StreamDiT: Real-Time Streaming Text-to-Video Generation

Akio Kodaira[1,2*] Tingbo Hou[2,*] Ji Hou[2] Markos Georgopoulos[2]
Felix Juefei-Xu[2] Masayoshi Tomizuka[1] Yue Zhao[2,*]

[1]UC Berkeley [2]Meta

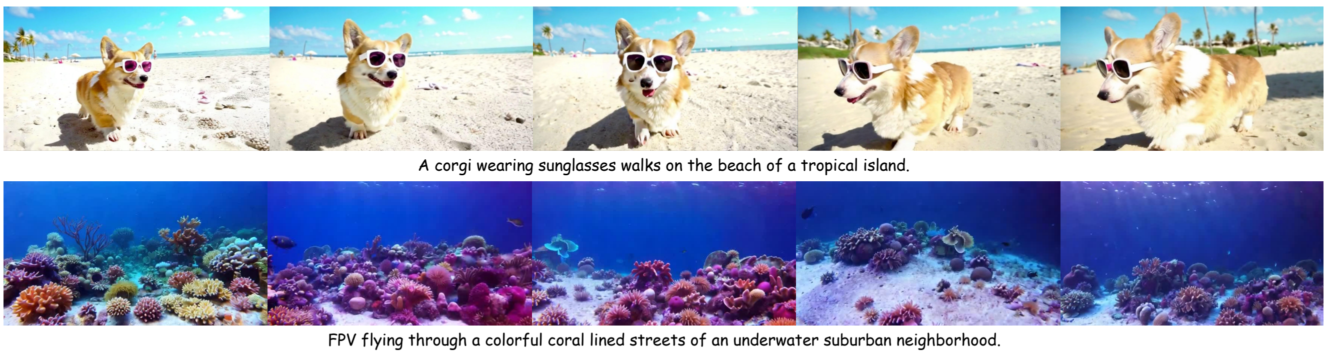


Figure 1. StreamDiT-4B: video generation can be streaming and real-time.

Abstract

Recently, great progress has been achieved in text-to-video (T2V) generation by scaling transformer-based diffusion models to billions of parameters. However, existing models typically produce only short clips offline, restricting their use cases in interactive and real-time applications. This paper addresses this challenge by proposing StreamDiT, an efficient trainable solution for streaming video generation. StreamDiT training modifies flow matching by adding a moving buffer with progressive denoising. We design mixed partitioning schemes of buffered frames to boost both content consistency and visual quality. StreamDiT modeling is based on adaLN DiT with varying time embedding and window attention. In addition, we propose a multistep distillation method tailored for StreamDiT. Sampling distillation is performed in each segment of a chosen partitioning scheme. After distillation, the total number of function evaluations (NFEs) is reduced to the number of chunks in a buffer. We apply StreamDiT to a 4B-parameter model, which reaches real-time performance for 512p resolution at 16 FPS on one GPU. We evaluate our method through both quantitative metrics and human evaluation. Our model enables real-time applications, e.g. *streaming generation, interactive generation, and video-to-video editing. More video results are available at the project website: https://cumulo-autumn.github.io/StreamDiT/.*

1. Introduction

Video generation is a trending problem in computer vision, with rapid progress in recent years. Diffusion models have been adopted for text-to-video (T2V) generation, and demonstrated capabilities of generating high-quality videos, with a potential as world simulators. The success was built upon the scalability of transformer-based architectures and huge compute resources to train billions of parameters. Despite achievements, many existing models are designed to generate short video clips, due to the increasing cost after scaling. As a result, generating long videos with low latency remains an extreme challenge [18], which is required in interactive and real-time applications.

Modern T2V models [17, 24, 29] are based on Diffusion Transformers (DiTs) [28] with bidirectional attention. Increasing video length is expensive due to the quadratic complexity of transformers with respect to sequence length. To make diffusion models capable of generating long videos, some approaches [13, 38, 44] attempted to combine autoregressive (AR) and diffusion models, by adding progressive noise or causal masking. In an AR setup, a generation frame can only see previous context frames in attention, posing a limitation on its quality. As shown in [19], replacing causal attention with bidirectional attention leads

* denotes equal contributions
This work was done while the first author was an intern at Meta.

to a massive quality gain. Modern video generation models [17, 29, 47] are also based on bidirectional attention. On another axis, AR frame-by-frame prediction has low streaming latency. However, it still needs to wait for a full denoising process to obtain a clean output frame. Diffusion with bidirectional attention can have high throughput, but its latency is much lower than AR. These observations lead us to seek a solution with low latency, high throughput, and high quality for streaming video generation.

We are inspired by non-trainable streaming generation *e.g.* StreamDiffusion [16] and FIFO-Diffusion [15]. StreamDiffusion utilizes pre-trained image diffusion models to edit an input video stream via batch denoising. It suffers from a lack of frame-to-frame consistency, leading to visual artifacts when generating continuous video streams. FIFO-Diffusion enqueues a sequence of frames with different noise levels. It pops up a clear frame after one denoising step and enqueues a noise frame in the queue. However, they do not come with any training solution, which limits their quality. Another missing effort is sampling distillation, which is crucial for real-time applications. Since the queue is updated at every denoising step, popular distillation methods *e.g.* step distillation [27] and consistency distillation [23] cannot be applied directly.

To address the challenges, we propose a complete solution for streaming video generation including training, modeling, and distillation, named StreamDiT. For training, we modify the flow matching [20] by introducing a moving buffer of frames. Within a buffer, we adopt full attention on latent frames so that our method can be seamlessly applied to modern T2V models. At inference, the buffer is moving along the frame dimension, generating clean frames sequentially. Similar to previous work [15, 38], we allow buffered frames with different noise levels. We design a unified partitioning of latent frames with different chunk sizes and micro steps. The uniform noise [20] and the diagonal noise [15, 38] are special cases of our partitioning. With mixed training of different schemes, we can enhance video consistency and avoid overfitting on a specific scheme.

For modeling, we design our StreamDiT model architecture to fit its training process while being efficient for real-time applications. We modify adaLN DiT [28] by adding varying time embedding and replacing full attention with window attention [21]. We first train our StreamDiT as a basic T2V model, and then adapt it to streaming video generation. To achieve real-time applications, we also design a multistep distillation tailored for StreamDiT with a chosen partitioning scheme. With that, we distill our model to 8 sampling steps without classifier-free guidance (CFG). Finally, our model can generate video frames at 16 FPS on one H100 GPU, achieving real-time performance. Fig. 1 shows some video results of our real-time streaming generation.

Our contributions are summarized as follows:

- We propose StreamDiT training for streaming video generation, which is based on flow matching by introducing a moving buffer. We articulate a generalized partitioning of buffered frames that covers uniform noise, diagonal noise, and others in between. We also point out a requirement for generation models such that time embedding needs to be separable in the frame dimension.
- We design StreamDiT model as adaLN DiT with varying time embedding and window attention. Our StreamDiT has 4B parameters and is trained to generate videos at 512p resolution. We show that using mixed training with different partitioning schemes can improve the quality and flexibility of streaming video generation.
- We build a real-time solution using multistep distillation tailored for StreamDiT. Choosing a partitioning scheme that balances inference speed and quality, we distill micro steps to one for each segment. With all the efforts combined, our distilled model achieves real-time performance with 16 FPS on one GPU. In addition, we show some results to highlight potential real-time applications using our model.

2. Related Work

2.1. Text-to-Video Generation

Arising with the development of text-to-image generation (T2I) [5], video generation bursts by introducing temporal modules. Earlier work [2, 3, 9, 10, 48] was based on UNet structures with temporal layers added to T2I models. As a pivot, DiT [28] demonstrated that transformer architecture is more scalable than UNet architecture for diffusion models. VDT [22] proposes to use separated spatial and temporal attentions in a transformer block. GenTron [4] introduces a family of transformer-based diffusion models for image and video generation. Snap Video [25] proposes a transformer architecture with joint spatiotemporal blocks, which is trained faster than U-Nets. ViD-GPT [7] uses causal attention in temporal domain and frames as prompts, following the design of GPT in LLMs. Due to the limitations of compute, these models are trained at small or medium scales, up to a few of billion parameters, generating videos of a few seconds. In the latest work of MovieGen [29], the DiT model has been scaled to 30B parameters and can generate videos up to 16 seconds, with high motion and aesthetic qualities.

In parallel, the open-source community is also growing fast. OpenSora [47] released a small T2V model with about 1B parameters, using factorized transformers for spatial and temporal domains. Hunyuan [17] video model reaches 13B parameters, which uses MMDiT [6] with concatenated text and video tokens in self-attention. For auto-encoder, it adopts causal 3D convolution [45] in its VAE. Step-Video-T2V [24] with 30B parameters is a large open-source model for video generation, which reduces the model size gap between open-source and closed-source.

2.2. Extensible Video Generation

The literature of extensible long video generation can be roughly categorized as training and training-free methods. Training new models or new modules injected to existing models requires a large amount of compute resources. NUWA-XL [42] proposes a diffusion over diffusion architecture to generate long videos in a coarse-to-fine manner. StreamingT2V [12] proposes an autoregressive approach for long video generation, using a short-term memory block and long-term memory block of previous chunks. In [38], a similar method was proposed by assigning latent frames with progressively increasing noise levels. VideoTetris [35] trains a ControlNet branch of condition frames for long video generation. A recent work by Loong [37] proposes an autoregressive LLM-based approach for video generation. The model is trained on 10-second videos and can be extended to one-minute long. To reduce the training and inference gap of AR video generation, self-forcing [13] was proposed, which uses student predictions as previous frames in the context. To reduce the streaming latency, it generates one latent frame at a time. The current AR models are hard to compete with full-attention diffusion models on quality.

For training-free methods, Gen-L-Video [36] discovers that the denoising path of a long video can be approximated by joint denoising of overlapping short videos in the temporal domain. MimicMotion [46] extends MultiDiffusion [1] to the temporal domain. It blends frames within overlap region progressively along the denoising process. Video-Infinity [33] distributes a long-form video task to multiple GPUs, with dual-scope attention that modulates temporal self-attention to balance local and global contexts efficiently across the devices. FreeNoise [30] proposes a simple solution to generate consistent long videos by rescheduling a sequence of noises for long-range correlation and performing temporal attention over them by window-based function. Reuse-and-Diffuse [8] proposes an iterative denoising for T2V, with staged guidance. It reuses intermediate denoising results from the previous clip to condition the current clip. FIFO-Diffusion [15] proposes an inference technique that iteratively performs diagonal denoising, which concurrently processes a series of consecutive frames with increasing noise levels in a queue.

2.3. Sampling Efficiency

Step distillation has been widely adopted for accelerating diffusion models. Compared to advanced numerical solvers, sampling distillation demonstrates better quality with fewer number of sampling steps. Progressive distillation [31] distills two denoising steps of a teacher model to one step of a student model. Guided distillation [27] proposed to first distill two function evaluations of CFG to one, and then perform progressive distillation on sampling steps. Consistency distillation [23, 32] maps any points on a diffusion trajectory to the same origin, and therefore reduces the number of steps needed for sampling. Multistep consistency models [11] splits the diffusion process to multiple segments, and consistency distillation can be applied at each segment. By allowing more budget on sampling steps, Multistep consistency distillation can achieve higher quality.

To further improve sampling efficiency, one-step approaches have been proposed, including UFOGen [40] and DMD [43]. The idea is to directly match distributions without the Gaussian assumption. They require to host auxiliary models (discriminator or fake score networks) during training, posing challenges for applying to large video models.

3. StreamDiT Training

3.1. Buffered Flow Matching

Flow Matching: Our training is based on Flow Matching (FM) [20]. FM produces a sample from the target data distribution by progressively transforming a sample from an initial prior distribution, such as a Gaussian. For a data sample $\mathbf{X}_1$, FM samples a time step $t \in [0, 1]$, and noise $\mathbf{X}_0 \sim \mathcal{N}(0, 1)$. These are used to create a training sample $\mathbf{X}_t$. FM predicts the velocity $\mathbf{V}_t$ that moves the sample $\mathbf{X}_t$ in the direction of data sample $\mathbf{X}_1$.

A simple linear interpolation or the optimal transport (OT) path [20] is used to construct $\mathbf{x}_t$, *i.e.*

$$\mathbf{X}_t = t\,\mathbf{X}_1 + (1 - (1 - \sigma_{\min})t)\,\mathbf{X}_0, \tag{1}$$

where $\sigma_{\min}$ is the standard deviation of x at $t = 1$. Thus, the ground truth velocity can be derived as

$$\mathbf{V}_t = \frac{d\mathbf{X}_t}{dt} = \mathbf{X}_1 - (1 - \sigma_{\min})\mathbf{X}_0. \tag{2}$$

It is worth noting that this target is irrelevant to timestep t. With parameters Θ and text prompt $\mathbf{P}$, the predicted velocity is written as $u(\mathbf{X}_t, \mathbf{P}, t; \Theta)$, and the training objective is represented as

$$\mathbb{E}_{t,\mathbf{X}_t} \| u(\mathbf{X}_t, \mathbf{P}, t; \Theta) - \mathbf{V}_t \|^2. \tag{3}$$

At inference, an FM model predicts the velocity to a clean sample on every denoising step. With the Euler solver, the inference can be formulated as

$$\mathbf{X}_{t+\Delta t} = \mathbf{X}_t + u(\mathbf{X}_t, \mathbf{P}, t; \Theta)\Delta t, \tag{4}$$

where Δt is the step size.

Buffered Flow Matching: We consider streaming video generation as a sequence of (possibly latent) frames

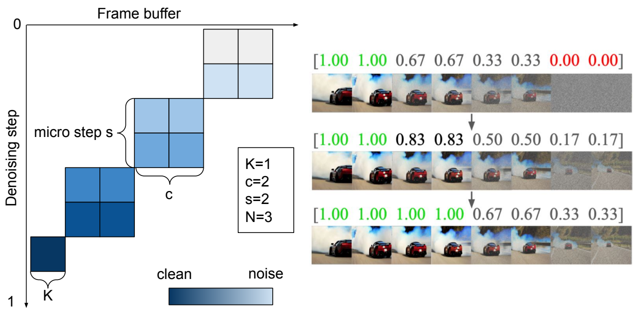


Figure 2. Illustration of StreamDiT partitioning. We partition the buffer to K reference frames and N chunks. Each chunk has c frames and s micro denoising steps.

$[f_1, f_2, \ldots, f_N]$, and N can be infinite. For a video diffusion model with frame buffer B, the clean data sample starting with frame f_i is denoted as $\mathbf{X}_1^i = [f_{i+1}, \ldots, f_{i+B}]$. We allow different noise levels to the frames: $\tau = [\tau_1, \ldots, \tau_B]$ as a monotonically increasing sequence. Thus a training example can be constructed as

$$\mathbf{X}_\tau^i = \tau \circ \mathbf{X}_1^i + (1 - (1 - \sigma_{\min})\tau) \circ \mathbf{X}_0, \tag{5}$$

where $\circ$ denotes element-wise product. Please note that the noise sample X_0 remains the same.

At inference, the buffer is updated by model predicted flow

$$\mathbf{X}_{\tau+\Delta\tau}^i = \mathbf{X}_\tau^i + u(\mathbf{X}_\tau^i, \mathbf{P}, \tau; \Theta) \circ \Delta\tau, \tag{6}$$

where $\Delta\tau$ is a sequence of step sizes. If one or more frames achieve the final denoising step, they are popped out of the buffer, and new noise frames are pushed at the beginning of the buffer. Therefore, it can generate streaming video sequences.

3.2. Partitioning Scheme

To facilitate flexible training and inference of StreamDiT, we design a unified partitioning of buffered frames. As illustrated in Fig. 2, the buffer is partitioned to K reference (context) frames and N chunks. Each chunk has c frames and s micro denoising steps.

Context Reference Frames: To enhance temporal consistency, we can optionally cache the last K fully denoised frames at the beginning of the buffer. We refer to these clean frames as reference frames. They participate in the denoising step as input but are no longer denoised. The reference frames are updated in the same way as other frames when the buffer moves. Allowing optional reference frames matches the design of FIFO-Diffusion [15], which can be viewed as a special case. Since our method is trainable, we found that reference frames can be skipped for streaming video generation. Hence, we set $K = 0$ in the rest of our work.

Method	Scheme	Consistency	Streaming
Uniform	$c = B, s = 1$	High	No
Diagonal	$c = 1, s = 1$	Low	Yes
StreamDiT	$c = [1, \ldots, B], s = \frac{T}{N}$	High	Yes

Table 1. StreamDiT unifies different partitioning schemes.

Chunked Denoising: Instead of denoising frame by frame, we group frames into stream chunks, with each chunk containing a specified number of frames indicated by chunk size. Noise levels are now applied at the chunk level, and each time a chunk of frames exits the pipeline. Let N denote the number of stream chunks and c the number of frames in each chunk. Then the total number of frames processed at any time is $K + N \times c$, and the number of denoising steps is constrained to N.

Micro Step: For better performance, additional denoising steps are usually helpful. A straightforward approach to address this limitation is to increase the size of the buffer, which will increase the serving cost, and therefore needs to be bounded by a computation budget.

To overcome this, we introduce an additional dimension of the design called micro-denoising step, illustrated in Fig. 2. The core idea of micro step is to denoise stream chunks along the temporal axis while stagnating at a fixed spatial position for a certain amount of time. Let s denote the micro step; then each stream chunk undergoes s denoising steps before advancing to the next noise level and moving toward the output. This modification effectively extends the total number of denoising steps to $s \times N$ without increasing the buffer size.

With the incorporation of reference frames, chunked frames, and micro-step denoising, the following equations hold:

$$\begin{aligned} B &= K + N \times c \\ T &= s \times N \end{aligned} \tag{7}$$

where B is the total length of the frame fed into the model, N is the number of stream chunks, c is the number of frames in each stream chunk, and T represents the total number of inference denoising steps.

Mixed Training: As shown in Tab. 1, StreamDiT unifies partion schemes from bidirectional attention with uniform noise to progressive diagonal noise [15, 38]. The latter enables streaming generation, but hurts consistency of generated content. To enhance consistency and avoid overfitting, we adopt mixed training with different schemes. This drives the model to learn generalized denoising with different noise levels, instead of memorizing fixed noise levels. It is worth noting that our mixed training covers diffusion and FM training. Therefore, our model can work as a standard T2V generation without streaming. This is also used

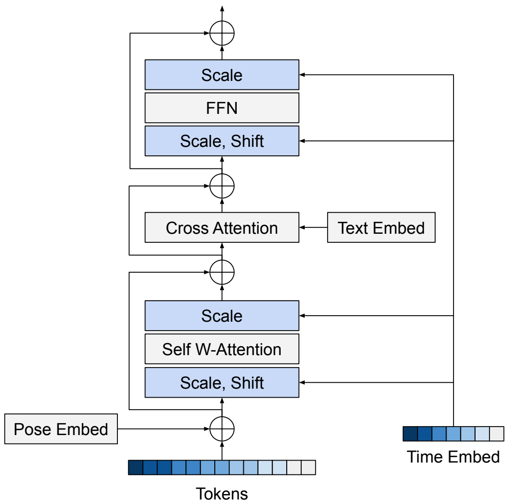


Figure 3. StreamDiT with varying time embedding and window attention. We modified the standard adaLN DiT with varying time embeddings applied to scale and shift modulations.

to initialize our streaming generation.

According to Fig. 2, frames in each chunk correspond to a distinct segment of the overall time step range. Thus, during training, we sample a random time step for the i-th chunk as follows:

$$\tau_i \sim \text{Uniform}\left(\left[\frac{T}{N} \cdot (i-1), \frac{T}{N} \cdot i\right]\right) \tag{8}$$

Interestingly, the StreamDiT training can be viewed as parallel training of the full range of denoising.

4. StreamDiT Modeling

4.1. Model Architecture

Time-Varying DiT: As shown in Sec. 3.1, our StreamDiT changes the scalar condition t to a sequence τ in the model. This requires that the time condition should be separable in the frame dimension. We follow the standard adaLN DiT [28] architecture, where time embedding is used for scale and shift modulations, and modify it with varying time embedding. Specifically, we reshape the latent tensor to 3D: $[F, H, W]$, and apply time embedding along the first dimension. As shown in Fig. 3, we modify the adaLN DiT with varying time embedding. Input tokens are also corrupted with different levels of noise. This architecture matches the design of our StreamDiT.

Window Attention: To make the DiT model more efficient, we adopt window attention [21] for all self-attention layers. Specifically, we partition a 3D latent with shape $[F, H, W]$ to non-overlapped windows with size $[F_w, H_w, W_w]$, and apply masking when computing self-attention, such that a token can only see tokens within the same window. To ensure consistency across windows, we shift by half of the window size every other layer. As illustrated in Fig. 4, voxels at axis ends will be warped over to the beginnings in the shifting. Accumulatively, global token communication can broadcast to all tokens, while each computation is efficient with local attention. The complexity of window attention is only $\frac{F_w \times H_w \times W_w}{F \times H \times W}$ of full attention.

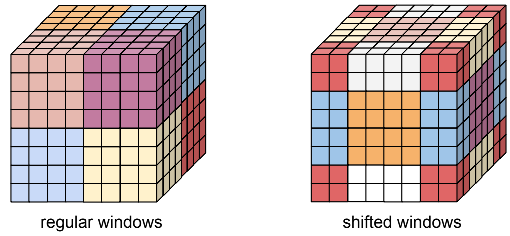


Figure 4. Illustration of window partitioning: regular windows and shifted windows.

Other Components: Since we are targeting real-time applications, we chose a moderate size (4B) for our DiT. We reuse a temporal auto-encoder (TAE) in Movie Gen [29] with compression rate 4 in temporal domain and 8 in spatial domain. The latent channel size is 8, with the consideration that a moderate latent space is easier to learn for small generation models. The trade-off is that with a smaller temporal compression rate, the model needs to generate more latent frames to reach the same FPS.

For text encoders, we adopt the same ones from Movie Gen [29], including UL2 [34], ByT5 [41], and MetaCLIP [39]. The text encoders run fast on GPU. We only run text encoders when the prompt is changed for streaming video generation. Therefore, its computation time is negligible.

4.2. Multistep Distillation

Sampling distillation is a key component to build real-time streaming video generation. Due to the modifications of frame partitioning and micro denoising steps made by StreamDiT, standard sampling distillation methods [11, 27, 31] cannot be applied. Thus, a customized sampling distillation for StreamDiT is needed.

As described in Sec. 3.2, StreamDiT partitions frames in a buffer to N chunks, each of which has s denoising steps before moving the buffer. In particular, we aim at reducing the micro-step s in Eq. (7). Our StreamDiT model is trained with mixed partitioning schemes, to make it flexible to different scenarios. In distillation, we first choose a partitioning scheme. Specifically, we set $K = 0$, $c = 2$, $s = 16$, $N = 8$ for the teacher model. This indicates that the teacher model has $s * N = 128$ denoising steps with CFG. The FM trajectory is split to N segments. We then perform step distillation in each segment separately. In practice, we perform both step and guidance distillation at the same time,

	Subject Consistency	Background Consistency	Temporal Flickering	Motion Smoothness	Dynamic Degree	Aesthetic Quality	Quality Score
ReuseDiffuse	0.9501	0.9615	0.9838	0.9912	0.2900	0.5993	0.8019
FIFO	0.9412	0.9576	0.9796	0.9889	0.3094	0.6088	0.7981
Ours	0.9622	0.9625	0.9671	0.9861	0.5240	0.6026	**0.8185**
Ours-distill	0.9491	0.9555	0.9649	0.9831	0.7040	0.5940	0.8163

Table 2. VBench quality metrics of our evaluation. Our models outperform others, and our distilled model is close to our teacher model.

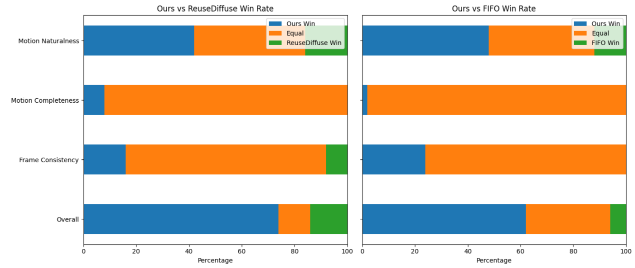


Figure 5. Human evaluations of our method compared with others, where our model shows higher win rates across all axes.

Chunk size	[1]	[1,2]	[1,2,4]	[1,2,4,8]	[1,2,4,8,16]
Quality score	0.8129	0.8100	0.8080	0.8076	**0.8144**

Table 3. Ablation of mixed training. Chunk size 1 represents the Progressive AR Diffusion [38], and chunk size 16 represents the original T2V without streaming. A mixed training with all chunk sizes show the best quality score.

by distilling multiple CFG steps of the teacher into a single conditional forward pass of the student.

For the multistep distillation, we reduce the micro step s to 1, resulting in N-step sampling without CFG, which still follows the design of StreamDiT for streaming. Moreover, it powers up a significant speed-up that enables real-time video generation.

5. Experiments

5.1. Implementation Details

We finetune a T2V model [29] with 4B parameters using the architecture introduced in Sec. 4.1. The latent size is $[16, 64, 64]$. With a $[4\times, 8\times, 8\times]$ TAE, the base model generates 64 frames at 512p resolution.

Then we adapt the model for streaming video generation. Our StreamDiT training has three stages: task learning, task generalization, and quality fine-tuning. In the first stage, we use a small amount of high-quality video data (3K videos) with a large learning rate ($1e-4$) to adapt the original T2V into a video streaming model. The second stage involves further training on the pretraining dataset (2.6M videos) with a small learning rate ($1e-5$) to improve generalization for video streaming. In the final stage, we finetune the model on the high-quality video dataset with a small learning rate ($1e-5$) to optimize output quality. Each stage is trained with 10K iterations on 128 NVIDIA H100 GPUs.

To achieve real-time inference, we additionally apply the multistep distillation described in Sec. 4.2. We found a partitioning scheme $c = 2, s = 16, N = 8$ is good for our multistep distillation considering both quality and efficiency, so we used it for teacher model inference. The distillation is also conducted on the 3K high-quality video dataset and 64 NVIDIA H100 GPUs for 10K iterations. After distillation, the micro step is reduced to 1, and the total number of sampling steps is 8. For more training details, please refer to the Appendix.

5.2. Evaluation

We compare our method with ReuseDiffuse [8] and FIFO-Diffusion [15] for streaming generation of long videos. To make a fair comparison, we implemented their methods on our base model, so they share the same visual quality. We select 50 prompts from the evaluation dataset of Movie Gen [29] that are suitable for long videos. We adopt VBench [14] quality metrics for quantitative evaluation.

As shown in Tab. 2, our models outperform others with high quality scores. All methods have similar aesthetic quality and imaging quality scores. This is because they are derived from the same base model. ReuseDiffuse and FIFO achieve higher temporal consistency and motion smoothness. However, by examining their generated videos, the content is more static. This is also reflected in the dynamic degree column, where our models are much better.

In addition to VBench metrics, we also conduct human evaluations following the guidance in [29]. A human evaluation compares two model results on the same prompts side-by-side. Annotators were asked to evaluate pairs of video samples to determine which one was superior or if they were of comparable quality along several axes. The evaluation criteria encompassed four aspects: overall quality, frame consistency, motion completeness, and motion naturalness. For these evaluations, we employ the same set of 50 prompts used in the VBench evaluation, generating 8-second videos at a resolution of 512p for each prompt. As shown in Fig. 5, our proposed method surpasses existing approaches across all evaluation metrics.

We show some selected frames from generated videos in Fig. 6. The videos are one-minute long. We can see that our models have more consistent content with more motions, while others are static. This observation aligns with the VBench metrics.

5.3. Ablation Study

We conduct an ablation study to analyze the mixed training discussed in Sec. 3.2. Specifically, we train models with different mixing schemes as shown in Tab. 3. Chunk size 1

Figure 6. Visual results selected from our evaluation. Our models show better consistency and higher quality than others. Our distilled model has a similar quality with our teacher model.

represents the Progressive AR Diffusion [38] implemented on our model. Recall that our base model has 16 latent frames, so chunk size 16 is the basic T2V. After the models are trained, we generated videos using chunk size $c = 1$, since this is the case covered by all models. Among them, we see that the mixture of all chunk sizes achieves the best quality score, although the inference is biased to chunk size 1 as being set in inference. Please note that we use a different inference scheme in Tab. 2, so the numbers of our model are different. The first one with chunk size 1 and no mixing achieves the second-best quality score, indicating an overfitting to this case.

5.4. Applications

Real-Time Streaming: Our distilled model has a chunk size 2 and 8 sampling steps without CFG. We benchmark its performance on one H100 GPU. It takes 482 ms for one denoising step to generate 2 latent frames and 8 video frames after a $4\times$ TAE. The latency of text encodering and TAE decoding are negligible. Thus, our distilled model can reach real-time performance at **16 FPS**. Fig. 7 shows frames from a one-minute long video generated in real-time. With real-time streaming generation, our model can potentially work as a game engine.

Infinite Streaming: To explore the ability of infinite streaming, we try our distilled model to generate a video over 5 minutes long. As shown in Fig. 8, the video content and quality are consistent after a very long generation, demonstrating the potential for infinite streaming.

Interactive Streaming: Additionally, we showcase the interactive storytelling capabilities of our model using a sequence of semantically related prompts. The qualitative results in Fig. 9 demonstrate that our model effectively controls video events interactively, based on user-specified prompts.

Video-to-Video Streaming: Our model can also be applied to real-time video-to-video tasks. The video-to-video process is based on the noise addition and denoising strategy proposed in SDEdit [26]. First, we add noise to the original input video and then denoise the noisy video with new prompt guidance. We achieve significant content editing while maintaining good temporal consistency. The visual results shown in Fig. 10 highlight the model's powerful video editing capabilities. In this example, a pig in the input video is modified to a cat in the output video, while the background is preserved.

6. Conclusion and Limitations

In this work, we present StreamDiT for streaming video generation. It includes a novel training framework and efficient modeling that can run in real-time. StreamDiT training is based on flow matching with a moving buffer. We de-


A rally car accelerating through a muddy forest track.

Figure 7. Real-time video streaming. With real-time streaming generation, our model can potentially work as a game engine.

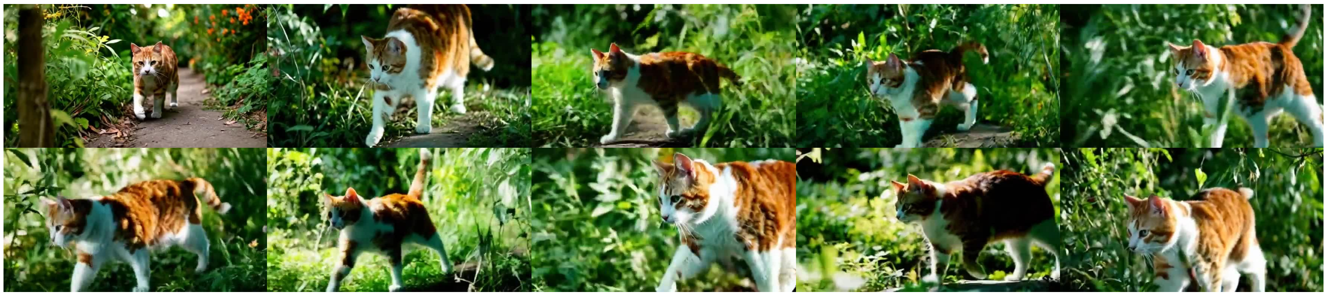
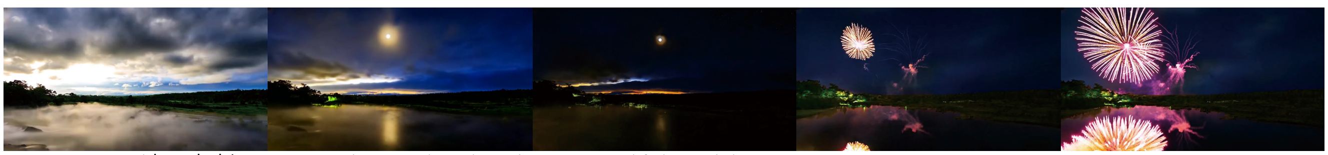

White and orange tabby cat runs through a dense garden.

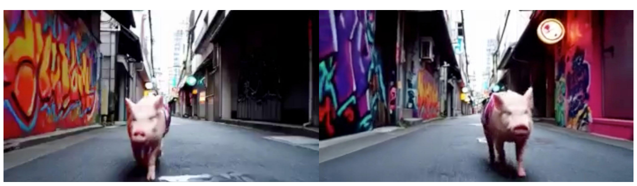
Figure 8. Infinite streaming. We generate a 5-minute video to demonstrate the potential for infinite streaming.


Serene nature with a calm lake and cloudy sky in daylight

Quiet lake at night under a glowing moon and fading twilight

Fireworks exploding over a lake

Figure 9. Interactive video streaming. The user can enter prompts to navigate the video generation on the fly.


Input video

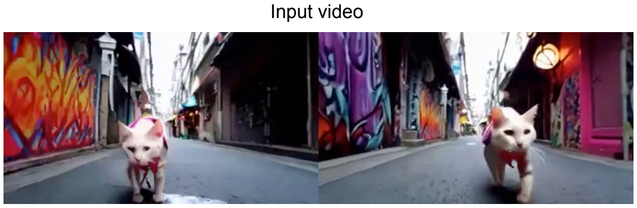

Output video: A cat walking on a street

Figure 10. Video-to-video streaming. The top row is input and the bottom row is output, where a pig is modified to a cat by prompt.

sign generalized partitioning that unifies different schemes. Through experiments, we show that mixed training with different schemes can improve the consistency and quality of generations. To achieve real-time performance, we train an efficient StreamDiT model based on time-varying DiT with window attention. We further distill the model to 8 sampling steps, using the proposed multistep distillation tailored for StreamDiT. Our model achieves high temporal consistency across frames, addressing critical challenges in long-form and interactive video applications.

At the end, we discuss two limitations of our StreamDiT-4B, regarding model capacity and context length. StreamDiT-4B has a moderate model size, which limits the quality of basic T2V generation. As a result, we noticed artifacts in some of the generated videos.To verify the capacity limitation and the scalability of our method, we also applied StreamDiT to a 30B-parameter model with a light-weight training process. We found that StreamDiT-30B shows much better quality in terms of visual aesthetic, content, and low artifacts. Please refer to the Appendix and project website for the results. Another limitation of StreamDiT-4B is the short context length. Flaws may be observed whereby out-of-context objects exhibit altered visual appearance after they re-enter the video sequence. We anticipate that this limitation can be mitigated by employing an extended context window in conjunction with a key–value (KV) cache during inference, thereby enabling more efficient computational acceleration. Therefore, the two limitations are not fundamental to the StreamDiT method, and can be improved with future work.

7. Acknowledgment

We would like to express our sincere gratitude to Yen-Cheng Liu, Luxin Zhang, and Tao Xu for their valuable discussions and insightful contributions, which greatly enriched our work.

References

[1] Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. Multidiffusion: Fusing diffusion paths for controlled image generation. In *ICML*, 2023. 3

[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 *CVPR*, 2023. 2

[3] 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. 2

[4] Shoufa Chen, Mengmeng Xu, Jiawei Ren, Yuren Cong, Sen He, Yanping Xie, Animesh Sinha, Ping Luo, Tao Xiang, and Juan-Manuel Perez-Rua. Gentron: Delving deep into diffusion transformers for image and video generation. In *CVPR*, 2024. 2

[5] Xiaoliang Dai, Ji Hou, Chih-Yao Ma, Sam Tsai, Jialiang Wang, Rui Wang, Peizhao Zhang, Simon Vandenhende, Xiaofang Wang, Abhimanyu Dubey, et al. Emu: Enhancing image generation models using photogenic needles in a haystack. *arXiv preprint arXiv:2309.15807*, 2023. 2

[6] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, Dustin Podell, Tim Dockhorn, Zion English, Kyle Lacey, Alex Goodwin, Yannik Marek, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis, 2024. 2

[7] Kaifeng Gao, Jiaxin Shi, Hanwang Zhang, Chunping Wang, and Jun Xiao. Vid-gpt: Introducing gpt-style autoregressive generation in video diffusion models, 2024. 2

[8] 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, 2023. 3, 6

[9] Yuwei Guo, Ceyuan Yang, Anyi Rao, Zhengyang Liang, Yaohui Wang, Yu Qiao, Maneesh Agrawala, Dahua Lin, and Bo Dai. Animatediff: Animate your personalized text-to-image diffusion models without specific tuning. In *ICLR*, 2024. 2

[10] Yingqing He, Tianyu Yang, Yong Zhang, Ying Shan, and Qifeng Chen. Latent video diffusion models for high-fidelity long video generation, 2023. 2

[11] Jonathan Heek, Emiel Hoogeboom, and Tim Salimans. Multistep consistency models, 2024. 3, 5

[12] Roberto Henschel, Levon Khachatryan, Daniil Hayrapetyan, Hayk Poghosyan, Vahram Tadevosyan, Zhangyang Wang, Shant Navasardyan, and Humphrey Shi. Streamingt2v: Consistent, dynamic, and extendable long video generation from text, 2024. 3

[13] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *NeurIPS*, 2025. 1, 3

[14] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, Yaohui Wang, Xinyuan Chen, Limin Wang, Dahua Lin, Yu Qiao, and Ziwei Liu. Vbench: Comprehensive benchmark suite for video generative models. In *CVPR*, 2024. 6

[15] Jihwan Kim, Junoh Kang, Jinyoung Choi, and Bohyung Han. Fifo-diffusion: Generating infinite videos from text without training. In *NeurIPS*, 2024. 2, 3, 4, 6

[16] Akio Kodaira, Chenfeng Xu, Toshiki Hazama, Takanori Yoshimoto, Kohei Ohno, Shogo Mitsuhori, Soichi Sugano, Hanying Cho, Zhijian Liu, and Kurt Keutzer. Streamdiffusion: A pipeline-level solution for real-time interactive generation, 2023. 2

[17] Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, Kathrina Wu, Qin Lin, Junkun Yuan, Yanxin Long, Aladdin Wang, Andong Wang, Changlin Li, Duojun Huang, Fang Yang, Hao Tan, Hongmei Wang, Jacob Song, Jiawang Bai, Jianbing Wu, Jinbao Xue, Joey Wang, Kai Wang, Mengyang Liu, Pengyu Li, Shuai Li, Weiyan Wang, Wenqing Yu, Xinchi Deng, Yang Li, Yi Chen, Yutao Cui, Yuanbo Peng, Zhentao Yu, Zhiyu He, Zhiyong Xu, Zixiang Zhou, Zunnan Xu, Yangyu Tao, Qinglin Lu, Songtao Liu, Dax Zhou, Hongfa Wang, Yong Yang, Di Wang, Yuhong Liu, Jie Jiang, and Caesar Zhong. Hunyuanvideo: A systematic framework for large video generative models, 2025. 1, 2

[18] Chengxuan Li, Di Huang, Zeyu Lu, Yang Xiao, Qingqi Pei, and Lei Bai. A survey on long video generation: Challenges, methods, and prospects, 2024. 1

[19] Tianhong Li, Yonglong Tian, He Li, Mingyang Deng, and Kaiming He. Autoregressive image generation without vector quantization, 2024. 1

[20] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *ICLR*, 2023. 2, 3

[21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2, 5

[22] Haoyu Lu, Guoxing Yang, Nanyi Fei, Yuqi Huo, Zhiwu Lu, Ping Luo, and Mingyu Ding. Vdt: General-purpose video diffusion transformers via mask modeling. In *CVPR*, 2024. 2

[23] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference, 2023. 2, 3

[24] Guoqing Ma, Haoyang Huang, Kun Yan, Liangyu Chen, Nan Duan, Shengming Yin, Changyi Wan, Ranchen Ming, Xiaoniu Song, Xing Chen, Yu Zhou, Deshan Sun, Deyu Zhou, Jian Zhou, Kaijun Tan, Kang An, Mei Chen, Wei Ji, Qiling Wu, Wen Sun, Xin Han, Yanan Wei, Zheng Ge, Aojie Li, Bin Wang, Bizhu Huang, Bo Wang, Brian Li, Changxing Miao, Chen Xu, Chenfei Wu, Chenguang Yu, Dapeng Shi,

Dingyuan Hu, Enle Liu, Gang Yu, Ge Yang, Guanzhe Huang, Gulin Yan, Haiyang Feng, Hao Nie, Haonan Jia, Hanpeng Hu, Hanqi Chen, Haolong Yan, Heng Wang, Hongcheng Guo, Huilin Xiong, Huixin Xiong, Jiahao Gong, Jianchang Wu, Jiaoren Wu, Jie Wu, Jie Yang, Jiashuai Liu, Jiashuo Li, Jingyang Zhang, Junjing Guo, Junzhe Lin, Kaixiang Li, Lei Liu, Lei Xia, Liang Zhao, Liguo Tan, Liwen Huang, Liying Shi, Ming Li, Mingliang Li, Muhua Cheng, Na Wang, Qiaohui Chen, Qinglin He, Qiuyan Liang, Quan Sun, Ran Sun, Rui Wang, Shaoliang Pang, Shiliang Yang, Sitong Liu, Siqi Liu, Shuli Gao, Tiancheng Cao, Tianyu Wang, Weipeng Ming, Wenqing He, Xu Zhao, Xuelin Zhang, Xianfang Zeng, Xiaojia Liu, Xuan Yang, Yaqi Dai, Yanbo Yu, Yang Li, Yineng Deng, Yingming Wang, Yilei Wang, Yuanwei Lu, Yu Chen, Yu Luo, Yuchu Luo, Yuhe Yin, Yuheng Feng, Yuxiang Yang, Zecheng Tang, Zekai Zhang, Zidong Yang, Binxing Jiao, Jiansheng Chen, Jing Li, Shuchang Zhou, Xiangyu Zhang, Xinhao Zhang, Yibo Zhu, Heung-Yeung Shum, and Daxin Jiang. Step-video-t2v technical report: The practice, challenges, and future of video foundation model, 2025. 1, 2

[25] Willi Menapace, Aliaksandr Siarohin, Ivan Skorokhodov, Ekaterina Deyneka, Tsai-Shien Chen, Anil Kag, Yuwei Fang, Aleksei Stoliar, Elisa Ricci, Jian Ren, and Sergey Tulyakov. Snap video: Scaled spatiotemporal transformers for text-to-video synthesis. In *CVPR*, 2024. 2

[26] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided Image Synthesis and Editing with Stochastic Differential Equations. In *ICLR*, 2022. 7

[27] Chenlin Meng, Robin Rombach, Ruiqi Gao, Diederik P. Kingma, Stefano Ermon, Jonathan Ho, and Tim Salimans. On distillation of guided diffusion models. In *CVPR*, 2023. 2, 3, 5

[28] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *ICCV*, 2023. 1, 2, 5

[29] Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, David Yan, Dhruv Choudhary, Dingkang Wang, Geet Sethi, Guan Pang, Haoyu Ma, Ishan Misra, Ji Hou, Jialiang Wang, Kiran Jagadeesh, Kunpeng Li, Luxin Zhang, Mannat Singh, Mary Williamson, Matt Le, Matthew Yu, Mitesh Kumar Singh, Peizhao Zhang, Peter Vajda, Quentin Duval, Rohit Girdhar, Roshan Sumbaly, Sai Saketh Rambhatla, Sam Tsai, Samaneh Azadi, Samyak Datta, Sanyuan Chen, Sean Bell, Sharadh Ramaswamy, Shelly Sheynin, Siddharth Bhattacharya, Simran Motwani, Tao Xu, Tianhe Li, Tingbo Hou, Wei-Ning Hsu, Xi Yin, Xiaoliang Dai, Yaniv Taigman, Yaqiao Luo, Yen-Cheng Liu, Yi-Chiao Wu, Yue Zhao, Yuval Kirstain, Zecheng He, Zijian He, Albert Pumarola, Ali Thabet, Artsiom Sanakoyeu, Arun Mallya, Baishan Guo, Boris Araya, Breena Kerr, Carleigh Wood, Ce Liu, Cen Peng, Dimitry Vengertsev, Edgar Schonfeld, Elliot Blanchard, Felix Juefei-Xu, Fraylie Nord, Jeff Liang, John Hoffman, Jonas Kohler, Kaolin Fire, Karthik Sivakumar, Lawrence Chen, Licheng Yu, Luya Gao, Markos Georgopoulos, Rashel Moritz, Sara K. Sampson, Shikai Li, Simone Parmeggiani, Steve Fine, Tara Fowler, Vladan Petrovic, and Yuming Du. Movie gen: A cast of media foundation models, 2024. 1, 2, 5, 6

[30] Haonan Qiu, Menghan Xia, Yong Zhang, Yingqing He, Xintao Wang, Ying Shan, and Ziwei Liu. Freenoise: Tuning-free longer video diffusion via noise rescheduling. In *ICLR*, 2024. 3

[31] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022. 3, 5

[32] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency models. In *ICML*, 2023. 3

[33] Zhenxiong Tan, Xingyi Yang, Songhua Liu, and Xinchao Wang. Video-infinity: Distributed long video generation. *arXiv preprint arXiv:2406.16260*, 2024. 3

[34] Yi Tay, Mostafa Dehghani, Vinh Q. Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, Huaixiu Steven Zheng, Denny Zhou, Neil Houlsby, and Donald Metzler. Ul2: Unifying language learning paradigms. In *ICLR*, 2023. 5

[35] Ye Tian, Ling Yang, Haotian Yang, Yuan Gao, Yufan Deng, Jingmin Chen, Xintao Wang, Zhaochen Yu, Xin Tao, Pengfei Wan, Di Zhang, and Bin Cui. Videotetris: Towards compositional text-to-video generation. In *NeurIPS*, 2024. 3

[36] Fu-Yun Wang, Wenshuo Chen, Guanglu Song, Han-Jia Ye, Yu Liu, and Hongsheng Li. Gen-l-video: Multi-text to long video generation via temporal co-denoising, 2023. 3

[37] Yuqing Wang, Tianwei Xiong, Daquan Zhou, Zhijie Lin, Yang Zhao, Bingyi Kang, Jiashi Feng, and Xihui Liu. Loong: Generating minute-level long videos with autoregressive language models, 2024. 3

[38] Desai Xie, Zhan Xu, Yicong Hong, Hao Tan, Difan Liu, Feng Liu, Arie Kaufman, and Yang Zhou. Progressive autoregressive video diffusion models, 2024. 1, 2, 3, 4, 6, 7

[39] Hu Xu, Saining Xie, Xiaoqing Ellen Tan, Po-Yao Huang, Russell Howes, Vasu Sharma, Shang-Wen Li, Gargi Ghosh, Luke Zettlemoyer, and Christoph Feichtenhofer. Demystifying clip data. In *ICLR*, 2024. 5

[40] Yanwu Xu, Yang Zhao, Zhisheng Xiao, and Tingbo Hou. Ufogen: You forward once large scale text-to-image generation via diffusion gans. In *CVPR*, 2024. 3

[41] Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models, 2022. 5

[42] Shengming Yin, Chenfei Wu, Huan Yang, Jianfeng Wang, Xiaodong Wang, Minheng Ni, Zhengyuan Yang, Linjie Li, Shuguang Liu, Fan Yang, Jianlong Fu, Gong Ming, Lijuan Wang, Zicheng Liu, Houqiang Li, and Nan Duan. Nuwa-xl: Diffusion over diffusion for extremely long video generation. In *ACL*, 2023. 3

[43] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *CVPR*, 2024. 3

[44] Tianwei Yin, Qiang Zhang, Richard Zhang, William T. Freeman, Fredo Durand, Eli Shechtman, and Xun Huang. From slow bidirectional to fast autoregressive video diffusion models. In *CVPR*, 2025. 1

[45] Lijun Yu, José Lezama, Nitesh B. Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, Vighnesh Birodkar, Agrim Gupta, Xiuye Gu, Alexander G. Hauptmann, Boqing Gong, Ming-Hsuan Yang, Irfan Essa, David A. Ross, and Lu Jiang. Language model beats diffusion – tokenizer is key to visual generation. In *ICLR*, 2024. 2

[46] Yuang Zhang, Jiaxi Gu, Li-Wen Wang, Han Wang, Junqi Cheng, Yuefeng Zhu, and Fangyuan Zou. Mimicmotion: High-quality human motion video generation with confidence-aware pose guidance, 2024. 3

[47] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, 2024. 2

[48] Daquan Zhou, Weimin Wang, Hanshu Yan, Weiwei Lv, Yizhe Zhu, and Jiashi Feng. Magicvideo: Efficient video generation with latent diffusion models, 2023. 2