TÉCNICO LISBOA

Viewport Adaptive Streaming for Omnidirectional Video Delivery

Miguel Mateus D'Avó Luís

Thesis to obtain the Master of Science Degree in

Aerospace Engineering

Supervisors: Prof. Maria Paula dos Santos Queluz Rodrigues

Prof. João Miguel Duarte Ascenso

Examination Committee

Chairperson: Prof. José Fernando Alves da Silva

Supervisor: Prof. Maria Paula dos Santos Queluz Rodrigues

Member of the Committee: Prof. Pedro António Amado Assunção

July 2020

Acknowledgements

First, I would like to show my gratitude to my supervisors, Professor Maria Paula Queluz and Professor João Ascenso for their guidance, patience, and constant support throughout the development of this thesis.

I would like to thank my mother Zélia and my father Paulo for their love and support during all these years of academic life, especially during the particularly different last few months.

I would also like to thank my sister Beatriz and her fiancé Lars for their advices.

I would like to thank my family and friends.

This work was funded by FCT/MCTES through national funds, and when applicable co-funded by EU funds, under the project UID/EEA/50008/2019. I would like also to thank Instituto de Telecomunicações for providing me the required means for the completion of this dissertation.

I declare that this document is an original work of my own authorship and that fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Abstract

Nowadays, there is an increasing interest in 360º or omnidirectional video, which allows the user to navigate through the video by changing its viewing direction. Although omnidirectional videos can be displayed on conventional 2D monitors, a much better immersive experience can be achieved using head mounted displays (HMD). Regardless of the used display type, the user only sees a fraction – known as *viewport* – of the entire sphere, at a time. Accordingly, and in order to reduce the transmission bandwidth, several viewport adaptive streaming (VAS) strategies have been proposed in the literature; these provide the user's viewport with higher quality (and higher bitrate), while the remaining part of the video is transmitted with lower quality. Although several VAS strategies have been proposed in the literature, there is still no generally recognized optimal solution.

In this thesis, the offset cubemap projection (OCM) – a VAS strategy developed by Facebook's Oculus – is optimized to provide an enhanced omnidirectional video streaming strategy. The OCM projection allows higher spatial resolution (thus, higher quality) around a predefined direction of the omnidirectional video – the offset orientation – by distorting spherical angles. The Oculus implementation of the OCM streaming considers a fix offset magnitude of 0.7, which was proven (in this thesis) to not provide the optimal viewport quality *vs.* bitrate results. To determine an appropriate offset value, for improved quality, 11 offset values were objectively evaluated using three omnidirectional videos, having distinct spatial-temporal activities. The best offset magnitude value was found to be 0.42, allowing viewport quality gains (measured using the V-PSNR metric) between 1.7 dB and 2.3 dB, relatively to an offset magnitude of 0.7.

The assessment of the optimized OCM projection, for adaptive omnidirectional video streaming, was then conducted considering several state-of-the-art streaming strategies as benchmarks; these were of two types: 1) the monolithic or conventional strategies - which encode the entire omnidirectional frame with the same quality; 2) tiles-based strategies, which use tiles encoded with different quality/spatial resolution to provide VAS. For static viewports, OCM based video streaming achieved better results than the benchmarks, either when the requested (to the server) viewport center orientation matched the viewing direction, either when an error of up to 20° between these two directions was introduced. Using previously recorded head trajectories from real users, the OCM based streaming provided better viewport quality than the conventional monolithic streaming, and competed with the considered tiled based streaming strategies.

Keywords: Omnidirectional Video Streaming, Offset Cubemap Projection, Tile-based Streaming, Objective Quality Assessment.

Resumo

Atualmente, há um interesse crescente em vídeos 360° ou omnidirecionais, em que é permitido ao utilizador navegar no conteúdo da imagem reproduzida no ecrã, alterando a direcção de visualização. Embora os vídeos omnidirecionais possam ser visualizados em monitores 2D convencionais, obtém-se uma melhor experiência imersiva usando *head mounted displays* (HMD). Independentemente do tipo de ecrã usado, em cada momento o utilizador observa apenas uma fracção – designada por *viewport* – da imagem esférica. Consequentemente, e com o intuito de reduzir a largura de banda necessária para a transmissão, têm sido propostas na literatura estratégias de *streaming* adaptadas ao *viewpor*t (*viewport adaptive streaming* – VAS), que se baseiam na transmissão do conteúdo do *viewport* com qualidade superior (e taxa de bits mais alta) à do restante conteúdo do vídeo. Apesar de terem sido até agora propostas na literatura várias estratégias de VAS, não existe uma solução reconhecida globalmente como sendo a óptima.

Nesta tese de mestrado, optimizou-se a projeção *offset cubemap* (OCM) – uma solução de VAS desenvolvida pela empresa Oculus da Facebook – de forma a obter uma solução melhorada de *streaming* de vídeo omnidirecional. A projeção OCM permite uma resolução espacial mais alta (e em consequência, uma melhor qualidade) em torno de uma direção pré-definida do vídeo omnidirecional – designada por *offset* – através da distorção dos ângulos esféricos. A implementação OCM utilizada pela Oculus considera uma magnitude de *offset* com o valor fixo de 0.7 que, como comprovado nesta tese, não é a óptima em termos da qualidade do *viewport vs.* bit rate resultantes. Para avaliar o impacto do *offset*, foram considerados 11 valores de *offset*, aplicados a três vídeos omnidirecionais com atividades espácio-temporais distintas. O melhor valor de *offset* encontrado foi de 0.42, permitindo ganhos na qualidade do *viewport* (medida com a métrica V-PSNR) entre 1.7 dB e 2.3 dB, relativamente à que se obtém com um valor de *offset* de 0.7.

Após a determinação do valor de offset óptimo, a projeção OCM foi avaliada e comparada com estratégias de *streaming* de vídeo omnidirecional de dois tipos: 1) monolítica ou convencional, que codifica todo o vídeo omnidirecional com a mesma qualidade; e 2) baseada em *tiles*, que utiliza *tiles* codificados com qualidade/resolução espacial diferentes para obter VAS. Para orientações de visualização do vídeo pré-definidas, o *streaming* baseado em OCM obteve resultados superiores às técnicas de referência, desde que a orientação pedida (ao servidor) corresponda à direção de visualização, ou desde que o erro entre as duas orientações não ultrapasse 20°. Ao utilizar trajetórias resultantes do movimento da cabeça de utilizadores reais, previamente gravadas, a estratégia baseada em OCM proporcionou *viewports* com qualidade superior ao método monolítico, e competiu com as estratégias de *streaming* baseadas em *tiles*.

Palavras-chave: Streaming de Vídeo Omnidirecional, Projeção Offset Cubemap, Streaming Baseado em Tiles, Avaliação Objetiva de Qualidade.

Table of Contents

Index of Figures

Index of Tables

List of Acronyms

2D	Two Dimensional
3D	Three Dimensional
6×4-Qual	Tiles 6×4 Quality
6×4-SRes	Tiles 6×4 Spatial Resolution
ABS	Adaptive Bitrate Streaming
AN-SNR	Anti-noise signal-to-noise ratio
AVC	Advanced Video Codec
BD	Bjøntegaard Delta
BL	Base Layer
BR	Bitrate
CABAC	Context-Adaptive Binary Arithmetic Coding
CDN	Content Delivery Network
CM	Cubemap
CTU	Coding Tree Unit
CU	Coding Unit
DASH	Dynamic Adaptive Streaming over HTTP
DCT	Discrete Cosine Transform
DLM	Detail Loss Metric
EL	Enhancement Layer
ERP	Equirectangular Projection
FoV	Field of View
GOP	Group of Pictures
HDist	High Distance
HEVC	High Efficiency Video Coding
HM	HEVC Test Model
HMD	Head Mounted Display
HSR	High Spatial Resolution
HQ	High Quality
HTTP	Hypertext Transfer Protocol
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector
ISO	International Organization for Standardization
IEC	International Electrotechnical Commission
JVET	Joint Video Experts Team
LQ	Low Quality
LSR	Low Spatial Resolution
MCPD	Mean Co-Located Pixel Difference
MCTS	Motion Constrained Tile Set

MonoCM	Monolithic Cubemap
MonoEqui	Monolithic Equirectangular
MPD	Media Presentation Description
MPEG	Moving Picture Experts Group
MSal	Maximum Saliency
MSE	Mean Squared Error
MSR	Medium Spatial Resolution
MS-SSIM	Multi-Scale Structural Similarity Index
OCM	Offset Cubemap
OMAF	Omnidirectional Media Format
OMAF-Qual	Tiles OMAF Quality
OMAF-SRes	Tiles OMAF Spatial Resolution
PSNR	Peak Signal-to-Noise Ratio
QEC	Quality Emphasized Center
QER	Quality Emphasis Region
QoE	Quality of Experience
QP	Quantization Parameter
RD	Rate-Distortion
RTP	Real-time Transport Protocol
SAP	Stream Access Point
SAO	Sample Adaptive Offset
SHVC	Scalable HEVC
S-PSNR	Spherical Peak Signal-to-Noise Ratio
SRD	Spatial Relationship Description
SVM	Support Vector Machine
SSIM	Structural Similarity Index
UDP	User Datagram Protocol
UHDTV	Ultra High Definition TV
VAS	Viewport Adaptive Streaming
VIF	Visual Information Fidelity
VMAF	Video Multimethod Assessment Fusion
V-PSNR	Viewport Peak Signal-to-Noise Ratio
VR	Virtual Reality
WLR	Weighted Linear Regression
WS-PSNR	Weighted to Spherically Uniform Peak Signal-to-Noise Ratio

Chapter 1. Introduction

1.1 Context and Motivation

The increasing interest for virtual reality (VR) environments has generated the development of many related technologies. One of these developments was the 360-degree or omnidirectional video, depicted in Figure 1.1a), which allows the user to navigate through the video by changing its viewing direction. Although omnidirectional videos can be displayed on conventional 2D monitors, a much better immersive experience can be achieved using head mounted displays (HMD), which allow the user to navigate on the video, while controlling the viewing direction with head movements. Several HMDs have been produced by companies such as Oculus, HTC and Sony's PlayStation. Furthermore, it is also possible to turn common smartphones into HMDs, by displaying the screen as presented in Figure 1.1b), using devices such as Google Cardboard and Samsung Gear VR, presented in Figure 1.1c) and d), respectively. Besides HMD development, omnidirectional video streaming has also grown due to the increasing availability of omnidirectional video cameras and streaming platforms, such as YouTube and Facebook, as well as by several standardization activities, e.g. the MPEG Immersive Media (MPEG-I) project [1] and its recent Omnidirectional MediA Format (OMAF) standard [2].

a) Omnidirectional video frame.

b) Smartphone screen + Google Cardboard displaying of an omnidirectional video.

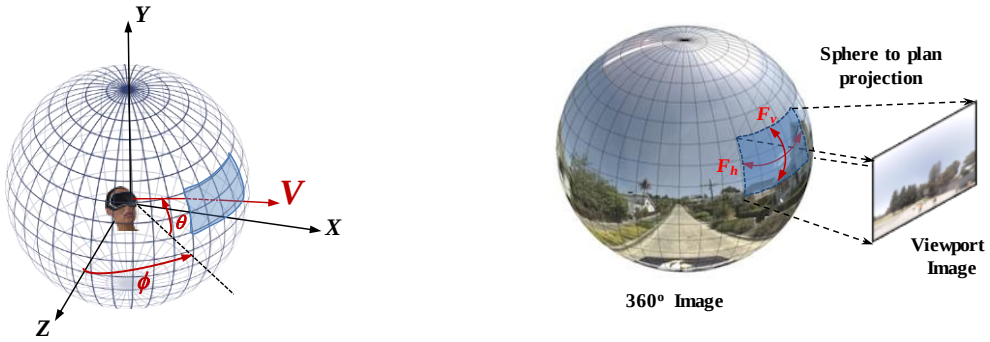
c) Google Cardboard [3].

d) Samsung Gear VR [4].

Figure 1.1 – Omnidirectional video and HMDs.

Regardless of the used display type, the user only sees a fraction – or *viewport* – of the entire sphere, at a time, whose content is defined by the viewing direction (***V***, in Figure 1.2a)), and by the horizontal and vertical field of views (F_h and F_v, in Figure 1.2b)). To provide a good quality of experience (QoE) to the user, the viewport content should have high spatial (and temporal) resolution, resulting in an even higher resolution for the entire omnidirectional video. As an example, for a viewport with an

horizontal field of view (FoV) of 120°, the full video requires at least three times the spatial resolution of the viewport resolution; a viewport with spatial resolution of 4K (3840×2160 pixels) requires a 12K (11520×5760 pixels) omnidirectional video. This results in much higher video bitrates, and required channel bandwidth, when compared to the 2D video. Accordingly, efficient omnidirectional video streaming strategies should be developed, to reduce the transmission bitrate and to avoid the interruption of the video playback due to insufficient channel bandwidth. This is especially important for video streaming over wireless networks due to their lower, and variable, available bandwidth, comparatively to wired networks.



a) Coordinate system. *b) Sphere to plan projection.*

Figure 1.2 – Viewport rendering [5].

To minimize the impact of the channel variability, Adaptive Bitrate Streaming (ABS) was adopted by the most recent streaming solutions, including omnidirectional video streaming. In ABS, the multimedia information is represented in different bitstreams, each one with a specific bitrate (and subjective quality), and each stream is divided into independent segments. This allows the receiver to switch between the available streams, during the transmission process, asking the server the most appropriate segments, according to its buffer occupancy level, the available network bandwidth, or the capabilities of the displaying device. ABS was first implemented in proprietary formats by Apple, Microsoft and Adobe. While the resulting streaming formats were conceptually quite similar, they were not totally compatible. To solve this incompatibility issue, the Moving Picture Experts Group (MPEG) has developed a new standard for the streaming of multimedia information, named as DASH (Dynamic Adaptive Streaming over HTTP) [6].

Concerning the particular case of omnidirectional video streaming, and to minimize the bitrate, only the user's viewport content should be streamed. However, current and near future internet infrastructure do not allow sufficiently low delay response to user's head directions and, as such, content outside the viewport must be also transmitted to avoid absence of content when the viewing direction has a fast change. Since, at each time instant, the user only sees a part of the whole spherical content, bandwidth savings may be achieved by transmitting that part with high quality (and higher bitrate), while the rest of the content is transmitted with lower quality, a strategy known as Viewport Adaptive Streaming (VAS). There are two main approaches for providing VAS: Quality Emphasis Region (QER) based streaming [7] [8], and tile-based streaming [9] [10] [11] [12]. QER based strategies generate several representations of the omnidirectional video, each providing better visual quality to a possible viewport region, either by encoding different regions with different qualities/spatial resolutions [7], or by projection

methods that distort the frame in such a way that more information is provided in a certain region [8]; the client then chooses the representation more appropriate to its viewing direction. Tile-based streaming segments (spatially) the omnidirectional video in tiles, and encodes each tile in multiple qualities/spatial resolutions; the client then selects higher quality/spatial resolution tiles for viewport regions and lower quality/spatial resolution tiles for the remaining regions. While several strategies for providing VAS have been proposed in the literature, there is still no clear winner solution.

Using VAS in DASH based streaming, besides conventional rate adaptation to the bandwidth fluctuations, an additional spatial adaptation to the viewing direction (or viewport) must be applied. For tile-based streaming, this is supported in DASH by a recently introduced feature, known as Spatial Relationship Description (SRD), that allows to describe spatially related parts of a same media.

1.2 Objectives

In the previous section, two approaches for VAS were referred, QER-based and tile-based; while both provide bandwidth waste reduction, the latter may lead to less efficient video encoding due to the tile independence; also, spatial discontinuities between low/high quality tiles may become visible, reducing the user QoE. The QER-based approach for omnidirectional video streaming was introduced in [7]. In that work, the cubemap (CM) projection was used, with the center of one cube's face - the face represented at highest quality - localized at the center of the most likely viewing directions; the other faces are represented with lower quality. A variant of this strategy is the offset cubemap (OCM) projection, that has been used by Facebook' s Oculus [8]. In this case, the spherical angle interval near a pre-defined direction (*offset angle*) is distorted by the projection, so that this interval is mapped on a larger cube area, and thus represented with higher spatial resolution than the other spherical angles.

The main objective of this thesis is to implement, assess and improve a VAS strategy based on the OCM projection, and compare it with tiled-based approaches, including some of the tile-based strategies proposed in OMAF.

1.3 Main Contributions

The OCM projection is characterized by the offset magnitude, which determines the area of the omnidirectional video that is projected on the cube front face; the higher the offset magnitude, the lower this area will be. This thesis evaluates the impact of the offset angle on the performance of the OCM projection and concludes about the optimum (in a rate-distortion sense) offset angle value, providing an enhanced video streaming quality comparatively to the OCM solution described in [8].

The optimized OCM projection was assessed and compared with the conventional streaming strategy – which encodes the entire omnidirectional frame with the same quality – and with diverse tile-based strategies, where the considered frame division in tiles was set according to the usual procedure in related literature: 6×4 and OMAF (Annex D.6.3) tiles structure. This comparison considered both static viewports and moving viewports resulting from real head motion trajectories.

1.4 Thesis Outline

This thesis is organized in six chapters, with the first one introducing the work in terms of context, motivation, objectives and contributions.

Chapter 2 overviews the MPEG-DASH standard, its main concepts and design principles. This includes a brief evolution of multimedia streaming towards ABS, the main motivations for the MPEG-DASH standardization, and the media distribution architecture for MPEG-DASH, from content creation to content consumption. The MPEG-DASH SRD feature is also described.

Chapter 3 describes the state-of-the-art in omnidirectional video streaming. Firstly, the main procedures involved in omnidirectional video streaming are described. Secondly, several VAS strategies are presented. Thirdly, the metrics used in omnidirectional video objective quality assessment are described. Finally, three state-of-the-art approaches for providing viewport adaptive streaming of omnidirectional video are overviewed.

Chapter 4 presents the OCM projection, its application to omnidirectional video streaming (as described in [8]) and proposes an optimization of it. As the basis of the OCM projection, a formal description of the CM projection is given. Then, a conceptual description of the OCM projection and the conditions used by Oculus to provide VAS, are presented. Lastly, an optimization of the OCM conditions used by Oculus is proposed.

Chapter 5 evaluates the OCM projection, with the optimized offset magnitude (as proposed in Chapter 4) for omnidirectional video streaming and compares it with other state-of-the-art viewport adaptive solutions. This evaluation is conducted using an objective evaluation of rendered viewports for each solution, and accounting for the resulting bitrate. In a first phase, the evaluation uses static viewports, with predefined orientations. In a final phase, the evaluation considers recorded head trajectories obtained from real users while navigating on the video, providing results closer to a realistic scenario.

Chapter 6 presents the conclusions of this thesis and some suggestions for future work.

Chapter 2. MPEG-DASH Overview

2.1 Introduction

The distribution of multimedia content on the internet has developed over the years. Streaming websites, such as YouTube and Netflix, are now available and accessible to everyone. However, bandwidth changes on the connection between server and user equipment may have a negative impact on the provided streaming service; in particular, an insufficient bandwidth can lead to an empty buffer on the user side, stopping the media playback, and compromising the user's Quality of Experience (QoE). Wireless networks are especially affected, as the radio channel is more unstable, when compared to wired connections. To minimize the impact of the channel variability, Adaptive Bitrate Streaming (ABS) was adopted by the most recent streaming solutions. In ABS, the multimedia information is represented in different bitstreams, each one with a specific bitrate (and subjective quality), and each stream is divided into independent segments. The characteristics of each segment – such as timing, server location or associated bitrate (among others) – are defined in a description file, that is sent before the transmission starts. This strategy allows the client to switch between the available streams, during the transmission process, asking the server the most appropriate segments, according to its buffer occupancy level, the available network bandwidth, or the capabilities of the displaying device. ABS was first implemented in proprietary formats by Apple, Microsoft and Adobe. While the resulting streaming formats were conceptually quite similar, they were not totally compatible. To solve this incompatibility issue, the Moving Picture Experts Group (MPEG) has developed a new standard for the streaming of multimedia information, called DASH (Dynamic Adaptive Streaming over HTTP) [13].

This chapter presents the main concepts and design principles subjacent to the MPEG-DASH standard. In section 2.2, a brief evolution of multimedia streaming towards ABS, and the main motivations for the MPEG-DASH standardization, are given. In section 2.3 and 2.4, the MPEG-DASH media distribution architecture is described, from content creation to content consumption. Finally, section 2.5 presents a new feature, recently added to the MPEG-DASH, known as Spatial Relationship Description (SRD), and highlights its relevance in the context of omnidirectional video streaming.

2.2 Adaptive HTTP streaming

Media streaming is the continuous deliver of multimedia data, from a service provider to an end-user, while the data is simultaneously presented on the user terminal. Streaming is an alternative to file downloading, a process in which the end-user obtains the entire content before watching or listening to it. In the last years, several streaming solutions have been proposed; Figure 2.1 describes, schematically, their evolution.

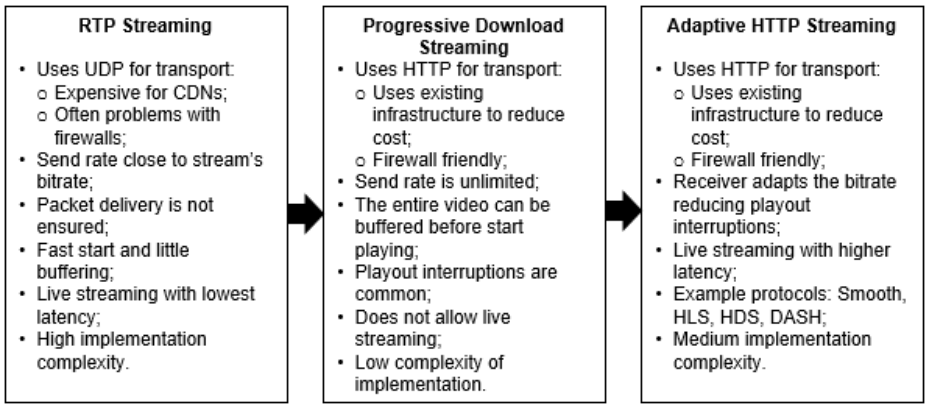


Figure 2.1 - Evolution of streaming toward adaptive HTTP streaming.

In the earliest streaming solution, referred to as "RTP Streaming", the Real-time Transport Protocol (RTP) was used; RTP is an application layer protocol for delivering audio and video over IP networks. The RTP streaming solution prioritizes the timely delivery of information and, to achieve this goal, it often tolerates some packet loss; as such, it is built on the User Datagram Protocol (UDP). UDP is a transport layer protocol that uses a simple connectionless communication model. It does not guarantee delivery, ordering, or duplicate protection of data. An RTP streaming server has just one representation of the media and sends it at a rate close to the streamed media bitrate, which is shown by the client's media player as the data arrives. This implies that, to avoid information loss during the streaming session, the connection bandwidth between the server and a client must be equal to, or higher than, the bitrate of the streamed media; if the connection bandwidth drops below the media bitrate, data will be lost, which will result on a temporal discontinuity on the media playback. Another issue with RTP streaming is that UDP often causes firewall problems. Moreover, nowadays it is common the use of content delivery networks (CDN) for storing and distributing content, and many do not support RTP [14].

In "Progressive Download Streaming", the Hypertext Transfer Protocol (HTTP) is the protocol used for media transport. Since it is the main application layer protocol on Internet, it is allowed by most firewalls and also supported by CDNs, making it a good option for the application layer protocols in streaming services. In progressive HTTP download, the server has just one representation of the media content, which the client obtains using HTTP requests. While the content file is downloaded, the client media player is able to start reproducing the data, as the client does not need to download the whole file before playback parts of it. The data is received at the available bandwidth between server and client. With only one content representation accessible to the client, a connection bandwidth lower than the media bitrate might lead to stalls on the media playback, reducing the user's QoE. Furthermore, Progressive Download Streaming does not allow live streaming.

To solve the problems of progressive download, while taking advantage on the use of HTTP, adaptive HTTP streaming solutions were developed. In this case, the main idea is to adapt the media bitrate to the available network bandwidth and to the client buffer fullness, seeking to continuously playing the media without interruptions. To achieve this, the media is encoded with different bitrates (and qualities), each one corresponding to a media representation, and each representation is divided in temporal segments. The information about the available media, as its bitrate, quality and location (i.e., its URL) is available in a manifest file, which is sent to the client at the beginning of the streaming

session. The media can also be separated by types (e.g., video, audio, subtitles, etc.), so that the client may choose between different camera angles, or between different languages, to just mention a few possibilities.

Figure 2.2 shows a streaming session workflow, and where the client changes the requested media quality over time, reacting to the changes on the available network bandwidth.

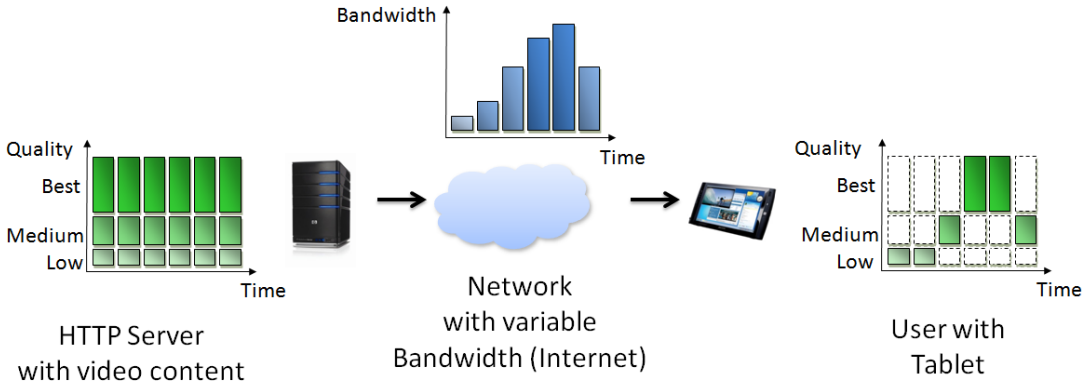


Figure 2.2 – Workflow of adaptive HTTP streaming [15]

Several proprietary adaptive HTTP streaming solutions were developed, including Apple's HTTP Live Streaming, Microsoft's Smooth Streaming and Adobe's HTTP Dynamic Streaming. All these solutions are closed systems, with their own manifest and media content formats, although also having several characteristics in common. To allow interoperability and an open-source standard for media streaming, MPEG developed the first adaptive HTTP based streaming standard, the MPEG-DASH [13]. This standard resulted from a collaboration with other international organizations (such as 3GPP) and companies (including Apple, Microsoft and Adobe), and as inherited many characteristics from their proprietary solutions.

2.3 Media Distribution Architecture of MPEG-DASH

Figure 2.3 shows the media distribution architecture of MPEG-DASH. The content creation encodes and segments the media, and creates the corresponding Media Presentation Description (MPD) file, in a way compliant with the MPEG-DASH standard; the resulting files are then stored in an HTTP server. Additionally, part of these files may be stored on HTTP caches, located close to the client sites, in order to reduce bandwidth requirements, server load and response times.

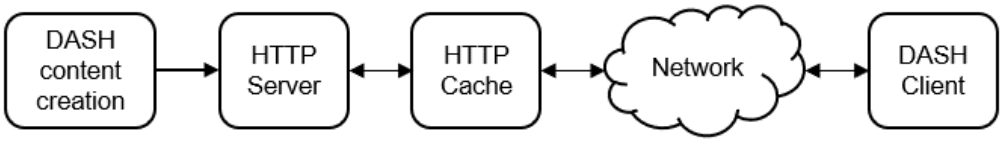


Figure 2.3 – Media distribution architecture of MPEG-DASH.

The MPEG-DASH does not define the adaptive bitrate streaming logic; instead, it specifies [16]:

- A normative definition of a Media Presentation, with Media Presentation defined as a structured collection of data that is accessible to the DASH client, through the **Media Presentation Description**.
- A normative definition of the formats of a **segment**, with a segment defined as an integral data unit of a media presentation that can be uniquely referenced by a HTTP-URL.

The following sections details the main blocks of the MPEG-DASH architecture.

2.3.1 Content creation

The media to be streamed needs to be prepared to comply with the DASH format. In the following, this preparation process is explained only for video data since this is the media type that will be used on this work.

First, the original video is spatially and/or temporally subsampled, creating different video sources with different resolutions and/or frame rates, respectively. Then, each video source is encoded with several bitrates, resulting in the so-called video representations. Although MPEG-DASH allows the use of any video encoder, the most commonly used video encoders are, nowadays, H.264/AVC [17] and H.265/HEVC [18]. Each video representation is then segmented in time intervals of fixed length (typically, between 1 second and 15 seconds), and different lengths may be applied to the same video representation. Each segment contains at least one stream access point (SAP) [14]; this is a position where playback of a media stream may start, while using only information contained from that position onwards [13]. An initialization segment might be created, which is used to initialize the video for playback and that contains information about the media SAPs. If not created, every segment shall be self-initialized.

Concerning the temporal segmentation, and as video encoders explore time redundancy, it reduces the overall compression efficiency - the shorter the time intervals are, the larger the overall media data size is; also, more requests are necessary from the client, which can lead to server overloading. However, shorter segments allow a faster adaptation to bandwidth changes, which is preferable in unstable connections, like wireless networks.

Finally, the Media Presentation Description is created, which contains the relevant information about the available video representations, such as spatial resolution, frame rate, bitrate, segment time intervals and server localization (URL).

Figure 2.4 illustrates a simplified version of the content creation process for a video track. It is assumed that the spatial subsampling results in three different resolutions, the temporal subsampling is kept constant and, for each spatial resolution version, three different bitrates are produced by the encoder.

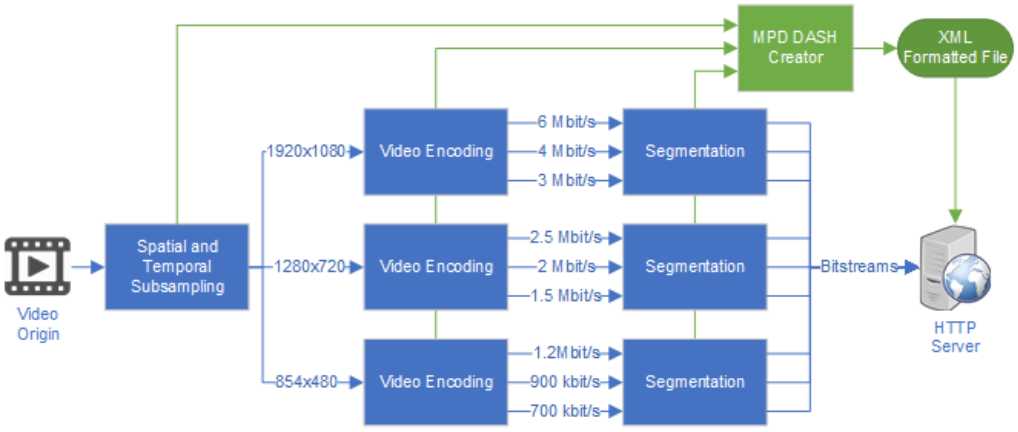


Figure 2.4 – Content creation process.

2.3.2 DASH server and client

Once created, the different video representations are stored on ordinary HTTP servers, and their URLs are linked to the segments by the MPD information. This allows the use of HTTP GET requests from the client, to download each media segment. Figure 2.5 shows the basic architecture of the HTTP server and a DASH client.

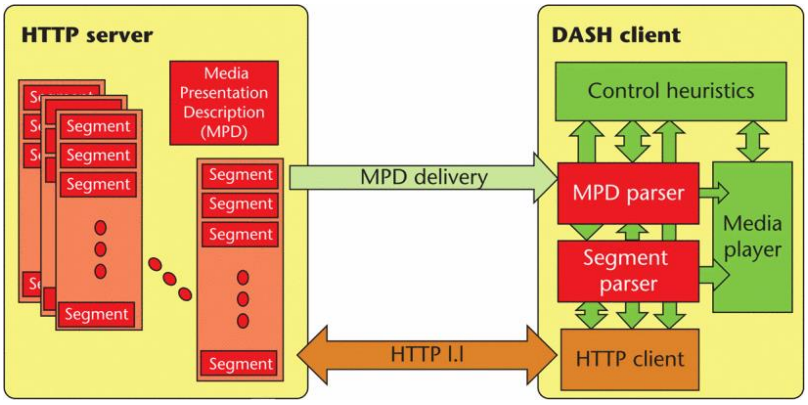


Figure 2.5 – Basic architecture of the server-client system, in DASH [14].

The MPD is transmitted to the client at the start of the transmission. The client then parses the MPD and starts requesting the selected video segments. For a continuous playback, without pauses (or video stalls), the requested video representation may change from segment to segment, according to the measured connection bandwidth. This strategy results in a DASH session with a flow like the one presented in Figure 2.6. As already mentioned, the DASH scope does not cover the client's bitrate adaptation heuristics, nor the media players.

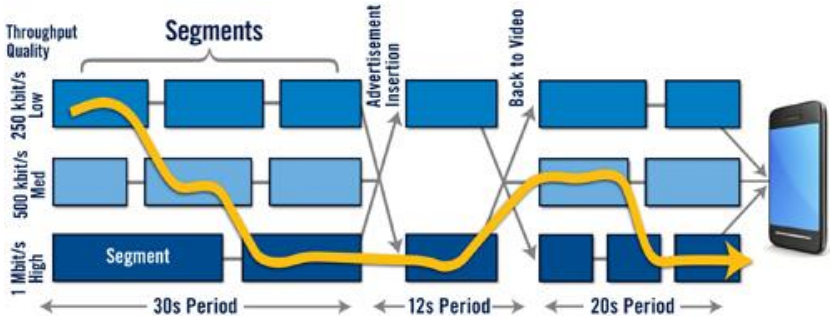


Figure 2.6 – Flow of a DASH session [19].

2.4 DASH Media Presentation Description

When streaming with DASH, the Media Presentation Description (MPD) informs the client about the available media content. This includes information about the media segments, and their relationships, and about other metadata that may be needed by the clients. The MPD is an XML document that respects a hierarchy established by the DASH standard, as depicted in Figure 2.7. From the highest to the lowest, the different hierarchy levels comprise periods, adaptation sets, representations and segments, and are all explained below.

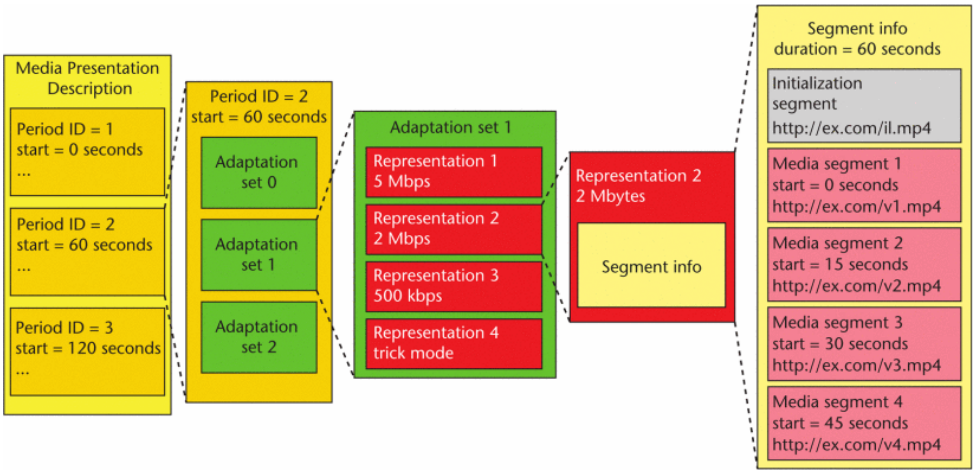


Figure 2.7 – MPD hierarchical data model [16].

I. Period

An MPD file contains one or more periods. Typically, a period represents a media content period during which a consistent set of encoded versions of the media content is available, i.e., the set of available bitrates, languages, captions, subtitles, etc., does not change during a period. Each period has an associated start time. Multiple periods may be created to enable splicing of the content, for example for ad insertion. They can also provide a synchronization point to avoid drift in segment numbering [13] [20].

II. Adaptation set

Adaptation sets are used to separate different parts of the media content in a period. During a streaming session, the client chooses only the adaptation sets that are of interest to him. A simple example consists on a separation of video and audio in different adaptation sets. This allows the client

to request only the audio component, or only the video component, or both. Separation between different audio tracks is done with adaptation sets. Similarly, it is at this level that different spatial areas of the video are selected when using the Spatial Relationship Description (SRD) feature [21] (described in section 2.5). Adaptation sets can also contain subtitles or arbitrary metadata [13].

III. Representation

Each adaptation set is divided in different representations. Every representation of one adaptation set is assumed to represent the same content but with different properties. A representation is defined by its encoded conditions, such as bitrate, resolution and used encoder. Changing between representations allows the client to adapt to the channel bandwidth variations, and device conditions [13].

IV. Segment

Segments represent a split in time of the media contained in one representation, and are the final result of the content creation step, described in section 2.3.1 . In order to access a segment, a URL is provided for each one. Typically, all segments within a representation have the same or roughly similar duration. To allow seamless switching during media playback across different representations, the segment duration might be made equal to all representations of an adaptation set. However, this is not normative as different representations of an adaptation set can have different segment durations [13].

2.5 MPEG-DASH SRD

DASH has the potential to evolve through the addition of new features. One problem that surged after its development was how to represent different spatial parts of a video, so that the representation of each part could be also adapted to the channel and client conditions. Omnidirectional video is one instance where this functionality is quite useful. Since typical 360º displays only show a fraction (or *viewport*) of the whole viewing sphere at a time, streaming the whole 360º video at constant quality would be a waste of bandwidth. As such, different regions of the 360º video can be sent at different qualities, so that the region of interest is sent with better quality. This streaming method is known as viewport adaptive streaming (VAS). MPEG-DASH SRD serves as a way to describe spatially related parts of a same media [21].

2.5.1 Tiled streaming

Tiled streaming allows a client to stream different spatial parts of the complete video with different bitrates, selected by the client. This allows a better bandwidth allocation, since a region-of-interest (ROI) on the video might be prioritized and streamed with the highest bitrate, and the other spatial parts can either be retrieved at a lower quality or discarded.

Preparing the video for tiled streaming requires an extra step to the content creation process described in subsection 2.3.1 – a spatial segmentation of the video has to be applied before the encoding process; it consists on spatially splitting each video frame into several parts, named tiles. The tiles that, along time, belong to the same single area of the video are taken together, encoded and stored as a new independent video stream, or spatial segment. The result is a large number of video files, each

representing a specific area of the original video [22]. As the encoder explores spatial redundancy to maximize compression, spatial segmentation leads to a decrease on the compression efficiency, and to an increase in overall media data size.

Figure 2.8 presents a tiled streaming example of a conventional 2D video, where tiles are rectangular shaped and the client only retrieves the tiles that contain the ROI.

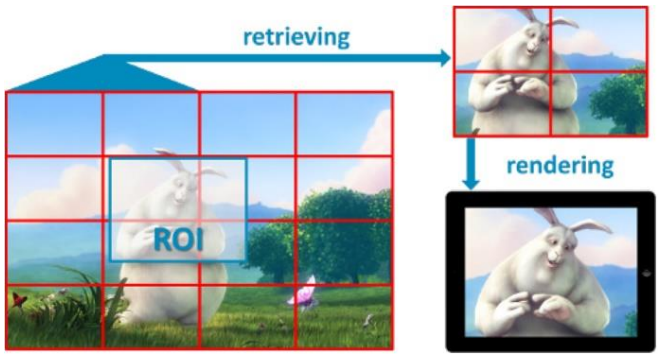


Figure 2.8 – Tiled streaming example [23]

2.5.2 Feature specifications

The SRD feature describes how media assets (video, audio, etc.) relate to each other, i.e., how the content is spatially organized. It does not presume anything about how a player shall use this information. For instance, considering a tiled video described by SRD, a player may decide to display a group of tiles at the same time, while another player only displays one tile at a time. Additionally, the information in SRD describes how the media assets are spatially related from a content creator perspective. Even though it can be used with other types of media, the SRD feature is only explained for video data since this is the media type that will be used on this work.

To spatially position the tile, a 2D Cartesian coordinate system, with the origin of the coordinate system at the top-left corner of the video, is used. This coordinate system represents an arbitrary coordinate system in which the positioning of the video tile is provided. With this, it is possible to describe from simple grid cell indices, like nonoverlapping tiled videos, to more complex representations, like overlapping ones. Processing of spatially related videos by a client is not defined and, like DASH, SRD is agnostic to the used encoder.

To express the SRD functionality on the MPD file, a property is added to adaptation sets, as different adaptation sets contain different spatial parts of the media. A number is used as an identifier of a reference space which implicitly defines a coordinate system. In the coordinate system, x and y represent the position of the top-left corner of the associated media assets. The width and height of each component and the total width and height of the media are also given, in coordinate system units. Finally, certain use cases require the selection of tiles within the same resolution layer, e.g. when panning, or from another resolution layer, e.g. when zooming in or out. This presence of different resolution layers is indicated by an optional identifier [21].

Figure 2.9 presents an example of application of MPEG-DASH SRD. The user is able to navigate and zoom within the video, allowing full control over what the user want to watch. This is achievable

using tiled streaming, allowing streaming of tiles with 4K resolution to be played on mobile devices like smartphones and tablets.

Figure 2.9 –MPEG-DASH SRD use case [21].

Chapter 3. Omnidirectional Video Adaptive Streaming

3.1 Introduction

The rise of streaming media allowed new types of content to develop, such as omnidirectional video streaming. Omnidirectional video provides an immersive experience not achieved by regular 2D video, by covering the 360º viewing angle with three degrees of freedom. The release of consumer available electronics for omnidirectional video, such as cameras and head mounted displays, made the consumption of this type of media easier. Furthermore, MPEG developed new standards for omnidirectional video as the Omnidirectional Media Format (OMAF) [24]. OMAF specifies the means to allow the consumption of omnidirectional video. Due to requiring more data than traditional 2D video, streaming of omnidirectional video currently faces limitations mainly due to required bandwidth.

This chapter presents the state-of-the-art in omnidirectional video streaming. In Section 3.2, the processes involved in omnidirectional video streaming are described, with emphasis on mapping, region-wise packing and coding. In Section 3.3, the main challenges of omnidirectional video streaming are presented. In Section 3.4, the state-of-the-art in omnidirectional video streaming is described with different possible approaches on this subject. In Section 3.5, the metrics used for omnidirectional video objective quality assessment are described. Finally, Sections 3.6, 3.7 and 3.8 provide a summary of the strategies developed by Corbillon et al. [7], Graf et al. [25] and Nasrabadi et al. [26].

3.2 The Omnidirectional Video Transmission Chain

This section describes the end-to-end processes required for the creation, transmission and consumption of omnidirectional video content. The architecture of the omnidirectional video processing chain is first presented; after, mapping, region-wise packing and video coding are briefly explained.

3.2.1 Omnidirectional Video Transmission Architecture and Walkthrough

Figure 3.1 presents the key steps involved on in a typical omnidirectional video transmission chain. This involves several processing steps to create, transmit and visualize omnidirectional frames, from stitching to rendering.

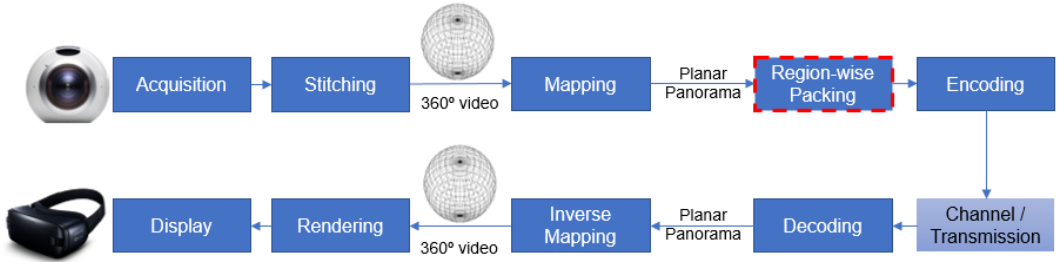


Figure 3.1 – Omnidirectional video transmission architecture [27] [28]

The main functions of each module are [27] [28]:

- **Acquisition:** The acquisition of omnidirectional videos is typically done with multiple cameras, that are time synchronized and uniformly placed along a rig. Each camera's lens is oriented in different directions, so that the whole visual scene around the camera rig is covered, thus obtaining a full 360º field of view, captured by different cameras. All cameras sense the environment to produce synchronized 2D images or video frames and the field of view of a single camera overlaps with other cameras field of view to allow the next processing step.
- **Stitching:** Stitching consists on the process of combination of the 2D frames captured during the acquisition with overlapped regions (between frames) to create a single omnidirectional frame representation. This is a complex process that involves several steps such as registration, calibration, blending and illumination compensation to assemble the omnidirectional frame. The output of this step is a non-planar spherical representation of the visual scene.
- **Mapping:** The omnidirectional spherical frame must be projected onto a 2D plane in order to be coded and transmitted. While there are many projection schemes, two popular projections, equirectangular projection (ERP) and cubemap projection (CMP) are usually employed. The output of the mapping process is a projected 2D omnidirectional frame. This process is explained with more detail in Subsection 3.2.2 .
- **Region-wise packing**: This optional region-wise packing step enables manipulations (resize, reposition, rotation, and mirroring) of any region in the projected omnidirectional frame and allows to distinguish regions in the mapped frame with different quality levels (after coding). This process may require the additional transmission of metadata. This process is explained with more detail in Subsection 3.2.3 .
- **Encoding:** After having a planar representation, a 2D video codec is used, such as H.264/AVC or HEVC. If the region-packed frame is separated in tiles (or slices), each tile (or slice) might be independently encoded from each other, using HEVC or H.264/AVC. Another option is to use a scalable HEVC (SHVC) encoder to obtain a set of hierarchical layers from low to high quality. This process is explained with more detail in Subsection 3.2.4 .
- **Channel/Transmission:** The bit stream generated by the encoding step can be transmitted to the client over a fixed or wireless communication channel. Nowadays, video streaming protocols such as MPEG-DASH could be used to transmit video content as described in Chapter 2.
- **Decoding:** In the decoding step of this processing chain the inverse operation of the encoder is performed and at the end the decoded projected omnidirectional video is obtained.
- **Inverse Mapping:** The planar omnidirectional video is mapped into a spherical representation, by applying the corresponding inverse mapping transformation.
- **Rendering:** In omnidirectional video, the frames that are shown to the user correspond only to a part of the entire viewing sphere. Thus, it is necessary to process the spherical representation (or render) to obtain a 2D plane of a fraction of the sphere (also known as viewport) which can then be shown to the users. Users can look around freely and only a small field of view (much less than 360º) is shown according to the user viewing direction.
- **Display:** The final step is to display the obtained viewport to users. The displays for omnidirectional video are of two types: the first corresponds to a navigable frame on a standard

2D display (e.g. a computer or a smartphone screen), where the user controls the viewing direction with a mouse or by moving the display; the second type corresponds to a head mounted display (HMD), which is a display that a user wears on his head, and tracks user's head movements to compute the corresponding viewport.

When streaming omnidirectional video, only a viewport is displayed at any particular time to the user. This feature may be utilized to improve the performance of omnidirectional video systems, through selective delivery depending on the user's viewport, known as viewport-dependent streaming. For generating a viewport-dependent video from an omnidirectional video, the most relevant steps of the omnidirectional video transmission chain are mapping, region-wise packing and encoding.

3.2.2 Mapping

The spherical representation of omnidirectional video cannot be straightforward encoded. As such, a mapping process is required to convert from the spherical representation to a planar representation at the encoder. After decoding, this planar representation is converted into a spherical one through inverse mapping. However, mapping a sphere to a flat plane always introduces some deformation and changes in the area to be encoded. The two planar projections supported by OMAF are described below:

- **Equirectangular projection:** The ERP is the most often used planar projection. The projection maps meridians to vertical straight lines, equally spaced, and parallels to horizontal straight lines, also equally spaced. To keep the rectangular shape, the points are stretched more and more as they approach the poles, increasing also the deformation of the content, as shown in Figure 3.2 by the distortion of the circles on the sphere to ellipses on the plane. This implies that the stretched content uses more pixels for its projection, which increases the number of pixels to encode. ERP is a relatively simple transformation and it is a friendly format to preview the whole omnidirectional video. However, the distortion present in the projection affect the coding performance, due to the larger number of redundant pixels representing small areas near the poles.

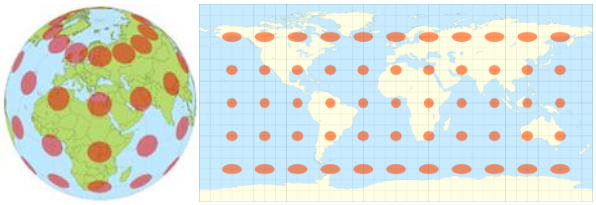


Figure 3.2 – Equirectangular projection [29] [30]

- **Cubemap projection:** The CMP is a planar projection of a cube. As seen on the left side of Figure 3.3, the sphere is surrounded by cube faces and then is stretched to cover the cube faces completely. The content close to the cube edges and corners has to be stretched but not as much as for the ERP near the poles. The cube faces are then organized to obtain a rectangular layout. One possible layout is presented on the right side of Figure 3.3, which gives continuity to the frame in each row. The final flat plane contains less distortion than the ERP, but the coding performance may be affected due to discontinuities between the faces.

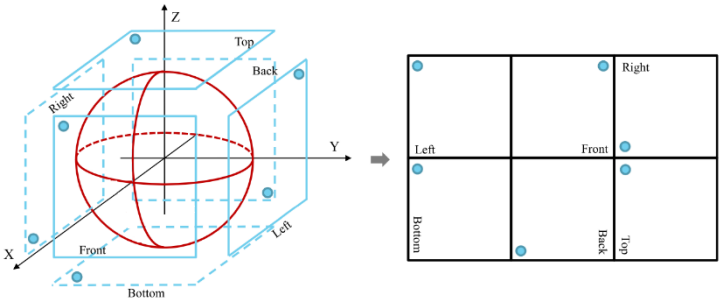


Figure 3.3 – Cubemap projection [28]

3.2.3 Region-wise packing

Additional changes can be made to the planar representation through region-wise packing. When region-wise packing is used, 2D projected regions (e.g. after equirectangular mapping) are mapped onto a packed frame by indicating the location, shape, and size of each projected frame region in the packed frame (metadata that is also transmitted). Each projected frame region usually has a rectangular shape, since this is the only shape currently supported by OMAF [24]. It is not mandatory that the packed frame covers the entire sphere. Figure 3.4 illustrates one simple example of generating a packed frame through region-wise resampling and repositioning, changing an equirectangular frame to a packed frame where the “top” and “bottom” regions are resampled to half the width and positioned above the middle part of the equirectangular frame, which remains with the same resolution [28].

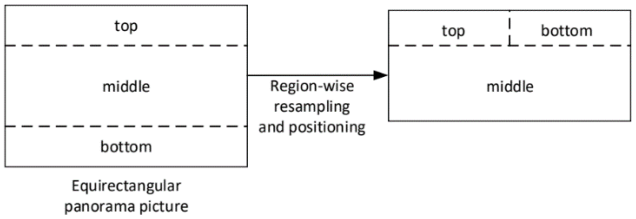


Figure 3.4 – Region-wise packing applied to a ERP frame [28]

A known problem of region-wise packing consists on the appearance of artifacts along the boundaries of the regions after rendering. To avoid or reduce these artifacts, a guard band might be used. The guard band allows addition of some extra pixels at the boundaries of the regions when the region-packed frame is created. The resampling of different regions (in the example a lower spatial region) allows a region of interest to keep high quality, while reducing overall resolution of the frame to be encoded and thus obtaining coding efficiency improvements.

One possible application of region-wise packing is viewport dependent streaming that will be described in Section 3.4.

3.2.4 Coding

The bitrate required to transmit raw video data is rather large and thus requires enormous resources for storage and transmission. Thus, video compression aims to lower the bitrate requirements by exploiting different characteristics of the video, e.g. spatial or temporal redundancy but also perceptual irrelevance. In the past several coding standards for video were developed and are nowadays widely used for several applications. Figure 3.5 shows a brief timeline of the main video

codecs developed by the ITU-T and ISO/IEC committees. Video codecs have been improving their coding performance (e.g. bitrate for the same quality), mainly due to advances in terms of hardware which allow the development of more complex compressing algorithms. Nowadays, the most commonly used video codecs are H.264/AVC (MPEG-4 part 10) and H.265/HEVC (or just HEVC).

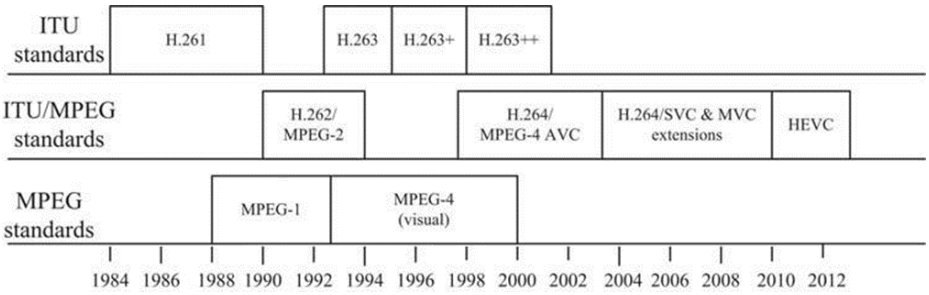


Figure 3.5 – History line of video coding standards by the ITU-T and ISO/IEC committee [31]

HEVC was standardized in 2013 and is the current state-of-the-art video codec in terms of rate-distortion (RD) performance. HEVC is the first codec to support UHDTV, supporting resolutions up to 8192×4320. As the previous ITU and MPEG standards, it typically uses the 4:2:0 color sampling, meaning that the chrominance components (responsible for colors) are half, in both directions compared to the luminance components. HEVC achieves better compression efficiency for the same perceptual quality than its predecessors, reducing 50% the bitrate when comparing with the previous standard AVC. HEVC also defines tiles, which are self-contained and independently decodable rectangular regions of the frame. The HEVC encoding process [32] starts by partitioning each frame into multiple units called Coding tree units (CTUs), usually 64x64. Then, each CTU is split recursively in a quadtree structure, resulting in coding units (CUs), which can have a size from 8x8 to the size of the CTU. The next step of HEVC encoding consists on exploiting redundancies spatially, through Intra-frame prediction, or exploiting redundancies temporally, with inter-frame prediction. For intra-frame prediction, HEVC has 33 directional modes (prediction is obtained by extrapolation from neighboring pixels), a DC mode (mean of left and upper neighboring pixels) and a planar mode (approximation of a smooth gradient between near left and upper neighboring pixels). For inter-frame prediction, an 8th order finite impulse filter is used to obtain motion vectors with ¼ precision. Within the inter-frame prediction loop, a deblocking filter is used, followed by a sample adaptive offset (SAO) filter. Both filters are used to have decoded frames with higher quality that are also used as reference during inter-frame prediction. The prediction type of each CU is either intra-frame or inter-frame and this is selected with a mode decision process that usually considers both rate and distortion. The difference between the original frame and the prediction, referred as residual, is transformed and quantized. HEVC defines two transformations to be used, the Discrete Cosine Transform (DCT) and one derived from the Discrete Sine Transform. The decoded quality of the video is conditioned by the quantization parameter (QP) which regulates how much spatial detail is saved. By increasing the QP the bitrate is decreased, but at the price of an increase in distortion. The final coded bitstream is obtained by exploiting the statistical correlation between symbols (DCT coefficients, motion vectors, etc.) through context-adaptive binary arithmetic coding (CABAC).

After its initial version in 2013, the HEVC standard has been extended to be better adapted to some applications. Scalable HEVC (SHVC) is an extension that offers a way of coding video in multiple

layers, where each layer represents a different quality representation of the same video scene. The base layer (BL) is the lowest quality representation and is backwards compatible with conventional HEVC, while enhancement layers (ELs) provide improved video quality (at the cost of more bitrate) and may be coded using lower quality layers frames as reference. To decode an EL with SHVC, its reference layers must first be fully decoded to make them available as prediction references; interlayer prediction tools are used to improve the prediction quality at some layer. The main types of video scalability are temporal and spatial. Temporal scalability allows an EL to increase the frame rate of the video. Spatial scalability allows an EL to increase the spatial resolution of the video.

3.3 Challenges in Omnidirectional Video Streaming

Nowadays, several platforms (such as YouTube) and network providers are working on systems to stream omnidirectional video with a good quality of experience. Most often, to show high video quality it is necessary to acquire, code and transmit the full high-quality omnidirectional frames. These high-quality omnidirectional frames achieve high resolutions, up to 8K, and high frame rates, e.g. 120fps.Thus, the resulting bitrate required to fully transmit omnidirectional videos with high quality between 50 to 100 Mbit/s, which demands transmission channels with large capacities difficult to achieve nowadays, especially on wireless networks, especially when a large amount of users want to consume this type of content.

During playback of omnidirectional video, only the user's viewport is displayed. To reduce required bandwidth to streaming, a solution that only streams the visual data corresponding to the current viewport is streamed might be considered. This solution is similar to adaptive HTTP streaming, with the spatial dimension added to the adaptation space. The server stores versions of the omnidirectional video, each containing a different area of the omnidirectional frame covering a possible viewport. These versions are encoded in different bitrates and can be temporally segmented to allow some adaptation to the network bandwidth conditions. The clients choose which areas to download and their fidelity (e.g. resolution) based on the user's viewing direction and available bandwidth. However, the latency of the network limits the speed of spatial adaptation. For example, when a user changes its viewport quickly to an area not covered (entirely or partially) by the current video data, no content is available in this new area (covered by the new viewport but not the old viewport) and thus it is necessary to wait for the arrival of the new area data. This situation is only completely prevented for network latency lower than 10 milliseconds [24], which is currently not feasible for many transmission channels. As such, it is necessary to prefetch future segments (spatio-temporal regions of the video), i.e. anticipate which areas of the videos the users will need in the future. However, prefetching future segments requires knowledge about the user's future viewport and thus, attempts have been made to predict the user's future viewport based on the viewport history. Yet, the mismatch between user's predicted and actual viewport is likely to occur [33]; moreover, the accuracy of viewport prediction decreases with time.

As usually happens in the case of temporal segmentation (forcing Intra frames), spatial segmentation also lowers the compression efficiency of a codec (not allowing spatial predictions). Furthermore, it is possible that different video versions covering possible viewports have overlapped areas, and thus being somewhat redundant between them. Moreover, spatial adaptation requires more

frequent temporal segmentation, to allow quicker adaptation to user's viewport changes. All these issues increase overall data size needed to be stored on a server and the number of files to be managed by a server and to be included on a manifest file.

3.4 State-of-the-art in Omnidirectional Video Streaming

Recently, several solutions for omnidirectional video streaming have been suggested in the literature, mainly to deal with the high amount of video data that needs to be stored, transmitted and rendered compared to traditional 2D video streaming. With the OMAF specification, interoperability among several VR related services, systems and devices are now possible. In addition, OMAF also enables quality control for regions of the omnidirectional video according to their importance. From all solutions available in the literature (some of them proposed in the context of the OMAF specification), the following classes were identified by abstracting the differences between solutions and understanding their commonalities:

- **Conventional approach:** This solution is also known as viewport-independent streaming and monolithic streaming. The encoding of omnidirectional video is done like traditional 2D video, meaning that a single encoded bit stream contains the full omnidirectional video, where quality is similarly distributed on the entire frame. In this approach, the client decodes the bit stream continuously to obtain the omnidirectional video data (complete frames) and thus, any viewport can be rendered. However, the bandwidth required for transmission of the entire omnidirectional video is rather large and a large part of each omnidirectional frame of the video will not be covered by any viewport, thus wasting a significant amount of bandwidth [24].
- **Personalized viewport approach:** In this solution, the client requests the viewport that it should display to the user. The server receives the client request, performs a suitable spherical to plane projection using the omnidirectional video frames and encodes the result, i.e. the requested viewport. With this approach, only the necessary bitrate to transmit the requested viewport to the user is spent. However, since it is only sent the viewport, the absence of content after a quick change of head direction (which requires a new viewport) might occur, especially if network latency issues are considered. Additionally, the extra encoding (or re-encoding) step of a personalized viewport further increases latency and requires a server capable of encoding (highly complex process) a different video for each client and thus, doesn't scale well when the number of users increase.
- **QER-based approach:** In this solution, the full omnidirectional video frames are also encoded (as in the conventional approach), but a specific spatial region, called Quality Emphasis Region (QER), is encoded in higher quality and the rest with lower quality. This is achievable using different methods: by choosing different qualities for different areas of the omnidirectional video during encoding [7]; by using projections which represent areas of the omnidirectional video with higher quality, such as the Offset cubemap projection [8]. QER-based approach requires multiple bitstreams to be encoded, each one with a different QER, which are stored on a server to be transmitted as requested by clients. The client requests the video representation that includes

the user's viewport in the QER. The choice of QERs available at the server may take different approaches. For example, the computation of QERs can be made to assure that every region of the omnidirectional video has a high quality representation. However, increasing the number of different QERs available at the server significantly increases the required storage. As such, available QERs are chosen based on typical viewing directions [7] [34].

- **MCTS-based approach:** One or multiple spatially subsampled versions of the omnidirectional video are divided in motion constrained tiles. These HEVC tiles are coded independently, which means that intra prediction cannot use as reference, regions that do not belong to the tile as in simple tiles. However, in motion constrained tiles it is not allowed to use motion vectors that reference past frames and "point" outside the spatial region of the tile. The motion constrained tile set (MCTS) technique limits the temporal inter prediction of the tile at the same position of the current frame and the reference past frames. Motion constrained tiles allow to encode different spatial-temporal regions of the video independently without a significant decrease in terms of coding efficiency. Each tile is encoded in multiple bitstreams at different bitrates, using different QPs and/or different resolutions, which are stored on a server. The idea is to stream one tile or a combination of tiles covering the viewport in high quality and thus reducing the required bandwidth transmission. In a streaming session, the client requests tiles covering its viewport in better quality and the rest of the omnidirectional video in lower quality tiles, to compensate for any latency on the transmission. However, the division of omnidirectional video in tiles reduces the compression efficiency, as every tile is independently decodable and spatial redundancy decreases [24] [25].
- **SHVC with MCTS-based enhancement layer:** The omnidirectional video is encoded with a scalable codec such as SHVC. This codec structures the compressed video into layers which allow the transmission of data that is combined with previous already received data to enhance the quality of tiles. The BL corresponds to the omnidirectional video with the lowest quality and can be encoded with AVC or HEVC. The EL layers can be divided into MCTS tiles, which are encoded using inter prediction from the same layer (with restricted motion vectors) and inter-layer prediction from the co-located region in an already decoded reference layer. The BL is always streamed and ELs are used to cover the viewport requested by the client. Since the BL is always streamed, it does not require temporal segmentation as frequently as an EL, and thus allows to better exploit the video temporal redundancy [26].
- **Simulcast with MCTS-based HEVC high-resolution representation:** Similar to the previous approach but a scalable codec is not used. In this case, a low quality representation of the entire omnidirectional video is streamed; this video is not used as reference and thus is transmitted simultaneously (simulcast) with a better quality representation of the omnidirectional video. Then, the omnidirectional video is encoded using the MCTS technique. The resulting tiles can be used to enhance some spatial regions of the omnidirectional video, namely the parts which contain the user viewport. However, this approach does not support inter-layer prediction, which leads to an increase in bandwidth and server storage required for streaming. Moreover, it is required

two HEVC decoders, one for the base layer and another for the high quality motion constrained tiles [10].

The last Sections of this Chapter present more details about some of these solutions. More specifically, Section 3.6 presents a QER-based approach, Section 3.7 presents a MCTS-based solution, and Section 3.8 presents a SHVC with MCTS-based enhancement layer solution. For the offset cubemap projection, a QER-based approach, more details and some optimization of the method are presented in Chapter 4.

3.5 Objective Quality Assessment Metrics

During the entire omnidirectional video transmission chain, the video suffers several transformations (such as mapping projection, lossy coding, etc.) which may introduce artifacts that can impact the perceived quality. Furthermore, viewport dependent streaming may introduce discontinuities on the quality of the omnidirectional video both spatially and temporally, which might be seen by users after head movements that are not immediately followed by adaptation algorithms (unpredicted head movements). This section provides a description of some objective quality metrics used for omnidirectional video. Most metrics used for omnidirectional video are variations of metrics used for 2D video. As such, it is first described the most commonly used metrics for 2D video:

- **Mean Squared Error (MSE):** MSE measures the mean of the squared differences between pixels of a reference and of the corresponding impaired frames. This is described by (3.1), where (i, j) refers to a pixel position, X refers to the impaired frame, Y refers to the reference frame, and H and W refer to the frame height and width in pixels, respectively. To obtain a single value for the video sequence, the resulting MSE values per frame are averaged along all the video frames.

$$MSE = \frac{1}{WH} \sum_{i=1,j=1}^{W,H} \left(X_{ij} - Y_{ij}\right)^2 \tag{3.1}$$

- **Peak Signal-to-Noise Ratio (PSNR):** PSNR is defined by (3.2), where P is the peak value of the signal (255 for bit depth of 8 bits).

$$PSNR = 10 \log_{10} \frac{P^2}{MSE} \tag{3.2}$$

As seen in previous sections, conventional 2D video and omnidirectional video are different in the way they are created and consumed. As such, metrics for objective quality assessment applied for spherical surfaces were created. The following metrics are adaptations of the PSNR metric to omnidirectional video:

- **Viewport PSNR (V-PSNR)** [35]: V-PSNR applies traditional PSNR evaluation only to the pixels of a given viewport, i.e., between corresponding viewports of the reference and impaired videos. This metric requires explicit head motion data (which may be difficult to acquire in some cases) and thus is dependent on how the user(s) interact with the content.

- **Spherical PSNR (S-PSNR)** [35]: S-PSNR was created to summarize the average quality over all possible viewports. S-PSNR is calculated based on the squared value differences of points uniformly sampled on two conceptual unit spheres, one generated by a reference frame and one by an impaired frame. One of the S-PSNR disadvantages results from the fact that a sample on the sphere might not correspond to an integer position on the planar projections, requiring the use of some interpolation procedure, which may condition the metric result. It is also not suitable for situations where quality is not equally distributed over the entire sphere.
- **Weighted to Spherically Uniform PSNR (WS-PSNR)** [36]: WS-PSNR uses two 2D frames of the same resolution and of the same projection type. WS-PSNR weights the pixel error computed between pixels on the reference and impaired frames, by the corresponding pixel area on the spherical surface. As for S-PSNR, it is not suitable for situations where quality is not equally distributed over the entire sphere.

When using a PSNR related metric on a content encoded in multiple bitrates, it is obtained a rate-distortion (RD) curve characterized by a set of N bitrate values $(R_1, \dots, R_N)$ with corresponding PSNR measurements $(D_1, \dots, D_N)$. In [37], G. Bjøntegaard has proposed a method to measure the coding efficiency between two different algorithms which may produce different pairs of rate and distortion (neither rate or distortion values are aligned). This method is used to calculate the average PSNR and average bitrate differences (in %) between two RD curves obtained from PSNR and bitrate measurements. Considering $r = \log R$, two values can be obtained with this model:

- **Bjøntegaard Delta PSNR (BD-PSNR)** [37]: BD-PSNR corresponds to the average PSNR difference in decibel (dB) for the same bitrate. The RD curves are represented by the distortion as a function of the (the logarithm of the) bitrate and approximated with the following third order polynomial fitting:

$$\widehat{D}(r) = ar^3 + br^2 + cr + d \tag{3.3}$$

where $\widehat{D}$ is the fitted distortion in PSNR and a, b, c, and d are the parameters of $\widehat{D}$. The average PSNR difference ΔD between two RD curves is the BD-PSNR and is given by (3.4), where the integration bounds, r_L and r_H, are given by (3.5) and (3.6) [38].

$$\Delta D = \mathrm{E}[D_2 - D_1] \approx \frac{1}{r_H - r_L} \int_{r_L}^{r_H} \left[\widehat{D}_2(r) - \widehat{D}_1(r)\right] dr \tag{3.4}$$

$$r_L = \max\left\{\min\left(r_{1,1}, \dots, r_{1,N_1}\right), \min\left(r_{2,1}, \dots, r_{2,N_2}\right)\right\} \tag{3.5}$$

$$r_H = \min\left\{\max\left(r_{1,1}, \dots, r_{1,N_1}\right), \max\left(r_{2,1}, \dots, r_{2,N_2}\right)\right\} \tag{3.6}$$

- **Bjøntegaard Delta Bitrate (BD-BR)** [37]: BD-BR corresponds to the average bitrate difference in percentage for the same PSNR. The RD curves are represented by (the logarithm of the) bitrate as a function of the distortion and approximated with the following third order polynomial fitting:

$$\hat{r}(D) = aD^3 + bD^2 + cD + d \tag{3.7}$$

where $\hat{r}$ is the fitted bitrate function and a, b, c, and d are the $\hat{r}$ parameters. The average bitrate difference ΔR between two RD curves is the BD-BR and is given by (3.8), where the integration bounds, D_L and D_H, are given by (3.9) and (3.10) [38].

$$\Delta R = \mathrm{E}\left[\frac{R_2 - R_1}{R_1}\right] \approx 10^{\frac{1}{D_H - D_L}\int_{D_L}^{D_H}[\hat{r}_2(D) - \hat{r}_1(D)]dD} - 1 \tag{3.8}$$

$$D_L = \max\left\{\min\left(D_{1,1}, \dots, D_{1,N_1}\right), \min\left(D_{2,1}, \dots, D_{2,N_2}\right)\right\} \tag{3.9}$$

$$D_H = \min\left\{\max\left(D_{1,1}, \dots, D_{1,N_1}\right), \max\left(D_{2,1}, \dots, D_{2,N_2}\right)\right\} \tag{3.10}$$

PSNR is the simplest and most widely used quality metric. However, several studies have shown that is not very well matched to the perceptual video quality and other metrics with better performance have been proposed in the past. In omnidirectional video, a common practice is to adapt existing 2D video metrics that correlate better with subjective evaluation to evaluate only the viewport based on recorded users head movements, as is the case of V-PSNR. In this scenario, the most commonly used metrics are:

- **Structural Similarity Index (SSIM)** [39]: SSIM is a perceptual metric that quantifies image quality degradation caused by many processing steps, such as lossy image compression or transmission errors, and is highly adapted to extract structural information from images. SSIM is calculated based on comparisons of luminance $l(X,Y)$, contrast $c(X,Y)$, and structure $s(X,Y)$ between two image signals X and Y, computed by

$$l(X,Y) = \frac{2\mu_X\mu_Y + C_1}{{\mu_X}^2 + {\mu_Y}^2 + C_1} \tag{3.11}$$

$$c(X,Y) = \frac{2\sigma_X\sigma_Y + C_2}{{\sigma_X}^2 + {\sigma_Y}^2 + C_2} \tag{3.12}$$

$$s(X,Y) = \frac{\sigma_{XY} + C_3}{\sigma_X\sigma_Y + C_3} \tag{3.13}$$

  where μ_X and μ_Y refer to the mean, σ_X and σ_Y to the standard deviation and σ_{XY} to the cross correlation, and C_1, C_2, and C_3 are model parameters. Considering that α, β, and γ are parameters used to adjust the relative importance of the three components, the SSIM metric is given by

$$SSIM(X,Y) = [l(X,Y)]^{\alpha} \cdot [c(X,Y)]^{\beta} \cdot [s(X,Y)]^{\gamma}. \tag{3.14}$$

- **Multi-Scale Structural Similarity Index (MS-SSIM)** [40]: MS-SSIM is an extension of the SSIM, which has more flexibility than SSIM by including variations of image resolution and viewing condition. MS-SSIM considers different spatial resolutions by iterative downsampling and weighting the different par of each SSIM component (luminance, contrast and structure) at different scales. The source image and the test image are low-pass filtered and then down sampled by a factor of 2 successively. The source and the test images are denoted as scale 1 and the highest scale is denoted as k, which is obtained after $k-1$ interactions. The luminance

is compared at the highest scale only, while contrast and structure are compared for each scale z. The MS-SSIM metric is given by

$$MS - SSIM(X,Y) = [l_k(X,Y)]^{\alpha_k} \cdot \prod_{z=1}^{k} [c_z(X,Y)]^{\beta_z} \cdot [s_z(X,Y)]^{\gamma_z}. \quad (3.15)$$

where α_k, β_z, and γ_z are parameters to define the relative importance of the different components.

- **Video Multimethod Assessment Fusion (VMAF)** [41] [42]: VMAF was developed by Netflix and uses a machine learning model to predict the subjective video quality; this model is trained and tested using opinion scores obtained through subjective experiments. Also, it is a metric focused on quality degradation due to compression and rescaling, the most common artifacts in video streaming. VMAF uses four image quality metrics – Visual Information Fidelity (VIF), Detail Loss Metric (DLM), Mean Co-Located Pixel Difference (MCPD), and Anti-noise signal-to-noise ratio (AN-SNR) – and fuses them using a Support Vector Machine (SVM). The result is a single score in the range of 0 to 100 per video frame (100 corresponds to identical quality), which then can be averaged to obtain a single value for the entire video.

3.6 Viewport-Adaptive Navigable 360-Degree Video Delivery

This section presents of the omnidirectional video streaming solution proposed by Corbillon et al. [7]. First, its context and objectives are summarized, followed by a description of the solution. Lastly, tests conditions and benchmarks are provided.

3.6.1 Context and objectives

Nowadays, omnidirectional video streaming is very challenging especially in the Internet and in wireless networks due to their characteristics in terms of delay, available bandwidth, varying channel conditions, etc. To provide good Quality of Experience (QoE) and avoid motion sickness, it is recommended that video streaming systems react to head movements as fast as the HMD refresh rate. Since the HMD refresh rates can be as high as 120Hz (and thus 10ms of delay) it is necessary efficient mechanisms to provide the necessary fast adaptation to changes in the viewport (seen by the user) location. Thus, it is typically transmitted the full omnidirectional video (and not only the viewport) for the HMD to extract the viewport and thus avoid any QoE reduction. However, there is a significant bandwidth waste since the user only sees a part of the entire omnidirectional video frames (1/3 of the omnidirectional video for 120º FoV). This is critical since omnidirectional videos require very large spatial and temporal resolution (up to 8K with 120fps) and thus bitrate (between 50 to 100 Mbit/s).

To reduce the bandwidth consumption, this solution proposed by [7] follows the same principles as in rate-adaptive streaming technologies like DASH but the server offers multiple representations of the same omnidirectional video that not only differ by bitrate (and thus quality or spatial resolution) but differ by having better quality in a given spatial region of the video. The bandwidth necessary for streaming is reduced due to the lower quality (and thus bitrate) required for representation of regions outside this region, which should correspond as much as possible to the viewport seen by the user.

3.6.2 Technical Solution

The solution can be characterized by the creation of different representations of the omnidirectional video, each one with an associated Quality Emphasis Region (QER). The center of the QER is the Quality Emphasized Center (QEC), which is a position of the spherical video where the quality is maximum. Outside of the QER, the quality of the video decreases. Depending on the user head motion, the client selects the best representation to stream, this means the one that has the best viewport quality; however, the entire video is available at the client so any viewport can be generated. Figure 3.6 shows the QER-based omnidirectional video streaming system proposed.

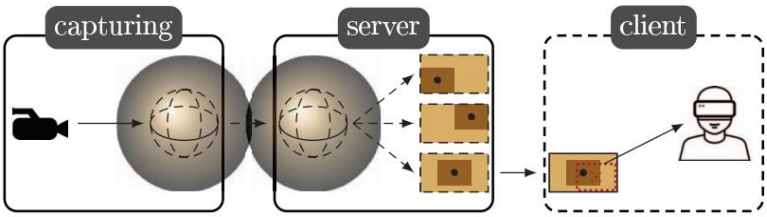


Figure 3.6 – QER-based streaming system: The server offers video representations for three QERs. The dark brown is the part of the video encoded at high quality, the light brown the low quality. The viewport is the dotted red rectangle, the viewport center the cross [7]

The most important modules of the proposed QER-based streaming solution are detailed below:

- **Server:** The server receives as input the omnidirectional video in the equirectangular format which is split into multiple segments from 1s to 10s. Then, each segment is encoded into multiple representations, each one characterized by its QEC and bitrate. The output is n QER-based videos, encoded at k global quality levels. The resulting number of available representations of the omnidirectional video segment is $n \times k$. The manifest file contains a description about all segment representations and the QEC associated to each one.
- **Client:** During the streaming session, the user head orientation changes as well as the available bandwidth. The client periodically sends a request to the server for a new segment representation that matches both the viewport center and the available bandwidth (throughput). This is accomplished by an adaptation algorithm that selects the next segment representation. First, it selects the QER of the video based on an estimation of the future position of the user head; in this case, the algorithm implemented for QER selection consists on the selection of the representation that has the QEC at the smallest orthodromic distance (smallest distance between two points on the surface of a sphere) to the viewport center at the time the client runs the adaptation algorithm. This is a simple algorithm, but better head motion prediction algorithms could be employed, thus making better QEC selections. After selecting the QER, the client chooses a representation using throughput estimation to reduce the mismatch between the requested bitrate and the available bandwidth.

The solution takes advantage of the geometric structure of the mapping projection (before coding) to optimize the video encoding based on QER. Each geometric projection is characterized by a number of faces (e.g. 6 for the cube map) and a given central point (which corresponds to a position on the sphere). For each QER version of the omnidirectional video, the front face of the geometric projection

corresponds to the QER and its central point is aligned with the QEC. This is achieved through rotation of the omnidirectional video assuring that the QEC is always at the same position on the 2D layout. The other faces of the geometric projection are encoded with reduced quality. The position of QECs is determined using the Thomson positioning problem [43], meaning that the QECs selected are spread on the sphere. However, the positioning of QECs could be improved based on saliency maps (probability of viewing directions), extracted from the feedback of previous viewers.

3.6.3 Performance Assessment

For the performance assessment, a software was developed that includes several features:

- Projection from a spherical video into four geometric projection layouts (equirectangular, cube map, pyramid, and dodecahedron). The software rotates the video so that the QEC is always at the same position on the 2D layout.
- Adjustments of the video quality for each geometric face of any layout.
- Viewport extraction for any viewport center on the sphere. Measure of QoE was done by measuring the quality of several extracted viewports instead of the full spherical video.

For the geometric projection layout evaluation described next, a 4K omnidirectional video using ERP was taken from YouTube and used as a reference. As for the number of QERs evaluation, a dataset of recorded head movements of real users watching omnidirectional videos was used. The number of videos included were eleven and each one was ten seconds long. The head movements were taken from eleven people who were asked to watch the videos on a current state-of-the-art HMD while standing, to enable a high degree of freedom.

I. Geometric Projection Layout Evaluation

The first experiment measures the viewport video quality for several geometric projections and with several face quality arrangements (quality of regions outside the QER); this will allow to measure the efficiency of each projection. The extracted viewports had a 1080p resolution and the original equirectangular video has a 4K resolution. The solutions compared with the reference video were the following:

- **UniEqui:** The equirectangular video re-encoded with HEVC at a constant bitrate of 6 Mbit/s, which corresponds to 75% of the original video bitrate. This solution is referred as *uniEqui* since all regions have a similar quality.
- **Equirec, CubeMap, Pyramid, Dodeca:** Projection of the spherical video into an equirectangular projection with 8x8 tiles, cube map, pyramid, and dodecahedron representations encoded with QER at 6 Mbit/s. For each projection, only the “best” quality arrangement for the overall bitrate budget is presented. For the CMP projection, the QEC is located at the center of the cube front face at full quality while the other faces have 25% of the full quality target.

The performance of the geometric projection can be studied with two aspects: the best viewport quality when the viewport center and the QEC perfectly matches; and the degradation of the viewport quality when the user head direction (viewport center) moves away from the QEC. To examine both aspects, one QEC was selected on the spherical video. Then, the orthodromic distance between the

center of the viewport and the QEC was varied from 0 to π. The MS-SSIM objective quality metric was used to compare the viewport obtained from each representation with the viewport obtained from the original reference video (at the same location). Figure 3.7 shows the results obtained for several geometric projections. The cubemap provided the best results, since the MS-SSIM obtained was 0.98 when the viewport center matches the QEC and stays higher than any other projection. The cubemap projection has a better result for the MS-SSIM metric than the *uniEqui* solution up to 2 units of distance.

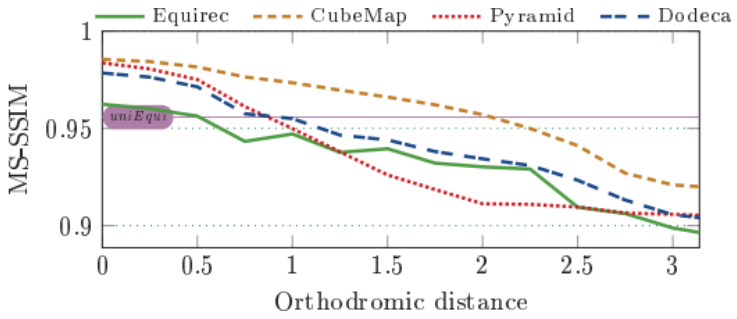


Figure 3.7 – Average MS-SSIM depending on the distance to the QEC for the four geometric layouts. Global bitrate budget: 6 Mbit/s [7]

II. Number of QERs Evaluation

The number of QER representations that are needed to perform efficient omnidirectional streaming represent a key trade-off in this solution. The more QERs are computed, the better the coverage of the spherical video which allows more fine adaptation, but the storage required at the server increases (as well as the MPD file). In this experiment, the average was computed for all users from our dataset assuming a different number of equally spread QECs on the spherical representation. The adaptation algorithm referred in Subsubsection 3.6.2 uses the head positions in the dataset to determine the QER for the next segment. The solutions considered were the following:

- **CMP+QER:** Omnidirectional video encoding with QER and cube-map projection (CMP) at 6 Mbit/s. For each QER version, the video is rotated in order to assure that the CMP has the QEC located at the center of the front face set at full quality. The other cube faces are encoded at 25% of the full quality target.
- **UniEqui**: The original equirectangular video re-encoded at 6 Mbit/s, which corresponds to 75% of the original video bitrate.

Different durations for the time segments were considered. Figure 3.8 presents the median PSNR gap between the viewports of the CMP+QER and *uniEqui* solutions, depending on the number of QERs. The best number of QERs is between 5 and 7, since the gains obtained for higher number of QERs are not significant enough to justify the extra-cost in terms of storage. With multiple QERs and short segments, the median PSNR gap is higher due to the better re-synchronization between the QERs and the viewport centers.

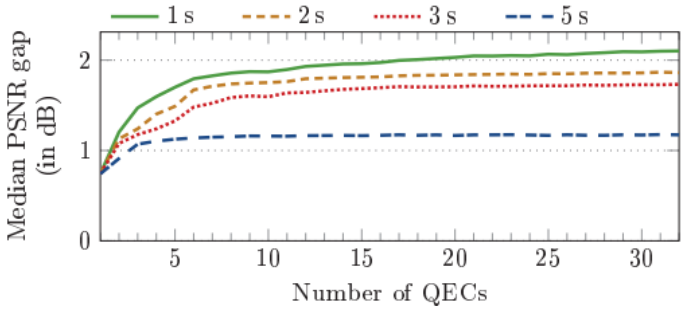


Figure 3.8 – Median PSNR gap between the viewports of the cube map layout and the uniEqui depending on the number of QERs. Bitrate: 6Mbit/s [7]

3.7 Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP

This section presents the omnidirectional video streaming solution proposed by Graf et al. [25]. First, its context and objectives are summarized, followed by a description of the solution. Last, tests conditions and benchmarks are provided.

3.7.1 Context and objectives

The streaming of omnidirectional video content can be performed by using the same strategy as 2D video, which means streaming the entire omnidirectional video with equally distributed quality (spatially) disregarding the user's viewport area or region (the only part seen by the user). However, this may lead to a significant waste of bandwidth or quality of experience. Region of interest-based coding is one of the most promising candidate to solve this problem and is supported by the tiling mechanism existing in recent video standards such as HEVC. Tiles may divide a video spatially and temporally into regular-sized, rectangular regions which are independently decodable and allow efficient parallel processing.

To reduce bandwidth consumption without significantly increasing required storage, the solution proposed by [25] builds on rate-adaptive streaming technologies like DASH and adds spatial adaptation provided by the tiling mechanism. Spatially segmenting the omnidirectional video into tiles and storing each one in different bitrates allows the client to download each tile (and thus region) at different qualities according to a policy that maximizes the user quality of experience. More precisely, this strategy allows the visualization of any viewport with high quality by a client while saving bandwidth comparing to a solution where the entire omnidirectional video is streamed. In such case, the regions outside the viewport can be streamed with lower quality or not even transmitted at all.

3.7.2 Technical Solution

The proposed solution for adaptive omnidirectional video streaming aims to provide bandwidth efficient omnidirectional video streaming using a tiling mechanism. Figure 3.9 depicts the proposed system architecture. The projection format used is ERP, since it is the format most widely supported and deployed in many applications. The omnidirectional video is encoded with HEVC motion constrained tiles (MCTS as described in Section 3.4). Then, each tile is re-encoded in multiple quality representations and time segmented. The resulting segmented tiles are made available at the server to

enable spatial and bandwidth adaptation to the content during streaming. The MPEG-DASH SRD (see Section 2.5) is used to describe the tile structure to client devices. Then, the clients request some tiles with an appropriate quality depending on the client conditions, such as bandwidth and current viewport. The adaptation logic that a client runs to choose representations based on the client conditions is not optimized, as it adopts only a very basic buffer management and throughput measurements.

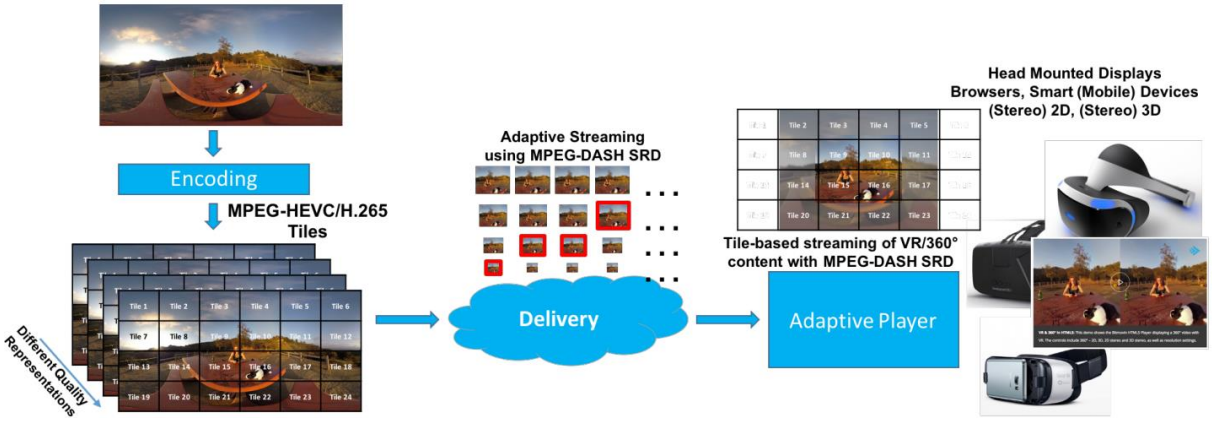


Figure 3.9 – System architecture for bandwidth efficient tiled streaming. Adapted from [25]

The three streaming strategies proposed are described below:

- **Full Delivery Basic**: All tiles of the omnidirectional video are provided to the client resulting in a full omnidirectional frame. The tiles visible to the user's *current* viewport are requested in the highest possible quality representation. The remaining of the tiles are requested in the lowest available quality representation. Depending on the available bandwidth and the bitrate of the tiles outside of the current viewport, the bitrate of the highest possible quality representation for the tiles necessary to fully represent the current viewport changes. In the best case, it is the same as the highest available quality representation and, in the worst case, it is the lowest available quality representation.
- **Full Delivery Advanced**: This strategy is an improvement of the full delivery basic approach and takes into consideration the quality of areas close to the viewport that are visible when the user's viewport (head) moves. As such, tiles around the visible viewport are requested in a lower (but not the lowest) quality. This can be improved with prediction of the user's viewport motion, followed by the request of tiles that do not overlap with the current viewport but are highly likely to be visible. Full delivery advanced is not evaluated, only suggested as a strategy to be studied in the future.
- **Partial Delivery**: Only the tiles covering the current viewport are delivered and thus the available bandwidth is consumed only in the transmission of these tiles. The tiles of the current viewport are requested at the highest possible quality representation. It is important to note that user head motions could lead to the rendering of blank areas or the corresponding tiles need to be rendered with delay, which decreases the QoE. Thus, this strategy is considered as impractical but is used as a benchmark to show what can be achieved by the other strategies in comparison.

3.7.3 Performance Assessment

The performance assessment was made using two omnidirectional videos downloaded from YouTube: *ExploreTheWorld* and *AssassinsCreed*. While the first is a computer-generated video the other is a documentary. However, the results shown in this section were only obtained for the documentary video. The spatial resolution for the entire omnidirectional video were 1920x960, 3840x1920 and 7680x3840.

I. Bandwidth Requirements

The goal of this experiment is to compare the traditional streaming solution to the proposed tiled streaming based on bandwidth constraints. As such, the required bitrate that achieves the same objective quality in the viewport area was measured for the different strategies. Then, the resulting measurements were compared to a traditional streaming solution in percentage of bitrate savings. To evaluate the bandwidth required, it was considered a static viewport, at a pitch and yaw angle of 0º. The viewport had a horizontal FoV of 96º. One solution and three different strategies for tile streaming were compared:

- **Traditional streaming solution (aka monolithic)**: The entire equirectangular video (all regions) HEVC encoded with a QP of 27 and segment duration of 4 seconds.
- **Tiles Monolithic strategy**: Delivery of the entire omnidirectional video in tiles with the highest quality. The segment duration was 1 second and the tiles were encoded with QP of 27.
- **Full Delivery Basic strategy**: Delivery of the viewport tiles at highest quality and other tiles at lowest quality as described in subsection 3.7.2. The segment duration was 1 second, the high-quality tiles were encoded with QP of 27, and the low-quality tiles were encoded with QP of 42.
- **Partial Delivery strategy**: Delivery of only the viewport tiles at highest quality as described in Subsection 3.7.2. The segment duration was 1 second and the viewport tiles were encoded with QP of 27.

Different tile patterns were considered for the tiled strategies (column x rows): 3x2, 5x3, 6x4, and 8x5. Larger tiles provide improved coding efficiency but less flexibility for viewport selection, while smaller tiles can better cover a given viewport but have lower coding efficiency (temporal and spatial correlation is not fully exploited).

Table 3.1 shows the results for bandwidth saving in comparison with the monolithic solution, where ‘positive’ values indicate an overhead and ‘negative’ values show actual savings. The results show better bandwidth savings for partial delivery, but this strategy is impractical due to the delay in usual network environments. Thus, partial delivery results are mainly to show the potential of tiled streaming. The full delivery basic strategy shows better results for the tilling pattern 6x4, resulting in a bandwidth reduction of over 64% comparing with the monolithic solution. The tile monolithic strategy allows to evaluate the overhead caused by spatial segmentation and more frequent temporal segmentation than the simple monolithic strategy.

Table 3.1 – Bitrate savings in percent relative to monolithic video for different resolutions and tiling patterns. Values in bold represent the highest and lowest bitrate savings for full delivery basic. [25]

		Monolithic [kbps]	Tiles, Bitrate Saving [%]		
Resolution	Tiling	Monolithic 4s	Tiles Monolithic	**Full Delivery Basic**	Partial Delivery
1920x960	3x2	3,537.32	19.13	**-19.14**	-23.12
1920x960	5x3	3,537.32	23.51	-42.81	-50.73
1920x960	6x4	3,537.32	25.93	**-64.92**	-77.36
1920x960	8x5	3,537.32	30.28	-45.47	-57.55
3840x1920	3x2	9,857.76	14.28	**-22.04**	-26.72
3840x1920	5x3	9,857.76	16.25	-45.91	-54.36
3840x1920	6x4	9,857.76	17.92	**-66.37**	-78.74
3840x1920	8x5	9,857.76	19.92	-49.25	-60.21
7680x3840	3x2	22,390.45	9.89	**-23.16**	-28.99
7680x3840	5x3	22,390.45	10.90	-46.20	-56.36
7680x3840	6x4	22,390.45	11.80	**-64.69**	-78.82
7680x3840	8x5	22,390.45	12.77	-50.36	-62.48

II. Viewport PSNR

To obtain a full evaluation of the proposed strategy and closer to a realistic streaming environment, the V-PSNR quality metric (explained in Section 3.5) using the recorded head motion of three users was used. The head motions were recorded while watching the omnidirectional video with resolution 1920x960. Additionally, one of the three users also had its head motion recorded while watching the omnidirectional video with resolution 3840x1920. The head motion traces were used to generate viewports simulating the streaming strategies described in the previous Subsection. One solution and two different strategies (for the proposed solution) for tile streaming were considered:

- **Monolithic solution (reference)**: Equirectangular video encoded with HEVC with QP = {22, 27, 32, 37} and segment duration of 4 seconds.
- **Tiles monolithic strategy**: Delivery of the entire omnidirectional video in tiles with the highest quality. The segment duration was 1s and the tiles were encoded with QP = {22, 27, 32, 37}.
- **Full delivery basic strategy**: Delivery of the viewport tiles at highest quality possible (considering some bandwidth constrain) and the tiles outside the viewport at the lowest quality. The segment duration was 1 second, tiles within the viewport were encoded with QP = {22, 27, 32, 37, 42}, and tiles outside the viewport were encoded with QP of 42.

Partial delivery (as described in the previous section) was not considered for this test, since using V-PSNR with partial delivery is impractical due to calculations evolving not available tiles. Similar tile patterns that were used for bandwidth requirements experiment were also considered here for a spatial resolution of 1920x960. In addition, results were provided for the tile pattern 6x4 which was considered the best in the previous experiment for a spatial resolution of 3840x1920. After the measure of the V-PSNR for each strategy, the Bjøntegaard-Delta (BD) Bitrate (BR) was used for comparison of monolithic solution (reference curve) with the two tiled strategies. Table 3.2 shows the obtained results. The results show that the full delivery basic tiling scheme provides the best results for a 6x4 tiling pattern, achieving a bitrate saving up to 40%.

Table 3.2 – BD-BR of tiled content over monolithic content with a segment duration of 4 seconds using V-PSNR. [25]

			BD-BR [%]	
Head Movements	Resolution	Tiling	Tiles Monolithic	Tiles With Full Delivery Basic
User 1	1920x960	3x2	30.538	-9.008
User 1	1920x960	5x3	34.732	-35.427
User 1	1920x960	6x4	38.680	**-35.433**
User 1	1920x960	8x5	45.682	-35.360
User 1	3840x1920	6x4	25.874	**-38.982**
User 2	1920x960	3x2	30.779	-15.075
User 2	1920x960	5x3	34.513	-28.976
User 2	1920x960	6x4	38.501	**-40.896**
User 2	1920x960	8x5	45.748	-29.970
User 3	1920x960	3x2	31.042	-11.317
User 3	1920x960	5x3	34.926	-31.786
User 3	1920x960	6x4	38.884	**-38.389**
User 3	1920x960	8x5	46.439	-32.282

3.8 Adaptive 360-Degree Video Streaming using Scalable Video Coding

This section presents the omnidirectional video streaming solution proposed by Nasrabadi et al. [26]. First, its context and objectives are summarized, followed by a description of the solution. Lastly, tests conditions and benchmarks are provided.

3.8.1 Context and objectives

The issues caused by network latency in omnidirectional video streaming can be mitigated with several techniques, such as the prediction of the client's viewport. Nowadays, viewport location can be accurately predicted for one second and if longer predictions are employed, a mismatch between actual and predicted viewport occurs and may lead to rebuffering, reducing the quality of experience offered to the user. Other option is for the client prefetch more future time segments and replace them with new ones every time the predicted viewport changes, which results in bandwidth waste. Thus, there is a tradeoff between QoE and bandwidth waste which forces the client to not prefetch chunks for more than the high accuracy viewport prediction interval, currently around 1 to 2 seconds. However, with this short buffer, abrupt variations in network conditions could be enough for a client to consume all the data in the buffer, leading to rebuffering.

To improve user's QoE during omnidirectional video streaming, it is necessary to minimize the amount of time spent on rebuffering, especially under challenging network conditions. To address this issue, this solution proposes an adaptation method for omnidirectional video streaming using scalable video coding, specifically SHVC. Since in SHVC, the BL is always required for decoding the ELs, the BL can be prefetched and buffered at the client for a long duration, significantly reducing rebuffering. ELs are then prefetched to increase the quality of the regions within the user's predicted viewport.

3.8.2 Technical Solution

The proposed solution for adaptive omnidirectional video streaming aims to provide the best possible QoE for the user viewport while minimizing the amount of bandwidth used. The CMP projection

is used due to its lower bandwidth requirements to convert the spherical representation to a planar representation before encoding and transmission; however, any other projection can be used. Each cube face is divided into a grid of tiles of equal size. Tiles are encoded with motion constrained in different quality levels according to the layered scheme of SHVC. As in conventional streaming, temporal segmentation is equally applied to all tiles of all layers, and during the streaming session the client makes all the decisions. Figure 3.10 shows the client components of the scalable tiled solution.

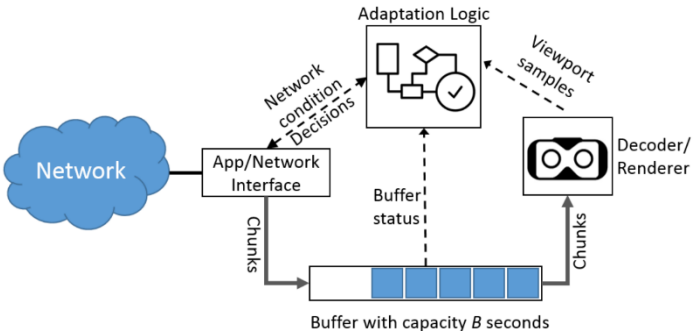


Figure 3.10 – Client components for scalable tiled streaming [26].

The adaptation logic shown in Figure 3.10 is responsible for the adaptation of video quality to network conditions and to the user's viewport. Adaptation to network is done based on buffer occupancy (buffer-based) and throughput estimation (rate-based). The buffer-based adaptation uses the amount of data that is currently stored in the buffer of the video player and the throughput estimation is obtained by smoothing throughput measures from previous chunk downloads. The adaptation logic uses buffer-based adaptation for base layer buffering, since the proposed method can buffer the BL for longer durations. Whenever the BL buffer is full, throughput estimation is used for viewport quality enhancement. The decision of the tiles requested for viewport quality enhancement is done with viewport prediction using previous viewport samples. The most important modules of the proposed scalable tiled streaming solution are detailed below:

- **Base layer buffering:** The BL is the bottom layer, enabling the client to continue video playback at the lowest quality during bandwidth drops, avoiding rebuffering. Base layer buffering occurs when the buffer is not full, this means that all BL tiles are downloaded as soon as possible, i.e. the client prefetches (downloads even if not immediately required) all base layer tiles for the chunks sequentially. This allows to have low-quality BL chunks for longer duration compared to the viewport prediction high-accuracy interval, which allows the client to continue playback during bandwidth drops or mismatches in the viewport prediction.
- **Viewport prediction:** The client requests more data to enhance the BL quality, this means ELs to be played in the future. Thus, to request EL tiles collocated in the viewport location some type of prediction about the future viewport location is necessary. The selected prediction method consists on the application of a Weighted Linear Regression (WLR) on the last 10 viewport location samples, as proposed in [33]. The average prediction accuracy within 10 degrees of deviation from the actual viewport is 96.6% for 1 second and 71.2% for 2 seconds. The prediction interval defined for the solution was 1 second, i.e. at time t, the user's viewport at time t + 1 is predicted.

- **Viewport quality enhancement:** When the client's BL buffer is full, the client tries to improve the quality of the user's viewport. Requesting a higher quality representation that requires higher than available throughput causes delay on delivery, and consequently shrink of the BL buffer. Then, during BL buffering, the quality is low until the buffer is full and finally quality improvement of user's viewport is possible, switching to higher quality. Repeating a request of a higher quality representation not delivered in time by the network would repeat the previous process. To avoid this change in quality, throughput estimation is used for viewport quality enhancement. Throughput estimation is done through sampling of the available throughput, where the total size in bits of the tiles downloaded in the near past is divided by its download time. It is assumed that the BL has a bitrate of R_0, and there are ELs up to quality l, with the bitrate of the BL + ELs up to quality l equal to R_l. The percentage of tiles covered by the predicted viewport is denoted by p. The adaptation logic chooses the highest quality level l for the tiles within the predicted viewport such that:

$$R_0 + (R_l - R_0) \times p < throughput_{est} \quad (3.16)$$

The average of the last three samples of available throughput is denoted by $throughput_{est}$. The EL tiles with quality l covered by the viewport prediction are downloaded and received by the client. In (3.16) it is assumed that the bitrate of a video is uniformly distributed among tiles and the bitrate of the BL + EL up to quality l cannot exceed the estimated bandwidth.

3.8.3 Performance Assessment

The performance evaluation of the proposed solution includes two parts, the first focused on pure coding efficiency while the other about the streaming performance. The first part is not reported here, but it was shown that SHVC suffers from a lower performance with respect to HEVC and that coding efficiency decreases when the number of tiles increase.

Regarding the streaming performance, the setup used for evaluation of the proposed solution consisted of one client connected to a HTTP web server hosting the video chunks. This connection can be emulated to have the behavior of a wireless connection in a mobile network. In all streaming performance experiments, the client played 120 seconds of a fast motion omnidirectional video. Two solutions were tested:

- **Scalable tiled streaming:** This corresponds to the proposed solution where tiles are encoded with SHVC quality scalability. Only two quality layers were used, with a QP of 34 for BL and a QP of 28 for EL. For the video used for evaluation, the resulting bitrate is 3230 Kbit/s for BL and 8229 Kbit/s for BL+EL. When streaming, the adaptation logic used by the client corresponds to the one described in the previous section. A buffer of 10 seconds for the BL is considered.
- **Non-scalable tiled streaming:** Tiles are encoded with HEVC in two quality levels: QP of 34 for quality level 0 and QP of 28 for quality level 1. For the tested video, the resulting bitrate is 3230 Kbit/s for quality level 0 and 7148 Kbit/s for quality level 1. When streaming with this solution, the client predicts viewport for the next 1 second. Then, it uses throughput estimation to decide to get either all tiles with quality level 0 or the viewport covered tiles at quality level 1 and the rest

with quality level 0. All requested tiles of a video chunk are buffered before playback of the correspondent video chunk.

The client runs the player application which supports streaming of both non-scalable and scalable tiled videos with respective adaptation logic included. The vertical and horizontal FoV considered are 90 and 100 degrees, respectively.

I. Bandwidth saving experiment

For this experiment, both non-scalable and scalable tiled streaming solutions were compared in terms of bandwidth saving with the conventional approach, which streams the whole video at QP of 28. For this experiment, the following conditions are applied:

- Two static viewport scenarios: a best-case scenario, with the viewport entirely located at one face of the cube, and a worst-case scenario, with the viewport located at the intersection point between three faces of the cube.
- Two scenarios with different number of tiles per cube face: 1 and 4.
- Constant bandwidth of 50 Mbit/s to assure streaming of highest quality without any interruption.

Figure 3.11 shows the results for bandwidth saving. In the best-case scenario, only one face of the cube is enough to cover viewport and scalable tiled has approximately 2% lower bandwidth savings, mainly due to the coding overhead of the scalable representation. In the worst-case scenario, the difference between 1 and 4 tiles per cube face is more visible. The reason is that using 4 tiles per face means that the worst-case scenario corresponds to 7 tiles of size 512x512 pixels, while for the 1 tile case, the viewport is covered with 3 tiles of size 1024x1024.

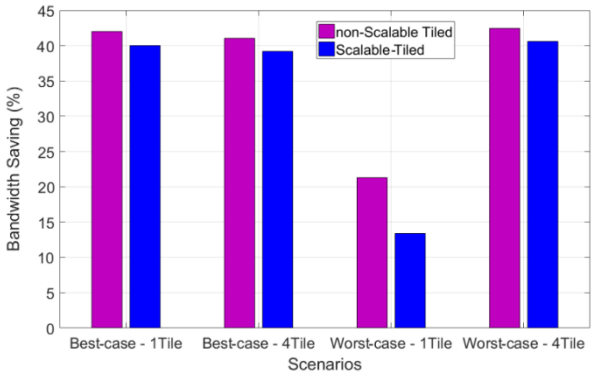


Figure 3.11 – Bandwidth saving results [26]

II. Quality of Experience experiment

For this experiment, non-scalable and scalable tiled streaming solutions were compared in terms of QoE. For this experiment, the following conditions were applied:

- Two static viewport scenarios used on bandwidth saving experiment.
- Four tiles per face.
- Five pre-recorded traces of network packet transmissions and a constant bandwidth trace of 50Mbit/s.

Figure 3.12 shows the first 120 seconds of a streaming session (playback only). The right column is for scalable streaming and the left column shows non-scalable streaming. Blue parts show playback at high quality, red parts show playback at low quality, and white spaces show rebufferings. Results show reduced buffering for scalable tiled streaming and thus much better overall quality of experience. Also, the number of quality switches and average quality is improved using scalable tiled streaming. The main reason is that the buffering of the base layer enables the adaptation algorithm used on scalable tiled streaming to react better to network changes.

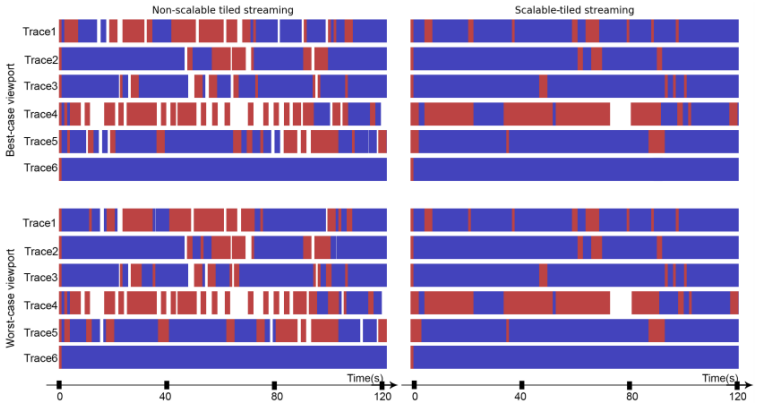


Figure 3.12 – First 120 seconds of experiments with best and worst case viewport, and 4 tiles per face (white = rebuffering, red = low quality playback, blue = high quality playback) [26]

Chapter 4. Offset Cubemap Projection for Omnidirectional Video Streaming

4.1 Introduction

The solution currently used by Facebook's Oculus to address omnidirectional video streaming over bandwidth limited channels is a variant of the cubemap (CM) projection, known as offset cubemap (OCM) projection [8]. This projection distorts the image in such a way that results in a higher spatial resolution (thus, in a higher quality) for the frame regions mapped to the cube front-face, than for other directions. Encoding several versions of the omnidirectional video, where each version corresponds to a different frame region projected on the cube´s front face, enables the implementation of a viewport adaptive video streaming strategy. This chapter is focused on the OCM projection and on its application to omnidirectional video streaming, as described in [8], and proposes an optimization of it.

Section 4.2 provides a formal description of the CM projection, since it is the basis of the OCM one. Section 4.3 presents a conceptual description of the OCM projection, and the conditions used by Oculus to provide viewport adaptive video streaming using this projection. The OCM projection is then formally described in Section 4.4, together with the process of directly rendering a viewport from an OCM projected planar image. Finally, Section 4.5 proposes an optimization of the OCM conditions used by Oculus, which will be assessed and compared with other streaming strategies in Chapter 5.

4.2 Cubemap Projection

This section presents the method to create a CM projected omnidirectional video. The rendering of a rectilinear projected viewport from a CM projection is also described.

4.2.1 Cubemap Creation

Consider the sphere inscribed in a cube presented in Figure 4.1a), with Cartesian coordinates, (X, Y, Z), centered at point O and with unit radius. Points on the sphere surface can be described by longitude (ϕ), with origin on the Z-axis and range $[-\pi, \pi]$, and latitude (θ), with origin on the XZ plane and range $[-\pi/2, \pi/2]$. The Cartesian coordinates of a point at the sphere surface are obtained from the corresponding spherical coordinates by:

$$X = \cos\theta \sin\phi \tag{4.1}$$

$$Y = \sin\theta \tag{4.2}$$

$$Z = \cos\theta \cos\phi. \tag{4.3}$$

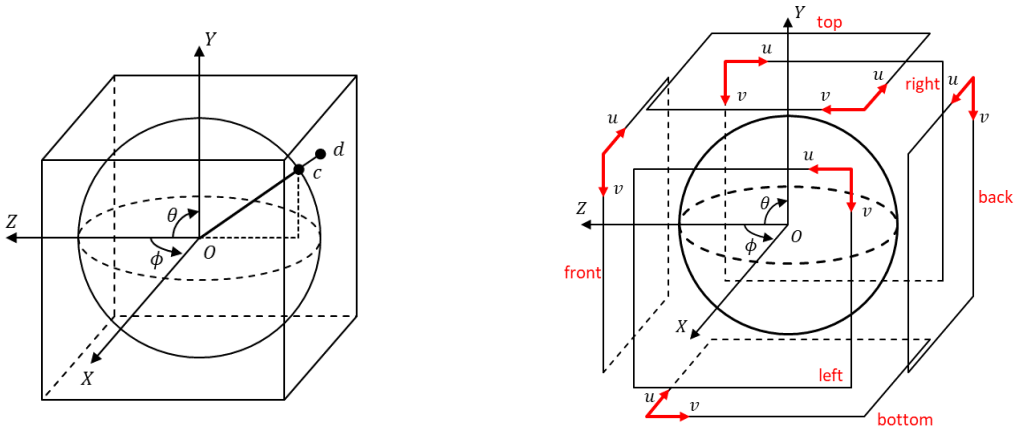
Longitude and latitude can be computed by:

$$\phi = \tan^{-1}\frac{X}{Z} \tag{4.4}$$

$$\theta = \tan^{-1}\frac{Y}{\sqrt{X^2 + Z^2}}. \tag{4.5}$$

The CM projection consists on the projection of a point at the sphere surface, described by a vector $\boldsymbol{c}$, onto a cube containing the sphere. The result is a vector $\boldsymbol{d}$ with same longitude and latitude as vector $\boldsymbol{c}$.

Let (u, v) be a 2D coordinate system, associated to each cube face with a length of two units, as represented in Figure 4.1b); for each cube face, the relationship between coordinates (d_X, d_Y, d_Z) of vector $\boldsymbol{d}$, and the (u, v) coordinates of the same vector, is given in Table 4.1.



a) Projection of a point from the sphere to the cube. *b) Coordinate system for each cube face.*

Figure 4.1 – Coordinates definition for cubemap.

Table 4.1 – Vector $\boldsymbol{d}$ coordinates for a given (u, v) and cube face.

Cube Face	Cartesian coordinates		
	d_X	d_Y	d_Z
Front	$-u+1$	$-v+1$	1
Back	$u-1$	$-v+1$	-1
Top	$-u+1$	1	$v-1$
Bottom	$-u+1$	-1	$-v+1$
Left	1	$-v+1$	$u-1$
Right	-1	$-v+1$	$-u+1$

Consider that W, H are, respectively, the width and the height (in pixels) of a cube face, and that the coordinate system (m, n) defines the pixel position in a CM face, as shown in Figure 4.2; the relationship between these coordinates and the (u, v) coordinates, is given by:

$$u = \frac{2}{W}(m + 0.5)\,, \qquad 0 \leq m < W \tag{4.6}$$

$$v = \frac{2}{H}(n + 0.5)\,, \qquad 0 \leq m < H \tag{4.7}$$

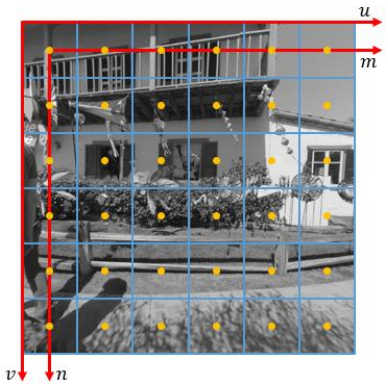
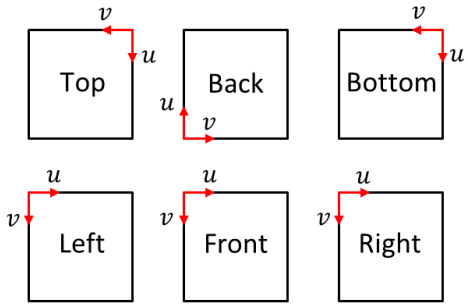


Figure 4.2 – CM face coordinate systems.

The spherical coordinates (in longitude and latitude) of a pixel with coordinates (m, n) in one given CM face can then be obtained by applying (4.6) and (4.7), followed by computing its (X, Y, Z) coordinates using Table 4.1, and then its longitude and latitude with (4.4) and (4.5), respectively. Figure 4.3a) shows a cube face layout that assures horizontal spatial continuity between faces; Figure 4.3b) and c) depict, respectively, an equirectangular video frame and the resulting CM projected frame.



a) Face layout with (u, v) coordinate system

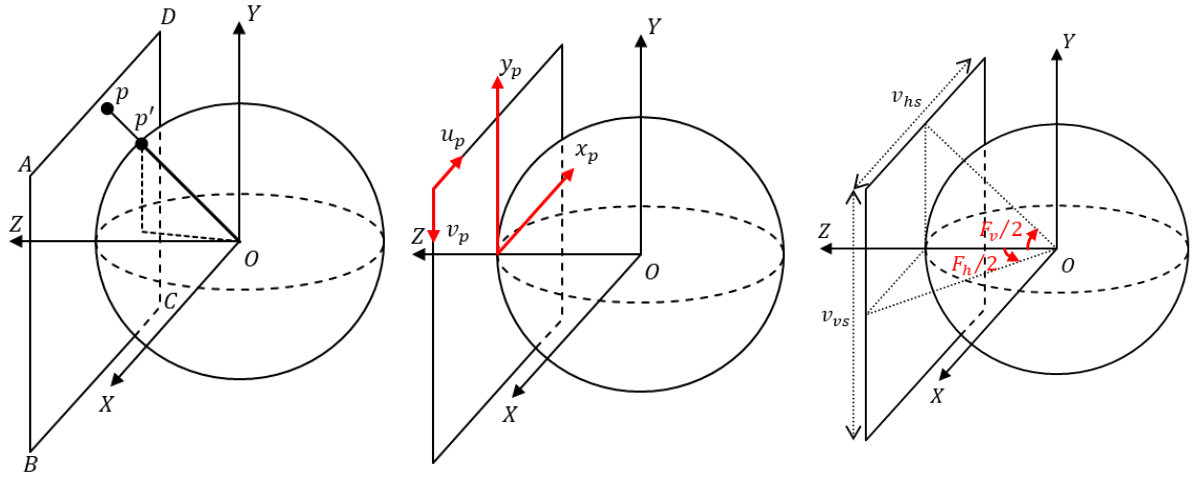
b) Equirectangular video frame

c) CM video frame.

Figure 4.3 – CM projection face layout.

4.2.2 Viewport Rendering

Figure 4.4a) shows the rectilinear projection of a viewport. The projection plane, denoted as $ABCD$, is perpendicular to the Z-axis and tangent to the sphere at $Z = 1$. The projection center is located at point O. Point $\boldsymbol{p}$ (on the plane) is the projection of point $\boldsymbol{p}'$ (on the sphere). Figure 4.4b) defines the coordinate systems (x_p, y_p) and (u_p, v_p) at the viewport. Figure 4.4c) defines the viewport horizontal and vertical sizes (in length units), v_{hs} and v_{vs}, and horizontal and vertical field of views, F_h and F_v.



a) Projection of one point. *b) Viewport coordinate systems* *c) FoV and viewport size.*

Figure 4.4 – Viewport rendering.

From Figure 4.4 c), the viewport Relation between viewport sizes and field of views is given by:

$$v_{hs} = 2\tan\frac{F_h}{2} \tag{4.8}$$

$$v_{vs} = 2\tan\frac{F_v}{2} \tag{4.9}$$

The aspect ratio of the viewport is defined by:

$$Aspect\ Ratio = \frac{v_{hs}}{v_{vs}} = \frac{W_p}{H_p} = \frac{\tan\frac{F_h}{2}}{\tan\frac{F_v}{2}} \tag{4.10}$$

where W_p and H_p are, respectively, the width and the height (in pixels) of the viewport. On CM projection, described in the previous subsection, the projection on each cube face is done using a rectilinear projection with horizontal and vertical FoVs of 90°; generalizing for different FoV, it is possible to use the coordinate systems of Figure 4.2. The viewport pixel (m_p, n_p) has viewport coordinates (u_p, v_p) given by:

$$u_p = (m_p + 0.5)\frac{v_{hs}}{W_p}, \qquad 0 \le m_p < W_p \tag{4.11}$$

$$v_p = (n_p + 0.5)\frac{v_{vs}}{H_p}, \qquad 0 \le n_p < H_p \tag{4.12}$$

where W_p and H_p are, respectively, the width and the height (in pixels) of the viewport. The viewport coordinates (u_p, v_p) and (x_p, y_p) are related by:

$$x_p = u_p - \frac{v_{hs}}{2} \tag{4.13}$$

$$y_p = -v_p + \frac{v_{vs}}{2} \tag{4.14}$$

The resulting cartesian coordinates of viewport point $\boldsymbol{p}$ are given by:

$$\boldsymbol{p} = (-x_p, y_p, 1) \tag{4.15}$$

Since point $\boldsymbol{p}'$ is contained on the spherical surface with unitary radius, its norm is one. Coordinates of point $\boldsymbol{p}'$ can then be obtained by normalizing $\boldsymbol{p}$, resulting in:

$$\boldsymbol{p}' = \frac{\boldsymbol{p}}{\|\boldsymbol{p}\|} \tag{4.16}$$

For a viewport centered at $(\theta_v, \phi_v) = (0°, 0°)$, the viewport matches the omnidirectional coordinates. In this condition, the projected point $\boldsymbol{p}'$ has the same cartesian coordinates as vector $\boldsymbol{c}$. For a viewport centered at $(\theta_v, \phi_v) \neq (0°, 0°)$, it is required to relate the projected point $\boldsymbol{p}'$ coordinates with the corresponding omnidirectional coordinates, $\boldsymbol{c}$, given by:

$$\boldsymbol{c} = R_{\phi v} R_{\theta v} \boldsymbol{p}' \tag{4.17}$$

where $R_{\theta v}$ and $R_{\phi v}$ are the rotations matrices for viewport center angles θ_v and ϕ_v, respectively. The rotation matrices R_θ and R_ϕ for given rotation angles θ and ϕ are obtained by:

$$R_\theta = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & \sin\theta \\ 0 & -\sin\theta & \cos\theta \end{bmatrix}. \tag{4.18}$$

$$R_\phi = \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \tag{4.19}$$

The projection of vector $\boldsymbol{c}$ in the CM, vector $\boldsymbol{d}$, has one of its coordinates (d_X, d_Y, d_Z) equal to 1 or -1 and the other coordinates contained in the interval $[-1,1]$. This is obtained by dividing vector $\boldsymbol{c}$ coordinates by the absolute value of the coordinate of $\boldsymbol{c}$ with maximum absolute value, expressed as follows:

$$\boldsymbol{d} = \frac{\boldsymbol{c}}{\max\{|c_X|, |c_Y|, |c_Z|\}}. \tag{4.20}$$

The relationship between vector $\boldsymbol{d}$, CM face coordinates (u, v) and CM face direction are given by Table 4.2.

Table 4.2 – Correspondence of vector $\boldsymbol{d}$ with cube face and coordinates (u, v).

Max. absolute index	Sign	Face	$\boldsymbol{u}$	$\boldsymbol{v}$
$\lvert\boldsymbol{d_X}\rvert > \lvert\boldsymbol{d_Y}\rvert \wedge \lvert\boldsymbol{d_X}\rvert > \lvert\boldsymbol{d_Z}\rvert$	$d_X = 1$	Left	$d_Z + 1$	$-d_Y + 1$
	$d_X = -1$	Right	$-d_Z + 1$	$-d_Y + 1$
$\lvert\boldsymbol{d_Y}\rvert > \lvert\boldsymbol{d_X}\rvert \wedge \lvert\boldsymbol{d_Y}\rvert > \lvert\boldsymbol{d_Z}\rvert$	$d_Y = 1$	Top	$-d_X + 1$	$d_Z + 1$
	$d_Y = -1$	Bottom	$-d_X + 1$	$-d_Z + 1$
$\lvert\boldsymbol{d_Z}\rvert > \lvert\boldsymbol{d_X}\rvert \wedge \lvert\boldsymbol{d_Z}\rvert > \lvert\boldsymbol{d_Y}\rvert$	$d_Z = 1$	Front	$-d_X + 1$	$-d_Y + 1$
	$d_Z = -1$	Back	$d_X + 1$	$-d_Y + 1$

Using CM face coordinates (u, v) is then possible to compute the CM face pixel position (m, n), solving (4.6) and (4.7) to (m, n), resulting in:

$$m = 0.5(Wu - 1), \qquad 0 \leq u \leq 2 \tag{4.21}$$

$$n = 0.5(Hv - 1), \qquad 0 \leq n \leq 2 \tag{4.22}$$

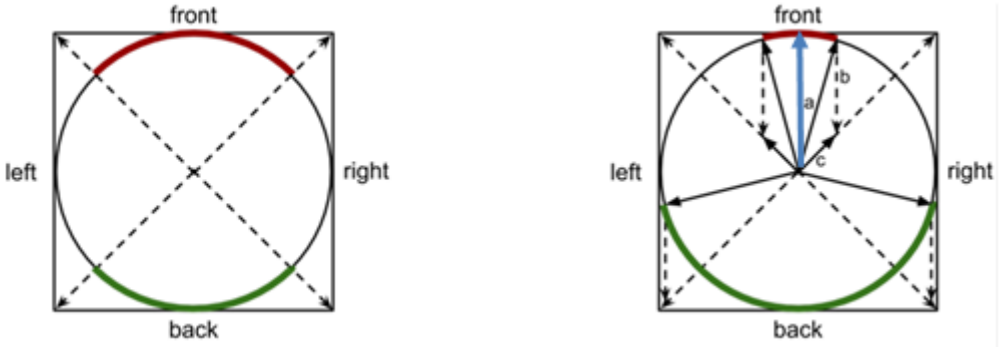
A viewport pixel with coordinates (m_p, n_p) can then be rendered from a CM projection by applying (4.11) to (4.17) and (4.20), followed by computation of cube face and coordinates (u, v) using Table 4.2, and finally obtaining the CM face pixel coordinates (m, n) with (4.21) and (4.22).

4.3 Offset Cubemap Projection

To provide viewport adaptive streaming, thus improving the bandwidth efficiency relatively to a monolithic streaming solution, Oculus uses a projection format referred to as offset cubemap (OCM)

projection [8]. This projection is similar to the standard CM projection, described in Section 4.2, mapping the pixels from the spherical image to six cube faces. However, the spherical angle interval near pre-established orientations – *offset orientation*s, defined by $(\theta_{offset}, \phi_{offset})$ – is distorted by the projection, so that the corresponding sphere area is mapped on a larger cube area, and thus represented with higher spatial sampling density (pixel per unit area on the spherical surface) than the rest of the sphere. By creating several video versions with different $(\theta_{offset}, \phi_{offset})$ values, the client may request the version that better matches the user viewing orientation.

Figure 4.5 shows, side by side, the standard CM and the OCM projections; the red curve represents the sphere area that is projected to the front face, and the green curve represents the sphere area projected to the back face; the blue vector represents a pre-defined offset orientation. The conversion between the standard CM and the OCM projections is done by considering a unit vector, $\boldsymbol{a}$, pointing to a pixel on the sphere surface, and adding an *offset vector*, $\boldsymbol{b}$, with an orientation opposite to the vector defining the offset orientation (blue vector in Figure 4.5). The resulting vector, $\boldsymbol{a} + \boldsymbol{b} = \boldsymbol{c}$, points to the resulting pixel position on the cube surface. Since the cube faces have the same number of pixels, the area on the sphere projected to the front face is represented with higher spatial sampling density, thus encoded in higher quality, than areas covered by the back face. Information about $(\theta_{offset}, \phi_{offset})$ and vector $\boldsymbol{b}$ is provided to the clients through the DASH MPD file.



a) Standard CM projection. *b) Offset CM projection.*

Figure 4.5 – Comparison of the standard CM projection and the OCM projection. The red portion and the green portion of the circle indicate the portions of the sphere mapped to the cubes' front and back face, respectively [8].

In the Oculus' implementation of the OCM projection, $\|\boldsymbol{b}\| = 0.7$. This causes the front face to cover roughly 30 degrees, horizontally and vertically, and the back face to cover roughly 150 degrees, horizontally and vertically. Figure 4.6a) presents the limits of the OCM faces in a equirectangular projected image and Figure 4.6b) depicts the correspondent OCM projected image, using $\|\boldsymbol{b}\| = 0.7$.

a) Equirectangular image, with the OCM face limits. *b) Image resulting from the OCM projection.*

Figure 4.6 – The OCM projection for $\|\boldsymbol{b}\| = 0.7$.

Oculus encodes a omnidirectional video using 22 offset orientations, listed in Table 4.3; Figure 4.7 presents the regions of a omnidirectional video that are projected on the cube front face, for all 22 offset orientations. The combination of the 22 front faces does not cover the full spherical surface, as each front face only covers a field of view (FoV) of 30º (in both directions). To account for bandwidth adaptation, Oculus uses four spatial resolutions of the cube faces: 272×272 (272w), 400×400 (400w), 528×528 (528w) and 656×656 (656w). This results in 88 versions of an omnidirectional video [8].

Table 4.3 – Pre-defined orientations for viewport adaptive streaming, used by Oculus [8].

θ_{offset} (in degrees)	ϕ_{offset} (in degrees)
90	0
45	15, 105, 195, 285
0	0, 30, 60, 90, 120, 150, 180, 210, 240, 270, 300, 330
- 45	15, 105, 195, 285
- 90	0

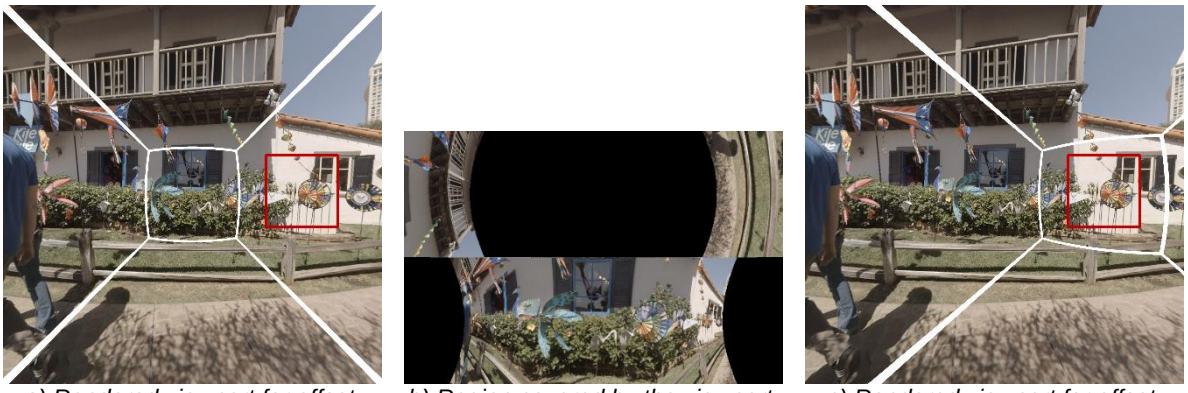
Figure 4.7 – Representation of the front faces of all 22 OCMs on a equirectangular image.

Positioning the six faces of an OCM projection in a 3×2 layout (three columns and two rows), an OCM projection with a face resolution of W×H pixels results in an OCM planar image with a resolution of 3W×2H pixels, representing the entire omnidirectional image. Depending on the viewport FoV and spatial resolution, the quality of a viewport extracted from an OCM projection changes:

- The wider the viewport FoV, the viewport coverage of areas not represented at the OCM front face center increase. Since these areas have lower spatial sampling density, the quality of the viewport decreases.
- The higher the viewport spatial resolution, the higher the distortion caused by spatial upsampling of the lower spatial sampling density regions represented outside the OCM front face, reducing the quality of the viewport.

The highest spatial resolution considered (by Oculus) for each OCM face, 656w, results in an omnidirectional video with spatial resolution of 1968×1312 pixels. The viewport configuration used in [8] has a spatial resolution of 2000×2000 pixels; accordingly, the number of pixels of the viewport surpasses the number of pixels of the OCM projection. The same viewport configuration has horizontal and vertical FoV of 96°. Figure 4.8a) shows the rendered viewport from the OCM projection, with added white lines

to represent the borders between OCM faces; the viewport center is the same as the OCM front face center and has orientation (0°,0°). Figure 4.8b) depicts the pixels of the OCM used for rendering the viewport of Figure 4.8a), showing that the rendered viewport uses a considerable portion of the entire OCM for a horizontal and vertical FoV of 96°. However, since the total number of pixels of the OCM is already smaller than the number of pixels of the viewport, the content used of the OCM requires spatial upsampling to render the viewport. Moreover, in Figure 4.8a), only a small portion of the viewport is covered by the OCM front face. Thus, viewport regions covered by other OCM faces are highly spatially upsampled, causing noticeable distortions. To visualize this distortion, the viewport presented in Figure 4.8c) was rendered with the same viewport direction, (0°,0°), but using an OCM with offset orientation (0°,30°). Figure 4.8d) and e) correspond to the red delimited regions of Figure 4.8a) and c), respectively. For the offset orientation (0°,0°) it is possible to notice a quality degradation from the left to the right as the distance from the offset orientation increases, where the content is highly upsampled. For offset orientation (0°,30°) no noticeable distortions are present since the region showed is covered by the OCM front face. Since a viewport centered with the offset orientation presents noticeable quality degradation, an offset of 0.7 does not provide good quality for a viewport with spatial resolution of 2000×2000 pixels and horizontal and vertical FoVs of 96°.



a) Rendered viewport for offset orientation (0°,0°). *b) Region covered by the viewport for offset orientation (0°,0°).* *c) Rendered viewport for offset direction (0°,30°).*

d) Red delimited region for offset orientation (0°,0°). *e) Red delimited region for offset orientation (0°,30°).*

Figure 4.8 – Viewport rendering for OCM with offset 0.7 and viewport of 2000×2000 spatial resolution and 96º horizontal and vertical FoV. White lines on a) and c) correspond to OCM face limits. Red lines on a) and c) delimit the regions shown in c) and d), respectively. Black regions on b) are not used for the viewport rendered in a).

4.4 Implementation of the Offset Cubemap

Figure 4.9 presents the OCM projection applied to the same coordinate systems defined in section 4.2; vector $\boldsymbol{a}$ describes a point at the sphere surface with unitary radius, vector $\boldsymbol{b}$ is the offset vector, vector $\boldsymbol{c}$ points to the resulting pixel position on the cube surface, and vector $\boldsymbol{d}$ describes the pixel position on the cube surface.

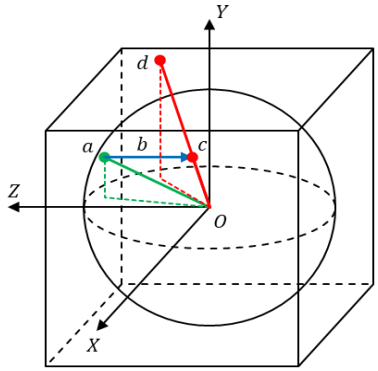


Figure 4.9 – OCM projection and related 3D coordinate system.

The following subsections provide details about the implementation of the OCM projection, notably the creation of the corresponding OCM planar image and the viewport rendering from this image.

4.4.1 Offset Cubemap Creation

Vector $\boldsymbol{d}$, with Cartesian coordinates (d_X, d_Y, d_Z), has the same direction of vector $\boldsymbol{c}$, which results in:

$$\boldsymbol{c} = K\boldsymbol{d}, \tag{4.23}$$

where K is a positive value. Since vector $\boldsymbol{b}$ has Cartesian coordinates $(0,0,b_Z)$, $b_z \in]-1;0]$, vector $\boldsymbol{a}$ is given by:

$$\boldsymbol{a} = \boldsymbol{c} - \boldsymbol{b} = (Kd_X, Kd_Y, Kd_Z - b_Z). \tag{4.24}$$

As the sphere has unity radius:

$$\|\boldsymbol{a}\| = 1 \Rightarrow K^2{d_X}^2 + K^2{d_Y}^2 + K^2{d_Z}^2 - 2Kd_Zb_Z + {b_Z}^2 = 1. \tag{4.25}$$

Applying the quadratic formula and considering that K is a positive value, results:

$$K = \frac{d_Zb_Z + \sqrt{{d_Z}^2{b_Z}^2 - \|\boldsymbol{d}\|^2\left({b_Z}^2 - 1\right)}}{\|\boldsymbol{d}\|^2}. \tag{4.26}$$

The OCM projection with $\left(\theta_{offset}, \phi_{offset}\right) = (0,0)$ can be obtained adding an additional step to the CM projection: starting with the pixel coordinates (m,n) on one given OCM face, equations (4.6) and (4.7) are first applied, followed by the computation of cartesian coordinates of vector $\boldsymbol{d}$, using Table 4.1; by applying (4.26) and (4.24) the Cartesian coordinates of vector $\boldsymbol{a}$ are obtained, which can be converted to longitude and latitude with (4.4) and (4.5), respectively. For an OCM projection with $\left(\theta_{offset}, \phi_{offset}\right) \neq (0,0)$, and after computing $\boldsymbol{a}$, it is required to further relate this coordinate with the corresponding omnidirectional coordinates, $\boldsymbol{a}'$, given by:

$$\boldsymbol{a}' = R_{\phi off}R_{\theta off}\boldsymbol{a}, \tag{4.27}$$

where $R_{\theta off}$ and $R_{\phi off}$ are the rotations matrices for angles θ_{offset} and ϕ_{offset}, respectively, computed using (4.18) and (4.19).

4.4.2 Viewport Rendering

Viewport rendering from an OCM projection processes differently after determination of the vector of coordinates on the sphere surface, $\boldsymbol{p}'$. Two coordinates transformation are required. First, a rotation is made to match the omnidirectional coordinates, $\boldsymbol{a}'$, given by:

$$\boldsymbol{a}' = R_{\phi v} R_{\theta v} \boldsymbol{p}'. \tag{4.28}$$

Then, a rotation is made to match the coordinate system of the OCM projection:

$$\boldsymbol{a} = {R_{\phi off}}^{-1} {R_{\theta off}}^{-1} \boldsymbol{a}', \tag{4.29}$$

where ${R_{\phi off}}^{-1}$ and ${R_{\theta off}}^{-1}$ are easily obtained by changing the sign of ϕ_{offset} and θ_{offset} in (4.19) and (4.18).

With the coordinates of the correspondent viewport point $\boldsymbol{a} = (a_X, a_Y, a_Z)$ and the offset vector $\boldsymbol{b} = (0,0,b_z)$, it is possible to obtain vector $\boldsymbol{c}$:

$$\boldsymbol{c} = (a_X, a_Y, a_Z + b_z). \tag{4.30}$$

The interception of vector $\boldsymbol{c}$ with the cube face gives the coordinates of the point on the surface of the OCM, vector $\boldsymbol{d}$. One of its coordinates, (d_X, d_Y, d_Z), is equal to 1 or -1 and the other coordinates are contained in the interval $[-1,1]$, given by:

$$\boldsymbol{d} = \frac{\boldsymbol{c}}{\max\{|a_X|, |a_Y|, |a_Z + b_z|\}}, \tag{4.31}$$

A viewport with coordinates (m_p, n_p) can then be rendered from a OCM projection by applying (4.11) to (4.16), (4.28) to (4.31), followed by computation of cube face and coordinates (u, v) using Table 4.2, finally obtaining the OCM face pixel coordinates (m, n) with (4.21) and (4.22).

4.5 Assessment of the Offset Magnitude Impact on Quality

This section provides an evaluation to determine the best OCM offset magnitude, to be used in omnidirectional video streaming. First, the test conditions are described. Then, an evaluation of the rate-distortion curve obtained for different offset values is presented.

4.5.1 Test Conditions

The test video sequences were selected from the JVET dataset [44]. From this dataset, three videos were used: *ChairliftRide*, *SkateboardInLot* and *KiteFlite*. *ChairliftRide* has moderate motion content, *SkateboardInLot* has fast motion content, and *KiteFlite* has low motion but contains high spatial detail. All videos are in the equirectangular format, with spatial resolution of 7680×3840 pixels, temporal resolution of 30 frames/s and have a duration of 10 seconds. The first frame of *ChairliftRide*, *SkateboardInLot* and *KiteFlite* are presented in Figure 4.10.

a) ChairliftRide *b) SkateboardInLot* *c) KiteFlite*

Figure 4.10 – First frame of omnidirectional video test sequences.

A total of 11 offset values, presented in Table 4.4, were assessed. The spatial resolutions of the OCM frames were selected according to the equirectangular frames resolutions, notably:

- The OCM front face spatial sampling density should be approximately equal to the spatial sampling density of regions near the equator of the equirectangular frames, to not spatially

oversample regions covered by the OCM front face. The width of a cube face, W_F, is approximately given by:

$$W_F \cong \frac{W_{equi}}{360^\circ} \times F_{FF}, \tag{4.32}$$

where W_{equi} is the width of the equirectangular frame and F_{FF} is the front face angle in degrees.

- Each OCM face size should be a multiple of 64, to allow correct split of the frame in slices, during HEVC encoding.

The equivalent equirectangular spatial resolutions selected were 7680×3840 for high spatial resolution (HSR), 6144×3072 for medium spatial resolution (MSR), and 4608×2304 for low spatial resolution (LSR), resulting in three RD points. For each spatial resolution, five OCM offset orientations were considered: (0°, 0°), (60°, 0°), (-60°, 0°), (0°, 90°) and (0°, -90°). Since rendered viewports were fixed with respective center matching the offset orientation, these offset orientations provided five different viewports directions, covering a larger area of the video. The five OCM offset orientations also provided three viewports for evaluation around the equator, with low overlap between viewports, plus evaluation of regions near each pole. The OCM videos were built from the original version of the videos. The OCM pixel values were obtained by using bicubic interpolation.

Table 4.4 – Offset values tested and corresponding OCM face width.

Offset $\|\boldsymbol{b}\| = \|\boldsymbol{b}_z\|$	**Front Face Angle**	**OCM Face Width**		
		HSR (7680×3840 ERP)	MSR (6144×3072 ERP)	LSR (4608×2304 ERP)
0.70	**30º**	640	512	384
0.64	**36º**	768	640	448
0.58	**41º**	896	704	512
0.54	**45º**	960	768	576
0.48	**50º**	1088	832	640
0.42	**55º**	1152	960	704
0.36	**60º**	1280	1024	768
0.30	**65º**	1408	1088	832
0.24	**70º**	1472	1216	896
0.18	**75º**	1600	1280	960
0.12	**80º**	1728	1344	1024

The videos were encoded with the reference software for HEVC, HEVC Test Model (HM) version 16.20 [45], using the Random Access configuration with a GOP size of 16 frames. Only the first 80 frames of each video were encoded; no temporal segmentation was considered in this test. For each video, offset value, and spatial resolution, the Quantization Parameter (QP) was adjusted in order to result in similar bitrate values for the same spatial resolution, thus, regardless of the used offset value and video sequence. Table 4.5 presents the QP values used for the test. All OCM videos were encoded with two slices, one corresponding to the top, back, and bottom cube faces, and the other corresponding to the left, front, and right cube faces (see Figure 4.3-a); this avoids artifacts produced around the border

between the two slices, since it blocks intra and inter encoding between the spatial discontinuity of the frame.

Table 4.5 – QP values used

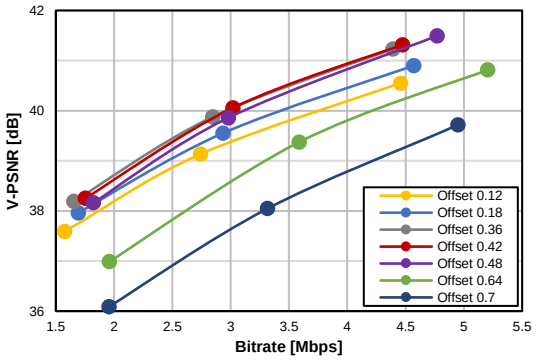
Offset $\|\boldsymbol{b}\| = \|\boldsymbol{b}_z\|$	**Front Face Angle**	**QP**								
		ChairliftRide			SkateboardInLot			KiteFlite		
		HSR	MSR	LSR	HSR	MSR	LSR	HSR	MSR	LSR
0.70	**30º**	24	25	26	27	27	28	26	27	27
0.64	**36º**	25	26	27	28	28	29	27	29	29
0.58	**41º**	26	27	28	29	29	30	28	30	30
0.54	**54º**	27	28	29	30	30	31	30	31	31
0.48	**50º**	28	29	30	31	31	32	31	32	32
0.42	**55º**	29	30	31	32	32	33	32	34	33
0.36	**60º**	30	31	32	32	32	33	33	34	34
0.30	**65º**	31	32	33	32	33	34	33	35	35
0.24	**70º**	31	32	33	33	33	34	34	36	36
0.18	**75º**	32	33	34	33	34	35	34	37	37
0.12	**80º**	33	34	35	34	34	35	35	37	38

The considered viewport dimension was 2000×2000 pixels, with horizontal and vertical FoV of 96º. The viewports are rendered using the rectilinear projection, meaning that the projection center is located at the center of the omnidirectional video sphere. The viewport pixel values were obtained by using bicubic interpolation.

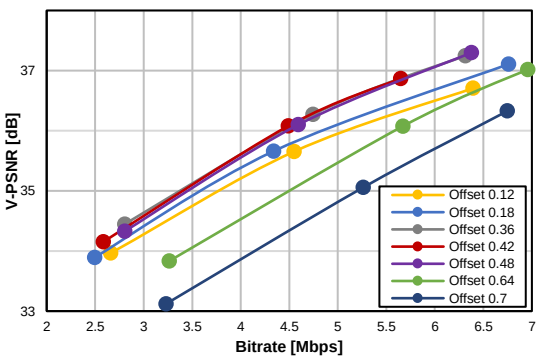
4.5.2 Results and analysis

The goal of the test was to determine the offset value that provides the best results for the used viewport configuration. For each offset values and spatial resolution, viewports were rendered for all 80 frames and for five static viewing directions (latitude, longitude): (0°, 0°), (60°, 0°), (-60°, 0°), (0°, 90°) and (0°, -90°). Each viewport was rendered using the OCM with the offset orientation that matches the center of the viewport. Each rendered viewport was compared to the same viewport rendered from the original video using the PSNR metric, this means computing the V-PSNR for each frame, followed by averaging the V-PSNR for the 80 frames of each sequence.

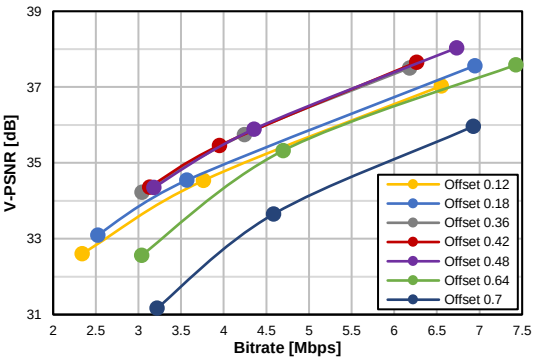
The rate-distortion plots obtained for the viewport directions (0°, 0°) and (0°, -90°), and for the three considered video sequences, are presented in Figure 4.11, where each point corresponds to the RD obtained for each spatial resolution. The offset 0.7 provides the worst rate-distortion curves. As previously discussed in section 4.3, the low spatial sampling density of regions outside the front face, but that are still included in the viewport, reduce the quality of the viewport. Reducing the offset magnitude from 0.7 provides an initial viewport quality increase, followed by an eventual decrease in quality as the offset magnitude reaches low values. Depending on the direction and content, offsets of 0.36, 0.42, and 0.48 provided the best results. As an example, for direction (0°, -90°) and sequence SkateboardInLot, depicted in Figure 4.11e), offsets of 0.42 and 0.48 present a slight advantage compared to an offset 0.36, while for direction (0°, 0°) and sequence KiteFlite depicted in Figure 4.11c), results for offsets of 0.36, 0.42, and 0.48 are quite similar.



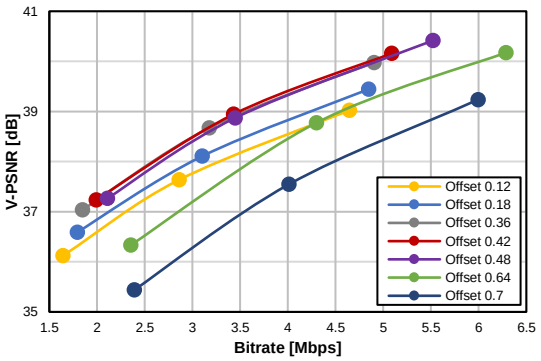
a) Direction (0º, 0º). Sequence: ChairliftRide.



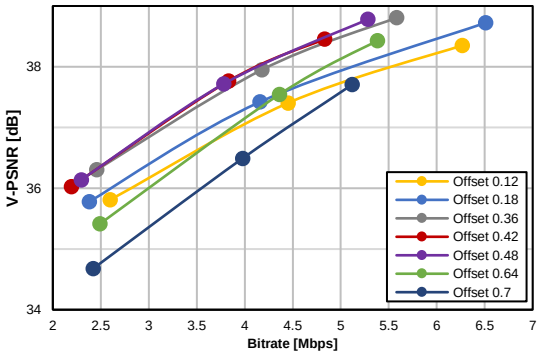
b) Direction (0º, 0º). Sequence: SkateboardInLot.



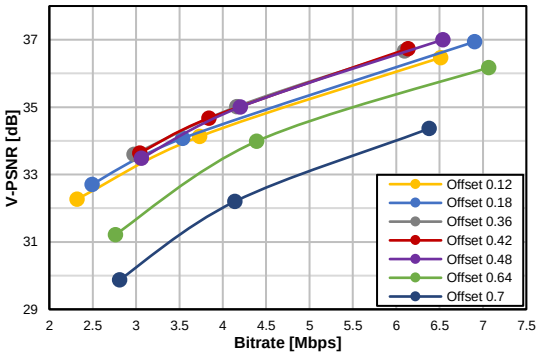
c) Direction (0º, 0º). Sequence: KiteFlite.



d) Direction (0º, -90º). Sequence: ChairliftRide.



e) Direction (0º, -90º). Sequence: SkateboardInLot.

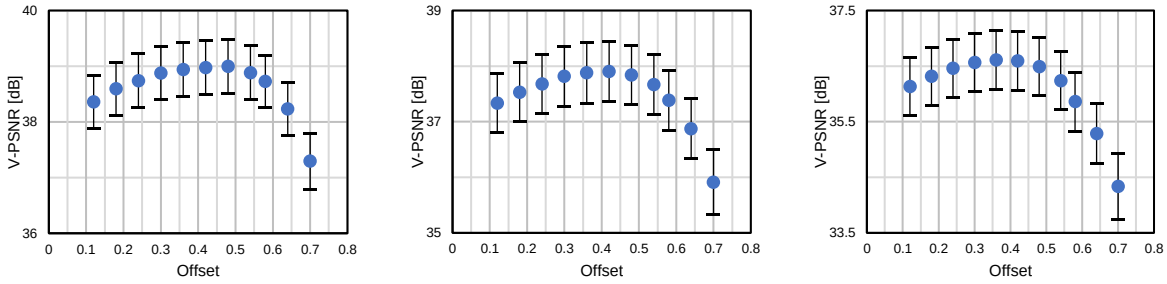


f) Direction (0º, -90º). Sequence: KiteFlite.

Figure 4.11 – RD performance for several offset magnitudes.

To better visualize the evolution of the viewport quality with the offset value, each RD curve was approximated with a second order polynomial fitting. Then, for each direction and sequence, three bitrate values were selected: 1) the minimum bitrate obtained for all HSR RD points, 2) the mean bitrate obtained for all MSR RD points, and 3) the maximum bitrate obtained for all LSR RD points. For these three bitrate values the corresponding V-PSNR values were computed for each RD curve using the corresponding second order polynomial fitting. Finally, the resulting V-PSNR values for each offset magnitude were averaged for all sequences and directions. Figure 4.12 presents the resulting average and standard deviation of the V-PSNR with the offset magnitude for the three bitrate values. Depending on the spatial resolution of the points, the best offset value changes between 0.48 for HSR, 0.42 for

MSR, and 0.36 for LSR. These three offset values outperform offset 0.7. For example, offset 0.42 provides gains between 1.7dB and 2.3dB compared to offset 0.7.



a) Minimum HSR bitrate *b) Mean MSR bitrate* *c) Maximum LSR bitrate*

Figure 4.12 – Viewport quality evolution with offset value.

The results of this test suggest a possible offset value that maximizes the quality of the video for a target bitrate, dependent on the video content; however, it is possible to assume that it is contained in the interval [0.36; 0.48]. As such, the offset value 0.42 is selected for the comparative tests, as it is close to the middle of the interval, providing a good approximation of the best offset value for the considered viewport dimensions and FoV.

Chapter 5. Comparative Study of Different Omnidirectional Video Streaming Strategies

5.1 Introduction

The main objective of this Chapter is to evaluate the proposed offset cubemap (OCM) projection defined in Chapter 4 for omnidirectional video streaming with other state-of-the-art strategies (benchmarks). The performance evaluation is done by measuring objectively viewport quality and accounting for the rate. First, in section 5.2, several omnidirectional video streaming strategies to be used as benchmarks are described. In section 5.3, some initial tests are made considering static viewports, which allows representation of the best and worst case scenarios, providing results that can be seen as the upper and lower bound of what is possible to achieve with several omnidirectional video streaming strategies. Finally, section 5.4 provides a test close to a realistic scenario where viewport evaluation is made with a set of selected trajectories obtained from users that were interacting with the content in the past.

5.2 Selected Omnidirectional Streaming Benchmarks

This section presents the several omnidirectional video streaming strategies which will be compared with the proposed OCM Projection. These benchmark solutions were obtained from the OMAF standard [24], which supports two planar projections for omnidirectional video streaming and describes a spatial segmentation method, and from [25], which recommends only a spatial segmentation method. The omnidirectional video streaming strategies presented can be divided into two categories, namely monolithic streaming and tile based streaming, and are described in the following subsections.

5.2.1 Monolithic Streaming Strategies

Monolithic streaming strategies are one of the most commonly used strategies for omnidirectional video streaming. For monolithic strategies, a single encoded bit stream that contains the full omnidirectional video, where quality is similarly distributed on the entire frame, is streamed. The client decodes the bit stream continuously to obtain the omnidirectional video data, and consequently, any viewport can be rendered. The projection used for a monolithic strategy allows the generation of a viewport for any position and angle in the sphere without any information loss. The two monolithic strategies considered use the two projections currently supported by OMAF, equirectangular and cubemap, and are described next:

- **Monolithic Equirectangular (MonoEqui):** The projection used is the equirectangular projection. The entire equirectangular projected omnidirectional video is encoded with the original resolution and with some target QP. Naturally, the QP value is changed to obtain several rate-distortion points and thus evaluate the solution for several target qualities and rates.

MonoEqui is one of the most common strategy used for omnidirectional video streaming. However, a large part of each omnidirectional frame of the video will not be covered by any viewport, thus wasting a significant amount of bandwidth. Moreover, the distortion near the poles present in the equirectangular projection has some impact on the coding performance.

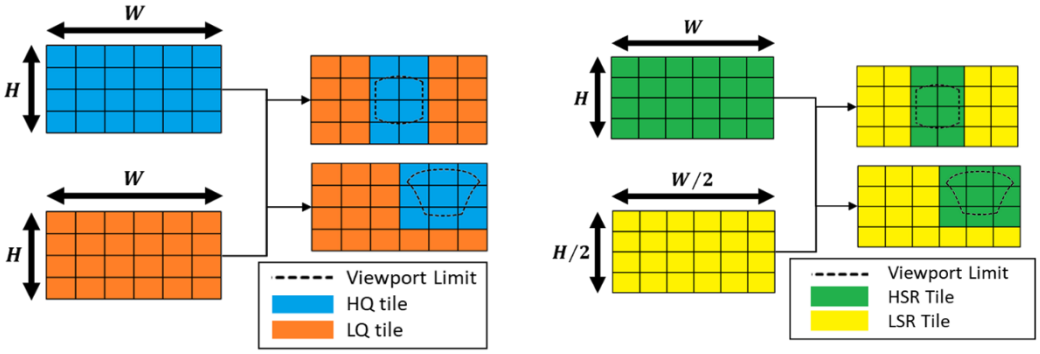
- **Monolithic CM (MonoCM):** The projection used is the CM projection described in section 4.2, with the face layout of Figure 4.3a). The entire cube-map omnidirectional video is encoded with a spatial resolution that approximates the spatial sampling density of regions near the equator of the original video and with some target QP. The QP value is changed to obtain several rate-distortion points. Two slices are considered, one in top of the frame, comprising the top, back and bottom faces of the cube, and one in the bottom of the frame, containing the left, front and right faces of the cube. Cubemap projection produces less quality degradation across the video than the equirectangular projection. MonoCM also provides a comparison to the OCM projection streaming strategy, as the CM projection is an OCM projection with offset zero.

5.2.2 Tiles Streaming Strategies

Tiles streaming strategies divide the omnidirectional video in motion constrained tiles. All tiles strategies layouts use equirectangular projection. Tiles have high quality/spatial resolution or low quality/spatial resolution. The selection of the high quality/spatial resolution tiles is based on the coverage of the viewport, i.e., the entire viewport area must be covered with high quality/spatial resolution tiles. Ideally, only the tiles covering the viewport would be streamed. However, in that case the user head motion could lead to the rendering of blank areas or the corresponding tiles need to be rendered with delay, which decreases the QoE. As such, tiles not covering viewport area are streamed with low quality/spatial resolution. The tiles strategies tested are described next:

- **Tiles 6×4 Quality (6×4-Qual):** Spatial segmentation of the equirectangular omnidirectional video in 6×4 motion constrained tiles, as recommended in [25]. All tiles are generated from the original full omnidirectional video spatial resolution. The low quality (LQ) tiles QP is (+5) higher than the high quality (HQ) tiles QP. The different rate-distortion points are obtained by changing the QP of all tiles, while keeping the QP difference between HQ and LQ tiles fixed. Figure 5.1a) presents two examples of tiles streamed for different viewport positions, where W and H are the width and height of the original omnidirectional video. Blue tiles correspond to HQ tiles, covering the viewport region, and orange tiles correspond to LQ tiles, covering the rest of the omnidirectional frame. Figure 5.2 presents the number of HQ tiles required to completely cover a viewport with 96° horizontal and vertical FoV using a 6×4 tiled layout. Each pixel of the map shown in Figure 5.2 represents the number of tiles with a color, if the viewport center is located at the pixel location. As shown, the number of tiles increases significantly as the viewport center moves to the poles. 6×4-Qual strategy is based on tiles strategy proposed in [25], with the condition of fixed QP difference between HQ and LQ tiles to reduce the impact in QoE due to the difference in quality between HQ and LQ tiles.
- **Tiles 6×4 Spatial Resolution (6×4-SRes):** Spatial segmentation of the equirectangular omnidirectional video in 6×4 uniform tiles, as recommended in [25]. This is similar to the previous

one but the spatial resolution changes instead of the coding quality. The low spatial resolution (LSR) tiles have half the resolution of the high spatial resolution (HSR) tiles but the QP is the same. The four rate-distortion points are obtained by changing the resolution of all tiles, while keeping the resolution ratio between HSR and LSR tiles. This solution avoids coding artifacts by changing the spatial resolution, but some blurring may occur. Figure 5.1b) presents two examples of tiles streamed for different viewport positions, where W and H are the omnidirectional video width and height selected for the HSR tiles. Green tiles correspond to HSR tiles, covering the viewport region, and yellow tiles correspond to LSR tiles, covering the rest of the omnidirectional frame. Since 6×4-SRes uses the same tiles layout as 6×4-Qual, Figure 5.2 also presents the number of HSR tiles required to completely cover a viewport with 96° horizontal and vertical FoV. Since no comparative study between streaming tiles with different quality and different spatial resolution was found, 6×4-SRes strategy is also included as benchmark.



a) 6×4-Qual *b) 6×4-SRes*

Figure 5.1 – Omnidirectional video streaming with tiles 6×4 layout.

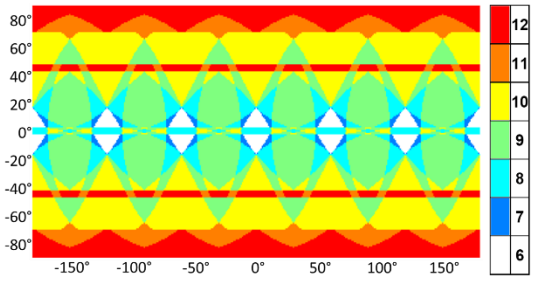
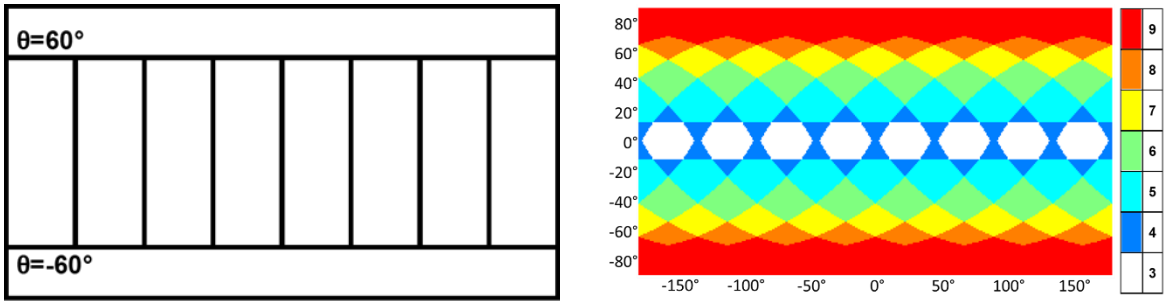


Figure 5.2 – Number of HQ/HSR tiles by viewport center for 6×4 tiles layout. Viewport with 96° horizontal and vertical FoV.

- **Tiles OMAF Quality (OMAF-Qual):** Spatial segmentation of the equirectangular omnidirectional video in motion constrained tiles according to the layout described in the OMAF Annex D.6.3 [1], represented in Figure 5.3a). All tiles are generated from the original full omnidirectional video resolution. The LQ tiles QP is (+5) higher than the HQ tiles QP. The different rate-distortion points are obtained by changing the QP of all tiles, while keeping the QP difference between HQ and LQ tiles fixed. Figure 5.4a) presents two examples of tiles streamed for different viewport positions, where W and H are the width and height of the original omnidirectional video. Blue tiles correspond to HQ tiles, covering the viewport region, and orange tiles correspond to LQ

tiles, covering the rest of the omnidirectional frame. Figure 5.3b) presents the number of HQ tiles required to completely cover a viewport with 96° horizontal and vertical FoV using the OMAF tiled layout. Each pixel of the map shown in Figure 5.3b) represents the number of tiles with a color, if the viewport center is located at the pixel location. As shown, the number of tiles increases as the viewport center moves to the poles, starting from 3 or 4 tiles at the equator to requiring 9 out of 10 tiles near the poles. The same difference between HQ and LQ tiles of 6×4-Qual are given to OMAF-Qual to obtain a comparable strategy.

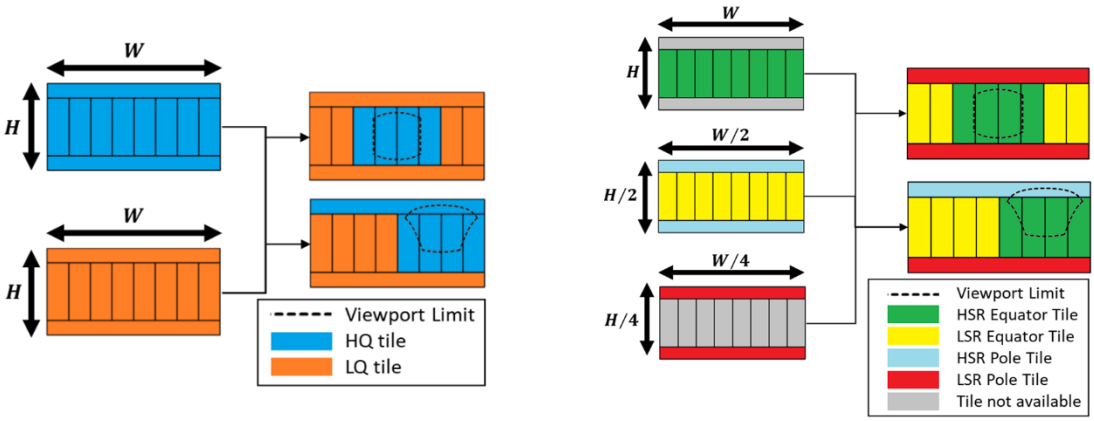
- **Tiles OMAF Spatial Resolution (OMAF-SRes):** Spatial segmentation of the equirectangular omnidirectional video in motion constrained tiles according to the OMAF Annex D.6.3, represented in Figure 5.3a). This strategy takes advantage of the fact that the tiles containing the poles of the omnidirectional video are regions with double (or higher) spatial sampling density compared to the equator. As in 6x4-SRes, the LSR tiles have half the resolution of the HSR tiles but the QP is the same. In addition, tiles of the poles are represented with half the spatial resolution comparing to the tiles of the equator. The four different bitrates are obtained by changing the resolution of all tiles, while keeping the resolution ratio between HSR and LSR tiles (and naturally also of the poles). Figure 5.4b) presents two examples of tiles streamed for different viewport positions, where W and H are the omnidirectional video width and height selected for the HSR tiles contained in the equator. Green and light blue tiles correspond to HSR equator and pole tiles, respectively, covering the viewport region, yellow and red tiles correspond to LSR equator and pole tiles, respectively, covering the rest of the omnidirectional frame, and gray tiles are not available for streaming. Since OMAF-SRes uses the same tiles layout as OMAF-Qual, Figure 5.3b) also presents the number of HSR tiles required to completely cover a viewport with 96° horizontal and vertical FoV. Since no comparative study between tiles OMAF layout and tiles 6×4 layout was found, the same difference between the HSR and LSR tiles resolution of 6×4-SRes are given to OMAF-SRes.



a) Tile layout described in OMAF Annex D.6.3.

b) Number of HQ/HSR tiles by viewport center for OMAF tiles layout. Viewport with 96° horizontal and vertical FoV.

Figure 5.3 – Tiles OMAF layout.



a) OMAF-Qual *b) OMAF-SRes*

Figure 5.4 –Omnidirectional video streaming with tiles OMAF layout.

5.3 Fixed Viewport Evaluation

This section evaluates the proposed OCM solution and the benchmark streaming strategies considering fixed viewports, i.e. viewports which do not change the location. First, the test conditions for the tests are described. Then, an evaluation of the rate-distortion obtained for a viewport covered always by the highest quality is made. Finally, some quality drop is measured in function of the distance of the actual viewport to the requested viewport.

5.3.1 Test Conditions

The test video sequences were selected from the JVET dataset [44]. From this dataset, the same three videos defined in Section 4.5 were used: ChairliftRide, SkateboardInLot and KiteFlite. All videos are in the equirectangular format with spatial resolution of 7680×3840, sampling rate of 30 frames per second and have a duration of 10 seconds.

The spatial resolutions of the omnidirectional video for each benchmark are the following: 7680×3840 for MonoEqui, 6×4-Qual and OMAF-Qual and 5760×3840 for MonoCM. Table 5.1 shows the spatial resolution for 6×4-SRes while the spatial resolutions for OMAF-SRes are shown in Table 5.2.

Table 5.1 – Spatial resolutions of the omnidirectional video for 6×4-SRes.

RD point	HSR	LSR
1	7680×3840	3840×1920
2	6144×3072	3072×1536
3	4608×2304	2304×1152
4	3072×1536	1536×768

Table 5.2 – Spatial resolutions of the omnidirectional video for OMAF-SRes.

RD point	HSR Equator	LSR Equator / HSR Pole	LSR Pole
1	7680×3840	3840×1920	1920×960
2	6144×3072	3072×1536	1536×768
3	4608×2304	2304×1152	1152×576
4	3072×1536	1536×768	768×384

For the proposed OCM projection the resolutions are shown in Table 5.3. In this case, five offset orientations $(\theta_{offset}, \phi_{offset})$ were considered: (0°, 0°), (60°, 0°), (-60°, 0°), (0°, 90°) and (0°, -90°).

Table 5.3 – Resolutions of the omnidirectional video used in OCM projection.

RD point	ERP equivalent resolution	OCM resolution	OCM face size
1	7680×3840	3456×2304	1152
2	6144×3072	2880×1920	960
3	4608×2304	2112×1408	704
4	3072×1536	1344×896	448

The videos were encoded with the HEVC reference software, HEVC Test Model (HM) version 16.20, using the Random Access configuration with GOP size of 16 frames. All 300 frames of each video were encoded; no temporal segmentation was considered in this initial test. For each video and resolution, the Quantization Parameter (QP) was adjusted so that the bitrate of each RD point would not differ too much from four predefined bitrates. These bitrates are 6Mbps, 4Mbps, 2.5Mbps and 1 Mbps for RD points 1, 2, 3, and 4 respectively. Table 5.4 to Table 5.6 present the QP values used for the test for all the solutions to be evaluated (proposed solution and benchmarks). All CM and OCM projected videos were encoded with two slices, one including three cube faces on top of the video (top, back, and bottom) and the other including the three bottom cube faces of the video (left, front, and right).

Table 5.4 – QP values used for fixed viewport evaluation. Sequence: ChairliftRide.

RD point	Mono Equi	6×4-Qual		6×4-SRes	OMAF-Qual		OMAF-SRes	Mono CM	OCM
	QP	HQ QP	LQ QP	QP	HQ QP	LQ QP	QP	QP	QP
1	34	31	36	30	32	37	30	32	27
2	37	33	38	31	35	40	31	35	28
3	40	37	42	32	38	43	32	38	29
4	46	42	47	33	45	51	33	44	31

Table 5.5 – QP values used for fixed viewport evaluation. Sequence: SkateboardInLot.

RD point	Mono Equi	6×4-Qual		6×4-SRes	OMAF-Qual		OMAF-SRes	Mono CM	OCM
	QP	HQ QP	LQ QP	QP	HQ QP	LQ QP	QP	QP	QP
1	36	33	38	32	33	38	32	35	30
2	38	36	41	33	36	41	33	37	32
3	41	40	45	33	40	45	33	41	32
4	51	46	51	36	46	51	36	50	34

Table 5.6 – QP values used for fixed viewport evaluation. Sequence: KiteFlite.

RD point	Mono Equi	6×4-Qual		6×4-SRes	OMAF-Qual		OMAF-SRes	Mono CM	OCM
	QP	HQ QP	LQ QP	QP	HQ QP	LQ QP	QP	QP	QP
1	37	34	39	33	34	39	33	36	32
2	40	37	42	34	37	42	34	39	34
3	43	41	46	34	41	46	35	42	33
4	51	46	51	36	46	51	36	50	36

The viewport dimensions considered were 2000×2000 pixels, with 96 degrees horizontal and vertical FoV. The viewports are rendered using the rectilinear projection, meaning that the projection center is located at the center of the omnidirectional video sphere. The viewport pixel values were obtained by using bicubic interpolation.

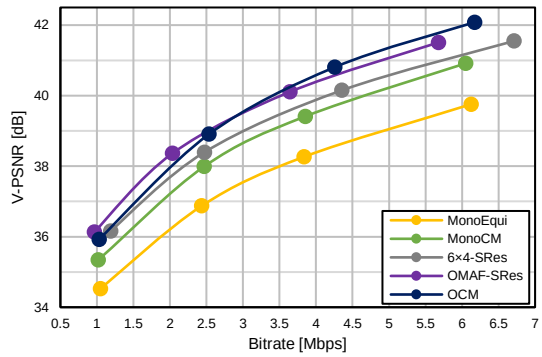
5.3.2 Fixed Viewport Evaluation: Ideal Conditions

The goal of the test was to provide the first evaluation to the several strategies in an ideal scenario, where the viewport location does not change and receives the best quality representation. This is an ideal case, since there is no mismatch between the video representation received and the viewport location.

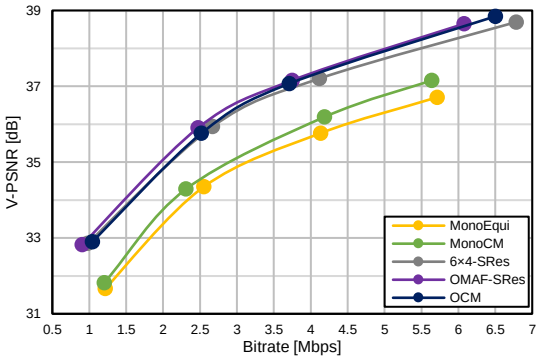
For each strategy, viewports were rendered for all 300 frames and for five static viewing directions (latitude, longitude): (0°, 0°), (60°, 0°), (-60°, 0°), (0°, 90°), and (0°, -90°). An ideal streaming scenario was considered, i.e., for tiles-based strategies only tiles covering the rendered viewport are HQ/HSR and for OCM streaming, the offset orientation matches the rendered viewport center. Each rendered viewport was compared to the same viewport rendered from the original video using the PSNR metric, this means computing the V-PSNR for each frame, followed by averaging the V-PSNR for the 300 frames of each sequence. Note that the LQ/LSR tiles are not considered in the quality evaluation but account for rate. The Bjøntegaard Delta PSNR (BD-PSNR) was computed for each viewport direction, strategy and sequence always using the MonoEqui as reference.

For each tiles strategy, a partial delivery streaming strategy was also considered. For these partial delivery streaming strategies, LQ/LSR tiles are not transmitted, as only the tiles that cover viewport regions are transmitted and have HQ/HSR. Partial delivery tiles strategies provide an upper bound performance by only transmitting the viewport but note that in real scenarios cannot be applied due to head motion and network delays in obtaining high quality tiles for the viewport area.

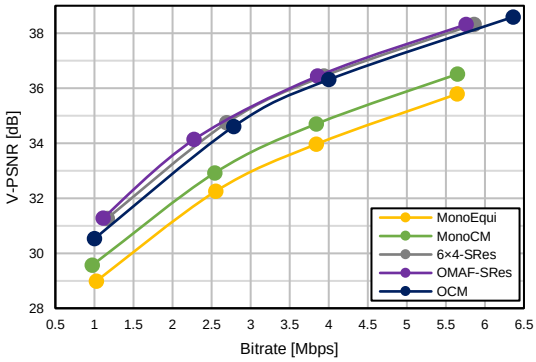
The rate-distortion curves for strategies MonoEqui, MonoCM, 6×4-SRes, OMAF-SRes, and OCM, for directions (0°, 0°), (0°, 90°), and (60°, 0°) for sequences SkateboardInLot, and KiteFlite, and for directions (0°, 0°) and (60°, 0°) for sequence ChairliftRide, are shown in Figure 5.5, where each point corresponds to a RD point. 6×4-Qual and OMAF-Qual obtained worse performance than 6×4-SRes and OMAF-SRes and, as such, rate-distortion curves for strategies 6×4-Qual and OMAF-Qual are not shown in Figure 5.5. Table 5.7 presents the average BD-PSNR values for directions (0°, 0°), (0°, 90°), and (0°, -90°) which covers regions of the omnidirectional video with latitude zero and thus more perceptually important (head directions more frequently assessed). Table 5.8 presents the average of BD-PSNR values for the other directions (60°, 0°) and (-60°, 0°), respectively. The average of BD-PSNR values for all directions are shown in Table 5.9.



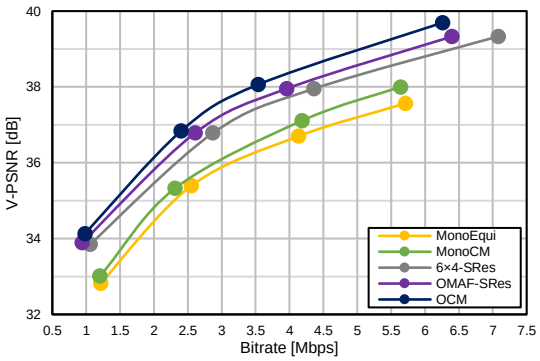
a) Direction (0°,0°). Sequence: ChairliftRide.



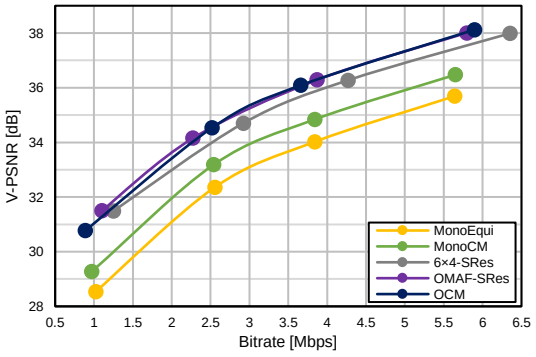
b) Direction (0°,0°). Sequence: SkateboardInLot.



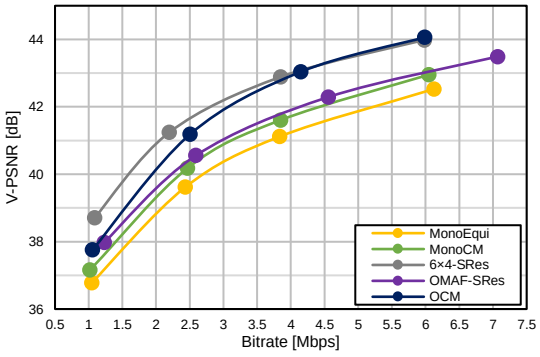
c) Direction (0°,0°). Sequence: KiteFlite.



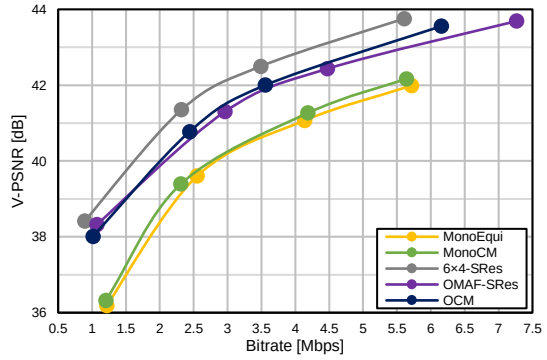
d) Direction (0°,90°). Sequence: SkateboardInLot.



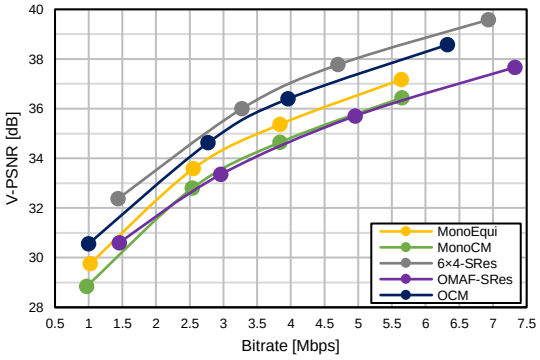
e) Direction (0°,90°). Sequence: KiteFlite.



f) Direction (60°,0°). Sequence: ChairliftRide.



g) Direction (60°,0°). Sequence: SkateboardInLot.



h) Direction (60°,0°). Sequence: KiteFlite.

Figure 5.5 – RD Performance for several streaming strategies.

Table 5.7 – Average of BD-PSNR for directions (0°,0°), (0°,-90°) and (0,90°). Dark green, green, and light green cells represent the best, second best, and third best BD-PSNR values (excluding partial delivery strategies), respectively.

Strategy		ChairliftRide	SkateboardInLot	KiteFlite
Partial Delivery	6×4-Qual	3.00	2.42	3.99
	6×4-SRes	3.13	2.55	4.05
	OMAF-Qual	2.95	2.40	3.69
	OMAF-SRes	3.13	2.55	3.78
6×4-Qual		0.84	0.59	1.26
6×4-SRes		1.38	1.31	1.96
OMAF-Qual		0.85	0.71	1.29
OMAF-SRes		1.91	1.56	2.27
MonoCM		0.94	0.28	0.83
OCM		2.00	1.71	2.08

Table 5.8 – Average of BD-PSNR for directions (60°,0°) and (-60°,0°). Dark green, green, and light green cells represent the best, second best, and third best BD-PSNR values (excluding partial delivery strategies), respectively.

Strategy		ChairliftRide	SkateboardInLot	KiteFlite
Partial Delivery	6×4-Qual	2.94	3.07	3.86
	6×4-SRes	3.13	3.18	3.92
	OMAF-Qual	1.05	1.15	1.39
	OMAF-SRes	0.97	1.43	0.67
6×4-Qual		0.77	0.78	1.20
6×4-Sres		1.38	1.62	1.87
OMAF-Qual		0.23	0.38	0.56
OMAF-SRes		0.50	1.06	0.14
MonoCM		0.55	0.12	-0.25
OCM		1.37	1.09	1.42

Table 5.9 – Average of BD-PSNR for all five directions. Dark green, green, and light green cells represent the best, second best, and third best BD-PSNR values (excluding partial delivery strategies), respectively.

Strategy		ChairliftRide	SkateboardInLot	KiteFlite
Partial Delivery	6×4-Qual	2.98	2.68	3.94
	6×4-SRes	3.13	2.80	4.00
	OMAF-Qual	2.19	1.90	2.77
	OMAF-SRes	2.27	2.10	2.53
6×4-Qual		0.81	0.66	1.24
6×4-SRes		1.38	1.43	1.92
OMAF-Qual		0.60	0.58	1.00
OMAF-SRes		1.34	1.36	1.42
MonoCM		0.78	0.22	0.40
OCM		1.75	1.46	1.82

From the BD-PSNR results, it is possible to observe that partial delivery tiles strategies obtain the best quality for the same bitrate. This was expected and the gap between the best non-partial delivery

and the best partial delivery streaming strategies is rather high which shows that there is still room for improvement.

From Table 5.9 it is possible to conclude that the overall best performing tiles-based strategies are 6×4-SRes and OMAF-SRes. However, the performance for strategies 6×4-SRes and OMAF-SRes depend of the latitude of the viewport center:

- **For directions with latitude zero:** OMAF-SRes performs better than 6×4-SRes. This behavior can be observed in Figure 5.5a) to e) and Table 5.7, with OMAF-SRes providing 0.25dB to 0.53dB gains compared to 6×4-SRes. For viewport directions (0°, 0°), (0°, 90°) and (0°, -90°), the HSR tiles for 6×4 and OMAF layouts cover 33.3% of the equirectangular projected omnidirectional frame. The 6×4 layout results in more tiles than OMAF layout, as can be seen comparing Figure 5.2 and Figure 5.3b), meaning that spatial redundancy is less exploited for 6×4-SRes compared to OMAF-SRes, increasing the bitrate for 6×4-SRes.
- **For directions with latitude different than zero:** 6×4-SRes performs better than OMAF-SRes. This behavior can be observed in Figure 5.5f) to h) and Table 5.8, with 6×4-SRes providing 0.56dB to 1.73dB gains compared to OMAF-SRes. For viewport directions (60°, 0°) and (-60°, 0°), the region covered by the HSR tiles is larger for the OMAF layout (covering 66.7% of the equirectangular frame) than for the 6×4 layout (covering 41.6% of the equirectangular frame). This leads to the OMAF-SRes requiring more bitrate for the same viewport representation. Furthermore, when regions near the poles have high frequency content, the lower spatial resolution of OMAF-SRes pole regions decreases the detail of these regions, compared to the same regions for 6×4-SRes, thus decreasing quality for viewports covering these regions using OMAF-SRes. This can be observed on Figure 5.5h), for sequence KiteFlite and direction (60°, 0°), which covers leaves motion with the wind.

6×4-Qual and OMAF-Qual strategies present similar behavior with latitude compared to 6×4-SRes and OMAF-SRes, respectively, as shown in Table 5.7 and Table 5.8, due to the similar tile layout. However, 6×4-Qual and OMAF-Qual perform worse than tiles spatial resolution strategies, showing tiles with different spatial resolutions provide better performance than tiles with different encoding qualities.

The OCM projection strategy provides the best or second-best overall results, according to the results in Table 5.9. These results can be further investigated with the behavior to latitude:

- **For directions with latitude zero:** As it can be observed in Table 5.7, the OCM projection results are rather close to those obtained by OMAF-SRes, with gains up to 0.15dB and losses up to 0.19dB. Moreover, OCM projection results provided gains between 0.12dB and 0.62dB compared to 6×4-Sres.
- **For directions with latitude different than zero:** As it can be observed in Table 5.8, OCM results are second best to 6×4-SRes, with losses up to 0.53dB. However, OCM projection leads to gains up to 1.28dB compared to OMAF-SRes.

These results demonstrate that OCM projection not only provides similar quality of the OMAF-SRes for latitudes close to zero, which are the most viewed regions, but also provides reduced quality loss compared to 6×4-SRes for latitudes closer to the poles.

5.3.3 Fixed Viewport Evaluation with Mismatch

The goal of this experiment was to determine the degradation of the viewport quality when the rendered viewport does not match the viewport requested. This may occur during typical streaming situations where the network delay doesn't allow to quickly receive the areas covered by the viewport with high quality and when the head motion prediction algorithm fails to predict the new regions of the viewport. To examine the viewport quality degradation, the viewport center requested was fixated at (0°, 0°). Next, 36 viewport directions uniformly distributed along latitude zero degrees (which are most frequently requested by streaming clients in a realistic scenario) were selected for viewport extraction. For each evaluated sequence and viewport direction, viewports were rendered from the first frame of each second, totalizing 10 frames per sequence, in order to keep computational costs low while still being representative. Each rendered viewport (after decoding) was compared to the same viewport rendered from the original video by computing the V-PSNR for each frame, followed by averaging the V-PSNR for the 10 frames of each sequence. Lastly, for each viewport direction, the resulting V-PSNR for MonoEqui was subtracted to the V-PSNR obtained for all strategies, obtaining the relative V-PSNR gain to MonoEqui, ΔV-PSNR. Figure 5.6, Figure 5.7, and Figure 5.8 present the ΔV-PSNR evolution for all streaming strategies evaluated when the longitude of the viewport center varies along the sphere. The bitrates are shown on the legend of all Figures, and results were obtained for sequences ChairliftRide, SkateboardInLot, and KiteFlite, respectively. Naturally, the ΔV-PSNR for MonoEqui is equal to zero.

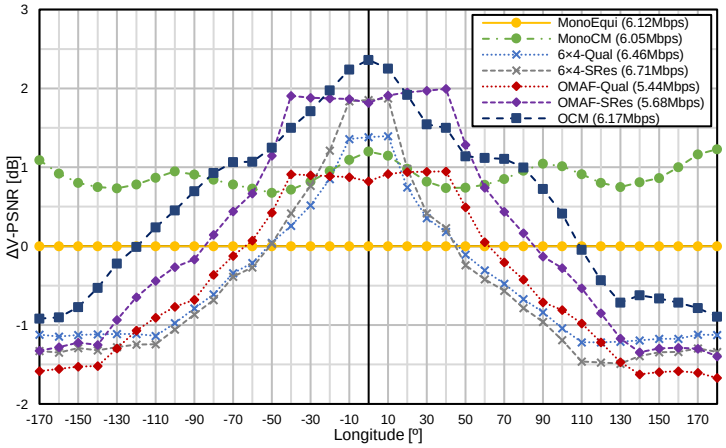


Figure 5.6 – V-PSNR relative gain to MonoEqui. Sequence: ChairliftRide.

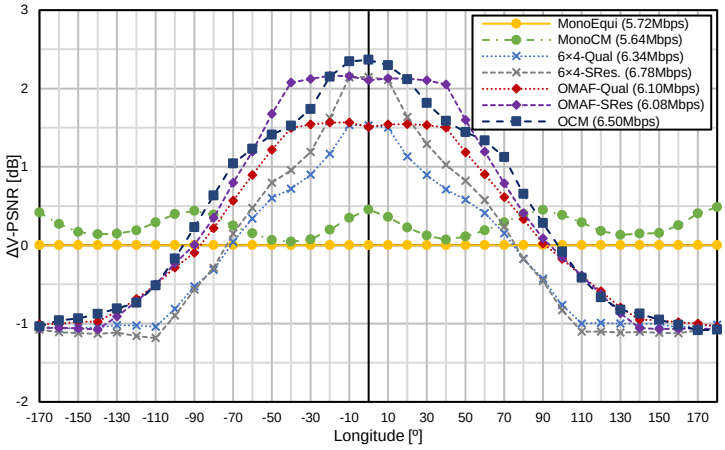


Figure 5.7 – V-PSNR relative gain to MonoEqui. Sequence: SkateboardInLot.

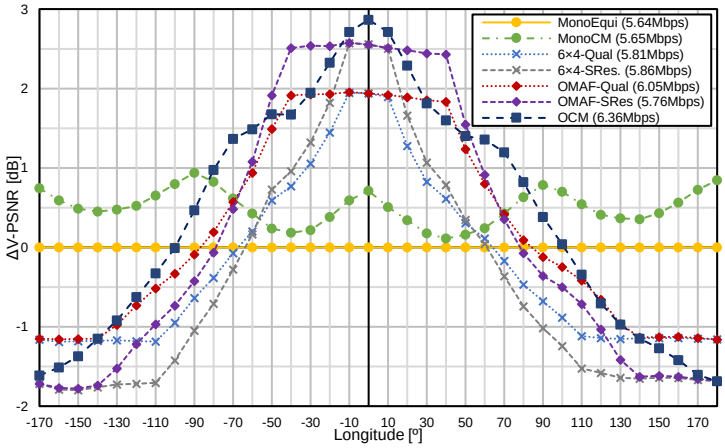


Figure 5.8 – V-PSNR relative gain to MonoEqui. Sequence: KiteFlite.

From Figure 5.6 to Figure 5.8 it is possible to identify three gain regions for tiles-based strategies: 1) approximately constant gain around longitudes close to 0°, 2) approximately constant loss around longitudes close to 180°, and 3) dropping gain between longitudes close to 0° and longitudes close to 180°. Analyzing tiles-based strategies for approximately constant ΔV-PSNR longitudes:

- **For longitudes close to 0°:** Among the tiles-based strategies, OMAF-SRes provided the highest gain for a larger longitude interval compared to other tiles-based strategies although it is not the best for longitudes close to zero. As can be seen from the results in Figure 5.6 to Figure 5.8, OMAF-SRes achieved higher gains than 6×4-Qual and OMAF-Qual, and the same gain as 6×4-SRes. However, for OMAF-SRes the region with the highest gain is 60° wider compared to 6×4-SRes. This is due to the OMAF-SRes HSR tiles covering in total 180° horizontal FoV, while 6×4-SRes HSR tiles only cover 120° horizontal FoV total.
- **For longitudes close to 180°:** Results for tiles-based strategies depend on the characteristics of the sequence. For example, results for sequence SkateboardInLot, presented in Figure 5.6, show similar results for all tiles strategies, while results for sequence KiteFlite, presented in Figure 5.8, depict losses for tiles spatial resolution strategies of approximately 0.6dB compared to tiles quality strategies.

Depending on the distance to the viewport center requested, the OCM strategy has a different performance behavior compared to OMAF-SRes. As shown in Figure 5.6 to Figure 5.8, the OCM

provides higher quality than OMAF-SRes for up to a longitude distance to the viewport requested between 10° and 20°, with gains up to 0.54dB for sequence ChairliftRide. Thus, for a requested viewport with low distance error, OCM provides the best quality. Between 20° and 50° longitude distance to the viewport requested, the OCM provides lower quality than OMAF-SRes, with losses up to 0.84dB for sequence KiteFlite. However, the OCM strategy provides a smoother quality decrease with respect to the angular distance to the viewport requested center, while OMAF-SRes quality drop is quicker for angular distances above 50° due to the difference in quality on a viewport when it covers both HQ/HSR and LQ/LSR tiles. This difference in quality is often perceptible, creating an unnatural border on the observed viewport, which may decrease the overall QoE. To illustrate these differences, Figure 5.9 presents sections of three viewports rendered using OMAF-SRes and OCM projection. For the OMAF-SRes it is possible to notice the difference in quality between HQ and LQ tiles, in Figure 5.9a) and b), on the back of the car, in Figure 5.9c) and d), by the difference in blur of the bushes, and in Figure 5.9e) and f), by the discontinuity on the floor and on the base of the publicity stand.. Note that these differences are more perceptual visible in an HMD where the display is close to the user's eyes. In contrast, OCM projection presents a smoother loss of quality due to the proposed projection algorithm. As such, visual quality decay is usually not as noticeable as for tiles strategies.

a) OMAF-SRes. Sequence: SkateboardInLot.

b) OCM Projection. Sequence: SkateboardInLot.

c) OMAF-SRes. Sequence: ChairliftRide.

d) OCM Projection. Sequence: ChairliftRide.

e) OMAF-SRes. Sequence: KiteFlite.

f) OCM Projection. Sequence: KiteFlite.

Figure 5.9 – Viewport section examples for viewport center (0°, 50°).

5.4 Trajectory Based Viewport Evaluation

This section evaluates the proposed OCM solution and the benchmark streaming strategies considering viewports following real trajectories. First, the test conditions are described. Then, an evaluation of the rate-distortion obtained for viewports extracted using selected trajectories obtained from users that were interacting with the content in the past.

5.4.1 Test Conditions

The test video sequences were selected from the Salient360 dataset [46], since no recorded head movements were available for the sequences used in section 5.3. The videos of Salient360 dataset are available encoded with H.264/AVC always using the same video configuration. From the Salient360 dataset, three videos were used: Turtle, UnderwaterPark and Touvet. These sequences were selected considering three conditions: 1) frame rate equal to the sequences used for fixed viewport evaluation (30 frames/s); 2) low amount of stitching artifacts; 3) different bitrate levels of the available encoded videos – Turtle with medium bitrate, UnderwaterPark with higher bitrate, and Touvet with lower bitrate. Figure 5.10 presents the first frame of Turtle, UnderwaterPark and Touvet. All videos are in the equirectangular format, with spatial resolution of 3840×1920, sampling rate of 30 frames per second and have a duration of 20 seconds.

a) Turtle *b) UnderwaterPark* *c) Touvet*

Figure 5.10 – First frame of omnidirectional video test sequences.

The spatial resolutions of the omnidirectional video for each streaming strategy are the following: 3840×1920 for MonoEqui, 6×4-Qual, and OMAF-Qual, and 2880×1920 for MonoCM. Table 5.10 shows the spatial resolution for 6×4-SRes, while the spatial resolution for OMAF-SRes are shown in Table 5.11.

Table 5.10 – Spatial resolutions of the omnidirectional video for 6×4-SRes.

RD point	HSR	LSR
1	3840×1920	1920×960
2	3072×1536	1536×768
3	2304×1152	1152×576
4	1536×768	768×384

Table 5.11 – Spatial resolutions of the omnidirectional video for OMAF-SRes.

RD point	HSR Equator	LSR Equator / HSR Pole	LSR Pole
1	3840×1920	1920×960	960×480
2	3072×1536	1536×768	768×384
3	2304×1152	1152×576	576×288
4	1536×768	768×384	384×192

For the proposed OCM projection the spatial resolutions are shown in Table 5.12. In this case, 13 offset orientations $(\theta_{offset}, \phi_{offset})$ were considered and are presented in Table 5.13. These offset orientations were based on the 22 offset orientations used by Oculus, listed in Table 4.3, by covering a similar area of the omnidirectional frame with OCM front faces, represented in Figure 4.7. Figure 5.11 presents the regions of an omnidirectional video that are projected on the cube front face, for all 13 offset orientations considered.

Table 5.12 – Spatial resolutions of the omnidirectional video used in OCM projection.

RD point	ERP equivalent resolution	OCM resolution	OCM face size
1	3840×1920	1728×1152	576
2	3072×1536	1344×896	448
3	2304×1152	1152×768	384
4	1536×768	768×512	256

Table 5.13 – OCM projection offset orientations.

θ_{offset} (in degrees)	ϕ_{offset} (in degrees)
55	26, 146, 266
0	0, 51, 103, 154, 206, 257, 309
-55	26, 146, 266

Figure 5.11 – Representation on an equirectangular image of the front faces of all 13 OCMs with the offset orientations considered for the OCM streaming strategy.

The videos were encoded with the HEVC reference software, HEVC Test Model (HM) version 16.20, using the Random Access configuration with GOP size of 16 frames. All 600 frames of each video were encoded. Considering the GOP size, two different temporal segmentation intervals were selected to each streaming strategy: 32 frames (approximately one second and referred as 1s) and 64 frames (approximately two seconds and referred as 2s). The QP values were chosen based on the QP values selected in section 5.3.1 to provide RD points between similar bitrate intervals for each sequence. Table 5.14, Table 5.15, and Table 5.16 present the QP values used for the test for all the strategies to be evaluated (proposed solution and benchmarks). All CM and OCM projected videos were encoded with two slices, one including three cube faces on top of the video (top, back, and bottom) and the other including three bottom cube faces of the video (left, front, and right).

Table 5.14 – QP values used for trajectory based viewport evaluation. Sequence: Turtle.

RD point	Mono Equi	6×4-Qual		6×4-SRes	OMAF-Qual		OMAF-SRes	Mono CM	OCM
	QP	HQ QP	LQ QP	QP	HQ QP	LQ QP	QP	QP	QP
1	34	32	37	32	32	37	32	34	28
2	40	38	43	33	37	42	33	40	29
3	45	41	46	34	40	45	34	45	30
4	51	43	48	35	42	47	35	50	32

Table 5.15 – QP values used for trajectory based viewport evaluation. Sequence: UnderwaterPark.

RD point	Mono Equi	6×4-Qual		6×4-SRes	OMAF-Qual		OMAF-SRes	Mono CM	OCM
	QP	HQ QP	LQ QP	QP	HQ QP	LQ QP	QP	QP	QP
1	34	32	37	32	32	37	30	33	28
2	40	38	43	33	37	42	33	40	29
3	45	41	46	34	40	45	34	45	30
4	51	43	48	35	42	47	35	50	32

Table 5.16 – QP values used for trajectory based viewport evaluation. Sequence: Touvet.

RD point	Mono Equi	6×4-Qual		6×4-SRes	OMAF-Qual		OMAF-SRes	Mono CM	OCM
	QP	HQ QP	LQ QP	QP	HQ QP	LQ QP	QP	QP	QP
1	34	32	37	32	32	37	30	33	30
2	40	38	43	33	37	42	33	40	33
3	45	41	46	34	40	45	34	45	34
4	51	43	48	35	42	47	35	50	35

The viewport dimensions considered were 2000×2000 pixels, with 96 degrees horizontal and vertical FoV. The viewports are rendered using the rectilinear projection. The viewport pixel values were obtained with bicubic interpolation.

5.4.2 Results and Analysis

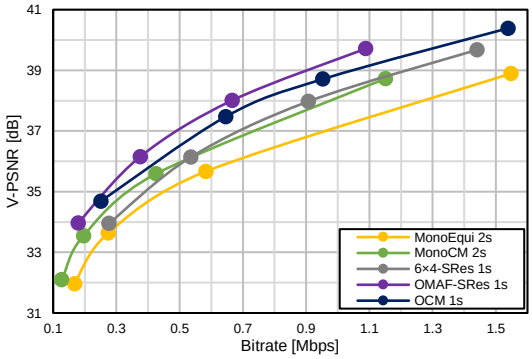
The goal of the test was to evaluate the different streaming strategies using head motions obtained in a realistic scenario. The head motions used are also available in the Salient360 dataset [46]. This dataset contains head motion data obtained from 57 users observing each sequence for its entire duration with an HMD. Each head motion trajectory contains 100 samples indicating the longitude, latitude and fixation timestamp. For each sequence, eight head motion trajectories divided in two types were considered:

- Maximum saliency (MSal): four trajectories were selected based on maximum saliency, this means cover the more commonly watched regions of the video.
- High distance (HDist): another four trajectories were chosen based on maximum distance covered, which cover faster head movements.

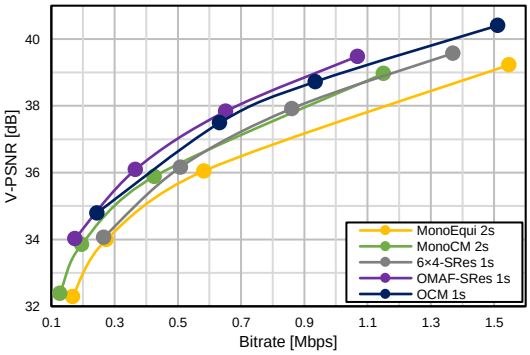
The streaming scenario considered requested the next segment higher quality/spatial resolution region (e.g. tiles) based on the coordinates of the trajectory sample that immediately precedes it. These coordinates are then used as the requested viewport center, for tiles-based strategies, or for determining the closest offset orientation, for OCM streaming. The viewports were rendered for all 600 frames using the trajectories coordinates as viewport center. Each viewport was compared to the same viewport

rendered from the original video using the PSNR metric, this means computing the V-PSNR for each frame, followed by averaging the V-PSNR for the 600 frames of each sequence. Then, both rate and PSNR for each strategy were averaged for the MSal and HDist trajectories. The BD-PSNR was computed for each trajectory type, streaming strategy, temporal segmentation and sequence always using the MonoEqui 2s as reference.

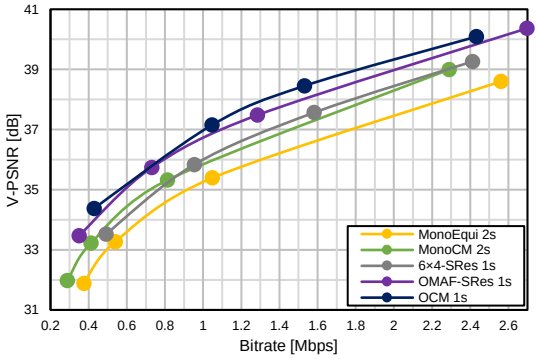
The RD curves for strategies MonoEqui, MonoCM, 6×4-SRes, OMAF-SRes, and OCM, and for each trajectory type and sequence are shown in Figure 5.12, where each point corresponds to the average of RD points obtained for four trajectories. MonoEqui and MonoCM are only presented for 2s segment duration, since longer segments provide better results for monolithic strategies. 6×4-SRes, OMAF-SRes, and OCM are only presented for 1s segmentation, as 1s provided better overall results than 2s for these streaming strategies. 6×4-Qual and OMAF-Qual are not included since they perform worse than 6×4-SRes and OMAF-SRes.



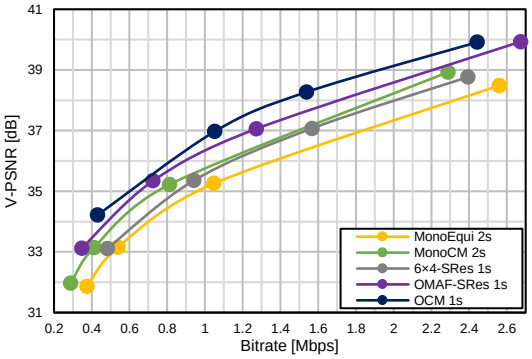
a) MSal trajectories. Sequence: Turtle.



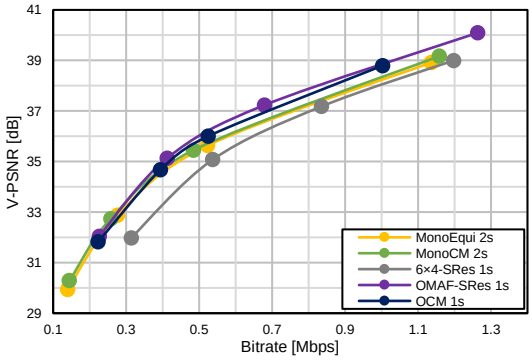
b) HDist trajectories. Sequence: Turtle.



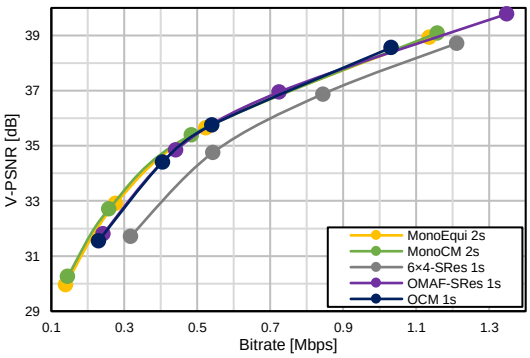
c) MSal trajectories. Sequence: UnderwaterPark.



d) HDist trajectories. Sequence: UnderwaterPark.



e) MSal trajectories. Sequence: Touvet



f) HDist trajectories. Sequence: Touvet

Figure 5.12 – RD Performance for several streaming strategies.

BD-PSNR values for all strategies, temporal segmentations, sequences, and trajectory type are presented in Table 5.17. Naturally, the BD-PSNR for MonoEqui 2s is zero since it is the reference.

Table 5.17 – BD-PSNR for the different trajectory types and streaming strategies compared to MonoEqui 2s. Dark green, green, and light green cells represent the best, second best, and third best BD-PSNR values, respectively.

Strategy	Segment duration	Turtle		UnderwaterPark		Touvet	
		MSal	HDist	MSal	HDist	MSal	HDist
MonoEqui	1s	-0.20	-0.19	-0.23	-0.23	-0.33	-0.33
	2s	0.00	0.00	0.00	0.00	0.00	0.00
MonoCM	1s	0.64	0.57	0.57	0.60	-0.17	-0.24
	2s	0.85	0.76	0.79	0.82	0.16	0.10
6×4-Qual	1s	0.44	0.25	0.21	-0.06	-0.68	-0.91
	2s	0.57	0.11	0.26	-0.21	-0.57	-0.84
6×4-SRes	1s	0.78	0.56	0.76	0.45	-0.68	-1.05
	2s	0.85	0.30	0.78	0.22	-0.81	-1.25
OMAF-Qual	1s	0.98	0.63	0.32	0.15	0.09	-0.26
	2s	1.14	0.58	0.44	0.08	0.20	-0.19
OMAF-SRes	1s	1.80	1.41	1.51	1.25	0.46	-0.13
	2s	1.90	1.26	1.60	1.11	0.30	-0.35
OCM	1s	1.48	1.21	1.78	1.71	0.26	-0.13
	2s	1.61	1.07	1.90	1.66	0.24	-0.27

The different segment duration leads to different results for monolithic, for tiles strategies and for OCM. The following conclusions regarding the *segment duration* can be taken:

- **MonoEqui and MonoCM streaming strategies:** 2s segments performs better than 1s, as shown in Table 5.17. Longer segments allow the encoder to exploit inter frame prediction and doesn't require so many Intra frames which are costly in terms of bitrate, resulting in lower bitrates for the same quality.
- **Tiles and OCM streaming strategies:** Table 5.18 presents the difference between the BD-PSNR shown in Table 5.17 for 1s and 2s segments for tiles and OCM streaming strategies; a positive difference means a better performance for 1s segments and a negative difference means a better performance of 2s segments. As shown in Table 5.18, there are many cases where the BD-PSNR of 2s segments compared to 1s is worst for all tiles strategies and OCM. To understand this behavior a more detailed analysis is made. Consider that the viewport average error, Er, is equal to the average angular distance between the centers of the current and the requested viewport (requested viewport center coordinates for tiles strategies and front face center coordinates for OCM Projection). Table 5.19 shows the viewport average error for 1s and 2s segments, Er_{1s} and Er_{2s}, respectively, and the difference between 2s and 1s viewport average error, $Er_{2s} - Er_{1s}$; the higher the difference, the lower the accuracy of the requested viewports for 2s segments compared to 1s and, thus, lower viewport quality for 2s. As shown in Table 5.19, MSal trajectories for Turtle and UnderwaterPark provide the lowest difference between 2s and 1s viewport average error, meaning that is not required correction of the

requested viewport center more frequently (i.e. with smallest 1s segments) to have a viewport with high quality, resulting in better performance for 2s segments for MSal trajectories for Turtle and UnderwaterPark as shown in Table 5.18. The remaining trajectories have higher difference between 2s and 1s viewport average error and, as shown in Table 5.18, for HDist trajectories for Turtle and UnderwaterPark and all trajectories for Touvet, 1s segments perform better than 2s segments for all tiles strategies and OCM, except for 6×4-Qual and OMAF-Qual for Touvet.

Table 5.18 – BD-PSNR difference between 1s and 2s for tiles-based and OCM streaming strategies.

Strategy	Turtle		UnderwaterPark		Touvet	
	MSal	HDist	MSal	HDist	MSal	HDist
6×4-Qual	-0.13	0.14	-0.05	0.15	-0.11	-0.07
6×4-SRes	-0.07	0.26	-0.02	0.23	0.13	0.2
OMAF-Qual	-0.16	0.05	-0.12	0.07	-0.11	-0.07
OMAF-SRes	-0.1	0.15	-0.09	0.14	0.16	0.22
OCM	-0.13	0.14	-0.12	0.05	0.02	0.14

Table 5.19 – Viewport average error (in degrees).

Strategy	Error	Turtle		UnderwaterPark		Touvet	
		MSal	HDist	MSal	HDist	MSal	HDist
Tiles	Er_{1s}	8.4	27.8	16.3	35.0	22.6	34.1
	Er_{2s}	14.6	42.0	24.3	53.0	36.8	51.6
	$Er_{2s} - Er_{1s}$	6.2	14.2	8.0	18.0	14.2	17.5
OCM	Er_{1s}	17.4	33.2	28.8	40.2	29.0	39.3
	Er_{2s}	20.6	45.6	35.9	56.3	40.8	56.4
	$Er_{2s} - Er_{1s}$	3.2	12.4	7.1	16.1	11.8	17.1

From Table 5.17 it is possible to conclude that the overall best performing tile-based strategy is OMAF-SRes. This is expected since, as discussed in sections 5.3.2 and 5.3.3, among the tiles strategies, OMAF-SRes provided the best quality for regions near the equator, which are the most viewed regions, and better tolerance to error between the current viewport and the requested viewport. OMAF-SRes and OCM with 1s segmentation provided the best overall results. However, results for these two strategies depend on the sequence:

- **Sequence Turtle:** better results are achieved for OMAF-SRes. As shown for Table 5.17, OMAF-SRes 1s provides gains of 0.32dB for MSal and 0.2dB for HDist compared to OCM 1s. This can also be confirmed for the RD curves of Figure 5.12a) and b).
- **Sequence UnderwaterPark:** better results were obtained for the OCM Projection. As shown in Figure 5.12c) and d) and Table 5.17, OCM 1s provides gains of 0.27dB for MSal and 0.46dB for HDist compared to OMAF-SRes 1s.
- **Sequence Touvet:** close results between strategies were obtained. As shown in Figure 5.12e) and f) and Table 5.17, OMAF-SRes 1s provides gains of 0.20dB for MSal trajectories compared to OCM 1s. However, for HDist trajectories, OMAF-SRes and OCM strategies with 1s segments provide losses of 0.13dB to MonoEqui.

The difference in RD performance between OMAF-SRes and OCM can be explained with the OMAF-SRes behavior. Table 5.20 presents the following results obtained for OMAF-SRes 1s: 1) the average number of HSR tiles requested for each sequence and trajectory type; 2) the percentage of times that an HSR pole tiles is requested. As shown in Table 5.20, the average number of HSR tiles per frame is lower for Turtle and for Touvet MSal, resulting in better performance for OMAF-SRes compared to OCM. On the other hand, the average number of HSR tiles per frame is higher for UnderwaterPark and Touvet HDist. The number of frames with one HSR pole tile is also higher for UnderwaterPark and Touvet HDist and, as seen in subsection 5.3.2, viewports covering HSR pole tiles decrease OMAF-SRes performance. Furthermore, UnderwaterPark contains high frequency content in the pole regions, resulting in worst performance for OMAF-SRes compared to OCM. Results for Touvet HDist between OCM and OMAF-SRes are similar, even though both provided worse performance than MonoEqui and MonoCM.

Table 5.20 – OMAF-SRes 1s tile streaming analysis.

Sequence	Trajectory	Average # HSR tiles requested	Requested HSR pole tile (%)
Turtle	MSal	3.54	10.5
	HDist	3.57	15.8
Underwater Park	MSal	4.17	57.9
	HDist	4.07	42.1
Touvet	MSal	3.68	21.1
	HDist	4.45	46.1

Sequence Touvet provided the biggest difference in performance compared to the other sequences. As shown in Table 5.17, performances of all strategies compared to MonoEqui 2s are worse for Touvet than for Turtle and UnderwaterPark, with only MonoCM 2s providing better results than MonoEqui 2s for Touvet HDist trajectories. This behavior can be attributed to Touvet content, as it consists on a very low motion environment captured by a traveling camera moving towards direction (0°,0°). For MonoEqui, the encoding of Touvet can efficiently exploit temporal redundancy in the entire frame, as content moves from the center of the equirectangular projection to the sides, increasing encoding efficiency. For tiles-based strategies, by introducing spatial segmentation, motion vectors are restricted, decreasing encoding efficiency and reducing performance. For OCM, due to the distortion caused by mapping and the discontinuities between the borders of the three faces on the top and the three faces on the bottom of the OCM frame, temporal redundancy is somewhat reduced, also decreasing encoding efficiency and thus reducing overall performance. Moreover, as shown in Table 5.19, Touvet HDist trajectories have high viewport average error for segment durations 1s and 2s and for tiles-based and OCM strategies, further decreasing the performance of these strategies for Touvet HDist trajectories.

Chapter 6. Conclusions

6.1 Summary

In this thesis, the offset cubemap projection was optimized to provide an enhanced omnidirectional video streaming strategy, enabling high quality transmission through channels with limited and varying bandwidth. Developed by Oculus ®, the OCM projection allows higher spatial resolution (thus, higher quality) around a predefined direction of the omnidirectional video – the offset orientation – by distorting spherical angles. The distortion level is dictated by the offset magnitude: the higher the offset magnitude, the higher the distortion and the smaller the region around the offset orientation provided with higher resolution.

The Oculus ® implementation of the OCM streaming considers a fix offset magnitude of 0.7, which was proven (in this dissertation) to not provide a good viewport quality. To determine an appropriate offset value, for improved quality, 11 offset values were objectively evaluated using three omnidirectional videos, having distinct spatial-temporal activities. The best offset magnitude value was found to be 0.42, providing V-PSNR gains of up to 2.3 dB relatively to an offset magnitude of 0.7.

The assessment of the optimized OCM projection, for adaptive omnidirectional video streaming, was then conducted considering several state-of-the-art streaming strategies as benchmarks; these were of two types: 1) the monolithic or conventional strategies – which encode the entire omnidirectional frame with the same quality – and considering the two most common sphere-to-plan projections, namely equirectangular and cubemap; 2) tiles-based strategies, which use tiles encoded with different quality/spatial resolution to provide viewport adaptive streaming; in this case, tiles corresponding to the viewport regions are transmitted with the highest possible quality (allowed by the available bandwidth), while the others are transmitted with lower quality. The considered frame division in tiles was set according to the usual procedure in related literature: 6×4 and OMAF (Annex D.6.3) tiles structure. In an initial phase, the proposed OCM streaming strategy and benchmarks were tested considering static viewports (i.e., static viewing directions) to analyze the upper and lower bound of performance for each strategy. Results for static viewports showed the potential of OCM based video streaming, either when the requested (to the server) viewport center orientation matched the viewing direction, either when an error of up to 20° between these two directions was introduced. Finally, the considered streaming strategies were evaluated and compared using real head motion trajectories. For this test, the OCM strategy showed to provide better results than the monolithic strategies and to compete with the best tiles based solutions, although results are dependent on content and trajectory.

6.2 Future Work

The quality of experience provided to the user, with OCM based video streaming, can be improved by minimizing the distance between the requested offset direction and the actual user viewing direction; this could be achieved by improving the offset orientation encoding selection. While in this dissertation only 13 offset orientations were considered, the quality provided to the users may increase with a larger

number of offset values at the server, at the expense of an increase in the required storage space. Also, instead of using pre-defined offset angles, these could be obtained from visual attention maps, seeking to increase the quality of those parts of the omnidirectional video that are more likely to be observed by the users.

In the trajectory based evaluation of the streaming strategies, the requested offset direction (and also the requested high quality tiles, in the tiles based approaches) corresponds to the last known user's head orientation. This may lead to the request of an offset orientation too much far from the user viewing direction, reducing the viewport quality. To improve the performance of the OCM streaming (as well as of the tiles based strategies) user´s head motion prediction should be included at the client side.

For the quality evaluation of the different streaming strategies, the V-PSNR metric was used. This metric was adopted from the traditional PSNR metric, commonly used in 2D video and, as such, has some limitations for the evaluation of omnidirectional video. In particular, tiles based streaming produce noticeable quality differences when different quality/spatial resolution tiles are present in the viewport, notably at the tiles frontier, where spatial discontinuities may appear; this was clearly visible in the 2D computer monitor used for rendering the viewports and is expected to be even more visible (and annoying) if HMD displays are used, due to content magnification. The V-PSNR averages the error in each pixel and is not able to measure the visibility of the above-mentioned spatial discontinuity. To improve the evaluation and comparison of the considered omnidirectional video streaming strategies, a better objective metric should be applied (and, eventually, developed); additionally, subjective assessment tests, with users wearing HMD devices, should be also conducted.

Bibliography

[1] M. Wien, J. Boyce, T. Stockhammer and W.-H. Peng, "Standardization Status of Immersive Video Coding," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems,* vol. 9, no. 1, pp. 5-17, March 2019.

[2] ISO/IEC 23090-2:2019, Information technology – Coded representation of immersive media – Part 2: Omnidirectional media format.

[3] Google, "Google Cardboard," [Online]. Available: https://arvr.google.com/cardboard/. [Accessed 20 May 2020].

[4] Samsung, "Samsung Gear VR with Controller," [Online]. Available: https://www.samsung.com/global/galaxy/gear-vr/. [Accessed 2020 May 20].

[5] F. Jabar, J. Ascenso and M. Queluz, "Content-Aware Perspective Projection Optimization for Viewport Rendering of 360° Images," in *IEEE International Conference on Multimedia and Expo (ICME)*, Shanghai, China, 2019.

[6] ISO/IEC, Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats, 2014.

[7] X. Corbillon, A. Devlic, G. Simon and J. Chakareski, "Viewport-Adaptive Navigable 360-Degree Video Delivery," in *2017 IEEE International Conference on Communications (ICC)*, Paris, France, 2017.

[8] C. Zhou, Z. Li and Y. Liu, "A Measurement Study of Oculus 360 Degree Video Streaming," in *MMSys'17 Proceedings of the 8th ACM on Multimedia Systems Conference*, Taipei, Taiwan, 2017.

[9] M. Graf, C. Timmerer and C. Mueller, "Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP," in *MMSys'17 Proceedings of the 8th ACM on Multimedia Systems Conference*, Taipei, Taiwan, 2017.

[10] R. Ghaznavi-Youvalari et al., "Comparison of HEVC Coding Schemes For Tile-based Viewport-adaptive Streaming of Omnidirectional Video," in *2017 IEEE 19th International Workshop on Multimedia Signal Processing*, Luton, UK, 2017.

[11] A. T. Nasrabadi, A. Mahzari, J. D. Beshay and R. Prakash, "Adaptive 360-Degree Video Streaming using Scalable Video Coding," in *MM '17 Proceedings of the 25th ACM international conference on Multimedia*, Mountain View, California, USA, 2017.

[12] R. Schatz, A. Zabrovskiy and C. Timmerer, "Tile-based Streaming of 8K Omnidirectional Video: Subjective and Objective QoE Evaluation," in *Eleventh International Conference on Quality of Multimedia Experience*, Berlin, 2019.

[13] ISO/IEC 23009-1, Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats.

[14] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet," *IEEE MultiMedia,* vol. 18, no. 4, pp. 62-67, 2011.

[15] C. Timmerer and C. Griwodz, "Dynamic adaptive streaming over HTTP: From content creation to consumption," in *Proceedings of the 20th ACM international conference on Multimedia*, Nara, Japan, 2012.

[16] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP – Design Principles and Standards," in *Proceedings of the Second Annual ACM SIGMM Conference on Multimedia Systems, MMSys 2011*, Santa Clara, CA, USA, 2011.

[17] ISO/IEC 14496-10, Information technology – Coding of audio-visual objects – Part 10: Advanced Video Coding.

[18] ISO/IEC 23008-2, Information technology – High efficiency coding and media delivery in heterogeneous environments – Part 2: High efficiency video coding.

[19] "Understanding MPEG-DASH," 14 March 2014. [Online]. Available: https://www.hometoys.com/content.php?post_type=2239. [Accessed 19 March 2019].

[20] 3GPP TS 26.247, Transparent end-to-end Packet-switched Streaming Service (PSS); Progressive Download and Dynamic Adaptive Streaming over HTTP (3GP-DASH).

[21] O. A. Niamut et al., "MPEG DASH SRD - Spatial Relationship Description," in *MMSys '16: Proceedings of the 7th International Conference on Multimedia Systems*, Klagenfurt, Austria, 2016.

[22] R. v. Brandenburg, O. Niamut, M. Prins and H. Stokking, "Spatial Segmentation For Immersive Media Delivery," in *2011 15th International Conference on Intelligence in Next Generation Networks*, Berlin, Germany, 2011.

[23] L. D'Acunto, J. v. d. Berg, E. Thomas and O. Niamut, "Using MPEG DASH SRD for zoomable and navigable video," in *MMSys '16: Proceedings of the 7th International Conference on Multimedia Systems*, Klagenfurt, Austria, 2016.

[24] ISO/IEC 23090-2, Information technology – Coded representation of immersive media (MPEG-I) – Part 2: Omnidirectional media format.

[25] M. Graf, C. Timmerer and C. Mueller, "Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP," in *MMSys'17 Proceedings of the 8th ACM on Multimedia Systems Conference*, Taipei, Taiwan, 2017.

[26] A. T. Nasrabadi, A. Mahzari, J. D. Beshay and R. Prakash, "Adaptive 360-Degree Video Streaming using Scalable Video Coding," in *MM '17 Proceedings of the 25th ACM international conference on Multimedia*, Mountain View, California, USA, 2017.

[27] F. Lopes, J. Ascenso, A. Rodrigues and M. P. Queluz, "Subjective and Objective Quality Assessment of Omnidirectional Video," in *Applications of Digital Image Processing XLI*, San Diego, CA, USA, 2018.

[28] Y. Xu et al., "Omnidirectional Media Format and Its Application to Immersive Video Streaming: An Overview," 2018.

[29] S. Kühn, "File:Tissot world from space.png," Wikimedia Commons, 9 December 2004. [Online]. Available: https://commons.wikimedia.org/wiki/File:Tissot_world_from_space.png. [Accessed 3 May 2019].

[30] E. G. –. W. C. u. Sting, "File:Tissot indicatrix world map equirectangular proj.svg," Wikimedia Commons, June 2008. [Online]. Available: https://commons.wikimedia.org/wiki/File:Tissot_indicatrix_world_map_equirectangular_proj.svg. [Accessed 3 May 2019].

[31] J. Zeng et al., "A Tutorial on Image/Video Coding Standards," in *2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, Kaohsiung, Taiwan, 2013.

[32] G. J. Sullivan, J.-R. Ohm, W.-J. Han and T. Wiegand, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Trans. Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1649-1668, December 2012.

[33] F. Qian, L. Ji, B. Han and V. Gopalakrishnan, "Optimizing 360 video delivery over cellular networks," in *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*, New York City, New York, 2016.

[34] X. Corbillon, A. Devlic, G. Simon and J. Chakareski, "Optimal Set of 360-Degree Videos for Viewport-Adaptive Streaming," in *MM '17 Proceedings of the 25th ACM international conference on Multimedia*, Mountain View, California, USA, 2017.

[35] M. Yu and B. Girod, "A Framework to Evaluate Omnidirectional Video Coding Schemes," in *2015 IEEE International Symposium on Mixed and Augmented Reality*, Fukuoka, Japan, 2015.

[36] Y. Sun, A. Lu and L. Yu, "Weighted-to-Spherically-Uniform Quality Evaluation for Omnidirectional Video," *IEEE Signal Processing Letters,* vol. 24, no. 9, pp. 1408-1412, 2017.

[37] G. Bjøntegaard, "Calculation of average PSNR differences between RD-curves," document VCEG-M33, 2001, Austin, Texas, USA.

[38] P. Hanhart and T. Ebrahimi, "Calculation of average coding efficiency based on subjective quality scores," *J. Vis. Commun. Image Represent.,* vol. 25, no. 3, pp. 555-564, 2014.

[39] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transactions on Image Processing,* vol. 13, no. 4, pp. 600-612, 2004.

[40] H. Zhang, X. Ma, Y. Zhao and H. Yang, "Perceptual Quality Assessment Metric MS-SSIM," in *Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11*, Geneva, Switzerland, 2016.

[41] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy and M. Manohara, "Toward a practical perceptual video quality metric," Netflix TechBlog 6, Los Gatos, CA, USA, 2016.

[42] Wikipedia, "Video Multimethod Assessment Fusion - Wikipedia," Wikipedia, the free encyclopedia, 9 April 2019. [Online]. Available: https://en.wikipedia.org/wiki/Video_Multimethod_Assessment_Fusion. [Accessed 23 July 2019].

[43] E. Rakhmanov, E. Saff and Y. Zhou, "Electrons on the sphere," *Series in Approximations and Decompositions,* no. 5, pp. 293-310, 1994.

[44] J. Boyce, E. Alshina, A. Abbas and Y. Ye, "JVET common test conditions and evaluation procedures for 360° video," in *Joint Video Exploration Team (JVET) of ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11, JVET-G1030v2, 7th Meeting*, Torino, Italy, July, 2017.

[45] ITU-T Q.6/SG 16, ISO/IEC JTC 1/SC 29/WG 11, "High Efficiency Video Coding (HEVC)," Fraunhofer Heinrich Hertz Institute, [Online]. Available: https://hevc.hhi.fraunhofer.de. [Accessed 07 April 2020].

[46] E. J. David, J. Gutiérrez, A. Coutrot, M. P. D. Silva and P. L. Callet, "A Dataset of Head and Eye Movements for 360° Videos," in *Proceedings of the 9th ACM on Multimedia Systems Conference (MMSys'18)*, Amsterdam, Netherlands, Jun. 2018.