Adaptation of the 3D-HEVC coding tools to arbitrary locations of cameras

Jarosław Samelak, Jakub Stankowski, Marek Domański
Poznań University of Technology, Chair of Multimedia Telecommunications and Microelectronics, Poznań, Poland,
{jsamelak, jstankowski}@multimedia.edu.pl, domanski@et.put.poznan.pl

Abstract *– **In the paper, we describe the extensions of the 3D-HEVC compression technology aimed at improved compression efficiency for multi-view sequences acquired from arbitrarily located cameras. Our proposal refines the inter-view prediction by replacing the horizontal shifts with the true mapping in the 3D space. This implies changes in several coding tools, which we describe in details. The paper also reports experimental results on the comparison of the proposed solution to the 3D-HEVC standard codec. We also discuss the influence of the number of views and the view coding order on the compression efficiency.***

Keywords ***— 3D-HEVC, augmented reality, free navigation, arbitrary camera setup, multiview video compression***

I. Introduction

For the natural 3D scenes, the "multiview plus depth" (MVD) representation [12] is probably the most often considered representation aimed at applications in 3D video, virtual navigation, free-viewpoint television and augmented reality. Obviously, the multiple views and the corresponding depth maps can be compressed as simulcast using the standard coding techniques like Advanced Video Coding (AVC) [15] or High Efficiency Video Coding (HEVC) [16]. Such a straightforward approach has been studied elsewhere, e.g. in [17]. More efficient techniques exploit the redundancy resulting from similarities between the individual views. The respective compression methods have been developed through many years. More recently, they have been standardized as Multiview Video Coding (MVC) and MV-HEVC (or MHEVC, i.e. Multiview HEVC) extensions of the AVC and HEVC, respectively.

Even more compression gain can be achieved by exploitation of the depth information [13,14]. Such techniques have been already standardized as the extensions of the AVC and HEVC standards: 3D-AVC and 3D-HEVC. Among these two MVD compression techniques, 3D-HEVC is the most efficient as it is built on the top of the state-of-the-art HEVC compression technology.

The 3D-HEVC technology is designed for the multiview video obtained by a number of cameras with the parallel optical axes and the optical centers located on a straight line. Such camera arrangements are relevant for applications related to autostereoscopic displays. Nevertheless, the recent research is devoted to such applications as virtual navigation, free-viewpoint television and augmented reality, where the cameras are usually located around a scene. This implies a need for adaptation of the 3D-HEVC technology to the multiview video obtained from multiple cameras with arbitrary positions. This need was already identified by Moving Pictures Experts Group (ISO/IEC JTC1/SC29/WG11, MPEG), that has issued Call for Evidence of the existence the respective efficient coding technology [18]. One of the two responses to that Call came from Poznań University of Technology, and this paper describes a technique that is related to this response.

The idea of a respective modification of 3D-HEVC has been already described in [1], while here we are going to discuss a more mature proposal that includes the modifications of several coding tools, the modifications of the 3D-HEVC Test Model [2] and the reference software [3], and the proposal for modifications of the standard bitstream syntax.

II. Point Mapping in 3D space

The 3D-HEVC was designed for encoding multiview video acquired only from the linear camera setup [2]. In such an arrangement, all the cameras are located on a line and their optical axes are in parallel. This implies that all the views are vertically aligned. The 3D-HEVC Test Model utilizes the restriction of linear camera setup e.g. by assuming that the corresponding blocks in the individual views are only shifted horizontally (Figure 1). The implementation of several encoding tools is limited to the linear camera arrangement, which is vital for the encoding time. On the other hand, due to these simplifications, the 3D-HEVC encoder is inefficient for compression of data acquired from non-linear camera arrangement.

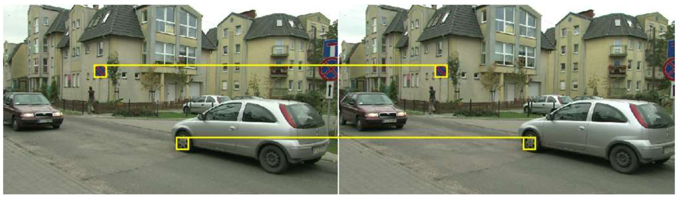

Fig. 1. Vertically aligned views of Poznan Street test sequence [4].

In our proposal, we remove the assumption of vertical alignment of the cameras and generalize the implementation of

Reseach project was supported by The National Centre for Research and Development, Poland. Project no. TANGO1/266710/NCBR/2015.

encoding tools in the 3D-HEVC Test Model. Such a solution allows to distribute cameras around the scene freely, but requires more effort to match the corresponding picture samples within different views. Since the views are not vertically aligned anymore, we need to perform a true mapping of samples in 3D space. The mapping is from a reference view to a target view as presented in Figure 2.

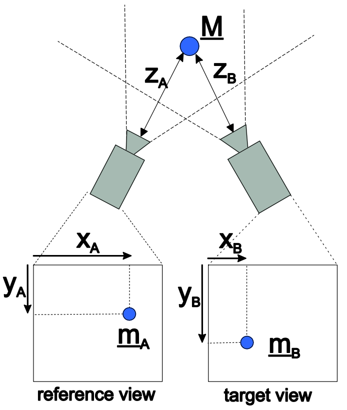

Fig. 2. Point mapping points in 3D space.

Points m_A and m_B (Figure 2) are the representations of point M in the reference and target view, respectively. The position of m_B in the target view can be calculated by projecting point m_A into the 3D space (which results in M) and then mapping it onto the desired view. To achieve that, the scene has to be first fully described by intrinsic and extrinsic camera parameters, enumerated in Table I. Some of these parameters are already present in the 3D-HEVC reference software, while the rest of them is not required with the linear camera setup assumption. The derivation of the camera parameters is out of scope of this paper. It is assumed that all the camera parameters are available on the input of the encoder.

TABLE I. CAMERA PARAMETERS NEEDED IN THE CASE OF LINEAR AND ARBITRARY CAMERA LOCATIONS.

Parameter name	Parameters needed for arbitrary locations	Parameters needed for linear locations
Horizontal focal length	f_x	f_x
Vertical focal length	f_y	-
Horizontal optical center	o_x	o_x
Vertical optical center	o_y	-
Skew factor	c	-
Nearest distance to camera	Z_{near}	Z_{near}
Farthest distance to camera	Z_{far}	Z_{far}
Translation	$\mathbf{T} = [t_x\ t_y\ t_z]$	t_x
Rotation	**R**	-

The intrinsic camera parameters can be gathered into the calibration matrix **K** as in equation (1). The translation vector **T** and the rotation matrix **R** are the extrinsic parameters. For a given camera, a projection matrix **P** can be derived using (2). A projection matrix of a given view allows to map the points from 3D space onto this view. To map the position of a sample from the view A to B, equation (3) is used. $\mathbf{P_A}$ and $\mathbf{P_B}$ are the projection matrices of the view A and B, respectively. The remaining symbols are explained in Figure 2 and in the following equations:

$$\mathbf{K} = \begin{bmatrix} f_x & c & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (1)$$

$$\mathbf{P} = \begin{bmatrix} \mathbf{K} & 0 \\ 0^T & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathbf{R} & -\mathbf{R} \cdot \mathbf{T} \\ 0^T & 1 \end{bmatrix}, \quad (2)$$

$$\begin{bmatrix} z_B \cdot x_B \\ z_B \cdot y_B \\ z_B \\ 1 \end{bmatrix} = \mathbf{P_B} \cdot \mathbf{P_A^{-1}} \cdot \begin{bmatrix} z_A \cdot x_A \\ z_A \cdot y_A \\ z_A \\ 1 \end{bmatrix}. \quad (3)$$

The above equations describe the mapping of the positions between two views. In our solution, this approach is used in the 3D-HEVC coding tools, replacing the simple horizontal shifts.

III. MODIFICATIONS OF THE 3D-HEVC CODING TOOLS

The adaptation of the 3D-HEVC to the arbitrary locations of the cameras requires modifications of several encoding tools. This section provides a detailed description of the most important changes introduced into the Test Model.

A. Disparity Compensated Prediction

The idea of Disparity Compensated Prediction (DCP) tool is based on widely used Motion Compensated Prediction. The difference is that DCP uses already coded pictures of other views as a reference. The DCP derives a disparity vector, which in the case of linear camera arrangement has only the horizontal component. In our solution, it may have two non-zero components.

Suppose that the encoder tries to predict point m_B (Figure 2) using the inter-view prediction. Thus, it has to derive the disparity vector. In our proposal, we map the position of the given point onto the reference view using (3), resulting in the point m_A. Then, we obtain the difference between m_A and m_B and use it as a disparity vector dv (4).

$$dv = m_A - m_B = \begin{bmatrix} x_A \\ y_A \end{bmatrix} - \begin{bmatrix} x_B \\ y_B \end{bmatrix} = \begin{bmatrix} x_A - x_B \\ y_A - y_B \end{bmatrix} \quad (4)$$

B. Inter-View Motion Prediction (IvMP)

In the original 3D-HEVC Reference Software, to calculate the position of the block in the reference picture for deriving candidate motion parameters, the maximum depth value within the associated depth block is converted to a disparity vector. Now, this value is used together with position of the coded block to obtain the 2D disparity vector, following the new derivation process presented in Section II. The resulting vector is then used to point the reference block in another view, which contains candidate motion vectors. These vectors represent direction of motion in the scene, which is different depending on the position of the camera (Figure 3). Thus, in our solution the candidate motion vectors mv_1 should be scaled before using them in the coded view. The scaling can be performed simply by mapping both points a_1 and b_1 into the coded view (resulting in a_2 and b_2) and calculate the difference in their positions.

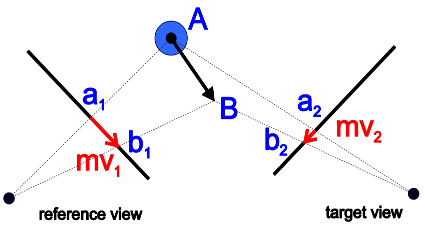

Fig. 3. Different motion vectors, representing the same motion.

C. Neighbouring Block Based Disparity Vector (NBDV)

In the NBDV tool, the disparity vector can be derived from a spatial neighboring block of the Coding Unit. In that case, we must take into account also the position of this block. Moreover, if the candidate is derived from another view, it is scaled as in the Inter-View Motion Prediction tool.

In a similar tool, called Disparity oriented Neighbouring Block Based Disparity Vector (DoNBDV), the disparity vector acquired from NBDV is used to point the virtual depth block. In the unmodified 3D-HEVC, the maximum depth value of the four edge samples of this block is converted to disparity. Our modifications require retrieving also the position of the chosen block corner and using it for deriving the disparity vector.

D. View Synthesis Prediction (VSP)

In View Synthesis Prediction, we use modified neighboring block disparity vector to obtain depth block from a reference view depth image. It estimates the depth information of the coded Prediction Unit. Then, the disparity vectors are derived at the sub-block level using our solution with projection matrices. The last stage is warping samples from reference view and use them as a predictor for the current Prediction Unit. In our solution, warping is performed as presented in Section II.

IV. Modification of the Software

The described modifications were implemented both in the encoder and the decoder, on top of the HTM 13.0, which is a 3D-HEVC Reference Software [3]. This software exploits the assumption of one-dimensional disparity vectors by creating look-up tables that allow to quickly convert depth to disparity. In our solution disparity vectors depend not only on depth, but also on the position of the sample so creating such tables seems pointless due to a very large number of possible combinations. Thus, we derive disparity vectors by performing the 3D mapping on the fly, taking into account depth, position of the sample and the appropriate projection matrices. This results in an increased encoding time and number of calculations, compared to the HTM 13.0. We can reduce this increase by calculating homography matrices, which we derive from projection matrices using (5). Homography matrix **H** is defined for a pair of cameras and allows to warp a sample directly from one view to another ($\mathbf{P_A}$ and $\mathbf{P_B}$ are projection matrices of the source and the target view, respectively)

$$\mathbf{H} = \mathbf{P_B} \cdot \mathbf{P_A^{-1}}. \qquad (5)$$

V. Modification of the Bitstream

The proposed modifications of the 3D-HEVC reference software require an increased number of camera parameters. The additional information has to be included in the bitstream, however transmitting raw camera parameters is not optimal. First of all, the number of camera parameters is high. Secondly, the transmission causes rounding errors which accumulate when calculating projection matrices. Thus, a better solution is to transmit rounded projection matrices after obtaining them from the original camera parameters. Also, transmitting projection matrices is more beneficial than homography matrices because for *N* views there are *N* projection matrices and *N·(N-1)* homography matrices.

Another issue is the precision of transmitted components. The range of projection matrices can be very high so in our solution the bit precision is adjusted dynamically, such that the error caused by rounding is less than 0.05%. This value has been defined experimentally as a compromise between precision and the number of bytes spent on transmitting projection matrices.

All the parameter values related to the modifications are cumulatively sent in the dedicated extension of the Video Parameter Set with minor influence on the syntax, replacing scales and offsets from the unmodified 3D-HEVC. Table II presents the modified syntax. It should be noticed that the projection matrices have 16 values but only the first 12 of them have to be transmitted, since the last four values are fixed.

TABLE II. PROPOSED MODIFIED SYNTAX OF THE 3D EXTENSION OF THE VIDEO PARAMETER SET.

vps_3d_extension() {	Value
cp_precision	ue(v)
for (n = 0; n < NumViews; n++) {	
i = ViewOIdxList[n]	
cp_in_slice_segment_header_flag[i]	u(1)
if (!cp_in_slice_segment_header_flag[i]) {	
vps_cp_znear[i]	se(v)
vps_cp_far[i]	se(v)
for (j = 0; j < 12; j++)	
vps_cp_projection_matrix[i][j]	se(v)
for (j = 0; j < 12; j++)	
vps_cp_projection_matrix_prec[i][j]	ue(v)
}	
}	
}	

VI. Compression Efficiency Evaluation

Two types of experiments were performed in order to evaluate the proposed solution. The goal of the first experiment was to compare the compression efficiency of our proposal, the unmodified 3D-HEVC encoder, MV-HEVC and HEVC simulcast. Another experiment, presented in Section VII, compares different encoding scenarios. All the state-of-the-art encoders are provided within the HTM-13.0 reference software.

The configuration parameters were the same in both experiments and for all encoders. The most important settings are presented in Table III. Parameters in **bold** do not apply to the HEVC simulcast encoder. They are used only by the encoders that utilize the inter-view prediction.

TABLE III. ENCODING CONFIGURATION PARAMETERS.

Parameter	Value
Number of encoded frames	50
Quantization Parameter for views	{25,30,35,40}
Quantization Parameter for depth	{34,39,42,45}
GOP size	8
Intra period	24
Slices per picture	1
Sample Adaptive Offset	on
View Synthesis Prediction	**on**
View Synthesis Optimization	**off**
Inter-view Motion Prediction	**on**
Neighboring Block Disparity Vector	**on**
Depth oriented Neighboring Block Disparity Vector	**on**

The encoding of each sequence was performed at four pairs of Quantization Parameter (QP) values. For comparison we used the averaged bitrate reduction for luma PSNR (Peek Signal-to-Noise Ratio), calculated with the Bjøntegaard formula [11]. The HEVC simulcast encoder was used as a reference. This way, the inter-view compression efficiency for each multi-view encoder can be observed.

In the first experiment, we encoded 3, 5 and 7 views of 10 commonly known multi-view test sequences. These sequences can be divided into two groups:

- linear camera arrangement - Poznan Street, Poznan Hall 2 [4], Dancer [7], Balloons, Kendo [8], Newspaper [9],
- circular camera arrangement - Poznan Blocks [5], Big Buck Bunny Flowers [10], Ballet, Breakdancers [6].

Figures 4, 5, and 6 present the results.

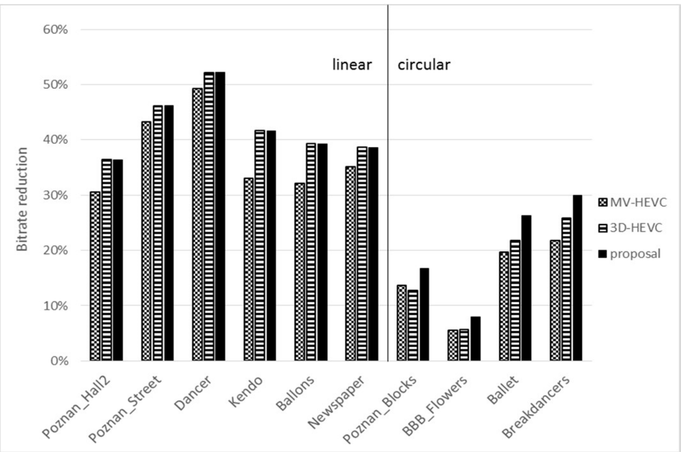

Fig. 4. Bitrate reduction against HEVC simulcast, encoding 3 views.

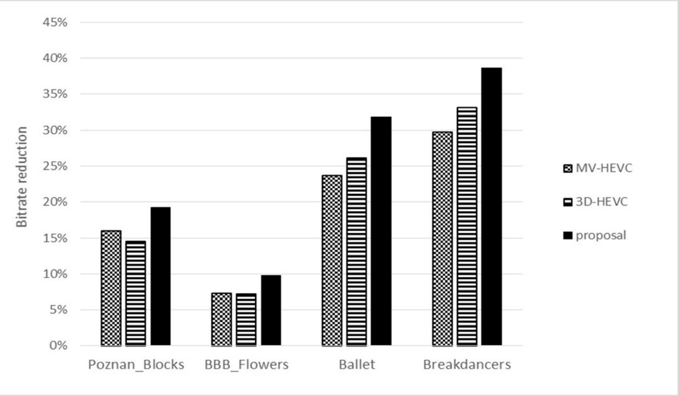

Fig. 5. Bitrate reduction against HEVC simulcast, encoding 5 views.

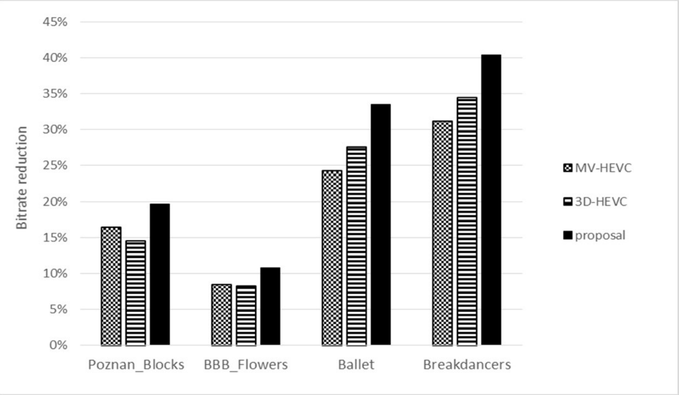

Fig. 6. Bitrate reduction against HEVC simulcast, encoding 7 views.

The first conclusion from these results is that the inter-view compression tools provide a significant gain in the compression efficiency compared to the HEVC simulcast, which encodes each view independently. Next, it can be noticed that the more views are compressed, the higher this gain is. The gain from increasing the number of encoded views is the highest for the proposed solution.

Moreover, our proposal outperforms other encoders in the case of circular camera arrangements. For our proposal, the reduction in bitrate may be as high as 7% when compared to 3D-HEVC. On the other hand, in the case of linearly distributed views, the proposed solution exhibits roughly the same the compression efficiency as the unmodified 3D-HEVC. Slightly increased bitrate is caused by the modifications introduced into the bitstream (Section V), which require transmission of more parameters than for the original solution.

VII. VIEW CODING ORDER

The goal of the second experiment was to determine the compression efficiency related to different view coding orders. Authors propose several coding orders that could be desired for some applications.

A. *Fountain coding order*

It is the most basic order of encoding a multi-view video. First, the central view is compressed and then the side views are encoded as presented in Figure 7. The reference view is always the closest one. The numbers below the views correspond to the coding order. The letters describe the number of views used as a reference for the inter-view prediction (I – no reference view, P – one reference view, B – two reference views).

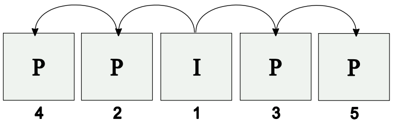

Fig. 7. Fountain coding order.

B. Cascade coding order

The leftmost view is the first encoded view. The remaining views are encoded sequentially from left to right. The reference view is always located on the left (Figure 8).

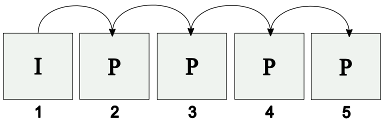

Fig. 8. Cascade coding order.

C. Hierarchical coding order

The compression in this case starts with the central view. Then, the edge views are encoded with the central view as a reference. At the end, the views closest to the central view are compressed with both central and edge views as a reference (Figure 9). The hierarchical coding order would be beneficial e.g. in virtual navigation because the density of the views would be easily scalable, without limiting the angle of observing the scene.

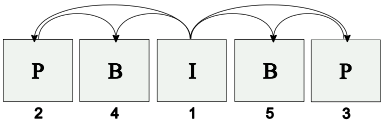

Fig. 9. Hierarchical coding order.

D. Single reference view

The coding order in this case is the same as in the fountain scheme. The difference is that the central view is always used as a reference for the inter-view prediction of other views (Figure 10). Such a solution can highly decrease the encoding time because all the side views can be compressed simultaneously.

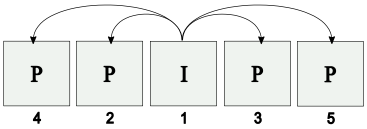

Fig. 10. Coding order with only central view as a reference.

In the experiment, the 3D-HEVC encoder with the proposed modifications was used to compress 5 and 7 views of three multi-view sequences (Poznan Blocks, Ballet, Breakdancers). The configuration parameters were the same as in the previous experiment. The fountain coding scenario was used as a reference. Again, the Bjøntegaard formula for the luma PSNR was used to calculate the average bitrate reduction. The results are presented in Figures 11 and 12.

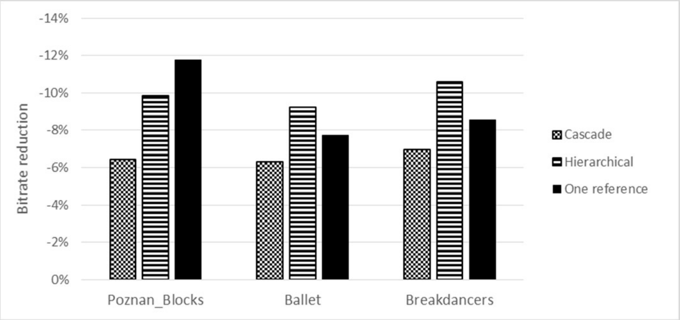

Fig. 11. Bitrate reduction against fountain coding order, encoding 5 views.

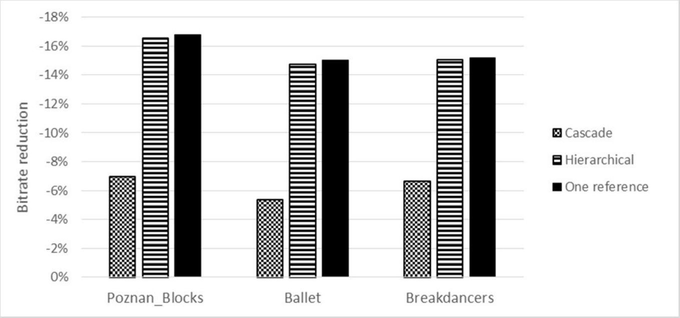

Fig. 12. Bitrate reduction against fountain coding order, encoding 7 views

The results show that the bitrate has increased in all cases, compared to the common fountain coding order. The least increase is observed for the cascade coding scenario, however its usability is limited. For compression of 7 views, the remaining coding scenarios cause roughly 15% of bitrate increase, which may be an acceptable trade-off between compression efficiency and scalability, or encoding time.

VIII. Conclusions

In the paper, authors described in details the modifications of several 3D-HEVC coding tools that allow to efficiently compress multiview video acquired from arbitrarily located cameras. The proposed solution introduces only a minor change into the 3D-HEVC bitstream. The compression efficiency for multiview video with linear camera arrangement is roughly the same as for the unmodified encoder. Nevertheless, for the proposed modifications, the compression efficiency for multiview video with circular camera arrangement is significantly increased (up to 7% bitrate reduction) as compared to the state-of-the-art encoders. This gain is a result of more accurate Disparity Compensated Prediction, which we improved by adding vertical component to the disparity vector, scaling motion vectors candidates and implementing an accurate method of mapping points between views.

The experiments show that the inter-view prediction can provide 5-50% bitrate reduction, compared to the independent encoding of each view. Furthermore, the paper presented a comparison of different coding schemes of multi-view video. The fountain coding order was proven to be the best in terms of the compression efficiency, but the other coding scenarios may be beneficial in the case of some multi-view applications.

Currently, the number of applications for the multiview video is growing. The augmented reality, free navigation in the scene and many others require locating the cameras around the scene. Thus, our proposal to improve the compression efficiency of any camera arrangement is a key issue in the further development of many technologies. The results will be helpful for the development of the forthcoming extensions of the 3D-HEVC standard as well as for the development of the 3D and multiview profiles of the next generation video coding that is expected to be concluded early 2020s.

References

[1] J. Stankowski, Ł Kowalski, J. Samelak, M. Domański, T. Grajek and K. Wegner, "3D-HEVC extension for circular camera arrangements," 2015 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video *(3DTV-CON)*, Lisbon, 2015, pp. 1-4.

[2] Y. Chen, G. Tech, K. Wegner, S. Yea, "Test Model 6 of 3D-HEVC and MV-HEVC", Geneva, October 2013.

[3] 3D HEVC reference codec available online https://hevc.hhi.fraunhofer.de/svn/svn_3DVCSoftware/tags/HTM-13.0.

[4] M. Domański, T. Grajek, K. Klimaszewski, M. Kurc, O.Stankiewicz, J. Stankowski, K. Wegner, "Poznań Multiview video test sequences and camera parameters," ISO/IECJTC1/SC29/WG11, Doc MPEG M17050, Xian, China, Oct.2009.

[5] M. Domański, A. Dziembowski, A. Kuehn, M. Kurc,A. Łuczak, D. Mieloch, J. Siast, O. Stankiewicz, K. Wegner,„Poznan Blocks – a multiview video test sequence and camera parameters for Free Viewpoint Television" ISO/IECJTC1/SC29/WG11, Doc. MPEG M32243, San Jose, USA, Jan. 2014.

[6] C.L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder and R. Szeliski, "High-quality video view interpolation using a layered representation", ACM Transactions on Graphics, vol. 23, pp. 600-608, 2004

[7] D. Rusanovskyy, P. Aflaki, M.M. Hannuksela, "UndoDancer 3DV sequence for purposes of 3DV standardization,"ISO/IEC JTC1/SC29/WG11, Doc. MPEG M20028, Geneva,Switzerland, Mar. 2011.

[8] M. Tanimoto, T. Fujii, N. Fukushima, "1D parallel test sequences for MPEG-FTV," ISO/IEC JTC1/SC29/WG11, Doc. MPEG M15378, Archamps, France, Apr. 2008.

[9] Y.S. Ho, E.K. Lee, C. Lee, "Multiview video test sequence and camera parameters," ISO/IECJTC1/SC29/WG11 Doc. MPEG M15419, Archamps, France, Apr. 2008.

[10] Big Buck Bunny test sequence available online http://www.bigbuckbunny.org/.

[11] G. Bjontegaard, "Calculation of Average PSNR Differences between RD-curves," ITU-T SG16, Doc. VCEG-M33, Austin, USA, Apr. 2001.

[12] Müller K., Merkle P., and Wiegand T., "3D Video representation using depth maps", Proceedings of the IEEE, vol. 99, no. 4, pp. 643–656, Apr. 2011.

[13] K. Müller; H. Schwarz; D. Marpe; Ch. Bartnik; S. Bosse; H. Brust; T. Hinz; H. Lakshman; Ph. Merkle; F. Hunn Rhee; G. Tech; M. Winken; Th. Wiegand "3D High Efficiency Video Coding for multi-view video and depth data", IEEE Trans. Image Processing, IEEE Trans. Image Processing, vol. 22, pp. 3366-3378, 2013.

[14] M. Domański, O. Stankiewicz, K. Wegner, M. Kurc, J. Konieczny, J. Siast, J. Stankowski, R. Ratajczak, T. Grajek "High efficiency 3D video coding using new tools based on view synthesis", IEEE Trans. Image Processing, IEEE Trans. Image Processing, vol. 22, pp. 3517 – 3527, 2013.

[15] ISO/IEC IS 14496-10, "Coding of audio-visual objects, Part 10: Advanced Video Coding," 2014, also as ITU-T Rec. H.264.

[16] ISO/IEC IS 23008-2, "High efficiency coding and media delivery in heterogeneous environments -- Part 2: High Efficiency Video Coding," 2015 also as ITU-T Rec. H.265.

[17] K. Klimaszewski, O. Stankiewicz, K. Wegner, M. Domański, "Quantization optimization in multiview plus depth video coding," IEEE International Conference on Image Processing ICIP 2014, 27-30 October 2014, Paris, France.

[18] G. Lafruit, K. Wegner, M. Tanimoto, "Call for Evidence on Free-Viewpoint Television: Super-Multiview and Free Navigation", ISO/IEC JTC1/SC29/WG11, Doc. MPEG N15348, Warsaw, June 2015.