TC-Padé: Trajectory-Consistent Padé Approximation for Diffusion Acceleration

Supplementary Material

6. More Quantitative Results

Text-to-Image Generation. To provide a more comprehensive assessment of our method's performance, we present additional evaluation results on the DrawBench benchmark using Aesthetic Score and ImageReward metrics, as referenced in Tabel 7.

Table 7. Quantitative results for **text-to-image** generation on **DrawBench**. Higher is better for quality metrics, and lower is better for efficiency metrics. The best results are in **bold**, and the second best are underlined.

DrawBench	Visual Quality	
	ImgRwd↑	Aesthetic↑
Flux.1[dev], 20steps	1.01	5.83
Δ-DiT (N = 3)	0.52	5.42
ToCa(N=5)	0.82	5.24
TeaCache($\delta = 0.25$)	0.85	<u>5.73</u>
TeaCache($\delta = 0.4$)	<u>0.86</u>	5.71
TeaCache($\delta = 0.6$)	0.74	5.62
TaylorSeer ($\mathcal{N} = 5, \mathcal{O} = 2$)	0.69	4.75
TaylorSeer ($\mathcal{N} = 6, \mathcal{O} = 2$)	0.68	4.40
TC-Padé (fast)	**0.93**	**5.83**

Text-to-Video Generation. For text-to-video generation, we provide a more detailed analysis of our method's performance on VBench-2.0 beyond the Total Score reported in the main text. Figure 7 presents a radar chart showing the performance across five key dimensions: Creativity Score, Commonsense Score, Controllability Score, Human Fidelity Score, and Physics Score, comparing our method against baseline approaches. Our method demonstrates balanced performance between generation quality and efficiency.

User Study. We conducted a user study to evaluate text-to-image generation quality on FLUX.1-dev across different caching methods. Fifty volunteers participated in a blind comparison study, where they were presented with generated images alongside their corresponding text prompts and asked to select the method that best preserved image quality. To ensure unbiased evaluation, participants were not informed which method produced each image. The results are presented in Figure 8. our method achieves a preference rate of 45%, indicating superior perceptual quality compared to other caching approaches.

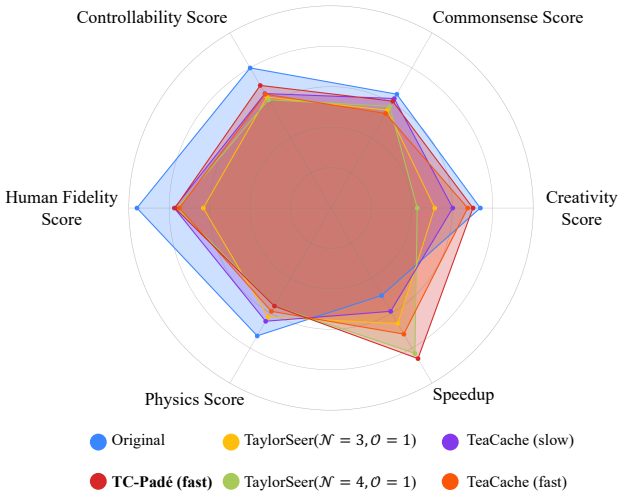


Figure 7. VBench-2.0 metrics and acceleration ratio of TC-Padé and other methods.

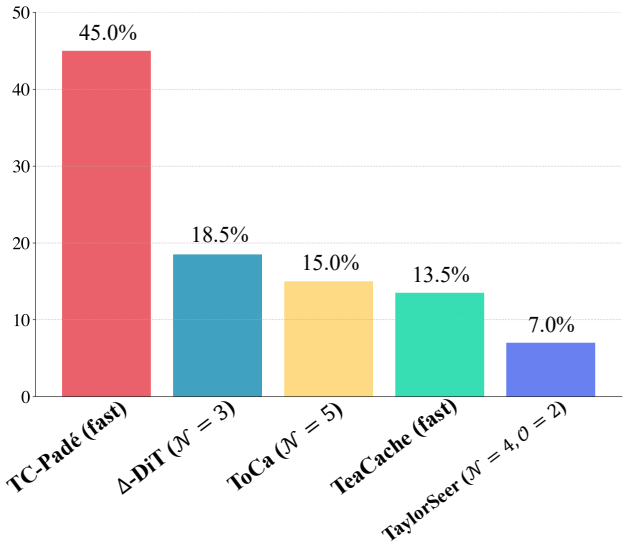


Figure 8. User preference comparison for text-to-image generation on FLUX.1-dev across different caching methods.

7. Rationale for Coefficient Design

7.1. Stability-Aware Coefficient Analysis

The application of Padé approximation to diffusion model cache prediction requires careful adaptation from its classical formulation. In traditional Padé approximation, coefficients are analytically derived by matching Taylor series expansions to achieve optimal rational function approxima-

tion of smooth, deterministic functions. However, direct application of fixed coefficients leads to two critical failure modes:

- **Over-extrapolation**: In regions where residuals change rapidly (early denoising steps), aggressive prediction amplifies errors.
- **Under-utilization**: In stable regions (late denoising steps), conservative prediction fails to leverage temporal coherence.

To address this, we introduce a stability factor σ_{stab} that quantifies local trajectory smoothness and modulates coefficient aggressiveness accordingly. In Eq. 7, the numerator $\|\mathcal{R}_{t+1} - \mathcal{R}_{t+2}\|$ measures the magnitude of recent residual change, while the denominator $\|\mathcal{R}_{t+1} + \mathcal{R}_{t+2}\|$ provides scale normalization to ensure consistency across different noise levels. When the trajectory exhibits rapid variation (large numerator), the exponential term $exp(-\lambda \cdot)$ drives $\sigma_{stab} \to 0$, effectively reducing reliance on potentially misleading historical patterns. Conversely, when residuals evolve smoothly, $\sigma_{stab} \to 1$ enables full utilization of temporal correlations for accurate prediction. $\sigma_{stab} \in (0, 1)$ ensures numerical stability and prevents coefficient explosion.

The coefficients b_0 and b_1 maintain a fixed ratio of $2 : -1$, which implements a second-order finite difference approximation that is optimal for smooth trajectory segments when $\sigma_{stab} \approx 1$. The denominator coefficient $a_1 = \frac{1}{\lambda}\sigma_{stab}$ scales inversely with λ. Since λ is set to a large value (e.g., $\lambda \geq 10$, this ensures a_1 remains small, preventing division instabilities while still allowing adaptive normalization. Since all coefficients scale linearly with σ_{stab}, the formulation guarantees smooth transitions across different stability regimes and avoids discontinuous prediction behavior at regime boundaries.

7.2. Parameter Sensitivity Analysis

Table 8. Impact of sensitivity parameter λ on generation quality.

λ	Visual Quality					
	Aesthetic↑	PickScore↑	ImgRwd↑	PSNR↑	SSIM↑	LPIPS↓
5	5.79	22.48	0.866	23.850	0.8080	0.2929
10	**5.83**	**22.64**	**0.9378**	**24.436**	**0.8376**	**0.2463**
15	5.79	22.43	0.8408	23.752	0.7981	0.3087

The sensitivity parameter λ controls the sharpness of the stable-to-unstable transition.

- **When λ is small (1-5)**, the stability factor transitions gradually between regimes, but this fails to sufficiently suppress predictions during unstable phases, leading to potential extrapolation errors.
- **Medium values λ (5-15)** provide balanced sensitivity that sharply distinguishes stable from unstable trajectories while maintaining smooth coefficient adaptation.
- **When λ exceeds 15**, the exponential decay becomes extremely sharp, which may cause abrupt switching between prediction modes and introduce discontinuity artifacts in the generated samples.

We evaluate different λ values on DrawBench prompts. Results in Tabel 8 show that $\lambda = 10$ achieves the best balance between stable trajectory exploitation and unstable regime suppression, while smaller values ($\lambda = 5$) lead to oversmoothed results due to insufficient suppression of unstable predictions, and $\lambda = 15$ introduces subtle discontinuity artifacts from overly sharp transitions. In our main experiments, we set $\lambda = 10$ across all tasks.

8. More Qualitative Results

This section we show more qualitative results between several cache methods.

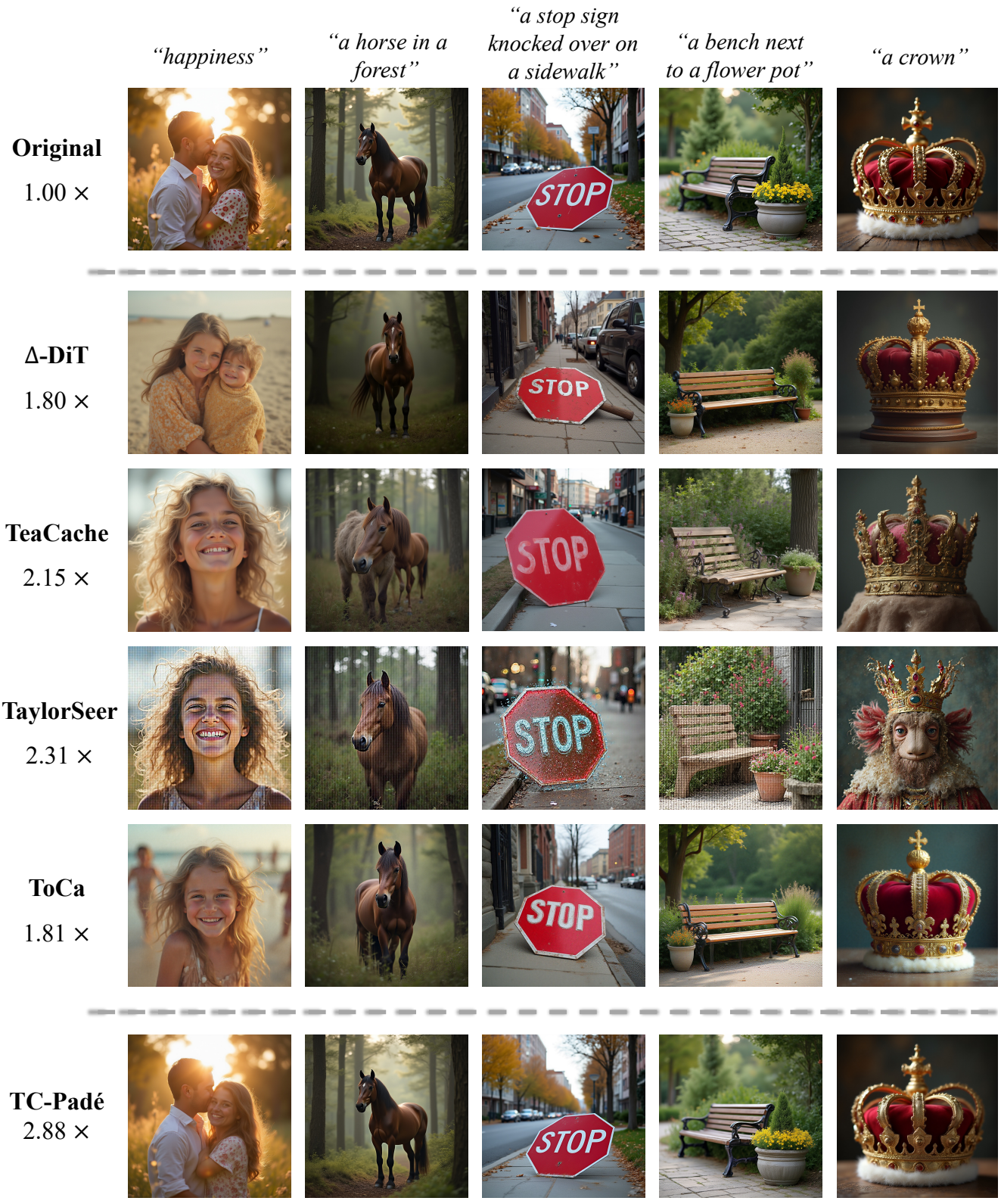
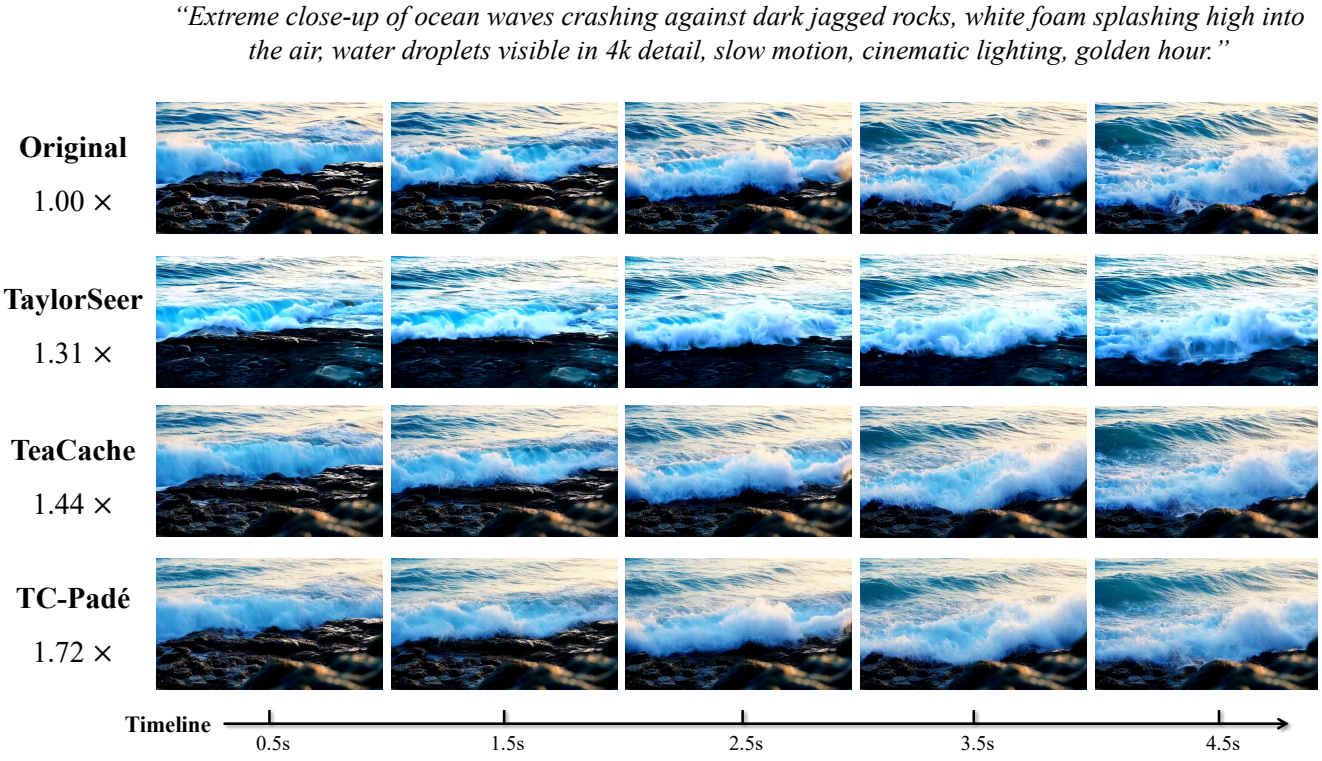


Figure 9. Qualitative examples for TC-Padé and other cache methods on FLUX.1-dev using 20 sampling steps. The text prompts are sampled from Parti prompts.

Figure 10. Qualitative examples for TC-Padé and other cache methods on FLUX.1-dev using 20 sampling steps. The text prompts are sampled from Parti prompts.



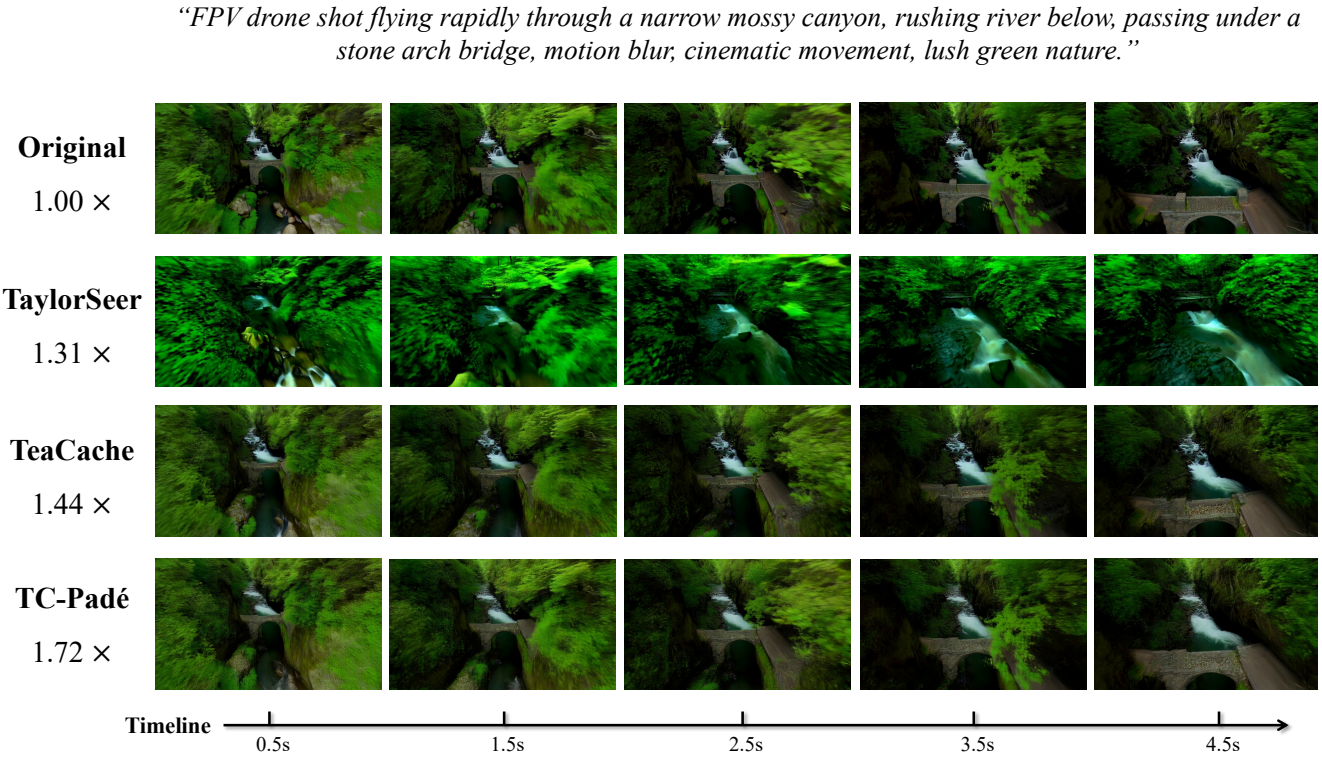


Figure 11. Qualitative examples for TC-Padé and other cache methods on Wan2.1-1.3B using 20 sampling steps.

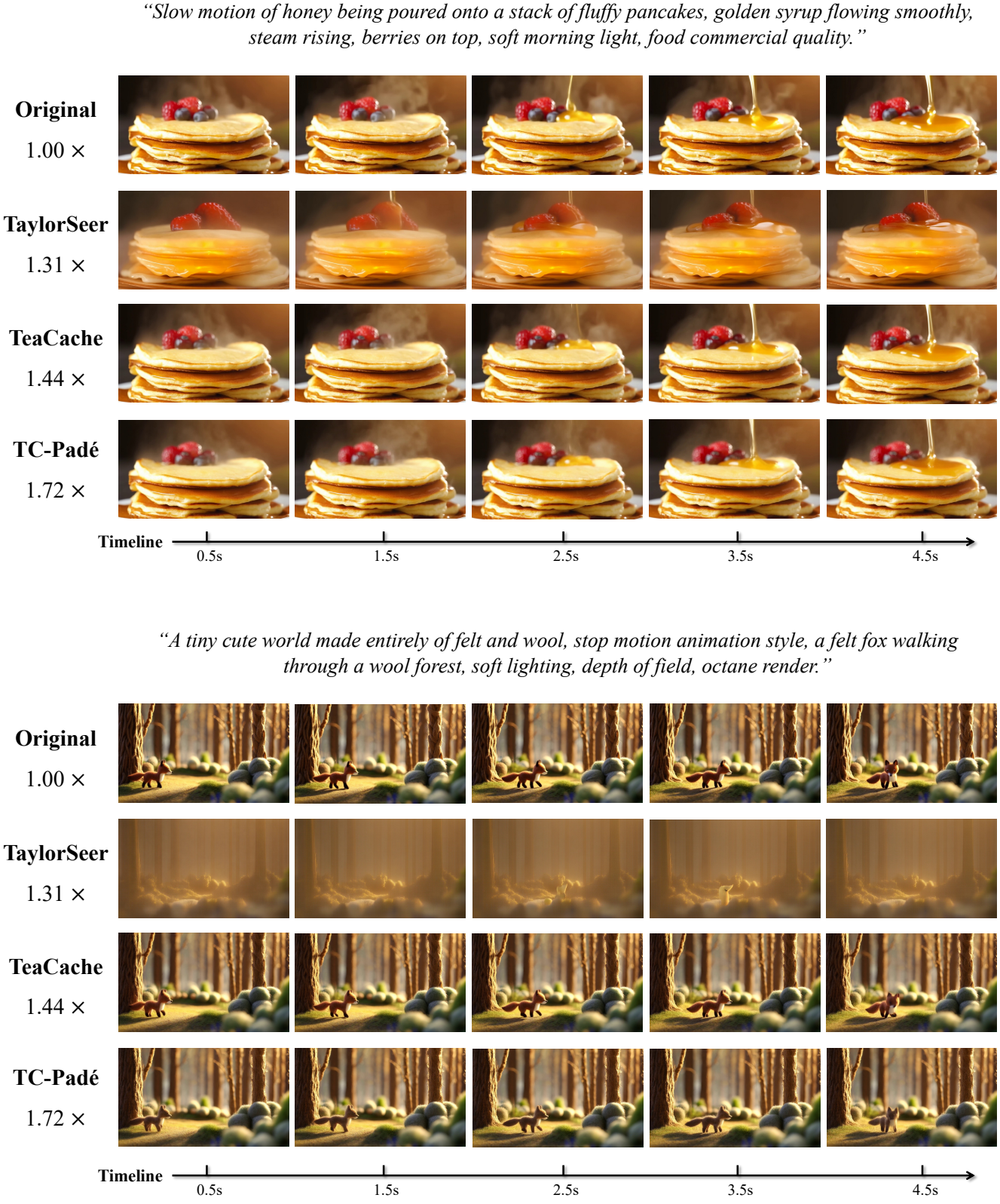


Figure 12. Qualitative examples for TC-Padé and other cache methods on Wan2.1-1.3B using 20 sampling steps.