Res-NERV : Residual blocks for a practical implicit neural video decoder

Marwa Tarchouli[12], Thomas Guionnet[1], Marc Riviere[1], Wassim Hamidouche[2],
Meriem Outtas[2], and Olivier Deforges[2],
[1]Ateme, Rennes, France
[2]Univ. Rennes, INSA Rennes, CNRS, IETR - UMR 6164, Rennes, France

Abstract

This paper proposes the integration of residual blocks into neural representation for videos (NeRV)-based architectures with the aim of enhancing the reconstruction of detailed patterns and high-level features. Additionally, a coding pipeline is introduced, placing the implicit neural decoder in a real-life video streaming framework. Indeed, DeepCABAC is employed for model compression, applying a quantization scheme followed by the context-adaptive binary arithmetic coding (CABAC) entropy coding algorithm, ultimately leading to bitstream generation. Our method outperforms NeRV, as well as x264 and x265, achieving BD-rate gains against NeRV : -12.06% using PSNR and -14.25% using MS-SSIM. Furthermore, it exhibits superior subjective quality compared to NeRV, attributed to enhanced high-level feature reconstruction. This observed behavior encourages the application of our method to other NeRV-based models, such as E-NeRV. This paper has been accepted at the 2024 IEEE International Conference on Image Processing.

Keywords

Implicit Neural Representation, Learned Video Compression, Video Compression.

1 Introduction

Learned video codecs have disrupted the landscape of traditional codecs, including versatile video coding (VVC) [1]. Notable, algorithms such as [2, 3, 4] are built upon large generic variational autoencoders (VAEs). The latest learned video codec, [4], has surpassed the performance of the state-of-the-art conventional codec VVC [1]. However, the success of these algorithms introduces heightened complexity, particularly on the decoder side, posing challenges to practical deployment.

To address this issue, implicit neural representation (INR) were applied in image and video compression. The core concept of INR involves representing the video as a function of temporal and/or spatial coordinates using a lightweight model, individually overfitted to each input video. In the realm of video compression, the encoding process involves training the INR model. The resulting weights are transmitted to the decoder, and the decoding stage involves retrieving these weights and applying the model to reconstruct the approximate version of the source video. In 2021, NeRV [5] emerged as an image-wise implicit model, recovering the entire video frame from its temporal index. This advancement not only improved encoding and decoding speed but also enhanced overall video quality. Serving as the baseline for multiple approaches such as E-NeRV[6], HNeRV [7], CNeRV [8], PS-NeRV [9], FFNeRV [10], and HiNeRV [11], NeRV has become the focal point for endeavors seeking to augment its performance.

NeRV has exhibited noteworthy success and demonstrated significant potential. However, based on our experimental observations, it encounters challenges in efficiently capturing high-level features, especially detailed elements, and textures that appear sporadically in the sequence. This limitation, prevalent among other NeRV-based models, motivates our proposal in this paper to enhance NeRV's architecture and its variants by incorporating residual blocks [12]. Furthermore, the ultimate objective for any video decoder is industrial deployment. Therefore, we proactively consider practical deployment requirements by introducing a compression system tailored for NeRV-based architectures, encompassing quantization, efficient entropy coding techniques, and bitstream generation.

The rest of this paper is organized as follows. In Section 2, we provide a brief introduction to NeRV and its variations. Subsequently, Section 3 outlines the integration of residual blocks in the NeRV codec and describes our proposed coding pipeline. The experimental results of the proposed framework are presented and analyzed in Section 4. Finally, Section 5 concludes the paper.

2 Related Work

As mentioned earlier, NeRV [5] proposed an image-wise architecture comprising a positional encoding step that transforms the temporal index into a high-dimensional embedding space. Subsequently, a multilayer perceptron (MLP) network, followed by convolutional blocks, learns the video features to reconstruct the output video. In comparison to pixel-wise approaches [13, 14, 15], this work significantly accelerates both the encoding and decoding processes by 25x to 70x and 38x to 132x, respectively. Notably, it achieves superior video quality, enhancing the peak signal to noise ratio (PSNR) by 1 dB to 3.35 dB, all at a comparable model size.

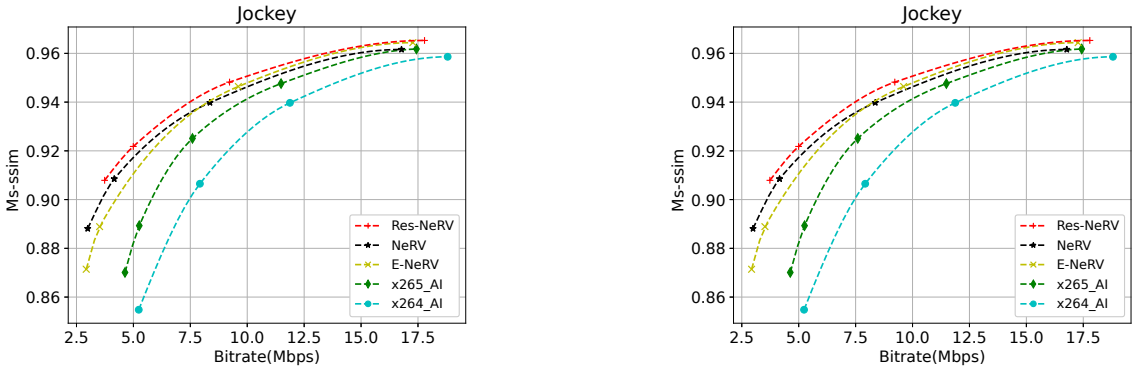

Figure 1: *Rate-distortion results: A performance comparison of Res-NeRV with NeRV, E-NeRV, x264, and x265.*

Several methods have been proposed to enhance NeRV, encompassing strategies such as incorporating spatial coordinates information into the network [6, 9, 11], introducing content information [7, 8, 16], and improving temporal correlation between frames [10, 17].

It is important to highlight that none of the previously mentioned methodologies investigated the incorporation of residual blocks within their decoder architecture. Furthermore, a significant gap exists as most of these approaches overlook the essential aspect of adapting the INR model to real-world video streaming environments.

3 Proposed method

3.1 Res-NeRV Architecture

In a video sequence, two different types of content can be distinguished: persistent content, which spans a large number of frames and includes constant logos, static backgrounds, or dynamic elements that persist throughout the video; and transient content, which appears briefly in the video, such as fast-moving objects. While NeRV-based algorithms excel in achieving high reconstruction quality for persistent content, they encounter challenges with transient content. This difficulty can be attributed to the overfitting nature of INR models. During training, the video sequence serves as the training dataset, where persistent content dominates over transient content, leading to superior model performance on the former.

To overcome this limitation, we propose incorporating residual blocks into the decoder blocks of NeRV-based architectures. Residual blocks, as defined in [12], are distinguished by their unique structure featuring skip connections. These shortcuts enable information to flow between layers, facilitating the learning of features at various levels of granularity. Additionally, the inclusion of residual blocks contributes to a stable training process, enhancing the efficiency of the resulting model [12].

3.2 Video Coding Pipeline

Our coding pipeline, depicted in Fig. 2, outlines the various steps involved in video coding using INR models, while adhering to real-life application requirements. The encoding and decoding processes are segregated into two distinct stages. The encoding stage encompasses the training of the INR model and the compression of its weights ϕ. The original model compression pipeline utilized by NeRV and its variations involves pruning, quantization, and Huffman entropy coding [18]. In this paper, CABAC [19], a more efficient algorithm, is employed as the entropy coding technique for the model's weights. To implement this, the DeepCABAC [20] algorithm, tailored for deep neural network compression, is utilized. This algorithm also introduces a quantization scheme that accounts for the impact of quantization on the model's performance. In summary, our model compression pipeline comprises quantization and CABAC entropy coding, as proposed by DeepCABAC.

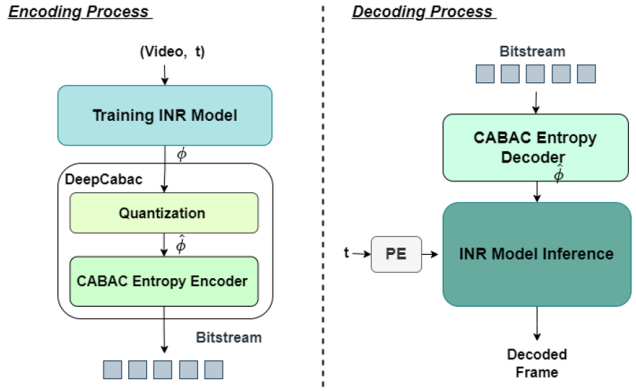

Figure 2: *INR-based coding in a practical video streaming pipeline.*

4 Experiments

The Big Buck Bunny sequence and the UVG-HD dataset were used to evaluate the performance of the proposed method. Additionally, the approach was benchmarked against two image-wise INR methodologies, NeRV and E-NeRV, as well as the traditional video compression algorithms x264 and x265. The evaluations were conducted using an NVIDIA RTX 3090 GPU.

4.1 Coding efficiency

Table 1 presents the Bjontegaard delta (BD)-rate of Res-NeRV compared to NeRV, using PSNR and multi-scale

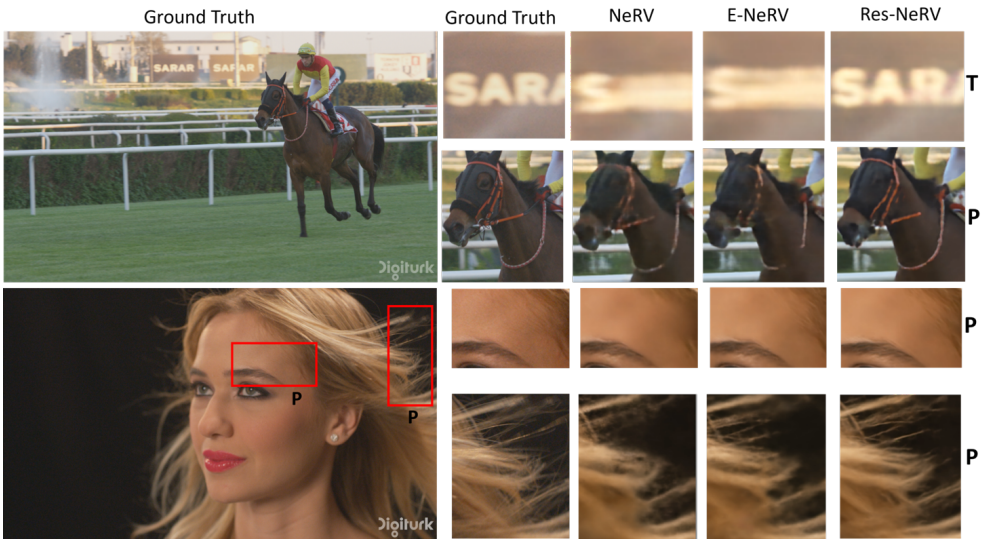

Figure 3: *Visual results for Jockey (∼ 3.6 Mbps), Bunny (∼ 3Mbps) and Beauty (∼ 3.5Mbps). P stands for persistent regions, T stands for transient regions*

SSIM (MS-SSIM). Res-NeRV achieves a significant BD-rate gains, ranging from -9.20% to -20.36% using PSNR and from -8.83% to 20.11% using MS-SSIM. Despite a slight loss in BD-rate using PSNR for the Beauty video sequence, Res-NeRV still attains a gain of -12.06% using the MS-SSIM metric, which is known to be more correlated to the perceived video quality than PSNR.

Tableau 1: *BD-rate results of Res-NeRV compared to NeRV.*

Sequence	BD-rate (PSNR)	BD-rate (MS-SSIM)
Bunny	-20.36%	-17.53%
Beauty	4.18%	-12.06%
Jockey	-10.12%	-10.50%
HoneyBee	-15.88%	-8.83%
ReadySetGo	-17.82%	-19.88%
YachtRide	-9.20%	-10.90%
Bosphorus	-14.79%	-20.11%
ShakenDry	-12.51%	-14.26%
Average	-12.06%	-14.25%

The rate-distortion curves for both PSNR and MS-SSIM metrics are illustrated in Fig. 1. Res-NeRV yields superior results compared to NeRV and traditional codecs x264 and x265, while remaining competitive with E-NeRV. As previously mentioned, the Res-NeRV block architecture proposed in this paper is compatible with all NeRV-based models, including E-NeRV. The obtained results for its integration within NeRV encourage its application to E-NeRV, especially for complex sequences such as *Jockey*.
Visual illustrations are provided in Fig. 3. For the *Jockey* sequence, Res-NeRV outperforms NeRV, enhancing its ability to preserve fine details and textures in both transient areas (denoted T) and persistent areas (denoted P). In this particular sequence, Res-NeRV achieves a better subjective quality than E-NeRV, refining high-level features. These results highlight the potential of integrating Res-NeRV blocks into E-NeRV. The *Beauty* sequence, is mostly dominated by persistent content. While NeRV succeeds in providing a good reconstruction of the video, Res-NeRV demonstrates a better reconstruction of fine details such as blown hair and intricate textures on the woman's face.
To sum up, Res-NeRV surpasses NeRV results using both objective and subjective metrics. Res-NeRV shows an average BD-rate gain of -12.06% using PSNR and -14.25% using MS-SSIM for the *Bunny* sequence and ultra video group (UVG)-HD dataset. Furthermore, Res-NeRV provides a better reconstruction of details for both persistent and transient regions of the video.

4.2 Decoder computational complexity

The decoding complexity of the three INR-based models is evaluated at the same bitrate ($\tilde{3}$.5 Mbps) on the *ReadySetGo* video sequence in Table 2.
Res-NeRV has proven to be computationally expensive, increasing the number of multiply-accumulate operations (MACs) by a factor of 3 compared to NeRV and E-NeRV. This is explained by the architecture of the residual blocks that require more computations. However, the decoding time of our proposed method does not increase proportionally. In fact, it increases by 26% compared to NeRV and E-NeRV at a constant bitrate. Overall, the coding efficiency results of Res-NeRV come with a rise in complexity. We believe this limitation can be an area of improvement in future works.

5 Conclusion

In this paper, we proposed Res-NeRV, which introduces residual blocks to the NeRV-based architecture. Additionally, we presented a INR coding pipeline aligned with real-life video streaming application requirements. Res-NeRV

Tableau 2: *Complexity performance on ReadySetGo.*

Model	Model Size (M)	Bitrate (Mbps)	GMACs	Decoding Time (s)
NeRV	2.85	3.5	226.94	161.19
E-NeRV	2.72	3.5	228.21	165.40
Res-NeRV	3.07	3.6	684.20	202.80

outperforms NeRV in both objective (BD-rate gains of -12.06% using PSNR and -14.25% using MS-SSIM) and subjective quality, preserving finer details across different content types. Compared to E-NeRV, Res-NeRV exhibited less favorable results, but it efficiently enhances high-level features, especially on transient content. Thus, integrating Res-NeRV blocks within E-NeRV holds the potential to further enhance performance. However, the rate-distortion gain of our method comes with an increase in computational complexity and decoding time. This limitation can be addressed in future works.

References

[1] J. Ohm et G. Sullivan. Versatile video coding–towards the next generation of video compression. Dans *Picture Coding Symposium*, 2018.

[2] T Ladune, P. Philippe, W. Hamidouche, L. Zhang, et O. Déforges. Conditional coding for flexible learned video compression. *preprint arXiv:2104.07930*, 2021.

[3] J. Li, B. Li, et Y. Lu. Neural video compression with diverse contexts. Dans *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22616–22626, 2023.

[4] X. Sheng et al. Temporal context mining for learned video compression. *IEEE Transactions on Multimedia*, 2022.

[5] H. Chen et al. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021.

[6] Z. Li et al. E-nerv: Expedite neural video representation with disentangled spatial-temporal context. Dans *European Conference on Computer Vision*, pages 267–284, 2022.

[7] H. Chen, M. Gwilliam, S-N. Lim, et A. Shrivastava. Hnerv: A hybrid neural representation for videos. Dans *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10270–10279, 2023.

[8] H. Chen, M. Gwilliam, B. He, S-N. Lim, et A. Shrivastava. Cnerv: Content-adaptive neural representation for visual data. *preprint arXiv:2211.10421*, 2022.

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

[10] JC. Lee, D. Rho, JH. Ko, et E Park. Ffnerv: Flow-guided frame-wise neural representations for videos. Dans *31st ACM International Conference on Multimedia*, pages 7859–7870, 2023.

[11] H-M. Kwan, G. Gao, F. Zhang, A. Gower, et D. Bull. Hinerv: Video compression with hierarchical encoding based neural representation. *preprint arXiv:2306.09818*, 2023.

[12] K. He, X. Zhang, S. Ren, et J. Sun. Deep residual learning for image recognition. Dans *IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[13] E. Dupont, A. Goliński, M. Alizadeh, Y.W Teh, et A. Doucet. Coin: Compression with implicit neural representations. *preprint arXiv:2103.03123*, 2021.

[14] S. Kim, S. Yu, J. Lee, et J. Shin. Scalable neural video representations with learnable positional features. *Advances in Neural Information Processing Systems*, 35:12718–12731, 2022.

[15] E. Dupont, H. Loya, M. Alizadeh, A. Goliński, Y. W Teh, et A. Doucet. Coin++: Neural compression across modalities. *preprint arXiv:2201.12904*, 2022.

[16] B. He et al. Towards scalable neural representation for diverse videos. Dans *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6132–6142, 2023.

[17] Q. Zhao, MS. Asif, et Z. Ma. Dnerv: Modeling inherent dynamics via difference neural representation for videos. Dans *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2031–2040, 2023.

[18] S. Han, H. Mao, et W.J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *preprint arXiv:1510.00149*, 2015.

[19] D. Marpe, H. Schwarz, et T. Wiegand. Context-based adaptive binary arithmetic coding in the h. 264/avc video compression standard. *IEEE Transactions on circuits and systems for video technology*, 13(7):620–636, 2003.

[20] S. Wiedemann et al. Deepcabac: A universal compression algorithm for deep neural networks. *IEEE Journal of Selected Topics in Signal Processing*, 14(4):700–714, 2020.