LeMiCa: Lexicographic Minimax Path Caching for Efficient Diffusion-Based Video Generation

Appendix

A Pseudocode: Lexicographic Minimax Path Selection

Below we present the full pseudocode implementation of the lexicographic minimax path selection algorithm, as introduced in Section 3.3. The algorithm leverages dynamic programming to efficiently compute an optimal caching path from source s to target t under a step budget constraint B. Unlike standard shortest-path methods, our approach handles the non-additive and non-Markovian nature of error accumulation by minimizing edge weights in a lexicographically ordered manner.

Algorithm 1 Lexicographic Minimax Path Selection

Input: Directed acyclic graph $G = (V, E)$, start node s, end node t, step limit B
Output: Lexicographic Minimax Path P^*
Initialization:
  $dp[v][k]$: maximum edge weight on any k-step path to v
  $paths[v][k], edges[v][k]$: corresponding node and edge sequences
  $dp[s][0] \leftarrow 0$, $paths[s][0] \leftarrow [[s]]$, $edges[s][0] \leftarrow [[\,]]$
Main Loop:
for $k = 0$ to $B - 1$ **do**
  for each node v with $dp[v][k] < \infty$ **do**
    for each neighbor u of v **do**
      $w \leftarrow$ weight of edge (v, u)
      $m \leftarrow \max(dp[v][k], w)$
      if $m < dp[u][k+1]$ **then**
        $dp[u][k+1] \leftarrow m$
        Update $paths[u][k+1]$, $edges[u][k+1]$ from v
      else if $m = dp[u][k+1]$ **then**
        Append new paths and edges from v to $paths[u][k+1]$, $edges[u][k+1]$
      end if
    end for
  end for
end for
Final Selection:
$P^* \leftarrow \min\big(\text{zip}(paths[t][B], edges[t][B]), \text{key} = \lambda(p, e) : \text{sorted}(e, \text{reverse=True})\big)$

B Experiment Settings

B.1 Models

In this paper, we introduce LeMiCa, a novel caching technique designed to accelerate and enhance a range of state-of-the-art video synthesis models, including Open-Sora 1.2 [55], Latte [24], and CogVideoX [47]. Open-Sora 1.2 integrates 2D/3D VAEs and ST-DiT blocks for efficient video compression and generation. Latte leverages spatio-temporal tokenization and Transformer layers to model video distributions in the latent space. CogVideoX employs a 3D VAE and expert Transformers with adaptive LayerNorm for modality fusion and high-fidelity generation. In our experiments, we adopt the CogVideoX-2B variant.

B.2 Details of the Compared Methods

PAB introduces a pyramid-style broadcasting mechanism to reduce redundant attention computations in diffusion models. By observing a U-shaped pattern in attention differences across steps, PAB

applies adaptive broadcast strategies based on the variance of different attention types (e.g., spatial, temporal, cross-modal). Stable attention outputs are efficiently reused in later steps, reducing computation. All experiments use PAB's default parameter settings.

TeaCache is a training-free, architecture-agnostic caching method that exploits the correlation between timestep embedding changes and model output differences across adjacent steps. By introducing a unified threshold-based strategy, TeaCache decides when to activate caching through an accumulated error-based discriminator. Since this method operates solely along the temporal dimension without modifying specific model components, it offers strong generalization and broad applicability.

B.3 Model Forward Steps

Model Forward Steps. In this work, we control the acceleration efficiency of LeMiCa via the Model Forward Steps B. Smaller values of B reduce the denoising time, leading to higher speed-up ratios. We consider two variants: LeMiCa-slow, which emphasizes visual fidelity, and LeMiCa-fast, which prioritizes inference efficiency. The corresponding B values for each variant across different models are listed in Table 4.

Table 4: Model forward steps B under different configurations.

Model	Configuration	Model Forward Steps B
Open-Sora 1.2	Original	30
	LeMiCa-slow	19
	LeMiCa-fast	11
Latte	Original	50
	LeMiCa-slow	27
	LeMiCa-fast	14
CogVideoX	Original	50
	LeMiCa-slow	27
	LeMiCa-fast	16

C OOD Generalization Analysis

We analyze the out-of-distribution (OOD) generalization ability of LeMiCa through two evaluation setups: VBench and IP-VBench.

OOD Evaluation on VBench. LeMiCa is evaluated on VBench [11], which follows a distribution distinct from T2V-CompBench [41] used for DAG construction. We quantify the distributional shift by computing prompt-level distances using text embeddings and PCA (see Table 5).

Table 5: Distributional distance analysis between VBench and T2V-CompBench.

Attribute	Description	VBench
Distance	VBench vs. T2V-CompBench	0.61
Radius	1 Std. Deviation of T2V-CompBench	0.39
Distance / Radius	Ratio of distance to radius	1.58
OOD Status*	Is it OOD?	✓

Despite this evident OOD setting, LeMiCa consistently maintains strong acceleration and visual quality (Table 1), demonstrating robustness to unseen prompt distributions.

OOD Evaluation on IP-VBench. We further test LeMiCa on IP-VBench [1], which contains intentionally unrealistic and semantically diverse prompts across four domains: *Physical*, *Biological*, *Social*, and *Geographical*. These prompts differ significantly from training data, with Distance/Radius nearly doubling compared to T2V-CompBench (see Table 6).

Across all domains, LeMiCa substantially outperforms TeaCache in LPIPS, SSIM, and PSNR, underscoring its strong generalization and robustness under severe OOD conditions.

Table 6: Quantitative OOD performance on IP-VBench across four semantic domains.

Method	Domain	LPIPS (↓)	SSIM (↑)	PSNR (↑)	Distance/Radius	OOD Status*
TeaCache	Physical	0.093	0.911	26.7	2.09	✓
	Biological	0.171	0.839	24.0	1.90	✓
	Social	0.144	0.842	24.9	1.93	✓
	Geographical	0.072	0.914	29.8	2.13	✓
	Overall	**0.120**	**0.877**	**26.4**	**2.01**	✓
LeMiCa	Physical	0.039	0.954	34.6	2.09	✓
	Biological	0.054	0.905	31.4	1.90	✓
	Social	0.040	0.946	33.1	1.93	✓
	Geographical	0.038	0.945	34.9	2.13	✓
	Overall	**0.042**	**0.938**	**33.5**	**2.01**	✓

D Offline Cost

The graph construction in LeMiCa is an entirely offline, three-stage process. First, Edge Weight Estimation estimates reconstruction errors by running full-generation passes on approximately 20 sampled prompts; this task is fully parallelizable and only needs to be run once per model configuration. The subsequent stages, Graph Construction (fusing jump edges into a sparse DAG) and Path Optimization (employing a lexicographic minimax search to find acceleration paths), are both highly efficient, each completing in under 1 second. As detailed in Table 7, these offline procedures incur negligible overhead, yet this low-cost offline computation yields up to **2.44**× acceleration during inference generation.

Table 7: Offline cost analysis of LeMiCa on OpenSora.

Stage	Description	Time Cost (Ref)	Affects Inference
Edge Weight Estimation	Full-generation error estimation	∼3.18 min / prompt	No
Graph Construction	Build sparse DAG	<1 sec	No
Path Optimization	Minimax search for jump paths	<1 sec	No
Inference Acceleration	Execute jump paths with caching	Up to 2.44× faster	Yes

E More Visual Results

We present additional visual comparisons across three foundational models: Open-Sora [55], Latte [24], and CogVideoX [47]. Results are grouped into two settings: fidelity-focused and speed-focused.

E.1 Fidelity-Focused

We perform frame-by-frame comparisons to assess fine-grained differences in quality (LeMiCa-slow vs. TeaCache-slow). Since this setting uses relatively low acceleration ratios, artifacts are less obvious in real-time playback. To address this, we extract representative frames that highlight detail preservation, object integrity, and temporal consistency. As shown in Figures 8, 9, 10, 11, and 12, our method consistently produces more coherent results across all baselines.

E.2 Speed-Focused

To evaluate robustness under aggressive acceleration, we compare videos generated with higher speed-up ratios (LeMiCa-fast vs. TeaCache-fast). This setting is designed to prioritize generation speed without significantly compromising visual quality. Under such conditions, baseline methods are more prone to issues such as flickering, object drift, and reduced temporal consistency. In contrast, our method maintains strong temporal and semantic coherence, even at high generation speeds.

As part of the supplementary material, we include the following video files: **Speed-Focused Open-Sora.mp4**, **Speed-Focused Latte.mp4**, and **Speed-Focused CogVideoX.mp4**.

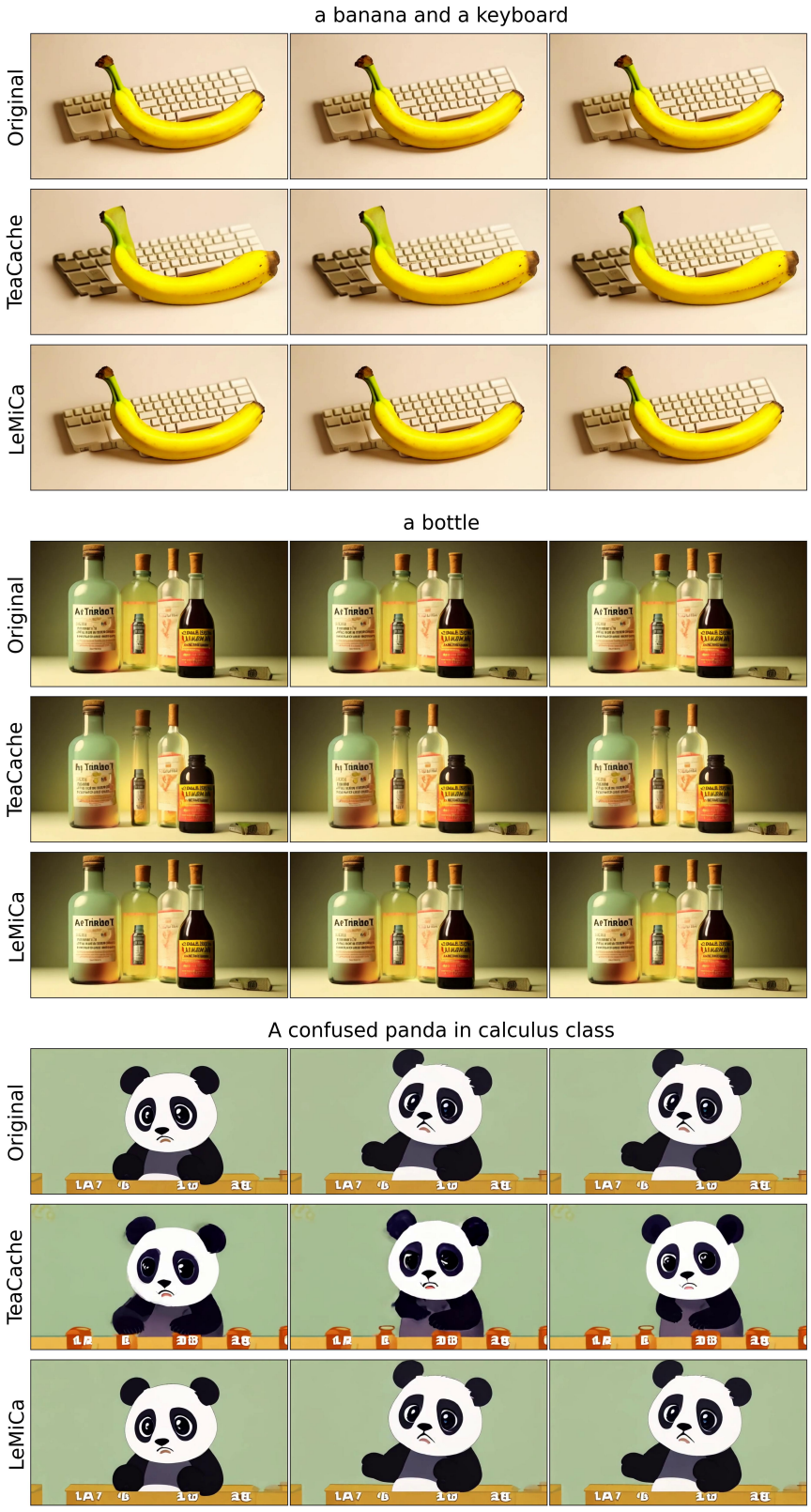

Figure 8: More visual results on Open-Sora (Part I).

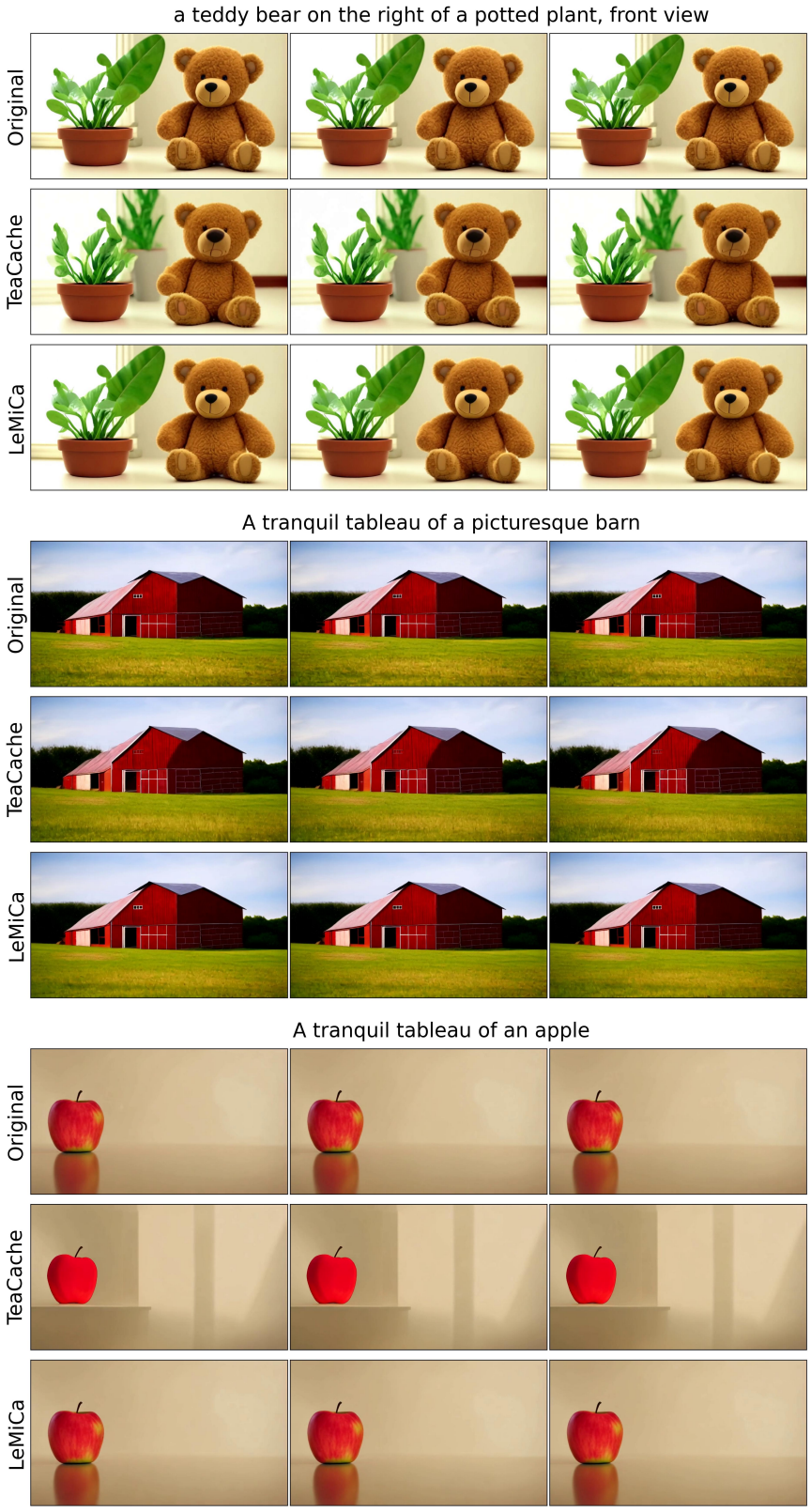

Figure 9: More visual results on Open-Sora (Part II).

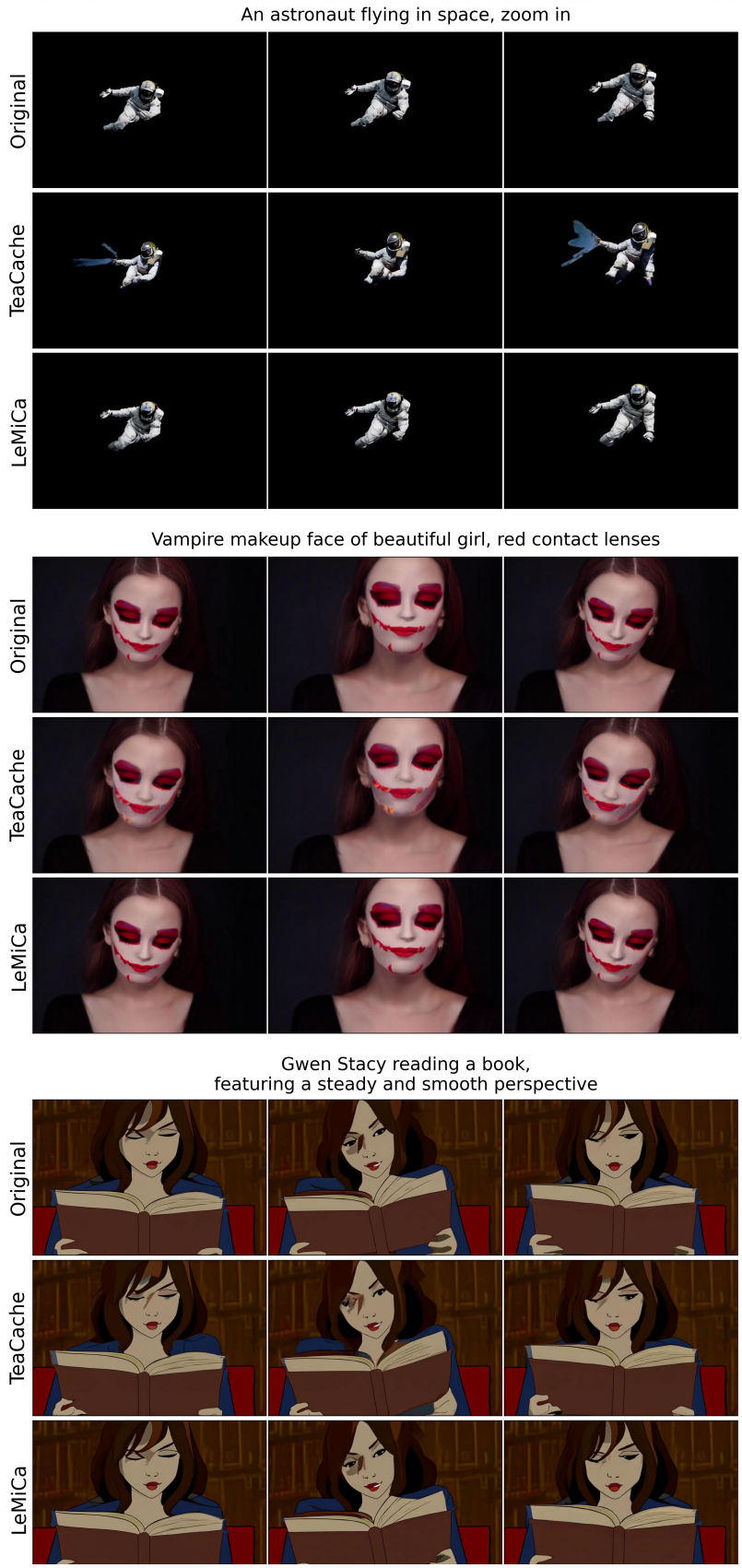

Figure 10: More visual results on CogVideoX.

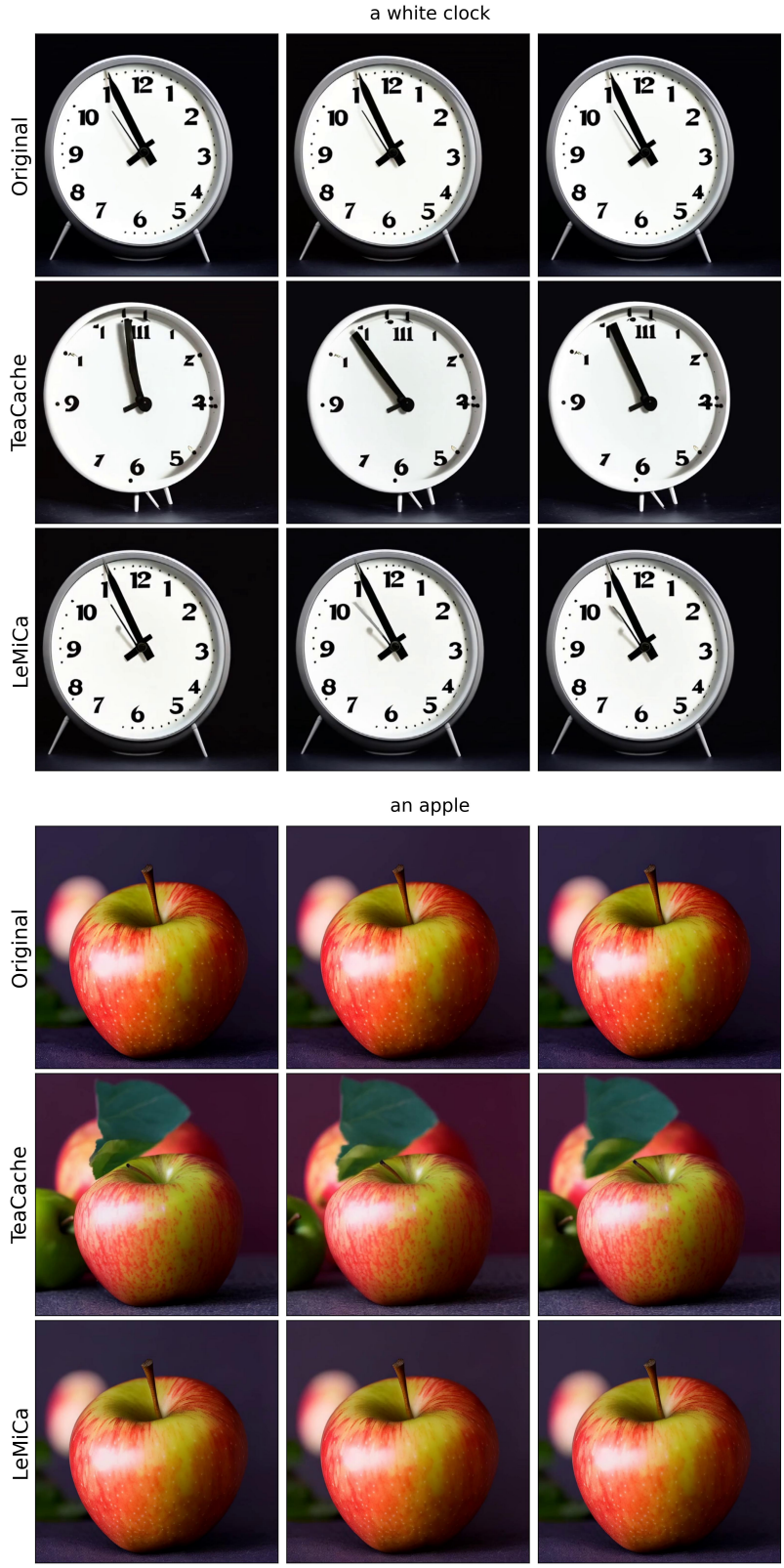

Figure 11: More visual results on Latte (Part I).

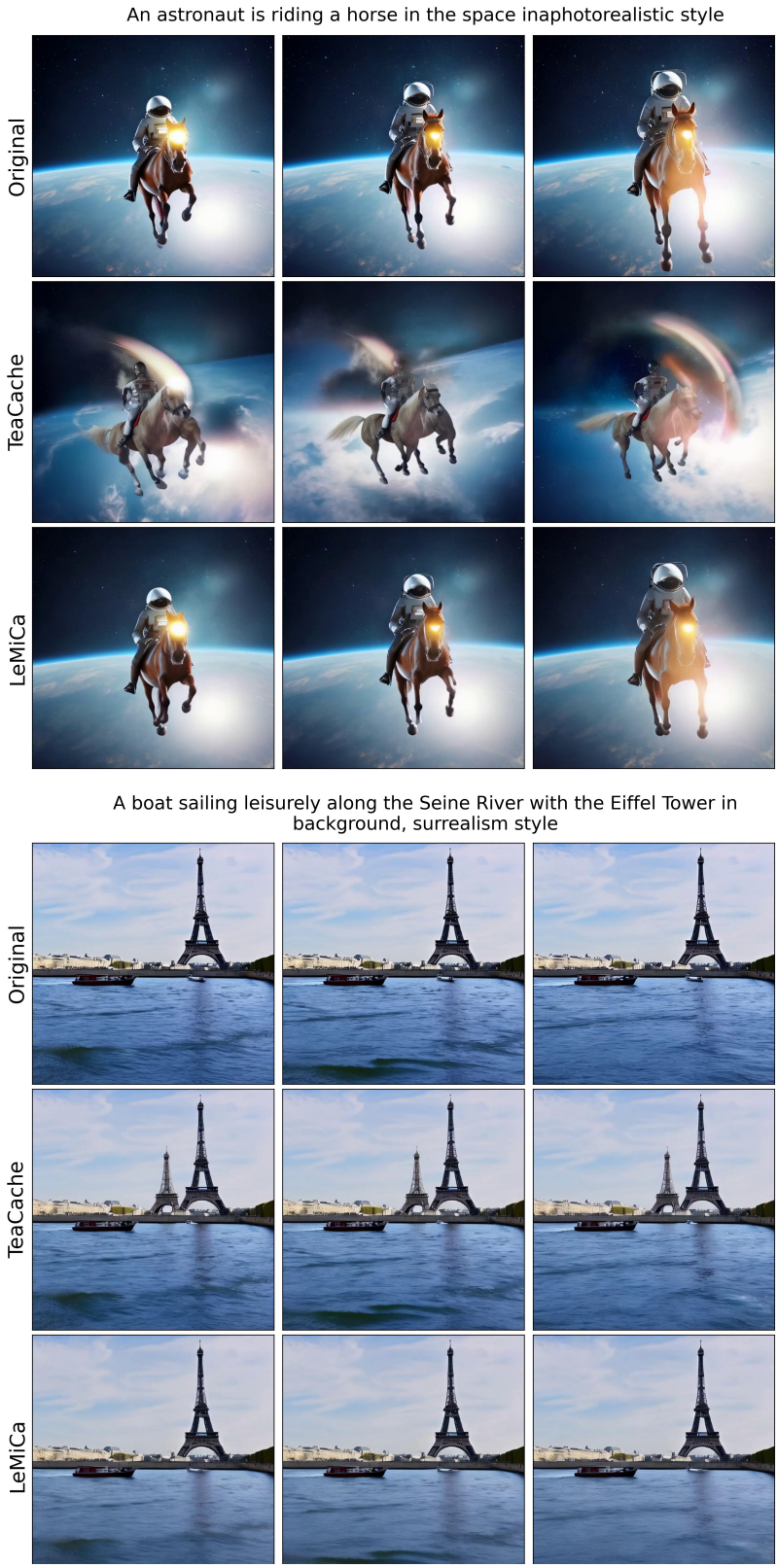

Figure 12: More visual results on Latte (Part II).

F Extension to text-to-image model

To further evaluate the scalability of our approach, we extend **LeMiCa** to the commercial text-to-image model **Qwen-Image**[46]. As shown in Figure 13 and Table 8, LeMiCa achieves strong generalization while maintaining high efficiency and visual fidelity, consistently outperforming TeaCache and DBCache[42] across both qualitative and quantitative metrics.

Figure 13: Qualitative results of Qwen-Image with LeMiCa.

Table 8: Comparison of inference efficiency and visual quality of LeMiCa and other acceleration methods on the Qwen-Image.

Method	Efficiency		Visual Quality				
	Latency (s)↓	Speed↑	Image Reward↑	CLIP Score↑	LPIPS↓	SSIM↑	PSNR↑
Original: 50 steps	32.68	1.00×	1.18	33.63	—	—	—
30% steps	9.86	3.31×	1.10	31.62	0.36	0.73	15.83
TeaCache (l=0.6)	18.52	1.76×	1.09	32.60	0.42	0.70	14.90
DBCache (Fn8, Bn0, r0.6)	13.95	2.34×	0.87	32.62	0.36	0.78	21.45
LeMiCa (B=17)	**13.31**	**2.46×**	**1.16**	**33.39**	**0.09**	**0.93**	**27.92**
DBCache (Fn4, Bn2, r1.5)	9.57	3.41×	-2.06	15.50	0.89	0.13	5.56
DBCache + TS (O=4)*	9.70	3.37×	-0.23	29.75	0.63	0.65	16.57
LeMiCa (B=10)	9.75	3.35×	**1.15**	**33.49**	**0.27**	**0.80**	**18.58**

* DBCache is derived from the Cache-DiT acceleration framework, and TS denotes the Taylorseer acceleration method.

G Limitation

Although our method achieves strong performance in both acceleration and video fidelity, it still has certain limitations. First, when the original video quality is low, particularly in scenarios involving complex motion dynamics, it struggles to consistently generate satisfactory results. This reflects a dependency on the representational capacity of the underlying diffusion model. Second, under high acceleration ratios, some degree of quality degradation remains inevitable due to the significantly reduced number of model forward steps. We believe that continued progress in foundational video generation models will help alleviate these issues. Moreover, since our approach focuses solely on

temporal step scheduling and is agnostic to model architecture, it can be quickly adapted to future, more powerful diffusion models.

H Social Impact

Diffusion-based video generation models are often limited by high inference time and computational cost. Our method alleviates this by significantly improving efficiency without requiring additional training. This enables broader access to high-quality video synthesis, particularly in resource-constrained settings. By reducing computation during the inference process, our approach also lowers energy use and carbon emissions, contributing to more sustainable AI development. Furthermore, we will release our code to support future research.