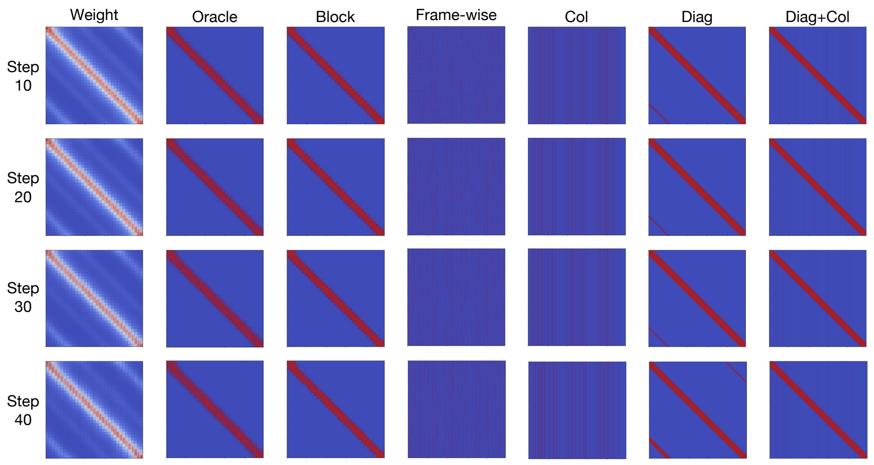
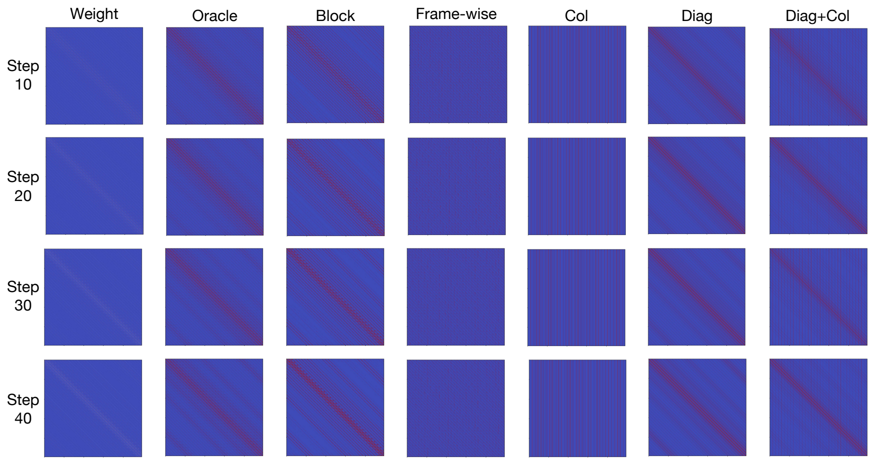
A Visualization of Attention Weights and Sparse Patterns

In Section 3, we conclude that the sparse patterns in the DiT model remain invariant across steps but vary with respect to heads and layers. This conclusion is preliminarily validated through the results presented in Figure 6a. In this section, to provide a more intuitive demonstration of the aforementioned conclusion, we visualize some representative attention weights of *HunyuanVideo* along with the search results of various patterns, as shown in Figures A, B, and C. Here, Oracle denotes the theoretically optimal pattern, which is searched using the block pattern with a block size of $B = 1$. The Oracle pattern serves only as a theoretical optimal guideline but is impractical for real-world use since using $B = 1$ would lead to extremely low inefficiency in the attention computation. Its highly dispersed nature is incompatible with GPU computation, leading to extremely low processing efficiency and failing to achieve any acceleration.

From these figures, it is evident that although the specific values of attention weights may fluctuate across time steps, the optimal pattern (Oracle) remains largely unchanged over time. Furthermore, the visualized results clearly indicate that many heads do not follow a discernible structured pattern. In such cases, applying static patterns or an approximate search based on dynamic patterns becomes ineffective. It is also apparent from the figures that regardless of the complexity or lack of discernible structure in the patterns, our blockified pattern achieves the closest approximation to the Oracle pattern.

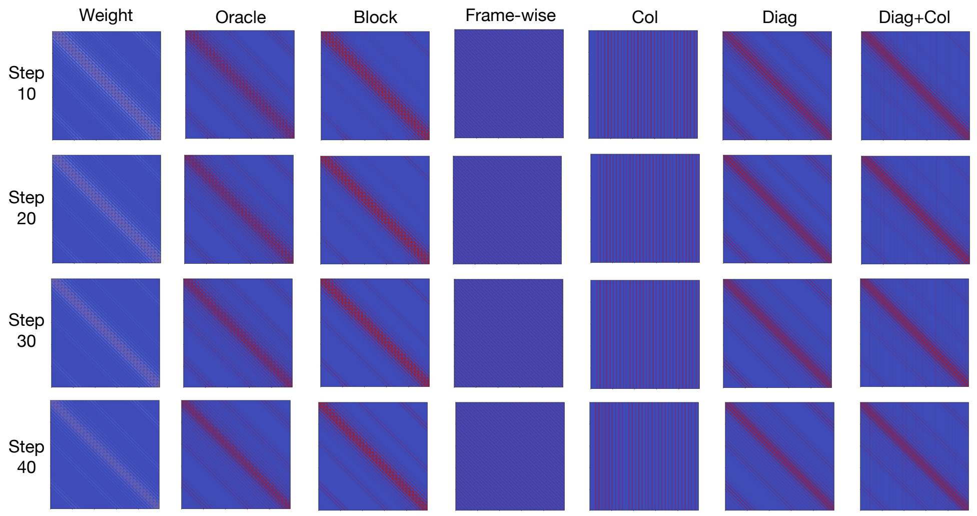


(a) Layer0-Head0.

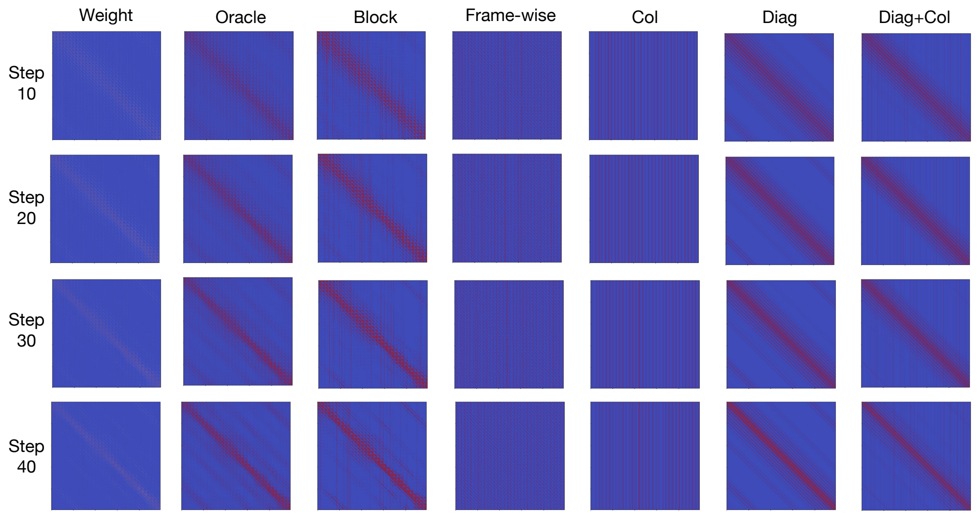


(b) Layer0-Head6.

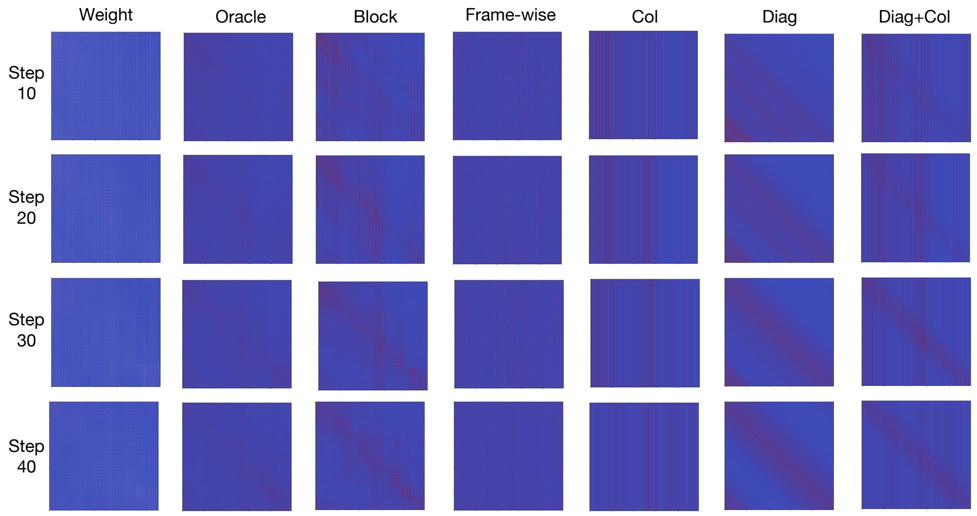
Figure A: (Part 1) The visualization of attention weight and different sparse patterns. These figures show a clear invariance of sparse patterns with respect to steps.



(a) Layer0-Head18.

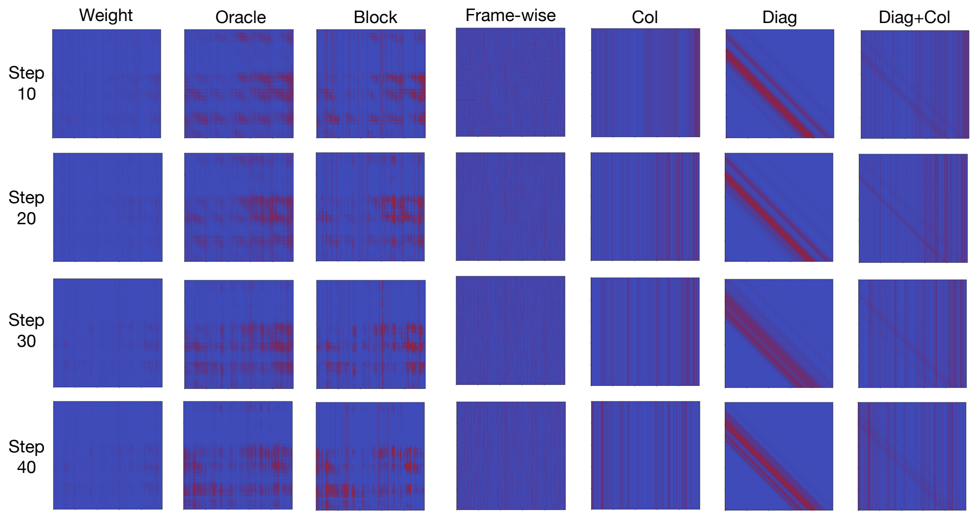


(b) Layer15-Head12.

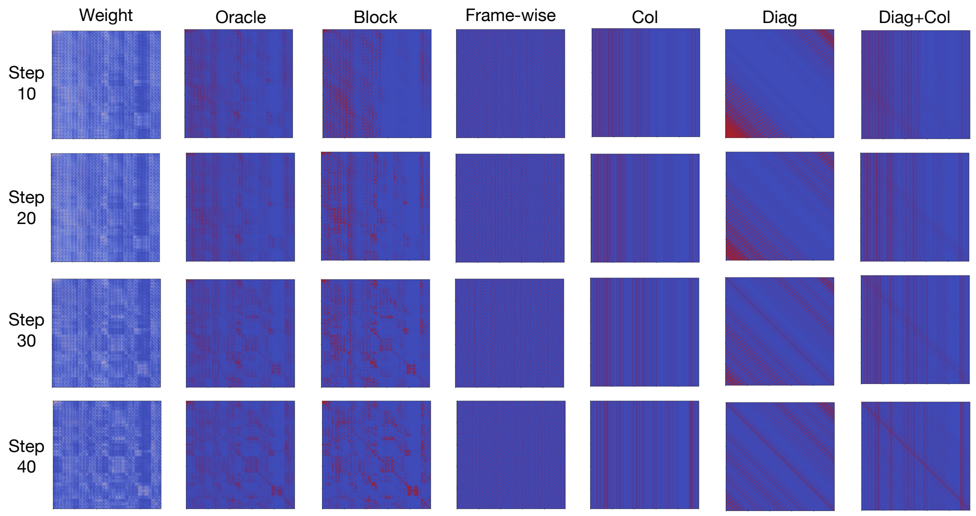


(c) Layer15-Head18.

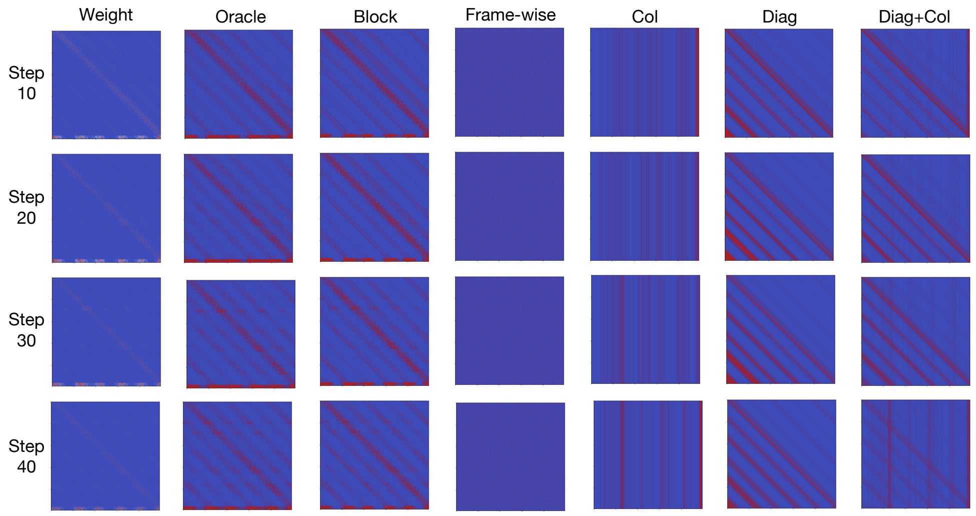
Figure B: (Part 2) The visualization of attention weight and different sparse patterns. These figures show a clear invariance of sparse patterns with respect to steps.



(a) Layer30-Head12.



(b) Layer45-Head6.



(c) Layer45-Head18.

Figure C: (Part 3) The visualization of attention weight and different sparse patterns. These figures show a clear invariance of sparse patterns with respect to steps.

B More Experimental Results

B.1 Quality-Latency Trade-off

We provide a comparison of the trade-off between quality and latency, corresponding to Figure 9 in the main text. The results show that under the condition of achieving the same quality, AdaSpa has a significantly higher generation speed than other methods.

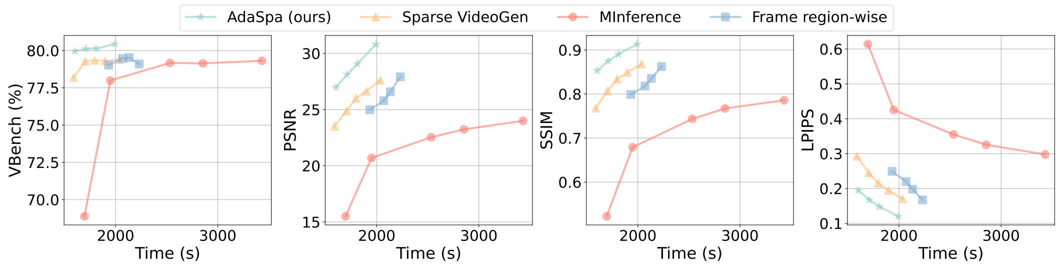


Figure D: The trade-off between quality and latency.

B.2 Evaluation on Short-step Model

To accelerate the video generation process, there is a trend to distill long-step DiTs into short-step variants that require fewer diffusion steps. To validate the effectiveness of AdaSpa in short-step models, we consider the distilled 6-step model *FastHunyuan* (13B). We set $\gamma = 0.85$, $B = 64$, $\mathcal{T} = \{0\}$. The results are presented in Table A. It can be seen that AdaSpa still achieves the best performance across all metrics while maintaining the highest speedup.

Table A: Quantitative evaluation of quality and latency for AdaSpa and other methods on *FastHunyuan*.

Method	VBench (%) ↑	PSNR ↑	SSIM ↑	LPIPS ↓	Latency (s)	Speedup
FastHunyuan (Full Attention)	43.79	-	-	-	372.13	1.00×
+ MInference	37.21	12.55	0.3908	0.5851	301.30	1.24×
+ Sparse VideoGen	43.34	14.11	0.4585	0.4838	252.01	1.48×
+ Frame region-wise	43.22	14.01	0.4432	0.5032	262.58	1.32×
+ AdaSpa (w/o Head Adaptive)	43.64	14.30	0.4585	0.4828	241.90	1.54×
+ AdaSpa (**ours**)	**43.71**	**14.49**	**0.4784**	**0.4732**	**238.11**	**1.56×**

B.3 Comparison with Other DiT Optimization Methods

We also conduct experiments to compare AdaSpa with other widely used training-free methods in video generation, as shown in Table B.

Adaptive Cache (AdaCache) and Pyramid Attention Broadcast (PAB) are well performed cached-based methods that cache and reuse intermediate activations instead of recomputing them at every step. Token Merge (ToMeSD) is a strategies that consolidate near-duplicate tokens to shorten sequence length.

Table B: Comparison with Other DiT Optimization Methods.

Method	VBench (%) ↑	PSNR ↑	SSIM ↑	LPIPS ↓	Latency (s)	Speedup
HunyuanVideo (Full Attention)	80.10	-	-	-	3213.76	1.00×
+ AdaCache	74.61	24.93	0.8070	0.2926	2124.32	1.51×
+ PAB	78.39	27.61	0.8683	0.1703	1791.02	1.79×
+ ToMeSD	70.55	14.03	0.3161	0.6606	**1711.70**	**1.88×**
+ AdaSpa (**ours**)	**80.13**	**29.07**	**0.8905**	**0.1478**	1810.23	1.78×

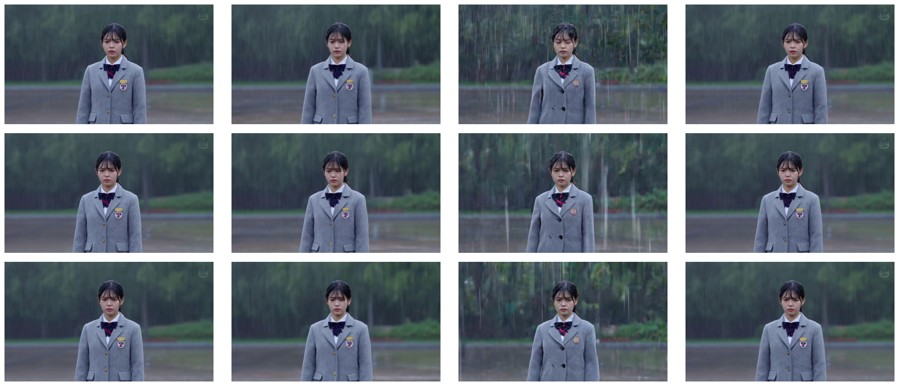
Cache methods yield high visual similarity (PSNR, etc.) due to its imitation of the original video, but they often introduce pixel-level blur, lowering video quality (VBench). As for ToMeSD, its similarity assumptions for UNet-based image models don't generalize well to DiT-based video models, leading to poor quality, though it shows good acceleration potential.

C More Visualization Results of Generated Videos

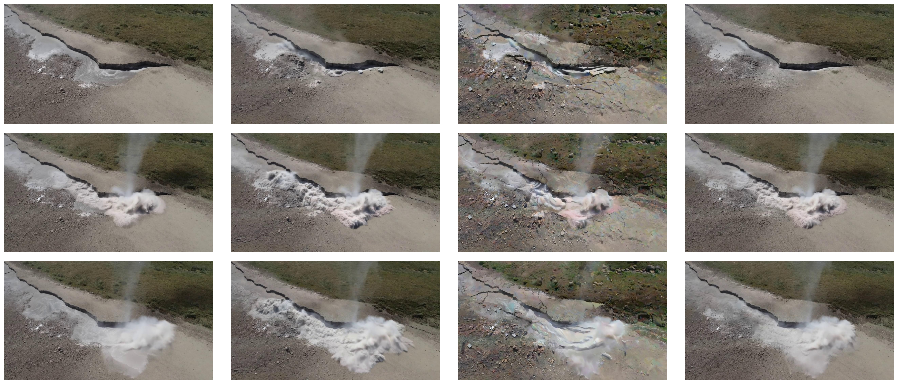
In this section, we provide more visualization results of the generated videos of AdaSpa, MInference (dynamic pattern + approximate search), and Sparse VideoGen (static pattern) in Figure E and F. AdaSpa consistently achieves higher quality and generation speed than the counterparts.

HunyuanVideo (129 frames, 720p)

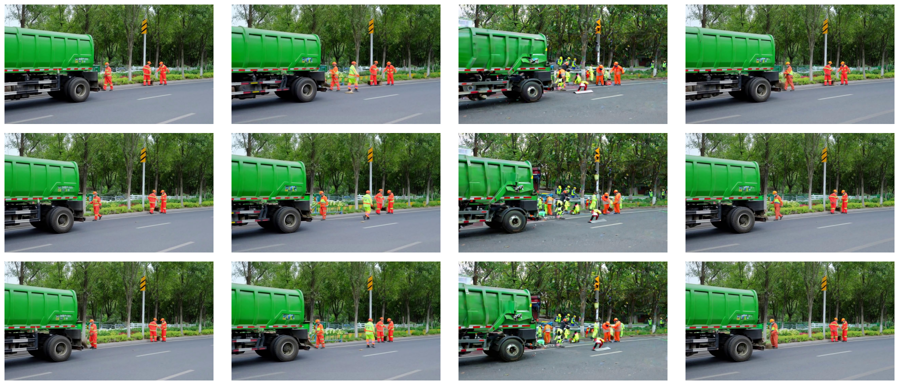
Full Attention	Sparse VideoGen	MInference	**AdaSpa (ours)**
	PSNR = 27.61	PSNR = 22.53	PSNR = **29.07**
Latency = 54 min	Latency = 34 min	Latency = 42 min	Latency = **30** min



Prompt: A female student in a gray coat slowly stands up in the rain. The entire video presents a melancholic atmosphere.



Prompt: A violent earthquake caused the ground to shake violently, and subsequently, a huge crack appeared in the ground.



Prompt: A green garbage truck is parked by the roadside. Several sanitation workers are cleaning the road behind the garbage truck.

Figure E: Comparison of the generated videos of different methods on *HunyuanVideo*.

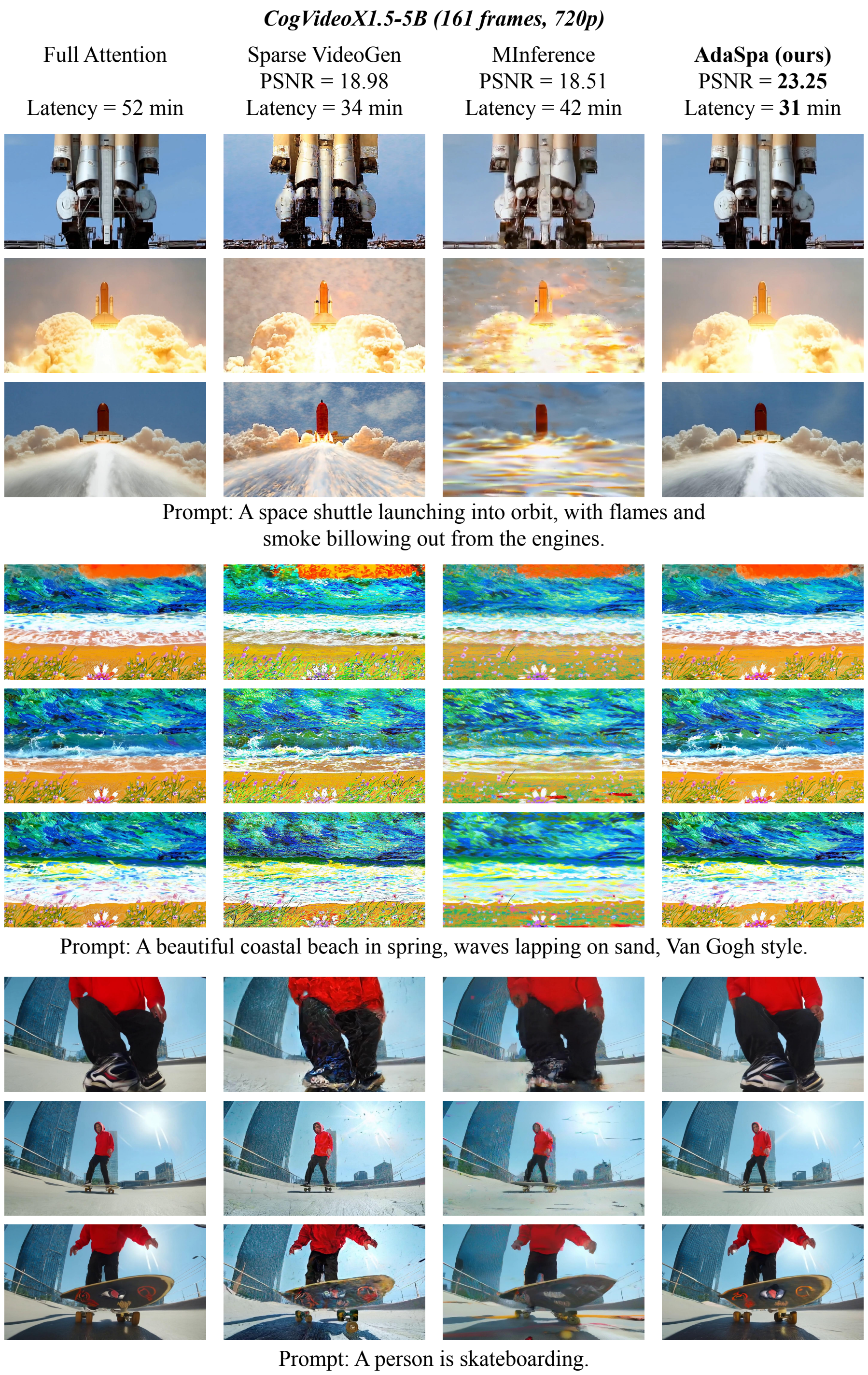

CogVideoX1.5-5B (161 frames, 720p)
Full Attention
Latency = 52 min
Sparse VideoGen
PSNR = 18.98
Latency = 34 min
MInference
PSNR = 18.51
Latency = 42 min
AdaSpa (ours)
PSNR = **23.25**
Latency = **31** min
Prompt: A space shuttle launching into orbit, with flames and smoke billowing out from the engines.
Prompt: A beautiful coastal beach in spring, waves lapping on sand, Van Gogh style.
Prompt: A person is skateboarding.

Figure F: Comparison of the generated videos of different methods on *CogVideoX1.5-5B*.