**INTERNATIONAL ORGANISATION FOR STANDARDISATION**

**ORGANISATION INTERNATIONALE DE NORMALISATION**

**ISO/IEC JTC1/SC29/WG11**

**CODING OF MOVING PICTURES AND AUDIO**

**ISO/IEC JTC1/SC29/WG11 MPEG2019/m52429**

**January 2020, Brussels, Belgium**

|  |
| --- |
| **Source Interdigital** |
| **Status Input contribution** |
| **Title Auxiliary Patches** |
| **Author Renaud Doré, Bertrand Chupeau** |
|  |

# Context

The current MIV specification determines the way how to carry information of a volumetric scene based on Texture and Depth patches. In fact, it is based on a surfacic description of the scene.

As a matter of fact, the scene is composed of open forms because they have been captured by a set of cameras typically located at one given side of the scene. The backs of the object or characters are not present, and it does not matter as the viewing zone should necessary match with the capture zone.

The contribution points out that

* there is very little to do to achieve a description of the scene to be conveyed by MIV syntax with a full definition of all surfaces composing the real scene.
* The modification of the syntax is very minor to allow this.
* This extended scene description may be useful for purposes which are definitively out of the scope of MIV specification

# Extended description of the scene with auxiliary patches

It is possible to describe the non visible parts of the scene from the original set of cameras by adding auxiliary patches. These patches can be generated:

* By other cameras pointing from the back to the center scene, possibly with active depth sensors as well.
* By closing open forms algorithmically
* ..

These patches are as dynamic as the T+D patches to be rendered and can integrate naturally the patch flow with a minor penalty in bitrate, providing they are defined in lower resolution and/or for example do not put any information for the texture.

It is however necessary to signal to the MIV decoder that these patches are NOT for the renderer:

* These auxiliary patches would typically overlap with the regular patches
* They could be in lower resolution as said before, possibly without any texture
* They could severely degrade the rendering if taken as true regular patches

# Syntax modification proposal

The signaling of the auxiliary nature of the patches can be done

1. at the patch level,
2. or (better) using the entityId introduced at MPEG128.

## At entity level

Auxiliary entities can be identified in the iv\_sequence\_parameters by modifying the IV sequence parameters syntax as shown in Table 3:

|  |  |
| --- | --- |
| iv\_sequence\_params( ) { | **Descriptor** |
| ivs\_profile\_tier\_level( ) |  |
| **depth\_params\_num\_bits\_minus8** | u(4) |
| view\_params\_list( ) |  |
| **depth\_low\_quality\_flag** | u(1) |
| **num\_groups\_minus1** | ue(v) |
| **max\_entities\_minus1** | ue(v) |
| **auxiliary\_flag** | u(1) |
| if ( auxiliary\_flag and max\_entities\_minus1 != 0) {  for (e==0; e <= max\_entities\_minus1; e ++) { |  |
| **auxiliary\_entity\_flag** [ e ] | u(1) |
| } |  |
| **viewing\_space\_present\_flag** | u(1) |
| if(viewing\_space\_present\_flag) | u(1) |
| viewing\_space( ) |  |
| **ivs\_sp\_extension\_present\_flag** | u(1) |
| if( ivs\_sp\_extension\_present\_flag ) { |  |
| while( more\_data\_in\_payload( ) ) |  |
| **ivs\_sp\_extension\_data\_flag** | u(1) |
| rbsp\_trailing\_bits( ) |  |
| } |  |

1. Table 3: Modification to IV sequence parameters in MIV [1] (clause 7.3.1)

**auxiliary\_flag** equal to 1 indicates that the auxiliary description is present for each entity structure.

**auxiliary\_entity\_flag**[ e ] equal to 1 indicates that the patches related to the entity e are not for the viewport rendering.

## Alternative syntax modification at patch level

Alternatively, the auxiliary patch can be signalled at patch level by modifying the atlas parameters syntax as shown in Table 4.

|  |  |
| --- | --- |
| atlas\_params( a ) { | **Descriptor** |
| **num\_patches\_minus1**[ a ] | u(16) |
| **atlas\_width\_minus1**[ a ] | u(16) |
| **atlas\_height\_minus1**[ a ] | u(16) |
| **depth\_occ\_params\_present\_flag**[ a ] | u(1) |
| auxiliary\_flag[ a ] | u(1) |
| for( p = 0; p <= num\_patches\_minus1; p++ ) { |  |
| **view\_id**[ a ][ p ] | u(v) |
| if( max\_entities\_minus1 > 0 ) |  |
| **entity\_id**[ a ][ p ] | u(v) |
| if( auxiliary\_flag[ a ] ) |  |
| **auxiliary\_patch\_flag**[ a ][ p] | u(1) |
| **patch\_width\_in\_view**[ a ][ p ] | u(v) |
| **patch\_height\_in\_view**[ a ][ p ] | u(v) |
| **patch\_pos\_in\_atlas\_x**[ a ][ p ] | u(v) |
| **patch\_pos\_in\_atlas\_y**[ a ][ p ] | u(v) |
| **patch\_pos\_in\_view\_x**[ a ][ p ] | u(v) |
| **patch\_pos\_in\_view\_y**[ a ][ p ] | u(v) |
| **patch\_rotation**[ a ][ p ] | u(3) |
| if( depth\_occ\_params\_present\_flag[ a ] ) |  |
| depth\_occupancy( a, p ) |  |
| } |  |
| } |  |

1. Table 4: Modification to atlas parameters syntax in MIV WD3 [1] (clause 7.3.4.1)

**auxiliary\_flag** equal to 1 indicates that the auxiliary description is present for each patch structure.

**auxiliary\_patch\_flag**[ a ][ p ] equal to 1 indicates that the patch p of the atlas a is not for the viewport rendering.

# Possible Use Cases using auxiliary out of MIV scope

The use cases are definitively out of scope of MIV specification and are more related to system architecture and scene compositing, but for information we would like to mention a couple of possible use cases:

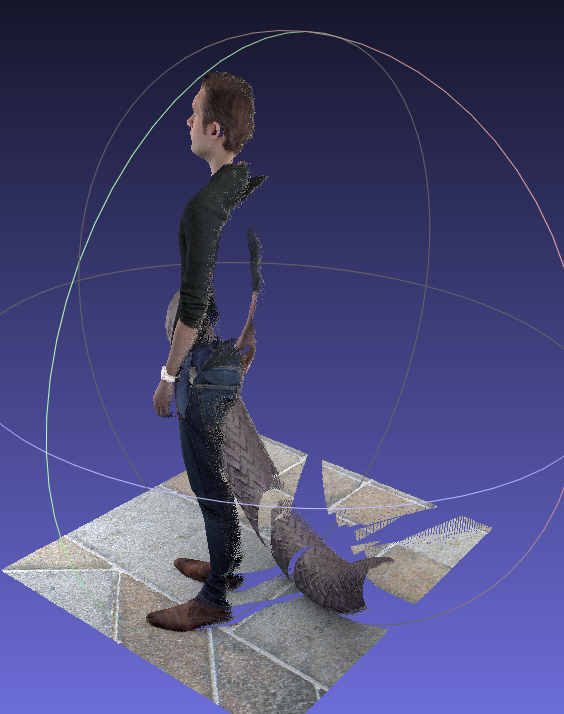
* Relighting of a MIV scene
* Definition of invisible and dynamic volume like haptic, interaction, etc.

An illustration of relighting issue is given below. The figure below shows a piece of TechnicolorMuseum scene which has been extracted, as seen from the camera rig:



Figure 1: piece of TechnicolorMuseum content seen from rig position

Of course, the same piece seen from a completely different point of view shows open forms:



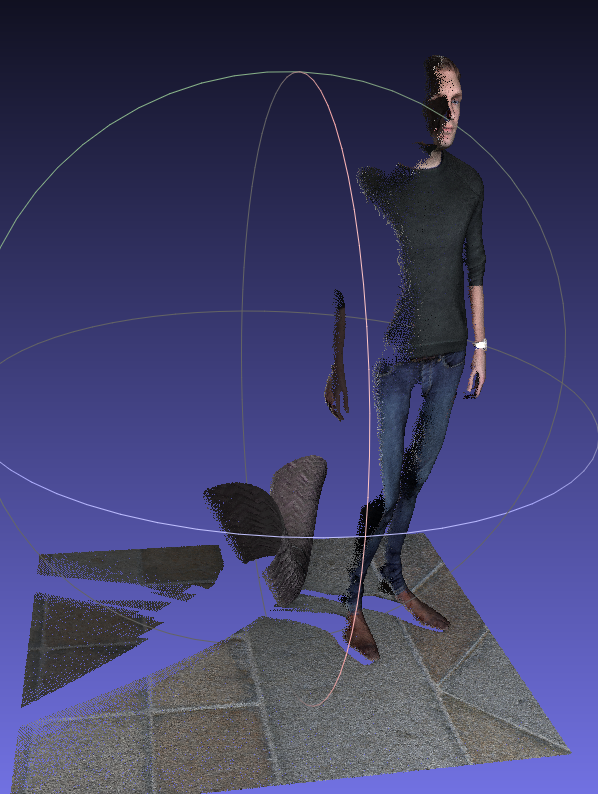


Figure 2: same content piece seen outside the camera rig

As a result, if an application would like to re-lighten this scene, it would lead to inconsistent results, as one can see in the figure below where a very simple sun-like relighting of remeshed scene has been tried:



Figure 3: example of scene relighting with inconsistent shadow on floor

In this case, complementing the MIV scene description with forms in the back, even very roughly, would be sufficient to make a realistic re-enlighting. It would consist in putting a few depth cameras shooting from the back, typically at a resolution way below the capture cameras.

# Conclusion and recommendation

We recommend the group to add in the next specification the possibility in the MIV specification to convey patches which are not meant for direct rendering. The change of the syntax is fully negligible, when articulated with the notion of entity Id which has been introduced at MPEG128, and even at patch level.

As this information complementing the description of the scene geometry is intimately linked to the geometry of the dynamic content to be streamed according to the MIV spec, we don’t see how to make that possible without this tiny hook in the MIV spec.

We do not argue in favor of use cases outside the scope of MIV specification, but only want to make such that MIV is open enough to enable use cases which are not necessary clear today.

# References

1. J. Jung, B. Kroon, J. Boyce, Common Test Conditions for Immersive Video, ISO/IEC JTC1/SC29/WG11 MPEG/N18789, October. 2019, Geneva, Switzerland.