Beyond Perspective: Neural 360-Degree Video Compression

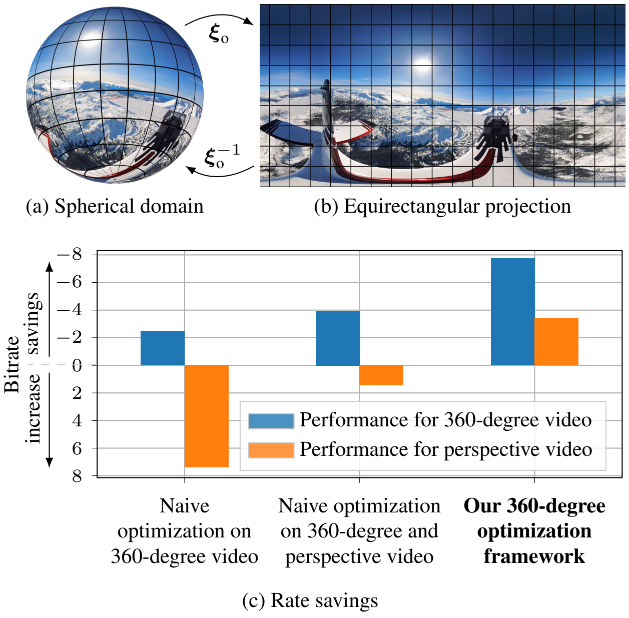
Andy Regensky, Marc Windsheimer, Fabian Brand, André Kaup
Multimedia Communications and Signal Processing
Friedrich-Alexander-Universität Erlangen-Nürnberg, Germany
andy.regensky, marc.windsheimer, fabian.brand, andre.kaup@fau.de

Abstract

Neural video codecs (NVCs) have seen fast-paced advancement in recent years and already perform close to state-of-the-art traditional video codecs like H.266/VVC. However, NVC investigations have so far focused on improving performance for classical perspective video leaving the increasingly important 360-degree video format unexplored. In this paper, we address this issue and present how existing NVCs can be optimized for 360-degree video while also improving performance on perspective video. As no suitable datasets for neural 360-degree video compression exist, we publish a large-scale 360-degree video dataset consisting of more than 6000 user generated 9-frame sequences with resolutions ranging from 0.5K to 8K. We propose a novel method for training data augmentation exploiting the spherical characteristics of 360-degree video that shows to be crucial for achieving maximum compression performance. An additional positional feature encoding further supports the NVC in dynamic bitrate allocation notably improving the performance for both 360-degree and perspective video. Overall, we achieve rate savings of almost 8% for 360-degree video and more than 3% for perspective video with minimal complexity overhead. The dataset is available at: https://huggingface.co/datasets/FAU-LMS/UGC360. Source code and pre-trained model weights are available at: https://github.com/FAU-LMS/NVC360.

1. Introduction

360-degree video allows to provide a feeling of presence and immersion that is unprecedented in traditional perspective video technologies. This feeling of presence has shown to support focus, emotional attachment, and motivation in diverse application areas including education, communication, and entertainment [5, 25, 46]. However, 360-degree video requires immense resolutions in order to match the human visual systems' angular resolution such that individual pixels are indistinguishable. While this effect is reached with 8K video for conventional planar perspective video [15], resolutions of up to 21K are required in order to reach the same effect with 360-degree video [12]. This leads to major challenges in storing and transmitting 360-degree videos and highlights the importance of efficient 360-degree video compression technologies.



(a) Spherical domain (b) Equirectangular projection

(c) Rate savings

Figure 1. 360-degree image in (a) spherical domain representation and (b) equirectangular projection format. Bjøntegaard Delta rate savings (c) of our proposed framework for optimizing neural video codecs for 360-degree video. Tested with DCVC-HEM [29] using the JVET360 [19], HEVC [7], and UVG [37] datasets. The anchor is DCVC-HEM trained on perspective video.

As 360-degree video provides image information for an all-around field of view, its natural representation lies on the surface of a sphere in 3D space as shown in Fig. 1(a). Most state-of-the-art image and video processing tech-

The authors gratefully acknowledge that this work has been supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under project number 418866191.

niques, however, have been specifically designed for planar 2D image and video formats. 360-degree video is thus typically stored and processed in a projected 2D format such as the popular equirectangular projection format (ERP) shown in Fig. 1(b). The mapping from the spherical domain to the projection plane is defined by the projection function $\boldsymbol{\xi}_\text{o}$. With the inverse projection function $\boldsymbol{\xi}_\text{o}^{-1}$, the content can be projected back to the spherical domain, allowing, e.g., to render viewports aligned with a user's head orientation.

In their projected 2D representation, 360-degree videos can readily be compressed by existing state-of-the-art video compression techniques like the hybrid video coding standards H.265/HEVC [52] or H.266/VVC [9], or recent neural video codecs like the DCVC series [28–31, 50]. For hybrid video codecs, various coding tools have been investigated that improve the compression efficiency of 360-degree video by taking their special characteristics into account. This includes, among others, alternative projection formats [64], adjusted bit allocation schemes [20, 66], and 360-degree specific motion models [14, 32, 43, 58].

For NVCs, there are two general approaches that can be followed. First, provide specialized NVCs for every content category. Second, provide a versatile NVC that addresses a variety of content categories. The first approach would allow highly specialized adjustments for 360-degree video even if they break compatibility with perspective video. The second approach would simplify standardization and industry adoption, as only one codec would have to be developed and maintained. The question, which of these approaches will be favorable, cannot be answered yet.

We contribute to this field of research and show that it is possible to optimize an NVC on a new content category (360-degree) while maintaining and even improving the performance for an existing content category (perspective). This mutual benefit is possible, because there is a significant overlap of both content domains. Both represent frame-by-frame image data. Though high-level features like spatial distortions in 360-degree videos differ from perspective video, low-level features like edges, textures or gradients are similar. However, as Fig. 1(c) shows, a naive optimization on 360-degree video or 360-degree and perspective video lacks considerably behind the possibilities of our proposed 360-degree optimization framework. The framework we propose is applicable to any NVC and is based on three key contributions:

- We publish a large-scale 360-degree training dataset (UGC360) consisting of more than 6000 full-frame user-generated 360-degree video sequences with 9 frames per sequence and resolutions ranging from 0.5K to 8K, eliminating the lack of 360-degree video training data.
- We propose flow-guided reprojection, a novel data augmentation method for 360-degree video that shows to be essential to reach the highest possible compression performance for both 360-degree and perspective video.
- We introduce a positional feature encoding into the entropy model that supports the model in dynamic bitrate allocation and further improves compression performance for both 360-degree and perspective video. It is a lightweight extension that requires only minimal finetuning for integration into an existing NVC

As shown in Fig. 1(c), we reach average rate savings of almost 8% for 360-degree video and more than 3% for perspective video by employing our proposed 360-degree optimization framework with the DCVC-HEM [29] compression model.

2. Related Work

Since the first pioneering investigations on end-to-end learned neural image codecs (NICs) in 2017 [3, 55], they have evolved significantly [4, 11, 26, 39] and already outperform state-of-the-art image codecs like VVC intra [9]. Motivated by this, JPEG started the standardization process for an AI-based image codec called JPEG AI [1, 2], which is expected to be released in early 2025 [1, 22].

As an extension to NICs, neural video codecs (NVCs) exploit temporal correlations during coding to further improve compression efficiency. Common concepts for incorporating temporal correlations are residual coding and conditional coding. The early DVC and DVCPro [34, 35] follow a residual coding concept which is inspired by traditional hybrid video codecs. A motion compensated prediction is formed based on estimated optical flow, subtracted from the current frame, and only the residual is coded. The state-of-the-art DCVC series [28–31, 41, 50] follows a conditional coding concept [8], where context for coding the current frame is provided as side information to the encoder, decoder and entropy model. This context is generated from the last decoded frame or feature space using a context generation network. Similar to residual coding, it incorporates an optical flow-based motion compensation.

While considerable efforts have gone into investigating improved 360-degree video compression in hybrid video codecs [43–45, 47, 48, 57, 64], NVCs have not yet been optimized and aligned for 360-degree video. One of the main obstacles for investigating NVCs for 360-degree video compression is the lack of suitable training datasets. While some 360-degree video datasets exist [10, 27, 54, 59, 62], none provide sufficient unique and high-quality 360-degree video clips suitable for training competitive NVCs.

However, several related approaches exist for improving NICs for 360-degree image compression. In [33], *Li et al.* propose to estimate a code structure map that is generated based on image content and 360-degree image latitude. This code structure map determines how many latent channels are coded in the bitstream for each latent position to reduce bits spent on less important channels. In the context of im-

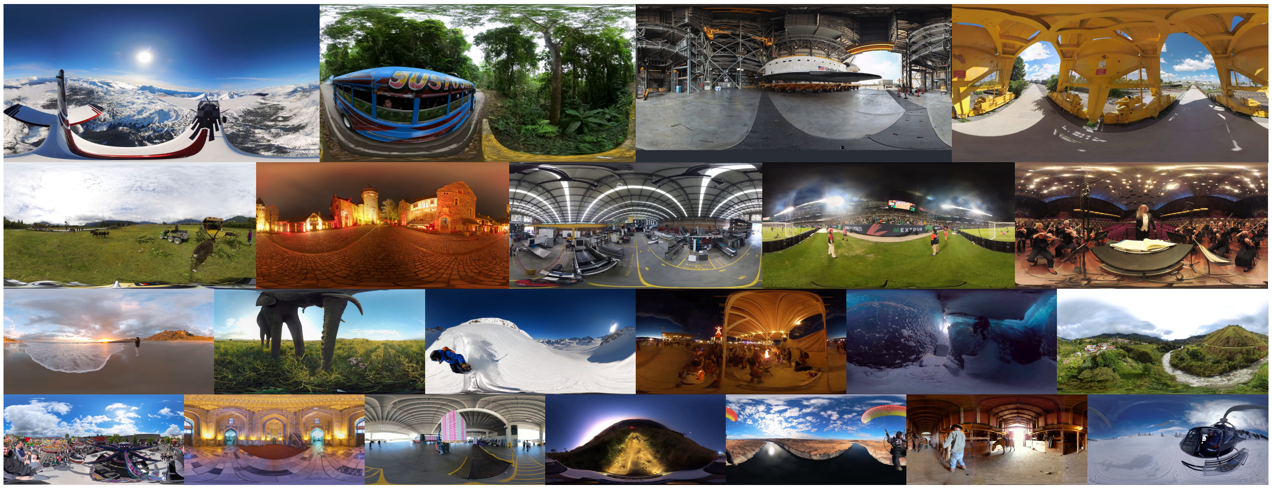


Figure 2. Sample frames from our UGC360 dataset showcasing its diverse content.

age coding for machines, *Zheng et al.* propose to estimate the significance of different pixels for the 360-degree machine task at hand at the encoder and use that to optimize the bit distribution in the bitstream [65]. The estimation is based on content and 360-degree image latitude. A similar idea is followed in [18], where *Gungordu and Tekalp* perform a saliency estimation at the encoder optimizing the bit distribution for the human observer. Similar to our proposed approach, all of these works incorporate position information in order to improve the 360-degree image compression performance. However, unlike our proposed approach, they focus on 360-degree images only and do not consider potential losses for classical perspective images.

In [36], *Mahmoudian et al.* follow a different concept, where they perform coding in the spherically uniform HEALPix format [17] with specially designed spherical convolutions. While they achieve notable rate savings for 360-degree images, the resulting NICs are not applicable to perspective images.

3. Method

3.1. 360-Degree Video Dataset

We collected a large-scale 360-degree video dataset for training neural video compression networks. Our UGC360 (user-generated content 360) dataset consists of 6866 user generated 360-degree video clips licensed under Creative Commons [13], which have been collected from the Vimeo [56] (5053 clips) and YouTube [16] (1813 clips) online video platforms. Each clip consists of 9 frames. We provide 9 frames per clip instead of the 7 frames per clip provided by vimeo90k [63] in order to increase the potential frame distance for investigating bi-directional coding concepts similar to traditional video codecs [9, 52, 60]. To ease data handling, the dataset is split intro three subsets UGC360-S, UGC360-M and UGC360-L grouped by resolution as described in Table 1. Overall resolutions range from 640×320 pixels up to 7680×4320 pixels. All clips are in the common equirectangular projection format. The 9-frame clips have been extracted from 1321 unique videos employing scene detection using TransNetV2 [51]. Clips with similar content have been removed from the dataset using their gist descriptors [40] similar to the process used for the vimeo90k dataset [63]. Furthermore, scenes with flat content and low contrast have been excluded. Fig. 2 showcases the diverse nature of our UGC360 dataset, which is publicly accessible via https://huggingface.co/datasets/FAU-LMS/UGC360.

Table 1. UGC360 dataset subsets. Subsets split the overall dataset into smaller parts based on their resolution to ease data handling.

Subset	Resolutions	Minimum	Maximum	Clips
UGC360-S	.5K - 2.5K	640×320	2560×1440	5080
UGC360-M	3K - 6K	3072×1536	5760×2880	1618
UGC360-L	8K	7680×3840	7680×4320	168

3.2. Flow-Guided Reprojection

Reprojection. Typically, NVCs are trained using training patches with limited resolutions that are randomly cropped from full-size training videos. A common training patch resolution is $M \times N = 256 \times 256$ pixels. For 360-degree videos, which capture the full spherical view, we can augment training data not just by cropping from random positions, but also through random rotations of the spherical data in 3D space. As the strength of distortions in the equirectangular projection format varies heavily between the equator and the poles, this data augmentation supports the network in learning a broad range of 360-degree video distortions. All frames of a training video share the same

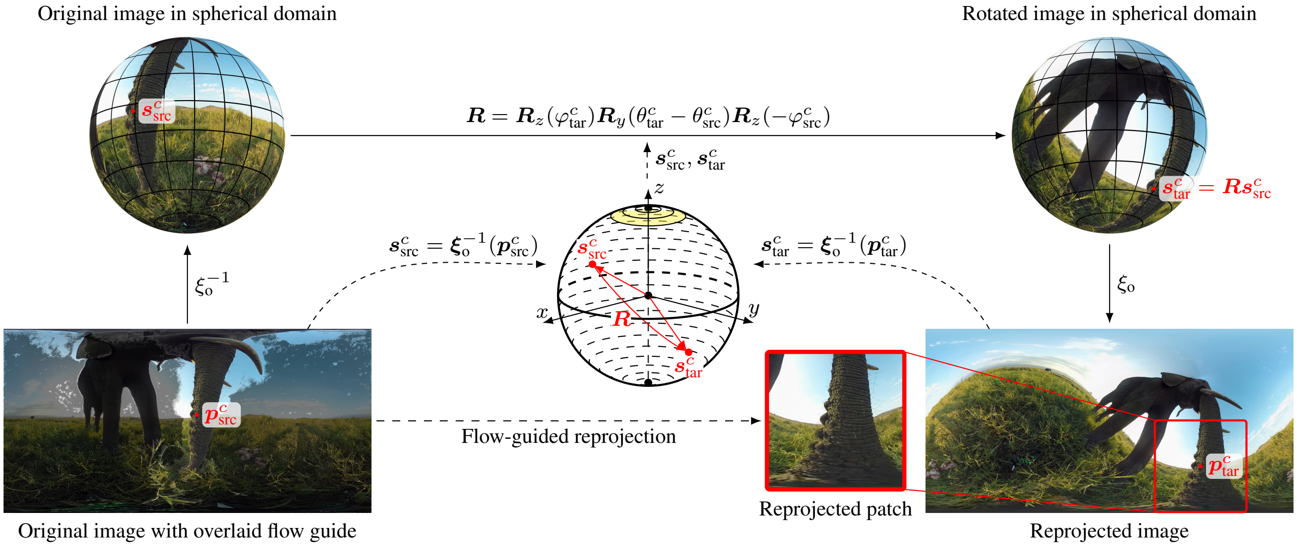

Original image in spherical domain
Rotated image in spherical domain
$\boldsymbol{s}^c_{\text{src}}$
$\boldsymbol{R} = \boldsymbol{R}_z(\varphi^c_{\text{tar}})\boldsymbol{R}_y(\theta^c_{\text{tar}} - \theta^c_{\text{src}})\boldsymbol{R}_z(-\varphi^c_{\text{src}})$
$\boldsymbol{s}^c_{\text{src}}, \boldsymbol{s}^c_{\text{tar}}$
z
$\boldsymbol{s}^c_{\text{tar}} = \boldsymbol{R}\boldsymbol{s}^c_{\text{src}}$
$\boldsymbol{s}^c_{\text{src}} = \boldsymbol{\xi}_{\text{o}}^{-1}(\boldsymbol{p}^c_{\text{src}})$
$\boldsymbol{s}^c_{\text{tar}} = \boldsymbol{\xi}_{\text{o}}^{-1}(\boldsymbol{p}^c_{\text{tar}})$
ξ_{o}^{-1}
ξ_{o}
x
y
$\boldsymbol{R}$
$\boldsymbol{s}^c_{\text{src}}$
$\boldsymbol{s}^c_{\text{tar}}$
$\boldsymbol{p}^c_{\text{src}}$
Flow-guided reprojection
$\boldsymbol{p}^c_{\text{tar}}$
Reprojected patch
Original image with overlaid flow guide
Reprojected image

Figure 3. Visualization of our flow-guided reprojection. In a first step, the source position $\boldsymbol{p}^c_{\text{src}}$ and target position $\boldsymbol{p}^c_{\text{tar}}$ are sampled. The source position is uniformly sampled within the valid region defined by the flow guide. The target position is uniformly sampled in the entire image area. In a second step, the source and target positions are projected onto the unit sphere according to (1) and (2). To align the source position with the target position, the rotation matrix $\boldsymbol{R}$ is then derived according to (3). Applying this rotation to all pixels in the source image yields the reprojected image from which the target patch is extracted.

3D space rotation. The mathematical procedure is detailed in the following.

For any 360-degree video projection format, a corresponding projection function $\boldsymbol{\xi}_{\text{o}} : \mathcal{S} \rightarrow \mathbb{R}^2$ describes the relation between a 3D coordinate $\boldsymbol{s} = (x, y, z)^{\text{T}} \in \mathcal{S}$ on the unit sphere and the corresponding pixel coordinate $\boldsymbol{p} \in (u, v)^{\text{T}} \in \mathbb{R}^2$ on the 2D image plane. $\mathcal{S} = \{\, \boldsymbol{s} \in \mathbb{R}^3 \mid \|\boldsymbol{s}\|_2 = 1\}$ describes the set of all pixel coordinates on the unit sphere. The inverse projection function $\boldsymbol{\xi}_{\text{o}}^{-1} : \mathbb{R}^2 \rightarrow \mathcal{S}$ maps the 2D image plane coordinate back to the unit sphere. This allows to uniquely map 360-degree video between its planar image plane representation and its natural spherical domain representation.

For the reprojected patch extraction, a source patch center position $\boldsymbol{p}^c_{\text{src}} = (u^c_{\text{src}}, v^c_{\text{src}})^{\text{T}} \in \mathcal{I}_{\text{src}}$ in the original image and a target patch center position $\boldsymbol{p}^c_{\text{tar}} = (u^c_{\text{tar}}, v^c_{\text{tar}})^{\text{T}} \in \mathcal{I}_{\text{tar}}$ in the reprojected image are randomly sampled. $\mathcal{I}_{\text{src}}$ and $\mathcal{I}_{\text{tar}}$ denote the set of all pixel coordinates in the original image $\boldsymbol{x}_{\text{src}} \in \mathbb{R}^{U \times V}$ and the reprojected image $\boldsymbol{x}_{\text{tar}} \in \mathbb{R}^{U \times V}$ with resolution $U \times V$, respectively. The goal of the reprojected patch extraction is to extract a patch positioned at $\boldsymbol{p}^c_{\text{tar}}$ in the reprojected image, whose content stems from a position $\boldsymbol{p}^c_{\text{src}}$ in the original image. This is achieved by rotating the source image in 3D space such that its source patch position aligns with the desired target patch position.

To derive the required rotation, the source and target patch positions are first projected onto the unit sphere using the inverse projection function $\boldsymbol{\xi}_{\text{o}}^{-1}$ as

$$\boldsymbol{s}^c_{\text{src}} = \boldsymbol{\xi}_{\text{o}}^{-1}(\boldsymbol{p}^c_{\text{src}}), \tag{1}$$

$$\boldsymbol{s}^c_{\text{tar}} = \boldsymbol{\xi}_{\text{o}}^{-1}(\boldsymbol{p}^c_{\text{tar}}). \tag{2}$$

With $(\theta^c_{\text{src}}, \varphi^c_{\text{src}})$ and $(\theta^c_{\text{tar}}, \varphi^c_{\text{tar}})$ denoting the spherical coordinate representation of $\boldsymbol{s}^c_{\text{src}}$ and $\boldsymbol{s}^c_{\text{tar}}$, respectively, the rotation matrix rotating the sampled source patch position to the sampled target patch position is then derived as

$$\boldsymbol{R} = \boldsymbol{R}_z(\varphi^c_{\text{tar}})\boldsymbol{R}_y(\theta^c_{\text{tar}} - \theta^c_{\text{src}})\boldsymbol{R}_z(-\varphi^c_{\text{src}}). \tag{3}$$

Using the rotation matrix that rotates the source patch position $\boldsymbol{p}^c_{\text{src}}$ to the target patch position $\boldsymbol{p}^c_{\text{tar}}$, the positions of the unknown target samples $\boldsymbol{p}_{\text{tar}}$ in the source format are obtained as

$$\boldsymbol{p}_{\text{tar}\rightarrow\text{src}}(\boldsymbol{p}_{\text{tar}}) = \boldsymbol{\xi}_o \left(\boldsymbol{R}^{-1}\boldsymbol{\xi}_o^{-1}(\boldsymbol{p}_{\text{tar}})\right) \;\forall\, \boldsymbol{p}_{\text{tar}} \in \mathcal{P}_{\text{tar}}, \tag{4}$$

where $\mathcal{P}_{\text{tar}}$ describes the set of pixel coordinates in the target patch of resolution $M \times N$ (typically 256×256 pixels).

The unknown target patch sample values $\boldsymbol{x}_{\text{tar}}[\boldsymbol{p}_{\text{tar}}]$ can then be obtained through interpolation from the original 360-degree video frames $\boldsymbol{x}_{\text{src}}$ as

$$\boldsymbol{x}_{\text{tar}}[\boldsymbol{p}_{\text{tar}}] = \boldsymbol{x}_{\text{src}}(\boldsymbol{p}_{\text{tar}\rightarrow\text{src}}(\boldsymbol{p}_{\text{tar}})) \;\forall\, \boldsymbol{p}_{\text{tar}} \in \mathcal{P}_{\text{tar}} \tag{5}$$

using a suitable interpolation technique. We found that a mipmapped bilinear interpolation [49, 61] yields the best training performance (further details in supplementary material).

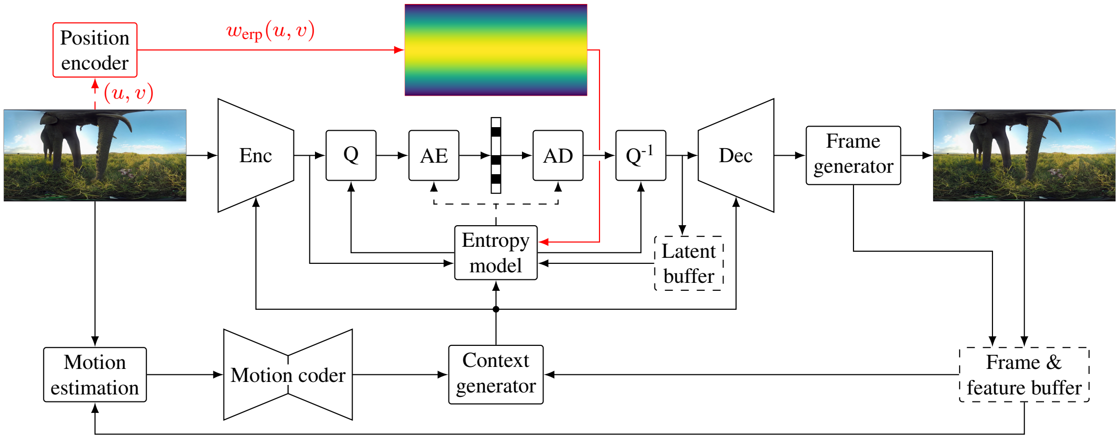
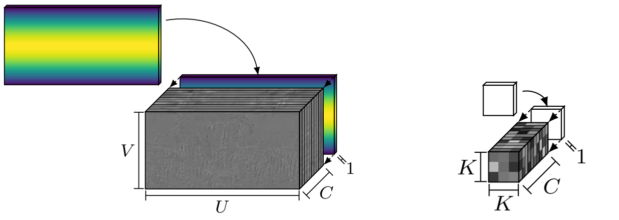


Figure 4. Positional feature encoding at the example of the DCVC-HEM model [29].



(a) Feature encoding.

(b) Kernel extension.

Figure 5. Depiction of (a) the insertion process of the additional positional feature encoding channel and (b) the extension of the respective convolution kernels.

Flow-guide. Despite the added flexibility of the described reprojected patch extraction procedure, a challenge with 360-degree video are scenes with a static camera. In such scenes, large areas might inhibit no or close to no motion. This lack of motion can impair the training of NVCs, as the primary goal lies in effectively coding innovation in dynamic scenes. To ensure that training patches inhibit sufficient motion, we propose to restrict the extraction of patches to video areas that exceed a minimum optical flow magnitude, which we estimate using SPyNet [42]. The flow guide is obtained by averaging the optical flow magnitude over all frames and filtering all pixel positions that surpass a given threshold. The source patch center position $\boldsymbol{p}_{\text{src}}^{c}$ for the reprojection procedure is then uniformly sampled only from pixel positions in the flow guide. Our proposed flow-guided reprojection is summarized in Fig. 3.

3.3. Positional Feature Encoding

The spatial position $\boldsymbol{p}$ of a pixel in the projected 360-degree video determines two aspects that are relevant for the coding procedure: The strength of distortions and the importance of that pixel for final video quality. The strength of distortions might affect the relevant filters for a position, i.e., different features might be relevant for different distortion characteristics. The importance of a pixel affects the desired rate-distortion tradeoff for that position, i.e., the less important a position is, the fewer bits should be used to code that position. Both depend on the 360-degree projection format.

We address this position dependency by providing positional information as additional input to the network. The best performance has been reached by introducing the position information only into the entropy model. Fig. 4 shows the extended NVC network architecture on the basis of the popular DCVC-HEM [29] framework.

The raw pixel positions $\boldsymbol{p}$ are preprocessed by a positional encoder. During training, the patch extraction provides these positions. During inference, the encoder and decoder can independently construct the position grid from the known height and width of the 360-degree video. No additional data has to be transmitted.

A position encoding inspired by the WS-PSNR quality metric for 360-degree video [53] proved to be most effective. It encodes the area a pixel covers on the unit sphere relative to its area on the 2D image plane and contains no learnable parameters. For the equirectangular projection with height V, the positional encoding at each pixel location $\boldsymbol{p} = (u,v)^{\mathrm{T}}$ is derived as [53]

$$w_{\text{erp}}(u,v) = \cos\left(\frac{(v - V/2)\cdot\pi}{V}\right). \tag{6}$$

While we focus on the equirectangular projection in this work, please note that this position encoding can also be derived for other projection formats. For perspective video, the position encoding is set as $w_{\text{p}}(u,v) = 1$.

The position encoding is then concatenated with the input feature space of the entropy model as depicted in Fig. 5(a), extending the channel dimension by an additional positional feature encoding channel. The involved convolution kernels are extended accordingly as shown in Fig. 5(b).

Table 2. BD-Rate (%) in YUV color space with respect to the baseline DCVC-HEM finetuned on vimeo90k. Different models and training configurations. WS-PSNR used as quality metric for the 360-degree JVET360 dataset, PSNR for the remaining perspective datasets. Column *Average* shows the average BD-Rate over all perspective dataset sequences. Highest rate savings for each dataset are marked bold.

Model	Training set	FGR	JVET360	HEVC-B	HEVC-C	HEVC-D	HEVC-E	UVG	Average
DCVC-HEM [29]	vimeo90k		0.00	0.00	0.00	0.00	0.00	0.00	0.00
DCVC-HEM [29]	UGC360		-2.48	11.74	3.32	8.88	-3.93	9.91	7.39
		✓	-6.98	5.02	-0.78	6.52	-10.00	4.05	2.22
DCVC-HEM [29]	UGC360+ vimeo90k		-3.92	2.77	4.50	4.28	-7.06	1.05	1.44
		✓	-7.12	-0.65	1.41	1.71	-10.55	-0.15	-1.13
DCVC-HEM-360	UGC360+ vimeo90k		-5.68	4.52	1.35	2.18	-6.22	1.10	1.15
		✓	**-7.73**	**-1.34**	**-1.47**	**-2.19**	**-12.21**	**-2.86**	**-3.40**
HM-18.0 [52]			22.83	11.06	-2.74	16.02	3.09	10.19	8.27
VTM-22.2 [9]			-10.12	-19.43	-28.84	-13.00	-28.82	-16.46	-20.34

4. Experimental Results

4.1. Experimental Setup

Datasets. As 360-degree training data, we use the proposed UGC360-S and UGC360-M subsets. UCG360-L is used for validation. As perspective training data, we use the vimeo90k dataset [63]. For combined training on UGC360 and vimeo90k (UGC360+vimeo90k), samples from both datasets are drawn in equilibrium (50:50). For testing, we use the JVET360 [19] dataset for 360-degree video at a resolution of 2048×1024 pixels, which is the official dataset used in standardization. The HEVC Class B, C, D, E [7] and UVG [37] datasets are used to evaluate performance on perspective video.

Loss function. For finetuning on 360-degree video, we use an updated loss function that uses the weighted mean squared error (WMSE) d_{WMSE} from [53] instead of MSE to optimize for the 360-degree specific WS-PSNR quality metric. For conventional perspective video, the WMSE is identical to the MSE $d_{\text{WMSE}} = d_{\text{MSE}}$. The loss function results as

$$L(\hat{\boldsymbol{x}}, \boldsymbol{x}) = \frac{1}{N} \sum_{i=1}^{N} (\lambda \cdot d_{\text{WMSE}}(\hat{\boldsymbol{x}}_i, \boldsymbol{x}_i) + r_i), \qquad (7)$$

where i denotes the frame index, N the number of consecutive frames used for training, $\hat{\boldsymbol{x}}_i$ the reconstructed frame, $\boldsymbol{x}_i$ the original frame, and r_i the coded number of bits.

Model. Though our proposed contributions can be applied to any recent NVC, our investigations focus on DCVC-HEM [29]. As noted in [24], reproducing the performance of the most recent DCVC-DC [30] and DCVC-FM [31] models remains an unsolved challenge, since their training procedures are not open-sourced. To ensure a fair comparison, we thus use the DCVC-HEM model, whose training performance could be successfully reproduced.

Training setup. We initialize DCVC-HEM with the pre-trained weights from [38]. The extended model with our proposed positional feature encoding is termed DCVC-HEM-360. For DCVC-HEM-360, the additional convolution kernel channels are randomly initialized. The networks are finetuned using 7 consecutive frames, whereby the first frame is intra coded. The number of frames used for training the video network thus results as $N = 6$. Training is performed in RGB color space similar to the original model [33] with a patch size of 256×256 pixels. In each optimization step, a random rate point is trained corresponding to the λ values $(85, 170, 380, 840)$. We finetune for 500,000 iterations using a batch size of 4.

Test settings. For each video, 96 frames are coded with an intra period of 32 similar to [29, 30, 41]. Rate savings are evaluated using the Bjøntegaard Delta metric [6, 21] with pchip interpolation. For 360-degree videos, the WS-PSNR metric [53] is used to assess the quality of the reconstructed frames. For perspective videos, the conventional PSNR is used. The quality metrics are evaluated in YUV color space. As the model input is RGB, color conversion is performed using BT.709 color conversion coefficients [23]. DCVC-HEM finetuned on vimeo90k is used as baseline as the finetuned weights improve slightly over the publicly available pre-trained weights [38].

4.2. Performance Evaluation

Table 2 shows the rate savings achieved by our framework for neural 360-degree video compression for DCVC-HEM and the extended DCVC-HEM-360 with positional feature encoding. The column FGR denotes whether the proposed flow-guided reprojection is applied for 360-degree data augmentation. For perspective training data or if FGR is disabled, patches are extracted via random cropping. For context, Table 2 also reports the performance of the traditional

Table 3. Ablation study investigating flow-guided reprojection. BD-Rate (%) in YUV color space with respect to default configuration T_A for each model individually. All instances trained on UGC360+vimeo90k.

		T_A	T_B	T_C	T_D	T_E
	Flow-Guide	✓		✓		✓
	Reprojection	✓	✓	✓		
	Mipmap	✓	✓			
DCVC-HEM	JVET360	0.00	1.35	4.34	3.38	2.42
	HEVC+UVG	0.00	1.61	3.39	2.56	1.44
DCVC-HEM-360	JVET360	0.00	1.53	1.73	2.14	2.22
	HEVC+UVG	0.00	1.60	1.64	4.73	6.88

HEVC [52] and VVC [9] video coding standards.

Flow-Guided Reprojection. FGR shows significant rate savings for each combination of model and training set for both 360-degree and perspective video data. If DCVC-HEM is finetuned exclusively on the new UGC360 dataset, the application of FGR improves rate savings from 2.48% to 6.98% for the 360-degree JVET360 dataset. For perspective data, FGR significantly reduces the losses from 7.39% to 2.22% on average. The same holds true for DCVC-HEM (DCVC-HEM-360) trained on the combined UGC360+vimeo90k, where rate savings increase from 3.92% (5.68%) to 7.12% (7.73%) for 360-degree video, and the average bitrate increase of 1.44% (1.15%) for perspective video is turned into average rate savings of 1.13% (3.40%).

Training set. Training on the combined UGC360+vimeo90k dataset improves the compression performance for both 360-degree and perspective video data compared to training on UGC360 alone. Rate savings improve from 6.98% to 7.12% for 360-degree video, and the average losses of 2.22% for perspective video are turned into slight bitrate savings of 1.13%. As discussed in the introduction, this mutual benefit is possible because of the significant domain overlap between perspective and 360-degree video data. Nonetheless, without positional feature encoding, slight losses of 1.41% and 1.71% still occur for the perspective HEVC-C and -D datasets.

Positional Feature Encoding. With the extended DCVC-HEM-360, we eliminate losses for perspective video while achieving even higher rate savings of 7.73% for 360-degree video. For the perspective datasets, DCVC-HEM-360 with FGR achieves robust average rate savings of 3.40% consistently outperforming the baseline DCVC-HEM finetuned on vimeo90k. This shows that the introduced positional feature encoding helps the network in reliably differentiating between 360-degree and perspective video content and allows the network to further profit from the increased diversity of the joint training set. Similar tests for the DCVC [28] and DCVC-TCM [50] models validate that our approach yields gains for other NVC architectures as well (further details in supplementary material).

Table 4. Per-frame complexity metrics for the default DCVC-HEM and DCVC-HEM-360 with positional feature encoding.

Model	Parameters	GMACs	Enc time	Dec time
DCVC-HEM	17.523 M	872.00	122 ms	89 ms
DCVC-HEM-360	17.528 M	872.01	122 ms	90 ms

*Evaluated on an RTX 3090 with input video of size 1024×512.

4.3. Ablation Study

Table 3 shows the results of an ablation study investigating the influence of the different elements of FGR. Results are shown for DCVC-HEM and DCVC-HEM-360. We train both models using five training configurations T_A - T_E on UGC360+vimeo90k. Each training configuration employs a unique set of features from FGR.

Disabling the flow guide (T_B) leads to a bitrate increase of 1.35% (1.53%) for 360-degree video and 1.61% (1.60%) for perspective video compared to DCVC-HEM (DCVC-HEM-360) with all FGR features enabled (T_A), demonstrating the relevance of dynamic content for NVC training. Disabling the mipmap (T_C) yields even higher bitrate increases of 4.34% (1.73%) for 360-degree video and 3.39% (1.64%) for perspective video, highlighting the importance of considering signal theoretic limits (aliasing) during interpolation of the reprojected samples. If not considered, disabling FGR entirely (T_D) would be the better option for DCVC-HEM as this does not require resampling. For DCVC-HEM-360, the effect of disabling the mipmap is less severe than for DCVC-HEM because it does not rely on local features to estimate relative pixel positions. Enabling only the flow-guide (T_E) also yields significant bitrate increases of 2.42% (2.22%) for 360-degree video and 1.44% (6.88%) for perspective video. The reprojection procedure shows to be crucial to reach the highest rate savings. The obtained rate savings cannot be achieved by exclusion of static content from training data alone.

4.4. Complexity

Despite the notable benefits for 360-degree and perspective video compression performance, the positional feature encoding in DCVC-HEM-360 shows only minimal complexity overhead over the default DCVC-HEM model, as visible in Table 4. This is achieved by introducing the positional feature encoding only in the entropy model, which works at a spatial resolution that is reduced by a factor of 16 compared to the input resolution. Introducing the positional feature encoding into additional network components yields additional complexity, but no benefit in compression performance (further details in supplementary material).

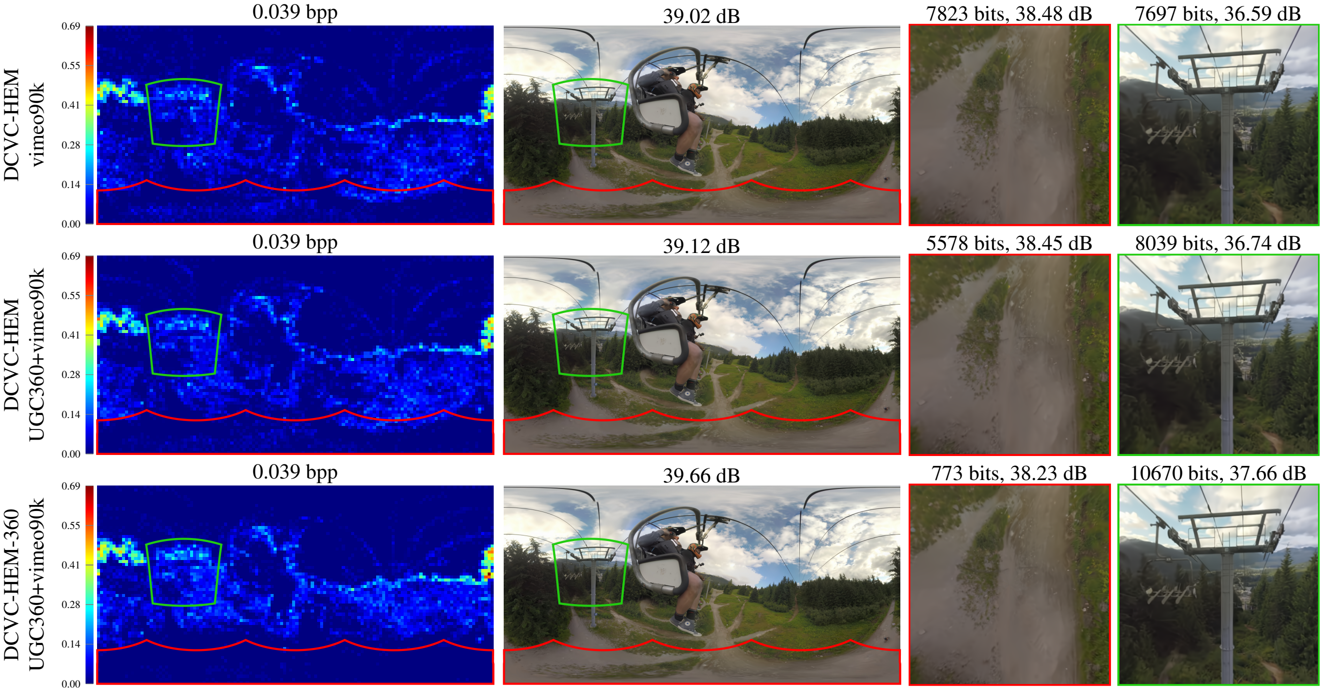


Figure 6. Visual results of our DCVC-HEM and DCVC-HEM-360 for the JVET360 *ChairliftRide* sequence finetuned on vimeo90k or UGC360+vimeo90k with flow-guided reprojection. The first column shows the bitrate allocation with the overall bitrate, the second column shows the reconstructed frame with the overall WS-PSNR, the third and fourth columns show two exemplary viewports with their allocated number of bits and corresponding viewport PSNR.

4.5. Visual Results

Fig. 6 shows a comparison of the bitrate allocation and quality of the default DCVC-HEM finetuned on vimeo90k (top row), the default DCVC-HEM finetuned on UGC360+vimeo90k with flow-guided reprojection (middle row), and our extended DCVC-HEM-360 finetuned on UGC360+vimeo90k with flow-guided reprojection (bottom row) for the JVET360 *ChairliftRide* sequence. The global quantization scale has been set for each model individually in order to achieve the same overall bitrate for all models.

DCVC-HEM finetuned on UGC360+vimeo90k already learned to allocate less bitrate to the polar areas. However, introducing spatial context into DCVC-HEM-360 further increases the focus towards the central equatorial area. The resulting improvement in visual quality also shows in the viewports. While DCVC-HEM finetuned on UGC360+vimeo90k already saves bits for the bottom viewport, DCVC-HEM-360 cuts down bitrate significantly, while trading only little in quality. For the green viewport, DCVC-HEM-360 is thus able to spend more bits and achieve a significantly improved quality over the other models. This showcases the efficacy of positional feature encoding for visual quality through improved bitrate allocation.

5. Conclusion and Outlook

In this paper, we introduced a broadly applicable 360-degree optimization framework for neural video codecs that outperforms naive optimization on 360-degree video significantly. Our framework improves compression performance for both 360-degree video and perspective video by leveraging the significant domain overlap between both content domains. We eliminate the lack of 360-degree video training data by publishing a dataset of more than 6000 full-frame 360-degree video clips with resolutions ranging from 0.5K to 8K. Our proposed flow-guided reprojection shows to be crucial to achieve maximum rate savings. It improves the diversity of content seen during training by exploiting the spherical characteristics of 360-degree video for data augmentation. Extending the entropy model by a positional feature encoding further boosts performance and yields robust rate savings for both 360-degree video and perspective video. Using the recent DCVC-HEM compression model, rate savings of almost 8% are reached for 360-degree video and more than 3% for perspective video with minimal complexity overhead. In the future, we will investigate training on additional projection formats to improve NVC versatility towards different 360-degree video representations.

References

[1] Elena Alshina, João Ascenso, and Touradj Ebrahimi. JPEG AI: The First International Standard for Image Coding Based on an End-to-End Learning-Based Approach. *IEEE MultiMedia*, 31(4):60–69, 2024. 2

[2] João Ascenso, Elena Alshina, and Touradj Ebrahimi. The JPEG AI Standard: Providing Efficient Human and Machine Visual Data Consumption. *IEEE MultiMedia*, 30(1):100–111, 2023. 2

[3] Johannes Ballé, Valero Laparra, and Eero P. Simoncelli. End-to-End Optimized Image Compression. In *Proc. Int. Conf. Learn. Represent.*, 2017. 2

[4] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational Image Compression with a Scale Hyperprior. In *Proc. Int. Conf. Learn. Represent.*, 2018. 2

[5] Jesús Bermejo-Berros and Miguel Angel Gil Martínez. The relationships between the exploration of virtual space, its presence and entertainment in virtual reality, 360º and 2D. *Virtual Reality*, 25(4):1043–1059, 2021. 1

[6] Gisle Bjøntegaard. Calculation of Average PSNR Differences between RD-curves, VCEG-M33. In *Proc. 13th Meet. Video Coding Experts Group*, pages 1–5, 2001. 6

[7] Frank Bossen, Jill Boyce, Karsten Sühring, Xiang Li, and Vadim Seregin. VTM Common Test Conditions and Software Reference Configurations for SDR Video, JVET-T2010. In *Proc. 20th Meet. Jt. Video Experts Team*, pages 1–2, 2020. 1, 6

[8] Fabian Brand, Jürgen Seiler, and André Kaup. Conditional residual coding: A remedy for bottleneck problems in conditional inter frame coding. *IEEE Trans. Circuits Syst. Video Technol.*, 34(7):6445–6459, 2024. 2

[9] Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J. Sullivan, and Jens-Rainer Ohm. Overview of the Versatile Video Coding (VVC) Standard and its Applications. *IEEE Trans. Circuits Syst. Video Technol.*, 31(10): 3736–3764, 2021. 2, 3, 6, 7

[10] Mingdeng Cao, Chong Mou, Fanghua Yu, Xintao Wang, Yinqiang Zheng, Jian Zhang, Chao Dong, Gen Li, Ying Shan, Radu Timofte, Xiaopeng Sun, Weiqi Li, Zhenyu Zhang, Xuhan Sheng, Bin Chen, Haoyu Ma, Ming Cheng, Shijie Zhao, Wanwan Cui, Tianyu Xu, Chunyang Li, Long Bao, Heng Sun, Huaibo Huang, Xiaoqiang Zhou, Yuang Ai, Ran He, Renlong Wu, Yi Yang, Zhilu Zhang, Shuohao Zhang, Junyi Li, Yunjin Chen, Dongwei Ren, Wangmeng Zuo, Qian Wang, Hao-Hsiang Yang, Yi-Chung Chen, Zhi-Kai Huang, Wei-Ting Chen, Yuan-Chun Chiang, Hua-En Chang, I-Hsiang Chen, Chia-Hsuan Hsieh, Sy-Yen Kuo, Zebin Zhang, Jiaqi Zhang, Yuhui Wang, Shuhao Cui, Junshi Huang, Li Zhu, Shuman Tian, Wei Yu, and Bingchun Luo. NTIRE 2023 Challenge on 360deg Omnidirectional Image and Video Super-Resolution: Datasets, Methods and Results. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshop*, pages 1731–1745, 2023. 2

[11] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned Image Compression With Discretized Gaussian Mixture Likelihoods and Attention Modules. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 7936–7945. IEEE, 2020. 2

[12] Xavier Corbillon, Gwendal Simon, Alisa Devlic, and Jacob Chakareski. Viewport-Adaptive Navigable 360-Degree Video Delivery. In *Proc. IEEE Int. Conf. Commun.*, pages 1–7. IEEE, 2017. 1

[13] Creative Commons. About CC Licenses. https://creativecommons.org/share-your-work/cclicenses/, 2024. 3

[14] Francesca De Simone, Pascal Frossard, Neil Birkbeck, and Balu Adsumilli. Deformable Block-Based Motion Estimation in Omnidirectional Image Sequences. In *Proc. IEEE 19th Int. Workshop Multimed. Signal Process.*, pages 1–6, 2017. 2

[15] Masaki Emoto, Kenichiro Masaoka, Masayuki Sugawara, and Yuji Nojiri. The Viewing Angle Dependency in the Presence of Wide Field Image Viewing and its Relationship to the Evaluation Indices. *Displays*, 27(2):80–89, 2006. 1

[16] Google LLC. YouTube Online Video Platform. https://www.youtube.com. 3

[17] K. M. Gorski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, and M. Bartelmann. HEALPix: A Framework for High-Resolution Discretization and Fast Analysis of Data Distributed on the Sphere. *ApJ*, 622(2): 759–771, 2005. 3

[18] Oguzhan Gungordu and A. Murat Tekalp. Saliency-aware End-to-end Learned Variable-Bitrate 360-degree Image Compression. In *Proc. IEEE Int. Conf. Image Process.*, pages 1–7, 2024. 3

[19] Yuwen He, Jill Boyce, Kiho Choi, and Jian-Liang Lin. JVET Common Test Conditions and Evaluation Procedures for 360° Video, JVET-U2012. In *Proc. 21st Meet. Jt. Video Explor. Team*, pages 1–8, 2021. 1, 6

[20] Christian Herglotz, Mohammadreza Jamali, Stephane Coulombe, Carlos Vazquez, and Ahmad Vakili. Efficient Coding of 360° Videos Exploiting Inactive Regions in Projection Formats. In *Proc. IEEE Int. Conf. Image Process.*, pages 1104–1108, 2019. 2

[21] Christian Herglotz, Hannah Och, Anna Meyer, Geetha Ramasubbu, Lena Eichermüller, Matthias Kränzler, Fabian Brand, Kristian Fischer, Dat Thanh Nguyen, Andy Regensky, and André Kaup. The Bjøntegaard Bible: Why Your Way of Comparing Video Codecs May Be Wrong. *IEEE Trans. on Image Process.*, 33:987–1001, 2024. 6

[22] ISO/IEC. ISO/IEC PRF 6048-1: Information Technology — JPEG AI Learning-Based Image Coding System — Part 1: Core Coding System, 2025. 2

[23] ITU-R. Rec. ITU-R BT.709-6: Parameter Values for the HDTV Standards for Production and International Programme Exchange, 2015. 6

[24] Wei Jiang, Junru Li, Kai Zhang, and Li Zhang. ECVC: Exploiting Non-Local Correlations in Multiple Frames for Contextual Video Compression, 2024. 6

[25] Jihyung Kim, Kyeongsun Kim, and Wooksung Kim. Impact of Immersive Virtual Reality Content Using 360-Degree Videos in Undergraduate Education. *IEEE Trans. Learning Technol.*, 15(1):137–149, 2022. 1

[26] A. Burakhan Koyuncu, Han Gao, Atanas Boev, Georgii Gaikov, Elena Alshina, and Eckehard Steinbach. Contextformer: A Transformer with Spatio-Channel Attention for Context Modeling in Learned Image Compression. In *Proc. 17th Eur. Conf. Comput. Vis.*, pages 447–463, 2022. 2

[27] Chen Li, Mai Xu, Xinzhe Du, and Zulin Wang. Bridge the Gap Between VQA and Human Behavior on Omnidirectional Video: A Large-Scale Dataset and a Deep Learning Model. In *Proc. 26th ACM Int. Conf. Multimed.*, pages 932–940, 2018. 2

[28] Jiahao Li, Bin Li, and Yan Lu. Deep Contextual Video Compression. In *Proc. Adv. Neural Inf. Process. Syst.*, pages 18114–18125, 2021. 2, 7

[29] Jiahao Li, Bin Li, and Yan Lu. Hybrid Spatial-Temporal Entropy Modelling for Neural Video Compression. In *Proc. 30th ACM Int. Conf. Multimed.*, pages 1503–1511, 2022. 1, 2, 5, 6

[30] Jiahao Li, Bin Li, and Yan Lu. Neural Video Compression With Diverse Contexts. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 22616–22626, 2023. 6

[31] Jiahao Li, Bin Li, and Yan Lu. Neural Video Compression With Feature Modulation. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 26099–26108, 2024. 2, 6

[32] Li Li, Zhu Li, Xiang Ma, Haitao Yang, and Houqiang Li. Advanced Spherical Motion Model and Local Padding for 360° Video Compression. *IEEE Trans. Image Process.*, 28 (5):2342–2356, 2019. 2

[33] Mu Li, Jinxing Li, Shuhang Gu, Feng Wu, and David Zhang. End-to-End Optimized 360° Image Compression. *IEEE Trans. on Image Process.*, 31:6267–6281, 2022. 2, 6

[34] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: An End-To-End Deep Video Compression Framework. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 10998–11007, 2019. 2

[35] Guo Lu, Xiaoyun Zhang, Wanli Ouyang, Li Chen, Zhiyong Gao, and Dong Xu. An End-to-End Learning Framework for Video Compression. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(10):3292–3308, 2021. 2

[36] Navid Mahmoudian Bidgoli, Roberto G. De A. Azevedo, Thomas Maugey, Aline Roumy, and Pascal Frossard. OSLO: On-the-Sphere Learning for Omnidirectional Images and Its Application to 360-Degree Image Compression. *IEEE Trans. on Image Process.*, 31:5813–5827, 2022. 3

[37] Alexandre Mercat, Marko Viitanen, and Jarno Vanne. UVG Dataset: 50/120fps 4K Sequences for Video Codec Analysis and Development. In *Proc. 11th ACM Multimed. Syst. Conf.*, pages 297–302, 2020. 1, 6

[38] Microsoft Asia. Deep Contextual Video Compression Open-Source Repository. https://github.com/microsoft/DCVC. 6

[39] David Minnen, Johannes Ballé, and George D Toderici. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. *Proc. Adv. Neural Inf. Process. Syst.*, 31, 2018. 2

[40] Aude Oliva and Antonio Torralba. Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *Int. J. Comput. Vis.*, 42(3):145–175, 2001. 3

[41] Linfeng Qi, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Motion Information Propagation for Neural Video Compression. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 6111–6120. IEEE, 2023. 2, 6

[42] Anurag Ranjan and Michael J. Black. Optical Flow Estimation Using a Spatial Pyramid Network. In *IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 4161–4170, 2017. 5

[43] Andy Regensky, Christian Herglotz, and André Kaup. Motion Plane Adaptive Motion Modeling for Spherical Video Coding in H.266/VVC. In *Proc. IEEE Int. Conf. Image Process.*, pages 875–879. IEEE, 2023. 2

[44] Andy Regensky, Christian Herglotz, and André Kaup. Multi-Model Motion Prediction for 360-Degree Video Compression. *IEEE Access*, 11:117004–117017, 2023.

[45] Andy Regensky, Fabian Brand, and André Kaup. Analysis of Neural Video Compression Networks for 360-Degree Video Coding. In *Proc. Pict. Coding Symp.*, pages 1–5. IEEE, 2024. 2

[46] Taehyun Rhee, Stephen Thompson, Daniel Medeiros, Rafael Dos Anjos, and Andrew Chalmers. Augmented Virtual Teleportation for High-Fidelity Telecollaboration. *IEEE Trans. Visual. Comput. Graphics*, 26(5):1923–1933, 2020. 1

[47] Johannes Sauer, Jens Schneider, and Mathias Wien. Improved Motion Compensation for 360° Video Projected to Polytopes. In *Proc. IEEE Int. Conf. Multimed. Expo*, pages 61–66, 2017. 2

[48] Johannes Sauer, Mathias Wien, Jens Schneider, and Max Blaser. Geometry-Corrected Deblocking Filter for 360° Video Coding using Cube Representation. In *Proc. Pict. Coding Symp.*, pages 66–70, 2018. 2

[49] Mark Segal and Kurt Akeley. The OpenGL Graphics System: A Specification - Version 4.6 (Core Profile). pages 1–829, 2022. 4

[50] Xihua Sheng, Jiahao Li, Bin Li, Li Li, Dong Liu, and Yan Lu. Temporal Context Mining for Learned Video Compression. *IEEE Trans. Multimed.*, 25:7311–7322, 2023. 2, 7

[51] Tomáš Souček and Jakub Lokoč. TransNet V2: An Effective Deep Network Architecture for Fast Shot Transition Detection, 2020. 3

[52] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Trans. Circuits Syst. Video Technol.*, 22(12):1649–1668, 2012. 2, 3, 6, 7

[53] Yule Sun, Ang Lu, and Lu Yu. Weighted-to-Spherically-Uniform Quality Evaluation for Omnidirectional Video. *IEEE Signal Process. Lett.*, 24(9):1408–1412, 2017. 5, 6

[54] Ahmed Telili, Ibrahim Farhat, Wassim Hamidouche, and Hadi Amirpour. ODVista: An Omnidirectional Video Dataset for super-resolution and Quality Enhancement Tasks, 2024. 2

[55] Lucas Theis, Wenzhe Shi, Andrew Cunningham, and Ferenc Huszár. Lossy Image Compression with Compressive Autoencoders. In *Proc. Int. Conf. Learn. Represent.*, 2017. 2

[56] Vimeo.com Inc. Vimeo Online Video Platform. https://vimeo.com. 3

[57] Bharath Vishwanath, Tejaswi Nanjundaswamy, and Kenneth Rose. Motion Compensated Prediction for Translational

Camera Motion in Spherical Video Coding. In *Proc. IEEE 20th Int. Workshop Multimed. Signal Process.*, pages 1–4, 2018. 2

[58] Bharath Vishwanath, Tejaswi Nanjundaswamy, and Kenneth Rose. A Geodesic Translation Model for Spherical Video Compression. *IEEE Trans. Image Process.*, 31:2136–2147, 2022. 2

[59] Qian Wang, Weiqi Li, Chong Mou, Xinhua Cheng, and Jian Zhang. 360DVD: Controllable Panorama Video Generation with 360-Degree Video Diffusion Model. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 6913–6923, 2024. 2

[60] T. Wiegand, G.J. Sullivan, G. Bjontegaard, and A. Luthra. Overview of the H.264/AVC Video Coding Standard. *IEEE Trans. Circuits Syst. Video Technol.*, 13(7):560–576, 2003. 3

[61] Lance Williams. Pyramidal Parametrics. *SIGGRAPH Comput. Graph.*, 17(3):1–11, 1983. 4

[62] Mai Xu, Chen Li, Yufan Liu, Xin Deng, and Jiaxin Lu. A Subjective Visual Quality Assessment Method of Panoramic Videos. In *Proc. IEEE Int. Conf. Multimed. Expo*, pages 517–522. IEEE, 2017. 2

[63] Tianfan Xue, Baian Chen, Jiajun Wu, Donglai Wei, and William T. Freeman. Video Enhancement with Task-Oriented Flow. *Int J Comput Vis*, 127(8):1106–1125, 2019. 3, 6

[64] Yan Ye and Jill Boyce. Algorithm Descriptions of Projection Format Conversion and Video Quality Metrics in 360Lib Version 12, JVET-T2004-v2. In *Proc. 20th Meet. Jt. Video Experts Team*, pages 1–65, 2020. 2

[65] Silin Zheng, Xuelin Shen, Qiudan Zhang, Zhuo Chen, Wenhan Yang, and Xu Wang. Towards 360° Image Compression for Machines via Modulating Pixel Significance. *Multimed. Tools Appl.*, 2024. 3

[66] Yimin Zhou, Ling Tian, Ce Zhu, Xin Jin, and Yu Sun. Video Coding Optimization for Virtual Reality 360-Degree Source. *IEEE J. Sel. Top. Signal Process.*, 14(1):118–129, 2020. 2