INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG04 MPEG VIDEO CODING

ISO/IEC JTC1/SC29/WG04/M74633
Oct. 2025, Geneva, CH

Title **[GSC][JEE 6.3-related] Block-based FLAS for video-based GSC**

Source **Poznan University of Technology**
Electronics and Telecommunications Research Institute
Authors **Adrian Dziembowski, Błażej Szydełko, Dawid Mieloch,**
Gwangsoon Lee, Jun Young Jeong, Kwan-Jung Oh

1 Abstract

This document presents a modification to the FLAS algorithm, which adapts the output yuv files to the block-based characteristics of a video encoder.

V2 of the document adds new section of the document, regarding the block-based non-FLAS sorting algorithm.

V3: a simple correction in that section.

2 Algorithm

The proposed modification of the FLAS sorting algorithm is adapted to the block-based characteristics of a video-encoder. We assumed that instead having smooth images it is better to have images with visible blocks, but the interior of these blocks is even smoother.

2.1 *Original FLAS*

In the FLAS algorithm, multidimensional vectors of splat attributes are arranged on a 2D grid in a way which ensures, that neighboring pixels contain similar vectors.

We start with a random assignment of the vectors to grid pixels and a large value of a smoothing radius r, and then we iteratively optimize arrangement of the vectors.

In each iteration we:

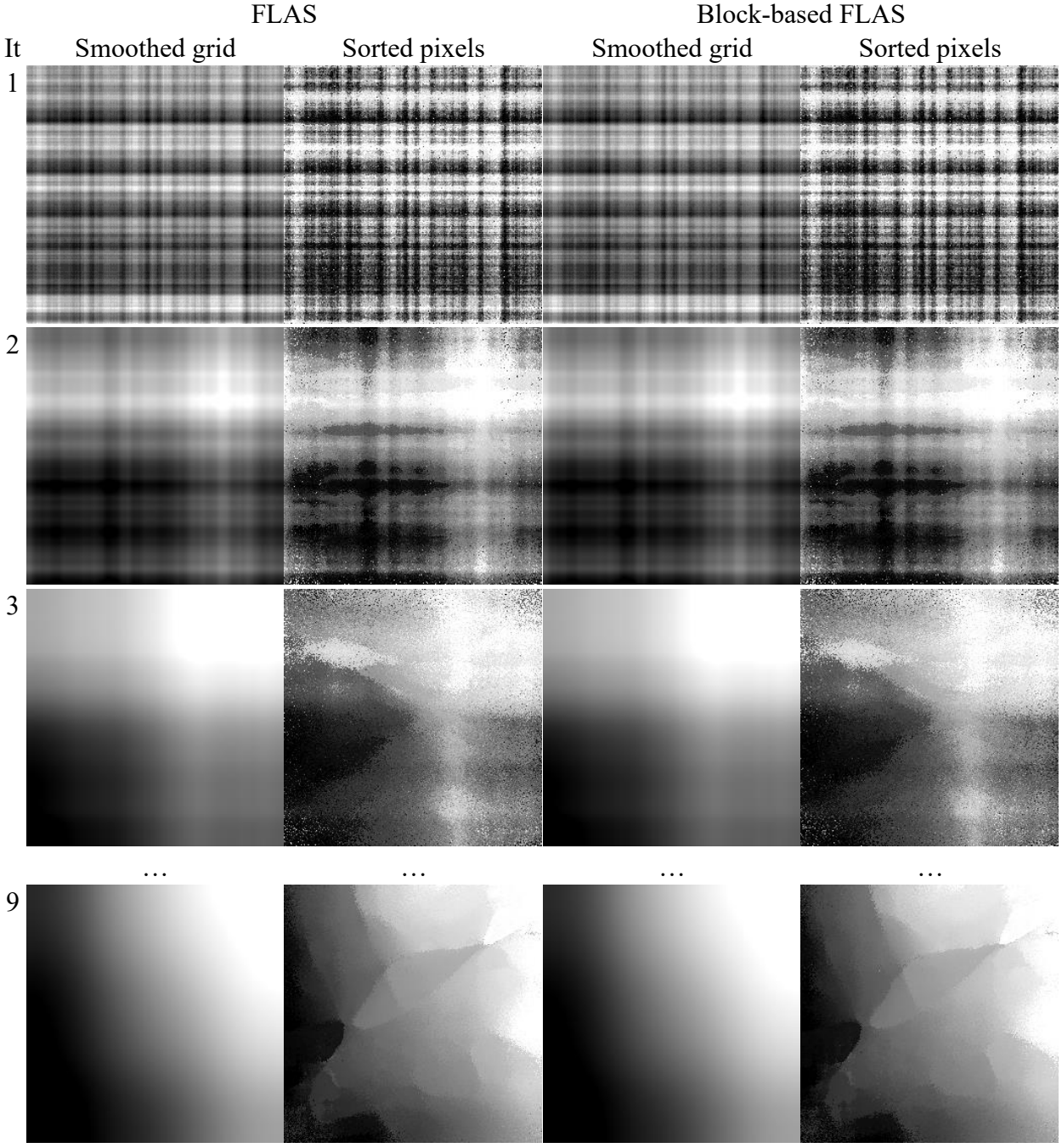
1. smooth the current grid (e.g., using Gaussian filter with filter radius equal to r) – this grid defines the “ideal” local content,
2. pick a random position and a small set (e.g., 49) of candidate vectors within radius r,
3. find a best assignment of those candidates (by minimizing their distance to the smoothed grid),
4. decrease the radius r (until it is larger than 1).

2.2 *Proposed modification*

In the proposed block-based modification, first iterations are performed in the same manner. However, when r becomes smaller than the desired block size (e.g., 64x64), the grid smoothing (first step of the algorithm for each iteration) is performed differently. While for original FLAS, the smoothing is performed on the entire image, in the block-based FLAS it is performed on each

block independently (each 64x64 block is smoothed separately, and the pixels from neighboring blocks are not used in the smoothing operation).

The step-by-step comparison between these two approaches is presented on Fig. 1. It is presented that for first iterations (9 iterations in the presented case) both approaches output the same results. When the r becomes small enough, the grid smoothing operation in the block-based FLAS changes creating visible edges between blocks.



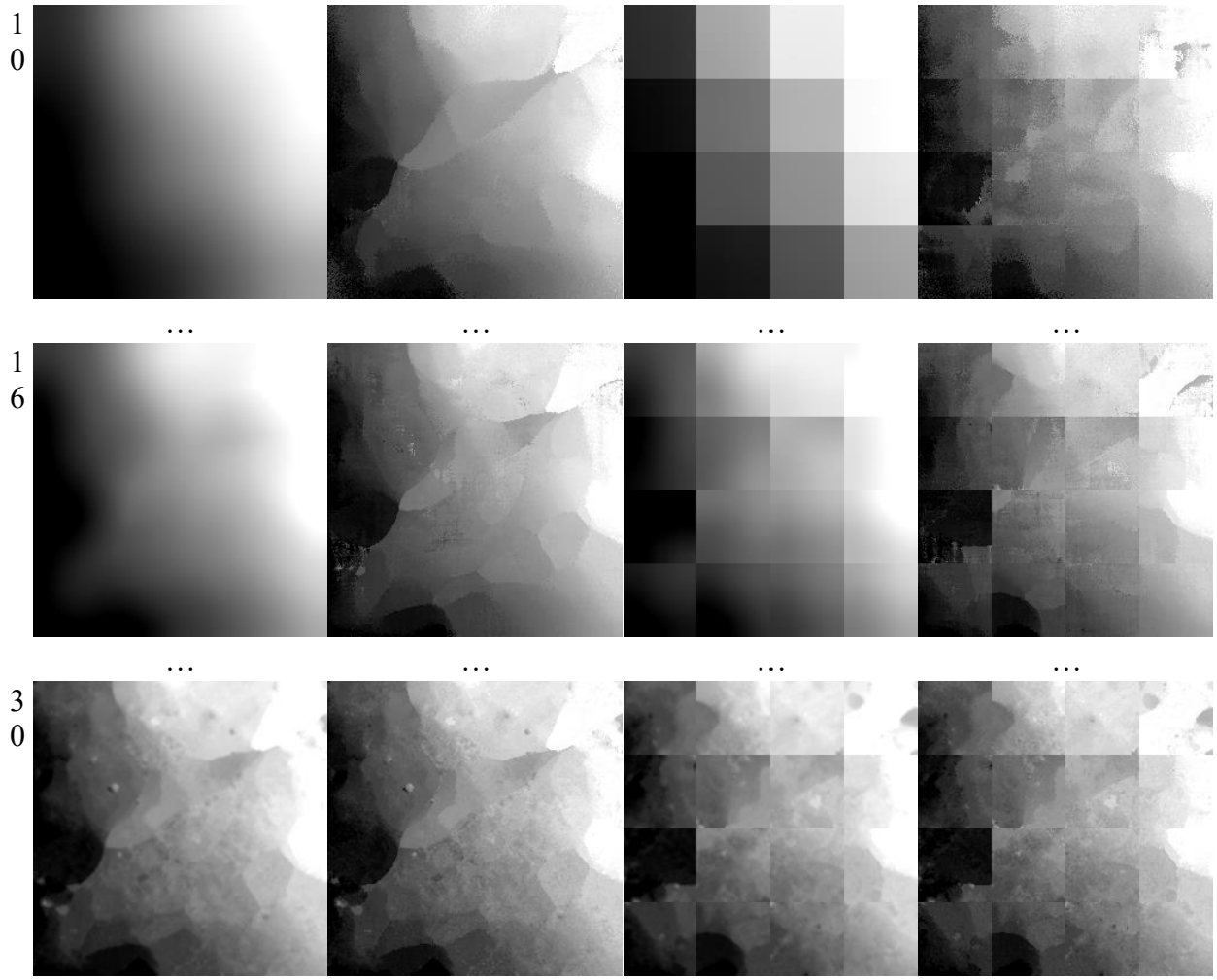


Fig. 1. Comparison between FLAS (two left columns) and proposed block-based FLAS (two right columns), attribute PX, Bartender (part of the scene).

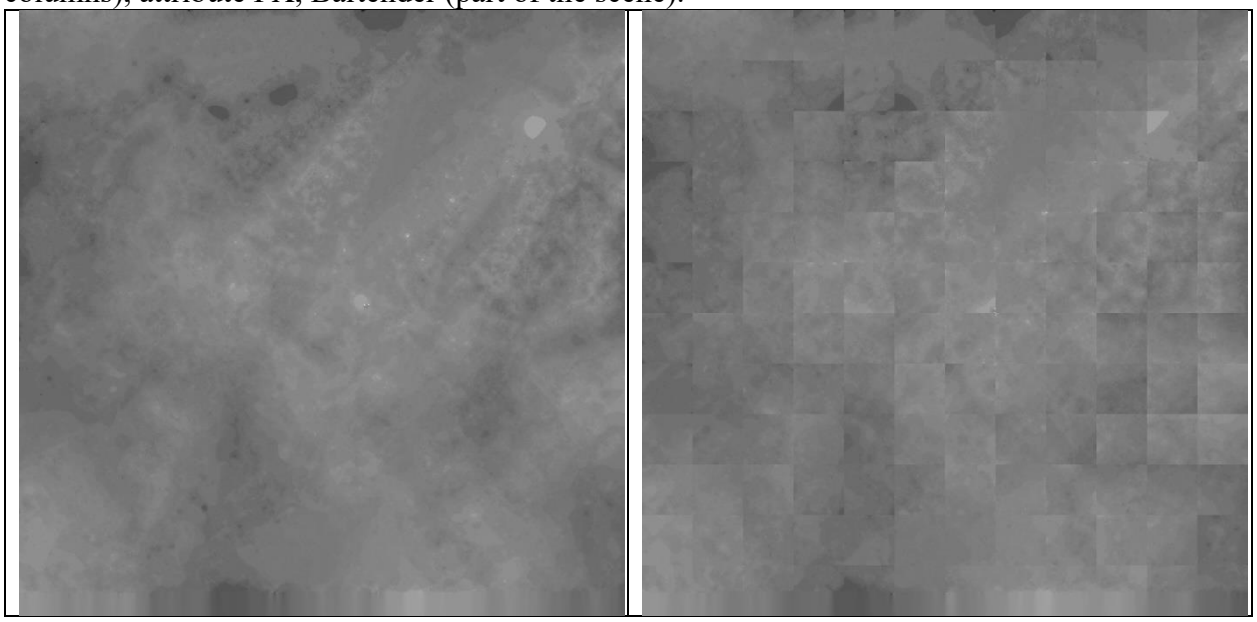
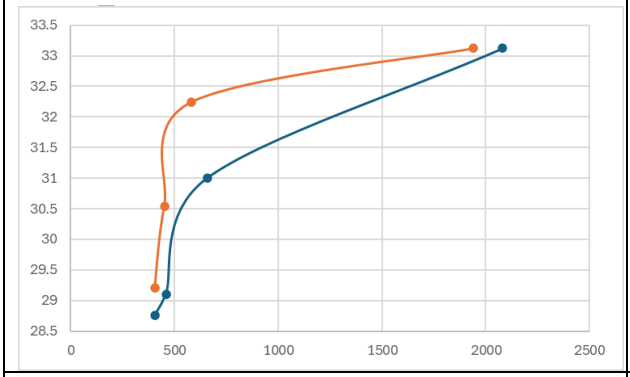
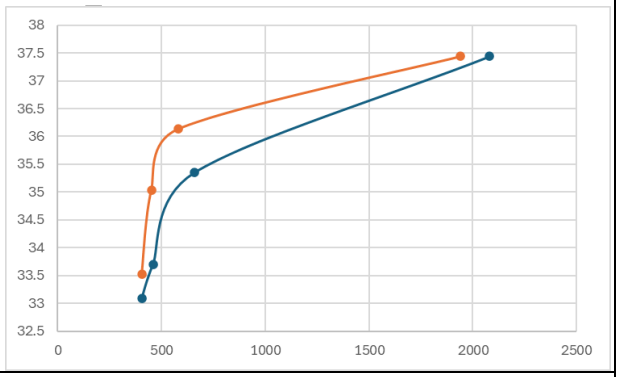
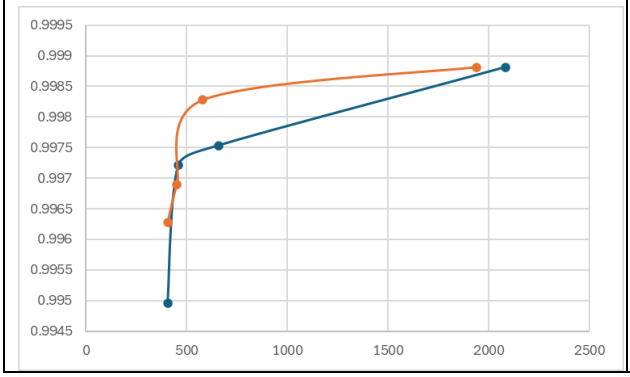
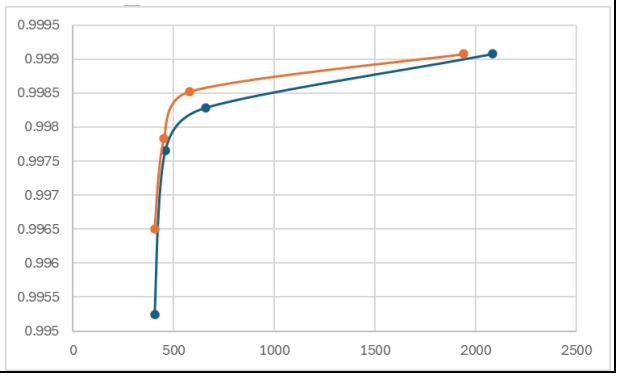
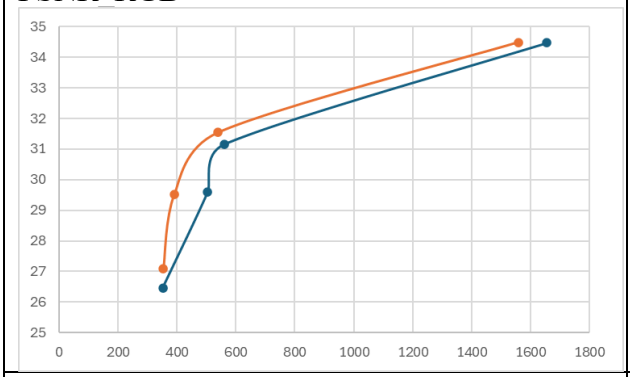
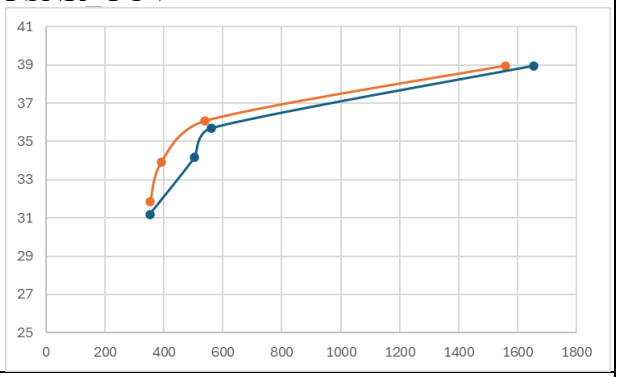
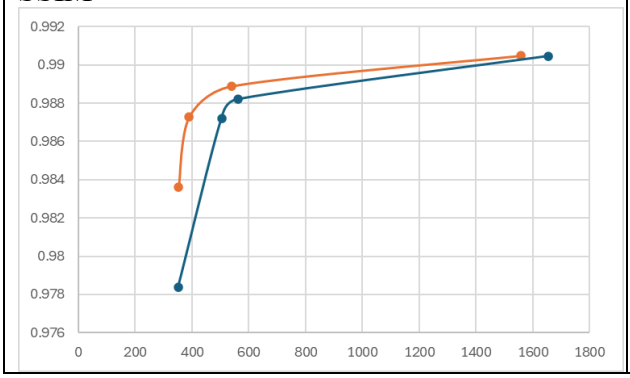
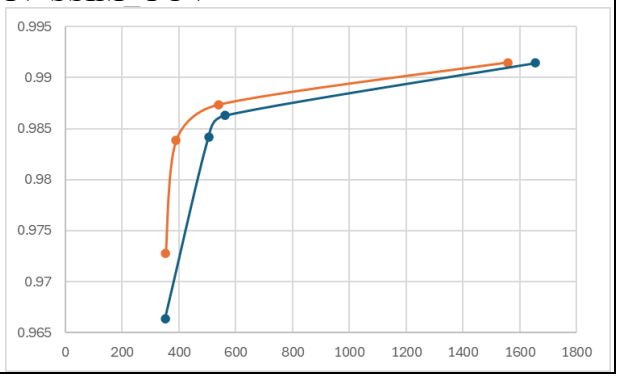


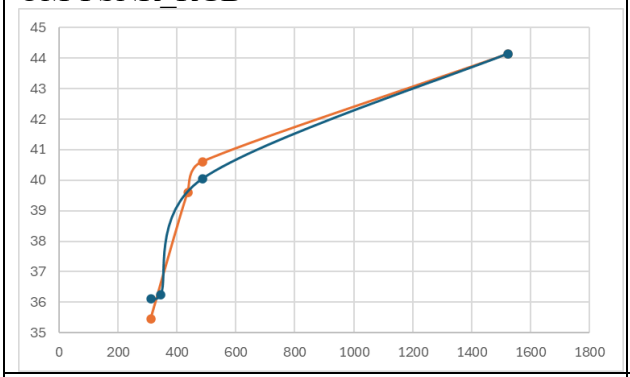
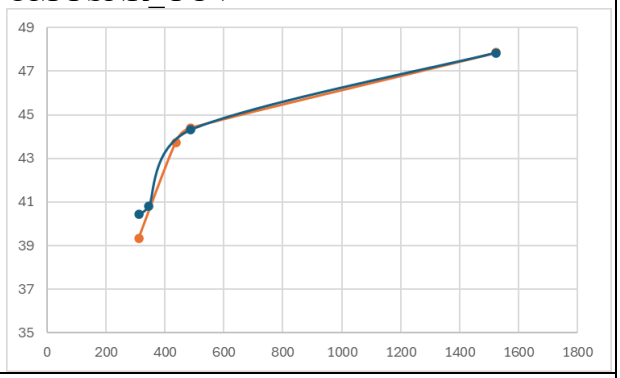
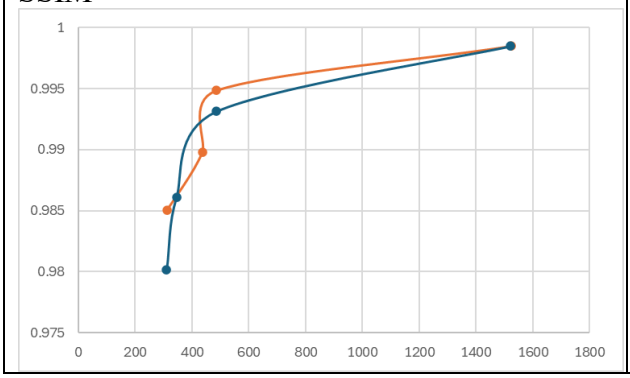
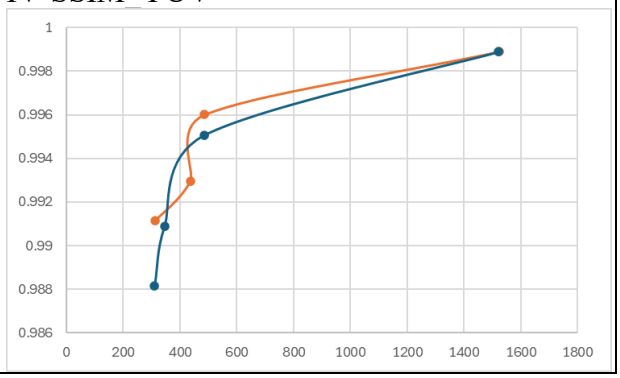
Fig. 2. Attributes sorted using FLAS (left) and proposed block-based FLAS (right), attr. PX, Bartender.

3 Results

The algorithm has been implemented and tested in our internal GSC software, but the implementation in MPEG GSC TMCV would be possible. **Block FLAS vs. FLAS**.

Bartender	
PSNR_RGB	PSNR_YUV
	
SSIM	IV-SSIM_YUV
	

Breakfast	
PSNR_RGB	PSNR_YUV
	
SSIM	IV-SSIM_YUV
	

Cinema	
OM-PSNR_RGB	OM-PSNR_YUV
	
SSIM	IV-SSIM_YUV
	

Remarks:

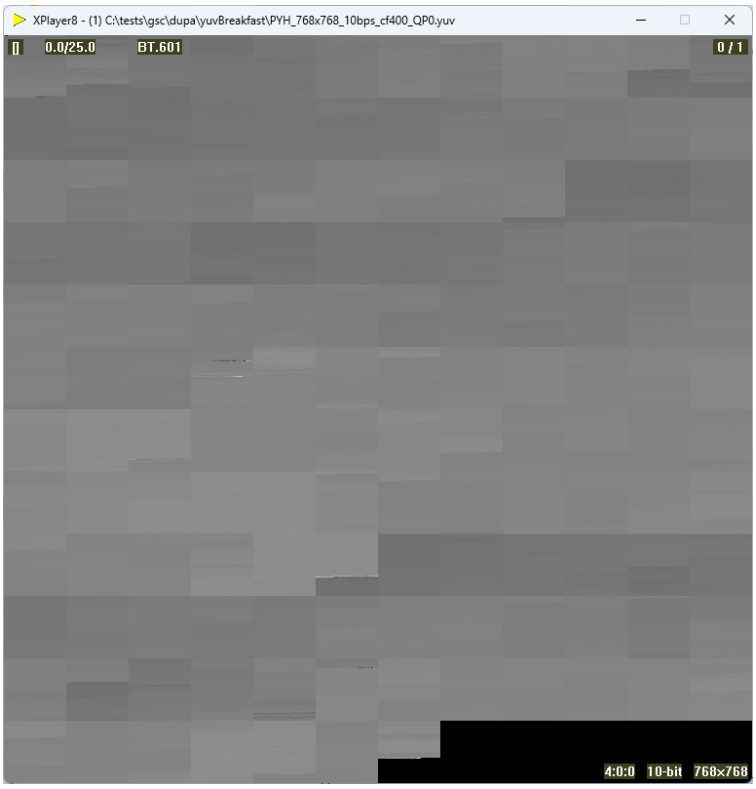
- The results were obtained for 16 frames with GOP = 16.
- Block size was set to 256.

4 Or maybe a non-FLAS block-based sorting?

Inspired by the results presented in m74027 about the Morton-based sorting, we noticed that sorting based on positions (and maybe a couple more attributes) is an efficient way to arrange splats in 2D videos. Therefore, we do not really need FLAS, as we do not utilize its main advantage – sorting based on everything. If not, we should try other, faster methods that are not based on multidimensional optimization.

In our proposal, we decided to use a two-step greedy algorithm, which is way faster than FLAS, while significantly increasing the coding efficiency.

In the first step, we use the sorting based on positions with binary interleaving creating a 1-D vector of splats. Then, we arrange the sorted splats into the grid of 64x64 blocks. In each block, splats are arranged line by line, creating a video containing 64x64 blocks with horizontal lines of similar positions in each.



In the second step, each block is processed independently. For each block, we insert the first splat at the beginning of the list and the second splat at the end of the list. Then, we use the insertion sort algorithm, trying to minimize the total difference between all the neighboring splats. This difference is calculated based on XYZ positions (with weight = 1), luma SH0 (with weight = 0.5), and XYZ rotations (with weight = 0.5). After inserting a third splat, a fourth one is being inserted in the same way.

Example with 3 attributes and 4 splats:

Splats before sorting:

A	B	C	D
2	2	5	5
3	7	7	8
1	10	3	4

Initialization of the list:

A	B		
2	2		
3	7		
1	10		

Total cost = abs(2-2) + abs(3-7) + abs(1-10)

Insertion of the third splat – three possibilities:

C	A	B
5	2	2
7	3	7
3	1	10

A	C	B
2	5	2
3	7	7
1	3	10

A	B	C
2	2	5
3	7	7
1	10	3

Cost CAB: abs(5-2)+abs(2-2)+abs(7-3)+abs(3-7)+abs(3-1)+abs(1-10) = 22

Cost ACB: 19

Cost ABC: 23

Insertion of the fourth splat – four possibilities:

D	A	C	B
5	2	5	2
8	3	7	7
4	1	3	10

A	D	C	B
2	5	5	2
3	8	7	7
1	4	3	10

A	C	D	B
2	5	5	2
3	7	8	7
1	3	4	10

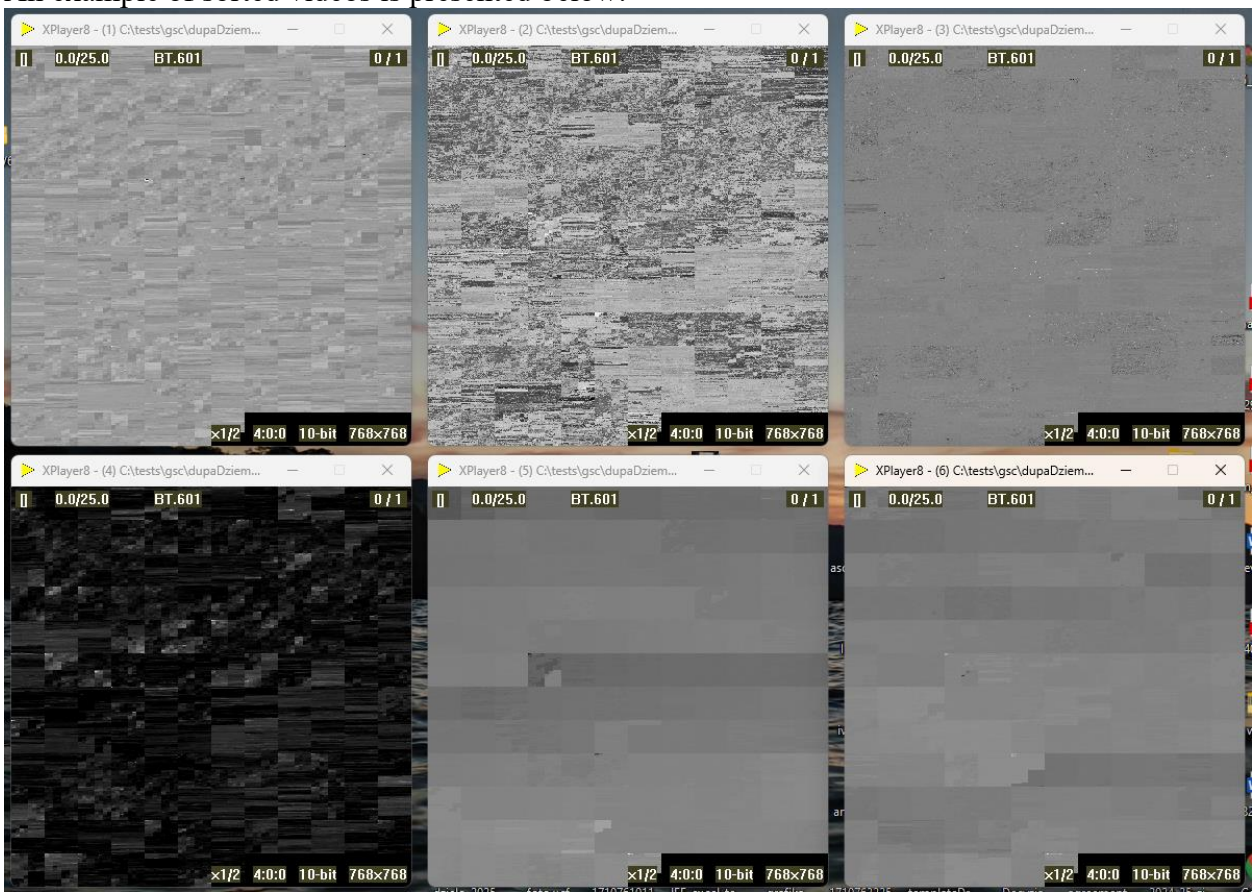
A	C	B	D
2	5	2	5
3	7	7	8
1	3	10	4

Cost DACB: 30
Cost ADCB: 23
Cost ACDB: 21
Cost ACBD: 29

Once inserted, splat is not removed from the list, making the algorithm greedy and the output suboptimal. However, it allows for achieving negligible sorting time.

Then, rearranged splats are written into the block of the output videos line by line, unless the total cost exceeds the threshold. In our experiment, we set a threshold to 25 * blockW * blockH. If the cost is higher, the block is split into two smaller blocks, and the procedure (of writing, not sorting, sorting was already finished) is repeated. It is repeated recursively until the block is bigger than 8x8.

An example of sorted videos is presented below:



Top row: rotation X, scale X, chroma U SH0
Bottom row: luma SH0, position X, position Y

The splats were sorted based on positions with weight = 1 (so position videos are the smoothest), luma (with weight = 0.5), and rotations (with weight = 0.5). Rotations and chromas were not used in sorting, but it can be seen that these attributes are also “kind of sorted”.

In the proposed approach there is no need for dummy patches. The remaining area (right bottom corner) is empty.

4.1 *Results*

Block-based FLAS BD-rate change vs no sorting

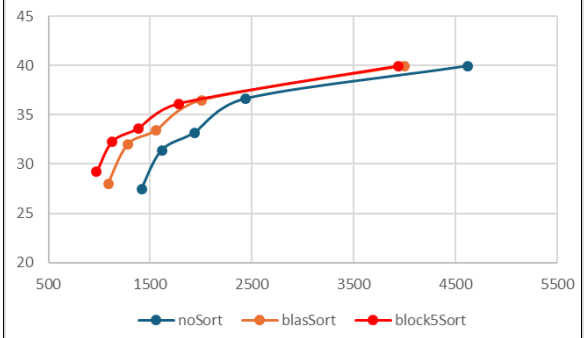
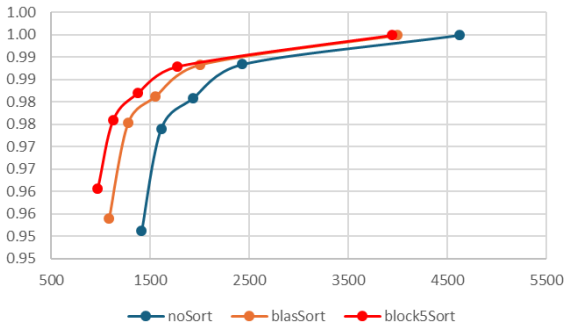
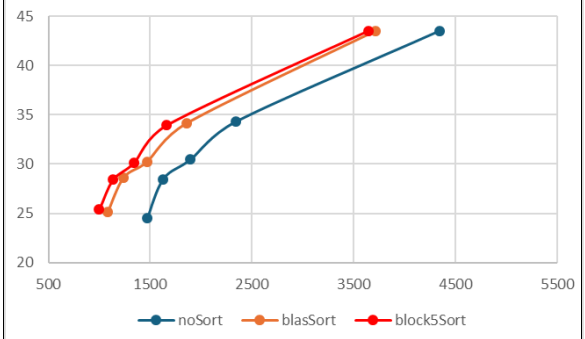
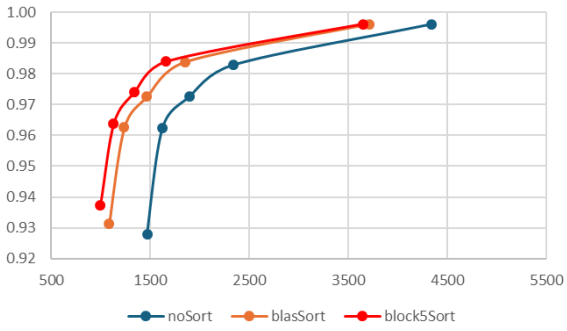
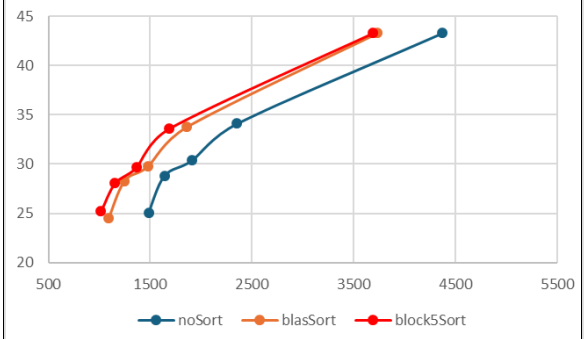
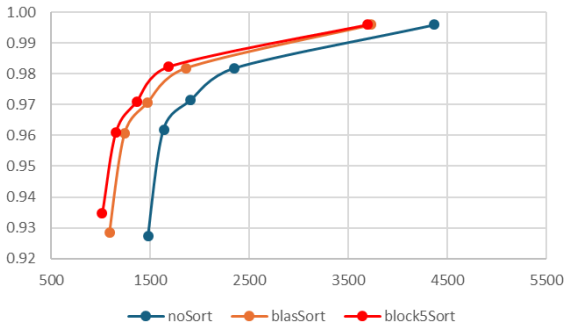
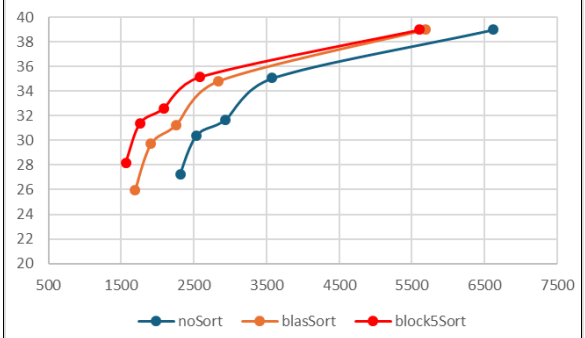
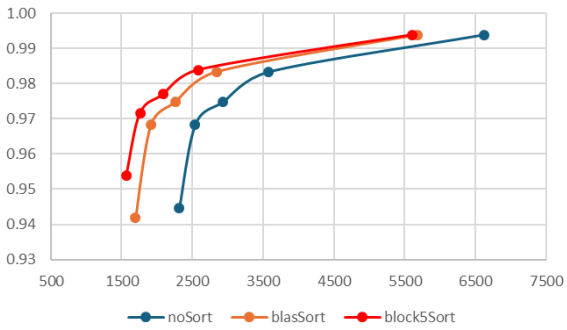
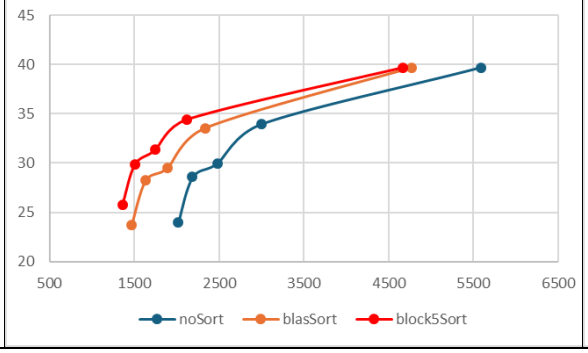
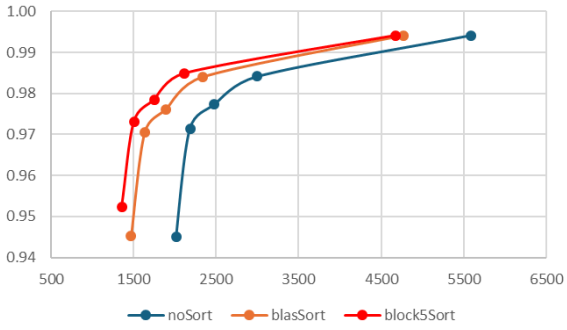
Sequence	*PSNR YUV*	*SSIM*	*IV-SSIM YUV*
Bartender tracked	**-22%**	**-22%**	**-21%**
Breakfast tracked	**-23%**	**-24%**	**-23%**
Breakfast partially tracked	**-21%**	**-24%**	**-22%**
Bartender partially tracked	**-21%**	**-25%**	**-22%**
Cinema partially tracked	**-23%**	**-25%**	**-24%**
Average	**-22%**	**-24%**	**-22%**

Block5Sort (proposed) BD-rate change vs no sorting

Sequence	*PSNR YUV*	*SSIM*	*IV-SSIM YUV*
Bartender tracked	**-32%**	**-32%**	**-30%**
Breakfast tracked	**-29%**	**-32%**	**-31%**
Breakfast partially tracked	**-27%**	**-30%**	**-28%**
Bartender partially tracked	**-34%**	**-33%**	**-31%**
Cinema partially tracked	**-35%**	**-33%**	**-33%**
Average	**-31%**	**-32%**	**-31%**

Block5Sort (proposed) BD-rate change vs block-based FLAS

Sequence	*PSNR YUV*	*SSIM*	*IV-SSIM YUV*
Bartender tracked	**-13%**	**-13%**	**-11%**
Breakfast tracked	**-8%**	**-10%**	**-10%**
Breakfast partially tracked	**-7%**	**-8%**	**-8%**
Bartender partially tracked	**-16%**	**-12%**	**-12%**
Cinema partially tracked	**-16%**	**-11%**	**-14%**
Average	**-12%**	**-11%**	**-11%**

Seq.	*PSNR YUV*	*SSIM*
Bartender tracked		
Breakfast tracked		
Breakfast partially tracked		
Bartender partially tracked		
Cinema partially tracked		

5 Recommendations

We recommend further exploration of the proposed approach.

6 Acknowledgment

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00207, Immersive Media Research Laboratory).