**3GPP TSG SA WG 4 Meeting #124 S4-230786**

**Berlin, 22 - 26 May 2023**

**Source: Tencent**

**pCR Title: [MeCAR] Update on available Visualization Space (8.5)**

**Draft Spec: Permanent document**

**Agenda item: 9.5 (MeCAR)**

**Document for: Approval**

# 1. Introduction

This contribution proposes updates to clause 8.5.3 visual space, addressing the raised questions in the NOTE:

NOTE: It is expected to analyze for further study:

* the implement ability of the potential solutions together with the OpenXR runtime for the acquisition of the available space. OpenXR as an example provides this in an extension https://registry.khronos.org/OpenXR/specs/1.0/html/xrspec.html#XR\_FB\_scene, using xrGetSpaceBoundary2DFB or xrGetSpaceBoundingBox3DFB
* the persistence of the available space (static vs dynamic including how frequent the information is refreshed and used)
* the accuracy of the space, i.e., units used and the mapping of 3D scene and objects to these units.

#  Answers to the Note’s comments

The openXR facebook scene extension allows to define the boundary space. But the boundary space is only defined as cubical and the semantic of it is not defined either.

The OpenXR XR\_FB\_scene defines the meter as the unit for each dimension and uses floating point for this purpose:

“This structure is used for component values that may be fractional (floating-point). If used to represent physical distances, values must be in meters. The width, height, and depth values must be non-negative.”

# Proposed update to the PD text

# 8.5 Available visualization space

8.5.1 General

A visual 3D scene may be required to be rendered within a specific visualization area to ensure that visual objects will fit into the available space surrounding the user. Primarily addressing a need for AR games, the proposed approach is meant to be applicable for any type of service such as AR streaming, AR conversational and other applications.

An AR experience is achieved by the integration of visual objects into the user environment. Depending on the available space around the user, the AR experience may be unreal by the perceived collision between the virtual 3D objects and the real environment.

8.5.2 Possible solution

In order to appropriately render an AR scene into the real environment, the system needs to know the available space in which visual objects can be rendered. The visual area capability defines the available volume in size and coordinates in space inside which the 3D objects can be easily rendered.

Such a rendering space may be defined with a simple shape (e.g., cylinder, cube, sphere) for which the size and coordinates can be easily signaled.

A more complex and precise space may also be defined thanks to the device capabilities to capture accurately the shape of objects (e.g., with LiDAR sensors). In this case, the available rendering space may either be calculated in the device itself of by the server if the LiDAR scene is uploaded to the server. But this second case may cause some privacy issues. For many applications, defining the free space with a simple shape is adequate.

8.5.3 Potential implementation

Assuming the visualization space is represented by a simple shape, the UE needs to signal the type and dimensions of the shape.

In the example of a cylinder as illustrated below the radius, the height (if not infinite) and the angle (if only part of the cylinder is covered, as shown in green below) is signaled.



**Figure 23 - Example of cylinder-shape visualization space**

In the example of a cube, only the distance from the center of the cube to any face of it (half the distance of an edge) is signaled, assuming that the cube is centered on the viewer’s viewpoint. An extension of cube signaling is when the 3 sizes of a cubic space (depth, width and height) are signaled.



**Figure 24 - Example of cube-shape visualization space**

In the case of a sphere, only its radius is signaled, also assuming that the center if the sphere is the user’s viewpoint.



**Figure 25 - Example of sphere-shape visualization space**

For the above examples here is the required signaling:

**Table 10 - Visualization space shape properties**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Shape | Cylinder | Partial cylinder | Cuboid | Sphere |
| Measures | * Height (if not infinite)
* Radius
 | * Height (if not infinite)
* Radius
* Angle
 | * Side 1
* Side 2
* Side 3
 | * Radius
 |

The parameters Height, Side X and Radius are defined in meter, with floating point representation.

The parameter for Angle is defined in degree, ranges from 0 to 360, with floating point representation.

At this point, only Cuboid visualization space is supported by this specification.

## 8.5.3.1 Persistency

The capturing the available visual space by the device is outside of the document. A device may use a manual input for defining the available visual space at the beginning of session or use an assisted method to define that space. In these cases, the available visual space is static.

A device may use its capturing sensors and derives the available visual space automatically. In this case, it may possibly update the visual space dynamically during the session. Since this document does not define the process of capturing the visual space, this specification only defines whether the available visualization space is static (no changing) or dynamic in IF-2 by Application, and if it is dynamic the frequency of update and/or an event-driven (action) update.

8.5.3.2 Implementation in OpenXR

The openXR XR\_FB\_scene extension allows to define the boundary room and also boundary space and objects in the space:

1. xrGetSpaceBoundingBox3DFB provides the defined rectangular cube XrRect3DfFB (by defining the offset XrOffset3DfFB and the extend XrExtent3DfFB, in x,y,z dimensions).

2. xrGetSpaceSemanticLabelsFB may provide a way to describe the semantic meaning of an space entity.

This extension provides a mechanism to retrieve a rectangular bounding box that possibly can be marked as the available visualization space. However, at least one semantic keyword for such indication needs to be defined by this document to mark the space as “available visualization space”.

8.5.4 Use of available space information by the server

With the knowledge of the available rendering space, a server can ensure that the virtual objects fit into it. What the server decides to do remains out of scope of 3GPP and is the responsibility of the service provider. The following is just an illustration of the possible content adaptations at the server side:

* The server may decide to downscale the 3D scene so that all objects fit into the rendering volume.
* The server may decide to clip the scene and only send the virtual objects fully present in the rendering volume.
* The server may decide to deny the service due to incompatibility between the immersive experience and the available space around the user.
* In case of multiple users in different spaces, the server may decide to downscale the 3D scene for some or upscale for others to create a symmetric experience for all users.

As an example in Figure 23, the following scene with 3 objects has one object (orange cylinder) within the rectangle visualization space one object partially inside the cube (yellow rectangle) and one object out of the rendering volume (green cylinder).



**Figure 26 - Example scene before adaptation**

|  |  |
| --- | --- |
| Une image contenant graphique  Description générée automatiquementFigure 27 - Example scene after clipping | Une image contenant graphique  Description générée automatiquementFigure 28 - Example scene after downscaling |

# Proposal

We propose:

1. To adopt the above text in PD

2. To include only cuboid format in the MeCAR specification, aligned with OpenXR FB extension

3. To define a semantic keyword “available\_visualization\_space” for Available Visualization Space (AVS), to be used to mark the space

4. To add AVS type (static and dynamic) and frequency of update/event-driven action to IF-2.