DS-NeRV: Implicit Neural Video Representation with Decomposed Static and Dynamic Codes

Hao Yan Zhihui Ke Xiaobo Zhou† Tie Qiu Xidong Shi Dadong Jiang
College of Intelligence and Computing, Tianjin University
{yan_hao, kezhihui, xiaobo.zhou, qiutie, suif, patrickdd}@tju.edu.cn

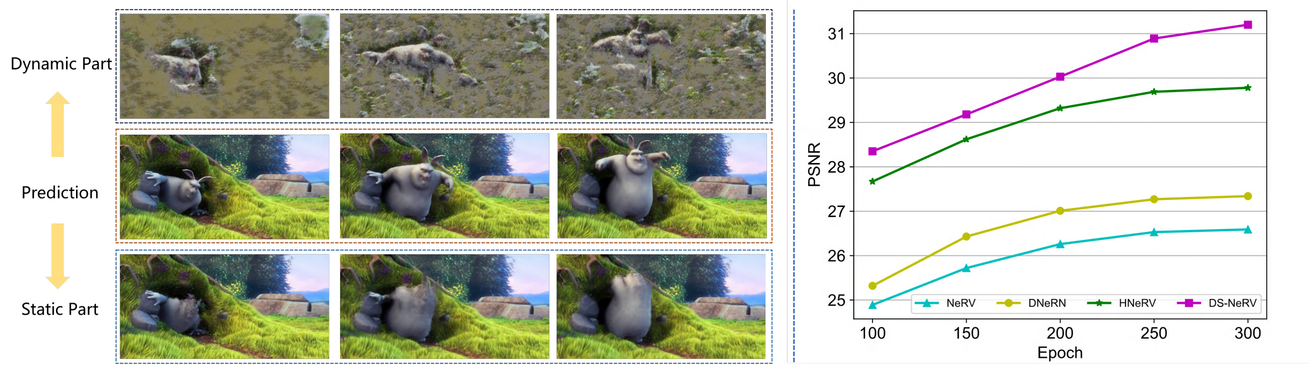

Figure 1. (Left) Our proposed DS-NeRV decomposes the video into learnable static and dynamic codes, which represent static elements and dynamic elements in the video. (Right) Video reconstruction results for various implicit neural representations with 0.35M.

Abstract

Implicit neural representations for video (NeRV) have recently become a novel way for high-quality video representation. However, existing works employ a single network to represent the entire video, which implicitly confuse static and dynamic information. This leads to an inability to effectively compress the redundant static information and lack the explicitly modeling of global temporal-coherent dynamic details. To solve above problems, we propose DS-NeRV, which decomposes videos into sparse learnable static codes and dynamic codes without the need for explicit optical flow or residual supervision. By setting different sampling rates for two codes and applying weighted sum and interpolation sampling methods, DS-NeRV efficiently utilizes redundant static information while maintaining high-frequency details. Additionally, we design a cross-channel attention-based (CCA) fusion module to efficiently fuse these two codes for frame decoding. Our approach achieves a high quality reconstruction of 31.2 PSNR with only 0.35M parameters thanks to separate static and dynamic codes representation and outperforms existing NeRV methods in many downstream tasks. Our project website is at https://haoyan14.github.io/DS-NeRV/.

† Corresponding Author

1. Introduction

In the first half of 2022, video traffic accounted for a substantial 65.93% share of the overall network traffic and constituted as much as 80% of the total downstream traffic during the evening peak hours [1, 2]. This causes tremendous pressure on network communication and storage. Thus, it is crucial to explore more efficient video representations for compression.

In recent years, implicit neural representations (INR) have emerged as a promising solution due to their remarkable capacity to represent diverse forms of signals [9, 34, 38, 41]. With the development of INR, it has been applied to video representation tasks, such as NeRV [9], which has transformed the challenge of video compression into a problem of model compression. Additionally, INR-based video representations often exhibit a simpler training process and a higher decoding speed [11] compared to traditional video compression methods [13, 22, 43, 47] and learning-based video compression methods [3, 14, 26, 28, 52].

Typically, INR-based video representations can be categorized into two types: (1) *index-based* [4, 9, 25] methods that model the video as a neural network, where the positional encoding of the frame index is taken as input to reconstruct the corresponding frame. (2) *hybrid-based* methods [11, 54] that employ an encoder-decoder architecture where they input each frame into the encoder to obtain the corresponding embedding, which is then forwarded to the decoder for reconstruction. Compared with index-based methods which is content-agnostic, hybrid-based methods leverage frame embedding to encapsulate frame information, thereby enhancing reconstruction quality. However, the aforementioned two methods model the video as a whole, confusing both static and dynamic information within the video implicitly in the model parameters. Therefore, they cannot effectively compress static redundant information and model globally coherent dynamic elements in the video.

Typically, a video consists of time-invariant static elements and time-varying dynamic elements. As shown in Fig. 1 (Left), the grassy and rocks in the background either remain static or change minimally, while the bunny's posture exhibits noticeable changes over time. Thus, to reduce the size of the video INR, it is beneficial to compress these redundant static information. On the other hand, the dynamic elements require smooth modeling across the entire video to preserve high-frequency details.

In this paper, we draw inspiration from the above insight and propose DS-NeRV, a method that decomposes video into sparse learnable static codes $\mathcal{C}^s$ and dynamic codes $\mathcal{C}^d$, which respectively represent the static and dynamic elements in the video. The design of the learnable codes bears resemblance to the learnable noise vector used in *Generative Latent Optimization* GLO) [7]. By assigning different sampling rates and sampling methods for two codes, DS-NeRV effectively decomposes the static and dynamic components of the video without the need for explicit optical flow or residual supervision and compresses redundant static information while preserving high-frequency dynamic details. For a given frame index t, we compute the corresponding static code by finding the two closest static codes c^s_i and c^s_j, then performing a weighted sum based on their distances. The corresponding dynamic code is obtained by interpolating dynamic codes $\mathcal{C}^d$ to the video's length and then selecting the corresponding code with index t. In addition, we propose a cross-channel attention-based (CCA) fusion mechanism for efficiently fusing static and dynamic codes.

In summary, our contributions are as follows:

- We propose DS-NeRV, a novel video INR, that decomposes the video into sparse learnable static and dynamic codes, which respectively represent static and dynamic elements within the video. This decomposition appears without the need for explicit optical flow or residual supervision.
- We carefully design the different sampling rates and sampling strategies for two codes to efficiently exploit the characteristics of videos. Moreover, we develop a cross-channel attention-based fusion module to fuse static and dynamic codes for video decoding.
- We conduct extensive experiments on three datasets and various downstream tasks to validate the effectiveness of DS-NeRV. The experimental results demonstrate that DS-NeRV achieves more efficient video modeling over existing INR methods through decomposed static and dynamic codes representation.

2. Related Work

Implicit Neural Representation. The purpose of INR is to model various signals through a function $\mathcal{F}$ that maps the input coordinate θ to corresponding value $y = \mathcal{F}(\theta), \theta \in \mathbb{R}^n, y \in \mathbb{R}^m$. Starting from NeRF [34], INR combined with neural rendering methods have developed rapidly in the field of novel view synthesis for static [5, 6, 8, 19, 35] and dynamic [16, 17, 38, 45] scenes, and 3D reconstruction [33, 36]. Recently, INR have been increasingly applied in the video representation. Different from Siren [41] which maps frame pixel coordinates to their corresponding RGB, NeRV [9] introduces an approach by mapping frame index directly to corresponding video frame, thus enhancing both efficiency and performance. The proposal of NeRV promoted the development of INR for video [4, 10, 11, 18, 20, 21, 25, 30, 54]. In contrast to existing studies that model the video as a whole, DS-NeRV decomposes the video into learnable static and dynamic codes, both of which are jointly learned during training. Thus, DS-NeRV can be seen as a novel INR for videos.

Video Compression. Traditional video compression methods (*e.g*. H.264 [47], HEVC [43]) utilize predictive coding architectures to encode motion information and residual data of videos. With the development of deep learning, video compression algorithms based on neural networks [12, 23, 27, 28, 39, 42, 49, 52] have garnered significant attention. However, these methods are limited to the conventional video compression workflow, severely impacting their capabilities. In NeRV-like methods, the problem of video compression can be converted to a model compression problem. Through techniques such as model pruning, model quantization, and entropy encoding, DS-NeRV achieves comparable performance with traditional video compression approaches and other INR methods.

Latent Optimization For Representation learning. Latent Optimization is employed in generative adversarial networks (GAN) to enhance the quality of samples z [50]. GLO [7] constructs a learnable noise vector for each image in the dataset, thereby offering a novel approach for image generation. This method has also been introduced in the field of novel view synthesis. To improve the reconstruction quality, [24, 31, 44] parameterize scene motion and appearance changes with a compact set of latent codes. Inspired from GLO, DS-NeRV models the static and dynamic elements of videos using learnable codes which resemble the

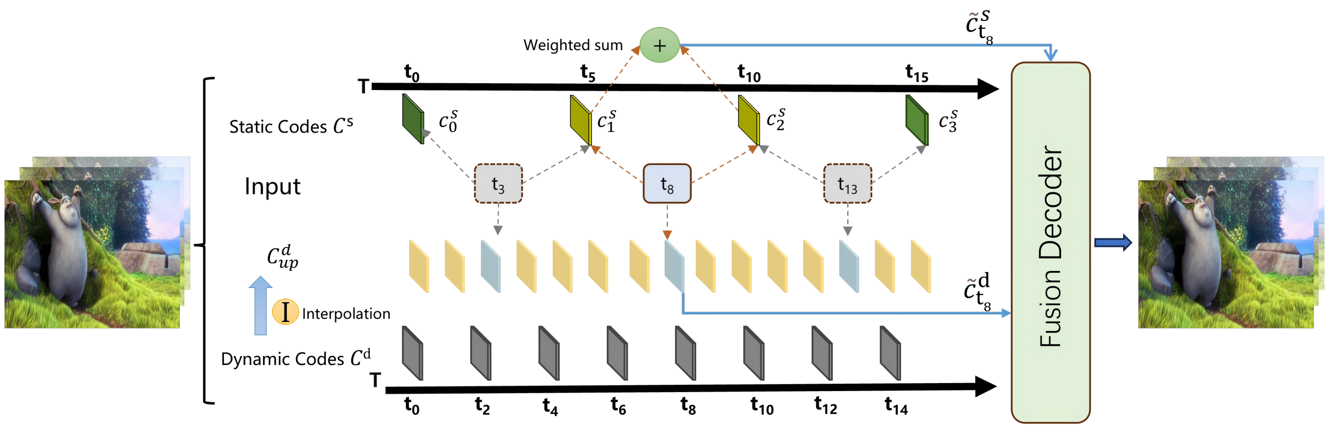

Figure 2. **DS-NeRV framework overview**. DS-NeRV decomposes the video into learnable static and dynamic codes. **Static Codes.** The two orange static codes shown above are the two nearest selected. After weighted sum, they are forwarded to the fusion decoder. **Dynamic Codes.** We interpolate the dynamic codes to match the length of the video. Then the dynamic code corresponding to t is selected in blue.

learnable noise vector in GLO. In this way, DS-NeRV can achieve higher performance in an end-to-end training manner thanks to the greater expressive ability of the codes.

3. Method

3.1. Overview

Given a video sequence $\mathcal{V} = \{v_t\}_{t=0}^{T-1} \in \mathbb{R}^{T\times H\times W\times 3}$, our target is to reconstruct the frame v_t based on the frame index t. To achieve this, we decompose the video into learnable static codes $\mathcal{C}^s \in \mathbb{R}^{l_s\times h_s\times w_s\times dim_s}$ and dynamic codes $\mathcal{C}^d \in \mathbb{R}^{l_d\times h_d\times w_d\times dim_d}$. Given the frame index t, we obtain the corresponding static code $\widetilde{c}_t^s$ by weighted sum and dynamic code $\widetilde{c}_t^d$ through interpolation. The obtained $\widetilde{c}_t^s$ and $\widetilde{c}_t^d$ are then forwarded to the fusion decoder module to reconstruct the frame v_t, as shown in Fig. 2.

3.2. Video Modeling

Traditional video compression pipelines [43, 47] use I-frames (Intra-frames) and P-frames (Predictive frames) for efficient video encoding and decoding. The former contain complete information and are independent of other frames, serving as key reference points in the video sequence. On the other hand, the latter store motion and residual data, relying on the preceding decoded I-frames or P-frames for reference to decode.

Inspired by this design concept, we utilize static codes $\mathcal{C}^s$ with a low sampling rate r_s to represent static elements in the video that can be shared to compress redundancy, while using dynamic codes $\mathcal{C}^d$ with a relatively high sampling rate r_d to represent rich dynamic information.

Static Codes. As Fig. 2 (Top) shows, the static codes $\mathcal{C}^s = \{c_0^s, \cdots, c_i^s, \cdots, c_{l_s-1}^s\}$ is evenly distributed along the timeline at interval z_s. Consequently, given sampling rate r_s, the length of static codes is defined as $l_s = T\cdot r_s$, $l_s \ll T$ and the interval is computed as $z_s = T/l_s$. More sampling details can be found in supplementary material.

According to E-NeRV [25], the MLP used for feature map initialization before NeRV blocks often results in large parameters. To solve this problem, we prefer storing each static code c_i^s in a 3D vector with dimensions $h_s\times w_s\times dim_s$, rather than a 1D vector which will be upsampled to initialize the feature map as adopted in [9, 11]. The 3D vector design eliminates the parameter overhead associated with the MLP before NeRV blocks. In our experiments, we set the size of each static code c_i^s to $4\times 8\times 64$ for $960\times 1920\times 3$ video frame.

The frames between two adjacent static codes can be similarly considered as a GOP (Group of Pictures) [43] in HEVC, containing massive redundant static information that can be shared. So to effectively leverage the information stored in static codes, we design an innovative sampling method to obtain the static code $\widetilde{c}_t^s$ corresponding to frame index t. Given t, instead of solely relying on the nearest static code to obtain static information, we integrate information from two adjacent static codes, summing weighted by their respective distances to t.

As illustrated in Fig. 2, for a given frame index t, we first obtain the two adjacent static codes indices i and j ($0 \le i < j < l_s$) and then calculate their corresponding wights w_i and w_j according to their distances to t.

$$dis_i = \mid t-(i\cdot(z_s+1))\mid, dis_j = \mid t-(j\cdot(z_s+1))\mid \quad (1)$$

$$w_i = \frac{dis_j}{(dis_i+dis_j)}, w_j = \frac{dis_i}{(dis_i+dis_j)} \quad (2)$$

Based on weights and indices, we then perform a weighted sum to obtain the final static code $\widetilde{c}_t^s$ as follows:

$$\widetilde{c}_t^s = w_i\cdot c_i^s + w_j\cdot c_j^s \quad (3)$$

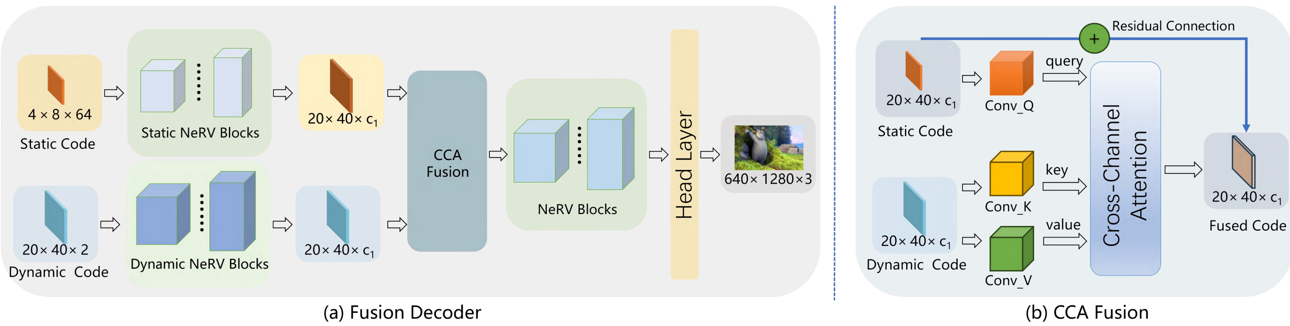

Figure 3. (a) The pipeline of Fusion Decoder. Decoder takes the static code and dynamic code corresponding to index t as input and fuses their information to output frame. (b) Architecture of CCA Fusion Module. The CCA module fuses static code $\widetilde{c}_t^s$ and dynamic code $\widetilde{c}_t^d$ by cross-channel attention.

In this way, the associated static content can be computed for each frame index, effectively sharing static information throughout the video. Additionally, the sparse codes design also helps compress redundant static information.

Dynamic Codes. To characterize the rich dynamic information in the video, the length of dynamic codes is $l_d = T \cdot r_d$ with a higher sampling rate r_d. Similar to static codes representation, we store each dynamic code as a 3D vector to reduce the parameters. Therefore, we set the overall dynamic codes $\mathcal{C}^d$ with a size of $l_d \times h_d \times w_d \times dim_d$. In our experiments, we set the size of each dynamic code to $20 \times 40 \times 2$ for $960 \times 1920 \times 3$ video frame by default.

Different from the sampling method used in the static codes, we obtain the corresponding dynamic code $\widetilde{c}_t^d$ through the interpolated dynamic codes $\mathcal{C}_{up}^d$. The interpolation sampling method establishes global temporal coherence among dynamic codes through internal interaction, aligning with the perceptual continuity of motion in the real world.

Specifically, we firstly interpolate the dynamic codes to match the length of the original video, while keeping the height, width, and number of channels unchanged.

$$\mathcal{C}_{up}^d = interpolate(\mathcal{C}^d, T, h_d, w_d, dim_d) \tag{4}$$

We subsequently retrieval the dynamic code $\widetilde{c}_t^d$ from the interpolated one $\mathcal{C}_{up}^d$ with index t, as depicted in Fig. 2.

Our dynamic codes representation offers low storage overhead by avoiding per-frame code storage while remaining compact total size and realizes the modeling of the global dynamic information through interpolation. Moreover, the interpolation enables the generation of frames that were not seen during training, thereby supporting smooth and meaningful frame interpolation [7, 20].

3.3. Fusion Decoder

Pipeline. Since the obtained static code $\widetilde{c}_t^s$ and dynamic code $\widetilde{c}_t^d$ have different heights and widths, we firstly employ NeRV blocks to align their spatial dimensions, as shown in Fig. 3 (a). They are then forwarded to the cross-channel attention-based (CCA) module for fusion. Once the fusion module integrates information from $\widetilde{c}_t^s$ and $\widetilde{c}_t^d$, the fused code is then processed by the stacked NeRV Blocks to progressively upsample to the corresponding frame.

CCA Fusion. When considering fusion, a natural way to fuse $\widetilde{c}_t^s$ and $\widetilde{c}_t^d$ is to simply add them together. However, this is not an appropriate approach [18] as they encode features from different domains, where static codes capture the static information but dynamic codes represent the motion-related information. To fuse the two types of information effectively, inspired by cross attention [55] and channel attention [48], we design a CCA fusion module based on cross-channel attention mechanism.

Compared to the more commonly used spatial attention [15], we choose channel attention because during the CCA fusion stage, the spatial dimensions of $\widetilde{c}_t^s$ and $\widetilde{c}_t^d$ are identical, but their channels are different. We can think that at the same spatial position (u,v) in two codes, each code represents the static or dynamic information corresponding to the same region in original frame. Therefore, in video representation task, we do not focus the interaction between different positions (u_1, v_1) and (u_2, v_2) in the two codes as this interaction does not contribute to the fusion between two features with the same spatial distribution. Instead, we prioritize the interaction between different channels of the two codes given their distinct channel structures. Hence, we choose cross-channel attention to capture the information interaction between two codes for effective fusion.

Specifically, as illustrated in Fig. 3 (b), we treat each channel in the static code $\widetilde{c}_t^s$ as a query and each channel in the dynamic code $\widetilde{c}_t^d$ as a key-value pair. To achieve this, we firstly utilize three convolutions to extract the query, key, and value components from $\widetilde{c}_t^s$ and $\widetilde{c}_t^d$. Subsequently, we flatten these components along the spatial dimension to do channel attention, as follows:

$$Q(t) = Flatten(Conv_q(\widetilde{c}_t^s)) \tag{5}$$

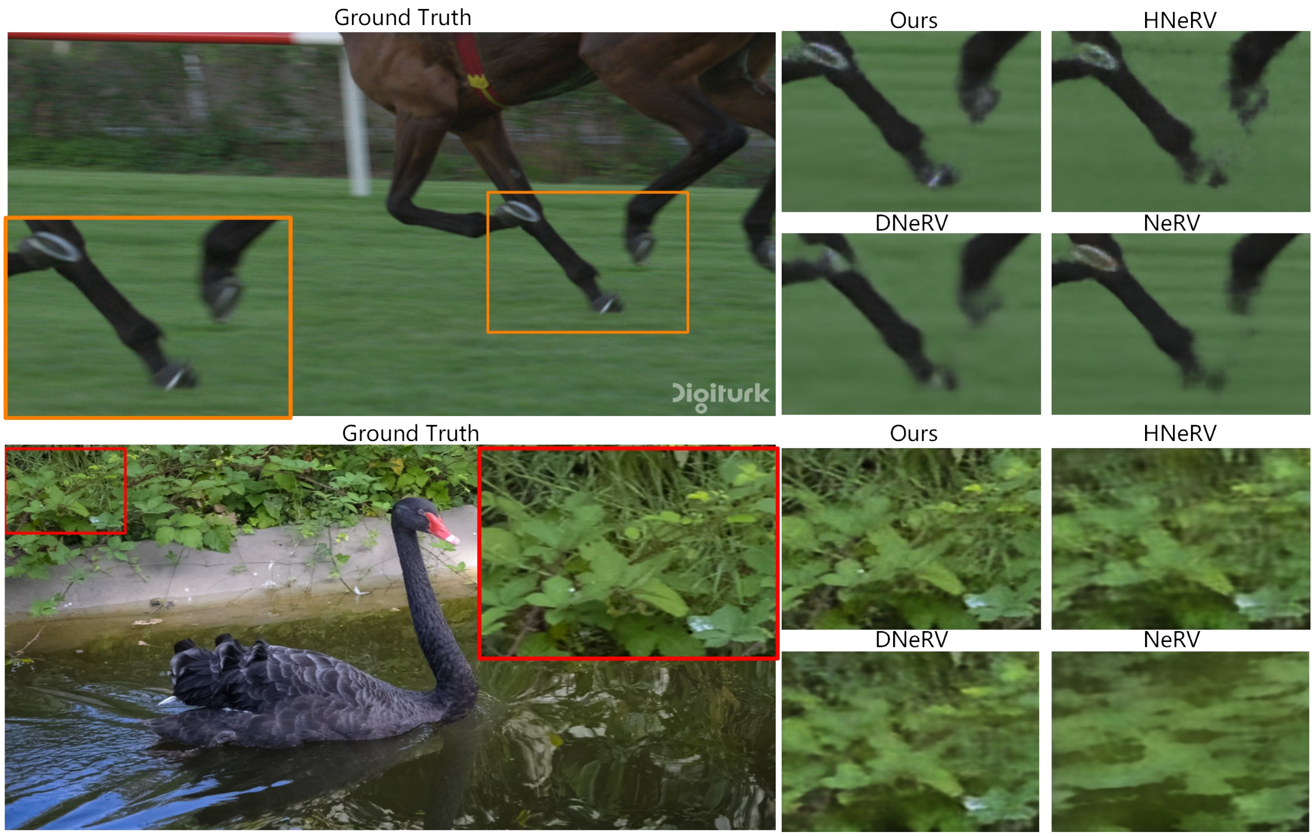

Figure 4. **Video reconstruction results on UVG and DAVIS.** (Top) Jockey. (Bottom) Blackswan.

$$K(t), V(t) = Flatten(Conv_k(\widetilde{c}_t^d)), Flatten(Conv_v(\widetilde{c}_t^d)) \tag{6}$$

After the attention mechanism, we integrate the static code $\widetilde{c}_t^s$ into the obtained attention output through residual connections.

$$FusedCode(t) = softmax(QK^T)V + \widetilde{c}_t^s \tag{7}$$

4. Experiments

4.1. Setup

Datasets. Extensive experiments are conducted on the Big Buck Bunny [40], UVG [32] and DAVIS [37] datasets. The Bunny dataset has 132 frames with size of 720×1280. The UVG dataset has 7 videos with resolution of 960×1920 and lengths of 600 or 300. We select 10 videos from the DAVIS dataset for additional testing, which have a fewer frame number. For a fair comparison, we follow the settings in [11] to crop the Bunny to 640×1280 and crop the UVG and DAVIS to 960×1920 and also crop a 480×960 version of UVG for additional comparison. More details please refer to supplementary material.

Evaluation. We employ PSNR and MS-SSIM [46] as metrics to evaluate video reconstruction quality, and bits per pixel (bpp) as an indicator of video compression performance. We conduct a comparative analysis between DS-NeRV and other implicit methods, namely NeRV, HNeRV, and DNeRV, in terms of video reconstruction as well as various downstream tasks, including video interpolation and inpainting. Moreover, we compare video compression performance with existing compression techniques.

Loss Functions. We only use the L2 loss functions to supervise DS-NeRV, i.e., the decomposition of static codes and dynamic codes is unsupervised.

$$L_2(y_i, \hat{y}_i) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{8}$$

where y_i represents the ground truth and $\hat{y}_i$ is the reconstructed frame.

Implementation Details. During training, we use the Adan [51] optimizer with betas as (0.98,0.92,0.99) and a weight decay of 0.02. Moreover, the learning rate is set to 7×10^{-3} , and we employ a cosine annealing learning rate schedule with a warm-up ratio of 0.2. We empirically find that setting the learning rate of static and dynamic code to 10 times the learning rate can achieve better results. We employ a batch size of 1 on Bunny and DAVIS, while a batch size of 8 on UVG. Unless stated otherwise, all models are

sizes	0.35M	0.75M	1.5M	3M
NeRV[9]	26.59	28.70	30.60	34.37
DNeRV[54]	27.34	30.01	31.19	34.09
HNeRV[11]	29.78	32.35	35.20	37.74
Ours	**31.20**	**33.82**	**36.44**	**38.65**

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

epochs	100	150	200	250	300
NeRV[9]	24.89	25.72	26.26	26.53	26.59
DNeRV[54]	25.32	26.43	27.01	27.27	27.34
HNeRV[11]	27.67	28.62	29.32	29.69	29.78
Ours	**28.35**	**29.18**	**30.03**	**30.89**	**31.20**

(b) PSNR(↑) On Bunny with varying **epochs**.

960x1920	Beauty	Bosph	Honey	Jockey	Ready	Shake	Yacht	avg.
NeRV[9]	33.33	33.34	38.79	28.97	23.89	33.89	27.05	31.32
DNeRV[54]	33.16	32.96	38.43	31.08	24.76	33.71	27.30	31.63
HNeRV[11]	33.88	35.02	39.41	31.69	25.72	34.95	29.09	32.82
Ours	**33.97**	**35.22**	**39.56**	**32.86**	**27.10**	**35.04**	**29.40**	**33.31**

(c) PSNR(↑) On UVG at resolution 960x1920.

480x960	Beauty	Bosph	Honey	Jockey	Ready	Shake	Yacht	avg.
NeRV[9]	34.98	34.98	40.73	31.23	24.92	34.95	28.59	32.91
DNeRV[54]	34.48	33.9	38.66	31.36	25.30	33.00	28.56	32.18
HNeRV[11]	**35.42**	36.13	41.47	32.64	26.54	36.04	30.22	34.07
Ours	35.37	**36.25**	**41.67**	**33.48**	**27.82**	**36.14**	**30.33**	**34.44**

(d) PSNR(↑) On UVG at resolution 480x960.

Table 1. Video **reconstruction** results on Bunny and UVG.

Video	b-swan	b-trees	boat	b-dance	camel	c-round	c-shadow	cows	dance	dog	avg.
NeRV[9]	25.04	25.22	30.25	25.78	23.69	24.08	25.29	22.44	25.61	27.15	25.30
DNeRV[54]	29.84	28.73	30.52	26.58	26.24	28.50	28.88	24.44	28.42	30.64	27.79
HNeRV[11]	29.23	28.67	32.27	31.39	25.93	28.72	31.21	24.67	28.43	30.72	28.91
Ours	**32.55**	**29.76**	**34.39**	**32.21**	**27.26**	**29.48**	**35.88**	**25.08**	**28.79**	**33.29**	**30.36**

Table 2. Video **reconstruction** results on DAVIS, PSNR(↑) reported.

3M and trained for 300 epochs. All experiments are performed on the Tesla V100. More implementation details can be found in the supplementary material.

4.2. Video Reconstruction

We first compare DS-NeRV with other INR methods on Bunny, UVG and DAVIS. As shown in Tab. 1a, we evaluate video reconstruction for various model sizes with 300 epochs on Bunny. Remarkably, DS-NeRV achieves impressive video reconstruction quality with a PSNR of 31.20, despite having only 0.35M parameters. Furthermore, we evaluate the reconstruction performance with different epochs with a fixed model size of 0.35M, as presented in Tab. 1b and Fig. 1 (right), from which we can see that DS-NeRV converges faster with a higher performance.

We subsequently extend our evaluation on UVG and DAVIS, with qualitative results shown in Fig. 4. DS-NeRV achieves clearer contour reconstruction for the horseshoe in Jockey. Additionally, DS-NeRV captures high-frequency texture details for the leaves in Blackswan, while other methods exhibit noticeable artifacts. This is mainly attributed to our proposed static and dynamic codes representation, which efficiently preserves more detail with compact model size through the utilization of the shared static information and the global temporal-coherence within the video. More quantitative experimental results are listed in Tabs. 1c, 1d and 2, which demonstrate a significant improvement achieved by DS-NeRV when compared to other methods, especially in two high-dynamic videos Ready and Jockey. We also provide experiment results on the standard UVG of 1080p in supplementary material.

4.3. Video Inpainting

We further investigate the video inpainting on DAVIS. Following the configuration in DNeRV [54], we apply masks to the original videos using either five boxes with the size of 50x50 or a central mask with dimensions equal to 1/4 of the width and height of original video. For DS-NeRV and HNeRV, the model is trained on the masked video, while DNeRV is trained on the original video according to their setting. All methods are tested on the masked videos and the qualitative details of inpainted frames are shown in Fig. 5. Note that the windows in Car-shadow (Top) and water flow in Boat (Bottom) are masked in some certain video frames. DS-NeRV almost perfectly inpaints the masked areas thanks to the utilization of global temporal coherence and its capacity to learn and then fill the masked areas using information visible in other frames. Moreover, DS-NeRV successfully reconstruct high-frequency details, such as the manhole cover in Car-shadow and the distant electric wire tower in Boat. More quantitative experimental results are presented in the Tab. 3, demonstrating the superiority of our proposed DS-NeRV over other methods.

4.4. Video Interpolation

We use even-numbered frames from the video as the training set and odd-numbered frames as the test set to conduct the interpolation experiment. During testing, DS-NeRV utilizes trained interpolated static and dynamic codes as inputs, in this way, our method can naturally generalize to frames that are not seen in the training set. The way we conduct the test is similar to typically video interpolation task [29, 53], where the frames to be interpolated are not

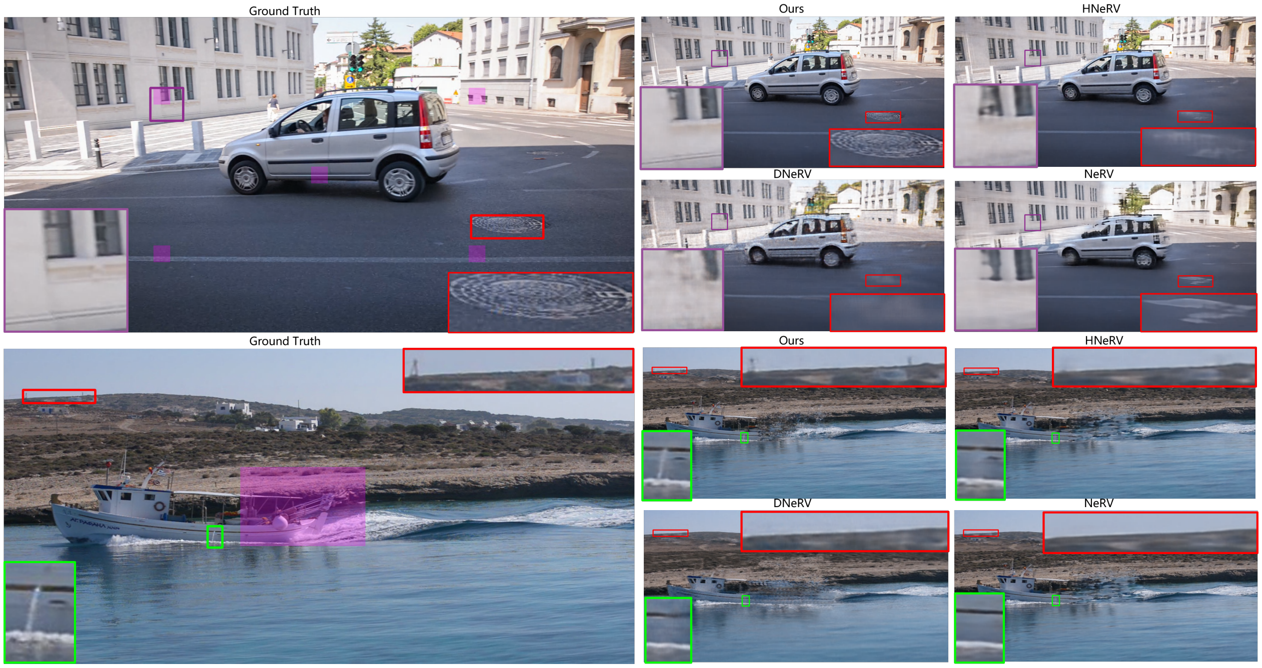

Figure 5. **Video inpainting results on DAVIS.** (Top) Car-Shadow with 5 masks of width 50.(Bottom) Boat with a central mask with width and height both 1/4 of the video.

Video	b-swan	b-trees	boat	b-dance	camel	c-round	c-shadow	cows	dance	dog	avg.
NeRV[9]	24.98	25.16	30.12	25.53	23.65	24.05	25.17	22.38	25.46	27.05	25.36
DNeRV[54]	29.52	28.14	29.52	25.76	25.48	28.00	25.66	24.05	27.81	26.44	27.03
HNeRV[11]	29.10	28.67	29.10	28.67	26.07	28.31	30.92	24.4	28.44	30.58	28.90
Ours	**32.28**	**29.58**	**34.09**	**31.50**	**27.21**	**29.34**	**35.35**	**24.99**	**28.64**	**33.03**	**30.60**

(a) PSNR(↑) On DAVIS with disperse mask.

Video	b-swan	b-trees	boat	b-dance	camel	c-round	c-shadow	cows	dance	dog	avg.
NeRV[9]	22.72	22.44	25.56	20.79	20.96	20.97	22.15	20.58	21.34	24.00	22.15
DNeRV[54]	26.47	21.71	24.74	21.96	23.10	**24.41**	**28.25**	22.06	23.12	24.03	23.99
HNeRV[11]	26.16	24.21	25.96	22.20	22.61	22.38	16.32	21.84	22.56	26.05	23.03
Ours	**28.33**	**25.42**	**27.71**	**22.96**	**23.36**	24.08	24.89	**22.71**	**23.31**	**27.83**	**25.06**

(b) PSNR(↑) On DAVIS with central mask.

Table 3. Video **inpainting** results on DAVIS.

visible during training and testing. However, HNeRV and DNeRV use test frames itself as input during testing to obtain embeddings, which are subsequently used to generate corresponding ground truth, which is not practical because the test frames are typically unknown. The quantitative results on the training and test sets are shown in Tab. 4, which demonstrates the superior performance of DS-NeRV on the training set compared to existing methods. Furthermore, DS-NeRV also achieves comparable performance on the test set even without seeing the ground truth during testing. The qualitative results on interpolation can be found in Fig. 6, where DS-NeRV achieves better reconstruction of the flag pattern and exhibits clearer contour in the human head region.

4.5. Video Compression

We follow the process in HNeRV to compress the model through model quantization, model pruning, and entropy coding. We compare DS-NeRV with H.264 [47], HEVC [43], NeRV [9] and HNeRV [11]. We present the results of video compression in Fig. 7. From the figure we can see that DS-NeRV surpasses HNeRV, exhibiting significant improvements. Additionally, in many cases, our method outperforms traditional methods such as H.264 and HEVC, achieving superior performance. The experimental results validate the effectiveness of our compression strategy.

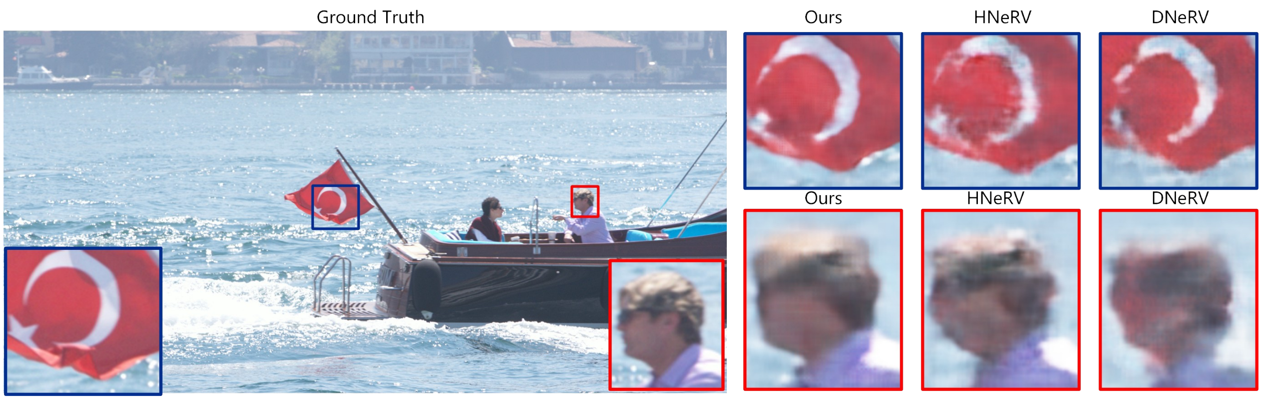

Figure 6. **Video interpolation results on UVG, interpolated frame shown above.** (a) Ours. (b) HNeRV. (c) DNeRV.

video	Beauty	Bosph	Honey	Jockey	Ready	Shake	Yacht	avg.
HNeRV[11]	34.02/31.26	34.69/34.54	39.26/39.10	32.58/22.86	26.25/20.51	**34.91**/32.79	29.20/27.41	32.99/29.78
DNeRV[54]	33.46/**32.48**	30.96/30.77	38.55/38.36	32.22/**29.79**	25.78/**24.29**	34.41/**33.34**	26.37/25.96	31.68**30.71**
Ours	**34.08**/31.84	**34.96/34.82**	**39.48/39.27**	**33.60**/22.96	**27.48**/21.26	34.54/33.17	**29.55/27.52**	**33.30**/30.09

Table 4. Video **interpolation** results on UVG with train/test split, PSNR(↑) reported.

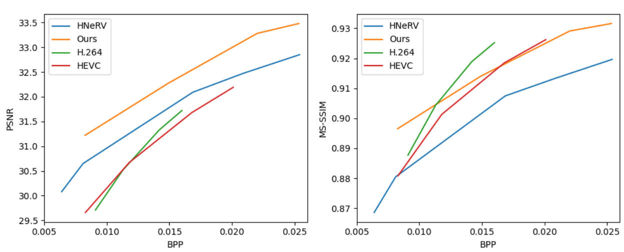

Figure 7. Compression performance on UVG dataset.

$t_s \backslash t_d$	0	100	200	300
0	29.91/38.80	30.78/39.49	31.10/39.41	31.62/39.21
30	28.68/39.13	30.87/39.53	32.16/39.52	32.56/39.43
60	29.91/39.23	31.04/39.53	32.25/39.52	32.75/39.43
90	30.88/39.38	31.19/39.54	32.23/39.52	32.69/39.42

Table 5. Ablation study for codes length on Jockey/Honey. The (0,0) combination refers to vanilla NeRV.

4.6. Ablation Studies

Static/Dynamic codes. To evaluate the effectiveness of the static and dynamic code designs and the impact of their lengths on video reconstruction, we conduct ablation experiments on Jockey and Honey from the UVG. Jockey exhibits strong dynamics, while Honey features nearly static video frames.

The results of the ablation experiments are presented in the Sec. 4.6. The results demonstrate the varying effects of different combinations of static and dynamic code lengths on videos with different levels of dynamics. For Jockey, increasing the length of the dynamic codes gradually improves the video quality, while the effect is less pronounced for Honey. When one of the lengths is set to 0, indicating the absence of the corresponding code, it further confirms that both static and dynamic codes are essential elements for achieving high-quality reconstruction, highlighting the necessity of their collaboration. Appropriately setting the lengths under a certain model size enables the model to fully utilize and compress the redundant static information contained in static codes and the dynamic information in dynamic codes. The static and dynamic parts of the video, decoded from the corresponding static and dynamic codes, are shown in the Fig. 1 (Left). More ablation results can be found in the supplementary material.

5. Conclusion

In this paper, we propose DS-NeRV, a novel INR for video, that decomposes the video into sparse, learnable static and dynamic codes. By computing a weighted sum of the static codes and interpolating the dynamic codes, DS-NeRV effectively utilizes the redundancy of static information in videos and models global temporal-coherent dynamic information. According to our extensive experiment, DS-NeRV outperforms the state-of-the-art methods in many downstream tasks.

Acknowledgements. This work is support in part by the National Science Fund for Distinguished Young Scholars (No. 62325208), in part by NSFC Joint Funds (No. U2001204), and in part by the National Natural Science Foundation of China (No. 62232002, 62072330).

References

[1] Cisco: Service provider solutions. https://www.cisco.com/c/en/us/solutions/service-provider/index.html#~products-and-solutions. 1

[2] 2023 global internet phenomena report. https://www.sandvine.com/phenomena. 1

[3] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2020. 1

[4] Yunpeng Bai, Chao Dong, Cairong Wang, and Chun Yuan. Ps-nerv: Patch-wise stylized neural representations for videos. In *2023 IEEE International Conference on Image Processing (ICIP)*, pages 41–45. IEEE, 2023. 1, 2

[5] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[6] Wenjing Bian, Zirui Wang, Kejie Li, Jia-Wang Bian, and Victor Adrian Prisacariu. Nope-nerf: Optimising neural radiance field with no pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4160–4169, 2023. 2

[7] Piotr Bojanowski, Armand Joulin, David Lopez-Paz, and Arthur Szlam. Optimizing the latent space of generative networks. *arXiv preprint arXiv:1707.05776*, 2017. 2, 4

[8] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision*, pages 333–350. Springer, 2022. 2

[9] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021. 1, 2, 3, 6, 7

[10] Hao Chen, Matt Gwilliam, Bo He, Ser-Nam Lim, and Abhinav Shrivastava. Cnerv: Content-adaptive neural representation for visual data. *arXiv preprint arXiv:2211.10421*, 2022. 2

[11] Hao Chen, Matthew Gwilliam, Ser-Nam Lim, and Abhinav Shrivastava. Hnerv: A hybrid neural representation for videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10270–10279, 2023. 1, 2, 3, 5, 6, 7, 8

[12] Tong Chen, Haojie Liu, Qiu Shen, Tao Yue, Xun Cao, and Zhan Ma. Deepcoder: A deep neural network based video compression. In *IEEE Visual Communications and Image Processing Conference*, 2017. 2

[13] Yue Chen, Debargha Murherjee, Jingning Han, Adrian Grange, Yaowu Xu, Zoe Liu, Sarah Parker, Cheng Chen, Hui Su, Urvang Joshi, et al. An overview of core coding tools in the av1 video codec. In *2018 picture coding symposium (PCS)*, pages 41–45. IEEE, 2018. 1

[14] Zhibo Chen, Tianyu He, Xin Jin, and Feng Wu. Learning for video compression. *IEEE Transactions on Circuits and Systems for Video Technology*, 30(2):566–576, 2019. 1

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[16] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 2

[17] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12479–12488, 2023. 2

[18] Bo He, Xitong Yang, Hanyu Wang, Zuxuan Wu, Hao Chen, Shuaiyi Huang, Yixuan Ren, Ser-Nam Lim, and Abhinav Shrivastava. Towards scalable neural representation for diverse videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6132–6142, 2023. 2, 4

[19] Hanwen Jiang, Zhenyu Jiang, Kristen Grauman, and Yuke Zhu. Few-view object reconstruction with unknown categories and camera poses. *arXiv preprint arXiv:2212.04492*, 2022. 2

[20] Subin Kim, Sihyun Yu, Jaeho Lee, and Jinwoo Shin. Scalable neural video representations with learnable positional features. *Advances in Neural Information Processing Systems*, 35:12718–12731, 2022. 2, 4

[21] Joo Chan Lee, Daniel Rho, Jong Hwan Ko, and Eunbyung Park. Ffnerv: Flow-guided frame-wise neural representations for videos. *arXiv preprint arXiv:2212.12294*, 2022. 2

[22] D LEGALL. A video compression standard for multimedia applications. *Commun. ACM*, 34:226–252, 1993. 1

[23] Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. *Advances in Neural Information Processing Systems*, 34:18114–18125, 2021. 2

[24] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5521–5531, 2022. 2

[25] Zizhang Li, Mengmeng Wang, Huaijin Pi, Kechun Xu, Jianbiao Mei, and Yong Liu. E-nerv: Expedite neural video representation with disentangled spatial-temporal context. In *European Conference on Computer Vision*, pages 267–284. Springer, 2022. 1, 2, 3

[26] Jianping Lin, Dong Liu, Houqiang Li, and Feng Wu. M-lvc: Multiple frames prediction for learned video compression.

In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3546–3554, 2020. 1

[27] Zhenyu Liu, Xianyu Yu, Yuan Gao, Shaolin Chen, Xiangyang Ji, and Dongsheng Wang. Cu partition mode decision for hevc hardwired intra encoder using convolution neural network. *IEEE Transactions on Image Processing*, 25 (11):5088–5103, 2016. 2

[28] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. Dvc: An end-to-end deep video compression framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 1, 2

[29] Liying Lu, Ruizheng Wu, Huaijia Lin, Jiangbo Lu, and Jiaya Jia. Video frame interpolation with transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3532–3542, 2022. 6

[30] Shishira R Maiya, Sharath Girish, Max Ehrlich, Hanyu Wang, Kwot Sin Lee, Patrick Poirson, Pengxiang Wu, Chen Wang, and Abhinav Shrivastava. 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*, pages 14378–14387, 2023. 2

[31] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 2

[32] 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*, pages 297–302, 2020. 5

[33] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4460–4470, 2019. 2

[34] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 1, 2

[35] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics (ToG)*, 41(4):1–15, 2022. 2

[36] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 2

[37] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 724–732, 2016. 5

[38] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1, 2

[39] Oren Rippel, Alexander G Anderson, Kedar Tatwawadi, Sanjay Nair, Craig Lytle, and Lubomir Bourdev. Elf-vc: Efficient learned flexible-rate video coding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14479–14488, 2021. 2

[40] Ton Roosendaal. Big buck bunny. In *ACM SIGGRAPH ASIA 2008 computer animation festival*, pages 62–62. 2008. 5

[41] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020. 1, 2

[42] Rui Song, Dong Liu, Houqiang Li, and Feng Wu. Neural network-based arithmetic coding of intra prediction modes in hevc, 2017. 2

[43] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (hevc) standard. *IEEE Transactions on circuits and systems for video technology*, 22(12):1649–1668, 2012. 1, 2, 3, 7

[44] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P Srinivasan, Jonathan T Barron, and Henrik Kretzschmar. Block-nerf: Scalable large scene neural view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8248–8258, 2022. 2

[45] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. 2

[46] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, pages 1398–1402. Ieee, 2003. 5

[47] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the h. 264/avc video coding standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):560–576, 2003. 1, 2, 3, 7

[48] Sanghyun Woo, Jongchan Park, Joon-Young Lee, and In So Kweon. Cbam: Convolutional block attention module. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 4

[49] Chao-Yuan Wu, Nayan Singhal, and Philipp Krahenbuhl. Video compression through image interpolation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 2

[50] Yan Wu, Jeff Donahue, David Balduzzi, Karen Simonyan, and Timothy Lillicrap. Logan: Latent optimisation for generative adversarial networks. *arXiv preprint arXiv:1912.00953*, 2019. 2

[51] Xingyu Xie, Pan Zhou, Huan Li, Zhouchen Lin, and Shuicheng Yan. Adan: Adaptive nesterov momentum algorithm for faster optimizing deep models. *arXiv preprint arXiv:2208.06677*, 2022. 5

[52] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for video compression with hierarchical quality and recurrent enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6628–6637, 2020. 1, 2

[53] Guozhen Zhang, Yuhan Zhu, Haonan Wang, Youxin Chen, Gangshan Wu, and Limin Wang. Extracting motion and appearance via inter-frame attention for efficient video frame interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5682–5692, 2023. 6

[54] Qi Zhao, M Salman Asif, and Zhan Ma. Dnerv: Modeling inherent dynamics via difference neural representation for videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2031–2040, 2023. 1, 2, 6, 7, 8

[55] Brady Zhou and Philipp Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13760–13769, 2022. 4