**3GPP TSG SA WG4#116e S4-211369**

**E-meeting, 10th – 19th November 2021**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.1* | | | | | | | | |
| **PSEUDO CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  | 26.998 | **CR** |  | **rev** |  | **Current version:** | 1.0.3 |  |
|  | | | | | | | | |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* | | | | | | | | |
|  | | | | | | | | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME |  | Radio Access Network |  | Core Network |  |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | |
| ***Title:*** | [FS\_5GSTAR] On Spatial Computing | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** | Qualcomm Incorporated | | | | | | | | | |
| ***Source to TSG:*** |  | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** | FS\_5GSTAR | | | | |  | ***Date:*** | | | 2021-11-02 |
|  |  | | | |  | |  | | |  |
| ***Category:*** | C |  | | | | | ***Release:*** | | | Rel-17 |
|  | *Use one of the following categories:* ***F*** *(correction)* ***A*** *(mirror corresponding to a change in an earlier release)* ***B*** *(addition of feature),* ***C*** *(functional modification of feature)* ***D*** *(editorial modification)*  Detailed explanations of the above categories can be found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | | | | | | | | *Use one of the following releases: Rel-8 (Release 8) Rel-9 (Release 9) Rel-10 (Release 10) Rel-11 (Release 11) … Rel-15 (Release 15) Rel-16 (Release 16) Rel-17 (Release 17) Rel-18 (Release 18)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **Y** | **N** |  | | | |  | | |
| ***Other specs*** | |  |  | Other core specifications | | | | TS/TR ... CR ... | | |
| ***affected:*** | |  |  | Test specifications | | | | TS/TR ... CR ... | | |
| ***(show related CRs)*** | |  |  | O&M Specifications | | | | TS/TR ... CR ... | | |
|  | |  | | | | | | | | |
| ***Other comments:*** | |  | | | | | | | | |
|  | |  | | | | | | | | |
| ***This CR's revision history:*** | |  | | | | | | | | |

**===== CHANGE =====**

## 4.3 Basic Processes in an AR Session

### Introduction

Two asynchronous sessions in parallel may occur

1. AR media session for rendering
2. Spatial compute for spatial mapping

### 4.3.1 AR Media Session

In this clause, we provide basic processes and generic workflow description for setting up AR Media sessions for media is accessed over the network. This generic basic process may be extended to address specific applications and use cases. The call flow as shown in Figure 4.3.1-1 aligns with the STAR/EDGAR architecture and serves as a baseline for defining use-case specific call flows.

Figure 4.3.1-1: Basic workflow for AR media sessions

A description of the steps of the general workflow is provided as follows:

1. The application contacts the application provider to fetch the entry point for the content. The acquisition of the entry point may be performed in different ways and is considered out of scope. An entry point may for example be a URL to a scene description.

2. The application initializes the Scene Manager using the acquired entry point.

3. The Scene Manager retrieves the scene description from the scene provider based on the entry point information. It then establishes a scene session with the scene provider.

4. The Scene Manager parses the entry point and creates the immersive scene.

5. The Scene Manager requests the creation of a new AR/MR session from the AR Runtime.

6. The AR Runtime creates a new AR/MR session and performs registration with the local environment.

7. The Scene Manager will inform the MAF about its QoS and compute needs

8. The MAF will request the Media Delivery Functions, such as AF, in the network to allocate the requested resources.

9. For each component or group of components of an object/node in the scene:

a. the Scene Manager triggers the MAF to fetch the related media

b. the MAF creates a dedicated media pipeline to process the input.

c. the MAF establishes a transport session for each component of the media object.

10. The application starts the media fetching and rendering loop

a. the MAF may receive updates to the scene description from the scene provider.

b. the MAF passes the scene update to the Scene Manager.

c. the Scene Manager updates the current scene.

d. The Scene Manager acquires the latest pose information and the user’s actions

e. The Scene Manager shares that information with the AR/MR application on the server

f. For each object:

i. The media pipeline fetches the media data. It could be static, segmented, or real-time media streams

ii. The media pipeline processes the media and makes it available in buffers

g. For each object to be rendered:

i. The Scene Manager gets processed media data from the media pipeline buffers

ii. The Scene Manager reconstructs and renders the object

h. The Scene Manager passes the rendered frame to the AR/MR Runtime for display on the user’s HMD.

### 4.3.2 AR Spatial Mapping Session

In this clause, we provide basic processes and generic workflow description for setting up an AR session using the AR Runtime and possibly the network in order to provide the AR device with a continuous mapping of the scene to the real-world spaces. This requires registering with the surrounding spaces requiring spatial coordinate systems for precisely positioning and orienting 3D media objects at meaningful places in the world. Beyond the registration within a world coordinate system, additionally spatial mapping of objects is essential in order to place 3D objects on real surfaces, but also provides the ability to occlude objects behind surfaces, doing physics-based interactions based on surface properties, providing navigation functions or providing a visualization of the surface. Thirdly, for the purpose of understanding and perceiving the scene semantically, machine-learning and/or artificial intelligence may be used to provide context of the observed scene. The above processes may be carried out on the AR device or may be partially or completely be delegated to the edge or cloud.

For this purpose, it may be needed to exchange data that is supporting registration, spatial mapping and semantical perception information collected by sensors (e.g., cameras, microphones, etc,) to spatial compute servers on the edge or cloud, i.e. the information needs to be exchanged. Secondly, spatial data may be provided to the AR run time in order to support registration and spatial mapping. This includes for example:

* Spatial maps, e.g., sparse or dense point clouds of the environment, spatial meshes of key surfaces in the environment, possibly with associated propertiesSpatial anchors, possibly associated with 3D objects (also referred to as trackables), in order to permit world-space experiences

A basic call flow and processes for providing spatial information to AR run times is shown in Figure 4.3.2-1 with focus on the data exchange between the spatial compute functionality on the device and the network.

tbd

Figure 4.3.2-1: Basic workflow for AR spatial mapping and perception sessions

A description of the steps of the general workflow is provided as follows:

1. The application initiates the run-time.

2. The AR Runtime creates a new AR/MR session to perform spatial computing functionalities including registration, spatial mapping and semantical perception.

3. The AR Runtime, possibly in coordination with the app, will inform the MAF about spatial compute needs and possibly other 5G System functionalities such as edge compute, QoS, etc for both uplink streaming of sensor data, as well as for accessing spatial information on the cloud.

4. The MAF will request the Media Delivery Functions, such as AF, in the network to allocate the requested resources.

5. For each sensor in the uplink to be sent to the network:

a. the AR Runtime triggers the MAF to push the related sensor information.

b. the MAF creates media pipelines to process the sensor output.

c. the MAF establishes a transport session for each relevant sensor.

6. For spatial mapping information:

a. the AR Runtime triggers the MAF to create a session for accessing spatial mapping information.

b. the MAF creates a media pipeline to access spatial mapping information.

c. the MAF establishes a transport session for spatial mapping information.

7. The application starts the spatial processing loop to exchange sensor data and spatial mapping information with the network

a. the AR Runtime fetches spatial mapping information from the spatial compute server

b. the AR Runtime provides sensor data or spatial mapping information updates to the spatial compute server

c. the spatial compute server processes the sensor data and may provide spatial mapping information updates to the AR Runtime.

**===== CHANGE =====**

### 4.4.3 User pose information

User’s position may be represented as a geolocation with longitude and latitude. The position may also be represented as a point in a scene. The scene may be represented as a bounding box on a geometry which represents user’s real environment. When an AR/MR device reports the user position to obtain a split render of the immersive media from a server, the device calculating the user pose should be either a geolocation, a point in a scene or a point in a user’s geometry. Depending on the representation, the server should be aware of the underlying scene or the geometry. A device should update whenever there is any change in the scene or the geometry through user interaction (e.g., rotating a scene by hand gesture) and/or SLAM (e.g., finer modelling of surrounding environment).

A direction may be represented with a rotation matrix, or roll, pitch, and yaw. The direction is relative to a scene/geometry and the scene/geometry has an origin and default direction of the three axes.

The device representing a user’s pose moves continuously, and if the device is worn on the user’s head, it is assumed that he or she frequently turns their head around. A set of position and direction information is only meaningful at a certain moment in time. Since the device reports the user pose at around a frequency of 1 KHz, any pose information should include a timestamp to specify when it was measured or created. A pose corrector (e.g., ATW and LSR) in a server may estimate the user’s future pose, whilst a pose corrector in a device may correct the received rendered image to fit the latest user pose.

- Formats for user pose

A position in Cartesian coordinate system may be represented by either X, Y and Z or by a translation matrix. A direction may be represented by a rotation matrix or by quaternions.

OpenXR describes a possible format for user pose [4]. It consists of 4 quaternions for orientation and 3 vectors for position. Timestamp is represented by a 64 bit monotonically increasing nano-second-based integer.

#### 4.4.3.2 Camera Paramaters

Immersive media is captured by camera(s). The camera parameters such as focal length, principal points, calibration parameters and the pose of the camera all contribute in understanding the relevance between points in the volumetric scene and pixels in the captured image. Photogrammetry is the technology used to construct immersive media from a continuous capturing of images. Depth sensor-based cameras may be used to capture immersive media from one capturing of the volumetric scene

- Formats for camera information

Camera intrinsic parameters may be represented by a camera matrix. Extrinsic parameters may be represented by a transform matrix.

**===== CHANGE =====**

### 4.4.7 Spatial Mapping Information

#### 4.4.7.1 Overview

According to clause 4.3.2, it may be needed to exchange data that is supporting registration, spatial mapping and semantical perception information collected by sensors (e.g., cameras, microphones, etc,) to spatial compute servers on the edge or cloud, i.e. the information needs to be exchanged. Details are provided in clause 4.4.7.2.

Secondly, spatial data may be provided by the spatial compute server to the AR run time in order to support registration and spatial mapping. This includes for example:

* Spatial maps, e.g., sparse or dense point clouds of the environment, spatial meshes of key surfaces in the environment, possibly with associated properties
* Spatial anchors, possibly associated with 3D objects (also referred to as trackables), in order to permit world-space experiences

#### 4.4.7.2 Camera and sensor information

2D :

LiDAR.

Depth

3D ToF

3D

Sound

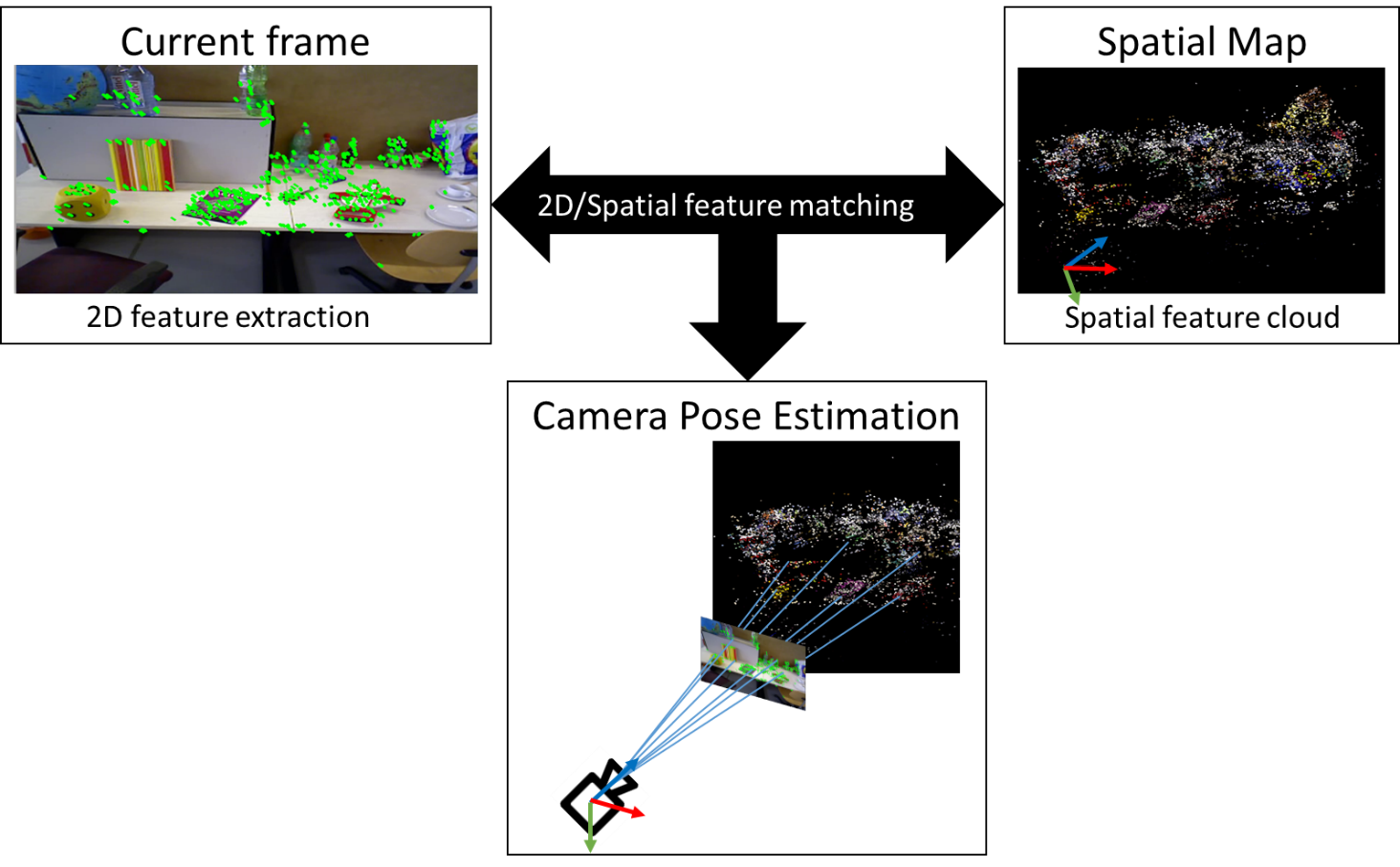
#### 4.4.7.3 Spatial Maps

Describe details, surfaces, meshes? etc.

Visual features are characteristics of a real world element that can be searched, recognized or tracked in frames captured from an AR device visual sensor as it moves in a real environment, using Simultaneous Localization And Mapping approach (SLAM). They are the overlapping points that are recognizable in multiple images of the real environment. Visual features are extracted from frames from a single moving camera or multiple cameras in SLAM systems. A 3D Map, sparse or dense point cloud, of the real world can be generated from keyframes (i.e. frames of interest) and their matched visual features. The keyframes must be attached to camera information defined in 4.4.3.2 to triangulate 3D points correctly from multiple cameras. This mapping process can be performed either at runtime or offline. A 3D map is then used at runtime to relocalize and thus register the AR device by matching the features extracted from the current frame with the ones stored in the previously built 3D map. The mapping approach is one of well-known keyframe-based SLAM techniques [X].

Add to references [X] <https://arxiv.org/abs/1607.00470>

Therefore, a spatial map consists of spatial visual features (e.g. 3D points associated with their descriptor such as SIFT [XX], SURF [XXX], ORB [XXXX]) and additional information to match them with visual features extracted at runtime from the frames captured by sensors (2D or 3D depending on sensor capabilities). Note that the 2D-3D feature matching (e.g for RGB or B&W cameras) is widely used to estimate the pose of the sensor (using a Perspective-n-Points algorithm), and thus of the AR device. Note that 3D-3D feature matching can also be implemented for depth sensors.



For this reason a spatial map consists at least of:

* A spatial feature cloud, e.g. 3D points (Vector of 3 floats) with their associated descriptors such as SIFT, SURF, ORB. These descriptors are generally vectors of numbers (e.g vector of 128 floats for SIFT, vector of 64 floats for SURF, vector of 32 integers for ORB). Note that other features such as 3D segments can be also used.

But additionnaly, to speed-up the 2D-3D matching process, a spatial map usually includes:

* Information required for a keyframe retrieval. For example, a keyframe retrieval can use Bag-Of-visual-Words (BoW) model. In this case, the information consists of the vocabulary of the BoW model and corresponding descriptors for each keyframe (vector of occurrence counts of a vocabulary in the keyframe). Depending on the visual descriptor used, the vocabulary size is usualy a few dozen Mb, and this vocabulary can be reused for any spatial map using the same descriptor. Note that neural networks can be used instead of a BoW model.
* The 2D features of each keyframes (e.g. 2D points with their associated descriptors such as SURF, SIFT, ORB represented by a vector of numbers). The number of features extracted per keyframe can vary between 200 and 1000.
* The matches between 2D features of keyframes and 3D features of the spatial feature cloud.

Thanks to this additional information, instead of comparing all descriptors of 2D visual features extracted from the current frame with all spatial feature descriptors, resulting in a very high complexity computation, the vision based localization system can:

* Retrieve the closest keyframe of the spatial map from the current frame (e.g. using the BoW model),
* Match the 2D features of the current frame with the 2D features of the retrieved keyframe,
* Match the 2D features of the current frame with the spatial features of the spatial feature cloud (based on the matches between 2D features of the keyframes and spatial features of the spatial feature cloud).

Add to reference [XX] <https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>

Add to reference [XXX] <https://people.ee.ethz.ch/~surf/eccv06.pdf>

Add to reference [XXXX] <https://ieeexplore.ieee.org/document/6126544>

#### 4.4.7.4 Spatial Anchors and Trackables

AR objects can be positioned in reference to the real world (e.g., placing a vase on a table) using spatial anchors. A spatial anchor provides a fixed position and orientation in the real world based on a common frame of reference that can be used by multiple AR devices. Spatial anchors should refer to trackables for accurate positioning relative to the physical space. Spatial anchors can also be used alone (not referring to trackable) if global coordinates are used. In this case, the anchors are treated as global anchors without trackable as they have global coordinates which positions can be determined.

Trackables areelements of the real world of which features (visual or non-visual) are available and/or could be extracted. A 3D Map trackable, for instance, may define a full environment composed of a floor walls, furnitures in the real world consisting of several 3D points with visual features. However, there are other types of trackables as well. For example:

* A controller with LEDs that can be tracked by an AR headset’s vision sensor. The feature in this case is the constellation of LEDs.
* A fiducial marker that is detected as a black and white pattern by an AR device vision sensor. The feature in this case is the black and white pattern.
* Hands visible through an AR headset’s vision sensor. The feature is a learnt model for hands.

All of the above examples give a position of the trackable in reference to the position of the sensor (generally embedded in the AR headset). A spatial description data structure describing the spatial organisation of the real world using anchors, trackables, camera parameters and visual features can be used for exchanging spatial data and updates between AR Runtime and spatial compute server.

Way forward after first offline:

* Thomas would work on 4.4.7.2. other volunteers please shout
* Jerome would provide some updates on 4.4.7.3.
* Offline Friday, 4PM CEST – Discuss how to combine with the cognitive and spatial computing call flows of 409 and 475.