PIXEL-SPACE VIDEO GENERATION WITH U-JIT

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent diffusion and flow matching models are ubiquitous in state-of-the-art generative video models. While highly effective, relying on pretrained autoencoders to compress the data into a lower-dimensional latent space also comes with drawbacks. First, training proceeds in two stages: autoencoder pretraining, with compression and reconstruction objectives, followed by latent generative model training, precluding fully end-to-end training. Second, autoencoders also lead to an inherent loss of signal, and the trade-off between reconstruction quality and generative performance in the latent space is still an active area of research. In this paper we explore the viability of generative video models directly in pixel space. We analyze recent approaches in pixel-space generative modeling of images that rely on basic transformer designs, and identify several shortcomings that limit their effectiveness for the high-dimensional pixel-space video generation. To overcome these limitations, we propose a UNet-like transformer architecture along with improvements to the training recipe and sampling strategy. We conduct controlled experiments, training latent and pixel-space models from scratch on Kinetics-600. For the standard frames-to-video task at 128 resolution, our U-JIT model improves generative FVD by 27% over prior state-of-the-art results, and by 14% against the JiT pixel-space baseline. Scaling to higher 256 resolution, we find the improvement over pixel-space and latent-based models grows as we increase the spatiotemporal compression rate up to $64\times64\times8$. Finally, we find that U-JIT improves per-frame FID by 27% over latent models and 61% over pixel-space models on class-to-video generation, with respectively 9% and 56% reduction in FVD.

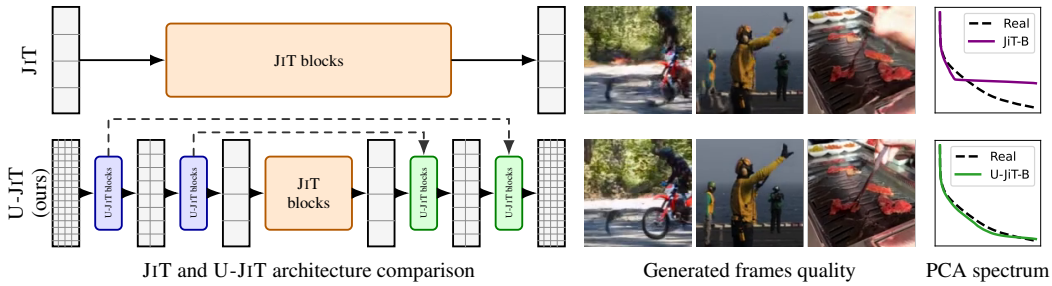

Figure 1: **Comparison of JIT and our proposed U-JIT on video generation.** While JIT processes the video with a single stack of large-patch blocks, U-JIT varies the patch size along model depth. On the right, frames generated by each model, and the PCA spectrum of generated *vs*. real videos.

1 INTRODUCTION

Recent advances on pixel-space image generation (Hoogeboom et al., 2025; Chen et al., 2026a;b; Pavasovic et al., 2026; Baade et al., 2026) have shown that Flow-Matching models can achieve high quality generation in ambient space, without relying on the latent spaces generated by autoencoders or "tokenizers". By using transformer-based models with large patch size, they also preserve the scalability and efficiency of their latent counterparts. Flow-Matching models are trained to predict, from a noised input $x_t = (1-t)\varepsilon + tx_0$, the velocity $v_t = x_0 - \varepsilon$ that transports the distribution of noise ε to the distribution of data x_0. The key idea behind pixel-space models is the hypothesis that the natural data lies on a low-dimensional manifold embedded in the high-dimensional ambient space (Turk and Pentland, 1991; Tipping and Bishop, 1999; Roweis and Saul, 2000; Carlsson, 2009;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Chapelle et al., 2006). This allows the predicted velocity to be decomposed into a compressible signal x_0 and the incompressible Gaussian noise ε, either explicitly through the objective (Li and He, 2026), or implicitly through the architecture design (Yu et al., 2026; Ma et al., 2026; Chen et al., 2026c).

However, state-of-the-art video generation models (Wan et al., 2025; Polyak et al., 2025; Jin et al., 2025; Yang et al., 2025; Gupta et al., 2025) rely on highly compressed latent representations (HaCohen et al., 2024; Team et al., 2026; Chen et al., 2025b) to achieve fast generation at scale, taking temporal redundancy. Direct application of pixel-space models to video is limited by the additional temporal dimension. Simple linear patchification layers cannot properly compress the high-dimensional data, as per-patch dimension becomes significantly higher than model hidden dimension.

In our work, we propose to address this limitation directly at the architectural level. After demonstrating how plain transformers struggle to generate high frequencies in video, we introduce a minimal adaptation of the transformer architecture, U-JiT (Figure 1), based on U-shaped transformers (Tian et al., 2024). We propose to keep the *stack of transformer blocks* as the main backbone, but to vary the patch size along the depth of the network, using single-pixel patches at the finest level. This design allows fine-grained modeling of high-dimensional data at the finest level coupled with a learned, progressive, non-linear compression of the pixel-space video to the internal hidden representations. Skip connections are added to allow propagation of high-frequency information, and adaptations to the attention mechanism in the early and late blocks are used to preserve efficiency.

We evaluate our method on the Kinetics-600 (Carreira et al., 2018) dataset, and show on the frames-to-video task that U-JiT improves FVD and FID by respectively 29% and 90% over the JiT pixel-space baseline, outperforms by 27% the FVD of the best published results on the benchmark, and achieves competitive FVD with 64% reduction in FID compared to our best latent-space baseline. We additionally show that among those models, U-JiT scales the best with respect to the patch size, supporting strong spatiotemporal compression that JiT models struggle with. Finally, for the more challenging class-conditional generation setting, we improve FVD by 9% over the best latent baseline, and by 56% over the JiT pixel-space baseline, with a high compression rate of f32t8.

We summarize our contributions as follows: (1) We provide a detailed experimental analysis of the plain transformer architecture for pixel-space video generation, and show its practical limitations. (2) We propose U-JiT, a simple and efficient adaptation of the transformer architecture for pixel-space video generation, with minimal changes to preserve efficiency and scalability. (3) We study three key behaviors of two-stage latent-space video generation models, and show how standard Flow-Matching mechanisms can be used as their replacements, improving the results of both JiT and U-JiT. (4) We conduct extensive experiments across model sizes and tasks on Kinetics-600, showing that U-JiT outperforms existing latent-space and pixel-space baselines across all metrics.

2 Related work

Diffusion and Flow-matching. Diffusion (Sohl-Dickstein et al., 2015; Ho et al., 2020; Song et al., 2021b; Dhariwal and Nichol, 2021; Song et al., 2021a; Nichol and Dhariwal, 2021) is a class of generative process modeling stochastic paths between a source (noise) and a target (data) distribution by learning to reverse a forward noising process. Flow Matching (Lipman et al., 2023; Liu et al., 2023; Albergo and Vanden-Eijnden, 2023) can be seen as a variant learning a deterministic velocity field between source and target. Notable design choices of the process itself include noise scheduling (Chen, 2023), sampling method (Karras et al., 2022), guidance mechanism (Ho and Salimans, 2021; Kynkäänniemi et al., 2024), training timestep sampling (Esser et al., 2024) and log-SNR shift with data dimensionality (Hoogeboom et al., 2023). Those methods are considered state-of-the-art for image (Peebles and Xie, 2023; Ma et al., 2024; Esser et al., 2024) and video (Ho et al., 2022b;a; Singer et al., 2023; Blattmann et al., 2023; Ma et al., 2025; Gupta et al., 2025; Lu et al., 2024) generation.

Latent-space generation. Latent diffusion models (Rombach et al., 2022) were introduced to allow high-dimensional generation by first compressing the data into a lower-dimensional latent space with an autoencoder (Kingma and Welling, 2022; van den Oord et al., 2017; Esser et al., 2021). These models inherently carry a trade-off between compression and generation quality (Chen et al., 2025a). Alignment with foundation models (Yao et al., 2025; Shi et al., 2026) and direct use of SSL encoders (Zheng et al., 2026; Singh et al., 2026) have been shown to improve the quality of

latent generation and to allow larger latent dimensions. On the other end, diffusion decoders (Zhao et al., 2025; Chen et al., 2025c; Vallaeys et al., 2026; Lu et al., 2026) focus on generating missing frequencies from the latent encoding. These autoencoders have been successfully extended to discrete (Yu et al., 2024) and continuous video tokenization (Guo et al., 2025; Gupta et al., 2025; Yang et al., 2025; Chen et al., 2025b; NVIDIA et al., 2025; Polyak et al., 2025; Wan et al., 2025; HaCohen et al., 2024; Kong et al., 2025). Various improvements have also been proposed for the latent model architecture, such as using and improving a transformer backbone (Peebles and Xie, 2023; Yao et al., 2025; Wang et al., 2026), and using U-Net (Ronneberger et al., 2015) inspired architectures (Hatamizadeh et al., 2025; Bao et al., 2023; Tian et al., 2024; Crowson et al., 2024).

End-to-end pixel-space generation. The main limitation of latent models is the two-stage training, which can introduce a loss of information and a latent space that is hard to diffuse. One way to address it is to jointly train the tokenizer and the latent model, either as a first-stage improvement (Heek et al., 2026), a finetuning stage after pretraining (Leng et al., 2025), or as a fully end-to-end training (Duggal et al., 2026; Chu et al., 2026; Meshchaninov et al., 2026). Another line of work tries to address the issue by directly generating in pixel space. Transformers are adapted by using additional convolutional layers (Hoogeboom et al., 2023; 2025), pixel-space specialized output heads (Ma et al., 2026; Chen et al., 2026c; Yu et al., 2026), or by keeping a plain transformer backbone with a reformulation of the target velocity (Li and He, 2026). We build upon this line of work, by identifying and overcoming some of the limitations of current pixel-space approaches.

3 METHOD

3.1 THE ARCHITECTURAL BOTTLENECK ON HIGH-DIMENSIONAL DATA

Natural images and videos lie on low-dimensional manifolds (Turk and Pentland, 1991; Tipping and Bishop, 1999; Roweis and Saul, 2000; Carlsson, 2009; Chapelle et al., 2006) embedded in high-dimensional ambient space, which implies that the intrinsic dimension of the data is much lower than the noise or velocity dimension. This has been partially addressed through objective reformulation in JiT (Li and He, 2026), which separates the incompressible noise from the compressible signal while maintaining a simple architectural design. We show here that this is not sufficient to capture the distribution of high-dimensional data such as videos. The relationship between the low-dimensional manifold and the high-dimensional ambient space is non-linear, implying that standard transformers are insufficient to capture this manifold with large patch sizes, resulting in lost high frequencies.

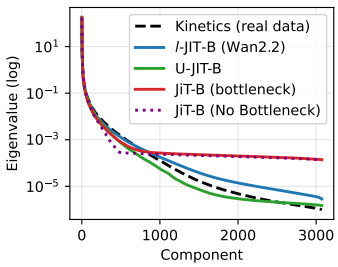

Figure 2: **PCA spectrum** over $16 \times 16 \times 4$ patches, computed over 1k real and sampled videos.

To empirically display this behavior on real data, we train a pixel-space JiT-B model on Kinetics-600 for 200 epochs on the standard task of generating 12 frames from 5 starting frames, with a patch size of $16 \times 16 \times 4$ and a frame resolution of 256×256. We additionally train a latent-space model (l-JiT-B (Wan2.2 Tokenizer)) and our custom architecture (U-JiT-B, see Section 3.2) on the same task. We display in Figure 2 the PCA spectrum of real and generated space-time video patches. We observe that JiT fails to capture high frequencies, as there is a sharp cut between two regimes of the spectrum: the first one roughly follows a power law, similarly to real data, while the second one is close to a flat Gaussian noise spectrum.[1] On the other hand, both the latent model and our architecture (U-JiT-B) manage to capture the real data spectrum more closely on a much wider range of frequencies.

To confirm this observation, we propose to model the distribution of real and generated patches as a mixture of two signals: a lower-dimensional signal P that captures the compressible part of the data (the signal subspace), and random Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$. This model is well aligned with the Flow-Matching generation process, which shifts samples from pure noise to data.

We follow Gavish and Donoho (2014) to empirically estimate the threshold between signal and noise in the PCA spectrum[2]. We find an estimated signal-subspace dimension of 1308 for real Kinetics-600

[1] By removing the input bottleneck projection, we confirmed that this frequency cut is related to the inner feature dimension of the model and not to that projection.

[2] We follow Eq. 26, Section E, to empirically estimate the noise level from the data.

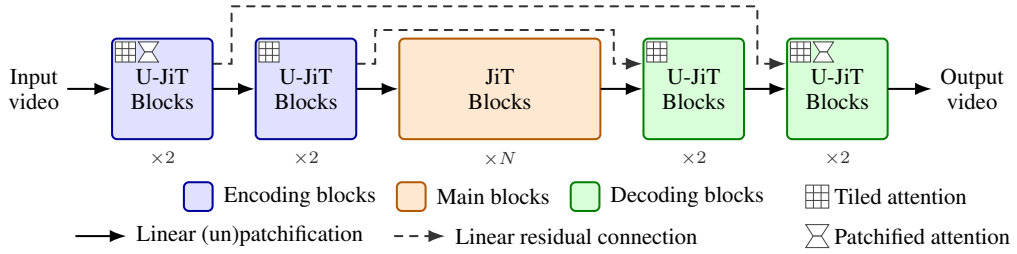

Figure 3: **U-JiT architecture** overview. Encoding and decoding blocks are parametrized to match standard JiT blocks used in the main backbone in computational cost.

videos, 1258 for videos generated in the Wan2.2 tokenizer latent space, 1284 for our U-JiT-generated videos, and only 596 for JiT-generated videos. This confirms that JiT fails to capture a large part of the signal, while the U-JiT architecture allows a better split between signal and noise, capturing a much wider range of frequencies.

3.2 U-JiT: VARYING PATCH-SIZE TRANSFORMER

In order to support efficient video generation, a model needs to handle a large spatiotemporal patch size, which reduces the number of tokens. The analysis in Section 3.1 motivates the need for architectural changes to natively handle large patch sizes in pixel space. Existing architectures from the image generation literature (Wang et al., 2026; Yu et al., 2026; Ma et al., 2026; Chen et al., 2026c) address this with a specialized output head, but suffer from two issues: (1) scalability: due to the large embedding dimension (Wang et al., 2026) or large, repeated, conditioning projection (Yu et al., 2026), their FLOPs and parameter count increase quadratically with the patch size, which makes them impractical for the stronger compression needed for video generation; (2) expressivity: they use a linear projection layer to compress the input, which is significantly less expressive than the multi-layer encoders employed by latent space methods to extract compacts latents, thereby losing access to useful signal throughout the model up until the output head.

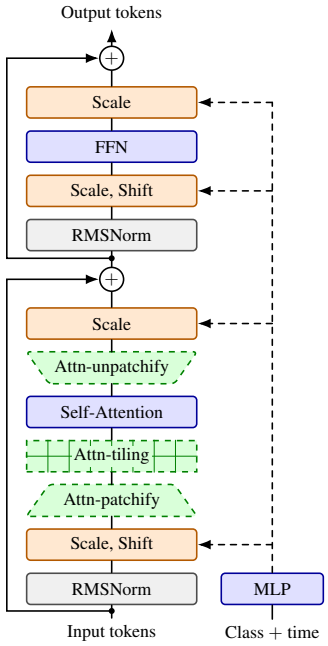

Figure 4: **U-JiT block, with attention tiling and patchification blocks.**

We propose instead to design our U-JiT as a varying-patch-size, U-Net shaped transformer, based on U-DiT (Tian et al., 2024), using JiT blocks (Li and He, 2026) and additional adaptations to attention to remain FLOPs-efficient. This ensures scalability by keeping standard transformer blocks, with only minimal changes in the early (*encoding*) and late (*decoding*) layers. The overall architecture is shown in Figure 3. We follow the standard DiT (Peebles and Xie, 2023) model sizes and use 3 levels (2 encoding / decoding levels, and one main level), adjusting the overall number of blocks to keep the same computational cost as JiT. The central (main) level uses exactly the same blocks as JiT models, with the same patch size and hidden dimension.

The encoding and decoding blocks, referred to as U-JiT blocks (see Figure 4), use a smaller patch size (one per level), and scale their hidden dimension relative to their patch size to preserve the same number of FLOPs as JiT models. Similarly to U-DiT, we use a single residual connection per level, projected with a zero-initialized linear layer. Downsampling and upsampling operations are learned as a linear operation combined with spacetime-to-depth / depth-to-spacetime reshaping, similarly to the input / output projections of JiT models. Furthermore, to allow patchification of individual frames, this linear projection is expressed as two linear layers, one for the spatial dimension and one for the temporal dimension, with the second one skipped for tokens representing individual frames. We additionally perform the adaptations detailed below to the attention mechanism of the encoding /

decoding blocks to remain FLOPs-efficient with small patch sizes on videos. Ablations on the impact of each architetural component of our model are provided in Table S4.

Attention tiling. Similarly to U-DiT (Tian et al., 2024), we use attention tiling to reduce attention complexity inside encoding / decoding layers, by splitting the token grid without changing the attention operation itself. Instead of fixing the number of grid splits, we use an attention tiling window size of $8 \times 8 \times 4$ for all our models, which scales linearly with video length.

Attention patchification. In order to support small spatiotemporal patch sizes of $1 \times 1 \times 1$ or $2 \times 2 \times 1$, we use a patchified attention, following the linear compressed attention of PixelDiT (Yu et al., 2026). We group tokens into spatiotemporal patches that are linearly projected to obtain a smaller token grid, with a higher embedding dimension per token, before applying the attention. A similar linear expansion is applied to the output of the attention, to recover the original token grid. In practice, we use attention patchification only for the first encoding / last decoding blocks (pixel-level blocks), with the same compression patch size as the second level of the U-JiT. We scale the attention hidden dimension to preserve FLOPs.

3.3 Analyzing the two-stage generative process of latent-space models

Latent space generative models can be seen as a two-stage generative process, where the first stage (latent flow-matching) iteratively generates a low-dimensional latent representation of the video, and the second stage decodes this latent representation into a high-dimensional video as a single decoding step. As these two stages behave differently, we make three key observations that we detail below. We support each one by performing an analysis on a latent l-JiT-B model trained using the Wan2.2 tokenizer on 256×256 resolution, on 16-frame Kinetics-600 clips.

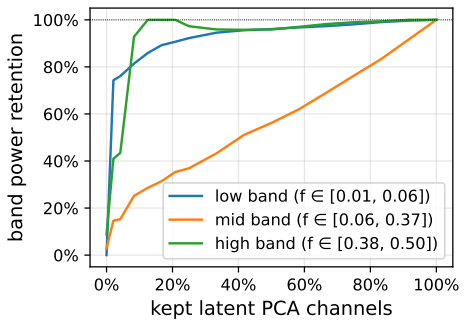

Figure 5: **Band power retention in decoded videos, depending on latent capacity.**

1. Low/high frequency split between latent and decoder. The latent dimensionality is usually significantly lower than that of the video, which limits the latent to mainly capturing low and mid frequencies, while the decoder is responsible for generating high frequencies. This makes latent-space models effective two-stage generative processes.

To identify which frequency bands are captured by the latent and which ones are generated by the decoder, we reduce the latent capacity by applying a PCA decomposition on it, keeping only the first k PCA components. We sweep over k and display in Figure 5 the power retention ratio of the low, mid and high frequency bands in the decoded video, compared to the full-latent decoded video. We observe that low frequencies are mostly captured by the first PCA components (around 90% power retention with 20% of the PCA components), while mid frequencies increase almost linearly with the number of PCA components, indicating that successive PCA components capture from low to mid frequencies. Conversely, high frequencies consistently keep from 90% to 100% of their power, even when removing 90% of the PCA components, at which point the latent encodes almost only low frequencies. This indicates that the decoder is responsible for generating most of the high frequencies, while the latent process generates low to mid bands.

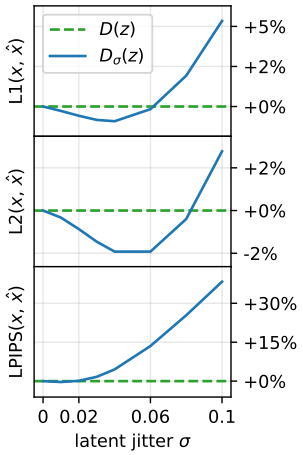

Figure 6: **Reconstruction quality over latent jittering.**

2. On-manifold high-frequency generation. Adversarially-trained decoders are optimized to generate on-manifold high-frequency details: from a given latent encoding z, the decoder does not generate the most likely video in the ambient space (the closest in MSE on average), but the most likely one on the embedded manifold. The theoretical insights behind this effect are discussed by Blau and Michaeli (2018).

To identify if the video decoder effectively benefits from this effect, we design a simple variant D_σ of the original decoder D to increase ambient space reconstruction fidelity. Given an input video x and its latent encoding z, we reconstruct the video by averaging the output of D over $K = 16$ randomly jittered latent encodings, yielding $\hat{x} = D_\sigma(z) = \frac{1}{K}\sum_{k=1}^{K} D(z + \sigma\varepsilon_k)$ where $\varepsilon_k \sim \mathcal{N}(0, I)$. In Figure 6 we plot reconstruction metrics, sweeping over the hyperparameter σ. For $\sigma \in [0.02, 0.06]$, our jittered decoder pulls the decoded video closer to the initial datapoint, as measured by L_1 and L_2 distances. Conversely, the perceptual quality of the video is degraded, as measured by LPIPS. This confirms that the decoder actively pushes the generated video on-manifold, away from the closest reconstruction in ambient space.

3. Perceptually-aligned variance. The latent Flow-Matching model is trained with an isotropic Gaussian noise schedule and an L_2 (MSE) objective, which is agnostic to any perceptual metric and weights every direction in latent space equally. As a result, its residual error is approximately isotropic in the latent space.

We investigate how the decoder maps isotropic latent variations to anisotropic, perceptually aligned variations in the video pixel space. To this end we decode a latent z into $\hat{x}$, and a perturbed latent $z + \sigma_z\varepsilon_z$ with isotropic noise into $\hat{x}_{\sigma_z}$. Averaging across noise and samples, we measure the effective pixel-space noise scale as the L_2 distance σ_x between the clean decoded video $\hat{x}$ and the perturbed decoded video $\hat{x}_{\sigma_z}$. We similarly perturb the reconstructed video $\hat{x}$ directly with isotropic noise in pixel space, as $\hat{x} + \sigma_x\varepsilon_x$. We compare L_1 and LPIPS metrics between both latent (z-noise) and pixel-space (x-noise) perturbations in Figure 7. For a similar, increasing pixel-space noise magnitude, the LPIPS metric deteriorates much faster for the pixel-space perturbation than for the latent perturbation. This shows that the decoder effectively maps isotropic latent noise to perceptually aligned variations in the video space, which is crucial to generate on-manifold high-frequency details. Additionally, the L_1 distance to the original decoded sample also increases slightly faster, indicating that the latent perturbation is mapped to a sparser variation.

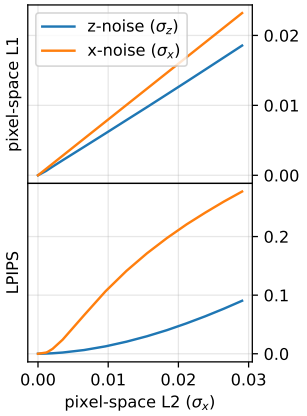

Figure 7: **Effect of two types of noise on metrics.**

3.4 Sampling on-manifold in pixel space

The analysis in Section 3.3 identifies three key mechanisms, specific to latent-space models, that improve the generation of high-quality videos. To bridge the gap between pixel-space and latent-space generation, we propose to adapt the training and sampling processes of pixel-space models, using known methods from Flow-Matching literature, to address the identified issues.

Table 1: **U-JiT-B sampling sweep on Kinetics-600, at 200 epochs.** A CFG of 1 reverts to non-CFG sampling, and $\sigma_t = 1$ to linear time sampling.

		CFG					
	σ_t	1	1.5	2	2.5	3	3.5
FVD@5k $\rightarrow$	1	3.43	3.21	3.15	3.15	3.16	3.17
	2	3.37	3.17	3.14	3.13	3.13	3.15
	4	3.33	3.14	3.12	3.11	3.12	3.15
	8	3.29	3.14	**3.10**	3.14	3.20	3.29
	16	3.32	3.15	3.14	3.26	3.47	3.78
FID@5k $\rightarrow$	1	21.9	18.4	17.1	16.5	16.1	15.9
	2	19.0	16.3	15.4	15.0	14.9	14.9
	4	17.2	15.0	14.2	13.9	14.0	14.2
	8	15.7	14.0	13.5	13.6	14.1	14.9
	16	14.9	**13.4**	**13.4**	14.2	15.8	18.0

Time-shifting: focusing the sampling on low-frequencies. Latent models use most of their compute on the low-frequency structure of the video (latent flow-matching), and a single decoding step to generate high-frequency details. We use time-shifting (Hoogeboom et al., 2023) *at inference only* to shift the sampling process toward low frequencies, keeping only a few steps to generate the high-frequency details. In practice, given N sampling steps, we use the sampling schedule $t_i = \frac{i}{i+\sigma_t(N-i)}$, reverting to the linear time sampling schedule for $\sigma_t = 1$. We validate in a simple setting in Table 1 that time-shifting can lead to significant improvements in FVD and FID metrics.

Classifier-free guidance: on-manifold high-frequency details. On the standard task of video completion (such as on the Kinetics dataset), where the strong conditioning is coupled with a GAN-trained decoder, classifier-free guidance is usually deemed unnecessary for latent-space models. We find in Table 1 that, for pixel-space models, applying classifier-free guidance (Ho and Salimans,

2021) with an interval (Kynkäänniemi et al., 2024) during sampling noticeably improves the quality of generated videos, as it allows the model to generate on-manifold high-frequency details.

Perceptual loss: aligning generation variations along perceptually-correct directions. Flow-Matching models are trained to capture the distribution of real training samples, and to generalize to realistic non-training samples. To ensure that the variations in high frequencies are aligned with perceptually-correct directions, we apply a perceptual loss (Zhang et al., 2018) during training. Ablation on the loss impact is included in Table S4.

We found time-shifting and classifier-free guidance to also improve, to a lesser extent, the performance of latent-space models on all tasks (see Table S3). To ensure a fair comparison, we use them in all our experiments. We find that a time shift $\sigma_t = 8$ performs best on average across tasks and models for pixel-space models, and $\sigma_t = 8$ for latent-space models. We select the CFG value for each model on each task with a grid search (step of 0.5), as the optimal CFG value differs substantially across tasks and models. Following (Kynkäänniemi et al., 2024), we apply CFG only in the ODE time interval of $[0.1, 1]$ for all models.

3.5 Model architecture and training details

We follow Yao et al. (2025) by using DiT models (Peebles and Xie, 2023) with the same improvements: SwiGLU activations (Shazeer, 2020), RMSNorm layers (Zhang and Sennrich, 2019), RoPE positional embeddings (Su et al., 2024), and qk-norm (Henry et al., 2020), together with the additional context tokens introduced by Li and He (2026). We refer to these models as JiT models in pixel space, l-JiT models when trained in latent space, and U-JiT when incorporating our architecture modifications. We train all models with the same training objective as JiT (Li and He, 2026), using x-prediction and a velocity loss, which we found more effective than v-prediction for both pixel-space and latent-space video models. We use noise scaling to scale across video resolutions and lengths. We refer to Appendix A for additional implementation details and per-model FLOPs and parametrization.

4 Experiments

4.1 Experimental Setup

We train and evaluate our models on the Kinetics-600 dataset (Carreira et al., 2018), and use the FVD metric (Unterthiner et al., 2019) to evaluate overall video quality. As FVD assesses distribution match using spatio-temporal features from an action recognition model, we also evaluate FID (Heusel et al., 2017) to assess per-frame quality. We compare our pixel-space models to latent-space models trained using the following state-of-the-art video tokenizers: Wan2.1 and Wan2.2 (Wan et al., 2025), LTX (HaCohen et al., 2024), and the continuous Cosmos tokenizer (NVIDIA et al., 2025). We write fFtT for a spatiotemporal compression rate of F along the spatial axes and T along the temporal axis. As the Wan2.2 and LTX video tokenizers have respective compression rates of f16t4 and f32t8, we only use them in strong compression settings. Additional details about training and evaluation are provided in Appendix A.

4.2 Frames to video generation

We evaluate our models on the standard frames-to-video generation task (Ho et al., 2022b), where the model is given 5 frames as input, and is evaluated on generating the next 11 frames (in practice, 12 are generated due to patchification). We follow recent work (Pavasovic et al., 2026; Heek et al., 2026; Gupta et al., 2025) by training and evaluating at resolution 128×128 with spatiotemporal compression f8t4. We first train l-JiT-B, JiT-B and U-JiT-B models on this task, and report their results in Table 2. Our latent-space models (l-JiT-XL) outperform existing latent-space video generation methods, and existing pixel-space using Wan2.1. Results show that our U-JiT is on par with the best latent-space model in terms of FVD, and significantly better in terms of FID. We also compare U-JiT-B with JiT-B under both our sampling strategy from Section 3.4 and *base sampling* that does not use CFG nor time-shifting.[3]. This confirms that both our sampling strategy

[3]For both latent l-JiT and pixel-space JiT and U-JiT models we otherwise sweep for the best CFG and time shifting, where a values of 1 are equivalent to base sampling.

Table 2: Frames to video: 5 conditioning frames, 12 generated frames, at resolution 128×128 with spatiotemporal compression f8t4. B- and XL-sized models. FLOPs are computed for a single forward pass of the model. *Pixel* column indicates pixel-space (✓) and latent-space (×) generative models.

Model (f8t4)	Pixel	#	FLOPs	↓ FVD	↓ FID
Video Diffusion Models (Ho et al., 2022b)	✓	1.1B	4.1T	16.2	-
RIN (Jabri et al., 2023)	✓	411M	386G	10.8	-
MAGVIT-v2 (Yu et al., 2024)	×	307M	-	4.3	-
DeRA (Guo et al., 2025)	×	632M	-	4.1	-
W.A.L.T (Gupta et al., 2025)	×	313M	-	3.3	-
Unified Latents (Heek et al., 2026)	×	-	-	1.3	-
WaiT-XL (Pavasovic et al., 2026)	✓	678M	1.3T	0.84	-
l-JiT-B (Cosmos Tokenizer)	×	124M	264G	1.50	8.36
l-JiT-B (Wan2.1 Tokenizer)	×	124M	264G	1.09	12.13
JiT-B (base sampling)	✓	125M	266G	1.68	45.20
JiT-B	✓	125M	266G	1.22	7.28
U-JiT-B (ours, base sampling)	✓	98M	265G	1.25	7.17
U-JiT-B (ours)	✓	98M	265G	1.06	**3.08**
l-JiT-XL (Cosmos Tokenizer)	×	643M	1.3T	1.00	8.78
l-JiT-XL (Wan2.1 Tokenizer)	×	643M	1.3T	<u>0.62</u>	11.05
JiT-XL (base sampling)	✓	644M	1.3T	0.86	32.43
JiT-XL	✓	644M	1.3T	0.71	5.29
U-JiT-XL (ours)	✓	587M	1.3T	**0.61**	<u>3.15</u>

and architecture each significantly contribute, on their own, to closing the gap between latent-space and pixel-space models, while combining both is necessary to match the FVD surpass the FID of latent-space models.

Scaling our models. To validate the improvements brought by our model at larger scales, we increase the model size to XL on the same task in Table 2. We observe clear FVD improvements for all methods, and similar FID for U-JiT and *l*-JiT. Our model stays on par with the best latent-space model in terms of FVD, with a clear per-frame FID quality improvement, and is noticeably better than JiT. We also reach state-of-the-art performance on the Kinetics-600 benchmark, with a 27% improvement in FVD compared to the previous best published results.

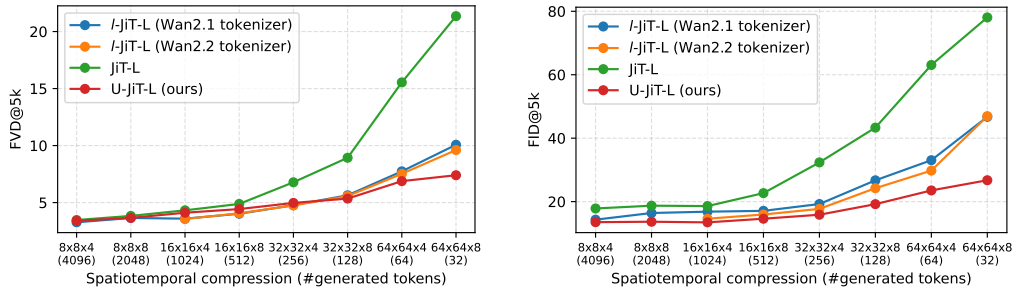

Figure 8: **Scaling the spatiotemporal compression rate.** FVD@5k (left) and FID@5k (right) as functions of the compression. Evaluated at 400 epochs, 1 to 15 frames, resolution 256×256.

4.3 Strong Video Compression

As established in Section 3.1, the main challenge of pixel-space video generation lies in the high dimensionality of the video data, which is exacerbated by the use of strong spatiotemporal compression, necessary for fast, high-resolution generation. To this end, we study the effect of increased spatiotemporal compression on our models, compared to latent-space and vanilla JiT.

Scaling along the compression rate. We first train U-JiT-L, JiT-L and *l*-JiT-L (using the Wan2.1 and Wan2.2 tokenizers) models on the single-frame-to-video generation task, at resolution 256×256, where the model is given 1 frame as input and is evaluated on the next 15 frames (in practice, 16 are generated and the last one is discarded). Models are trained for 400 epochs and evaluated on 5k samples. We report results in Figure 8. The JiT base-

line shows a clear degradation in performance compared to latent-space models, as it struggles with high-dimensional data. On the other hand, our U-JiT model is as robust as latent-space models to increased spatiotemporal compression, following closely their FVD, and improving over them for the strongest $f64t8$ compression setting. It additionally shows improved per-frame quality (FID) at high compression rates.

Strong compression analysis. We validate our findings by scaling to XL-sized models on the same task at a strong compression rate (f32t8), training until convergence (600 epochs) and evaluating on 50k samples. Results in Table 3 confirm our patch size scaling analysis, with U-JiT-XL outperforming all latent-space models in terms of both FVD and FID, and JiT-XL suffering from a clear degradation in performance.

Table 3: **Image to video, strong compression**. 1 to 16 frames, at resolution 256×256 with spatiotemporal compression f32t8.

Model (f32t8)	#	↓ FVD	↓ FID
l-JiT-XL (Cosmos Tokenizer)	644M	3.94	12.41
l-JiT-XL (Wan2.1 Tokenizer)	644M	2.73	12.86
l-JiT-XL (Wan2.2 Tokenizer)	644M	<u>2.66</u>	<u>9.97</u>
l-JiT-XL (LTX Tokenizer)	643M	5.09	21.06
JiT-XL (base sampling)	700M	8.98	52.53
JiT-XL	700M	5.86	30.27
U-JiT-XL (ours)	478M	**2.55**	**7.20**

4.4 Class to video generation

Video completion evaluates the ability to continue a sequence of frames, with a strong conditioning signal that specifies most of the structure and background of the video. This mainly focuses on the capacity of the model to output coherent motion, and high-quality frames similar to the input, which is close to an auto-encoding task. We also evaluate our method on the harder class-to-video generation task, where the full distribution of the content is learned by the model. We use the same strong compression setting of f32t8 as in Section 4.3, training *l*-JiT-XL, JiT-XL and U-JiT-XL models to generate 65 frames from a class condition, corresponding to 8 seconds of video, at resolution 256×256. To compute FVD, we split each of the 50k generated videos into 4 clips of 16 frames, yielding a total of 200k clips for evaluation. Results reported in Table 4 show that our U-JiT outperforms all latent-space models, increasing the gap in FVD improvement from 4% to 9% compared to the image-to-video task (Table 3). This confirms that U-JiT generalizes to general video generation tasks, learning the full distribution of low- and high-level features.

Table 4: **Class to video**. 65 frames generated from a class condition, at resolution 256×256 with strong spatiotemporal compression f32t8.

Model (f32t8)	#	↓ FVD	↓ FID
l-JiT-XL (Cosmos Tokenizer)	647M	4.78	<u>17.44</u>
l-JiT-XL (Wan2.1 Tokenizer)	647M	<u>3.31</u>	21.52
l-JiT-XL (Wan2.2 Tokenizer)	647M	3.55	18.08
l-JiT-XL (LTX Tokenizer)	646M	3.96	22.24
JiT-XL (base sampling)	707M	9.45	65.49
JiT-XL	707M	6.88	32.60
U-JiT-XL (ours)	478M	**3.00**	**12.65**

5 Conclusion

Two-stage latent-space generation models are inherently limited by the quality of the tokenizer, whose objectives are not easily aligned with the generation task: reconstruction fidelity, compression ratio and diffusability are all competing objectives that are hard to optimize simultaneously. End-to-end pixel-space generation offers a paradigm that bypasses the tokenizer bottleneck, but is limited by the high dimensionality of the data space. Previous work has shown how to separate high-dimensional noise from the compressible signal to provide accurate velocity prediction in ambient space. In this work, we show that this is not sufficient to capture the distribution of high-dimensional data such as videos, and propose a simple adaptation of the transformer architecture to address this issue. Our findings show that U-JiT, a multi-resolution pixel-space model, can be trained end-to-end to achieve results competitive with latent-space models, while providing better scalability to strong compression settings. We hope that this work will encourage further research on end-to-end video generation.

AI use statement

In this work, we used generative AI tools to implement methods, support data analysis, create TikZ figures, format LaTeX code (but not to write text), check grammar and spelling, and identify additional relevant literature (but not to interpret papers). We have not used generative AI tools to propose

or refine hypotheses, design or provide feedback on research methodology or experiments, clean and reformat datasets, interpret results, suggest experimental parameters, write or structure the text of the paper, or format references. *Generating synthetic data sets, developing theoretical models, formulating mathematical claims, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, assisting with translation* are not applicable to this work. We have reviewed all AI-assisted work. We have verified all code generated by AI tools, and implemented critical parts of the method manually. We conducted the analysis and interpretation ourselves, limiting AI to implementing analysis code. We have verified all figures generated from code written by AI tools, as well as the LaTeX output and grammar edits. We only used AI agents to complement our own literature review, without relying on them to interpret or summarize papers.

Reproducibility statement

We ensure that all experiments are reproducible by providing detailed implementation details in Appendix A, with all the model configurations and training hyperparameters used in our result tables. We additionally detail data processing and evaluation procedures, referring to original implementations of the metrics used.

References

Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=li7qeBbCR1t.

Alan Baade, Eric Ryan Chan, Kyle Sargent, Changan Chen, Justin Johnson, Ehsan Adeli, and Li Fei-Fei. Latent forcing: Reordering the diffusion trajectory for pixel-space image generation. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=LFUaHUCO1a.

Fan Bao, Shen Nie, Kaiwen Xue, Yue Cao, Chongxuan Li, Hang Su, and Jun Zhu. All are worth words: A vit backbone for diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22669–22679, June 2023.

Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, Varun Jampani, and Robin Rombach. Stable video diffusion: Scaling latent video diffusion models to large datasets, 2023. URL https://arxiv.org/abs/2311.15127.

Yochai Blau and Tomer Michaeli. The perception-distortion tradeoff. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Gunnar Carlsson. Topology and data. *Bulletin of the American Mathematical Society*, 46(2):255–308, 2009.

Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600, 2018. URL https://arxiv.org/abs/1808.01340.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien, editors. *Semi-Supervised Learning*. MIT Press, Cambridge, MA, USA, 2006.

Hansheng Chen, Jan Ackermann, Minseo Kim, Gordon Wetzstein, and Leonidas Guibas. Asymmetric flow models, 2026a. URL https://arxiv.org/abs/2605.12964.

Junyu Chen, Han Cai, Junsong Chen, Enze Xie, Shang Yang, Haotian Tang, Muyang Li, and Song Han. Deep compression autoencoder for efficient high-resolution diffusion models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu, editors, *International Conference on Learning Representations*, volume 2025, pages 96539–96560, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/f062da1973ac9ac61fc6d44dd7fa309f-Paper-Conference.pdf.

Junyu Chen, Wenkun He, Yuchao Gu, Yuyang Zhao, Jincheng Yu, Junsong Chen, Dongyun Zou, Yujun Lin, Zhekai Zhang, Muyang Li, Haocheng Xi, Ligeng Zhu, Enze Xie, Song Han, and Han Cai. Dc-videogen: Efficient video generation with deep compression video autoencoder, 2025b. URL https://arxiv.org/abs/2509.25182.

Ting Chen. On the importance of noise scheduling for diffusion models, 2023. URL https://arxiv.org/abs/2301.10972.

Yinbo Chen, Rohit Girdhar, Xiaolong Wang, Sai Saketh Rambhatla, and Ishan Misra. Diffusion autoencoders are scalable image tokenizers, 2025c. URL https://arxiv.org/abs/2501.18593.

Zhennan Chen, Junwei Zhu, Xu Chen, Jiangning Zhang, Jiawei Chen, Zhuoqi Zeng, Wei Zhang, Chengjie Wang, Jian Yang, and Ying Tai. L2p: Unlocking latent potential for pixel generation, 2026b. URL https://arxiv.org/abs/2605.12013.

Zhennan Chen, Junwei Zhu, Xu Chen, Jiangning Zhang, Xiaobin Hu, Hanzhen Zhao, Chengjie Wang, Jian Yang, and Ying Tai. Dip: Taming diffusion models in pixel space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 36136–36146, June 2026c.

Wenda Chu, Bingliang Zhang, Jiaqi Han, Yizhuo Li, Linjie Yang, Yisong Yue, and Qiushan Guo. End-to-end autoregressive image generation with 1d semantic tokenizer. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=fSnxuRhWoC.

Katherine Crowson, Stefan Andreas Baumann, Alex Birch, Tanishq Mathew Abraham, Daniel Z Kaplan, and Enrico Shippole. Scalable high-resolution pixel-space image synthesis with hourglass diffusion transformers. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 9550–9575. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/crowson24a.html.

Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 16344–16359. Curran Associates, Inc., 2022. doi: 10.52202/068431-1189. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/67d57c32e20fd0a7a302cb81d36e40d5-Paper-Conference.pdf.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8780–8794. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/49ad23d1ec9fa4bd8d77d02681df5cfa-Paper.pdf.

Shivam Duggal, Xingjian Bai, Zongze Wu, Richard Zhang, Eli Shechtman, Antonio Torralba, Phillip Isola, and William T. Freeman. End-to-end training for unified tokenization and latent denoising, 2026. URL https://arxiv.org/abs/2603.22283.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12873–12883, June 2021.

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, and Robin Rombach. Scaling rectified flow transformers for high-resolution image synthesis. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 12606–12633. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/esser24a.html.

Matan Gavish and David L. Donoho. The optimal hard threshold for singular values is $4/\sqrt{3}$. *IEEE Transactions on Information Theory*, 60(8):5040–5053, 2014. doi: 10.1109/TIT.2014.2323359.

Pengbo Guo, Junke Wang, Zhen Xing, Chengxu Liu, Daoguo Dong, Xueming Qian, and Zuxuan Wu. Dera: Decoupled representation alignment for video tokenization, 2025. URL https://arxiv.org/abs/2512.04483.

Agrim Gupta, Lijun Yu, Kihyuk Sohn, Xiuye Gu, Meera Hahn, Fei-Fei Li, Irfan Essa, Lu Jiang, and José Lezama. Photorealistic video generation with diffusion models. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 393–411, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72986-7.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, Poriya Panet, Sapir Weissbuch, Victor Kulikov, Yaki Bitterman, Zeev Melumian, and Ofir Bibi. Ltx-video: Realtime video latent diffusion, 2024. URL https://arxiv.org/abs/2501.00103.

Ali Hatamizadeh, Jiaming Song, Guilin Liu, Jan Kautz, and Arash Vahdat. Diffit: Diffusion vision transformers for image generation. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 37–55, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-73242-3.

Jonathan Heek, Emiel Hoogeboom, Thomas Mensink, and Tim Salimans. Unified latents (ul): How to train your latents, 2026. URL https://arxiv.org/abs/2602.17270.

Alex Henry, Prudhvi Raj Dachapally, Shubham Shantaram Pawar, and Yuxuan Chen. Query-key normalization for transformers. In Trevor Cohn, Yulan He, and Yang Liu, editors, *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4246–4253, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.findings-emnlp.379. URL https://aclanthology.org/2020.findings-emnlp.379/.

Karl Heun et al. Neue methoden zur approximativen integration der differentialgleichungen einer unabhängigen veränderlichen. *Z. Math. Phys*, 45(23-38):7, 1900.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/8a1d694707eb0fefe65871369074926d-Paper.pdf.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Jonathan Ho, William Chan, Chitwan Saharia, Jay Whang, Ruiqi Gao, Alexey Gritsenko, Diederik P. Kingma, Ben Poole, Mohammad Norouzi, David J. Fleet, and Tim Salimans. Imagen video: High definition video generation with diffusion models, 2022a. URL https://arxiv.org/abs/2210.02303.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David Fleet. Video diffusion models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 8633–8646. Curran Associates, Inc., 2022b. doi: 10.52202/068431-0628. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/39235c56aef13fb05a6adc95eb9d8d66-Paper-Conference.pdf.

Emiel Hoogeboom, Jonathan Heek, and Tim Salimans. simple diffusion: End-to-end diffusion for high resolution images. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 13213–13232. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/hoogeboom23a.html.

Emiel Hoogeboom, Thomas Mensink, Jonathan Heek, Kay Lamerigts, Ruiqi Gao, and Tim Salimans. Simpler diffusion: 1.5 fid on imagenet512 with pixel-space diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18062–18071, June 2025.

Allan Jabri, David J. Fleet, and Ting Chen. Scalable adaptive computation for iterative generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 14569–14589. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/jabri23a.html.

Yang Jin, Zhicheng Sun, Ningyuan Li, Kun Xu, Hao Jiang, Nan Zhuang, Quzhe Huang, Yang Song, Yadong MU, and Zhouchen Lin. Pyramidal flow matching for efficient video generative modeling. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu, editors, *International Conference on Learning Representations*, volume 2025, pages 23378–23402, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/3ab228c4703c4459b1a600ebadc5732c-Paper-Conference.pdf.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 26565–26577. Curran Associates, Inc., 2022. doi: 10.52202/068431-1926. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/a98846e9d9cc01cfb87eb694d946ce6b-Paper-Conference.pdf.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2022. URL https://arxiv.org/abs/1312.6114.

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. URL https://arxiv.org/abs/2412.03603.

Tuomas Kynkäänniemi, Miika Aittala, Tero Karras, Samuli Laine, Timo Aila, and Jaakko Lehtinen. Applying guidance in a limited interval improves sample and distribution quality in diffusion models. In *NeurIPS*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/dd540e1c8d26687d56d296e64d35949f-Paper-Conference.pdf.

Xingjian Leng, Jaskirat Singh, Yunzhong Hou, Zhenchang Xing, Saining Xie, and Liang Zheng. Repa-e: Unlocking vae for end-to-end tuning of latent diffusion transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 18262–18272, October 2025.

Tianhong Li and Kaiming He. Back to basics: Let denoising generative models denoise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 36115–36125, June 2026.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

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 B. Kim, Y. Yue, S. Chaudhuri, K. Fragkiadaki, M. Khan, and Y. Sun, editors, *International Conference on Learning Representations*, volume 2024, pages 19259–19286, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/543ec10715d964122ab7cb15f648772b-Paper-Conference.pdf.

Yifan Lu, Qi Wu, Jay Zhangjie Wu, Zian Wang, Huan Ling, Sanja Fidler, and Xuanchi Ren. Pid: Fast and high-resolution latent decoding with pixel diffusion, 2026. URL https://arxiv.org/abs/2605.23902.

Nanye Ma, Mark Goldstein, Michael S. Albergo, Nicholas M. Boffi, Eric Vanden-Eijnden, and Saining Xie. Sit: Exploring flow and diffusion-based generative models with scalable interpolant transformers. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 23–40, Cham, 2024. Springer Nature Switzerland. ISBN 978-3-031-72980-5.

Xin Ma, Yaohui Wang, Xinyuan Chen, Gengyun Jia, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *Transactions on Machine Learning Research*, 2025. ISSN 2835-8856. URL https://openreview.net/forum?id=ntGPYNUF3t.

Zehong Ma, Longhui Wei, Shuai Wang, Shiliang Zhang, and Qi Tian. Deco: Frequency-decoupled pixel diffusion for end-to-end image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 43600–43610, June 2026.

Viacheslav Meshchaninov, Alexander Shabalin, Egor Chimbulatov, Nikita Gushchin, Ilya Koziev, Alexander Korotin, and Dmitry Vetrov. How to train your latent diffusion language model jointly with the latent space. In *ICML 2026 Workshop on Foundations of Deep Generative Models: Understanding Memorization, Generalization, and Reasoning*, 2026. URL https://openreview.net/forum?id=IZ1jSNabMs.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8162–8171. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/nichol21a.html.

NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical ai, 2025. URL https://arxiv.org/abs/2501.03575.

Krunoslav Lehman Pavasovic, Théophane Vallaeys, Stéphane Mallat, Giulio Biroli, Luke Zettlemoyer, Brian Karrer, and Jakob Verbeek. Wait for the signal: Simple frequency-aware flow-matching, 2026. URL https://arxiv.org/abs/2607.28760.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4195–4205, October 2023.

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, 2025. URL https://arxiv.org/abs/2410.13720.

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 (CVPR)*, pages 10684–10695, June 2022.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, 2015.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202.

Minglei Shi, Haolin Wang, Wenzhao Zheng, Ziyang Yuan, Xiaoshi Wu, Xintao WANG, Pengfei Wan, Jie Zhou, and Jiwen Lu. Latent diffusion model without variational autoencoder. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust, editors, *International Conference on Learning Representations*, volume 2026, pages 154506–154537, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/fa5ddd6bac0d665c72969d79221b680a-Paper-Conference.pdf.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, Devi Parikh, Sonal Gupta, and Yaniv Taigman. Make-a-video: Text-to-video generation without text-video data. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=nJfylDvgzlq.

Jaskirat Singh, Boyang Zheng, Zongze Wu, Richard Zhang, Eli Shechtman, and Saining Xie. Improved baselines with representation autoencoders, 2026. URL https://arxiv.org/abs/2605.18324.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

FSVideo Team, Qingyu Chen, Zhiyuan Fang, Haibin Huang, Xinwei Huang, Tong Jin, Minxuan Lin, Bo Liu, Celong Liu, Chongyang Ma, Xing Mei, Xiaohui Shen, Yaojie Shen, Fuwen Tan, Angtian Wang, Xiao Yang, Yiding Yang, Jiamin Yuan, Lingxi Zhang, and Yuxin Zhang. Fsvideo: Fast speed video diffusion model in a highly-compressed latent space, 2026. URL https://arxiv.org/abs/2602.02092.

Yuchuan Tian, Zhijun Tu, Hanting Chen, Jie Hu, Chao Xu, and Yunhe Wang. U-dits: Downsample tokens in u-shaped diffusion transformers. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 51994–52013. Curran Associates, Inc., 2024. doi: 10.52202/079017-1648. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/5d2e24df9cfaad3189833b819c40b392-Paper-Conference.pdf.

M. E. Tipping and C. M. Bishop. Mixtures of probabilistic principal component analysers. *Neural Computation*, 11(2):443–482, 1999.

M. Turk and A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1):71–86, 1991.

Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges, 2019. URL https://arxiv.org/abs/1812.01717.

Théophane Vallaeys, Jakob Verbeek, and Matthieu Cord. Ssdd: Single-step diffusion decoder for efficient image tokenization, 2026. URL https://arxiv.org/abs/2510.04961.

Aaron van den Oord, Oriol Vinyals, and koray kavukcuoglu. Neural discrete representation learning. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/7a98af17e63a0ac09ce2e96d03992fbc-Paper.pdf.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models, 2025. URL https://arxiv.org/abs/2503.20314.

Shuai Wang, Zhi Tian, Weilin Huang, and Limin Wang. Ddt: Decoupled diffusion transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 40633–40642, June 2026.

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, Da Yin, Yuxuan Zhang, Weihan Wang, Yean Cheng, Xu Bin, Xiaotao Gu, Yuxiao Dong, and Jie Tang. Cogvideox: Text-to-video diffusion models with an expert transformer. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu, editors, *International Conference on Learning Representations*, volume 2025, pages 83048–83077, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ce31378e9f41d8907e97dab172b6c559-Paper-Conference.pdf.

Jingfeng Yao, Bin Yang, and Xinggang Wang. Reconstruction vs. generation: Taming optimization dilemma in latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15703–15712, June 2025.

Lijun Yu, Jose Lezama, Nitesh Bharadwaj Gundavarapu, Luca Versari, Kihyuk Sohn, David Minnen, Yong Cheng, 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 *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=gzqrANCF4g.

Yongsheng Yu, Wei Xiong, Weili Nie, Yichen Sheng, Shiqiu Liu, and Jiebo Luo. Pixeldit: Pixel diffusion transformers for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14273–14282, June 2026.

Biao Zhang and Rico Sennrich. Root mean square layer normalization. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/1e8a19426224ca89e83cef47f1e7f53b-Paper.pdf.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

Long Zhao, Sanghyun Woo, Ziyu Wan, YANDONG LI, Han Zhang, Boqing Gong, Hartwig Adam, Xuhui Jia, and Ting Liu. Epsilon-VAE: Denoising as visual decoding. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=oK4ot2wAJU.

Boyang Zheng, Nanye Ma, Shengbang Tong, and Saining Xie. Diffusion transformers with representation autoencoders. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust, editors, *International Conference on Learning Representations*, volume 2026, pages 35791–35820, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/3c4141c12660ad3625eb4ae845e0a6f9-Paper-Conference.pdf.

A Implementation details

We base our implementation on a reproduction of the publicly available JiT codebase[4] (Li and He, 2026). All our configurations are available in Table S1. We apply the following modifications on top of JiT:

- **Register tokens**: similarly to how JiT uses in-context class tokens, we use register tokens in the model. On frames-to-video completion, they are a set of learnable embeddings that are concatenated to the input of the transformer blocks. On class-to-video generation, they are strictly equivalent to JiT in-context class tokens, as we additionally embed the class label into the register tokens.
- **Bottleneck**: we do not use the bottleneck in the input projection of JiT models, unless explicitly specified (preliminary and ablation experiments).
- **Training time sampler**: we use a heavy-tailed training time sampler for pixel-space diffusion, as a mixture of uniform and logit-normal distributions.
- **Noise scaling**: we sweep over noise scaling factors of the form 2^k for each video resolution and length, and for each individual tokenizer, as we find that the optimal scaling factor varies greatly across both tokenizers, and pixel-space resolutions.
- **Attention implementation**: we use FlashAttention (Dao et al., 2022) for all attention layers to support the large token count of video models.

Table S1: **Configurations of experiments.**

	JiT-B	JiT-L	JiT-XL
architecture			
depth	12	24	28
hidden dim	768	1024	1152
heads	12	16	18
dropout		0	
register tokens		32	
registers start block	4	8	8
training			
epochs		600 (unless specified otherwise)	
warmup epochs		5	
optimizer		AdamW, $\beta_1, \beta_2 = 0.9, 0.95$	
batch size		128	
learning rate		2e-4 (JiT, U-JiT), 4e-4 (l-JiT)	
learning rate schedule		constant	
weight decay		0.01	
EMA decay		0.9999	
time sampler		$\begin{cases} t \sim \mathcal{U}(0,1) & \text{w.p. } p_l \\ \text{logit}(t) \sim \mathcal{N}(\mu_l, \sigma_l^2) & \text{w.p. } 1-p_l \end{cases}$	
time mean		$\mu_l = -0.8$ (JiT, U-JiT), $\mu_l = 0$ (l-JiT)	
time variance		$\sigma_l = 0.8$ (JiT, U-JiT), $\sigma_l = 1$ (l-JiT)	
time heavy tail		$p_l = 0.1$ (JiT, U-JiT), $p_l = 0$ (l-JiT)	
clip of $(1-t)$ in division		0.05	
class token drop (for CFG)		0.1	
LPIPS loss weight		0.25 (U-JiT), 0 (JiT, l-JiT)	
sampling			
ODE solver		Heun (Heun et al., 1900)	
ODE steps		$N = 50$	
time steps		$t_i = \frac{i}{i+\sigma_t(N-i)}$	
CFG scale sweep step		0.5	
CFG interval		$[0.1, 1]$	

[4] https://github.com/LTH14/JiT

U-JiT blocks parametrization. The U-JiT encoding and decoding blocks are parametrized by their patch size and hidden dimension, with additional attention patchification (patch and dimension) and tiling parameters. The dimensions and number of those blocks are adjusted per setting (video resolution, length, main level patch size, and model size) to keep a similar or lower computational cost as matching JiT models. We provide in Table S2 the parametrization of each level of the U-JiT models for each setting used in our experiments, together with the number of parameters and FLOPs compared to JiT. Main blocks follow the same parametrization as JiT (Table S1).

Table S2: **U-JiT blocks parametrization.** Per-setting configuration of the U-JiT multi-level blocks across video resolutions, lengths, tokenizers, and model sizes, together with parameter counts and per forward-pass FLOPs compared to JiT.

Setting				
Video resolution	128×128		256×256	
Video length	17 frames		17 frames	65 frames
Tokenizer patch size	f8t4 ($8 \times 8 \times 4$)		f32t8 ($32 \times 32 \times 8$)	
Model size	B	XL	XL	XL
Pixel-space noise scaling	2		2	4
First level				
Patch size	$1 \times 1 \times 1$		$2 \times 2 \times 1$	
Depth	2 (encoding) + 2 (decoding)			
Hidden dim	32		24	
Attention patchification	$4 \times 4 \times 1$		$8 \times 8 \times 2$	
Attention dim	128	384	128	64
Attention tiling	$8 \times 8 \times 4$			
Second level				
Patch size	$8 \times 8 \times 1$		$8 \times 8 \times 2$	
Depth	2 (encoding) + 2 (decoding)			
Hidden dim	256	384	128	
Attention tiling	$8 \times 8 \times 4$			
Main level				
Depth	8	20	20	
Efficiency				
JiT parameters	124.8M	644.5M	700.0M	706.5M
U-JiT parameters	98.2M	587.5M	478.1M	478.2M
JiT FLOPs	266.3G	1.3T	199.1G	596.1G
U-JiT FLOPs	264.8G	1.3T	195.2G	575.2G

First frame embedding. A standard practice in state-of-the-art video tokenizers (NVIDIA et al., 2025; HaCohen et al., 2024; Wan et al., 2025) is to embed the first frame of the video independently, and use temporal patchification on the remaining frames. Through the video tokenizers, our latent models implicitly use this single-frame embedding. We follow this practice for our pixel-space models: for U-JiT, each spatiotemporal linear downsampling and upsampling projection (as input, output and between each level) is decomposed as a spatial projection followed by a temporal projection. On the first frame, the temporal projections are skipped. For JiT, we use a separate linear projection for the first frame, and a spatiotemporal projection for the remaining frames.

Dataset. We train on the Kinetics-600 dataset (Carreira et al., 2018), subsampling each video to 8 FPS, resizing the shorter side to the target resolution, and using a center crop. During training, we apply data augmentation by randomly flipping the video horizontally with probability 0.5. We filter out videos shorter than the target length, depending on the task.

Evaluation. Following common practices, we evaluate FVD (Unterthiner et al., 2019) for video quality and FID (Heusel et al., 2017) for per-frame quality. For the FVD evaluation, we use the pytorch adaptation[5] of I3D models trained on the Kinetics dataset (Carreira and Zisserman, 2017). FID uses the commonly-used torch-fidelity implementation[6]. Evaluations are conducted on the eval set for frames-to-video completion, and on the training set for class-to-video generation. We use

[5] https://github.com/piergiaj/pytorch-i3d

[6] https://github.com/toshas/torch-fidelity

the first 16 frames of the output (conditioning + generated frames) for FVD evaluation, and the last generated frame for FID. Both metrics are evaluated on 50k samples. FVD@5k and FID@5k indicate evaluation on 5k samples instead.

B Additional results and analysis

Sampling latent-space models. In Section 3.4 we consider sampling methods to fruther bridge the gap between latent-space and pixel-space flow-matching models. While we observe in Section 3.3 that latent decoders focus the generative process on lower frequencies and push the generated high frequencies on-manifold through adversarial and perceptual losses, it is not clear whether their action is sufficient for faithful video generation. We therefore apply both sampling methods (classifier-free guidance and time-shifting) to latent-space models. In Table S3, we study their effect on l-JiT-B (Wan2.2 Tokenizer), and observe an improvement on both the FID and FVD metrics when combining the two methods. The improvement is smaller than the one observed on pixel-space models: FVD and FID improve respectively by 5% and 28% when applied to latents, compared to 10% and 39% for pixel space. In our main experiments we therefore report results using the best time-shifting and classifier-free guidance settings for latent-space l-JiT models.

Table S3: **l-JiT-B (Wan2.2 Tokenizer) latent sampling sweep on Kinetics-600 (epoch 200), on 5-to-11 frames video generation.** FVD@5k and FID@5k over σ_t (time shift) and CFG. Lower is better; best per metric in **bold**.

		CFG					
	σ_t	1	1.5	2	2.5	3	3.5
FVD@5k ↓	1	3.17	3.02	3.11	3.20	3.25	3.27
	2	3.13	3.02	3.12	3.21	3.26	3.28
	4	3.08	**3.01**	3.13	3.22	3.28	3.29
	8	3.05	3.01	3.15	3.22	3.29	3.34
	16	3.13	3.10	3.25	3.39	3.58	3.82
FID@5k ↓	1	19.1	15.5	14.7	14.4	14.4	14.4
	2	18.5	15.1	14.3	14.1	14.0	14.1
	4	17.9	14.8	14.0	**13.8**	**13.8**	13.9
	8	18.3	15.2	14.5	14.4	14.7	15.3
	16	21.6	17.2	16.9	18.3	20.8	24.6

Architectural and sampling ablations. To study the impact of each component of our method, we perform a series of architectural and sampling-time ablations on our model. We train variants of the U-JiT-L model on the 1-to-15 frame prediction task, with a spatiotemporal compression rate of f32t8, as in Table 3, and report results in Table S4. Line (1) is our default configuration used in our main experiments.

First, when we compare our main model to line 2 we observe that LPIPS loss contributes *significantly* to per-frame quality (FID), while also improving the overall video quality (FVD).

Secondly, we analyze the impact of several of our architectural design choices. Attention tiling in the encoding and decoding blocks (see line 3) is *necessary* to keep the model efficient and to prevent the attention cost from exploding and more than doubling the total number of FLOPs. Enabling it only slightly degrades both metrics. Attention patchification (line 4) provides a 5% improvement in FVD while degrading FID by 5%, and slightly reduces the computational cost: while it slightly improves our model overall, this component is *dispensable*. Next, we remove either one (line 5) or both (line 6) of the encoding-decoding levels of our U-JiT architecture, keeping the overall number of layers the same by increasing the main level depth. Removing the pixel-level (first level) layers (line 5) only causes a small degradation of the metrics. On the other hand, keeping only the main level (line 6), which results in a JiT-L model, deteriorates generation quality significantly. We additionally validate our choice of removing the input bottleneck (line 7) from JiT, as it further degrades both metrics on video generation. We conclude that the progressive compression from pixel space to the internal representation is *central* to our method, while the exact number of levels and patch sizes are hyperparameters with lower impact. These ablations validate that our U-shaped design is key to

Table S4: **Architectural and sampling ablations.** Variants of U-JiT-L trained for 400 epochs on 1 conditioning frame to 15 evaluated frames, at resolution 256×256 with spatiotemporal compression f32t8. The first block ablates the loss, the second the architectural choices (attention adaptations, patchification levels and network structure), and the third the sampling-time methods. FLOPs are computed for a single forward pass of the model.

	Model (f32t8)	FLOPs		FVD@5k		FID@5k	
		Default configuration					
(1)	U-JiT-L (ours)	137G		5.35		19.20	
		Loss ablations					
(2)	w/o LPIPS	137G	–	6.33	***+0.98***	35.47	***+16.27***
		Architecture ablations					
(3)	w/o attention tiling	358G	***+221G***	5.32	*−0.03*	19.13	*−0.07*
(4)	w/o attention patchification	149G	***+12G***	5.63	*+0.28*	18.38	*−0.82*
(5)	w/ 2 U-JiT levels only	130G	*−7G*	5.44	*+0.09*	19.65	*+0.45*
(6)	w/ main level only (JiT-L)	137G	–	8.98	***+3.63***	43.33	***+24.13***
(7)	w/ main level only and bottleneck	132G	*−5G*	10.08	***+4.73***	42.37	***+23.17***
(8)	w/o residual connections	137G	–	6.09	***+0.74***	20.01	*+0.81*
(9)	w/o U-JiT blocks	131G	*−6G*	7.47	***+2.12***	23.33	***+4.13***
		Sampling ablations					
(10)	w/o time-shift	137G	–	5.49	*+0.14*	27.59	***+8.39***
(11)	w/o CFG	137G	–	5.86	***+0.51***	21.09	***+1.89***
(12)	w/o time-shift and w/o CFG	137G	–	6.43	***+1.08***	34.74	***+15.54***

our method, while attention adaptations mostly focus on enabling efficient implementation of this architecture.

In lines 8 and 9 study the impact of the two components of the U-shaped architecture: residual connections, and progressive patchification. In line 8 we keep the varying patch size across blocks without the residuals, while in line 9 we use a plain single-resolution transformer (as in JiT), but adding the residual connections. We observe that while both components contribute to improving FVD and FID metrics, the progressive patchification and unpatchification of U-JiT blocks is the most significant one.

Finally, we study sampling variants in this setting, complementing the results from Section 3.4. While both classifier-free guidance and time-shifting impact FVD and FID, removing the time shift mostly has most impact on the per-frame quality (FID), while removing CFG bast most impact on the overall video quality (FVD). Removing both leads to the largest degradations in both metrics, suggesting that time0shifting and CFG act as complementary mechanisms.