Media Operations Use Case for an Extended Reality Application on Edge Computing Infrastructure
draft-ietf-mops-ar-use-case-05

Abstract

This document explores the issues involved in the use of Edge Computing resources to operationalize media use cases that involve Extended Reality (XR) applications. In particular, we discuss those applications that run on devices having different form factors and need Edge computing resources to mitigate the effect of problems such as a need to support interactive communication requiring low latency, limited battery power, and heat dissipation from those devices. The intended audience for this document are network operators who are interested in providing edge computing resources to operationalize the requirements of such applications. We discuss the expected behavior of XR applications which can be used to manage the traffic. In addition, we discuss the service requirements of XR applications to be able to run on the network.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 12 January 2023.

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

Extended Reality (XR) is a term that includes Augmented Reality (AR), Virtual Reality (VR) and Mixed Reality (MR) [XR]. AR combines the real and virtual, is interactive and is aligned to the physical world of the user [AUGMENTED_2]. On the other hand, VR places the user inside a virtual environment generated by a computer [AUGMENTED]. MR merges the real and virtual world along a continuum that connects completely real environment at one end to a completely virtual environment at the other end. In this continuum, all combinations of the real and virtual are captured [AUGMENTED].
XR applications will bring several requirements for the network and the mobile devices running these applications. Some XR applications such as AR require a real-time processing of video streams to recognize specific objects. This is then used to overlay information on the video being displayed to the user. In addition, XR applications such as AR and VR will also require generation of new video frames to be played to the user. Both the real-time processing of video streams and the generation of overlay information are computationally intensive tasks that generate heat [DEV_HEAT_1], [DEV_HEAT_2] and drain battery power [BATT_DRAIN] on the mobile device running the XR application. Consequently, in order to run applications with XR characteristics on mobile devices, computationally intensive tasks need to be offloaded to resources provided by Edge Computing.

Edge Computing is an emerging paradigm where computing resources and storage are made available in close network proximity at the edge of the Internet to mobile devices and sensors [EDGE_1], [EDGE_2]. These edge computing devices use cloud technologies that enable them to support offloaded XR applications. In particular, the edge devices deploy cloud computing implementation techniques such as disaggregation (breaking vertically integrated systems into independent components with open interfaces using SDN), virtualization (being able to run multiple independent copies of those components such as SDN Controller apps, Virtual Network Functions on a common hardware platform) and commoditization (being able to elastically scale those virtual components across commodity hardware as the workload dictates) [EDGE_3]. Such techniques enable XR applications requiring low-latency and high bandwidth to be delivered by mini-clouds running on proximate edge devices.

In this document, we discuss the issues involved when edge computing resources are offered by network operators to operationalize the requirements of XR applications running on devices with various form factors. Examples of such form factors include Head Mounted Displays (HMD) such as Optical-see through HMDs and video-see-through HMDs and Hand-held displays. Smart phones with video cameras and GPS are another example of such devices. These devices have limited battery capacity and dissipate heat when running. Besides as the user of these devices moves around as they run the XR application, the wireless latency and bandwidth available to the devices fluctuates and the communication link itself might fail. As a result algorithms such as those based on adaptive-bit-rate techniques that base their policy on heuristics or models of deployment perform sub-optimally in such dynamic environments [ABR_1]. In addition, network operators can expect that the parameters that characterize the expected behavior of XR applications are heavy-tailed. Such workloads require appropriate resource management policies to be used on the Edge. The service...
requirements of XR applications are also challenging when compared to the current video applications. In particular several QoE factors such as motion sickness are unique to XR applications and must be considered when operationalizing a network. We motivate these issues with a use-case that we present in the following sections.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Use Case

We now describe a use case that involves an application with AR systems’ characteristics. Consider a group of tourists who are being conducted in a tour around the historical site of the Tower of London. As they move around the site and within the historical buildings, they can watch and listen to historical scenes in 3D that are generated by the AR application and then overlaid by their AR headsets onto their real-world view. The headset then continuously updates their view as they move around.

The AR application first processes the scene that the walking tourist is watching in real-time and identifies objects that will be targeted for overlay of high resolution videos. It then generates high resolution 3D images of historical scenes related to the perspective of the tourist in real-time. These generated video images are then overlaid on the view of the real-world as seen by the tourist.

We now discuss this processing of scenes and generation of high resolution images in greater detail.

3.1. Processing of Scenes

The task of processing a scene can be broken down into a pipeline of three consecutive subtasks namely tracking, followed by an acquisition of a model of the real world, and finally registration [AUGMENTED].

Tracking: This includes tracking of the three dimensional coordinates and six dimensional pose (coordinates and orientation) of objects in the real world[AUGMENTED]. The AR application that runs on the mobile device needs to track the pose of the user’s head, eyes and the objects that are in view. This requires tracking natural features that are then used in the next stage of the pipeline.
Acquisition of a model of the real world: The tracked natural features are used to develop an annotated point cloud based model that is then stored in a database. To ensure that this database can be scaled up, techniques such as combining a client side simultaneous tracking and mapping and a server-side localization are used[SLAM_1], [SLAM_2], [SLAM_3], [SLAM_4].

Registration: The coordinate systems, brightness, and color of virtual and real objects need to be aligned in a process called registration [REG]. Once the natural features are tracked as discussed above, virtual objects are geometrically aligned with those features by geometric registration. This is followed by resolving occlusion that can occur between virtual and the real objects [OCCL_1], [OCCL_2]. The AR application also applies photometric registration [PHOTO_REG] by aligning the brightness and color between the virtual and real objects. Additionally, algorithms that calculate global illumination of both the virtual and real objects [GLB_ILLUM_1], [GLB_ILLUM_2] are executed. Various algorithms to deal with artifacts generated by lens distortion [LENS_DIST], blur [BLUR], noise [NOISE] etc are also required.

3.2. Generation of Images

The AR application must generate a high-quality video that has the properties described in the previous step and overlay the video on the AR device’s display - a step called situated visualization. This entails dealing with registration errors that may arise, ensuring that there is no visual interference [VIS_INTERFERE], and finally maintaining temporal coherence by adapting to the movement of user’s eyes and head.

4. Requirements

The components of AR applications perform tasks such as real-time generation and processing of high-quality video content that are computationally intensive. As a result, on AR devices such as AR glasses excessive heat is generated by the chip-sets that are involved in the computation [DEV_HEAT_1], [DEV_HEAT_2]. Additionally, the battery on such devices discharges quickly when running such applications [BATT_DRAIN].
A solution to the heat dissipation and battery drainage problem is to offload the processing and video generation tasks to the remote cloud. However, running such tasks on the cloud is not feasible as the end-to-end delays must be within the order of a few milliseconds. Additionally, such applications require high bandwidth and low jitter to provide a high QoE to the user. In order to achieve such hard timing constraints, computationally intensive tasks can be offloaded to Edge devices.

Another requirement for our use case and similar applications such as 360 degree streaming is that the display on the AR/VR device should synchronize the visual input with the way the user is moving their head. This synchronization is necessary to avoid motion sickness that results from a time-lag between when the user moves their head and when the appropriate video scene is rendered. This time lag is often called "motion-to-photon" delay. Studies have shown \cite{PER_SENSE}, \cite{XR}, \cite{OCCL_3} that this delay can be at most 20ms and preferably between 7-15ms in order to avoid the motion sickness problem. Out of these 20ms, display techniques including the refresh rate of write displays and pixel switching take 12-13ms \cite{OCCL_3}, \cite{CLOUD}. This leaves 7-8ms for the processing of motion sensor inputs, graphic rendering, and RTT between the AR/VR device and the Edge. The use of predictive techniques to mask latencies has been considered as a mitigating strategy to reduce motion sickness \cite{PREDICT}. In addition, Edge Devices that are proximate to the user might be used to offload these computationally intensive tasks.

Towards this end, the 3GPP requires and supports an Ultra Reliable Low Latency of 0.1ms to 1ms for communication between an Edge server and User Equipment (UE) \cite{URLLC}.

Note that the Edge device providing the computation and storage is itself limited in such resources compared to the Cloud. So, for example, a sudden surge in demand from a large group of tourists can overwhelm that device. This will result in a degraded user experience as their AR device experiences delays in receiving the video frames. In order to deal with this problem, the client AR applications will need to use Adaptive Bit Rate (ABR) algorithms that choose bit-rates policies tailored in a fine-grained manner to the resource demands and playback the videos with appropriate QoE metrics as the user moves around with the group of tourists.

However, heavy-tailed nature of several operational parameters make prediction-based adaptation by ABR algorithms sub-optimal \cite{ABR_2}. This is because with such distributions, law of large numbers works too slowly, the mean of sample does not equal the mean of distribution, and as a result standard deviation and variance are unsuitable as metrics for such operational parameters \cite{HEAVY_TAIL_1}, \cite{HEAVY_TAIL_2}. Other subtle issues with these distributions include
the "expectation paradox" [HEAVY_TAIL_1] where the longer we have waited for an event the longer we have to wait and the issue of mismatch between the size and count of events [HEAVY_TAIL_1]. This makes designing an algorithm for adaptation error-prone and challenging. Such operational parameters include but are not limited to buffer occupancy, throughput, client-server latency, and variable transmission times. In addition, edge devices and communication links may fail and logical communication relationships between various software components change frequently as the user moves around with their AR device [UBICOMP].

Thus, once the offloaded computationally intensive processing is completed on the Edge Computing, the video is streamed to the user with the help of an ABR algorithm which needs to meet the following requirements [ABR_1]:

* Dynamically changing ABR parameters: The ABR algorithm must be able to dynamically change parameters given the heavy-tailed nature of network throughput. This, for example, may be accomplished by AI/ML processing on the Edge Computing on a per client or global basis.

* Handling conflicting QoE requirements: QoE goals often require high bit-rates, and low frequency of buffer refills. However, in practice, this can lead to a conflict between those goals. For example, increasing the bit-rate might result in the need to fill up the buffer more frequently as the buffer capacity might be limited on the AR device. The ABR algorithm must be able to handle this situation.

* Handling side effects of deciding a specific bit rate: For example, selecting a bit rate of a particular value might result in the ABR algorithm not changing to a different rate so as to ensure a non-fluctuating bit-rate and the resultant smoothness of video quality. The ABR algorithm must be able to handle this situation.

5. AR Network Traffic

5.1. Traffic Workload

As discussed earlier, the parameters that capture the characteristics of XR application behavior are heavy-tailed. Examples of such parameters include the distribution of arrival times between XR application invocation, the amount of data transferred, and the inter-arrival times of packets within a session. As a result, any traffic model based on such parameters are themselves heavy-tailed. Using these models to predict performance under alternative resource
allocations by the network operator is challenging.

5.2. Traffic Performance Metrics

The performance requirements for AR/VR traffic have characteristics that need to be considered when operationalizing a network. We now discuss these characteristics.

The bandwidth requirements of XR applications are substantially higher than those of video based applications.

The latency requirements of XR applications have been studied recently [AR_TRAFFIC]. The following issues were identified:

* The uploading of data from an AR device to a remote server for processing dominates the end-to-end latency.

* A lack of visual features in the grid environment can cause increased latencies as the AR device uploads additional visual data for processing to the remote server.

* AR applications tend to have large bursts that are separated by significant time gaps.

The packet loss rates in wireless links between XR devices and the Edge server can be as high as 2% or more [WIRELESS_1]. 5G systems however promise a reliability of 99.999% with an air interface latency of 0.5ms[RELIABILITY_1], [RELIABILITY_2].

Finally, XR applications interact with each other on a time scale of a round-trip-time propagation and this must be considered when operationalizing a network.

6. Acknowledgements

Many Thanks to Spencer Dawkins, Rohit Abhishek, Jake Holland, Kiran Makhijani and Ali Begen for providing very helpful feedback suggestions and comments.

7. Informative References


[RELIABILITY_1]

[RELIABILITY_2]


Authors' Addresses

Renan Krishna
InterDigital Europe Limited
64, Great Eastern Street
London
EC2A 3QR
United Kingdom
Email: renan.krishna@interdigital.com

Akbar Rahman
InterDigital Communications, LLC
1000 Sherbrooke Street West
Montreal H3A 3G4
Canada
Email: Akbar.Rahman@InterDigital.com