ETSI TR 126 926 V18.2.0 (2024-05)

TECHNICAL REPORT

LTE;
5G;
Traffic Models and Quality Evaluation Methods
for Media and XR Services in 5G Systems
(3GPP TR 26.926 version 18.2.0 Release 18)

Reference

DTR/TSGS-0426926vI20

Keywords

5G,LTE

Contents

Foreword

This Technical Report has been produced by the 3rd Generation Partnership Project (3GPP).

The contents of the present document are subject to continuing work within the TSG and may change following formal TSG approval. Should the TSG modify the contents of the present document, it will be re-released by the TSG with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

- x the first digit:
  - 1 presented to TSG for information;
  - 2 presented to TSG for approval;
  - 3 or greater indicates TSG approved document under change control.
- y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.
- z the third digit is incremented when editorial only changes have been incorporated in the document.

In the present document, modal verbs have the following meanings:

shall	indicates a mandatory requirement to do something
shall not	indicates an interdiction (prohibition) to do something

The constructions "shall" and "shall not" are confined to the context of normative provisions, and do not appear in Technical Reports.

The constructions "must" and "must not" are not used as substitutes for "shall" and "shall not". Their use is avoided insofar as possible, and they are not used in a normative context except in a direct citation from an external, referenced, non-3GPP document, or so as to maintain continuity of style when extending or modifying the provisions of such a referenced document.

should	indicates a recommendation to do something
should not	indicates a recommendation not to do something
may	indicates permission to do something
need not	indicates permission not to do something

The construction "may not" is ambiguous and is not used in normative elements. The unambiguous constructions "might not" or "shall not" are used instead, depending upon the meaning intended.

can	indicates that something is possible
cannot	indicates that something is impossible

The constructions "can" and "cannot" are not substitutes for "may" and "need not".

will	indicates that something is certain or expected to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document
will not	indicates that something is certain or expected not to happen as a result of action taken by an agency the behaviour of which is outside the scope of the present document
might	indicates a likelihood that something will happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

might not	indicates a likelihood that something will not happen as a result of action taken by some agency the behaviour of which is outside the scope of the present document

In addition:

is	(or any other verb in the indicative mood) indicates a statement of fact
is not	(or any other negative verb in the indicative mood) indicates a statement of fact

The constructions "is" and "is not" do not indicate requirements.

Introduction

Media services in general, but also in particular eXtended Reality (XR) and Cloud Gaming are some of the most important 5G media applications under consideration in the industry. XR is an umbrella term for different types of realities and refers to all real-and-virtual combined environments and human-machine interactions generated by computer technology and wearables. It includes representative forms such as Augmented Reality (AR), Mixed Reality (MR) and Virtual Reality (VR) and the areas interpolated among them.

On XR and Cloud Gaming traffic with high throughput, low latency and high reliability requirements, it is important to consider system aspects of such services. If the traffic requirements of the XR and Cloud Gaming service are flexible (e.g., the underlying architecture allows adaptation of content), then the capacity of the service can be studied by assessing the delay, throughput and reliability variations with increasing number of users in the system. In order to properly study this, detailed traffic characteristics are necessary.

Based on this, the present document provides traffic models and quality evaluation methods for different media and eXtended Reality (XR) Services. In order to address this, generic modelling considerations are introduced and for different services reference designs, simulation models and suitable quality metrics are reported. This information permits to obtain accurate information on exact bitrate and delay requirements in uplink and downlink, develop detailed traffic traces, develop suitable statistical models for media and XR traffic and to evaluate the expected media quality of such services. The information may be used by other 3GPP groups in order to assess media quality for different configuration of 5G System parameters, as well as for evaluating the requirements in terms of QoS for XR and media services.

1 Scope

The present document provides traffic models and quality evaluation methods for different media and eXtended Reality (XR) Services. In order to address this, generic modelling considerations are introduced and for different services reference designs, simulation models and suitable quality metrics are reported. This information permits to obtain accurate information on exact bitrate and delay requirements in uplink and downlink, develop detailed traffic traces, develop suitable statistical models for media and XR traffic and to evaluate the expected media quality of such services. The information may be used by other 3GPP groups in order to assess media quality for different configuration of 5G System parameters, as well as for evaluating the requirements in terms of QoS for XR and media services.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TR 26.925, "Typical traffic characteristics of media services on 3GPP networks"

[3] 3GPP TR 26.928, "Extended Reality (XR) in 5G"

[4] 3GPP TR 38.838, "Study on XR (Extended Reality) Evaluations for NR"

[5] 3GPP TS 26.501, "System architecture for the 5G System (5GS)"

[6] 3GPP TS 26.118, "Virtual Reality (VR) profiles for streaming applications"

[7] VR-IF Guidelines, https://www.vr-if.org/wp-content/uploads/vrif2020.180.00-Guidelines-2.3_clean.pdf

[8] IETF RFC 5052, Forward Error Correction (FEC) Building Block

[9] IETF RFC 6330, RaptorQ Forward Error Correction Scheme for Object Delivery

[10] IETF RFC 6681, Raptor Forward Error Correction (FEC) Schemes for FECFRAME

[11] IETF RFC 8627, RTP Payload Format for Flexible Forward Error Correction (FEC)

3 Definitions of terms, symbols and abbreviations

3.1 Terms

For the purposes of the present document, the terms given in 3GPP TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in 3GPP TR 21.905 [1].

trace: a well-defined format to describe a sequence of timed data units together with relevant metadata for system simulation.

3.2 Symbols

For the purposes of the present document, the following symbols apply:

void

3.3 Abbreviations

For the purposes of the present document, the abbreviations given in 3GPP TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in 3GPP TR 21.905 [1].

AAC	Advanced Audio Coding
ACK	ACKnowledgment message
ADU	Application Data Unit
ALR	Area Loss Rate
AMR	Adaptive Multi Rate
API	Application Programming Interface
ATW	Asynchronous Time Warping
AVC	Advanced Video Coding
CAR	Correct Area Rate
CAT	CATegory
CBR	Constant BitRate
CDF	Cumulative Distribution Function
CDN	Content Delivery Network
CGI	Computer Graphics and Images
CQP	Contant QP
CRF	Constant Rate Factor
CSV	Comma Separated Version
CTU	Coding Tree Unit
CU	Coding Unit
DAR	DAmage Rate
DASH	Dynamic Adaptive Streaming over HTTP
DRX	Discontinous Receive
ERP	Equi-Rectangular Projection
EVS	Enhanced Voice Service
FEC	Forward Error Correction
FFS	For Further Study
FOV	Fielf-of-View
FPS	First Person Shooter
GBR	Guaranteed Bit Rate
GDR	Gradual Decoder Refresh
GFBR	Guaranteed Flow Bit Rate
HARQ	Hybrid Automatic Repeat 9equest
HDR	High Dynamic Range
HEVC	High Efficiency Video Coding
HLS	HTTP Live Streaming
HTTP	HyperText Transfer Protocol
IAT	Inter arrival times
IBC	International Broadcasting Convention
IDR	Instantenous Decoder Refresh
IVAS	Immersive Voice and Audio Service
LDR	Loss and Damage Rate
MDBV	Maximum Data Burst Volume
MFBR	Maximum Flow BitRate
MMO	Massive Multiplayer Online
MOS	Mean Opinion Score
MPEG	Moving Pictures Expert Group
MTHR	Medium-to-High Ratio
MTU	Maximum Transfer Unit

NACK	Negation ACKnowledgement message
OMAF	Omnidirectional MediA Format
PDB	Packet Delay Budget
PDCP	Packet Data Convergence Protocol
PDU	Packet Data Unit
PER	Packet Error Rate
PLR	Packet Loss Rate
POC	Picture Order Count
PSNR	Peak Signal to Noise Ratio
QFI	QoS Flow Indicator
QP	Quantization Parameter
RAN	Radio Access Network
RLC	Radio Link Control
RPG	Role-Player Game
RPSNR	Recovered PSNR
RTCP	Real-Time Control Protocol
RTP	Real-Time Protocol
RTS	Real-Time Strategy
RTT	Round Trip Time
SLR	Slice Loss Rate
SSIM	Structural Similarity Index Measure
STD	Standard Deviation
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
UPF	User Plane Function
VBR	Variable BitRate
VDP	Viewport DePendent

4 Overview and Scope

In an initial version of TR26.925, Typical Traffic Characteristics for Operator and Third-Party Services have been collected. The work was initiated based on communication between SA4 and SA1. During the course of the work, additional requests from SA1 were received that have been partially addressed in the initial version of TR26.925.

3GPP TSG SA WG4 (Codec) addressed their mandate on "Guidance to other 3GPP groups concerning required QoS parameters and other system implications, including channel coding requirements, imposed by different multimedia codecs in both circuit-switched and packet-switched environments."

Furthermore, during the study for eXtended Reality (XR) over 5G (FS_5GXR) documented in TR26.928, several initial considerations for XR services including cloud gaming had been collected. Specifically, parameters such as downlink and uplink bitrates, packet delay budgets, error rates and round-trip times are collected, but several of those for different cases are kept FFS and need more work.

In particular, eXtended Reality (XR) and Cloud Gaming are some of the most important 5G media applications under consideration in the industry. XR is an umbrella term for different types of realities and refers to all real-and-virtual combined environments and human-machine interactions generated by computer technology and wearables. It includes representative forms such as Augmented Reality (AR), Mixed Reality (MR) and Virtual Reality (VR) and the areas interpolated among them.

This Technical Report attempts to provide the following information:

- Collect and document traffic characteristics including for different services, but not limited to
    - Downlink data rate ranges
    - Uplink data rate ranges
    - Maximum packet delay budget in uplink and downlink
    - Maximum Packet Error Rate,
    - Maximum Round Trip Time

  - Traffic Characteristics on IP level in uplink and downlink in terms of packet sizes, and temporal characteristics. XR Services and Cloud Gaming based on the initial information documented in TR26.928 including.
- Collect additional information, such as codecs and protocols in use.
- Provide the information from above at least for the following services (initial services)
  - Viewport independent 6DoF Streaming
  - Viewport dependent 6DoF Streaming
  - Simple Single Buffer split rendering for online cloud gaming
  - Cloud gaming
  - MTSI-based XR conversational services
- Identify additional relevant XR and other media services and document their traffic characteristics
- Document additional developments in the industry that impact traffic characteristics in future networks
- Identify the applicability of existing 5QIs/PQIs for such services and potentially identify requirements for new 5QIs/PQIs or QoS related parameters.

The work is expected to be carried out with other 3GPP groups and external organizations on relevant aspects related to the scope of the work.

In addition, extensions to TR 26.925 are developed to document traffic characteristics for XR applications.

5 General Design and Modelling

5.1 Introduction

The system and RAN model used in this Technical Report for XR Traffic analysis follows the 5G System architecture as defined in TS 23.501 [5.1i]. In particular, the user plane aspects are considered as showing in Figure 5.1-1 for which an XR Server is in the external DN or in a trusted DN, and the XR Device is a 5G UE. The exchange of data is assumed to be carried out using the 5G System. Interfaces to the 5G Core functions for charging, QoS control, etc. are not considered in the below diagram, for details refer to TS 23.501.

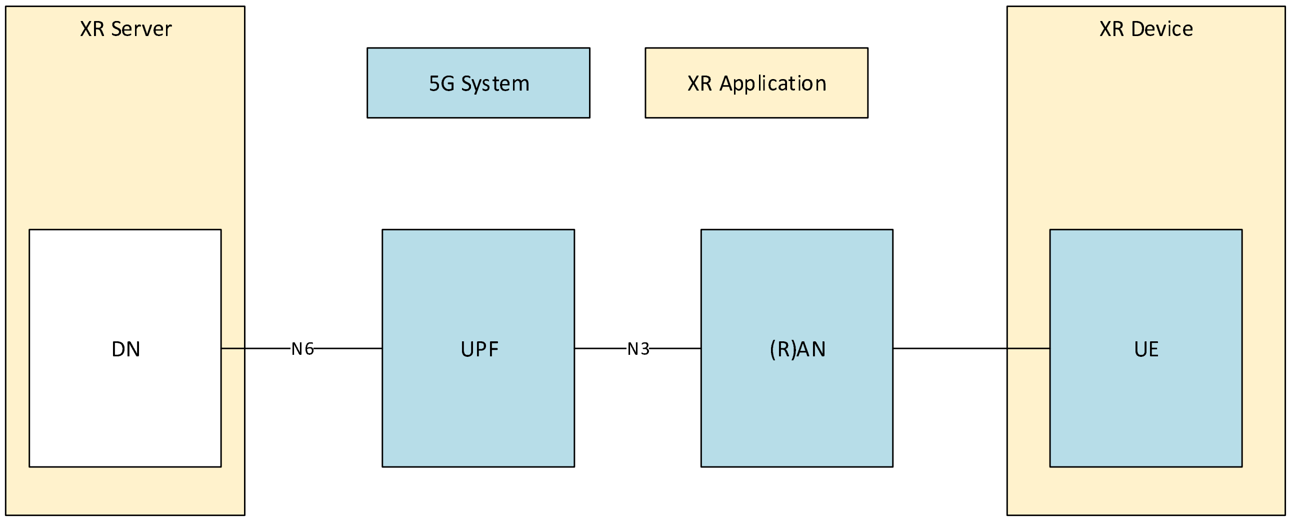


Figure 5.1-1 Simplified Architecture for XR end-to-end services

In order to identify the traffic characteristics and the impacts of media and XR applications on 5G Systems and radio access networks, an appropriate modelling of both, the XR traffic as well as of the 5G System is desirable, also to identify the interaction of the two components. Note that the XR Server may be hosted in the cloud or may be hosted on an edge.

Details on the modelling of the 5G System and RAN as well as simulation parameters are provided in clause 5.2.

5.2 End-to-end Modelling of XR Traffic

5.2.1 Architecture and System Model

In order to support modelling of end-to-end XR Traffic, an abstracted architecture and system model is introduced in Figure 5.2.1-1. This system is used as a baseline and may be refined for specific traffic classes. The system follows the basic building blocks from clause 5.1 but provides details for each of those aspects. In order to support simpler system modelling and simulation, the interfaces are supported by *traces*. Traces define a well-defined format to describe a sequence of timed data units together with relevant metadata for system simulation. An overview of traces is introduced in clause 5.2.2, details are provided in each of the functions.

In order to split the tasks across different 3GPP working groups and companies, an approach as introduced in Figure 5.2.1-1 is considered:

- A content model based on traces may be provided based on company input providing sufficient information on a rendered application, e.g. a game or a scene.
- Based on these content model traces, codec centric experts defines a content encoding and delivery model taking into account a system design, for example based on TR26.928 [3]. The system model includes encoding, delivery, decoding and also a quality definition, providing packet traces.
- 5G System and RAN simulations may be carried out by the 3GPP radio experts using the packet traces to simulate different traffic characteristics and evaluate different performance options.

An overview of the functions is provided below:

1) Content Model: Provides typical characteristics for the XR content model for video, audio and potentially other data. This content is rendered for XR/CG consumption – for details refer to clause 5.5.
2) Content Encoding Model: Provides the details for the content encoding in order to meet certain objectives. This includes the generation of sequences of application data units (slices, video frames, audio frames, etc.) and the incurred timestamp of each of the units is available. For details refer to clause 5.6.
3) Content Delivery Model: provides details on content delivery, for example packetization, delay jitter, but possibly also more sophisticated models such as retransmission, TCP operations and so on. This also includes emulation of 5G Core Network. It produces packet traces (timestamp and size of each packet) based on traces of application data units. For details refer to clause 5.7.
4) Core Network Model: This model emulates the core network behaviour in terms of delay, latency and packet losses in the core network, i.e. the interface between the UPF and gNB. For details refer to clause 5.4.
5) Packet Radio Model: The packet radio model receives sequences/traces of packets at a given time and of a specified size. The packets may have additional metadata assigned that can potentially be used by the radio simulator. The packet traces are provided for multiple users and reflect typical traffic characteristics. The RAN simulator provides packet traces after delivery that reflect the occurred delays and losses for each user. For details refer to clause 5.4.
6) Content Receiver Model: The content receiver converts the packet traces into application units taking into account delays and losses occurred. It may also model additional functions such as retransmissions or FEC, if applicable. For details refer to clause 5.7.
7) Content Decoding Model: The decoding model uses the received application units to model the reconstruction of timed data (video frames, audio) considering delays, losses and also content model properties (error propagation, refresh data and so on). For details refer to clause 5.6.
8) Quality Evaluation: A quality evaluation tool is provided that takes into account traces to compute different quality metrics based on packet, application unit and media quality. For details refer to clause 5.8.

9) Uplink Model: A model for the uplink traffic in a similar fashion also providing packet traces. For details refer to clause 5.9.

Several of the functions are initialized by appropriate configurations. An overview of configurations is introduced in each of the functions, if needed.

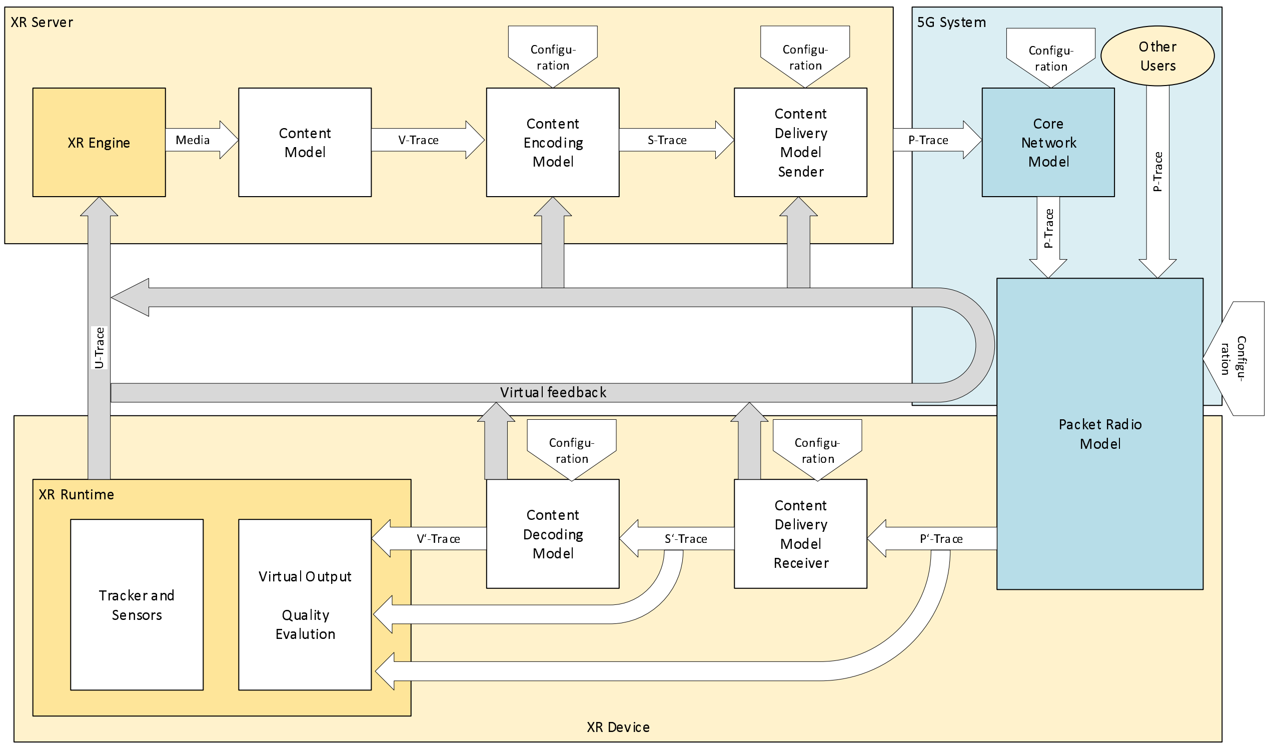


Figure 5.2.1-1 General architecture and system model for XR Traffic

5.2.2 Interfaces and Traces

5.2.2.1 Introduction

Generally, traces include the following information:

- Time series of data units
- Size of each data units
- Association to the application
- Additional "metadata"

Traces can also directly be fed to a quality evaluation tool obtain the application quality based on the simulation as shown in clause 5.8. The benefit of traces compared to first order statistical model are as follows:

- traces to represent time correlations.
- traces can be used to assign additional metadata in the time series
- statistical models can be developed based on time series
- traces can be directly fed to quality evaluation tool

The trace formats are defined as CSV files and follow the recommendation in RFC4180 "Common Format and MIME Type for CSV Files". For CSV files, headers are added to identify the data types. From existing CSV types, the online tool https://csv.openbridge.dev/ has been used to

1) Validate the csv file

2) Create a schema for the files.

5.2.2.2 Overview of Traces

Figure 5.2.1-1 shows different traces, associated to different interfaces. The following traces are defined:

1) V-Traces: Video traces provide a time series of the statistics/complexity of a video frame. Such a V-Trace is expected to provide sufficient information such that a coding model can create suitable statistics statics for encoded video streams at the output. For details refer to clause 5.5.

2) S-Traces: Slice traces provide a time series of encoded video application data units, referred to as slices as known from H.264/AVC and H.265/HEVC suitable application data units as outputs from a video codecs. S-Traces are expected to provide sufficient information to the content delivery module to create IP packets. For details refer to clause 5.6.

3) P-Traces: P Traces provide a time series of IP packets, possibly associated with different application flows and/or QoS Flows. In addition, they provide metadata information that may be beneficial for advanced delivery. For details refer to clause 5.3.

4) P'-Traces have the same format as P-Traces, but in addition may include losses and delays observed due to the delivery over a network.

5) S'-Traces are have the same format as S-Traces, but in addition may include information on losses and distortions due to the delivery over a network. For details refer to clause 5.7.

6) V'-Traces: Video receive traces provide a time series of received video frames together with an associated encoding and delivery quality. For details refer to clause 5.6.

5.2.3 Benefits and Limitations

This aspect is for further study.

5.3 P-Traces

Packet Traces (P-Traces) are defined to emulate the arrival of IP packets with associated metadata at certain network nodes. A P-Trace format together with semantics is defined in Table 5.3-1.

The first row just keeps an index of the packets.

The second to fourth data row information is considered to be available in a general delivery environment and are summarized as baseline parameters.

- `availability_time`: A relative time from the start of the trace when this packet is available in the delivery function
- `size`: the size of the packet in octets
- `flow_id`: an id to a flow in order to differentiate different QoS flows to identify a PDU session aligned with the QFI.

Table 5.3-1 P-Trace format

Name	Type	Semantics
`number`	BIGINT	Unique packet number in the delivery
Baseline parameters		
`availability_time`	BIGINT	Availability time of packet for next processing step relative to start time 0 in microseconds (0 means lost).
`size`	BIGINT	packet size in bytes.
`flow_id`	BIGINT	an id to the flow in order to differentiate different QoS flows
Cross-Layer Parameter		

buffer	BIGINT	The associated eye buffer 1=left 2=right In general, differentiates application traffic for different buffers, for example audio, video, left eye, right eye.
delay	BIGINT	Delay observed of the packet in the last processing step (-1 means lost)
render_timing	BIGINT	the rendering generation timing associated to the media included in the packet.
number_in_unit	BIGINT	The number of the packet within the unit (slice), start at 1
last_in_unit	BIGINT	Indicates if this is the last packet in the slice/unit 0=no, 1=yes
type	BIGINT	The data type of the unit 0 unknown For video 1=intra 2=inter
importance	BIGINT	assigned relative importance information (higher number means higher importance)

In addition, it is considered that an application may provide additional metadata per packet that if present, may be used by the delivery system. Different definitions are provided in the P-Trace format. The signaling of this data a 5G System is for further study. This information would not be available to RAN delivery in the incoming IP packets. These parameters are summarized as cross-layer parameters.

5GS through the QoS model may provide the ability that application streams are separated into different streams or the streams are at least marked accordingly. Details would need further discussion with SA2 on how application traffic and packet properties or stream properties can be matched to the 5G System architecture. This aspect may be considered, once it is identified that certain “cross-layer” markings make sense.

Based on this, it is suitable to use the basic packet trace structure as defined in Table 5.3-1 and that RAN1 can assume that RAN1 may have access to the above packet information. The P-Trace may be extended based on new findings.

For any system simulations, it is important that simulation are carried out only using the baseline parameters. Additional cross-layer parameters may be used as defined above. New packet-based markers may be added.

It is also considered that realization in the 5G System is only addressed once the benefits of an approach are identified.

5.4 5G System Model and Simulation

5.4.1 5G System Model

In order to further model the 5G system, an alignment with the 5GS QoS model as defined in clause 5.7 of TS 23.501 is considered appropriate. The interface between the application domain and the 5G System is assumed to be based on QoS Flows. A QoS Flow is the finest granularity of QoS differentiation in the PDU Session. A QoS Flow ID (QFI) is used to identify a QoS Flow in the 5G System. User Plane traffic with the same QFI within a PDU Session receives the same traffic forwarding treatment (e.g. scheduling, admission threshold).

The QFI is carried in an encapsulation header on N3 without any changes to the e2e packet header and is used to uniquely identify a PDU Session. The principle for classification and marking of User Plane traffic and mapping of QoS Flows to AN resources is illustrated in Figure 5.7.1.5-1 of TS 23.501 and repeated in Figure 5.1-2.

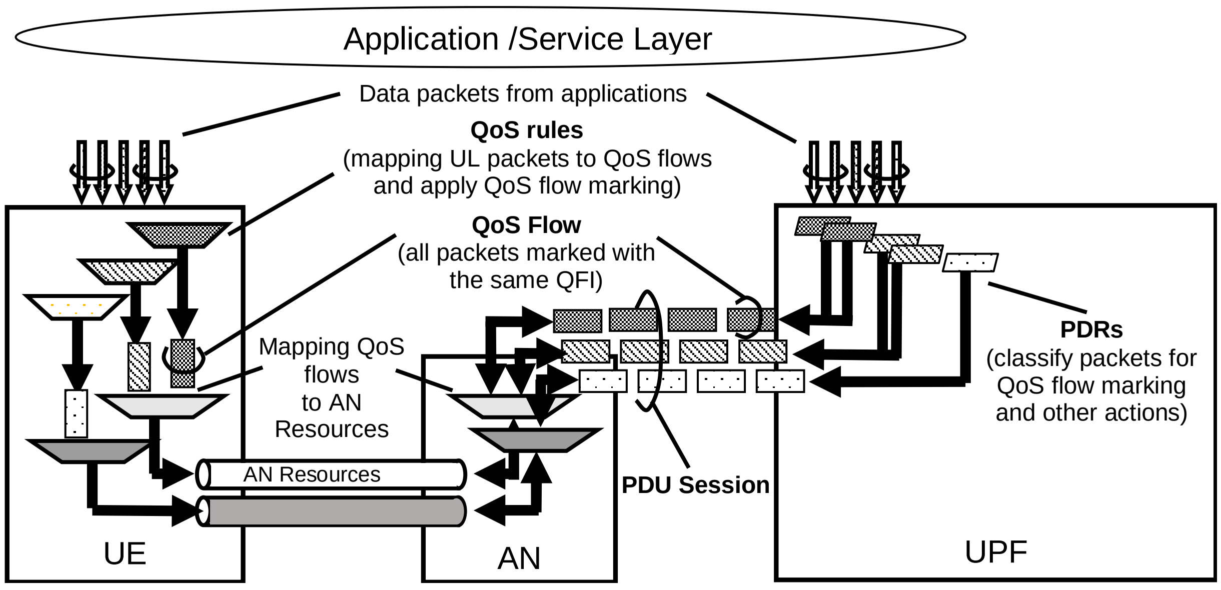

Application /Service Layer
Data packets from applications
QoS rules
(mapping UL packets to QoS flows and apply QoS flow marking)
QoS Flow
(all packets marked with the same QFI)
Mapping QoS flows to AN Resources
AN Resources
PDU Session
PDRs
(classify packets for QoS flow marking and other actions)
UE
AN
UPF

Figure 5.2.1-1: The principle for classification and User Plane marking for QoS Flows and mapping to AN Resources (see TS 23.501 [5.1i], Figure 5.7.1.5-1)

In the downlink, incoming data packets are classified by the UPF based on the Packet Filter Sets of the DL PDRs in the order of their precedence. The UPF conveys the classification of the User Plane traffic belonging to a QoS Flow through an N3 User Plane marking using a QFI. The AN binds QoS Flows to AN resources (i.e. Data Radio Bearers of in the case of 3GPP RAN).

In UL, for relevant PDU Session of Type IP, the UE evaluates UL packets against the UL Packet Filters in the Packet Filter Set in the QoS rules based on the precedence value of QoS rules in increasing order until a matching QoS rule is found. The UE uses the QFI in the corresponding matching QoS rule to bind the UL packet to a QoS Flow. The UE then binds QoS Flows to AN resources.

TS 23.501 defines the following QoS characteristics:

- Clause 5.7.3.2: Resource type (Non-GBR, GBR, Delay-critical GBR)
    - A GBR QoS Flow uses either the GBR resource type or the Delay-critical GBR resource type. The definition of PDB and PER are different for GBR and Delay-critical GBR resource types, and the MDBV parameter applies only to the Delay-critical GBR resource type.
    - A Non-GBR QoS Flow uses only the Non-GBR resource type.
- Clause 5.7.3.3: Priority Level;
    - The Priority Level associated with 5G QoS characteristics indicates a priority in scheduling resources among QoS Flows. The lowest Priority Level value corresponds to the highest priority.
- Clause 5.7.3.4: Packet Delay Budget (including Core Network Packet Delay Budget);
    - The Packet Delay Budget (PDB) defines an upper bound for the time that a packet may be delayed between the UE and the N6 termination point at the UPF. For a certain 5QI the value of the PDB is the same in UL and DL. In the case of 3GPP access, the PDB is used to support the configuration of scheduling and link layer functions (e.g., the setting of scheduling priority weights and HARQ target operating points).
    - For GBR QoS Flows using the Delay-critical resource type, a packet delayed more than PDB is counted as lost if the data burst is not exceeding the MDBV within the period of PDB and the QoS Flow is not exceeding the GFBR. For GBR QoS Flows with GBR resource type not exceeding GFBR, 98 percent of the packets shall not experience a delay exceeding the 5QI's PDB.
- Clause 5.7.3.5: Packet Error Rate;
    - The Packet Error Rate (PER) defines an upper bound for the rate of PDUs (e.g. IP packets) that have been processed by the sender of a link layer protocol (e.g. RLC in RAN of a 3GPP access) but that are not

successfully delivered by the corresponding receiver to the upper layer (e.g. PDCP in RAN of a 3GPP access).

  - Thus, the PER defines an upper bound for a rate of non-congestion related packet losses. The purpose of the PER is to allow for appropriate link layer protocol configurations (e.g. RLC and HARQ in RAN of a 3GPP access). For every 5QI the value of the PER is the same in UL and DL. For GBR QoS Flows with Delay-critical GBR resource type, a packet which is delayed more than PDB is counted as lost, and included in the PER unless the data burst is exceeding the MDBV within the period of PDB or the QoS Flow is exceeding the GFBR.

- Clause 5.7.3.6: Averaging window (for GBR and Delay-critical GBR resource type only);

  - Each GBR QoS Flow shall be associated with an Averaging window. The Averaging window represents the duration over which the GFBR and MFBR shall be calculated (e.g. in the (R)AN, UPF, UE).

- Clause 5.7.3.7: Maximum Data Burst Volume (for Delay-critical GBR resource type only).

  - Each GBR QoS Flow with Delay-critical resource type shall be associated with a Maximum Data Burst Volume (MDBV).

  - MDBV denotes the largest amount of data that the 5G-AN is required to serve within a period of 5G-AN PDB.

  - Every standardized 5QI (of Delay-critical GBR resource type) is associated with a default value for the MDBV (specified in QoS characteristics Table 5.7.4.1). The MDBV may also be signalled together with a standardized 5QI to the (R)AN, and if it is received, it shall be used instead of the default value.

  - The MDBV may also be signalled together with a pre-configured 5QI to the (R)AN, and if it is received, it shall be used instead of the pre-configured value

5.4.2 Core Network Model

The core network is expected to operate on constant bitrate channel from the UPF to the gNB.

In the model showing in Figure 5.4.2-1, the core network model gets as input a P-Trace and delivers a P-Trace. Packets provided to the core network are expected to be delivered from the UPF to the gNB at a constant bitrate of bitrate *R*, whereby each packet is delivered at a delivery time being the later of the two

1) The arrival time at the UPF

2) The delivery end time of the previous packet

The arrival time at the gNB is modelled as the sum of the delivery time and the packet size divided by the bitrate *R*.

The core network delivery is considered to be error-free.

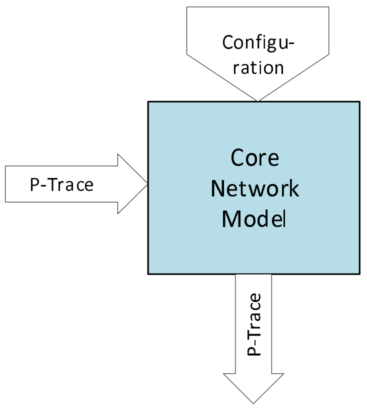


Figure 5.4.2-1 Core Network Model

The modelling of such a delay is for example show in Figure 5.4.2-2 for different packet size models.

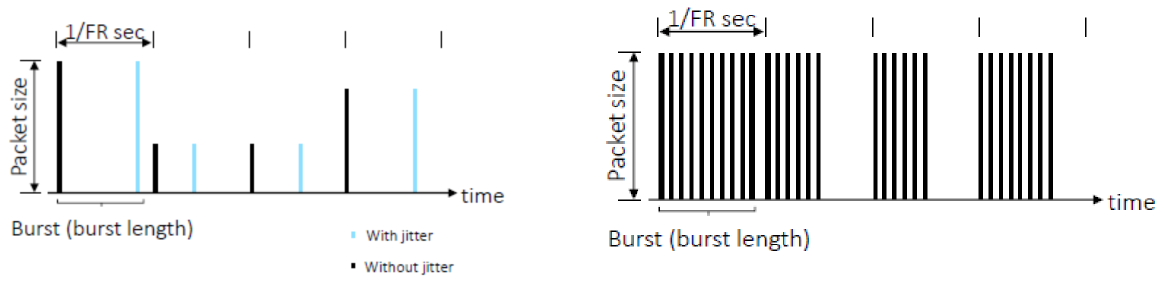


Figure 5.4.2-2 Packet size and latency addition in core network model

The configuration parameters for the core network model are

- the input packet trace
- the core network bitrate
- the output packet trace

5.4.3 RAN Model

The RAN delivery may be modelled by a function that receives P-Traces as input for each simulated user as shown in Figure 5.4.3-1 and produces a P'-Trace at the output for the primary user. The other users are only used from statistical competing traffic. Configuration parameters for the simulation are left to RAN experts.

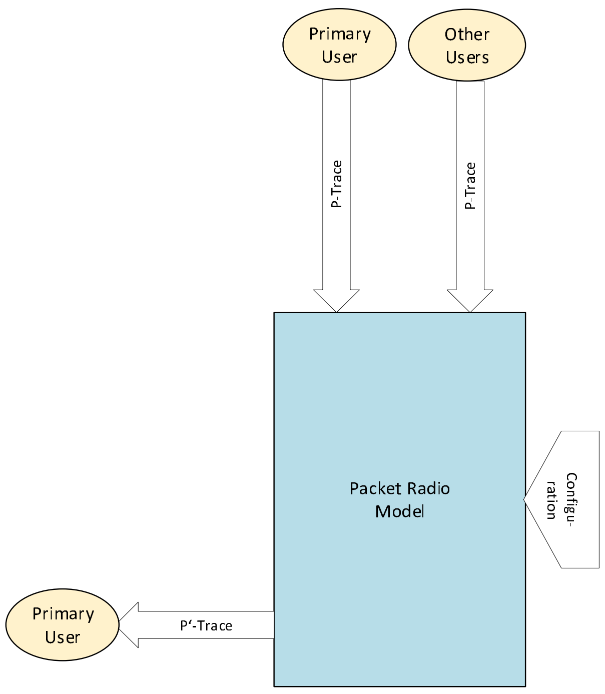


Figure 5.4.3-1 Packet Radio Network Model

However, the usage of traces for RAN simulations is complex and there is a preference to operate with statistical models that emulate the arrival time and size of the packets. The main reason is for example, that RAN1 needs to simulate multiple cells at the same time, for example 63, for interference considerations. A statistical model is preferred.

5.4.4 Test Channels

P-Traces as produced by XR applications may be used by RAN for complex RAN simulations. However, at the same time it is important to evaluate different XR application and encoding configurations that result in a P-Trace and then compare different settings. For this purpose, test channels are defined addressing two aspects:

- Permit to emulate typical radio conditions in terms delays and losses.
- Permit to evaluate the application quality for different representative radio conditions.

The test channel aligns with the 5G System and QoS Model as introduced in clause 5.4.1.

A test channel is defined by the following models:

- Packet Error Rate:
  - Definition: the rate of PDUs (e.g. IP packets) that have been processed by the sender of a link layer protocol (e.g. RLC in RAN of a 3GPP access) but that are not successfully delivered by the corresponding receiver to the upper layer (e.g. PDCP in RAN of a 3GPP access).
  - Model: iid loss model independent of the packet size with parameter *PLR*
- Packet Delay:
  - Definition: the time that a packet may be delayed between the UE and the N6 termination point at the UPF.
  - Model: iid distributed latency between 0 and a max_delay

Other parameters and modelling aspects for test channels are for further study.

5.5 Content Modelling

5.5.1 Introduction

In order to evaluate the traffic characteristics and the quality of XR applications, it is relevant to operate with realistic source data. Preferably, the traffic of existing services is evaluated, for example taking P-Traces from existing XR services. However, in doing so, there are limitations

1) in accessing the data
2) in terms of legal restrictions on providing such data
3) in terms of making use of such data for quality evaluation
4) in terms of understanding the structure and details of the contained traffic
5) in terms of understanding the options and impact of different system configurations
6) and for other reasons

Based on this, an approach is proposed in this report as shown in Figure 5.5.1-1.

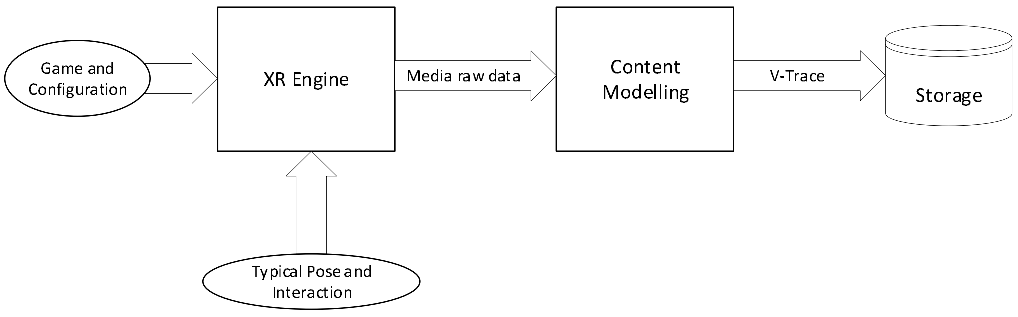


Figure 5.5.1-1 Content Modelling Approach

It is assumed that the raw media data is generated based on an XR/game session for one or several popular games, using typical pose traces and interactions, for example by a human playing the game. This information may be running over several minutes or even hours in order to create sufficiently representative statistics.

The output of the XR rendering engine is video representing texture, primitive and eye buffers.

For service and applications not relying on XR engines, other approaches may be taken. This is for further study and discussed as part of the specific traffic characteristics.

5.5.2 V-Trace Generation

The basic idea behind the generation of V-Traces is motivated by two main aspects:

1) Handling of several minutes or hours of raw data from a game engine is too significant. A statistical version is preferred in order to properly handle the amount of data
2) Representative games are typically attached with copyright and licensing terms and hence, the raw media data cannot be shared publicly.

Based on this V-Trace generation has been initiated by

- using a representative game
- using a repeatable pose and interaction trace for the game
- using a high-quality output of a game engine for each eye buffer, encoded for example in H.264(AVC)
- decode these video sequences and store this raw data for proper model encoding to generate a trace. As an example, the decoded video sequence is 2K x 2K at 120 fps for each source buffer.

A shorter version of the raw data is then encoded in with different parameters to create some statistical output that allows to estimate the performance of a video codec when used with different configurations. For this purpose, initially a set of encoding parameters are used to understand the impact of each of the parameters for the resulting bitrate and quality. This information is then used for the modelling.

Based on the above model findings, it considered to only generate two V-Traces for the entire sequence to keep data handling manageable as shown in Figure 5.5.2-1. One V-Trace is generated for Intra coding only, the other one is generated for predictive coding.

The V-Traces are combined to document relevant statistics for each frame.

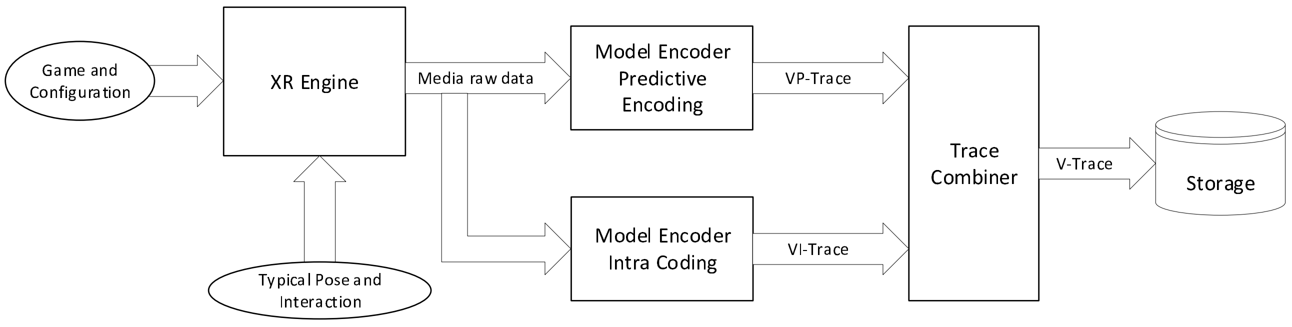


Figure 5.5.2-1 V-Trace Generation

For the purpose of this TR, a modelling is provided Annex A.

5.5.3 V-Trace Format

For each frame in the video sequence, the information as documented in Table 5.5.3-1 is provided.

Table 5.5.3-1 V-Trace Format

Name	Type	Semantics
time_stamp_in_micro_s	BIGINT	Associated rendering time of the frame
encode_order	BIGINT	The display order of the frames
i_qp	BIGINT	Quantization Parameter decided for the I frame.
i_bits	BIGINT	Number of bits consumed by the I frame.
i_y_psnr	BIGINT	Peak signal to noise ratio for Y planes in dB and multiplied by 1000 for I frame.
i_u_psnr	BIGINT	Peak signal to noise ratio for U planes in dB and multiplied by 1000 for I frame.
i_v_psnr	BIGINT	Peak signal to noise ratio for V planes in dB and multiplied by 1000 for I frame.
i_yuv_psnr	BIGINT	Peak signal to noise ratio for weighted Y, U and V planes in dB and multiplied by 1000 for I frame.
i_ssim	BIGINT	Quality metric that denotes the structural similarity between frames for I frame.
i_ssim_d_b	BIGINT	Quality metric that denotes the structural similarity between frames in dB for I frame.
i_total_frame_time_ms	BIGINT	Total time spent to encode the frame for I frame.
p_poc	BIGINT	The display order of the frames for P
p_qp	BIGINT	Quantization Parameter decided for the P frame.
p_bits	BIGINT	Number of bits consumed by the P frame.
p_y_psnr	BIGINT	Peak signal to noise ratio for Y planes in dB and multiplied by 1000 for P frame.
p_u_psnr	BIGINT	Peak signal to noise ratio for U planes in dB and multiplied by 1000 for P frame.
p_v_psnr	BIGINT	Peak signal to noise ratio for V planes in dB and multiplied by 1000 for P frame.
p_yuv_psnr	BIGINT	Peak signal to noise ratio for weighted Y, U and V planes in dB and multiplied by 1000 for P frame.
p_ssim	BIGINT	Quality metric that denotes the structural similarity between frames for P frame.
p_ssim_d_b	BIGINT	Quality metric that denotes the structural similarity between frames in dB for P frame.
p_total_frame_time_ms	BIGINT	Total time spent to encode the frame for P frame.
intra	DOUBLE PRECISION	Percentage of intra Coding Units in P frame
merge	DOUBLE PRECISION	Percentage of merge Coding Units in P frame
skip	DOUBLE PRECISION	Percentage of skip Coding Units in P frame
inter	DOUBLE PRECISION	Percentage of inter Coding Units in P frame

5.5.4 Other media types

For other media types, no dedicated trace format is defined.

5.6 Media Coding and Decoding Modelling

5.6.1 General

Media may be encoded in many different ways in order to meet application, service and delivery requirements. Whereas decoders in media coding are typically fully specified, there typically exist significant options in the encoding, in particular for video, but also in general, apply different encoding methods and functionalities.

These functionalities may impact:

1) Bitrate of the media stream, both in terms of short-term (for example over a window of 1 second) as well as average bitrate of the media stream. The bitrate may also be dynamically adjusted, for example based on content properties and/or network conditions. Changes in bitrate impact the quality of the media stream.

2) Delays in encoder or decoder due to processing or algorithmic functionalities. For example the time when captures/generated media sample hits encoder and the time when it leaves the decoder may be configurable. Permitting longer delays typically results in better quality.

3) Output data unit properties determining the size of the generated data units and packets. Adding constraints typically results in additional bitrates

4) Error resilience and random access includes functionalities in the media stream that support mitigation of potential losses of data. Adding such functionalities typically increases the bitrate or reduces the quality.

More details for video media type are provided in clause 5.6.2. A configuration overview for video is provided in clause 5.6.3. Trace formats are defined in clause 5.6.4.

Other media types are defined in clause 5.6.5.

5.6.2 Video Coding and Decoding

Figure 5.6.2-1 provides an overview of content encoding and decoding models with focus on video. Encoding makes use of V-Traces and based on configurations and possibly feedback from the decoder, encoding of individual media samples/frames is carried out. The encoder produces a sequence of slices according to the format defined in clause 5.6.3. After delivery, a content decoding model generates a sequence of V'-Traces that can be used by a Quality evaluation tool. The decoding model may also provide feedback to the encoder, for example on lost or late frames.

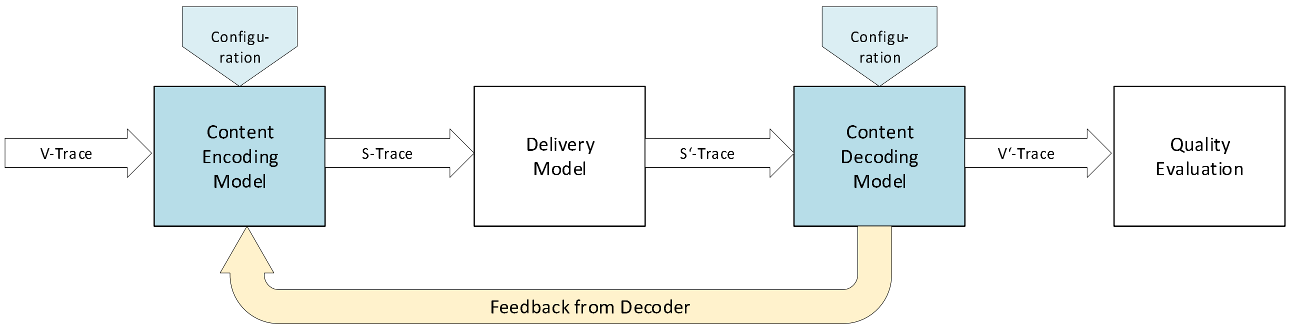


Figure 5.6.2-1 Media Coding and Decoding Modelling

As an example, rate control plays an important role in video coding, although it's not a normative tool for any video coding standard. In video communications, rate control ensures that the coded bitstream can be transmitted successfully and make full use of the limited bandwidth while maximizing the quality. The rate control not only determines the quality, it also determines the latency when for example delivering a video bitstream via constant bitrate link.

For video encoding, among others the following parameters may be considered:

- Codec in use, for example H.264/AVC or H.265/HEVC

- Encoding patterns:

    - Usage of B and P pictures
    - Latency settings: P pictures only, look-ahead units only 0 (for minimum latency), i.e. no future frames are taken into account in the encoding
- Rate control to be used, for example
    - CBR with bitrate
    - Capped VBR with bitrate
    - constant rate factor (CRF), i.e. constant quality with weighted rate adjustments
    - constant QP (CQP), i.e. constant quantization parameters, or
    - feedback-based using information on loss and delay statistics

  For a detailed modelling refer to clause 5.6.2.3
- Slice settings, examples
    - 1 slice per frame,
    - 1 slice per every 16 pixel row,
    - 8 slices per video frame
- Intra settings and error resilience:
    - Regular IDR Patterns, e.g. 16-th frame,
    - Regular Gradual Decoder Refresh Pattern,
    - adaptive Intra,
    - feedback based Intra,
    - feedback based predication and ACK-based,
    - feedback-based prediction and NACK based
- Complexity settings for encoders
- Pre-Encoding delays: generation timestamp of the rendered frame until this frame is encoded (add models)
    - constant delay
    - equally distributed between a min and max delay
    - truncated Gaussian model with mean, variance and maximum delay
- Encoding delays: the time of encoder from receiving the frame until a slice is produced (add models).
    - constant delay
    - equally distributed between a min and max delay
    - truncated Gaussian model with mean, variance and maximum delay.
- Handling of multiple video buffers, for example left and right eye
    - All buffers operate on the same sampling/render time with the same pre-encoding delay
    - Buffers are staggered, i.e. sampling/render time is uniformly distributed across the sampling rate.

Based on these settings, a slice trace S-Trace can be generated resulting in the format as defined clause 5.6.4. A slice delivery unit now uses a slice trace and provides as output an S’-Trace, for more details refer to clause 5.7. The delivery impacts the slice delivery in one or multiple of the following ways:

1) The slice is delayed, i.e. a delivery timestamp is assigned to the slice in the S'-Trace. This can be used to determine if the slice is received in time

2) The slice is corrupted, i.e. the slice is entirely, only a correct prefix of the slice available or the slice was fully received for the decoding model.

A decoding model now takes into account the S'-Trace as well as the V-Trace in order to generate a decoding model. For this, coding units in a video frame may be viewed individually and get assigned a state:

- A coding unit spanning a certain area of the video frame is *correct* if and only if it is received correctly and it predicts for a non-damaged coding unit. Predicting from non-damaged units means that both spatial prediction as well as temporal prediction coding units are correct

- If any of the two conditions do not hold, the coding unit state is *damaged*.

Note that only non-predicted coding units will eventually remove damaged areas. The resulting trace documenting correct and damaged coding units is referred to as V'-Trace, see clause 5.6.4 for more details.

An important factor for the quality of the video is the percentage of lost area.

A specific algorithm to model the generation of slices as well as the decoding of received slices is provided in Annex A.

5.6.3 Video Coding and Decoding Configuration

Detailed video encoding configuration parameters may be derived from the list of parameters provided in clause 5.6.2. An example encoding parameters are provided in Table 5.6.3-1. The below example sets the following parameters:

- Two source buffers, each defined by V-Traces, width, height, frame rates

- slice configuration mode, 8 rows are configured.

- Bitrate mode is configured to CBR, with parameters for the rate control

- Error resilience settings using 1 intra rotating intra slice for every frame, restricted to no error propagation, also referred to gradual decoder refresh (GDR).

- Parameters for encoder pre-delay and encoder processing delay

- staggering of buffers are set to be true, i.e. the frames of each buffer are sent with different presentation times.

Table 5.6.3-1 Example video encoding configuration

```
{
    "Buffers": [{
        "Source": {
            "V-Trace": "V-Trace-left.csv",
            "frame_width": 2048,
            "frame_height": 2048,
            "frame_rate": 60,
            "start_frame": 1,
            "total_frames": 3600,
            "buffer": "left"
        },
        "Source": {
            "V-Trace": "V-Trace-right.csv",
            "frame_width": 2048,
            "frame_height": 2048,
            "frame_rate": 60,
            "start_frame": 1,
            "total_frames": 3600,
            "buffer": "right"
        }
    }],
    "Slice": {
        "mode": "row",
        "parameter": "8"
    },
    "Bitrate": {
        "mode": "CBR",
        "bitrate": "10000000",
        "window_framerate": "1",
        "QPmin": 10,
        "QPmax": 25
    },
    "ErrorResilience": {
        "mode": "pIntra",
        "parameter": "1"
    },
    "PreDelay": [{
        "Delay": {
            "mode": "equally",
            "parameter1": "10",
            "parameter1": "30"
        }],
    "EncodingDelay": [{
        "Delay": {
            "mode": "GaussianTrunc",
            "parameter1": "8",
            "parameter2": "3"
            "parameter3": "1"
        }],
    },
    "bufferinterleaving": true,
    "S-Trace": "S-Trace.csv"
}
```

For video coding, each received S'-trace would be decoded independently and converted into a V'-Trace. The V'-Trace holds status of each coding unit it is determined if the coding unit is correct or damaged according to clause 5.6.2. No specific other configuration parameters are provided.

5.6.4 S-Trace and V'-Trace Format

For each generated slice, an S-Trace format is provided according to Table 5.6.4-1.

Table 5.6.4-1 S-Trace Format

Name	Type	Semantics
`index`	`BIGINT`	Unique index increased by 1 and indexing this row in the S-Trace file.
`time_stamp_in_micro_s`	`BIGINT`	Availability time of slice after encoder relative to start time 0 in microseconds.
`size`	`BIGINT`	Slice size in bytes.
`render_timing`	`BIGINT`	the rendering generation timing associated to the frame
`buffer`	`BIGINT`	The associated eye buffer 1=left 2=right In general, differentiates application traffic for different buffers, for example audio, video, left eye, right eye.
`frame_index`	`BIGINT`	Frame index to identify the frame buffer
`type`	`BIGINT`	The slice type 1=intra 2=inter
`importance`	`BIGINT`	assigned relative importance information (higher number means higher importance)
`start_address_cu`	`BIGINT`	start address of CU in slice
`number_cus`	`BIGINT`	total number of CUs in slice
`frame_file`	`STRING`	Reference to frame file containing information for each CU including index and eyebuffer

For each frame, an entry into a binary file is generated that documents the information according to Table 5.6.4-2.

Table 5.6.4-2 Frame-related metadata from encoding

Name	Type	Semantics
`address`	`BIGINT`	Address of CU in frame.

size	BIGINT	CU size in bits
mode	BIGINT	The mode of the CU 1=intra, 2=merge, 3=skip, 4=inter
reference	BIGINT	The reference frame of the CU 0 n/a, 1=previous, 2=2 in past, etc.
qpnew	BIGINT	the QP decided for the CU
psnr_y	BIGINT	the estimated Y-PSNR for the CU in dB multiplied by 1000
psnr_yuv	BIGINT	the estimated weighted YUV-PSNR for the CU dB multiplied by 1000

For each slice after delivery, on entry in the V'-Trace is provided according to the format provided in Table 5.6.4-3.

Table 5.6.4-3 V'-Trace Format

Name	Type	Semantics
time_stamp_in_micro_s	BIGINT	Availability time of frame after decoder relative to start time 0 in microseconds.
render_timing	BIGINT	the rendering generation timing associated to the frame
QP	BIGINT	Quantization Parameter decided for the frame.
bits	BIGINT	number of bits consumed by the frame.
psnr_y	BIGINT	the encoded Y-PSNR for the frame in dB multiplied by 1000
psnr_yuv	BIGINT	The encoded weighted YUV-PSNR for the frame in dB multiplied by 1000
rpsnr_y	BIGINT	the estimated recovered Y-PSNR for the frame in dB multiplied by 1000
rpsnr_yuv	BIGINT	the estimated weighted recovered YUV-PSNR for the frame in dB multiplied by 1000
total_CUs	BIGINT	Total number of CUs
correct_CUs	BIGINT	Number of correct CUs
lost_CUs	BIGINT	Number of lost CUs
damaged_CUs	BIGINT	Number of damaged CUs due to error propagation

5.6.5 Other media types

For other media types, no dedicated modelling is defined.

5.7 Content Delivery Modelling

5.7.1 General

Content delivery deals with the content delivery protocol and packetization, delay jitter, but possibly also more sophisticated models such as RTP retransmission, TCP operations and so on. Typically, an application data unit is received by the content delivery system and is then mapped to one or several transport packets including information on timing, etc. At the receiving end, the received packets are used to re-construct the application data units.

There may be network deployments that uniformly support the transport of packets with larger Maximum Transfer Unit (MTU) sizes (for example with ethernet jumbo frames of transport MTU size up to 9216 octets), but according to TS 23.501, clause 5.6.10.4 and Annex J, the typical link MTU size is 1358 bytes. This means that for typical network deployments the IP packets size larger than 1358 bytes will imply IP fragmentation.

The sending part of the content delivery can typically be modelled by at least the following parameters:

- Maximum Transfer Unit: This determines the maximum packet size of the application. Typically, an application data unit is segmented into multiple packets for transfer for which none of the packets exceeds the maximum transfer unit. The delivery This is addressed in clause 5.7.2

- Bitrate restrictions: the available bitrate from packager to UPF, i.e. from the content delivery sender to the 5G System.
- Potential retransmissions of packets. This is addressed in clause 5.7.3.
- Potentially adding forward error correction on application layer for one or several application data units. This is addressed in clause 5.7.4.

The receiving part of the content delivery can typically be modelled by at least the following parameters:

- Slice recovery strategy: This determines how a slice is recovered from lost and received packets assigned to a slice, for details for different configurations refer to clauses 5.7.2, 5.7.3 and 5.7.4.
- Maximum Latency: This determines the maximum permitted latency for a slice to be useful and if exceeded, the slice will be declared as lost.

A content delivery modelling is provided in Figure 5.7.1-1. An S-Trace is received by the content delivery model sender which converts the data to a packet trace (P-Trace) according to clause 5.3. After delivery through the 5G-System model, a P'-Trace is provided to the content delivery model sender which has the same format as the P-Trace. From the P'-Trace, an S'-Trace is reconstructed following the format defined in 5.7.5.

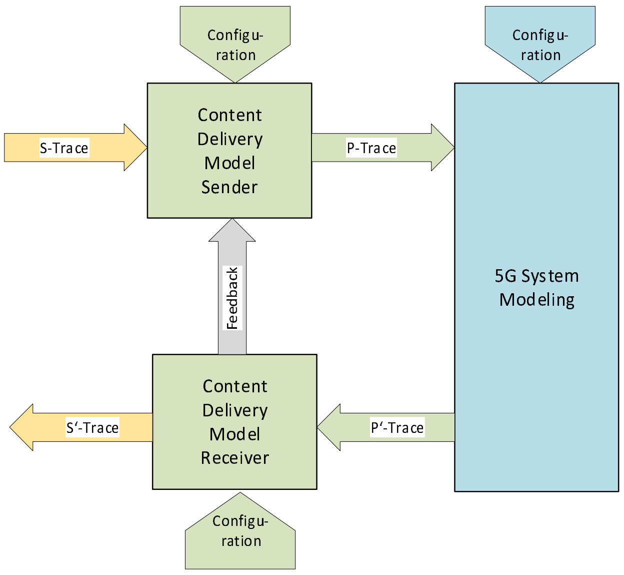


Figure 5.7.1-1 Content Delivery Modelling

5.7.2 Content Delivery Modelling for ADU Fragmentation

Payload formats of modern video codecs, in particular H.264/AVC and H.265/HEVC allow fragmentation of a single application data unit (ADU) into multiple transport units. Such a fragment of an ADU consists of an integer number of consecutive octets of that ADU unit. Each octet of the ADU is part of exactly one fragment of that ADU unit. Fragments of the same ADU are typically sent in consecutive order, for example with ascending RTP sequence numbers. Based on this approach, every ADU can be mapped to a packet transport stream for which a maximum transfer unit (MTU) of a specific size exist, for example something like 1500 bytes. In addition, mapping a ADU to an IP packet system, results in some overhead for the packet header, for IPv4 with RTP/UDP typically 40 bytes.

Secondly, the availability time of the generated packets to RAN layer depends on the delivery from the encoder to the UPF/RAN.

Based on this, from an S-Traces, a P-Trace can be generated based on the following parameters:

- MaxSize: maximum size of an output packet in bytes (0 means infinite)
- PacketOverhead: packet overhead added in bytes (default 40 bytes)

- Packet Delivery Bitrate: Provides the bitrate at which the packets are delivered from Encoder to RAN. Packets arrive this value divided by the size of the packet.

NOTE: assumes 1 hop with a bitrate as above. Consideration of multiple hops is not included.

Figure 5.7.2-1 provides an illustration on the content delivery modelling delays and packets based on the measurements presented in Annex B. A video frame may be delayed by the encoder. ADUs/slices are produced and are delayed by the encoder. The resulting packets are fragmented and then delivered, possibly delayed by the connection from the encoder to the UPF/RAN.

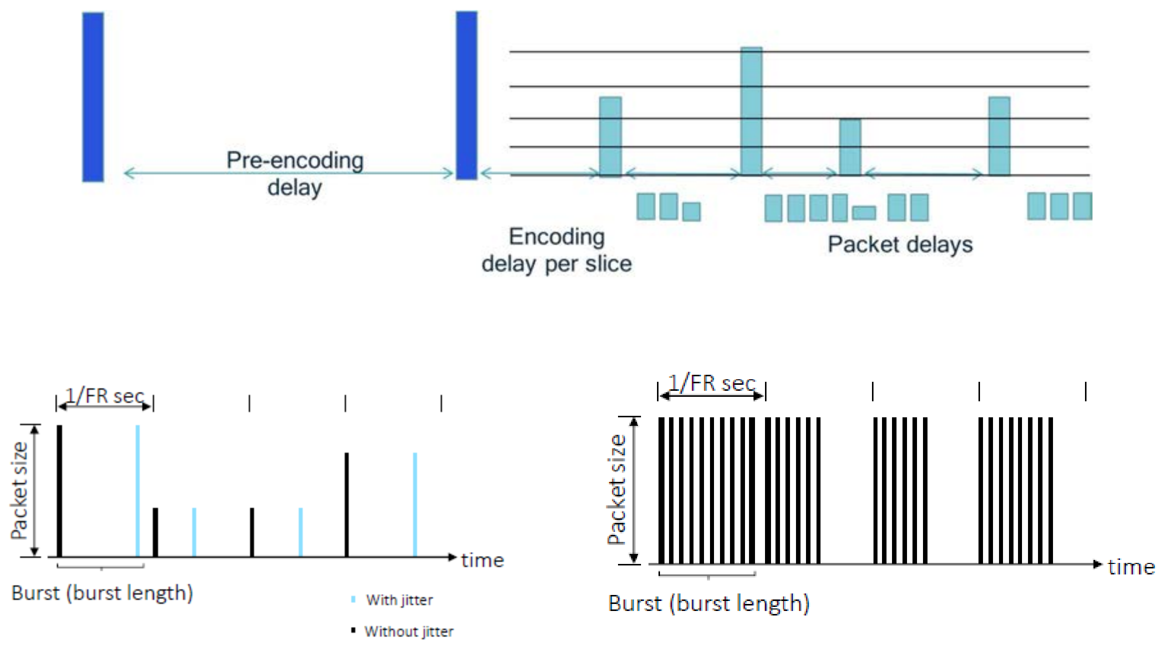


Figure 5.7.2-1 Packet sizes and delays

A possible configuration is provided in Table 5.7.2-1. In this case the trace data from an S-Trace is used and ADU fragmentation and packetization is done using a maxSize of 1468 and an overhead of 40. The access bitrate from encoder to RAN is assumed to be 10 Mbit/s and may result in packet delays when delivered.

Table 5.7.2-1 Example Content Delivery sender configuration

```
{
    "S-Trace": {
        "source": "S-Trace.csv",
        "startTime": 0
    },
    "Packet": {
        "maxSize": "1468",
        "overhead": "40"
    },
    "Bitrate": "10000000",
    "P-Trace": "P-Trace.csv"
}
```

At the receiving end the packets assigned to an ADU are recovered to reconstruct an ADU. For the case of ADU fragmentation, the ADU is reconstructed by removing the header from each packet and concatenating the fragment of the ADUs in sequence order. Only when all packets associated to the ADU are available, the ADU is available to the next processing unit, i.e. the timestamp is determined by the last received packet associated to the ADU.

If one or more packets associated to the ADU are lost, then the timestamp of the loss is the time at which the first lost packet is detected. However, as in order delivery cannot be assumed, a maximum delay of an ADU needs to be set, typically compared to the render time, after which only received packets are processed as part of the ADU. For the potential recovery of the ADU one of two different modes of recovery apply:

- ADU loss: If one of the packets associated to the ADU is lost, the entire ADU is lost.
- Suffix loss: If one packet is lost, then only the information from packets are earlier in the concatenated ADU are used to generate a partially received ADU.

In addition, a maximum ADU delay is set after which the data in the ADU compared to the render time is no longer helpful and the ADU is discarded.

A possible configuration is provided in Table 5.7.2-2. In this case the trace data from an P'-Trace is analysed and all buffer S'-Traces are generated. The loss mode is set to ADU to indicate that any lost fragment results in a loss of the entire ADU. ADUs not recovered within the maxDelay of 60 are considered as lost.

Table 5.7.2-2 Example Content Delivery receiver configuration

```
{
    "Input": {
        "Pp-Trace": "Pp-Trace-1.csv",
    },
    "Output": [{
        "Sp-Trace": {
             "buffer": "left",
             "Sp-Trace": "Sp-Trace-left.csv",
             "maxDelay": 60,
             "lossMode": "ADU"
          },
        "Sp-Trace": {
             "buffer": "right",
             "Sp-Trace": "Sp-Trace-right.csv"
             "maxDelay": 60,
             "lossMode": "ADU"
          }
    }]
}
```

5.7.3 Content Delivery Modelling for Retransmission

This work is for further study.

5.7.4 Content Delivery Modelling for Application Layer FEC

Commercial XR split rendering and cloud gaming services use Application Layer Forward Error Correction (FEC).

For example, Nvidia CloudXR™ supports FEC as indicated here https://web.archive.org/web/20240129165957/https:/forums.developer.nvidia.com/t/possible-to-configure-tune-cloudxr-encoding/208977/3. Other cloud gaming and XR services also report about the use of FEC.

In the following a possible implementation for application layer FEC assuming the system model for XR traffic (see Figure 5.2.1-1) is described. Commercially available XR split rendering and cloud gaming services as introduced above follow the same or at least similar principles.

In this case, the Application Data Units (ADUs) are not sent directly to the network, but they are added to a source block that then generates packets of basically equal size in order to then distribute the content. The basic concept is shown in Figure 5.7.4-1.

Each Application Data Unit (for example a video frame, or an object) has assigned a size F and additional properties, for example the type of the ADU, its importance, its delay constraints and so on. The properties are typically different for each ADU. Each ADU forms a source block with K encoding symbols, each of size T. Typically, the number of K is different for each ADU in a sequence of ADUs. Each of the initial K encoding symbol forms the payload of K source packets, whereby each packet may include some of the properties, and includes the source block size K as well as the encoding symbol id (ESI). The size of the object, F, may be carried as part of the source block as shown in Figure 1. In addition to K source packets, N-K repair packets may be sent as part of this ADU. The repair packets would be assigned to the same ADU, for example using a unique Transport Object Identifier (TOI) for ADU.

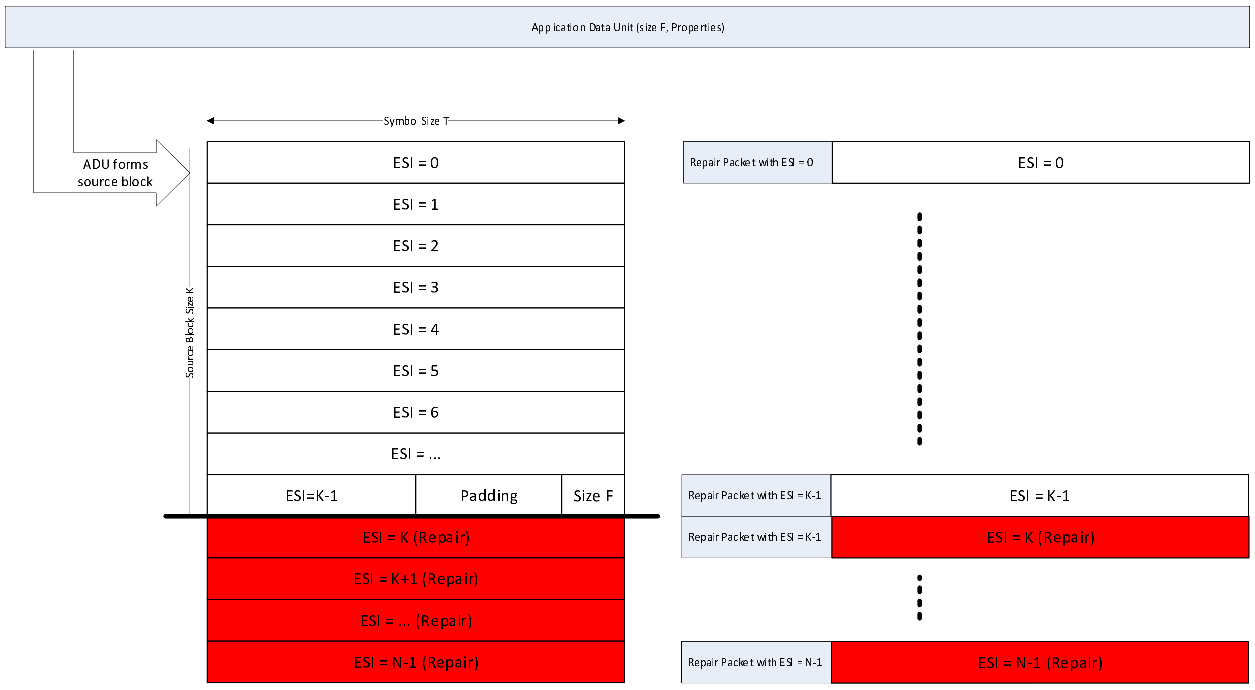


Figure 5.7.4-1 Packet Generation for Application Layer FEC

At the receiving end, assuming that the code is maximum distance separable (MDS) as the case for RaptorQ or Reed-Solomon codes, i.e. K out of the N packets are sufficient to recover the ADU, the receiver collects K symbols, determines the symbol size T based on the payload size, applies FEC decoding, recovers the source block, reads the size F from the K-th source symbols and recovers the ADU for the next layer in the protocol stack.

Such a system is aligned with Forward Error Correction (FEC) Building Block as defined in RFC 5052 [7].

The FEC Payload ID (i.e. the information carried in every packet header), essentially only requires carrying the encoding symbol ID and the source block size K. As an example, for RaptorQ as defined in RFC 6330 [8], the maximum source block size is 56403, i.e. 16 bits are sufficient. It is also expected that to signal the ESI, 1 or 2 bytes would be sufficient for most applications. In addition, a TOI may be carried, again using 1 or 2 bytes. While the above FEC configuration only serves as one reference, it may be considered as typical implementation.

There are several FEC schemes defined in the IETF that either permit or require that the RTP source packets are sent unmodified or are sent to be compatible with the payload format they comply to. This restriction for example applies to IETF RFC 8627 [11], the Flexible Forward Error Correction (FEC). In a similar fashion, for Raptor FEC in the context of FECFRAME as defined in IETF RFC 6681 [10], for example for the single sequenced flow in clause 8 of IETF RFC 6681 [10], the source packets are unmodified. In these cases, the repair packets need to contain sufficient information to form the source block from a sequence of unmodified source packets. For example, IETF RFC 6681 [10] adds to every repair packet the initial sequence number (of the RTP source packet that is included in the source block), the source block length and the encoding symbol ID, summing up to 48 bit in total. FEC encoding ID, encoding symbol size and maximum source block length are signalled as part of the SDP. The information in the SDP and in the repair packets allows the receiver to add each received source packet to the appropriate ESI in the source block at the receiver. Once decoded, the information in the source block (the size F), can be used to recover the length of the included packet.

Based on the information in IETF RFC 6681 [10], for an RTP based delivery for which source packets are unmodified, only the repair packets add the headers. To obtain the source block number a source packet belongs to as well as the source block length of this source block, at least one repair packet needs to be received.

An example sender configuration for FEC is provided in Table 5.7.4-1

Table 5.7.4-1 Example Content Delivery sender configuration for Application Layer FEC

```
{
    "S-Trace": {
        "source": "S-Trace.csv",
        "startTime": 0
    },
    "FEC": {
        "symbolSize": "1468",
        "packet-overhead": "46",
        "fec-overhead-percent": "30"
    },
    "Bitrate": "10000000",
    "P-Trace": "P-Trace.csv"
}
```

At the receiver, if one or more packets associated to the ADU with are lost, then the timestamp of the loss is the time at which the first lost packet is detected. However, as in-order delivery cannot be assumed, a maximum delay of an ADU needs to be set, typically compared to the render time, after which only received packets are processed as part of the ADU. If at least K packets are received for an ADU within the time budget, the ADU can be fully recovered. If less than K packets are received, the ADU cannot be recovered and one of the following two error handling modes can be configured:

- ADU loss: If more than N-K of the packets associated to the ADU are lost, the entire ADU is lost.
- Suffix loss: If more than N-K of the packets associated to the ADU are lost, then only the correct prefix preceding the first loss of a source packet is used to generate a partially received ADU.

5.7.5 S’-Trace Format

The S’-Trace format after the content delivery modelling receiver is shown in Table 5.7.4-3.

Table 5.7.4-3 S’-Trace Format

Name	Type	Semantics
index	BIGINT	Unique slice index increased by 1 and indexing this row in the S-Trace file.
time_stamp_in_micro_s	BIGINT	Availability time of slice at decoder relative to start time 0 in microseconds.
recovery_position	BIGINT	Recovered unit position (0 => lost, in between, full)
size	BIGINT	Original size of slice/data unit in bytes.
render_timing	BIGINT	the rendering generation timing associated to the frame
start_address_cu	BIGINT	start address of CU in slice
number_cus	BIGINT	total number of CUs in slice
frame_file	STRING	Reference to frame file containing information for each CU

5.8 Quality Metrics and Computation

5.8.1 General

Quality Evaluation is based on two aspects, namely the encoding quality and the quality degradation due to lost or late data. An evaluation framework is shown in Figure 5.8-1. The quality evaluation takes into account information from

- P’-Traces
- S’-Traces

- V'-Traces

For each user and each buffer, the quality is provided individually following the format of a Quality Trace Q-Trace as shown in Figure 5.8.1-1.

Table 5.8.1-1 Q-Trace Format

Name	Type	Semantics
user	BIGINT	User id.
buffer	BIGINT	the buffer information
type	TEXT	video, audio or data
total_packets	BIGINT	Total amount of packets that have been sent
duration	BIGINT	Duration of the simulation in ms
PLoR	DOUBLE PRECISION	Packet loss rate
PLaR	DOUBLE PRECISION	Packet late rate
SLR	DOUBLE PRECISION	Slice loss rate for video, if not present, n/a
CAR	DOUBLE PRECISION	Correct Area rate for video, if not present, n/a
ALR	DOUBLE PRECISION	Area loss rate for video, if not present, n/a
DAR	DOUBLE PRECISION	Area damage rate for video, if not present, n/a
LDR	DOUBLE PRECISION	Area loss and damage rate for video, if not present, n/a
PSNR	DOUBLE PRECISION	Average encoded PSNR for video, if not present, n/a
PSNRyuv	DOUBLE PRECISION	Average encoded PSNRyuv for video, if not present, n/a
RPSNR	DOUBLE PRECISION	Average recovered PSNR for video, if not present, n/a
RPSNRyuv	DOUBLE PRECISION	Average recovered PSNRyuv for video, if not present, n/a sum_frame (sum_cu PSNR[frame][cu] * indicator(correct[frame][cu]))/total_CU[frame] with indicator(condition) = 1 if condition is TRUE and 0 if condition is wrong
MOS	DOUBLE PRECISION	Average audio MOS for audio, if not present, n/a

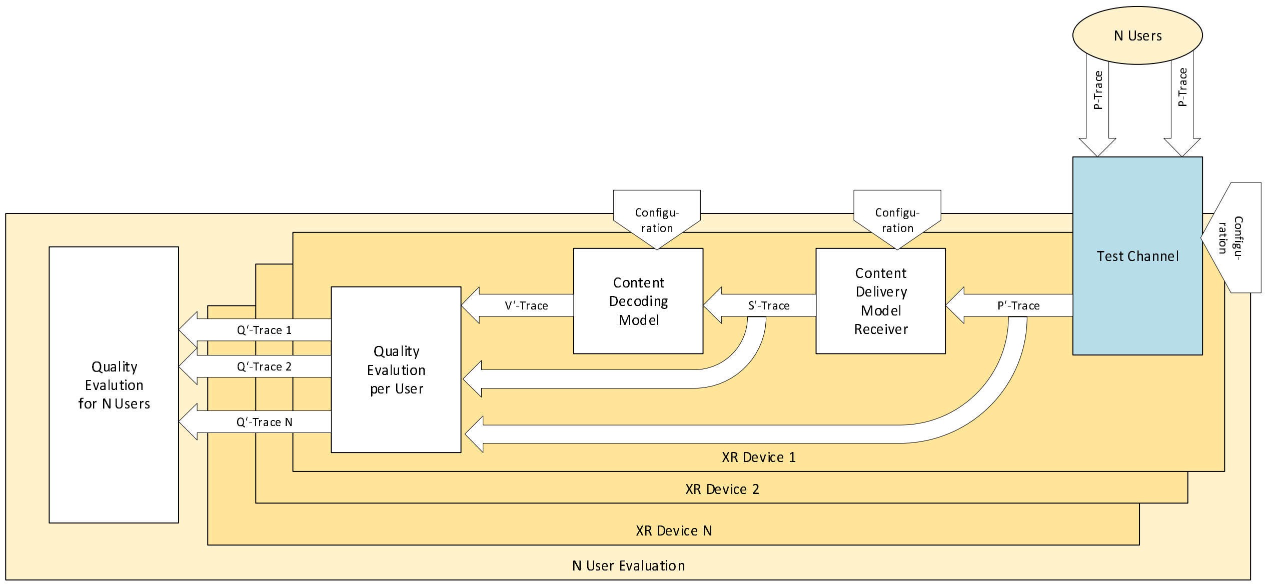


Figure 5.8-1 Packet sizes and delays

5.8.2 Test Channels

In order to evaluate the performance of certain configurations, test channel are defined and applied to P-Traces. The test channel address two aspects:

- Permit to emulate typical radio conditions in terms delays and losses.
- Permit to evaluate the application quality for different representative radio conditions for different different application configurations.

The test channel is aligned with the QoS Model as defined in TS 23.501, clause 5.7. Note that TS 23.501 defines the following QoS characteristics:

- Clause 5.7.3.2: Resource type (Non-GBR, GBR, Delay-critical GBR)
  - A GBR QoS Flow uses either the GBR resource type or the Delay-critical GBR resource type. The definition of PDB and PER are different for GBR and Delay-critical GBR resource types, and the MDBV parameter applies only to the Delay-critical GBR resource type.
  - A Non-GBR QoS Flow uses only the Non-GBR resource type.
- Clause 5.7.3.3: Priority Level: The Priority Level associated with 5G QoS characteristics indicates a priority in scheduling resources among QoS Flows. The lowest Priority Level value corresponds to the highest priority.
- Clause 5.7.3.4: Packet Delay Budget (including Core Network Packet Delay Budget):
  - The Packet Delay Budget (PDB) defines an upper bound for the time that a packet may be delayed between the UE and the N6 termination point at the UPF. For a certain 5QI the value of the PDB is the same in UL and DL. In the case of 3GPP access, the PDB is used to support the configuration of scheduling and link layer functions (e.g., the setting of scheduling priority weights and HARQ target operating points).
  - For GBR QoS Flows using the Delay-critical resource type, a packet delayed more than PDB is counted as lost if the data burst is not exceeding the MDBV within the period of PDB and the QoS Flow is not exceeding the GFBR. For GBR QoS Flows with GBR resource type not exceeding GFBR, 98 percent of the packets shall not experience a delay exceeding the 5QI's PDB.
- Clause 5.7.3.5: Packet Error Rate;
  - The Packet Error Rate (PER) defines an upper bound for the rate of PDUs (e.g. IP packets) that have been processed by the sender of a link layer protocol (e.g. RLC in RAN of a 3GPP access) but that are not successfully delivered by the corresponding receiver to the upper layer (e.g. PDCP in RAN of a 3GPP access).
  - Thus, the PER defines an upper bound for a rate of non-congestion related packet losses. The purpose of the PER is to allow for appropriate link layer protocol configurations (e.g. RLC and HARQ in RAN of a 3GPP access). For every 5QI the value of the PER is the same in UL and DL. For GBR QoS Flows with Delay-critical GBR resource type, a packet which is delayed more than PDB is counted as lost, and included in the PER unless the data burst is exceeding the MDBV within the period of PDB or the QoS Flow is exceeding the GFBR.
- Clause 5.7.3.6: Averaging window (for GBR and Delay-critical GBR resource type only): Each GBR QoS Flow shall be associated with an Averaging window. The Averaging window represents the duration over which the GFBR and MFBR shall be calculated (e.g. in the (R)AN, UPF, UE).
- Clause 5.7.3.7: Maximum Data Burst Volume (for Delay-critical GBR resource type only):
  - Each GBR QoS Flow with Delay-critical resource type shall be associated with a Maximum Data Burst Volume (MDBV).
  - MDBV denotes the largest amount of data that the 5G-AN is required to serve within a period of 5G-AN PDB.
  - Every standardized 5QI (of Delay-critical GBR resource type) is associated with a default value for the MDBV (specified in QoS characteristics Table 5.7.4.1). The MDBV may also be signalled together with a standardized 5QI to the (R)AN, and if it is received, it shall be used instead of the default value.

- The MDBV may also be signalled together with a pre-configured 5QI to the (R)AN, and if it is received, it shall be used instead of the pre-configured value

For the initial version of a test channel, the following model and parameters are defined:

1) Packet Error Rate:
    a. Parameters: Packet Error Rate
    b. Model: iid losses independent of the packet size
2) Packet delays:
    a. Parameter: max_delay
    b. Model: iid distributed latency between 0 and a max_value

Additional aspects of the QoS model and the RAN model are for further study.

5.9 Uplink Modeling

5.9.1 Considerations

In several scenarios, only pose and action data is sent in the uplink channel. Pose information may be used by the edge/cloud for network rendering. In general, the sampling rates are controlled by the application for which the pose information is applied. The more frequently the pose information is captured, the more accurate the rendered scenes and gaming control are. According to TR 26.928, in order to always be able to respond to the latest XR Viewer Pose, tracking needs to be done frequently and the minimum update rates should be 1000Hz and beyond. However, this applies to the hardware tracking of the device. Hence, an XR application can continuously query the XR Runtime (for example using an OpenXR API) to provide the viewer pose for a particular display time. This time is typically the target display time for a frame to be rendered. Repeatedly querying the pose for the same display time may not necessarily return the same result. Instead, the pose prediction gets increasingly accurate as the function is called closer to the given time for which a prediction is made. The application may also query the XR runtime for the predicted pose at different display times.

In case the pose is used for pre-rendering in the network (edge/cloud), an accurate and most recent pose information is preferable. There is a tradeoff between how often the latest pose is sent and whether it is sent for only one predicted display time or several consecutive display times. As a first estimate it can be assumed that sending a viewer pose aligned with the frame rate of the rendered video may be sufficient, for example at 60fps. The size of such information is typically 32 bytes per pose, and with several poses sent and header overhead, it may be up to few 100 bytes in a single flow. With such assumption the mapping to bitrates, periodicity and PDB is straightforward.

In addition to pose information, media uplink may be sent, for example data from cameras. In this case, a similar modelling as for downlink media applies.

5.9.2 Pose Uplink Model

According to TR 26.928, clause 4.1.3, the following applies:

> To maintain a reliable registration of the virtual world with the real world as well as to ensure accurate tracking of the XR Viewer pose, XR applications require highly accurate, low-latency tracking of the device at about 1kHz sampling frequency. An XR Viewer Pose consists of the orientation (for example, 4 floating point values in OpenXR) and the position (for example, 3 floating point values in OpenXR). In addition, the XR Viewer Pose needs to have assigned a time stamp. The size of a XR Viewer Pose associated to time typically results in packets of size in the range of 30-100 bytes, such that the generated data is around several hundred kbit/s if delivered over the network.

In addition, according to clause 5.9.1, an application permits to control the uplink pose information sending.

Based on these considerations, the following is proposed for the uplink modelling:

1) XR Viewer Pose information is provided to the application in 1kHz sampling frequency

2) The size of the XR Viewer Pose packets are considered 100 byte (60 payload, 40 header)

3) The Pose is sent with a frequency of controlled by the application, but we assume alignment with the rendering frame rate, i.e. 10 ms.

In order to minimize the rendering impact, the expected latency of the pose information is at most 10ms in the uplink. This means that the content is rendered with a pose of typically 10-15ms age.

6 XR Split Rendering

6.1 Introduction

Raster-based split rendering was introduced in TR 26.928 as an important scenario in order to serve end devices with restricted capabilities.

6.2 Reference System Design

6.2.1 Overview

The system design for split rendering follows the discussion and requirements from TR26.928 [3], clause 6.2.5. The architecture us shown in Figure 6.2.1-1.

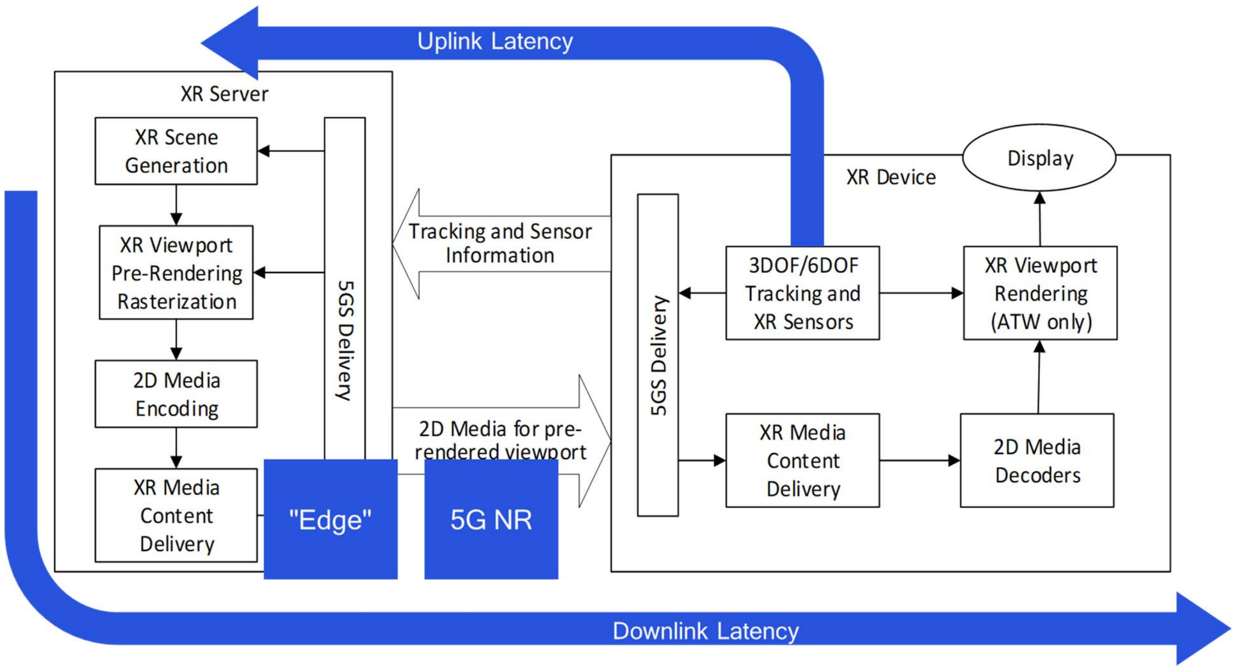

Uplink Latency
XR Server
XR Scene Generation
XR Viewport Pre-Rendering Rasterization
2D Media Encoding
XR Media Content Delivery
5GS Delivery
Tracking and Sensor Information
2D Media for pre-rendered viewport
"Edge"
5G NR
XR Device
Display
5GS Delivery
3DOF/6DOF Tracking and XR Sensors
XR Viewport Rendering (ATW only)
XR Media Content Delivery
2D Media Decoders
Downlink Latency

Figure 6.2.1-1 Split Rendering with Asynchronous Time Warping (ATW) Correction

Raster-based split rendering refers to the case where the XR Server runs an XR engine to generate the XR Scene based on information coming from an XR device. The XR Server rasterizes the XR viewport and does XR pre-rendering.

According to Figure 6.2.1-1, the viewport is pre-dominantly rendered in the XR server, but the device is able to do latest pose correction, for example by asynchronous time-warping (see clause 4.1 of TR26.928) or other XR pose correction to address changes in the pose.

- XR graphics workload is split into rendering workload on a powerful XR server (in the cloud or the edge) and pose correction (such as ATW) on the XR device

- Low motion-to-photon latency is preserved via on device Asynchronous Time Warping (ATW) or other pose correction methods.

The following call flow highlights the key steps:

1. An XR Device connects to the network and joins XR application
    a) Sends static device information and capabilities (supported decoders, viewport)
2. Based on this information, the XR server sets up encoders and formats
3. Loop
    a) XR Device collects XR pose (or a predicted XR pose)
    b) XR Pose is sent to XR Serve
    c) The XR Server uses the pose to pre-render the XR viewport
    d) XR Viewport is encoded with 2D media encoders
    e) The compressed media is sent to XR device along with XR pose that it was rendered for
    f) The XR device decompresses video
    g) The XR device uses the XR pose provided with the video frame and the actual XR pose for an improved prediction using and to correct the local pose, e.g. using ATW.

According to TR 26.928, clause 4.2.2, the relevant processing and delay components are summarized as follows:

- User interaction delay is defined as the time duration between the moment at which a user action is initiated and the time such an action is taken into account by the content creation engine. In the context of gaming, this is the time between the moment the user interacts with the game and the moment at which the game engine processes such a player response.
- Age of content is defined as the time duration between the moment a content is created and the time it is presented to the user. In the context of gaming, this is the time between the creation of a video frame by the game engine and the time at which the frame is finally presented to the player.

The **roundtrip interaction delay** is therefore the sum of the *Age of Content* and the *User Interaction Delay*. If part of the rendering is done on an XR server and the service produces a frame buffer as rendering result of the state of the content, then for raster-based split rendering, the following processes contribute to such a delay:

- User Interaction Delay (Pose and other interactions)
    - capture of user interaction in game client,
    - delivery of user interaction to the game engine, i.e. to the server (aka network delay),
    - processing of user interaction by the game engine/server,
- Age of Content
    - creation of one or several video buffers (e.g. one for each eye) by the game engine/server,
    - encoding of the video buffers into a video stream frame,
    - delivery of the video frame to the game client (a.k.a. network delay),
    - decoding of the video frame by the game client,
    - presentation of the video frame to the user (a.k.a. framerate delay).

As ATW is applied the motion-to-photon latency requirements (of at most 20 ms) are met by XR device internal processing. What determines the network requirements for split rendering is time of pose-to-render-to-photon and the roundtrip interaction delay. According to clause TR 26.928, clause 4.5, the permitted downlink latency is typically 50-60ms.

6.2.2 Considered Content Formats

Rasterized 3D scenes available in frame buffers (see clause 4.4 or TR 26.928 [3]) are provided by the XR engine and need to be encoded, distributed and decoded. According to TR 26.928, clause 4.2.1, relevant formats for frame buffers are 2k by 2k per eye, potentially even higher. Frame rates are expected to be at least 60fps, potentially higher up to 90 fps. The formats of frame buffers are regular texture video signals that are then directly rendered. As the processing is graphics centric, formats beyond commonly used 4:2:0 signals and YUV signals may be considered.

In practical considerations, the NVIDIA Encoding functions may be used. The parameters of such an encoder are documented here https://developer.nvidia.com/nvidia-video-codec-sdk.

6.2.3 Considered System Parameters

Based on the discussion on clause 6.2.1 and 6.2.2, several parameters are relevant for the overall system design.

- Game: Type of game, state of game, multi-user actions, etc.
- User Interaction:
    - 6DOF pose based on head and body movement,
    - Game interactions by controllers
- Formats of rasterized video signal. Typical parameters are:
    - 1.5K x 1.5K per eye at 60, 90, 120fps
    - 2K x 2K at 60, 90, 120fps
    - YUV 4:2:0 or 4:4:4
- Encoder configuration
    - Codec: H.264/AVC or H.265/HEVC
    - Bitrate: Bitrate setting to a specific value (e.g. 50 Mbit/s)
    - Rate control: CBR, Capped VBR, Feedback based, CRF, QP
    - Slice settings: 1 per frame, 1 per MB row, X per frame
    - Intra settings and error resilience: Regular IDR, GDR Pattern, adaptive Intra, feedback based Intra, feedback based predication and ACK-based, feedback-based prediction and NACK based
    - Latency settings: P pictures only, look-ahead units
    - Complexity settings for encoder
- Content Delivery
    - Sending of eye buffers
        - At the same time
        - staggered
    - Slice to IP mapping: Fragmentation
    - RTP-based time codes and packet numbering
    - RTP/RTCP-based feedback ACK/NACK
    - RTP/RTCP-based feedback on bitrate
- 5G System/RAN Configuration:
    - QoS Settings (5QI): GBR, Latency, Loss Rate

    - HARQ transmissions, scheduling, etc.
- Content Delivery Receiver configuration:
    - Loss Detection: sequence numbers
    - Delay/Latency handling
    - Error Resilience
    - ATW
- Quality Aspects
    - Video quality (encoded)
    - Lost data
    - Immersiveness

6.3 Simulation System

Figure 6.3-1 shows the simulation system used for XR split rendering based on the architecture in Figure 5.2.1-1. The following instantiation is provided:

- Two S-Traces are generated to address a video output for each eye buffer
- No feedback on video encoding or content delivery is considered

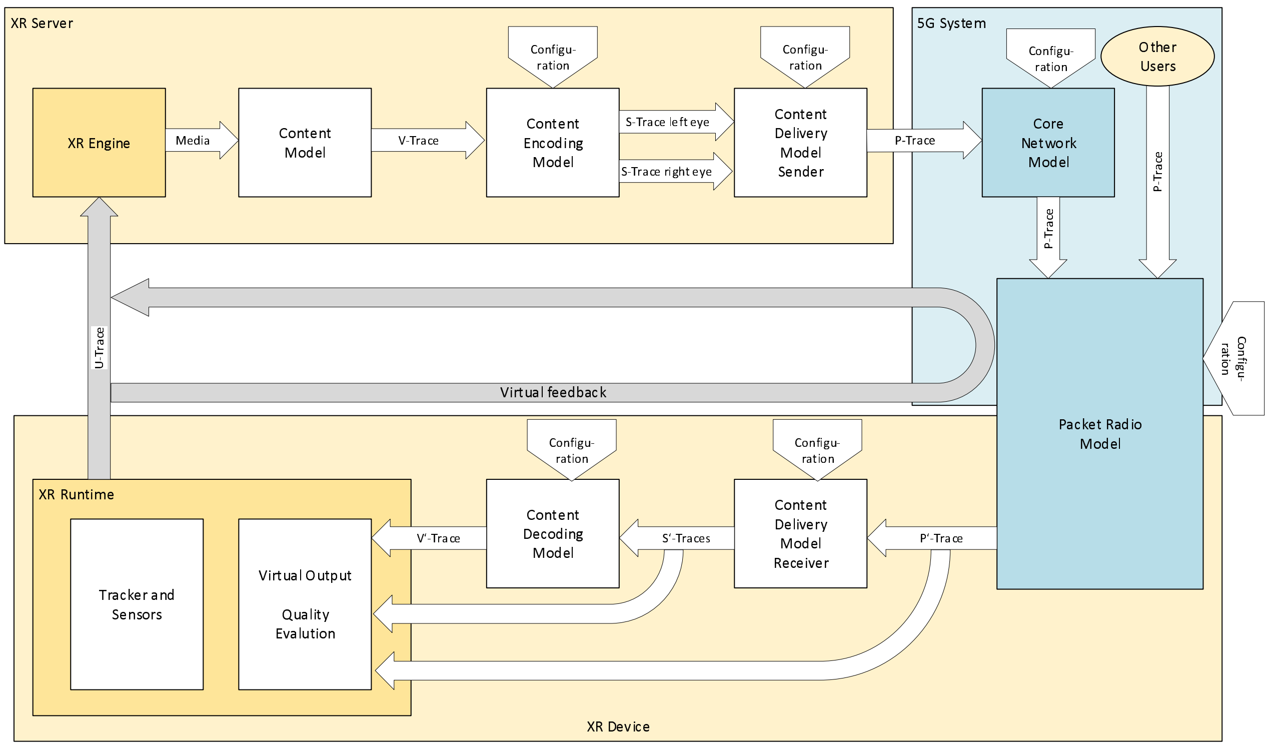


Figure 6.3-1 XR Split Rendering Simulation System

6.4 Recommended Configurations

The following parameters are recommended and fixed for the simulartion:

1) Content Model:

   a. Rendered scene output with 2 eye buffers at 2Kx2K at 60 fps, 8bit.
   b. Content and Trace Preview is here: http://dash.akamaized.net/WAVE/3GPP/XRTraffic/Traces/Qualcomm-VR2
   c. No audio

NOTE: Audio Modeling may be added by providing another P-Trace for audio, for example every 20ms a packet is provided according to an AMR/EVS/AAC model. Typical bitrates are in the range of 15-80 kbit/s which is negligible compared to video.

2) Encoding Model
   a. Encoding Models according to clause 5.5.
   b. HEVC, target bitrate 30 Mbit/s or 45 Mbit/s (CBR, capped VBR), equally split across eye buffers, independently encoded.
   c. Slice based encoding (8 slices) or 1 frame
   d. Intra Refresh (1 slice per frame) or every 8th frame.
   e. Left and right eye buffer are independently encoded.
   f. Pre-encoding delay: Encoder pre-delay is varying between 10 to 20ms
   g. Encoding delay is modelled to vary with mean `4/slice_numbers` and standard deviation `3/slice_numbers` and maximum being the frame interval (aligned with clause 5.7.2 and Annex B).
3) Content Delivery Model
   a. Content Delivery Model see clause 5.7.2 with, detailed configurations in Table 6.4-1.
   b. Packet MaxSize 1500 byte (cloud) or unlimited (edge)
   c. Edge/Cloud to gNB bitrate is 1.5 media bitrate => delay variance
4) Delivery receiver
   a. Modeling according to clause 5.6.
   b. Packets are dropped if late
   c. Slice loss model (1 lost packets per slice results in slice loss)
   d. Timestamp of slice if time stamp of latest packet of slice
   e. Maximum latency for slice: 60ms (see TR 26.928, clause 4 and 6.2.5.1)
5) Decoding Model
   a. Modeling according to clause 5.6.
   b. Takes into account slice structure, spatial and temporal error propagation, intra refresh
6) Quality evaluation tool.
   a. Modeling according to clause 5.8.
   b. The following metrics are considered for each user and buffer
      i. IP Packet loss rate
      ii. IP Packet late rate
      iii. Slice loss rate
      iv. Area loss rate (total amount of Coding Units)

        v. Area damage rate (total amount of Coding Units)

        vi. Average encoded PSNR

        vii. Average PSNR

            a. Average over all buffers

            b. Multi-user

                i. Average over all users

                ii. Percentile of support

7) A model for the uplink traffic in a similar fashion also providing packet traces.

    a. XR Pose is sent uplink

    b. Details are in clause 5.8.

    c. The uplink bitrate for the pose if 200 kbit/s CBR, with 4ms packet interval and packet size 100 byte. This means that the content is rendered with a pose of typically 10-15ms age.

In addition, the following configurations are considered:

- Start frame for N=16 users at different starting positions [1,1801,901,2701,451,2251,1351,3251,226,2026,1126,2926,676,2476,1576,3476]
- Slice and Error Resilience:
    - For configuration 1, 2, 3, 4, and 6 use slice setting 8 with one intra slice per frame
    - For configuration 5: No slice setting with one I-frame very 8 frames
- Bitrates:
    - For configurations 1, 2, 5 and 6 capped VBR at 30 Mbit/s total with window size 12 frame (200ms) (target frame size is 31,250 byte per buffer, but buffer may exceed slightly – only over 200ms it is not exceeding). (aligns with findings in Annex B).
    - For configurations 3 and 4: Constant bitrate at 30 Mbit/s total with window size 1 frame (maximum frame size is 31,250 byte per buffer, almost constant).
- Delay:
    - Encoder pre-delay is varying between 10 to 20ms.
    - Encoding delay is modelled to vary with mean 4/slice_numbers and std 3/slice_numbers and maximum being the frame interval.
- Buffer Interleaving:
    - For configuration 1, 2, 3, 4 and 5, buffer interleaving set to false.
    - For configuration 6, buffer interleaving set to false.
- Content Delivery Modeling
    - For configurations 1, 3, 5, and 6, 1500 byte max packet size (addressing the cloud server case in Annex B)
    - For configurations 2 and 4, unlimited packet size, i.e. each slice results in a packet (addressing the edge serve case in Annex B)
    - The bitrate is set to an excess of 1.5, i.e. 45 Mbit/s for 30 Mbit/s content bitrate and 67.5 Mbit/s for 45 Mbit/s content bitrate. This aligns with the parameters in Annex B.

Beyond the above, the configurations provided in Table 6.4-1 are recommended with priority according to order.

Table 6.4-1 XR Split Rendering Recommended Configurations

[1]	Configuration	[2]	Basic Content Parameters
[3]	**VR2-1**	[4]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffer sent at same time, 1500 byte max packet size
[5]	**VR2-2**	[6]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffer sent at same time, unlimited packet size
[7]	**VR2-3**	[8]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s CBR with window 1 frame, buffer sent at same time, 1500 byte max packet size packets
[9]	**VR2-4**	[10]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s CBR with window 1 frame, buffer sent at same time, unlimited packet size
[11]	**VR2-5**	[12]	1 slice per eye buffer, every 8th frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffer sent at same time, 1500 byte max packet size
[13]	**VR2-6**	[14]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffers sent interleaved, 1500 byte max packet size
[15]	**VR2-7**	[16]	8 slices per eye buffer, 1 slice per frame is intra coded, 45Mbit/s capped VBR with window 200ms, buffer sent at same time, 1500 byte max packet size
[17]	**VR2-8**	[18]	8 slices per eye buffer, 1 slice per frame is intra coded, 45Mbit/s capped VBR with window 200ms, buffer sent at same time, unlimited packet size

The following provides an idea of the latencies of packets from the time that a frame is rendered in the XR server to the time the packet arrives at the gNB/Radio:

- Min-Latency:
  - Pre-encoding delay: 10ms
  - Encoding delay: 2ms
  - Encoder-to-gNB delay: 3ms
  - Total delay: 15ms
- Max-Latency:
  - Pre-encoding delay: 20ms
  - Encoding delay: 17ms (considered exceptionally large)
  - Encoder-to-gNB delay: 10ms (considered exceptionally large)
  - Total delay: 47ms
- Typical-Latency:
  - Pre-encoding delay: 15ms
  - Encoding delay: 4ms
  - Encoder-to-gNB delay: 11ms
  - Total delay: 30ms

Note that

- the maximum delay of 47ms aligns with the 32ms from Annex B assuming without constant delay of around 15ms.
- typical latency of 30ms aligns with the 32ms from Annex B assuming without constant delay of around 15ms

6.5 Traces and Statistical Models

6.5.1 Traces

6.5.1.1 Overview

A set of P-Traces for 16 users are provided, each 1 minute duration for the 8 configurations as provided in clause 6.4.

The traces can be accessed here

http://dash.akamaized.net/WAVE/3GPP/XRTraffic/Traces/Candidate/VR2/

6.5.1.2 VR2-1 Plots and Insights

Figure 6.5.1.2-1 provides the S-Trace statistics for both eyes and one user 0. The bitrate over time varies, but is bound to 30 Mbit/s for almost all cases. Frame size variation is more distributed than VR2-3, with much smaller frames occasionally. Due to window size 12 (i.e. 200ms), larger frame sizes occur. Slice size distribution also follows the peak at 3500 bytes with distributions up to 12000 bytes.

Overall, the total bitrate is likely too low to express best quality, as the rate control forces the frame size to be constant for most of the time.

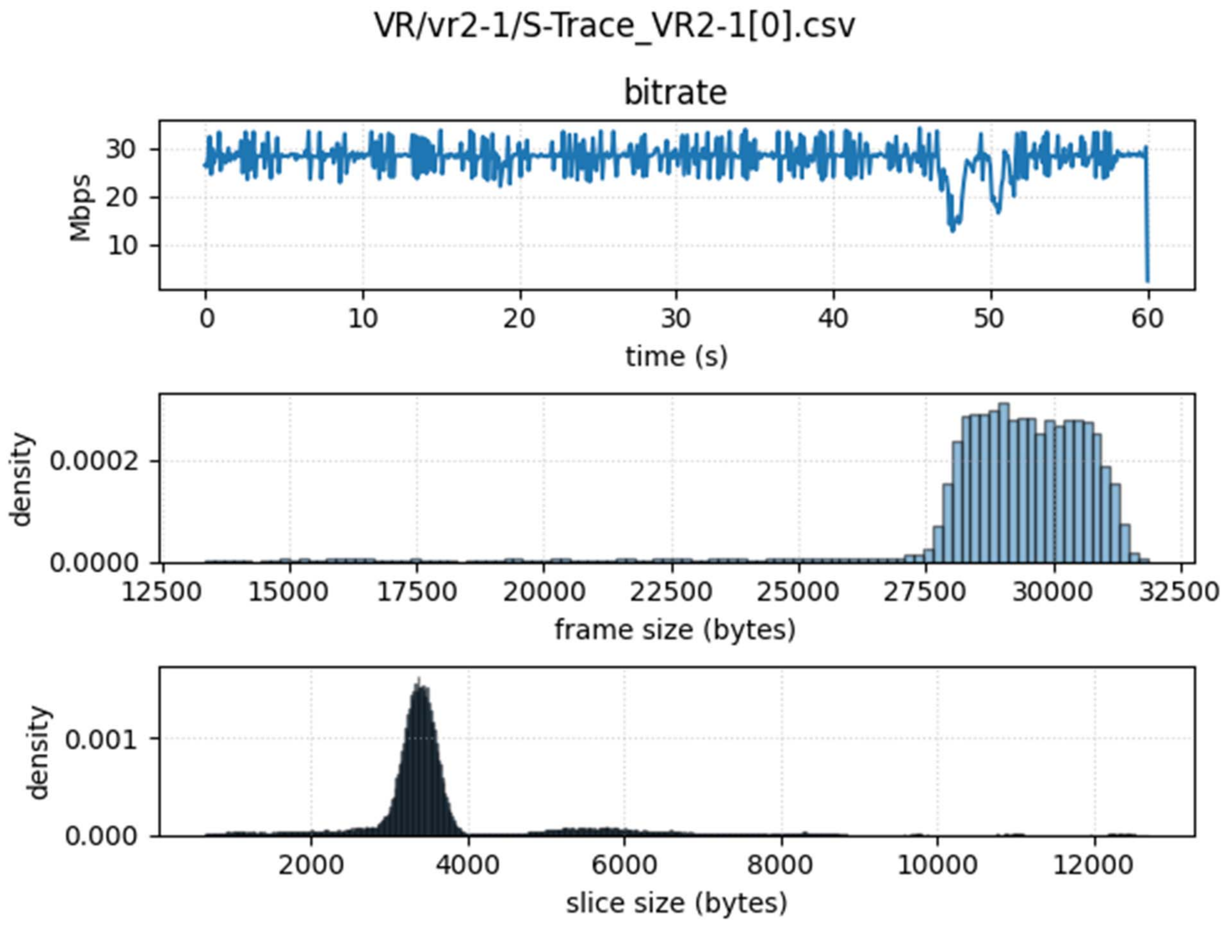


Figure 6.5.1.2-1 S-Trace Statistics for VR2-1

6.5.1.3 Initial VR2-7 Plots and Insights

Figure 6.5.1.3-1 and Figure 6.5.1.3-2 provide the S-Trace and P-Trace statistics for VR2-7 for both eyes and one user 0, respectively. The bitrate over time varies, but is bound to 45 Mbit/s. This also shows significantly more variance. Frame size variation is more distributed than VR2-1, with much smaller frames occasionally. Due to window size 12 (i.e.

200ms), larger frame sizes occur. Slice size distribution also follows the peak at 5500 bytes with distributions up to 12000 bytes.

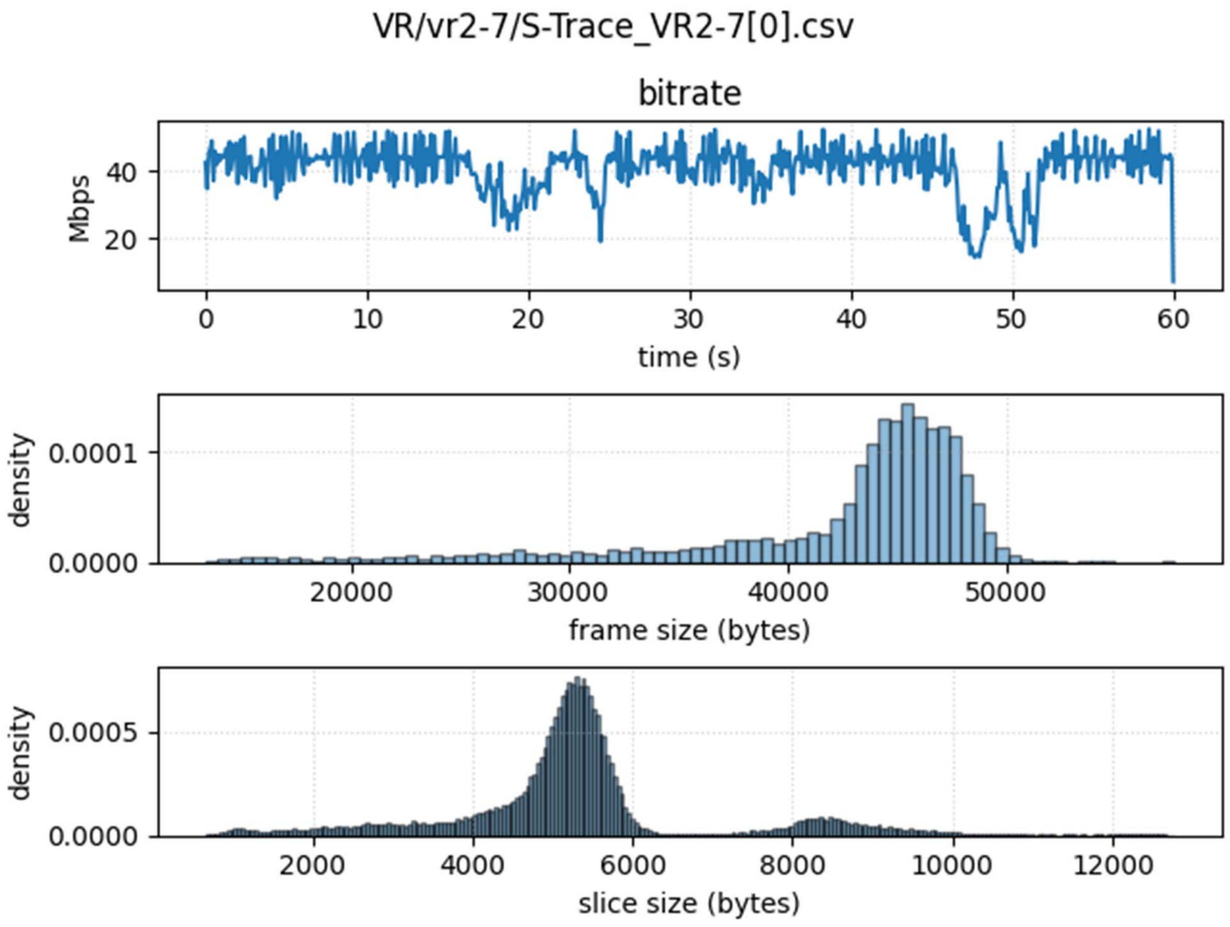


Figure 6.5.1.3-1 S-Trace Statistics for initial VR2-7

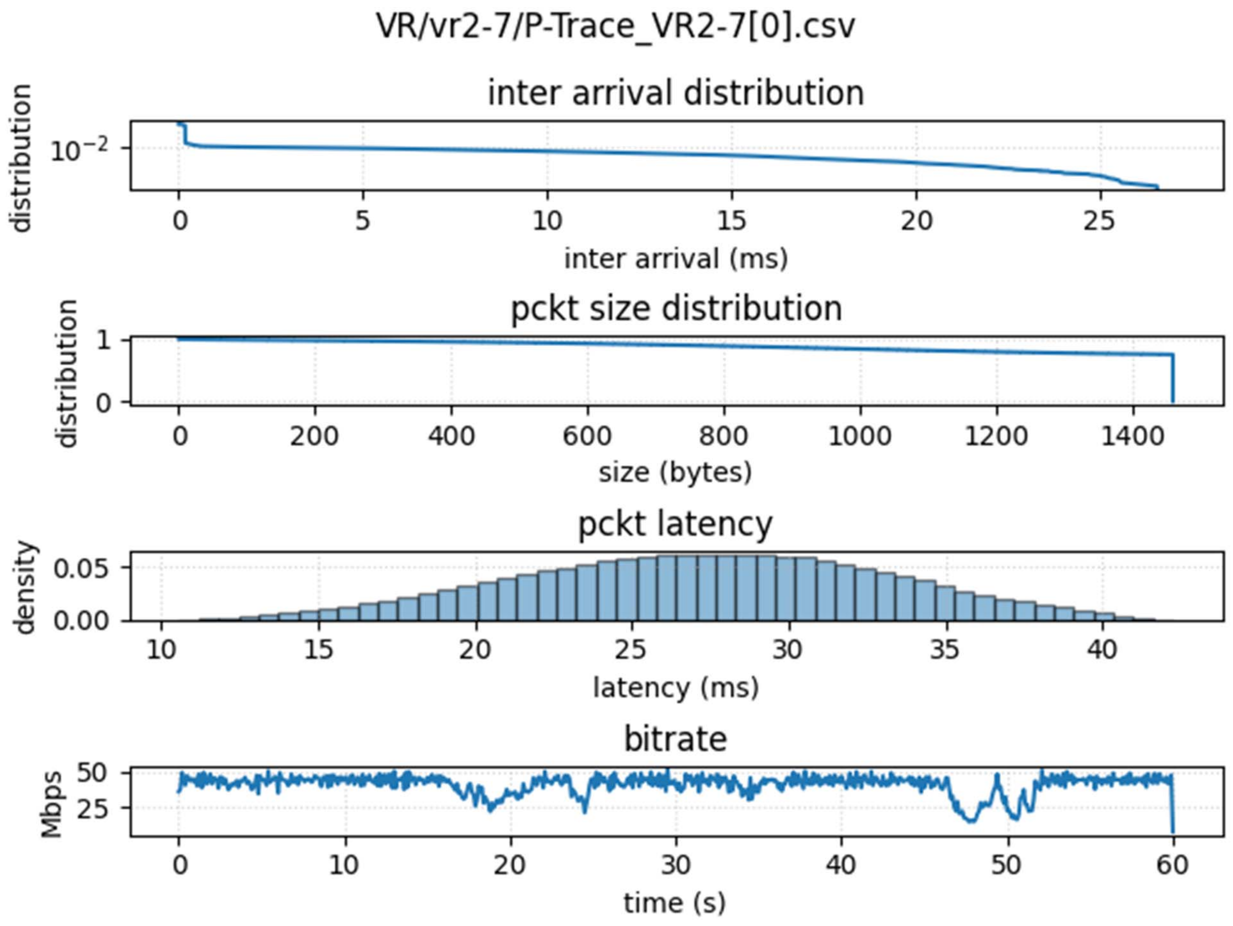


Figure 6.5.1.3-1 P-Trace Statistics for initial VR2-7

This bitrate of 45 Mbit/s seems to better express a realistic content model.

6.5.2 Statistical Models

Analyzing the initial traces that were derived and they are attached to this document for VR2-3 and VR2-4.

We extract several statistical models for VR2-3 and VR2-4 in Figure 6.5.2-1 and Figure 6.5.2-2, respectively.

- Inter arrival distribution: what is the typical difference between packet arrival
- Packet size distribution: what are the different packet sizes
- Packet latency distributions

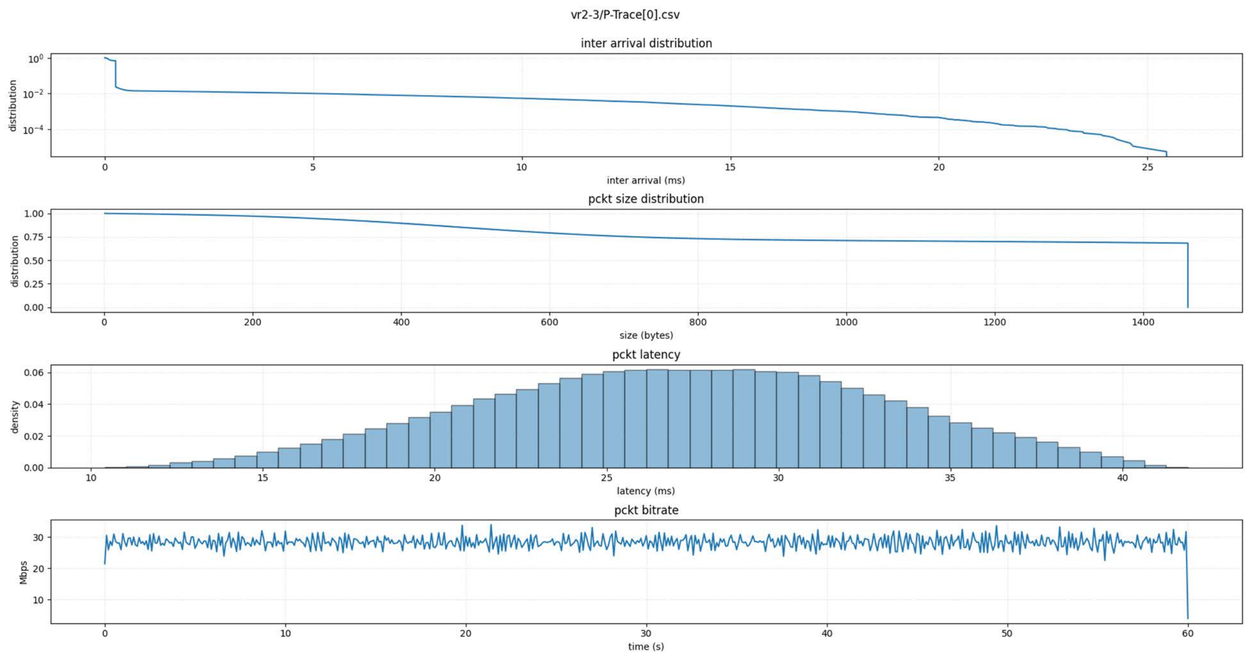


Figure 6.5.2-1 VR2-3 Statistics

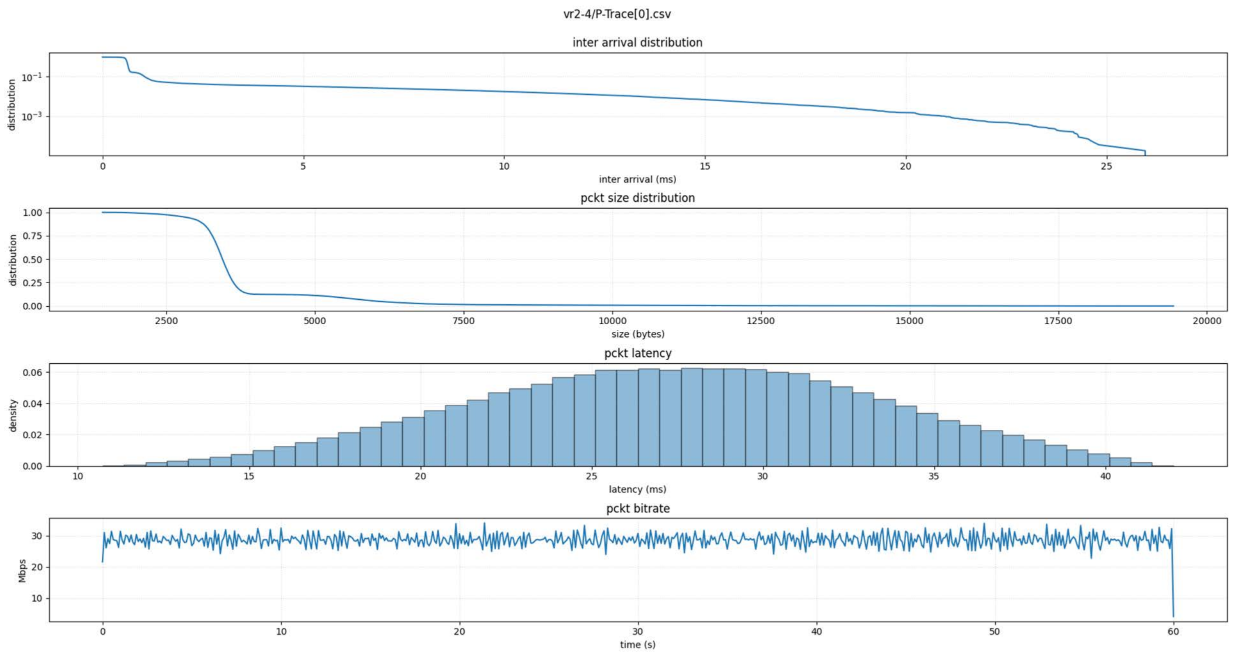


Figure 6.5.2-2 VR2-4 Statistics

Observations:

- The traffic statistics can be mapped to statistics, but we need to know which statistics are needed
- It is clear that second order statistics are not represented in the above distributions
- It is also clear that depending on the configurations, the statistics are quite different.

More data can be extracted based on the attached traces and distributions.

Based on the observations it is agreed

- that SA4 develops statistical models based on traces

- that RAN1 may develop statistical models based on traces
- that RAN1 may ask SA4 to provide statistical models for specific setups and parameters, if needed, for example
  - Statistical models for packets associated to I-frames and P-frames
  - Statistical models for slices and video frames
  - Statistical models for different importance settings

6.5.3 Analysis of the Traces and Statistical Models

6.5.3.1 Frame Size

Statistical models are in Table 5.1.1.5-1 in TR 38.838 [4], repeated below in Table 6.5.3.1-1. The size of a frame is determined by the given data rates and frame rates, which is modelled as a random variable following truncated Gaussian distribution, as shown in Table 5.1.1.5-1 in TR 38.838 [4].

Note that a frame in the context of TR 38.838 is defined as the collection of all packets of a packet trace with the same timestamp for presentation.

Table 6.5.3.1-1: Statistical parameter values for dual eye buffer frame size (TR 38.838)

Parameter	unit	values for evaluation	Optional values for evaluation
Mean: M	byte	R×1e6 / (2×F) /8	R×1e6 / (2×F) / 8
STD	byte	10.5% of M	4% of M
Max	byte	150% of M	112% of M
Min	byte	50% of M	88% of M
R: data rate of the flow in Mbps F: frame generation rate of the flow in fps			

An analysis was done using the traces defined in clause 6.5.1 and the statistical model. Table 6.5.3.1-2 show the frame size distribution for the P-trace according to the traces. It can be seen that the frame size distribution based on the P-traces in clause 6.5.1 are aligned with frame size model in TR 38.838 [4].

Table 6.5.3.1-1 Frame size distribution deriving from the P-trace files

Parameter	VR 2-1	VR 2-2	VR 2-3	VR 2-4	VR 2-5	VR 2-6	VR 2-7	VR 2-8
Bitrate (Mbit/s)	30	30	30	30	30	30	45	45
Mean (byte)	7378	7211	7640	7467	6699	3689	10506	10265
STD (byte)	560	543	176	173	847	296	1574	1530
Min (byte)	3531	3488	7115	6974	1522	1719	3544	3480
Max (byte)	9224	9117	8164	7996	13106	4836	13523	13147
STD/Mean	7.60%	7.53%	2.30%	2.32%	12.64%	8.02%	14.99%	14.91%
Min/Mean	47.86%	48.37%	93.14%	93.40%	22.72%	46.61%	33.74%	33.90%
Max/Mean	125%	126%	107%	107%	196%	131%	129%	128%

Figure 6.5.3.1-1 shows the distribution of the slice size for different P-Traces.

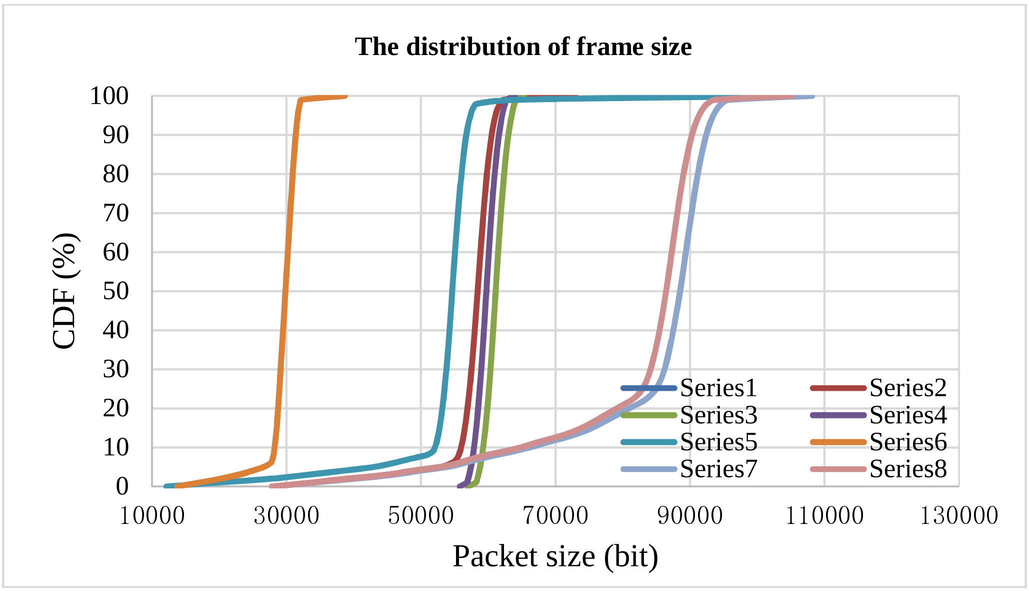


Figure 6.5.3.1-1 The statistic of packet size by using the P-trace data from SA4

From the figure the following is observed:

- VR2-1, VR2-2, VR2-3, VR2-4 and VR2-5 share similar frame size distribution as they all operate at the same bitrate of 30 Mbit/s and basically every frame uses the same bitrate to maximize the quality within the delay constraints. VR2-1, VR2-2 and VR2-5 have higher likelihood of smaller packets as they operate in capped VBR.
- VR2-6 has lower frame sizes, because it sends the buffers in a staggered manner, resulting in twice as many frames, but each at half size.
- VR2-7 and VR2-8 operate at higher bitrate and hence the quality for several frames is higher making use of the higher bit budget, but some frames do not need the full bit budget and hence stay at lower size.

6.5.3.2 Jitter

An analysis was done using the traces defined in clause 6.5.1 for jitter statistical modelling. Table 6.5.3.2-1 shows the jitter distribution for the P-trace according to the traces.

Table 6.5.3.2-1: Jitter distribution deriving from the P-trace files

Parameter	VR 2-1	VR 2-2	VR 2-3	VR 2-4	VR 2-5	VR 2-6	VR 2-7	VR 2-8
Mean (ms)	0	0	0	0	0	0	0	0
STD (ms)	4.96	5.00	4.85	4.89	5.31	4.02	5.19	5.18
Min (ms)	-15.89	-16.00	-11.87	-11.96	-18.95	-14.40	-16.48	-16.18
Max (ms)	+11.27	+10.30	+10.10	+10.51	+18.69	+8.08	+11.59	+11.13
Jitter range (ms)	[-8.39, +7.89]	[-8.42, +7.91]	[-8.26, +7.69]	[-8.30, +7.77]	[-8.74, +8.42]	[-7.05, +6.24]	[-8.82, +8.19]	[-8.79, +8.20]
jitter range = [5%-tile in CDF, 95%-tile in CDF] ms								

Figure 6.5.3.2-1 shows the distribution of the jitter value for different P-traces.

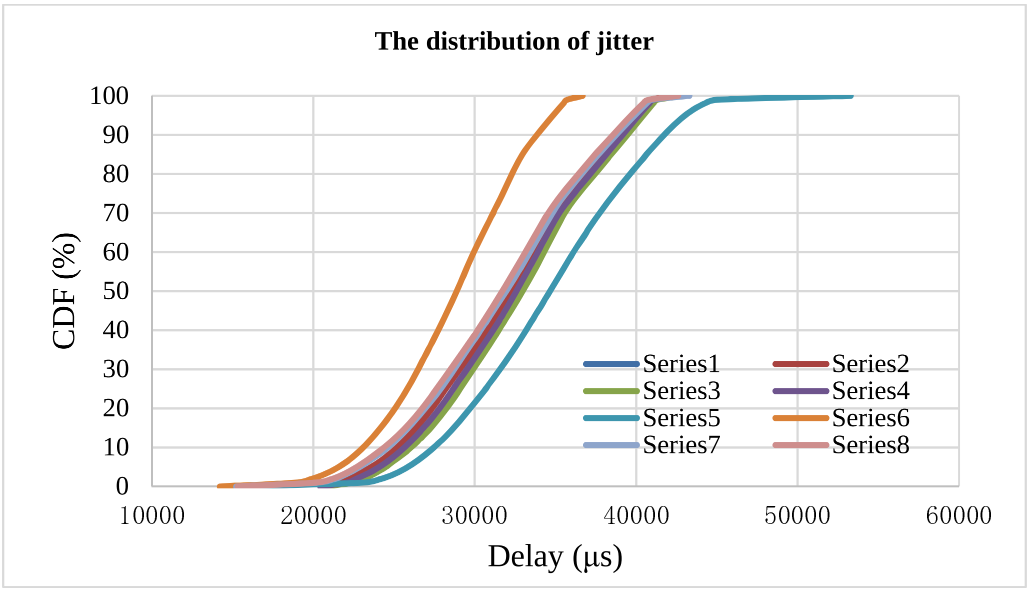


Figure 6.5.3.2-1 The statistic of jitter from the P-trace files

From Table 6.5.3.2-1 and Figure 6.5.3.2-1 the following is observed:

- VR2-1, VR2-2, VR2-3, VR2-4, VR2-5, VR2-6, VR2-7 and VR2-8 share similar jitter distribution with Mean 0ms and STD 5ms in rang [-8, 8] ms, since they all experience the same encoder and transport network.

Note that the jitter statistical model refers to the description in TR 38.838 [4]. The jitter is modelled as a random variable added on top of periodic arrivals, which follows truncated Gaussian distribution.

6.6 Test Channel Results

In clause 5.8.2, test channels are introduced in order to

- Permit to emulate typical radio conditions in terms delays and losses.
- Permit to evaluate the application quality for different representative radio conditions.

The 3GPP radio experts are expected to simulate based on statistical models whereby the statistical models may for example differentiate packet classes, as an example, different radio QoS is associated to different packets.

3GPP radio experts may ask 3GPP application service experts to evaluate the simulation and evaluation results for different test channels in order to identify the benefit of specific radio settings. For this purpose, 3GPP radio experts should support 3GPP application service experts to define test channels models and configurations. Examples are

- Use Test channel with PLR 0.1% and iid delay between 0 and 20ms and simulate the quality
- Use Test channel with
  - PLR 0.1% and iid delay between 0 and 20ms for packets marked with type P
  - PLR 0.01% and iid delay between 0 and 20ms for packets marked with type I
  - and provide the resulting quality

It is agreed

- that 3GPP application service experts evaluates the quality of different configurations and radio settings through test channels
- that 3GPP radio experts supports 3GPP application service experts in the development of different test channel models

- that 3GPP radio experts may ask 3GPP application service experts to evaluate the simulation and evaluation results for different test channels in order to identify the benefit of specific radio settings, if needed.

Table 6.6-1: traces and configuration for P-file generation

[19]	Configuration	[20]	Basic Content Parameters
[21]	**VR2-1**	[22]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffer sent at same time, 1500 byte max packet size
[23]	**VR2-2**	[24]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffer sent at same time, unlimited packet size
[25]	**VR2-3**	[26]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s CBR with window 1 frame, buffer sent at same time, 1500 byte max packet size packets
[27]	**VR2-4**	[28]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s CBR with window 1 frame, buffer sent at same time, unlimited packet size
[29]	**VR2-5**	[30]	1 slice per eye buffer, every 8th frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffer sent at same time, 1500 byte max packet size
[31]	**VR2-6**	[32]	8 slices per eye buffer, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, buffers sent interleaved, 1500 byte max packet size
[33]	**VR2-7**	[34]	8 slices per eye buffer, 1 slice per frame is intra coded, 45Mbit/s capped VBR with window 200ms, buffer sent at same time, 1500 byte max packet size
[35]	**VR2-8**	[36]	8 slices per eye buffer, 1 slice per frame is intra coded, 45Mbit/s capped VBR with window 200ms, buffer sent at same time, unlimited packet size

Based on the traces and configuration as available in Table 6.6-1 we have produced some initial results with the following test channel.

- "loss_rate": 0.01,
- "max_delay_ms": 50,
- "max_bitrate": 60000000,

The results were produced for 4 users each. We observe that the loss rates and delays may be too high, but it was for an initial plausibility checking.

The results are provided here: https://dash.akamaized.net/WAVE/3GPP/XRTraffic/Test-Results/sample-results-2021-03-31-01.zip and include P', S', V', and Q traces as well as inter arrival plots.

Table 6.6-2 provides the averaged results for the 4 users and for each of the eye buffers.

Table 6.6-2: traces and configuration for P-file generraration

config	total_packets	PLoR	PLaR	SLR	CAR	ALR	DAR	LDR	PSNR	PSNR yuv	RPSNR	RPSNR yuv
1	712134	1.004 %	20.890%	24.283%	24.781%	24.283%	50.936%	75.219%	40057	40419	10417	10535
2	230400	1.010 %	15.105%	16.115%	28.803%	16.115%	55.082%	71.197%	40050	40412	12067	12203
3	727022	0.989 %	22.214%	25.836%	24.137%	25.836%	50.027%	75.863%	39739	40101	10083	10198
4	230400	0.973 %	15.820%	16.793%	28.587%	16.793%	54.620%	71.413%	39742	40104	11904	12039
5	528639	0.989 %	33.336%	57.392%	14.175%	57.392%	28.433%	85.825%	40617	40979	6189	6253
6	712220	1.007 %	7.459 %	11.120%	30.522%	11.120%	58.358%	69.478%	40054	40415	12772	12916
7	924161	1.017 %	28.905%	32.619%	21.278%	32.619%	46.102%	78.722%	37290	37652	8326	8427
8	230400	0.972 %	20.822%	21.793%	26.494%	21.793%	51.713%	73.506%	37297	37659	10322	10446

The following is observed:

1. The packet loss rates are too high and in particular the channel latencies are too high to get meaningful results
    - The packet late rate is too high
    - In between 11 to 50% of the slices are lost
    - The Loss and Damaged Area is 70 to 85%
2. There is a problem with the PSNR for config 7 and 8, which is lower than for lower bitrates. We are investigating this.
3. Best performing is config 6, which does eye buffer interleaving. In this case, the second eye buffer is not blocked in the sending for the right as they are staggered. This is a problem in all other configurations that the second eye buffer causes the packet and slice late losses.
4. Worst performing is the single slice configuration as a single loss results in the loss of an entire frame.
5. The same packet loss rate for larger packets (i.e. mapped to a slice) results in better quality as for smaller packets (as each loss aggregates later to a slice loss).
6. We also identified that handling the frame data too complex, so we plan to create a binary pseudo bitstream.

The following is concluded:

1. We are moving into a good direction, but some open issues still
2. We need to fix the bug on PSNR
3. We need to use identify good channel parameters for
    a. PLR = 1e-3, 1e-4, 1e-5
    b. Latencies = 10ms, 20ms, 30ms (iid), 10ms (iid) for 98% of packets

We need to identify to what extent these configurations are useful.

7 Cloud Gaming

7.1 Introduction

Cloud Gaming according to TR 26.928 refers to the case that rendering of the views is done in the cloud. It differentiates from clause 6 XR Split Rendering in a sense that it is expected to played on a single screen device, e.g. a smartphone or tablet.

7.2 Reference System Design

7.2.1 Overview

The system design for cloud gaming is provided in Figure 7.2.1-1 and is similar to what is provided for XR Split rendering.

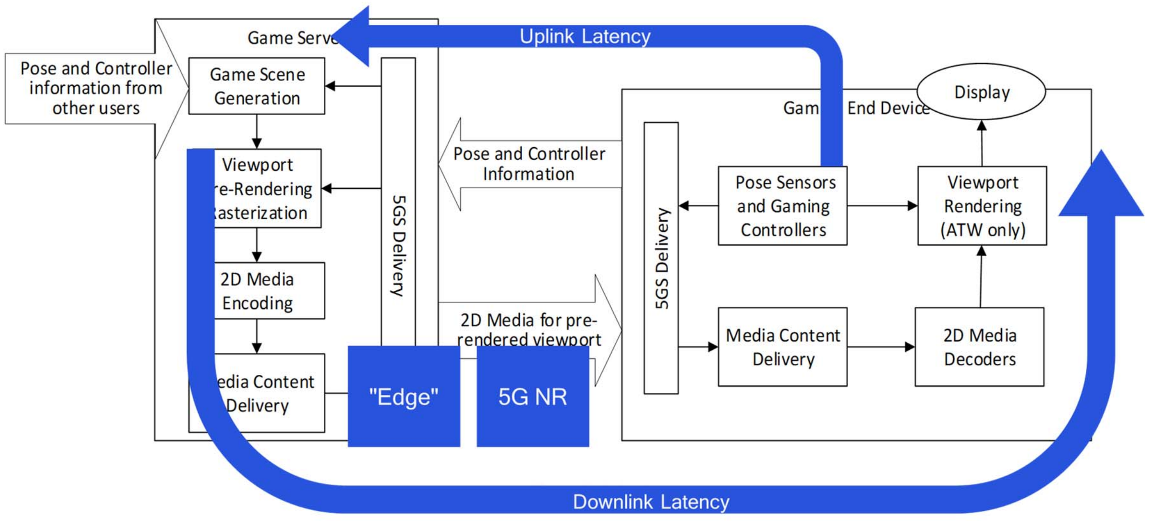

Game Server
Uplink Latency
Pose and Controller information from other users
Game Scene Generation
Viewport Pre-Rendering Rasterization
2D Media Encoding
Media Content Delivery
5GS Delivery
Pose and Controller Information
2D Media for pre-rendered viewport
"Edge"
5G NR
Gaming End Device
Display
Pose Sensors and Gaming Controllers
Viewport Rendering (ATW only)
Media Content Delivery
2D Media Decoders
Downlink Latency

Figure 7.2.1-1 Cloud Gaming Reference Architecture

The cloud scene is rendered in the device. The following assumptions are taken. The Game Server runs a game engine to generate the game scene based on information coming from a gaming end device, and in multi-user cases also what comes from other users. The gaming server rasterizes the viewport and does a scene pre-rendering.

According to Figure 7.2.1-1, the viewport is pre-dominantly rendered in the gaming server, but the device is able to do latest pose correction, for example by asynchronous time-warping (see clause 4.1 of TR26.928) or other pose correction to address changes in the pose.

- Graphics workload is split into rendering workload on a powerful cloud gaming server (in the cloud or the edge) and pose correction (such as ATW) on the gaming device
- Low motion-to-photon latency is preserved via on device Asynchronous Time Warping (ATW) or other pose correction methods.

The following call flow highlights the key steps:

1) A gaming device connects to the network and joins a gaming application
   a) Sends static device information and capabilities (supported decoders, viewport)
2) Based on this information, the gaming server sets up encoders and formats
3) Loop
   a) gaming end device collects pose and controller information
   b) pose and controller information is sent to gaming server
   c) The gaming uses the pose to pre-render the gaming viewport
   d) the scene viewport is encoded with 2D media encoders
   e) The compressed media is sent to gaming device along with the pose that it was rendered for
   f) The gaming device decompresses video
   g) The gaming device uses the pose provided with the video frame and the actual pose for an improved prediction using and to correct the local pose, e.g. using ATW.

According to TR 26.928, clause 4.2.2, the relevant processing and delay components are summarized as follows:

- **User interaction delay** is defined as the time duration between the moment at which a user action is initiated and the time such an action is taken into account by the content creation engine. In the context of gaming, this is the time between the moment the user interacts with the game and the moment at which the game engine processes such a player response.

- **Age of content** is defined as the time duration between the moment a content is created and the time it is presented to the user. In the context of gaming, this is the time between the creation of a video frame by the game engine and the time at which the frame is finally presented to the player.

The **roundtrip interaction delay** is therefore the sum of the *Age of Content* and the *User Interaction Delay*. If part of the rendering is done on a gaming server and the service produces a frame buffer as rendering result of the state of the content, then for raster-based split rendering (as defined in clause 6.2.5) in cloud gaming applications, the following processes contribute to such a delay:

- User Interaction Delay (Pose and other interactions)
  - capture of user interaction in game client,
  - delivery of user interaction to the game engine, i.e. to the server (aka network delay),
  - processing of user interaction by the game engine/server,
- Age of Content
  - creation of one or several video buffers (e.g. one for each eye) by the game engine/server,
  - encoding of the video buffers into a video stream frame,
  - delivery of the video frame to the game client (a.k.a. network delay),
  - decoding of the video frame by the game client,
  - presentation of the video frame to the user (a.k.a. framerate delay).

As ATW is applied corrections to pose are applied by device internal processing. What determines the network requirements for split rendering is time of pose-to-render-to-photon and the roundtrip interaction delay. According to clause TR 26.928, clause 4.5, the permitted downlink latency is typically 50-60ms.

7.2.2 Considered Content Formats

7.2.2.1 Introduction

Video games have different characteristics that are important to take into account when encoding the rasterized frames produced by the game engine. In TR 26.928 [6], clause 4.2.2, a few different types of games and their interaction delay tolerance are documented. However, TR 26.928 [6] does not differentiate the characteristics of the content. This aspect is addressed in the following.

In particular, the following characteristics are important:

- Dynamicity of content: how frequent rasterized frames change when compared to previous frame
- Complexity of content: how much content changes between frames and how complex such changes are
- Type of content: traditional CGI, photo-realistic CGI or natural images/video

Depending on these characteristics as well as the interaction delay tolerance, video games can be organized into different categories as document in the remainder of this clause.

A detailed analysis is provided in TR26.955, clause 6.6.

7.2.2.2 Category A: Low/medium dynamicity with low/medium complexity.

This category includes games such as board games, turn-by-turn strategy games, management/simulation games or non-realtime role-playing games (RPG) in which content may not change over several consecutive frames and changes are typically limited.

This category also includes games such as adventure games, casual games, or platform games in which although content may change at every single frame, changes are limited to animation of sprites or simple global movements of the content.

The common characteristics of the games in this category is that their playability can support longer interaction delay tolerance (500 – 1000ms according to TR 26.928 [6]) and their content is typically considered to video encode.

7.2.2.3 Category B: games with high dynamicity and low/medium complexity.

This category includes games such as fighting games, racing games, real-time strategy (RTS) games or real-time RPGs in which content is very dynamic but changes are either limited or simple transforms.

The common characteristics of the games in this category is that their playability requires shorter interaction delay tolerances (100ms according to TR 26.928 [6]) while their content is still considered simple to video encode (with high benefits from prediction coding).

7.2.2.4 Category C: games with high dynamicity and high complexity.

This category includes games such as first-person shooters (FPS), Massive Multiplayer Online (MMO) games and racing games in which content is very dynamic with possibly very significant changes regularly in the content.

The common characteristics of the games in this category is that their playability requires shorter interaction delay tolerances (100ms according to TR 26.928) and their content is typically considered as complex content to video encode.

7.2.2.5 Category D: photo-realistic games or games based on natural images/video.

The main characteristics of the games in this category is that their content is typically considered as more complex content to video encode.

7.2.2.6 Category E: XR game content

XR Game content is covered as part of XR Split rendering and not further discussed.

7.2.2.7 Summary

Table 7.2.2.7-1 provides an overview of the different source signal properties for Online Gaming. This information is used to select proper test sequences.

Table 7.2.2.7-1 Online Gaming source properties

Source format properties	Category A low/med dynamicity & low/med complexity games	Category B high dynamicity & low/med complexity games	Category C high dynamicity & high complexity games	Category D photo-realistic or natural video games
Spatial resolution	1280x720, 1920x1080, 3840x2160	1280x720, 1920x1080, 3840x2160	1280x720, 1920x1080, 3840x2160	1280x720, 1920x1080, 3840x2160
Chroma format	Y'CbCr	Y'CbCr	Y'CbCr	Y'CbCr
Chroma subsampling	4:2:0, 4:4:4	4:2:0, 4:4:4	4:2:0, 4:4:4	4:2:0, 4:4:4
Picture aspect ratio	16:9	16:9	16:9	16:9
Frame Buffers	1	1	1	1
Frame rates	30, 50, 60 Hz	30, 50, 60, 90, 120 Hz	30, 50, 60, 90, 120 Hz	30, 50, 60, 90, 120 Hz
Bit depth	8	8	8, 10	8, 10
Colour space formats	BT.709, BT.2020	BT.709, BT.2020	BT.709, BT.2020	BT.709, BT.2020
Transfer characteristics	N/A	N/A	BT.2100 (HDR)	BT.2100 (HDR)

In practical considerations, the NVIDIA Encoding functions may be used. The parameters of such an encoder are documented here https://developer.nvidia.com/nvidia-video-codec-sdk.

7.2.3 Considered System Parameters

Based on the discussion on clause 7.2.1 and 7.2.2, several parameters are relevant for the overall system design.

- Game:
    - Type of game
    - state of game,
    - multi-user actions, etc.
- User Interaction:
    - Game pose by the device,
    - Game interactions by controllers
- Formats of rasterized video signal. Typical parameters are:
    - See above
- Encoder configuration
    - Codec: H.264/AVC or H.265/HEVC
    - Bitrate: Bitrate setting to a specific value (e.g. 50 Mbit/s)
    - Rate control: CBR, Capped VBR, Feedback based, CRF, QP
    - Slice settings: 1 per frame, 1 per MB row, X per frame
    - Intra settings and error resilience: Regular IDR, GDR Pattern, adaptive Intra, feedback based Intra, feedback based predication and ACK-based, feedback-based prediction and NACK based
    - Latency settings: P pictures only, look-ahead units
    - Complexity settings for encoder
- Content Delivery
    - Slice to IP mapping: Fragmentation
    - RTP-based time codes and packet numbering
    - RTP/RTCP-based feedback ACK/NACK
    - RTP/RTCP-based feedback on bitrate
- 5G System/RAN Configuration:
    - QoS Settings (5QI): GBR, Latency, Loss Rate
    - HARQ transmissions, scheduling, etc.
- Content Delivery Receiver configuration:
    - Loss Detection: sequence numbers
    - Delay/Latency handling
    - Error Resilience
    - ATW
- Quality Aspects
    - Video quality (encoded)

- Lost data
- immersiveness

7.3 Simulation System

Figure 7.3-1 shows the simulation system used for Cloud gaming based on the architecture in Figure 5.2.1-1. The following instantiation is provided:

- Single S-Traces is generated to address a video output for a screen
- No feedback on video encoding or content delivery is considered

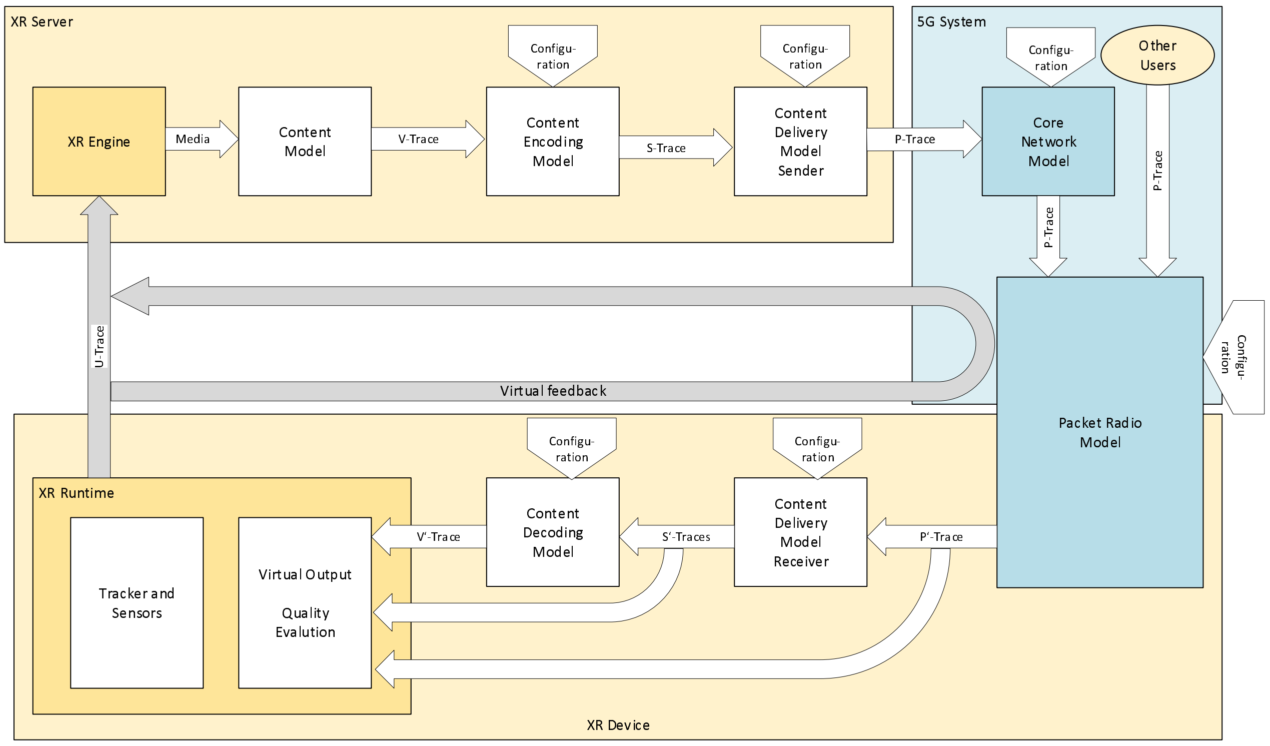


Figure 7.3-1 Cloud Gaming Simulation System

7.4 Recommended Configurations

The following parameters are considered:

1) Content Model:
   a. Game output with 1920 x 1080 and 4096 x 2048 at 60fps.
   b. Content and Trace Preview is here:
      i. Baolei HD (to be produced)
      ii Baolei 4K (to be produced)
   c. No audio
2) Encoding Model
   a. HEVC encoding model as defined in 5.6
   b. Pre-encoding delay: Encoder pre-delay is varying between 10 to 20ms

   c. Encoding delay is modelled to vary with truncated Gaussian

3) Content Delivery Model

   a. Edge/Cloud to gNB bitrate is 1.5 media bitrate

4) Delivery receiver

   a. Slice loss model (1 lost packets per slice results in slice loss)

   b. Maximum latency for slice: 80ms (see TR 26.928, clause 4 and 6.2.5.1)

Beyond the above, the configuration options provided in Table 7.4-1 are recommended with priority according to order.

Table 7.4-1 Online Gaming source properties

Configuration	Basic Content Parameters
CG-1	4K content, 8 slices, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, 1500 byte max packet size
CG-2	4K content, 8 slices, 1 slice per frame is intra coded, 30Mbit/s capped VBR with window 200ms, unlimited packet size
CG-3	FullHD content, 8 slices, 1 slice per frame is intra coded, 8Mbit/s capped VBR with window 200ms, 1500 byte max packet size
CG-4	4K content, 1 slice, every 8th frame is intra coded, 30Mbit/s capped VBR with window 200ms, 1500 byte max packet size

In total there are 4 configurations. Additional configurations may be added.

For uplink modelling, no specific content modelling is considered.

1) A model for the uplink traffic in a similar fashion also providing packet traces.

   a. Game pose and actions are sent uplink

   b. Details are in clause 5.8.

   c. The uplink bitrate for the pose is 200 kbit/s CBR, with 4ms packet interval and packet size 100 byte. This means that the content is rendered with a pose of typically 10-15ms age.

   d. The E2E Requirements are

      i. Bitrate: 200 kbit/s

      ii. PLR: 1e-3

      iii. 10ms

In addition, the configurations as documented in clause 6.4 are to be considered.

7.5 Traces and Statistical Models

Traces and statistical models for cloud gaming are for further study.

7.6 Test Channel Results

Test Channel Results for cloud gaming are for further study.

8 AR Conversational

8.1 Introduction

In contrast to the scenarios presented in clause 6 and 7, AR Conversational provides audio and video also in the uplink. A more specific example is the use case 8 in TR 26.928 [3] (AR guided assistant at remote location (industrial services)), which requires one or two video streams in the uplink. The use case states:

- A XR device with glasses of some sort boasting view-port dependent streaming and split rendering
- The device has one 2D camera which is used for conveying the local scene to a remote location. (Use case 8 includes depth information but this could be disregarded for the purposes of this study.)

For example, it is expected that mobile network operators may configure 5G networks for less capacity in the uplink than the downlink, thus it would be relevant to confirm whether we can expect bottlenecks in the uplink, related to XR use cases. Also, it would be relevant to confirm whether the transmission of time-critical pose information in the uplink is impacted by the additional uplink video traffic.

As uplink video may in some use cases happen at the same time as split rendering in the downlink, it is relevant to add an uplink video signal with suitable characteristics to the scenario for AR Conversational.

8.2 Reference System Design

8.2.1 Overview

In order to re-use an existing architecture, the XR distributed computing architecture in TR 26.928 [3] is applicable. The camera in the XR device may be viewed as a "sensor" in this context as shown in Figure 8.2.1-1.

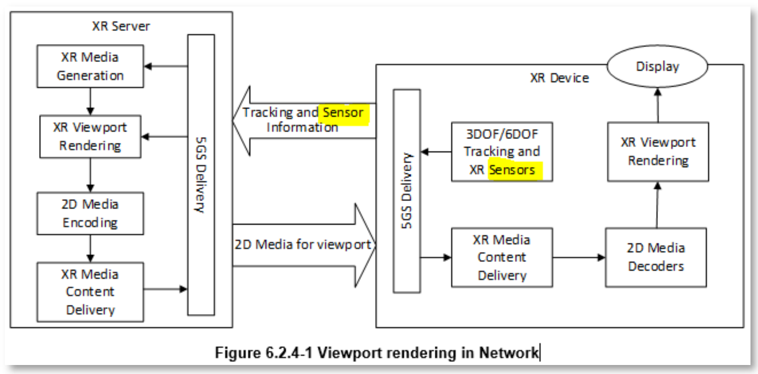
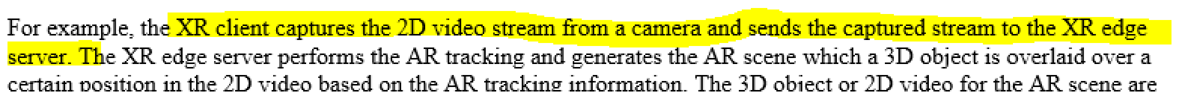


Figure 8.2.1-1 AR Conversational reference system as XR distributed compute system


For example, the XR client captures the 2D video stream from a camera and sends the captured stream to the XR edge server. The XR edge server performs the AR tracking and generates the AR scene which a 3D object is overlaid over a certain position in the 2D video based on the AR tracking information. The 3D object or 2D video for the AR scene are

Detailed system design and modeling assumptions are provided in clause 8.3.

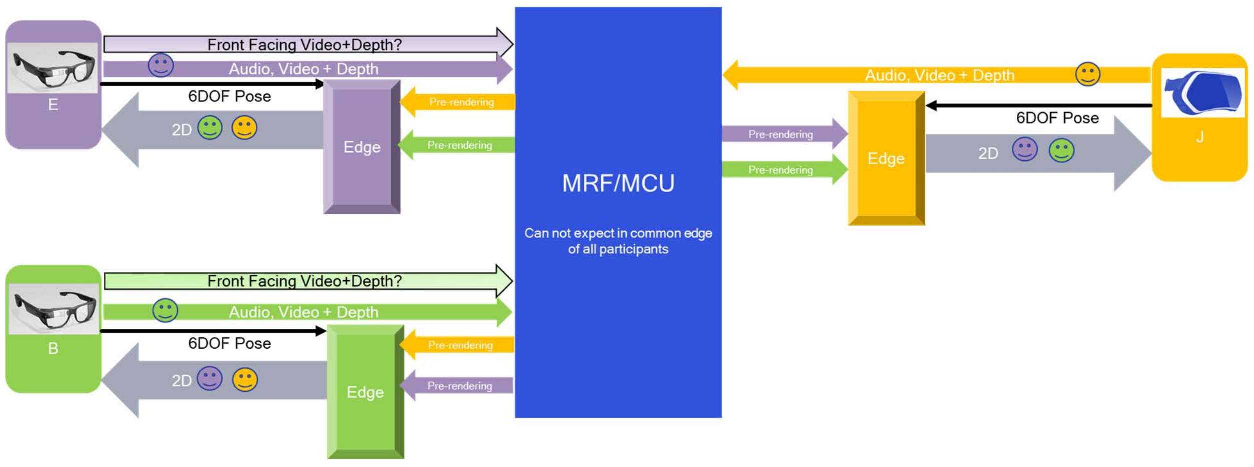
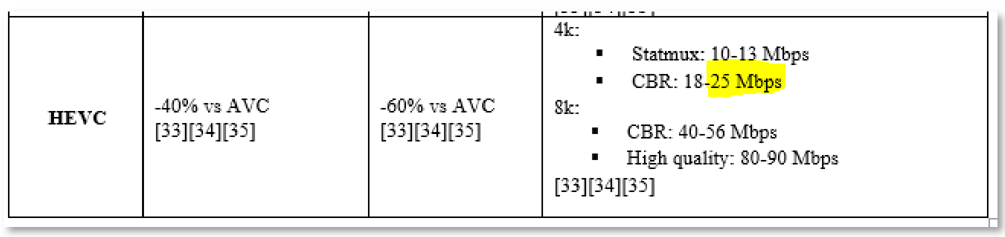


Figure 8.2.1-1 AR Conversational reference system

8.2.2 Considered Content Formats

As a starting point consider 4K 60 fps, at 0, 10 and 25 Mbps, based on 2D video as documented in TR 26.928 [3] are considered and provided in Figure 8.2.2-1.



HEVC	-40% vs AVC [33][34][35]	-60% vs AVC [33][34][35]	4k: ▪ Statmux: 10-13 Mbps ▪ CBR: 18-25 Mbps 8k: ▪ CBR: 40-56 Mbps ▪ High quality: 80-90 Mbps [33][34][35]

Figure 8.2.2-1 Content formats for AR conversational

This is based on the highest profiles in TS 26.118 [6] for VR streaming may be a starting point to derive also the 2D uplink video. In practice the rate would be highly use-case-dependent, it is advisable to select realistic values covering a variety of use case. The considered Content formats and properties are provided in Table 8.2.2-1.

Table 8.2.2-1 Considered content formats and properties

Media	Format and Model	E2E Latency requirement
3/6DOF Pose	Same as for split rendering	UL: 5-10 ms
Video + Depth	1080p, Capped VBR 10/20 Mbit/s for UL	Conversational 100ms, 200ms
2D Video is split rendering	1080p or 4K (2 eyes) same model as split rendering	60ms 100ms
Front Facing Camera*	720p, CBR 3 Mbit/s for UL	Conversational 100ms, 200ms
Audio (MPEG-H)	256/512 kbps for both UL/DL	Conversational 100ms, 200ms
Data Stream	0.5 Mbps for both UL/DL	Conversational 100ms, 200ms

8.3 Simulation System

8.3.1 Overview

A simulation overview for downlink and uplink is provided.

8.3.2 Simulation Downlink

Downlink simulation model is provided in the following

1) Content Model:
    a. Video is identical to split rendering simulation in clause 6.3
    b. Audio
        i. Max Sampling Rate: 48 kHz
        ii. Inter-frame time: 20-21.3 ms

NOTE: For the simulation purposes, the inter-frame time can be assumed to be 21.3 ms considering MPEG-H, or if we consider that the actual conversational audio codec might be a different one, we could assume 20 ms, as this has so far been used for several 3GPP speech codecs

    c. Data Stream
        - Inter-frame time: 10 ms
2) Encoding Model
    a. Video is identical to split rendering simulation in clause 6.3
    b. Audio
        - Operation Point (following 3GPP TS 26.118 [6] Table 6.1-1): 3GPP MPEG-H Audio (this Operation Point is specified for VR streaming in SA4 and can be used for simulation purposes for conversational services since the IVAS Codec is not yet available)
        - Average data rate : 256 / 512 kbps
        - Packet Loss rate should be below 1e-3
    c. Data Stream
        - Average data Rate : <0.5 Mbps
        - Packet Loss rate should be below 1e-3
3) Content Delivery Model
    a. Video is identical to split rendering simulation in clause 6.3
    b. Audio and data are for further study
4) Delivery receiver
    a. Video is identical to split rendering simulation in clause 6.3
    b. Audio and data are for further study
5) Decoding Model
    a. Video is identical to split rendering simulation in clause 6.3
    b. Audio and data are for further study

6) Quality evaluation tool.
    a. Video is identical to split rendering simulation in clause 6.3
    b. The following metrics are considered for each user and buffer
        - IP Packet loss rate
        - IP Packet late rate
        - Slice loss rate
        - Area loss rate (total amount of Coding Units)
        - Area damage rate (total amount of Coding Units)
        - Average encoded PSNR
        - Average PSNR
    c. Average over all buffers
    d. Multi-user
        - Average over all users
        - Percentile of support
    e. Audio and Data are for further study.
7) A model for the uplink traffic in a similar fashion also providing packet traces.
    a. Uplink Pose information
        - identical to split rendering simulation in clause 6.3
    b. Reverse uplink audio and video encoding see clause 8.3.3

8.3.3 Simulation Uplink

The simulation for the reversed uplink is considered as follows:

1) Content Model:
    a. Video
        - Camera Signal with 1920 x 1080 at 60fps.
        - Content and Trace Preview is for further study
    b. Audio
        - Max Sampling Rate: 48 kHz
        - Inter-frame time: 20-21.3 ms

NOTE: For the simulation purposes, the inter-frame time can be assumed to be 21.3 ms considering MPEG-H, or if we consider that the actual conversational audio codec might be a different one, we could assume 20 ms, as this has so far been used for several 3GPP speech codecs

    c. Data Stream
        - Inter-frame time: 10 ms
2) Encoding Model
    a. Video: Encoding Models see clause 6.3 with the following

- HEVC, target bitrate 10 Mbit/s (capped VBR) or AVC target bitrate 20 Mbit/s (capped VBR).
- Slice based encoding (4 slices) or 1 frame
- Intra Refresh (1 slice per frame) or every 60th frame.
- Pre-encoding delay: Encoder pre-delay is varying between 10 to 20ms
- Encoding delay is modelled to vary with mean 4/slice_numbers and std 3/slice_numbers and maximum being the frame interval (aligned with S4aV200607)

b. Audio

- Operation Point (following 3GPP TS 26.118 [6] Table 6.1-1): 3GPP MPEG-H Audio (this Operation Point is specified for VR streaming in SA4 and can be used for simulation purposes for conversational services since the IVAS_Codec is not yet available)
- Average data Rate : 256 / 512 kbps
- Packet Loss rate should be below 1e-3

c. Data Stream

- Average data Rate : <0.5 Mbps
- Packet Loss rate should be below 1e-3

3) Content Delivery Model

a. Video is identical to split rendering simulation in clause 6.3

b. Audio and data are for further study

4) Delivery receiver

a. Video is identical to split rendering simulation in clause 6.3 with following parameters

- Packets are dropped if late
- Slice loss model (1 lost packets per slice results in slice loss)
- Timestamp of slice if time stamp of latest packet of slice
- Maximum latency for slice: 80ms (see TR 26.928 [3], clause 4 and 6.2.5.1)

b. Audio and data are for further study

5) Decoding Model

a. Video is identical to split rendering simulation in clause 6.3

b. Audio and data are for further study

6) Quality evaluation tool.

a. Video is identical to split rendering simulation in clause 6.3

b. The following metrics are considered for each user and buffer

- IP Packet loss rate
- IP Packet late rate
- Slice loss rate
- Area loss rate (total amount of Coding Units)
- Area damage rate (total amount of Coding Units)

- Average encoded PSNR
- Average PSNR

c. Average over all buffers

d. Multi-user

- Average over all users
- Percentile of support

b. Audio and data are for further study

8.4 Simulation Framework

This work is for further study.

9 Viewport Dependent VR Streaming

9.1 Overview

The system design for viewport dependent 3DoF streaming is basically similar to the discussion and requirements from TR26.928 [3], clause 6.2.3. The architecture is shown in Figure 9.1-1.

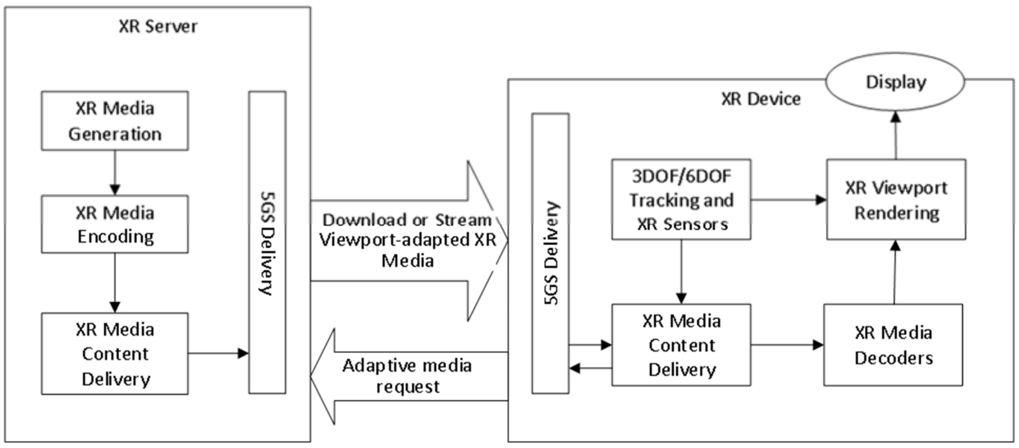


Figure 9.1-1 Viewport-dependent Streaming defined in TR26.928

Viewport Dependent 3DoF Streaming refers to the case where the XR server runs a 3DoF XR media generation engine, and the XR device collects local 3DoF tracking information to get the viewport related content to display.

According to Figure 1, the XR media is generated, encoded, and delivered by XR server depending on the adaptive media request sent by the XR device. The device works in two channels. One is to accept the XR media stream and the workflow of decoding, rendering, displaying is followed. Another is to use the tracking sensors to collect 3DoF parameters continuously, and 'translate' them into adaptive media request which will be sent to XR server by using 5GS delivery system.

The following call flow highlights the key steps:

1. An XR device connects to the network and joins Viewport Dependent 3DoF Streaming application
2. The device sends static device information and capabilities to server (supported decoders, display resolution)

3. Based on these information, the XR server sets up encoders and formats, and pre-spilts the XR scene into blocks (e.g. several tiles used in HEVC encoders)
4. Loop
    a. The device collects XR 3DoF pose
    b. XR pose is converted to media request by the device in order to cover the viewport of user and match the network condition
    c. The XR server responses to these requests and provides the media segments
    d. Compressed media is sent to XR device
    e. The device decodes the media content
    f. The device displays the media

According to TR 26.928, clause 6.2.3.5, for viewport dependent streaming, updated tracking and sensor information impacts the network interactivity. Typically, due to updated pose information, HTTP/TCP level information and responses are exchanged every 100-200 ms in viewport-dependent streaming.

In actual implementation process, in order to ensure the smooth transmission and viewing, sometimes it is adopted to transmit a channel of background data independent to the viewport in addition to the data described above, so as to replace these data to fill the playback when network condition is poor.

Attached to this document, please find a paper from IBC 2017 addressing network optimizations for VR Streaming. An author of the paper provided information beyond the paper.

It is considered that the model for VDP 3DOF streaming also applies for VDP 6DOF streaming.

Further information is provided based on the from the VR-IF Guidelines [7] for 3DoF Viewport dependent Streaming:

1) Downlink data rate ranges
    - for tiled streaming, 4K around 5 Mbit/s, up to 10.
    - 8K between 10 and 13, but can go under for static scenes and over for dynamic ones.
    - 5.5K stereo is the same as 8K mono.
    - Interestingly, for tiled streaming 360 or 180 make little difference if the viewport is the same.
2) Maximum packet delay budget in uplink and downlink → for mass distribution I don't care about the uplink much as long as the delay is not too jittery. Relate this to the segment size in an HLS pull – 2 secs is a good target segment size.
3) Maximum Packet Error Rate, → for mass distribution applications, preferably use something reliable, like HLS push or pull or a proprietary reliable transport.
4) Maximum Round Trip Time
    - for viewport-dependent, minimizing round-trip is crucial especially with head motion. That said there are no maximums.
    - There is a degradation as it increases. You should think in the order of frame duration, and know that head motion is about 500 degrees per second max, so 60 msec per tile, and that eyes also take some 60 msec to adapt to a new position, so you have a few frames to work with.

9.2 System Parameters

9.2.1 Content Delivery Setting

According to 3GPP TS 26.118 [6] and TR 26.923 [3], the tiled stream approach can be used for VR 360 video delivery. It allows emphasizing the current user viewport through transmitting non-viewport samples with decreased resolution, i.e. selecting the tiles from the viewport at a high-resolution version and the tiles that do not belong to the viewport at a lower resolution.

With tiled streaming approach, one or multiple decoders are needed, which depends on each of the tiles is encoded as motion-constrained HEVC tiles or separate video streams. When using motion constrained tile HEVC streams, the varying spatial resolution in video picture can be achieved by merging all tiles into a single common bitstream with one decoder needed. When using separately encoded video streams, several decoders are required at the receiver side.

Similarly to the separately encoded video streams, GSMA Cloud AR/VR white paper proposes the Tile Wise Streaming according to MPEG OMAF specification [D]. The tiled based approach with separately encoded video streams are used for Viewport Dependent 3DoF Streaming delivery method. The details can be summarized as below:

1. Split the original 8K 360° panoramic video into multiple tiles (such as 42 tiles) and generate a low-resolution 360° panoramic video (such as a 4K 360° panoramic video);
2. All tiles and the low-resolution panoramic video are packaged in DASH using spatial relationship descriptors and distributed via the CDN;
3. The client player retrieves low-resolution panoramic video and those tiles which are located at the user's field of view. When the user turns around and causes the field of view to change, the client requests new tiles from the CDN for the new field of view. The multiple tiles would be obtained separately and merged into the FoV-Area pictures in the receiver side. Because it requires a certain amount of time from when the new tiles are requested until they can be rendered, the low-resolution panoramic video will be displayed until the new tiles can be played.

9.2.2 Considered Content Formats

In this scenario, the vast majority of transmitted media content is 2D or 3D 360 degree video. According to TR 26.928, clause 4.2.1, it is commonly accepted that 2k by 2k per eye provides acceptable quality. Thus the equivalent content resolution will be 6k~8k at 90 ~ 110 degree FOV. Framerate is 60 fps for common cases and 30 fps for some low demand services.

H.264/AVC Progressive High Profile Level 5.1 with additional restrictions or H.265/HEVC Main-10 Profile Main Tier Profile Level 5.1 are recommended as encoding codecs by TR 26.928 in clause 4.5.1. Network adaptive encapsulation format is usually used to ensure smooth transmission, e.g. MPEG-DASH.

9.2.3 Considered System Parameters

The following system parameters are considered.

- Streaming: Types of streaming: video-on-demand
- Frame rate setting: 30fps
- Formats of transmitted content: ERP, 8K(7680 * 3840) , YUV 4:2:0
- Video Encoder configuration
    - Codec:H.265/HEVC
    - Rate control: VBR
    - Tile settings: 42 per frame
    - Tiles of FoV Area: 18
    - Complexity settings for encoder
    - Segment Length: 1.067s
- Audio Encoder configuration
    - Max Sampling Rate: 48 kHz
    - Operation Point (following 3GPP TS 26.118 [6] Table 6.1-1): 3GPP MPEG-H Audio

- Content Delivery
    - MPEG-DASH-based media delivery (tiled based approach with separately encoded video streams)
- RAN Configuration :
    - QoS Settings (5QI): GBR, Latency, Loss Rate, Mobility, Power Consumption
- Receiver configuration:
    - Buffer size: 3s
- E2E Downlink Budget:
    - 50ms

9.2.4 Output traffic characteristics

The following encoder output traffic characteristics are considered

- Data rate range:
    - per tiled streaming: 0.71~1.43 Mbps
    - FoV Area Streaming: (0.71~1.43)*18 Mbps
    - low-resolution 4K omnidirectional streaming: 6-8Mbps
- Periodical segment request: 1s per request
- Per tiled Segment size range: 8000 ~ 180000 byte
- Low resolution 4K omnidirectional segment size range: 848386~1681920 byte
- Packet size distribution: fixed size as 1500 bytes

9.2.5 Additional Discussion

Figure 9.2.5-1 provides an overview of the system design and delay components involved in viewport-dependent streaming according to the above referred paper.

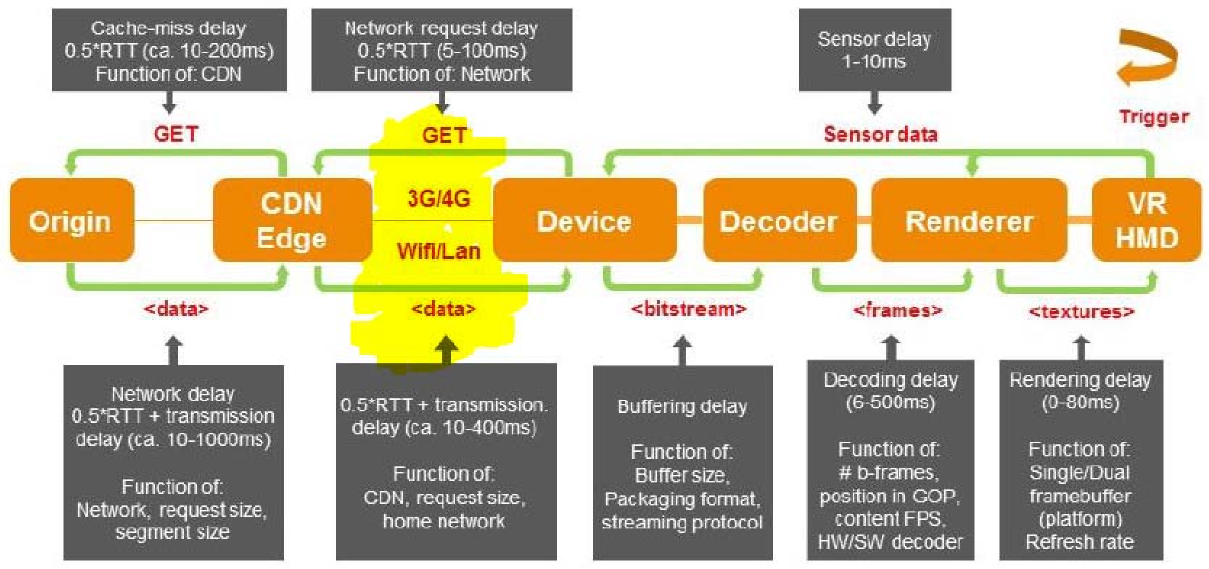
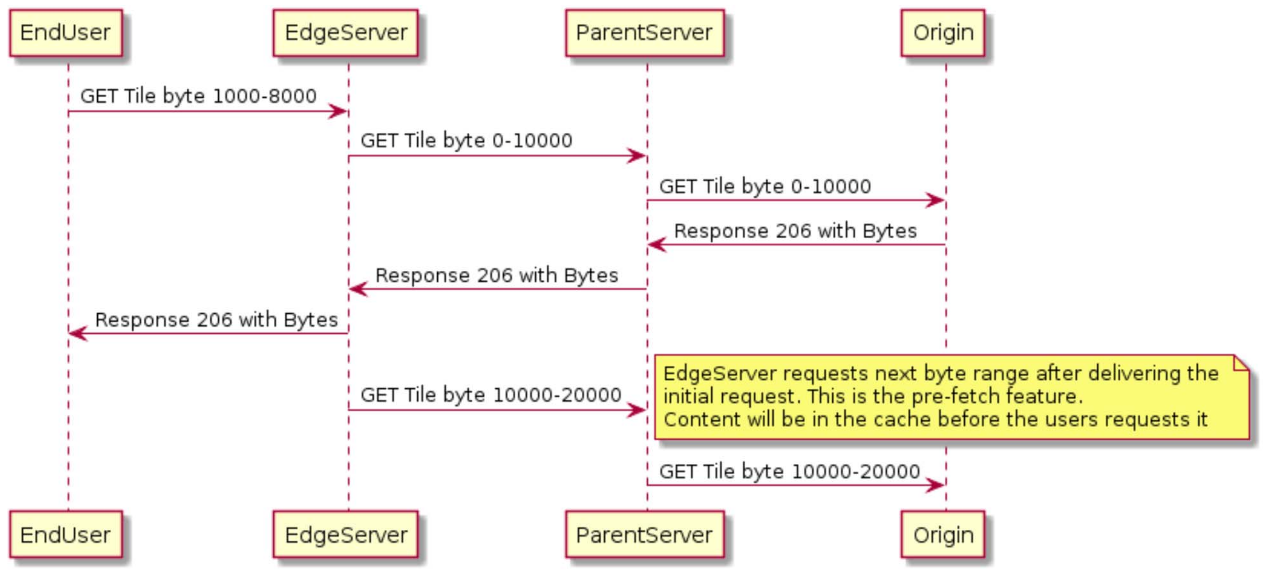


Figure 2 – Overview of latency factors in a Tiled VR Streaming system.

Figure 9.2.5-1 Overview of latency factors in a Tiled VR Streaming system (see [7])

Furthermore, Figure 9.2.5-2 provides the network exchange and caching process.



EdgeServer requests next byte range after delivering the initial request. This is the pre-fetch feature.
Content will be in the cache before the users requests it

Figure 9.2.5-2 Caching Process for VR Streaming (see [7])

Finally, clause 5 of the attached paper includes a typical set of test conditions. Tests with simulated (i.e., recorded), but significant head motion using 30 frames per second, 8k x 4k panorama encoded using 96 high-resolution tiles. The content was encoded using separate HEVC encoders for each of the tiles, with tile size fixed at 512 x 512 pixels. The bitrate from the edge to the end user was between 12 and 16 Mbit/s.

9.3 System Design and Simulation Model

9.3.1 Initial Considerations

The simulation model is similar to the one provided in Figure 9.2-1, but traces are not provided.

1) Content Model:
   - Not considered
2) Encoding Model
   - 96 tiles
   - HEVC
3) Content Delivery Model
   a) Options:
      - HTTP/1.1. (HTTP + TCP/IP)
      - HTTP3 (HTTP2 + QUIC + UDP)

   b) 96 concurrent requests every 2 seconds

   c) 20 in viewport at 500 kbit/s (Gaussian distributed with variance 100 kbit/s)

   d) 6 background tiles 500 kbit/s (Gaussian distributed with variance 100 kbit/s)

   e) Packet size is 1500 byte, provided in sequence

   f) Server to gNB bitrate is 1.5 times of the media bitrate

   g) Packet Trace settings
4) Delivery receiver
   a) HTTP1.1 => Object is impacted by the lost packet for each object
      - Model needs to be defined based on traces

   b) HTTP3 => Object is impacted by the lost packet for each object
      - Model needs to be defined based on traces

   d) Objects are either lost or delayed
5) Decoding Model
   - none
6) Quality evaluation tool. The following metrics are considered for each user and buffer
   a) IP Packet loss rate

   b) Viewport object loss rate

   c) Non-viewport object loss rate

   d) Streaming Metrics
      - Video stuck rate
      - Percentage of video playing stuck time
      - MTHR delay: average video time from low definition to high definition (FOV)

- Times of black edge/shadow/blur when turning head

e) Average over all buffers

f) Multi-user

- Average over all users
- Percentile of support

7) A model for the uplink traffic in a similar fashion also providing packet traces.

a) 26 HTTP requests are sent

b) HTTP/1.1 and HTTP 3 feedback for TCP/IP and QUIC

c) Uplink traffic is minimal

d) The end-to-end RTT delay and packet loss rate recorded by the test under different configurations can also be used as the actual network conditions.

e) Interaction: turning frequency

- Low frequency turned: like every 10s
- high frequency turned: like every ls

8) Potential RAN Configurations

- to be decided by RAN
- Devices: support QoS Profile. The recommended configuration of 5QI is as follows：

5QI	MinBR（Mbps）	Relative Priority Level (ratio to 5QI 9)
6	10	4:1
8	10	2:1
9	-	1:1

- Since the 5QI have mapping relation to characteristics of 5G QoS, the RAN configuration, such as Packet Delay Budget (PDB) and Packet Error Rate (PER), can be configured according to the value of 5QI. In the TS 23.501, table 5.7.4-1. shows standardized 5QI to QoS characteristics mapping. It should be noted that the PDB value in this table includes AN-PDB and CN-PDB which is a fixed value generally.

A summary is provided as follows:

Configuration	Basic Content Parameters
VR1-1	See above for HTTP/1.1
VR1-2	See above for HTTP/3

9.3.2 Open Issues

Among others, the following issues are for further study

- Late and loss model for objects based on HTTP/1.1 and HTTP3 taking into account radio model
- Uplink traffic model for objects based on HTTP/1.1 and HTTP3
- Definition of late threshold

- Quality impact for lost and late objects in viewport and not in viewport
- Packet trace model and mapping.

10 XR Distributed Computing

Traffic models XR Distributed computing are for further study.

Annex A: Example Content Coding Modelling

A.1 Introduction

Content encoding with different parameters is complex and so is decoding modelling with errors. Hence a simplified model is derived using model encoding and decoding. Different aspects are modelled in the following.

A.2 Encoding modeling

A.2.1 Impact of encoding parameters

A version of the video sequence is sent through a model encoder for identifying the impacts of different parameter settings. The following x.265 parameters are used resulting in total with 24 different configurations. The configurations take into account different frame rates, different quality factors, an all intra and an all inter predicted sequence, as well as no slices or 8 slices, as follows:

```
./x265 --input input.yuv --input-res 2048x2048 --preset medium --fps [30,60]
  --crf [22, 28,34] --keyint [-1,1] --slice [1,8] --csv
  /csv_analfiles/[i,p]_[30,60]_[22,28,34]_[1,8].csv --csv-log-level 2 --log-
  level full  --rc-lookahead 0 --no-deblock  --bframes 0  --frames 480 --psnr
  --ssim  --output /output/xxxx.h265
```

A summary of the bitrates and resulting quality is provided in Table A.2.1-1 for each of the 24 parameter settings.

Table A.2.1-1 Bitrates and resulting quality for different configurations

Mode	Frame rate	CRF	Slice	Data Rate in Mbit/s	Average QP	Average Rate Factor	Y PSNR	U PSNR	V PSNR	YUV PSNR	SSIM	SSIM(dB)	Frames
I	60	22	1	43.77	27.29	19.09	42.58	46.58	46.12	43.52	0.979	16.82	480
I	60	22	8	44.16	27.29	19.09	42.57	46.57	46.11	43.51	0.979	16.77	480
P	60	22	1	14.14	23.82	21.99	43.46	48.27	47.83	44.60	0.983	17.80	480
P	60	22	8	29.64	23.71	21.99	43.37	47.94	47.49	44.45	0.983	17.57	480
I	60	28	1	21.56	33.30	25.09	39.07	44.11	43.64	40.27	0.960	14.00	480
I	60	28	8	21.85	33.30	25.09	39.06	44.11	43.64	40.26	0.960	13.98	480
P	60	28	1	3.86	30.17	27.99	40.36	46.12	45.57	41.73	0.972	15.51	480
P	60	28	8	12.33	29.91	27.99	40.13	45.48	44.96	41.40	0.969	15.07	480
I	60	34	1	10.55	39.35	31.09	35.68	42.62	42.16	37.36	0.928	11.44	480
I	60	34	8	10.76	39.34	31.09	35.65	42.63	42.17	37.34	0.928	11.43	480
P	60	34	1	1.30	36.75	33.99	37.16	43.84	43.32	38.77	0.949	12.94	480
P	60	34	8	5.59	36.20	33.99	36.85	42.99	42.47	38.32	0.942	12.40	480
I	30	22	1	29.43	24.88	19.28	44.08	47.70	47.27	44.93	0.984	17.99	250
I	30	22	8	29.66	24.87	19.28	44.07	47.70	47.26	44.92	0.984	17.91	250
P	30	22	1	12.78	21.87	22.18	44.43	48.95	48.51	45.50	0.986	18.41	250
P	30	22	8	20.59	21.82	22.18	44.42	48.72	48.29	45.44	0.985	18.25	250
I	30	28	1	14.54	30.88	25.27	39.07	44.11	43.64	40.27	0.960	14.00	250
I	30	28	8	14.71	30.88	25.27	39.06	44.11	43.64	40.26	0.960	13.98	250
P	30	28	8	3.82	28.07	28.17	41.23	46.69	46.19	42.53	0.976	16.11	250
P	30	28	8	8.34	27.94	28.17	41.09	46.17	45.67	42.30	0.974	15.80	250
I	30	34	1	7.14	36.91	31.26	37.07	43.19	42.73	38.54	0.943	12.44	250
I	30	34	8	7.26	36.91	31.26	37.05	43.20	42.74	38.53	0.943	12.43	250
P	30	34	1	1.29	34.38	34.16	38.09	44.36	43.83	39.59	0.957	13.62	250
P	30	34	8	3.71	34.11	34.16	37.86	43.57	43.12	39.23	0.952	13.21	250

An analysis of the bitrate decrease per CRF/QP increase is shown in Table A.2.1-2.

Table A.2.1-2 Bitrate decrease per CRF/QP increase

1 Slices		
I only		P only

22->28/60	28->34/60	22->28/30	28->34/30		22->28/60	28->34/60	22->28/30	28->34/30
2.03	2.04	2.02	2.04		3.85	3.23	3.46	3.18
8 Slices								
I only					P only			
22->28/60	28->34/60	22->28/30	28->34/30		22->28/60	28->34/60	22->28/30	28->34/30
2.02	2.04	2.02	2.04		3.85	3.23	3.46	3.18

The bitrate increase for I-frames vs P-frames for different configurations is shown in Table A.2.1-3.

Table A.2.1-3 Bitrate decrease per CRF/QP increase

Frame	I/P 22/60/1	I/P 28/60/1	I/P 34/60/1	I/P 22/60/8	I/P 28/60/8	I/P 34/60/8
Average	3.17	6.10	10.03	1.51	1.81	1.81

Finally, the bitrate increase for 60fps vs 30fps is shown in Table A.2.1-4.

Table A.2.1-4 Bitrate increase for 60 fps vs. 30fps

Frame	22-1	28-1	34-1	22-8	28-8	34-8
Average	1.83	2.13	2.28	1.39	1.38	1.36

From these results some initial conclusions:

- Slices as available in x265 being constrained across pictures are very costly if used without prediction across slice boundaries
- Prediction over 2 P-frames results roughly in factor 2
- Rate decrease per CRF/QP for I is consistently 2 for QP increase 6
- Rate decrease per CRF/QP for P is consistently between 3 and 4 for QP increase 6
- Bitrate increase for I frame coding when doubling frame rate is typically in the range of 2.
- Bitrate increase for P frame coding when doubling frame rate is typically in the range of 1.4.

A.2.2 Global Configuration

The following global configuration parameters are considered (considered, *preferred*, ~~ignored~~)

- Bitrate Control (one of the following):
  - Constant Bitrate – bitrate & buffer
  - Feedback-based Variable Bitrate - dynamic
  - *Constant Rate Factor – rate factor with CRF (default: CRFref is used)*
- Slice Setting (one of the following)
  - Default – no slices
  - number of slices – number of slices (typical numbers are 4, 8, 16)
  - ~~maximum slice size – number in bytes~~
- Error Resilience (one of the following)
  - Intra-refresh frame parameter would be the period (default is no intra refresh)

- Intra-refresh slice: period (1 ➔ 1 slice every 1 frame with the slice being picked as POC mod #slices) ~~2 ➔ 1 slice every 2 frames~~)
- Feedback-based
  - Mode:
    - *intra refresh: add an intra for the lost slice*
    - ACK-mode: only use acknowledge slices in prediction
    - NACK-mode: use an old reference frame or intra in case of loss

A.2.3 Dynamic status:

Dynamic feedback status may be considered as follows

- Max number of bits for next frame (external rate control)
- frame Number, slice number ACK/NACK/unknown

A.2.4 Content Encoding Modelling

The content encoding per frame is modelled as shown in Listing A.2.4-1.

Listing A.2.4-1 Content Encoding per frame

```
•   Create a map of slices, CTU maps (64 x 64) and reference frames
        •   Example: 2048 x 2048, 8 slices, 3 reference frames
                •   Addresses for 2048 / (8 * 64) = 4 rows with 32 CTUs for 8 slices in 3 frames maintained.
                •   For each CTU store encoding mode:
                        •   intra/inter+merge/skip
                        •   largest reference frame (only previous one is used in simple config)
•   For frame i from V-Trace (based on sample time)
        •   Read frame i from V-Trace (timestamp)
        •   Read latest dynamic information from dynamic status info, if applicable
        •   Do a model encoding
                •   Input parameters: V-Trace, global configuration, dynamic status (if applicable)
                •   Output: s slices with parameters
-   Based on the map of intra and inter for each slice/MB
        o   Constant CRF
                ▪   Re-use the CRFref for which the Trace was generated.
                ▪   For every CTU
                        •   Take P-frame intra/inter/merge/skip percentage and decide
                            intra/inter/merge/skip randomly. This is done that the number of MBs
                            is matching the trace entry.
                ▪   For every slice,
                        •   apply intra/(inter + reference) decision as follows
                                o   Periodic: slice mode follows pattern, other inter + reference 1
                                o   Feedback intra: slice was lost => intra, else inter + reference 1
                                o   Feedback ack: only ACK slices => reference inter + reference
                                    backward (typically more than 1), such that the slice was acked.
                                o   Feedback NACK: slice nack => reference inter + reference
                                    backward (typically more than 1) prior to NACK or intra.
                        •   For intra slices,
                                o   overwrite the above CTU decision and make it an intra
                        •   draw a number of bits for the slice
                                o   that takes into account
```

            - the type of the CTU
            - the information in the trace file
        - the total amount of the bits for this slice by the following modelling
            - intra size:
                - take total intra size and divide by number of CTUs as medium value
                - apply Gaussian drawing with 10% variance of the total number of bits
                - Use the CRF and apply adjustment as follows
                    - FinalBits = Bits * pow(2, (CRFref – CRF)/6)
                    - QPnew = QPref - (CRF – CRFref)
            - skip:
                - 1 byte
            - merge + inter
                - *compute medium value as total inter size (total P-size - intra-percentage*intra-size – skip-percentage) and divide by number of inter CTUs (total CTUs*(1 – intra_percentage – skip_percentage))*
                - *reference frame 1*
                    - *apply Gaussian drawing with 20% variance of the total number of bits with medium value*
                    - *Use the CRF and apply adjustment as follows*
                        - *FinalBits = Bits * pow(2, (CRFref – CRF)/6)*
                        - *QPnew = QPref - (CRF – CRFref)*
                - reference frame more than 1, it is X
                    - take total inter size and divide by number of CTUs as medium value
                    - multiply the medium value with X
                    - apply Gaussian drawing with 20% variance of the total number of bits
                    - do also intra test as above, if intra is lower, apply intra, else this inter mode.
                    - Use the CRF and apply adjustment as follows
                        - FinalBits = Bits * pow(2, (CRFref – CRF)/6)
                        - QPnew = QPref - (CRF – CRFref)
        - sum up the size of each CTU for the slice
- Dump the following information:
    - Slice Timing/frame count
    - Left or right eye
    - Slice availability (after the slice timing) relative to 0.

                - Without encoding delay this is the same as the slice. time.
            - Quality/QPnew
            - New PSNR – add a function
            - Slice size
            - Slice type
            - CTU types
    - Bitrate constrained – a total max of bits available max_bits
        - Do the same as for constant CRF
        - Iterate to the smallest CRF that fulfill max_bits

The following issues are not yet included:

- Modelling of encoding times – no priority
    - Model encoding delay.
- two independent buffers for left and right eye need to be addressed.
    - for now create the same process for
        - Option 1: Same timing
        - Option 2: staggered left right at half the frame rate
- Can a new PSNR model for the updated QP be derived?
    - check for model
    - In the absence of a model, 1 QP step up reduces PSNR by 1dB (linear)
- What about slice modelling and all the issues that we saw
    - Ignore slice modelling for now, no bitrate change compared to not using slices.
- Also ACK/NACK based feedback needs to be addressed.
    - NACK already addressed above.
    - ACK you only take acknowledged slice/frames for references – this basically extends the reference frame the feedback delay per slice.

A.2.5 Rate Control

The rate control is modelled as shown in Listing A.2.5-1.

Listing A.2.5-1 Rate Control Modelling

- Parameters
    - Total average bit budget per frame: Tbits
    - Window size is W
    - Allocated bits to frame i: Bits[i]
    - Allocated bits in model encoding (from V-Trace): A[i]
    - Forced intra period – intra_period = 1, 2, … (0 means no forced intra)
- Algorithm
    - Available Bits B[i] = Tbits*W – sum (j=i-W+1) (i-1) Bits[j]
    - If A[i] <= Tbits + (B[i] – Tbits)/(W/factor)
    - Then
        - Bits[i] = A[i]
    - Else

```
        • Increase QP such that Bits [i] results in both of the following being fulfilled
            • sum (j=i-W+1) (i) Bits[j] < Tbits (total budget not exceeded over window)
            • If intra_period is 0: Bits[i] <= Tbits + (B[i] – Tbits)/(W/factor) (only take a fair portion
              of the excess bitrate such that you do not overshoot too much)
            • if intra_period is > 0: Bits[i] <= (Tbits + (B[i] – Tbits)/(W/factor)) *
              intra_period/((intra_period -1) + ifactor)
• Factor is set to 2. Ifactor is set to 3.
```

A.3 Decoding modeling

A.3.1 Overview

Quality Evaluation is based on two aspects, namely the encoding quality and the quality degradation due to lost slices. This evaluation is shown in Figure A.3.1-1.

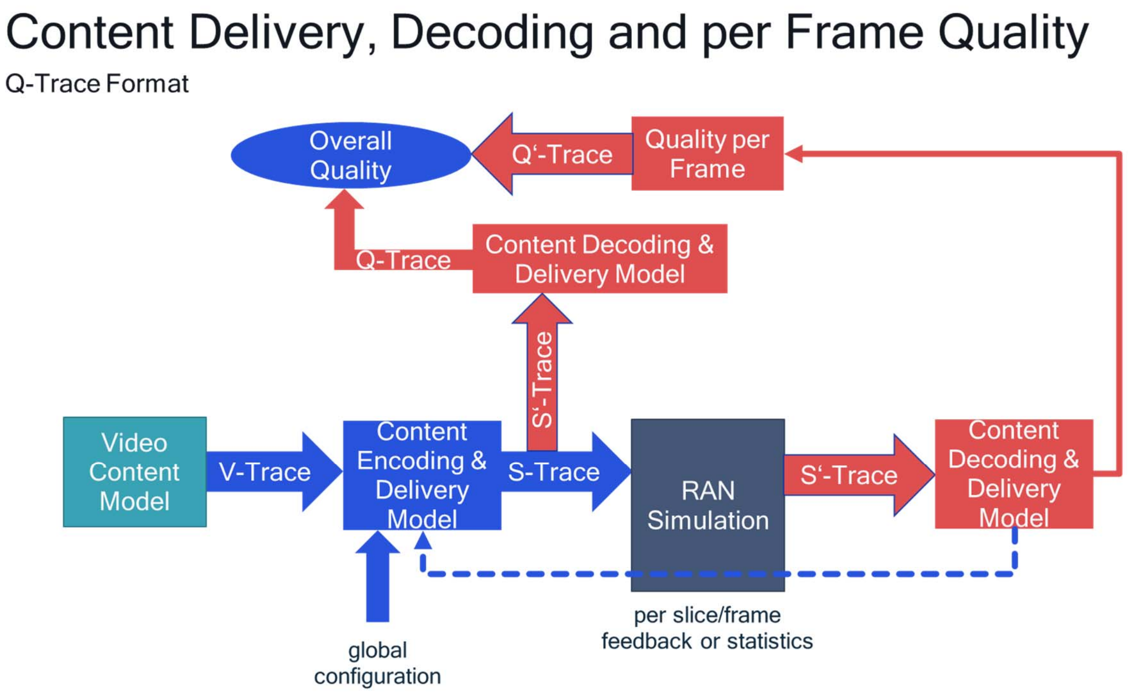

Content Delivery, Decoding and per Frame Quality
Q-Trace Format
Overall Quality
Q'-Trace
Quality per Frame
Q-Trace
Content Decoding & Delivery Model
S'-Trace
Video Content Model
V-Trace
Content Encoding & Delivery Model
S-Trace
RAN Simulation
S'-Trace
Content Decoding & Delivery Model
global configuration
per slice/frame feedback or statistics

Figure A.3.1-1 Content Delivery, decoding and quality modeling

The following simulation is defined for identifying damaged macroblocks:

- Keep a state for each macroblock
    - Damaged
    - Correct
- Macroblock is damaged
    - If it is part of a slice that is lost for this transmission
    - If it is correctly received, but it predicts from a wrong macroblock
- Macroblock is correct
    - If it is received correctly and it predicts for a non-damaged macroblock

- Predicting from non-damaged macroblock means
  - Spatial prediction is correct
  - Temporal prediction is correct
  - Recovering MBs done by Intra Refresh and predicting from correct MBs again.

Depending on the configuration and the setting of the delivered video quality, different results may be obtained. An example is shown in Figure A.3.1-2.

A quality threshold may for example be to have at most 0.1 % of damaged video area. Also the quality of the original content may be a threshold. Details are ffs.

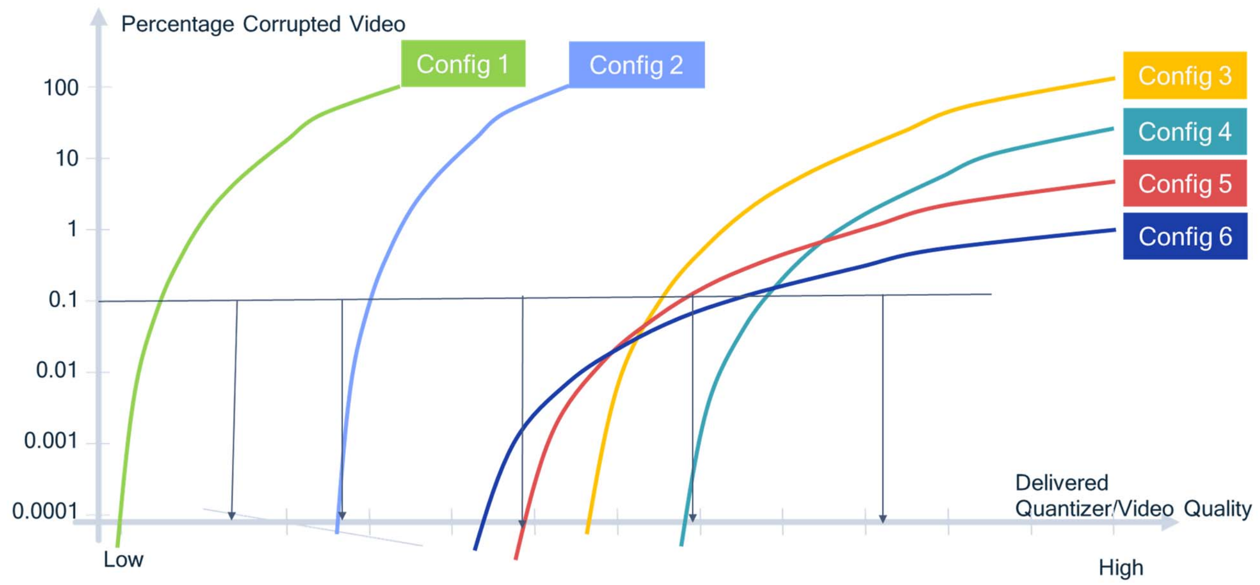


Figure A.3.1-2 Potential evaluation graph for different configurations

A.3.2 Slice Recovery

At the recovery the following happens for recovering RTP/IP packet traces:

- if one packet of a slice is lost the entire slice is lost
- the delay of the latest packet determines the arrival time of the slice.
  - Dump the following information:
    - Slice Timing/frame count
    - Left or right eye
    - Slice availability (after the slice timing) relative to 0.
      - The time it took through system
      - It can also be infinite = lost
    - Quality/QPnew
    - New PSNR – add a function
    - Slice size

    - Slice type
    - CTU types

The result is a slice delay trace

- Slice Timing/frame count
- Left or right eye
- Slice availability (after the slice timing) relative to 0.
    - The time it took through system
    - It can also be infinite = lost
- Quality/QPnew
- New PSNR – add a function
- Slice size
- Slice type
- CTU types

A.3.3 Quality Measurement

The quality measurement algorithm is provided in Listing A.3.3-1.

Listing A.2.5-1 Quality Measurement Algorithm

```
Input information
  - Trace of
      - Slice Timing/frame count
      - Left or right eye
      - Slice availability (after the slice timing) relative to 0.
          - The time it took through system
          - It can also be infinite = lost
      - Quality/QPnew
      - New PSNR – add a function
      - Slice size
      - Slice type
      - CTU types
Run the following algorithm:
•   Input parameters.
        •   Parameters of source
                •   Resolution
                •   reference frames
        •   Slice Trace
        •   Loss delay
        •   Decoding delay: do we want to model decoding of late arriving slices, i.e. they can still be decoded and
            used as reference, but not as part of the presentation. For now this is not assumed.
•   Create a map of slices, CTU maps (64 x 64) and reference frames
```

- Example: 2048 x 2048, 8 slices, 3 reference frames
    - Addresses for 2048 / (8 * 64) = 4 rows with 32 CTUs for 8 slices in 3 frames maintained.
    - For each CTU of each frame, store mode:
        - Correct
        - Damaged
        - Unavailable
    - Initialize all CTUs as unavailable
- For each frame i
    - Get all slices from trace for the frame
    - For all slices that are lost or later than decoding delay
        - Mark all CTUs as unavailable
        - Indicate the slice loss for feedback
    - For all slices are received
        - Indicate the slice received for “feedback”
        - If it is an intra CTU, mark it correct
        - If it is an inter CTU and it *references* a damaged or unavailable CTU, mark it as damaged, otherwise mark it as correct
            - *Referencing* is determined as follows (note a better model may be developed in the future)
                - The CTU in the new frame references the CTU at the same position in the referencing frame is 100%
                - The probability of referencing a neighbouring CTU top/bottom/left/right is 50%
                - The probability of referencing a neighbouring CTU is 50%, if one of the two top/botton/left/right is referenced, and 100% if both are referenced, and is 0% if none are referenced.
    - Compute the totally unavailable and damaged CTUs in this frame
    - Compute the average PSNR for this frame
        - avPSNR = PSNR * correctCTUs/totalCTUs + PSNRwrong (1- correctCTUs/totalCTUs) with PSNRwrong = 0
- Run this independently for each eye buffer

A.3.4 Video Decoding and Reconstruction

A.3.4.1 Overview

Video decoding and reconstruction primarily addresses to identify areas of the image that are correct and those that are damaged. It also includes the encoding quality of correctly received images.

The reconstruction quality evaluation is based on two aspects, namely the encoding quality and the quality degradation due to lost and late packets. This evaluation is shown in Figure A.3.4.1-1.

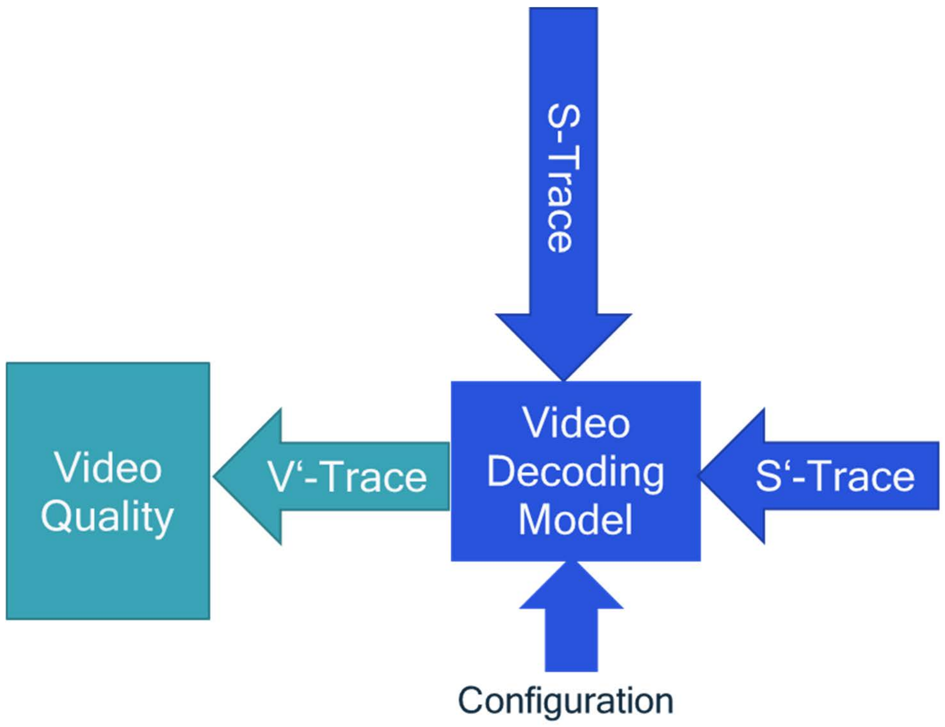


Figure A.3.4.1-1 Video Decoding Model

Video decoding is based on input from S'-Trace and S-Trace, resulting in V'-Trace.

The following simulation is proposed for identifying damaged CUs:

- Keep a state for each CU
  - Damaged
  - Correct
- CU is damaged
  - If it is part of a slice that is lost for this transmission
  - If it is correctly received, but it predicts from a wrong CU
- CU is correct
  - If it is received correctly and it predicts for a non-damaged CU
- Predicting from non-damaged CU means
  - Spatial prediction is correct
  - Temporal prediction is correct
- Recovering CU done by Intra Refresh and predicting from correct CUs again.

Detailed modelling is provided in clause A.3.4.3.

A.3.4.2 Configuration

No configuration is applied, but the input parameters are the S and S'-Trace.

A.3.4.3 Modelling

Input information

- S-Trace
    - Slice index
    - Slice metadata for reconstruction
        - For every CU
            - Size
            - Mode
            - Reference
            - Quality/QPnew
            - PSNR/PSNRnew
- S'-Trace
    - Slice index
    - Slice availability time
    - Recovered slice position (0 => lost, in between, full)

Run the following algorithm:

- Input parameters.
    - Parameters of source
        - Resolution
        - reference frames
            - S'-Trace
- Create a map of slices, CU maps (64 x 64) and reference frames
    - Example: 2048 x 2048, 8 slices, 3 reference frames
        - Addresses for 2048 / (8 * 64) = 4 rows with 32 CUs for 8 slices in 3 frames maintained.
        - For each CU of each frame, store mode:
            - Correct
            - Damaged
            - Unavailable
        - Initialize all CUs as unavailable
- For each frame i
    - Get all slices from trace for the frame
    - For all slices that are lost
        - Mark all CUs as unavailable
        - Indicate the slice loss for feedback
    - For all slices are received

    - Indicate the slice received for "feedback"
    - If it is an intra CU, mark it correct
    - If it is an inter CU and it *references* a damaged or unavailable CU, mark it as damaged, otherwise mark it as correct
        - *Referencing* is determined as follows (note a better model may be developed in the future)
            - The CU in the new frame references the CU at the same position in the referencing frame is 100%
            - The probability of referencing a neighbouring CU top/bottom/left/right is 50%
            - The probability of referencing a neighbouring CU is 50%, if one of the two top/botton/left/right is referenced, and 100% if both are referenced, and is 0% if none are referenced.
  - Compute the totally unavailable and damaged CUs in this frame
  - Compute the average PSNR for this frame
    - avPSNR = PSNR * correctCUs/totalCUs + PSNRwrong (1- correctCUs/totalCUs) with PSNRwrong = 0
- Run this independently for each eye buffer
- Dump this information into a V'-Trace according to format in 5.6.4.
- Add the availability time stamp

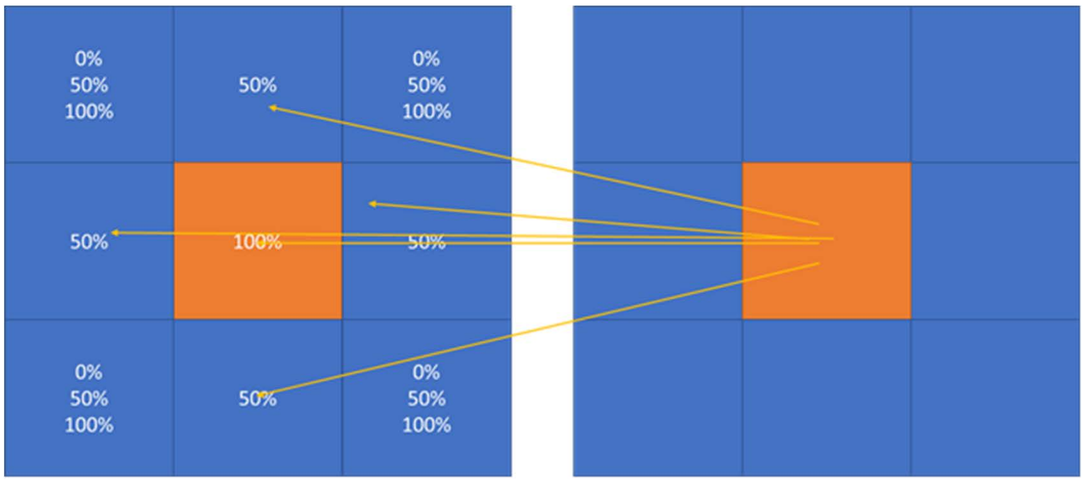


Figure A.3.4.3-1 Probability for wrong CU propagation inter

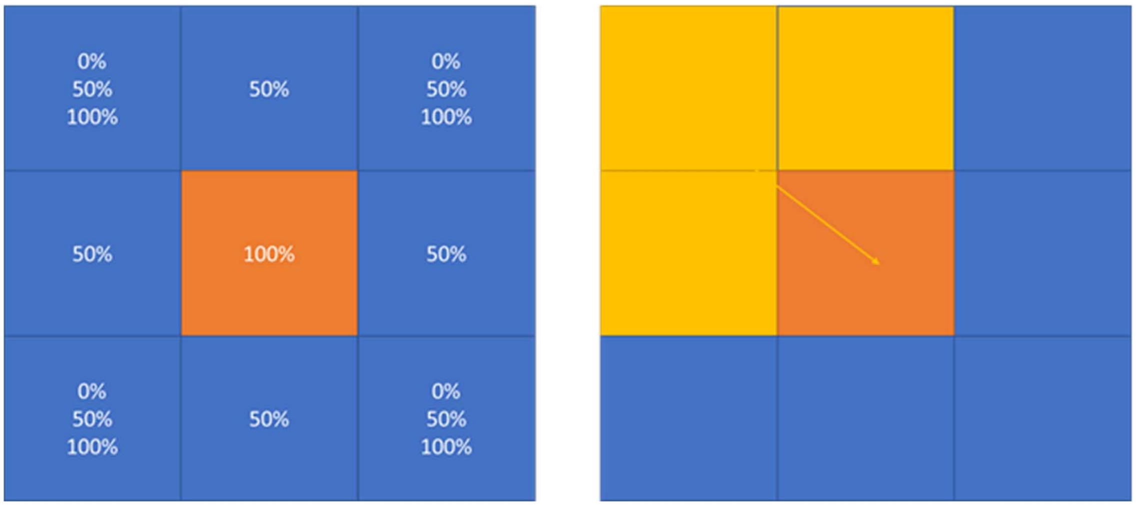


Figure A.3.4.3-2 Probability for wrong CU propagation intra

Annex B: Cloud Gaming & XR Traffic Model

B.1 Introduction

In order to have some insights into realistic traffic models for cloud gaming and XR, measurements are derived traffic models from different gaming platforms such as Stadia ™, GEFORCE Now ™, PS Now ™ and different game types such as role player games or racing. It is considered to use rendering in the edge (with the operators network) and rendering in the cloud with a public Internet connection.

The following assumptions where taken for a cloud gaming:

- Platform: Google Stadia
- Game: misc. e.g. Grid (Racing)
- Frame rate: 60fps
- Screen resolution: 4K
- Capturing packets from Router
- WiFi AP: .11ac, 5GHz, 80MHz, 867Mbps
- Wireline: Data rate: 70Mbps DL / 20Mbps UL
- Ping: 20ms

B.2 Results

From the results of the measurement a bursty traffic model is obtained with burst periodicity of indirectly proportional to the frame rate of the video content. Packets are of fixed size, with almost fixed inter-packets arrival time. The burst Length follows truncated Gaussian distribution.

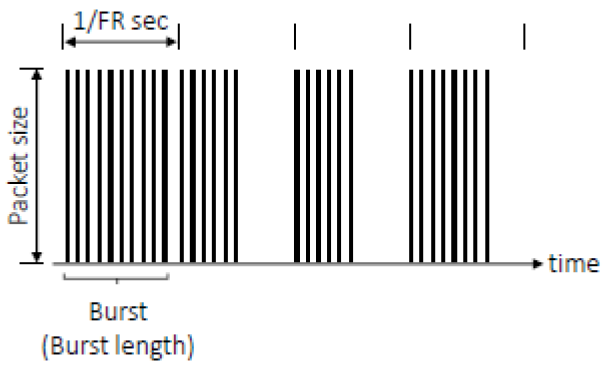
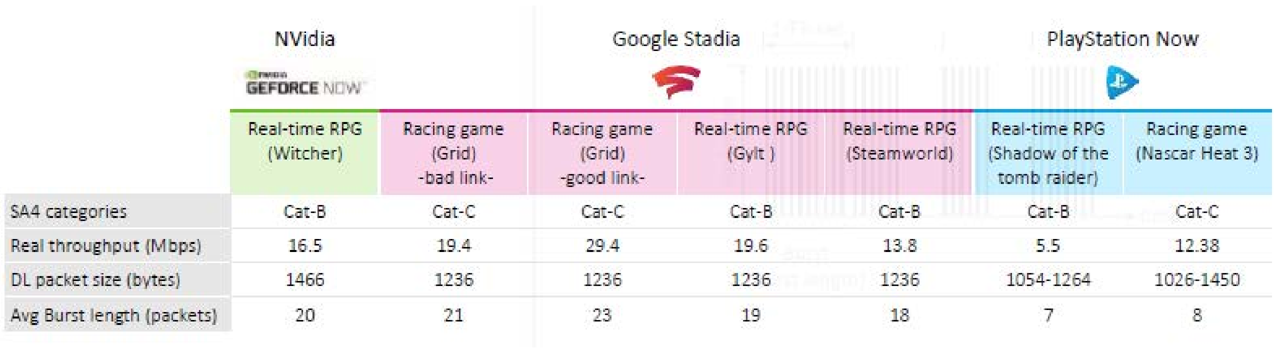


Figure B.2-1 Illustration of Traffic model

Figure B.2-2 provides the results for different games on different platforms in terms of throughput, downlink packet sizes and burst lengths for:

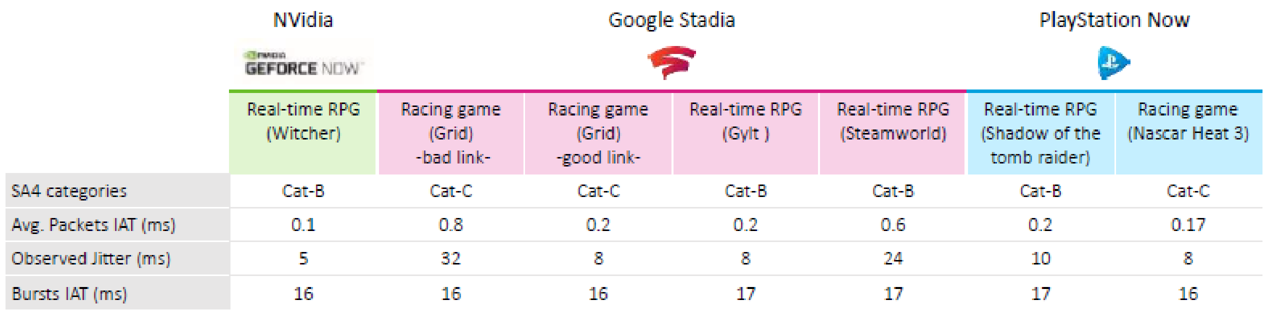
- CAT-B: High dynamicity & low/med complexity games
- CAT-C: High dynamicity & high complexity games



	NVidia	Google Stadia				PlayStation Now	
	Real-time RPG (Witcher)	Racing game (Grid) -bad link-	Racing game (Grid) -good link-	Real-time RPG (Gylt)	Real-time RPG (Steamworld)	Real-time RPG (Shadow of the tomb raider)	Racing game (Nascar Heat 3)
SA4 categories	Cat-B	Cat-C	Cat-C	Cat-B	Cat-B	Cat-B	Cat-C
Real throughput (Mbps)	16.5	19.4	29.4	19.6	13.8	5.5	12.38
DL packet size (bytes)	1466	1236	1236	1236	1236	1054-1264	1026-1450
Avg Burst length (packets)	20	21	23	19	18	7	8

Figure B.2-2Statistics for different games and platforms: Throughput, packet size, burst length

Figure B.2-3 provides the results for different games on different platforms in terms of Inter arrival times (IAT), jitter, and burst IAT.



	NVidia	Google Stadia				PlayStation Now	
	Real-time RPG (Witcher)	Racing game (Grid) -bad link-	Racing game (Grid) -good link-	Real-time RPG (Gylt)	Real-time RPG (Steamworld)	Real-time RPG (Shadow of the tomb raider)	Racing game (Nascar Heat 3)
SA4 categories	Cat-B	Cat-C	Cat-C	Cat-B	Cat-B	Cat-B	Cat-C
Avg. Packets IAT (ms)	0.1	0.8	0.2	0.2	0.6	0.2	0.17
Observed Jitter (ms)	5	32	8	8	24	10	8
Bursts IAT (ms)	16	16	16	17	17	17	16

Figure B.2-3Statistics for different games and platforms: Inter arrival times (IAT), jitter, burst IAT

B.3 Derived Traffic Models

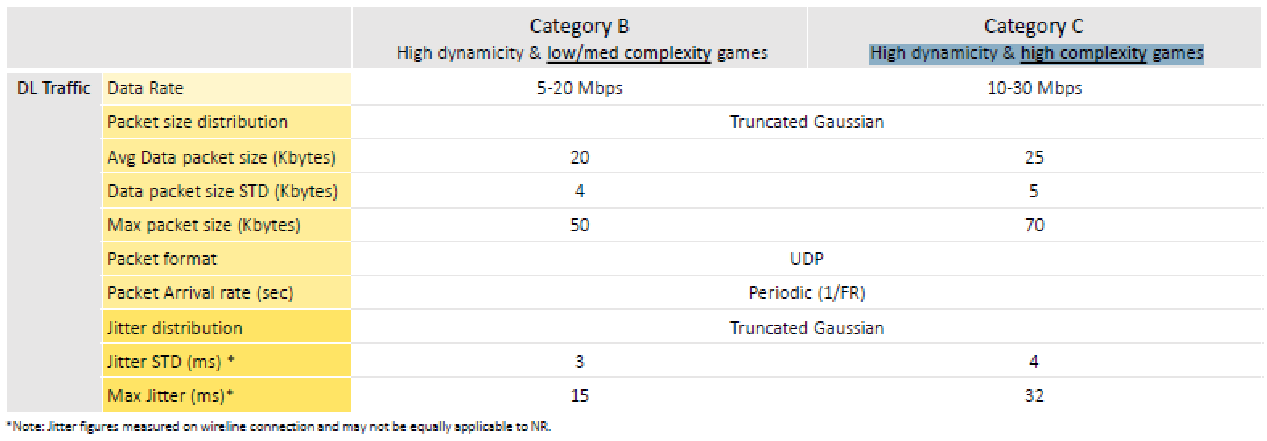
B.3.1 Traffic Models

Based on the measurement and results, the following traffic models are proposed for cloud-gamin:

- Packet Size & Traffic Shape
  - Fixed packet size. E.g. 1500 bytes
  - Bursty (Segmentation based on Ethernet MTU size limit)
- Arrival time
  - Inter-burst period of 1/FR sec
  - Burst length follows truncated Gaussian distribution and depends on channel conditions and games/XR req.
  - Jitter

B.3.2 Parameters

Figure B.3.2-1 provides recommended parameters for cloud-gaming traffic



		Category B High dynamicity & low/med complexity games	Category C High dynamicity & high complexity games
DL Traffic	Data Rate	5-20 Mbps	10-30 Mbps
	Packet size distribution	Truncated Gaussian	
	Avg Data packet size (Kbytes)	20	25
	Data packet size STD (Kbytes)	4	5
	Max packet size (Kbytes)	50	70
	Packet format	UDP	
	Packet Arrival rate (sec)	Periodic (1/FR)	
	Jitter distribution	Truncated Gaussian	
	Jitter STD (ms) *	3	4
	Max Jitter (ms)*	15	32

*Note: Jitter figures measured on wireline connection and may not be equally applicable to NR.

Figure B.3.2-1 recommended parameters for cloud-gaming traffic

B.3.3 Jitter Modelling

Jitter is the effect of variable packet delays observed in the system. Reasons of jitter

- XR Server at the Cloud (outside 3GPP network): Variation in frame encoding time, Network routing and congestion, gNB buffering latency
- XR server at the Edge (within 3GPP network): Network routing and gNB buffering latency

Jitter affects QoE, i.e. results indicate that, even for slight variations of the underlying jitter distribution, the QoE distribution shows significant variations. Jitter can lead to high power consumption due to uncertainty in the arrival time of the periodic traffic could impact the DRX operation affecting power consumption. Edge compute may reduce baseline latency, but congestion in access still causes some jitter.

Annex C: (informative): Change history

Change history							
Date	**Meeting**	**TDoc**	**CR**	**Rev**	**Cat**	**Subject/Comment**	**New version**
2021-04	SA4#113e	S4-210663				Initial Version	0.1.0
2021-05	SA4#114e	S4-210894				Version agreed during SA4#114e (adds S4-210750)	0.2.0
2021-08	SA4#115e	S4-211208				Version agreed during SA4#115e (adds S4-211028)	0.3.0
2021-11	SA4#116e	S4-211208				Version agreed during SA4#116e (adds S4-211028)	0.4.0
2021-12	SA#94-e	SP-21339				For Presentation to plenary	1.0.0
2022-02	SA4#117e	S4-220220				Version agreed during SA4#117e (adds S4-220220)	1.1.0
2022-08	SA4#120e	S4-221170				Version agreed during SA4#120e (adds S4-221170)	1.2.0
2022-11	SA4#121	S4-221325				Version agreed during SA4#121 (adds S4-221325)	1.3.0
2023-02	SA4#122	S4-230293				Version agreed during SA4#122 (adds S4-230293)	1.4.0
2023-05	SA4#124	S4-230990				Version agreed during SA4#124 (adds S4-230810)	1.5.0
2023-08	SA4#125	S4-231413				Version agreed during SA4#125 (adds S4-231217 and S4-231219)	1.6.0
2023-09	SA#101	SP-230979				Version 2.0.0 created by MCC to be sent to TSG SA#101 for approval	2.0.0
2023-09	SA#101					Version 18.0.0 created by MCC	18.0.0
2023-12	SA#102	SP-231372	0001	2	B	[FS_XRTraffic] Application Layer FEC Traffic characteristics	18.1.0
2024-03	SA#103	SP-240043	0002	1	C	[FS_XRTraffic] RTP-based Application Layer FEC	18.2.0

History

Document history		
V18.2.0	May 2024	Publication