DNeRV: Modeling Inherent Dynamics via Difference Neural Representation for Videos

Qi Zhao
Nanjing University
qizhao@smail.nju.edu.cn

M. Salman Asif
University of California Riverside
sasif@ucr.edu

Zhan Ma*
Nanjing University
mazhan@nju.edu.cn

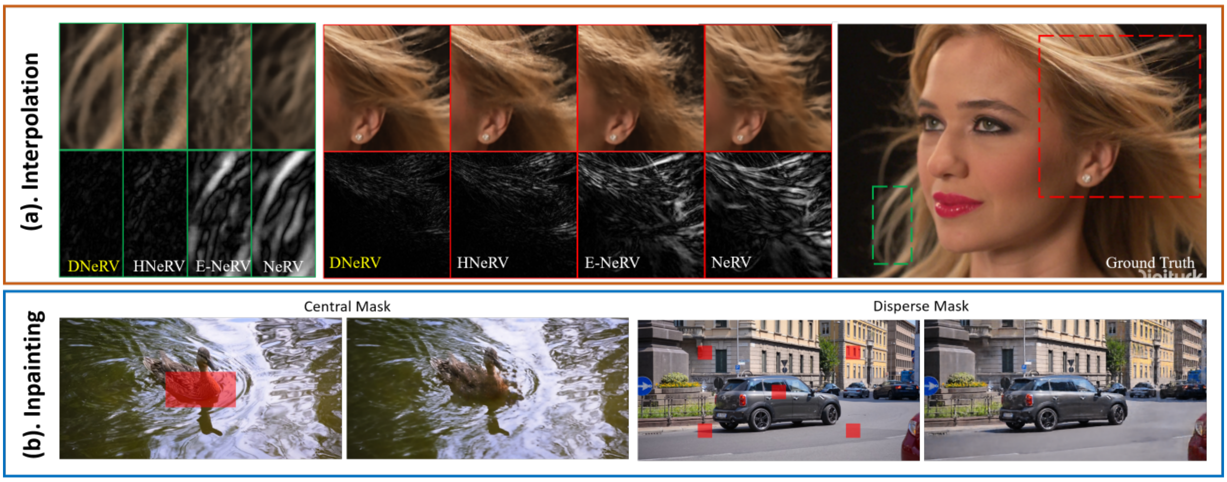

Figure 1. Results of the proposed DNeRV with 3M parameters for (a) video interpolation on UVG [26] and (b) video inpainting on Davis [30]. The superior performance shows the effectiveness and generalization capability of DNeRV on neural representation for videos.

Abstract

Existing implicit neural representation (INR) methods do not fully exploit spatiotemporal redundancies in videos. Index-based INRs ignore the content-specific spatial features and hybrid INRs ignore the contextual dependency on adjacent frames, leading to poor modeling capability for scenes with large motion or dynamics. We analyze this limitation from the perspective of function fitting and reveal the importance of frame difference. To use explicit motion information, we propose Difference Neural Representation for Videos (DNeRV), which consists of two streams for content and frame difference. We also introduce a collaborative content unit for effective feature fusion. We test DNeRV for video compression, inpainting, and interpolation. DNeRV achieves competitive results against the state-of-the-art neural compression approaches and outperforms existing implicit methods on downstream inpainting and interpolation for 960×1920 *videos.*

1. Introduction

In recent years, implicit neural representations (INR) have gained significant attention due to their strong ability in learning a coordinate-wise mapping of different functions. The main principle behind INR is to learn an implicit continuous mapping f using a learnable neural network $g_\theta(\cdot) : \mathbb{R}^m \to \mathbb{R}^n$. The idea was first proposed for the neural radiance fields (NeRF) [28] and since then has been applied to various applications [4, 8, 58]. INR attempts to approximate the continuous f by training g_θ with m-dimensional discrete *coordinates* $\mathbf{x} \in \mathbb{R}^m$ and corresponding quantity of interest $\mathbf{y} \in \mathbb{R}^n$. Once trained, the desired f can be fully characterized using g_θ or the weights θ, and it would be benefit for the tasks which need to model the intrinsic generalization for given data, such as interpolation or inpainting tasks shown in Fig. 1.

The success of INR can be attributed to the insight that a learnable and powerful operator with a finite set of data samples $\mathcal{S} = \{x_i, y_i\}_{i=0}^N$, can fit the *unknown* mapping f. The accuracy of the mapping depends on the number of samples (N) and the complexity of the map f. INR for

[1]Corresponding author: Zhan Ma (mazhan@nju.edu.cn)

Figure 2. Examples of neighboring frames with large mismatch. Learning continuous INR with such dynamics is challenging.

videos requires a large N, which primarily depends on the size and internal complexity of the video sequence. Furthermore, video representation is complicated due to different sampling or frames-per-second (FPS) rates of videos. Large motion (in terms of direction, speed, rotation, or blur) and transformations of the objects or scene can make adjacent frames quite different. Figure 2 shows examples of such mismatch between consecutive frames, which we attribute to *adjacent dynamics*.

Adjacent dynamics are the short-term transformations in the spatial structure, which are difficult to represent using existing methods for neural representation of videos (NeRV). Existing NeRV approaches can be broadly divided into two groups: (1) *Index-based* methods, such as [4] and [21], use positional embedding of the index as input and lack content-specific information for given videos. (2) *Hybrid-based* methods [3] use frames for index embedding and neglect the temporal correlation between different frames. Therefore, neither index nor frame-based NeRV are effective against adjacent dynamics.

In this work, we propose Difference NeRV (DNeRV) that attempts to approximate a *dynamical system* by absorbing the difference of adjacent frames, $\mathbf{y}_t^D = \mathbf{y}_t - \mathbf{y}_{t-1}$ and $\mathbf{y}_{t+1}^D = \mathbf{y}_{t+1} - \mathbf{y}_t$, as a diff stream input. Further analysis for the importance of diff stream is presented in Section 3. An illustration of DNeRV pipeline is presented in Figure 3. Diff encoder captures short-term contextual correlation in the diff stream, which is then merged with the content stream for spatiotemporal feature fusion. In addition, we propose a novel gated mechanism, collaborative content unit (CCU), which integrates spatial features in the content stream and temporal features in the diff stream to obtain accurate reconstruction for those frames with adjacent dynamics.

The main contribution of this paper are as follows.

- Existing NeRV methods cannot model content-specific features and contextual correlations simultaneously. We offer an explanation using adjacent dynamics. Furthermore, we reveal the importance of diff stream through heuristic analysis and experiments.
- We propose the Difference NeRV, which can model the content-specific spatial features with short-term temporal dependence more effectively and help network fit the implicit mapping efficiently. We also propose a collaborative content unit to merge the features from two streams adaptively.
- We present experiments on three datasets (Bunny, UVG, and Davis Dynamic) and various downstream tasks to demonstrate the effectiveness of the proposed method. The superior performance over all other implicit methods shows the efficacy of modeling videos with large motion. As a result, DNeRV can be regarded as a new baseline for INR-based video representation.

2. Related Work

Implicit neural representations (INRs) have been used in various vision tasks in recent years [6, 28]. In 3D vision, [24, 27, 31, 51, 51] aim to use INRs from static and simple to dynamic and complicated visual data. In image analysis, INRs have been used to learn the mapping between 2D spatial coordinates $\mathbf{x} \in \mathbb{R}^2$ and corresponding RGB value $\mathbf{y} \in \mathbb{R}^3$ via various positional embedding techniques [9, 39, 43, 55] or meta-learning [40]. In video analysis, INRs learn the mapping from frame index $x_t \in \mathbb{R}$ to RGB frame $\mathbf{y} \in \mathbb{R}^{3 \times w \times h}$ [4, 5, 25]. In other visual tasks, INRs can encode both spatial coordinates and specific feature vectors [36, 53].

Neural representation for videos (NeRV) methods can be broadly divided into two groups. *Index-based* NeRV methods use the positional embedding of t as the input [4, 21, 25]. The follow-up work has improved network structure for acceleration or disentangled feature modeling, but those methods could not capture the content-specific information, causing spatial redundancy. Hybrid-based NeRV [3] provides an insightful view by treating the current frame itself as the index embedding. The method shows much better performance over index-based ones via content-adaptive fashion. The main limitation of hybrid NeRV is that they ignore temporal relationship between frames, resulting in poor performance with adjacent dynamics.

Video compression methods based on INR use the traditional pipeline but change some intermediate components into networks [15, 19, 22, 23, 33]. However, their utility is limited by large number of parameters and computational redundancy, causing long coding time and limited generalized ability. One significance of INRs for video is that the video compression can be regarded as a model compression problem, and the existing model compression and acceleration techniques [11, 12] could be used for INR-based network. Neural compression is expected to break through the limitation of traditional pipeline with the help of INRs, resulting in better performance.

Two stream vision and fusion resemble the idea of adopting multi-stream mechanism for disentangling feature learning in video analysis. [35] observe the importance of optical flow, which is widely used in action recogni-

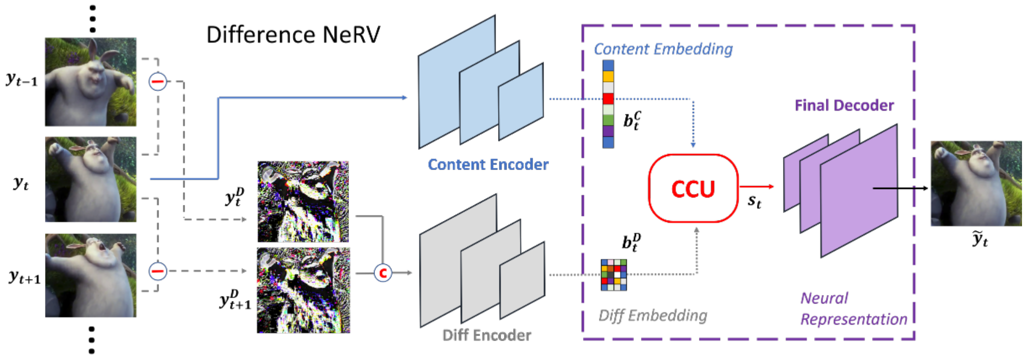

Figure 3. The pipeline of DNeRV. Blue part indicates content stream, grey for diff stream and red for fusion. The purple part is the implicit neural representation for the given video, consisting of embeddings, lightweight CCU, and decoder.

tion [2, 10, 38] or other visual tasks [16, 42]. Separating motion and content as two stream is also widely used in video prediction [44] and image-to-video generation [57]. Furthermore, hidden state and time-dependent input in RNN/LSTM could be thought as different stream [7, 14, 46]. More effective fusion module for different stream information is a fundamental task in sequence learning, generating lots of novel gated mechanism [13, 45, 56]. The two-stream fusion aims to reduce the *spatiotemporal redundancy* caused by the inherent continuity of video, which motivated us to propose DNeRV based on diff stream. Additionally, inspired by the gated mechanism in RNN, we introduce a fusion unit for adaptive feature fusion.

3. Motivation

In this section, we discuss how the frame differences benefit the adjacent dynamics. A video sequence is a collection of images captured over time; therefore, motion of different objects in video is often spatially continuous and smooth. The main objective of INR is to learn an implicit mapping $f(\mathbf{x}) \mapsto \mathbf{y}$. Thus, we usually assume that training tuples $(\mathbf{x}, \mathbf{y})$ are dense enough so that the continuous f can be approximated. In the case of NeRV, the difference between frames (or the so-called adjacent dynamics) violates the expected smoothness. That is the main reason for the failure of NeRV in the presence of large motion and highly dynamic scenes.

For video INRs, the training data are $\{(t, \mathbf{y}_t)\}_{t=0}^{N}$, where $\mathbf{y}_t$ is the frame at time index t. We assume that the implicit (unknown) mapping f is continuous and defined as

$$f : \Phi \to \mathbb{R}^{3\times H\times W}, \quad \Phi = [0, N]. \tag{1}$$

Let us note that although NeRV methods use various positional encoding techniques [39, 43] to map the discrete index into a higher dimension for better performance (w

the frame itself as an embedding of the index for hybrid-based methods, the network g_θ is actually a mapping from a time index in Φ to $\mathbb{R}^{3\times H\times W}$. Once trained, we can generate frames as

$$g_\theta\,(t) = \mathbf{y}_t, \quad t \in \{1, 2, \cdots, N\}. \tag{2}$$

Since video is usually continuous along spatial and temporal dimensions, f can also be described as a dynamical system:

$$\dot{f} = A(f(t), t),$$

where $A(\cdot,\cdot)$ represents a nonlinear function determined by the system. The goal of INR is to use the neural network g_θ to fit the f with the training tuples $\{(t, f(t))\}_{t=0}^{N}$. However, in general, the training tuple $(t, f(t))$ is not enough for networks to learn a dynamical system. For example, we consider a simple time-invariant system:

$$\dot{f} = A \cdot f(t), \quad f(0) = \mathbf{y}_0,$$

where A is a constant. The solution of the problem can be written as $f(t) = \exp(At)\mathbf{y}_0$. If g_θ is a general feed-forward ReLU network, then we can achieve $\|g_\theta(t) - f(t)\| \le \epsilon$ for all t under some conditions. In particular, we require $O(\log(\epsilon)^d)$ non-zero hidden units to train $g_\theta(t)$ using training tuples $\{(t, f(t))\}$, where d is the dimension of $f(t)$ if f is invertible [34, 49].

Interestingly, if we use $\{(t, \dot{f}(t)), f(t)\}$ as training tuples, then one-layer linear network is sufficient to learn $f(t)$ and

$$\|g_\theta(t, \dot{f}(t)) - f(t)\| = 0, \quad \forall t \in [1, N].$$

This is because the learning problem simplifies to learning the constant A instead of learning a continuous function $f(\cdot)$. Hence, considering the high order differentials $(\dot{f}, \ddot{f}, \cdots)$ as the network input can significantly benefit in ...ng a dynamical system.

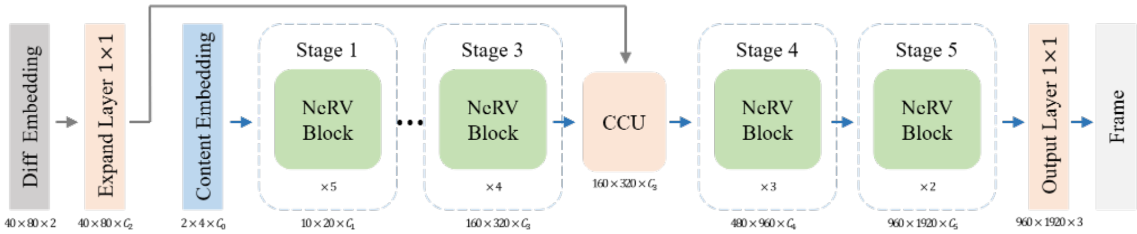

Figure 4. Architecture of decoder with CCU as fusion module for 960 × 1920.

For hybrid-based INR methods, we introduce the *diff stream* $\mathbf{y}_t^D = \mathbf{y}_t - \mathbf{y}_{t-1}$ as an auxiliary input with the main content stream $\mathbf{y}_t$ (for more details, see the ablation studies). The difference between two adjacent frames could be treated as discrete differential ∇f,

$$\begin{aligned} \nabla f\Big|_{\tau=t} &= \frac{f(\tau) - f(\tau - \Delta\tau)}{\Delta\tau}\Big|_{\tau=t} \\ &\simeq \frac{f(t) - f(t-1)}{t-(t-1)} = \mathbf{y}_t - \mathbf{y}_{t-1}. \end{aligned} \tag{3}$$

In addition to more efficiently fitting the implicit function f, we believe that diff stream could also help g_θ capture the adjacent dynamic and a more robust video representation.

4. Method

Overview. Our proposed DNeRV method is a hybrid-based approach that uses two input streams. The encoders in D-NeRV process two streams separately; the intermediate output embeddings pass through a fusion step before entering the decoders. We follow the *hybrid* philosophy in [3] that treats the *video-specific embeddings* and *video-specific decoder* as the INRs for videos. The model architecture is shown in Fig. 3.

Encoder. The encoder consists of Diff Encoder and Content Encoder. For the content encoder, we use the settings in [3] for a fair comparison, and use input as the current frame $\mathbf{y}_t$. Meanwhile, for the given frame $\mathbf{y}_t$, DNeRV calculates the frame differences, $\mathbf{y}_t^D = \mathbf{y}_t - \mathbf{y}_{t-1}$ and $\mathbf{y}_{t+1}^D = \mathbf{y}_{t+1} - \mathbf{y}_t$, and concatenates them before feeding into diff encoder. Both content encoder and diff encoder adopt one ConvNeXT block at one stage, with a scale transformation layer and layer norm adopted at the beginning of each stage. For 1920 × 960 video, the strides are [5, 4, 4, 3, 2] in content encoder and [4, 3, 2] in diff encoder, providing content embedding $\mathbf{b}_t^C$ in $16 \times 2 \times 4$ and diff embedding $\mathbf{b}_t^D$ in $2 \times 40 \times 80$. It is worth mentioning that the shape of diff embedding is flexible, smaller diff embedding (e.g., $2 \times 10 \times 20$) only brings slight performance drops and remain comparable against other NeRV methods. See more details in ablation studies and Tab. 3. The encoder can be described as

$$\begin{aligned} \mathbf{b}_t^C &= \mathsf{CONT\_ENC}(\mathbf{y}_t), \\ \mathbf{b}_t^D &= \mathsf{DIFF\_ENC}(\mathsf{concat}[\mathbf{y}_t^D, \mathbf{y}_{t+1}^D]). \end{aligned} \tag{4}$$

Decoder. We adopt NeRV block as the basic block in each decoding stage. For the embeddings from encoder, they would be fused in same shape. We explored different connections for fusion, and the final architecture is shown in Fig. 4. For the fusion module, it could be sum fusion, conv fusion, or other gated mechanisms. Once the features are merged, they pass through other stages and map into pixel domain via channel reduction.

Fusion. To fuse features from diff stream and content stream, conv fusion $\mathbf{s}_t = \mathsf{Conv}(\mathbf{b}_t^C) + \mathbf{b}_t^D$ or concat fusion $\mathbf{s}_t = \mathsf{concat}[\mathbf{b}_t^C, \mathbf{b}_t^D]$ may not be suitable. This is because the features come from different domains as $\mathbf{b}^D$ is the discretization of differential of f, while $\mathbf{b}^C$ represents the value of f. To merge the two streams, inspired by gated mechanism in temporal sequence learning [7,45,54], we introduce a collaborative content unit (CCU). Our motivation is that diff stream needs to be fused with *content* stream *collaboratively* to obtain the refined content features by adding high-order information. Specifically, the content stream could be treated as hidden state in an RNN-fashion, which contains time-varying features helping reconstruction. CCU can be represented as

$$\begin{aligned} \mathbf{z}_t^D &= GELU(\mathsf{PS}(\mathbf{W}_z^{1\times1} * (\mathbf{b}_t^D))), \\ \tilde{\mathbf{b}}_t^C &= \mathsf{BLOCK}^{(2)}(\mathbf{b}_t^C), \\ \mathbf{u}_t &= tanh(\mathbf{W}_{ub} * \tilde{\mathbf{b}}_t^C + \mathbf{W}_{uz} * \mathbf{z}_t^D), \\ \mathbf{v}_t &= Sigmoid(\mathbf{W}_{vb} * \tilde{\mathbf{b}}_t^C + \mathbf{W}_{vz} * \mathbf{z}_t^D), \\ \mathbf{s}_t &= \mathbf{u}_t \odot \mathbf{v}_t + (1 - \mathbf{v}_t) \odot \tilde{\mathbf{b}}_t^C, \end{aligned} \tag{5}$$

where PS is PixelShuffle, $*$ is convolution operator and $\odot$ is Hadamard product. $\mathbf{v}_t$ could be treated as the update gate in GRU [7], to decide how much information in content feature could be remained. Finally, two streams are merged and the adjacent dynamics collaboratively captured by CCU can help network g_θ learn the implicit mapping. To balance the parameter quantity, we reduce the channels in the last two stages of the decoder. Final output is given as follows

$$\begin{aligned} \mathbf{s}_t &= \mathsf{FUSION}(\mathbf{b}_t^C, \mathbf{b}_t^D), \\ \tilde{\mathbf{y}}_t &= Sigmoid(\mathbf{W}_y^{1\times1} * (\mathsf{BLOCK}^{(3)}(\mathbf{s}_t))), \end{aligned} \tag{6}$$

FUSION represents CCU in our implementation.

size	0.35M	0.75M	1.5M	3M
NeRV [4]	26.99	28.46	30.87	33.21
E-NeRV [21]	27.84	30.95	32.09	36.72
H-NeRV [3]	30.15	32.81	35.19	37.43
D-NeRV	**30.80**	**33.30**	**35.22**	**38.09**

(a) PSNR on Bunny with varying **model size**.

epochs	300	600	1200	1800	2400	3600
NeRV [4]	28.46	29.15	29.57	29.73	29.77	29.86
E-NeRV [21]	30.95	32.07	32.79	33.10	33.36	33.67
H-NeRV [3]	32.81	33.89	34.51	34.73	34.88	35.03
D-NeRV	**33.30**	**34.28**	**34.83**	**35.16**	**35.25**	**35.34**

(b) PSNR on Bunny with varying **epochs**.

960×1920	Beaut	Bosph	Honey	Jocke	Ready	Shake	Yacht	avg.
NeRV [4]	33.25	33.22	37.26	31.74	24.84	33.08	28.03	31.63
E-NeRV [21]	33.17	33.69	37.63	31.63	25.24	34.39	28.42	32.02
HNeRV [3]	33.58	34.73	38.96	32.04	25.74	34.57	29.26	32.69
DNeRV (L1+SSIM)	**40.19**	36.59	**43.23**	35.75	28.17	**38.25**	30.73	**36.13**
DNeRV (L2)	40.00	**36.67**	41.92	**35.75**	**28.67**	36.53	**31.10**	35.80

(c) PSNR on UVG in 960 × 1920.

480×960	Beaut	Bosph	Honey	Jocke	Ready	Shake	Yacht	avg.
NeRV [4]	36.27	35.07	40.76	32.58	25.81	35.33	30.11	33.70
E-NeRV [21]	36.26	36.06	43.26	32.70	26.19	35.64	30.38	34.35
HNeRV [3]	36.91	36.95	42.05	33.33	27.07	36.97	30.96	34.89
DNeRV (L1+SSIM)	**40.24**	37.35	**43.98**	**35.85**	28.70	**38.84**	31.03	**36.58**
DNeRV (L2)	39.64	**37.49**	42.45	35.44	**29.21**	36.83	**31.30**	36.05

(d) PSNR on UVG in 480 × 960.

Table 1. Video regression results on Bunny and UVG, where DNeRV uses different loss functions for ablation.

Discussion of optical flow. Although optical flow captures adjacent temporal relationship as well as the difference stream, we could not achieve comparable performance when using optical flow. The main reason is that INR-based video representation task is different from semantic video tasks. In the case of NeRV, pixel-level features that directly help decoder reconstruction are more vital. More details can be found in the supplementary materials.

Comparison with NeRV. Now, we look back to the philosophy of NeRV and compare it with DNeRV. For NeRV, we search for an operator g_θ by solving the following optimization problem:

$$\underset{\theta}{\operatorname{argmin}} \|g_\theta\left(h(t)\right) - f(t)\|, \tag{7}$$

where h represents the embedding of time index t and $f(t) = \mathbf{y}$ represents the frames in pixel-domain. In the case of hybrid methods [3], we solve the following optimization problem:

$$\underset{\theta}{\operatorname{argmin}} \|g_\theta\left(f(t)\right) - f(t)\|, \tag{8}$$

where the embedding is the frame itself. The hybrid method attempts to fit a series of invariant point transformations in function space for every training tuple $(t, \mathbf{y})$. This explains why existing methods only work well on fixed background scene with few dynamics, such as "HoneyBee" or "ShakeNDry" in UVG. In other words, they only take effect when $\mathbf{y}_i$ is within a small neighborhood of training samples. In other words, g_θ only learns the behavior of f near the mean of whole training samples, where adjacent dynamics would not be apparent. In the case of DNeRV, we solve the following optimization problem:

$$\underset{\theta}{\operatorname{argmin}} \|g_\theta\left(f, \nabla f, ..., \nabla^{(i)} f\right) - f\|, \tag{9}$$

where $i = 1$ in our realization. DNeRV attempts to learn a *dynamical system* that represents f in implicit way.

5. Experiments

Settings. We verify DNeRV on Bunny [18], UVG [26] and DAVIS Dynamic. Bunny owns 132 frames for 720 × 1280. UVG has 7 videos at 1080×1920 with length of 600 or 300. DAVIS Dynamic is a subset of DAVIS16 validation [30] which containing 22 videos[1] in 1080 × 1920. Most of the selected videos contain dynamic scenes or moving targets, which are quite difficult for existing methods. Following the settings in [3] for fair comparison, we center-crop the videos into 640 × 1280 or 960 × 1920 and reshape UVG into 480 × 960 for additional comparison.

During training, we adopt Adam [17] as the optimizer with learning rate of 5×10^{-4} and cosine annealing learning rate schedule [47] and the batch size is set to 1. We use PSNR and SSIM to evaluate the video quality. The stride list, kernel size and reduction rate remain to be same as [3], except for the channels in the last two stages of decoder.

We compare DNeRV with others in video regression and three downstream visual tasks consist of video compression, interpolation and inpainting. In video interpolation we train the model on the sequence of even frames and test it on the odd frames sequence from UVG and DAVIS Dynamics. In video inpainting, we directly use the models trained in regression without any fine-tuning and test them using masked videos in disperse mask or central mask from DAVIS Dynamics. All experiments are conducted in Pytorch with GPU RTX2080ti, with 3M size and 300 epochs unless otherwise clarified.

Discussion of Loss functions. We conduct loss objective ablation between L2 and L1+SSIM, shown in Tab. 1c and Tab. 1d. L1+SSIM is the loss objective in NeRV [4], $L(\hat{\mathbf{y}}, \mathbf{y}) = \alpha\|\hat{\mathbf{y}}, \mathbf{y}\|_1 + (1-\alpha)(1 - SSIM(\hat{\mathbf{y}}, \mathbf{y})), \alpha = 0.7$. Owing to L1 norm is the convex approximation of L0 norm [1], it is better for scenes with complex textures and high-frequency subtle spatial structure but few motion between

[1] blackswan, bmx-bumps, camel, breakdance, car-roundabout, bmx-trees, car-shadow, cows, dance-twirl, dog, car-turn, dog-agility, drift-[illegible] drift-turn, goat, libby, mallard-fly, mallard-water, parkour, [illegible]ade, scooter-black, strolle.

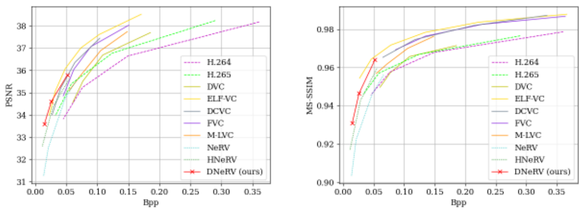

Figure 5. Compression results on 960 × 1920 UVG. DNeRV outperforms other INR-based methods.

frames. While during experiments, L2 is set as default loss function as it is better for low-frequency scenes with large motion.

5.1. Video Regression

Bunny. The comparison between different implicit methods trained in 300 epochs on Bunny is shown in Tab. 1a. DNeRV outperforms others. Also, we compare various implicit methods in same 0.75M but different training epochs reported in Tab. 1b. DNeRV surpasses other methods relying on the reasonable structure with no additional gradient propagation difficulties.
UVG. The PSNR results on UVG are given in Tab. 1c and Tab. 1d. DNeRV shows large improvements at resolution 960 × 1920. The excellent results are attributed to the high-resolution diff stream, containing a great deal of content-specific spatial information. The adjacent dynamic hidden among frames could be captured in a more reasonable way, helping the network converge faster.
DAVIS Dynamic. The results of regression on DAVIS Dynamic are shown together with inpainting results in Tab. 2. DNeRV achieves an impressive performance when processing those videos with complicated scene transformation or object movements. Another difficulty of DAVIS Dynamic videos is that the number of frames is quite smaller, e.g., 25 for "dog-agility" and 43 for "Scooter-black". Fewer frames and ubiquitous adjacent dynamics present extreme difficulty for implicit methods to fit the latent mapping, indicating the effectiveness of DNeRV.

5.2. Video Compression

We show the compression results in PSNR and SSIM on UVG dataset in Fig. 5. Without pruning and only 8 bits quantization with entropy encoding adopted, DNeRV outperforms other NeRV methods especially on PSNR. D-NeRV optimizes the network structure under the condition of parameter amount, thus reducing the redundancy in weights. Although it couldn't maintain performance in 40% pruning like [4,21], the 10% pruned DNeRV is still competitive compared with other implicit methods. We also report VMAF [32] results on UVG in the appendix.
Compare with state-of-the-arts methods. We compare D-NeRV with H.264 [48], H.265 [41], DVC [23], FVC [illegible], M-LVC [22], DCVC [19] and ELF-VC [33]. Without any specific modification, DNeRV is better than traditional video codecs H.264 or H.265 in both PSNR and SSIM, and it is also competitive with the state-of-the-art deep compression methods. We will explore the potential of DNeRV in video compression in the follow-up studies. The complexity comparison of video decoding with two rapid neural compression methods [20, 37] in model size, decoding time and FPS is shown in Tab 5.

size of diff embedding	stage 2	stage 3	stage 4	params	PSNR/SSIM
N/A				0.311M	29.64/0.908
				0.347M	29.95/0.914
2 × 40 × 80	A			0.288M	29.31/0.907
	A	C		0.339M	29.93/0.914
		<u>C</u>		0.339M	<u>30.60/0.924</u>
	A		C	0.348M	29.73/0.909
			C	0.348M	30.38/0.922
		C	C	0.348M	30.33/0.921
	A	C	C	0.348M	26.96/0.835
2 × 10 × 20		U		0.343M	29.66/0.907
Final		U		0.349M	**30.80/0.930**

Table 3. Ablation study for fusion module and diff embedding size, training on Bunny in 300 epochs. **A** indicates sum fusion, **C** is conv fusion and **U** is the CCU. The final version of DNeRV consists of diff embedding in shape of 2×40×80, 3rd stage where merging and CCU as the fusion module. The first two rows which are marked as N/A represent HNeRV baseline, where the size of diff embedding is not available.

input of diff stream	conv fusion	CCU
$\Delta f(t)$	30.359/0.920	30.425/0.922
$\Delta f(t+1)$	30.354/0.919	-/-
$(\Delta f(t-1) + \Delta f(t+1))/2$	30.213/0.918	30.278/0.920
concat$[\Delta f(t), \Delta f(t+1)]$	<u>30.598/0.924</u>	**30.804/0.930**
concat$[\Delta f(t), \Delta f(t+1), \nabla f(t)]$	30.310/0.919	30.392/0.921

Table 4. Ablation study of various difference on Bunny in 0.35M and 300 epochs, where Δ is first order and ∇ is second order difference.

method	1920	params↓	dec time↓	FPS↑
DCVC [19]	× 1080	35M	35590ms	0.028
Li 2022 [20]	× 1080	67M	525ms	1.9
Sheng 2021 [37]	× 1080	41M	470ms	2.12
HNeRV [3]	× 960	3.2M	30ms	33.3
DNeRV	× 960	3.5M	39ms	25.6

Table 5. Complexity comparison.

5.3. Video Interpolation

Since the INR-based video representation is over-fitting to the given video, we evaluate the generalization of different methods by video interpolation task. We follow the setting in [3] and demonstrate the quantitative results on UVG in Tab. 6 and qualitative results in Fig. 6. Thanks to the consideration of adjacent dynamics, DNeRV outperforms [illegible] implicit methods especially on the videos owning dy-

video	regression				Inp-Mask-S		Inp-Mask-C	
	NeRV	E-NeRV	HNeRV	DNeRV	HNeRV	DNeRV	HNeRV	DNeRV
Blackswan	28.48/0.812	29.38/0.867	30.35/0.891	**30.92/0.913**	26.51/0.825	**29.01/0.886**	24.64/0.783	**27.45/0.858**
Bmx-bumps	29.42/0.864	28.90/0.851	29.98/0.872	**30.59/0.890**	23.16/0.728	**25.70/0.819**	20.39/0.665	**22.95/0.767**
Bmx-trees	26.24/0.789	27.26/0.876	28.76/0.861	**29.63/0.882**	22.93/0.720	**26.57/0.841**	20.26/0.653	**21.62/0.752**
Breakdance	26.45/0.915	28.33/0.941	30.45/0.961	**30.88/0.968**	27.63/0.945	**29.16/0.961**	23.84/0.907	**25.18/0.938**
Camel	24.81/0.781	25.85/0.844	26.71/0.844	**27.38/0.887**	20.94/0.661	**24.71/0.832**	21.85/0.733	**23.72/0.815**
Car-round	24.68/0.857	26.01/0.912	27.75/0.912	**29.35/0.937**	23.96/0.752	**27.63/0.854**	22.06/0.810	**24.82/0.886**
Car-shadow	26.41/0.871	30.41/0.922	31.32/0.936	**31.95/0.944**	25.87/0.875	**27.90/0.914**	**28.67**/0.908	28.18/**0.923**
Car-turn	27.45/0.813	29.02/0.888	29.65/0.879	**30.25/0.892**	23.96/0.752	**27.63/0.854**	24.43/0.773	**25.67/0.821**
Cows	22.55/0.702	23.74/0.819	24.11/0.792	**24.88/0.827**	21.37/0.682	**22.91/0.770**	20.81/0.668	**21.87/0.733**
Dance-twril	25.79/0.797	27.07/0.864	28.19/0.845	**29.13/0.870**	23.05/0.743	**26.13/0.830**	21.10/0.704	**23.00/0.783**
Dog	28.17/0.795	30.40/0.882	30.96/0.898	**31.32/0.905**	25.34/0.739	**27.43/0.824**	23.09/0.677	**24.96/0.763**
Dog-ag	29.08/0.821	29.30/0.905	28.75/0.893	**29.94/0.923**	27.70/0.884	**28.28/0.913**	**25.12**/0.856	24.85/**0.886**
Drift-straight	26.65/0.860	29.10/0.941	30.80/0.932	**31.50/0.940**	24.79/0.833	**27.00/0.892**	20.15/0.725	**22.61/0.823**
Drift-turn	26.70/0.812	27.94/0.875	29.72/0.834	**30.37/0.862**	22.27/0.677	**26.20/0.816**	19.95/0.636	**22.65/0.749**
Goat	23.90/0.746	25.25/0.855	26.62/0.858	**27.79/0.887**	21.11/0.675	**22.65/0.755**	20.21/0.639	**21.76/0.727**
Libby	29.08/0.821	31.43/0.890	32.69/0.917	**33.43/0.927**	27.59/0.825	**29.51/0.875**	24.33/0.752	**26.07/0.826**
Mallard-fly	26.83/0.757	28.84/0.847	**29.22/0.848**	28.77/0.833	23.81/0.709	**25.83/0.774**	20.83/0.618	**22.77/0.689**
Mallard-water	25.20/0.824	27.28/0.896	29.08/0.908	**29.69/0.922**	23.55/0.803	**24.11/0.845**	**21.18**/0.743	21.15/**0.796**
Parkour	25.14/0.794	25.31/0.845	**26.56/0.851**	25.75/0.827	21.32/0.685	**24.51/0.799**	19.97/0.650	**21.55/0.754**
Rollerblade	29.28/0.898	**33.32/0.964**	32.19/0.935	32.49/0.940	29.32/0.911	**30.41/0.931**	27.31/0.901	**27.63/0.917**
Scooter-black	22.73/0.835	25.79/0.927	27.38/0.923	**28.53/0.940**	21.05/0.794	**24.27/0.897**	19.76/0.789	**21.00/0.844**
Stroller	29.28/0.859	30.37/0.914	31.31 0.894	**32.73/0.928**	25.90/0.796	**28.46/0.876**	22.86/0.734	**24.52/0.822**
Average	26.56/0.819	28.19/0.887	29.21/0.886	**29.86/0.902**	24.09/0.774	**26.53/0.854**	22.40/0.742	**23.91/0.812**

Table 2. Video regression and inpainting results on 960×1920 Davis Dynamic in PSNR/SSIM, larger is better.

UVG	Beauty	Bospho	Honey	Jockey	Ready	Shake	Yacht	avg.
NeRV [4]	28.05	30.04	36.99	20.00	17.02	29.15	24.50	26.54
E-NeRV [21]	27.35	28.95	38.24	19.39	16.74	30.23	22.45	26.19
H-NeRV [3]	31.10	34.38	**38.83**	23.82	20.99	32.61	27.24	29.85
D-NeRV	**35.99**	**35.19**	37.43	**30.61**	**24.05**	**35.34**	**28.70**	**32.47**

Table 6. Video interpolation results on 960 × 1920 UVG in PSNR.

namic scene and large motion. More results on DAVIS are reported in appendix.

Case study. Interpolation is quite challenging for the generalization ability of INRs. Shown in Tab.6, DNeRV earns 28.5% and 14.6% improvement against the best results on "Jockey" and "ReadySetGo", where exist strong dynamics and large motion. Especially for "Jockey" in Fig. 6, we could recognize some numbers or letters from DNeRV's prediction, but it's impossible for HNeRV's.

5.4. Video Inpainting

We conduct video inpainting experiments on DAVIS Dynamic with central mask and disperse mask. The central mask follows the setting in [29, 52] as the rectangular mask with width and height both 1/4 of the original frame. The disperse mask is five square masks in 100×100 average distributed in fixed positions. The quantitative results are listed in Tab. 2 and qualitative performance is shown in Fig. 6. We train the implicit methods on raw videos but test them with masked videos. We only conduct the comparison with HNeRV because it beats other implicit methods on robust inpainting, reported in [3]. Although the diff is also masked before input, DNeRV still obtains impressive and rob

2037

sults.

Case study. Detailed texture is a major difficulty for D-NeRV to encode videos, because difference of neighboring frames is more like than high frequency details close to noise, such as "Mallard-fly" and "Cows". However, D-NeRV outperform other implicit methods in robustness and generalization. Although in "Mallard-fly", DNeRV's training PSNR is less than HNeRV, but severe trivial-fitting phenomenon happens. Those similar textures which in the same position among frames, seem as copies in HNeRV's results, shown as (a, d) in Fig 6.

5.5. Ablation Study

We provide various ablation studies on Bunny with PSNR and MS-SSIM as the metrics, reported in Tab 3 and Tab. 4.

Diff stream. We compare the backward difference, forward difference, central difference and second order difference as input and merge one of them with content stream in 3rd stage. The results are shown in Tab. 4, indicating that higher-order difference may not be helpful for fitting. We concatenate both backward and forward difference as our default diff stream.

Param quantity. To verify the effectiveness of diff stream, we compare the hybrid ones without diff stream in varying model size (by changing the channels in decoder) with DNeRV. More parameters may not attributed to better performance. The params, converge speed and generalization ability should be balanced.

mbedding. The smaller shape of diff embedding

Figure 6. Visualization comparison for inpainting (a, b, c, d) and interpolation (e, f) results on Davis Dynamic. The left is ground truth, DNeRV's results in mid and HNeRV's on the right. White number is the best PSNR for each method training on original video, while yellow ones are testing PSNR on masked videos. Zoom in for details.

would reduce both the model size and reconstruction quality. The shape in $2 \times 40 \times 80$ is adopted as default.

Fusion module. The ablation results reported in Tab 3 and Tab. 4 verify the effectiveness of proposed CCU. Compared with conv fusion or sum fusion, CCU improves PSNR by adaptively merging two-stream features.

Fusion stages. It is vital for DNeRV to select the stage where two streams are merged, we do ablations on every possible connection. Decoder should balance the params and the difficulty of gradient descent. DNeRV adopts merging in 3rd stage without sum fusion. Further we upgrade conv fusion to CCU.

6. Conclusion

In this paper, we propose the difference neural representation for videos (DNeRV) for modeling the inherent dynamics of contextual frames. Relying on the diff strea collaborative content unit, DNeRV retains its advantages in reconstruction quality for video regression, compression, inpainting and interpolation. Consequently, the experimental results show that the proposed DNeRV could achieve effective and robust representation for videos by achieving a better approximation to the implicit mapping than existing NeRV methods.

Future directions. DNeRV shows its potential on various visual tasks. The improved task-specific approach based on DNeRV is promising to challenge the state-of-the-art methods. Also, rigorous theoretical analysis needs to be improved for INR-based networks g_θ fitting the continuous f on finite training tuple via gradient descent.

Acknowledgements. We thank Xingyu Xie for comments that greatly improved the manuscript. The work was supported in part by National Key Research and Development t of China (2022YFF0902402).

References

[1] Emmanuel J. Candès and Terence Tao. Decoding by linear programming. *IEEE Transactions on Information Theory*, 2005. 5

[2] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017. 3

[3] Hao Chen, Matthew Gwilliam, Ser-Nam Lim, and Abhinav Shrivastava. Hnerv: A hybrid neural representation for videos. 2022. 2, 4, 5, 6, 7, 11

[4] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser-Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. In *NeurIPS*, 2021. 1, 2, 5, 6, 7

[5] 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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, 2022. 2

[6] Zhiqin Chen and Hao Zhang. Learning implicit fields for generative shape modeling. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*, 2019. 2

[7] Junyoung Chung, Caglar Gülcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv:1412.3555*, 2014. 3, 4

[8] Zhiwen Fan, Yifan Jiang, Peihao Wang, Xinyu Gong, Dejia Xu, and Zhangyang Wang. Unified implicit neural stylization. In *ECCV 2022*, 2022. 1

[9] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J. Zico Kolter. Multiplicative filter networks. In *9th International Conference on Learning Representations, ICLR 2021*, 2021. 2

[10] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, 2016. 3

[11] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv: Learning*, 2019. 2

[12] Amir Gholami, K. Kwon, Bichen Wu, Zizheng Tai, Xiangyu Yue, Peter H. Jin, Sicheng Zhao, and Kurt Keutzer. Squeezenext: Hardware-aware neural network design. *CVPRW*, 2018. 2

[13] Albert Gu, Çaglar Gülçehre, Thomas Paine, Matt Hoffman, and Razvan Pascanu. Improving the gating mechanism of recurrent neural networks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020*, pages 3800–3809. 3

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 1997. 3

[15] Zhihao Hu, Guo Lu, and Dong Xu. Fvc: A new framework towards deep video compression in feature space. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2, 6

[16] Zhewei Huang, Tianyuan Zhang, Wen Heng, Boxin Shi, and Shuchang Zhou. Real-time intermediate flow estimation for video frame interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 3

[17] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015*, 2015. 5

[18] Janus B. Kristensen. Big buck bunny. 2010. 5

[19] Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. 2021. 2, 6

[20] Jiahao Li, Bin Li, and Yan Lu. Hybrid spatial-temporal entropy modelling for neural video compression. In *MM '22: The 30th ACM International Conference on Multimedia*, 2022. 6

[21] Zizhang Li, Mengmeng Wang, Huaijin Pi, Kechun Xu, Jianbiao Mei, and Yong Liu. E-nerv: Expedite neural video representation with disentangled spatial-temporal context. *arXiv:2207.08132*, 2022. 2, 5, 6, 7

[22] Jianping Lin, Dong Liu, Houqiang Li, and Feng Wu. M-lvc: Multiple frames prediction for learned video compression. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 6

[23] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. Dvc: An end-to-end deep video compression framework. 2019. 2, 6

[24] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue Wang, and Pedro V. Sander. Deblur-nerf: Neural radiance fields from blurry images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, 2022. 2

[25] Long Mai and Feng Liu. Motion-adjustable neural implicit video representation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, 2022. 2

[26] Alexandre Mercat, Marko Viitanen, and Jarno Vanne. UVG dataset: 50/120fps 4k sequences for video codec analysis and development. In *Proceedings of the 11th ACM Multimedia Systems Conference, MMSys 2020*, 2020. 1, 5

[27] Ben Mildenhall, Peter Hedman, Ricardo Martin-Brualla, Pratul P. Srinivasan, and Jonathan T. Barron. Nerf in the dark: High dynamic range view synthesis from noisy raw images. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, 2022. 2

[28] 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. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 1, 2

[29] Hao Ouyang, Tengfei Wang, and Qifeng Chen. Internal video inpainting by implicit long-range propagation. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 7

[30] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus H. Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, 2016. 1, 5

[31] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021*, 2021. 2

[32] Reza Rassool. Vmaf reproducibility: Validating a perceptual practical video quality metric. *2017 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB)*, 2017. 6, 11

[33] Oren Rippel, Alexander G. Anderson, Kedar Tatwawadi, Sanjay Nair, Craig Lytle, and Lubomir D. Bourdev. Elf-vc: Efficient learned flexible-rate video coding. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021. 2, 6

[34] Johannes Schmidt-Hieber. Nonparametric regression using deep neural networks with relu activation function. *ArXiv*, abs/1708.06633, 2020. 3

[35] Laura Sevilla-Lara, Yiyi Liao, Fatma Güney, Varun Jampani, Andreas Geiger, and Michael J. Black. On the integration of optical flow and action recognition. In *Pattern Recognition - 40th German Conference, GCPR*, 2018. 2

[36] Wentao Shangguan, Yu Sun, Weijie Gan, and Ulugbek S. Kamilov. Learning cross-video neural representations for high-quality frame interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 2

[37] Xihua Sheng, Jiahao Liu Li, Bin Li, Li Li, Dong Liu, and Yan Lu. Temporal context mining for learned video compression. *arXiv:2111.13850*, 2022. 6

[38] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *NeurIPS*, 2014. 3

[39] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NeurIPS*, 2020. 2, 3

[40] Yannick Strümpler, Janis Postels, Ren Yang, Luc Van Gool, and Federico Tombari. Implicit neural representations for image compression. In *ECCV*, 2022. 2

[41] Gary J. Sullivan, Jens-Rainer Ohm, Woojin Han, and Thomas Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Trans. Circuits Syst. Video Technol.*, 2012. 6

[42] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2018. 3

[43] Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. In *NeurIPS*, 2020. 2, 3

[44] Ruben Villegas, Jimei Yang, Seunghoon Hong, Xunyu Lin, and Honglak Lee. Decomposing motion and content for natural video sequence prediction. In *5th International Conference on Learning Representations, ICLR 2017*, 2017. 3

[45] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S. Yu. Predrnn++: Towards A resolu[illegible] the deep-in-time dilemma in spatiotemporal predictive learning. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, 2018. 3, 4

[46] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S. Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. In *NeurIPS*, 2017. 3

[47] Zhou Wang, Eero P. Simoncelli, and Alan Conrad Bovik. Multiscale structural similarity for image quality assessment. *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, 2003. 5

[48] Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Trans. Circuits Syst. Video Technol.*, 2003. 6

[49] Xingyu Xie, Hao Kong, Jianlong Wu, Wayne Zhang, Guangcan Liu, and Zhouchen Lin. Maximum-and-concatenation networks. 2020. 3

[50] Xingyu Xie, Pan Zhou, Huan Li, Zhouchen Lin, and Shuicheng Yan. Adan: Adaptive nesterov momentum algorithm for faster optimizing deep models. *ArXiv*, abs/2208.06677, 2022. 11

[51] Qiangeng Xu, Zexiang Xu, Julien Philip, Sai Bi, Zhixin Shu, Kalyan Sunkavalli, and Ulrich Neumann. Point-nerf: Point-based neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, 2022. 2

[52] Rui Xu, Xiaoxiao Li, Bolei Zhou, and Chen Change Loy. Deep flow-guided video inpainting. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 7

[53] 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. In *The Tenth International Conference on Learning Representations, ICLR 2022*, 2022. 2

[54] Wei Yu, Yichao Lu, Steve Easterbrook, and Sanja Fidler. Efficient and information-preserving future frame prediction and beyond. In *8th International Conference on Learning Representations, ICLR 2020*, 2020. 4

[55] Gizem Yüce, Guillermo Ortiz-Jiménez, Beril Besbinar, and Pascal Frossard. A structured dictionary perspective on implicit neural representations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, 2022. 2

[56] Lu Zhang, Ju Dai, Huchuan Lu, You He, and Gang Wang. A bi-directional message passing model for salient object detection. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018*, 2018. 3

[57] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N. Metaxas. Learning to forecast and refine residual motion for image-to-video generation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 3

[58] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation. In *ICCV*, 2021. 1