INTERNATIONAL ORGANIZATION FOR STANDARDIZATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC 1/SC 29/WG 07
MPEG 3DGH

ISO/IEC JTC 1/SC 29/WG 04 m76580
April 2026, Ibiza, ES

Title **[GSC][VPCC][AMD1] Block-based splat sorting within raw patches**

Source **Adrian Dziembowski, Błażej Szydełko, Dawid Mieloch (PUT)**
Gwangsoon Lee, Kwan-Jung Oh, Jun Young Jeong (ETRI)

Abstract

This document proposes a block-based sorting algorithm for Gaussian splats within raw patches to improve video coding efficiency. To fully exploit the block-based characteristics of video encoders and ensure perfect alignment of the components within the coding grid, the sorting method is complemented with a planar data arrangement syntax enabling non-continuous raw patch data. It is recommended to adopt the proposed syntax and semantics, and to establish the block-based sorting method as a CTC default strategy.

1 Proposed solution

We propose to sort the Gaussian splats within each patch using a block-based sorting proposed in [m74633v3] on a top of the NN sorting proposed in [m76156]. The splats are sorted based on their geometry and – optionally – also key attributes (luma SH0 and rotations). The block-based approach allows for adapting the video to the block-based characteristics of the video encoder.

1.1 Sorting description

1.1.1 Initial sorting

In the first step, the splats within a raw patch are sorted using the KD-tree based nearest-neighbor chain sorting, as described in [m76156]. This step results in a sorted 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 patch containing 64x64 blocks with horizontal lines of similar positions in each.

1.1.2 Greedy sorting within the block

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), and optionally also 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 4 splats, sorting on 3 geometry values only:

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.

1.1.3 Remarks

Remark 1: In order to make the proposed sorting efficiently work, the spatial resolution of the raw patches was changed, and both AtlasPatch2dSizeX and AtlasPatch2dSizeY are set to be multiples of B. B is set to 16. If the patch contains lower number of pixels (i.e., N < B^2), B is lowered by a factor of two. If needed (if the patch contains a few splats), this step is performed recursively until either N >= B^2 or B = 1.

Remark 2: In order to efficiently pack the patches within an atlas, they are first sorted by their size, descendingly. The biggest patches are packed first.

1.2 Geometry atlas comparison

Fig. 1 presents a fragment of a geometry atlas sorted using the proposed method. Orange box highlights a single raw patch. As presented, first of three components, X, is perfectly aligned with the 16x16 grid. Two other components, however, are not – they contain sharp rectangular regions, but their boundaries do not fit the grid. The reason is the data arrangement in the raw patches, which is strictly continuous (Fig. 3).

Therefore, we propose to allow for planar data arrangement, in which the data are continuous within a single component, but all the components have to be horizontally aligned (Fig. 4). With such a modification, all the data are perfectly aligned with the grid, increasing the video coding efficiency.

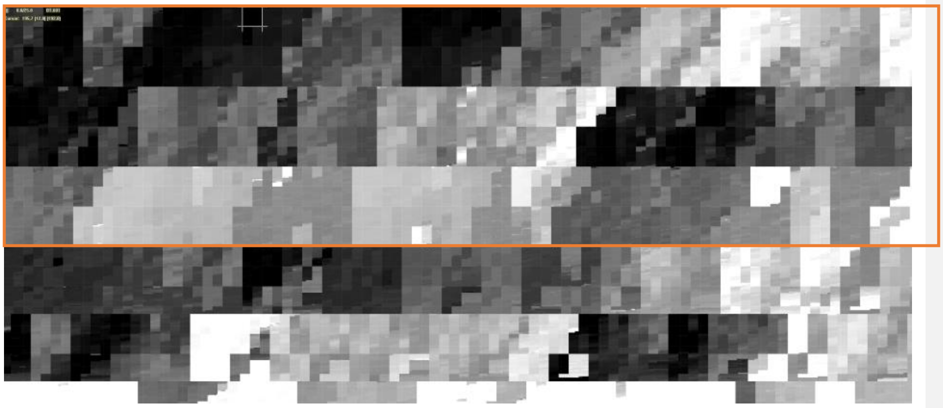


Fig. 1. Fragment of geometry atlas without any syntax changes.

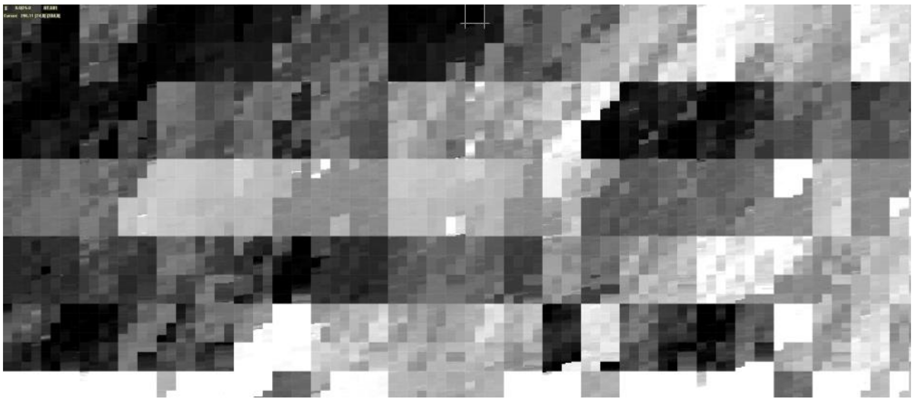


Fig. 2. Fragment of geometry atlas with proposed syntax changes.

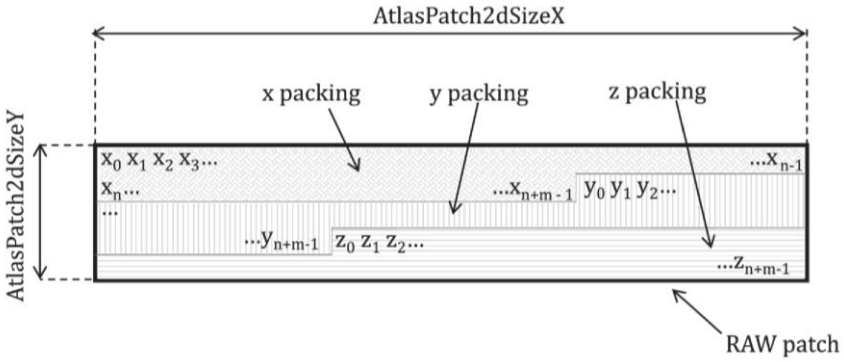


Fig. 3. Default, continuous data arrangement within raw patch. Fig. from 23090-5 spec.

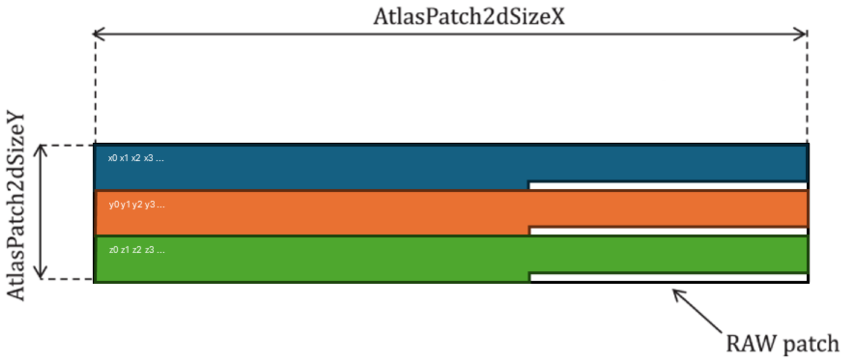


Fig. 4. Proposed planar data arrangement within raw patch.

1.3 Syntax and semantics

H.8.3.6.3.4 AAPS V-PCC GS extension syntax

aaps_vpcc_gs_extension( ) {	**Descriptor**
aaps_vpcc_gs_dequantization_present_flag	u(1)
aaps_vpcc_gs_adaptive_quantization_enable_flag	u(1)
aaps_vpcc_gs_sh_property_present_flag	u(1)
aaps_vpcc_gs_2d_tr_present_flag	u(1)
aaps_vpcc_gs_raw_patch_data_arrangement_flag	u(1)
aaps_vpcc_gs_reserved_zero_3bits	u(3)
if(aaps_vpcc_gs_dequantization_present_flag)	
gs_dequantization( )	
if(aaps_vpcc_gs_sh_property_present_flag)	
gs_spherical_harmonics_property( )	
if(aaps_vpcc_gs_raw_patch_data_arrangement_flag)	

gs_raw_patch_data_arrangement( )	
for(i = 0; i < 16; i++)	
ColTr[i] = 1	
if(aaps_vpcc_gs_2d_tr_present_flag)	
gs_2d_tr( )	
}	

H.8.4.6.3.4 AAPS V-PCC GS extension semantics

…

aaps_vpcc_gs_raw_patch_data_arrangement_flag equal to 1 specifies that gaussian splat raw patch data arrangement syntax structure shall be present in the current atlas adaptation parameter set. aaps_vpcc_gs_raw_patch_data_arrangement_flag equal to 0 specifies that gaussian splat raw patch data arrangement syntax structure shall not be present.

…

H.8.3.6.3.X Gaussian splat raw patch data arrangement syntax

gs_raw_patch_data_arrangement( ) {	**Descriptor**
gsrpda_data_arrangement_method	ue(v)
}	

H.8.4.6.3.X Gaussian splat raw patch data arrangement semantics

gsrpda_data_arrangement_method specifies the method of raw patch data arrangement as specified in Table H.X. When not present, gsrpda_data_arrangement_method is equal to 0.

Table H.X — Interpretation of gsrpda_data_arrangement_method

Value	Interpretation
0	Sequential arrangement (as defined in H.11.4)
1	Vertical planar arrangement
2-	Reserved

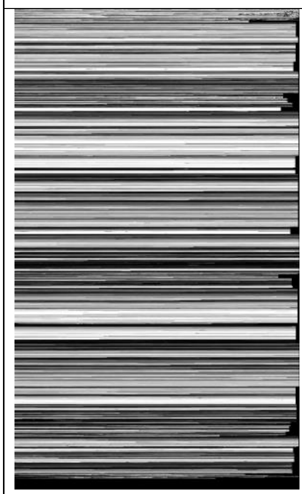
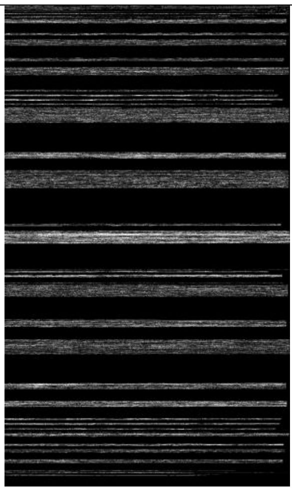
H.11.4. Reconstruction of RAW patches

```
if( gsrpda_data_arrangement_method == 0 ) {
   m = 0
   for( k = 0; k < 3; k++ ) {
      for( pointIdx = 0; pointIdx < AtlasPatchRawPoints[ pIdx ]; pointIdx++ ) {
         y = AtlasPatch2dPosY[ pIdx ] + ( m / AtlasPatch2dSizeX[ pIdx ] )
         x = AtlasPatch2dPosX[ pIdx ] + ( m % AtlasPatch2dSizeX[ pIdx ] )
         rawPos1D[ m ] = gFrame[ y ][ x ]
         m++
      }
   }
} else if( gsrpda_data_arrangement_method == 1 ) {
   compHeight = AtlasPatch2dSizeY[ pIdx ] / 3
   for( pointIdx = 0; pointIdx < AtlasPatchRawPoints[ pIdX ]; pointIdx++ ) {
      u = pointIdx % AtlasPatch2dSizeX[ pIdx ]
      v = pointIdx / AtlasPatch2dSizeX[ pIdx ]
```

```
      x = AtlasPatch2dPosX[ pIdx ] + u
      for( k = 0; k < 3; k++ ) {
         y = AtlasPatch2dPosY[ pIdx ] + v + ( k * compHeight )
         rawPos1D[ pointIdx + k * AtlasPatchRawPoints[ pIdx ] ] = gFrame[ y ][ x ]
      }
   }
}
for( pointIdx = 0; n < AtlasPatchRawPoints[ pIdx ]; n++ ) {
   recPcGeo[ pointCnt ][ 0 ] = rawPos1D[ pointIdx ]
      + AtlasPatch3dOffsetU[ pIdx ]
   recPcGeo[ pointCnt ][ 1 ] = rawPos1D[ pointIdx + AtlasPatchRawPoints[ pIdx ] ]
      + AtlasPatch3dOffsetV[ pIdx ]
   recPcGeo[ pointCnt ][ 2 ] = rawPos1D[ pointIdx + 2 * AtlasPatchRawPoints[ pIdx ] ]
      + AtlasPatch3dOffsetD[ pIdx ]
   y = AtlasPatch2dPosY[ pIdx ] + ( pointIdx / AtlasPatch2dSizeX[ pIdx ] )
   x = AtlasPatch2dPosX[ pIdx ] + ( pointIdx % AtlasPatch2dSizeX[ pIdx ] )
   for( attrIdx = 0; attrIdx < ai_attribute_count[ RecAtlasID ]; attrIdx++ ) {
      attrDim = ai_attribute_dimension_minus1[ RecAtlasID ][ attrIdx ] + 1
      for( compIdx = 0; compIdx < attrDim; compIdx++ )
         recPcAttr[ pointCnt ][ attrIdx ][ compIdx ] =
            aFrame[ attrIdx ][ compIdx ][ y ][ x ]
   }
   if( ai_attribute_count[ RecAtlasID ] > 0 )
      attrPresent[ pointCnt ] = 1
      pointToPixel[ pointCnt ][ 0 ] = -1
      pointToPixel[ pointCnt ][ 1 ] = -1
      pointToPatch[ pointCnt ] = pIdx
      pointCnt++
}
if( ai_attribute_count[ RecAtlasID ] > 0 )
      attrPresent[ pointCnt ] = 1
      pointToPixel[ pointCnt ][ 0 ] = -1
      pointToPixel[ pointCnt ][ 1 ] = -1
      pointToPatch[ pointCnt ] = pIdx
      pointCnt++
}
```

2 Experimental results

2.1 Visual atlas comparison

Geometry	Opacity	Scales
		

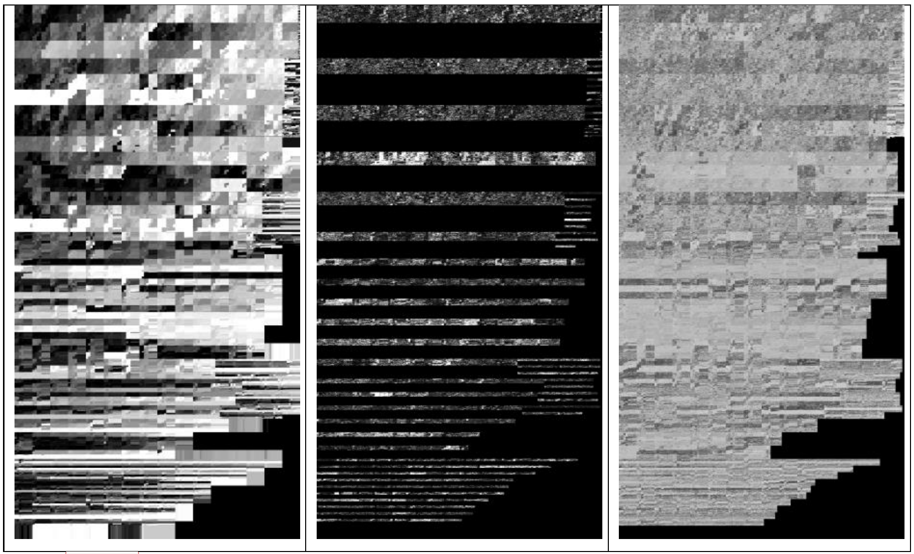


Fig. 5. Anchor (top) vs. proposed solution (bottom)

Commented [AD1]: To check

Remark: geometry atlas was additionally padded. If a 64x64 block contained any occupied pixel, the entire block was padded. Otherwise, it was zero. The padding was not performed on attributes.

2.2 Objective evaluation

2.2.1 Proposal (geo-only sorting) vs. Anchor

Sequence	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM	EncT Total	DecT Total	EncT Geometry	DecT Geometry
bartender_semitracked	-4.8%	-4.3%	-5.0%	67.0%	41.9%	46.4%	60.6%
cinema_semitracked	-5.8%	-6.2%	-5.7%	68.8%	44.1%	55.5%	74.1%
breakfast_semitracked	-4.8%	-4.8%	-6.0%	61.9%	44.0%	52.4%	54.8%
breakfast_untracked	-4.9%	-5.0%	-7.0%	86.0%	62.0%	65.9%	85.3%
breakdance_untracked	No overlap	No overlap	No overlap				
bartender_tracked	-5.1%	-6.0%	-6.2%	71.1%	47.7%	51.9%	65.6%
cinema_tracked	-6.7%	-6.8%	-6.8%	85.1%	63.9%	55.2%	92.6%
breakfast_tracked	-5.2%	-5.5%	-5.0%	62.0%	46.5%	57.5%	67.4%
trio	No overlap	No overlap	No overlap				
makeup	No overlap	No overlap	No overlap				
musical	No overlap	No overlap	No overlap				
Average	**-5.3%**	**-5.5%**	**-6.0%**	**71.7%**	**50.0%**	**55.0%**	**71.5%**
Average over mandatory V-PCC sequences	**-5.3%**	**-5.5%**	**-6.0%**	**71.7%**	**50.0%**	**55.0%**	**71.5%**

Sequence	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM	EncT Total	DecT Total	EncT Geometry	DecT Geometry
manwithfruit_tracked	-5.9%	-5.9%	-6.6%	33.7%	36.5%	35.5%	44.8%
lego_ferrari	-7.0%	-7.2%	-7.1%	40.7%	40.3%	41.5%	52.7%
lego_bugatti	-5.5%	-5.6%	-5.5%	47.9%	41.9%	45.8%	57.1%
cricket_player	-5.0%	-5.4%	-5.1%	88.6%	52.9%	70.8%	73.4%
plant	-4.6%	-5.6%	-6.2%	85.8%	61.7%	64.9%	90.5%
solo_tango_female	-6.7%	-8.6%	-6.4%	70.7%	49.0%	54.3%	68.8%
solo_tango_male	-7.1%	-7.6%	-6.1%	82.9%	70.1%	39.8%	96.8%
tango_duo	-3.4%	-3.6%	-2.2%	75.5%	49.5%	69.3%	67.4%
tennis_player	-5.0%	-5.5%	-4.6%	89.9%	61.5%	76.1%	94.8%
library	No overlap	No overlap	No overlap				
flowerdance	-4.2%	-4.8%	-4.7%	53.2%	39.0%	62.3%	67.5%
gymnast	-6.3%	-7.0%	-6.1%	44.7%	34.9%	54.7%	64.0%
Average	**-5.5%**	**-6.1%**	**-5.5%**	**64.9%**	**48.9%**	**55.9%**	**70.7%**
Average over mandatory V-PCC sequences	**-5.5%**	**-6.1%**	**-5.5%**	**64.9%**	**48.9%**	**55.9%**	**70.7%**

2.2.2 Proposal (geo-only sorting) vs. [m76156]

Sequence	Proposal BD-rate RGB-PSNR	Proposal BD-rate YUV-PSNR	Proposal BD-rate YUV-SSIM	[m76156] BD-rate RGB-PSNR	[m76156] BD-rate YUV-PSNR	[m76156] BD-rate YUV-SSIM
bartender_semitracked	-4.8%	-4.3%	-5.0%	-4.6%	-4.4%	-3.8%
cinema_semitracked	-5.8%	-6.2%	-5.7%	-3.8%	-3.9%	-4.0%
breakfast_semitracked	-4.8%	-4.8%	-6.0%	-3.2%	-3.1%	-3.7%
breakfast_untracked	-4.9%	-5.0%	-7.0%	-4.7%	-4.6%	-3.7%
breakdance_untracked	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
bartender_tracked	-5.1%	-6.0%	-6.2%	-3.9%	-4.7%	-4.1%
cinema_tracked	-6.7%	-6.8%	-6.8%	-4.3%	-4.3%	-3.9%
breakfast_tracked	-5.2%	-5.5%	-5.0%	-3.4%	-3.1%	-2.6%
trio	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
makeup	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
musical	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
Average	**-5.3%**	**-5.5%**	**-6.0%**	**-4.0%**	**-4.0%**	**-3.7%**
Average over mandatory V-PCC sequences	**-5.3%**	**-5.5%**	**-6.0%**	**-4.0%**	**-4.0%**	**-3.7%**

1F ([m76156])

Sequence	Proposal BD-rate RGB-PSNR	Proposal BD-rate YUV-PSNR	Proposal BD-rate YUV-SSIM	[m76156] BD-rate RGB-PSNR	[m76156] BD-rate YUV-PSNR	[m76156] BD-rate YUV-SSIM
manwithfruit_tracked	-5.9%	-5.9%	-6.6%	-3.9%	-3.9%	-4.7%
lego_ferrari	-7.0%	-7.2%	-7.1%	-4.2%	-4.2%	-4.3%
lego_bugatti	-5.5%	-5.6%	-5.5%	-3.3%	-3.3%	-3.4%
cricket_player	-5.0%	-5.4%	-5.1%	-3.9%	-3.9%	-3.3%
plant	-4.6%	-5.6%	-6.2%	-4.5%	-4.6%	-5.3%
solo_tango_female	-6.7%	-8.6%	-6.4%	-5.0%	-5.4%	-5.4%
solo_tango_male	-7.1%	-7.6%	-6.1%	-5.5%	-5.6%	-5.6%
tango_duo	-3.4%	-3.6%	-2.2%	-3.0%	-3.2%	-3.2%
tennis_player	-5.0%	-5.5%	-4.6%	-3.4%	-3.3%	-2.8%
library	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
flowerdance	-4.2%	-4.8%	-4.7%	-8.4%	-8.4%	-7.5%
gymnast	-6.3%	-7.0%	-6.1%	-10.6%	-10.6%	-10.3%
Average	**-5.5%**	**-6.1%**	**-5.5%**	**-5.1%**	**-5.1%**	**-5.1%**
Average over mandatory V-PCC sequences	**-5.5%**	**-6.1%**	**-5.5%**	**-5.1%**	**-5.1%**	**-5.1%**

2.2.3 Proposal vs. proposal but with Morton instead of [m76156]

Sequence	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM
bartender_semitracked	-4.8%	-4.3%	-5.0%	-4.9%	-4.9%	-4.8%
cinema_semitracked	-5.8%	-6.2%	-5.7%	-4.4%	-4.6%	-5.1%
breakfast_semitracked	-4.8%	-4.8%	-6.0%	-3.3%	-3.3%	-4.4%
breakfast_untracked	-4.9%	-5.0%	-7.0%	-3.2%	-3.3%	-6.0%
breakdance_untracked	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
bartender_tracked	-5.1%	-6.0%	-6.2%	-3.3%	-3.8%	-4.7%
cinema_tracked	-6.7%	-6.8%	-6.8%	-5.2%	-5.4%	-5.4%
breakfast_tracked	-5.2%	-5.5%	-5.0%	-3.5%	-3.4%	-3.5%
trio	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
makeup	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
musical	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
Average	**-5.3%**	**-5.5%**	**-6.0%**	**-4.0%**	**-4.1%**	**-4.9%**
Average over mandatory V-PCC sequences	**-5.3%**	**-5.5%**	**-6.0%**	**-4.0%**	**-4.1%**	**-4.9%**

2.2.4 Geometry sorting vs. Geometry + key attributes sorting

Sequence	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM
bartender_semitracked	-4.8%	-4.3%	-5.0%	-5.0%	-5.1%	-4.5%
cinema_semitracked	-5.8%	-6.2%	-5.7%	-5.9%	-6.5%	-5.8%
breakfast_semitracked	-4.8%	-4.8%	-6.0%	-4.2%	-4.4%	-4.4%
breakfast_untracked	-4.9%	-5.0%	-7.0%	-4.8%	-4.9%	-8.0%
breakdance_untracked	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
bartender_tracked	-5.1%	-6.0%	-6.2%	-3.8%	-4.8%	-5.1%
cinema_tracked	-6.7%	-6.8%	-6.8%	-6.2%	-6.2%	-6.7%
breakfast_tracked	-5.2%	-5.5%	-5.0%	-4.9%	-5.0%	-5.1%
trio	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
makeup	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
musical	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
Average	**-5.3%**	**-5.5%**	**-6.0%**	**-5.0%**	**-5.3%**	**-5.6%**
Average over mandatory V-PCC sequences	**-5.3%**	**-5.5%**	**-6.0%**	**-5.0%**	**-5.3%**	**-5.6%**

Sequence	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM	BD-rate RGB-PSNR	BD-rate YUV-PSNR	BD-rate YUV-SSIM
manwithfruit_tracked	-5.9%	-5.9%	-6.6%	-6.3%	-6.3%	-6.4%
lego_ferrari	-7.0%	-7.2%	-7.1%	No overlap	No overlap	No overlap
lego_bugatti	-5.5%	-5.6%	-5.5%	No overlap	No overlap	No overlap
cricket_player	-5.0%	-5.4%	-5.1%	No overlap	No overlap	No overlap
plant	-4.6%	-5.6%	-6.2%	No overlap	No overlap	No overlap
solo_tango_female	-6.7%	-8.6%	-6.4%	No overlap	No overlap	No overlap
solo_tango_male	-7.1%	-7.6%	-6.1%	No overlap	No overlap	No overlap
tango_duo	-3.4%	-3.6%	-2.2%	No overlap	No overlap	No overlap
tennis_player	-5.0%	-5.5%	-4.6%	No overlap	No overlap	No overlap
library	No overlap	No overlap	No overlap	No overlap	No overlap	No overlap
flowerdance	-4.2%	-4.8%	-4.7%	No overlap	No overlap	No overlap
gymnast	-6.3%	-7.0%	-6.1%	No overlap	No overlap	No overlap
Average	**-5.5%**	**-6.1%**	**-5.5%**	**-6.3%**	**-6.3%**	**-6.4%**
Average over mandatory V-PCC sequences	**-5.5%**	**-6.1%**	**-5.5%**	**-6.3%**	**-6.3%**	**-6.4%**

2.3 Computational time (encoding)

Content	Anchor	Morton Block Sort (geo-only)	Proposal (geo-only)	Proposal (geo, rot, sh0)
bartender_semitracked	1732.45	1583.07	1588.83	1555.91
cinema_semitracked	1422.50	1314.75	1310.85	1318.95
breakfast_semitracked	1116.80	957.14	945.92	955.58
breakfast_untracked	748.81	727.77	711.87	774.76
bartender_tracked	1144.01	1031.34	1020.01	644.90
cinema_tracked	711.15	696.34	688.24	727.35
breakfast_tracked	1095.49	936.13	923.68	931.02
manwithfruit_tracked	16.74	16.10	15.11	17.04

3 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).

4 Recommendations

We recommend (1) adopting the proposed syntax and semantics, (2) establishing the proposed method as a CTC default sorting strategy.

5 References

[m74633v3] A. Dziembowski, B. Szydełko, G. Lee, J.Y. Jeong, "[GSC][JEE 6.3-related] Block based FLAS for video-based GSC," ISO/IEC JTC 1/SC 29/WG 7 M76156, 10.2025.

[m76156] J.Y. Jeong, G. Lee, K.J. Oh, "[V-PCC][Amd1][CE2.3-related] Nearest-Neighbor Chain Sorting for Raw Points Patch," ISO/IEC JTC 1/SC 29/WG 7 M76156, 04.2026.