Efficient Neural Video Representation with Temporally Coherent Modulation

Seungjun Shin*, Suji Kim*, and Dokwan Oh

Samsung Advanced Institute of Technology
{sj0216.shin, sujii.kim, dokwan.oh}@samsung.com

Abstract. Implicit neural representations (INR) has found successful applications across diverse domains. To employ INR in real-life, it is important to speed up training. In the field of INR for video applications, the state-of-the-art approach [25] employs grid-type parametric encoding and successfully achieves a faster encoding speed in comparison to its predecessors [6]. However, the grid usage, which does not consider the video's dynamic nature, leads to redundant use of trainable parameters. As a result, it has significantly lower parameter efficiency and higher bitrate compared to NeRV-style methods [6, 27, 5] that do not use a parametric encoding. To address the problem, we propose *Neural Video representation with Temporally coherent Modulation* (NVTM), a novel framework that can capture dynamic characteristics of video. By decomposing the spatio-temporal 3D video data into a set of 2D grids with flow information, NVTM enables learning video representation rapidly and uses parameter efficiently. Our framework enables to process temporally corresponding pixels at once, resulting in the fastest encoding speed for a reasonable video quality, especially when compared to the NeRV-style method, with a speed increase of over 3 times. Also, it remarks an average of 1.54dB/0.019 improvements in PSNR/LPIPS on UVG (Dynamic) (even with 10% fewer parameters) and an average of 1.84dB/0.013 improvements in PSNR/LPIPS on MCL-JCV (Dynamic), compared to previous grid-type works. By expanding this to compression tasks, we demonstrate comparable performance to video compression standards (H.264, HEVC) and recent INR approaches for video compression. Additionally, we perform extensive experiments demonstrating the superior performance of our algorithm across diverse tasks, encompassing super resolution, frame interpolation and video inpainting.

Keywords: Implicit Neural Representation · Neural Video Compression · Parametric encoding

1 Introduction

Implicit neural representation (INR) is a technique that represents a signal as a continuous function of its corresponding coordinates. Because it is effective

* Equally contributed.

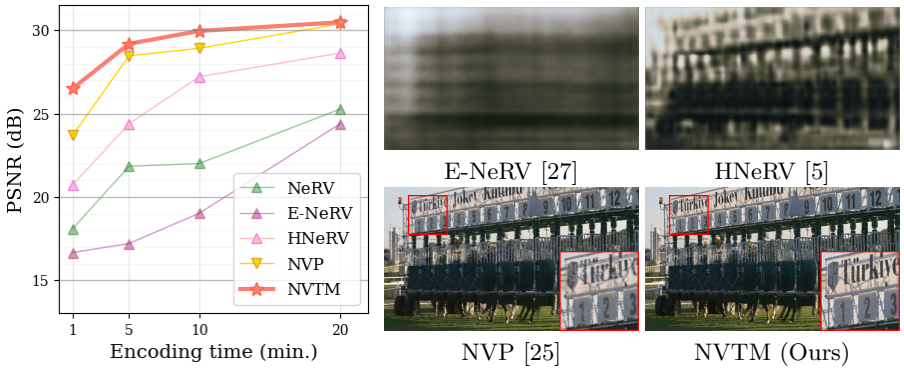


Fig. 1: **Fast encoding speed with high image quality.** (Left) The encoding speed in UVG, where all models are configured at **0.1bpp** and evaluated on the same resource conditions. NVTM learns quickly and achieves 30dB **3×faster than the NeRV-series**. (Right) Video reconstruction results on ReadySetGo sequence **after training for 1 minutes**. While E-NeRV and HNeRV exhibit blurry outputs, NVP and NVTM, based on parametric encoding, quickly capture complex representations. Further, NVTM excels at representing fine details such as text and numbers.

to handle complex signals, INR has gained considerable attention across various domains such as images [38, 7], sounds [41, 39], 3D objects and scenes [16, 42, 12, 33, 19], and compression [14, 13]. Following this trend, video applications of INR are now being explored in many studies [25, 8]. They have unique advantages such as the ability to play videos at arbitrary resolutions and frame rates, as well as the capability for video inpainting. Furthermore, leveraging INR in video compression leads to remarkable breakthroughs [27, 5, 30, 49, 22, 21].

NeRV [6], a framework that iteratively combines convolution and pixel-shuffle operators, was proposed for the application of INR in video reconstruction and compression. Numerous follow-up studies are conducted [27, 5, 30, 49, 22, 21] and they emphasize the applicability in video compression by diminishing input dimensions or replacing input coordinates with image features. However, they lost one of INR's major advantages, which is the capacity to produce outputs at various resolutions using a single learned model. In addition, the slow encoding speed still remains as the main challenge in those architectures. Since all parameters of network must be updated for every pixel, serious computational inefficiency occurs and the encoding time increases. To overcome this challenge, parametric encoding (e.g., grid) is being widely adopted [4, 18, 28, 9, 24]. They achieve a faster training speed via their strong locality (computing-efficient), however, they have the drawback that the model size needs to increase according to the input dimension (parameter-inefficient). Also, the attempts to directly embedding videos into a 3D grid [20, 35] or decomposing into three 2D grids [25] did not sufficiently consider the dynamic nature of videos. This results in the duplication of parameters, and large parameter size being required for reasonable performance.

In the field of video processing, it is important to deal with temporal redundancy between adjacent frames [10, 47]. Video codecs [46, 40] also efficiently encode videos by dealing such temporal redundancy with motion compensation similar to the philosophy, leaving only residual information in each frame. Several studies [26, 37] in the INR fields have adopted a warping-residual structure to eliminate temporal redundancy. However, no studies have been conducted to consider removing temporal redundancy while using parametric encoding. Therefore, we propose a computing-efficient (fast encoding speed, shown in Figure 1) and parameter-efficient (high reconstruction quality, shown in Table 3). INR framework that takes into account the dynamic characteristics of videos. The key idea is utilizing a series of 2D grids to represent videos by employing the same modulation to the corresponding pixels. Overall, we make the following contributions:

- We propose a novel framework *Neural Video representation with Temporally coherent Modulation* (NVTM), which applies consistent modulation equally to corresponding along the time axis.
- Our framework achieves a fast training speed and high parameter efficiency on video representation.
- We validate the performance on extensive experiments with various datasets and various tasks including video reconstruction, video compression, video super resolution, video frame interpolation, and video inpainting compared to state-of-the-art methods.

2 Related Works

2.1 Implicit Neural Representation (INR)

INR, also known as neural fields or coordinate-based neural representation (CNR), has emerged as a new paradigm for representing complex and continuous signals. It interprets data as a continuous signal and proposes a methodology where data is encoded into a neural network using coordinate inputs.

2.2 INR for videos

Video data is composed of consecutive frames, and many studies have attempted to find a better framework to apply INR to video. Since pixel-wise INR, which output the (r, g, b) for 3D coordinate input $(x, y, t) \in \mathbb{R}^3$, has slow encoding speed and low parameter efficiency, NeRV [6] proposed to frame-wise INR with 1D coordinate input $t \in \mathbb{R}^1$. Although this frame-wise INR on does not consider spatial input $(x, y) \in \mathbb{R}^2$, it could efficiently represent videos with comparable performance in video compression. Subsequent studies also have provided notable improvements. [27] improved performance by eliminating redundancy in model parameters, and [21] leveraged coding efficiency by imposing constraints on weight entropy. Furthermore, [30, 2] have extended the frame-wise INR to the patch-wise INR, enabling an improved representation of videos. However, they

Table 1: The performance on NVP [25] drops when the parameter size of temporal axis (T) is decreased while the overall size of parameters ($X \times Y \times T$) is maintained. These results are based on 600-frame HD videos of UVG, and demonstrate a degradation in performance as the T becomes smaller than the video length.

T	600	300	200	100	60
PSNR	36.34	35.59 (-0.75)	33.99 (-2.35)	31.65 (-4.69)	29.47 (-6.87)

still have a limitation in that they cannot be expanded spatially, then can only be decoded at a fixed resolution size.

On the other hand, some approaches try to encode the difference between frames, instead of directly encoding the frames themselves. [22] generates the entire video using warping and upsampling from given compressed keyframes, and [26] reconstructs the final frame by generating a flow map and independent frames between adjacent frames and aggregating them. These methods effectively reduce the temporal redundancy which is inherent in video data, and demostrate outstanding performance in video compression. In addition, [5, 49] suggested that 1D coordinate input $t \in \mathbb{R}^1$ in frame-wise INR was insufficient for accurately modeling the video's context feature. Based on this finding, [5] proposed a structure that integrates context features with a video-specific decoder, whereas [49] established a structure that combines context features with difference features. These studies exhibited superior performance compared to traditional frame-wise INRs.

Despite implementing several structural improvements, subsequent studies on NeRV still exhibit a very slow learning speed. As shown in Figure 1, unlike other models that successfully capture numbers and text in just one minute of training, HNeRV [5] and E-NeRV [27] fail to do the same and exhibits a blurry artifact.

To address these problems, NVP [25] indicates a new direction of pixel-wise INR for video, while achieving a fast encoding speed. It effectively learns the video representation by using parametric encoding (e.g., grid) that are used to improve the learning speed in INR's field [35, 11]. Specifically, by decomposing the 3D coordinates into three 2D coordinates $(x, y), (y, t), (t, x)$ and employing a 3D sparse grid, they successfully trained an pixel-wise implicit video representation framework. However, this approach has a definite limitation as it simply treats videos as 3D data, without considering their dynamic nature at all.

As shown in Table 1, performance degradation occurs if the grid parameters of the sparse 3D grid are not sufficiently secured along the time axis when the parameter size of temporal axis is lower than video length. Particularly, despite the overall parameters remaining the same, as the grid parameters decrease along the time axis, the performance degradation becomes more severe. This implies that it does not properly remove temporal redundancy. In this paper, we propose a fast and parameter-efficient video representation using a grid-type parameter encoding that considers the dynamics of the video. while having 3D coordinate input $(x, y, t) \in \mathbb{R}^3$.

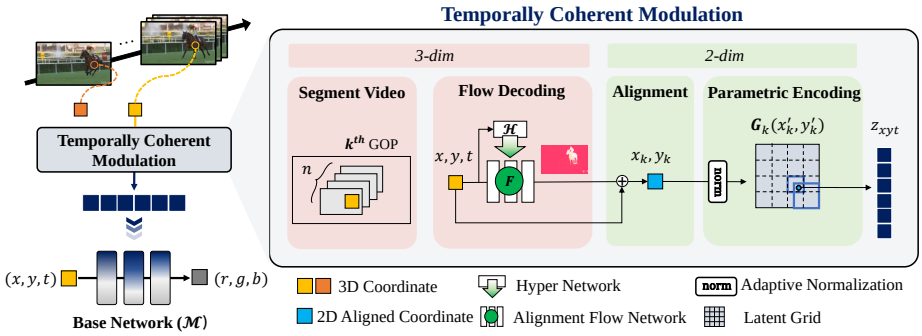


Fig. 2: **Overview of NVTM.** NVTM generates the same modulation latent for temporally correlated pixels between consecutive frames, and the latent is used to modulate the base network. To obtain this latent, 1) input video is split into GOP units, 2) network F generates an alignment flow to transform 3D coordinate (x, y, t) to specific time t_k in k-th GOP unit, 3) 2D aligned coordinated (x_k, y_k) is obtained by adding (x, y) and the alignment flow. 4) The temporally coherent latent (z_{xyt}) is extracted from the latent grid G_k using normalized (x'_k, y'_k). Following the process, the temporally correlated 3D coordinates (yellow square and orange square) are mapped to the same 2D coordinate, thereby ensuring they share the same modulation latent representation. This shared modulation helps in the fast and efficient learning of video representation.

2.3 Modulation for INR

Although INR can represent each specific data instance successfully, it lacks generalization and requires re-training from scratch whenever different data instances are applied. Therefore, unlike conventional paradigm which inputs coordinates and outputs data values, several studies [31, 15, 3] have researched to further modulate network operations. [31] introduced an auxiliary modulator in parallel to the base network, controlling the frequency and phase of the base network to increase its representational power. [15, 3] proposed to learn the instance-specific shift modulation latent, allowing the base network to represent the entire dataset while each shift modulation latent represent each instance. However, alternative approaches [41, 17] utilize a hyper-network, which determines the weights of the base network according to each data, completely altering the operation of the base network. While primary studies focus on an instance-wise modulation, in this paper, we introduce the concept of pixel-wise modulation to represent video data more efficiently.

3 Methodology

3.1 Overall Framework

The overview is described on Figure 2. The base network $\mathcal{M}$ takes the 3D coordinate (x, y, t) as the input and produces (r, g, b) as the output, and this can be expressed as $(r, g, b) = \mathcal{M}(x, y, t)$. The base network $\mathcal{M}$ also takes the latent

z_{xyt}, obtained from each pixel, as a modulation value which affects the network's behavior. The process of computing the latent is detailed in following subsection.

$$(r, g, b) = \mathcal{M}_{\theta(z_{xyt})}(x, y, t). \tag{1}$$

Utilizing the latent z_{xyt} to modulate the base network enables temporally coherent modulation $\mathcal{M}_{\theta(z_{xyt})}$. We adopt the modulated-SIREN [31] as the modulation scheme for the base network.

3.2 Temporally Coherent Modulation

NVTM is composed of an alignment flow network and multiple 2D latent grids. As mentioned previously, our key idea is to group similar pixels together and apply the same modulation, allowing the model to learn the pixel values quickly and sufficiently, even with fewer model parameters.

Segment Video As mentioned before, it is needed to group similar pixels within the video. However, grouping all the pixels in the video is challenging and not effective. Therefore we aim to segment the input video into group-of-pictures (GOPs) of size n and match corresponding pixels along time within each GOP. The number of GOPs, m, is calculated by dividing the total number of video frames with n. k-th GOP is composed of $\{t|(k-1)n \leq t < kn\}$-th frames, and the grouping alignment will executed within each GOP unit.

Flow Decoding To match corresponding pixels, we align 3D coordinates into 2D coordinates at a specific time t_k, which is defined each GOP unit. We refer to this time as the keyframe time, and used the first frame of the each GOP unit as the keyframe in this work. For this process, the network F generates a flow from input time t to keyframe time t_k for each 3D coordinate input (x, y, t).

$$\text{Flow}_{t \to t_k}(x, y) = F(x, y, t) \tag{2}$$

Since this alignment flow $F(x, y, t)$ is likely to be similar the optical flow from time t towards keyframe time t_k, we utilize the optical flow as a guidance to train F, and we add an auxiliary loss at the beginning steps of training. In the decoding phase, the optical flow is not necessary as the output of F is used directly.

Meanwhile, the optical flow appears to move spatially over time, reflecting the movements of objects in video sequences [17]. Based on this flow observation, we make network F be influenced from t for easily learning flows with fewer parameters. For this, we adopt a hyper-SIREN [17] as $F_{\mathcal{H}(t)}(x, y)$, which uses a hyper-network $\mathcal{H}(t)$ to generate SIREN F's weights over time t. Further, to offset the difference in flow scale caused by the interval to t_k at each t, we incorporated the log scale factor into the output scaling of the model.

$$\text{Flow}_{t \to t_k}(x, y) = \log(t - t_k) F_{\mathcal{H}(t)}(x, y) \tag{3}$$

Alignment Using the alignment flow, each 3D coordinate (x, y, t) is warped into the 2D aligned coordinate (x_k, y_k) of keyframe time t_k.

$$(x_k, y_k) = (x, y) + \log(t - t_k) F_{\mathcal{H}(t)}(x, y) \tag{4}$$

Parametric Encoding Parametric encoding takes the form of extracting values from a parameter group corresponding to each input. We utilize a parameter group structured as a 2D-grid type for each GOP and designate as the latent grid G_k for the k-th GOP. The 2D aligned coordinate (x_k, y_k) serves as the input to G_k. The input of the latent grid must satisfy $\in [0, 1]$. However, given that both alignment flow and the initial 3D coordinate (x, y, t) range in $[0, 1]$, their sum, which results in (x_k, y_k), may not satisfy this condition. To adjust them, some naive approaches such as clipping or simple re-normalization $(x_k - \min(x_k))/(\max(x_k) - \min(x_k))$ can be considered, but they have unacceptable side effects. Clipping occurs information loss and re-normalization decreases grid parameter efficiency.

Therefore, we propose an adaptive normalization method, which can optimize the spatial utilization of the grid while containing the maximum amount of information. We search the largest area with a higher pixel density than the predefined threshold r_{th} and define the area as $\{x_k^{min}, y_k^{min}, x_k^{max}, y_k^{max}\}$. Finally, (x_k, y_k) are normalized into (x'_k, y'_k) using the calculated min and max values.

$$\begin{aligned} x'_k &= \text{Clip}\{(x_k - x_k^{min})/(x_k^{max} - x_k^{min}), (0, 1)\} \\ y'_k &= \text{Clip}\{(y_k - y_k^{min})/(y_k^{max} - y_k^{min}), (0, 1)\} \end{aligned} \tag{5}$$

This adaptive normalization ensures that areas with a high pixel occupancy are properly normalized, whereas sparse regions are effectively handled by clipping the coordinates.

Modulation latent is obtained from the normalized coordinate, as $z_{xyt} = G_k(x'_k, y'_k)$. Additionally, we extend modulation latent to utilizing two or more latent grids of neighboring GOPs. We define a neighbor index set P, and the final latent is obtained by concatenating all latents computed from the latent grids $\{G_k, G_{k+1}, .., G_{k+p}\}$ of the neighboring GOPs belonging to $P = \{0, 1, .., p\}$.

$$z_{xyt} = \text{concat}\{G_{k+p}(x'_{k+p}, y'_{k+p}) | p \in P\} \tag{6}$$

Loss Total loss is a combination of the reconstruction loss and the auxiliary loss with the weight factor w_{aux}. The reconstruction loss L_{recon} is the Mean Squared Error (MSE) between the original and reconstructed pixels, while auxiliary loss L_{aux} is the MSE between the alignment flow and the optical flow. Then the total loss is calculated as $L_{total} = L_{recon} + w_{aux} \cdot L_{aux}$.

4 Experimental Results

4.1 Implementation Details

Dataset. We conduct experiments on UVG [32] and MCL-JCV [45] datasets, which are widely used in various video tasks such as compression and quality

Table 2: Encoding speed on video reconstruction. All models are configured as 0.1bpp and we compare their reconstruction performance (PSNR) based on the encoding time (i.e., the training time). **Bold** values represent the best value for each encoding time, and evaluated epoch or step (e/s) of each model is denoted.

UVG (Dynamic)

Models	Encoding time				
	∼ 1min.	∼ 5min.	∼ 10min.	∼ 20min.	∼ 60min.
NeRV [6]	$18.10_{/1e}$	$21.84_{/9e}$	$22.01_{/18e}$	25.27	30.88
E-NeRV [27]	$16.66_{/1e}$	$17.20_{/5e}$	$19.02_{/10e}$	24.38	30.00
HNeRV [5]	$20.73_{/2e}$	$24.39_{/13e}$	$27.22_{/26e}$	28.64	**32.51**
NVP [25]	$23.37_{/250s}$	$28.48_{/1250s}$	$28.93_{/2500s}$	30.39	31.40
NVTM (Ours)	$\mathbf{26.52}_{/111s}$	$\mathbf{29.20}_{/556s}$	$\mathbf{29.97}_{/1111s}$	**30.49**	31.85

MCL-JCV (Dynamic)

Models	Encoding time				
	∼ 1min.	∼ 5min.	∼ 10min.	∼ 20min.	∼ 60min.
NeRV [6]	$19.09_{/10e}$	$21.93_{/50e}$	$23.10_{/100e}$	24.67	28.33
E-NeRV [27]	$16.26_{/7e}$	$16.67_{/36e}$	$17.80_{/72e}$	23.76	28.11
HNeRV [5]	$20.66_{/12e}$	$24.36_{/63e}$	$26.30_{/126e}$	29.73	32.27
NVP [25]	$25.34_{/294s}$	$29.26_{/1471s}$	$29.41_{/2941s}$	31.13	32.53
NVTM (Ours)	$\mathbf{27.71}_{/250s}$	$\mathbf{30.85}_{/1250s}$	$\mathbf{31.65}_{/2500s}$	**32.09**	**33.57**

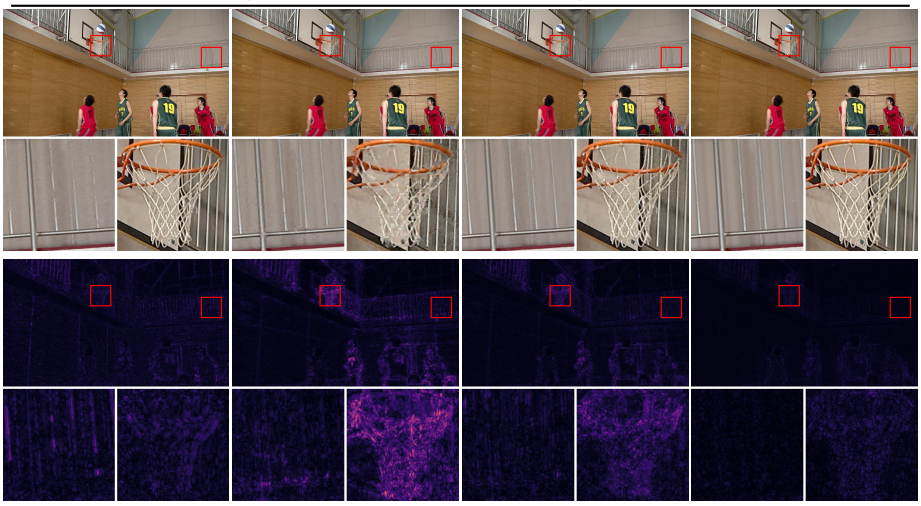
assessment. Since our proposed approach is designed for videos which contain temporal dynamic information, we target on dynamic sequences among them. Hence, we select 4 sequences from UVG and 5 sequences from MCL-JCV, which have large motion and sufficient spatial/temporal information. We convert those from raw YUV videos into RGB format and use the complete set of 600 frames for each sequence in UVG HD and initial 100 frames for each sequence in MCL-JCV HD. Details of statistics and data procedures are described in supplementary.

Model configuration and training details. These are our default experiment setting and more exploration are experimented on Section 4.5. We configure NVTM as a default setting with a GOP size n as 10. And we configure the index set P as {0, 1}. To capture some static characteristics of video (e.g., still images), we additionally add a single 2D grid as static feature, similar to NVP [25]. We utilize RAFT [43] to generate the optical flows, and set w_{aux} as 0.5 for auxiliary loss of the alignment flow network. We use threshold value r_{th} as 0.5 for adaptive normalization to ensure that at least half of the area is considered effective. All experiments are conducted on a single NVIDIA A100 GPU. More details are described in supplementary materials.

Evaluation. We evaluate with Peak Signal-to-Noise Ratio (PSNR) and Learned Perceptual Image Patch Similarity (LPIPS) [48]. We compare our model both with grid-type models (fast encoding time) and NeRV-style models (efficient

Table 3: Video reconstruction performance with grid-type models. Each value represents the average on UVG and MCL-JCV respectively. **Bold** is the best value and * indicates that ours uses 10% fewer parameters compared to other methods. We display the video reconstruction visualizations on videoSRC05 sequence as the pairs of decoded image and crop-zoomed images. The below images are visualization of FLIP [1] calculated from the original frame, the bright regions represent errors, while the darker colors indicate better performance.

Model	UVG (Dynamic)			MCL-JCV (Dynamic)		
	Params.	PSNR↑	LPIPS↓	Params.	PSNR↑	LPIPS↓
Instant-NGP [35]	145M	37.08	0.126	29M	39.32	0.093
3D ModSIREN [31]	134M	37.24	0.095	29M	36.96	0.134
NVP [25]	136M	39.00	0.090	29M	39.55	0.093
NVTM (Ours)	122M*	**40.54**	**0.071**	28M	**41.39**	**0.080**



Instant-NGP [35] 3D ModSIREN [31] NVP [25] NVTM (Ours)

parameter size). NeRV-style models, including NeRV [6], E-NeRV [27] and HNeRV [5], are reproduced by author's implementation. Since they have low bpp-levels, we also evaluate ours with 0.1bpp and compare with them. This experiment demonstrate real-world video settings, as Netflix recommends 5Mbps[1] as the minimum speed for Full HD video streaming, and 0.1bpp corresponds to a closely aligned bitrate of 4.97Mbps at 24 fps. In addition, we experiment with grid-type models, including 3D ModSIREN, Instant-NGP [35] and NVP [25]. 3D ModSIREN refers to the use of 3D grid as modulation latents in modulated-SIREN [31] without any dimension reduction, Instant-NGP is implemented by adjusting network size and NVP is reproduced according to author setting. Since the performance is dependent on target video scale, we additionally design them as a smaller parameter size for smaller resolution or short video length settings.

[1] https://help.netflix.com/en/node/306

4.2 Video Reconstruction: Encoding Speed

To apply on practical service, INR method must be quickly encoded. Then we first report the performance of the models when trained for 1/5/10/20/60 minutes under the same resource conditions on Figure 1 and Table 2. All models are configured in 0.1bpp, as following their bit range. Our model prominently exhibits fast encoding and quickly reaches over 30dB compared to the NeRV series. NVP, which also uses grid-based parametric encoding, encodes quickly but its performance is inferior to ours.

4.3 Video Reconstruction: Parameter Efficiency

We also compared the performance of NVTM with other parametric encoding methods to explore how parameter efficient it is. Table 3 shows that NVTM outperforms on various video sequences. NVTM has 1.54dB/0.019 improvements of PSNR/LPIPS even with 10% fewer parameters on UVG (Dynamic), and 1.84dB/0.013 improvements on MCL-JCV (Dynamic). From qualitative comparison on decoded images, we can observe how well our model preserves the fine details such as the thin iron bar and the basketball hoop. These results can be interpreted as NVTM has better parameter efficiency by dealing the coherent information of consecutive frames.

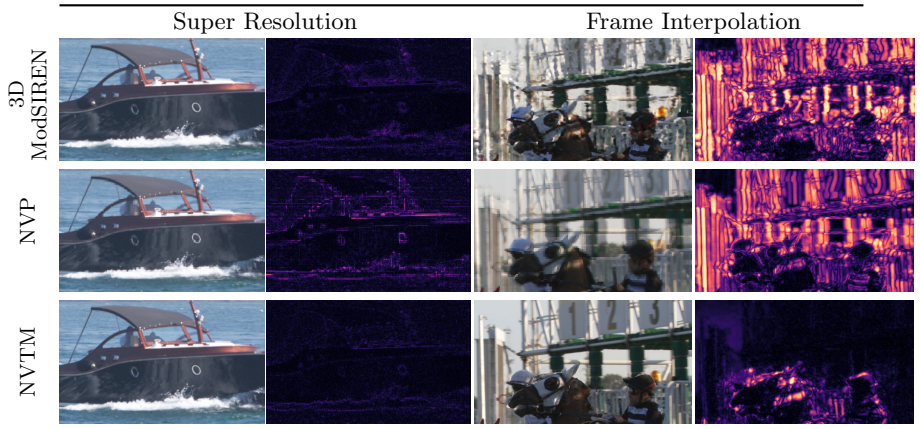
4.4 Downstream Tasks

Video super resolution and frame interpolation. One of the major advantages of INR is its capability to capture intermediary points in both temporal and spatial dimensions. For video super resolution, we decode all models with doubled spatial coordinate and evaluate with early-defined 4K resolution videos (T,H,W→T,2H,2W). Similarly, for video frame interpolation, we first train models with odd number images and decode them with doubled temporal coordinate and evaluate with original video sequence (T/2,H,W→T,H,W). The evaluated results are in Table 4, NVTM shows much fewer errors for both intermediate spatial and temporal values than others. Meanwhile, since 3D ModSIREN can densely encode pixels utilizing 3D coordinates directly, it might be slightly advantageous in generating intermediate values and outperforms NVP. However, our approach, despite not using 3D-shaped grid parameters, demonstrates impressive results, indicating its successful decomposition of 3D video data.

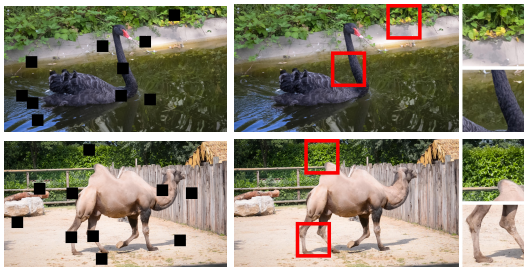
Video inpainting. We further explore the potential of NVTM in the video inpainting task. We use DAVIS2017 [36] HD dataset. We conduct a random box experiment, training with random box masked images and targeting to reconstruct the complete frames, as previous works [25]. We generate masked images with 10 random boxes masking with 100×100 sized on every frame. As seen in Figure 5, our method exhibits a remarkable restoration performance on the masked regions. This highlights that our representation is learned as aggregated by reference on similar pixels along the temporal axis.

Table 4: Video super resolution and frame interpolation on UVG (Dynamic). We expand spatial coordinates (×2) and temporal coordinates (×2) respectively on decoding time. Below figures are cropped images and FLIP visualizations from super resolution results on Bosphorus sequence (left) and frame interpolation results on Jockey sequence (right). The bright regions in FLIP figures represent errors, while the darker colors indicate better performance.

Model	Super Resolution		Frame Interpolation	
	PSNR↑	LPIPS↓	PSNR↑	LPIPS↓
3D ModSIREN [31]	34.72	0.265	25.41	0.221
NVP [25]	31.87	0.396	23.88	0.394
NVTM (Ours)	**35.82**	**0.240**	**30.49**	**0.134**



(a) Video Inpainting



(b) Video Compression

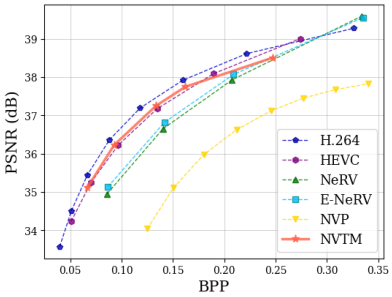


Fig. 3: Video inpainting and compression performance. (a) Visualization of video inpainting on Blackswan and Camel sequences in DAVIS2017. Although the masked regions are excluded during encoding, the NVTM successfully decodes them by utilizing temporally coherent modulation latent from adjacent frames. (b) BPP-PSNR plot of video compression on UVG (Dynamic). We encode all models with each video sequence and evaluate as following authors guided.

Video compression. Video compression is one of main applications in INR for video [6, 27, 25, 5, 49, 22, 21], and they attempt to prune, quantize, or

Table 5: Ablation study on framework design in UVG (Dynamic).

(a) Framework design

Adaptive norm.	Static feature	PSNR
x	x	40.24
✓	x	40.36
✓	✓	**40.54**

(b) Latent process

Modulation	PSNR
x	38.90
✓	**40.54**

(c) Index set

P	PSNR
$\{0\}$	39.45
$\{0, 1\}$	**40.54**

Table 6: Ablation study on GOP size. The performance for each video sequence vary depending on the GOP size. As the smaller size of GOP results in dividing video sequences into more segments for coordinate alignment, we modify the model configuration to ensure similar overall model parameters in each experiment.

GOP	Bosphorus	Jockey	ReadySetGo	YachtRide
5	43.27	**40.41**	**39.80**	**39.57**
10	**43.30**	40.17	39.70	38.88
20	43.00	39.29	38.54	35.65
60	40.34	35.39	33.50	32.53

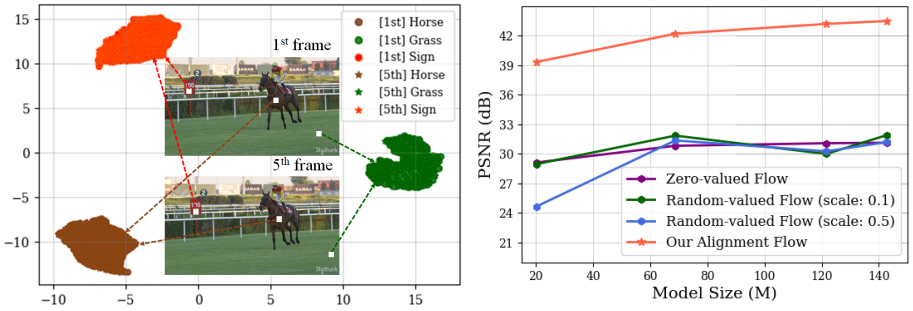
Table 7: Temporal scalability on video length. Experiments on video lengths exceeding 600 frames are conducted on videos composed of concatenated sequences, each labeled according to the initial letter of sequence names.

Model	100	200	300	600	1200 (B+J)	2400 (B+J+R+Y)
3D ModSIREN [31]	38.48	39.37	40.06	40.92	37.37	38.82
NVP [25]	41.23	41.08	41.12	41.47	40.23	40.21
NVTM (Ours)	**43.70**	**43.54**	**43.50**	**43.41**	**41.54**	**41.05**

compress the model parameters after training. We also compress model parameter by applying existing codecs, as grid-type INR approaches tried [25, 35]. Especially, since our model is decomposed with a series of 2D grids notated as $\mathrm{G} := (G_1, ..., G_m)$, we applied HEVC video compression on the grid parameters and further compress effectively. In Figure 3b, we compare our model with standard video codecs (H.264 [46], HEVC [40]) and state-of-the-art methods [6, 27, 25, 22]. The details of the evaluation are described in supplementary. NVTM demonstrates compression performance similar or slightly better than [6, 27] which have lower training speeds, and notably outperforms [25] which has faster training speed as ours. Here, we can confirm the superiority of NVTM when considering both parameter-efficiency and computing-efficiency.

4.5 Ablations Studies

Framework design We conduct ablation studies on our framework modules. Table 5a demonstrates the positive impacts of adaptive normalization and static features, with improvements of 0.12dB and 0.19dB respectively. We compare the



(a) Temporal consistency analysis (b) Alignment flow analysis

Fig. 4: (a) t-SNE visualization of modulation latent z_{xyz} from our alignment module on corresponding pixels ($1st$ and $5th$ frame). We select areas with similar pixel information, i.e. RGB values, and for ease of verification, these are denoted as {Horse, Grass, Sign}. The latent derived from the $1st$ frame and $5th$ are marked with circle and star respectively. The analysis is based on segments, each consisting of 400 pixels. (b) Effects of alignment flow. Each line represents the performance with replacing our alignment method on Bosphorus sequence. **Purple** indicates aligning with zero-valued flow (i.e., its spatial coordinate). **Green** and **blue** indicates aligning with random-valued flow in a notated scale of source video resolution.

effectiveness of using latent as a modulation to the base network versus using it as a direct input. Table 5b indicates that using it as a modulation is more effective for representing video. Also we experiment on effect of neighbor index set P on Table 5c.

GOP size Our framework uses a fixed GOP size to divide the video into segments. We experiment with different GOP sizes on Table 6. From our analysis about the degree of motion for each sequence (described on Section A.2), we find that some sequences with relatively large motion energy exhibit improved performance when the video is divided more finely with a GOP value of 5. Conversely, for sequence which has relatively small motion, a GOP value of 10 yielded better performance. From this tendency, we believe that NVTM can achieve better performance if it uses variable size of GOP.

Video duration We verify temporal scalability that NVTM consistently achieves the standout performance across various video lengths on Table 7.

4.6 Analysis

Temporal Consistency Modulation We propose that by assigning the same modulation latent (z_{xyz}) to similar pixels across consecutive frames, the network could learn more rapidly and achieve higher performance. To confirm this, we analysis on z_{xyz} corresponding to pixels that appeared to be similar in Figure 4.

We can observe that the latent values derived from similar pixel areas across different frames are represented as similar embeddings. These findings validate our intention that our network produces identical modulation latents from similar pixels in consecutive frames.

Alignment Flow We propose to align the 3-dimensional (x, y, t) to the 2-dimensional (x, y), using an alignment flow derived from Equation 3. To validate the effectiveness and usefulness of this method, we compare it with other alignment methods in Figure 4b. Both zero-valued flow and the random-valued flow, unlike our method, does not consider motion or pixel similarity, and simply map the video into 2D. From the results, we can verify that the proposed method demonstrated sufficient performance (much over 30dB) even with fewer parameters, whereas other methods were significantly deficient in performance.

5 Conclusion

In this study, we proposed a novel approach for implicit neural video representation, which involves temporal coordinate alignment and modulation latent encoding to effectively capture video dynamics at the pixel level. Extensive experiments verified that the NVTM outperforms existing methods of implicit neural video representation on various video related tasks. We anticipate that our framework will provide inspiration for the follower on INR for videos.

Bibliography

[1] Andersson, P., Nilsson, J., Akenine-Möller, T., Oskarsson, M., Åström, K., Fairchild, M.D.: Flip: A difference evaluator for alternating images. Proc. ACM Comput. Graph. Interact. Tech. **3**(2), 15–1 (2020)

[2] Bai, Y., Dong, C., Wang, C., Yuan, C.: Ps-nerv: Patch-wise stylized neural representations for videos. In: 2023 IEEE International Conference on Image Processing (ICIP). pp. 41–45. IEEE (2023)

[3] Bauer, M., Dupont, E., Brock, A., Rosenbaum, D., Schwarz, J.R., Kim, H.: Spatial functa: Scaling functa to imagenet classification and generation. arXiv preprint arXiv: 2302.03130 (2023)

[4] Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16. pp. 608–625. Springer (2020)

[5] Chen, H., Gwilliam, M., Lim, S.N., Shrivastava, A.: Hnerv: A hybrid neural representation for videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023)

[6] Chen, H., He, B., Wang, H., Ren, Y., Lim, S.N., Shrivastava, A.: Nerv: Neural representations for videos. Advances in Neural Information Processing Systems **34**, 21557–21568 (2021)

[7] Chen, Y., Liu, S., Wang, X.: 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 (2021)

[8] Chen, Z., Chen, Y., Liu, J., Xu, X., Goel, V., Wang, Z., Shi, H., Wang, X.: 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)

[9] Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 6970–6981 (2020)

[10] Chu, M., Xie, Y., Mayer, J., Leal-Taixé, L., Thuerey, N.: Learning temporal coherence via self-supervision for gan-based video generation. ACM Transactions on Graphics (TOG) **39**(4), 75–1 (2020)

[11] Deng, C.L., Tartaglione, E.: Compressing explicit voxel grid representations: fast nerfs become also small. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1236–1245 (2023)

[12] Dong, Z., Guo, C., Song, J., Chen, X., Geiger, A., Hilliges, O.: Pina: Learning a personalized implicit neural avatar from a single rgb-d video sequence. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 20470–20480 (2022)

[13] Dupont, E., Loya, H., Alizadeh, M., Golinski, A., Teh, Y., Doucet, A.: Coin++: neural compression across modalities. Transactions on Machine Learning Research **2022**(11) (2022)
[14] Dupont, E., Golinski, A., Alizadeh, M., Teh, Y.W., Doucet, A.: Coin: Compression with implicit neural representations. In: Neural Compression: From Information Theory to Applications–Workshop@ ICLR 2021 (2021)
[15] Dupont, E., Kim, H., Eslami, S.A., Rezende, D.J., Rosenbaum, D.: From data to functa: Your data point is a function and you can treat it like one. In: International Conference on Machine Learning. pp. 5694–5725. PMLR (2022)
[16] Fang, S., Xu, W., Wang, H., Yang, Y., Wang, Y., Zhou, S.: One is all: Bridging the gap between neural radiance fields architectures with progressive volume distillation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 37, pp. 597–605 (2023)
[17] Figueirêdo, P., Paliwal, A., Kalantari, N.K.: Frame interpolation for dynamic scenes with implicit flow encoding. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 218–228 (2023)
[18] Fridovich-Keil, S., Yu, A., Tancik, M., Chen, Q., Recht, B., Kanazawa, A.: Plenoxels: Radiance fields without neural networks. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5501–5510 (2022)
[19] Genova, K., Cole, F., Sud, A., Sarna, A., Funkhouser, T.: Local deep implicit functions for 3d shape. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4857–4866 (2020)
[20] Girish, S., Shrivastava, A., Gupta, K.: Shacira: Scalable hash-grid compression for implicit neural representations. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 17513–17524 (2023)
[21] Gomes, C., Azevedo, R., Schroers, C.: Video compression with entropy-constrained neural representations. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18497–18506 (2023)
[22] He, B., Yang, X., Wang, H., Wu, Z., Chen, H., Huang, S., Ren, Y., Lim, S.N., Shrivastava, A.: Towards scalable neural representation for diverse videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6132–6142 (2023)
[23] Installations, T., Line, L.: Subjective video quality assessment methods for multimedia applications. Networks **910**(37), 5 (1999)
[24] Jiang, C., Sud, A., Makadia, A., Huang, J., Nießner, M., Funkhouser, T., et al.: Local implicit grid representations for 3d scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6001–6010 (2020)
[25] Kim, S., Yu, S., Lee, J., Shin, J.: Scalable neural video representations with learnable positional features. arXiv preprint arXiv:2210.06823 (2022)
[26] Lee, J.C., Rho, D., Ko, J.H., Park, E.: Ffnerv: Flow-guided frame-wise neural representations for videos. In: Proceedings of the 31st ACM International Conference on Multimedia. pp. 7859–7870 (2023)

[27] Li, Z., Wang, M., Pi, H., Xu, K., Mei, J., Liu, Y.: E-nerv: Expedite neural video representation with disentangled spatial-temporal context. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXV. pp. 267–284. Springer (2022)
[28] Liu, L., Gu, J., Zaw Lin, K., Chua, T.S., Theobalt, C.: Neural sparse voxel fields. Advances in Neural Information Processing Systems **33**, 15651–15663 (2020)
[29] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
[30] Maiya, S.R., Girish, S., Ehrlich, M., Wang, H., Lee, K.S., Poirson, P., Wu, P., Wang, C., Shrivastava, A.: Nirvana: Neural implicit representations of videos with adaptive networks and autoregressive patch-wise modeling. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2023)
[31] Mehta, I., Gharbi, M., Barnes, C., Shechtman, E., Ramamoorthi, R., Chandraker, M.: Modulated periodic activations for generalizable local functional representations. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 14214–14223 (2021)
[32] Mercat, A., Viitanen, M., Vanne, J.: Uvg dataset: 50/120fps 4k sequences for video codec analysis and development. In: Proceedings of the 11th ACM Multimedia Systems Conference. pp. 297–302 (2020)
[33] Mildenhall, B., Srinivasan, P., Tancik, M., Barron, J., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: European conference on computer vision (2020)
[34] Müller, T.: tiny-cuda-nn (4 2021), `https://github.com/NVlabs/tiny-cuda-nn`
[35] Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Transactions on Graphics (ToG) **41**(4), 1–15 (2022)
[36] Pont-Tuset, J., Perazzi, F., Caelles, S., Arbeláez, P., Sorkine-Hornung, A., Van Gool, L.: The 2017 davis challenge on video object segmentation. arXiv:1704.00675 (2017)
[37] Rho, D., Cho, J., Ko, J.H., Park, E.: Neural residual flow fields for efficient video representations. In: Proceedings of the Asian Conference on Computer Vision. pp. 3447–3463 (2022)
[38] Sitzmann, V., Martel, J., Bergman, A., Lindell, D., Wetzstein, G.: Implicit neural representations with periodic activation functions. Advances in Neural Information Processing Systems **33**, 7462–7473 (2020)
[39] Su, K., Chen, M., Shlizerman, E.: Inras: Implicit neural representation for audio scenes. Advances in Neural Information Processing Systems **35**, 8144–8158 (2022)
[40] Sullivan, G.J., Ohm, J.R., Han, W.J., Wiegand, T.: Overview of the high efficiency video coding (hevc) standard. IEEE Transactions on circuits and systems for video technology **22**(12), 1649–1668 (2012)
[41] Szatkowski, F., Piczak, K.J., Spurek, P., Tabor, J., Trzciński, T.: Hypersound: Generating implicit neural representations of audio signals with hypernetworks. arXiv preprint arXiv:2211.01839 (2022)

[42] Takikawa, T., Litalien, J., Yin, K., Kreis, K., Loop, C., Nowrouzezahrai, D., Jacobson, A., McGuire, M., Fidler, S.: Neural geometric level of detail: Real-time rendering with implicit 3d shapes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11358–11367 (2021)
[43] Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16. pp. 402–419. Springer (2020)
[44] Tomar, S.: Converting video formats with ffmpeg. Linux journal **2006**(146), 10 (2006)
[45] Wang, H., Gan, W., Hu, S., Lin, J.Y., Jin, L., Song, L., Wang, P., Katsavounidis, I., Aaron, A., Kuo, C.C.J.: Mcl-jcv: a jnd-based h. 264/avc video quality assessment dataset. In: 2016 IEEE international conference on image processing (ICIP). pp. 1509–1513. IEEE (2016)
[46] Wiegand, T., Sullivan, G.J., Bjontegaard, G., Luthra, A.: Overview of the h. 264/avc video coding standard. IEEE Transactions on circuits and systems for video technology **13**(7), 560–576 (2003)
[47] Yang, J., Zheng, W.S., Yang, Q., Chen, Y.C., Tian, Q.: Spatial-temporal graph convolutional network for video-based person re-identification. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3289–3299 (2020)
[48] Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: 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)
[49] Zhao, Q., Asif, M.S., Ma, Z.: 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)