



V.I.P. : Iterative Online Preference Distillation for Efficient Video Diffusion Models

Supplementary Material

In this supplementary material, we provide,

- **A. Limitations**
- **B. Future Research**
- **B. Experimental Details**
 - B.1. Pruning algorithm details
 - B.2. Data curation details
 - B.3. Evaluation details
 - B.4. Dynamic degree analysis
 - B.5. User study details
- **C. Additional Experiments**
 - C.1. SFT Weight Experiment
 - C.2. Further Experiments for VideoCrafter 2
 - C.3. Experiments for step-distilled model
 - C.4. Experiments for reward model
- **D. Additional Explanations on Motivation**
- **E. Prompt Filtering**
 - E.1 Dynamic Degree
 - E.2 Visual Quality
 - E.3 Text Alignment
- **F. Qualitative Results**

A. Limitations

Since we used VideoScore as the reward model, further advancements in reward modeling could further enhance our performance. As our experiments are conducted only on U-Net models, further research on Transformer-based models could provide additional insights and improvements.

B. Future Research

As pruning methods for DiT-based video diffusion model[51] are beginning to emerge, V.I.P., which is *orthogonal* to any pruning strategy, can be applied to DiT-based models by simply adapting such methodology into the framework. Moreover, while DiT-based models benefit from the scalability of transformer architecture, they are computationally heavy. We believe that V.I.P. could also serve as an effective distillation method by leveraging large, capable DiT models as teachers and smaller models as students (e.g. Wan2.1 13B & 1.3B)[55], contributing to building practical T2V models.

C. Experimental details

In this section, we report the additional details of training. As for the prompt filtering process, we detail it in Section F for better readability.

C.1. Pruning algorithm details

At the pruning stage, we iteratively removed blocks from the model one by one and evaluated each model using VideoScore [18]. We then sorted the models by total VideoScore and selected four motion module blocks for AnimateDiff and four U-Net blocks for VideoCrafter2 from the top-ranked models at each stage.

However, as previously mentioned, we observed cases where a high total score was misleading—some models achieved a high total score due to extremely high dynamic degree, despite having low consistency. This suggests that motion quality was poor, but the overall score was inflated because the drop in consistency was offset by an unusually high dynamic degree.

To address this issue, we sorted models by both total VideoScore and consistency to ensure that motion quality was properly considered. Finally, our pruning stage concluded by selecting blocks based on the intersection of the highest-ranked models in total VideoScore and the highest-ranked models in consistency.

Algorithm 1 Step-by-Step Pruning Algorithm

Input: Model M with N modules, Benchmark V_{bench} , Number of modules to prune per stage k

Output: Pruned and Distilled Model M_{pruned}

while Pruning not completed **do**

for $i \leftarrow 1$ to N **do**

Compute $\Delta_i = V_{\text{metric}}(M) - V_{\text{metric}}(M - \text{module } i)$

end for

Select k modules in ascending order of Δ_i : $\mathcal{S} = \{m_1, \dots, m_k\}$

Prune \mathcal{S} from M : $M_{\text{pruned}} \leftarrow M - \mathcal{S}$, $N \leftarrow N - k$

Perform dataset curation and train M_{pruned}

Update $M \leftarrow M_{\text{pruned}}$

end while

C.2. Data curation details

After pruning four modules, we re-evaluated the model using VideoScore to identify which properties had decreased compared to the full model. Note that, as previously mentioned, removing certain redundant blocks can sometimes improve performance rather than degrade it. If multiple

properties exhibited a drop, we selected the property with the largest gap from the full model as the primary target.

Based on this, we used target-filtered prompts obtained by prompt filtering, and used them to generate preferred-unpreferred datasets from the full model and pruned model each. If multiple target properties were selected, we ensured that all selected properties were considered when forming preferred-unpreferred pairs. Moreover, since unpreferred pairs are also part of the training dataset, their quality is crucial. To maintain meaningful learning, we introduced a lower bound for unpreferred pairs, ensuring that their scores remained above $mean - \alpha * std$, and α is fixed at 0.3 in all stages.

Additionally, to prevent unintended learning where other properties dominate the preference learning, we imposed a threshold condition ensuring that the gap in the targeted property (preferred vs. unpreferred) is greater than the gap in any other property.

C.3. Dynamic Degree analysis

When evaluating dynamic degree on VBench, we divide the prompt test set of dynamic degree into static and dynamic. We use the same instruction of Listing 1 to label dynamic and static prompts. We mark score 3 as dynamic and score 1, 2 as static. Then, we redefine dynamic degree by adding the portion of dynamic videos in filtered dynamic prompts and static videos in filtered static prompts.

C.4. User study details

Assessing the quality of generated content is often complicated by its inherent subjectivity. To support our findings and gain deeper insights into human preferences, we conducted a comprehensive user study involving 30 participants. Participants were given a prompt, and a set of videos, consisting of outputs from V.I.P., SFT-on, and Full. The participants were given 18 sets of the data from VideoCrafter 2 and 18 sets from AnimateDiff, resulting in a total of 1080 responses. They were asked to rank the overall preference of the videos based on three given criteria: 1-Visual Quality, 2-Motion Quality, and 3-Text Alignment. The samples used for the user study were chosen randomly from a large, unbiased pool. An example question of the user study is provided in Figure E.

D. Additional Experiments

In this section, we present the results of additional experiments that could not be included in the main paper due to page constraints.

D.1. SFT Weight Experiment

Since SFT Weight is a new parameter that we introduce, we conduct extensive experiments on VideoCrafter 2 in order to understand the effect of the parameter. We search through

a total of 6 values, ranging from $1e2$ to $1e7$. The setup is identical to the main experiment, the only difference lying in w_{SFT} .

Stage	weight	Visual Quality	Temporal Consist.	Dynamic Degree	Text Align.
Stage 1	$1e2$	2.561	2.494	2.793	2.477
	$1e3$	2.607	2.569	2.751	2.500
	$1e4$	2.613	2.591	2.750	2.505
	$1e5$	2.620	2.597	2.734	2.504
	$1e6$	2.630	2.608	2.731	2.510
	$1e7$	2.622	2.599	2.735	2.499
Stage 2	$1e2$	2.392	2.292	2.775	1.982
	$1e3$	2.424	2.349	2.763	2.016
	$1e4$	2.621	2.601	2.737	2.518
	$1e5$	2.634	2.618	2.720	2.516
	$1e6$	2.629	2.617	2.728	2.518
	$1e7$	2.449	2.403	2.712	1.959

Table A. Ablation on SFT weight for VideoCrafter2. The colored rows are the actual parameters used in the main experiment.

Results in table A demonstrate that SFT weight plays a crucial role in the performance of the models. While a typically low w_{SFT} results in abnormally high dynamics that lead to video quality degradation, over a certain critical point, the model’s overall performance just drops. Results show that such performance drop is more extreme in the second stage, meaning that the performance drop is likely resulted by overly fitting to the teacher’s output as discussed in Section 3.1.

Stage	Method	Visual Quality	Temporal Consist.	Dynamic Degree	Text Align.
1	Pruned	2.609	2.588	2.744	2.487
	V.I.P.	2.630	2.608	2.731	2.510
2	Pruned	2.627	2.595	2.725	2.486
	V.I.P.	2.629	2.617	2.728	2.518
3	Pruned	2.436	2.372	2.749	1.923
	V.I.P.	2.594	2.580	2.718	2.429
	Full	2.627	2.602	2.728	2.491

Table B. Experimental results on VideoCrafter 2. While our method shows consistent performance in recovering the degraded performance due to pruning, when VC2 is pruned for a third stage, the performance drops drastically, making it unreasonable to report the numbers. However, even so, V.I.P. show great recovery of performance.

D.2. Further Experiments for VideoCrafter2

In this section, we present the results of training VC up to Stage 3. As shown in Table B, after pruning two additional U-Net blocks, the performance of the pruned model drops significantly. Despite further training, the model fails to

reach optimal performance. However, as observed in the table, after applying ReDPO, the performance gap dramatically improves, demonstrating that even with severe performance degradation, ReDPO and VIP effectively facilitate learning and recovery.

D.3. Experiments for step-distilled model

In this section, to demonstrate V.I.P.’s effectiveness on a step-distilled model, we experiment on AnimateDiff Lightning[33], a 4-step distilled model. As shown in Table C, our method meets the performance of the full model in both stages, which is remarkable considering that it has already been distilled once. Contrarily, Table D show that SFT struggles significantly, even with the same V.I.P. framework with only a difference in loss. These findings underscore the robustness of V.I.P., especially in heavily pruned, capacity-limited settings like step-distilled models. The clear advantage over SFT in such scenarios emphasizes the effectiveness of our targeted, preference-driven distillation strategy.

Stage	Method	Visual Quality	Temporal Consist.	Dynamic Degree	Text Align.	AVG.
S1	Full	2.644	2.560	2.542	2.411	2.539
	Pruned	2.640	2.566	2.532	2.410	2.537
	ReDPO	2.642	2.562	2.537	2.410	2.538
S2	Pruned	2.640	2.564	2.535	2.393	2.533
	ReDPO	2.641	2.565	2.550	2.397	2.538

Table C. Experiments on AD Lightning with VideoReward.

D.4. Experiments for reward model

In this section, to examine the robustness of our method across different reward models, we replaced VideoScore with a more recent reward model, VideoReward[36], in our two-stage pruning experiment on AnimateDiff Lightning. As shown in Table D, using VideoReward led to improved performance compared to VideoScore. This result highlights that our framework is *reward-model agnostic*—it uses reward models only to generate preference pairs, without any direct propagation of reward values. Consequently, improvements in the reward models translate directly into better distillation outcomes.

Loss	Visual Quality	Temporal Consist.	Dynamic Degree	Text Align.
SFT	2.637	2.572	2.525	2.388
ReDPO(videoscore)	2.641	2.564	<u>2.540</u>	<u>2.396</u>
ReDPO(videoreward)	2.641	<u>2.565</u>	2.550	2.397

Table D. Experiments on AnimateDiff Lightning.

E. Additional Explanations on Motivation

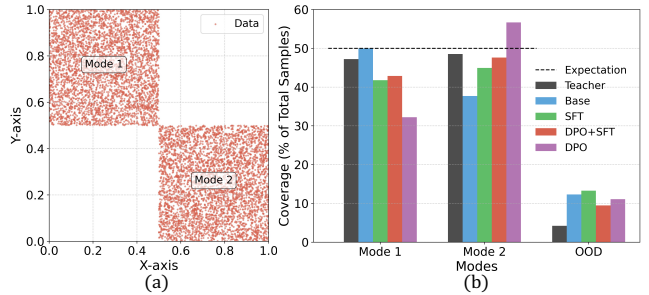


Figure A. Analysis of toy experiment results. (a) Ground-truth distribution used in the toy experiment. (b) Number of samples assigned to each mode and the number of out-of-distribution (OOD) samples.

In this section, we illustrate the limitations of conventional knowledge distillation methods in diffusion models, which rely on SFT loss—particularly when applied to capacity-constrained student models. We then propose an alternative distillation approach and validate its effectiveness through a controlled toy experiment.

Conventional knowledge distillation methods for diffusion models typically transfer knowledge from the teacher to the student by directly minimizing SFT loss. However, in models with limited capacity, this objective forces the student to align with the teacher as closely as possible, often resulting in distributional averaging and overly smoothed outputs. This occurs because minimizing the SFT loss implicitly prioritizes fitting the mean over preserving sharpness [5, 13]. To address this, it is crucial to provide explicit guidance—here, using DPO [45]—that prioritizes important features, ensuring the student allocates its limited capacity effectively rather than blindly mimicking the teacher.

To investigate this, we conducted a toy experiment by training a high-capacity *teacher* and a low-capacity *base student* on a two-dimensional dataset. We implemented the teacher model’s diffusion backbone as a 4-layer MLP with a hidden dimension of 64. Since pruning small MLPs does not behave similarly to pruning large U-Nets—where the goal is typically initializing the student model to closely resemble the teacher—we trained a separate, smaller-scale student model, named the base student, to replicate this phenomenon. Specifically, we trained the base student with a diffusion backbone consisting of a 2-layer MLP with a hidden dimension of 32.

Both models learn to approximate the data distribution shown in Figure A (a). However, the base student exhibits an imbalanced learned distribution—as illustrated in Figure A (b)—due to its limited capacity. We distilled the teacher’s knowledge into the base student using three distinct loss variants: L_{SFT} , L_{DPO} , and $L_{DPO} + L_{SFT}$. The

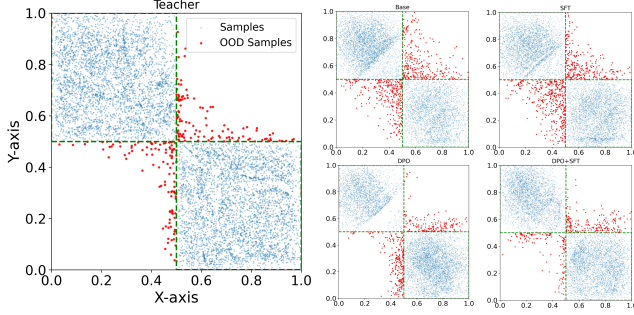


Figure B. **Visualization of learned distributions.** Combining DPO with SFT yields a distribution that most closely aligns with the teacher distribution.

L_{SFT} loss minimizes the L_2 distance between predictions, while L_{DPO} explicitly prioritizes Mode 2 to address the base student’s difficulty in generating samples in this region.

To construct the L_{DPO} -based variants, we first created a DPO dataset by setting the reward function to prioritize samples in Mode 2. A preference dataset was then built, selecting winning samples from the teacher within Mode 2 and losing samples from the student outside Mode 2. Using this dataset, we trained both the L_{DPO} student and the $L_{DPO} + L_{SFT}$ student. The only difference among all trained models was their loss formulation.

Figure A and Figure B presents three key findings: (1) Distillation using L_{SFT} leads to excessive distributional smoothing, resulting in more out-of-distribution (OOD) samples than even the base student. (2) Distillation using L_{DPO} reallocates model capacity toward favoring Mode 2; however, due to inherent over-optimization issues, the model becomes misdirected in uncertain regions of the reward distribution—specifically demonstrated by degradation in Mode 1. (3) Combining L_{DPO} with L_{SFT} balances these effects, effectively prioritizing Mode 2 while mitigating over-optimization. As a result, it reduces the number of OOD samples compared to the other methods and more closely follows the teacher distribution. These observations suggest that L_{DPO} facilitates efficient reallocation of the model’s limited capacity toward critical generative properties, while avoiding over-optimization.

F. Prompt filtering

For prompt filtering, we use Gemini 2.0 Flash to score each prompt from 0 to 3. The score distribution of two properties after LLM filtering is shown in Figure C. The word cloud of prompts before and after filtering is shown in Figure D.

F.1. Dynamic Degree

As shown in Listing 1, we designed an LLM-based filtering process to assign dynamic motion scores to prompts. We

instruct LLM to score 1 if the prompt contains no motion, score 2 if the prompt contains minimal motion, and score 3 if the prompt contains considerable motion. When configuring an example of dynamic degree on an instruction, we use an evaluation set of prompt from VBench. It contains multiple prompts with dynamic motions. We select the score 3 prompt in Dynamic Degree for video curation.

F.2. Visual Quality

As shown in Listing 2, we designed an LLM-based filtering process to assign visual quality scores to the prompts. We instruct LLM to score 1 if the prompt contains simple or generic descriptions, score 2 if the prompt contains moderate visual attributes, and score 3 if the prompt contains highly descriptive, rich in visual attributes. When configuring an example of visual quality, we use LLM to generate appropriate examples. We select the score 3 prompt in Visual Quality for video curation.

F.3. Text Alignment

To filter a prompt set that enhances text alignment, we hypothesize that high-quality text prompts are essential for generating videos that accurately capture semantic meaning. To ensure quality, we establish criteria to exclude prompts that are too short or too long, overly complex, or dominated by location names. Specifically, we retain single-sentence prompts with 5 to 25 words, excluding articles. Additionally, we employ LLM-based filtering during the initial selection stage by assigning a score of 0 to eliminate unusable prompts. By applying these constraints, we ensure that the input prompt set maintains linguistic clarity and relevance, facilitating the construction of a dataset optimized for text alignment. We select non-zero score prompts from Dynamic Degree and Visual Quality.

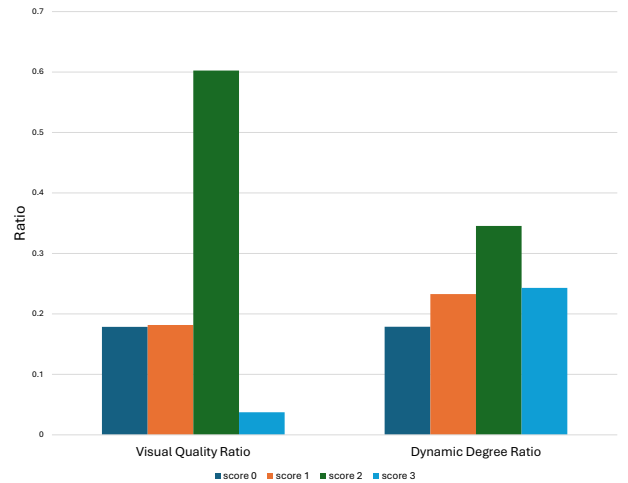


Figure C. The score distribution of prompt after LLM filtering

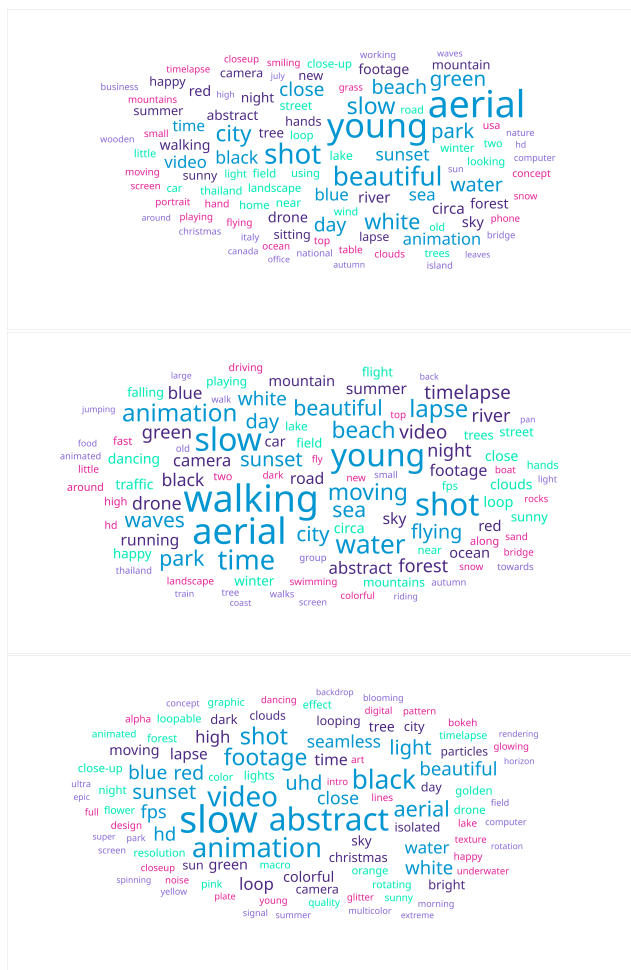


Figure D. **Word clouds of prompt sets.** The word cloud of prompt set before LLM filtering (Top). After filtering, the word cloud of dynamic quality (Middle) and visual quality (Bottom) grounded to its property each.

G. Qualitative results

We additionally report qualitative results of our work from Figure F to Figure O.

Video distillation user study

In this task, you will evaluate videos generated by three different video generation models. Your role is to rank the videos based on their overall quality by considering **three key aspects**:
Visual Quality – How clear, detailed, and realistic each frame appears.
Motion Quality – The smoothness, naturalness, and realism of movement within the video.
Text Alignment – How well the video corresponds to the given text prompt. Each question presents one prompt and one generated video.

The videos are displayed in a fixed order:
The leftmost video is A.
The middle video is B.
The rightmost video is C.
You must rank the videos from **best to worst (1st, 2nd, 3rd place)** based on their overall quality.
Your evaluation should consider **all three factors** (visual quality, motion quality, text alignment).

Text prompt: a panda in a suit speaks to the camera



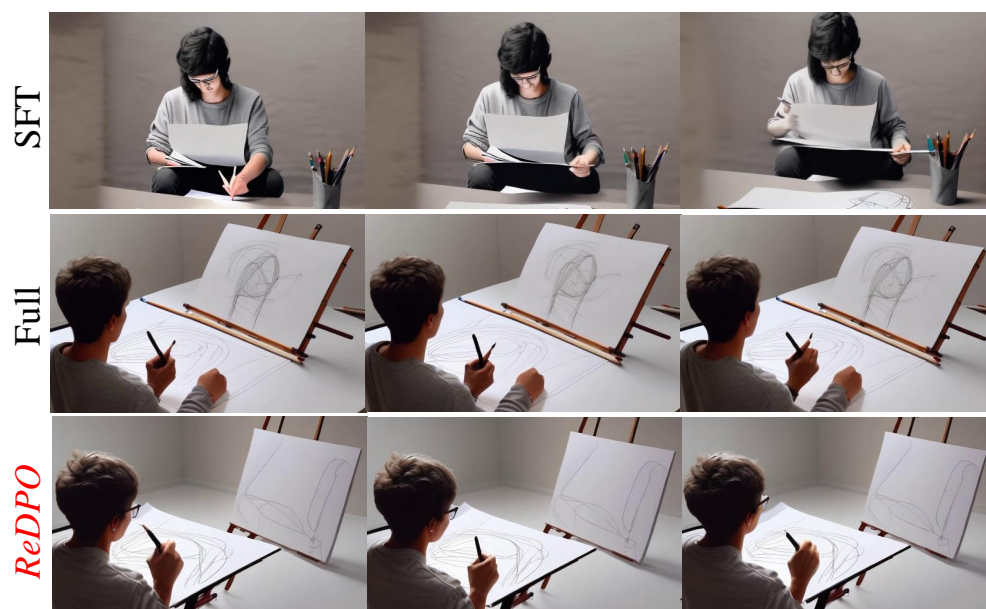
Rank the video from top.

	A	B	C
1st	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
2nd	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
3rd	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure E. Example of the user study instructions that the participants received and a sample of an actual question.



Figure F. Qualitative example of VideoCrafter 2



Prompt: “A person is drawing”

Figure G. Qualitative example of VideoCrafter 2



Prompt: “A bird flying over a snowy forest”

Figure H. Qualitative example of VideoCrafter 2



Prompt: “A person is milking cow”

Figure I. Qualitative example of VideoCrafter 2

Task Overview:

You are a model responsible for scoring prompts for a video diffusion model. Your job is to evaluate and determine the level of dynamic motion present in each prompt. The final output should be a score from 0 to 3.

Task Description:

1. Assign score 0 if the prompt is unusable due to:
 - Fragmented, unclear, or incoherent sentences.
 - Excessive mentions of country names (distracts from motion evaluation).
2. Otherwise, analyze the degree of motion and assign a score from 1 to 3:
 - 1: Static Scene No motion or movement (e.g., a still scene, a stationary object).
 - 2: Minimal Motion Slight transitions or small repetitive actions (e.g., a person blinking, tree leaves rustling, a slow tilt upward).
 - 3: Considerable Motion Significant movement or scene transformation (e.g., running, a car driving, waves crashing, person walking, a smooth tracking shot following person).

Examples:

- 1: A still painting of a landscape with a sunset.
- 2: A person slowly turning the pages of a book.
- 3: A cyclist racing through a city, dodging traffic.

Output format:

Always return your result in this format:

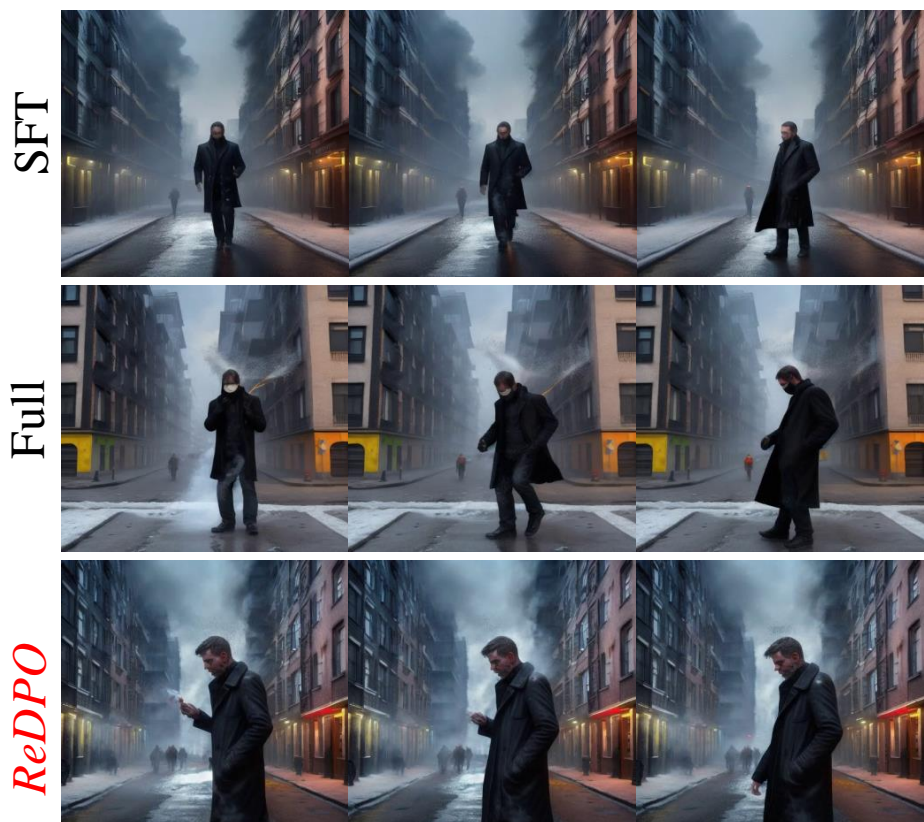
[RESULT] <a score between 0 and 3>

Listing 1. LLM Instruction for Dynamic Motion Scoring



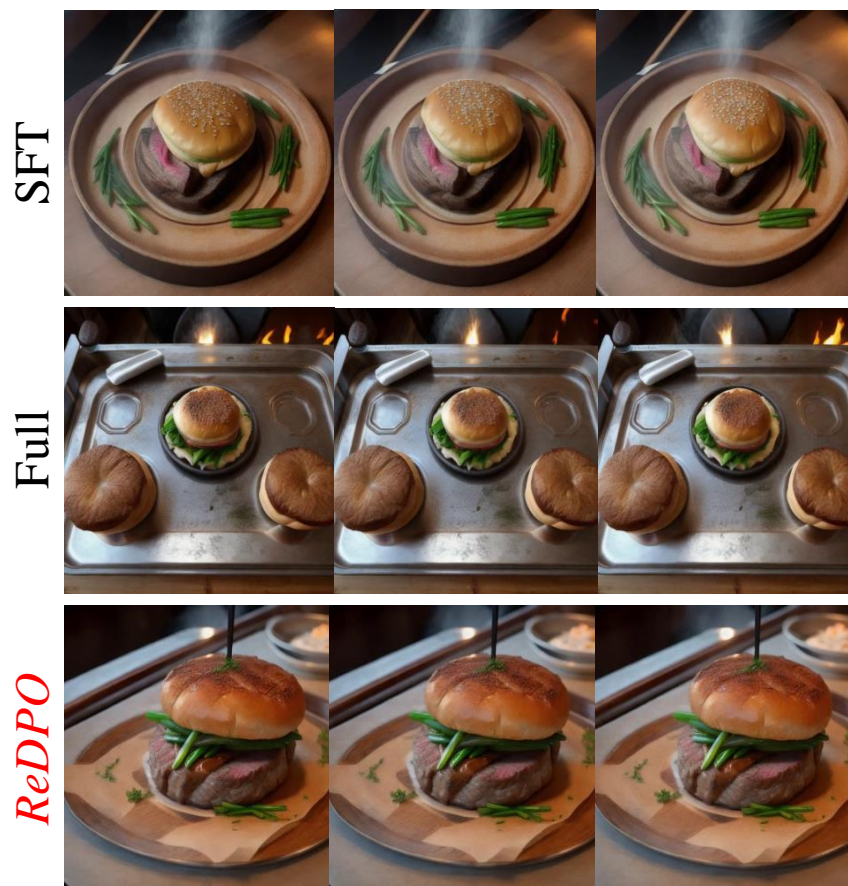
Prompt: “Illustrate a bustling market scene, with fresh produce displayed on stalls, attracting villagers eager to purchase. cartoon style”

Figure J. Qualitative example of AnimateDiff



Prompt: “man in black coat getting covered in explosion and smoke on street with colorful tenement houses around, photorealistic 8k”

Figure K. Qualitative example of AnimateDiff



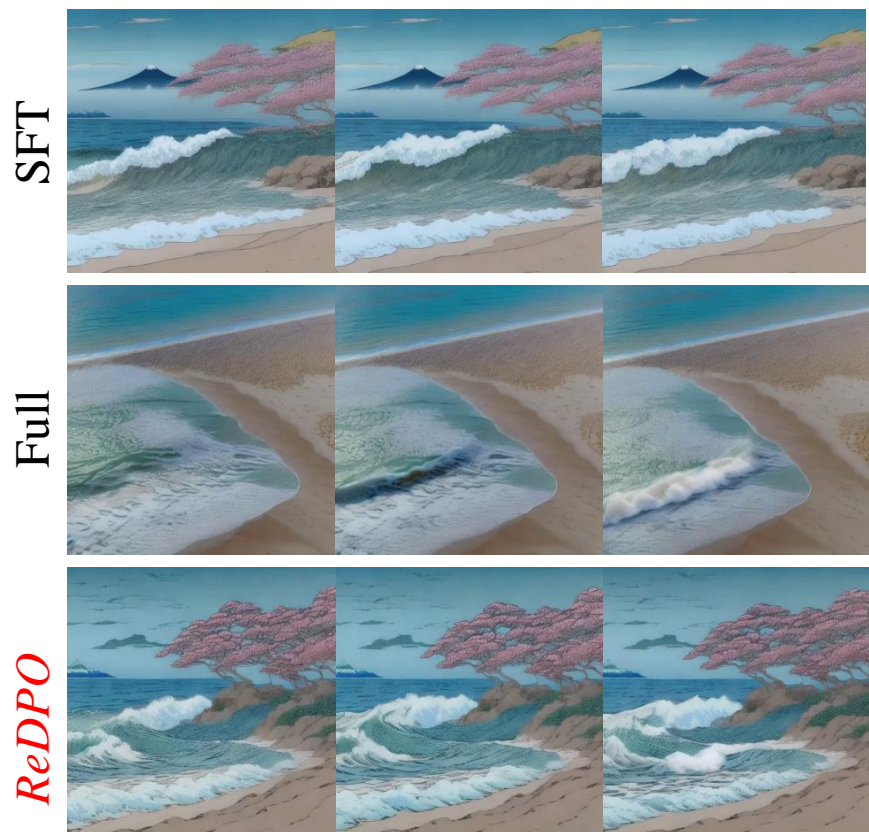
Prompt: “steak bun steamy table shot tasty food”

Figure L. Qualitative example of AnimateDiff



Prompt: "a computer laptop sitting on a beach at sunrise with a volcano erupting from it"

Figure M. Qualitative example of AnimateDiff



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

Figure N. Qualitative example of AnimateDiff



Prompt: “Robot petting a cat on the background of a full moon”

Figure O. Qualitative example of AnimateDiff

```
###Task Overview:
You are a model responsible for scoring prompts for a video diffusion model.
The definition of "Visual Quality" is the quality of the video in terms of clearness,
resolution, brightness, and color. Your job is to evaluate and determine the level of
Visual Quality present in each prompt. The final output should be a score from 0 to 3.

### Task Description:
1. Assign score 0 if the prompt is unusable due to:
  - Fragmented, unclear, or incoherent sentences.
  - Excessive mentions of country names (distracts evaluation).

2. Otherwise, analyze the degree of Visual Quality and assign a score from 1 to 3:
  - Score 1: Low Visual Quality: Vague or generic descriptions with minimal details. No
    mention of visual attributes like lighting, colors, resolution, or atmosphere.
  - Score 2: Moderate Visual Quality: Some visual attributes are present but lack
    specificity or coherence. Colors, lighting, and resolution are mentioned but not in
    depth.
  - Score 3: High Visual Quality: The prompt is highly descriptive, rich in visual
    attributes. Specific details about lighting, resolution, colors, textures, and
    clarity are included.

### Examples:
- Score 1: A beach with waves.
- Score 2: A snow-covered mountain with a few clouds in the sky.
- Score 3: An elderly man sitting on a worn leather armchair beside a crackling fireplace,
  the warm glow casting deep shadows on the wooden walls.

### Output format:
Always return your result in this format:
[RESULT] <a score between 0 and 3>
```

Listing 2. LLM Instruction for Visual Quality Scoring