# Exploiting View Synthesis for Super-multiview Video Compression

Pavel Nikitin, Marco Cagnazzo, Joël Jung, Attilio Fiandrotti

# Exploiting View Synthesis for Super-multiview Video Compression

Pavel Nikitin, Marco Cagnazzo, Attilio Fiandrotti, Joël Jung

# Exploiting View Synthesis for Super-multiview Video Compression

Pavel Nikitin
pavel.nikitin@orange.com
Orange Labs
Guyancourt, France
LTCI, Telecom Paris, Institut Polytechnique de Paris
Paris, France

Marco Cagnazzo
marco.cagnazzo@telecom-paristech.fr
LTCI, Telecom Paris, Institut Polytechnique de Paris
Paris, France

Joel Jung
joelb.jung@orange.com
Orange Labs
Guyancourt, France

Attilio Fiandrotti
attilio.fiandrotti@telecom-paristech.fr
LTCI, Telecom Paris, Institut Polytechnique de Paris
Paris, France

## ABSTRACT

Super-multiview video consists in a 2D arrangement of cameras acquiring the same scene and it is a well-suited format for immersive and free navigation video services. However, the large number of acquired viewpoints calls for extremely effective compression tools. View synthesis allows to reconstruct a viewpoint using nearby cameras texture and depth information. In this work we explore the potential of recent advances in view synthesis algorithms to enhance the compression performances of super-multiview video. Towards this end we consider five methods that replace one viewpoint with a synthesized view, possibly enhanced with some side information. Our experiments suggest that, if the geometry information (*i.e.* depth map) is reliable, these methods have the potential to improve rate-distortion performance with respect to traditional approaches, at least for some specific content and configuration. Moreover, our results shed some light about how to further improve compression performance by integrating new view-synthesis prediction tools within a 3D video encoder.

## KEYWORDS

Free viewpoint navigation, view synthesis, 2D-multiview

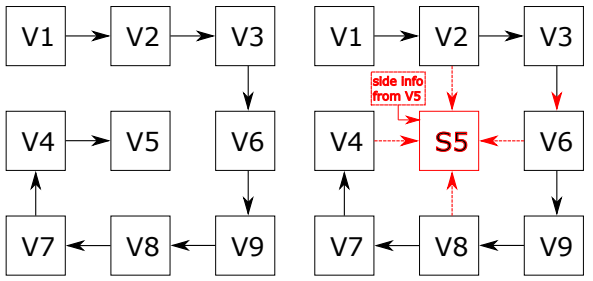
**ACM Reference Format:**
Pavel Nikitin, Marco Cagnazzo, Joel Jung, and Attilio Fiandrotti. 2019. Exploiting View Synthesis for Super-multiview Video Compression. In *Proceedings of ACM International Conference on Distributed Smart Cameras (ICDSC'19).* ACM, ICDSC, Trento, Italy, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

## 1 INTRODUCTION

Free viewpoint video allows a user to navigate in a scene captured by multiple cameras, enabling an experience comparable to the ability of a gamer to navigate in a computer-generated virtual environment. Free viewpoint video navigation promises to enable a number of innovative applications including interactive e-learning, sport events, concert events, museum tours and so forth. A typical implementation of free video navigation consists in selecting an arbitrary viewpoint in a bi-dimensional array of cameras. In this setup, the term "super-multiview" is sometimes employed to emphasize the dense bi-dimensional arrangement of cameras (*i.e.* horizontal and vertical parallax). For each view, the camera acquires both a natural light image (referred to as "texture") and the distance of the objects in each pixel position from the camera itself (referred to as "depth map"), a format often referred to as "multiview-video plus depth" (MVD) [3]. If the depth is not available, it can be estimated from the available textures. MVD allows to synthesize viewpoints at arbitrary positions (referred to as "virtual views") at the user side, and these are typically used to ensure smooth transitions when the user switches from one viewpoint to another. The huge amount of data (a dense bi-dimensional array of textures and depth) that must be conveyed to the user to enable free viewpoint navigation demands efficient encoding schemes.

State-of-the-art solutions for MVD encoding such as 3D-HEVC [8] usually achieve high compression efficiency exploiting the specific redundancy of this format. In addition to temporal and spatial redundancy, the encoder typically exploits the correlation among views (inter-view redundancy) and between texture and depth of the same view (inter-component redundancy). In particular, view-synthesis prediction (VSP) predicts a block of pixels of the current view using a view-synthesis algorithm applied to the available (*i.e.* already decoded) textures and depth. Recent advances in algorithms for view synthesis [1, 4, 7] have drastically improved the quality of synthesized views. Such advances may put into question the paradigm of delivering to the user a dense array of natural views and depth maps versus synthesizing some of the captured views at the decoder side.

In this preliminary work, we aim at exploring the potential of view synthesis techniques from a video compression perspective.




(a) Anchor (b) With skipped central view

**Figure 1: Configuration of the anchor and the proposed system with central view skip and synthesis.**

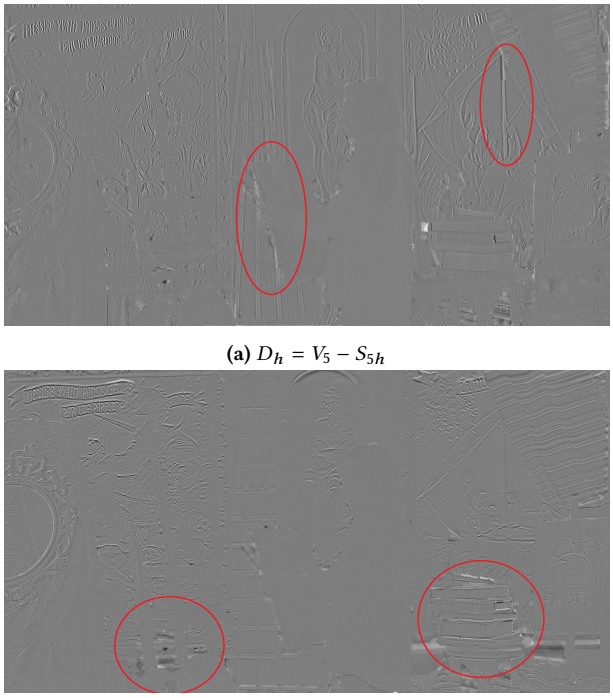
This approach was considered by Dricot at al. [2], who proposed to skip encoding certain views at the source and synthesizing such views at the decoder. This has the advantage of not requiring modifications of existing codecs, thus being easily extended to future 3D codecs. In this paper we propose different variants of this idea, which are easily deployable using existing coding tools with minor modifications only. Our experiments show that in some conditions and for some content it could be advantageous to synthesize a view at the decoder rather than compressing it at the encoder, at least in an All-Intra configuration. Moreover, the results presented here shed some light on a more sophisticated exploitation of the synthesis tool *i.e.* a new VSP version.

The rest of the paper is organized as follows. Sect. 2 describes the proposed method and relative variants. Then, the experimental setup and the results are described and commented in Sect. 3. Sect. 4 concludes the paper and highlights possible future developments.

## 2 PROPOSED METHOD AND VARIANTS

We consider the problem of encoding MVD content consisting into nine views arranged with horizontal and vertical parallax in a 3×3 coding structure. Please note that in the following, the term "view" refers to the texture and the depth acquired by a single camera. The coding structure is shown in Fig. 1. This kind of configuration is quite common in super-multiview set-ups, and there exist some sequences that have been captured or generated using 3D models and that are considered for experiments in the MPEG-I Visual subgroup, the goal of which is to provide a standard for the future video immersive technologies.

Our anchor consists in using MV-HEVC with the direct-P inter-view prediction shown in Fig. 1a. This coding structure is somewhat suboptimal [6], but we chose it because it allows to assess in a simple way the effect of "skipping" the encoding of the central view, since the latter is not used as reference for others. On the contrary, the proposed methods (Fig. 1b) all consist in skipping the encoding of the central view, which will be reconstructed at the decoder side. To this end, we consider five different methods, all based on the view synthesis provided by the most recent view synthesis algorithm called Versatile View Synthesis (VVS) [1]. One of the proposed methods also requires the transmission of some side information in order to produce the reconstructed view.



(a) $D_h = V_5 - S_{5h}$

(b) $D_v = V_5 - S_{5v}$

**Figure 2: Difference images obtained by subtraction the original view $V_5$ and the synthesized views from horizontal and vertical neighbors respectively.**

The first three methods consist in synthesizing view 5 by using respectively only the horizontal neighboring views, only the vertical neighboring views, and the four neighbors as inputs of VVS. The symbol $V_k$ refers to the texture data of view $k$. Moreover, we will refer to the respective synthesized textures from these three methods as follows: $S_{5v}$, which is synthesized from two vertical neighbors $V_2$ and $V_8$; $S_{5h}$, which is synthesized from horizontal neighbors $V_4$ and $V_6$; and $S_{5FR}$, which is synthesized from the four references.

For the first two methods, in Fig. 2 we show the errors between the synthesized and original textures: $D_h = V_5 - S_{5h}$ and $D_v = V_5 - S_{5v}$. We can observe that the errors on the difference image, computed between the original and the predicted from vertical neighbors are distributed along horizontal edges, and in the picture with prediction from horizontal references along vertical edges. As a consequence, we may expect to improve the synthesized image by suitably blending images $S_{5h}$ and $S_{5v}$. The third method actually takes advantage from having four references available, but in a "blind" way, only oriented to synthesis and not to compression. With the last two methods, we want to explore the idea of exploiting the four references explicitly for improving compression.

Let us start by considering two $N \times N$ blocks[1] from $S_{5h}$ and $S_{5h}$, and let us refer to them as $B^{\mathbf{p}}_{5h}$ and $B^{\mathbf{p}}_{5v}$, where the superscript $\mathbf{p} = (n, m)$ refers to the position of the top-left pixel in the block. A new block can be obtained by blending the two blocks via a linear

[1] Not to be confused with HEVC CUs. $N$ here can have any value

(convex) combination:

$$B_{5c}^{\mathbf{p}} = \alpha B_{5h}^{\mathbf{p}} + (1-\alpha) B_{5v}^{\mathbf{p}} \tag{1}$$

On one hand, this would allow for a better synthesized image quality, since we can give more weight to the best prediction between horizontal and vertical. On the other hand, it requires some additional rate since a coefficient should be encoded per block. The bit precision in representing $\alpha$ and the block size $N$ presides the rate-distortion trade-off of this enhanced synthesis method.

In the following we consider two alternative methods to produce $B_{5c}^{\mathbf{p}}$: in the first one, the coefficient $\alpha$ is inferred from $S_{5v}$ and $S_{5h}$ using the gradient operator; in the second the optimal coefficient $\alpha$ is computed by convex projection and then quantized on a suitable number of bits.

## 2.1 Gradient-based method

As shown in Fig. 2, the horizontal and vertical synthesis $S_{5h}$ and $S_{5v}$ have large errors respectively near vertical and horizontal contours. Then, the basic idea is very simple: we estimate the direction of the contours in each point of $V_5$ and then we set the value of $\alpha$ in Eq.(1) accordingly.

The first issue is the estimation of the gradient of $V_5$. In order to do this, we compute the gradients of $S_{5h}$ and $S_{5v}$ using the Sobel operators $S_x$ and $S_y$, and then estimate the gradient of $V_5$ as the average of the gradients of $S_{5h}$ and $S_{5v}$:

$$S_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} \qquad S_y = S_x^T \tag{2}$$

$$G_x = S_x * \frac{1}{2}(S_{5h} + S_{5v}) \qquad G_y = S_y * \frac{1}{2}(S_{5h} + S_{5v}) \tag{3}$$

In the previous equations, the symbol $*$ represent the bi-dimensional convolution. We can consider then the gradient components in any point $\mathbf{p}$, and compute the gradient direction $\theta(\mathbf{p})$ as follows

$$\theta(\mathbf{p}) = \tan^{-1} \frac{G_x(\mathbf{p})}{G_y(\mathbf{p})}.$$

Finally, the combination coefficient $\alpha$ can be computed as a function of $\theta$. In our experiments we found that an effective choice consists in the following:

$$\alpha = \begin{cases} 1, & \text{if } \theta \in \left[0, \frac{\pi}{8}\right[ \cup \left[\frac{7\pi}{8}, \frac{9\pi}{8}\right[ \cup \left[\frac{15\pi}{8}, 2\pi\right[ \\ 0, & \text{if } \theta \in \left[\frac{3\pi}{8}, \frac{5\pi}{8}\right[ \cup \left[\frac{11\pi}{8}, \frac{13\pi}{8}\right[ \\ \frac{1}{2} & \text{otherwise.} \end{cases} \tag{4}$$

This choice of $\alpha$ is easily interpreted: we use the horizontal interpolation $S_{5h}$ near vertical contours, and the vertical interpolation $S_{5v}$ near horizontal contours, while in other positions we use the average of the two.

The advantage of this method is that no side information is needed to be sent at the decoder, since the coefficient $\alpha$ is computed only by using available information. Moreover, we can use any integer value for the block size $N$, and , we can even compute a different coefficient per pixel (*i.e.* $N = 1$). The main disadvantage is that it is an "open-loop" technique: we do not control the quality of the synthesized image.

## 2.2 Convex combination method

In order to control the quality of the synthesized image, the most intuitive approach is to choose $\alpha$ in such a way that the MSE between the synthesized block and the original (i.e., the one coming from the uncompressed $V_5$) is minimized. Such a value of $\alpha$ is easily obtained as a convex combination [5, 9]. Let $\mathbf{h}$, $\mathbf{v}$ be respectively a vectorized representation of $B_{5h}^{\mathbf{p}}$ and $B_{5v}^{\mathbf{p}}$, and $\mathbf{x}$ a vectorized version of the original block of view $V_5$. Minimizing the MSE is equivalent to minimize $d(\alpha) = \|\mathbf{x} - \alpha\mathbf{h} - (1-\alpha)\mathbf{v}\|^2$. So we have:

$$\begin{aligned} d(\alpha) &= (\mathbf{x} - \alpha\mathbf{h} - (1-\alpha)\mathbf{v})^T(\mathbf{x} - \alpha\mathbf{h} - (1-\alpha)\mathbf{v}) \\ &= \|\mathbf{x} - \mathbf{v}\|^2 + \alpha^2\|\mathbf{h} - \mathbf{v}\|^2 + 2\alpha(\mathbf{x} - \mathbf{v})^T(\mathbf{v} - \mathbf{h}) \\ \frac{\partial d}{\partial \alpha} &= 2\alpha\|\mathbf{h} - \mathbf{v}\|^2 + 2(\mathbf{x} - \mathbf{v})^T(\mathbf{v} - \mathbf{h}) \\ \alpha^* &= \frac{(\mathbf{x} - \mathbf{v})^T(\mathbf{v} - \mathbf{h})}{\|\mathbf{h} - \mathbf{v}\|^2} \end{aligned}$$

Using the convex combination of horizontal and vertical synthesis with the optimal coefficient $\alpha^*$ might greatly improve the quality of the skipped view, in particular since this is done by minimizing the distortion. However, the associated side information can be very large since we need to encode one (real) coefficient per $N \times N$ block. Thus, we have to quantize $\alpha^*$ on a suitable number of bits and to choose a suitable block-size $N$. Even though this optimization might be performed in a RD-optimized fashion, for the sake of simplicity in this first exploratory work we just performed a few preliminary tests and find out that, for the sequences , using a 1/8 precision to quantize $\alpha^*$ and using $N = 256$ brings the best performances.

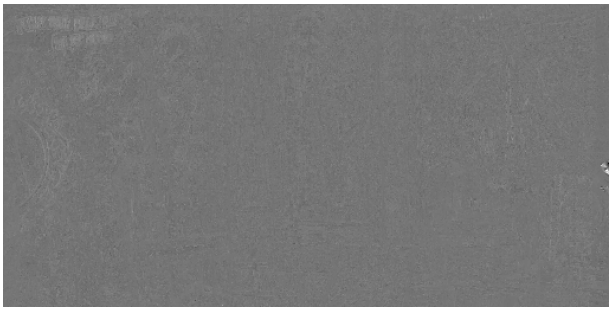
In Fig. 3 we show the values of $\alpha^*$ for the Technicolor Painter sequence and for four different values of $N$. We observe that these values show a small spatial correlation, which suggest that encoding them with *e.g.* a context-based arithmetic encoder (CBAE) would reduce their coding rate. In our implementation we used a CBAE with a two-pixel context (horizontal and vertical neighbors).

In conclusion of this section, we propose five synthesis-based methods in order to reconstruct $V_5$ at the decoder without sending it: using $V_4$ and $V_6$ to obtain an horizontal interpolation ("horizontal"); using $V_2$ and $V_8$ to obtain a vertical interpolation ("vertical"); using the four references $V_4$, $V_6$, $V_2$ and $V_8$ ("four references") ; using the gradient as in Eq. (3) to compute the coefficient for blending $S_{5h}$ and $S_{5v}$ pixel-by-pixel *i.e.* $N = 1$ ("gradient"); and computing the convex combination for $256 \times 256$ blocks and sending the coefficients as side information ("convex").

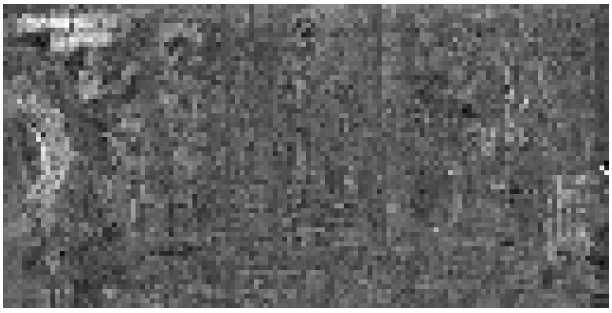
# 3 RESULTS

## 3.1 Experimental setup

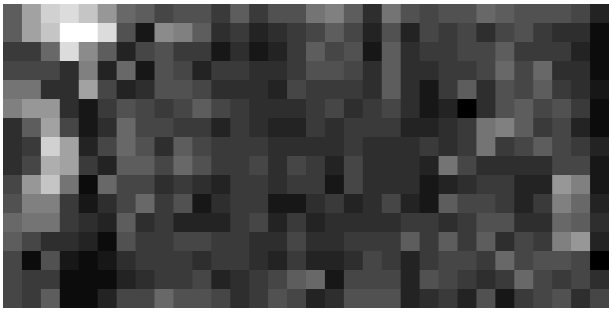
For our experiments we consider three SMV video sequences (Technicolor Painter, Orange Shaman and ULB Unicorn), characterized by good quality depth maps. The anchor is MV-HEVC with the inter-view prediction structure depicted in Fig. 1. We decided to use MV-HEVC for practical reasons related to some limitations of the current implementation of 3D-HEVC reference software, in particular the very large memory requirement of the latter. By using MV-HEVC this requirement is reduced by a factor of two, as texture and depth sequences are treated separately. The prediction



**(a) 4x4**



**(b) 16x16**



**(c) 64x64**

**(d) 256x256**

**Figure 3: $\alpha^*$ values for different block sizes, first frame of the Technicolor Painter sequence**

structure is sub-optimal and differs from MPEG-I Visual Common Test Conditions, but allows to easily compare anchor and proposed methods, since for the latter the central view is just removed from the encoded stream, which for the rest is identical to the anchor. Another limitation of our case-study is that we do not consider temporal prediction – *i.e.* we select an All-Intra configuration. The proposed methods use the same configuration as the anchor but does not encode view $V_5$. For the first four version of our method (horizontal, vertical, four references and gradient), there is no additional bit-rate. The last version (convex) uses the linear combination coefficients that are uniformly quantized with a quantization step of $\frac{1}{8}$ and then encoded with a CBAE with a two-pixel context.

Each sequence is encoded using five QPs: 25, 30, 35, 40 and 45. The first four are used to compute the Bjontegaard metrics at medium rate, while the last four are used for the low-bitrate range.

## 3.2 Experimental results

In Tab. 1, 2 and 3 we report the quality of the synthesized views of the four methods that do not require additional signaling, evaluated via the PSNR with respect to the original central view. The four-references method has the best synthesis quality in all the cases except one, where gradient is better. The latter method has often PSNR values close to the best, while the two simpler interpolation methods (horizontal and vertical) are less effective. While it was expected that the gradient technique performs better at higher rates than at lower rates (since the gradient estimation is more reliable), it was not sure that it could provide a better synthesis than simple horizontal or vertical, since the gradient method is open-loop. As for the convex method, the PSNR of the synthesized view is not directly comparable to that of the other methods, since the former requires additional side information, while the latter do not. The convex method can be compared to the others by using the Bjontegaard delta rate, as shown later on; however we observe that the PSNR of the central view synthesized with the convex method was always better than the PSNR of the other four methods, including four-references, except one case, where the gradient method was better.

In Fig. 4 we show the synthesized images for four of the methods, for the sequence Technicolor Painter and for QP equal to 25. The improved quality of the convex method can be seen in the better representation of some details, such as the small table on the bottom right part of the image.

| Method | QP25 | QP30 | QP35 | QP40 | QP45 |
|---|---|---|---|---|---|
| Vertical | 36.73 | 35.81 | 34.56 | 33.24 | 30.80 |
| Horizontal | 36.79 | 36.03 | 34.90 | 33.24 | 31.20 |
| Four references | **37.40** | **36.50** | **35.30** | **33.50** | **31.37** |
| Gradient | 37.38 | 36.44 | 35.25 | 33.45 | 30.85 |

**Table 1: Technicolor Painter, PSNR values for different methods.**

| Method | QP25 | QP30 | QP35 | QP40 | QP45 |
|---|---|---|---|---|---|
| Vertical | 32.09 | 31.09 | 30.46 | 30.11 | 29.57 |
| Horizontal | 33.29 | 32.28 | 31.43 | 30.69 | 29.97 |
| Four references | **34.22** | **33.08** | **32.08** | **31.36** | **30.42** |
| Gradient | 33.78 | 32.68 | 31.78 | 31.15 | 30.27 |

**Table 2: Orange Shaman, PSNR values for different methods.**

**(a) horizontal**

**(b) vertical**

**(c) original**

**(d) convex**

**Figure 4: Results for QP 25 and $N = 256$.**

However, we must underline that the first four method do not require any side information to be sent, while the convex method does. Therefore we compute the Bjontegaard delta rate (BD-rate ) of the five methods with respect to the anchor, for low and medium bit-rate ranges. The results are reported in Tab. 4.

| Method | QP25 | QP30 | QP35 | QP40 | QP45 |
|---|---|---|---|---|---|
| Vertical | 30.78 | 29.97 | 28.91 | 27.34 | 25.21 |
| Horizontal | 29.86 | 29.27 | 28.17 | 26.96 | 25.12 |
| Four references | **32.15** | **31.38** | **30.13** | **28.27** | 25.98 |
| Gradient | 31.39 | 30.71 | 29.63 | 27.97 | **26.31** |

**Table 3: ULB Unicorn, PSNR values for different methods.**

| | Average, low bit rates [%] | Average, medium bit rates [%] |
|---|---|---|
| Vertical | 1.03 | 4.40 |
| Horizontal | 1.62 | 5.38 |
| Four references | **-1.38** | 1.43 |
| Gradient | -0.66 | 2.35 |
| Convex | -1.24 | **1.38** |

**Table 4: Bjontegaard BD-rate results.**

We see that the synthesis-based methods may have the potential of improving the performance of the anchor, but only if more sophisticated method than simple horizontal or vertical interpolation are considered, and only in the lower bit-rate range. Moreover these results are obtained for an All-Intra configuration, thus the extension to other temporal prediction structures must be carefully considered. The results show some variance among the sequences: for example, for the Technicolor Painter sequence all the methods are better than the anchor, with the convex method gaining up to almost -4% BD-rate . For this sequence we also show the PSNR of the central view as a function of the total encoding rate in Fig. 5, showing the advantage of convex and gradient method over the simple horizontal and vertical syntheses. As it was shown from the Bjontegaard metric, the the four-references method has the best performance at lower bitrate, while the Convex is the best at medium bitrate (even though it is worse than the anchor in this case).

The fact that the synthesis methods achieve better results at lower rates was also expected, since at very high rates the quality of $V_5$ is bounded by the quality of the synthesis, while the anchor can encode the original view and achieve arbitrarily high PSNR values.

In Tab. 5 the proportion of metadata in total bitstream for the convex method is shown. The metadata rate remains nearly constant for all quality points, but with decreasing overall bitrate its percentage increases from 0.06% for QP 25 to up to 1.25% for QP45.

| Sequence | QP25 | QP30 | QP35 | QP40 | QP45 |
|---|---|---|---|---|---|
| Technicolor Painter | 0.05% | 0.11% | 0.20% | 0.37% | 0.71% |
| ULB Unicorn | 0.05% | 0.09% | 0.18% | 0.36% | 0.62% |
| Orange Shaman | 0.06% | 0.14% | 0.28% | 0.61% | 1.25% |

**Table 5: Proportion of the metadata in total bitrate.**

Moreover, a closer look to the synthesized images reveals that they show a small misalignment, due to the imperfections of the warping module in the synthesis. This suggests how the synthesis

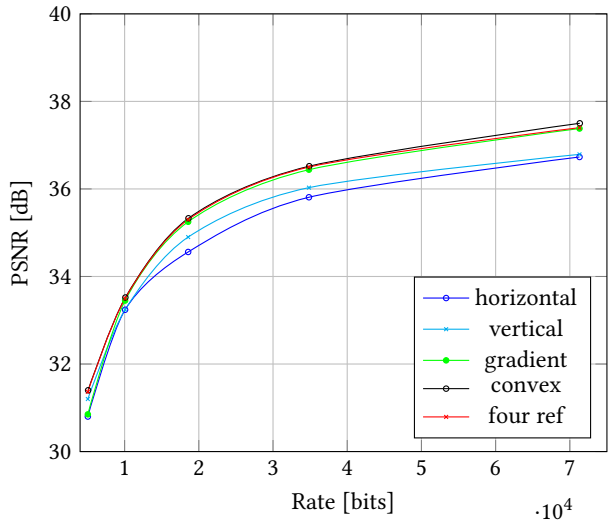



**Figure 5: Technicolor Painter, Rate vs PSNR for different Methods. PSNR indicates the quality of synthesized view.**

could be better used for compression: the synthesized images can just be put in the decoder frame buffer and used as additional references for image prediction. Nevertheless, implementing this method would require solving other problems including optimal selection of synthesized references, the position in the decoder frame buffer, etc. along with a significant implementation work, so it is left for future works.

## 4 CONCLUSIONS

In this work we consider view-synthesis as an helper for compression, by skipping a viewpoint in a SMV video and only using synthesis for reconstructing it at the decoder side. Five synthesis methods have been compared, four requiring no side information and a fifth that uses side information in order to perform a convex combination of the horizontal and vertical synthesis. Experiments show that the simpler synthesis method cannot beat the anchor for compression, while there is some potential for the more sophisticated methods, at least at low bit-rates, for some specific content and in an All-Intra configuration. We have also to recall that the prediction configuration of our anchor is sub-optimal but allows a simple comparison among methods and the anchor. Therefore, if skipping the central view had to achieve better performance than the anchor, it can only happen by using the more sophisticated synthesis method. Moreover, the synthesis methods will probably have to be implemented within the encoder, in order to provide an enhanced view-synthesis prediction tool. These issues, along with the study of more general configurations than the All-Intra , will be the subject of future works.

## REFERENCES

[1] Boissonade and Jung. 2019. Versatile View Synthesizer 2.0 (VVS 2.0) manual. In *ISO/IEC JTC1/SC29/WG11 MPEG*. w18172.

[2] Dricot, Jung, Cagnazzo, Pesquet, Dufaux, Kovacs, and Kiran Adhikarla. 2015. Subjective evaluation of super multi-view compressed content on high end light field 3D display. *Elsevier Signal Processing: Image Communication* 39 (November 2015), 369–385.

[3] Frederic Dufaux, Béatrice Pesquet-Popescu, and Marco Cagnazzo. 2013. *Emerging technologies for 3D video: creation, coding, transmission and rendering*. John Wiley & Sons.

[4] Fachada, Bonatto, Schenkel, Kroon, and Sonneveldt. 2018. Reference View Synthesizer (RVS) manual. In *ISO/IEC JTC1/SC29/WG11 MPEG*. N18068.

[5] C. Greco, M. Cagnazzo, and B. Pesquet-Popescu. 2010. H.264-Based Multiple Description Coding Using Motion Compensated Temporal Interpolation. In *IEEE Workshop on Multimedia Signal Processing*, Vol. 1. Saint-Malo, France, 239–244.

[6] P. Nikitin, M. Cagnazzo, and J. Jung. 2019. Compression Improvement via Reference Organization for 2D-multiview Content. In *IEEE International Conference on Acoustics, Speech and Signal Processing*. Brighton, UK, 1612–1616. https://doi.org/10.1109/ICASSP.2019.8682999

[7] Wegner, Stankiewicz, Tanimoto, and Domanski. 2013. Enhanced View Synthesis Reference Software (VSRS) for Free-viewpoint Television. In *ISO/IEC JTC1/SC29/WG11*. m31520.

[8] Zhang, Tech, Wegner, and Yea. 2013. 3D-HEVC test model 5. In *ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11*. JCT3V-E1005.

[9] Ce Zhu and Minglei Liu. 2009. Multiple Description Video Coding Based on Hierarchical B Pictures. *IEEE T Circ Syst Vid* 19, 4 (April 2009), 511–521.