Geometry-Corrected Deblocking Filter for 360° Video Coding using Cube Representation

Johannes Sauer, Mathias Wien, Jens Schneider and Max Bläser
Institut für Nachrichtentechnik
RWTH Aachen University

***Abstract*—In 360° video, a complete scene is captured, as it can be seen from a single point in any direction. Since the captured 360° images are spherical, they cannot be converted to planar images without introducing geometric distortions. The nature of these distortion depends on the used projection format.**

This paper introduces an approach to reduce artifacts occurring when encoding 360° video which has been projected to the faces of a cube. In order to achieve this, the operation of the deblocking filter is modified such that the correct pixels with respect to the 3D geometry are used for filtering of edges.

The method is evaluated on the set of sequences defined by the Joint Call for Proposals on Video Compression with Capability beyond HEVC. While the method has almost no impact on the objective coding performance, the visual quality is still clearly enhanced. Edges of the cube, previously visible as coding artifacts, are mostly removed with the proposed method.

***Index Terms*—Video compression, 360 degree, deblocking filter**

I. Introduction

Recently, 360° video has gained attention in the video coding research community [1]–[4]. 360° video is now being investigated in an Ad-hoc-group of the Joint Video Exploration Team (JVET) [5]–[7] as a new type of content for video sequences. 360° content has challenges specific to its kind, which have to be addressed.

Loop filters (LFs) are used in video coding in order to reduce coding artifacts or increase the coding performance. For video coding pictures are split into blocks of different sizes. The blocks are then coded using prediction and transform coding. After all video blocks are reconstructed, LFs are executed on a block-wise basis. In the context of HEVC and the exploration software JEM [8] these filter are:

- Deblocking filter (DBF)
- Sample adaptive offset (SAO)
- Adaptive loop filter (ALF)

All of these tools operate on groups of pixels spanning across block edges. They rely on the assumption that neighboring pixels in the coded video also are connected in the scene.

This assumption may be violated for some blocks in 360° video which has been projected to the faces of a cube. Depending on the arrangement of the cube faces, the pixels which are connected in the 3D cube are different from those connected in the 2D representation (e.g. edge 10/12' in Figure 2). Consequently, LFs operating across these faces do not work correctly. They may even produce additional artifacts.

This paper proposes modified LFs for 360° content in cube format with the aim of reducing artifacts occurring at face edges. While LFs are tools commonly used in video coding, the description here will be made in the context of the JEM reference software [8]. The DBF was modified such that instead of using the connected pixels in the 2D projection of the content it operates on pixels as they are connected on the 3D cube. The same should be applied for the ALF and SAO. Here however ALF was disabled at discontinuous face edges inside the picture. SAO was not modified, since it created no significant artifacts.

Other approaches to reduce the artifacts at face edges are:

- Using tiles, whose borders coincide with face boundaries. Then disabling LF across tile boundaries [9].
- Adding a padding area around the faces [10]–[12]. The padded area is discarded when rendering the viewports.

However, both of these approaches will have a decrease in the coding performance, since no prediction across tile boundaries can be performed or additional pixels have to be coded.

The rest of this paper is organized as follows. In Section II 360° representation formats relevant for this paper are introduced. Section III introduces the modifications made to the DBF in order to treat the artifacts correctly. Section IV shows the impact of the method on the coding performance as well as visual comparisons. Section V concludes this paper.

II. 360° video representation

Many different flat representations of 360° video have been developed, among them equirectangular projection, cube projection, octahedron projection, icosahedron projection and segmented sphere projection [13], [14]. In this paper we will focus on the equi-angular cubemap projection (EAC) format, which is a cube based format.

A. Cube representation

The sphere is mapped to the faces of a cube. Since the faces of a cube are quadratic it can be easily split into quadratic blocks. This makes this representation suitable for video coding, as state-of-the-art video codecs also work on pictures split into quadratic blocks.

Different layouts are possible for arranging the cube faces into a rectangular picture, particularly the cube 4x3 (Figure 1) layout and the compact cube 3x2 layout (CMP) [15] (Figure 2). The cube 4x3 layout can be obtained by unfolding a cube, the faces form a connected region shaped like a lying cross. However, there are empty faces in this arrangement, which introduce an overhead for coding. The compact cube

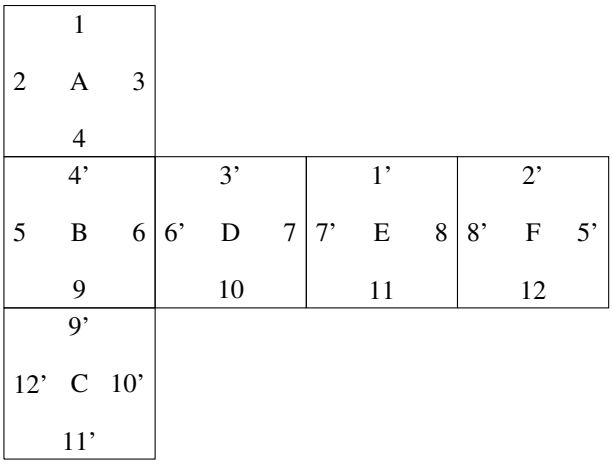


Fig. 1. Unfolded cube. All faces and edges are labeled. Edges have two sides, one side is marked as x, its paired edge as x'.

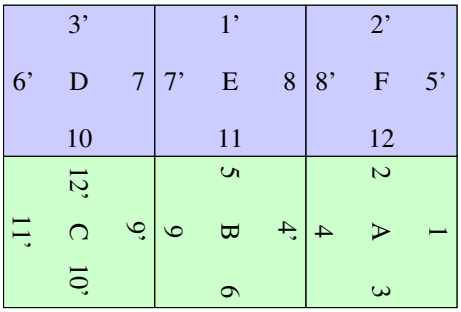


Fig. 2. Original compact 3x2 representation. Orientations shown relative to the unfolded cube (Figure 1). Connected areas (i.e. without discontinuous edges) are shown in green and blue.

3x2 layout avoids this problem. Here the six faces of the cube are arranged in two rows of three faces each. Each row is a connected region on the cube. The resulting video has no empty regions.

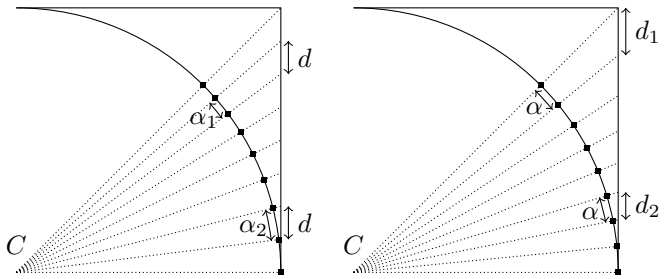
B. EAC representation

The faces of the compact 3x2 layout appear as if they were shot by a conventional camera. CMP is not optimal, since due to the relatively high field of view of 90° the sampling of the sphere significantly deviates from uniform sampling. This effect is illustrated in Figure 3a.

The equi-angular cubemap projection (EAC) addresses this problem, by additionally modifying the mapping of pixels on the sphere to the cube faces, such that equal angles are used between neighboring pixels (Figure 3b). Consequently, EAC provides a more uniform sampling of the sphere than CMP [16]. The coding performance of EAC is better than the one of CMP and EAC was used for the method described in this paper.

III. Corrected deblocking filter for cube representation

A complete description of the DBF is beyond the scope of this paper, but can be found in [17]. Instead we will focus on



(a) CMP (b) EAC

Fig. 3. Sampling of sphere when using compact cube projection (CMP) or EAC. For CMP equidistant locations (d) on the cube face are mapped to the sphere, where they no longer are equidistant. For EAC equal angles are used between neighboring pixels. As a consequence, EAC sphere coordinates project with more uniform resolution onto the cube surfaces. C: Camera center. Adapted from [9].

the modifications that have been made to the DBF.

For deblocking the DBF considers edges on a 8x8 pixel grid. If an edge is aligned with a coding, prediction or transform block boundary it is considered for deblocking. Consequently, if cube face edges should be filtered by the DBF, they have to be aligned with coding block edges. To this end the coded face size (1280x1280) was chosen as a multiple of the maximum coding block size (128x128).

For a correct filtering operation exactly one of each face edge x or its pair x' (Figure 2) must be aligned with edges processed by the DBF. The deblocking in JEM is implemented to operate on the top or left borders of a given block. In our implementation we also consider only the top of left borders of cube face edges. The right and bottom borders are filtered automatically when their counterparts are processed. To this end, the DBF was modified to also allow filtering of the top and left picture borders. This is necessary since 360° video features rotational symmetry. When moving out of the picture at a border, one will reach another part of the image, instead of moving out of the scene.

The compact 3x2 cube format (Figure 2) is not suitable to apply the face edge deblocking in this manner, since problems occur at the edges 10/10' and 2/2'. Both edges 10 and 10' are at the bottom of face borders and thus not filtered. The edges 2 and 2' are both aligned with top face borders. Thus they are filtered twice, once when the blocks containing edge 2' are processed and again for the blocks of edge 2. Note that this is purely a restriction of our implementation. A more sophisticated implementation could support arbitrary face arrangements.

An alternative compact cube format which is suitable to apply DBF with our method is shown in Figure 4. Here every edge x or its pair x' is exactly once aligned with the top or left borders of coding blocks. Consequently all cube face edges are filtered once. The face edges which are aligned with coding block edges considered by the DBF are: 2, 11', 8', 4, 10',

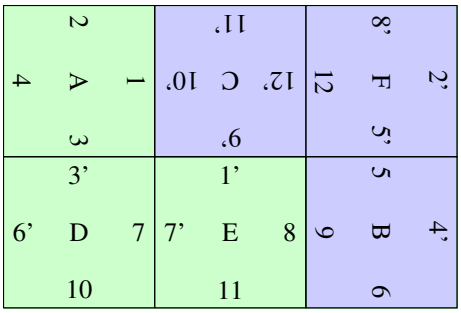


Fig. 4. Alternative compact 3x2 representation. Orientations shown relative to the unfolded cube (Figure 1). Connected areas (i.e. without discontinuous edges) are shown in green and blue.

12, 3', 1', 5, 6', 7' and 9. Blocks belonging to their paired edges are processed at the same time as the blocks belonging to these edges.

Note that the connected regions in both arrangements consist of three faces and they both have three discontinuous face edges (not counting the picture borders). Thus, a similar coding performance can be expected for both formats.

For the edges on the 8x8 pixel grid which are considered for filtering a boundary strength parameter is derived. A fixed value of 2 was used for the boundary strength parameter of edges aligned with cube face edges.

Before a 8x8 block is deblocked it is further split into 4x4 blocks. The filtering is then performed for each 4x4 block and its left or top neighboring 4x4 block. The values of the pixels in this region (Figure 5) are used are used to decide whether filtering is performed at all and if strong or weak filtering is performed. Thus all operations accessing these pixels have been modified to use the correct pixels as defined by the 3D arrangement of the faces. Specifically these operations are:

- Calculation of the absolute values of the second order derivatives for block filtering decision.
- Decision of strong or weak filtering.
- Deblocking of Luma pixels.
- Deblocking of Chroma pixels.

IV. Simulation Results

Results are reported for an implementation into the Joint Exploration Test Model (JEM) reference software, version JEM 7.1, using the 360Lib extension, version 5.0 [8], [19].

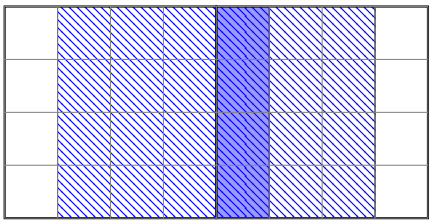


Fig. 5. The pixels accessed by the DBF for deblocking of a vertical edge between two 4x4 blocks. Blue,filled: Edge which is co-located with edge considered by the DBF. Blue, striped: Pixels which are potentially modified by the DBF. Right block is currently processed by the DBF, left block is modified along with it.

The proposed method was tested on the 360° sequences listed in Table I. It is the same set of 5 sequences which is used for the CfP [18]. Three of the sequences feature static cameras, the other two non-static cameras. The chosen coding scenario was a random access configuration, while the set of tested quantization parameters was chosen such that the target rates specified in the CfP could be matched. The sequences were coded using the EAC representation and the modified face arrangement (Figure 4). Only the first Intra period was encoded[1].

The coding performance of the method was measured using the Bjøntegaard Delta (BD) [20]. The BD-statistics are shown in Table II. The reference for the BD-statistics was the coding performance of the JEM reference software. The method has no significant impact on the objective measure of coding performance.

Despite the low impact on the coding performance the method clearly improves the artifacts occurring at face edges. Figure 8 shows a part of a discontinuous face edge and how it is affected by the various LFs. Our method avoids the creation of additional artifacts. Figures 6 and 7 show cube corners for the lowest target rate of the CfP. In Figure 6 discontinuous face edges appear as strong artifacts in the viewports, which are removed by our method. The viewports Figure 7 show face edges located at the picture borders. Here, no loop filtering is applied in the original encoder. However there are still weak artifacts visible.

A simple way to implement the method is building a lookup table for retrieving the geometry-corrected pixels which are used for the DBF. It is sufficient to build the lookup table once at the start of the en- or decoder. Consequently the method has a low complexity. However it may require storing reconstructed blocks for a longer time than is need for conventional deblocking. For example, blocks of edge 4 in Figure 4 have to be kept until edge 4' is completely decoded. Only then deblocking can be performed.

V. Conclusion

In this paper a method was presented which addresses the treatment of artifacts occurring at face edges of 360° video projected to the faces of a cube. This is achieved by correcting which samples are used for the DBF. While the impact on the coding performance of the proposed method is minor, the artifacts are clearly removed. The method requires using an alternative cube face arrangement and the relation of the faces to each other have to be known. This could be signaled at low cost. Further it has only minor impact on the complexity of en- and decoder.

The proposed method can be extended in order to also apply SAO and ALF with geometry-correction. However, this was beyond the scope of this paper.

[1]Simulations for the full sequences were not completed the time of writing. Full results will be available with the camera ready version of the paper

TABLE I
OVERVIEW OF THE USED 360° SEQUENCES. THE SAME SEQUENCES ARE USED FOR THE JOINT CALL FOR PROPOSALS ON VIDEO COMPRESSION WITH CAPABILITY BEYOND HEVC (CFP) [18].

Sequence	Original Resolution	Coded Resolution	Frame rate	Length	Bit depth	Camera type
Balboa	6144x3072	3840x2560	60	600	8	non-static
ChairliftRide	8192x4096	3840x2560	30	300	10	non-static
KiteFlite	8192x4096	3840x2560	30	300	8	static
Harbor	8192x4096	3840x2560	30	300	8	static
Trolley	8192x4096	3840x2560	30	300	8	static

(a) ChairliftRide, Original (b) ChairliftRide, Our method

(c) Harbor, Original (d) Harbor, Our Method

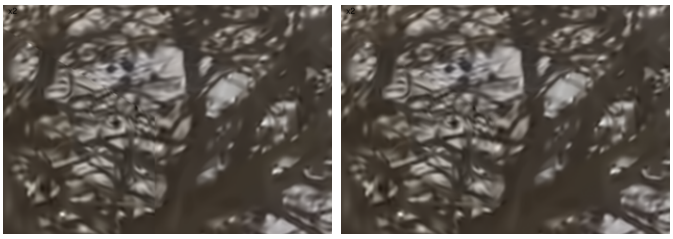
(e) Trolley, Original (f) Trolley, Our Method

Fig. 6. Viewports of corners for the lowest target rate of the CfP. There are discontinuous face edges in the viewports, which show up as strong artifacts in the original.

TABLE II
BJØNTEGAARD DELTA-STATISTICS OF THE PROPOSED METHOD. REFERENCE BITSTREAMS WERE ENCODED USING JEM 7.1 (WITH 360LIB 5.0).

Sequence	Rate	PSNR
Balboa	-0.58 %	0.02 dB
ChairliftRide	-0.39 %	0.01 dB
Harbor	-0.53 %	0.02 dB
KiteFlite	-0.17 %	0.01 dB
Trolley	0.00 %	0.00 dB
Average	-0.33 %	0.01 dB

(a) Balboa, Original (b) Balboa, Our method



(c) KiteFlite, Original (d) KiteFlite, Our Method

Fig. 7. Viewports of corners for the lowest target rate of the CfP. The face edges are at the picture borders, where no LFs are applied in the original. Still, weak artifacts are visible.

REFERENCES

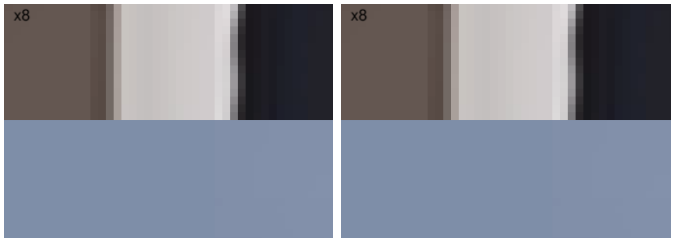
[1] M. Budagavi, J. Furton, G. Jin *et al.*, "360 degrees video coding using region adaptive smoothing," in *2015 IEEE International Conference on Image Processing (ICIP)*, Sept 2015, pp. 750–754.

[2] G. Jin, A. Saxena, and M. Budagavi, "Motion estimation and compensation for fisheye warped video," in *2015 IEEE International Conference on Image Processing (ICIP)*, Sept 2015, pp. 2751–2755.

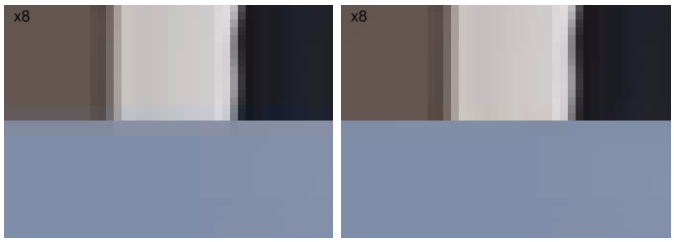
[3] Y. He, Y. Ye, P. Hanhart, and X. Xiu, "Geometry padding for motion compensated prediction in 360 video coding," in *2017 Data Compression Conference (DCC)*, April 2017, pp. 443–443.

[4] Y. Li, J. Xu, and Z. Chen, "Spherical domain rate-distortion optimization for 360-degree video coding," in *2017 IEEE International Conference on Multimedia and Expo (ICME)*, July 2017, pp. 709–714.

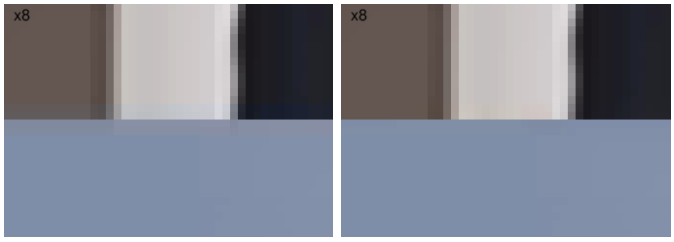
[5] J. Boyce, A. Abbas, E. Alshina, G. van der Auwera, and Y. Ye, "JVET AHG report: 360 video coding tools and test conditions (AHG8),"



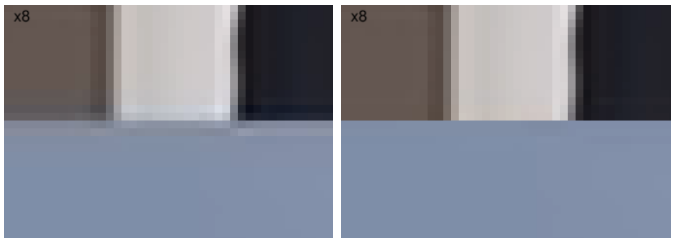
Original, before DBF | Our Method, before DBF



Original, after DBF | Our Method, after DBF



Original, after SAO | Our Method, after SAO



Original, after ALF | Our Method, after ALF

Fig. 8. Comparison of the the different loop filtering steps. Close up of edge 9'/1' of EAC in modified face arrangement (Figure 4). Note that the original face arrangement also has a discontinuous face edge at the same location.

online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Torino, IT, 7th meeting, Doc. JVET-G0008, Jul. 2017.

[6] ——, "JVET AHG report: 360 video coding tools and test conditions (AHG8)," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Macao, CN, 8th meeting, Doc. JVET-H0008, Oct. 2017.

[7] ——, "JVET AHG report: 360 video coding tools and test conditions (AHG8)," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Gwangju, KR, 9th meeting, Doc. JVET-I0008, Jan. 2018.

[8] "Joint Exploration Test Model, version 7.1," https://jvet.hhi.fraunhofer.de/svn/svn_HMJEMSoftware/tags/HM-16.6-JEM-7.1, 2017.

[9] M. Coban, G. Van der Auwera, and M. Karczewicz, "AHG8: Adjusted cubemap projection for 360-degree video," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Hobart, AU, 6th meeting, Doc. JVET-F0025, Apr. 2017.

[10] Y. He, Y. Ye, P. Hanhart *et al.*, "Geometry padding for 360 video coding," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Chengdu, CN, 4th meeting, Doc. JVET-D0075, Oct. 2016.

[11] C.-H. Shih, H.-C. Lin, J.-L. Lin, and S.-K. Chang, "AHG8: Face-based padding scheme for cube projection," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Geneva, CH, 5th meeting, Doc. JVET-E0057, Jan. 2017.

[12] G. Van der Auwera, M. Coban, and M. Karczewicz, "AHG8: Acp with padding for 360-degree video," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Torino, IT, 7th meeting, Doc. JVET-G0071, Jul. 2017.

[13] J. Boyce, "BoG Report on 360 Video," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Chengdu, CN, 4th meeting, Doc. JVET-D0188, Oct. 2016.

[14] Y. Sun, A. Lu, and L. Yu, "AHG8: A study on the influence of different projection schemes," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Chengdu, CN, 4th meeting, Doc. JVET-D0068, Oct. 2016.

[15] M. Zhou, "AHG8: A study on JEM3.0 compression efficiency on 360 video content," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Chengdu, CN, 4th meeting, Doc. JVET-D0024, Oct. 2016.

[16] ——, "AHG8: A study on equi-angular cubemap projection (EAC)," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Torino, IT, 7th meeting, Doc. JVET-G0056, Jul. 2017.

[17] M. Wien, *High Efficiency Video Coding – Coding Tools and Specification*. Berlin, Heidelberg: Springer, Sep. 2014.

[18] A. Segall, V. Baroncini, J. Boyce, J. Chen, and T. Suzuki, "Joint call for proposals on video compression with capability beyond HEVC," online, Joint Video Exploration Team (on Future Video coding) of ITU-T VCEG and ISO/IEC MPEG, Macao, CN, 8th meeting, Doc. JVET-H1002, Oct. 2017.

[19] "360 video conversion software, version 5.0," https://jvet.hhi.fraunhofer.de/svn/svn_360Lib/tags/360Lib-5.0, 2017.

[20] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," ITU-T SG16/Q6 VCEG, Austin, USA, Tech. Rep. Doc. VCEG-M33, 2001.