HAL
open science

Folding-based Compression of Point Cloud Attributes

Maurice Quach, Giuseppe Valenzise, Frédéric Dufaux

To cite this version:

Maurice Quach, Giuseppe Valenzise, Frédéric Dufaux. Folding-based Compression of Point Cloud Attributes. IEEE International Conference on Image Processing (ICIP'2020), Oct 2020, Abu Dhabi, United Arab Emirates. pp.3309-3313, 10.1109/ICIP40778.2020.9191180 . hal-02609423

HAL Id: hal-02609423

https://hal.science/hal-02609423v1

Submitted on 16 Jun 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

FOLDING-BASED COMPRESSION OF POINT CLOUD ATTRIBUTES

Maurice Quach *Giuseppe Valenzise* *Frederic Dufaux*

Université Paris-Saclay, CNRS, CentraleSupélec, Laboratoire des signaux et systèmes
91190 Gif-sur-Yvette, France

ABSTRACT

Existing techniques to compress point cloud attributes leverage either geometric or video-based compression tools. We explore a radically different approach inspired by recent advances in point cloud representation learning. Point clouds can be interpreted as 2D manifolds in 3D space. Specifically, we fold a 2D grid onto a point cloud and we map attributes from the point cloud onto the folded 2D grid using a novel optimized mapping method. This mapping results in an image, which opens a way to apply existing image processing techniques on point cloud attributes. However, as this mapping process is lossy in nature, we propose several strategies to refine it so that attributes can be mapped to the 2D grid with minimal distortion. Moreover, this approach can be flexibly applied to point cloud patches in order to better adapt to local geometric complexity. In this work, we consider point cloud attribute compression; thus, we compress this image with a conventional 2D image codec. Our preliminary results show that the proposed folding-based coding scheme can already reach performance similar to the latest MPEG Geometry-based PCC (G-PCC) codec.

Index Terms— point cloud, compression, neural network

1. INTRODUCTION

A point cloud is a set of points in 3D space which can have associated attributes such as color or normals. Point clouds are essential for numerous applications ranging from archeology and architecture to virtual and mixed reality. Since they can contain millions of points with complex attributes, efficient point cloud compression (PCC) is essential to make these applications feasible in practice.

When compressing a point cloud, we usually consider two aspects: the geometry, that is the 3D coordinates of each individual point, and the attributes, for example RGB colors. Moreover, we can differentiate dynamic point clouds, which change in the temporal dimension, from static point clouds. The Moving Picture Experts Group (MPEG) is leading PCC standardization efforts [1]. Specifically, two main solutions have emerged. The first one, Geometry-based PCC (G-PCC), uses native 3D data structures, while the second one, Video-based PCC (V-PCC), targets mainly dynamic point clouds, and projects the data on a 2D plane to make use of available video codecs such as HEVC.

Point clouds can be interpreted as 2D discrete manifolds in 3D space. Therefore, instead of compressing point cloud attributes using 3D structures such as octrees, we can fold this 2D manifold onto an image. This opens many avenues of research, as it provides, e.g., a way to apply existing image processing techniques straightforwardly on point cloud attributes. In this work, we propose a novel system for folding a point cloud and mapping its attributes to a 2D grid. Furthermore, we demonstrate that the proposed approach can be used to compress static point cloud attributes efficiently.

2. RELATED WORK

Our work is at the crossroads of static point cloud attribute compression and deep representation learning of 3D data. Compressing static point cloud attributes has been explored using graph transforms [2], the Region-Adaptive Hierarchical Transform (RAHT) [3] and volumetric functions [4]. Graph transforms take advantage of the Graph Fourier Transform (GFT) and the neighborhood structure present in the 3D space to compress point cloud attributes. The RAHT is a hierarchical transform which extends the Haar wavelet transform to an octree representation. In this paper, we propose a different perspective, and leverage the manifold interpretation of the point cloud by mapping its attributes onto a 2D grid, which can then be compressed as an image.

Deep learning methods have been used for representation learning and compression of point clouds [5]. In particular, the initial folding in our work is inspired by [6] where an autoencoder network is trained on a dataset to learn how to fold a 2D grid onto a 3D point cloud. In our work, we build on this folding idea; however, we employ it in a very different way. Specifically, we do not aim at learning a good representation that can generalize over a dataset; instead, we employ the folding network as a parametric function that maps an input 2D grid to points in 3D space. The parameters of this function (i.e., the weights of the network) are obtained by overfitting the network to a specific point cloud. In addition, the original folding proposed in [6] is highly inefficient for

Funded by the ANR ReVeRy national fund (REVERY ANR-17-CE23-0020).

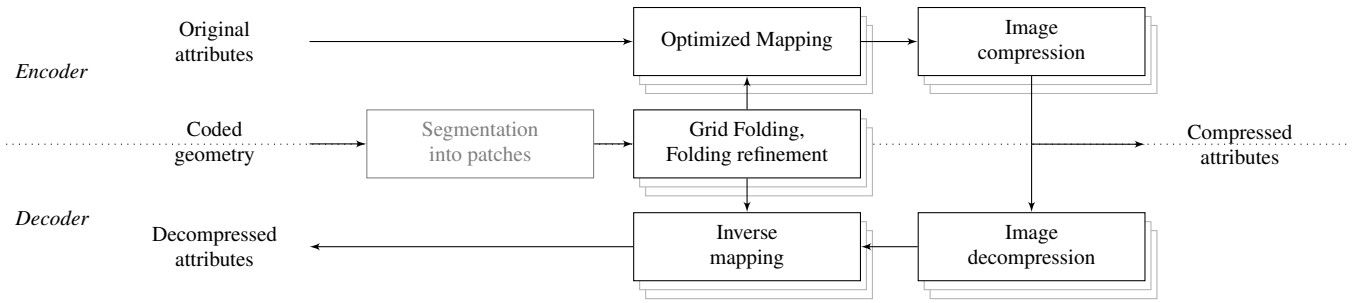

Fig. 1: Proposed system for attribute compression. Segmentation is optional and can help to adapt to local geometry complexity.

PCC as it poorly adapts to complex geometries. In our work, we propose a number of solutions to improve folding.

3. PROPOSED METHOD

We propose a novel system for compressing point cloud attributes based on the idea that a point cloud can be seen as a discrete 2D manifold in 3D space. In this way, we can obtain a 2D parameterization of the point cloud and we can map attributes from a point cloud onto a grid, making it possible to employ 2D image processing algorithms and compression tools. The overall system is depicted in Figure 1. In a nutshell, our approach is based on the following two steps: a) we find a parametric function (specifically, a deep neural network) to fold a 2D grid onto a 3D point cloud; b) we map attributes (e.g., colors) of the original point cloud to this grid. The grid and the parametric function contain all the necessary information to recover the point cloud attributes. Assuming the point cloud geometry is coded separately and transmitted to the decoder, the folding function can be constructed at the decoder side, and the 2D grid is fully decodable without any need to transmit network parameters. In practice, the 3D-to-2D mapping is lossy, which entails a mapping distortion in the step b) above. In the following, we propose several strategies to reduce this mapping distortion.

Notation. We use lowercase bold letters such as $\mathbf{x}$ to indicate 3D vectors (point cloud spatial coordinates), and uppercase letters such as X to indicate sets of 3D points (vectors). We denote with a tilde (like $\tilde{\mathbf{x}}$ or $\tilde{X}$) compressed (distorted) vectors or sets of vectors. We use the notation $\langle S \rangle = \sum_{x \in S} x / |S|$ for the average over a set S.

3.1. Grid folding

We propose a grid folding composed of two steps, namely, an initial folding step to get a rough reconstruction of X and a folding refinement step to improve the reconstruction quality, which is quintessential to map point cloud attributes with minimal mapping distortion.

We fold a grid onto a point cloud to obtain its 2D parameterization by solving the following optimization problem:

$$\min_f \mathcal{L}(X, \tilde{X}) \tag{1}$$

where X is the set of n points in the original point cloud, $\tilde{X} = f(X, G)$ is the set of n' points in the reconstructed point cloud obtained by folding G onto X where G the set of $n' = w \times h$ points of a 2D grid with 3D coordinates. In general, $n' \neq n$; however, we choose n' to be close to n. $\mathcal{L}$ is a loss function and f is a folding function.

We parameterize f using a neural network composed of an encoder f_e and a decoder f_d such that $\mathbf{y} = f_e(X)$ and $\tilde{X} = f_d(G, \mathbf{y})$. The encoder f_e is composed of four pointwise convolutions with filter sizes of 128 followed by a max-pooling layer. The decoder f_d is composed of two folding layers with $f_d(G, \mathbf{y}) = \mathrm{FL}(\mathrm{FL}(G, \mathbf{y}), \mathbf{y})$. Each folding layer has two pointwise convolutions with filter sizes of 64 and concatenates $\mathbf{y}$ to its input. The last pointwise convolution has a filter size of 3. We use the ReLU activation [8] for the encoder and LeakyReLU activation [9] for the decoder. A one-to-one mapping exists between each point $\tilde{\mathbf{x}}_i$ in the folded grid $\tilde{X}$ and their original position $\mathbf{g}_i$ in the grid G.

We propose the following loss function

$$\mathcal{L}(X, \tilde{X}) = \mathrm{d_{ch}}(X, \tilde{X}) + \mathrm{d_{rep}}(\tilde{X}) \tag{2}$$

where $\mathrm{d_{ch}}$ is the Chamfer distance:

$$\mathrm{d_{ch}}(X, \tilde{X}) = \sum_{\mathbf{x} \in X} \min_{\tilde{\mathbf{x}} \in \tilde{X}} \|\mathbf{x} - \tilde{\mathbf{x}}\|_2^2 + \sum_{\tilde{\mathbf{x}} \in \tilde{X}} \min_{\mathbf{x} \in \tilde{X}} \|\tilde{\mathbf{x}} - \mathbf{x}\|_2^2,$$

and $\mathrm{d_{rep}}$ is a novel *repulsion loss* computed as the variance of the distance of each point in $\tilde{X}$ to its nearest neighbor:

$$\mathrm{d_{rep}}(\tilde{X}) = \mathrm{Var}(\{ \min_{\tilde{\mathbf{x}}' \in \tilde{X} \setminus \tilde{\mathbf{x}}} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}'\|_2^2 \mid \tilde{\mathbf{x}} \in \tilde{X} \}).$$

The Chamfer distance ensures that the reconstruction $\tilde{X}$ is similar to X and the repulsion loss penalizes variations in the reconstruction's density.

We obtain the parameterized folding function f by training a neural network using the Adam optimizer [10]. We use the point cloud X as the single input which is equivalent to overfitting the network on a single sample.

(a) Original (b) Folded (27.63 dB) (c) Refined folded (30.62 dB) (d) Opt. refined folded (33.39 dB)

Fig. 2: Different steps of our proposed attribute mapping method for the first frame of phil9 [7]. Top row: phases of point cloud reconstruction; bottom row: the attributes mapped on a 2D grid, which is later compressed and transmitted. The initial folding (b) provides a rough reconstruction $\tilde{X}$ which is improved with folding refinement (c) and occupancy optimization (d) to reduce the density mismatch between X and $\tilde{X}$. We then map attributes from the point cloud onto a 2D grid. The holes in the grid are filled to facilitate compression with HEVC. We indicate Y PSNRs between original and colors distorted by mapping.

3.2. Folding refinement

The initial folding has difficulties reconstructing complex shapes accurately as seen in Figure 2b. Specifically, the two main issues are mismatches in local density between X and $\tilde{X}$ and inaccurate reconstructions for complex shapes. As a result, this introduces significant mapping distortion when mapping attributes from the original PC to the folded one; additionally, this mapping distortion affects the reconstructed point cloud attributes. For compression applications, this is a serious issue as there are now two sources of distortion from both mapping and compression. This is why we propose a folding refinement method that alleviates mismatches in local density and inaccurate reconstructions.

First, we reduce local density variations by considering density-aware grid structure preservation forces inside $\tilde{X}$. Specifically, each point $\tilde{\mathbf{x}}$ is attracted towards the inverse density weighted average of its neighbors $\mathbf{p}_{\text{grid}}$. Since a one-to-one mapping exists between $\tilde{X}$ and G, each point $\tilde{\mathbf{x}}_i$ in the folded grid $\tilde{X}$ has a corresponding point $\mathbf{g}_i$ in the grid G. We then define the inverse density weight ω_i for $\tilde{\mathbf{x}}_i$ as $\omega_i = \langle\{\, \|\tilde{\mathbf{x}}_i - \tilde{\mathbf{x}}_j\|_2 \mid \mathbf{g}_j \in \mathcal{N}_G(\mathbf{g}_i)\,\}\rangle$ with $\mathcal{N}_G(\mathbf{g}_i)$ the set of horizontal and vertical neighbors of $\mathbf{g}_i$ in the grid G. This encourages the reconstruction to have a more uniform distribution by penalizing high density areas. Given the set Ω comprising all weights ω_i, we define the normalized weights $\hat{\omega}_i = (\omega_i - \min(\Omega))/(\max(\Omega) - \min(\Omega))$. Finally, this allows us to define the weighted average $\mathbf{p}_{\text{grid}_i} = \langle\{\, \hat{\omega}_j \tilde{\mathbf{x}}_j \mid \mathbf{g}_j \in \mathcal{N}_G(\mathbf{g}_i)\,\}\rangle$.

Second, we set up bidirectional attraction forces between X and $\tilde{X}$ to solve two issues: incomplete coverage, when $\tilde{X}$ does not cover parts of X, and inaccurate reconstructions, when $\tilde{X}$ fails to reproduce X accurately. As a solution, we attract each point $\tilde{\mathbf{x}}$ towards two points $\mathbf{p}_{\text{push}}$ and $\mathbf{p}_{\text{pull}}$. Specifically, $\mathbf{p}_{\text{push}}$ is the nearest neighbor of $\tilde{\mathbf{x}}$ in X and *pushes* $\tilde{X}$ towards X which allows for more accurate reconstructions. On the other hand, $\mathbf{p}_{\text{pull}}$ is the average of the points in X which have $\tilde{\mathbf{x}}$ as their nearest neighbor and allows X to *pull* $\tilde{X}$ closer which alleviates incomplete coverage issues.

Finally, we combine these components into an iterative refinement system to update the point cloud reconstruction:

$$\tilde{\mathbf{x}}_{t+1,i} = \alpha \mathbf{p}_{\text{grid}_{t,i}} + (1-\alpha)(\mathbf{p}_{\text{push}_{t,i}} + \mathbf{p}_{\text{pull}_{t,i}})/2 \quad (3)$$

where $\tilde{\mathbf{x}}_{t,i}$ is the value of $\tilde{\mathbf{x}}_i$ after t iterations and $\tilde{\mathbf{x}}_0 = \tilde{\mathbf{x}}$. The inertia factor $\alpha \in [0,1]$ balances the grid structure preservation forces in $\tilde{X}$ with the bidirectional attraction forces set up between X and $\tilde{X}$. Preserving the grid structure preserves the spatial correlation of the attributes mapped on the grid and the density-aware aspect of these forces results in more uniformly distributed points. In addition, the bidirectional forces improve the accuracy of the reconstruction significantly.

3.3. Optimized Attribute Mapping

Once a sufficiently accurate 3D point cloud geometry is reconstructed (Figure 2c), we can map attributes from X to $\tilde{X}$. To this end, we first build a mapping $m_{X\to\tilde{X}}$ from each point in X to a corresponding point in $\tilde{X}$ (for example, the nearest neighbor). Hence, the inverse mapping $m_{\tilde{X}\to X}$ maps $\tilde{\mathbf{x}}$ back

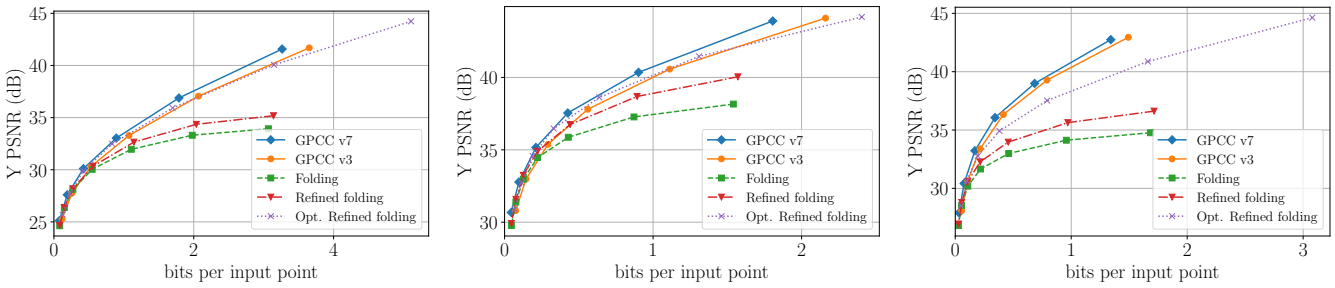

Fig. 3: RD curves showing the performance of the different steps of our method. From top to bottom: longdress_vox10_1300, redandblack_vox10_1550 and soldier_vox10_0690 [11].

to X. As $m_{X\to\tilde{X}}$ is not one-to-one (due to local density mismatches and inaccuracy of the reconstruction), several points in X can map to the same $\tilde{\mathbf{x}}$. Thus, a given $\tilde{\mathbf{x}}$ can correspond to zero, one or many points in X; we define the number of these points as its occupancy $o(\tilde{\mathbf{x}})$. Attribute mapping from X to $\tilde{X}$ is obtained using $m_{\tilde{X}\to X}$ as the attribute value for a point $\tilde{\mathbf{x}}$ is the average of the attribute values of $m_{\tilde{X}\to X}(\tilde{\mathbf{x}})$. In case $m_{\tilde{X}\to X}(\tilde{\mathbf{x}}) = \varnothing$, we simply assign to $\tilde{\mathbf{x}}$ the attribute of its nearest neighbor in X. As a consequence of this approach, points with higher occupancy tend to have higher mapping distortion, as more attributes are averaged.

To overcome this problem, we integrate the occupancy as a regularizing factor when building the mapping. For each point $\mathbf{x}$ in X, we consider its k nearest neighbors set $\mathcal{N}_k(\mathbf{x}) \in \tilde{X}$ and select $m_{X\to\tilde{X}}(\mathbf{x}) = \arg\min_{\tilde{\mathbf{x}}\in\mathcal{N}_k(\mathbf{x})} o(\tilde{\mathbf{x}})\|\tilde{\mathbf{x}} - \mathbf{x}\|_2$. Specifically, the mapping is built iteratively and the occupancies are updated progressively.

As noted above, when $o(\tilde{\mathbf{x}}) > 1$, the attributes are averaged which introduces distortion. We mitigate this problem by adding rows and columns in the 2D grid (see Fig. 2d) using the following procedure. Since $o(\tilde{\mathbf{x}})$ is defined on $\tilde{X}$ and there is a one-to-one mapping between $\tilde{X}$ and G, we can compute mean occupancies row-wise and column-wise. In particular, we compute mean occupancies with zeros excluded and we select the row/column with the maximum mean occupancy. Then, we reduce its occupancy by inserting additional rows/columns around it. We repeat this procedure until we obtain a lossless mapping or the relative change on the average of mean occupancies Δ_r is superior to a threshold $\Delta_{r,min}$.

4. EXPERIMENTAL RESULTS

We evaluate our system for static point cloud attribute compression and compare it against G-PCC v3 [12] and v7 [13]. We also study the impact of folding refinement and occupancy optimization on our method by presenting an ablation study. Since folding is less accurate on complex point clouds, we manually segment the point clouds into patches and apply our scheme on each patch. The patches are then reassembled in order to compute rate-distortion measures.

We use TensorFlow 1.15.0 [14]. For the folding refinement, we set α to $1/3$ and perform 100 iterations. When mapping attributes, we consider $k = 9$ neighbors for assignment. When optimizing occupancy, we set $\Delta_{r,min}$ to 10^{-6}. We then perform image compression using BPG [15], an image format based on HEVC intra [16], with QPs ranging from 20 to 50 with a step of 5.

In Figure 3, we observe that our method performs comparably to G-PCC for “longdress" and “redandblack". The performance is slightly worse for “soldier" as its geometry is much more complex making a good reconstruction difficult and introducing mapping distortion. We obtain significant gains in terms of rate-distortion by improving the reconstruction quality using folding refinement and occupancy optimization. This shows the potential of our method and confirms the importance of reducing the mapping distortion.

5. CONCLUSION

Based on the interpretation of a point cloud as a 2D manifold living in a 3D space, we propose to fold a 2D grid onto it and map point cloud attributes into this grid. As the mapping introduces distortion, this calls for strategies to minimize this distortion. In order to minimize mapping distortion, we proposed a folding refinement procedure, an adaptive attribute mapping method and an occupancy optimization scheme. With the resulting image, we compress point cloud attributes leveraging conventional image codecs and obtain encouraging results. Our proposed method enables the use of 2D image processing techniques and tools on point cloud attributes.

6. REFERENCES

[1] Sebastian Schwarz, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip A. Chou, Robert A. Cohen, Maja Krivokuca, Sebastien Lasserre, Zhu Li, Joan Llach, Khaled Mammou, Rufael Mekuria, Ohji Nakagami, Ernestasia Siahaan, Ali Tabatabai,

Alexis M. Tourapis, and Vladyslav Zakharchenko, "Emerging MPEG standards for point cloud compression," pp. 1–1.

[2] Cha Zhang, Dinei Florêncio, and Charles Loop, "Point cloud attribute compression with graph transform," in *2014 IEEE International Conference on Image Processing (ICIP)*, pp. 2066–2070, ISSN: 2381-8549.

[3] Ricardo L. de Queiroz and Philip A. Chou, "Compression of 3d point clouds using a region-adaptive hierarchical transform," vol. 25, no. 8, pp. 3947–3956, Conference Name: IEEE Transactions on Image Processing.

[4] Maja Krivokuća, Maxim Koroteev, and Philip A. Chou, "A volumetric approach to point cloud compression," .

[5] Maurice Quach, Giuseppe Valenzise, and Frederic Dufaux, "Learning convolutional transforms for lossy point cloud geometry compression," in *2019 IEEE International Conference on Image Processing (ICIP)*, pp. 4320–4324, ISSN: 1522-4880.

[6] Yaoqing Yang, Chen Feng, Yiru Shen, and Dong Tian, "FoldingNet: Point cloud auto-encoder via deep grid deformation," in *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[7] Charles Loop, Qin Cai, Sergio O. Escolano, and Philip A. Chou, "Microsoft voxelized upper bodies - a voxelized point cloud dataset," in *ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document m38673/M72012*.

[8] Vinod Nair and Geoffrey E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, Johannes Fürnkranz and Thorsten Joachims, Eds. pp. 807–814, Omnipress.

[9] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*.

[10] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *2015 3rd International Conference on Learning Representations*.

[11] Sebastian Schwarz, Gaëlle Martin-Cocher, David Flynn, and Madhukar Budagavi, "Common test conditions for point cloud compression," in *ISO/IEC JTC1/SC29/WG11 MPEG output document N17766*.

[12] Khaled Mammou, Philip A. Chou, David Flynn, and Maja Krivokuća, "PCC test model category 13 v3," in *ISO/IEC JTC1/SC29/WG11 MPEG output document N17762*.

[13] "G-PCC test model v7 user manual," in *ISO/IEC JTC1/SC29/WG11 MPEG output document N18664*.

[14] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," .

[15] Fabrice Bellard, "BPG image format," .

[16] "High efficiency video coding (HEVC) version 2 (ITU-t recommendation h.265)," .