# DOLLAR: Few-Step Video Generation via Distillation and Latent Reward Optimization

Supplementary Material

## Table of Contents


7. DOLLAR Method Details 1
- 7.1. Pseudo-code . . . 1
- 7.2. Diffusion Model Training and Inference . . . 1
- 7.3. Implementation Details . . . 2
- 7.4. Student-Teacher Parameterization . . . 2
- 7.5. Derivations . . . 2
- 7.6. Inference Time Analysis . . . 3

8. Reward Model Fine-Tuning 3
- 8.1. Evidence of Fine-tuning Effect . . . 3
- 8.2. Direct Reward Gradient . . . 3
- 8.3. Latent Reward Model For Different Reward Types 4
- 8.4. Latent Reward Model Training . . . 4
- 8.5. Latent Reward Model Fine-tuning . . . 4
- 8.6. Denoising Diffusion Policy Optimization . . . 5

9. Additional Experimental Results 7
- 9.1. Diversity Measure Details . . . 7
- 9.2. Human Evaluation . . . 7
- 9.3. Reward Overoptimization . . . 9
- 9.4. Additional Ablation Study . . . 9
- 9.5. Complete VBench Scores . . . 10

10 Challenges and Discussions 13

11 Visualization 13
- 11.1 More Qualitative Results . . . 13
- 11.2 Comparison of Reward Model Fine-tuning . . . 13
- 11.3 Inference Steps . . . 13
- 11.4 Diversity . . . 13
- 11.5 Prompt Length . . . 14
- 11.6 Sampling with Various Styles and Motions . . . 14


## 7. DOLLAR Method Details

### 7.1. Pseudo-code

The pseudo-code of our DOLLAR method is displayed as Alg. 1

**Algorithm 1** Training procedure of DOLLAR.

**Input:** Pretrained teacher model $v_{\theta'}$ by $\mathcal{L}_{\text{CV}}$ Eq. (13), pretrained encoder and decoder, dataset $\mathcal{D}=\{(c,i)\}$
**Output:** Distilled student few-step generator $G_\theta$.
`//Initialize student and fake score model from teacher`
$\theta \leftarrow \theta'$, $\theta_{\text{fake}} \leftarrow \theta'$
**while** train **do**
  Sample batch $(c,i)\sim\mathcal{D}$, encode $x \leftarrow \text{Encoder}(i)$
  `//Update the generator with distillation`
  $\hat{x} \leftarrow G_\theta(c,\varepsilon), \varepsilon\sim\mathcal{N}(0,\mathbf{I})$
  Uniformly sample $t_n$, forward diffusion $x_{t_{n+m}} \leftarrow F(x,t_{n+m})$
  $\mathcal{L}_{\text{G}} = \mathcal{L}_{\text{VSD}}(\theta;\theta',\theta_{\text{fake}},\hat{x},c) + \eta_1 \mathcal{L}_{\text{CD}}(\theta;\theta',x_{t_{n+m}},c)$ `//VSD by Eq. (8), CD by Eq. (4)`
  $G_\theta \leftarrow \text{GradientDescent}(\theta,\mathcal{L}_{\text{G}})$
  `//Update fake score model`
  Uniformly sample $t$, forward diffusion $x_t \leftarrow F(\hat{x},t)$
  $\theta_{\text{fake}} \leftarrow \text{GradientDescent}(\theta_{\text{fake}},\mathcal{L}_{\text{CV}}(x_t))$ `//Eq. (13)`
  `//Train latent reward model`
  Merge batch $\tilde{x} = x \bigcup \hat{x}$
  $\mathcal{R}^l_\phi \leftarrow \text{GradientDescent}(\phi,\mathcal{L}_{\text{LRM}}(\phi;\tilde{x},\mathcal{R}))$ `//Eq. (10)`
  `//Update the generator with latent reward fine-tuning`
  $G_\theta \leftarrow \text{GradientDescent}(\theta,\mathcal{L}_{\text{FT}}(\theta;\hat{x},\mathcal{R}^l))$ `//Eq. (11)`
**end while**

### 7.2. Diffusion Model Training and Inference

**Conjugate Prediction Objective.** Instead of applying noise prediction in previous work [17, 46] and the standard velocity prediction objective as in Instaflow [33], we apply a *conjugate* velocity prediction objective:

$$\mathcal{L}_{\text{CV}}(\theta) = \mathbb{E}_{x_0\sim q(x_0),\varepsilon\sim\mathcal{N}(0,\mathbf{I}),t}\left[||v_\theta(x_t,t)-(x_0-\varepsilon)||_2^2\right] \quad (13)$$

with the sample $x_t$ being diffused along the diffusion trajectory according to the schedule defined as Eq. (1). The model is parameterized to predict velocity $v_t$ on RF trajectory at each timestep $t$, with a constant target $(x_0-\varepsilon)$ (we take a reverse here as opposed to standard RF for notation clarity), as visualized in Fig. 7. The predicted velocity $v_\theta(x_t,t)=v_t^y$ is the velocity on RF as the conjugate point $y_t$ of sample $x_t$

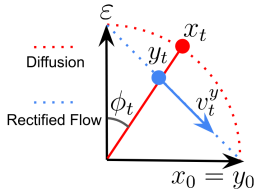



Figure 7. Demonstration of the conjugate velocity prediction: relationship of $v$-prediction for diffusion and rectified flow.

along the diffusion trajectory. This is practically easier to learn compared to the time-varying velocity as in Eq. (3).

**Why Conjugate Prediction Objective?** The rectified flow loss a commonly applied generative modeling objective in StableDiffusion3 [11], MovieGen [41], Hunyuan Video [26], etc. While prior video generation models apply DDPM noise schedule, direct velocity prediction along the diffusion trajectory following Eq. (2) has a time-varying target as Eq. (3), which is practically harder to learn compared with the constant velocity objective in Eq. (13). However, the standard rectified flow [29] target is not along the diffusion trajectory, which requires to model the trajectory with a different noise schedule other than the DDPM one. The conjugate objective generates variance-preserving noise samples along the diffusion trajectory while predicts a constant velocity, through the conjugate relationship between the diffusion trajectory and rectified flow, therefore it is easier to learn.

**Inference.** After training, the reverse diffusion process follows:

$$\begin{aligned} x_{t-1} &:= \text{Denoise}(x_t, t, \theta) \\ &= (\sqrt{\bar{\alpha}_{t-1}} - \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}})\hat{x}_0 + \frac{\sqrt{1-\bar{\alpha}_{t-1}}}{\sqrt{1-\bar{\alpha}_t}} x_t + \sigma_t \varepsilon \end{aligned} \quad (14)$$

with $\hat{x}_0 = \frac{x_t + \sqrt{1-\bar{\alpha}_t} v_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t}}$ as the predicted original samples. Proofs see Appendix 7.5.

## 7.3. Implementation Details

Both the teacher and student models are trained on internal image and video datasets with text captioning, comprising approximately $O(100\text{M})$ images and $O(1\text{M})$ videos. The teacher model employs standard DDPM settings with 1000 sampling steps: $t \in [1, \ldots, 1000]$. For inference, the teacher model utilizes DDIM sampling to generate high-quality samples in 50 steps, with $t_n \in [19, 39, \ldots, 999]$. After distillation, the student model adopts a default 4-step sampling protocol, as in previous work [74], using timesteps $[249, 499, 749, 999]$. Additionally, we explore 1-step ($[999]$) and 2-step ($[499, 999]$) generation configurations for the student model in Sec. 5.5. Consistency distillation (CD, discussed in Sec. 3.2) follows a DDIM schedule with $N = 50$ steps, as implemented in LCM [36]. For teacher inference, we apply classifier-free guidance (CFG) [16] augmentation with a weight of $w = 7.5$ in CD as specified in Eq. (6) and $w = 3.5$ for the real score network in VSD, The fake score network and distilled student inference do not employ CFG. In the VSD loss, we adhere to the update ratio as 5 for the fake score update over generator update, as suggested in previous work [74], to ensure training stability. All experiments are conducted with a batch size of 1 per GPU due to the large model size and limited VRAM, utilizing 8 GPUs in parallel for each run. All student models are distilled up to $4 \times 10^4$ iterations, with moderate model selection. Video samples are generated with 128 frames at a resolution of $192 \times 320$. We set $\beta_{\text{CD}} = 0.5$ and $\beta_{\text{FT}} = 1.0$ to roughly match the magnitude of each loss without more fine-grained balance. This simple strategy is sufficient that the dominance of one loss over others does not appear throughout our experiments, which verifies the robust training of our framework at large scale.

To reduce VRAM occupancy on GPUs, we employ gradient checkpointing and fully sharded data parallel (FSDP) [78], enabling sharding of model weights and gradients across GPUs in a data-parallel fashion. Additionally, we utilize mixed precision training with the Bfloat16 data type. For fine-tuning with LRMs, we apply gradient accumulation over 7 steps to stabilize training due to the small batch size (=1) used.

## 7.4. Student-Teacher Parameterization

There are two different ways for student-teacher parameterization: **homogeneous** and **heterogeneous**.

For homogeneous student-teacher parameterization, the networks of student and teacher both follow the same variable prediction, *i.e.*, $v$-prediction in our setting, with a transformation:

$$x_\theta(x_t, t) = \frac{x_t + \sqrt{1-\bar{\alpha}_t} v_\theta(x_t, t)}{\sqrt{\bar{\alpha}_t} + \sqrt{1-\bar{\alpha}_t}} \quad (15)$$

which is proved in Appendix. The student model $v_\theta$ will be initialized from teacher model $v_{\theta'}$ at the beginning of distillation.

For heterogeneous student-teacher parameterization, the student network can directly predict $x_\theta$ without leveraging Eq. (15). For the best usage of teacher model in student distillation, we adopt the homogeneous parameterization by default.

Experimental comparisons for two different parameterizations see Appendix 9.4.2.

## 7.5. Derivations

### 7.5.1. Proof of Eq. (14)

We start from the forward diffusion process of DDPM [17]. The distribution of one-step diffusion process $q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{\alpha_t} x_{t-1}, (1-\alpha_t)\mathbf{I})$ can be equivalently written as:

$$x_t = \sqrt{\alpha_t} x_{t-1} + \sqrt{1-\alpha_t}\varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \mathbf{I}) \quad (16)$$

with $t \in [T]$.

By chain rule, we have

$$x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon \quad (17)$$

with $\bar{\alpha}_t = \Pi_{i=1}^t \alpha_i$. Equivalently, we have $x_t \sim q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t} x_0, (1-\bar{\alpha}_t)\mathbf{I})$. This equation is also used to predict:

$$\hat{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} x_t - \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}} \epsilon_\theta \quad (18)$$

which is called the Tweedie's formula. $\epsilon_\theta$ is the approximated prediction of $\varepsilon$ with a parameterized model by $\theta$.

Proof of the denoising function Eq. (14) in reverse diffusion process is as follows:

$$
\begin{aligned}
x_{t-1} &= \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\epsilon_\theta + \sigma_t\varepsilon \\
&= \sqrt{\bar{\alpha}_{t-1}}\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\left(\frac{1}{\sqrt{1-\bar{\alpha}_t}}x_t\right. \\
&\left. - \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\hat{x}_0\right) + \sigma_t\varepsilon \\
&= \left(\sqrt{\bar{\alpha}_{t-1}} - \sqrt{1-\bar{\alpha}_{t-1}-\sigma_t^2}\frac{\sqrt{\bar{\alpha}_t}}{\sqrt{1-\bar{\alpha}_t}}\right)\hat{x}_0 \\
&+ \frac{\sqrt{1-\bar{\alpha}_{t-1}}}{\sqrt{1-\bar{\alpha}_t}}x_t + \sigma_t\varepsilon
\end{aligned}
$$

with the first equation follows the posterior sampling in DDIM paper [54]. The second is to plug in the Tweedie's formula. We have the variance term $\sigma_t^2 = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}$.

#### 7.5.2. Proof of Eq. (15)

Following the Instaflow objective as Eq. (13), the network directly predicts $v_\theta$, to approximate the target velocity $\tilde{v}^y$ along the rectified flow (RF) trajectory, as the difference of the clean sample and Gaussian noise:

$$
v_\theta \approx \tilde{v}^y = x_0 - \varepsilon \tag{19}
$$

Since the RF sample $y_t$ is a scaled version of diffusion sample $x_t$ as:

$$
y_t = \frac{x_t}{\sqrt{\bar{\alpha}_t}+\sqrt{1-\bar{\alpha}_t}} = \gamma_t x_0 + (1-\gamma_t)\varepsilon, \tag{20}
$$

$$
\gamma_t = \frac{\sqrt{\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}+\sqrt{1-\bar{\alpha}_t}}, \tag{21}
$$

which satisfies $y_0 = x_0$.

Given the velocity prediction $v_\theta$, we can derive the prediction of original sample $x_\theta$ as following, by replacing $x_0$ with prediction $x_\theta$ in Eq. (19) and (20):

$$
\gamma_t x_\theta = y_t - (1-\gamma_t)(x_\theta - v_\theta^y) \tag{22}
$$

$$
x_\theta = y_t + (1-\gamma_t)v_\theta = y_t + \frac{\sqrt{1-\bar{\alpha}_t}}{\sqrt{\bar{\alpha}_t}+\sqrt{1-\bar{\alpha}_t}}v_\theta \tag{23}
$$

which concludes the proof.

### 7.6. Inference Time Analysis

Here, we provide a detailed analysis of the inference time experimental results presented in the main paper, as shown in Sec. 5.4 Tab. 4. Absolute time costs are not reported, as they are influenced by hardware-specific factors and inference configurations such as batch size and the number of GPUs used. Instead, relative time consumption is emphasized as a more reliable metric for cross-configuration comparisons.

Notably, the relationship between diffusion sampling time and the number of sampling steps is not strictly linear. For example, the first diffusion sampling step accounts for only $0.33\%$ of the total inference time, making it approximately 6.2 times faster than subsequent steps. This discrepancy is likely due to the faster inference process for initial Gaussian noise inputs or the relatively low hardware cache occupation during early inference stages.

Furthermore, the difference between the total inference time and the diffusion sampling time includes additional costs for text preprocessing and encoding, as well as decoding from the latent space back to the original pixel space. These processes collectively account for approximately $7\%$ of the total inference time.

## 8. Reward Model Fine-Tuning

### 8.1. Evidence of Fine-tuning Effect

As shown in Fig. 8, the samples generated after reward model tuning can have a substantial difference from the original training samples in dataset (left in the figure), for aspects of aesthetic quality, lighting condition, colors, etc. More such examples are provided in Sec. 11.2 for comparison of models trained with or without reward fine-tuning.

Figure 8. Visualization of samples in training dataset (left) and samples generated with reward tuning using HPSv2 reward (right).

### 8.2. Direct Reward Gradient

In this section, we discuss in details why the direct reward gradient methods like ReFL [72] and DRaFT [8], cannot fit into the memory efficiently.

Take the HPSv2 [67] model as an example. It applies fine-tuned version of ViT-H/14 variant of CLIP model, which contains 32 image transformer layers and 24 text transformer layers, each with 16 heads. This constitutes a total of 633 million parameters. Even with FP16 data type, the model weights will occupy 1.25 GB memory. Even for a batch size of 1, the input video tensor of size $(128,3,192,320)$ occupies about 6 GB memory for forward inference only. Backpropagation through the model will drastically increases the memory cost due to gradients storage. Moreover, the memory occupancy roughly

scales linearly with the batch size, making it hard to scale up. PickScore [25] with CLIP-H model has the similar memory cost in practice. Comparison of parameter numbers and memory costs for reward models and LRMs is shown in Tab. 1. If we take sub-sampling in videos to extract frames for reward optimization, the backward memory (VRAM) cost for different number of frames $H$ is shown in Tab. 8. It indicates that even with frame sub-sampling, the memory cost can still be too large to afford in video model training.

Table 8. Backward memory (VRAM) costs for HPSv2, PickScore reward models with different numbers ($H$) of image ($192 \times 320$) frames.

| Model | $H=12$ | $H=24$ | $H=64$ | $H=128$ |
|---|---|---|---|---|
| HPSv2/PickScore | 12.373 GB | 20.577 GB | 48.413 GB | >90 GB |

Given the diffusion modeling in latent space, direct reward gradient methods will also need to backpropagate the gradients from reward model through the large pretrained decoder, this further increases the burden on memory usage.

### 8.3. Latent Reward Model For Different Reward Types

The proposed latent reward model method is compatible with any type of reward metrics as introduced previously, regardless of its differentiability and input formats. Here we consider several types of commonly used reward metrics: image reward, text-image reward, video reward and text-video reward. For each category, we provide examples and explain how LRM, with its diverse architectures, supports these metrics. A summary of this compatibility is provided in Tab. 9, with further details outlined below:

- Image reward: $\mathcal{I} \to \mathbb{R}$.
  The LRM is $\mathcal{R}^l_\phi(x) : \mathcal{X} \to \mathbb{R}, x = \text{Encode}(i), i \in \mathcal{I}$. It has the image backbone as a 2D convolutional neural network (CNN).
  Examples include LAION aesthetic quality [51], JPEG compressibility [3].
- Text-image reward: $\mathcal{C} \times \mathcal{I} \to \mathbb{R}$.
  The LRM is $\mathcal{R}^l_\phi(x,c) : \mathcal{X} \times \mathcal{C} \to \mathbb{R}, x = \text{Encode}(i), i \in \mathcal{I}$. It has the image backbone as a 2D CNN and text embedding $e_c$ as inputs, with a cross-attention module for mixing image features $e_x$ and text features $e_c$: $\text{Softmax}(\mathbf{Q}(e_x) \cdot \mathbf{K}(e_c)^\top) \cdot \mathbf{V}(e_c)$.
  Examples include human preference score (HPS) [67, 68], ImageReward [72], PickScore [25].
- Video reward: $\mathcal{I}^H \to \mathbb{R}$ where $H$ is the number of frames in each video.
  The LRM can be either (1). $\mathcal{R}^l_\phi(x) : \mathcal{X} \to \mathbb{R}, x = \text{Encode}(i), i \in \mathcal{I}$ using a 2D CNN image backbone with average frame reward $\frac{1}{H}\sum_{k=1}^{H} \mathcal{R}^l_\phi(x_k)$ as video reward or (2). $\mathcal{R}^l_\phi(x_1,...,x_H) : \mathcal{X}^H \to \mathbb{R}$ using a 3D CNN as video backbone.
  Examples include 7 quality scores in VBench (subject consistency, background consistency, motion smoothness, etc).
- Text-video reward: $\mathcal{C} \times \mathcal{I}^H \to \mathbb{R}$.
  The LRM can be either (1). $\mathcal{R}^l_\phi(x,c) : \mathcal{X} \times \mathcal{C} \to \mathbb{R}$ using a 2D CNN image backbone with average frame reward $\frac{1}{H}\sum_{k=1}^{H} \mathcal{R}^l_\phi(x_k,c)$ as video reward or (2). $\mathcal{R}^l_\phi(x_1,...,x_H,c)$: $\mathcal{X}^H \times \mathcal{C} \to \mathbb{R}$ using a 3D CNN as video backbone, with additional text embedding $e_c$ as inputs, and cross-attention for mixing image features $e_x$ and text features $e_c$: $\text{Softmax}(\mathbf{Q}(e_x) \cdot \mathbf{K}(e_c)^\top) \cdot \mathbf{V}(e_c)$.
  Examples include ViCLIP [64], VideoScore [15], InternVideo2 [65] and 9 semantic score metrics in VBench (object class, human action, color, etc).

**Architecture Details.** The image only LRM $\mathcal{R}^l_\phi(x)$ has architecture detailed in Tab. 10. The text-image LRM $\mathcal{R}^l_\phi(x,c)$ has architecture detailed in Tab. 11. For video LRM and text-video LRM, we apply the same architectures with frame averaging in our experiments.

**Discussions.** The latent reward model can be utilized in two ways: it can either be pretrained or trained concurrently with the student model during fine-tuning, as demonstrated in our experiments. Furthermore, this approach can also be extended to fine-tune the teacher model. Alternatively, one could bypass the reward model in pixel space entirely and directly employ a latent reward model from the outset. However, we argue that such an approach is likely to be limited to specific fixed latent spaces and may lack generalizability across models. This is because pretrained encoder-decoder models can vary significantly and often do not share a unified latent space, particularly in existing image and video models.

### 8.4. Latent Reward Model Training

Fig. 9 and Fig. 10 show the learning curves of latent reward models (LRMs) with two original pixel-space rewards HPSv2 and PickScore, respectively, during the distillation process. The loss for training is VSD+LRM. Left figure displays the MSE loss for LRM prediction against the ground-truth pixel-space reward value. Right figure displays the LRM predicted reward values $\mathcal{R}^l_\phi(x_0,c)$ and ground truth reward values $\mathcal{R}(x_0,c)$ on training samples from the dataset $x_0 \sim \mathcal{X}$. This demonstrates that the LRM achieves rapid convergence within 2000–3000 training iterations, even when operating in a significantly lower-dimensional latent space. The small approximation errors ensure the effectiveness of fine-tuning with learned LRM.

### 8.5. Latent Reward Model Fine-tuning

Fig. 11 displays the predicted reward values $\mathcal{R}^l_\phi(\hat{x}_0,c)$ with LRM for generated samples ($\hat{x}_0 \sim \mathcal{X}'$, by Eq. (15)) during the distillation process with VSD+LRM loss, for two reward metrics

Table 9. Summary of latent reward models for different pixel-space reward metrics.

| **Reward Type** | **LRM Function** | **Architecture** | **Examples** |
|---|---|---|---|
| Image Reward | $\mathcal{R}^l_\phi(x): \mathcal{X} \to \mathbb{R}$ | 2D CNN backbone | LAION aesthetic [51], JPEG compressibility [3] |
| Text-Image Reward | $\mathcal{R}^l_\phi(x,c): \mathcal{X} \times \mathcal{C} \to \mathbb{R}$ | 2D CNN + text embedding, cross-attention | HPS [67, 68], ImageReward [72], PickScore [25] |
| Video Reward | $\mathcal{R}^l_\phi(\boldsymbol{x}): \mathcal{X}^H \to \mathbb{R}$ | 2D CNN with average frame reward, or 3D CNN backbone | VBench quality scores (subject consistency, motion smoothness, etc) |
| Text-Video Reward | $\mathcal{R}^l_\phi(\boldsymbol{x},c): \mathcal{X}^H \times \mathcal{C} \to \mathbb{R}$ | 2D CNN with average frame reward, or 3D CNN backbone, + text embedding, cross-attention | ViCLIP [64], VideoScore [15], InternVideo2 [65], VBench semantic scores (object class, human action, color, etc) |

Table 10. Architecture of the image latent reward model

| Layer | Input Shape | Output Shape | Kernel Size | Stride | Padding | Number of Parameters |
|---|---|---|---|---|---|---|
| **Input** | (batch, C, H, W) | | | | | |
| **Conv2d + GroupNorm + SiLU** | (batch, C, H, W) | (batch, 128, 6, 10) | 4x4 | 4 | 1 | 24,704 |
| **Conv2d + GroupNorm + SiLU** | (batch, 128, 6, 10) | (batch, 128, 3, 5) | 3x3 | 2 | 1 | 147,584 |
| **AdaptiveAvgPool2d** | (batch, 128, 3, 5) | (batch, 128, 1, 1) | - | - | - | 0 |
| **Conv2d** | (batch, 128, 1, 1) | (batch, 128, 1, 1) | 1x1 | 1 | 0 | 16,512 |
| **Flatten** | (batch, 128, 1, 1) | (batch, 128) | - | - | - | 0 |
| **Linear** | (batch, 128) | (batch, 1) | - | - | - | 129 |
| **Total Parameters** | | | | | | 189,441 |

Table 11. Architecture of the text-image latent reward model

| Layer | Input Shape | Output Shape | Kernel Size / Projection | Stride | Padding | Number of Parameters |
|---|---|---|---|---|---|---|
| **Input Image** | (batch, C, H, W) | | | | | |
| **Conv2d + GroupNorm + SiLU** | (batch, C, H, W) | (batch, 128, 6, 10) | 4x4 | 4 | 1 | 24,704 |
| **Conv2d + GroupNorm + SiLU** | (batch, 128, 6, 10) | (batch, 128, 3, 5) | 3x3 | 2 | 1 | 147,584 |
| **AdaptiveAvgPool2d** | (batch, 128, 3, 5) | (batch, 128, 1, 1) | - | - | - | 0 |
| **Conv2d** | (batch, 128, 1, 1) | (batch, 128, 1, 1) | 1x1 | 1 | 0 | 16,512 |
| **Flatten (Image Features)** | (batch, 128, 1, 1) | (batch, 128) | - | - | - | 0 |
| **Input Text** | (batch, L, D) | | | | | |
| **Text MLP** | (batch, L, D) | (batch, 256, 128) | - | - | - | 524,544 |
| **Average Pooling (Text Features)** | (batch, 256, 128) | (batch, 128) | - | - | - | 0 |
| **Query Projection (Linear)** | (batch, 128) | (batch, 128) | - | - | - | 16,512 |
| **Key Projection (Linear)** | (batch, 128) | (batch, 128) | - | - | - | 16,512 |
| **Value Projection (Linear)** | (batch, 128) | (batch, 128) | - | - | - | 16,512 |
| **Attention Mechanism (Softmax)** | (batch, 1, 1) | (batch, 1, 1) | - | - | - | 0 |
| **Final Linear (Output Layer)** | (batch, 128) | (batch, 1) | - | - | - | 129 |
| **Total Parameters** | | | | | | 763,009 |

HPSv2 and PickScore, respectively. The horizontal dashed lines are the average reward values of the samples in training dataset. For HPSv2, the reward values of generated samples surpass the training data quickly with the LRM fine-tuning. For PickScore, the reward values of generated samples also gradually increase to be close to the training data.

## 8.6. Denoising Diffusion Policy Optimization

Denoising Diffusion Policy Optimization (DDPO) [3] serves as the baseline for comparison with our proposed LRM method. DDPO applies the REINFORCE algorithm to optimize the diffusion model by treating the diffusion process as a MDP. It requires to estimate the log-probabilities for the sample at all diffusion steps, which are then summed over and weighted by the

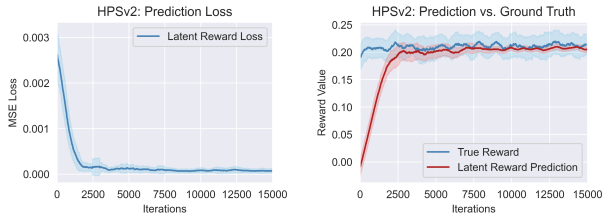



Figure 9. The learning process of LRM with HPSv2 reward.

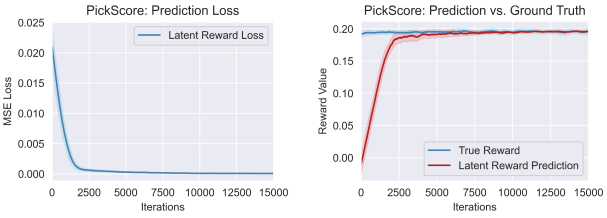



Figure 10. The learning process of LRM with PickScore reward.

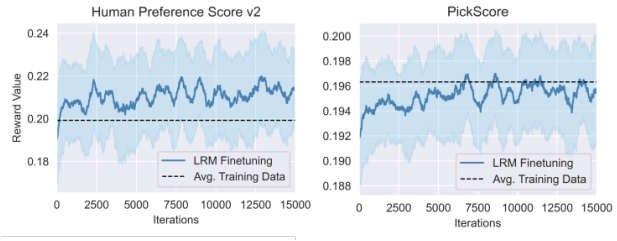



Figure 11. Latent reward model fine-tuning process under reward metrics HPSv2 and PickScore.

final reward as the optimization objective. Considering memory constraints, our method is suited for few-step sampling models or configurations with gradient truncation along the diffusion trajectory. In our experiments, memory limitations prevent log-probability estimation over more than 2 steps. Therefore, we employ a truncation step of 2 for the student model (*i.e.*, log-probability estimation at timesteps [249, 499]). This truncation approach has been validated in previous work [8, 45]. We apply $\text{DDPO}_{\text{SF}}$ for online policy gradient in our experiments.

By applying the REINFORCE algorithm on denoising process of diffusion models, the $\text{DDPO}_{\text{SF}}$ algorithm follows the score function policy gradient:

$$\nabla_\theta \mathcal{J} = \mathbb{E}[\sum_{t=1}^{T} \nabla_\theta \log p_\theta(x_{t-1}|x_t,c) R(x_0,c)] \tag{24}$$

This is the online version for gradient estimation, which requires to sample $x_{t-1}$ as well as calculating the probabilities $p_\theta(x_{t-1}|x_t,c)$ along the sampling process at the same time, such that the model parameters $\theta$ remain the same for sampling and probability estimation. The update will only take one step to preserve the online estimation property. Original paper [3] also proposes another version for offline policy gradient estimation with importance sampling to allow multi-step updates. As log-probability $\log p_\theta(x_{t-1}|x_t,c)$ needs to be estimated during the sampling process, we cannot take sampling process as Eq. (14), but estimating the posterior mean $\mu_\theta$ and standard deviation $\sigma$ instead:

$$\mu_\theta(x_{t-1};x_t) = \frac{(1-\alpha_t)\sqrt{\bar{\alpha}_t}}{1-\bar{\alpha}_t} x_\theta + \frac{\sqrt{\alpha}_t(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} x_t$$
$$\sigma_t = \sqrt{(1-\alpha_t)\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}} \tag{25}$$

with $x_\theta$ following Eq. (15). $x_{t-1}$ will be sampled from $\mathcal{N}(\mu_\theta(x_{t-1};x_t),\sigma_t)$, with log-probability of the sample as:

$$\log p_\theta(x_{t-1}|\mu_\theta,\sigma,c) = -\frac{1}{2}\left(\frac{(x_{t-1}-\mu_\theta)^2}{\sigma^2} + \log(2\pi\sigma^2)\right) \tag{26}$$

The practical procedure of $\text{DDPO}_{\text{SF}}$ is outlined in Alg. 2. Due to VRAM memory constraints, we employ the REINFORCE policy gradient with truncation, allowing gradient tracking for a maximum of $N=2$ steps during training. Specifically, for a student model with a sampling time sequence $[T,...,t_{\text{min}}] = [999,749,499,249]$, the gradient update steps will only take the last two steps $t_n \in \{499,249\}$, rather than all timesteps. This truncation is used to estimate the log-probabilities of samples at $t_{n-1}$. Here, $\text{Dec}(\cdot)$ represents the pretrained video decoder, while the reward model $R$ operates in the original pixel space. We use .detach() to indicate a stop-gradient function.

**Algorithm 2** DDPO practical procedure.

**Input:** Distilled student model $G_\theta$, dataset $\mathcal{D}=\{(c,i)\}$
**Output:** Fine-tuned student few-step generator $G_\theta$.
**while** train **do**
  `//Sample from random noise along entire diffusion trajectory`
  $x_T \leftarrow \varepsilon \sim \mathcal{N}(0,\mathbf{I})$
  **for** $t_n \in [T,...,t_{\text{min}}]$ **do**
    Get posterior Gaussian $(\mu_\theta,\sigma)$ with $v_\theta(x_{t_n}.\text{detach}(),t_n)$ `//Eq.` (25)
    Sample $x_{t_{n-1}} \sim \mathcal{N}(\mu_\theta,\sigma\mathbf{I})$
    Estimate $\log p_\theta(x_{t_{n-1}}|x_{t_n},c)$ `//Eq.` (26)
  **end for**
  Get reward $R = R(\text{Dec}(\hat{x}_0),c).\text{detach}()$
  `//REINFORCE policy gradient with truncation`
  $\mathcal{L}_{\text{DDPO}_{\text{SF}}} = -\sum_n^N \log p_\theta(x_{t_{n-1}}.\text{detach}()|x_{t_n},c)\cdot R$
  $G_\theta \leftarrow \text{GradientDescent}(\theta,\mathcal{L}_{\text{DDPO}_{\text{SF}}})$
**end while**

**Learning Curves.** The training process of VSD+DDPO for two reward metrics are shown in Fig. 12. The learning curve shows the reward values $\mathcal{R}(x_0,c)$ for generated samples $\hat{x}_0$


Figure 12. Reward model fine-tuning process with VSD+DDPO under reward models HPSv2 and PickScore.

through iterative denoising along the full diffusion trajectories, during the fine-tuning process.

The learning curves of DDPO are not directly comparable to those of the LRM methods shown in Fig. 11. This difference arises because DDPO samples across the entire diffusion trajectory to obtain the predicted $\hat{x}_0$ for reward evaluation, whereas LRM performs one-step prediction using $x_\theta = \frac{x_t+\sqrt{1-\bar{\alpha}_t}v_\theta^w(x_t,t)}{\sqrt{\bar{\alpha}_t}+\sqrt{1-\bar{\alpha}_t}}$, as defined in Eq. (15). Consequently, the LRM samples tend to be noisier and yield lower rewards during fine-tuning. A fair comparison involves evaluating the rewards of the final generated samples after the model fine-tuning, as presented in Tab. 5 of main paper.

## 9. Additional Experimental Results

### 9.1. Diversity Measure Details

For the diversity experiments in main paper Sec. 5.4, the videos are generated using VBench long prompts, with five videos produced for each prompt. To evaluate diversity, we randomly sample 500 prompts, resulting in a total of 2,500 videos. For assessing video sample diversity within a single prompt, we define the diversity metric as:

$$\text{Diversity} = \frac{1}{K}\sum_{k=1}^{K}\text{Vendi}\left([f_1^k,\ldots,f_n^k]\right) \tag{27}$$

For images, the function $\text{Vendi}(\cdot)$ quantifies the diversity of a set of image features, which can be derived either from raw pixel vectors or embeddings obtained via the Torchvision Inception v3 model. For videos, we uniformly extract $K$ keyframes with an equal spacing of 20 frames between consecutive keyframes. We then calculate $\text{Vendi}\left([f_1^k,\ldots,f_n^k]\right)$ for the 5 videos corresponding to the same frame index $k$. Finally, the diversity measure for a given prompt is obtained by averaging the Vendi values across all $K$ frames. The mean and standard deviations of this metric are computed and reported across all prompts to evaluate video diversity, as shown in Tab. 3 of main paper.

### 9.2. Human Evaluation

**Human Evaluation Details.** Fig. 13 displays the user interface for human evaluation experiments. The four choices include visual quality, text-video alignment, motion and general preference, which correspond to the four reported metrics in Fig. 14. For the pairwise comparison of methods, the videos are randomly sampled from 4730 videos with 946 VBench long prompts, with 5 videos generated for each prompt under different random seeds. The videos are all displayed at a resolution of $192\times320$ with 128 frames for our methods. For a fair comparison, videos for the baseline method Gen-3 ($768\times1280$) are resized to $192\times320$. Each pair of videos requires approximately 20–30 seconds for evaluation. To prevent positional bias, the left and right placement of the videos is randomly shuffled for each evaluation session.

**Human Evaluation Results.** We conduct 6 rounds of human evaluation on sampled videos with different methods, comparing models under the following settings:

- VSD+LRM with HPSv2 as reward model versus VSD method, to verify the effectiveness of LRM for fine-tuning.
- VSD+LRM versus VSD+DDPO, both with HPSv2 as the reward model, to compare the LRM and DDPO methods for reward fine-tuning.
- VSD+LRM with HPSv2 reward versus PickScore reward, to testify the effectiveness of two reward models.
- VSD+CD+LRM with HPSv2 reward versus Gen-3 model results, to compare our distilled models with one of the best present models in Tab. 2 according to VBench.
- VSD+CD+LRM with PickScore as reward model versus the teacher model.
- VSD+CD+LRM with HPSv2 as reward model versus the teacher model.

The results for above 6 experiments are summarized in Fig. 14. Each value indicates the winning rate, with the equal performance option excluded.

**Discussions.** In the comparison of VSD+CD+LRM with PickScore versus the teacher model, human evaluation results indicate that the student underperforms the teacher in text-video alignment, motion and general preference, although it has a much higher score in VBench evaluation (82.37 vs. 80.25) as Tab. 2. Specifically, the semantic score in VBench is 77.90 for the student and 73.71 for the teacher, while human evaluation arrives at the opposite conclusion. This discrepancy highlights a mismatch between VBench and human evaluation metrics, posing a challenge in accurately assessing video generation quality. Our empirical findings suggest that humans tend to reject videos exhibiting subtle flaws such as shape distortions, unnatural motions, or other elements that appear less natural or physically realistic. Humans are highly sensitive to these imperfections, which influence their preference. By contrast, VBench metrics, primarily based on pretrained image understanding models, are more influenced by factors such as coloring, lighting, aesthetics, and imaging quality, while being less sensitive to the naturalness and physical realism of videos. Measuring physical realism directly from pixels remains a challenge in general. We

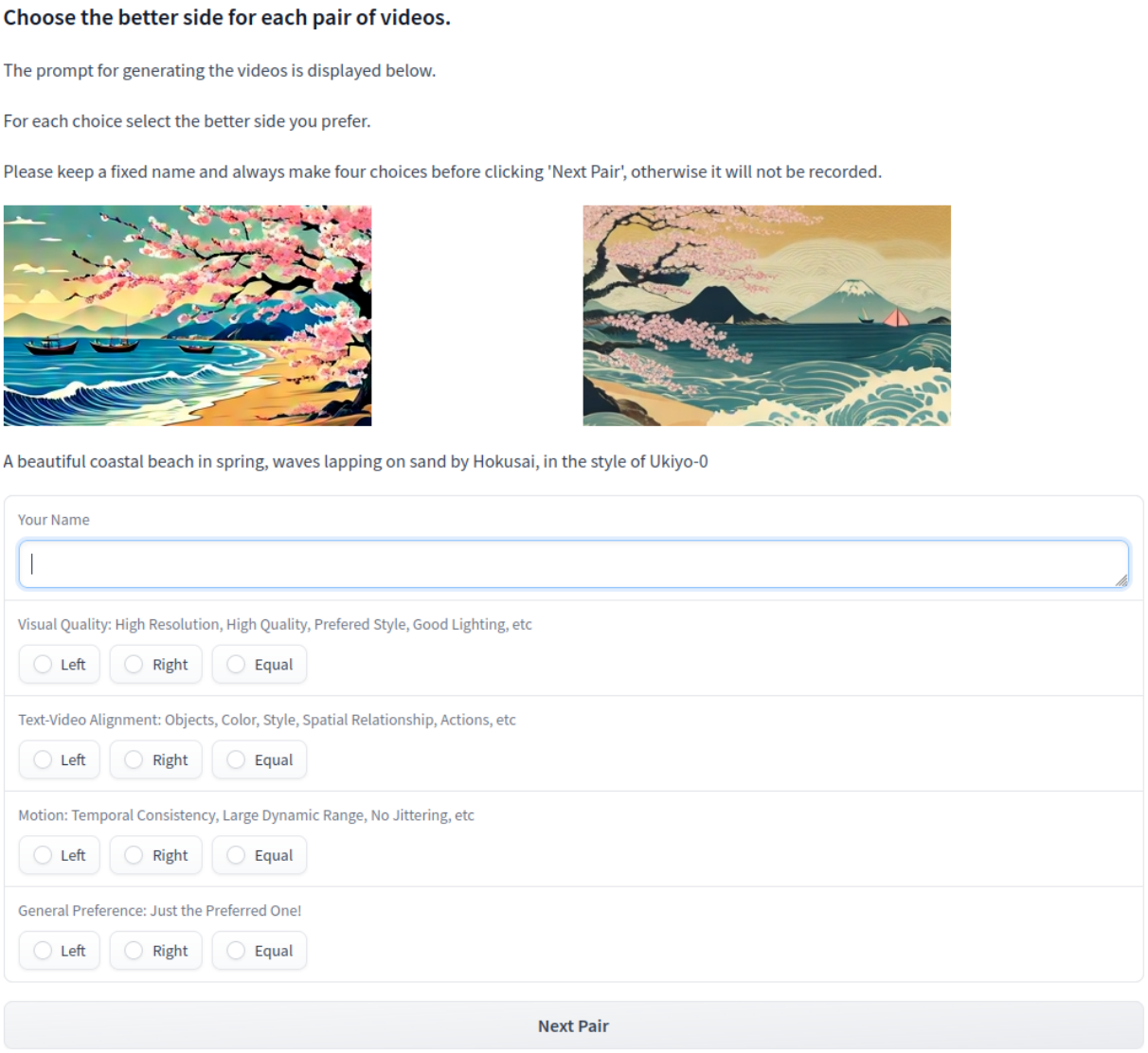



Figure 13. The user interface for human evaluation experiments.

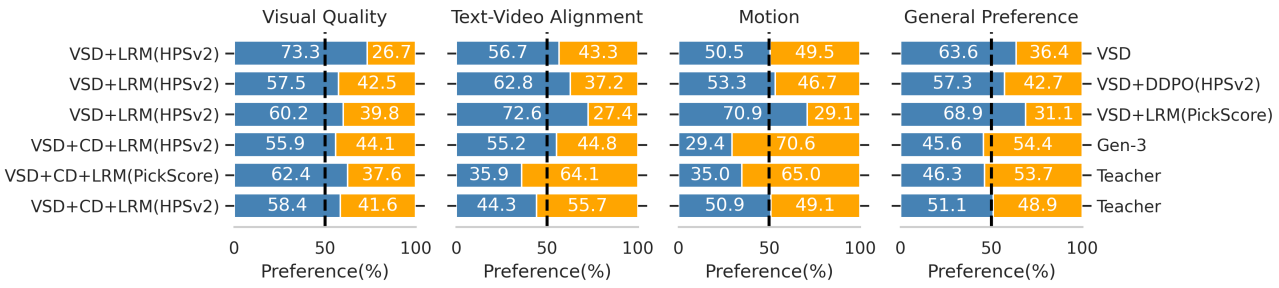



Figure 14. Human evaluation results over four independent metrics: visual quality, text-video alignment, motion and general preference, for six pair of models.

hypothesize that this difference contributes to the observed divergence between VBench scores and human preferences in our experiments.

### 9.3. Reward Overoptimization

We conduct additional experiments with latent reward fine-tuning on some VBench video-reward metrics, such as dynamic degree, and image-reward metrics, such as JPEG compressibility [3]. Fig. 15 shows the progress of the latent reward model fine-tuning with the dynamic degree metric in VBench. As the dynamic degree score increases, the generated samples begin to exhibit a "noise flow" effect that deteriorates the imaging quality. Despite this, the dynamic degree score can rise as high as 0.97, compared to the average score of 0.75 in the training data. These findings highlight the trade-off between optimizing for specific metrics and preserving overall visual quality. Fig. 16 visualize the training data and generated samples during reward fine-tuning.


Figure 15. Latent reward model fine-tuning process for dynamic degree.

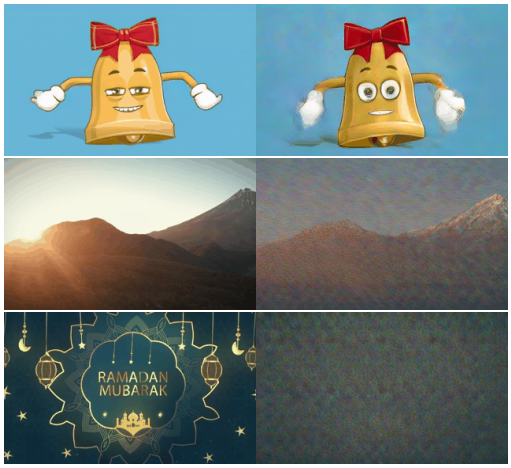



Figure 16. Reward model fine-tuning with dynamic degree: (left) ground-truth training samples, (right) generated samples. The noise level increases as training goes longer (from top to bottom).

As noted in [3], reward-based optimization is prone to overoptimization, stemming from the divergence between the reward maximization objective and the distribution matching objective used during pre-training. In our video generation experiments, this issue is even more pronounced, with overoptimization sometimes occurring within just a few hundred iterations of fine-tuning. This rapid onset is likely exacerbated by the sample variance inherent in stochastic gradient descent when using a small batch size.

Simply reducing the learning rate or loss weight to mitigate overoptimization is not an ideal solution, as it significantly increases the training time and does not effectively address the core issue. This highlights the need for alternative strategies to balance reward maximization and distribution preservation during fine-tuning.

### 9.4. Additional Ablation Study

#### 9.4.1. VBench Prompt Length

During our evaluation, we observed that the standard prompt suite in VBench includes very short prompts, such as "a bus," which lack context or motion descriptions. This does not align well with the text-video data distribution used to train our model, where most images and videos are accompanied by richly detailed captions to enhance the model's semantic capabilities. Our findings indicate that pretrained T2V models often exhibit a bias toward prompt length, performing better with longer, more descriptive prompts. To address this, VBench incorporates the prompt optimization technique introduced in CogVideoX [73], which utilizes GPT-4o [1] to extend the short prompts into more descriptive "long prompts" while preserving their original meanings. We refer to these as "long prompts", distinguishing them from the original "short prompts".

The VBench score comparison for long prompts and short prompts are summarized in Tab. 15. The evaluation includes five models:

- Teacher model
- VSD model with 1-step inference (VSD1)
- VSD model with 4-step inference (VSD4)
- Model distilled with VSD and CD joint loss, using CD denoising step $m=1$ (VSD4+CD1)
- Model distilled with VSD and LRM joint loss, using PickScore as reward function (VSD4+LRM)

Each pair of comparison is conducted using exactly the same model and evaluation protocol, differing only in prompt lengths. Most models achieve higher total scores when short prompts are replaced with long prompts, except for VSD1, which verifies our hypothesis on prompt length bias. According to this observation, we adopt the long prompt suite by default for VBench score evaluation. Full results of short-prompt VBench scores refer to Tab. 19. Sample visualization refers to Sec. 11.5.

#### 9.4.2. Homogeneous vs. Heterogeneous Parameterization

For the given teacher model $v_{\theta'}$ with $v$-prediction, we compare the student models with heterogeneous $x_\theta$ and homogeneous $v_\theta$ parameterization from the teacher, under the VSD+CD loss. The student model weights are initialized from the teacher model for both configurations. The evaluated VBench results are shown in Tab. 14. The homogeneous parameterization leads to slightly better performance over the heterogeneous parameterization and even the teacher model. Full results for VBench scores refer to Table 13.

Table 12. Comparison of VBench scores across models with different inference steps (values in percentage).

| Model | Teacher | Student (VSD) | | |
|---|---|---|---|---|
| Inference Steps | 50 | 1 | 2 | 4 |
| Subject Consistency | 83.99 | 90.09 | 92.27 | **93.26** |
| Background Consistency | 93.78 | 94.39 | 95.17 | **95.82** |
| Temporal Flickering | 96.42 | **96.79** | 95.73 | 95.79 |
| Motion Smoothness | **98.09** | 97.72 | 96.96 | 97.48 |
| Dynamic Degree | **99.44** | 86.39 | 93.33 | 58.61 |
| Aesthetic Quality | 61.21 | 60.26 | **61.55** | 61.34 |
| Imaging Quality | 63.87 | 61.82 | 66.09 | **68.21** |
| Object Class | 85.79 | 90.03 | 87.86 | **94.72** |
| Multiple Objects | 52.59 | 67.71 | 58.06 | **69.24** |
| Human Action | 99.60 | 98.40 | 99.60 | **99.80** |
| Color | **77.00** | 74.43 | 65.44 | 71.81 |
| Spatial Relationship | 51.40 | **69.17** | 63.96 | 64.80 |
| Scene | 49.99 | 49.74 | **52.21** | 51.89 |
| Temporal Style | **26.45** | 26.03 | 25.19 | 24.93 |
| Appearance Style | **24.83** | 23.90 | 23.77 | 24.31 |
| Overall Consistency | **27.89** | 27.14 | 26.91 | 26.38 |
| Quality Score | 81.89 | 81.61 | **82.71** | 80.95 |
| Semantic Score | 73.71 | **76.66** | 73.86 | 76.61 |
| Total Score | 80.25 | 80.62 | **80.94** | 80.08 |

### 9.5. Complete VBench Scores

The full version of main paper Tab. 2 is Tab. 16. The breakdown VBench scores and reward scores for main paper Tab. 5 are shown in Tab. 18 and Tab. 17. The breakdown VBench scores for main paper Tab. 6 are shown in Tab. 12. The breakdown VBench scores for main paper Tab. 7 and Tab. 14 are shown in Tab. 13. VSD4 indicates the VSD loss for 4-step inference of the student, as our default setting. CD1 and CD5 indicate the CD loss with denoising steps $m=1$ and $m=5$, respectively.

Table 13. Comparison of VBench scores for VSD+CD methods (values in percentage).

| Metric | Teacher | VSD4+CD1 | VSD4+CD5 | VSD4+CD5 |
|---|---|---|---|---|
| Parameterization | $v_\theta$ | $v_\theta$ | $v_\theta$ | $x_\theta$ |
| Subject Consistency | 83.99 | 86.36 | **86.37** | 85.47 |
| Background Consistency | 93.78 | **94.84** | 94.70 | 93.37 |
| Temporal Flickering | 96.42 | 95.60 | 96.48 | **96.51** |
| Motion Smoothness | **98.09** | 97.70 | 98.04 | 98.05 |
| Dynamic Degree | **99.44** | 87.50 | 90.28 | 95.28 |
| Aesthetic Quality | 61.21 | 60.16 | **62.16** | 60.95 |
| Imaging Quality | 63.87 | 62.85 | **65.24** | 63.39 |
| Object Class | 85.79 | 85.84 | **89.79** | 87.07 |
| Multiple Objects | 52.59 | 52.53 | **63.86** | 54.51 |
| Human Action | **99.60** | 99.40 | 99.20 | **99.60** |
| Color | **77.00** | 64.02 | 71.38 | 69.35 |
| Spatial Relationship | 51.40 | 55.34 | **59.50** | 54.89 |
| Scene | 49.99 | 49.29 | 49.49 | **53.85** |
| Temporal Style | **26.45** | 25.82 | 25.30 | 26.04 |
| Appearance Style | **24.83** | 23.22 | 23.81 | 24.09 |
| Overall Consistency | **27.89** | 27.24 | 27.08 | 27.73 |
| Quality Score | **81.89** | 80.75 | 82.16 | 81.65 |
| Semantic Score | 73.71 | 71.57 | **74.58** | 73.66 |
| Total Score | 80.25 | 78.92 | **80.65** | 80.05 |

Table 14. Comparison of different student-teacher parameterization for distillation with VSD+CD using VBench (long prompt).

| Model | Teacher | Student | |
|---|---|---|---|
| Parameterization | $v_\theta$ | Heterogeneous ($x_\theta$) | Homogeneous ($v_\theta$) |
| Quality Score | 81.89 | 81.65 | 82.16 |
| Semantic Score | 73.71 | 73.66 | 74.58 |
| Total Score | 80.25 | 80.05 | 80.65 |

Table 15. Effects on VBench scores with different prompt lengths: "S" for short prompts and "L" for long prompts.

| Model | Teacher | | VSD1 | | VSD4 | | VSD4+CD1 | | VSD4+LRM | |
|---|---|---|---|---|---|---|---|---|---|---|
| Prompt | S | L | S | L | S | L | S | L | S | L |
| Quality | 81.50 | 81.89 | 81.60 | 81.61 | 80.75 | 80.95 | 79.27 | 80.75 | 82.64 | 84.01 |
| Semantic | 74.64 | 73.71 | 77.10 | 76.66 | 76.67 | 76.61 | 67.52 | 71.57 | 60.04 | 72.51 |
| Total | 80.13 | 80.25↑ | 80.70 | 80.62↓ | 79.94 | 80.08↑ | 76.92 | 78.92↑ | 78.12 | 81.71↑ |

Table 16. Comparison of VBench scores for different models. Our `DOLLAR` method is VSD+CD+LRM.

| Model | Pika | Gen-2 | Gen-3 | Kling | T2V-Turbo (VC2) | T2V-Turbo (Data Juicer) | Teacher | Our DOLLAR (PickScore) | Our DOLLAR (HPSv2) |
|---|---|---|---|---|---|---|---|---|---|
| Subject Consistency | 96.76 | 97.61 | 97.10 | **98.33** | 96.28 | 97.92 | 83.99 | 93.77 | 92.57 |
| Background Consistency | 98.95 | 97.61 | 96.62 | 97.60 | 97.02 | **99.27** | 93.78 | 96.80 | 96.14 |
| Temporal Flickering | **99.77** | 99.56 | 98.61 | 99.30 | 97.48 | 98.14 | 96.42 | 96.30 | 97.48 |
| Motion Smoothness | 99.51 | **99.58** | 99.23 | 99.40 | 97.34 | 97.77 | 98.09 | 97.76 | 98.59 |
| Dynamic Degree | 37.22 | 18.89 | 60.14 | 61.21 | 49.17 | 38.89 | **99.44** | 75.83 | 81.67 |
| Aesthetic Quality | 63.15 | 66.96 | 63.34 | 46.94 | 63.04 | **67.39** | 61.21 | 63.80 | 63.14 |
| Imaging Quality | 62.33 | 67.42 | 66.82 | 65.62 | **72.49** | 70.41 | 63.87 | 69.40 | 65.61 |
| Object Class | 87.45 | 90.92 | 87.81 | 87.24 | 93.96 | **96.44** | 85.79 | 91.63 | 93.84 |
| Multiple Objects | 46.69 | 55.47 | 53.64 | 68.05 | 54.65 | 64.51 | 52.59 | 69.71 | **72.21** |
| Human Action | 88.00 | 89.20 | 96.40 | 93.40 | 95.20 | 95.40 | **99.60** | 99.00 | 99.00 |
| Color | 85.31 | 89.49 | 80.90 | 89.90 | 89.90 | **95.51** | 77.00 | 77.95 | 74.78 |
| Spatial Relationship | 65.65 | 66.91 | 65.09 | **73.03** | 38.67 | 47.17 | 51.40 | 68.56 | 68.35 |
| Scene | 44.80 | 48.91 | 54.57 | 50.86 | 55.58 | **57.30** | 49.99 | 55.06 | 52.72 |
| Temporal Style | 24.44 | 24.12 | 24.71 | 24.17 | 25.51 | 25.55 | **26.45** | 24.64 | 25.23 |
| Appearance Style | 21.89 | 24.31 | 24.86 | 19.62 | 24.42 | **26.82** | 24.83 | 24.45 | 23.50 |
| Overall Consistency | 25.47 | 26.17 | 26.69 | 26.42 | 28.16 | **29.25** | 27.89 | 26.93 | 26.85 |
| Quality Score | 82.68 | 82.47 | **84.11** | 83.39 | 82.57 | 83.38 | 81.89 | 83.49 | 83.83 |
| Semantic Score | 71.26 | 73.03 | 75.17 | 75.68 | 72.57 | **79.13** | 73.71 | 77.90 | 77.51 |
| Total Score | 80.40 | 80.58 | 82.32 | 81.85 | 81.01 | 82.53 | 80.25 | 82.37 | **82.57** |

Table 17. Comparison of LRM with DDPO using VBench (long prompt) and fine-tuning reward metrics HPSv2 and PickScore.

| Reward Model | PickScore | | | HPSv2 | | |
|---|---|---|---|---|---|---|
| Method | VSD+DDPO | VSD+LRM | VSD+CD+LRM | VSD+DDPO | VSD+LRM | VSD+CD+LRM |
| Quality Score | 82.99 | **84.01** | 83.49 | 82.97 | 83.53 | **83.83** |
| Semantic Score | 77.26 | 72.51 | **77.90** | 74.56 | 75.67 | **77.51** |
| Total Score | 81.84 | 81.71 | **82.37** | 81.29 | 81.96 | **82.57** |
| Reward | $0.207 \pm 0.011$ | $0.207 \pm 0.011$ | $\mathbf{0.210 \pm 0.011}$ | $0.271 \pm 0.027$ | $0.276 \pm 0.028$ | $\mathbf{0.277 \pm 0.029}$ |

Table 18. Comparison of VBench scores for DDPO and LRM methods (values in percentage).

| Model | Teacher | VSD4 | VSD4+DDPO (PickScore) | VSD4+LRM (PickScore) | VSD4+DDPO (HPSv2) | VSD4+LRM (HPSv2) |
|---|---|---|---|---|---|---|
| Subject Consistency | 83.99 | 93.26 | **95.26** | 94.34 | 93.27 | 91.99 |
| Background Consistency | 93.78 | 95.82 | 96.21 | 96.08 | 96.22 | **96.93** |
| Temporal Flickering | 96.42 | 95.79 | 96.56 | 95.85 | 96.64 | **96.80** |
| Motion Smoothness | **98.09** | 97.48 | 96.45 | 97.30 | 97.56 | 97.39 |
| Dynamic Degree | **99.44** | 58.61 | 85.83 | 94.44 | 81.67 | 85.56 |
| Aesthetic Quality | 61.21 | 61.34 | 61.85 | 61.84 | 61.66 | **63.14** |
| Imaging Quality | 63.87 | 68.21 | 65.98 | **68.49** | 66.39 | 67.35 |
| Object Class | 85.79 | **94.72** | 94.29 | 87.28 | 91.91 | 90.65 |
| Multiple Objects | 52.59 | 69.24 | **72.33** | 55.11 | 65.32 | 60.34 |
| Human Action | 99.60 | **99.80** | 98.00 | 98.80 | 98.20 | 99.40 |
| Color | **77.00** | 71.81 | 76.28 | 76.12 | 70.00 | 73.69 |
| Spatial Relationship | 51.40 | 64.80 | **65.19** | 54.25 | 61.75 | 63.81 |
| Scene | 49.99 | 51.89 | 52.60 | 49.17 | 49.65 | **53.43** |
| Temporal Style | **26.45** | 24.93 | 25.00 | 24.39 | 24.53 | 25.02 |
| Appearance Style | **24.83** | 24.31 | 23.99 | 23.68 | 23.66 | 24.53 |
| Overall Consistency | **27.89** | 26.38 | 26.43 | 25.97 | 26.67 | 26.76 |
| Quality Score | 81.89 | 80.95 | 82.99 | 84.01 | 82.97 | **83.53** |
| Semantic Score | 73.71 | 76.61 | **77.26** | 72.51 | 74.56 | 75.67 |
| Total Score | 80.25 | 80.08 | 81.84 | 81.71 | 81.29 | **81.96** |

Table 19. VBench scores with short prompts (values in percentage) for some models.

| Model | Teacher | VSD1 | VSD4 | VSD4+LRM (PickScore) | VSD4+LRM (HPSv2) | VSD4+CD1 |
|---|---|---|---|---|---|---|
| Subject Consistency | 84.80 | 89.39 | 92.98 | 94.13 | 91.72 | 84.83 |
| Background Consistency | 94.10 | 94.91 | 96.12 | 95.14 | 96.34 | 93.87 |
| Temporal Flickering | 96.12 | 96.96 | 96.55 | 95.29 | 96.50 | 94.84 |
| Motion Smoothness | 97.99 | 97.57 | 97.12 | 96.77 | 96.65 | 97.08 |
| Dynamic Degree | 97.78 | 91.94 | 61.39 | 93.06 | 94.17 | 93.61 |
| Aesthetic Quality | 57.74 | 57.33 | 58.24 | 58.08 | 60.20 | 55.32 |
| Imaging Quality | 65.41 | 62.10 | 67.79 | 68.97 | 67.12 | 62.28 |
| Object Class | 88.45 | 89.89 | 93.12 | 57.34 | 92.67 | 80.54 |
| Multiple Objects | 56.54 | 73.86 | 72.29 | 38.43 | 66.45 | 47.90 |
| Human Action | 99.60 | 98.00 | 98.20 | 92.00 | 96.60 | 96.60 |
| Color | 77.75 | 86.19 | 79.55 | 78.36 | 82.90 | 67.55 |
| Spatial Relationship | 51.21 | 70.13 | 70.09 | 44.49 | 63.32 | 55.34 |
| Scene | 50.89 | 35.32 | 42.95 | 13.31 | 36.90 | 29.53 |
| Temporal Style | 26.52 | 26.21 | 24.91 | 23.03 | 25.11 | 25.02 |
| Appearance Style | 24.76 | 23.93 | 23.87 | 23.47 | 24.45 | 23.03 |
| Overall Consistency | 27.96 | 28.06 | 26.42 | 24.81 | 27.05 | 27.13 |
| Quality Score | 81.50 | 81.60 | 80.75 | 82.64 | 83.03 | 79.27 |
| Semantic Score | 74.64 | 77.10 | 76.67 | 60.04 | 75.08 | 67.52 |
| Total Score | 80.13 | 80.70 | 79.94 | 78.12 | 81.44 | 76.92 |

## 10. Challenges and Discussions

**Long Prompt Bias.** The experiments in Sec. 9.4.1 show that, current models perform better for long and more descriptive prompt, which is inherited from the teacher model. The reason is hypothesized to be the well-captioned text-to-video training dataset, which emphasize detailed descriptions. With longer prompts, the text-video alignment, understanding of object relationships, and depiction of motion are generally more robust and accurate. To address this issue, the performance gap with short prompts could be reduced by incorporating more short-prompt datasets during training or fine-tuning. Additional results illustrating this phenomenon are provided in Fig. 30, with corresponding video samples available on the website.

**Reward Overoptimization.** As demontrated by experiments in Sec. 9.3, the reward overoptimization issue sometimes happens for some certain reward metrics for both LRM and DDPO methods. To address this issue, early stopping or checkpoint selection can be one rescue. Another approach involves incorporating additional explicit regularization to constrain the student model with the teacher model during the reward fine-tuning process, and implicit regularization like diffusion loss or VSD loss may not be sufficient for this purpose. Beyond early stopping and careful tuning of the loss coefficients between data modeling and reward tuning, adopting the memoryless noise schedule [9] shows promise in steering the model toward correctly converging to tilted distributions. Further investigation into these strategies and their effectiveness in resolving overoptimization remains an important direction for future work.

**Diversity.** From the Vendi score diversity measure of generated samples in main paper, we verifies the effectiveness of incorporating additional CD loss for improving the sample diversity. However, both qualitative comparisons and visual inspections reveal that a diversity gap remains between the distilled student models and the teacher model.

While prior research predominantly emphasizes sample quality, the diversity of T2V models is crucial for practical applications, where diverse outputs are often necessary. This aspect of diversity remains underrepresented even in the comprehensive VBench evaluation, highlighting an area that warrants further attention and improvement.

**Misalignment in Evaluation.** As discussed in Sec. 9.2, our experiments reveal a misalignment between VBench scores and human evaluations for videos generated using the same set of prompts. Humans may be more sensitive to unnatural flaws in videos, which can influence their preferences differently from the automatic evaluation metrics used in VBench. This discrepancy highlights the difficulty of aligning weighted score metrics with human preferences. As a result, models that achieve higher VBench scores may not necessarily be preferred by humans, and vice versa. Given the inherent complexity of video content, relying on a single or limited set of metrics may fail to fully capture video quality. This presents a challenge for the research community to develop more comprehensive evaluation protocols that are better aligned with human preferences.

## 11. Visualization

### 11.1. More Qualitative Results

More qualitative results of our methods (VSD+CD+LRM) are displayed in Fig. 17, 18 and 19.

Visual comparison of our methods with baselines in Tab. 2 for generated samples with the same prompt is shown in Fig. 20 and 21. For fair of comparison, we visualize all sampled frames with resolution $192 \times 320$ as the typical sample size of our models.

### 11.2. Comparison of Reward Model Fine-tuning

As additional results for Sec. 5.4, we provide visualization of samples with different reward model fine-tuning methods in Fig. 22 and 23. It compares:

- VSD;
- VSD with DDPO fine-tuning, using reward PickScore;
- VSD with DDPO fine-tuning, using reward HPSv2;
- VSD with LRM fine-tuning, using reward PickScore;
- VSD with LRM fine-tuning, using reward HPSv2.

All results are for 4-steps sampling after the distillation process.

### 11.3. Inference Steps

We provide visualization of samples with different sampling steps for the VSD method, as shown in Fig. 24 and 25. During the distillation process, the sampling steps is set to be 1, 2, 4, and at inference time it follows the same step number as in distillation. From visual inspection, it is clear to show that a larger number of sampling steps usually leads to better performances, which may not be well captured by the slight difference of VBench scores. As an example, in Fig. 25, the "astronaut" video with 1-step inference looks blurry but the 4-step sample has more sharper details and realistic surroundings. It should be clear to see that increasing the inference steps indeed improves the sample performances.

Fig. 26 visualizes the samples with 4-step teacher DDIM sampling, and with only CD loss for student distillation, as typically used in previous work like LCM and VideoLCM. Few-step teacher sampling without any distillation cannot generate high-quality samples. CD loss only tends to generate overly smoothed samples.

### 11.4. Diversity

For visualizing the difference of sample diversity across different methods, we provide sample visualization, including videos in Fig. 27 for comparing our DOLLAR method with DMD, Fig. 28 and Fig. 29 with extracted video frames for several models after training:

- VSD for 4-step sampling;
- VSD with CD for 4-step sampling and $m=5$ for CD;
- Teacher model with 50 steps DDIM sampling.

The CD improves the sample diversity from both visual inspection and the quantitative measurement with Vendi score as in Tab. 3 of the main paper.

### 11.5. Prompt Length

As additional results to Sec. 9.4.1, we visualize samples with long descriptive prompts and corresponding short prompts in Fig. 30. It further verifies the hypothesis that the trained models tend to align the videos better with longer and more descriptive prompts. According to this results, the VBench evaluation in our experiments takes the long prompts for video generation by default.

### 11.6. Sampling with Various Styles and Motions

The distilled student models with the proposed methods demonstrate great performances over various styles in prompts, including different artistic styles like *Ukiyo style, cuberpunk, surrealism, pixel art, oil painting, watercolor painting, black and white*, etc. It also supports different camera motions in the video, like *pan left, pan right, tilt down, tilt up, zoom in, racking focus*, etc. The visualization for generated samples with various styles and camera motions is shown in Fig. 31 and Fig. 32.


Figure 17. More qualitative results of our method (VSD+CD+LRM). Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).


Figure 18. More qualitative results of our method (VSD+CD+LRM). Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).

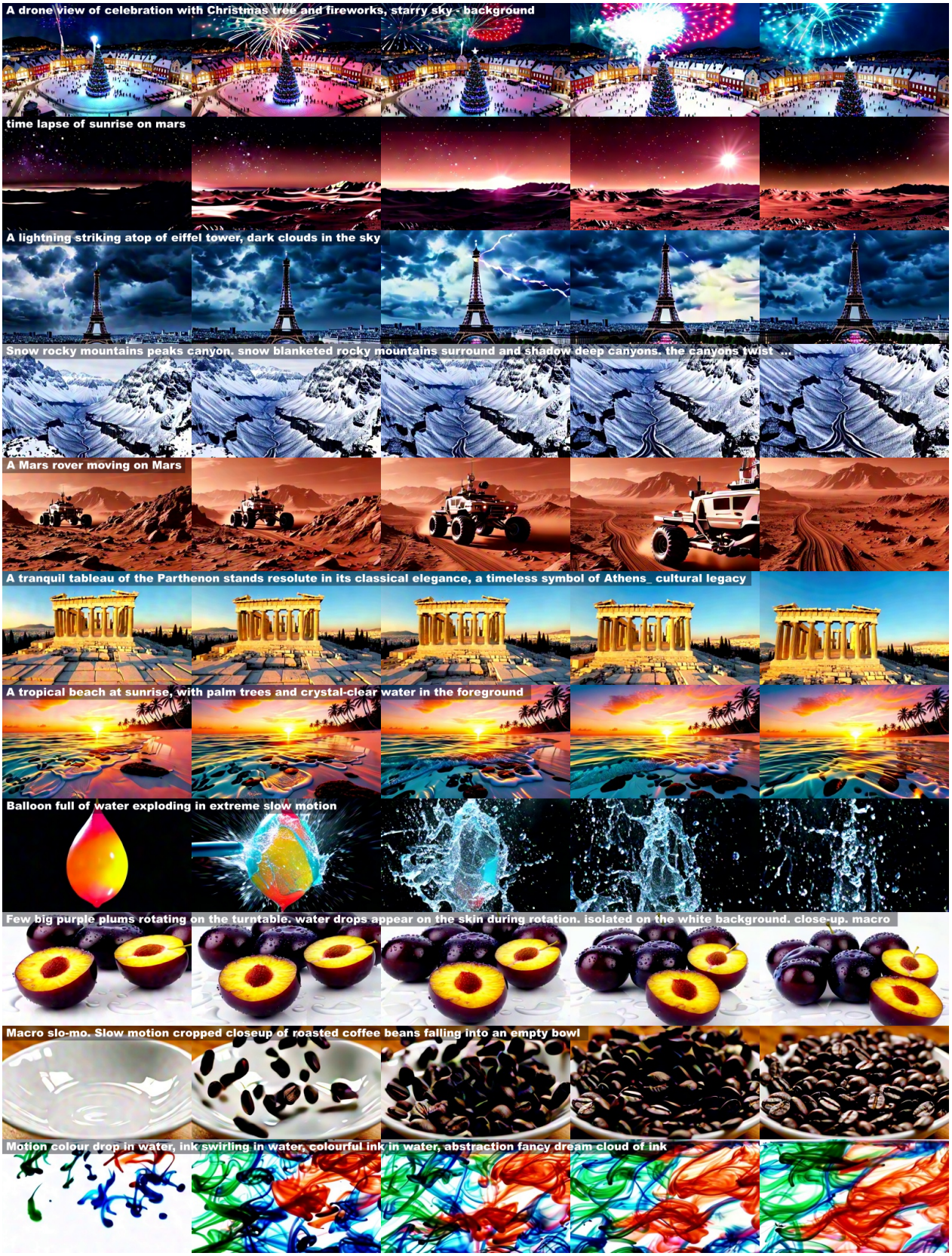



Figure 19. More qualitative results of our method (VSD+CD+LRM). Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).


Figure 20. Comparison of our method (VSD+CD+LRM) against several baselines. Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120). Videos from all baseline methods are transformed into $192 \times 320$ resolution for fair comparison, including Gen-2, Gen3, Kling, Pika. Our model shows superior performances in text-video alignment, motions, visual quality and fidelity.

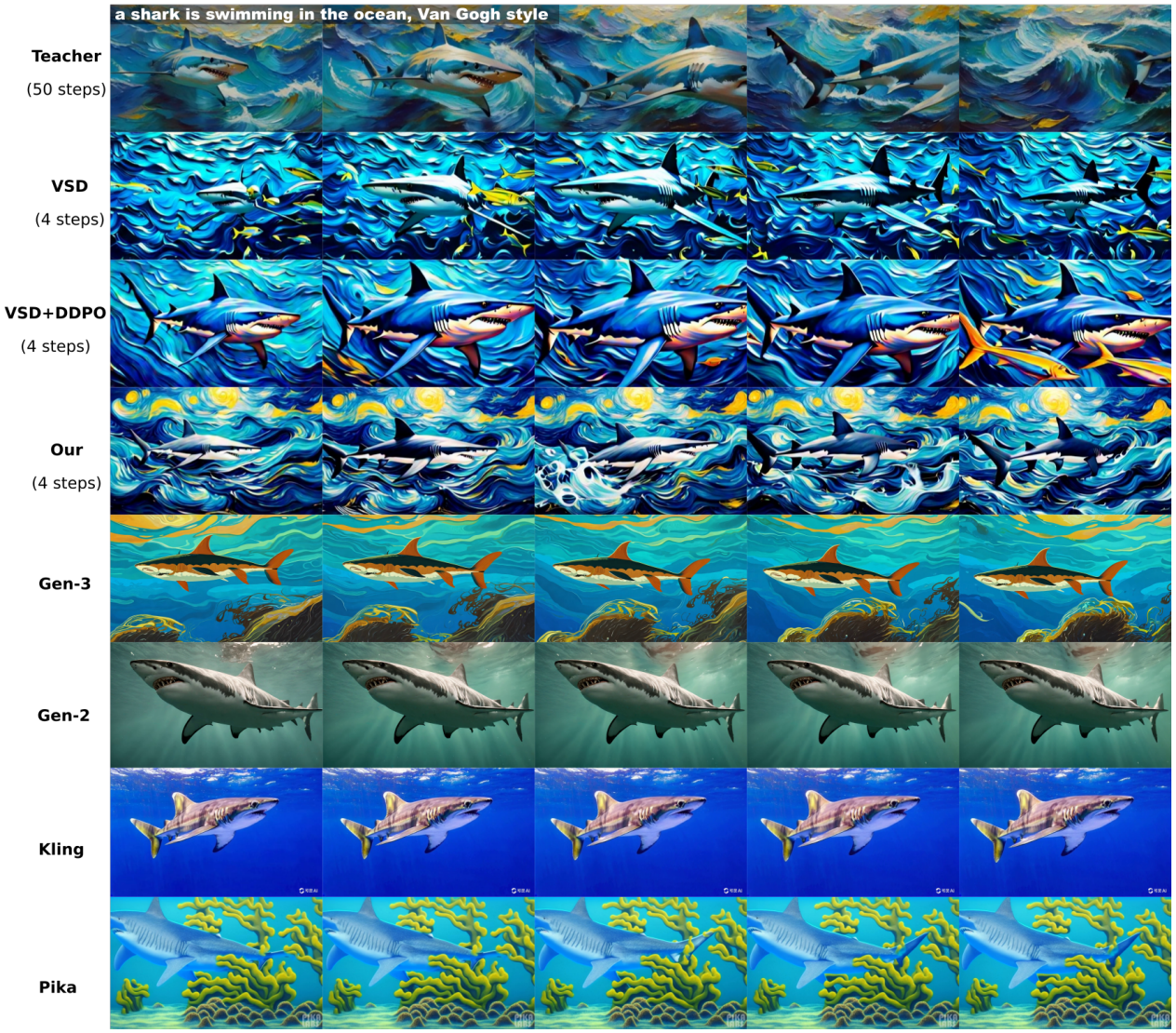
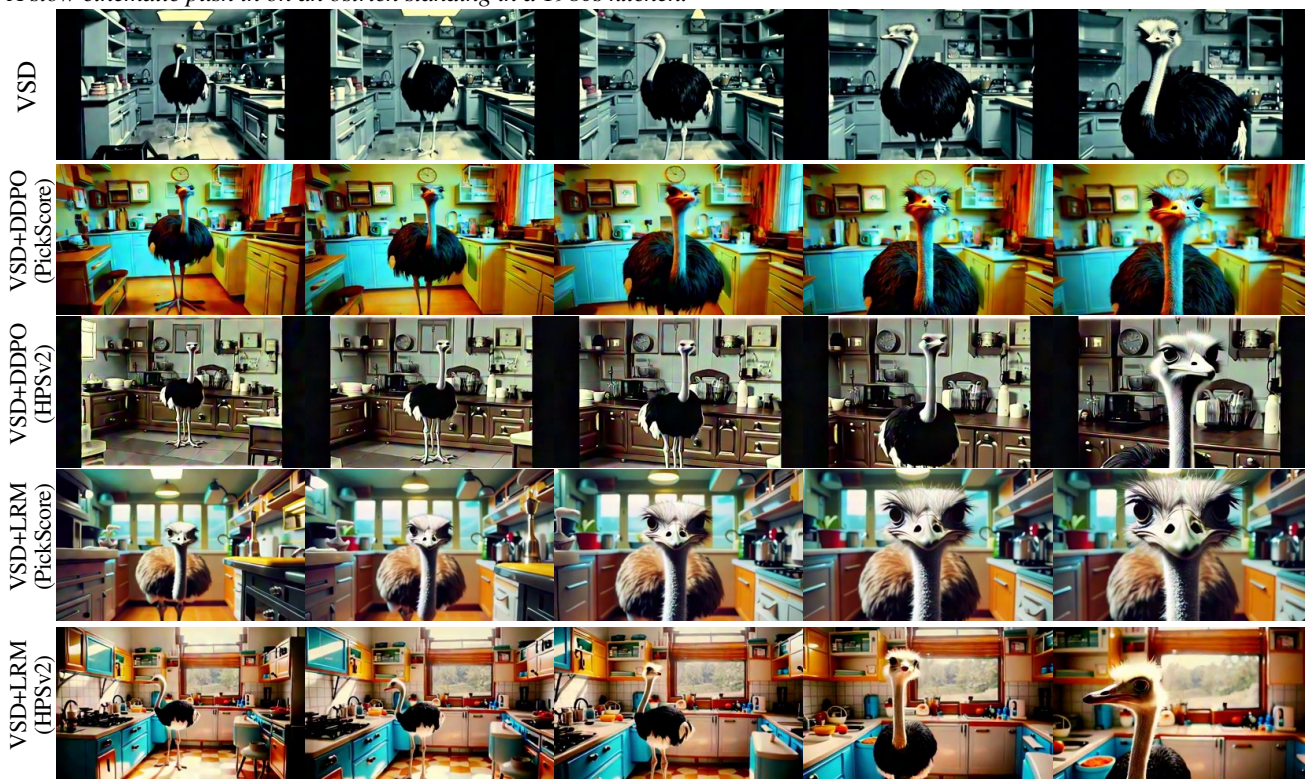



Figure 21. Comparison of our method (VSD+CD+LRM) against several baselines. Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120). Videos from all baseline methods are transformed into $192 \times 320$ resolution for fair comparison, including Gen-2, Gen3, Kling, Pika. Our model shows superior performances in text-video alignment, motions, visual quality and fidelity.

*A slow cinematic push in on an ostrich standing in a 1980s kitchen.*




*A breathtaking aurora dances across the night sky, vibrant green and purple hues illuminating snow-covered mountains and a serene lake.*

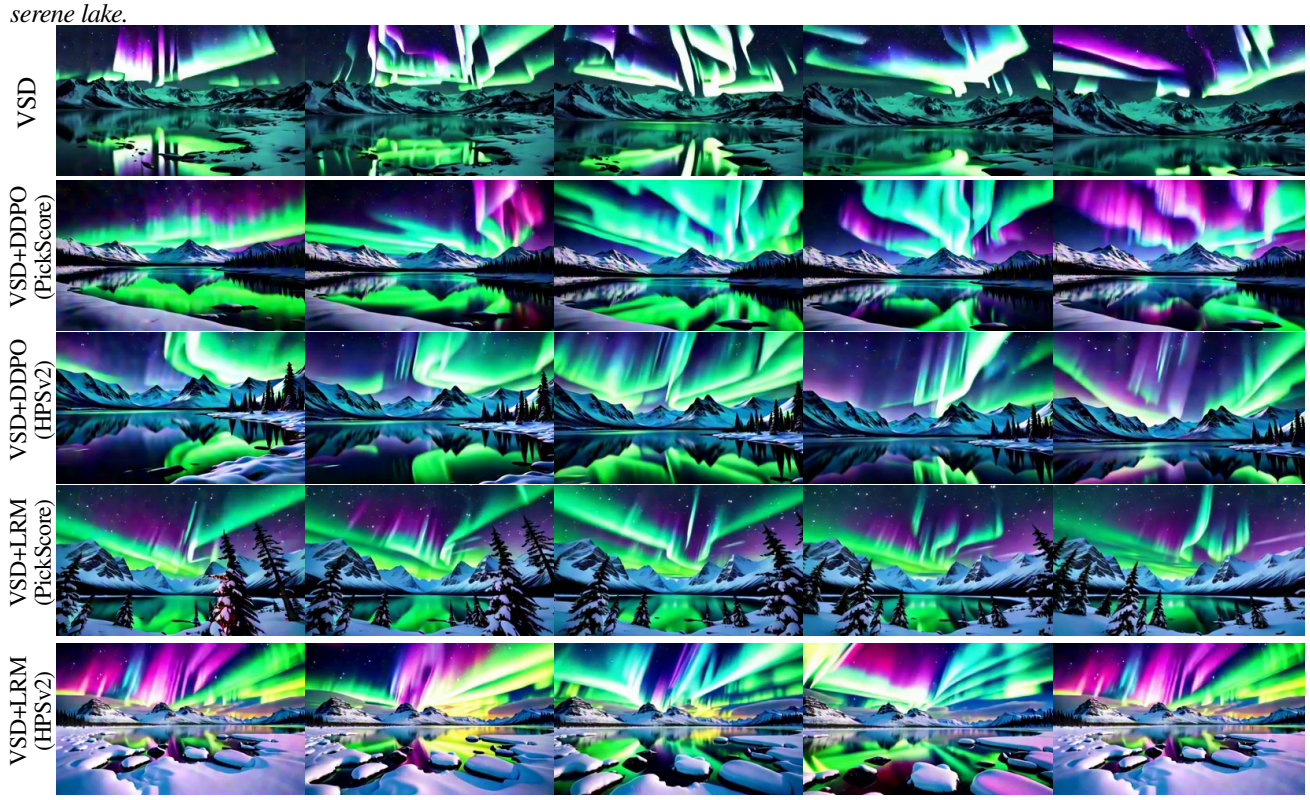



Figure 22. Visualization of video samples using different methods: VSD, VSD+DDPO(PickScore), VSD+DDPO(HPSv2), VSD+LRM(PickScore), VSD+LRM(HPSv2). Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).


Figure 23. Visualization of video samples using different methods: VSD, VSD+DDPO(PickScore), VSD+DDPO(HPSv2), VSD+LRM(PickScore), VSD+LRM(HPSv2). Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).

*FPV moving through a forest to an abandoned house to ocean waves.*


*A middle-aged sad bald man becomes happy as a wig of curly hair and sunglasses fall suddenly on his head.*


Figure 24. VSD 1, 2, 4 steps, teacher with 50 steps. Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).

*An astronaut running through an alley in Rio de Janeiro.*


*An older man playing piano, lit from the side.*


Figure 25. VSD 1, 2, 4 steps, teacher with 50 steps. Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).


Figure 26. Sample results of 4 steps DDIM by the teacher model and 4 steps student model with CD loss. One frame for each video. Same five prompts as previous results: (from left to right) (1). *A slow cinematic push in on an ostrich standing in a 1980s kitchen.* (2). *A breathtaking aurora dances across the night sky, vibrant green and purple hues illuminating snow-covered mountains and a serene lake.* (3). *Handheld camera moving fast, flashlight light, in a white old wall in a old alley at night a black graffiti that spells "Cool Baby".* (4). *FPV moving through a forest to an abandoned house to ocean waves.* (5). *A middle-aged sad bald man becomes happy as a wig of curly hair and sunglasses fall suddenly on his head.*

Figure 27. **[Click to play] Mode collapse issue:** By incorporating variational score distillation, consistency distillation, and latent reward fine-tuning, our DOLLAR method (top) with 4 inference steps increases sample diversity and fidelity, while DMD (bottom) [75] suffers from mode collapse issue. Three videos with same prompt and different random seeds are compared. The prompt is "A middle-aged sad bald man becomes happy as a wig of curly hair and sunglasses fall suddenly on his head". You can [*click*] each sample to play the video (20 extracted frames) in Adobe Acrobat.

*Close-up of an Asian man with a hopeful expression. He's wearing a knit navy sweater and leaning forward slightly. His eyes are wide and focused, giving a sense of urgency or excitement. Soft glowing light illuminates his face, highlighting his features and the texture of his skin. The mood is hopeful, as if he's in the middle of an exciting conversation or reacting to something surprising.*


*An older man playing piano, lit from the side.*


*A middle-aged sad bald man becomes happy as a wig of curly hair and sunglasses fall suddenly on his head.*


Figure 28. Diversity: VSD (top line), VSD+CD (middle line), teacher (bottom line), one frame from each video (5 videos).

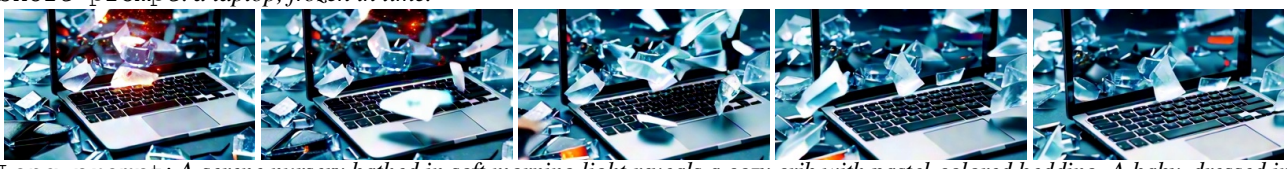
*A slow cinematic push in on an ostrich standing in a 1980s kitchen.*


*Handheld camera moving fast, flashlight light, in a white old wall in a old alley at night a black graffiti that spells "Cool Baby".*


Figure 29. Diversity: VSD (top line), VSD+CD (middle line), teacher (bottom line), one frame from each video (5 videos).

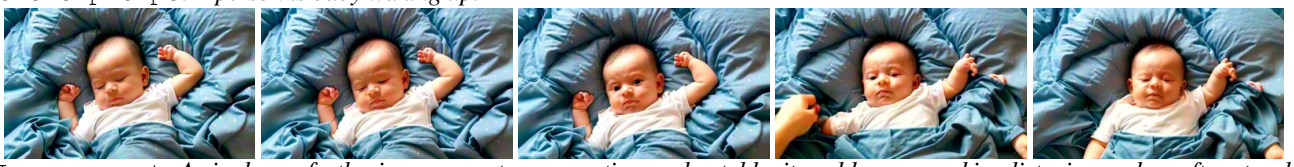
Long prompt: *A sleek, modern laptop, its screen displaying a vibrant, paused scene, sits on a minimalist wooden desk. The room is bathed in soft, natural light filtering through sheer curtains, casting gentle shadows. The laptop's keyboard is mid-illumination, with a faint glow emanating from the keys, suggesting a moment frozen in time. Dust particles are suspended in the air, caught in the light, adding to the stillness. A steaming cup of coffee beside the laptop remains untouched, with wisps of steam frozen in mid-air. The scene captures a serene, almost magical pause in an otherwise bustling workspace.*

Short prompt: *a laptop, frozen in time.*



Long prompt: *A serene nursery bathed in soft morning light reveals a cozy crib with pastel-colored bedding. A baby, dressed in a cute onesie adorned with tiny stars, stirs gently. The camera captures the baby's delicate eyelashes fluttering open, revealing curious, sleepy eyes. The baby stretches tiny arms and legs, yawning adorably. A mobile with soft, plush animals gently spins above, casting playful shadows. The room is filled with the soft hum of a lullaby, creating a peaceful atmosphere as the baby slowly awakens, ready to greet the new day with innocent wonder.*

Short prompt: *A person is baby waking up.*



Long prompt: *A single, perfectly ripe pear rests on a rustic wooden table, its golden-green skin glistening under soft, natural light. The pear's surface is dotted with tiny, delicate freckles, and its curved stem casts a gentle shadow. The background is a blurred, warm-toned kitchen scene, with hints of vintage decor and a window letting in a soft, diffused glow. The stillness of the frame captures the pear's natural beauty and simplicity, evoking a sense of calm and timelessness.*

Short prompt: *In a still frame, a pear.*

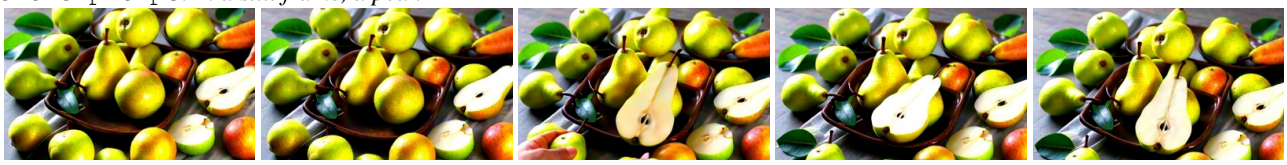
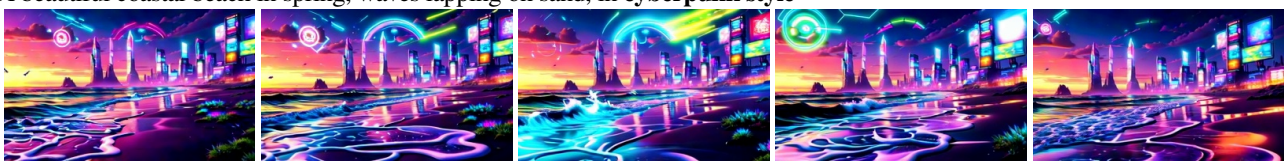


Figure 30. Comparison of sampled videos for VBench long and short prompts. Five frames are displayed for each video (frame index: 0, 30, 60, 90, 120).

A beautiful coastal beach in spring, waves lapping on sand by Hokusai, in the **style of Ukiyo**

A beautiful coastal beach in spring, waves lapping on sand, in **cyberpunk style**



A beautiful coastal beach in spring, waves lapping on sand, **oil painting**

A beautiful coastal beach in spring, waves lapping on sand, **pixel art**

A beautiful coastal beach in spring, waves lapping on sand, **surrealism style**

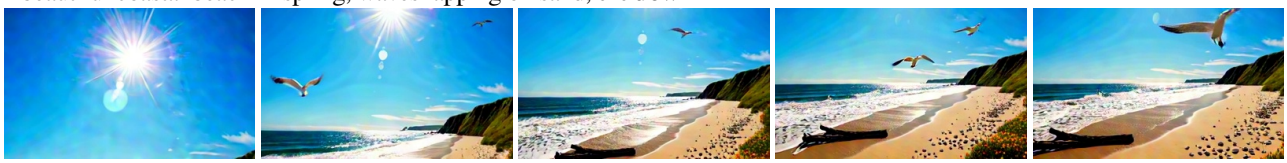
A beautiful coastal beach in spring, waves lapping on sand, **black and white**

A beautiful coastal beach in spring, waves lapping on sand, **watercolor painting**

Figure 31. Video generation with diverse styles, using prompts from VBench. Five frames are extracted uniformly from one video for each prompt (with frame index: 0, 30, 60, 90, 120).

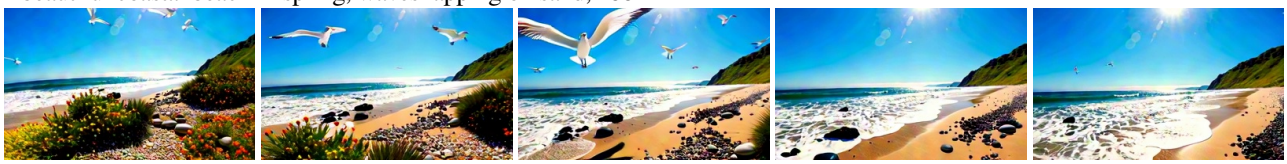
A beautiful coastal beach in spring, waves lapping on sand, **pan left**

A beautiful coastal beach in spring, waves lapping on sand, **tilt down**



A beautiful coastal beach in spring, waves lapping on sand, **tilt up**

A beautiful coastal beach in spring, waves lapping on sand, **zoom in**



A beautiful coastal beach in spring, waves lapping on sand, **racking focus**

A beautiful coastal beach in spring, waves lapping on sand, **in super slow motion**

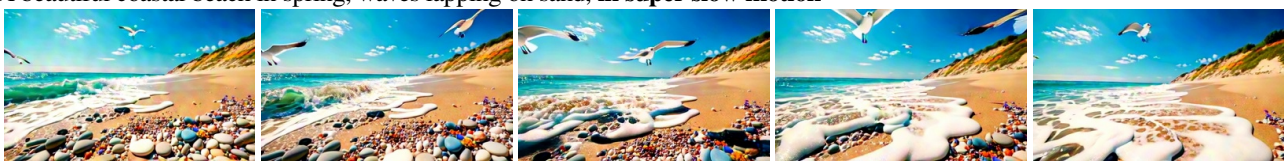


Figure 32. Video generation with diverse camera motions, using prompts from VBench. Five frames are extracted uniformly from one video for each prompt (with frame index: 0, 30, 60, 90, 120).