduling details. DDiT employs dynamic patch scheduling based on the third-order latent difference. Computing this quantity requires the first three latent values to determine the subsequent acceleration rate. We initialize these first three latent states using the largest patch size, *i.e.*, $\max(p_i)$ where $p_i \in p_{\text{new}}$. This choice is motivated by the observations in

Fig. 6 and Fig. 23, which show that during the initial denoising steps, $\boldsymbol{\sigma}_{t-1}^{2p,(\rho)}$ remains consistently low. We set $\rho = 0.4$ following Sec. 4, as values that are too small or too large tend to be overly sensitive to outliers. Empirically, we find that this initialization strategy yields better image quality.

Baseline details. We compare our method against two state-of-the-art caching-based acceleration approaches: TeaCache [63] and TaylorSeer [64]. For both the T2I and T2V tasks, we follow the official implementations provided in their public repositories.[†] [‡] To measure latency, we report the average inference speedup achieved by each method.

10. Details of the User Study

```
You will see two images displayed side by side. Your task is to choose which image you
    prefer (left or right) based on visual quality. In this study, "visual quality" refers
    to how realistic, clear, and visually pleasing an image appears. If you cannot
    determine which image is better, you may select "Can't decide."
```

Table 8. User study instructions.

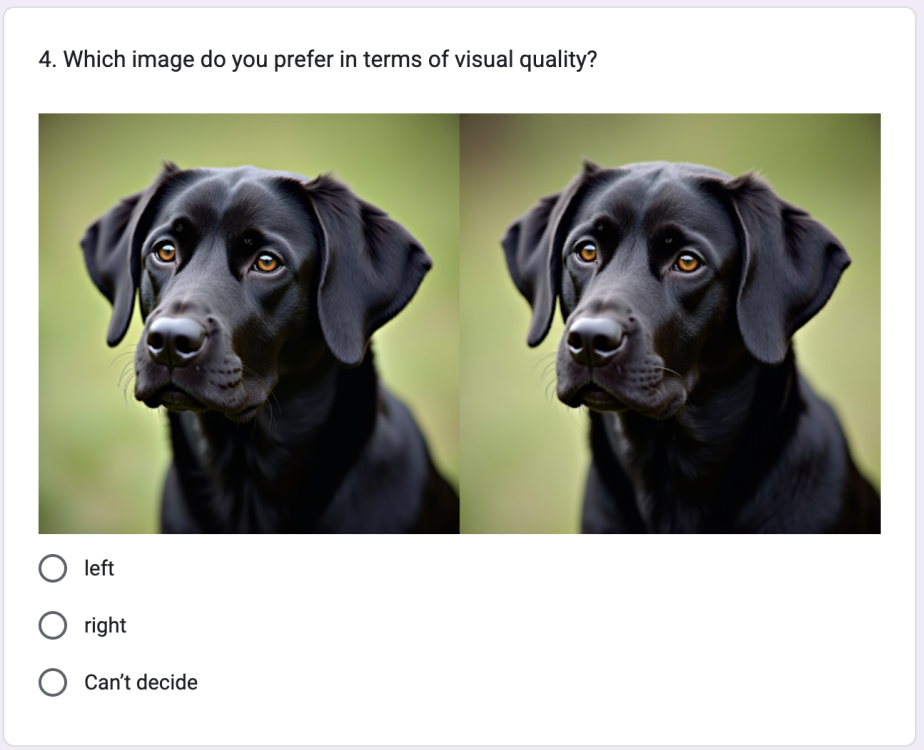

Figure 24. User study interface.

In this section, we provide additional details of our user study, whose results are reported in Sec. 4.4. Specifically, we generate images using DDiT and the baseline for the first 50 prompts from the DrawBench dataset. Each pair of generated images was randomly arranged side by side, and participants were asked to select the one with higher overall visual quality. The complete instructions provided to participants are shown in Table 8, and the user interface is illustrated in Fig. 24. Three participants with diverse backgrounds took part in the study.

[†]TeaCache: https://github.com/ali-vilab/TeaCache
[‡]TaylorSeer: https://github.com/Shenyi-Z/TaylorSeer