Rethink the Noise Prior of Initialization Gap in Video Diffusion Models

Tian Xia   Yinuo Yang   Sihan Xu
University of Michigan
EECS 442 Course Project

Abstract

Video Diffusion Models have advanced video generation by integrating text and image conditioning, offering enhanced control over generated content. However, maintaining consistency across frames remains a challenge, especially when using text prompts as control conditions. Some approaches, like FreeInit, address this by iteratively updating the initial noise to ensure video consistency, while methods like UniCtrl employ Attention Control to maintain spatiotemporal consistency. Yet, these techniques incur additional computational costs and inference time. Addressing the need for stable and consistent video generation without extra computational expense remains an open problem. In this paper, we revisit the noise prior to the initialization gap in video diffusion models and introduce a novel initialization method ***FastFreeInit****. By partially sharing the initial noise across different frames, we achieve enhanced consistency and stability in video generation without additional computational demands, as verified by our experiments.*

1. Introduction

Diffusion Models (DMs) have demonstrated superior performance in image synthesis, surpassing traditional methods such as GANs [12, 25, 26] and VAEs [27, 37, 49] in terms of stability and quality. Early research [19, 24, 29, 43–45] laid the essential groundwork for DMs, proving their effectiveness in scaling with varied datasets. Recent innovations [30, 32, 36, 38, 40, 58, 60] have enhanced their controllability and interaction with users, facilitating the generation of images that more accurately align with user specifications.

Recently developed Video Diffusion Models (VDMs) [20] have employed Diffusion Models (DMs) for generating videos. VDMs demonstrate their ability to produce videos depicting a range of motions in text-to-video synthesis tasks, facilitated by the integration of text encoders [35], as evidenced in works like [1, 2, 16, 18, 22, 59]. Various open-source text-to-video models have emerged, such as ModelScope [51], AnimateDiff [16], and VideoCrafter [6]. These models often rely on a pre-trained image generation model like Stable Diffusion (SD) [39] and incorporate additional temporal or motion components. Despite this, texts, unlike images which are rich in semantic content, struggle to maintain frame-to-frame consistency in video outputs. Concurrently, some research utilizes image conditions to foster image-to-video transformations with enhanced spatial semantic consistency [1,15,23]. While there are approaches proposing a text-to-image-to-video framework [11], relying solely on image conditions often falls short in controlling video motion. Combining both text and image conditions enhances spatiotemporal consistency in a mixed text-and-image-to-video process [6, 7, 14, 55, 61], though these techniques necessitate additional training.

Currently, Video Diffusion Models (VDMs) typically incorporate additional temporal layers into a 2D UNet; however, this modification fails to adequately address cross-frame constraints during the training of the 2D UNet model. Various training-free methods, such as those documented in recent research [8,13,34,54], have attempted to improve the smoothness of generated videos by refining the start noise or taking the use of attention control [4, 17, 48, 57]. Despite these efforts, the challenge of maintaining consistent cross-frame coherence in videos produced by VDMs remains unresolved. In this paper, we reevaluate the noise prior to the initialization gap in video diffusion models and introduce a new initialization technique, **FastFreeInit**. By sharing initial noise partially across different frames, we achieve enhanced consistency and stability in video generation without imposing additional computational costs, as confirmed by our experimental results.

2. Background

Video Generation Numerous studies have explored the realm of video generation, employing various approaches like GAN-based frameworks [3, 42, 47] and transformer-based architectures [21, 50, 52, 53]. Building on the success of Diffusion models (DMs) [19, 24, 29, 43–45], which have delivered impressive outcomes in image synthesis [31, 32, 36, 38, 40], video diffusion models (VDMs) [20] have also showcased their prowess in generating videos

[2, 6, 7, 11, 14, 16, 18, 22, 41, 51, 55, 59].

Presently, VDMs typically integrate extra temporal layers into a 2D UNet, yet this adaptation does not sufficiently address cross-frame constraints during the training of the 2D UNet model. Several techniques [8, 13, 34, 54] have experimented with training-free approaches to enhance the smoothness of the generated videos. Nonetheless, the challenge of maintaining consistent cross-frame coherence in videos produced by VDMs persists.

Noise in Diffusion Models Only a few studies have highlighted the drawbacks in the noise schedules of existing diffusion models. In the realm of image synthesis, [28] identifies that traditional diffusion noise schedules do not entirely obscure the information in natural images, which constrains the model to produce only images of moderate brightness. Building on this, [9] further investigates the issue of signal leakage and introduces a method to explicitly model this leakage for an improved inference noise distribution, resulting in images of greater brightness and color diversity.

In the context of video, PYoCo [10] meticulously formulates a progressive video noise prior to enhanced video generation. Echoing [28], PYoCo also emphasizes noise schedule adjustments during the training phase and necessitates extensive fine-tuning on video datasets. Recent initiatives [13,34] similarly focus on the initial noise at inference, albeit with a goal of producing longer videos. FreeInit [54] aims to elevate inference quality and further incorporates tailored frequency-domain operations to adjust various frequency components of the initial noise, but it needs additional computational costs and inference time.

3. Method

Although Video Diffusion Models (VDMs) have achieved notable success in video generation, most open-source VDMs still struggle with consistency and stability in their generated videos. Research has indicated that the consistency of VDM-generated videos can be influenced by the initial noise [54], while inconsistencies in the attention mechanism's value can lead to unstable video outputs [8]. FreeInit [54] addresses this by iteratively updating the initial noise to ensure consistency across frames, whereas UniCtrl [8] enhances video quality by managing the consistency of values within the attention mechanism. However, these methods require additional inference time and computational resources. FreeInit involves multiple sampling processes, and UniCtrl necessitates concurrent inference across multiple branches. Given the substantial memory and computation demands of VDMs, these additional costs are often impractical. Therefore, exploring ways to enhance the consistency and stability of video generation without increasing computational costs remains a pressing issue. Our approach, inspired by PYoCo [10], mixes the noise from the first frame with that of subsequent frames and, following FreeInit [54], blends this combined noise at various frequencies. This method achieves improved spatio-temporal consistency in generated videos without additional computational overhead.

Noise Mixing Consider the scenario where $\varepsilon^1, \varepsilon^2, \ldots, \varepsilon^{n_s}$ represent the specific noises associated with each frame of a video, with ε^i being the i^{th} noise element in the noise tensor ε. Following PYoCo [10], we define two distinct types of noise vectors: $\varepsilon_{\text{shared}}$ and ε_{ind}. The vector $\varepsilon_{\text{shared}}$ serves as a universal noise component common to all frames, whereas ε_{ind} consists of unique noise vectors tailored to each individual frame. These two vectors are then combined linearly to form the noise applied to each frame.

Mathematically, this is described by the following formulation:

$$\varepsilon_{\text{shared}} \sim N\left(0, \frac{\alpha^2}{1+\alpha^2} I\right), \varepsilon^i_{\text{ind}} \sim N\left(0, \frac{1}{1+\alpha^2} I\right) \quad (1)$$
$$\varepsilon^i_{new} = \varepsilon_{\text{shared}} + \varepsilon^i_{\text{ind}}$$

This structure ensures that while each frame benefits from a base level of commonality due to the shared noise, it also preserves a degree of uniqueness through the individual noise components, thus balancing consistency and variation across the video sequence. Here, we use the noise of the first frame from the latent as our shared ε.

Noise Reinitialization Like FreeInit [54], we utilize a spatio-temporal frequency filtering technique, whereby we amalgamate the low-frequency elements of the original noise latent vector z_T with the high-frequency elements of a newly generated random Gaussian noise η. This yields a dynamically reinitialized noisy latent vector z'_T. By employing this method, we retain the critical information embedded within the low-frequency components of z_T, while infusing variability in the high-frequency spectrum to enrich visual textures and details. The formulation of this process is detailed in the following mathematical expressions:

$$\mathcal{F}^L_{z_T} = \mathcal{FFT}3D(z_T) \odot \mathcal{H}, \quad (2)$$
$$\mathcal{F}^H_{\eta} = \mathcal{FFT}3D(\eta) \odot (1 - \mathcal{H}), \quad (3)$$
$$z'_T = \mathcal{IFFT}3D(\mathcal{F}^L z_T + \mathcal{F}^H \eta), \quad (4)$$

Here, $\mathcal{FFT}3D$ denotes the Three-Dimensional Fast Fourier Transformation applied across both spatial and temporal dimensions, and $\mathcal{IFFT}3D$ is the Inverse Fast Fourier Transformation that reconstructs the combined latent z'_T from the frequency domain to the time-space domain. The filter $\mathcal{H}$ represents a Spatial-Temporal Low Pass Filter (LPF), designed to match the dimensions of the latent, facilitating selective frequency blending.

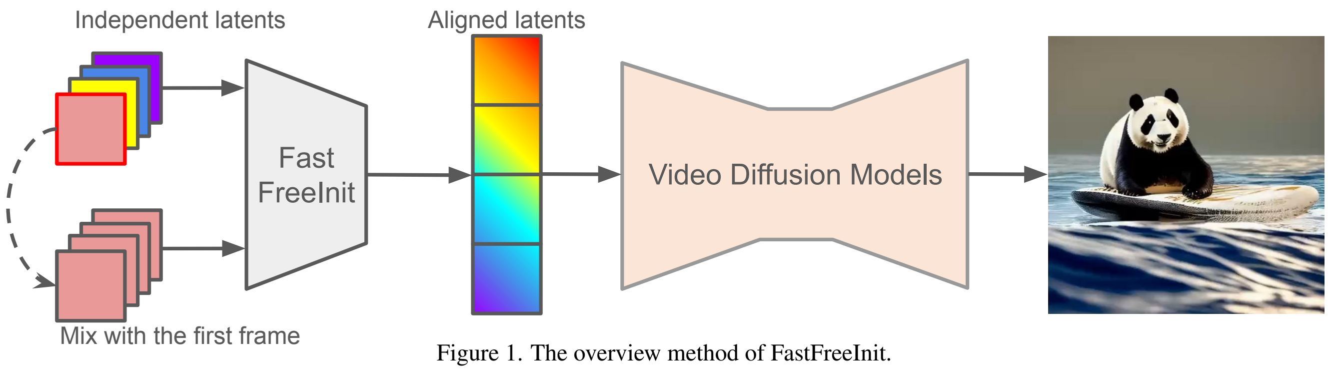

Figure 1. The overview method of FastFreeInit.

FastFreeInit To improve the consistency of videos in text-to-video tasks, we focus on manipulating the initial noise. According to Ge et al. [10], controlling the initial noise may sometimes result in sub-optimal consistency of the generated videos. Our method for manipulating initial noise is structured into two distinct steps:

1) Adopting the Mixed Noise Model methodology proposed by Ge et al. [10], we control the noise for each frame as comprising both shared and independent components. The shared component is directly sourced from the noise of the first frame, while the independent component consists of initial random noise. Consequently, the noise for each subsequent frame is composed of the first frame's noise combined with its original random noise. The corresponding formula is presented as 1 and the formula applied is

$$\varepsilon^i_{new} = \frac{\alpha^2}{1+\alpha^2} * \varepsilon^1 + (1 - \frac{\alpha^2}{1+\alpha^2}) * \varepsilon^i_{\text{orignal}}$$

2) Building on the findings of Wu et al., substantial spatio-temporal correlations exist within the low-frequency components, we extend those correlations from the first frame to others like 2. Therefore, we preserve the low-frequency components of the noise processed in the first step, adding the original latent high-frequency components.

Here is the full algorithm:

Algorithm 1 FastFreeInit Algorithm

1: $latents \leftarrow random\ noises for all frames$
2: $latents_{low} \leftarrow the\ first\ frame\ of\ latents$
3: $a \leftarrow \frac{\alpha^2}{1+\alpha^2}$ and $b \leftarrow 1 - a$
4: $latents_{low} \leftarrow a \times latents_{low} + b \times latents$
5: $latents \leftarrow freq\_mix\_3d(latents_{low}, latents)$

Implement Details We began with the official repository of FreeInit [54] and integrated our FastFreeInit pipeline based on the FreeInit template, incorporating an initial noise control algorithm prior to the denoising step. We utilized the formula and the freq_mix_3d function from FreeInit to manage the noise control. We ensured that the initial random noise was consistent across all three approaches and timed them within the main program. The core code is detailed in the appendix.

For the evaluation phase, we employed relevant models from Hugging Face to assess various metrics. We simultaneously generated and evaluated those videos with the prompt text. The relevant code is provided in the appendix for further reference.

4. Results

To assess the effectiveness of our model, we utilize prompts from two datasets: UCF-101 [46] and MSR-VTT [56] to generate videos. In line with the approach of Ge et al [10] and Chen et al [8], we employ identical prompts from the UCF-101 dataset for our experiments. Additionally, we select 100 unique prompts from the MSR-VTT dataset to further evaluate our model. These selections form our comprehensive dataset for evaluation. Next, we provide a brief introduction to evaluation metrics and backbone.

4.1. Metric

To quantitatively measure the performance of our model, we employ standard metrics as outlined in [41, 54].

• *Clip Similarity*: To measure the relevance between videos and texts, we compute the average Clip Similarity [52] for each generated video with their corresponding prompt. We calculate the score by averaging Clip Similarity for each frame in the video with its prompt. In our experiment, we compute the CLIP similarity utilizing TorchMetrics with clip-vit-base-patch32 model [35].

• *DINO*: To assess the spatiotemporal consistency of the generated videos, we utilize DINO [33] to compute the cosine similarity between the initial frame and subsequent frames. The average DINO score across all consecutive frames is then used as the video's overall score. In our experiments, we utilize the DINO-vits16 [5] model to compute the DINO cosine similarity.

A man narrates his minecraft gameplay

A car is shown

The woman sings into the microphone

Figure 2. Qualitative Comparisons

Table 1. Quantitative Comparisons on UCF-101 and MSR-VTT. FastFreeInit significantly improves the temporal consistency without adding too much extra time for generating video. I indicates the number of iterations for FreeInit.

Method	CLIP (↑)	DINO (↑)	Time (↓)
AnimateDiff [16]	95.18	97.18	74.40s
FreeInit + AnimateDiff ($I = 3$)	96.95	98.32	218.91s
FastFreeInit + AnimateDiff	97.69$_{(+00.74)}$	99.11$_{(+00.79)}$	74.40s$_{(-141.51s)}$

4.2. Backbones

Given the plug-and-play nature of our approach, we opted to test our methods using AnimateDiff. *AnimateDiff* [16] provides a practical framework to impart motion dynamics into personalized text-to-image models like those developed through Stable Diffusion. This is achieved without necessitating adjustments specific to each model. At the core of AnimateDiff lies a motion module. Once trained, this module can be universally applied to various personalized text-to-image models that share the same foundational model, utilizing transferable motion priors derived from real-world videos to enable animation.

4.3. Baseline

We chose FreeInit as our baseline because it is a training-free method that aims to enhance the appearance and temporal consistency of generated videos. It does this by iteratively refining the spatial-temporal low-frequency components of the initial latent code during inference. Given that both FreeInit and FastFreeInit are training-free methods designed to improve spatiotemporal consistency in video generation through diffusion models, it is logical to compare the performance of FastFreeInit against FreeInit.

4.4. Qualitative Comparisons

For qualitative comparisons, shown in Figure 2, reveal that our FastFreeInit method markedly improves spatiotemporal consistency and adheres closely to the text. For instance, using the text prompt 'A man narrates his Minecraft gameplay', the standard AnimateDiff method would cause abrupt transitions from a green-black box to a blue pool. In contrast, FastFreeInit maintains greater consistency with the elements. Additionally, FreeInit would have the ground details change inconsistently, whereas FastFreeInit preserves better consistency in these feature details as well. Furthermore, FastFreeInit is more aligned with the prompt, as it avoids visualizing the narrator, which is implied by the text that the man should not appear in the video. Both Normal AnimateDiff and FastFreeInit cause Minecraft characters to appear in the video.

4.5. Quantitative Comparisons

For quantitative comparisons, the results for UCF-101 and MSRVTT are presented in Table 1. We compare the base backbone with it augmented by FastFreeInit and FreeInit, respectively. The result demonstrates that our FastFreeInit method significantly enhances spatiotemporal consistency and preserves the meaning of text within the video. Additionally, it offers substantial time savings compared to the FreeInit method while improving both CLIP and DINO scores. The improvement in the CLIP score from 96.95 to 97.69 suggests that our method adheres more closely to the prompt instructions than the original method and FreeInit. Furthermore, the enhancement in the DINO score from 98.32 to 99.11 indicates that our method improves video consistency over the other two methods. In conclusion, our FastFreeInit method outperforms the other approaches in these three critical aspects.

5. Conclusion

We present FastFreeInit as a novel solution aimed at enhancing cross-frame consistency and stability in Video Diffusion Models without the need for additional training. By ingeniously managing the initial noise distribution across frames, FastFreeInit significantly improves the spatiotemporal consistency in generated videos. This method is distinguished by its ease of integration with existing models and does not require extensive fine-tuning, ensuring broad applicability across different video generation frameworks. The performance of FastFreeInit has been thoroughly validated through rigorous testing, confirming its effectiveness and showcasing its potential as a versatile tool for video generation models.

References

[1] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 1

[2] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023. 1, 2

[3] Tim Brooks, Janne Hellsten, Miika Aittala, Ting-Chun Wang, Timo Aila, Jaakko Lehtinen, Ming-Yu Liu, Alexei

Efros, and Tero Karras. Generating long videos of dynamic scenes. *Advances in Neural Information Processing Systems*, 35:31769–31781, 2022. 1

[4] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. Masactrl: Tuning-free mutual self-attention control for consistent image synthesis and editing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22560–22570, October 2023. 1

[5] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. 2021. 3

[6] Haoxin Chen, Menghan Xia, Yingqing He, Yong Zhang, Xiaodong Cun, Shaoshu Yang, Jinbo Xing, Yaofang Liu, Qifeng Chen, Xintao Wang, Chao Weng, and Ying Shan. Videocrafter1: Open diffusion models for high-quality video generation, 2023. 1, 2

[7] Haoxin Chen, Yong Zhang, Xiaodong Cun, Menghan Xia, Xintao Wang, Chao Weng, and Ying Shan. Videocrafter2: Overcoming data limitations for high-quality video diffusion models, 2024. 1, 2

[8] Xuweiyi Chen, Tian Xia, and Sihan Xu. Unictrl: Improving the spatiotemporal consistency of text-to-video diffusion models via training-free unified attention control, 2024. 1, 2, 3

[9] Martin Nicolas Everaert, Athanasios Fitsios, Marco Bocchio, Sami Arpa, Sabine Süsstrunk, and Radhakrishna Achanta. Exploiting the signal-leak bias in diffusion models. *arXiv preprint arXiv:2309.15842*, 2023. 2

[10] Songwei Ge, Seungjun Nah, Guilin Liu, Tyler Poon, Andrew Tao, Bryan Catanzaro, David Jacobs, Jia-Bin Huang, Ming-Yu Liu, and Yogesh Balaji. Preserve your own correlation: A noise prior for video diffusion models. In *ICCV*, 2023. 2, 3

[11] Rohit Girdhar, Mannat Singh, Andrew Brown, Quentin Duval, Samaneh Azadi, Sai Saketh Rambhatla, Akbar Shah, Xi Yin, Devi Parikh, and Ishan Misra. Emu video: Factorizing text-to-video generation by explicit image conditioning. *arXiv preprint arXiv:2311.10709*, 2023. 1, 2

[12] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 1

[13] Jiaxi Gu, Shicong Wang, Haoyu Zhao, Tianyi Lu, Xing Zhang, Zuxuan Wu, Songcen Xu, Wei Zhang, Yu-Gang Jiang, and Hang Xu. Reuse and diffuse: Iterative denoising for text-to-video generation. *arXiv preprint arXiv:2309.03549*, 2023. 1, 2

[14] Xianfan Gu, Chuan Wen, Jiaming Song, and Yang Gao. Seer: Language instructed video prediction with latent diffusion models. *arXiv preprint arXiv:2303.14897*, 2023. 1, 2

[15] Yuwei Guo, Ceyuan Yang, Anyi Rao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Sparsectrl: Adding sparse controls to text-to-video diffusion models. *arXiv preprint arXiv:2311.16933*, 2023. 1

[16] Yuwei Guo, Ceyuan Yang, Anyi Rao, Yaohui Wang, Yu Qiao, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. *arXiv preprint arXiv:2307.04725*, 2023. 1, 2, 5

[17] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-or. Prompt-to-prompt image editing with cross-attention control. In *The Eleventh International Conference on Learning Representations*, 2023. 1

[18] Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P Kingma, Ben Poole, Mohammad Norouzi, David J Fleet, et al. Imagen video: High definition video generation with diffusion models. *arXiv preprint arXiv:2210.02303*, 2022. 1, 2

[19] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 1

[20] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in Neural Information Processing Systems*, 2022. 1

[21] Wenyi Hong, Ming Ding, Wendi Zheng, Xinghan Liu, and Jie Tang. Cogvideo: Large-scale pretraining for text-to-video generation via transformers. *arXiv preprint arXiv:2205.15868*, 2022. 1

[22] Yaosi Hu, Chong Luo, and Zhenzhong Chen. Make it move: controllable image-to-video generation with text descriptions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18219–18228, 2022. 1, 2

[23] Zhihao Hu and Dong Xu. Videocontrolnet: A motion-guided video-to-video translation framework by using diffusion model with controlnet. *arXiv preprint arXiv:2307.14073*, 2023. 1

[24] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*, 35:26565–26577, 2022. 1

[25] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1

[26] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020. 1

[27] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014. 1

[28] Shanchuan Lin, Bingchen Liu, Jiashi Li, and Xiao Yang. Common diffusion noise schedules and sample steps are flawed. *arXiv preprint arXiv:2305.08891*, 2023. 2

[29] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *arXiv preprint arXiv:2310.04378*, 2023. 1

[30] Chong Mou, Xintao Wang, Liangbin Xie, Jian Zhang, Zhongang Qi, Ying Shan, and Xiaohu Qie. T2i-adapter: Learning adapters to dig out more controllable ability for text-to-image diffusion models. *arXiv preprint arXiv:2302.08453*, 2023. 1

[31] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pages 8162–8171. PMLR, 2021. 1

[32] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning*, pages 16784–16804. PMLR, 2022. 1

[33] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision. 2024. 3

[34] Haonan Qiu, Menghan Xia, Yong Zhang, Yingqing He, Xintao Wang, Ying Shan, and Ziwei Liu. Freenoise: Tuning-free longer video diffusion via noise rescheduling. *arXiv preprint arXiv:2310.15169*, 2023. 1, 2

[35] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 3

[36] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022. 1

[37] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021. 1

[38] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1

[39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022. 1

[40] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022. 1

[41] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 2, 3

[42] Ivan Skorokhodov, Sergey Tulyakov, and Mohamed Elhoseiny. Stylegan-v: A continuous video generator with the price, image quality and perks of stylegan2. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3626–3636, 2022. 1

[43] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015. 1

[44] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2020. 1

[45] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. *arXiv preprint arXiv:2303.01469*, 2023. 1

[46] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. 2012. 3

[47] Yu Tian, Jian Ren, Menglei Chai, Kyle Olszewski, Xi Peng, Dimitris N Metaxas, and Sergey Tulyakov. A good image generator is what you need for high-resolution video synthesis. *arXiv preprint arXiv:2104.15069*, 2021. 1

[48] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-play diffusion features for text-driven image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1921–1930, June 2023. 1

[49] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 1

[50] Ruben Villegas, Mohammad Babaeizadeh, Pieter-Jan Kindermans, Hernan Moraldo, Han Zhang, Mohammad Taghi Saffar, Santiago Castro, Julius Kunze, and Dumitru Erhan. Phenaki: Variable length video generation from open domain textual description. *arXiv preprint arXiv:2210.02399*, 2022. 1

[51] Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. Modelscope text-to-video technical report, 2023. 1, 2

[52] Chenfei Wu, Lun Huang, Qianxi Zhang, Binyang Li, Lei Ji, Fan Yang, Guillermo Sapiro, and Nan Duan. Godiva: Generating open-domain videos from natural descriptions. *arXiv preprint arXiv:2104.14806*, 2021. 1, 3

[53] Chenfei Wu, Jian Liang, Lei Ji, Fan Yang, Yuejian Fang, Daxin Jiang, and Nan Duan. Nüwa: Visual synthesis pretraining for neural visual world creation. In *European conference on computer vision*, pages 720–736. Springer, 2022. 1

[54] Tianxing Wu, Chenyang Si, Yuming Jiang, Ziqi Huang, and Ziwei Liu. Freeinit: Bridging initialization gap in video diffusion models. *arXiv preprint arXiv:2312.07537*, 2023. 1, 2, 3

[55] Jinbo Xing, Menghan Xia, Yong Zhang, Haoxin Chen, Xintao Wang, Tien-Tsin Wong, and Ying Shan. Dynamicrafter: Animating open-domain images with video diffusion priors. *arXiv preprint arXiv:2310.12190*, 2023. 1, 2

[56] Jun Xu, Tao Mei, Ting Yao, and Yong Rui. Msr-vtt: A large video description dataset for bridging video and language. June 2016. 3

[57] Sihan Xu, Yidong Huang, Jiayi Pan, Ziqiao Ma, and Joyce Chai. Inversion-free image editing with natural language. *arXiv preprint arXiv:2312.04965*, 2023. 1

[58] Sihan Xu, Ziqiao Ma, Yidong Huang, Honglak Lee, and Joyce Chai. Cyclenet: Rethinking cycle consistent in text-guided diffusion for image manipulation. In *Advances in Neural Information Processing Systems*, 2023. 1

[59] David Junhao Zhang, Jay Zhangjie Wu, Jia-Wei Liu, Rui Zhao, Lingmin Ran, Yuchao Gu, Difei Gao, and Mike Zheng Shou. Show-1: Marrying pixel and latent diffusion models for text-to-video generation. *arXiv preprint arXiv:2309.15818*, 2023. 1, 2

[60] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023. 1

[61] Yiming Zhang, Zhening Xing, Yanhong Zeng, Youqing Fang, and Kai Chen. Pia: Your personalized image animator via plug-and-play modules in text-to-image models. *arXiv preprint arXiv:2312.13964*, 2023. 1

A. Core Algorithm Code

```
latents_low = latents[:, :, 0, :, :]
z_T = latents_low.unsqueeze(2).expand(-1, -1, latents.shape[2], -1, -1)

alpha_sqr = alpha * alpha
a = alpha_sqr / (1 + alpha_sqr)
b = 1 - a
z_T = a * z_T + b * latents

latents = freq_mix_3d(
    z_T.to(dtype=torch.float32), latents, LPF=self.freq_filter
)
```

B. Evaluation Code

```
device = torch.device('cuda' if torch.cuda.is_available() else "cpu")
processor_clip = CLIPProcessor.from_pretrained("openai/clip-vit-base-patch32")
model_clip = CLIPModel.from_pretrained("openai/clip-vit-base-patch32").to(device)

processor_dino = ViTImageProcessor.from_pretrained("facebook/dino-vits16")
model_dino = ViTModel.from_pretrained("facebook/dino-vits16").to(device)

def clip_score(outputs, processor, model, device):
    image_features = []
    with torch.no_grad():
        for output in outputs:
            input_tmp = processor(images=output, return_tensors="pt").to(device)
            image_feature = model.get_image_features(**input_tmp)
            image_features.append(image_feature)
        cos = nn.CosineSimilarity(dim=0)
        Clip = 0
        for i in range(1,len(image_features)):
            sim = cos(image_features[i-1][0],image_features[i][0]).item()
            sim = (sim+1)/2
            Clip += sim
        return Clip / 15

def dino_score(outputs, processor, model, device):
    image_features = []
    with torch.no_grad():
        for output in outputs:
            input_tmp = processor(images=output, return_tensors="pt").to(device)
            image_feature = model(**input_tmp).last_hidden_state
            image_feature = image_feature.mean(dim=1)
            image_features.append(image_feature)
        cos = nn.CosineSimilarity(dim=0)
        Dino = 0
        for i in range(1,len(image_features)):
            sim = cos(image_features[i-1][0],image_features[i][0]).item()
            sim = (sim+1)/2
            Dino += sim
        return Dino / 15
```