INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC 1/SC 29/WG 4
MPEG VIDEO CODING

ISO/IEC JTC 1/SC 29/WG 4 m 59442
April 2022, Online

Title: [MIV] Geometry Packing Implementation in TMIV for Frame Packed Video

Source: Jong-Beom Jeong, Soonbin Lee, Eun-Seok Ryu (Sungkyunkwan University (SKKU))

Abstract

This contribution provides an implementation of geometry packing in TMIV for a practical use of the frame packed video. The implementation packs geometry atlases to align them vertically in pixel domain, thereby the packed geometry atlases have the same height with the texture atlases. Then, the atlases are encoded as an 1×1 tile / subpicture using the motion-constrained tile set (MCTS) of HM or VVenC. One texture and geometry atlas bitstreams are merged in bitstream domain, and the V3C bitstream is modified to represent the merged video bitstreams. The implementation supports not only the proposed geometry packing method where the number of regions is automatically calculated based on the scaling information, but also the previous frame packing method (proposed by m56827) at both encoder- and decoder-side; the implementation adds a new syntax which signals only 1-bit. Besides, the merging process is optional according to the number of maximum decoder instances at the client-side.

1 Encoder-side Geometry Packing

Implemented components of geometry packing in TMIV is shown in Figure 1.

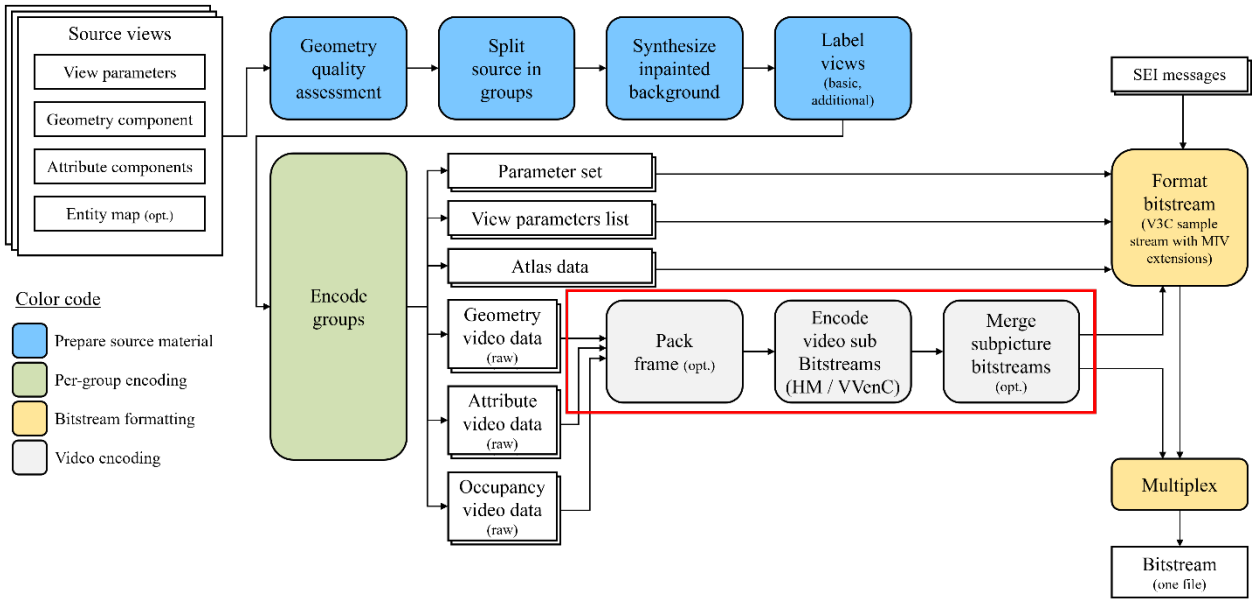

Figure 1. TMIV encoder with geometry packing and bitstream merging applied to the atlas components' video data

The proposed implementation divides the geometry (if present) and the occupancy (if existed) atlases into multiple regions vertically and align them, thereby the packed atlases have the same height with these of the texture atlases. Figure 2 shows

examples of the proposed geometry packing and bitstream merging methods. Figure at the middle shows the packed geometry atlases. Figure at the right represents a decoded picture of merged texture and geometry bitstreams, showing that the packed geometry atlas is attached right to the texture atlas. The reason why the geometry atlas is packed right to the texture atlas is that all the tiles / subpictures must be divisible by the CTU size (64 in HEVC and 128 in VVC), and it was proved that tiles / subpictures at the last column and row do not need to be divisible by the CTU size. Because most of the texture atlases which were generated using the test sequences in the common test conditions (CTC)[1] have width divisible by the CTU size and height non-divisible by the CTU size, the geometry atlas is packed right to the texture atlas. Otherwise, the merged bitstream cannot be decoded, or padding might be needed prior to the video encoding. More details will be explained in Subsection 1.1.

To represent the packed regions in the V3C bitstream, the proposed method uses the packing syntax[2]-[3] as the previous frame packing method did. Because the previous implementation proposed by m56827[4] only supports packing where the geometry (if present) and the occupancy (if present) atlases are attached under the texture atlas, a new option *geometryPacking* was added in TMIV implementation. When *geometryPacking* is true, *framePacking* is set to *false* in the proposed implementation. To reflect the proposed geometry packing in the syntax, a new syntax *vps_geometry_packing_enabled_flag* was added to distinguish the previous and the proposed methods, which will be explained in Subsection 1.2.
The proposed method is activated by **SKKU_GEOMETRY_PACKING** option which is defined in source/Common/include/TMIV/Common/Half.h, 38th line. This option is set to 1 in the attached source code. When this option is set to 0, the proposed method is deactivated.

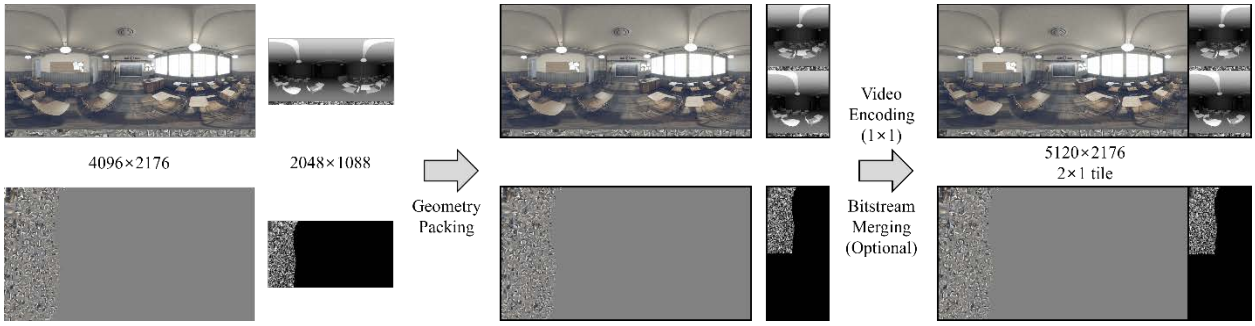

Figure 2. Geometry and bitstream merging for A17 – A (*ClassroomVideo*)

1.1 Video Encoding and Video Sub-bitstreams Merging

The generated atlases shown in the middle of Figure 2 are encoded by using the VVenC as explained in the CTC, where the atlases are considered as 1×1 tile. Figure 3 shows the encoding, subpicture merging processes with geometry packing, where two atlases are encoded as 1×1 tile. Since VVenC v0.3.1.0 which was defined in the CTC does not support the tiling option, VVenC v1.4.0 was used to the experiment. Further, the subpicture merging function proposed by Nokia[5] recommends to disable five options: joint chroma coding, ALF, CCALF, LMCS, and AMaxBT; thereby these five options were disabled in this experiment.

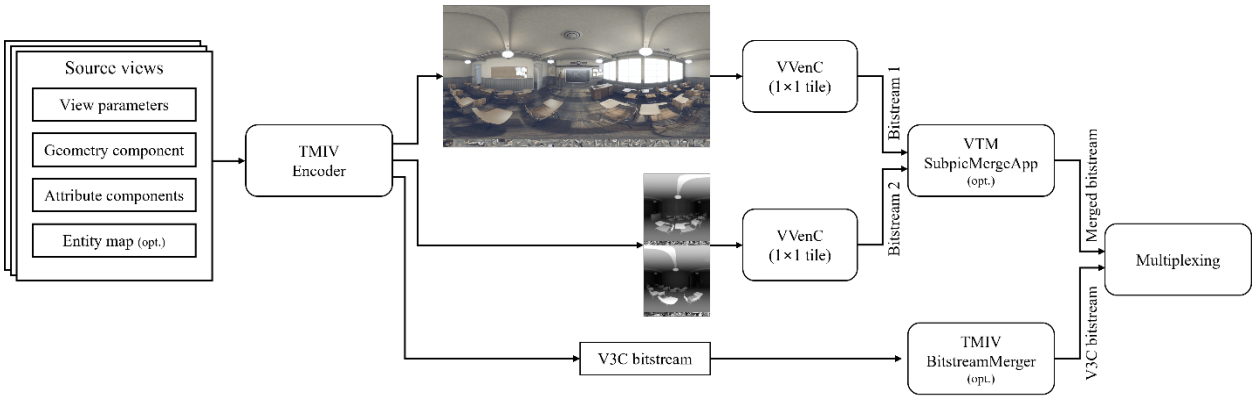

Figure 3. Encoding, subpicture merging of texture and geometry atlases with geometry packing

VVenC command line for texture atlas:

```
./vvencFFapp --preset slow -c cfg/A_2_VvenC_encode_tex_jacla_off.cfg -c cfg/GP17.cfg -i TMIV_GP17_A_tex_c00_4096x2176_yuv420p10le.yuv -s 4096x2176 -fr 30 --tiles 1x1 -q 26 -b files_encoded/ TMIV_GP17_A_tex_c00_4096x2176_yuv420p10le_26.266
```

VVenC command line for geometry atlas:

```
./vvencFFapp --preset slow -c cfg/A_2_VvenC_encode_geo_jacla_off.cfg -c cfg/GP17.cfg -i TMIV_GP17_A_geo_c00_1024x2176_yuv420p10le.yuv -s 1024x2176 -fr 30 --tiles 1x1 -q 7 -b files_encoded/ TMIV_GP17_A_geo_c00_1024x2176_yuv420p10le_7.266
```

The bitstream merging, which is optional, can be done by conducting two processes: (i) merging subpicture video sub-bitstreams, and (ii) modifying the V3C bitstream to reflect the changed region location during the subpicture merging. A new project *BitstreamMerger* was created in TMIV as shown in Figure 3 to conduct these two functions. However, the proposed implementation now only supports V3C bitstream modification. The subpicture merging is done by external application *SubpicMergeApp* in VTM v11.0. This is because the current TMIV includes HM, VVenC and VVdeC, therefore namespace problem occurs when implementing the bitstream parsing and re-encoding functions, thereby it requires more time to implement. Further, including the VTM to use the subpicture merging function cannot be done because the VTM does not have the namespace. The subpicture merging function will be implemented and introduced in the next meeting.
Note that the current *SubpicMergeApp* requires all the tiles / subpictures to be a multiple of the CTU size, which occurs errors during encoding for test sequences *Fan*, *Kitchen*, *Frog*, *Carpark*, and *Group* where the texture atlas size is 1920×4640 (the height is non-divisible by the CTU size). However, it was proved that the tiles / subpictures in the last column or row does not need to be divisible by the CTU size. Therefore, the VTM with modified *SubpicMergeApp* was attached in this proposal and used for the experiment, performing no errors during merging and decoding that was presented in 134th MPEG meeting[6] and VCIP 2021[7].

SubpicMergeApp command line for video sub-bitstream merging:

```
./SubpicMergeAppStatic -l cfg/TMIV_GPM17_A_pac_c00_5120x2176_yuv420p10le_QP1_merger.cfg -o files_merged/ TMIV_GPM17_A_pac_c00_5120x2176_yuv420p10le_QP1.266
```

SubpicMergeApp configuration file for video sub-bitstream merging:

```
4096 2176 0 0 files_encoded/TMIV_GP17_A_tex_c00_4096x2176_yuv420p10le_26.266
1024 2176 4096 0 files_encoded/TMIV_GP17_A_geo_c00_1024x2176_yuv420p10le_7.266
```

BitstreamMerger command line for V3C bitstream modification:

```
./BitstreamMerger -n 17 -s A -r R0 -c cfg/TMIV_GPM17_A_geometry_packing_subpicture_merge.json
```

BitstreamMerger configuration file V3C bitstream modification:

```
{
  “inputBitstreamPathFmt”: “A/TMIV_GP17_A.bit”,
  “inputDirectory”: “.”,
  “outputBitstreamPathFmt”: “A/TMIV_GPM17_A.bit”,
  “outputDirectory”: “.”
}
```

Mode	*vps_packing_information_ present_flag*	*vps_packed_video_ present_flag*	*vps_geometry_packing_ enabled_flag*
MIV main (A17)	0	0	0
Frame packing - m56827 (P17)	1	1	0
Proposed - geometry packing only (GP17)	1	0	1
Proposed - geometry packing + bitstream merging (GPM17)	1	1	1

Table 1. V3C parameter set syntax values for MIV main (A17), frame packing (P17), proposed method with geometry packing only (GP17), proposed method with geometry packing and bitstream merging (GPM17)

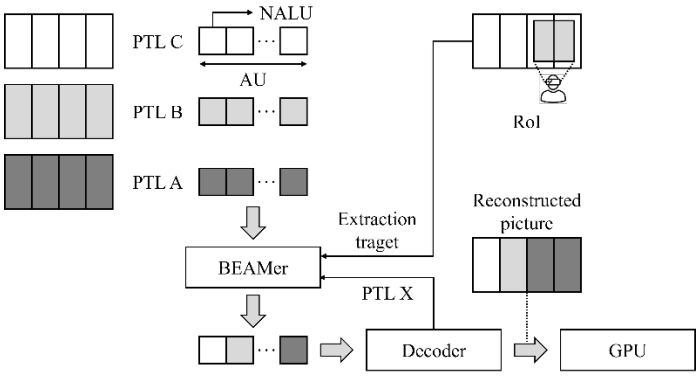

Figure 4. Bitstream extracting and merging operations in immersive media applications

1.2 V3C Syntax Setting

The proposed *BitstreamMerger* modifies *pin_region_top_left_x* following the merged video sub-bitstreams, and sets *vps_packed_video_present_flag* to 1. Table 1 shows the syntax values for MIV main (A17), frame packing (P17), the proposed method with geometry packing only (GP17), and the proposed method with geometry packing and bitstream merging (GPM17). In P17, GP17, and GPM17, *vps_packing_information_present_flag* is set to 1 because these three modes need the packing information. In P17, *vps_geometry_packing_enabled_flag* is set to 0. In GP17 which conducts only geometry packing and generates the same number of atlases with these of A17, *vps_packed_video_present_flag* is set to 0. In GPM17 which conducts geometry packing and bitstream merging, *vps_packed_video_present_flag* is set to 1 because frame unpacking is needed at the decoder side.

The reason why GP17 and GPM17 need to be distinguished is to use two or four decoder instances with the bitstream extractor and merger (BEAMer) introduced in ISO 23090-13: video decoding interfaces (VDI) for immersive media[8]. Figure 4 shows operations of the BEAMer in VDI, where NALUs are extracted and merged depending on the decoder's ability, which is simple and lightweight. Otherwise, when using P17, additional A17 atlases and following bitstreams need to be generated to support clients which cannot decode P17 bitstreams since P17 is a pixel-domain approach. Although P17 is simple, and not restricted to the MCTS, it may require additional server-side storage and computational complexity in terms of the VDI standard. When GP17 and GPM17 are not distinguished by *vps_packed_video_present_flag*, this problem may occur.

Consequently, by adding the *vps_geometry_packing_enabled_flag* which takes only 1-bit, the proposed and the pervious methods are distinguished and supported.

Figure 5 and 6 represents decoded atlases and syntax values for GP17 and GPM17. Because GP17 does not conduct video sub-bitstream merging, the texture and geometry atlases are separated, thereby *vps_packed_video_present_flag* is set to 0. In GPM17 where both geometry packing and video sub-bitstream merging are conducted, texture and geometry atlases are at the same decoded picture, therefore *pin_region_top_left_x* is set to 4096 for two geometry regions. *vps_packed_video_present_flag* is set to 1.

Syntax	Descriptor	Value (jth atlas)		
v3c_parameter_set(numBytesInV3CPayload) {				
vps_packing_information_present_flag[j]	u(1)	1		
vps_packed_video_present_flag[j]	**u(1)**	**0**		
vps_geometry_packing_enabled_flag[j]	**u(1)**	**1**		
}				
packing_information(j) {				
pin_codec_id[j]	u(8)	1 (HEVC_Main10), 3 (VVC_Main10)		
pin_regions_count_minus1[j]	ue(v)	2		
for(i = 0; i <= pin_regions_count_minus1[j]; i++) {				
pin_region_tile_id[j][i]	u(8)	0	0	0
pin_region_type_id_minus2[j][i]	u(2)	2 (for attribute)	1 (for geometry)	1 (for geometry)
pin_region_top_left_x[j][i]	u(16)	0	0	0
pin_region_top_left_y[j][i]	u(16)	0	0	1024
pin_region_width_minus1[j][i]	u(16)	4095	1023	1023
pin_region_height_minus1[j][i]	u(16)	2175	1087	1087
pin_region_unpack_top_left_x[j][i]	u(16)	0	0	1024
pin_region_unpack_top_left_y[j][i]	u(16)	0	0	0
pin_region_rotation_flag[j][i]	u(1)	0	0	0
}				
}				

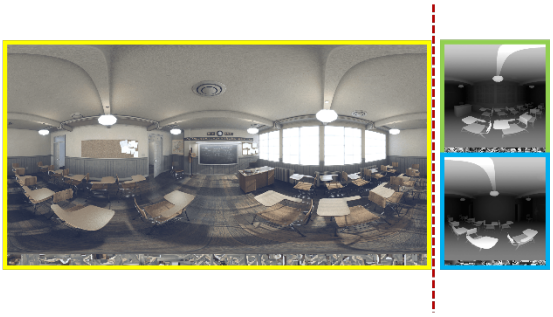

Figure 5. Packing information for GP17 – A for jth atlas

Syntax	Descriptor	Value (jth atlas)		
v3c_parameter_set(numBytesInV3CPayload) {				
vps_packing_information_present_flag[j]	u(1)	1		
vps_packed_video_present_flag[j]	**u(1)**	**1**		
vps_geometry_packing_enabled_flag[j]	**u(1)**	**1**		
}				
packing_information(j) {				
pin_codec_id[j]	u(8)	1 (HEVC_Main10), 3 (VVC_Main10)		
pin_regions_count_minus1[j]	uc(v)	2		
for(i = 0; i <= pin_regions_count_minus1[j]; i++) {				
pin_region_tile_id[j][i]	u(8)	0	0	0
pin_region_type_id_minus2[j][i]	u(2)	2 (for attribute)	1 (for geometry)	1 (for geometry)
pin_region_top_left_x[j][i]	u(16)	0	4096	4096
pin_region_top_left_y[j][i]	u(16)	0	0	1024
pin_region_width_minus1[j][i]	u(16)	4095	1023	1023
pin_region_height_minus1[j][i]	u(16)	2175	1087	1087
pin_region_unpack_top_left_x[j][i]	u(16)	0	0	1024
pin_region_unpack_top_left_y[j][i]	u(16)	0	0	0
pin_region_rotation_flag[j][i]	u(1)	0	0	0
}				
}				

Figure 6. Packing information for GPM17 – A for jth atlas

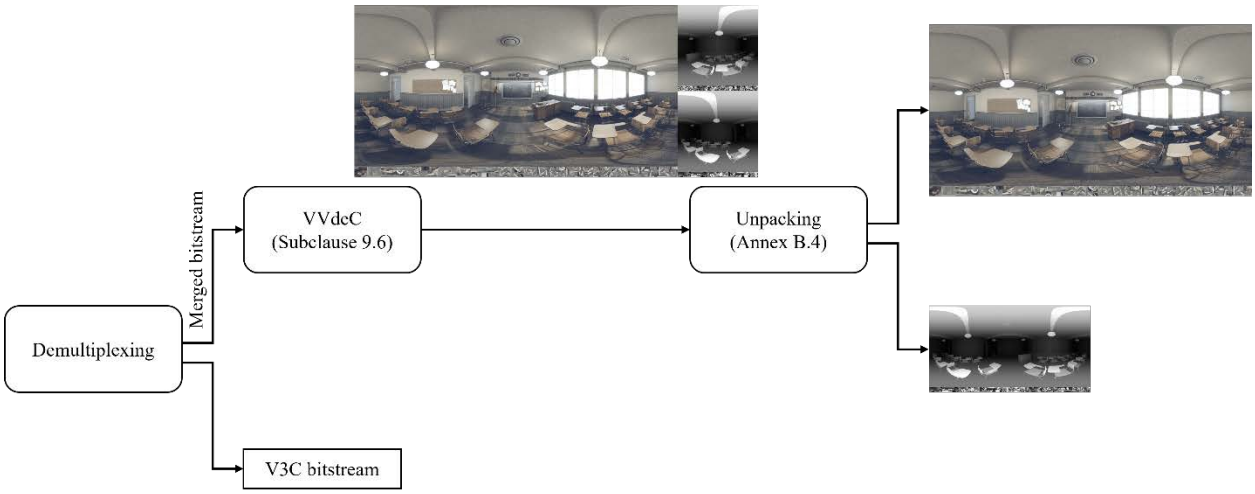

Figure 7. Decoding and unpacking of packed video

2 Decoder-side Geometry Packing

Figure 7 shows the decoding and unpacking processes of the packed video. At decoder side, the merged bitstream is decoded by the VVdeC, following 23090-5 subclause 9.6. Prior to the reconstruction (23090-5 clause 10), frame unpacking is conducted which is introduced in 23090-5 Annex B.4. The original atlas components are retrieved during the unpacking process. Then, intermediate view or pose trace rendering (reconstruction) is conducted.

3 V3C Syntax and Semantics for Geometry Packing

This section provides the modification in the specification for the proposed method with respect to FDIS of 23090-5 second edition[9] and WD of 23090-12 second edition[10].

8.3.4.1 General V3C parameter set syntax

v3c_parameter_set(numBytesInV3CPayload) {	**Desc -riptor**
profile_tier_level( )	
vps_v3c_parameter_set_id	u(4)
vps_reserved_zero_8bits	u(8)
vps_atlas_count_minus1	u(6)
for(k = 0; k < vps_atlas_count_minus1 + 1; k++) {	
vps_atlas_id[k]	u(6)
j = vps_atlas_id[k]	
vps_frame_width[j]	ue(v)
vps_frame_height[j]	ue(v)
vps_map_count_minus1[j]	u(4)
if(vps_map_count_minus1[j] > 0)	
vps_multiple_map_streams_present_flag[j]	u(1)
vps_map_absolute_coding_enabled_flag[j][0] = 1	
vps_map_predictor_index_diff[j][0] = 0	
for(i = 1; i <= vps_map_count_minus1[j]; i++) {	
if(vps_multiple_map_streams_present_flag[j])	
vps_map_absolute_coding_enabled_flag[j][i]	u(1)
else	
vps_map_absolute_coding_enabled_flag[j][i] = 1	

if(vps_map_absolute_coding_enabled_flag[j][i] == 0) {	
vps_map_predictor_index_diff[j][i]	ue(v)
}	
}	
vps_auxiliary_video_present_flag[j]	u(1)
vps_occupancy_video_present_flag[j]	u(1)
vps_geometry_video_present_flag[j]	u(1)
vps_attribute_video_present_flag[j]	u(1)
if(vps_occupancy_video_present_flag[j])	
occupancy_information(j)	
if(vps_geometry_video_present_flag[j])	
geometry_information(j)	
if(vps_attribute_video_present_flag[j])	
attribute_information(j)	
}	
vps_extension_present_flag	u(1)
if(vps_extension_present_flag) {	
vps_packing_information_present_flag	u(1)
vps_miv_extension_present_flag	u(1)
vps_extension_6bits	u(6)
}	
if(vps_packing_information_present_flag) {	
for(k = 0; k <= vps_atlas_count_minus1; k++) {	
j = vps_atlas_id[k]	
vps_packed_video_present_flag[j]	
vps_geometry_packing_enabled_flag[j]	
if(vps_packed_video_present_flag[j] || vps_geometry_packing_enabled_flag[j])	
packing_information(j)	
}	
}	
if(vps_miv_extension_present_flag)	
vps_miv_extension( ) /*Specified in ISO/IEC 23090-12[1] */	
if(vps_extension_6bits) {	
vps_extension_length_minus1	ue(v)
for(j = 0; j < vps_extension_length_minus1 + 1; j++) {	
vps_extension_data_byte	u(8)
}	
}	
byte_alignment( )	
}	

8.4.4.1 General V3C parameter set semantics (23090-5)

vps_geometry_packing_enabled_flag[j] equal to 0 indicates that the atlas with atlas ID j does not have packed geometry video data associated with it. vps_geometry_packing_enabled_flag[j] equal to 1 indicates that the atlas with atlas ID j shall have packed geometry video data associated with it. vps_geometry_packing_enabled_flag[j] equal to 1 and vps_packed_video_present_flag[j] equal to 1 indicates that the atlas with atlas ID j shall have packed video data associated with it while the geometry packing is applied.
When vps_geometry_packing_enabled_flag[j] is not present, it is inferred to be equal to 0.

[1] Under preparation. Stage at time of publication: ISO/IEC CD 23090-12:2020

It is a requirement of bitstream conformance if vps_geometry_packing_enabled_flag[j] is equal to 1 for an atlas with atlas ID j, at least one of the pin_occupancy_present_flag[j], pin_geometry_present_flag[j], or pin_attribute_present_flag[j] is equal to 1.

Table A-1: Allowable values of syntax element values for the MIV toolset profile components (23090-12)

	Profile name			
Syntax element	**MIV Main**	**MIV Extended**		**MIV Geometry Absent**
			Restricted Geometry	
vuh_unit_type	V3C_VPS, V3C_AD, V3C_GVD, V3C_AVD, or V3C_CAD	V3C_VPS, V3C_AD, V3C_OVD, V3C_GVD, V3C_AVD, V3C_PVD, or V3C CAD	V3C_VPS, V3C_AD, V3C_AVD, V3C_PVD, or V3C CAD	V3C_VPS, V3C_AD, V3C_AVD, or V3C CAD
ptl_profile_toolset_idc	64	65		66
ptl_profile_reconstruction_idc	255	255		255
ptc_restricted_geometry_flag	N/A	0	1	N/A
vps_miv_extension_present_flag	1	1	1	1
vps_packing_information_present_flag	0	0, 1	0, 1	0
vps_map_count_minus1[atlasID]	0	0	0	0
vps_auxiliary_video_present_flag[atlasID]	0	0	0	0
vps_occupancy_video_present_flag[atlasID]	0	0, 1	0	0
vps_geometry_video_present_flag[atlasID]	1	0, 1	0	0
vme_embedded_occupancy_enabled_flag	1	0, 1	0	0
oi_occupancy_MSB_align_flag[atlasID]	0	0	0	0
gi_geometry_MSB_align_flag[atlasID]	0	0	0	0
ai_attribute_count[atlasID]	0, 1	0, 1, 2	2	1
ai_attribute_type_id[atlasID][attrIdx]	ATTR_TEXTURE	ATTR_TEXTURE, ATTR_TRANSPARENCY	ATTR_TEXTURE, ATTR_TRANSPARENCY	ATTR_TEXTURE
ai_attribute_dimension_minus1[atlasID] [attrTextureIdx]	2	2	2	2
ai_attribute_dimension_minus1[atlasID] [attrTransparencyIdx]	N/A	0	0	N/A
ai_attribute_dimension_partitions_minus1[atlasID] [attrIdx]	0	0	0	0
ai_attribute_MSB_align_flag[atlasID][attrIdx]	0	0	0	0

pin_attribute_count[atlasID]	N/A	0, 1, 2	2	N/A
pin_attribute_type_id[atlasID][attrIdx]	N/A	ATTR_TEXTURE, ATTR_TRANSPARENCY	ATTR_TEXTURE, ATTR_TRANSPARENCY	N/A
pin_attribute_dimension_minus1[atlasID] [attrTextureIdx]	N/A	2	2	N/A
pin_attribute_dimension_minus1[atlasID] [attrTransparencyIdx]	N/A	0	0	N/A
pin_attribute_dimension_partitions_minus1[atlasID] [attrIdx]	N/A	0	0	N/A
pin_attribute_MSB_align_flag[atlasID][attrIdx]	N/A	0	0	N/A
asps_long_term_ref_atlas_frames_flag	0	0	0	0
asps_pixel_deinterleaving_enabled_flag	0	0	0	0
asps_patch_precedence_order_flag	0	0	0	0
asps_raw_patch_enabled_flag	0	0	0	0
asps_eom_patch_enabled_flag	0	0	0	0
asps_plr_enabled_flag	0	0	0	0
asps_vpcc_extension_present_flag	0	0	0	0
asme_patch_constant_depth_flag	0	0, 1	1	0
vps_geometry_video_present_flag[atlasID] || pin_geometry_present_flag[atlasID] || asme_patch_constant_depth_flag	N/A	1	1	N/A
vps_packed_video_present_flag[atlasID]	0	0, 1	0, 1	0
vps_geometry_packing_enabled_flag[atlasID]	0	0, 1	0, 1	0
afps_lod_mode_enabled_flag	0	0	0	0
afps_raw_3d_offset_bit_count_explicit_mode_flag	0	0	0	0
afti_single_tile_in_atlas_frame_flag	1	0, 1	0, 1	0, 1
ath_type	I_TILE	I_TILE	I_TILE	I_TILE
atdu_patch_mode[tileID][patchIdx]	I_INTRA	I_INTRA	I_INTRA	I_INTRA
asps_atlas_sequence_parameter_set_id	0..63, inclusive	0..63, inclusive	0..63, inclusive	0..63, inclusive
afps_atlas_frame_parameter_set_id	0..63, inclusive	0..63, inclusive	0..63, inclusive	0..63, inclusive
afps_atlas_sequence_parameter_set_id	0..63, inclusive	0..63, inclusive	0..63, inclusive	0..63, inclusive
aaps_atlas_adaptation_parameter_set_id	0..63, inclusive	0..63, inclusive	0..63, inclusive	0..63, inclusive
ath_atlas_frame_parameter_set_id	0..63, inclusive	0..63, inclusive	0..63, inclusive	0..63, inclusive

ath_atlas_adaptation_parameter_set_id	0..63, inclusive	0..63, inclusive	0..63, inclusive	0..63, inclusive

4 Conclusion and Recommendation

This contribution proposed the geometry packing and bitstream merging method to accomplish the following goals: (i) practical use of the frame packing (e.g. delta QP allocation to geometry atlases), and (ii) support two or four number of decoder instances using BEAMer, including V3C bitstream modification.

According to the advantages of the proposed method, it is proposed to integrate the text proposed in Section 2 to FDIS of 23090-5 Ed. 2 and to WD of 23090-12 Ed. 2, and the implementation in the recent version of TMIV software with its test cases and update the TMIV document accordingly.

5 References

[1] "Common Test Conditions for MPEG Immersive Video", Joel Jung, Bart Kroon, ISO/IEC JTC1/SC29/WG4 output document n00169, January 2022, online meeting.

[2] "[MIV] Editorial input on packed video", Lukasz Kondrad, ISO/IEC JTC1/SC29/WG5 input document m55343, October 2020, online meeting.

[3] "CE-1.3: frame packed video sub-bitstream type", Lukasz Kondrad, Vinod Kumar Malamal Vadakital, Lauri Ilola, ISO/IEC JTC1/SC29/WG11 input document m54274, June 2020, online meeting.

[4] "Frame Packing Implementation in TMIV", Basel Salahieh, Gousemoodhin Nadaf, Jill Boyce, ISO/IEC JTC1/SC29/WG4 input document m56827, April 2020, online meeting.

[5] "AHG3/AHG12: Subpicture merging software", Antti Hallapuro, Miska M. Hannuksela, ISO/IEC JTC1/SC29/WG11 input document m54168, June 2020, online meeting.

[6] "[MIV] Extraction and Merging on Frame Packed Video", Jong-Beom Jeong, Soonbin Lee, Eun-Seok Ryu, ISO/IEC JTC1/SC29/WG4 input document m56591, April 2021, online meeting.

[7] Jong-Beom Jeong, Soonbin Lee, Eun-Seok Ryu, "DWS-BEAM: Decoder-Wise Subpicture Bitstream Extracting and Merging for MPEG Immersive Video", IEEE Visual Communications and Image Processing 2021 (VCIP2021), Dec. 5-8, 2021.

[8] "Text of ISO/IEC DIS 23090-13 Video Decoding Interface for Immersive Media", ISO/IEC JTC1/SC29/WG3 output document n00484, January 2022, online meeting.

[9] "Text of ISO/IEC DIS 23090-5 Visual Volumetric Video-based Coding and Video-based Point Cloud Compression 2nd Edition", ISO/IEC JTC1/SC29/WG7 output document n00188, July 2021, online meeting.

[10] "Preliminary WD1 of ISO/IEC 23090-12 MPEG Immersive video Ed. 2", ISO/IEC JTC1/SC29/WG4 output document n00176, January 2022, online meeting.