Adaptive Video Distillation: Mitigating Oversaturation and Temporal Collapse in Few-Step Generation

Supplementary Material

Contents

DMD

rCM

Ours

An adorable happy otter confidently stands on a surfboard wearing a yellow lifejacket, riding along turquoise tropical waters near lush tropical islands, 3D digital render art style.

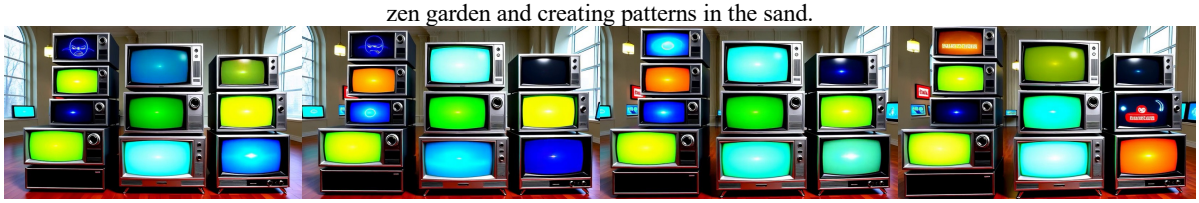
DMD

rCM

Ours

A close up view of a glass sphere that has a zen garden within it. There is a small dwarf in the sphere who is raking the zen garden and creating patterns in the sand.

DMD



rCM

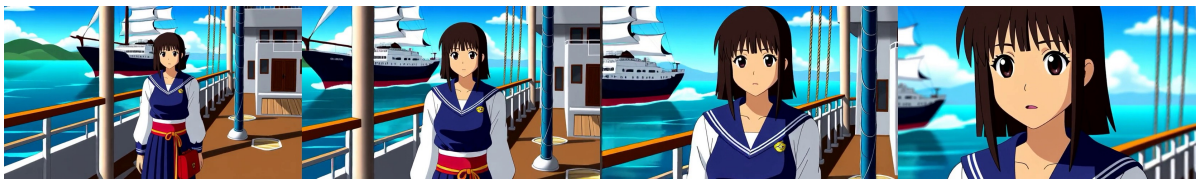
Ours

The camera rotates around a large stack of vintage televisions all showing different programs — 1950s sci-fi movies, horror movies, news, static, a 1970s sitcom, etc, set inside a large New York museum gallery.

DMD

rCM

Ours



A Japanese animated film of a young woman standing on a ship and looking back at camera.

DMD

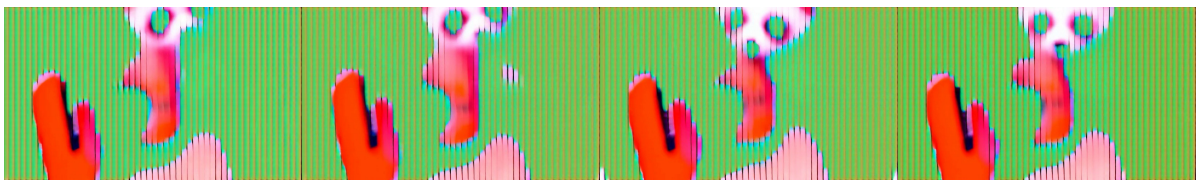
rCM

Ours

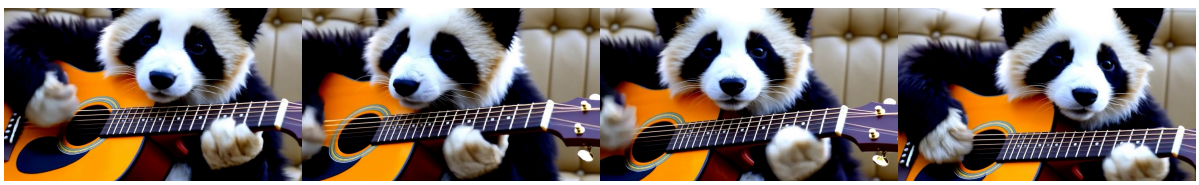
A woman singing and standing in a concert stage with a bright light in the background.

DMD

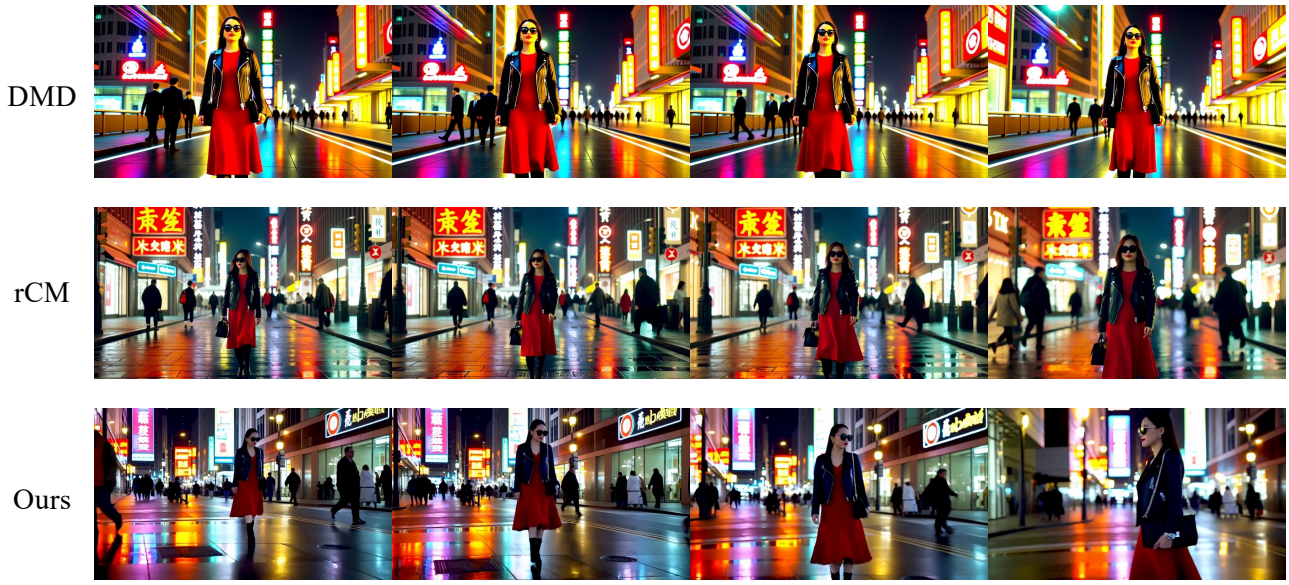
rCM



Ours



Panda playing the guitar



A stylish woman walks down a Tokyo street filled with warm glowing neon and animated city signage. She wears a black leather jacket, a long red dress, and black boots, and carries a black purse. She wears sunglasses and red lipstick. She walks confidently and casually. The street is damp and reflective, creating a mirror effect of the colorful lights. Many pedestrians walk about.

Animated scene features a close-up of a short fluffy monster kneeling beside a melting red candle. The art style is 3D and realistic, with a focus on lighting and texture. The mood of the painting is one of wonder and curiosity, as the monster gazes at the flame with wide eyes and open mouth. Its pose and expression convey a sense of innocence and playfulness, as if it is exploring the world around it for the first time. The use of warm colors and dramatic lighting further enhances the cozy atmosphere of the image.

Figure 10. **Comparison with baselines on MovieGen prompts.** While DMD and rCM (a strong performer from Sec. 4) produce videos with low motion dynamics and oversaturated colors, our method resolves both issues. It achieves superior motion, well-calibrated colors, and excellent detail and stability.More video examples are included in the supplementary zip file. (Note: We have confirmed that any visual artifacts in some videos are not code-level issues.)

6. Training Data

This section details the pipeline for the collection, filtering, and annotation of the video dataset used in our training.

6.1. Data Collection

For the construction of a high-fidelity training dataset, we curated video data from two primary channels: (1) publicly available, open-source videos from the internet, and (2) our internal, proprietary archives. To achieve broad diversity and coverage, we deliberately included videos from a multitude of categories. These range from everyday scenes like lifestyle, travel, and food, to more specialized domains such as aerial cinematography, medical imaging, and professional sports, among many others.

6.2. Data Filtering

Our data filtering pipeline consists of two main stages designed to ensure the quality of the training videos.

Preliminary Filtering by Resolution. We first decode each video to determine its spatial resolution. To ensure sufficient source quality, we filter out all videos with a width or height below 720p (1280×720). Subsequently, all remaining videos are uniformly resized to 480p for training, a process that helps preserve essential structural details from the high-resolution source material.

Frame-based Quality Assessment. From each remaining video, we uniformly sample four frames at fixed temporal positions (specifically, the 1st, 21st, 41st, and 61st frames). Each sampled frame must pass two complementary quality checks:

- **Monochromatic Scene Detection:** We convert the frame to the HSV color space and compute the entropy of its hue histogram. If the hue entropy is below a predefined threshold of 0.60, the frame is classified as part of a monochromatic scene (e.g., all-black, all-white, or solid-color backgrounds), and the corresponding video is rejected.
- **Blur Detection:** We assess the sharpness of the frame by calculating the variance of its Laplacian operator. If the Laplacian variance is lower than 20.0, the frame is deemed excessively blurry (potentially due to out-of-focus or motion blur issues), and the video is eliminated.

Optical Flow Analysis for Motion Dynamics. To ensure that our training videos contain sufficient dynamic content, we perform an optical flow analysis on all videos that passed the initial screening. We use an optical flow algorithm to extract motion vector fields between consecutive frames and then compute the average motion magnitude. Videos with a magnitude below 0.2 are considered to have insignificant motion (e.g., static scenes or overly stable footage) and are excluded from the training set.

Prompt:
"Please describe the details of this video, focusing on concrete aspects such as objects, people, colors, actions, and spatial relationships. Do not include any subjective descriptions related to symbolism, implication, or emotions."

Caption example:
"An elderly man with white hair and a mustache, wearing a red, blue, and white checkered shirt, is seen carefully hanging a large number of vibrant red chili peppers on a wall outdoors, likely in a rural or traditional setting. The scene is bathed in sunlight, highlighting the warmth of the day and the rich colors of the peppers. The man's focused expression and the meticulous way he handles the peppers suggest a deep connection to the cultural practice of preserving food."

Figure 11. Caption model prompt and caption style examples.

Temporal Consistency Filtering. To further enforce temporal coherence within video clips, we introduce an assessment based on frame-to-frame consistency. For the remaining videos, we calculate a temporal consistency score following the methodology of VBench1 [YourVBenchCitation]. We retain only the top 50% of videos ranked by this score.

Aesthetic Quality Filtering. Finally, we apply an aesthetic quality filter to ensure visual appeal. We employ a pre-trained visual aesthetic assessment model to score keyframes. Specifically, we uniformly sample 8 frames from each video, calculate their average aesthetic score, and keep only the top 50% of videos based on this metric.

6.3. Data Captioning

For the filtered data, we employ a proprietary 7B-parameter captioning model for annotation. The prompt and style examples used for the annotation process are shown in Figure 11.

7. Quality Visualize

To further evaluate our method's performance in practical video generation scenarios, we conducted a qualitative comparison on the official MovieGen prompt set against the

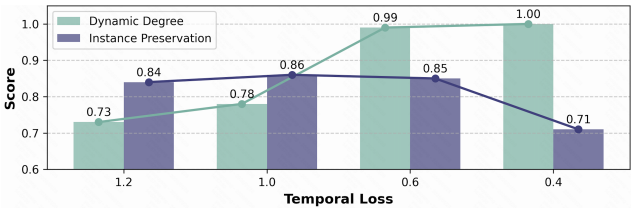


Figure 12. This figure illustrates the impact of the truncation threshold of the temporal regularization loss on model performance. The horizontal axis represents the truncation threshold of the regularization loss, while the vertical axis shows the motion score and the instance preservation score of the videos generated by the student model, respectively.

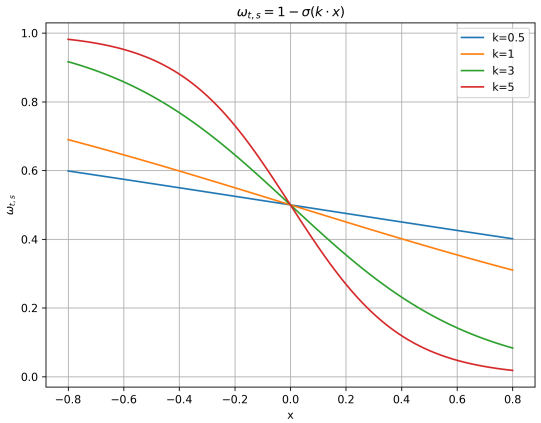


Figure 13. Visualization of the adaptive weight function in Eq. (7) for different values of k. The x-axis represents the deviation $\mathcal{L}_s - \bar{\mathcal{L}}_s$, while the y-axis represents the resulting adaptive weight.

DMD baseline and rCM, the latter of which performed favorably in our main experiments (Sec. 4). All methods generated videos under identical prompts and consistent sampling configurations to ensure a fair comparison. The results, as illustrated in Figures 8, 9, and 10, demonstrate that our method consistently achieves superior and more stable performance across multiple key dimensions. Specifically, in terms of **motion dynamics**, our approach generates videos with more extensive action, cinematic camera movements, and greater fluidity compared to the baselines. Regarding **color saturation and overall style**, our method produces visuals with well-calibrated saturation, avoiding the oversaturation artifacts common in other approaches and resulting in a style that aligns more closely with natural video distributions. Furthermore, concerning **detail quality**, our model excels at preserving fine-grained textures and sharp structural contours, effectively mitigating issues such as blurring and local structural degradation.

8. Frame-interpolation Module

Model Architecture. Our interpolation module is built upon a U-Net architecture designed to generate temporal intermediate frames between adjacent frames in the video's latent space. It takes as input the concatenated VAE-encoded latent features of two consecutive frames.

The network consists of a three-level downsampling encoder that progressively extracts high-level features, followed by a symmetric upsampling decoder that restores spatial resolution. Skip connections are employed to fuse high-resolution details from the encoder with deep semantic information from the decoder. The final output is an interpolated latent feature map with the same number of channels as a single frame's latent representation.

Specifically, the encoder is composed of three `ConvBlock` layers, each progressively expanding the channel dimensions and followed by max-pooling for downsampling. The decoder utilizes transposed convolutions for upsampling, concatenates the feature maps from the corresponding encoder level, and then fuses the information using another `ConvBlock`. A final 1×1 convolution is applied to produce the C-channel intermediate frame.

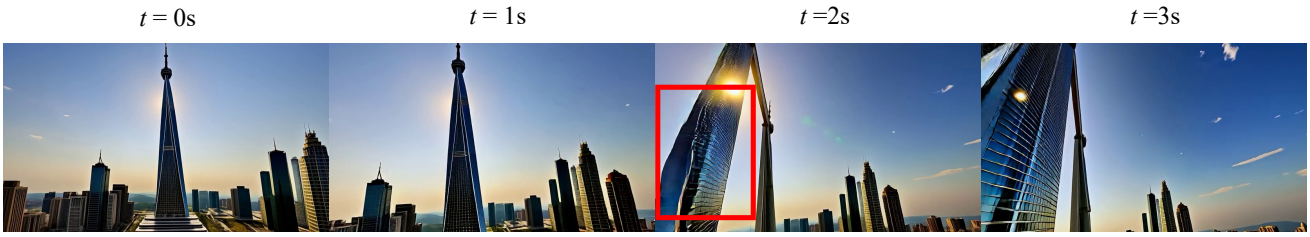
Training Details. We train the U-Net interpolation module on a dataset of 150,000 real-world video clips, each 5 seconds long, with a resolution of 480p. The training procedure is as follows:

For each video, we first encode it into the latent space using the pre-trained VAE. A standard video in our latent space consists of 21 frames (corresponding to 81 frames at 16 fps in the pixel space over 5 seconds). From this 21-frame latent sequence, we uniformly downsample it to 11 frames. The U-Net module then takes these 11 frames and performs interpolation to restore the sequence to its original length. The network is optimized by computing a regression loss between the interpolated latent video and the original 21-frame ground-truth latent video.

We use the AdamW optimizer with a learning rate of 1×10^{-5}, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. The model is trained for 10,000 iterations with a batch size of 32.

Inference Process. During inference, the interpolator is invoked along the temporal dimension. For each pair of adjacent frames in the input sequence, the original frames are preserved. The two frames are then concatenated and fed into the U-Net to generate one intermediate frame, which is inserted in order between them. This process expands the sequence length from F to $2F - 1$, effectively doubling the video's frame rate.

The overall architecture is simple yet highly effective, capable of generating smooth and structurally consistent



"Burj Khalifa, zoom out."

"Blue Lagoon, zoom out."

Figure 14. **Failure cases resulting from unclipped temporal regularization.** Without clipping, the student generator produces severe artifacts late in training. **(Top)** After the first second, a drastic content shift occurs, accompanied by a noticeable distortion of the building (highlighted by the red box). **(Bottom)** The scene content abruptly vanishes at the two-second mark and is replaced by another major content shift at the three-second mark (highlighted by the red box). These phenomena, inconsistent with plausible camera motion, are clear manifestations of hallucinations. This highlights the necessity of clipping the temporal loss to prevent it from excessively amplifying inter-frame variance.

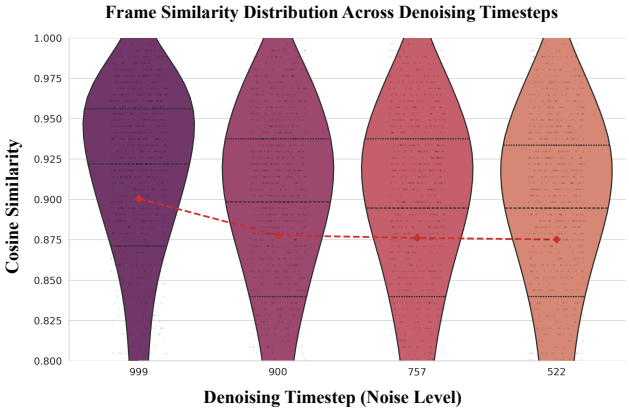


Figure 15. Violin plot showing the distribution of adjacent-frame cosine similarity across different denoising stages. The overall similarity is notably higher during the high-noise stage, which motivates our approach of halving the inference frame rate in this phase to reduce computational cost.

temporal intermediates in the latent space. It is suitable for applications such as video frame-rate up-sampling, transition generation, and interpolation tasks within diffusion models.

9. Hyperparameter Discuss

In this section, we discuss the hyperparameter values used during the training process. For most settings, we follow the configuration from Wan2.1. The specific parameters are as follows:

- **Teacher Model:** The number of training timesteps is set to 1000. The classifier-free guidance scale is 5.0, and the timestep-shift is also 5.0.
- **AdamW Optimizer:** The parameters are set as follows: learning rate (lr) = 2.0×10^{-6}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and the maximum gradient norm (`max_grad_norm`) is 10.0.
- **Two-Timescale Update Rule:** The update frequency is set to 5. This means that the student generator is updated once for every five updates of the online model.

9.1. Temporal Regularization

The temporal loss is updated using the formulation in Eq. (8); we simply set ϵ to 1×10^{-6}. Theoretically, this loss function incentivizes a continuous increase in the temporal variance of the model's output, as a larger variance results in a smaller negative logarithm and thus a lower loss. Consequently, the loss lacks a natural convergence mechanism. This can lead to numerical instability, excessively large gradients, and potentially exploding gradients. Furthermore, the model's output can be amplified indefinitely, causing anomalous temporal variations that manifest as excessive jitter or noise in the generated videos. This regularization term may also conflict with the primary training objective; for instance, it creates an optimization paradox if the main task promotes smoothness while this term encourages variance.

Empirically, we observed that without any clipping, this loss causes the model to generate videos with severe frame jumps or hallucinatory artifacts in the later stages of training (as shown in Figure 14).

Therefore, it is necessary to clip this loss once it falls below a certain value. To establish a reasonable clipping threshold, we first computed this loss on the VAE-encoded latents of 4,000 videos generated by the teacher model. The average value was found to be approximately 1.5. Based on this observation, we experimented with three distinct clipping thresholds: 1.2, 1.0, 0.6 and 0.4. We trained the model for a sufficient and equal number of epochs under each setting. The dynamic degree score and instance preservation score of the videos generated by the student are presented in Figure 12. We observed that setting the threshold to 0.6 significantly improves motion dynamics while maintaining object generation quality. Furthermore, to ensure that the distribution matching and adaptive regression losses remain the dominant sources of gradients during distillation, we found that setting the weight ω_{temp} for this regularization term to 0.05 provides an effective balance.At this weight, the temporal loss typically converges to near the clipping threshold during training, while its weighted magnitude remains a small fraction compared to the other two losses.

9.2. Adaptive Regression Loss

We now discuss the selection of key hyperparameters.

First, the parameter α in Eq. (6) governs the contribution of historical losses to the exponential moving average (EMA); we adopt a commonly used value of 0.95.

Next, in Eq. (7), the parameter k controls the slope of the sigmoid function. This, in turn, determines how sensitively the adaptive weight responds to the deviation of the current loss $\mathcal{L}_s$ from its historical trend $\bar{\mathcal{L}}_s$. Notably, for low-noise steps, the absolute value of the regression loss $\mathcal{L}_s$ is typically very small (often below 0.01). At this stage, the guidance from the regression loss is inherently limited, as the input already contains significant information from the real image. Consequently, the output of the sigmoid function remains close to 0.5, making the weight largely insensitive to the value of k. Therefore, our analysis primarily focuses on the impact of k during the high-noise denoising stages.

As visualized in Figure 13, which plots the weight function for several values of k (where the x-axis is $\mathcal{L}_s - \bar{\mathcal{L}}_s$), we identified two desired behaviors. **First**, during the initial training phases, when the distribution gap is large, $\mathcal{L}_s - \bar{\mathcal{L}}_s$ can reach values between 0.6 and 0.8. We want to avoid excessively penalizing these data points (i.e., the weight should not be too close to 0 in this range). **Second**, in the later stages, when $\mathcal{L}_s - \bar{\mathcal{L}}_s$ has largely converged to the [-0.3, 0.3] interval, the weight function should still retain sufficient discriminative power to differentiate between samples. Through experimentation, we found that setting $k = 3.0$ provides an effective trade-off that satisfies both conditions.

Finally, for w_{reg} in Eq. (9), we set its value to 2.0. This choice normalizes the weight to approximately 1.0 for data points where the current loss is near the historical average (i.e., $\mathcal{L}_s - \bar{\mathcal{L}}_s \approx 0$).