3GPP TSG-SA WG4 Meeting #131-bis-eS4-250525

Online, 11 – 17 April, 2025 *revision of S4aV250029*

**Source: InterDigital Canada**

**Title: [FS\_ARSpatial] Pseudo-CR on Diminished Reality Spatial Computing Functions**

**Spec: 3GPP TR 26.819 v0.3.0**

**Agenda item: 9.8**

**Document for: Agreement**

**1. Introduction**

The Study on Spatial Computing for AR Services (FS\_ARSpatial) was approved during SA#104 meeting. The objectives of the study include identifying where spatial computing functions run and which media, metadata, and description formats are used for exchange between these elements based on the architecture defined in the TS 26.506, notably in split processing scenarios.

This contribution introduces a new spatial computing function, real-object removal, that is necessary to enable diminished reality capabilities in an AR-based service.

**2. Reason for Change**

One of the main objectives of the study is to “Study how spatial computing functions such as relocalization, mapping, and semantic perception are realized and identify the necessary set of spatial mapping information.”. This contribution addresses this objective by documenting a real-object removal spatial computing function that supports diminished reality applications.

**3. Proposal**

It is proposed to agree the following changes to 3GPP TR 26.819 v0.3.0.

\* \* \* First Change \* \* \* \*

### 4.2.1 General

XR spatial computing encompasses a set of functions which process sensor data to generate information about the world 3D space surrounding the AR user. These include functions such as relocalization (establishing the position of users and objects within that space), anchoring, 3D Model construction, segmentation and labeling (semantic perception), light extraction, and real object removal are achieved through a combination of advanced sensors, cameras, and algorithms that enable devices to understand and interact with the three-dimensional space around them.

This requires accurately localizing the AR device worn by the end-user in relation to a spatial coordinate system of the real-world space. Vision-based localization systems reconstruct a sparse spatial mapping of the real-world space in parallel (e.g., SLAM). Beyond the localization within a world coordinate system, which is usually based on a sparse spatial map, dense spatial mapping of objects is also essential in order to place 3D objects on real surfaces and provides the ability to occlude objects behind surfaces, do physics-based interactions based on surface properties, provide navigation functions, or provide a visualization of the surface.

For the purpose of understanding and perceiving the scene semantically, machine-learning and/or artificial intelligence may be used to provide context for the observed scene.

The output of spatial computing is spatial mapping information that is organized in a data structure called the *XR Spatial Description* for storing and exchanging the information. Some spatial computing functions may also take an XR spatial description and may result in updates to the XR spatial description. Spatial computing functions typically include data exchange and require a network architecture.

An AR device may provide sensor data to the spatial computing function to create or update the spatial mapping information. The device may also access the spatial computing function to retrieve different spatial mapping information depending on the needs of the XR application.

The spatial computing functions can run locally on the AR device or can be executed remotely on the cloud or on the edge in a spatial computing server accessed through dedicated interfaces as detailed in TR 26.298 [2].

The main functions provided by a spatial computing service are given in Figure 4.2.1-1and explained in the following subclauses.

A diagram of a model

AI-generated content may be incorrect.

**Figure 4.2.1-1: Spatial computing functions**

\* \* \* Next Change \* \* \* \*

### 4.2.10 Real object removal

The removal of real objects in an XR application is also known as diminished reality (DR) which removes reality components from the viewer's perspective. Unwanted elements are recognized and replaced by other image elements, creating a plausible and consistent overall impression for the viewer.

Diminished reality has the potential to address a challenge in AR where virtual objects cannot be placed due to real-world obstacles. The IKEA Kreativ app, Figure 4.2.10-1, is an example of diminished reality.

|  |  |
| --- | --- |
| A room with a chair and boxes  AI-generated content may be incorrect. | A room with a wood floor and windows  AI-generated content may be incorrect. |

Figure 4.2.10-1: Example of object removal in IKEA Kreativ.

To remove a real object from the viewer’s perspective, the object needs to be hidden and replaced by what is behind it (background image).

NOTE: Due to hardware and latency constraints of optical and camera passthrough devices, diminished reality is realized through overlays, which may be prone to distortion with slight pose changes.

To hide real objects in a scene, several tasks must be performed:

- Capture the scene.

- Segment the scene (2D or 3D).

- Select objects to hide.

- Apply masks to selected objects.

- Fill the masked areas with an inpainting process.

Using deep learning, it is possible to identify and select objects in an image. The segmented 3D model of a real scene can also be used to select target objects. A mask is then positioned over the selected areas.

To hide selected objects, the masked areas are filled with background information. This process is known as inpainting.

In-painting diminished reality is a technique in which parts of a scene are reconstructed or "painted over" using information in the source image about textures and colors around the object or the area to be reduced. There are two common methods for removing targets from videos or images. The first method fills in the missing regions by using readily available background images or essential information captured beforehand. The second method uses the information around the target or the similarity of textures to artificially produce the filling content (usually using some deep learning technique), an approach also known as *image inpainting*.

To hide real objects in a scene, the following input data can be used:

- Sensor data

- Images captured by AR Device

- pose of AR Device

- Depth map (image or texture)

- Mesh captured by AR Device

When rendering the DR scene, two methods can be applied to hide objects and the shadows casted by the object to be removed:

- A set of images containing transparent areas and non-transparent areas is used. Non-transparent areas hide real objects and their shadows.

- 3D objects (meshes and textures) are used to hide reals objects and their shadows.

Image with transparency and 3D objects are possible outputs of the “Real Object Removal” function.

Hiding an object does not erase its shadow. To improve the XR experience, shadow removal could also be done. The aim of shadow removal is to restore the lighting conditions, color, and texture in the shadow regions. Recent shadow removal techniques predominantly adopt deep learning-based approaches.

Shadow removal methods typically involve two key steps: shadow detection and shadow removal, based on the detected shadow mask. Prior-based shadow removal methods are generally based on physical models and adopt the prior information such as gradient, illumination, and regions to remove shadows. The emergence of deep learning has greatly improved the performance of shadow removal. In all recent publications, deep learning is used to remove shadows.

To remove a shadow, several tasks must be performed:

- Capture the scene

- Detect shadows (associated to hidden objects)

- Apply masks to detected areas

- Fill the masked area with an inpainting process

When rendering the DR scene, the same methods used to hide objects can also be applied.

Sensor data and output are the same as before.

\* \* \* Next Change \* \* \* \*

### 4.2.10 Summary of spatial description formats

This section describes the common output data formats for the spatial computing functions defined in clause 4.2. Table 1 provides a list of the output data for each spatial computing function and the corresponding format.

Table – Output data of spatial computing functions.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function** | **Data** | **Description** | **Format Example** | **Possible Representation** |
| World Tracking | Feature Map | Features to compute a pose in XR Space. | Array of features |  |
| Feature | 3D point associated with a descriptor, the descriptor format depends on the feature type. | SIFT (Scale Invariant Feature Transform) | Typically, a vector of 128 floats. |
| ORB – Oriented FAST Rotated BRIEF (Binary descriptor) | Typically, a 32-bit descriptor. |
| BRIEF – Binary Robust Independent Elementary Features) | Typically, 32-bit descriptor. |
| FREAK – Fast Retina Keypoint | Typically, 64-bit descriptor. |
| SURF – Speeded-Up Robust Features | Typically, vector of 64-bit or 128-bit floats. |
| Relocalization | Pose | Pose of the AR device in XR Space | Position: 3D vector  Orientation: quaternion | Position: 3 floating point values.  Oriantation: 4 floating point values |
| XR\_space\_id | An identifier for the XR Space used as a reference. | String |  |
| Anchoring | Id | An identifier of the new trackable (when the anchoring function is invoked by a producer). | String |  |
| Pose | Pose of the anchor in relation to the device according to a trackable (when the anchoring function is invoked by a consumer). | Position: 3D vector  Orientation: quaternion | Position: 3 floating point values.  Oriantation: 4 floating point values |
| 3D Model Construction | Model | 3D model(s) of surrounding space. | Non-segmented model (e.g., point cloud, mesh with/without attributes) | OBJ, STL, PLY [41], FBX, glTF |
| Segmented model | glTF, USD |
| Collider Generation | Colliders | Set of colliders (not combined with associated objects). | Array of colliders |  |
| Collider | A collider object. | Primitive (e.g., sphere with radius) |  |
| Mesh | OBJ, STL, FBX |
| Colliders | Part of description of the 3D Model (when combined with associated objects). | Hierarchical node graph | MPEG-I SD, USD [46] |
| Segmentation and Labeling | Objects | A list of 3D objects resulting from the segmentation and labeling function. | Array or a hierarchical node graph. |  |
| Object | A 3D object in the captured world. |  |  |
| Mesh | Segmented object. | Mesh |  |
| Label | Label of segmented object. | String |  |
| Light Extraction | Lights | Set of extracted lights | Array |  |
| Light | A description of a light source that includes a set of parameters that depend on the type of the light. Possible types include: point, directional, area, spot, texture-based, or image-based light. | Object | For point light, the parameters include: pose, intensity, color, and range. |
| Real Object Removal | Objects | A set of textured 3D objects. | Mesh + Texture | OBJ, STL, PLY, FBX, glTF |
| Images | Images with transparency. |  | PNG, TIFF, TGA, BMP |

\* \* \* End of Changes \* \* \* \*