3GPP TSG- Meeting # *S4-250518r02*

**, , -**

**Source: InterDigital Communications**

**Title: [SR\_IMS] Pseudo-CR on Delay adaptation in UE**

**Spec: 3GPP TS 26.567 v1.0.1**

**Agenda item: 10.5**

**Document for: Discussion and agreement**

**1. Introduction**

Draft TS 26.567 specifies Processing Delay adaptation in clause 7.3.2 and defined message format in Annex A. Processing Delay adaptation may be used with the Split Adaptation defined in clause 7.3.1. In this case the SR-DCMTSI client in terminal renders some objects of the scene locally, and the Processing Delay Adaptation may be used on the locally rendered objects.

The MF may need to change the Level-of-Details (LoD) of objects of the scene rendered by the SR-DCMTSI client in terminal. However, there is no mean to synchronize the state of the objects of the scene rendered by the SR-DCMTSI client in terminal with the MF, and the SR-DCMTSI client in terminal needs to know the different LoDs available for locally rendered objects.

Processing Delay adaptation is essential to maintain the quality of the user experience at an acceptable level when the network conditions are changing. This delay adaptation relies on different scene representations to modify the delay of various processing tasks involved in the rendering loop such as:

- the update of the XR scene graph representation by considering animation, user interactivity, physics simulation,

- the culling techniques (e.g., camera frustrum, back-face) process with respect to the user current point of view to exclude objects or portions of the XR scene that are not visible to the user,

- the rendering of the XR scene with the proper lighting conditions.

An object may have several representations (i.e., several variants of its geometry, texture, media). These object representations are often called object Levels-of-Details (LoDs) to be selected by the XR application for optimizing the trade-off between rendering quality and processing delay during the runtime. This trade-off between object rendering quality and processing delay is mentioned in the introduction section of the Level of Detail for 3D Graphics book [1] p.30:

*“when rendering, use a less detailed representation for small, distant, or unimportant portions of the scene. This less detailed representation typically consists of a selection of several versions of objects in the scene,* ***each version less detailed and faster to render than the one before****.”*

The criteria to select one object representation among the others may vary and is application dependent.

For example, a distance criteria from the current user viewpoint can be used to represent far objects with lower geometry and/or texture details with respect to close objects as shown in Figure 1.

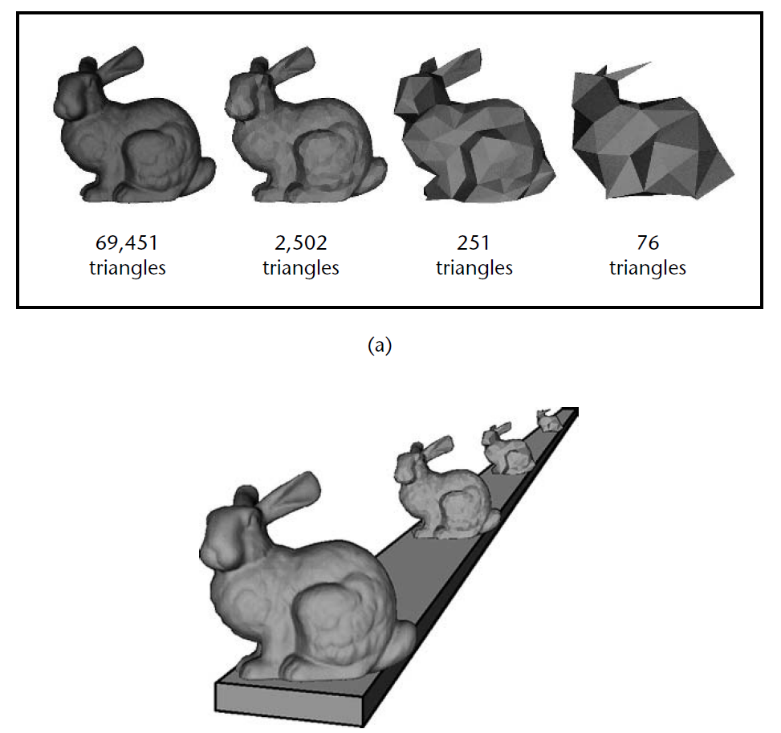


Figure 1: LOD based on the number of mesh triangles with a distance criteria from Level of Detail for 3D Graphics book [1]

LoD generation of 3D asset is widely supported by the Digital Content Creation (DCC) tools such as [Autodesk Maya](https://help.autodesk.com/view/MAYAUL/2024/ENU/?guid=GUID-79C7A942-0547-4AC4-8A4D-DCAC4ABB1EF2), [Unreal Engine](https://dev.epicgames.com/documentation/en-us/unreal-engine/creating-and-using-lods-in-unreal-engine), [Houdini](https://www.sidefx.com/docs/houdini/nodes/lop/createlod.html), [Unity 3D](https://docs.unity3d.com/2023.2/Documentation/Manual/LevelOfDetail.html).

A content creator may provide different objects LoDs in a scene description file for portability purpose between game engines. For example, the LoD functionality is supported the Khronos glTF 2.0 with the Microsoft MSFT\_lod [2] extension.

Figure 2 provides an example of creating three LOD levels for an object based on a screen coverage criteria using the glTF MSFT\_lod extension:

* The “High\_LOD” object representation is rendered for a screen coverage from 0.5 to 1.0,
* The “Medium\_LOD” object representation is rendered for a screen coverage from 0.2 to 0.5,
* The “Low\_LOD” object representation is rendered for a screen coverage from 0.01 to 0.2,
* The object is not rendered for a screen coverage below 0.01.

"nodes": [

{

"name": "High\_LOD",

"mesh": 0,

"extensions": {

"MSFT\_lod": {

"ids": [

1,

2

]

}

},

"extras": {

"MSFT\_screencoverage": [

0.5,

0.2,

0.01

]

}

},

{

"name": "Medium\_LOD",

"mesh": 1

},

{

"name": "Low\_LOD",

"mesh": 2

}

]

Figure 2: Example of glTF MSFT\_lod [2] extension defining three level of details

To point out that this glTF MSFT\_lod extension is generic in the sense that the selection criteria and the corresponding thresholds are provided in a MSFT\_screencoverage metadata specified in extras parameter (i.e., non-normative). Other criteria (e.g., viewing distance, field of view) may be defined to select these three object representations.

Another example of LoD feature is the MPEG\_media [3] extension of the Khronos glTF 2.0. The MPEG\_media extension with the “alternatives” parameter supports multiple media representations providing different rendering qualities.

MSFT\_lod and MPEG\_media extensions provide LoD, however, at different levels in the scene graph. MSFT\_lod offers LoD at the node level, while MPEG\_media offers LoD at the texture or mesh primitive levels.

The combination of both LoDs provides a finer granularity of the final object representation. Figure 3 shows various LoDs of an advertising vehicle.

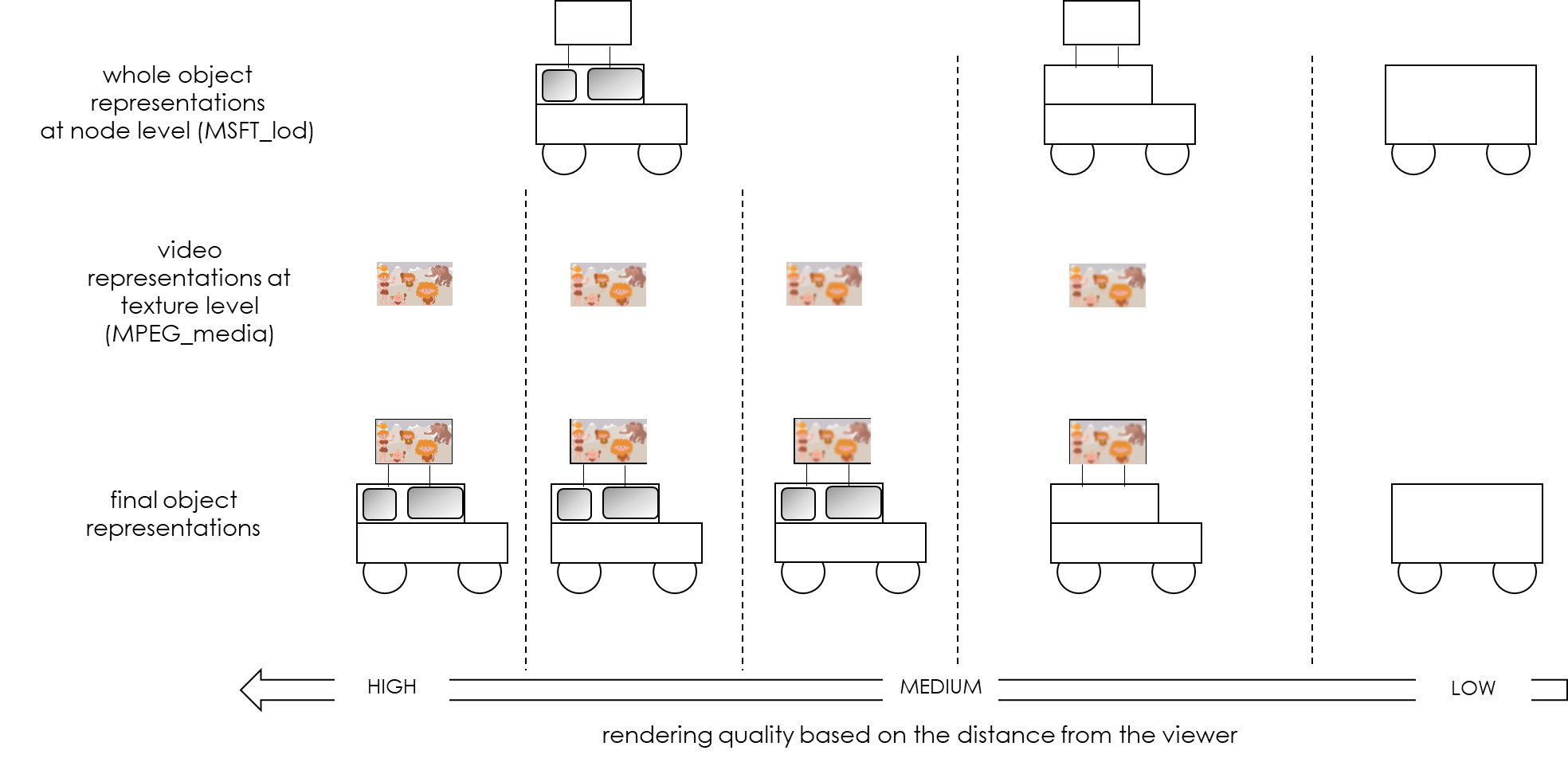


Figure 3: Example of combination of LoDs.

As described, the LoD mechanism handling multiple representations of an object is a generic method to introduce processing delay flexibility by modifying the object rendering quality in a controllable manner.

An XR scene may contain many objects having multiple LoDs, leading to offer to the XR application a wide range of processing delay capabilities.

**2. Reason for Change**

When Seamless Adaptive Split procedure is used, the SR-DCMTSI client in terminal renders some objects of the scene locally. The Processing Delay Adaptation may be used on the locally rendered objects.

The SR-DCMTSI client in terminal may determine and change the LoD of the locally rendered objects to adjust the processing delay. The SR-DCMTSI client in terminal may have to request the asset with the corresponding LoD from the MF.

We proposed to use the asset request message defined in clause A.2.7 to request the asset with one or more corresponding LoDs from the MF.

The SR-DCMTSI client must know the different LoDs available for determining the LoD of the locally rendered objects.

We proposed to clarify the definition of object identifiers in the adaptive split message defined in clause A.2.3, and in the seamless adaptive split message defined in clause A.2.4: the identifiers are named nodes of the graph corresponding to objects or the objects LoDs.

We proposed to extend the Processing Delay Adaptation Configuration format to include the reference of the different LoDs available for each object rendered by the SR-DCMTSI client in terminal.

**3. Proposal**

It is proposed to agree the following changes to 3GPP TS 26.567 v1.0.1.

**4. Reference**

[1] David Luebke, Martin Reddy, Jonathan D. Cohen, Amitabh Varshney, Benjamin Watson, Robert Huebner (2003). “Level of Detail for 3D Graphics”, <https://www.sciencedirect.com/book/9781558608382/level-of-detail-for-3d-graphics>.

[2] Microsoft MSFT\_lod extension of Khronos glTF 2.0: <https://github.com/KhronosGroup/glTF/tree/main/extensions/2.0/Vendor/MSFT_lod>.

[3] ISO/IEC SC29 WG3 (MPEG Systems) MPEG\_media extension of Khronos glTF 2.0: <https://github.com/KhronosGroup/glTF/tree/main/extensions/2.0/Vendor/MPEG_media>.

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

### 7.3.2 Processing Delay adaptation based on QoE metrics

An SR-DCMTSI client or an MF may trigger further procedures during a split rendering. An additional procedure may include adjusting various round-trip delays between the SR-DCMTSI client and the MF during a split rendering session. The delay adaptation information may be sent from the SR-DCMTSI client to the MF periodically when the measured QoE metrics (e.g., poseToRenderToPhoton, roundtripInteractionDelay) goes out of the target delay range. Upon receiving the notification, an MF may adjust the delay involved in some of the processing tasks. For example, an MF may change the Level of Detail (LoD) of the objects in an XR scene which impacts the processing complexity of the XR scene. Delay adaptation procedure may include data exchange, for example, exchange of information messages for delay adaptation. The following generic procedure may apply, while the exact details may depend on the DC-application being rendered.



Figure 7.3.2-1: General procedures for processing delay adaptation based on QoE metrics information

The steps are as follows:

1 to 3: As described in clause 7.3.1.

The steps for processing delay adaptation based on QoE metrics information during the rendering loop are as follows:

The SR-DCMTSI client measures and collects the QoE metrics negotiated in the delay adaptation configuration message. The QoE metrics considered for delay adaptation may contain all or a subset of the QoE latency metrics (e.g., poseToRenderToPhoton, roundtripInteractionDelay) negotiated in the metrics configuration message in clause 6.3.1.

4: The SR-DCMTSI client in terminal sends metadata required for rendering to the MF. In addition to the pose, pose predictions and user input metadata, the metadata may include a delay adaptation information message to the MF. The trigger to send the delay adaptation information message is based on the configured periodicity to inform the MF that the measured delay of a QoE latency metric is out of the target delay range. For example, the measured poseToRenderToPhoton QoE latency metric goes out of the target delay range due to new network conditions.

5a: The MF may adjust the processing delay based on the delay adaptation information message.

NOTE: For example, the MF may change the LoD of the objects that are part of the scene for the delay adaptation.

6 and 5b: The SR-DCMTSI client in terminal and the MF render the frame.

7 and 8: As described in clause 7.3.1.

When processing delay adaptation procedure is used ~~with split adaption procedure~~, the SR-DCMTSI client in terminal may render some objects of the scene and an MF may render other objects. The SR-DCMTSI client in terminal, may determine and change the LoD of the locally rendered objects to adjust the processing delay.

The SR-DCMTSI client in terminal shall use the asset request message defined in clause A.2.7 to request the asset with one or more corresponding LoDs from the MF.







\* \* \* 2nd Change \* \* \* \*

## A.2.7 Asset Request

An SR-DCMTSI client that supports the split rendering shall support the asset request messages below.

Table A.2.7-1 Message format for asset requests

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Cardinality | Description |
| id | string | 1..1 | A unique identifier of the message in the scope of the data channel session. |
| type | string | 1..1 | urn:3gpp:split-rendering:v1:asrp:sr-asset |
| message | Object | 1..1 | Message content |
| request | string | 1..1 | A request for assets, identifying the assets requested, for example as a list of nodes of a scene graph or a list of nodes of a scene graph corresponding to the Level-of-Details of the assets (objects), or a list of URIs which reference assets for nodes in the scene graph. |

\* \* \* 3rd Change \* \* \* \*

## A.2.5 Processing Delay Adaptation based on QoE metrics

### A.2.5.1 Configuration format

An SR-DCMTSI client that supports the processing delay adaptation shall support the processing delay adaptation configuration message format as below.

The MF shall share the configuration information of the delay adaptation procedure with the SR-DCMTSI client in terminal during the split rendering session negotiation and establishment processes. The configuration may be updated during the rendering loop. The configuration information of the delay adaptation procedure shall be in JSON format according to the Metadata Data Channel Message Format defined in clause 5.4.3. The message type shall be “urn:3gpp:split-rendering:v1:daqoe:configuration”.

Table A.2.5.1-1 – Configuration message format for Processing Delay adaptation based on QoE metrics

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Cardinality | Description |
| id | string | 1..1 | A unique identifier of the message in the scope of the data channel session. |
| type | string | 1..1 | urn:3gpp:split-rendering:v1:daqoe:configuration |
| message | Object | 1..1 | Message content |
| qoeMetrics | array | 1..1 | An array of the QoE metrics for which delay adaptation is considered. This qoeMetrics array may contain all or a subset of the QoE latency metrics negotiated in the metrics configuration message in clause 6.3.1. |
| qoeMetricId | string | 1..1 | A unique identifier of the QoE metric within the scope of the split rendering session. The name of that QoE metric is chosen as unique ID, this name should be consistent with the name provided in the metrics reporting configuration defined in clause 6.3.1. |
| periodicity | string | 1..1 | The periodicity of the delay adaptation information for that QoE metric. It may be expressed as a multiple of the "Measure-Resolution" defined in clause 16.3.2 of TS 26.114 [7].  Whenever a delay adaptation information message is sent, the SR-DCMTSI client in terminal shall reset its timer to the value of the periodicity property and it shall begin countdown of the timer again. |
| flexibleObjects | array | 0..1 | An array of objects for which several LoDs are available for the delay adaptation. |
| flexibleObjectId | string | 1..1 | A unique identifier of an object within the scope of the split rendering session. For example, the index of the node of the object in the scene description. |
| levelIds | Array(string) | 1..1 | An array of node identifiers in a scene description corresponding to the LoDs of the object identified by the flexibleObjectId. |
| criteria | string | 0..1 | An information to guide the application for determining the object and corresponding LoD available in the XR scene. This information can be either:  - “VIEWING\_DISTANCE”: for increasing the LoD (i.e., by increasing the processing delay) for close object(s) if the measured delay is below the target delay and by decreasing the LoD (i.e., by decreasing the processing delay) for far object(s) if the measured delay is above the target delay,  - “FIELD\_OF\_VIEW”: for increasing the LoD (i.e., by increasing the processing delay) for object(s) in the center of the FoV if the measured delay is below the target delay and by decreasing the LoD (i.e., by decreasing the processing delay) for object(s) located at the borders of the FoV if the measured delay is above the target delay,  - “SCREEN\_COVERAGE”: for increasing the LoD (i.e., by increasing the processing delay) for object(s) having larger screen coverage if the measured delay is below the target delay and by decreasing the LoD (i.e., by decreasing the processing delay) for object(s) having smaller screen coverage if the measured delay is above the target delay. |

NOTE: The target delay range for a QoE metric is delimited by a minimum threshold and a maximum threshold delay value. The thresholds and the target delay, for each QoE metric in the qoeMetrics array, can be provided by the DC Application Server to the SR-DCMTSI client in terminal.

### A.2.5.2 Metadata format

During a IMS-based split rendering session, the operating environment of the serving MF, the resources of SR-DCMTSI client or the network conditions may change. Consequently, the roundtrip delay may need to be adjusted to deliver a consistent QoE.

When delay adaptation procedure is enabled, the SR-DCMTSI client in terminal checks for the QoE metrics being monitored whether the measured delays are within the target delay range or not. When a measured delay is outside the target delay range for a QoE metric, the SR-DCMTSI client in terminal may report the measured delay based on the configured periodicity.An SR-DCMTSI client that supports the processing delay adaptation shall support the processing delay adaptation information message format as below.

The delay adaptation information message format that is used for IMS-based split rendering shall comply with the format defined in Table A.2.5.2-1. The delay adaptation information message shall be carried as part of the data channel messaging mechanism. The metadata data channel message shall be in JSON format according to the Metadata Data Channel Message Format defined in clause 5.4.3. The message type shall be “urn:3gpp:split-rendering:v1:daqoe:information”.

Table A.2.5.2-1 Metadata format for Processing Delay Adaptation information message based on QoE metrics

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Cardinality | Description |
| id | string | 1..1 | A unique identifier of the message in the scope of the IMS-based split rendering session. |
| type | string | 1..1 | urn:3gpp:split-rendering:v1:daqoe:information |
| message | object | 1..1 | Message content |
| qoeMetrics | array | 1..1 | An array of the QoE metrics for which delay adaptation is needed. This qoeMetrics array may contain all or a subset of the QoE metrics negotiated in the configuration message in clause A.1.5.1. |
| qoeMetricId | string | 1..1 | A unique identifier of the QoE metric within the scope of the split rendering session. |
| delayValue | number | 1..1 | The measured delay value of that QoE metric. |

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