Accelerating Diffusion via Hybrid Data-Pipeline Parallelism Based on Conditional Guidance Scheduling

Supplementary Material

Methods	Speed-Up ↑	Image Quality ↑	Model General. ↑	High-res Synth. ↑	Comm. Efficiency ↑
Distrifusion	2.5	3.5	2.5	3.3	5.0
AsyncDiff	3.0	4.5	5.0	3.5	1.0
Ours	**4.7**	**4.5**	**5.0**	**4.4**	**5.0**

Table 3. **Quantitative metrics comparison across five evaluation aspects.** Scores are normalized to a 5-point scale. Higher values (↑) indicate better performance.

A. Evaluation of Hybrid Parallelism

Evaluation Protocol. All scores are computed based on a 5-point scale unified min-max scaling scheme, where the normalized values are re-centered around an average score of 3. Table 3 summarizes the resulting normalized scores across the five evaluation aspects. Specifically, each metric is assessed as follows:

- **Speed-Up.** We measure the relative acceleration ratio with respect to the SDXL baseline latency in Table 1. The measured latencies are 13.53secs for DistriFusion, 12.54secs for AsyncDiff, and 7.12secs for our method.
- **Image Quality.** We evaluate image quality using FID scores reported in Table 1 from the main results of SDXL. The reported FID values are 4.864 for DistriFusion, 4.103 for AsyncDiff, and 4.100 for our method.
- **Model Generality.** We assign scores based on architecture compatibility. Each model receives 2.5 points for supporting U-Net and an additional 2.5 points for DiT support, resulting in scores of 2.5 for DistriFusion, 5 for AsyncDiff, and 5 for Ours.
- **High-resolution Synthesis.** The score reflects both high-resolution generation capability and inference latency. According to the results in Section 5.4 High-Resolution Generation, all three methods successfully generate three target resolutions. The corresponding average latencies are 14.73secs for DistriFusion, 14.27secs for AsyncDiff, and 11.99secs for Ours.
- **Communication Efficiency.** We evaluate the communication efficiency based on the measured inter-GPU data transfer communication volume in the SDXL multi-GPU setting reported in Table 1 from the main results. The measured communication volumes are 0.525 GB for DistriFusion, 9.830 GB for AsyncDiff, and 0.516 GB for our method.

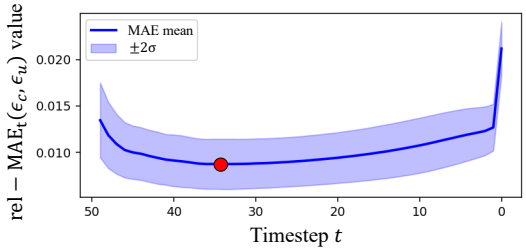


Figure 8. Empirical visualization of denoising discrepancy curve.

B. Empirical Visualization of Denoising Discrepancy

Figure 8 illustrates the average denoising discrepancy (rel-MAE$_t(\epsilon_c, \epsilon_u)$) value measured during the denoising process based on 5,000 prompts from the MS-COCO 2014 [15] validation set. The shaded region represents the $\pm 2\sigma$ range, and the denoising model used is Stable Diffusion XL. The red dot denotes $\tau_{\text{cap}} = \operatorname{argmin}_t \text{rel-MAE}_t(\epsilon_c, \epsilon_u)$, which is employed as a safety-cap in the main method.

C. Adaptive Parallelism Switching Algorithm

Algorithm 1 Adaptive Parallelism Switching via Denoising Discrepancy

Require: latent noise $\mathbf{x}_t$, prompt c, steps T, window L, slope threshold g, safety-cap τ_{cap}, interval k

$\tau_1, \tau_2 \leftarrow \varnothing$
for $t = T, T-1, \ldots, 1$ **do**
  $\epsilon_c, \epsilon_u \leftarrow \epsilon_\theta(\mathbf{x}_t, c, t),\ \epsilon_\theta(\mathbf{x}_t, t)$
  $M_t \leftarrow \dfrac{\mathbb{E}_{x,\epsilon}\|\epsilon_c - \epsilon_u\|_1}{\mathbb{E}_{x,\epsilon}\|\epsilon_u\|_1}$ ▷ rel-MAE$_t(\epsilon_c, \epsilon_u)$
  $G_t = \frac{M_t - M_{t-L}}{L}$
  if $\tau_1 = \varnothing$ **and** $t > L$ **and** $0 \le G_t < g$ **then**
    $\tau_1 \leftarrow \min(t, \tau_{\text{cap}})$; $\tau_2 \leftarrow \tau_1 + k$
  Denoise:
  if $t \ge \tau_1$ **then**
    WARM-UP
  else if $t > \tau_2$ **then**
    PARALLELISM
  else
    FULLY-CONNECTING
  end if
  $x_{t-1} \leftarrow$ STEP DENOISE$(\mathbf{x}_t, \epsilon_c, \epsilon_u, t)$
end for
return $x_0,\ (\tau_1, \tau_2)$

D. Derivation of Score-Based Interpretation of Denoising Discrepancy

The denoising discrepancy(rel-MAE$_t(\epsilon_c, \epsilon_u)$) criterion in Eq. (4) can be theoretically derived from the score decomposition of diffusion models. Following the ϵ-parameterization of score-based generative modeling [8, 32], the preconditioned score can be expressed as

$$s_\theta(\mathbf{x}_t, t) \approx -\frac{\epsilon_\theta(\mathbf{x}_t, t)}{\sigma_t}, \tag{5}$$

where σ_t denotes the noise standard deviation at timestep t. According to Bayes' rule, the conditional score function can be decomposed as

$$s_c(\mathbf{x}_t, t) = s_u(\mathbf{x}_t, t) + \nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t), \tag{6}$$

where $s_u(\mathbf{x}_t, t)$ is the unconditional data score, and $\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)$ denotes the conditional information flow [7]. Substituting Eq. ((5)) into Eq. ((6)) yields

$$\epsilon_c(\mathbf{x}_t, t) - \epsilon_u(\mathbf{x}_t, t) \propto \sigma_t \nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t), \tag{7}$$

which implies that the difference between conditional and unconditional denoiser outputs corresponds to the conditional gradient scaled by σ_t. Therefore, the rel-MAE at each timestep t can be approximated as

$$\text{rel-MAE}_t = \frac{\|\epsilon_c - \epsilon_u\|_1}{\|\epsilon_u\|_1} \approx \frac{\|\nabla_{\mathbf{x}_t} \log p(c|\mathbf{x}_t)\|_1}{\|s_u(\mathbf{x}_t, t)\|_1}. \tag{8}$$

This formulation reveals that rel-MAE$_t(\epsilon_c, \epsilon_u)$ quantifies the relative magnitude between the conditional information and the unconditional data prior—forming the theoretical basis for the main method equation (Eq. (4)).

E. Robustness of Determine τ_1 under Stochastic Denoising Noise

Diffusion inference is a stochastic denoising process; predicted noises $\epsilon_\theta(\mathbf{x}_t)$ are subject to random sampling. Consequently, the observed $\{M_t\}$ fluctuates slightly, and $G_t \approx 0$ may appear prematurely. To ensure robust detection, we define a finite-difference slope by

$$G_t = \frac{M_t - M_{t-L}}{L}, \tag{9}$$

which smooths out stochastic perturbations across L timesteps. The stability of G_t can be theoretically justified by Hoeffding's inequality:

$$\Pr(|G_t - \mathbb{E}[G_t]| > \delta) \leq 2\exp\Big(-\frac{2L\delta^2}{(b-a)^2}\Big). \tag{10}$$

Here, L denotes the window length used to compute the moving-average slope, δ represents the allowable deviation from the expected slope $\mathbb{E}[G_t]$, and a, b correspond to the minimum and maximum possible range of the observed rel-MAE$_t(\epsilon_c, \epsilon_u)$ values, typically normalized within $[0, 1]$.

As L increases, the variance of the estimated slope decreases, and the probability of false detection decreases exponentially. showing that larger L exponentially reduces false-alarm probability.

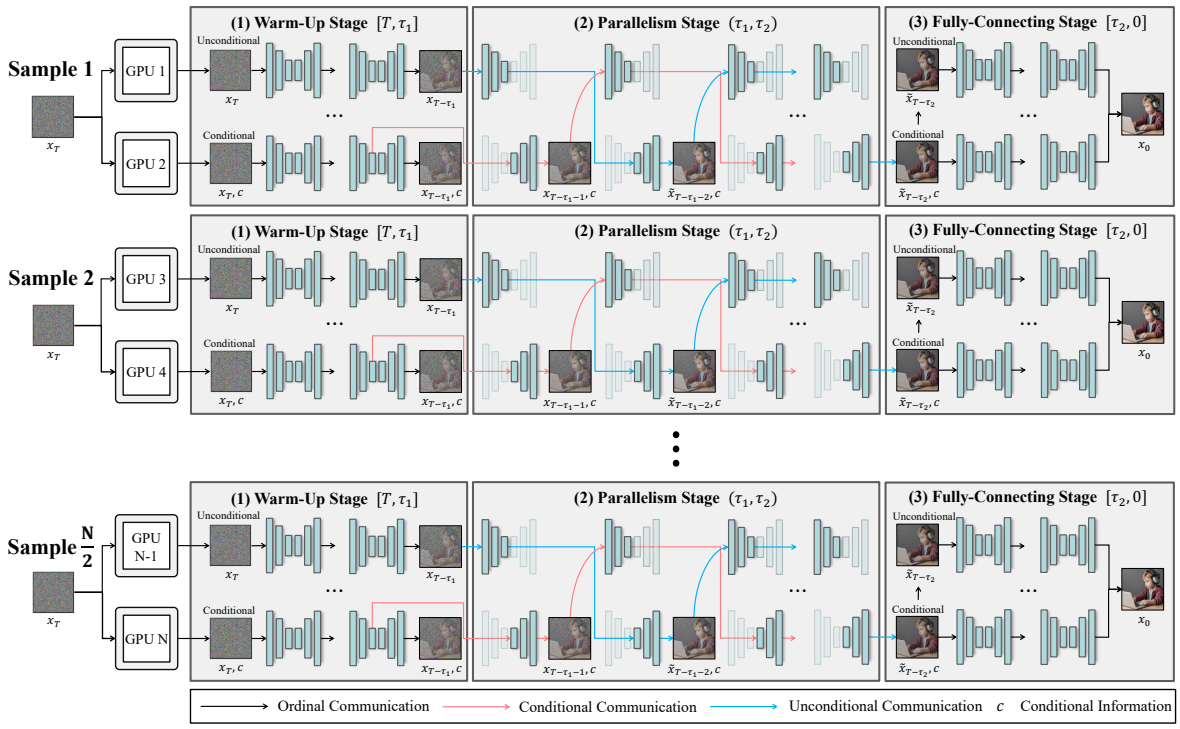
Empirically, L and g_{slope}, which are also established in our experiments, lie within a stable regime due to strong autocorrelation of rel-MAE$_t(\epsilon_c, \epsilon_u)$ sequences. Thus, τ_1 can be reliably detected as the earliest timestep satisfying $0 \leq G_t < g_{\text{slope}}$ and $t \leq \tau_{\text{cap}}$.

F. Extensibility to Many GPU Configurations Structures

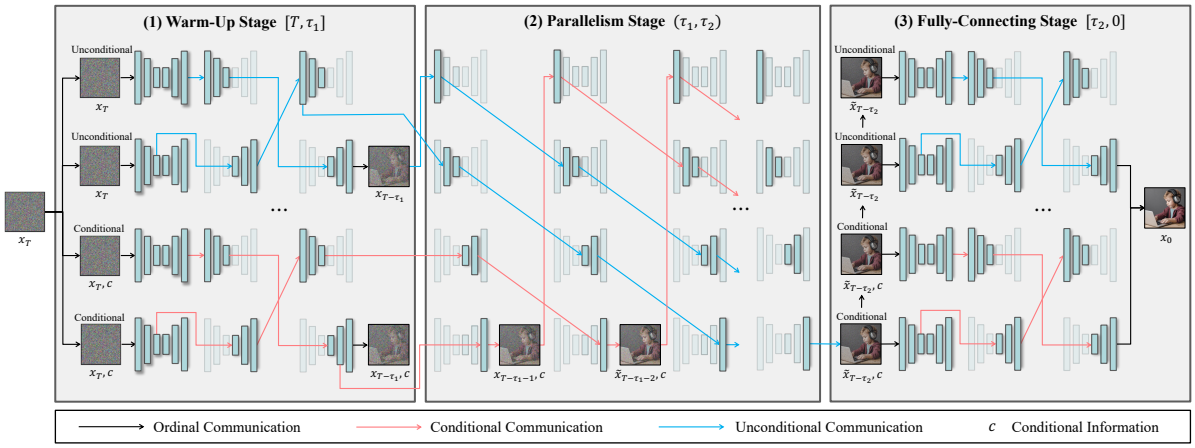
Figure 9 presents two extensibility structures that scale the proposed hybrid parallelism framework from the baseline 2 GPUs setup to many GPU configurations.

The first structure, shown in Figure 9a, demonstrates the batch-level extension under an N GPU configuration. In this scheme, each pair of GPUs collaboratively denoises a single sample while following the three stages hybrid parallelism framework. As a result, the system can generate $N/2$ samples concurrently with N GPUs, enabling near-linear throughput scaling when multiple samples are produced.

The second structure, shown in Figure 9b, demonstrates the layer-wise pipeline extension on a 4 GPU configuration. Here, the denoising network is partitioned into multiple layer-wise segments distributed across devices, allowing the hybrid parallelism strategy to be applied to single-sample generation. While this configuration may exhibit slightly reduced acceleration efficiency and minor quality degradation compared to the batch-level extension, it provides a fine-grained pipeline scheduling mechanism. Importantly, the same structural principles naturally generalize beyond the 4 GPUs example to arbitrary N GPU configurations, demonstrating the flexibility and scalability of the proposed framework.



(a) Batch-level extension under N GPU configuration.



(b) Layer-wise pipeline extension on a 4 GPU configuration.

Figure 9. **Extensibility to many GPU configurations structures.** This figure illustrates two strategies for scaling the proposed hybrid parallelism framework to larger GPU configurations. These structures demonstrate how the proposed framework naturally generalizes from the 2 GPUs setting to both batch-level and layer-wise many GPU configurations.

Base Model	Devices	Methods	Latency (s) ↓	Speed-Up ↑	Comm. (GB) ↓	FID ↓		LPIPS ↓		PSNR ↑	
						w/ G. T.	w/ Orig.	w/ G. T.	w/ Orig.	w/ G. T.	w/ Orig.
Stable Diffusion XL	1	Original Model	16.49	-	-	23.977	-	0.797	-	9.618	-
	2	DistriFusion [12]	13.53	1.22×	0.525	24.164	4.864	0.7978	0.146	9.597	24.634
		AsyncDiff [2] (stride=1)	12.54	1.31×	9.830	23.941	4.103	0.797	0.108	9.586	26.387
		Ours (k=5)	**7.12**	**2.31**×	**0.516**	**23.831**	**4.100**	**0.796**	**0.107**	**9.665**	**26.640**
	4	DistriFusion	7.13	2.31×	**0.486**	24.197	5.772	0.798	0.183	9.578	23.046
		AsyncDiff (stride=1)	9.45	1.74×	10.531	24.225	5.829	0.795	0.147	9.626	24.776
		AsyncDiff (stride=2)	6.57	2.51×	4.795	24.140	6.572	0.803	0.192	9.557	23.070
		Ours (k=5)	**4.83**	**3.41**×	0.751	**24.087**	**5.544**	**0.788**	**0.113**	**9.888**	**25.233**
	8	DistriFusion	5.86	2.81×	**0.424**	24.423	6.457	**0.799**	0.218	9.537	22.042
		Ours (k=5)	**3.97**	**4.15**×	0.883	**24.410**	**6.267**	0.808	**0.184**	**9.639**	**23.670**
Stable Diffusion 3	1	Original Model	19.36	-	-	33.433	-	0.810	-	8.086	-
	2	AsyncDiff [2] (stride=1)	9.82	1.97×	1.290	33.379	2.032	0.813	0.052	8.155	27.812
		xDiT-Ring [4]	14.31	1.35×	121.646	33.356	1.909	0.809	0.047	8.085	27.857
		Parastep [33]	9.98	1.94×	**0.032**	33.340	3.350	0.810	0.112	8.091	22.917
		Compact-2bit [21]	13.96	1.39×	9.049	33.560	4.712	0.810	0.196	7.999	19.242
		Ours (k=5)	**9.33**	**2.07**×	0.189	**33.322**	**1.878**	**0.780**	**0.046**	**8.229**	**27.875**
	4	AsyncDiff (stride=1)	6.51	2.97×	3.774	33.909	3.458	0.887	0.177	8.033	26.540
		AsyncDiff (stride=2)	**4.86**	**3.98**×	1.241	34.434	8.939	0.895	0.272	7.956	21.087
		xDiT-Ring	17.96	1.08×	130.792	33.374	2.232	0.861	0.127	8.001	27.101
		Parastep	6.29	3.08×	**0.038**	33.211	2.581	0.910	0.134	7.990	27.089
		Compact-2bit	9.25	2.09×	13.593	33.420	5.461	**0.811**	0.248	7.926	17.490
		Ours (k=5)	5.53	3.50×	0.572	**33.113**	**2.109**	0.857	**0.122**	**8.046**	**27.110**
	8	xDiT-Ring	21.04	0.92×	149.085	34.191	3.478	0.911	**0.198**	7.714	25.109
		Parastep	5.75	3.37×	**0.041**	**33.889**	3.596	0.889	0.223	**7.889**	24.924
		Ours (k=5)	**4.73**	**4.09**×	1.013	34.099	**3.456**	**0.872**	0.201	7.787	**25.414**

Table 4. **Additional quantitative comparison of parallelism methods on the Stable Diffusion XL and Stable Diffusion 3 models.** We compare our method with existing distributed inference techniques under 1, 2, 4, and 8 GPUs. We report both the baseline latency and the corresponding acceleration ratio (*Speed-Up*), Communication efficiency *(Comm.)*, and quantitative metrics assessing generation fidelity. Here, *w/ G.T.* denotes comparison with the ground-truth image, and *w/ Orig.* indicates comparison with the original (single-GPU) model output.

G. Implementation Details

All experiments adopt the DDIM scheduler [31] with $T = 50$ timesteps and generate images at a resolution of 1024×1024. Experiments are performed on NVIDIA GeForce 3090 GPUs (24GB each), connected via PCIe Gen3. The adaptive switching parameters are set as follows: for SDXL, we use $L = 12$, $g_{\text{slope}} = 0.4 \times 10^{-3}$, $k = 5$, and $\tau_{\text{cap}} = 15$; for SD3, we set $L = 15$, $g_{\text{slope}} = 0.1 \times 10^{-3}$, $k = 5$, and $\tau_{\text{cap}} = 40$.

H. Additional Quantitative Results with 8 GPUs

Table 4 further reports the quantitative results under an 8 GPU configuration. On SDXL, our method achieves up to a 4.15× speed-up over the single-GPU baseline while maintaining comparable generation quality. Similarly, on SD3, our framework attains a 4.09× acceleration and consistently outperforms existing distributed inference methods in terms of the speed-quality trade-off. These results further demonstrate the strong scalability of the proposed hybrid parallelism framework as the number of GPUs increases.

I. Additional Qualitative Results

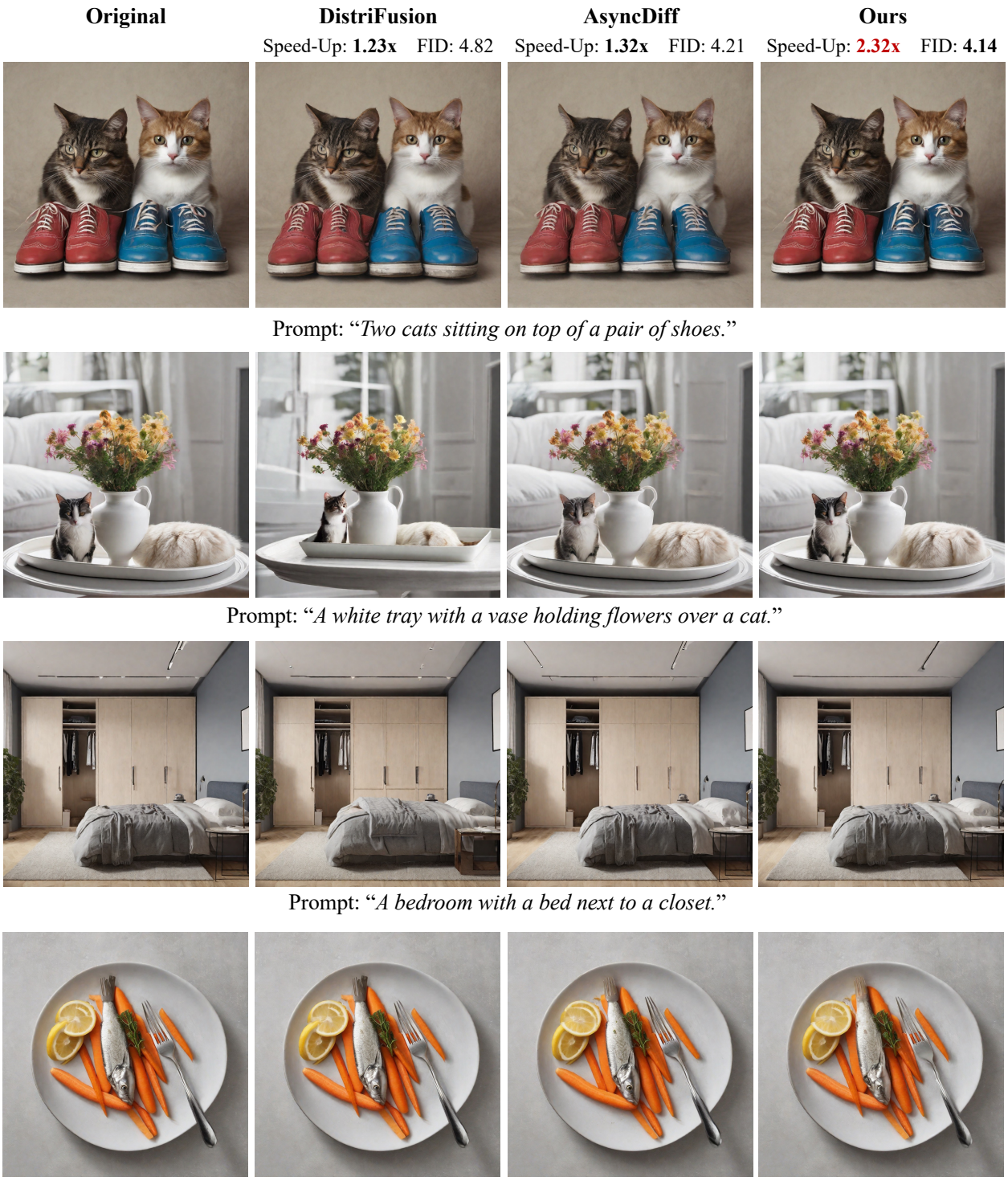


Figure 10. **Additional qualitative results of the main experiments.** We compare 1024×1024 image generations from the SDXL model. Our method achieves the best acceleration and FID performance, while producing visuals most similar to the original.

Parallelism Interval k	**Latency (s)** $\downarrow$	**Speed-Up** $\uparrow$	**FID** $\downarrow$ **(w/ Orig.)**
k=5	7.12	2.31×	**4.100**
k=10	6.89	2.39×	5.942
k=20	6.44	2.56×	7.966
k=30	**5.94**	**2.78**×	9.191

Table 5. **Effect of speed-quality trade-off across different parallelism intervals *k*.** All experiments are conducted on the SDXL model at 1024×1024 resolution with various parallelism intervals and 2 GPUs.

J. Quantitative Results on the Parallelism Interval *k*

Table 5 summarizes the numerical values corresponding to the speed-quality trade-off illustrated in Figure 6. As described in Section 5.4, smaller parallelism interval k preserve higher fidelity, whereas larger k values yield powerful acceleration. The table provides concrete measurements that reflect this trade-off, confirming the same trend observed in the pareto frontier visualization.

K. Qualitative Comparion Results via Different k

Figure 11. **Additional qualitative comparisons across different k values.** We compare 1024×1024 image generations from the SDXL model across various parallelism intervals. Smaller k values preserve higher visual fidelity, whereas larger k gradually reduce local detail due to the extended parallelism window. Although the overall appearance remains similar, fine-grained conditional attributes become subtly blurred as k increases.