Reducing the complexity of normalizing flow architectures for point cloud attribute compression

Rodrigo Borba Pinheiro, Jean-Eudes Marvie, Giuseppe Valenzise, Frédéric Dufaux

REDUCING THE COMPLEXITY OF NORMALIZING FLOW ARCHITECTURES FOR POINT CLOUD ATTRIBUTE COMPRESSION

Rodrigo B. Pinheiro[1,2] *Jean-Eudes Marvie*[1] *Giuseppe Valenzise*[2] *Frédéric Dufaux*[2]

[1] InterDigital, Inc.
[2]Université Paris-Saclay, CNRS, CentraleSupélec, L2S 91190 Gif-sur-Yvette, France

ABSTRACT

Existing learning-based methods to compress PCs *attributes* typically employ variational autoencoders (VAE) to learn compact signal representations. However, these schemes suffer from limited reconstruction quality at high bitrates due to their intrinsic lossy nature. More recently, normalizing flows (NF) have been proposed as an alternative solution. NFs are invertible networks that can achieve lossless reconstruction, at the cost of very large architectures with high memory and computational footprint. This paper proposes an improved NF architecture with reduced complexity called RNF-PCAC. It is composed of two operating modes specialized for low and high bitrates, combined in a rate-distortion optimized fashion. Our approach reduces the number of parameters of the existing NF architectures by over $6\times$. At the same time, it achieves state-of-the-art coding gains compared to previous learning-based methods and, for some PCs, it matches the performance of G-PCC (v.21).

***Index Terms*—** Point clouds, Learning-Based, Compression, Attributes, Normalizing Flow

1. INTRODUCTION

Content consumption has been evolving towards immersive formats [1], specially for entertainment. In this context, Point Clouds (PCs) are one of the most popular volumetric representations. PCs are sets of unordered points that contain the coordinates x, y, z, and the respective attribute information, in our case, color. However, to properly represent an object, PCs need to be very dense, with millions of points, becoming expensive to store and transmit. Compression is therefore necessary to make them a viable option to diffuse 3D content.

The standardization efforts in the MPEG [2] group have contributed significantly to the advances in the field of PC compression, and more recently, learning-based techniques have been gaining popularity for PCs [3]. Very recently, normalizing flow (NF) architectures have been considered for PC attribute compression [4]. NF architectures have the potential of yielding good coding performance at high bitrates, but the sparsity nature of PCs makes them challenging to implement, due to the necessity of using shuffling layers, which leads to an increased number of coefficients. This impacts the complexity of the architecture (memory consumption) and its storage space, making it hard to use in real world scenarios.

This paper introduces an enhanced NF scheme for attribute compression, referred to as Reduced Normalizing Flow PC Attribute Compression (**RNF-PCAC**), to achieve better computational and memory efficiency. The approach is composed of two operating modes, targeting low and high bitrate ranges, respectively: Back Projection mode (**BP**) and No-Average mode (**NA**). We dynamically choose the best coding mode using a rate-distortion optimization approach. Our focus is to control the number of parameters in the network while having a good rate-distortion performance. We achieve better results than the original NF, with a six times reduction in the number of parameters, and also achieve comparable results to G-PCC version v.21 on some PCs.

2. RELATED WORK

The baseline for today's research on PC static compression is the Geometry-based PC Compression (G-PCC) algorithm standardized by MPEG [2]. G-PCC encodes the geometry using an octree approach, whereas attribute coding is performed by a Region Adaptive Hierarchical Transform (RAHT) [5].

Following the recent success of learning-based coding (e.g., [6]), Variational Autoencoders (VAEs) have been also applied to the compression of PCs, in particular for geometry. In [7], the authors use 3D voxel convolutions, and cast the decoding problem as one of classifying which voxels are non-empty. This initial architecture led to some extensions and improvements, [8, 9]. In particular, [10] introduces the use of sparse convolutions for PC compression. Sparse convolutions are interesting for PCs tasks because they can help overcome the limitations of available memory as well as avoid the dilation of the features to empty spaces [11].

Learning-based coding of PC attributes has been less explored. In [12], attributes are mapped to a signal over a 2D manifold and coded using a conventional image codec. However, this method fails when the geometry is complex, limiting the coding performance. Deep-PCAC [13] employs second-order point convolutions [14], which are lighter than sparse convolutions, but have the inability to capture spatial dependencies, which leads to relatively poor coding performance.

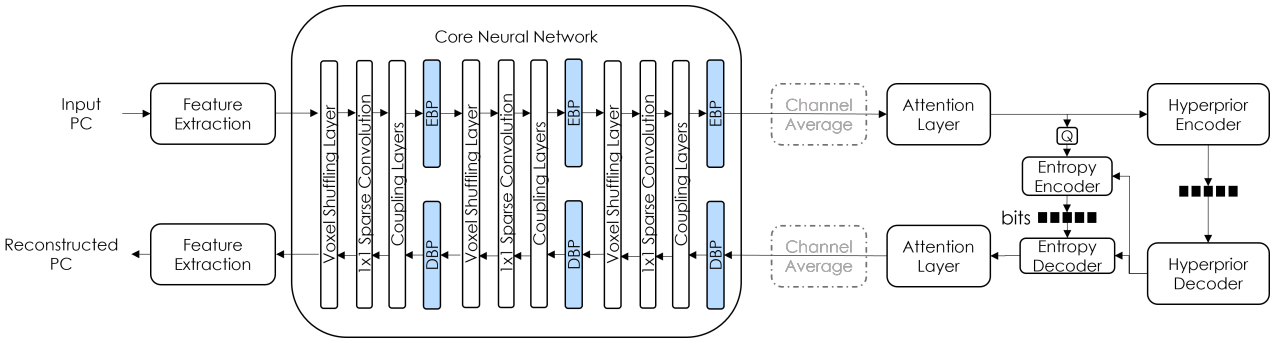


Fig. 1: **BP mode**: The Encoder Back Projection (EBP) layer on the encoding path and the Decoder Back Projection (DBP) layer on the decoding one allow us to control the number of channels through the network. The intermediate channels after the BPs modules are respectively 12, 24, 96. By fixing the number of channels we reduce the number of coefficients of the network. Added blocks in blue and removed blocks in grey and dashed, when compared to the original NF-PCAC [4]

Another method proposed in [15] consists of an extension of [10] in a VAE architecture to compress the attributes of PCs. VAEs are effective at low bitrates, as it reduces the dimensionality of the data. However, this reduction limits the maximum reconstruction quality at higher bitrates.

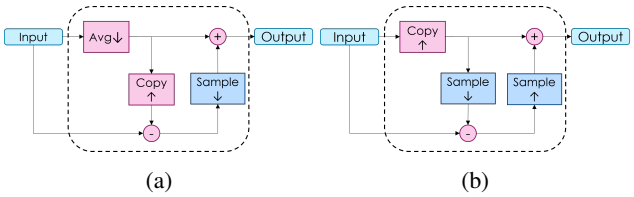
Recently, Normalizing Flows [16] have been applied to image compression [17] with promising results. NFs employ affine coupling layers to produce invertible transformations of the PC signal, similar to orthogonal transforms used in conventional coding. This can lead to coding gains compared to VAEs, especially at high bitrates. In our previous work [4], we have pioneered using NFs for PCAC, obtaining state-of-the-art performance over the current VAE approach. However, this approach employs shuffling layers to transform spatial information into channels and expand the receptive field of sparse convolutions. This leads to a higher number of coefficients, increasing its complexity and storage space. The goal of this paper is to address this drawback.

3. PROPOSED METHOD

To have a bigger receptive field for the convolutions without any loss in information, the NF-based coding network relies on the voxel shuffling layer – a layer that, in the 3D domain, increases the number of channels (N) at a rate of $N \times 8$ [4], causing the number of parameters in the network to explode. The addition of non-invertible blocks, such as channel averaging, to reduce the number of coefficients impacts the performance at high-bitrate, while the removal of those blocks will affect the low-bitrate range. To cope with these contrasting objectives, we adopt here a *divide-and-conquer* strategy. We devise an architecture (**RNF-PCAC**) composed of two operating modes, both extending NF-PCAC [4] to address a different bitrate range. The first mode (Section 3.1) introduces non-invertible blocks in the core of the network to control the number of coefficients and targets the low bitrate range. The second mode (Section 3.2) relies on a reduction of the architecture in [4] combined with an increased hyperprior model and targets the higher bitrate range. Our final proposed method **RNF-PCAC** consists in selecting dynamically one of the two modes using rate-distortion optimization, similar to conventional video coding.

3.1. Back Projection Mode (BP)

The first mode (see Figure 1) was designed to reduce the number of coefficients by controlling the number of channels in the inner layers of the NF core. By systematically reducing the number of channels before the voxel shuffling layers, we reduce the total number of parameters needed to ingest data in subsequent layers. To achieve that, one could directly apply a channel averaging module. However, this approach causes data loss, mainly losing high frequency information, since the channel averaging is equivalent to a low pass filter. To tackle that, we add back-projection blocks both on the encoder and decoder side of our model. Back projection is used in super-resolution context [18] and in image compression [19] with the goal of improving the sampling of features. The goal is to achieve the reduction in the number of channels, while maintaining most of the information contained in those channels.



(a) (b)

Fig. 2: **Back Projection Layers** in BP mode. (a) Encoder Back Projection (b) Decoder Back Projection, Avg stands for channel averaging, $Copy$ for channel copying and $Sample$ for convolutional sampling. $\uparrow$ represent an increase in the number of channels and $\downarrow$ a reduction. Light blue blocks are the sparse data, pink blocks are operators that require no parameters and darker blue blocks are convolutional layers.

On the encoder side, we have the Encoder Back Projection block (EBP), illustrated in Figure 2a. It aims to downsample channel data while retaining more information than

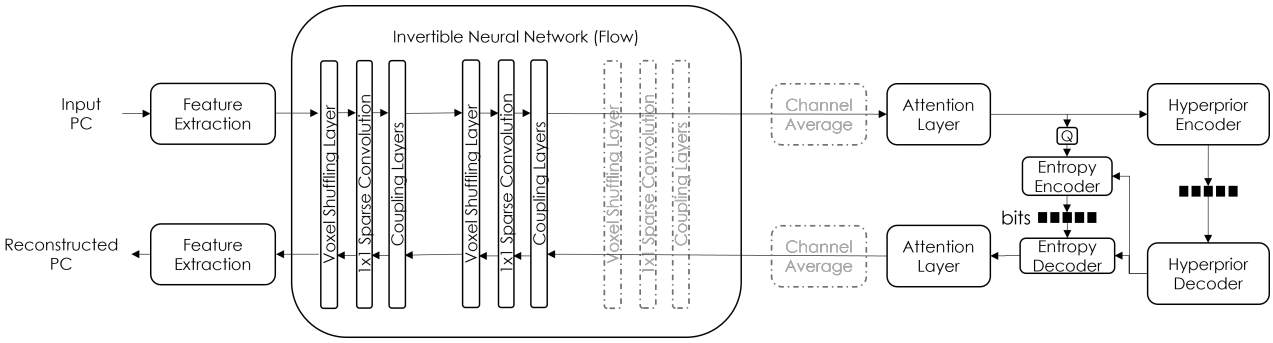


Fig. 3: NA mode: The only sources of non-invertibility are the attention layer and the feature extraction, which do not affect the dimensionality of the data. By removing the last block in the core and the channel average block, we obtain a smaller version of the initial network with a bigger latent space. Removed blocks in grey and dashed lines when compared to [4].

a naive channel averaging. We compute the naive channel average in $Avg \downarrow$, then copy the channels back to the original size in $Copy \uparrow$. We obtain the residual information between the copied data and the original and we use it as input to $Sample \downarrow$, a block composed of 3 sparse convolutional layers that downsamples the residuals channels while learning important information that should be re-injected on the averaged tensor.

On the decoder side, we have the presence of the Decoder Back Projection block (DBP), reported in Figure 2b. It aims to better reconstruct the original data than a naive copying of the channels. For that, we copy the content back to the original tensor size on $Copy \uparrow$ and feed it to a neural network, $Sample \downarrow$, that has the goal of downsampling the data. We then produce the residuals between the averaged tensor and the downsampled version of the copied tensor. We feed this to another neural network $Sample \uparrow$ to recover some details that were lost in the averaging. These details are then added to the copied tensor to recover the original tensor, minimizing the information loss that happened at the encoder side.

BP mode offers a good control over the size of model and the BP blocks, can reduce the number of parameters and achieve a good reconstruction that better maintains the data details. The number of parameters is reduced by over ten times when compared to the NF-PCAC [4] and we maintain the performance in lower bitrates. Results are available in Section 4. Since the BP mode contains more sources of non-invertibility, the high bitrate performance is degraded when compared to the baseline. Therefore, we propose a second mode, NA, to handle the high-bitrate cases.

3.2. No-Average Mode (NA)

With the NA mode (see Figure 3), we aim to reduce the number of parameters and also the sources of non-invertibility by reducing the initial architecture. The main contributor to the increasing number of parameters is the last block of voxel shuffling and coupling layers. We remove those blocks and the channel averaging block that acts as a low-pass filter on the channel data. With these two modifications, we reduce the number of channels at the output of the core network and we remove one of the biggest sources of non-invertibility, boosting the performance on high bitrates.

The architecture on this mode has fewer layers than the baseline NF-PCAC, while having more channels in the hyperprior modules to accommodate the channels added in the shuffling layers. We reduce the number of coefficients by over ten times and obtain a highly invertible network, obtaining good results in the high bitrate domain. The experimental results and comparisons are available in Section 4.

4. EXPERIMENTS AND RESULTS

4.1. Training

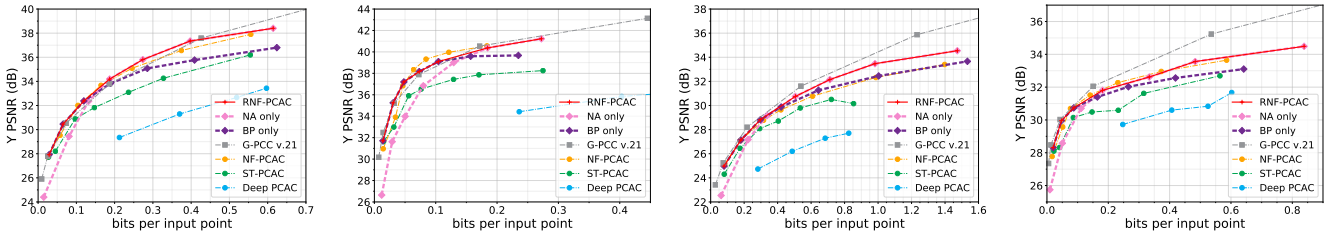
The training set is a mix of different PCs that represent real characters from 8i [20], owlii [21], volucap and xdprod [22], as well as PCs generated from axyz textured mesh bundles [23], sampled to produce PCs of roughly 1,000,000 points, a comparable density to the other PCs in our dataset.

To compensate for the lack of available data for training and testing, we retain for tests the PCs Soldier [20], Basketball Player [21] and Facade [2] while all the other sequences are used for training and we repeat the same protocol using Longdress and Redandblack [20] as test set for a broader validation. Furthermore, an octree partition of PCs is performed, generating blocks of size $128 \times 128 \times 128$. We train on the YUV color space with no data augmentation for 20 epochs with a learning rate of $1e^{-4}$ and a batch size of 8. We train a different model for each rate-point by varying the hyperparameter λ in the rate-distortion loss function.

4.2. Validation

For inference, we do not perform any partition. This helps us avoid blocking artifacts that would occur when coding a PC that has been divided into blocks. Since our architectures are completely convolutional, they can be fed with any input data size as long as it fits into memory.

For comparison purposes, we have a method with a VAE-type architecture similar to the one presented in [15] reimple-



(a) Soldier - 8iVFB (b) Basketball Player - Owlii (c) Longdress - 8iVFB (d) Facade - MPEG

Fig. 4: PSNR curves for four different PCs. The proposed method RNF-PCAC has competitive results against NF-PCAC, and outperforms other learning-based methods. G-PCC v.21 results displayed for reference.

Methods	Params	Storage	Peak GPU Memory				Time			
			10-bits		11-bits		10-bits		11-bits	
			Enc	Dec	Enc	Dec	Enc	Dec	Enc	Dec
NF-PCAC	278	1.10	4.99	4.39	10.02	10.04	0.99	0.95	3.37	3.27
NA only	**26**	**0.10**	3.31	2.74	8.48	9.03	0.65	0.56	2.20	2.02
BP only	**18**	**0.07**	2.79	2.19	7.19	7.36	0.49	0.43	1.59	1.56
RNF-PCAC	**44**	**0.17**	3.31	2.74	8.48	9.03	1.14	0.56	3.78	2.02
ST PCAC	12	0.05	1.85	1.08	3.82	3.47	0.11	0.07	0.33	0.22

Table 1: Complexity Comparison: Millions of parameters, storage in GB per rate-point, peak memory used in GB, and average time in seconds on GPU (Nvidia®. Tesla ®. V100, 32 GB). Encoding times between input and entropy encoding and decoding time between after entropy decoding and reconstructed PC. ST-PCAC results for informative purposes.

mented by us, Sparse Tensor PCAC (ST-PCAC), in addition to NF-PCAC [4]. All the models have the same mean-scale hyperprior [24] that can be enhanced in the future by a more robust model, and they were trained with the same dataset and hyperparameters. We present rate-distortion results for 4 PCs in Figure 4 with the Y-PSNR as quality metric. Although G-PCC can still be considered the best performing approach, our proposed method, **RNF-PCAC** achieves the best results between the learning-based approaches, while offering an important reduction in the number of parameters when compared with the approach on the original NF-PCAC [4].

As seen in Table 1, the number of parameters of RNF-PCAC is over six times smaller than NF-PCAC [4], corresponding to the sum of parameters on both modes. In our approach, we code the PCs with both modes and choose the best performing one signalling it to the decoder with an extra bit. By encoding sequentially with both, the encoding time is the sum of the times for each mode. Peak memory consumption corresponds to the biggest between both modes and the decoding time is also the biggest between both. One could opt to perform parallel encoding, increasing memory consumption and reducing encoding time. In any case, decoding time and memory are not impacted.

4.2.1. Contribution of each mode

The BP mode outperforms all other methods on lower bitrates. When compared to NF-PCAC [4], it achieves similar and sometimes better performance using over $10\times$ fewer parameters. Besides, with the addition of the BP layers, the number of channels throughout the network can be easily controlled. At high bitrates, due to the addition of non-invertible blocks, BP has degraded performance.

On the other end, the NA mode produces higher coding gains than other learning-based methods at higher bitrates. Due to the reduced number of non-invertible blocks, it allows the decoder to receive the information with little loss. It also has over $10\times$ fewer parameters than the original NF architecture. However, when it comes to lower bitrates, the other architectures are better performing. This happens because on the NA we have no channel averaging layer, so the number of coefficients to code is bigger than the other learning based approaches, as the voxel shuffling layers add more coefficients.

4.2.2. Overall coding gains and comparison with G-PCC

We present the BD rate results in Table 2, using G-PCC as reference. Our proposed method RNF-PCAC achieves the best BD rate among the learning-based approaches, and for some PCs it even has similar or slightly competitive performance compared to G-PCC.

Point Cloud	Basketball	Soldier	Longdress	Facade
Deep-PCAC	942.75	295.44	340.23	526.74
ST- PCAC	73.18	43.53	53.79	168.41
NF-PCAC	13.47	1.26	30.58	55.66
RNF-PCAC (Ours)	**-0.11**	**-4.80**	**17.99**	**44.00**

Table 2: Average BD-Rate (%) for luma channel: Learning-based methods against G-PCC v.21. Negative values mean bitrate savings.

5. CONCLUSIONS

We propose RNF-PCAC, a method comprising of two coding modes: back-projection mode and no-average mode. The two modes contribute to reducing the memory and computational footprint of NF-based coding of PC attributes, while increasing coding performance both at low and high bitrates. Differently from geometry coding, learning-based PCAC is still not competitive with the latest versions of G-PCC. Our results show that NF-based approaches are a promising avenue to bridge this performance gap, offering a viable alternative to the prevalent VAE-based approaches.

6. REFERENCES

[1] G. Valenzise, M. Alain, E. Zerman, and C. Ozcinar, *Immersive Video Technologies*, Elsevier, Oct. 2022.

[2] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai, "An overview of ongoing point cloud compression standardization activities: video-based (V-PCC) and geometry-based (G-PCC)," *APSIPA Trans. Signal and Inf. Process.*, 2020.

[3] M. Quach, J. Pang, T. Dong, G. Valenzise, and F. Dufaux, "Survey on Deep Learning-based Point Cloud Compression," *Frontiers in Signal Processing*, 2022.

[4] R. B. Pinheiro, J.-E. Marvie, G. Valenzise, and F. Dufaux, "Nf-pcac: Normalizing flow based point cloud attribute compression," in *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2023, pp. 1–5.

[5] R. L. de Queiroz and P. A. Chou, "Compression of 3d point clouds using a region-adaptive hierarchical transform," *IEEE Transactions on Image Processing*, 2016.

[6] D. Minnen, J. Ballé, and G. D. Toderici, "Joint autoregressive and hierarchical priors for learned image compression," in *Advances in Neural Information Processing Systems*, 2018.

[7] M. Quach, G. Valenzise, and F. Dufaux, "Learning convolutional transforms for lossy point cloud geometry compression," *IEEE International Conference on Image Processing*, 2019.

[8] M. Quach, G. Valenzise, and F. Dufaux, "Improved deep point cloud geometry compression," in *2020 IEEE 22nd International Workshop on Multimedia Signal Processing*, 2020.

[9] J. Wang, D. Ding, Z. Li, and Z. Ma, "Multiscale point cloud geometry compression," in *2021 Data Compression Conference (DCC)*, 2021.

[10] J. Wang, D. Ding, Z. Li, X. Feng, C. Cao, and Z. Ma, "Sparse tensor-based multiscale representation for point cloud geometry compression," 2021.

[11] B. Graham, M. Engelcke, and L. van der Maaten, "3d semantic segmentation with submanifold sparse convolutional networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[12] M. Quach, G. Valenzise, and F. Dufaux, "Folding-based compression of point cloud attributes," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020.

[13] X. Sheng, L. Li, D. Liu, Z. Xiong, Z. Li, and F. Wu, "Deep-PCAC: An end-to-end deep lossy compression framework for point cloud attributes," *IEEE Transactions on Multimedia*, 2021.

[14] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017.

[15] J. Wang and Z. Ma, "Sparse tensor-based point cloud attribute compression," in *IEEE International Conference on Multimedia Information Processing and Retrieval*, 2022.

[16] L. Dinh, J. Sohl-Dickstein, and S. Bengio, "Density estimation using real NVP," in *International Conference on Learning Representations*, 2017.

[17] Y. Xie, K. L. Cheng, and Q. Chen, "Enhanced invertible encoding for learned image compression," *29th ACM International Conference on Multimedia*, 2021.

[18] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Los Alamitos, CA, USA, jun 2018, pp. 1664–1673, IEEE Computer Society.

[19] G. Gao, P. You, R. Pan, S. Han, Y. Zhang, Y. Dai, and H. Lee, "Neural image compression via attentional multi-scale back projection and frequency decomposition," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 14657–14666.

[20] E. d'Eon, B. Harrison, T. Myers, and P. A. Chou, "8i voxelized full bodies - a voxelized point cloud dataset," ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006, 2017.

[21] Y. Xu, Y. Lu, and Z. Wen, "Owlii dynamic human mesh sequence dataset," ISO/IEC JTC1/SC29/WG11 m41658, 120th MPEG Meeting, Macau, 2017.

[22] R. Schaefer, P. Andrivon, J. Ricard, and C. Guede, "Volucap and XD Productions Datasets," Tech. Rep. M56192, ISO/IEC JTC1/SC29/WG7(MPEG), 2021.

[23] "Axyz design 3d people models and character animation software," https://secure.axyz-design.com - Acessed Mar. 03, 2023.

[24] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," 2018.