NeRV-Diffusion: Diffuse Implicit Neural Representations for Video Synthesis

Anonymous authors
Paper under double-blind review

Abstract

We present NeRV-Diffusion, an implicit latent video diffusion model that synthesizes videos via generating neural network weights. The generated weights can be rearranged as the parameters of a convolutional neural network, which forms an implicit neural representation (INR), and decodes into videos with frame indices as the input. Our framework consists of two stages: 1) A hypernetwork-based tokenizer that encodes raw videos from pixel space to neural parameter space, where the bottleneck latent serves as INR weights to decode. 2) An implicit diffusion transformer that denoises on the latent INR weights. In contrast to traditional video tokenizers that encode videos into frame-wise feature maps, NeRV-Diffusion compresses and generates a video holistically as a unified neural network. This enables efficient and high-quality video synthesis via obviating temporal cross-frame attentions in the denoiser and decoding video latent with dedicated decoders. To achieve Gaussian-distributed INR weights with high expressiveness, we reuse the bottleneck latent across all NeRV layers, as well as reform its weight assignment, upsampling connection and input coordinates. We also introduce SNR-adaptive loss weighting and scheduled sampling for effective training of the implicit diffusion model. NeRV-Diffusion reaches superior video generation quality over previous INR-based models and comparable performance to most recent state-of-the-art non-implicit models on real-world video benchmarks including UCF-101 and Kinetics-600. It also brings a smooth INR weight space that facilitates seamless interpolations between frames or videos.

1 Introduction

Video latent diffusion models (LDMs) have achieved impressive generative capability. However, their tokenizers usually inherit from those of image diffusion models and encode videos as individual frame-wise feature maps, ignoring the natural coherence across frames and resulting in redundant representations. Cross-frame attentions (Wang et al., 2023; Guo et al., 2023) are thus introduced to constrain temporal consistency in both generation and decoding processes, largely increasing the model size and leading to massive computation footprint. Moreover, a typical tokenizer for diffusion (Rombach et al., 2022) is usually built in the form of a large-scale variational autoencoder (VAE), which compresses the visual data into latent code with generalizable decoding quality on diverse data. During inference, the denoised latent must be processed by the decoder to be rendered into pixels, demanding high computation for visualization efficiency.

Implicit neural representations (INRs) are neural networks that fit on single data points. An INR takes unified coordinates as the input and outputs pixels as stored in its model weights. It has shown significant advantages on compression (Sitzmann et al., 2020; Dupont et al., 2021), fast decoding (Chen et al., 2021a), and easy transformation (Mildenhall et al., 2021; Kerbl et al., 2023) by representing data as an integral format of function. The continuity and differentiability of INRs facilitate advanced single-data generative tasks, such as super-resolution, restoration, style transfer and editing, via smooth interpolations within the data space. Its compact representation also contributes to reducing memory overhead, making them highly suitable in resource-constrained environments.

To harness the strengths of both latent generative models and implicit neural representations, we establish an implicit latent diffusion model, NeRV-Diffusion, for video synthesis by generating INR weights, where a video is represented as a holistic set of INR weights. It consists of two stages:

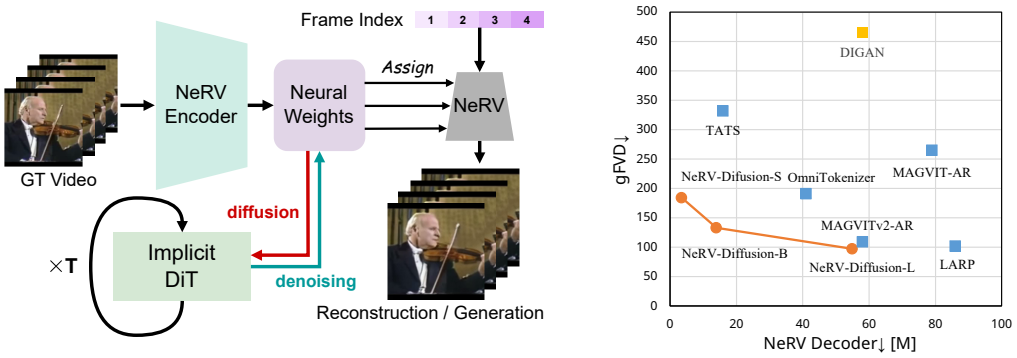


Figure 1: **Left:** Overview of our NeRV-Diffusion framework. In the tokenization stage, NeRV encoder projects RGB videos to neural network weights, forming up NeRVs and decoding for reconstruction. In the generation stage, an implicit diffusion transformer is trained to denoise on NeRV weights. During inference, the implicit DiT generates NeRV weights from random noise, which decode into RGB videos. **Right:** NeRV-Diffusion (orange) outperforms previous INR-based (yellow) as well as most recent non-implicit (blue) video generation models at all scales with more compact model sizes. The generative performance is evaluated in gFVD on UCF.

In the tokenization stage, a hypernetwork-based encoder compresses RGB videos into parametric latent tokens. The tokens instantiate an INR to decode for reconstruction with unified frame indices input. In the generation stage, a diffusion transformer denoises in the encoded implicit latent space, mapping random noise to INR weight tokens. Figure 1 (left) overviews the framework.

However, it is not trivial to acquire Gaussian-distributed neural network weights for smooth diffusion that are meanwhile able to represent diverse realistic data with high fidelity. We adopt a convolutional video INR, NeRV (Chen et al., 2021a), and build a transformer INR encoder based on FastNeRV (Chen et al., 2024a). They are originally designed toward video compression performance only and their produced INR weights are not generatable. To ensure the bottleneck latent tokens fitful for both faithful reconstruction and smooth diffusive generation, we have made several critical architectural modifications. The detailed architectures are illustrated in Figure 2.

Specifically, we reuse the encoded weight tokens with multiple linear affine layers such that each NeRV layer is modulated by all tokens independently. We also redesign the weight modulation approach, proposing to directly set the latent tokens to be the convolution kernels, instead of repeating and multiplying them with shared base weights. These upgrades fundamentally enlarge the expressiveness and smoothness of the implicit space while maintaining its compactness. We leverage vanilla diffusion transformer (Peebles & Xie, 2023) (DiT) to denoise on weight tokens that imply no spatial or temporal structures. We also handle the error accumulation with SNR-adaptive loss weighting and scheduled sampling for optimal denoising in the implicit latent space.

NeRV-Diffusion leverages video INRs as instance-specific decoders, offering faithful reconstruction, compact model and fast decoding compared to the large, shared decoders in traditional LDMs. It encodes and generates video frames holistically as integral INR weights, implies the keyframe-residue representation by reusing the same set of parameters to decode all frames, and thus maintains temporal associations without cross-frame attention. Furthermore, NeRV-Diffusion generates neural weights with only a single linear layer after the Gaussian bottleneck, and employs direct channel-wise parameterization to construct the INRs. This leads to multi-variant normal distribution of our generative NeRV weights and enables smooth interpolation between frames and videos.

In summary, our contributions are as follows:

- We propose a novel implicit video autoencoder that compresses videos into neural weight tokens of normal distribution, constituting generation-specialized video INRs.
- We propose an implicit diffusion model that denoises in neural weight space, achieving dynamic and diverse video synthesis via generating INR parameters.

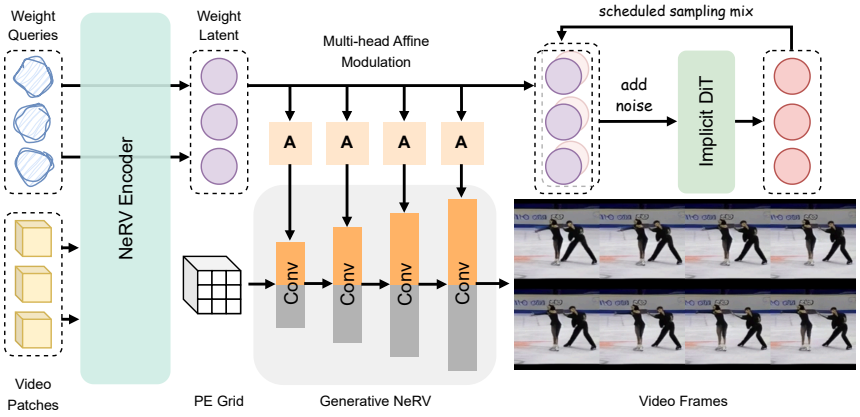


Figure 2: Detailed architectures of NeRV-Diffusion. **Left**: Patchified videos and initialized weight queries are concatenated and input into NeRV encoder, outputting latent weight tokens; **Middle top:** Weight tokens are reused and converted by multi-head affines to instantiate each generative NeRV layer; **Middle bottom:** Generative NeRV decodes spatiotemporal positional embeddings into RGB videos, using the instance-specific modulation weights (gold) and global shared weights (gray). Block details and side connections are omitted; **Right:** Weight tokens are added noise and an implicit diffusion transformer is trained to denoise in this implicit weight space.

- NeRV-Diffusion surpasses prior implicit and most recent non-implicit generative models on multiple real-world video benchmarks, and conveys smooth time and weight interpolations.

2 Related Work

2.1 Implicit Neural Representations

Implicit neural representations (INRs) are neural networks that fit on single data points. An INR takes in coordinates and outputs corresponding pixel values of the stored data. It has presented capacity and flexibility in various modalities, including images (Sitzmann et al., 2020; Dupont et al., 2021), 3D shapes (Park et al., 2019; Mildenhall et al., 2021) and videos (Chen et al., 2021a; 2022). They are primarily developed for image compression (Strümpler et al., 2022; Dupont et al., 2022) and editing (Fan et al., 2022; Yang et al., 2023), video compression (Li et al., 2022; Kwan et al., 2024; Zhao et al., 2023; Zhang et al., 2021; Lee et al., 2023) and editing (Ouyang et al., 2024), and novel view rendering (Kerbl et al., 2023; Barron et al., 2023; Cao & Johnson, 2023) and 3D scene editing (Yuan et al., 2022; Liu et al., 2024). Although some editing applications have been explored, they create the INRs after manipulating the data in pixel space.

A standard INR is trained via memorizing the pixel data, which is time-consuming in a backpropagation manner. Chen & Wang (2022); Kim et al. (2023a); Chen et al. (2024a) suggest using transformer-based hypernetworks to create INR weights given RGB data in a feed-forward fashion at scale. However, these methods are optimized solely toward reconstruction performance and incorporate no distribution regularization on the produced INR weights, leaving the implicit generative task that synthesizes novel data points from random noise under-addressed.

2.2 Implicit Neural Representation Generation

INR generation is a challenging task. Traditional generative models learn mapping random noise to pixels or latent features, while implicit generative models aim to associate neural parameters with Gaussian distribution. Several efforts have been made toward implicit generation. Skorokhodov et al. (2021) builds a GAN for image INRs (Sitzmann et al., 2020), and Yu et al. (2022) extends it to videos by involving the temporal axis. Erkoç et al. (2023); Chen et al. (2023); Müller et al. (2023);

Shue et al. (2023) study generating 3D NeRF parameters via diffusion models. (Chen et al., 2024b) applies latent diffusion models on image INRs (Chen et al., 2021b) for image synthesis, while their INR weights are derived by a complex decoder from the denoising latent space. Recently, Wang et al. (2024b; 2025b) propose to leverage the hypernetwork-INR architecture to conduct flow matching on image or 3D pixel data. Lee et al. (2025) also developed a masked image autoencoder for inpainting with a similar structure. Despite these efforts, no video diffusion model that generates INR weights has yet been explored, casting this a challenging task as videos embed more dynamic information and diffusion models have a more strict demand on its denoising space.

2.3 Latent Video Diffusion Models

Latent video diffusion models (Wang et al., 2023; Blattmann et al., 2023b; Guo et al., 2023) have achieved significant success in video generative modeling. However, traditional video tokenizers often encode video frames as individual feature maps, calling cross-frame attentions in the denoising network to constrain temporal consistency. Kim et al. (2023b); Wu et al. (2025) start to explore video autoencoders with motion awareness and temporal compression, splitting the complexity between the tokenization and generation stages. Recent 1D tokenization (Yu et al., 2024b; Wang et al., 2025a; Zha et al., 2025) encodes visual data into holistic tokens that project no spatial or temporal alignment with pixels, while they remain focused on images or auto-regressive generation only. In this work, we look to synthesize videos by generating INR weights via diffusion, obviating frame-wise representations by using the whole INR model to decode all frames given time indices. Moreover, symmetric autoencoders rely on a large-scale decoder to render synthesized latent to diverse RGB data with high fidelity, consuming non-negligible computational resources and time for end users to visualize. We explore the space of asymmetric hypernetwork-INR autoencoders, where the INR acts as an efficient instance-specific decoder as it only needs to represent a single data point.

3 NeRV Diffusion

NeRV-Diffusion is a two-stage generative framework. In the tokenization stage, an implicit autoencoder (§3.1) is trained to compress a video from pixels to latent neural weight tokens, and the tokens function as the parameters of an INR (§3.2) and self-decode to reconstruct the video. In the generation stage, an implicit diffusion transformer (§3.3) is trained to generate the weight tokens from random noise. Figures 1 (left) and 2 illustrate our full pipeline of both stages.

3.1 NeRV Autoencoder

In the first stage, we aim to tokenize a video into a latent space that represents the video through the parameters of an INR. This is achieved by training an implicit autoencoder, where the encoder $\mathcal{E}$ is a hypernetwork that produces INR parameters $\theta = \mathcal{E}(x)$ given pixel input x. The decoder is implemented as an INR $\mathcal{D}_\theta(\cdot)$, which decodes to pixel values given corresponding coordinates. We build the backbone of our INR encoder $\mathcal{E}$ upon ViT-based FastNeRV (Chen et al., 2024a), where we make several critical modifications to align the learned latent space with generative tasks.

The RGB video is first segmented into patches and converted to transformer input embeddings. Since the output weight tokens have no spatiotemporal correspondence to the input patches, instead of mapping them directly we introduce dedicated query tokens and concatenate them with the data patches following (Peebles et al., 2022). Only the output tokens corresponding to the queries are retained. They are batch normalized along the token embedding dimension.

KL Bottleneck. Two additional fully connected (FC) layers are appended after the NeRV encoder's output to create an information bottleneck of compact latent dimension. KL divergence loss is applied to align their distribution toward standard Gaussian distribution $\mathcal{N}(0, 1)$.

Multi-head Affine Mapping. FastNeRV use its encoded latent to modulate the parameters of a subset of the INR layers, which limits the capacity of the latent tokens especially when KL constraint is applied for generation tasks. Inspired by the multiple affine layers in Karras et al. (2019), we expand the post-bottleneck FC layer into multi-head affine mappings, and the single set of weight tokens are reused to modulate all NeRV layers independently. Specifically, for each NeRV layer, a

Figure 3: Video reconstruction of our NeRV autoencoder on UCF (left) and K600 (right).

dedicated affine head maps all the weight tokens into modulation parameters. This strategy significantly expands the expressiveness of the weight tokens, as a compact latent space will reduce the complexity of the diffusion process in the generation stage.

Channel-wise INR Parameterization. FastNeRV repeats the weight tokens and multiply them to the instance-agnostic INR base weights via dot product as the modulation. Skorokhodov et al. (2021); Yu et al. (2022) perform low-rank vector cross product to amplify the modulation matrix dimension from condensed weight latent. Inspired by Lin et al. (2021) that prunes a pretrained GAN generator by subsetting its kernels, we propose to directly set affined instance-specific weight tokens to be the convolutional kernels at a certain group of INR channels. Other parameters θ_s are shared among all training data and are learnable during training. All kernel values are normalized along all dimensions except the output channels, following the demodulation in Karras et al. (2019). In this way, the generated weight tokens are directly involved in decoding with maximal degrees of freedom. This also enables smooth parameter interpolation between our INR decoders.

Convolutional Discriminator. To generate realistic videos we incorporate adversarial training (Goodfellow et al., 2020). We choose a convolutional discriminator (Karras et al., 2019) over a transformer-based one, as we observe that the latter introduces flickering artifacts across frames.

Training Objectives. We train NeRV-VAE with the reconstruction objective. With an additional perceptual loss (Zhang et al., 2018) $\mathcal{L}_{\text{LPIPS}}$ and the adversarial loss $\mathcal{L}_{\text{GAN}}$, it is optimized via

$$\mathcal{L}_{\text{VAE}}(\mathcal{E}, \theta_s) = \|x - \tilde{x}\|^2 + \mathcal{L}_{\text{LPIPS}}(x, \tilde{x}) + \mathcal{L}_{\text{GAN}}(x, \tilde{x}) + D_{\text{KL}}(\mathcal{N}(0, 1), \tilde{\theta}). \tag{1}$$

3.2 Generative NeRV

The encoded weight tokens are formed into a video INR $\mathcal{D}_\theta(\cdot)$ that decodes to reconstruct the video. NeRV (Chen et al., 2021a) is a convolutional video INR that takes time index t as the input query and yields a whole frame at each forwarding. We construct our implicit decoder based on it while introducing several upgrades to enhance its capacity for generative purposes.

Spatiotemporal Embedding Input. Time-query video INRs upsamples from $\mathbb{R}^{T\times D\times 1\times 1}$ to $\mathbb{R}^{T\times 3\times H\times W}$, where no spatial dimension is input. With this structure, we observe distinct movement in the reconstruction, however the spatial content lacks clarity. To balance between the appearance and motion quality, we expand the input time embedding to 3D spatiotemporal, while time remains the sole query axis. Specifically, we sample a 3D positional embedding and reshape it to $\mathbb{R}^{T\times 3D\times h\times w}$. This spatiotemporal input supplements geometric prior and avoids the leading FC layer in vanilla NeRV that were designed for transforming 1D time embedding input. We observe that full convolutions fit optimally for generative quality with our multi-head affine modulation.

Scaling up Blocks. Benefited from the reused weight modulation with multi-head affine mappings, we are able to largely scale up our generative NeRV without extra weight tokens. We expand the upsampling layers to blocks, each performing one-level ($2\times$) upsampling with additional convolutions that don't change the shape. Compared to the assorted upsampling scales in limited layers in vanilla NeRV, this periodic upsampling structure evenly distributes the information from low to high resolutions, and cooperates well with our multi-head affine modulation. We also double the

Table 1: Model and bottleneck representation size comparison. rFVD and gFVD are results on UCF. † Note that for implicit GANs we conceptually separate the mapping network as the generator and the generator network as the decoder, as the latter takes in the frame index and decodes as the INR.

Method	#Params		#Tokens	rFVD↓		gFVD↓	
	Detokenizer	Generator		UCF	K600	UCF	K600
Non-Implicit Models							
CogVideo (Hong et al., 2022)	-	9.4B	-	-	-	626	109
TATS (Ge et al., 2022)	16M	362M	1024	162	-	332	-
MAGVIT-AR (Yu et al., 2023a)	79M	306M	1024	25	-	265	-
Latte (Ma et al., 2024)	49M	674M	512	21	-	202	-
OmniTokenizer (Wang et al., 2024a)	41M	650M	1280	42	-	191	33
VideoFusion (Luo et al., 2023)	-	2B	-	-	-	173	-
MAGVITv2-AR (Yu et al., 2024a)	58M	840M	1280	**8.6**	-	109	-
LARP (Wang et al., 2025a)	86M	343M	1024	<u>20</u>	-	<u>102</u>	**6.2**
Implicit Models							
DIGAN (Yu et al., 2022)	58M†	5.5M†	-	-	-	465	-
NeRV-Diffusion-S (Ours)	**3.5M**	467M	128	85	40	184	46
NeRV-Diffusion-B (Ours)	<u>14</u>M	467M	128	59	27	133	30
NeRV-Diffusion-L (Ours)	55M	467M	128	41	19	**97**	<u>22</u>

hidden dimensions of the layers in the last block following Karras et al. (2020) so that more native high-resolution information can be processed with sufficient capacity.

Upsampling Algorithm. While vanilla NeRV has tested that pixelshuffle results in the best reconstruction performance with similar amount of parameters, we again compare different upsampling algorithms for our generative NeRV. We find that transposed convolutions achieve non-negligible better generation quality to pixelshuffle with merely a quarter of parameters and computations. Therefore we choose transposed convolutions for all the upsampling layers in our generative NeRV.

Side Connections. With the increased depth of our generative NeRV by upscaled blocks, we further append side connections to effectively collate all intermediate resolution information with minimal computation overhead. We investigate the residual and skip connections as in Karras et al. (2020). If the side connection needs additional layers, they are also modulated by the same set of our weight tokens thanks to our multi-head affine mappings and no extra trainable parameter is introduced. Residual connection fuses latent features at different scales before decoding to RGB and is experimented to yield clearer appearance and stabler motion.

3.3 Implicit Diffusion

With visual data tokenized from pixel space to NeRV weight space by the implicit autoencoder described above, we perform diffusion process on these weight tokens by $\theta_t = \alpha_t\theta_0 + \sigma_t\epsilon$ and train a denoising network ϕ toward

$$\mathcal{L}_{\text{IDM}} = E_{\theta,\epsilon\sim\mathcal{N}(0,1),t}[\|\epsilon_0 - \epsilon(\epsilon_t, t)\|^2] \tag{2}$$

It is not trivial to model denoising process on neural weights. Previous diffusion models are designed for pixel data or their latent feature maps. Since NeRV weight tokens have no spatiotemporal structure, transformers are more suitable than U-Nets to process them, and temporal attention is unnecessary in our denoising network like those in traditional video diffusion models (Ma et al., 2024). G.pt (Peebles et al., 2022) uses transformers to evolve neural network weights in a meta-learning fashion but not on noisy data. DiT (Peebles & Xie, 2023) tailors transformers for image diffusion and Zha et al. (2025) also use it to process 1D image tokens in diffusion. We explored these backbone options and DiT reaches the optimal performance with a straightforward architecture. Besides, we curate the training scheme as below to fill the gap when adapting DiT to the implicit space.

Min-SNR-γ Loss Weighting. We observe that our implicit diffusion model converge slower on early denoising timesteps than late ones, i.e. it is tougher to learn to parse more noisy input. To

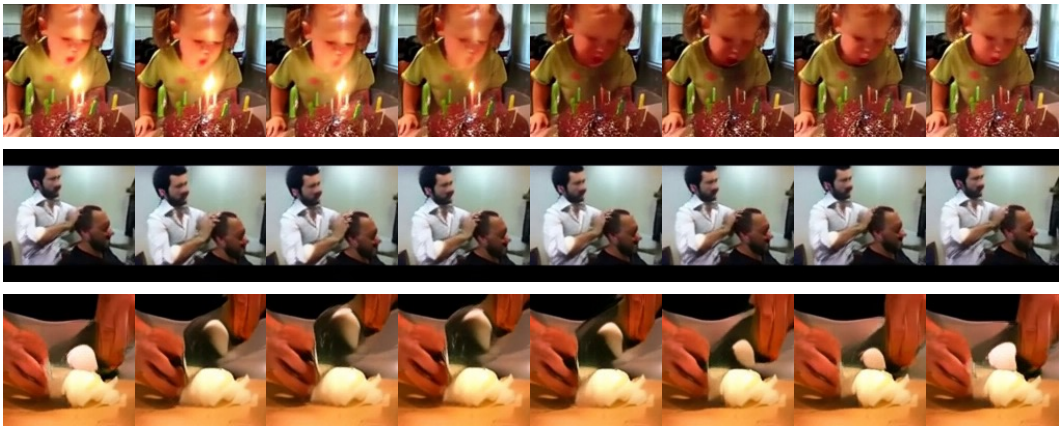


Figure 4: Class-conditioned video generation on UCF.

address this issue and speed up its convergence, we adopt Min-SNR-γ loss weighting (Hang et al., 2023) and apply the coefficient $w_t = \min\{\text{SNR}(t), \gamma\}$ on the denoising loss, where $\text{SNR}(t) = \frac{\alpha_t^2}{\sigma_t^2}$ reflects the signal-noise ratio at timestep t, and constant γ controls the minimum of w_t. This loss weighting strategy prevent the diffusion training to focus too much on the low noise levels and descends evenly toward the denoising directions at all timesteps.

Scheduled Sampling. To further enhance the implicit denoising chain and tackle the exposure bias issue, we introduce scheduled sampling (Bengio et al., 2015) into our training scheme. It is initially proposed for auto-regressive models, and has been applied on diffusion models (Ning et al., 2023; Ren et al., 2024) to fill the training-inference gap brought by Teacher Forcing. During training, after the first forward round at step t, we randomly use the model predictions $\tilde{\theta_{t-1}} = \theta_\phi(\theta_t, t)$ as the new input and execute another forward pass, and calculate the total losses. It aligns the the training and inference modes, ensures low input disparity and minimizes error accumulation during sampling.

4 Experiments

4.1 Setups

Datasets. We demonstrate NeRV-Diffusion on two real-world video benchmarks: video generation on UCF-101 (Soomro et al., 2012) (UCF) and frame prediction on Kinetics-600 (Kay et al., 2017) (K600). All experiments are conducted on 16 frames of 128^2 resolution. We use the train split of K600, and all videos from UCF, following prior work (Yu et al., 2023a; Wang et al., 2025a).

Implementations. We realize our NeRV encoder with the backbone of Vision Transformer (Dosovitskiy et al., 2021) (ViT). We scale up our generative NeRV decoder to three configurations of progressive sizes: -Small (3.5M), -Base (14M) and -Large (55M). We ablate our key design options in §4.4. Detailed model and training configurations are provided in Appendix A.

Metrics. We measure Fréchet Video Distance (FVD) (Unterthiner et al., 2018) to evaluate the reconstruction and generation quality of NeRV-Diffusion. We calculate FVD on 2,048 sampled videos following prior work (Yu et al., 2022; 2023a; Wang et al., 2025a) for fair comparison.

4.2 Video Reconstruction

Visualized reconstruction output of our implicit tokenizer are displayed in Figure 3. The quantitative results, together with the model and bottleneck latent size comparisons are listed in Table 1. NeRV-Diffusion achieves comparable performance to other non-implicit methods, with much more compact model and latent sizes. It also worth noting that NeRV-Diffusion features a small reconstruction-generation gap compared to other models, indicating our effective design of implicit video representations for generation purposes, and thus efficient usage of our latent space.

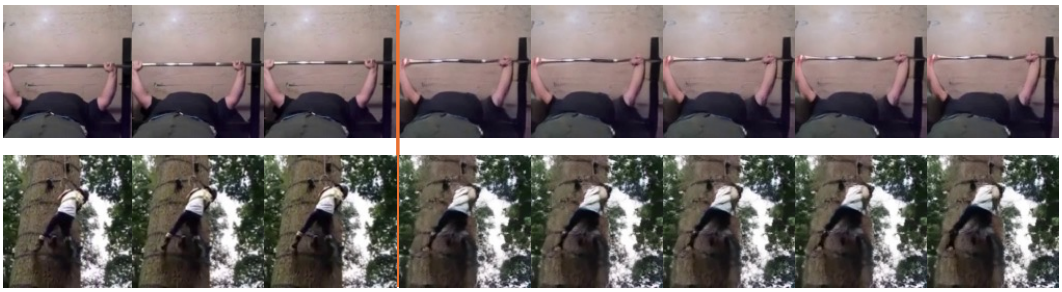


Figure 5: Frame prediction on K600. Frames in front of the orange line are input conditions.

4.3 Video Generation

Class-Conditioned Video Generation. We conduct class-conditioned video generation on UCF and present our visual results in Figure 4. We quantitatively compare NeRV-Diffusion with other models in Table 1. NeRV-Diffusion outperforms previous INR-based generative methods as well as most recent non-implicit models of various mechanisms, including GAN, diffusion and auto-regressive architectures. It is able to synthesize dynamic videos with diversity in both appearances and motions, ranging from detailed objects to complex scenes. More samples in Appendix D.

Frame Prediction. Following the settings in Hong et al. (2022), we train our implicit diffusion model given the initial 5 frames to predict the rest 11 frames. We construct a sequence of the 5 given frames and 11 duplicate 5th frames and encode them as a video clip to the NeRV weight space. We expand the input embedder of DiT by doubling its input channels to fuse the clean condition and noised ground truth. The quantitative results are listed in Table 1 and the visualizations are displayed in Figure 5. Our model faithfully propagates the spatial content and movement flows to future frames.

4.4 Ablation Studies

We conduct ablation studies to assess the key components we propose in §3 and validate the optimal design options for generative objectives. The quantitative results are tested with NeRV-Diffusion-S configuration on UCF and are listed in Tables 2 and 3.

Table 2a indicates that in our NeRV autoencoder, our channel-wise parameterization outperforms due to its maximal transparency to decode directly using the encoded weight tokens. In Table 2b, our multi-head affines significantly boost the capacity of NeRV by mapping the whole bottleneck weight tokens to different NeRV layers for reused modulation. Table 2c demonstrates that the spatiotemporal input embedding of shape $h = w = 8$ expands the input space with peak expressiveness, while smaller sizes lead to truncated space and bigger sizes result in fewer upsampling layers. We compare different upsampling operations in Table 2d, and find that transposed convolution surpasses pixelshuffle by much fewer parameters and computations without inflated channels. We further explore side connection types in Table 2e, and observe that residual connections fuse raw features at diverse scales without visible artifacts brought by skip connections when summing up multi-resolution RGB output. Finally we scale up our implicit latent space by increasing the number of tokens, as we meanwhile observe that the token dimension only makes slight impact on the output quality. 128 tokens reaches the peak performance and more tokens will lead to an over complex latent space for diffusion although the reconstruction error continues dropping.

For our implicit denoising network, we consider three backbone candidates in Table 3a. G.pt was designed for neural weight evolution, but not adapted to diffusion tasks. Latte was designed for video generation and incorporate with temporal attentions, which are not beneficial to our implicit generation as NeRV weight tokens lack spatiotemporal structure. Table 3b showcases that both Min-SNR-γ loss weighting and scheduled sampling scheme effectively minimize the gap between implicit diffusion training and inference, by emphasizing the denoising model more on high-noise predictions and imperfect input.

Table 2: Ablation studies of the key design options in our NeRV autoencoder and generative NeRV, tested with NeRV-Diffusion-S and the best implicit denoiser configuration on UCF.

Modulation	gFVD↓
Repeat	741
FMM	636
Channel	**570**

(a)

Reuse	gFVD↓
No reuse	570
Direct reuse	562
Multi-head affines	**283**

(b)

Spatial PE	gFVD↓
$h = w = 1$	283
$h = w = 4$	269
$h = w = 8$	**254**
$h = w = 16$	277

(c)

Upsampling	gFVD↓
PixelShuffle	254
Transposed Conv	**248**
Bilinear	287

(d)

Side Connection	gFVD↓
Vanilla	248
Residual	**219**
Skips	235

(e)

Token Shape	gFVD↓
32×128	219
64×128	193
128×128	**184**
256×128	206

(f)

Table 3: Ablation studies of the key design options in our implicit diffusion model, tested with NeRV-Diffusion-S and the best NeRV autoencoder configuration on UCF.

Model	gFVD↓
G.pt (Peebles et al., 2022)	550
DiT (Peebles & Xie, 2023)	**295**
Latte (Ma et al., 2024)	342

(a)

Configurations	gFVD↓
Vanilla DiT	295
w/ Min-SNR-γ	238
w/ Scheduled Sampling	261
w/ Both	**184**

(b)

4.5 Properties of Generative NeRV

4.5.1 Long Video Generation via Time Interpolation and Extrapolation

Benefited from the continuous frame index positional embedding, our generative NeRV features flexible time interpolation and extrapolation capability. In Figure A1, we interpolate the input time embeddings by a factor of $8\times$ to sample 128-frame videos with smooth and distinct motions. This property indicates that our generative NeRV efficiently encodes high-density information and understand the residual intrinsic of frame sequences. It enables compact representation of long videos and efficient training with fewer frames and large frame intervals.

4.5.2 Generative NeRV Weight Interpolation

Our generative NeRV also features smooth interpolation between two distinct videos by interpolating their instance parameters. Given two generative NeRVs' parameters θ_1 and θ_2, we perform linear interpolation $\lambda\theta_1 + (1 - \lambda)\theta_2$ between them. The visual outcomes are exhibited in Figure A2. Our model produces progressively interpretable results compared to DIGAN. We attribute this parametric continuity to not only the Gaussian distribution constraint of our weight latent, but also our simple yet effective linear bottleneck mapping and channel-wise parameterization.

5 Conclusion

We propose NeRV-Diffusion, a two-staged video synthesis model via NeRV weight generation. Our NeRV autoencoder projects videos into a Gaussian weight latent space for tokenization, where our implicit diffusion model denoises to generate neural weights that render into videos. NeRV-Diffusion outperforms both INR-based and most recent non-implicit video generative models on multiple real-world video benchmarks, demonstrating promising scaling law. It also features smooth temporal and parametric interpolation properties. The outstanding performance of NeRV-Diffusion highlights the potential of a new holistic video synthesis paradigm with efficient representations.

REFERENCES

Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19697–19705, 2023.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

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

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

Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 130–141, 2023.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

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*, pp. 6299–6308, 2017.

Hansheng Chen, Jiatao Gu, Anpei Chen, Wei Tian, Zhuowen Tu, Lingjie Liu, and Hao Su. Single-stage diffusion nerf: A unified approach to 3d generation and reconstruction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 2416–2425, 2023.

Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021a.

Hao Chen, Saining Xie, Ser-Nam Lim, and Abhinav Shrivastava. Fast encoding and decoding for implicit video representation. In *ECCV*, 2024a.

Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations. In *European Conference on Computer Vision*, pp. 170–187. Springer, 2022.

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8628–8638, 2021b.

Yinbo Chen, Oliver Wang, Richard Zhang, Eli Shechtman, Xiaolong Wang, and Michael Gharbi. Image neural field diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8007–8017, 2024b.

Zeyuan Chen, Yinbo Chen, Jingwen Liu, Xingqian Xu, Vidit Goel, Zhangyang Wang, Humphrey Shi, and Xiaolong Wang. Videoinr: Learning video implicit neural representation for continuous space-time super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2047–2057, 2022.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=YicbFdNTTy`.

Emilien Dupont, Adam Goliński, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. Coin: Compression with implicit neural representations. *arXiv preprint arXiv:2103.03123*, 2021.

Emilien Dupont, Hrushikesh Loya, Milad Alizadeh, Adam Goliński, Yee Whye Teh, and Arnaud Doucet. Coin++: Neural compression across modalities. *arXiv preprint arXiv:2201.12904*, 2022.

Ziya Erkoç, Fangchang Ma, Qi Shan, Matthias Nießner, and Angela Dai. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 14300–14310, 2023.

Zhiwen Fan, Yifan Jiang, Peihao Wang, Xinyu Gong, Dejia Xu, and Zhangyang Wang. Unified implicit neural stylization. In *European Conference on Computer Vision*, pp. 636–654. Springer, 2022.

Songwei Ge, Thomas Hayes, Harry Yang, Xi Yin, Guan Pang, David Jacobs, Jia-Bin Huang, and Devi Parikh. Long video generation with time-agnostic vqgan and time-sensitive transformer. In *European Conference on Computer Vision*, pp. 102–118. Springer, 2022.

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

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. *arXiv preprint arXiv:2307.04725*, 2023.

Yoav HaCohen, Nisan Chiprut, Benny Brazowski, Daniel Shalem, Dudu Moshe, Eitan Richardson, Eran Levin, Guy Shiran, Nir Zabari, Ori Gordon, et al. Ltx-video: Realtime video latent diffusion. *arXiv preprint arXiv:2501.00103*, 2024.

Tiankai Hang, Shuyang Gu, Chen Li, Jianmin Bao, Dong Chen, Han Hu, Xin Geng, and Baining Guo. Efficient diffusion training via min-snr weighting strategy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7441–7451, 2023.

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

Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21807–21818, 2024.

Huiwon Jang, Sihyun Yu, Jinwoo Shin, Pieter Abbeel, and Younggyo Seo. Efficient long video tokenization via coordinate-based patch reconstruction. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 22853–22863, 2025.

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

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 8110–8119, 2020.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023.

Chiheon Kim, Doyup Lee, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Generalizable implicit neural representations via instance pattern composers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11808–11817, 2023a.

Gyeongman Kim, Hajin Shim, Hyunsu Kim, Yunjey Choi, Junho Kim, and Eunho Yang. Diffusion video autoencoders: Toward temporally consistent face video editing via disentangled video encoding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6091–6100, 2023b.

Ho Man Kwan, Ge Gao, Fan Zhang, Andrew Gower, and David Bull. Hinerv: Video compression with hierarchical encoding-based neural representation. *Advances in Neural Information Processing Systems*, 36, 2024.

Joo Chan Lee, Daniel Rho, Jong Hwan Ko, and Eunbyung Park. Ffnerv: Flow-guided frame-wise neural representations for videos. In *Proceedings of the 31st ACM International Conference on Multimedia*, pp. 7859–7870, 2023.

Sua Lee, Joonhun Lee, and Myungjoo Kang. Minr: Implicit neural representations with masked image modelling. *arXiv preprint arXiv:2507.22404*, 2025.

Zizhang Li, Mengmeng Wang, Huaijin Pi, Kechun Xu, Jianbiao Mei, and Yong Liu. E-nerv: Expedite neural video representation with disentangled spatial-temporal context. In *European Conference on Computer Vision*, pp. 267–284. Springer, 2022.

Ji Lin, Richard Zhang, Frieder Ganz, Song Han, and Jun-Yan Zhu. Anycost gans for interactive image synthesis and editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14986–14996, 2021.

Xiangyue Liu, Han Xue, Kunming Luo, Ping Tan, and Li Yi. Genn2n: Generative nerf2nerf translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5105–5114, 2024.

I Loshchilov. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Zhengxiong Luo, Dayou Chen, Yingya Zhang, Yan Huang, Liang Wang, Yujun Shen, Deli Zhao, Jingren Zhou, and Tieniu Tan. Videofusion: Decomposed diffusion models for high-quality video generation. *arXiv preprint arXiv:2303.08320*, 2023.

Zhuoyan Luo, Fengyuan Shi, Yixiao Ge, Yujiu Yang, Limin Wang, and Ying Shan. Open-magvit2: An open-source project toward democratizing auto-regressive visual generation. *arXiv preprint arXiv:2409.04410*, 2024.

Xin Ma, Yaohui Wang, Gengyun Jia, Xinyuan Chen, Ziwei Liu, Yuan-Fang Li, Cunjian Chen, and Yu Qiao. Latte: Latent diffusion transformer for video generation. *arXiv preprint arXiv:2401.03048*, 2024.

Kangfu Mei and Vishal Patel. Vidm: Video implicit diffusion models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 9117–9125, 2023.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Norman Müller, Yawar Siddiqui, Lorenzo Porzi, Samuel Rota Bulo, Peter Kontschieder, and Matthias Nießner. Diffrf: Rendering-guided 3d radiance field diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4328–4338, 2023.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. *arXiv preprint arXiv:2308.15321*, 2023.

Hao Ouyang, Qiuyu Wang, Yuxi Xiao, Qingyan Bai, Juntao Zhang, Kecheng Zheng, Xiaowei Zhou, Qifeng Chen, and Yujun Shen. Codef: Content deformation fields for temporally consistent video processing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8089–8099, 2024.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 165–174, 2019.

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

William Peebles, Ilija Radosavovic, Tim Brooks, Alexei A Efros, and Jitendra Malik. Learning to learn with generative models of neural network checkpoints. *arXiv preprint arXiv:2209.12892*, 2022.

Zhiyao Ren, Yibing Zhan, Liang Ding, Gaoang Wang, Chaoyue Wang, Zhongyi Fan, and Dacheng Tao. Multi-step denoising scheduled sampling: Towards alleviating exposure bias for diffusion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 4667–4675, 2024.

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*, pp. 10684–10695, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

J Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3d neural field generation using triplane diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20875–20886, 2023.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10753–10764, 2021.

Ivan Skorokhodov, Willi Menapace, Aliaksandr Siarohin, and Sergey Tulyakov. Hierarchical patch diffusion models for high-resolution video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7569–7579, 2024.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

Yannick Strümpler, Janis Postels, Ren Yang, Luc Van Gool, and Federico Tombari. Implicit neural representations for image compression. In *European Conference on Computer Vision*, pp. 74–91. Springer, 2022.

Mingzhen Sun, Weining Wang, Gen Li, Jiawei Liu, Jiahui Sun, Wanquan Feng, Shanshan Lao, SiYu Zhou, Qian He, and Jing Liu. Ar-diffusion: Asynchronous video generation with auto-regressive diffusion. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 7364–7373, 2025.

Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 4489–4497, 2015.

Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Hanyu Wang, Saksham Suri, Yixuan Ren, Hao Chen, and Abhinav Shrivastava. LARP: Tokenizing videos with a learned autoregressive generative prior. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=Wr3UuEx72f.

Jiuniu Wang, Hangjie Yuan, Dayou Chen, Yingya Zhang, Xiang Wang, and Shiwei Zhang. Modelscope text-to-video technical report. *arXiv preprint arXiv:2308.06571*, 2023.

Junke Wang, Yi Jiang, Zehuan Yuan, Bingyue Peng, Zuxuan Wu, and Yu-Gang Jiang. Omnitokenizer: A joint image-video tokenizer for visual generation. *Advances in Neural Information Processing Systems*, 37:28281–28295, 2024a.

Shuai Wang, Ziteng Gao, Chenhui Zhu, Weilin Huang, and Limin Wang. Pixnerd: Pixel neural field diffusion. *arXiv preprint arXiv:2507.23268*, 2025b.

Yuyang Wang, Anurag Ranjan, Josh Susskind, and Miguel Angel Bautista. Inrflow: Flow matching for inrs in ambient space. *arXiv preprint arXiv:2412.03791*, 2024b.

Pingyu Wu, Kai Zhu, Yu Liu, Liming Zhao, Wei Zhai, Yang Cao, and Zheng-Jun Zha. Improved video vae for latent video diffusion model. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 18124–18133, 2025.

Shuzhou Yang, Moxuan Ding, Yanmin Wu, Zihan Li, and Jian Zhang. Implicit neural representation for cooperative low-light image enhancement. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12918–12927, 2023.

Lijun Yu, Yong Cheng, Kihyuk Sohn, José Lezama, Han Zhang, Huiwen Chang, Alexander G Hauptmann, Ming-Hsuan Yang, Yuan Hao, Irfan Essa, et al. Magvit: Masked generative video transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10459–10469, 2023a.

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*, 2024a. URL https://openreview.net/forum?id=gzqrANCF4g.

Qihang Yu, Mark Weber, Xueqing Deng, Xiaohui Shen, Daniel Cremers, and Liang-Chieh Chen. An image is worth 32 tokens for reconstruction and generation. *Advances in Neural Information Processing Systems*, 37:128940–128966, 2024b.

Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. Generating videos with dynamics-aware implicit generative adversarial networks. *arXiv preprint arXiv:2202.10571*, 2022.

Sihyun Yu, Kihyuk Sohn, Subin Kim, and Jinwoo Shin. Video probabilistic diffusion models in projected latent space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18456–18466, 2023b.

Sihyun Yu, Weili Nie, De-An Huang, Boyi Li, Jinwoo Shin, and Anima Anandkumar. Efficient video diffusion models via content-frame motion-latent decomposition. In *International Conference on Learning Representations*, 2024c.

Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: geometry editing of neural radiance fields. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 18353–18364, 2022.

Kaiwen Zha, Lijun Yu, Alireza Fathi, David A Ross, Cordelia Schmid, Dina Katabi, and Xiuye Gu. Language-guided image tokenization for generation. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 15713–15722, 2025.

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*, pp. 586–595, 2018.

Yunfan Zhang, Ties Van Rozendaal, Johann Brehmer, Markus Nagel, and Taco Cohen. Implicit neural video compression. *arXiv preprint arXiv:2112.11312*, 2021.

Qi Zhao, M Salman Asif, and Zhan Ma. Dnerv: Modeling inherent dynamics via difference neural representation for videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2031–2040, 2023.

A Implementation Details

A.1 Additional Model and Training Details

We use a medium configuration of ViT with 18 blocks, 14 heads and 896 hidden dimensions for our NeRV encoder. We set the scale of KL divergence loss to 1×10^{-5}. It patchifies the RGB videos into $8 \times 8 \times 1$ patches along height, width and time dimensions. We use sinusoidal positional embedding for our generative NeRV's input time index, instead of the exponential embedding in vanilla NeRV. For residual connections we use bilinear to upsample the earlier feature maps before merging them into the main branch.

Our discriminator is adapted from a 3D StyleGAN with 5 blocks, 64 unit hidden dimensions and a channel multiplier of 2 for each block. Its learning rate is set to one fifth of the NeRV autoencoder's, and it is updated every five iterations to stabilize the training. The scale of the GAN loss added to our NeRV autoencoder is 1.

Our implicit diffusion transformer adopts DiT-L configuration with 24 layers, 16 heads and 1024 hidden dimensions. Its patch size follows the token shape as output by our NeRV encoder. Our implicit DiT is optimized for predicting the noise ϵ at each timestep, and thus also adjust the Min-SNR-γ loss weighting accordingly. We employ CFG for class-conditioned sampling and the optimal guidance scale is 2.

Both our NeRV autoencoder and implicit DiT train with L2 reconstruction loss. We use AdamW(Loshchilov, 2017) optimizer with a linear warmup learning rate schedule and cosine decay. Both learning rates are set to 1×10^{-4}. We train our NeRV autoencoder for 2M iterations and train our implicit DiT for for 1M iterations.

A.2 Generative NeRV Architecture

We list the layer-wise architecture of our NeRV model in Table A1. "Feature Map" refers to the output feature map of each layer. "Modulation Weight" refers to the instance-specific weight latent to be assigned to each NeRV layers. T is the number of frames.

We set the dimensions of the time, height and width positional embeddings all to 16. We start from sampling a spatiotemporal positional embedding of shape $[8, 8, T, 48]$. It is transposed to queries along the time axis $[T, 48, 8, 8]$, and then spatial convolutions are applied on it.

We use kernel size $k = 4$ for all upsampling transposed convolutions and $k = 3$ for all other convolutions that don't change the feature map shape. We set the base hidden dimensions $D = 128, 256, 512$ for NeRV-Diffusion-S, -B and -L configurations, respectively. gelu is used for activations in all blocks while $\tanh$ is used after the tailing toRGB layer.

B Time Interpolation

As discussed in §4.5.1, we train NeRV-Diffusion on UCF with an interval of 8 frames, and interpolate the input time embeddings by an $8\times$ factor to sample 128-frame videos. The results are presented in Figure A1.

C INR Weight Interpolation

We further illustrate our generative NeRV's superiority in INR weight interpolation. DIGAN (Yu et al., 2022) proposes to interpolate between the latent noise vectors. When being interpolated between the whole weights, their video INR presents non-continuous transitions as shown in Figure A2 (top). This is because 1) their latent vectors are decoded from Gaussian noise with a complex non-linear mapping network; 2) their INR weights are modulated with low-rank cross product, termed as Factorized Matrix Multiplication (FMM) in Skorokhodov et al. (2021)), of the latent vectors, which break the arithmetic property. In contrast, our weight latent is directly used for modulation with a single linear affine layer from the KL bottleneck, and is directly assigned as NeRV parameters with minimal transforms. Our generative NeRV presents smooth interpolation effect as shown in

Table A1: Detailed architecture of our generative NeRV model and modulated weight shape. Batch size is omitted.

Layer	Feature Map Shape	Modulation Weight Shape
Input Init	$[8, 8, T, 48]$	-
Reshape	$[T, 48, 8, 8]$	-
Conv	$[T, D, 8, 8]$	$[48, D, 3, 3]$ or $[24, D/2, 3, 3]$
Transposed Conv	$[T, D, 16, 16]$	$[64, 64, 4, 4]$
Conv	$[T, D, 16, 16]$	$[64, 64, 3, 3]$
Conv	$[T, D, 16, 16]$	$[64, 64, 3, 3]$
Transposed Conv	$[T, D, 32, 32]$	$[64, 64, 4, 4]$
Conv	$[T, D, 32, 32]$	$[64, 64, 3, 3]$
Conv	$[T, D, 32, 32]$	$[64, 64, 3, 3]$
Transposed Conv	$[T, D, 64, 64]$	$[64, 64, 4, 4]$
Conv	$[T, D, 64, 64]$	$[64, 64, 3, 3]$
Conv	$[T, 2D, 64, 64]$	$[128, 64, 3, 3]$
Transposed Conv	$[T, 2D, 128, 128]$	$[64, 64, 4, 4]$
Conv	$[T, 2D, 128, 128]$	$[64, 64, 3, 3]$
toRGB	$[T, 3, 128, 128]$	$[D, 3, 3, 3]$
Reshape to Output	$[T, 128, 128, 3]$	-

Figure A2 (bottom). This property also opens up the potential of general direct manipulations on the tokenized NeRVs in a compositional manner with our NeRV autoencoder.

D Additional Results

We provide more generation samples of NeRV-Diffusion on UCF dataset in Figure A3. Video files in MP4 format are attached in the supplementary materials.

E Additional Experiments and Discussions

E.1 Inference Efficiency Comparison

We compare the inference speed and peak GPU memory of NeRV-Decoder and NeRV-Diffusion with other video decoders and generators, at 16 frames and both 128^2 and 256^2 resolutions. All results are tested on a single NVIDIA A6000 GPU in `bfloat16` at batch size 1 and averaged for 100 runs. All generators have enabled CFG and iterate for their default sampling timesteps. "-" denotes that the method was not developed for the resolution. We also extend NeRV-Diffusion to 256^2 resolutions with a sublinear increase in model and latent size. Full configurations are detailed in Appendix E.3.

The results are listed in Tables A2 and A3. We also include the VAE of Stable Diffusion (SD) and Stable Video Diffusion (SVD), which are also commercial large-scale foundation models that are trained on open-world data with considerable resources and extensive time. Overall, our models cost far less latency and VRAM footprint in both decoding and generation stages. It demonstrates the superior efficiency of our implicit framework, especially due to obviating temporal attentions by reusing the same set of parameters to decode all frames with redundancy.

E.2 Additional Quantitative Metrics.

The use of FVD for quantifying both reconstruction and generation quality on UCF and K600 datasets is a common protocol that have been widely practiced by SOTA work in recent top venues including MAGVITs (Yu et al., 2023a; 2024a), OmniTokenizer (Wang et al., 2024a),

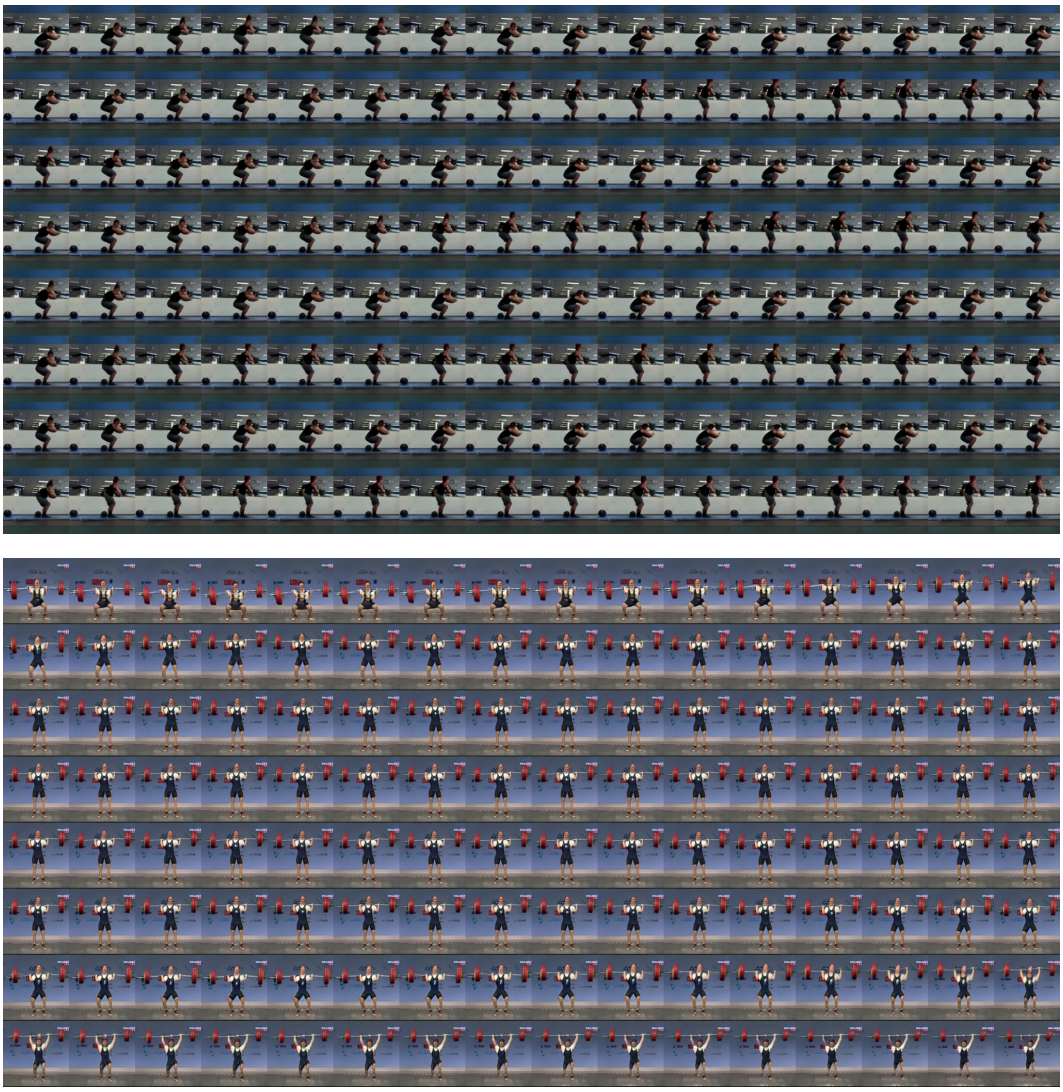


Figure A1: 128-frame video interpolation results on UCF. NeRV-Diffusion can be easily extended to efficient long video generation via smooth time interpolation after trained with large frame intervals.

Table A2: Decoding efficiency comparison.

Decoders	**#Params**		**Latency↓**		**VRAM↓**	
	128^2	256^2	128^2	256^2	128^2	256^2
Latte (Ma et al., 2024)	-	49M	-	0.288s	-	5.2G
CMD (Yu et al., 2024c)	24M	24M	0.129s	1.030s	1.2G	4.1G
Open-MAGVIT2 (Luo et al., 2024)	226M	-	0.127s	-	45G	-
SD-VAE (Rombach et al., 2022)	49M	49M	0.048s	0.260s	1.2G	4.3G
SVD-VAE (Blattmann et al., 2023a)	98M	98M	0.094s	0.411s	1.3G	4.5G
NeRV-VAE-L	55M	64.5M	0.032s	0.133s	0.99G	2.6G

HPDM (Skorokhodov et al., 2024), LARP (Wang et al., 2025a) and AR-Diffusion (Sun et al., 2025) etc. They didn't report extra metrics and now we perform additional comparisons with

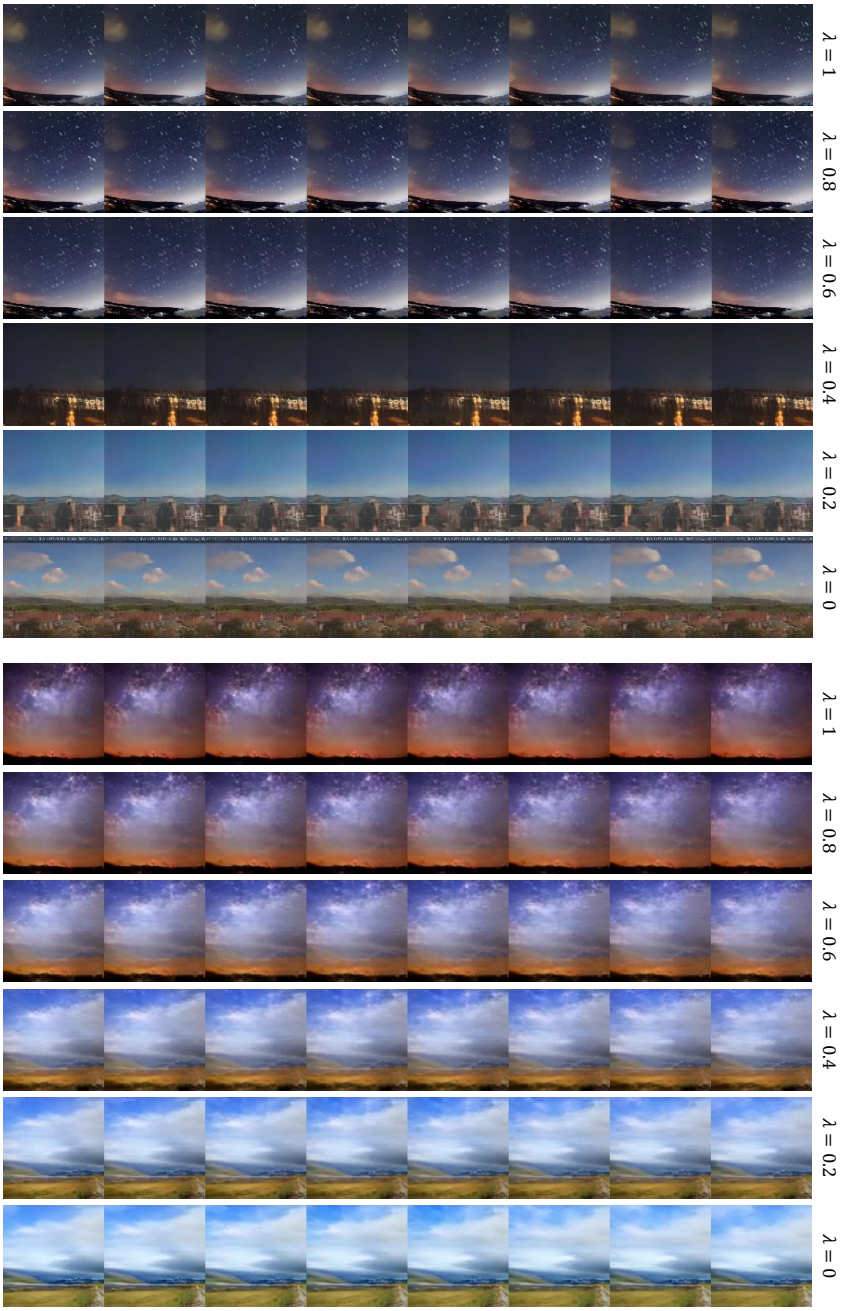


Figure A2: Interpolation of the whole parameters of the video INRs in DIGAN (top) and our generative NeRVs (bottom).

the open-sourced work that have released pre-trained checkpoints.

We extend our reconstruction metrics to PNSR, SSIM and LPIPS between the output and input videos, and extend our generation metrics to cross-frame LPIPS (fLPIPS) and C3D (Tran et al., 2015) / I3D (Carreira & Zisserman, 2017) -based Inception Score (Salimans et al., 2016) (IS) on the output videos. Due to the lack of pair-wise supervision in generation, we also evaluate generators on the non-text subsets of VBench Huang et al. (2024), including Subject Consistency (SC), Background Consistency (BC), Temporal Flickering (TF), Motion

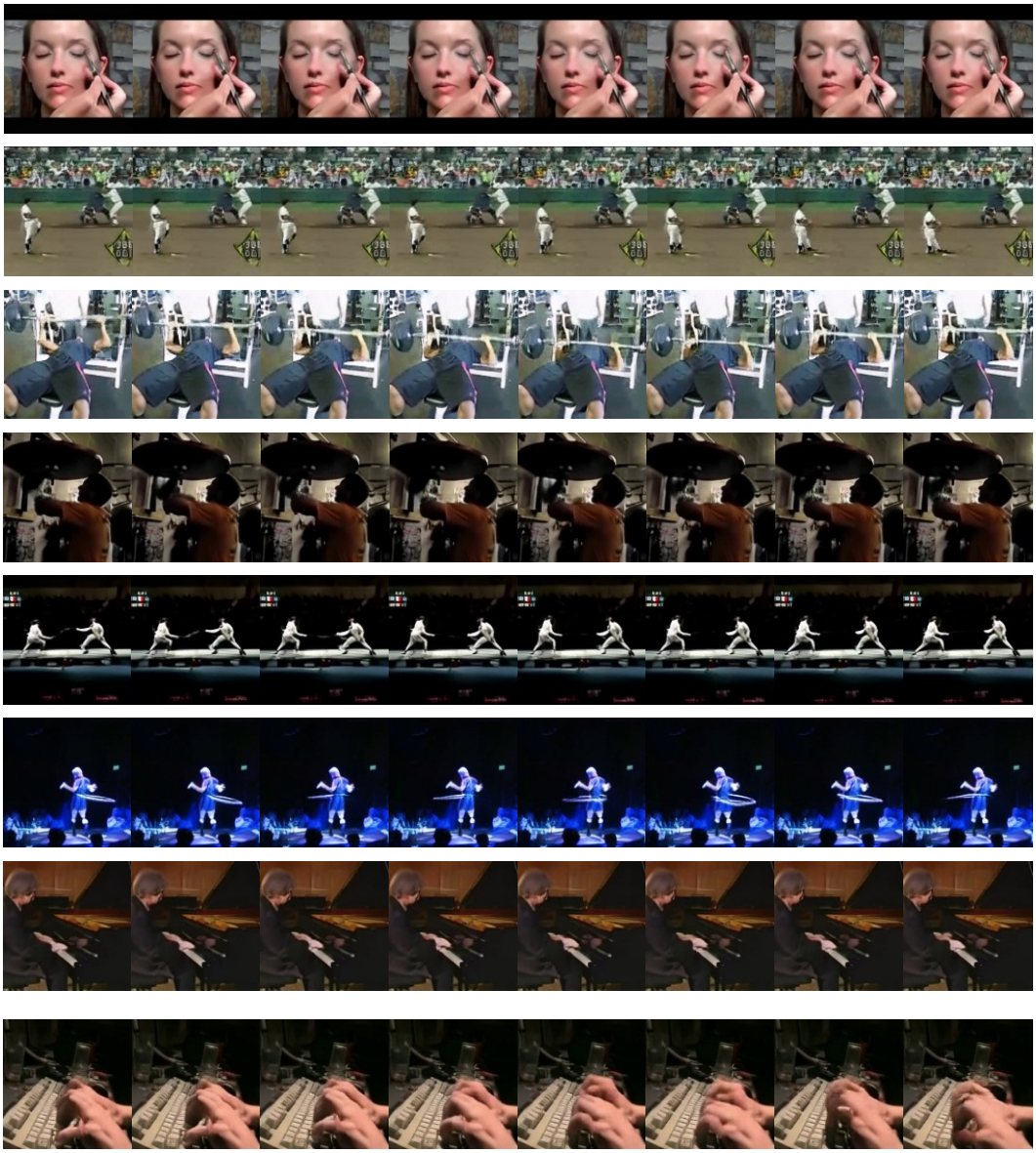


Figure A3: Additional class-conditioned generation samples on UCF.

Table A3: Generation efficiency comparison.

Generators	#Params	#Tokens		Steps	Latency↓		VRAM↓	
		128^2	256^2		128^2	256^2	128^2	256^2
Latte (Ma et al., 2024)	674M	-	512	250	-	37s	-	4.4G
LARP-L (Wang et al., 2025a)	343M	1024	-	1024	20s	-	1.6G	-
OmniTokenizer (Wang et al., 2024a)	650M	-	1280	5120	-	139s	-	4.5G
NeRV-Diffusion-L	467M	128	160	250	6.8s	8.2s	1.8G	2.1G

Smoothness (MS), Dynamic Degree (DD), Aesthetic Quality (AQ) and Imaging Quality (IQ). For comparing methods, we use their pretrained checkpoints for inference.

It worth noting that the ultimate purpose of NeRV-VAE (and the comparing tokenizers) is to encode RGB videos into smooth latents for high-quality video generation, and we optimize it for best generative quality instead of solely reconstruction faithfulness. Our models achieve SOTA generation performance on UCF across all dimensions, and reach comparable SOTA performance on K600 frame prediction.

Table A4: Reconstruction performance comparison on UCF.

Reconstruction (UCF)	**PNSR↑**	**SSIM↑**	**LPIPS↓**	**rFVD↓**
TATS (Ge et al., 2022)	22.43	0.765	0.108	162
Open-MAGVIT2 (Luo et al., 2024)	25.84	0.862	0.045	16
LARP-L (Wang et al., 2025a)	27.87	0.891	0.038	20
NeRV-VAE-L	26.63	0.879	0.043	22

Table A5: Generation performance comparison on UCF.

Generation (UCF)	**fLPIPS↕**	**IS↑**	**VBench↑**							**gFVD↓**
			SC	BC	TF	MS	DD	AQ	IQ	
Ground Truth	0.031	86.24	0.954	0.977	0.981	0.988	0.282	0.410	0.443	0
TATS (Ge et al., 2022)	0.048	68.79	0.910	0.958	0.978	0.980	0.314	0.348	0.437	332
VIDM (Mei & Patel, 2023)	-	64.17	-	-	-	-	-	-	-	263
VideoDiffusion (Luo et al., 2023)	-	80.03	-	-	-	-	-	-	-	173
LARP-L (Wang et al., 2025a)	0.025	68.79	0.955	0.977	0.985	0.991	0.218	0.398	0.393	102
NeRV-Diffusion-L	0.028	82.17	0.958	0.978	0.983	0.989	0.262	0.392	0.433	97

Table A6: Reconstruction performance comparison on K600.

Reconstruction (K600)	**PNSR↑**	**SSIM↑**	**LPIPS↓**	**rFVD↓**
LARP-L (Wang et al., 2025a)	28.22	0.867	0.035	11
NeRV-VAE-L	26.45	0.823	0.044	19

E.3 Scaling Up Video Resolution and Length

We scale up NeRV-Diffusion to 256^2 resolution and 128 frames on UCF. Conventional VAEs encode videos into frame-wise feature maps with a fixed spatial downsampling factor, and thus their latent size increases quadratically w.r.t. to RGB resolutions. In contrast, NeRV-Diffusion uses INRs as instance-specific decoders, and we append one additional layer/block to the end to perform an extra upsampling to double the output resolution. Therefore, our model parameters increase sublinearly, and the neural weight latent size also only needs to increase accordingly. Specifically, our NeRV decoder size increases by 17% as shown in Table A2, and we increase the neural latent token numbe by 25% for simple alignment of channel-wise parameterization. Besides, our GNeRV incorporates skip connections (§3.2) so multi-resolution output and joint training are also feasible.

For video length, conventional VAEs increase their frame-wise latent size linearly with a fixed temporal downsampling factor, while our NeRV features smooth native time interpolation and can be trained with large frame intervals as shown in Section 4.5.1. In our 128-frame experiment, we train NeRV-Diffusion at 16 frames with the downsampling interval of 8 and sample it with $8\times$ frame interpolation. This would be more challenging as no ground truth frames between the anchored frames are seen by our models.

Table A7: Generation performance comparison on K600.

Generation (K600)	fLPIPS↕	IS↑	VBench↑							gFVD↓
			SC	BC	TF	MS	DD	AQ	IQ	
Ground Truth	0.031	31.20	0.950	0.945	0.978	0.988	0.266	0.406	0.461	0
LARP-L (Wang et al., 2025a)	0.029	23.51	0.933	0.973	0.981	0.989	0.292	0.337	0.293	17
NeRV-Diffusion-L	0.034	27.09	0.928	0.967	0.977	0.979	0.354	0.392	0.432	22

We report the generation performance comparison in Tables A8 and A9. In both cases our NeRV-Encoder remains the same configuration. NeRV-Diffusion outperforms recent SOTAs with high compactness and easy extensibility.

Table A8: Generation performance comparison on UCF with 256^2 resolution.

Method	gFVD^{256}↓
VIDM (Mei & Patel, 2023)	263
Latte (Ma et al., 2024)	202
OmniTokenizer (Wang et al., 2024a)	191
AR-Diffusion (Sun et al., 2025)	186
HPDM-M (Skorokhodov et al., 2024)	143
NeRV-Diffusion-L (Ours)	140

Table A9: Generation performance comparison on UCF with 128 frames.

Method	gFVD$_{128}$↓
Latte (Ma et al., 2024)	1157
DIGAN (Yu et al., 2022)	1103
PVDM (Yu et al., 2023b)	505
VIDM (Mei & Patel, 2023)	426
CoordTok (Jang et al., 2025)	369
NeRV-Diffusion-L (Ours)	366

E.4 Dataset Scale and Complexity

UCF and K600 are the common standard benchmarks to evaluate novel video generation architectures, widely adopted by SOTA work in recent top venues including MAGVITs (Yu et al., 2023a; 2024a), OmniTokenizer (Wang et al., 2024a), HPDM (Skorokhodov et al., 2024), LARP (Wang et al., 2025a) and AR-Diffusion (Sun et al., 2025) etc. Our computation resources in academia are limited and training open-world text-to-video is not feasible. NeRV-Diffusion follows the generic diffusion pipeline and it can be extended to additional condition input (please refer to Appendix E.7).
Meanwhile, UCF dataset contains over 13K real-world videos sourced from YouTube of average 180 frames, totally about 27 hours. K600 dataset contains about 500K videos of average 250 frames, summing up to about 57 days in length. The total information is far more than our models' parameters. They are realistic and complex data and training and evaluating on them are non-trivial.

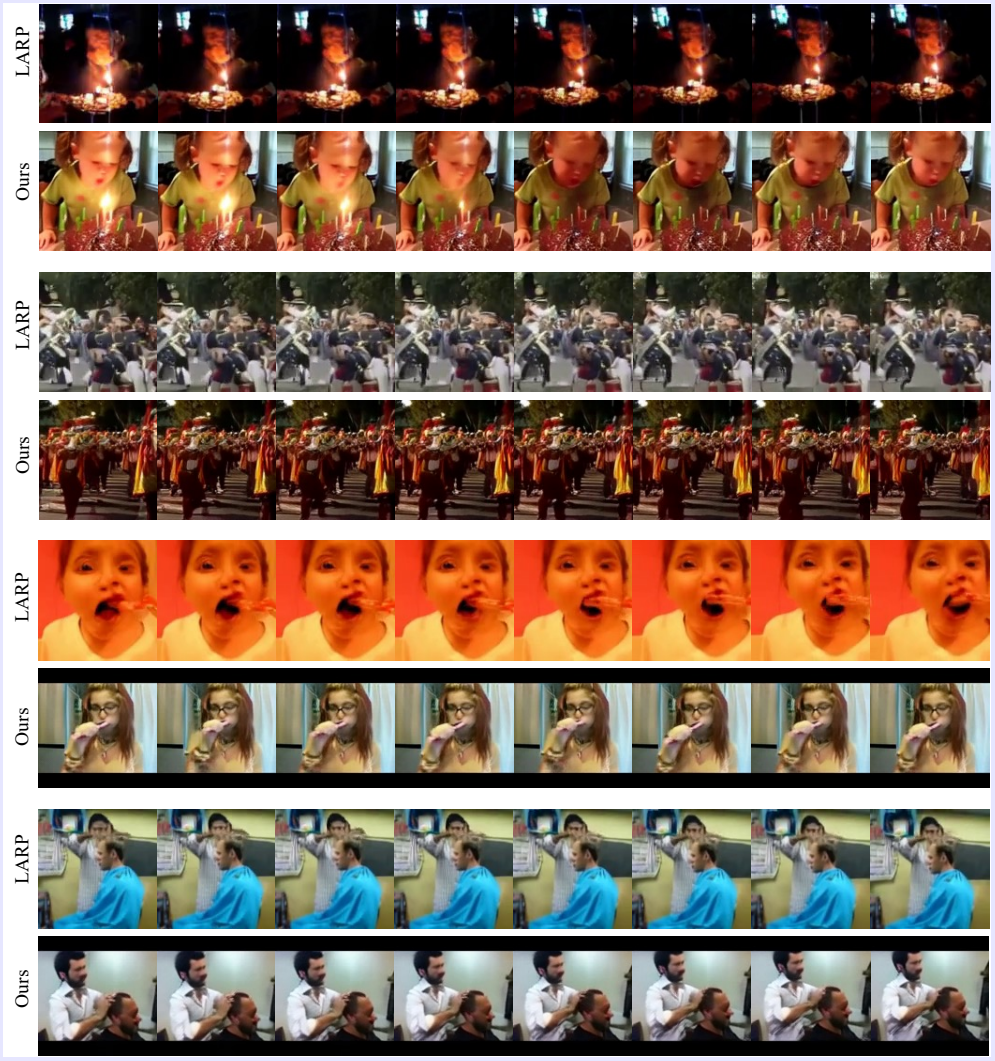


Figure A4: Qualitative comparisons with LARP on UCF.

E.5 Qualitative Comparisons

We present additional visual comparisons with LARP (Wang et al., 2025a) on UCF in Figures A4 and A5. We sample both models with the same class label input. We updated the attached supplementary materials to include all raw video files. We also embed the videos in Figure A6 that can be displayed directly in PDF. Compared to LARP, NeRV-Diffusion also constructs holistic video representations but meanwhile still maintain the spatiotemporal integrity via the query time indices and spatial input embeddings, and thus produces more structural videos with less morphing or tearing.

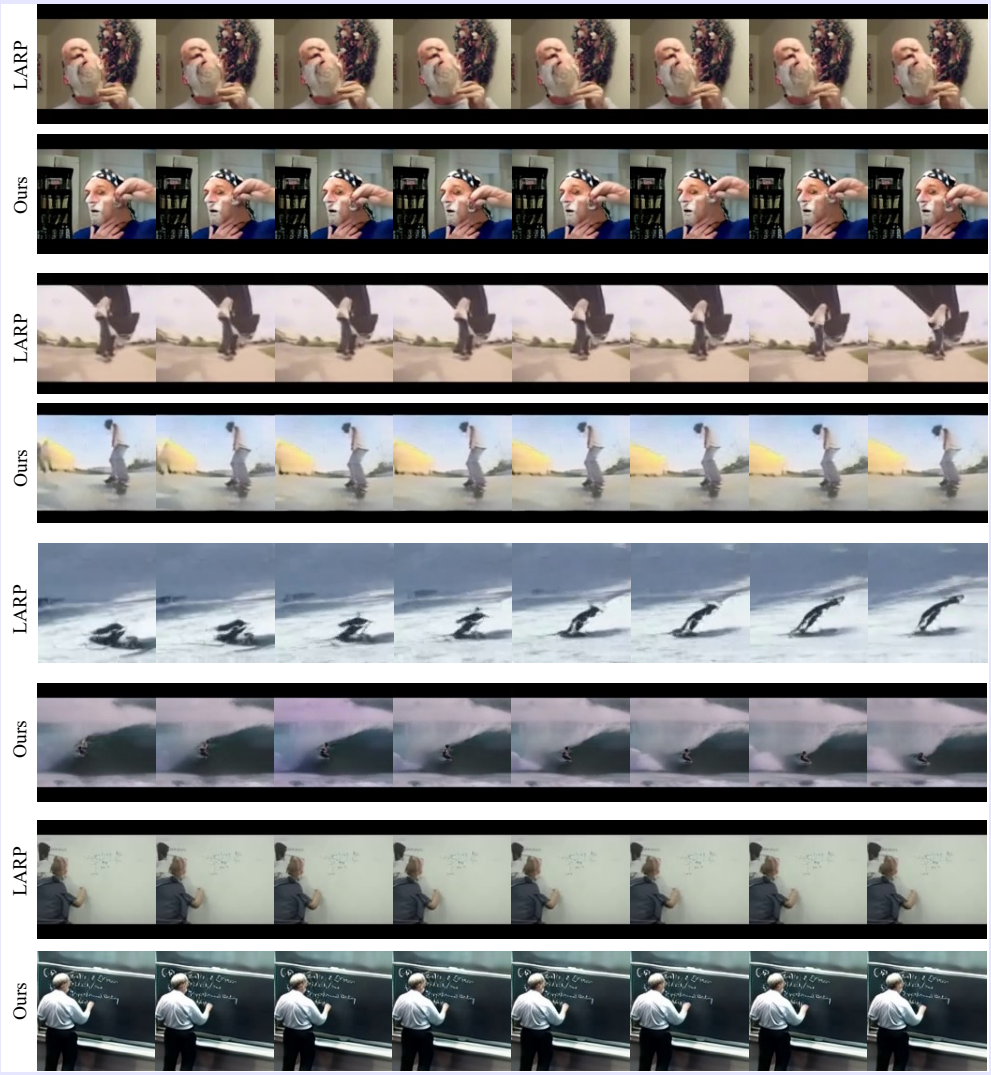
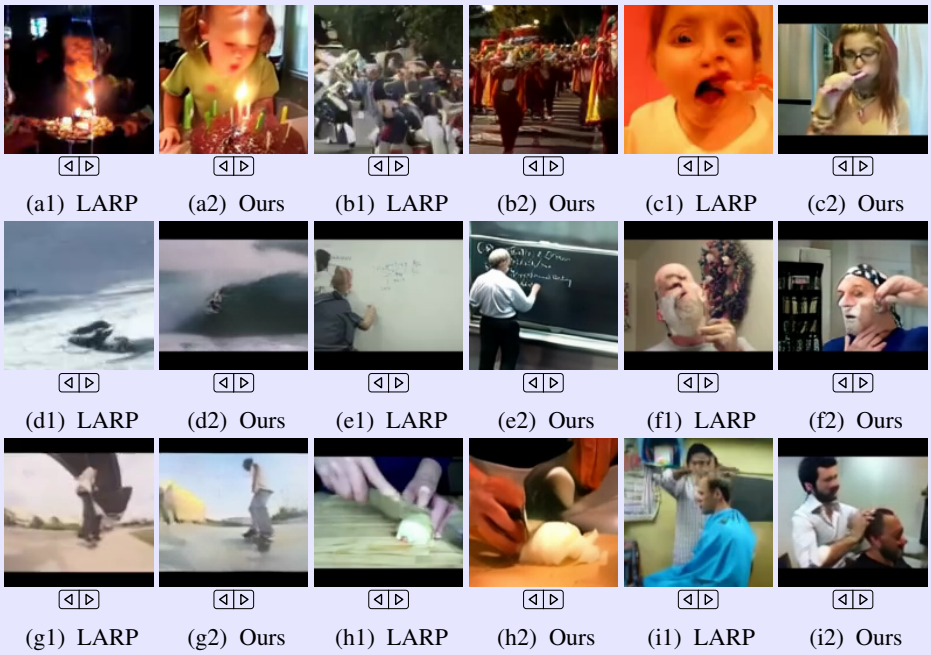


Figure A5: Qualitative comparisons with LARP on UCF class-conditioned generation.

E.6 Quantitative Verification of Time Interpolation

We quantitatively measure long video semantic consistency in terms of object identity and action logic preservation. Specifically, we leverage the Subject Consistency (SC) and Background Consistency (BC) metrics in VBench, which extract the subject and background features via pre-trained DINO (Caron et al., 2021), and calculate their pairwise similarities across all frames for appearance preservation. Inspired by it, we further employ C3D, a pre-trained action recognizer network, and feed it with all sub-clips from the long videos to extract action features. We calculate pairwise similarities of the action features across all windows to measure the action logic preservation. These metrics don't only rely on consecutive distance but average along the whole video, accurately reflecting the drifting issues for long video generation. The results are listed in Table A10.



(a1) LARP (a2) Ours (b1) LARP (b2) Ours (c1) LARP (c2) Ours

(d1) LARP (d2) Ours (e1) LARP (e2) Ours (f1) LARP (f2) Ours

(g1) LARP (g2) Ours (h1) LARP (h2) Ours (i1) LARP (i2) Ours

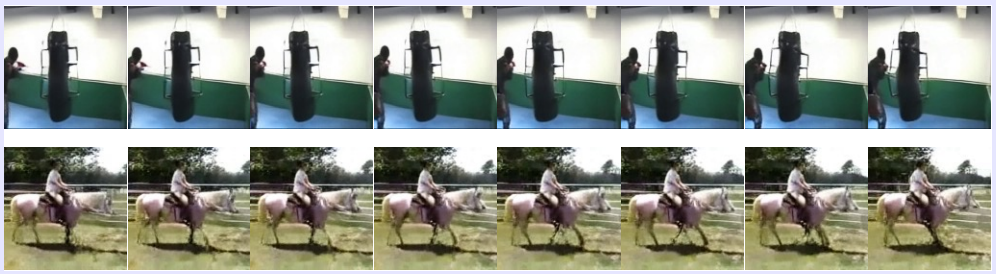
Figure A6: Videos of visual comparisons between LARP and NeRV-Diffusion, following the orders in Figures A4 and A5. Please use Adobe Acrobat Reader on laptop/desktop and click to play.

Table A10: Time interpolation semantic consistency.

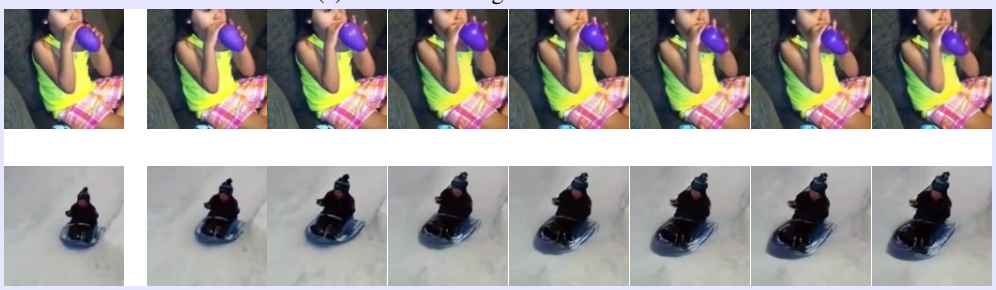
Long Consistency	**VBench↑**		**Action Sim.↑**
	SC	BC	
Ground Truth	0.931	0.956	0.912
Ours	0.919	0.942	0.901

E.7 Generalizability to Downstream Tasks with Controllability

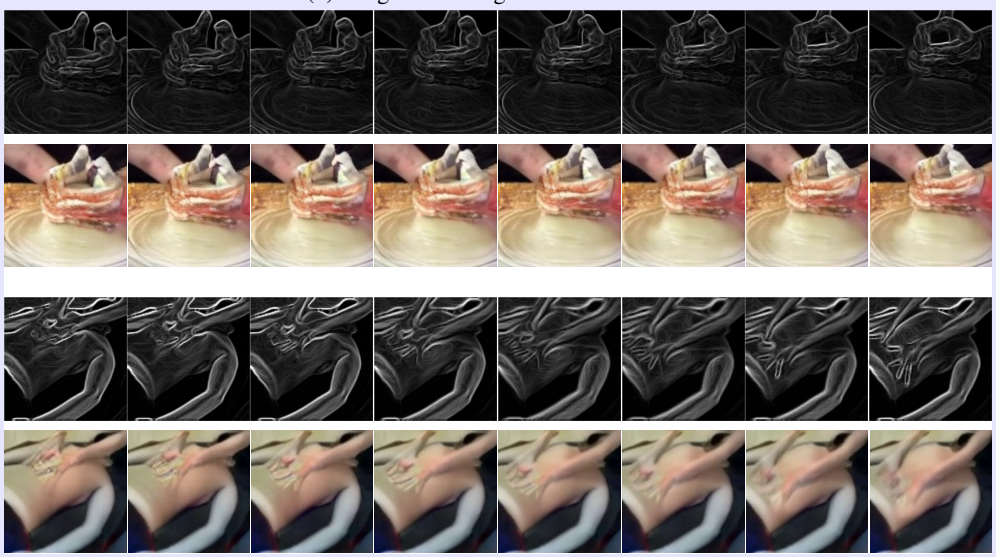
Our implicit diffusion model follows standard diffusion framework but only switches to neural weight latents, and thus can be trained with versatile condition types. Additional condition input can be integrated with standard cross attention so that different modalities other than videos or images are also supported. We follow the prior work that developed novel tokenizers to majorly test on the foundation tasks. Due to time limit and resource scale, we extend NeRV-Diffusion to several additional downstream tasks, including image-to-video generation and unconditional video generation. To showcase the support of granular controllability of our nerual weight latents, we also include an edge-to-video generation experiment. Previous work didn't evaluate on these setups, and we present our qualitative results in Figure A7. These experiments demonstrate the native extensibility of NeRV-Diffusion spanning from none to fine-grained condition controls with high quality.



(a) Unconditional generation on UCF.



(b) Image-to-video generation on K600.



(c) Edge-conditioned video generation on K600.

Figure A7: Versatile condition types of video generation supported by NeRV-Diffusion.

E.8 Novelty and Contribution Compared to DIGAN

GANs and diffusions are two distinct generative model families. The idea of generating INR weights has emerged before DIGAN and has been developed for various modalities with different model designs, and we have discussed them in §2. Our proposed modules such as multi-head affine and channel-wise parameterization has effectively boosted the performance of our models to achieve the first implicit video diffusion model, and we have depicted the impact of key design options in §4.4.

E.9 Two-stage Tokenizer-Generator Framework

A two-stage tokenizer-generator framework has been widely adopted for almost all large-scale generative models. Latent Diffusion Models (LDMs, or Stable Diffusions, SDs) encodes RGB data into continuous latents and train diffusion models on the latents. Vector-Quantized GANs (VQGANs) also encodes RGB data into discrete codes and train autoregressive (AR) models on the codes. They have achieved successes in various domains, including T2I, T2V, etc. Although they need to be trained in separate stages and need to balance between expressiveness and Gaussianity, two-stage frameworks are known for their stability and efficiency compared to single-stage generation models such as GANs, especially when scaled up to large-scale models and data. The expressiveness and dynamics of latent diffusion models for larger-scale and more complex data will not be limited by the KL trade-off.

E.10 Trade-off Between Expressiveness and Gaussianity

All tokenizers designed for generative modeling in a two-stage framework need to balance between the latent compactness and expressive richness. For VAEs it is the KL distance on the continuous latents, and for VQGANs it is the codebook alignment on the discrete code. One-stage generators like pixel diffusion models have to process raw RGB data and map bigger random noise, which are harder to scale up. An autoencoder without variational (KL) constraint leads to sparse and unbounded latent values, which have no impact on video compression but cannot be used to train diffusions. In our work, we are able to manage this balance well and provide a stable and clear recipe with high quality.

On the other hand, our proposed modules, such as multi-head affine and channel-wise parameterization, are mainly designed for adapting and enhancing the NeRV decoder interface. These components exist in prior work such as StyleGAN, DIGAN, TransINR and FastNeRV for encoder-decoder connection and latent space modulation, and we make simple yet effective upgrades to largely lift their expressiveness.

E.11 Granularity of Neural Network Weights

Our neural latents form up the convolution weights of NeRV decoder, taking in unified spatiotemporal input embeddings to render pixel frames. Although it is holistic and different from conventional frame-wise feature maps, on the other hand it still performs convolutions, i.e. matrix multiplications over the spatiotemporal input grid (in the opposite positions), and thus still maintains spatiotemporal information. This is a major difference of our implicit representations between discrete holistic video latents such as LARP. We demonstrate the smoothness of our neural network weight space in Appendix C by direct interpolations between them.

We further design another probing experiment to validate the spatiotemporal association between our neural weight latents and pixel frames. We crop RGB videos into random clips with random spatial and temporal ranges, and encode them using various video encoders. For each video encoder, we perform K-Means on its encoded latents, with the raw video sources as the class labels. We calculate the purity and normalized mutual information (NMI) between the latent clusters and their real labels, i.e. which raw video the corresponding clip is cropped from. Table A11 shows that our neural weight latents have spatiotemporal relations with pixel operations, indicating its potential to facilitate granular editing.

Table A11: Time interpolation semantic consistency.

Granularity	**Purity↑**	**NMI↑**
SD-VAE	0.484	0.753
LARP	0.285	0.591
Ours	0.366	0.656

E.12 Impact of KL Loss Scale.

In all two-stage frameworks, the reconstruction fidelity of the first-stage VAE determines the upper bound of the generation quality of the second-stage generator. Given other factors (e.g. KL loss scale) fixed, the generation quality will follow the change of reconstruction quality. Table 1 shows that for different model size configurations (NeRV-Diffusion-S, -B, -L), our gFVD changes following rFVD. The gap between gFVD and rFVD is controlled by the KL loss scale. A high KL loss scale will harm rFVD but reduce the gap between gFVD and rFVD, while a low KL loss scale will improve rFVD but enlarge the gap between gFVD and rFVD. We ablate its impact on NeRV-Diffusion-S configuration in Table A12.

Moreover, Table 1 displays that different methods have their own optimal gFVD-rFVD gap, corresponding to their individual optimal balance between the reconstruction fidelity and latent space smoothness constraint. NeRV-Diffusion features a relatively low gFVD-rFVD gap at all scales, indicating the smoothness of our implicit latent space. We also attribute this to that conventional frame-wise tokenizers rely on input videos to guide their motion flow for reconstruction, while usually needing temporal attentions to constrain temporal consistancy in the generator, which create extra challenge and thus reconstruction-generation gap. NeRV-Diffusion highlights its holistic video representations without cross-frame regularization in either VAE and the implicit DiT, leading to consistent performance across reconstruction and generation.

Table A12: Ablation on KL loss scales on NeRV-Diffusion-S configuration.

KL Loss Scale	1×10^{-6}	5×10^{-6}	1×10^{-5}	5×10^{-5}
rFVD	73	82	85	107
gFVD	198	186	184	202

E.13 Latent Gaussianity Diagnostics

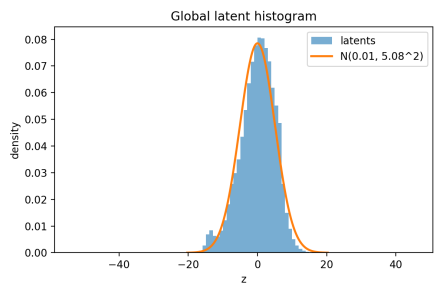
We conduct marginal tests and normality plots on our encoded neural weight latent as suggested, and compare it to LDM/SD's VAE latent. The statistics and visualizations are presented in Table A13 and Figure A8.

Table A13: Latent Gaussianity statistics.

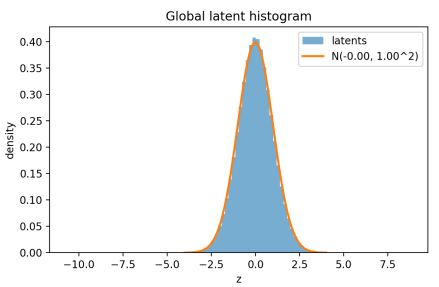
Gaussianity	**mean**	**std**	**skew**	**kurtosis**
LDM/SD-VAE	-0.0165	5.2505	-0.4756	0.1693
NeRV-VAE-L	-0.0036	0.9979	-0.0001	0.2568

E.14 Further Scaling Up

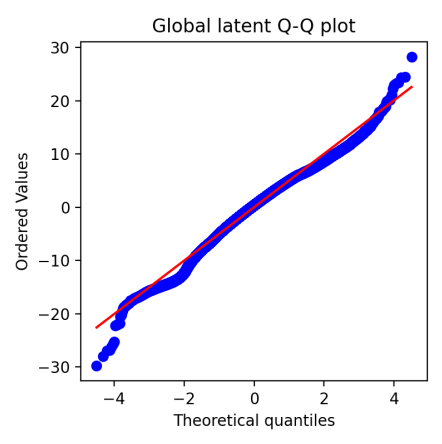
Our explorations started from compact NeRV decoders and we found that scaling up NeRV decoders would considerably improve the performance. We have ablated neural weight latent size in Table 2f, and it shows that oversized latents will be harder for DiT of the same size to train on. However this is the common pattern for all VAEs and latent diffusions, as the upper bound of latent size would be the raw RGB size without encoding, and LDMs usually outperform pixel diffusions. Due to resource limit, we haven't been able to further scale up the latent or decoder size as our current experiments have exhibited scaling-up boost.



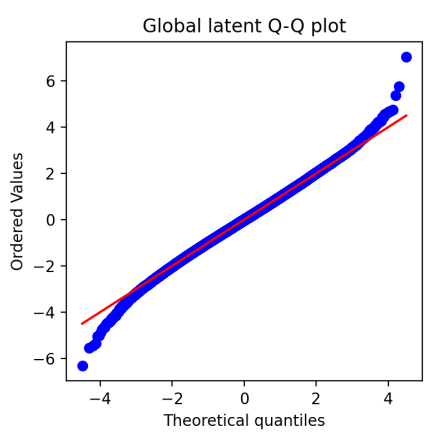
(a) Histogram of SD-VAE latents.



(b) Histogram of NeRV-VAE latents.



(c) Quantile-Quantile of SD-VAE latents.



(d) Quantile-Quantile of NeRV-VAE latents.

Figure A8: Gaussianity verifications of the latents encoded by NeRV-VAE and SD-VAE.

E.15 Comparison with Open-World Large-Scale Video Tokenizers.

LTX (HaCohen et al., 2024) and WAN (Wan et al., 2025) are commercial foundation models that are trained on open-world large-scale data with considerable resources and extensive time. As a research work, we are not able to compete against them but mainly to explore a novel architecture of implicit tokenization and generation that outperforms previous SOTAs under similar settings. Though, we list the quantitative comparisons in Table A14. We also include the VAE of Stable Diffusion (SD) and Stable Video Diffusion (SVD), which are also large-scale foundation models at earlier stages.

Table A14: Comparison of reconstruction performance on UCF with foundation video tokenizers.

Reconstruction (UCF)	**PNSR↑**	**SSIM↑**	**LPIPS↓**	**rFVD↓**
SD-VAE	24.79	0.783	0.068	63
SVD-VAE	27.24	0.856	0.058	39
LTX-VAE	31.29	0.909	0.061	32
WAN-VAE	31.14	0.935	0.022	7
NeRV-VAE-L	26.63	0.879	0.043	22

E.16 Encoding and Decoding Efficiency on 720P

We create random data and test the encoding/decoding latency on a 1280×720 video. The encoding takes approximately 1.9s, and the decoding takes approximately 1.8s, both on a single NVIDIA A6000 GPU.

E.17 Training efficiency Comparison

We train each of our first-stage NeRV-VAE and second-stage NeRV-Diffusion for 1 week on 8 NVIDIA A6000 GPUs. We train NeRV-VAE at a small batch size of 32 for 2M iterations. In comparison, LARP-L-Long is trained for 500K iterations at batch size of 128. MAGVIT-v2 is trained for 300K iterations at batch size 256. OmniTokenizer is trained for 1M iterations on 8 NVIDIA A100 GPUs for 2 weeks.

E.18 Explanation of Multi-head Affine Modulation

In standard configurations of previous INR autoendoers like TransINR and FastNeRV, the bottleneck is formed with two FC layers after the encoder and before the decoder. The two FC layers downsample and upsample the token dimensions to form the information bottleneck. Our Multi-head affine module contains multiple second FC layers after the bottleneck latent, upsampling it from one to multiple sets of latent tokens. In this way, our model is able to modulate all layers in the NeRV decoder with different values that sourced from one bottleneck latent, largely expand its capacity.

E.19 Neural Weight Latent Dimension Ablation

We ablate different neural latent token dimensions with NeRV-Diffusion-S configuration in Table A15. It results in a slight performance drop when being reduced, but not as significant as token numbers shown in Table 2f. We attribute this to that our neural weight latents are holistic representations unlike conventional frame-wise feature maps, and the token dimension doesn't correspond to the RGB color channel.

Table A15: Ablations on neural weight latent dimensions.

Token Shape	128×32	128×64	128×96	128×128
rFVD↓	114	99	92	85
gFVD↓	230	212	197	184

E.20 Efficient Reconstruction with Few Tokens

Our implicit video tokenizer features its INR decoder dedicated for each data point, i.e. the neural tokens directly serve as decoder parameters instead of input, showing its effectiveness over traditional feature map latents. Our multi-head affine module reuses the neural tokens for all layers in the INR decoder, which also enlarges the expressiveness.

E.21 Neural Latent Token Number Options

For generation, there is usually a balance that too few tokens will lack expressiveness, and too many tokens will harm the smoothness of the latent space. Given the certain amount of each token deviating from standard Gaussian distribution, more tokens will introduce higher total variance and the diffusion model will thereby be harder to converge. This applies to all two-stage generation frameworks, given fixed VAE and denoising model sizes. We ablated different token numbers in Table 2f, and adopted for the optimal generative quality. Due to time and resource limit, we didn't ablate on more tokens beyond.

E.22 Performance Ceiling over Efficiency

So far we haven't noticed such a ceiling as NeRV-Diffusion surpasses previous SOTA methods of both diffusion and autoregressive models. We attribute this to that our implicit neural weight latent is a holistic representation and leverages the redundancy in videos. Meanwhile, the outstanding efficiency of NeRV-Diffusion also indicates its extra potential when scaling up. If performance is prioritized over efficiency, NeRV-Diffusion can also further scale up to comparable efficiency for extra performance boost.

F Reproducibility Statement

Our code and trained checkpoints will be made publicly available upon publication. We have discussed our complete implementation details in §4.1 and Appendix A, including the model configuration and training recipe.