Video-T1: Test-Time Scaling for Video Generation

Supplementary Material

A. More Implementation Details

A.1. More Discussion of Image-level Alignment

In our generation process, inspired by recent work [2], each frame is generated with image-level TTS. Specifically, we employ a two-stage evaluation mechanism. First, a potential assessment reward model examines whether the partially generated frame exhibits sufficient visual clarity for meaningful evaluation and assigns a binary label. If the output is deemed unclear ('no'), the model skips further processing for that frame. Otherwise ('yes'), the frame advances to a secondary evaluation stage where its potential to yield a high-quality final image is assessed. Again, a binary decision is made: if the frame is unlikely to lead to an optimal outcome, the generation path is truncated immediately; if it passes, the synthesis continues to produce the final image.

A.2. More Discussion of Hierarchical Prompting

In our approach to hierarchical prompting, we employ DeepSeek-R1-8B [1], a large language model distilled from the LLaMA-8B [4] architecture. DeepSeek-R1-8B is used to decompose a given input prompt into three distinct hierarchical prompts, each tailored to represent a specific stage of the video sequence:

- Static scene description: The first prompt provides a detailed depiction of the static scene intended for the initial frame, establishing the starting visual context of the sequence.
- Action/motion directions: The second prompt outlines the actions or motion directions that guide the dynamic progression across the intermediate frames, ensuring a coherent evolution of the scene.
- Expected ending state: The third prompt delineates the expected ending state for the concluding frame, defining the desired outcome of the video sequence.

DeepSeek-R1-8B processes the input prompt and generates these three hierarchical prompts, which are then returned as an ordered list of length three. At each stage of the video sequence—initial, intermediate, and final—the test verifier evaluates the generated output with the corresponding hierarchical prompt. This stage-specific assessment ensures that the video content aligns with the intended descriptions, maintaining both accuracy and coherence throughout the sequence. Figure 6 gives an example of hierarchical prompting generation and test-time verification.

A.3. Detailed Complexity Analysis

In our method, the video generation process is modeled as the dynamic growth of a forest, where the trees are branched and pruned over time. Specifically, similar to the linear search, we begin by generating N initial frames, representing the roots of N trees in the forest. Each time step $t \in [0, T-1]$ corresponds to a level in the tree, and each frame is treated as a tree node. We consider the process in which each of the N trees grows by adding one level of nodes. At each time step, the k_{t-1} surviving parent nodes dynamically branch into b_t possible continuations. We then evaluate all $k_{t-1} \cdot b_t$ nodes using a heuristic reward function H, followed by pruning to retain only the top k_t branches. The time complexity of growing one level of the tree is:

$$O(k_{t-1}b_t + b_t \log(k_{t-1}b_t)). \tag{1}$$

Here, generating $k_{t-1}b_t$ nodes takes $O(k_{t-1}b_t)$ time, the evaluation cost per node is $O(1)$, and the total evaluation time is $O(k_{t-1}b_t)$. Heap sorting for pruning costs $O(b_t \log(k_{t-1}b_t))$. By iteratively applying dynamic branching and heuristic pruning, the deepest leaf nodes in the forest correspond to the final frames of the video, with the path to those nodes representing the optimal video sequence. The overall time complexity of this process is:

$$O(k_0 + \sum_{t=1}^{T-1} k_{t-1}b_t + b_t \log(k_{t-1}b_t)). \tag{2}$$

In practice, we set a branching limit b for dynamic branching, *i.e.*, $b_t \leq b$. In the heuristic pruning step, we use the heuristic reward function H to prune branches that fall below the average value. On average, each pruning step retains $k_t \approx \frac{k_{t-1}b_t}{2} = \frac{k_0 \prod_{i=1}^{t} b_i}{2^t} \leq \frac{k_0 b^t}{2^t}$ branches before k_t drops to 1. Therefore, we have:

$$\begin{aligned} &O(k_0 + \sum_{t=1}^{T-1} k_{t-1}b_t + b_t \log(k_{t-1}b_t)) \\ &= O(k_0 + \sum_{t=1}^{T-1} \frac{k_0 \prod_{i=1}^{t} b_i}{2^t} + b_t \log(\frac{k_0 \prod_{i=1}^{t} b_i}{2^t}) \end{aligned} \tag{3}$$

In practice, we set $k_0 = N$ and $b = 2$. In the worst-case scenario, assuming $b_i = b = 2$ for all i, the resulting time complexity is:

$$O(N + TN + 2T \log(N)) = O(TN). \tag{4}$$

This complexity is consistent with that of the linear search. However, in our actual experiments, we perform branching operations only at specific prompt stages to ensure a diverse and stable transition between stages. Consequently, b_t remains 1 for most timesteps, leading to the following update rule:

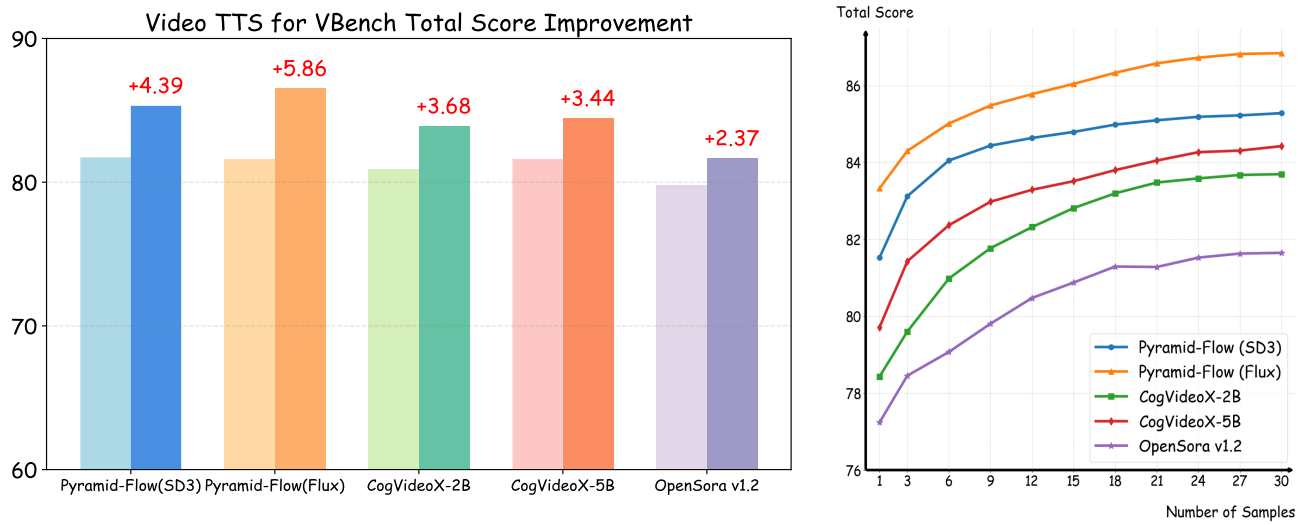


Figure 1. **Results of Test-Time Scaling for Video Generation.** As the number of samples in the search space increases by scaling test-time computation (TTS), the models' performance exhibits consistent improvement (In the bar chart, light colors correspond to the results without TTS, while dark colors represent the improvement after TTS.).

$$k_t = \begin{cases} \frac{k_{t-1} b_t}{2}, & 0 < t \leq \log(k_0) \\ 1, & t > \log(k_0). \end{cases} \tag{5}$$

Thus, Eq. 2 can simplify to:

$$\begin{aligned} & O(k_0 + \sum_{t=1}^{T-1} k_{t-1} b_t + b_t \log(k_{t-1} b_t)) \\ &= O(k_0 + \sum_{t=\log(k_0)+1}^{T-1} b_t + \sum_{t=1}^{\log(k_0)} \frac{k_0 \prod_{i=1}^{t} b_i}{2^t} + b_t \log(\frac{k_0 \prod_{i=1}^{t} b_i}{2^t})) \\ &= O(k_0 + T - \log(k_0) + k_0 + \log^2(k_0) - \frac{\log 2}{2} log^2(k_0)) \\ &= O(N + T). \end{aligned} \tag{6}$$

Compared to the quadratic complexity of linear search, our approach converge at a geometric rate, ultimately achieving linear complexity. It significantly reduces computational costs while maintaining high sample diversity. The logarithmic dependency on N ensures efficient scaling, making our method more suitable for high-dimensional video generation. Additionally, by dynamically adjusting the branching factor, we achieve a better trade-off between exploration in early timesteps and convergence in later stages.

B. More Experiments

B.1. More Quantitative Results

We performed multiple random linear search experiments across various video generation models with different verifiers. Figures 7-9 present more quantitative results on VBench across different dimensions. These indicate that TTS consistently delivers stable performance improvements across dimensions. Moreover, the evaluation accuracy of different verifiers varies across dimensions, justifying our use of multiple verifiers. Figure 1 present trends for performance gain across different video generation models as the number of samples increases.

B.2. More Qualitative Results

We provide additional visual results for Pyramid-Flow [3] and CogVideoX-5B [5] in Figures 10-15. Each example displays different video outputs as NFE increases with the same prompt input. The results show that video quality and text alignment improve with more TTS samples.

B.3. Discussion of RL from reward model feedback

TTS in video generation differs significantly from the Chain-of-Thought (CoT) approach employed in language models. It does not generate new contextual information nor leverage in-context learning capabilities; instead, it relies on a reward model to prune specific paths. If a video model has already undergone reinforcement learning based on feedback from a reward model during the training phase, will the TTS method still yield substantial benefits? To address this question, we conducted additional experiments. Specifically, we first trained CogVideo-2B with a reward model via reinforcement learning and then applied our TTS method to the fine-tuned model. The results indicate that while reinforcement learning prunes numerous low-reward paths, it fails to ensure convergence to the global optimum. As illustrated in Table 1 and Figure 3, the model after reinforcement learning still retains significant potential for im-

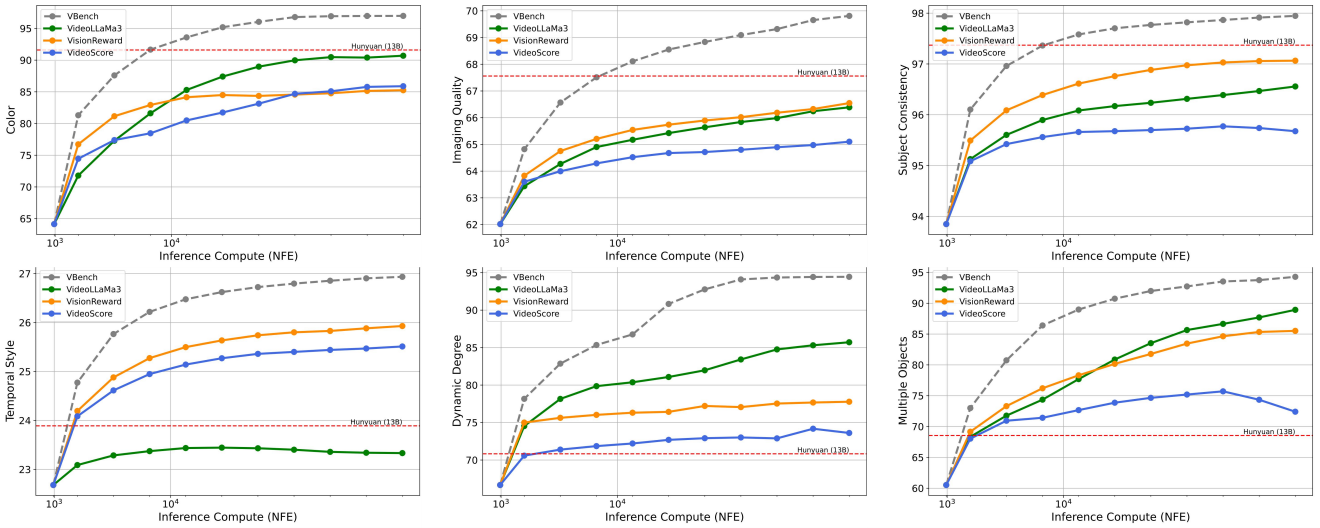


Figure 2. Using TTS, the small model (Pyramid-Flow) achieves scores that are close to, or even exceed, those of the 13B large model (HunyuanVideo) in many dimensions. The gray dashed horizontal line in the figures indicates HunyuanVideo's score in that dimension.

provement through the TTS method.

	Overall Consistency	Imaging Quality	Aesthetic Quality
CogVideoX-2B	24.16	61.34	55.22
+ TTS	26.34$^{+9.02\%}$	63.18$^{+2.99\%}$	57.78$^{+4.64\%}$
CogVideoX-2B (after RL)	25.80	61.10	58.15
+ TTS	27.00$^{+4.65\%}$	66.64$^{+9.07\%}$	61.12$^{+5.11\%}$

Table 1. Quantitative comparisons with RL fine-tuned model.

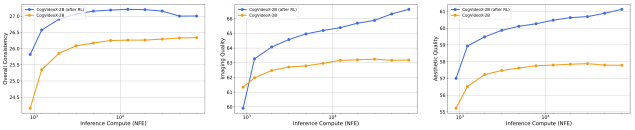


Figure 3. Qualitative comparison with RL fine-tuned model.

B.4. Comparison with Large Models

Figure 2 compares the outputs of the small model Pyramid-Flow (2B) with TTS and the large model HunyuanVideo (13B) single. As scaling increases, the small model's performance approaches or even surpasses that of the large model in many dimensions.

We also conduct experiments on latest open-source methods (Hunyuan Video and Wan2.1) to evaluate TTS on a subset of VBench. As shown in Tab. 2 and Fig. 4,larger models receive more improvements on several dimensions.

	Overall Consistency	Aesthetic Quality	Background Consistency
Hunyuan Video	27.27	64.75	95.47
+ TTS	30.68$^{+12.5\%}$	69.48$^{+7.30\%}$	99.12$^{+3.93\%}$
Wan 2.1	26.26	65.62	97.58
+ TTS	30.10$^{+14.6\%}$	71.40$^{+8.80\%}$	99.74$^{+2.21\%}$

Table 2. Experiment Results on latest opensource models.

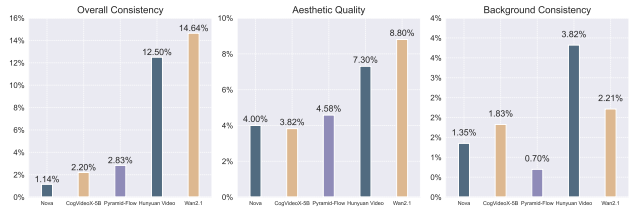


Figure 4. TTS Improvement Comparison with large models.

B.5. Failure Cases

Figure 5 shows failure cases in Pyramid-Flow experiments where TTS does not significantly improve video quality. Increasing inference computation fails to generate reasonable details (*e.g.*, hand movements), indicating that model performance limitations constrain TTS method improvements.

Figure 5. Failure cases on prompt "A person is clapping".

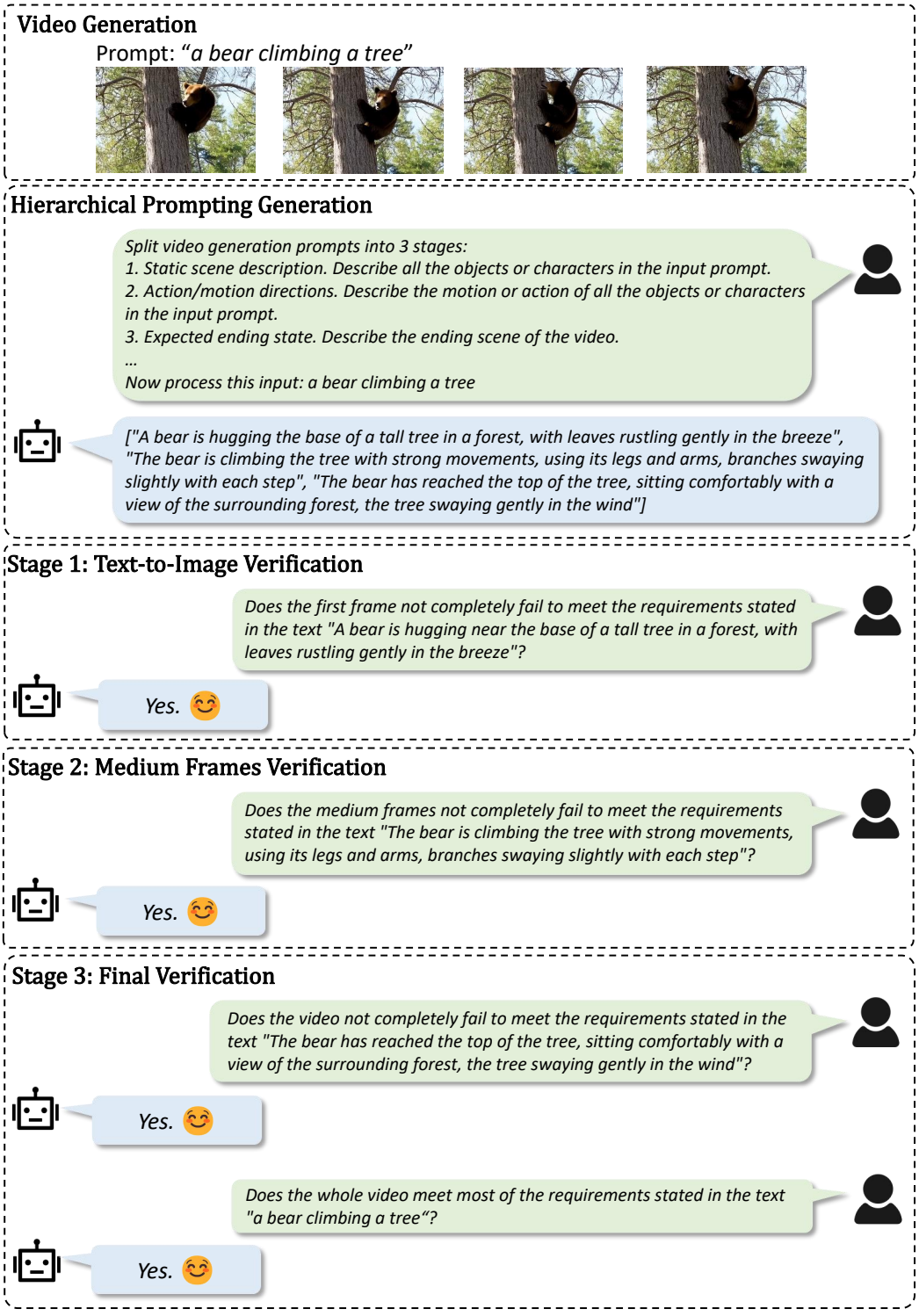


Figure 6. Verifications during TTS process.

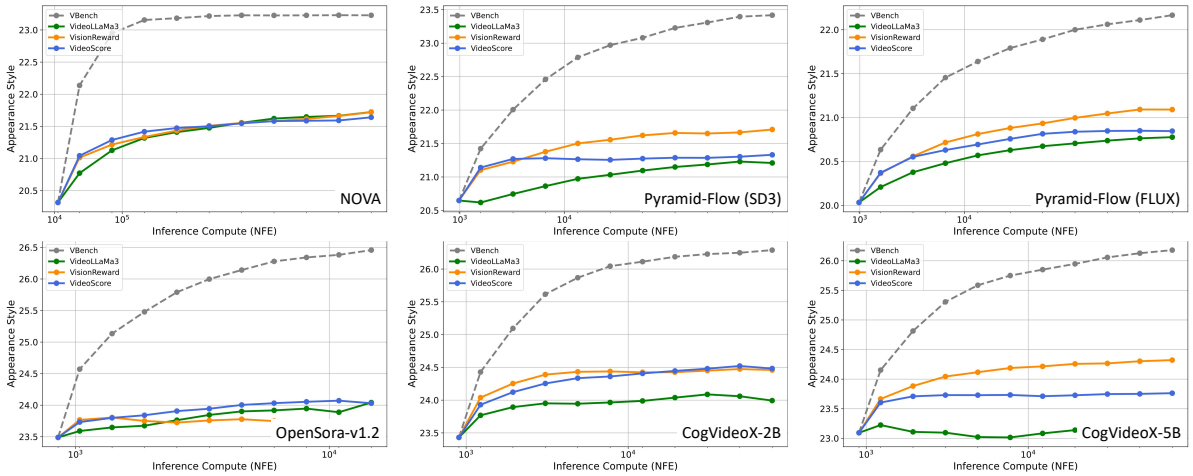


Figure 7. TTS performance on Appearance Style across diverse verifiers.

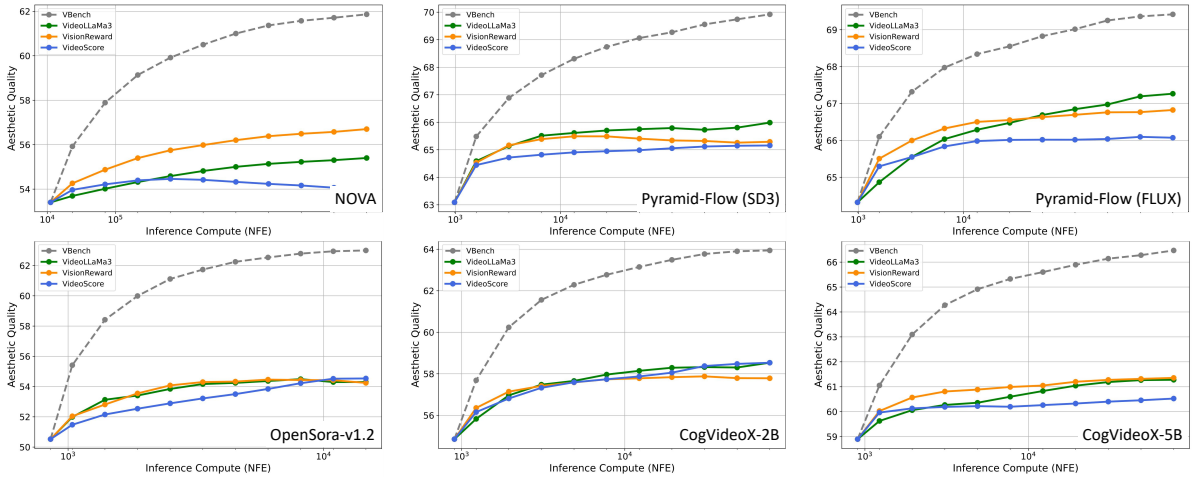


Figure 8. TTS performance on Aesthetic Quality across diverse verifiers.

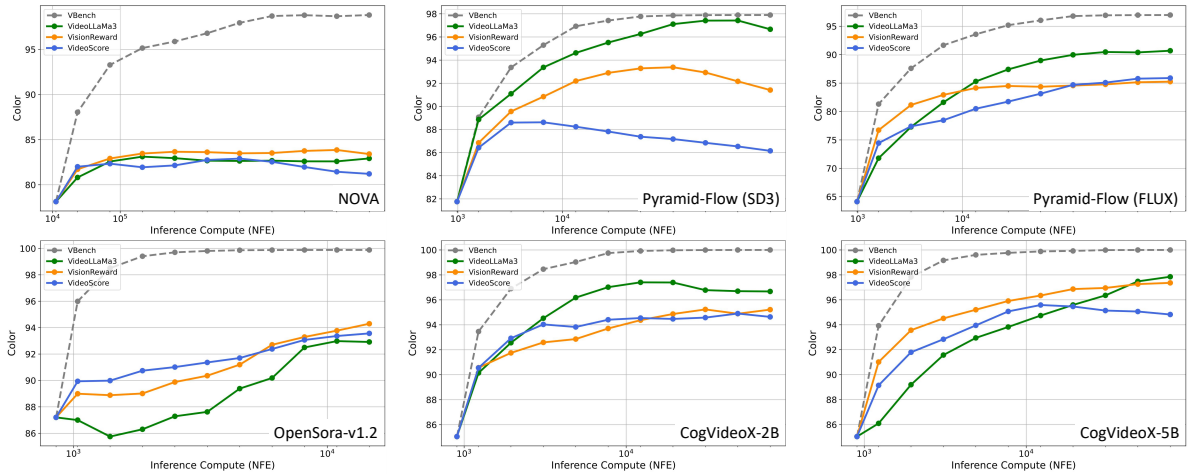


Figure 9. TTS performance on Color across diverse verifiers.

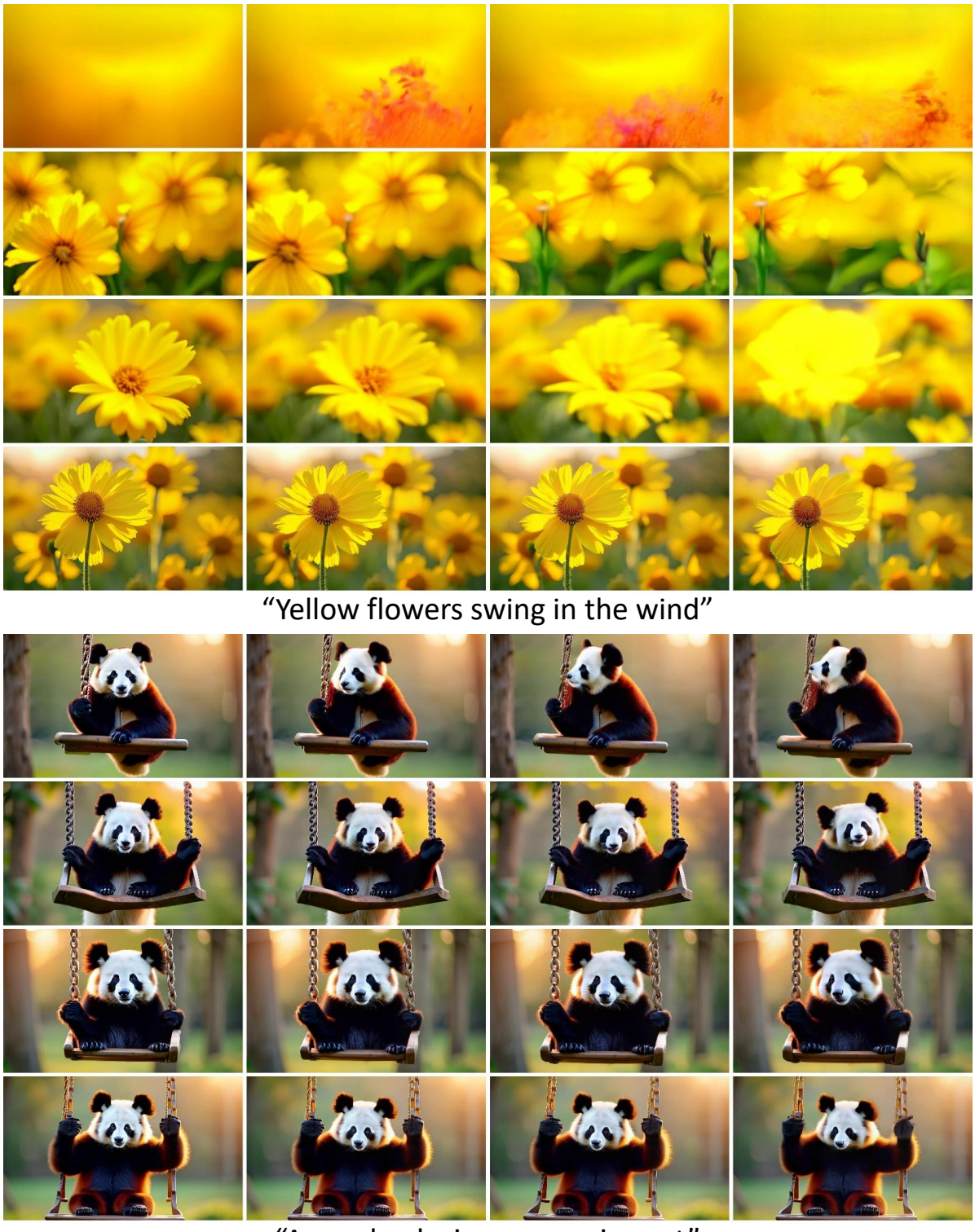


Figure 10. More visual results during TTS process on Pyramid-Flow. From left to right, each row of frames are extracted from a video sequence. From top to bottom, each row represents the output video results of TTS with an increasing number of samples.

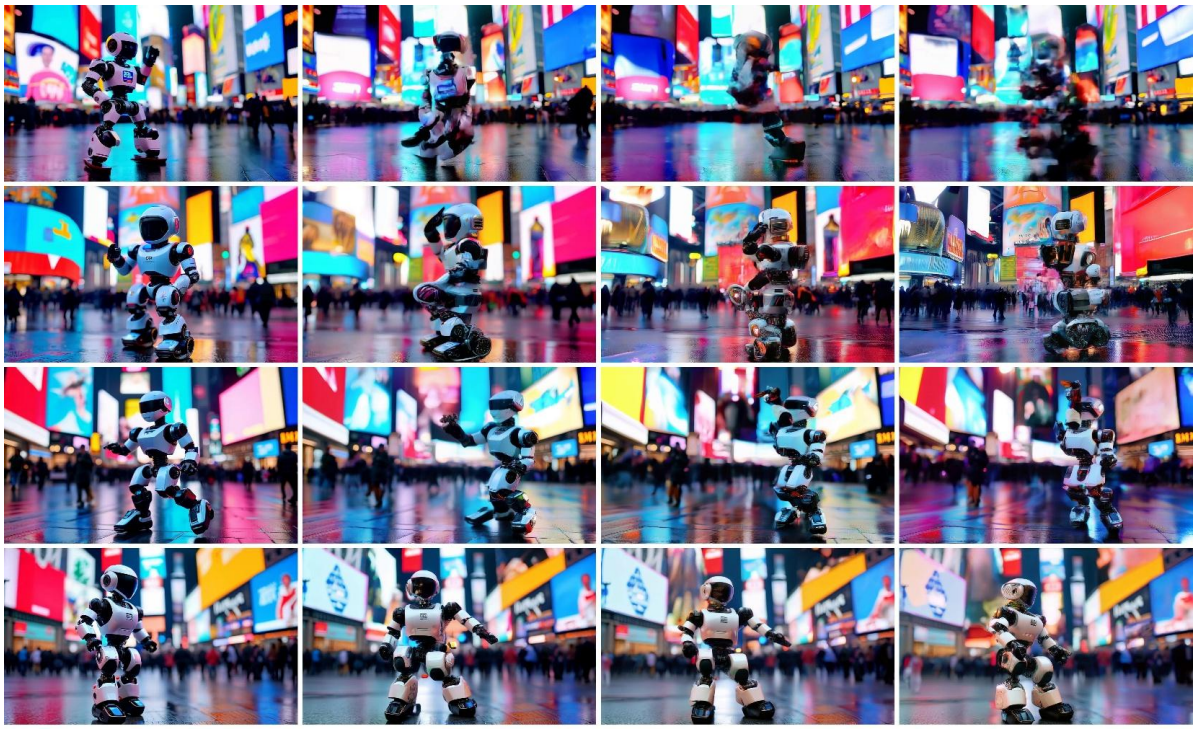
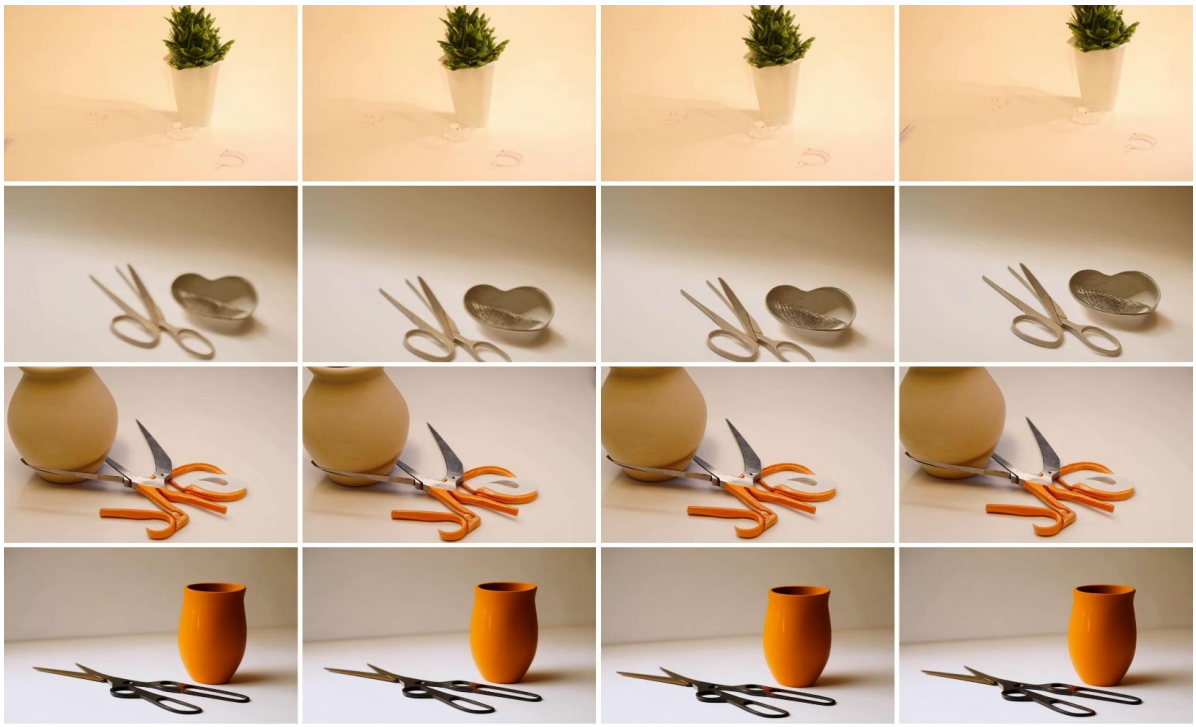


Figure 11. More visual results during TTS process on Pyramid-Flow. The TTS method can effectively alleviate common issues in video generation, such as those related to human motion and complex movements.



“A vase and scissors.”

“A happy fuzzy panda playing guitar nearby a campfire,
snow mountain in the background”

Figure 12. More visual results during TTS process on CogVideoX-5B.

Figure 13. More visual results during TTS process on CogVideoX-5B.

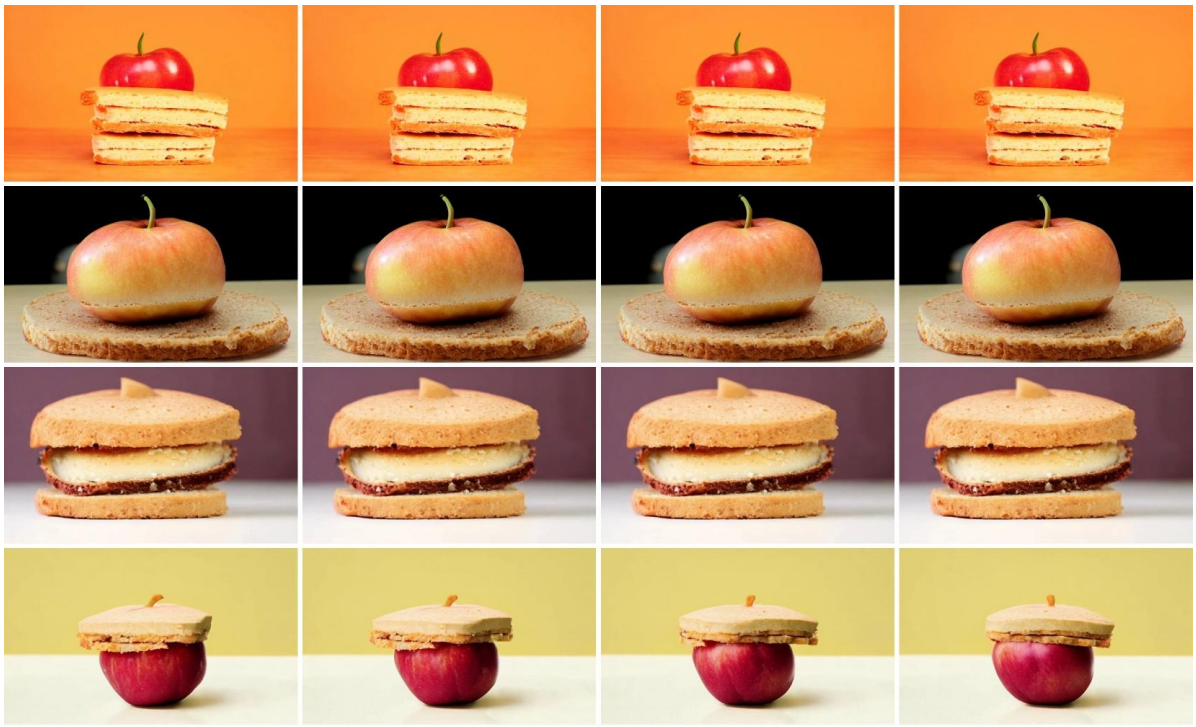


Figure 14. More visual results during TTS process on CogVideoX-2B. The TTS method can help to enhance multi-object spatial perception.

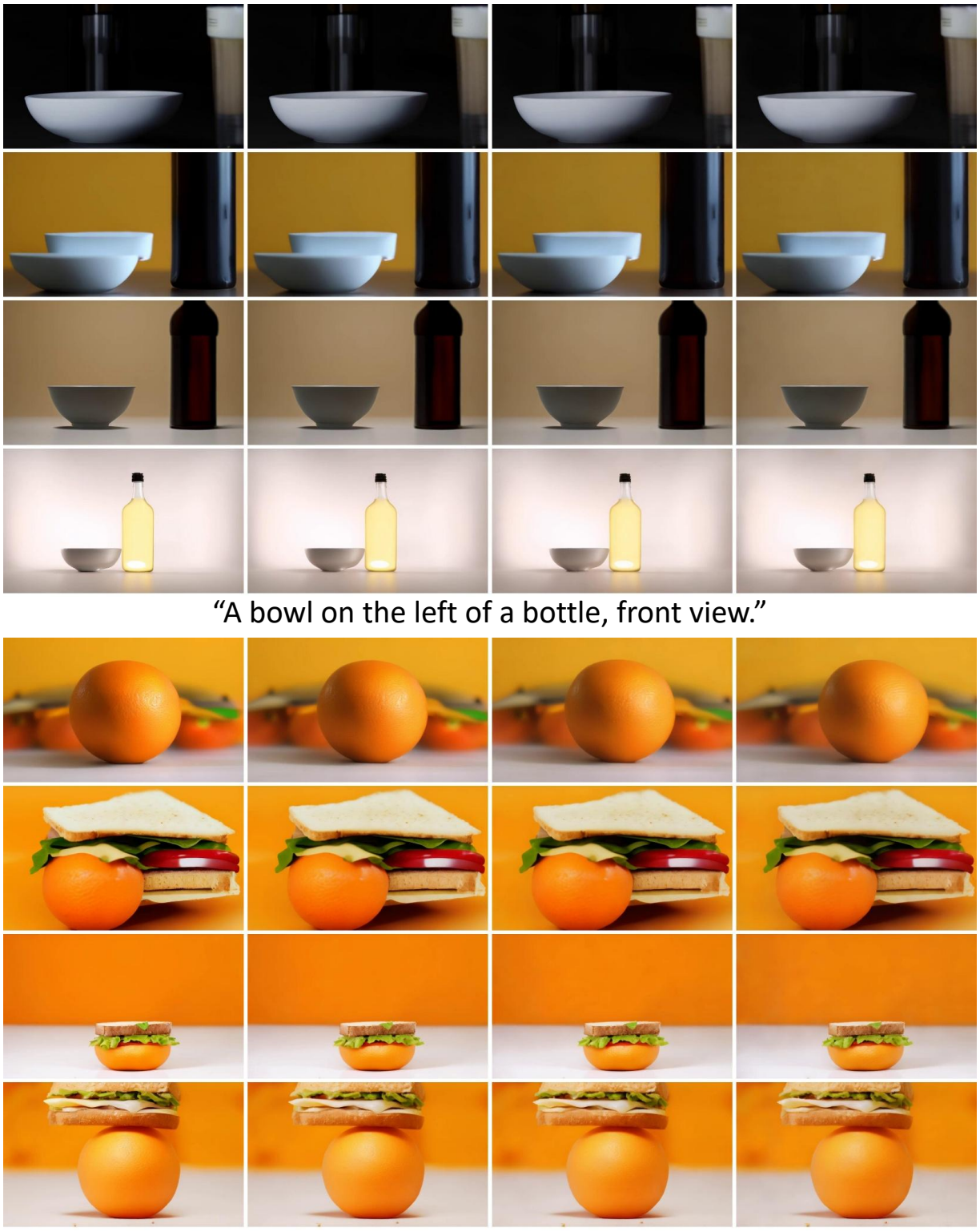


Figure 15. More visual results during TTS process on NOVA. The TTS method improves the alignment between the spatial relationships of objects in videos and the corresponding text prompt.

References

[1] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025. 1

[2] Ziyu Guo, Renrui Zhang, Chengzhuo Tong, Zhizheng Zhao, Peng Gao, Hongsheng Li, and Pheng-Ann Heng. Can we generate images with cot? let's verify and reinforce image generation step by step. *arXiv preprint arXiv:2501.13926*, 2025. 1

[3] Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong Mu, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. *arXiv preprint arXiv:2410.05954*, 2024. 2

[4] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 1

[5] Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024. 2