Enable the next generation of interactive video streaming

Xavier Corbillon

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THÈSE DE DOCTORAT DE

École nationale supérieure
Mines-Télécom Atlantique
Bretagne Pays de la Loire - IMT Atlantique
COMUE **U**NIVERSITE **B**RETAGNE **L**OIRE

Ecole Doctorale N° 601
Mathèmatique et Sciences et Technologies de l'Information et de la Communication (MathSTIC)
Spécialité : Informatique

Par

« Xavier CORBILLON »

« Enable the Next Generation Interactive Video Streaming »

«Rendre Possible la Transmission via l'Internet des Prochaines Générations de Vidéos Interactives»

Thèse présentée et soutenue à RENNES, le 30 octobre 2018
Unité de recherche : Institut de Recherche en Informatique et Systèmes Aléatoires (IRISA)
Thèse N° : 2018IMTA0103

Rapporteurs avant soutenance :

Vincent CHARVILLAT – Professor, ENSEEIHT Engineering School, University of Toulouse, France
Miska M. HANNUKSELA – Bell Labs Fellow, Nokia Technologies, Finland

Composition du jury :

Président :	Patrick LE CALLET – Professor, Polytech Nantes, University of Nantes, France
Examinateurs :	Vincent CHARVILLAT – Professor, ENSEEIHT Engineering School, University of Toulouse, France
	Miska M. HANNUKSELA – Bell Labs Fellow, Nokia Technologies, Finland
	Laura TONI – Lecturer (assistant professor), University College London, United-Kingdom
Dir. de thèse :	Gwendal SIMON – Professor, IMT Atlantique, IRISA, France

SUPERVISOR:
Gwendal Simon

LOCATION:
Rennes, France

DECLARATION

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements.

Xavier Corbillon
February 8, 2019

ACKNOWLEDGEMENT

First of all, I would like to thanks with all my heart my advisor, Prof. Gwendal Simon. I am thankful for his trust, his time, and his dedication. His valuable comments, ideas and feedbacks have considerably helped improving my thesis work and more generally my research activities. I will always remember the passionate discussions we had about either work or life. It was a great experience and a great honor to work with him.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Vincent Charvillat, Dr. Miska M. Hannuksela, Prof. Patrick Le Callet, and Dr. Laura Toni, for their perceptive comments and questions, and for their encouragement.

I am deeply thankful to Prof. Pascal Frossard who invited me to spend two months in the Signal Processing Laboratory (LTS4) in the École Polytechnique Fédérale de Lausanne (EPFL). It was a wonderful and rewarding experience.

Thanks a lot for Dr. Francesca De Simone, her directions and feedbacks were valuable. Your joy and commitment in your work was highly motivating. I am grateful for all I have learnt in two month with you.

My sincere thank to Dr. Alisa Devlic, Dr. Hristina Hristova, Mariem Ben Yahia, and Dr .Jacob Chakareski for the great collaborations. It was a great pleasure to work with wonderful people like you.

Then I would like to thanks my office colleagues Qipeng Song, François Lemercier, Saad El Jaouhari, and Hristina Hristova for the nice discussions about all and nothing. You were always there to produce productive critics and enlightening feedbacks. Tanguy Kerdoncuff, Baptiste Gaultier, Guillaume Habault and Benjamin Cama for the competitive "team building" moments 😉. Your warm welcome into the lab was highly appreciated.

Regarding the teaching and researcher life, I thank Christophe Couturier, Dr. Alberto Blanc, Dr. Jean-Pierre Le Narzul, Prof. Xavier Lagrange, and Dr. Géraldine Texier for their great and priceless advices.

Last but not the least, I would like to thank my friends for their support and the great moments we spent together. Above all, I thank my family for being always present and supporting despite the distance between us. Life did not allowed all of you to live through the end of this stage but you will always be a part of me ♥.

LIST OF PUBLICATIONS

INTERNATIONAL CONFERENCES

2018

- Hristina Hristova, Xavier Corbillon, Gwendal Simon, Viswanathan Swaminathan, and Alisa Devlic. "Heterogeneous Spatial Quality for Omnidirectional Video." In: *Proceeding of IEEE International Workshop on Multimedia Signal Processing (MMSP)*. 2018
- Xavier Corbillon, Francesca De Simone, Gwendal Simon, and Pascal Frossard. "Dynamic Adaptive Streaming for Multi-Viewpoint Omnidirectional Videos." In: *Proceedings of the 9th ACM Multimedia Systems (MMSys)*. 2018, pp. 237–249. DOI: 10.1145/3204949.3204968
- Jacob Chakareski, Ridvan Aksu, Xavier Corbillon, and Gwendal Simon. "Viewport-Driven Rate-Distortion Optimized 360-degree Video Streaming." In: *Proceedings of IEEE International Conference on Communications (ICC)*. 2018, pp. 1–7. DOI: 10.1109/ICC.2018.8422859

2017

- Xavier Corbillon, Alisa Devlic, Gwendal Simon, and Jacob Chakareski. "Optimal Set of 360-Degree Videos for Viewport-Adaptive Streaming." In: *Proceedings of ACM on Multimedia Conference, MM*. 2017, pp. 943–951. DOI: 10.1145/3123266.3123372
- Xavier Corbillon, Francesca De Simone, and Gwendal Simon. "360-Degree Video Head Movement Dataset." In: *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. 2017, pp. 199–204. DOI: 10.1145/3083187.3083215
- Xavier Corbillon, Alisa Devlic, Gwendal Simon, and Jacob Chakareski. "Viewport-Adaptive Navigable 360-Degree Video Delivery." In: *IEEE International Conference on Communications (ICC)*. 2017, pp. 1–7. DOI: 10.1109/ICC.2017.7996611

2016

- Xavier Corbillon, Florian Boyrivent, Gregoire Asselin De Williencourt, Gwendal Simon, Géraldine Texier, and Jacob Chakareski. "Efficient lightweight video packet filtering for large-scale video data delivery." In: *IEEE International Conference on Multimedia & Expo Workshops, ICME Workshops*. 2016, pp. 1–6. DOI: 10.1109/ICMEW.2016.7574700
- Xavier Corbillon, Ramon Aparicio-Pardo, Nicolas Kuhn, Géraldine Texier, and Gwendal Simon. "Cross-layer scheduler for video streaming over MPTCP." In: *Proceedings of the 7th International Conference on Multimedia Systems, MMSys 2016, Klagenfurt, Austria, May 10-13, 2016*. 2016, 7:1–7:12. DOI: 10.1145/2910017.2910594

2018

- Mariem Ben Yahia, Yannick Le Louedec, Gwendal Simon, Loutfi Nuaymi, and Xavier Corbillon. "HTTP/2-Based Frame Discarding for Low-Latency Adaptive Video Streaming." In: *IEEE Transactions on Multimedia Computing Communications and Applications* (2018). submitted

LIST OF AWARDS

- IEEE MMSP 2018 Best Paper Award — August 2018
  Best paper award obtained for the paper “Heterogeneous Spatial Quality for Omnidirectional Video”
- ACM MMSys 2018 Best Paper Award — June 2018
  Best paper award obtained for the paper “Dynamic Adaptive Streaming for Multi-Viewpoint Omnidirectional Videos”
- ICC 2017 Best Paper Award for CSSMA Symposium — May 2017
  Best paper award obtained for the paper “Viewport-adaptive navigable 360-degree video delivery”
- Best Poster Award — Feb 2017
  Awarded at the “Futur & Rupture” annual meeting by the academic members and the industrial partners.

ABSTRACT

Video streaming over the Internet consumes a larger ratio of the Internet bandwidth capacity, with a prediction that more than 80 % of Internet traffic will be video by 2020. Consumers are gradually deserting traditional broadband television to watch video stream over the Internet, requesting high quality contents. Consumer are now starting to request immersion inside the content they are watching.

With the decreasing price of Head-Mounted Display (HMD) and omnidirectional cameras, the popularity of omnidirectional video is increasing. Omnidirectional videos, also denoted as spherical videos or 360° videos, are videos with pixels recorded from a given viewpoint in every direction of space. A user watching such an omnidirectional content with a HMD can select the portion of the video to display by moving her head. The portion of the video displayed to the user is denoted as viewport and represents, with 2018's typical HMDs, around 20 % of the full omnidirectional content. To feel high immersion inside the content a user needs to see viewport with a very high resolution and a high frame rate (typically at least 4096×2048 pixels (4K) spatial resolution and 90 Hz frame rate). In order to get such viewport quality, streaming the whole video would require more than 100 Mbit s^{-1} bandwidth, which is much higher than the worldwide median Internet access connection speed. Applying traditional adaptive streaming technologies used by Over-The-Top (OTT) companies such as YouTube, Netflix, and Facebook would result, to match the available download bandwidth, in very low quality viewports.

In this dissertation I present my contributions to enable the streaming of highly immersive omnidirectional videos over the Internet. The contributions can be gathered into six contributions and into three main topics. First we propose new streaming architectures. Our goal is to stay as close as possible as existing HTTP Adaptive Streaming (HAS) architecture for obvious cost reduction reasons. We propose first a viewport-adaptive streaming architecture where omnidirectional video are encoded with a Quality Emphasized Region (QER) in order to stream video with high quality in the direction user is predicted to look in a few seconds while keeping traditional download throughput adaptation from HAS. Then we propose an extension to this streaming architecture to stream a next generation of omnidirectional video, denoted as Multi-ViewPoint (MVP) omnidirectional video, where users can not only perform rotational movement inside the content but also predefined translational movements. Secondly we perform theoretical studies. We study the relationship between the spherical pixel density and viewport distortion observed by users. We propose an extension to Facebook offset cube-map projection. We present a theoretical model to compute the optimal way to distribute the bit-rate inside an omnidirectional video, based on viewing statistics, to satisfy a majority of customer. Finally we propose practical tools to manipulate and study omnidirectional videos. First we developed a modular open-source C++ software, named *360Transformations*, to manipulate projected omnidirectional videos, extract viewports and compute objective quality metrics. Finally we recorded an openly available head-movement dataset of users watching in a free-of-task way omnidirectional videos.

This work resulted in nine publications including eight international conferences and one journal paper. Out of those nine publications, four are not discussed in this dissertation, for consistency purpose, as they are related to video frame filtering for the streaming of low latency videos or related to HTTP/2.0.

RÉSUMÉ

Chaque année, le streaming de vidéos sur Internet consomme une plus grande part de la bande passante totale de l'Internet, avec comme prévision que d'ici 2020 plus de 80 % du trafic Internet sera dédié à la vidéo. Les consommateurs désertent peu à peu la télévision traditionnelle pour regarder des vidéos sur Internet, en demandant toujours des contenus de plus hautes qualités. Les consommateurs commencent à réclamer de l'immersion à l'intérieur du contenu qu'ils regardent. Avec la baisse du prix des Casques de Réalité Virtuelle (CRV) et des caméras omnidirectionnelles, la popularité des vidéos omnidirectionnelles augmente. Les vidéos omnidirectionnelles, également appelées vidéos sphériques ou vidéos 360°, sont des vidéos avec des pixels enregistrés, à partir d'un seul point de vue, dans toutes les directions de l'espace. Un utilisateur qui regarde un tel contenu omnidirectionnel avec un CRV peut sélectionner la partie de la vidéo à afficher en bougeant la tête. La partie de la vidéo affichée à l'utilisateur est usuellement nommée viewport (fenêtre d'affichage en Français) et représente environ 20 % du contenu omnidirectionnel complet en considérant les caractéristiques des CRVs disponible en 2018. Pour se sentir totalement immergé à l'intérieur du contenu, l'utilisateur a besoin de voir un viewport avec une résolution spatiale de 4K et une fréquence d'images de 90 Hz. Pour obtenir une telle qualité d'affichage, le streaming de l'ensemble de la vidéo nécessiterait une bande passante de plus de 100 Mbit s^{-1}, ce qui est beaucoup plus élevé que la vitesse médiane d'accès à l'Internet dans le monde. L'utilisation des technologies traditionnelles de streaming adaptatif, utilisées par les sociétés dites Over-The-Top (OTT) telles que YouTube, Netflix et Facebook, rendrait possible le streaming de telles videos avec les bandes passantes actuellement disponibles mais au prix de viewport avec une très faible qualité.

Dans cette thèse, je présente mes contributions pour rendre possible le streaming de vidéos omnidirectionnelles hautement immersives sur l'Internet. On peut distinguer six contributions principales elles-mêmes regroupées en trois thèmes. Tout d'abord, nous proposons de nouvelles architectures de streaming. Notre objectif est de rester aussi proche que possible de l'architecture existante d'HTTP Adaptive Streaming (HAS) pour des raisons évidentes de réduction des coûts. Nous proposons d'abord une architecture de streaming omnidirectionnelle où la vidéo omnidirectionnelle est encodée avec une Région de Qualité plus Élevée (RQE) afin de diffuser la vidéo de haute qualité dans la direction où l'utilisateur est censé regarder dans quelques secondes tout en conservant l'adaptation au variation de débit du HAS classique. Nous proposons ensuite une extension de cette architecture de streaming pour diffuser une nouvelle génération de vidéo omnidirectionnelle, nommée vidéo omnidirectionnelle Multi-Points de Vue (MPV), où les utilisateurs peuvent non seulement effectuer des rotations à l'intérieur du contenu mais aussi effectuer des translations prédéfinies. Deuxièmement, nous avons réalisé des études théoriques. Nous avons étudié la relation entre la densité des pixels sur la sphère et la distorsion du viewport observée par l'utilisateur. Nous avons proposé une extension à la projection offset cube-map de Facebook. Nous présentons un modèle théorique pour calculer la manière optimale de distribuer le débit à l'intérieur d'une vidéo omnidirectionnelle, en utilisant des statistiques de visualisation, afin de satisfaire une majorité d'utilisateurs. Enfin, nous proposons des outils pratiques pour manipuler et étudier les vidéos omnidirectionnelles. Tout d'abord, nous avons développé en C++ un logiciel modulaire open-source, nommé *360Transformations*, pour manipuler les vidéos omnidirectionnelles projetées, extraire des viewports et calculer des métriques de qualité objective. Enfin, nous avons enregistré des traces de navigation de mouvements de têtes d'utilisateurs qui regardent des vidéos omnidirectionnelles.

Ce travail a donné lieu à neuf publications scientifiques, dont huit conférences internationales et un article de revues. Sur ces neuf publications, quatre ne sont pas abordées dans la présente thèse, par souci de cohérence, car elles traitent du filtrage de trames vidéo pour le streaming de vidéos à faible latence.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

16K 16 384 × 8192 pixels.

1DoF one Degree of Freedom.

2D two Dimensional.

3D Three Dimensional.

3DoF three Degrees of Freedom.

3GPP 3rd Generation Partnership Project.

4K 4096 × 2048 pixels.

6DoF six Degrees of Freedom.

8K 8192 × 4096 pixels.

ABR Adaptive Bit-Rate.

API Application Programming Interface.

AVC Advanced Video Coding.

B bidirectional-predicted.

CDF Cumulative Density Function.

CDN Content Delivery Network.

CPU Central Processing Unit.

DASH Dynamic Adaptive Streaming over HTTP.

EPFL Ecole Polytechnique Fédérale de Lausanne.

FoV Field of View.

GOP Group of Picture.

GUI Graphical User Interface.

HAS HTTP Adaptive Streaming.

HDK2 Hacker Development Kit 2.

HDS HTTP Dynamic Streaming.

HEVC High Efficiency Video Coding.

HLS HTTP Live Streaming.

HMD Head-Mounted Display.

I intra-predicted.

ILP Integer Linear Program.

ISO International Organization for Standardization.

ISOBMFF ISO base media file format.

JPEG 2000 Joint Photographic Experts Group 2000.

JVET Joint Video Exploration Team.

LTE Long Term Evolution.

MCTS motion-constrained tile sets.

MILP Mixed Integer Linear Programming.

MKV Matroska.

MOS Mean Opinion Score.

MP4 MPEG-4 Part 14.

MPD Media Presentation Description.

MPEG Moving Picture Experts Group.

MPEG-I MPEG – Immersive .

MPTCP Multi-Path Transmission Control Protocol.

MS-SSIM Multiscale - Structural Similarity.

MSE Mean Square Error.

MSS Microsoft Smooth Streaming.

MVP Multi-ViewPoint.

OMAF Omnidirectional MediA Format.

OpenGL Open Graphics Library.

OSVR Open-Source Virtual Reality.

OTT Over-The-Top.

P inter-predicted.

PCC Pearson Correlation Coefficient.

PDF probability density function.

POC picture order counts.

PSNR Peak Signal to Noise Ratio.

QEC Quality Emphasis Center.

QER Quality Emphasized Region.

QoE Quality of Experience.

QP Quantization Parameter.

RAP Random Access Point.

RoI Region of Interest.

RTMP Real-Time Message Protocol.

RTP Real-time Transport Protocol.

RTT Round-Trip Time.

S-PSNR Spherical - Peak Signal Noise to Ratio.

S-PSNR Spherical Peak Signal to Noise Ratio

SLERP spherical linear interpolation.

SRD Spatial Relationship Description.

SSIM Structural Similarity.

TV television.

URL Uniform Resource Locator.

VoD Video on Demand.

VQM Video Quality Metric.

VR Virtual Reality.

WRoI-S-PSNR Weighted Region of Interest - Spherical - Peak Signal Noise to Ratio.

WS-PSNR Weighted Spherical - Peak Signal Noise to Ratio.

Part I

INTRODUCTION

1

INTRODUCTION

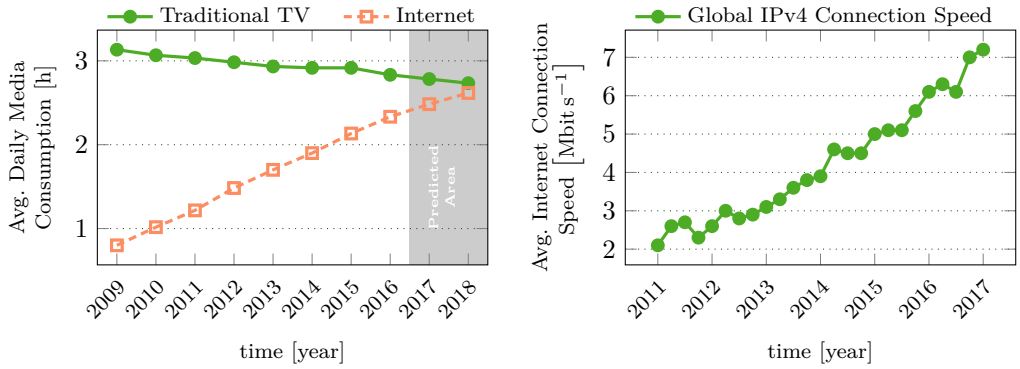
1.1 GENERAL CONTEXT

During the last decade, the time people spend consuming medias from the Internet has exploded. Analysts predict the Internet will become the main source for entertainment in the next decade [159]. This tendency, illustrated in Figure 1.1a, is mainly due to the better accessibility of high-quality media, which as been influenced by three factors: (*i*) As depicted in Figure 1.1b, the average Internet connection speed has highly increased during the last decade, allowing people to consume, without delay, content with better quality. The wide deployment of wireless access networks, such as Wi-Fi and 4G, allow users to consume content wherever and whenever they want. (*ii*) Devices able to decode high-quality content and able to connect to high speed network have become mainstream. And finally, (*iii*) the number of videos available on the Internet has exploded, thanks to content providers, such as *YouTube*, *Dailymotion*, *Netflix*, *Hulu*, *Twitch*, which allow content producers to offer multimedia content over the Internet. The diversity of content type, including Video on Demands (VoDs), television (TV) program replays, and live streaming of e-sport events, helps to attract a wide range of the population to those services.

Among all media consumed on the Internet, video is the one users spend most time on and the one that uses most Internet resources. For instance, in the first quarter of 2018, more than 4.80 billion hours of video content was streamed over the Internet (this is equivalent to 6 centuries of content streamed every single day), with an increase of 114 % compared to the same period in 2017 [27]. Moreover, video streaming represented up to 73 % of global Internet traffic in 2016 and is predicted to grow up to 82 % by 2021 [25].

Compared to traditional broadcasted TV programs, video streaming services over the Internet offer more freedom to the users and provide a real feeling of personalized services. The user can decide when and where to watch a content, can pause, skip a part, and continue to watch the content later. This flexibility is one of the main strengths of video streaming services. The feeling of personalized services comes from the possibility for the streaming service to adapt the video quality based on how the user is consuming the media (on a cellphone or on a TV, with a high or low speed Internet connection), but overall come from the very efficient content recommendation services offer by most service providers. The user is always able to find content he likes and so keep using the platform of the content provider. Those content providers are Over-The-Top (OTT) companies: they stream the multimedia content directly to the devices of their customers without intermediaries, which allows them to collect precious information on their users.

The key enabling technologies that allows OTT companies to stream so much content on the Internet is the Adaptive Bit-Rate (ABR) streaming implemented in HTTP Adaptive Streaming (HAS) protocols such as Dynamic Adaptive Streaming over HTTP (DASH). With HAS, content provider deliver videos to their customers like broadband companies (i. e. same content for multiple users) while giving to them the feeling of personalized services. The main idea is to encode the videos into different representations each having different resolutions and/or bit-rates. Each representation is split into segments of a few seconds long (typically between 2 s to 3 s long). The users' device run a video play, usually denoted as client, that decides which segment to download, based on its available download bandwidth and on the device resources, by sending HTTP requests. The service provider only needs to send the requested segment to the client with a regular HTTP server and without any dedicated processing because all the



(a) Average daily media consumption per person worldwide

(b) Average Global Internet Connection Speed

Figure 1.1: Evolution over time of the daily media consumption per person (source Zenith via Statista [159]) and of the global Internet access connection speed (source Akamai via Statista [4])

decisions are made by the client. This technology can easily scale with the number of client and the number of videos and adapt to users' needs.

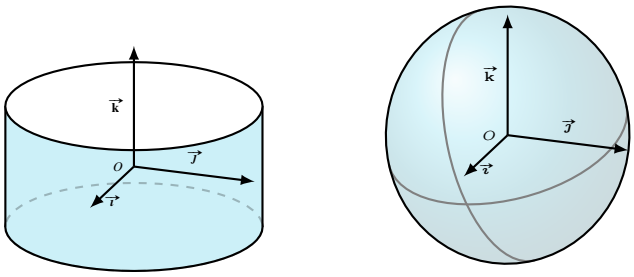
The last decade has also seen the increase of the interest of users in interactive media, mainly because of the technology needed for such contents is becoming more and more affordable. Interactive medias are multimedia content that can be actively modified by a user while being watched. They use new generations of video players that exploits the increasing computing capacities of users' devices to allow new forms of interaction. For instance, the user can change the position of the camera, zoom inside the video or even interact with objects inside the video. Multiple OTT services are based on interactive videos such as cloud gaming, multi-view videos, and omnidirectional videos.

An omnidirectional video, also called *spherical videos* or 360° *videos*, is a video with pixel captured in every direction of space. When a user watches such a video, only a small portion of the video, denoted as viewport, is displayed to the user screen. One can represent an omnidirectional video as a video on the inner surface of a sphere. To generate the viewport, a "virtual camera" is positioned at the center of the sphere and extracts only the portion of the video in its Field of View (FoV). The orientation of the virtual camera is controlled by user's feedback: if the video is watched using a Head-Mounted Display (HMD), the omnidirectional video player uses the head orientation of the user, otherwise it uses keyboard or remote control inputs.

Virtual Reality (VR) multimedia content aims to offer users a high feeling of immersion. The users should feel like they are living a real experience: the barrier between reality and virtuality disappears. Such high level of immersion can only be provided with a high level of interactivity between the users and the multimedia content.

To ensure a good immersion into the content, the spatial resolution of the viewport should be high enough for the users not to perceive the pixel borders when using a HMD. A viewport with a 4096 × 2048 pixels (4K) resolution usually provide imperceptible pixels. The vendors of HMD recommend for the whole system to be able to react to head movement as fast as the refresh period of the HMD to avoid *simulator sickness* [87]: 11 ms for 90 Hz HMDs. In other words, the motion-to-photon delay, defined as the time between a head movement and the display of the first viewport corresponding to this head movement, should be lower than 11 ms.

The streaming of such omnidirectional content on the Internet is challenging. To enable 4K viewports, the whole omnidirectional content should be equivalent to at least a 16 384 × 8192 pixels (16K) video. Streaming the whole content (16K video per eye) at 90 fps with existing streaming technologies would requires more than 150 Mbit s^{-1} [96] which is way higher than the median home



(a) Panoramic Videos **(b)** Omnidirectional Videos

Figure 1.2: Panoramic versus Omnidirectional videos. In blue, part with pixels.

download speed (64 Mbit s^{-1} on United-States fixed connection and 22 Mbit s^{-1} on United-States mobile connection [122]).

The goal of this thesis is to propose and evaluate modifications in today's OTT delivery architecture and to propose new ways to represent omnidirectional videos to enable the streaming of such interactive and immersive content through the Internet.

1.2 MOTIVATION

Since Neumann, Pintaric, and Rizzo [90] published in 2000 the first paper introducing an "omnidirectional" capturing video system, many researchers have worked on omnidirectional videos/images. But what is often called "omnidirectional video" or "panoramic video" in the literature is not exactly the same as what we denote by omnidirectional video in this dissertation. Often, it is cylindrical content, with pixels only close to the equator, captured in 360°, without pixel near the poles. It is a content where users can only turn on themself (i. e. yaw rotation only). In this dissertation, we denote by *panoramic videos* the cylindrical videos with one Degree of Freedom (1DoF) (see Figure 1.2a), and by *omnidirectional videos* the spherical videos with three Degrees of Freedom (3DoF) (see Figure 1.2b), and we mainly focus on the latter.

Before the beginning of this thesis, a small number of studies have dealt with the streaming of omnidirectional videos in the context of HAS for OTT companies, targeting HMD client devices. Existing works can be classified into three groups: (*i*) the streaming of panoramic videos to traditional TV displays, to tablets or to huge "omnidirectional" screens denoted as CAVE-like displays [36, 111], (*ii*) the study of efficient ways to render the content [8, 9], and (*iii*) the study of new possible user's interactions with the content [99]. Alface, Macq, and Verzijp [6] are the first, in 2012, to study the tiling of omnidirectional video. They model the tile quality selection, in the context of a bandwidth constrained streaming, into an Integer Linear Program (ILP). They did not performed their study with the tiling option of Moving Picture Experts Group (MPEG) High Efficiency Video Coding (HEVC) video codec but with the not-tile-compatible Joint Photographic Experts Group 2000 (JPEG 2000) image codec, nor did they discuss the integration of the system into HAS-like streaming architectures. Niamut et al. [92] were the first in 2013 to propose a full streaming architecture to stream panoramic videos. Their solution does not focus on omnidirectional videos and does not exploit the characteristics of such videos. No real consensus existed in the community on what is the best way to prepare the omnidirectional content before streaming, and how existing technologies can be upgraded to support efficient and high quality omnidirectional video streaming. The goal of this thesis is to contribute filling this gap.

In this thesis we focus on the streaming of omnidirectional videos to HMD devices, in the context of OTT content providers: HAS architectures, low motion-to-photon delay, and time dependent bandwidth.

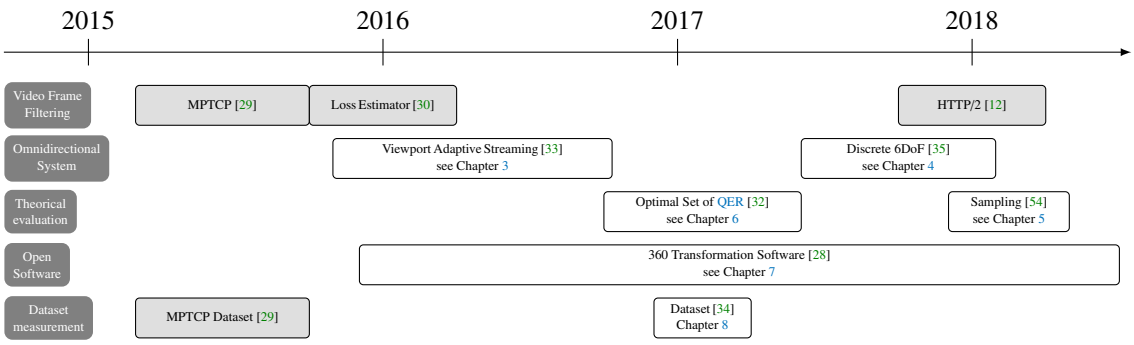


Figure 1.3: Contribution timeline. Dark gray boxes indication the topics of contributions on the same row. Each contribution is represented by a box with a short description inside. Contribution with white background are discussed inside the body of the dissertation, those with light-gray background are not discussed in this dissertation.

In the state-of-the-art, we can notice that few work have been done on the domain before 2015. Moreover no dataset of recorded trajectory of user watching omnidirectional videos with a HMD existed, making the study user behaviors inside the omnidirectional content more difficult. We will further discuss the related work in Chapter 2.

The goals of the thesis are then threefold:

i. **Streaming Architecture**: Propose and evaluate a streaming architecture compatible with HAS and especially with the widely deployed DASH protocol

ii. **Theoretical**: Propose theoretical analysis of the proposed architecture, on how to generate the video representations for this architecture and on how to objectively evaluate the quality of each representation.

iii. **Practical Tools**: Propose practical tools to evaluate the different propositions and to allow the community to reproduce the results of our work.

1.3 CONTRIBUTIONS AND ORGANIZATION OF THE MANUSCRIPT

1.3.1 *Introduction to the Contributions*

The contributions presented in this dissertation are related to omnidirectional videos. We exploited the characteristics of omnidirectional content to propose adaptive streaming solutions compatible with current OTT streaming architecture: mainly with DASH-like architectures. Our contributions in this domain can be classified into three groups, introduced in the previous Section and detailed further below: (*i*) Architecture; (*ii*) Theoretical studies; and (*iii*) Practical tools .

Reproducibility of research works is the cornerstone of science. To help the community to reproduce our works, we released, with all our publications, the software and datasets used to produce the results. The only exception is when dataset copyright owner forbid the release of the dataset.

The omnidirectional video stream contributions do not represent the full extent of the work done during the thesis. Figure 1.3 represents a timeline of the contributions of the thesis started in March 2015 and ended in May 2018. We chose not to talk about the video frame filtering contributions (with light-gray background in the Figure) in the body of this dissertation to keep a consistent document. Modern video player are able to decode a video bit-stream even if some of the encoded video frames are

missing. Missing frames introduce distortion in the decoded video (decoding artifacts due to missing information), but some missing frames introduce more distortion than others. In the context of low delay video streaming, we studied the possibility to not transmit some of the frames that introduce little distortion to mitigate temporary bandwidth shortage and avoid stalls. We applied this idea on a cross-layer video frame scheduler for Multi-Path Transmission Control Protocol (MPTCP), and on an applicative frame scheduler using the new features of HTTP/2. Readers interested about those contributions may read our published papers [29, 30].

The dissertation is structured as follow: Part I contains this introduction and exhibits the state-of-the-art, Part II contains a presentation and an evaluation of the architectures of viewport-adaptive streaming and of a possible extension to discrete six Degrees of Freedom (6DoF) video streaming, Part III comprehends some theoretical studies about omnidirectional videos, Part IV comprises the presentation of some practical tools developed to better study viewport-adaptive streaming, omnidirectional video projections, and users behaviors inside the 360° content, and finally Part V concludes the thesis.

The next sub-sections introduce the contributions that will be further discussed in the body of the dissertation.

1.3.2 *Streaming Architecture*

We propose a new adaptive streaming architecture, compatible with DASH-like protocols, to stream omnidirectional videos over the Internet. We introduce the concept of heterogeneous spatial quality representations, which are video representations where the quality is not the same everywhere. When an omnidirectional video is encoded with heterogeneous spatial quality, viewport extracted in an high quality area has a better visual quality than viewport extracted in lower quality areas. We introduce the concept of Quality Emphasized Region (QER), which is a connected subset of the sphere with higher quality than the rest of the spherical regions, and the concept of Quality Emphasis Center (QEC), which is the centroid of the QER, to propose a viewport-adaptive streaming architecture that not only adapts to the user's available bandwidth but also adapts to the user's head orientation. This work was published in the proceedings of IEEE ICC 2017 [33] and is presented in Chapter 3.

This architecture can only target 3DoF. Indeed, users can only rotate their head inside the content but they cannot do any translational movements. Offering 6DoF on demand content is still challenging because it either requires huge dedicated computing resources per user (like what is done with cloud gaming) or requires the usage of still not mature technologies such as lightfield signal.

We studied a possible next step toward 6DoF: multi-viewpoint omnidirectional videos. It consists in a set of synchronized omnidirectional videos that films the same scene from different viewpoints. The user can "teleport" from one viewpoint to another. It is not a full 6DoF but instead a *discrete 6DoF* scenario where the user can freely choose the orientation of her head but can only move to a predefined discrete set of position in space. Our study discuss different possible implementation of multi-viewpoint omnidirectional videos streaming, model the optimal download strategy and evaluate two radical download strategies. This study was partly done while I was visiting the Signal Processing Laboratory (LTS4) at the Ecole Polytechnique Fédérale de Lausanne (EPFL). This work was published in the proceedings of ACM MMSys'18 [35] and is presented in Chapter 4.

1.3.3 *Theoretical Study*

Sphere-to-plane projections are usually used to enable omnidirectional videos to be encoded with encoders designed for traditional two Dimensional (2D) rectangular videos. To better understand the impact of the sphere-to-plane projection on the quality of the extracted viewports, we studied the relation between the spherical pixel sampling and the distortion introduced inside the extracted viewports. We

denote by spherical sampling, or spherical pixel density, the number of pixel per surface units on the sphere. The spherical sampling approach is useful to study projection that continuously degrades the quality of the video such as the offset projection, proposed by Facebook to generate representation with QER and used in the special case of the cubemap projection [73, 162]. We generalize this offset transformation to any projection. This work is published in the proceedings of IEEE MMSP'18 [54] and introduced in Chapter 5.

Even if we know how to generate representations with a QER, we need tools to automatically decide the shape and the position of the QER. We aim to model the QER allocation independently of the actual sphere-to-plane projection used, using content specific users' viewing statistics. We model this problem into a bit-rate allocation problem within the spherical video. The goal is to generate n representations which fulfill a total bit-rate budget constraint, and which maximize the quality inside the viewports generated by recorded head movement trajectories. To perform this allocation, we introduce the notion of *surface bit-rate*. This work was published in the proceedings of ACM MM'17 [32] and is presented in Chapter 6.

1.3.4 *Practical Tools and Dataset*

To evaluate the performance of viewport-adaptive streaming with different sphere-to-plane projections, head-orientation prediction and head-orientation datasets, we developed an open source software able to convert omnidirectional videos from one projection to another, to extract viewports, to replay head movement trajectory datasets and to compute different objective distortion metrics. The software is designed to allow easy insertion of new sphere-to-plane projections. This software was used in most of our works to evaluate our proposition and is available for the community on *GitHub* [28]. Chapter 7 describes its design.

Datasets of recorded head movement trajectory of users watching omnidirectional videos are necessary to enable the community to evaluate the different viewport-adaptive streaming proposals, to understand global user behaviors within the omnidirectional content and to develop algorithms to predict future head positions. For instance, viewport-adaptive streaming supposes that users' head movements can be predicted a few seconds in the future. Defining a small set of QER supposes there exists some well defined Region of Interest (RoI) and that most users focus on those regions.

To alleviate this lack, we gathered a head movement dataset of 59 users watching five 70 s long videos and released this dataset openly. This dataset is among the three first open dataset on omnidirectional videos alongside with Lo et al. [82] and Wu et al. [145]. This dataset was published in the proceedings of MMSys'17 [34] and is described in Chapter 8.

2

RELATED WORK

2.1 INTRODUCTION

This chapter introduces the concepts needed to understand the body of the dissertation, and discusses the existing related work.

The interest of the research community related to adaptive streaming of omnidirectional video has strongly grown since 2015. Most of the work presented in this dissertation was done in parallel of the related work discussed in this Chapter.

This Chapter is organized as follow. Section 2.2 introduces the mathematical set-up used in the whole dissertation. Section 2.3 presents an abstract representation of an omnidirectional video. First, a definition of traditional two Dimensional (2D) video is given followed by a formal definition of omnidirectional video and of viewports. Finally, the sphere-to-plane projection are defined and the two main sphere-to-plane projections used in the context of omnidirectional video, namely the equirectangular and the cube-map projection, are presented. Section 2.4 presents the principles behind video encoding, and an overview of subjection and objective Quality of Experience (QoE) evaluation. Section 2.5 presents the HTTP Adaptive Streaming (HAS), which are the main group of protocols used to stream traditional video over the Internet. Section 2.6 presents the viewport-adaptive streaming: an evolution of HAS to enable optimized streaming of omnidirectional video. It also introduces the concept of heterogeneous spatial video encoding along with the new features of Moving Picture Experts Group (MPEG) High Efficiency Video Coding (HEVC)/H.265 codec useful in this context. Section 2.7 discusses the related work with regard to prediction of future head orientation of users watching omnidirectional videos. Finally, Section 2.8 presents the different standardisation efforts and innovative industrial solutions.

2.2 DEFINITIONS, NOTATIONS AND CONVENTIONS

$\triangleq$ denotes the definition equality and $=$ denote the standard deductive equality.

World Space and Reference Frame.

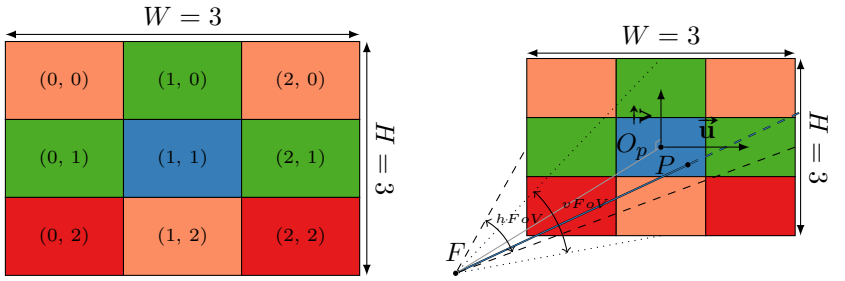
In the whole dissertation, we consider the world to be the affine Euclidean space $\mathbb{E}^3 = \mathbb{R}^3$, with the orthonormal direct world reference frame $\left(O,\ \vec{\imath},\ \vec{\jmath},\ \vec{\mathbf{k}}\right)$. The right hand rule is used to define rotations.

If A and B are two points, the vector to go from A to B is denoted as $\overrightarrow{\mathbf{AB}}$. Using traditional affine space notations, we have $\overrightarrow{\mathbf{AB}} \triangleq B - A$ and $B = A + \overrightarrow{\mathbf{AB}}$. To simplify the notation we denote by $\vec{\mathbf{A}}$ the vector $\overrightarrow{\mathbf{OA}}$.

Inner and Cross Products.

This space is equipped with an inner product, denoted as $\cdot$, and with a cross product, denoted as $\times$. We denote by $\|\vec{\mathbf{u}}\| \triangleq \sqrt{\vec{\mathbf{u}} \cdot \vec{\mathbf{u}}}$ the norm of $\vec{\mathbf{u}}$, and by $\theta^{\vec{\mathbf{v}}}_{\vec{\mathbf{u}}}$ the unsigned angle between $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$ in a plane containing both vectors. The inner product between two vectors $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$ verify:

$$\vec{\mathbf{u}} \cdot \vec{\mathbf{v}} = \|\vec{\mathbf{u}}\| \|\vec{\mathbf{v}}\| \cos(\theta^{\vec{\mathbf{v}}}_{\vec{\mathbf{u}}}) \tag{2.1}$$



(a) 2D picture with resolution 3x3 **(b)** 2D picture in Three Dimensional (3D) space

Figure 2.1: Traditional 2D picture

And the cross product verify

$$\|\vec{\mathbf{u}} \times \vec{\mathbf{v}}\| = \|\vec{\mathbf{u}}\|\|\vec{\mathbf{v}}\| \sin(\theta_{\vec{\mathbf{u}}}^{\vec{\mathbf{v}}}) \tag{2.2}$$

We define the norm of a point P as follow: $\|P\| \triangleq \|\overrightarrow{\mathbf{OP}}\|$.

Viewing Direction.

We denote by viewpoint a point A in the world. It is a point where an isolated observer is supposed to see the world. We denote by direction a unit vector $\vec{\mathbf{u}}$. We denote by *viewing direction* $(A, \vec{\mathbf{u}})$ a *ray* (or *half-line*) starting from a viewpoint A and having the direction $\vec{\mathbf{u}}$. It is the set of points $\left\{M \quad | \quad \exists t \in \mathbb{R}^+, \overrightarrow{\mathbf{AM}} = A + t\vec{\mathbf{u}}\right\}$.

A ray is uniquely defined by the couple $(A, \vec{\mathbf{u}})$.

2.3 OMNIDIRECTIONAL VIDEOS

2.3.1 *Traditional 2D Rectangular Videos*

We denote by 2D picture with resolution $W \times H$, a function C that maps any node from a rectangular regular $W \times H$ grid with a color (a point in a three dimensional color space $\mathbb{C} \subset \mathbb{R}^3$ [133]). In other words, $C(w,h)$ returns the color of the node with coordinate (w,h) in the grid. A 2D picture C with resolution $W \times H$ is often represented as a rectangle covered regularly into $W \times H$ identical rectangular surfaces, denoted as pixel, with all points inside a pixel (w,h) painted with the same color $C(w,h)$. This is a nearest neighbor interpolation. The only points with known color is the center of the pixels, all other points have a color interpolated from the color function C. Figure 2.1a depicts this 2D picture representation for a picture with 3×3 resolution. In this example, $C(1,1)$ is equal to the color "blue".

It is possible to visualize a 2D picture in the 3D space $\mathbb{E}^3$ by emulating a perfect calibrated camera [84]. Such camera is characterized by its focal point F, its horizontal Field of View (FoV) $hFoV$ and its vertical FoV $vFoV$. The focal point F is the point where all ray coming inside the camera are supposed to converge. The vertical (respectively horizontal) FoV is the angular distance between the top (respectively the left border) and the bottom (respectively the right border) of the picture measured from the focal point F. The picture is a planar rectangle positioned such that the orthogonal projection O_p of F on the plane containing the picture is the center of the rectangle. Figure 2.1b illustrates this camera model in the 3D space. Each point P inside the rectangular picture is mapped to a color by the function C. The light ray starting from the focal point F and going through a point P on the picture (i. e. the ray $(F, \overrightarrow{\mathbf{FP}}/\|\overrightarrow{\mathbf{FP}}\|)$) has the color of P.

We denote by $(O_p, \vec{\mathbf{u}}, \vec{\mathbf{v}})$ a reference frame associated with the picture. $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$ are coplanar with the picture plane: $\vec{\mathbf{u}}$ is in the direction of the "horizontal" and $\vec{\mathbf{v}}$ of the "vertical". If we decide to set $\|\vec{\mathbf{u}}\|$ (respectively $\|\vec{\mathbf{v}}\|$) as the horizontal (respectively vertical) dimension of a pixel, we have:

$$\|\vec{\mathbf{u}}\| = \frac{2}{W}\tan(\frac{hFoV}{2}) \tag{2.3}$$

and

$$\|\vec{\mathbf{v}}\| = \frac{2}{H}\tan(\frac{vFoV}{2}) \tag{2.4}$$

For the sake of simplicity, we suppose $\|\overrightarrow{\mathbf{FO_p}}\| = 1$.

Only rays coming from the focal point F and intersecting the picture plan inside the picture rectangle are associated with a color. A ray $(F, \vec{\mathbf{a}})$ intersects the plane that contains the picture if and only if:

$$\vec{\mathbf{a}} \cdot \overrightarrow{\mathbf{FO_p}} > 0 \tag{2.5}$$

In this case the intersection point P' with the plane is:

$$P' = F + \frac{\vec{\mathbf{a}}}{\vec{\mathbf{a}} \cdot \overrightarrow{\mathbf{FO_p}}} \tag{2.6}$$

By definition of a reference frame, as P' is in the picture plane, there exists two real w and h such that $P' = O_p + w\vec{\mathbf{u}} + h\vec{\mathbf{v}}$. P' is inside the picture if and only if:

$$w = \vec{\mathbf{P'}} \cdot \frac{\vec{\mathbf{u}}}{\vec{\mathbf{u}} \cdot \vec{\mathbf{u}}} \in \left[-\frac{W}{2}; \frac{W}{2}\right] \tag{2.7a}$$

$$h = \vec{\mathbf{P'}} \cdot \frac{\vec{\mathbf{v}}}{\vec{\mathbf{v}} \cdot \vec{\mathbf{v}}} \in \left[-\frac{H}{2}; \frac{H}{2}\right] \tag{2.7b}$$

In other words, a ray $(F, \vec{\mathbf{a}})$ is associated with a color if and only if Conditions (2.6), (2.7a) and (2.7b) are true, and in this case the ray has the same color as the point P' defined by Equation (2.6).

We denote by *video* a sequence of picture captured at successive equally spaced instants in time. The period between two pictures is usually given in hertz (Hz) by the number of picture per second or fps. Traditionally, only the color function C is time dependent, the FoVs and the resolution are constant.

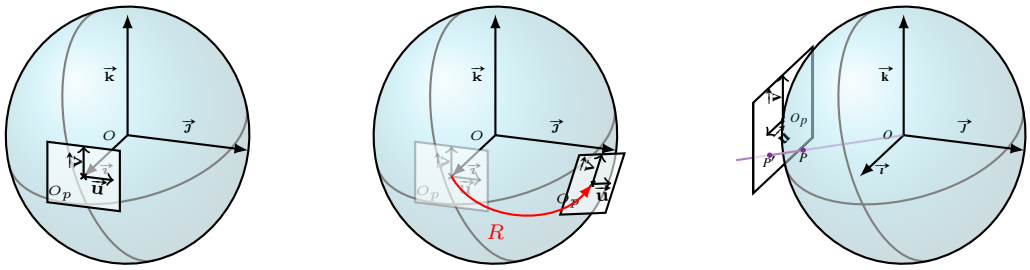
2.3.2 *Omnidirectional Videos*

An omnidirectional image is an image on a unit sphere of $\mathbb{R}^3$. We can consider such image was captured with a fully omnidirectional camera. An omnidirectional camera can be considered as a calibrated central camera [84]. Note that in practice, fully omnidirectional central camera cannot exists but can be approximated with a set of traditional camera, whose pictures are stitched together to emulate a fully omnidirectional central camera [17].

Like a traditional image, any point on the surface of the unit sphere is associated with a color by a function C. The focal point is the center of the sphere. For the sake of simplicity, except in Chapter 4, we always consider the unit sphere centered at the origin O of the world reference frame. Unlike 2D pictures, any ray starting from the focal point O intersect the unit sphere and so can be associated with a color.

Viewports

Traditional displays or Head-Mounted Displays (HMDs) can only display 2D pictures. To allow a high level of immersion inside the content, the users should feel like the display is a window through which



(a) Reference Viewport **(b)** Any Viewport **(c)** Gnomonic Projection

Figure 2.2: Viewport

they watch the omnidirectional video. To do so, the picture inside the display should have a wide FoV and should represent the very image of what the user would have perceived if the scene displayed was real. Such a 2D picture, called viewport, can be captured by placing a virtual camera (as modeled in Section 2.3.1) at the center of the omnidirectional picture (i. e. $F = O$). The viewport is positioned such that its plane is tangent to the sphere at the viewport center O_p. Figure 2.2a illustrates such a viewport. The color function of the viewport is computed using a *gnomonic projection* [118]: the color of the pixel centers P' is fixed as the same as the color of its projection P on the sphere following the light ray coming from the center of the sphere O (i. e. $C'(P') = C(P = \frac{P'}{\|P'\|})$). This is an azimuthal perspective projection with center O.

Figure 2.2c illustrates how this projection works. The pixel center P' got mapped to color purple because its gnomonic projection P on the sphere is purple.

With today's technologies, the viewports have a FoV close to 110°× 90°. This means that viewports, which are the pictures actually displayed to the user, represent only around 20 % of the whole omnidirectional image. It is commonly admitted in the community that a viewport resolution with a 4096 × 2048 pixels (4K) resolution would provide a high feeling of immersion inside the content for a user wearing a HMD. Indeed with such a high resolution the user does not perceive the pixel matrix because, with nowaday HMDs FoV the angular size of a pixel in a 4K viewport is smaller than the human eye sensitivity. As far as we known, no scientific study as been perform to evaluate what is the optimal resolution to use in the viewport to provide high immersion. The 4K resolution is inferred based on prior knowledge on the visual accuracy of the human visual system. Using a viewport with a 4K resolution means the total resolution of the omnidirectional video should be at least equivalent to a 16 384 × 8192 pixels (16K) planar video to enable extraction of 4K viewports.

Rotations and Viewports. A viewport can be uniquely identified by giving both its tangent point O_p and a vector collinear to $\vec{\mathbf{u}}$. O_p indicates the viewing direction of the center of the viewport, and $\vec{\mathbf{u}}$ indicates the tilt of the viewport. To properly define a viewport position, and to avoid any ambiguity at the poles, it is common to identify it to the *rotation* $R \in SO(3)$ that transform $\vec{\imath}$ into $\overrightarrow{\mathbf{O_p}}$ and $\vec{\jmath}$ into $\vec{\mathbf{u}}$. This rotation is unique and always exists (even at the poles). Figure 2.2a illustrates the reference viewport associated with the identity rotation, and Figure 2.2b shows how the viewport identified with rotation R can be obtained from the reference viewport.

2.3.3 *Sphere-To-Plane Map Projection*

State-of-the-art technologies to store and compress videos are optimized for 2D rectangular videos. To use those technologies it is needed to map the spherical images onto a 2D pictures. In other words, it is needed to define a color function C' for the 2D picture using the color function C of the spherical picture.

From "Ansgar Koreng / CC BY-SA 3.0 (DE)": https://commons.wikimedia.org/wiki/File:Magnolienbaum,_Wiesbaden-Biebrich,_360x180,_160409,_ako.jpg

Figure 2.3: Equirectangular Projection

The color function C' is usually defined by $C'(u,v) \triangleq C(F(u,v))$, with F a function that takes as input a point (u,v) on the 2D picture and return a point P on the sphere ($F : [-\frac{W}{2}; \frac{W}{2}] \times [-\frac{H}{2}; \frac{H}{2}] \to \mathbb{S}^2$). Note that the function F that maps the plane on the sphere is used to perform the sphere-to-plane projection: to generate the 2D picture C' from the spherical image C.

To preserve traditional encoder efficiency, the function F should be an isometric isomorphism: i. e. F should preserve object shapes, should preserve distance ratios, should be continuous, and should have an inverse function. With these properties, encoder can easily detect object movements inside the projected 2D video.

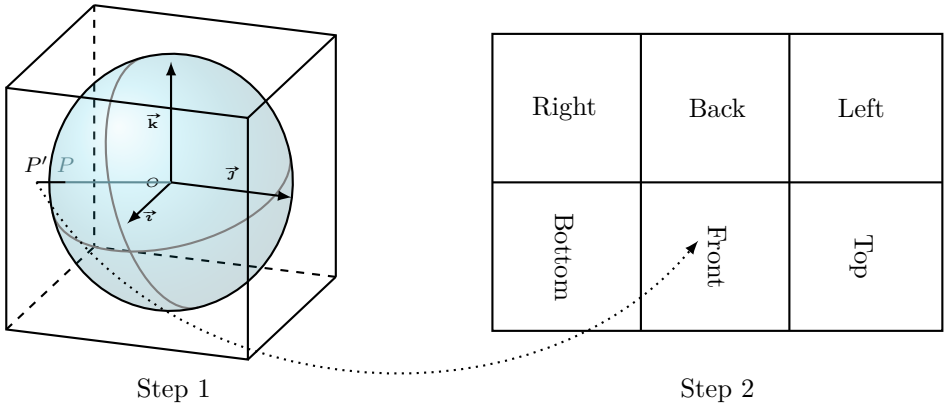
Theorem 1 (Gauss's Theorema Egregium [101])**.** *The Gaussian curvature of a surface is invariant under local isometry.*

So there exist no *nice* function between a 2D picture and a sphere. Theorem 1 implies that if two surfaces have not the same curvature, there exists no isometry between them. The curvature of the unit sphere is 1 everywhere and the curvature of the plane is 0 everywhere so a map between a sphere and a subset of the plane will necessarily distort some distances.

Sphere-to-plane projections have been widely studied in the context of cartographic projection [119] (i. e. projection of the earth or of the celestial sphere on a plane), but only two are usually used in the context of omnidirectional videos: the equirectangular projection [124] and the cube-map projection. Some other projections have been studied by the community [123, 150] such as the Equi-angular Cubemap projection and the Truncated Square Pyramid projection. Especially, Ye and Boyce [150] performed an extensive study of 13 projections. We will focus on some of them in Chapter 3.

Note that in many works, and in MPEG vocabulary, what we call projection is often described as a two steps operation. A first step usually called *projection* but denoted here as *geometrical projection* to avoid confusion, projects the sphere to a geometrical 3D object (for instance a cube or a pyramid). The faces of the geometrical object are then mapped to the planar picture. This mapping operation is called *packing*. Readers should be careful not to confuse the two definitions of projection. With our definition changing the packing (i. e. the position, orientation of a face on the 2D picture) changes the projection itself but does not influence the geometrical projection.

Equirectangular Projection. The equirectangular projection f maps the latitude and longitude of



(a) Cube

(b) 2D picture

Figure 2.4: Cube-Map Projection

a point on the sphere with the coordinate u and v in the 2D picture. With the world reference frame $(O, \vec{\imath}, \vec{\jmath}, \vec{\mathbf{k}})$ introduced previously we get:

$$f(u\vec{\mathbf{u}} + v\vec{\mathbf{v}}) \triangleq \cos(\frac{2u\pi}{W})\cos(\frac{v\pi}{H})\vec{\imath} \quad - \sin(\frac{2u\pi}{W})\cos(\frac{v\pi}{H})\vec{\jmath} \quad - \sin(\frac{v\pi}{H})\vec{\mathbf{k}}$$

and

$$f^{-1}(\vec{\mathbf{P}}) = \frac{W}{2\pi}\,\text{atan2}\left(\frac{\vec{\mathbf{P}}'}{\|\vec{\mathbf{P}}'\|}\cdot\vec{\jmath}, \frac{\vec{\mathbf{P}}'}{\|\vec{\mathbf{P}}'\|}\cdot\vec{\imath}\right)\vec{\mathbf{u}} + \left(\frac{H}{2} - \frac{H}{\pi}\,\text{acos}(\vec{\mathbf{P}}\cdot\vec{\mathbf{k}})\right)\vec{\mathbf{v}}$$

with $\vec{\mathbf{P}}' \triangleq \vec{\mathbf{P}} - \vec{\mathbf{P}}\cdot\vec{\mathbf{k}}\,\vec{\mathbf{k}}$, the orthogonal projection of $\vec{\mathbf{P}}$ on the plane $(O, \vec{\imath}, \vec{\jmath})$

The equirectangular projection is a continuous function whose inverse function is not well defined at the poles (i. e. when $\vec{\mathbf{P}} = \pm\vec{\mathbf{k}}$). All the points from the top border of the 2D picture ($v = \frac{W}{2}$) maps to the north pole of the sphere, and all the points from the bottom border of the 2D picture ($v = -\frac{W}{2}$) maps to the south pole of the sphere. We can see on Figure 2.3 that this property generate a lot of distortion in the 2D picture near the poles.

Cube-Map Projection. The cube-map projection can be split into two steps: (*i*) The sphere is projected, following the ray starting from its center, onto a cube circumscribed to the sphere, (*ii*) each

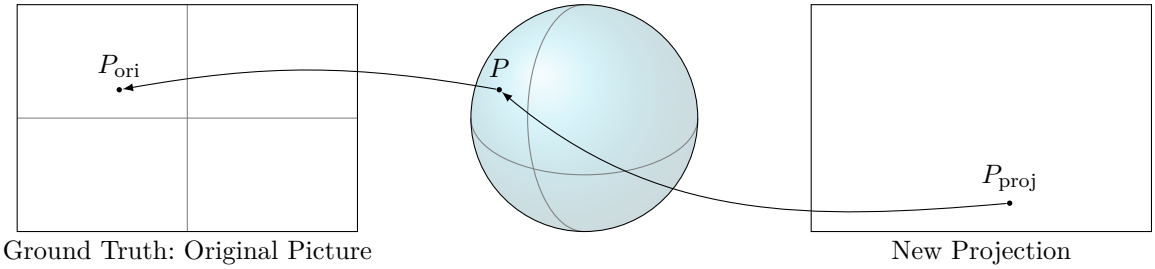


Figure 2.5: Projection from ground truth. Arrow indicates maps: pixels of the new projection are mapped to the sphere and then map to the original picture.

face of the cube is mapped to a part of the 2D picture. Figure 2.4a illustrates those two steps: first it project the point P from the sphere into P' on the cube and then map it to the 2D picture. The projection equation for a given face is the Equation (2.6).

The cube-map projection function is bijective once you fixed to which face belong each edge. In Figure 2.4 we can see that there is a discontinuity line at the middle of the picture.

2.3.4 *Ground Truth Omnidirectional Video*

With nowadays capture technologies, the original spherical video signal is never captured and stored directly. Most omnidirectional cameras use a set of synchronized traditional cameras with fish-eye lens [66]. The set of 2D pictures output from those cameras are stitched together to generate the output omnidirectional images. The output of the camera is not spherical images but projection on 2D pictures. Those projected 2D pictures are what we denote as reference, original or ground truth omnidirectional pictures.

Usually, to extract viewport or to project the omnidirectional video with a new sphere-to-plane projection, it is not possible to directly use the sphere-to-plane projections introduced in previous Section as the spherical image is not directly available, only a projection of it is available. It is first required to perform a plane-to-sphere projection followed by the sphere-to-plane projection. Figure 2.5 illustrates those steps: pixel centers from the 2D picture we want to generate (on the right side) are first mapped to the sphere and then mapped to the original ground truth picture. The map on the original picture will most likely not match a pixel center, and so the color of this point will not be directly available. The color of this pixel is interpolated based on neighbor pixel center colors thanks for instance to a nearest neighbor interpolation, a bilinear interpolation or a bicubic interpolation [138]. The neighborhood is usually established on the original 2D projection.

2.4 VIDEO ENCODING

In this Section we analyse the related work associated to the encoding of traditional 2D rectangular videos.

2.4.1 *Principles*

A raw, non encoded, RGB picture with 4K spatial resolution and 8 bit depth per color channel requires at least 199 Mbit of storage ($3840 \times 2160 \times 3 \times 8$ bit). Even YUV422p format, which use chroma downsampling on YUV color format, still requires at least 99.5 Mbit. A 4K video with 24 fps stored

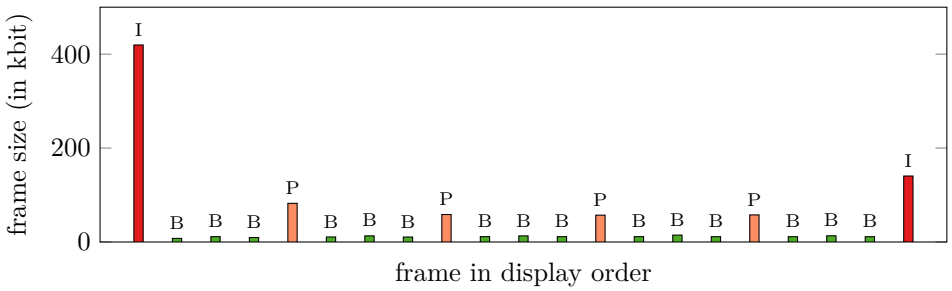


Figure 2.6: Frame size within a GOP for a video compressed with default parameter of *libx265* [146]

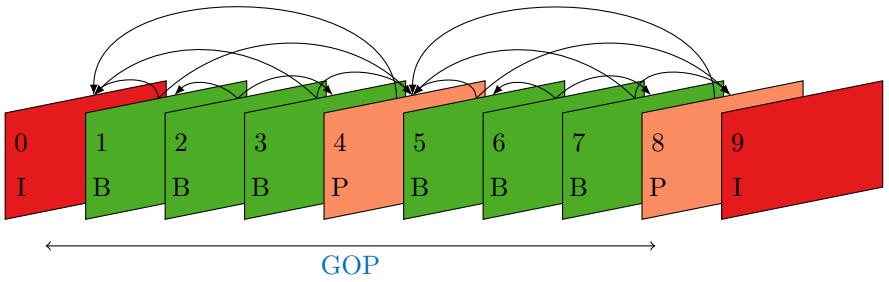


Figure 2.7: Hierarchical Encoding: GOP and Dependency Chain. Arrows indicate the dependencies between the frames. For instance, the frame 2 is a B frame which requires the B frames 1 and 3. Frames are ordered in display order.

with YUV422p format would then requires a bit-rate of at least 2.39 Gbit s^{-1}. It is not possible with current state-of-the-art storages and networks to store and stream non compressed video.

Videos contain redundant information that come from pixels' spatial and temporal correlations:

Spatial Correlation Neighbor pixels in videos pictures are usually consistent and are highly correlated, as they represent color from light ray coming from the same physical object. For instance, JPEG 2000 [128] codec uses this spatial correlation to compress the pictures.

Temporal Correlation Consecutive pictures in a video are also usually highly correlated as they represent the same scenes taken at different instant in time. Once object movements have been identified within the pictures, it is possible to map some highly correlated blocks of pixels together between two pictures, and to use this correlation to compress the information.

Video encoders, such as MPEG's HEVC / H.265 codec [2, 127], MPEG's Advanced Video Coding (AVC) / H.264 [1, 144], Google's VP9 [88] and the Alliance for Open Media's AV1 [21], use the spatial and temporal redundancies to compress the video signal. In this dissertation we mainly focus on the HEVC codec, but many results are compatible with other compression standards.

The HEVC/H.265 codec, uses a hierarchical structure to exploit the temporal and spatial redundancies in the video and efficiently compress it. Pictures from the original video are encoded into three types of frames: intra-predicted (I) frames, inter-predicted (P) frames and bidirectional-predicted (B) frames. I frames are encoded using only intra-predictions: the picture is compressed using only internal redundant information, as if it was only one picture. P frames are encoded using both intra and inter-predictions: the encoder can use a previously encoded picture to exploit temporal redundancy but can also use internal redundancy. B frames are similar to P frames with in addition the possibility to use a second picture to exploit temporal redundancy. One picture is usually displayed before the B and the second is usually displayed after. With B frames the frame encoding/decoding order is not necessarily the same

as the picture display order. Figure 2.6 illustrates how using some temporal correlation improve the compression efficiency. Figure 2.7 depicts the dependencies between the frames.

The dependency chains between the frames make it impossible for a decoder to directly decode most of the frames in the video. For instance, to decode the frame number 6 in Figure 2.7, it is mandatory to decode first all frames until the last I frame (i. e. frames 0, 1, 2, 3, 4, 5, 7 and 8). Indeed, I frames can be decoded independently, without any prerequisite. A video decoder can start decoding the video stream at any I frames, they are Random Access Points (RAPs). There is a trade-off between the frequency of RAP in the encoded video and the bit-rate of the stream. The more frequent are the RAPs, the easiest it is to seek to any position in the video but the bigger the bitstream. A group of frames that depend on each others are often denoted as Group of Picture (GOP). In Figure 2.7, frames from 0 to 8 form a GOP, and frame 9 starts a new one.

Even if most codecs allow lossless compression, content providers usually perform lossy compression to better control the bit-rate of the content.

2.4.2 *Video Quality Evaluation*

Lossy compression generates distortion in the video images that may impact the QoE of the users. The QoE is a subjective feeling perceived by a user while watching a multimedia content. It is often influenced by external elements such as the environment where the user is watching the video, the context and opinion of the user over the content itself. To avoid as much as possible environment and context biases and to allow measurements reproducibility, subjective evaluation of users' QoE are performed in controlled environments. The ITU-T Recommendation P.910 [63] provides methods to assess subjective quality of video content. It strictly defines parameters such as the room illumination, the user's viewing distance, the test duration and the selection of the evaluators.

Most subjective quality evaluations use the Mean Opinion Score (MOS) to measure the QoE. Each user note the evaluated content on a discrete scale, usually an integer ranging from 1 ("bad" quality) to 5 ("good" quality), and the score attributed by each user is averaged to generate the MOS. This score represents with a hight fidelity the average user QoE.

Performing valid subjective video quality evaluation is constraining, laborious, and requires time and resources which are not available for content provider to evaluate each encoded version of each video content. Those tools were designed for video encoder evaluation and for display evaluation and not for content providers that requires low cost, instantaneous evaluation. Objective metrics are needed to automatically assess the video quality and to estimate the quality a user would have perceived. Fidelity evaluation of objective quality assessment metrics is performed by evaluating their correlation with the MOS measured with a subjective assessment.

The objective factors that influence the MOS include:

IMAGE DISTORTION The distortion introduced by the lossy compression, if visible, can reduce the QoE. The users are even more sensitive to high variation in the quantity of distortion during time, as low distortion level anchor the user expectation.

START-UP DELAY The delay between when a user requested for the video and when the video actually start to be played.

STALLS A stall is a pause in the video display decided by the video play without user consent. It generally happen after a buffer starvation. Users are usually more sensitive to frequent stall than long stall

Many models has been proposed to predict the users QoE based on objective measurement of those factors [45, 108, 109]. In this dissertation, we suppose, except in Chapter 4, optimal scenarios where

the start-up delay is fixed and where stalls are always avoided. In this context (no stall and fixed start-up delay), only the displayed image quality can influence the user experience.

Multiple metrics have been developed to assess video quality by measuring distortion in the pictures, but they are not all well correlated with the subjective MOS. Chikkerur et al. [22] surveyed the objective video quality assessment tools available before 2011. Three types of video quality assessment metrics exists: (*i*) full-reference metrics which have access to the original, highest quality, video (usually the video used to generate the encoded videos), (*ii*) reduced-reference metrics which have access to a simplified representation of the original video, and (*iii*) no-reference metrics which have only access to the encoded bit-stream.

In this dissertation, to evaluate our proposals, we focus on full-reference metrics because we always have access to the original videos and because they usually has a better correlation with the subjective MOS. The main full-reference metrics are the Peak Signal to Noise Ratio (PSNR), the Multiscale - Structural Similarity (MS-SSIM), the Video Quality Metric (VQM) [58].

The MS-SSIM is a static image metric that computes the structural similarities between the original image and the compressed image [143]. It returns a dimensionless real value between 0 and 1, 1 being identical images. When the MS-SSIM is higher than 0.95, real users cannot distinguish the original image from the compressed image. For a video, the metric is computed frame by frame between the original video and the encoded video and averaged to get a unique value. The MS-SSIM have a good correlation with the subjective MOS [22].

The Mean Square Error (MSE) is a measure of the average squared error between the luma component of each pixel of the picture compared to a reference picture. The PSNR is often preferred to the MSE. It represents the same information but expressed in logarithmic scale in decibel (dB), and compared to the maximum value of the MSE: $PSNR = 10\log_{10}(\frac{MSE_{max}}{MSE})$. The PSNR alone does not represent well the actual perceptual quality felt by users. The absolute PSNR value is highly content dependent. The PSNR difference between two versions of the same video gives a good approximation of the subjective quality gap between the versions [70].

2.5 TRADITIONAL HTTP ADAPTIVE STREAMING ARCHITECTURE

Adaptive streaming, and HAS in particular, is a widely adopted technology to stream traditional planar videos over the Internet. The goal of HAS is to adapt the quality of the streamed video to the available bandwidth. MPEG Dynamic Adaptive Streaming over HTTP (DASH) [61] is a commonly used HAS international standard. Its main competitors inside the HAS family are Apple's HTTP Live Streaming (HLS) [97], Adobe HTTP Dynamic Streaming (HDS) [79] and Microsoft Smooth Streaming (MSS) [5, 154]. One advantage of using a protocol based on HTTP instead of a UDP such as Real-time Transport Protocol (RTP) or TCP such as Adobe's Real-Time Message Protocol (RTMP) is that it can be used through standard firewalls and proxy servers without the need of any extra configuration. Moreover, it enables the usage of stateless web server instead of dedicated streaming servers with state records for each ongoing streaming session. DASH can be used to stream on demand or live videos.

To use DASH, the video has to be preprocessed by the server once before being delivered to many clients without further processing. The video is encoded into multiple representations, each of them being characterized by its bit-rate, its spatial resolution, and its video codecs. Then representations are split into multiple chunks, denoted as segments. In other words, a segment is a chunk of a few seconds of a video, encoded with a specific quality level, that can be downloaded by a client independently of any other segments. The set of all segments, generated for a given media in a way that a client can switch from one representation to another without the need to reset the video decoder, is called an *adaptation set*. A client can only switch from representation belonging to the same adaptation set. The server can also prepare segments for subtitle and for audio but we will not consider this possibility in this dissertation as usually the size of subtitle and audio segments are insignificant compared to the size

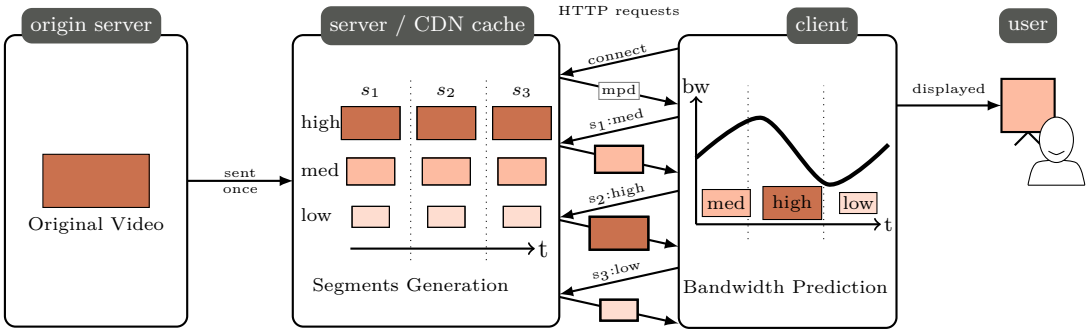


Figure 2.8: DASH delivery chain and video content preparation

of video segments. The diversity of codec, resolution and bit-rate of all segments allows the service provider to easily deliver the DASH video to a wide variety of client devices having each different screen resolution, Central Processing Unit (CPU) capacity and available bandwidth.

To start to play a video with DASH, a client requests the Media Presentation Description (MPD) file, which describes all video segments available on the server for a given media. The description contains among other things: the duration and bit-rate of the segments, the codec used, and the location of the segment described as an Uniform Resource Locator (URL) or as a template used to generate an URL. Based on those information and an estimation of the future available download bandwidth, the client selects which segments to download to maximize the user QoE [120] (maximize the displayed quality, minimize the quality switch, minimize the buffer starvations). Figure 2.8 illustrates the architecture of a DASH delivery system.

The DASH architecture scales easily with the number of users. The service provider only needs to generate the adaption set, with the associated MPD file, once, and then just need to answer regular HTTP requests with a standard HTTP server. The scheduling of segments downloading is a hard problem and a profusion of research works have focused on optimizing it [72], but the scheduling is only performed by the clients. The service provider does not solve the scheduling problem on its servers but in a decentralized way on each clients. Using HTTP requests allow an easy interconnection with Content Delivery Network (CDN) providers and HTTP cache service, enabling high scalability.

2.6 VIEWPORT-ADAPTIVE STREAMING

Viewport-adaptive streaming, also denoted as Viewport-Dependent Streaming, denotes streaming architecture that leverages viewport orientation prediction to stream high quality only in the area that are predicted to be attended by the user, and low quality elsewhere. Viewport-adaptive streaming extends DASH to perform viewport orientation adaptation in addition to the existing bandwidth adaptation. Viewport-adaptive streaming solutions usually exploits heterogeneous spatial quality encoding to deliver high quality only in a given area. As viewports represents only a small portion of total omnidirectional content (around 20 % of the spherical surface with today's HMDs), for the same bandwidth budget, the displayed viewport can have better average quality than when all the video is streamed with uniform quality (with viewport-independent streaming).

Viewport-adaptive streaming is needed because the naive strategy considering the streaming of only the attended viewport to the user is not conceivable. Indeed, with current video codecs, streaming only the attended viewport would require, for each user, the server to decode the video, extract the viewports and re-encode the viewport video stream. Such encoding process is hardly scalable and

Figure 2.9: Heterogeneous Quality Video

cost-efficient. Moreover, to avoid the simulator sickness [114], the motion-to-photon delay should be lower than 10 ms. To enable the streaming of only the attended viewport, the client needs to transmit the orientation of the user's head to the server, but the transmission delay of the orientation and then of the video bit-stream (i. e. the Round-Trip Time (RTT)), and the delay to decode/re-encode the video are, accumulated, usually greater than the advisable 10 ms.

The architecture of viewport-adaptive streaming is further described in Chapter 3. In this Section we mainly focus on different way to prepare the content to enable viewport-adaptive streaming.

2.6.1 *Heterogeneous Spatial Quality Encoding*

We denote by heterogeneous spatial quality video a video with non uniform spatial quality. Non uniform spatial quality means the quality of a group of pixels in the pictures vary depending on its spatial location. The video depicted in Figure 2.9 is an example of heterogeneous spatial quality video: the center of the picture has a higher quality than the edges of the picture.

In the context of omnidirectional video, as only a small portion of the video is displayed to the user, one can emphasize the quality in the part displayed to the user and decrease the quality elsewhere to reduce the total bit-rate of the video.

It is worth mentioning that even though the pixel information outside the user's viewport is less important, maybe even useless for a short video segment, it still needs to be delivered (even in low quality) to maintain the video interactivity. For instance, when the user suddenly changes her head position but no video content outside her current viewport has been sent in advance, the user will experience a continuous loss of information until the video content for the current viewport is received through the network. The latter would definitely decrease user's QoE. That is why, sending videos with heterogeneous quality (containing at least some content information for all the pixels of the video) is a good practice.

We distinguish two main approaches to generate omnidirectional videos with heterogeneous quality: one where the variation of the quality is done when the spherical video is projected into a plane [73, 74, 123, 162], and one where the encoding of the projected frames enables differentiating the quality within the frame [32, 49]. Readers should note that those two categories are not totally disjointed. We

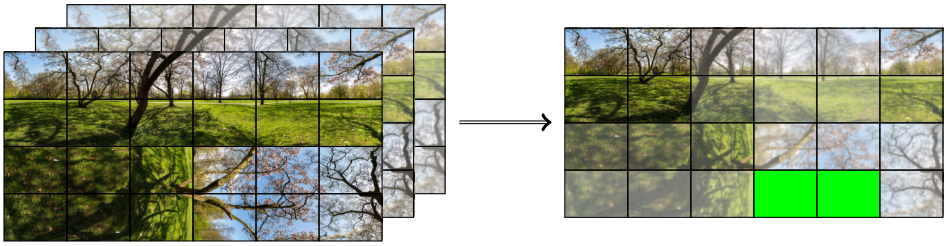


Figure 2.10: MCTS encoding and bit-stream recomposition for a single HEVC-compliant decoder

use them in this thesis to help distinguish different ways to generate videos with heterogeneous quality but some solution could fit in both categories depend on what step one considers more important.

Coding-Based Viewport-Adaptive Video Streaming

There exists two main approaches to generate video with heterogeneous quality at encoding time: the Quantization Parameter (QP) approach and the motion-constrained tile sets (MCTS) approach.

The QP approach changes the weight of the quantization steps during the encoding process to decrease the quantity of information stored outside the Region of Interest (RoI) and keep a high quantity of information inside the RoI. Lee, De Simone, and Ebrahimi [78] apply this idea to emphasize the quality in the area of a traditional video where users are predicted to focus their attention. They identify the RoI by using jointly audio and visual information.

The QP approach allows a fine grade tuning of how the information is distributed in the video pictures, but offers no choices to the client which can only get the video representations as the service provider has decided to prepare them.

The tile-based approach exploits the MCTS introduced by HEVC to spatially split a planar video into independently encoded rectangular regions [52, 75, 85, 113, 157, 158]. The key idea of tiling is to spatially split the video into a set of motion-constrained non-overlapping rectangular blocks, called tiles, which are encoded independently one from another [85]. By default tiles are just spatially independent but not temporally independent. Indeed tile has been originally introduced to allow parallel decoding of video pictures. Reference to blocks inside another tile of the current picture is forbidden (spatially independent) but reference to blocks in another tiles of a previously decoded picture is allowed (not temporally independent). The MCTS scheme adds extra encoding constraints to also forbid temporal prediction outside the tile boundaries and restrict the final in-loop filter to pixel belonging to the same tile. The MCTS constraints turn each tile into independently decodable.

It is possible to extract each tile from a MCTS bit-stream into its own track inside an ISO base media file format (ISOBMFF) media file [68]. Each track can then be downloaded independently from each other (and as well be stored into separate files). A single instance HEVC decoder cannot decode directly the N different tile tracks to get back the full picture. It is needed to merge the tile tracks together first. To perform this merging operation, a special track called the "extractor track" is used. The extractor track contains instructions to follow (called constructors) to merge the MCTS tracks. The merging operation generates a new ISOBMFF stream with a single HEVC-compliant track.

Encoding a video with the MCTS introduces a bit-rate overhead due to the tile headers[1] and due to losses in compression efficiency [26]. On the other hand, tiling allows more flexible streaming strategies. At the server side, a projected omnidirectional video (usually an equirectangular or a cube-map panorama) is encoded into multiple qualities. Using the bit-stream extractor mechanism [26, 75,

1 it is actually due to the slice headers, but in the context of spatial quality adaptation with tiles, in order to make the tiles independently downloadable, tiles are usually build in such a way that they each match a unique slice

115, 157], every tile is extracted, for each quality level, into a separate ISOBMFF bit-stream that can be independently downloaded. It is then possible at the client side to generate a decodable bit-stream, decodable by a single HEVC decoder, by merging together tiles with different quality. The constraints to generate this decodable bit-stream are: (*i*) each tile most fit into the tile grid and the slice grid of the new bit-stream, (*ii*) it is mandatory to include the initial metadata of the video (i. e. the extractor track used to re-write the bit-stream metadata) and (*iii*) the new bit-stream shall contain, for a given chunk, at most one version of each tile. Figure 2.10 presents on the left side the video prepared with the MCTS and the different quality levels, and on the right side a bit-stream recomposition with different quality level for each tile. The tiles in green are tiles without information (i. e. not downloaded). This approach, advocated in many studies [37, 48, 53, 98, 155, 157, 158], has been successfully implemented [37] and integrated into recent standards [52, 59, 93].

In 2014, an extension, denoted as DASH Spatial Relationship Description (SRD) [38], has been introduced into DASH to enable the streaming of individual tiles. DASH SRD allows a DASH server to announce the existence and the properties of the different tiled segments into the MPD, and enables tile level bit-rate adaptation at the client side.

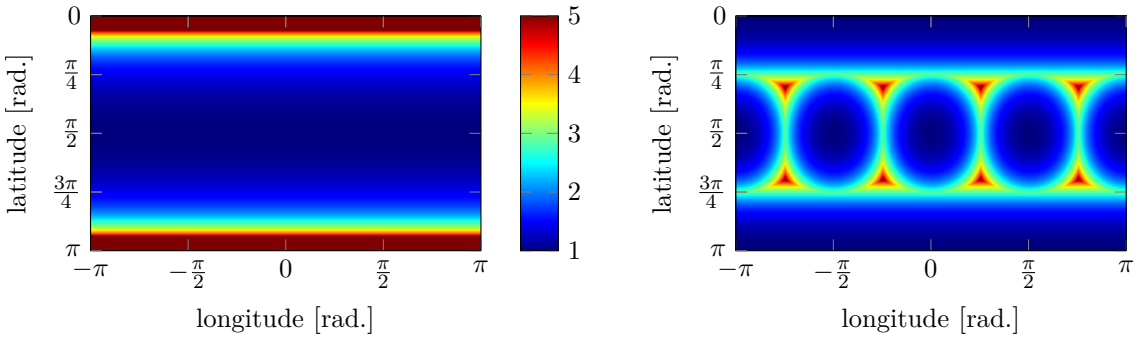
Le Feuvre and Concolato [75] and Concolato et al. [26] demonstrate the feasibility of tiled adaptive streaming with DASH SRD in the context of planar videos. However, no formal network evaluation is performed. To show the encoding efficiency of tiling, Zare et al. [157] have investigated the theoretical gains of tiling when applied to projected omnidirectional video frames. The authors have proposed a viewport-adaptive tiling approach which uses two video versions with high and low resolutions. The viewport tiles are transmitted in high resolution, whereas the regions outside the user's viewport correspond to tiles with low resolution. Petrangeli et al. [98] perform viewport-adaptive streaming over HTTP/2 on tiled videos. They evaluate their system using an Android phone over a Wi-Fi network, but they do not consider any delivery network. Ahmadi, Eltobgy, and Hefeeda [3] study a multicast DASH-based solution to stream live tiled omnidirectional video on a multicast Long Term Evolution (LTE) network. The system separates the users into different multicast groups, based on their network quality (not on their viewing direction). For a given tile, users in the same multicast group receive the same quality. Finally, Skupin et al. [115] present a solution to stream tiles with multiple resolutions in the context of omnidirectional video. They introduce a fast and coding efficient way to create frequent RAPs, but they do not evaluate any streaming scenario.

The MCTS approach offers a great level of flexibility to the client: if the user has a outlier behavior, the client can still select tiles that provide high quality in this direction.

Projection Based Viewport-Adaptive Video Streaming

Projection based heterogeneous quality gathers techniques that reduce the size of the encoded video by applying transformation on the omnidirectional signal before the encoding process. Two main strategies exists to reduce the video bit-rate after encoding[2] while generating a video with heterogeneous spatial quality: (*i*) Transforming the original projected planar picture into a new planar picture with the same resolution but applying a low-pass filter on some regions of the video. The filtering reduce the complexity of some regions of the video, allowing a more efficient compression by the video codec, and so reducing the video bit-rate, while keeping an equivalent quality than the original video in the zone where no filter was applied [19, 54]. (*ii*) Use a sphere-to-plane projection with non uniform spherical pixel sampling. With such projection, some region of the sphere have more pixel than others. By choosing carefully the projection parameters, it is possible to store on a lower resolution planar picture, a spherical picture with a region having as many pixel as in the original picture but with other regions having less pixels. The bit-rate gain is then mainly due to the decrease of the spatial resolution. Figures 2.11a and 2.11b depicts the pixel density ratio on the sphere for the equirectangular and the

2 compared to the bit-rate the video would have without using those techniques



(a) Equirectangular Projection

(b) Cube-Map Projection

Figure 2.11: Heatmap of pixel density on the sphere. To ease the reading, the spherical pixel density is projected on a planar picture using an equirectangular projection. The red areas represent zone on the sphere with a high density of pixel. The color scale between the two pictures is the same. It is truncated to 5 because for the equirectangular projection the density tend toward infinity at the poles.

cube-map projection. If the pixel density ratio is equal to r, its means that at this position on the sphere this projection generate r times more pixels than in the direction of $\vec{t}$.

Neither the equirectangular projection nor the cube-map projection directly enables viewport-adaptive video streaming, as neither of them emphasizes a given video region by compromising the quality of the remaining regions. Kuzyakov and Pio [74] have recently exploited the potential of another sphere-to-plane projection, i. e. the pyramid projection. They leverage a key property of this projection, i. e. its irregularity, to emphasize a particular region of the video (the Quality Emphasized Region (QER)). The sphere is put inside a pyramid and then projected onto each face of the pyramid. The part of the sphere mapped to the base of the pyramid is the QER. The QER is displayed in full resolution, whereas the quality of the video regions, mapped to the sides of the pyramid, decreases with the distance from the QER. This means that the best video quality corresponds to the QER, mapped to the base of the pyramid, whereas the worst quality corresponds to the region opposite the QER. Therefore, in order to emphasize the user's viewport, we need to apply a rotation of the spherical video to make sure that the viewport is projected onto the base of the pyramid. A variant of the pyramid projection called "truncated square pyramid" was evaluated by Zare, Aminlou, and Hannuksela [156] and compared to a tile-based encoding. They demonstrate that the truncated square pyramid is slightly more efficient regarding the streaming performances but requires more storage and more encoding time that the tile-based solution.

Facebook introduced the *cube-map offset projection* to perform such a projection based heterogeneous spatial quality encoding [73]. This transformation has been studied for the first time by Zhou, Li, and Liu [162] while they reverse engineered how the Facebook's Oculus Rift works. We [54] further studied this kind of projection by generalizing it to any projection, and we present some original results in Chapter 5 of this dissertation showing the correlation between spherical pixel sampling (or spherical pixel density) with visual quality.

2.7 VIEWPORT ORIENTATION PREDICTION

In the context of DASH streaming of traditional planar video, the prediction of the bandwidth that will be available to the client in the near future is a key information to perform an optimal adaptive bit-rate segment scheduling. In the context of omnidirectional video streaming, the fact (i) that most ($\simeq$80 %)

of the video is not displayed to the user and (*ii*) that resolution of the full content is very high, makes it very challenging to fit into the bandwidth budget while matching the low latency constraint.

Viewport orientation prediction is a key feature for omnidirectional video streaming as it may be used to maximize the downloaded bit-rate in the displayed part of the video (i. e. to perform viewport-adaptive streaming as described in Section 2.6). Of course bandwidth prediction is still important but stays similar to the bandwidth prediction in the context of traditional planar video streaming.

Petrangeli et al. [98] predict the position of the center of the viewport with a linear regression: the positions and the speeds are measured/estimated on the projected planar picture, not directly on the sphere. The next viewport center position on the 2D picture $\vec{\mathbf{p}}(t+\Delta T)$ in a period ΔT is estimated as $\vec{\mathbf{p}}(t+\Delta T) \simeq \vec{\mathbf{p}}(t) + \Delta T \frac{\vec{\mathbf{p}}(t)-\vec{\mathbf{p}}(t-\delta)}{\delta}$, with δ the viewport center velocity measurement step and t the current instant in time. In their work, they use this prediction on an equirectangular projection of the sphere. They do not predict the tilt of the viewport nor evaluate the accuracy of this viewport prediction, but they show that using such a simple prediction can substantially improve the performance of their adaptive streaming scenario.

Quan et al. [103] study also a linear regression and a weighted line regression to predict viewport orientations. They evaluated the accuracy of their prediction with different prediction window.

Bao et al. [10, 11] predict the viewport position using regressions on the past position time-series. They manipulate the orientation of the viewports using the Euler angles [83] yaw, pitch and roll. They shows that there exists a high short-term auto-correlation with the Euler angles, and a lower correlation between the three angles measured two by two. This means the viewport position can be predicted using a short-term time-series by performing a regression independently on each Euler angles. They decided to predict only the yaw and pitch angles, ignoring the roll angle, which is equivalent to predict the position of the center of the viewport ignoring the viewport tilt. They do not use any video/audio feature in the prediction. They study four regressions: a naive no movement regression, a linear regression, and two neural network regressions; and measured the prediction error.

Xu et al. [148] perform also a linear regression on the three Euler angles representing the viewport orientation. Unlike other cited work, they try to predict the full viewport orientation, not only its center. They study the forecast error over time.

2.8 STANDARDS AND INDUSTRIAL SOLUTIONS

When omnidirectional videos has emerged, the industry has immediately been active in studying potential standards, which could enable the whole chain of capturing, delivering, and consuming immersive data to share common tools. In the meantime, this innovation field has also been explored by start-up companies. We report in the following only the work that is directly related to this thesis, which means that capturing and consuming 360°, or work related to omnidirectional audio are typically ignored.

2.8.1 *Standards*

Many standard bodies have been active in starting studies related to Virtual Reality (VR) and in particular, to omnidirectional videos.

MPEG's ongoing work focuses on Omnidirectional MediA Format (OMAF) [100], which can be seen as a toolkit for omnidirectional video based on the ISOBMFF multimedia container, HEVC and DASH. It aims at the harmonization of VR video platforms and applications. The overall architecture that is under discussion at MPEG and in particular, in OMAF, is depicted in Figure 2.12. A scene is captured as a set of video signals (Bv). Before encoding (Ev), the content is projected and stitched onto

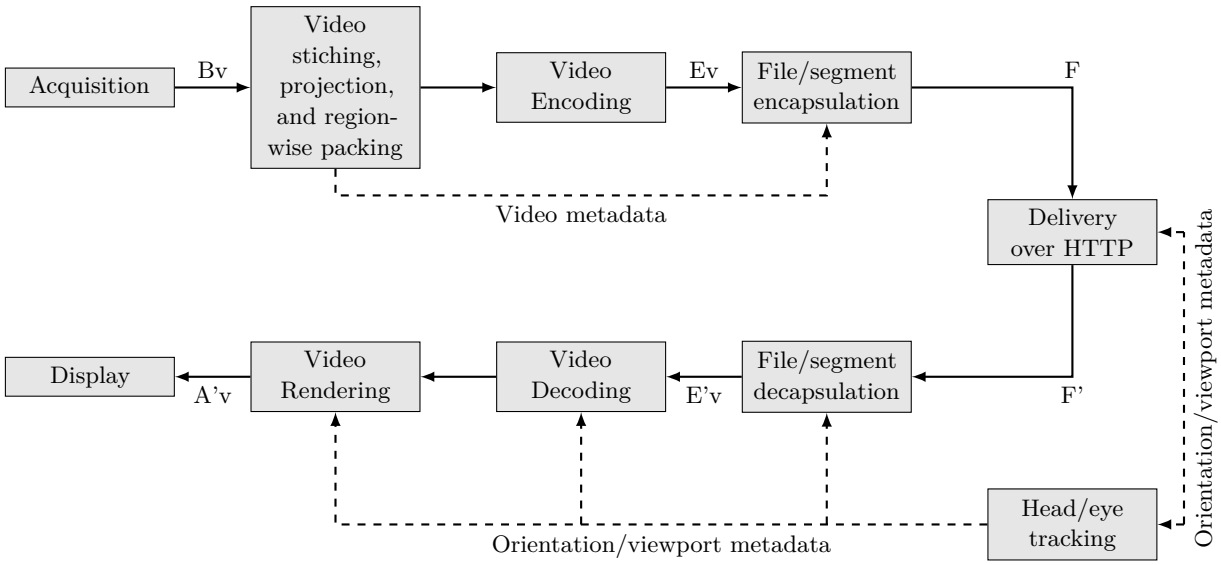


Figure 2.12: Overall architecture under discussion at OMAF (video part only)

the unit sphere, then a sphere-to-plane projection is applied to get a planar *projected* picture. In OMAF vocabulary and process, the sphere-to-plane projection is split into three steps: (*i*) the spherical image is projected onto a 3D object, usually the unit-sphere itself or a cube, with a gnomonic projection (also denoted as Dymaxion map or Fuller map when apply on a polyhedra); (*ii*) the image projected on the polyhedra (or on the sphere itself), is mapped on the plane. Each face of the 3D object is mapped to a predefined region of the plane. The regions and the mapping depend on the 3D object; (*iii*) The planar regions are *packed* into the final rectangular planar picture. The packing operation takes rectangular regions of the projected picture and can apply resampling, rotation (90°, 180° or −90°) or mirroring operation on it. The resulting rectangular piece of picture is then mapped to the final picture. The concept of *region-wise packing* is a core element of OMAF. OMAF defines the metadata needed to describe such operations. OMAF metadata can only describe a subset of sphere-to-plane projection. After encoding, the content is encapsulated into ISOBMFF segments (F) together with some metadata, which may be used for additional signalling for DASH clients. The segments are then delivered to the client. After downloading the file segments (F'), the client decapsulates the received stream (E'v) and extracts the corresponding metadata. Finally, the video is decoded, projected back to the sphere and displayed at the client device.

In OMAF, two projections are currently considered: the equirectangular and the cube-map projections. As previously mentioned, one major component of OMAF is the concept of region-wise packing [100, 117]. It is typically used to circumvent the main weaknesses of the projections: in particular the oversampling at the poles in the equirectangular projection. We show the main idea behind region-wise packing in Figure 2.13. Note that region-wise packing can also be used to describe videos with a preferred viewport orientation. OMAF introduces the concept of region-wise quality ranking to indicate that some region of a representation is available at higher quality than other. The quality ranking provide the possibility to fully order the regions of multiple representations of the same content. The region-wise quality ranking metadata can be spread into the DASH MPD file. With this information, the client can select the representation which maximize the quality in the prediction viewport positions.

The HEVC-based viewport-dependent profile of the current OMAF version (version 1) supports only video encoded with HEVC up to the Main 10 profile, Main tier, Level 5.1 [116]. In other words, to be OMAF compliant with current existing profiles, the video should not exceed a 4K spatial resolution at 60 fps. This resolution constraint implies that if one wants to provide a high effective resolution

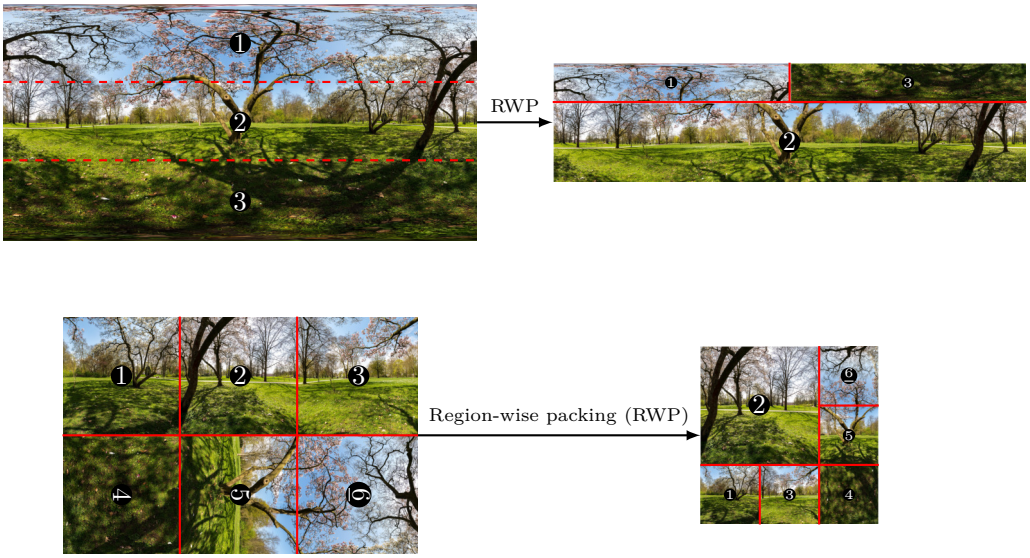


Figure 2.13: Illustration of Region-wise packing for the equirectangular (top) and the cube-map (bottom) projections [117].

inside the user's viewports with the 4K decoding constraint, smart region-wise packing/sphere-to-plane projection such as the one proposed by Zare, Aminlou, and Hannuksela [155] has to be considered.

OMAF specifies the delivery of omnidirectional video over DASH. It enables several approaches, including the following:

- Using the *viewport-independent* profile defined in OMAF, the full video is sent to the end-users regardless of the user's viewing orientation. In that case, regular DASH server and client can be used.
- Using the *viewport-dependent* profile, in which the user selects an adaptation set based on the viewing orientation (or its prediction). Several adaptation sets need to be available on the server, each for a specific user viewing orientation. The main idea here is to have, in each adaptation set, the full omnidirectional video where one region is with an enhanced quality/resolution compared to the rest of the omnidirectional video.
- The viewport-dependent profile of OMAF enables encoding MCTSs at several qualities, each MCTS encapsulated in its own Representation. Each set of MCTS Representations covering the same sphere region are encapsulated into the same Adaptation Set. Additionally, an extractor track, offered as its own Representation, contains instructions on how to merge selected MCTSs to a single video bitstream.
- The viewport-dependent profile of OMAF also enables encoding MCTSs at several resolutions and combining MCTSs originating from different resolutions into a single video bitstream [155]. In this scenario, an extractor track is required for each distinct viewing orientation.

Annex D of the OMAF standard [59] provides further information on viewport-dependent streaming schemes enabled by OMAF.

The 3rd Generation Partnership Project (3GPP) has also started a working group to work on the topic of VR in mobile networks [140]. The first reports have provided a complete overview of the challenges that need to be addressed for a high-quality delivery of VR in mobile network, but, so far, to the best of our knowledge, no standard has been actually defined in this standard body. The main document that gives an idea of what the 3GPP could discuss in the next couple of years is shown in another technical

report [102]. This document provides some typical QoE metrics for on-demand omnidirectional video streaming.

2.8.2 *Industry*

Start-ups

The main start-up that has been visible in the emerging market of VR is *Tiledmedia* [132]. The fundamental technology that they use is the MCTS *tiling*. Their main idea is as follows. At any time, they deliver two video streams: (*i*) one video at low resolution and low quality (as a backup for abrupt head orientation changes), this video may not be encoded with tile and (*ii*) only the tiles that are viewed in the viewport at very high quality in order to get an excellent QoE.

One of the claims of Tiledmedia is that the motion-to-high-quality delay is a handful of frames, which means that, in case of abrupt head movement, the delay to send the tiles at the highest quality for the new viewport is around 30 ms. To do so, Tiledmedia encodes the high-quality video in two distinct streams: (i) one video stream with a long RAP interval, sent in the case of relatively stable head orientations and (ii) one video stream with a ultra-short RAP interval (typically two frames only). Thus, this video can be played at any time once it is stored in the buffer. In case of brutal head movement, the system delivers the missing high-quality tiles in its short RAP interval version, so that the client can display the high-quality tiles upon reception of the tiles, and, at the end of the video segment, the video version with long RAP interval is delivered to ensure an idle delivery.

The system implemented by Tiledmedia is thus especially efficient regarding the motion-to-high-quality. It requires however to encode each video three times (one low-quality and two high-quality) with the MCTS feature of HEVC. Unfortunately, this feature is not implemented in fast open-source video encoders today.

Part II

ARCHITECTURE

3

OMNIDIRECTIONAL VIDEO STREAMING ARCHITECTURE: EVALUATION OF DIFFERENT PROJECTIONS

3.1 INTRODUCTION

To deliver omnidirectional video content on the Internet, the content providers have to deal with a problem of bandwidth waste: What is displayed on the device, denoted as *viewport*, is only a fraction of what is downloaded, which is an omnidirectional view of the scene (see Section 2.3.2 for more details on viewports). This bandwidth waste is the price to pay for interactivity. To prevent *simulator sickness* [87] and to provide good QoE, the vendors of HMDs recommend that the multimedia systems react to head movements as fast as the HMD refresh rate. Since the refresh rate of state-of-the-art HMDs is 120 Hz, the whole system should react in less than 10 ms. This delay constraint prevents the implementation of traditional delivery architectures where the client notifies a server about changes and awaits for the reception of content adjusted at the server. Instead, in the current VR video delivery systems, the server sends the full omnidirectional stream, from which the HMD extracts the viewport in real time, according to the user head movements. Therefore, the majority of the delivered video stream data are not used.

Let us provide some numbers to illustrate this problem. The viewport is defined by a device-specific viewing angle (typically 110°× 90°), which delimits horizontally the scene from the head direction center, called viewport center. To ensure a good immersion, the pixel resolution of the displayed viewport is high, typically 4K. So the resolution of the full omnidirectional video is at least 16K. In addition, the immersion requires a video frame rate on the order of the HMD refresh rate, so typically around 100 fps. Overall, high-quality omnidirectional videos combine both a very large resolution (up to 12K) and a very high frame rate (up to 100 fps). To compare, the bit-rate of videos with resolution 8192×4096 pixels (8K) and 60 fps encoded using HEVC is around 100 Mbit s^{-1} [96].

We propose in this Chapter a solution where, following the same principles as in rate-adaptive streaming technologies, the server offers multiple *representations* of the same omnidirectional video. But instead of offering representations that only differ by their bit-rate, the server offers here representations that differ by having a QER: a region of the video with a better quality than the remaining of the video. Our proposal is a *viewport-adaptive streaming system* and is depicted in Figure 3.1. The QER of each video representation is characterized by a *Quality Emphasis Center (QEC)*, which is the centroid of the QER and represents a given viewing direction in the spherical video. Around the QEC, the quality of the video is maximum, while it is lower for video parts that are far from the QEC. Similarly as in DASH, the video is cut into segments and the client periodically runs an *adaptive algorithm* to select a representation for the next segment. In a viewport-adaptive system, clients select the representation such that the bit-rate fits their receiving bandwidth and the QEC is closest to their viewport center.

This viewport-adaptive omnidirectional streaming system has three advantages: (*i*) the bit-rate of the delivered video is lower than the original full-quality video because video parts distant from the QEC are encoded at low quality. (*ii*) When the end-user does not move, the viewport is extracted from the highest quality part of the spherical video (supposing that the user has not an outlier orientation and so that a QEC exists near its current position). And (*iii*) when the head of the end-user moves, the device can still extract a viewport because it has the full spherical video. If the new viewport center is far from the QEC of the received video representation, the quality of the extracted viewport is lower but this degradation holds only until the selection of another representation with a closer QEC.

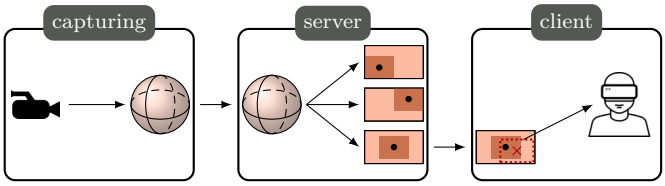


Figure 3.1: Viewport-adaptive 360-degree video delivery system: The server offers video representations for three QERs. The dark brown is the part of the video encoded at high quality, the light brown the low quality. The viewport is the dotted red rectangle, the viewport center the cross

The remainder of the Chapter is organized as follows. First, we present our viewport-adaptive streaming system, and we show how it can be integrated into MPEG – Immersive (MPEG-I) OMAF [117]. Second, we address the choice of the geometric layout into which the spherical video is projected for encoding. We evaluate several video quality arrangements for a given geometric layout and show that the cube map layout with full quality around the QEC and 25 % of this quality in the remaining faces offers the best quality of the extracted viewport. Third, we study the required video segment length for viewport-adaptive streaming. Based on a dataset of real users navigating the omnidirectional videos, we show that head movements occur over short time periods, hence the streaming video segments have to be short enough to enable frequent QER switches. Fourth, we examine the impact of the number of QERs on the viewport quality and we show that a small number of (spatially-distributed over the sphere) QERs suffices to get high viewport quality. Finally, we introduce a tool (released as open source), which creates video representations for the proposed viewport-adaptive streaming system. The tool is highly configurable: from a given omnidirectional video, it allows any arrangement of video quality for a given geometric layout, and it extracts the viewport from any viewport center. This tool, used in all our evaluations, is further detailed in Chapter 7.

In this Chapter, we compare our viewport-adaptive streaming architecture to the solutions existing at the time this work was realised.

3.2 BACKGROUND AND RELATED WORK

We introduce the necessary geometric concepts for spherical videos, and discuss prospective architecture proposals for navigable omnidirectional video delivery.

3.2.1 *Geometric Layouts for 360-degree Videos*

As described in Section 2.3.3, omnidirectional videos are usually projected onto planar rectangular images in order to exploit existing video encoders. In this Chapter, we consider the four projections that has been discussed for omnidirectional video encoding [152]. These layouts are depicted in Figure 3.2. From the images that are projected on an *equirectangular* panorama, a *cube map*, and a *rhombic dodecahedron*, it is possible to generate a viewport for any position and angle in the sphere without any information loss [47, 91]. However, some pixels are over-sampled (a pixel on the sphere is projected to a pair of pixels in the projected image). This is typically the case for the sphere pole when projected on the equirectangular panorama (see Section 2.6.1). This over-sampling degrades the performance of traditional video encoders [152]. On the contrary, the projection into a pyramid layout causes under-sampling: some pairs of pixels on the sphere are merged into a single pixel in the projected image by interpolating their color values. This under-sampling cause distortion and information loss in some extracted viewports. Previous work regarding projection of spherical videos into different

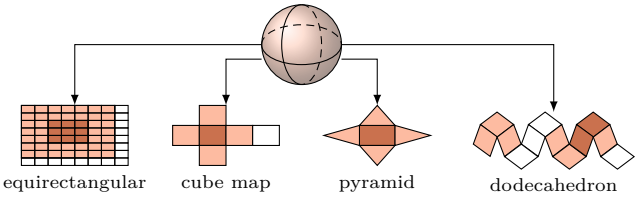


Figure 3.2: Projections into four geometric layouts

geometric layouts focuses on enabling efficient implementation of signal processing functions [67] and improving the video encoding [135].

Our contributions. We propose to leverage the geometric structure of the layouts to implement a video encoding based on QER. Each geometric layout is characterized by a number of *faces* (*e.g.*, 6 for the cube map, 12 for the dodecahedron) and a given *central point* (which corresponds to a position on the sphere). From the given central point and layout, our idea is to rotate the 3D geometric object to align the center of one of its face with the given central point, this face is then denoted as front face, and then to encode the front face in full quality while the quality of other faces is reduced. To our knowledge, such idea has not been studied yet. Another originality of our work is that we measure QoE by measuring the quality of several extracted viewports instead of the full projected video.

3.2.2 *Personalized Viewport-Only Streaming*

An intuitive idea to address the problem of resource waste due to the delivery of non-displayed video data is to stream only the part of the video that corresponds to the viewport. This solution however does not enable fast navigation within the omnidirectional video: When the client moves the head, the viewport orientation changes, requiring a new viewport to be immediately displayed. Since the device has no knowledge about other parts of the spherical video, it has to notify the server about the head movement and wait for the reception of the newly adjusted viewport. As seen in other interactive multimedia systems [24], this solution cannot meet the 10 ms latency requirement in the standard Internet, even with the assistance of CDN. In addition, this solution requires the server to extract a part of the video (thus to spend computing resources) for each client connection.

Our contributions. In our system, the server always delivers the full video, but it has different versions of this video depending on the QER (characterized by its QEC). The client device selects the right representation and extracts the viewport. The storage requirements at the server side increase but all the processing is done at the client side (representation selection and viewport extraction). This idea matches the adaptive delivery solutions that content providers have recently adopted (*e.g.* DASH), trading client-personalized delivery for simple server-side management operation.

3.2.3 *Tiling for Adaptive Video Streaming*

To deal with the cases of end-users consuming only a fraction of the video (navigable panorama [48, 113, 142] and large-resolution video [75]), the most studied delivery solution leverages the concept of *tiling* (see Section 2.6.1). In a short paper, Ochi et al. [94] have sketched a tile-based streaming system for omnidirectional videos. In their proposal, the spherical video is mapped onto an *equirectangular* video, which is cut into 8×8 *tiles*. More recently, Hosseini and Swaminathan [53] proposed a *hexaface sphere*-based tiling of a omnidirectional video to take into account projection distortion. They also present an approach to describe the tiles with MPEG DASH SRD formatting principles. Quan et al. [103] also propose the delivery of tiles based on a prediction of the head movements. Zare et al. [158] evaluate

the impact of different tiling scheme on the compression efficiency and on the transmission bit-rate saving.

A tile-based adaptive streaming system provides the same features as our proposed system regarding navigability (the clients get the full video), bandwidth waste reduction (the video at low quality for non-viewport part) and QoE maintenance (the downloaded video is at full quality near the viewport center). It has however several weaknesses. First, the client has to first reconstruct the video from independent tiles before the viewport extraction can take place, which might cause additional latency. Second, the more tiles there are, the less efficient the video encoding is due to the tile independence [113]. There is so a limit on how fine-grained tiling can be useful. For instance, Youvalari et al. [151] concluded that 6×3 tiles appears to be the best trade-off for viewport adaptive streaming of omnidirectional videos with equirectangular projection. Third, when the number of tiles is greater than the number of quality-emphasized versions, the management at the server is heavier because the number of files is larger. For example, a typical 8×8 tiling offered at six quality levels contributes to having 384 independent files for each video segment, and this can result in larger MPD files (or manifest files). Finally, the management at the client side is heavier. For each tile, the client should run a representation selection process to decide witch quality to fetch from the server.

Our contributions. In our system, the server prepares n QER-based videos, each of them being a pre-processed set of tile representations. Each QER-based video is then encoded at k *global* quality levels. The main advantages include an easier management for the server (fewer files hence a smaller MPD file), a simpler selection process for the client (by a distance computation), and no need for re-constructing the video before the viewport extraction.

3.2.4 *QER-Based Streaming*

A omnidirectional video provider (*Facebook*) has released detailed the implementation of its delivery platform [74]. The spherical video is projected onto a pyramid layout from up to 30 central points to generate a set of video representations. Since the front face of pyramid projection has a better image quality than the other faces, the system is in essence similar to our concept of QER. The end-users periodically select one of the representations based on their viewport center. This implementation suggests that Facebook was at the time considering the extra-cost of generating and storing multiple QER-based representations of the same video worth the bandwidth savings and enhanced system usability. However, as seen in Section 3.4, the pyramid projection is not the best regarding the viewport quality. Moreover, the system uses the same video quality on each face, which is less efficient than our proposal. Finally, the impact of the video encoding on the solution is not given.

Lee, De Simone, and Ebrahimi [78] studied in another context the coding of a regular video with a QER. The QER is generated near the area that is the most likely to attract gazes. They do not propose to generate different representations with different QERs.

Our contributions. Our approach is based on the same idea of offering multiple QER-based video representations. However, we provide a complete study of our system with the additional distinction of having varying quality across the geometrical layout. Moreover, our study includes an evaluation of several geometric layouts, an analysis of the best segment duration, an analysis of the best number of QERs, and a step towards integration into MPEG DASH.

3.3 SYSTEM ARCHITECTURE

This section describes the system architecture of the proposed navigable omnidirectional video delivery framework.

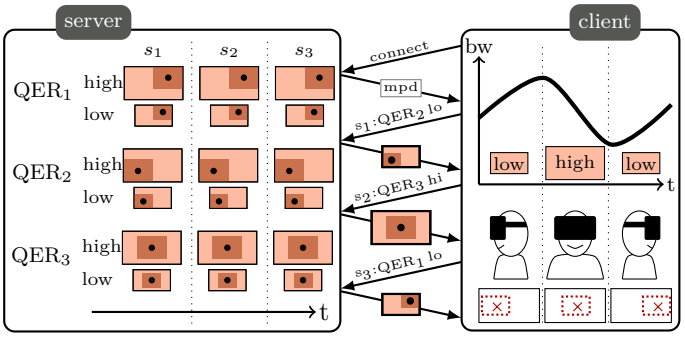


Figure 3.3: Viewport-adaptive streaming system: the server offers 6 representations (3 QERs at 2 bit-rates). The streaming session lasts for three segments. The client head moves from left to right, the available bandwidth varies. For each segment, the client requests a representation that matches both the viewport and the network throughput.

Server. The server takes as an input an omnidirectional video in equirectangular format and transforms each frame into the desired geometrical layout. Then, it creates n different video versions, each with a different QER and encoded into k different bit-rates (see Figure 3.3). The server splits all such encoded videos into segments, which are classified in $n \times k$ representations (based on their respective bit-rate and QER), enabling clients to regularly switch from one representation to another. The video quality around the QEC is the highest, while the remaining part is encoded at lower quality.

Client. Over time the viewer moves the head and the available bandwidth changes. Current HMDs record changes in head orientation through a rotation. Head movements modify the viewport orientation, requiring a new viewport to be displayed. State-of-the-art HMDs can perform the viewport extraction from a full omnidirectional picture [141]. The client periodically sends a request to the server for a new segment in the representation that matches both the predicted future viewport orientation and the available throughput.

Adaptation algorithm. Similarly to DASH, the client runs an adaptation algorithm to select the video representation. It first selects the QER of the video based on the angular distance between the viewport center and the QECs of the available QERs. This is an important addition to the DASH bit-rate adaptation logic, since the QER position determines the quality of the extracted viewports, the actual video displayed to the user. After the QER selection, the client chooses the video representation characterized by this QER and whose bit-rate fits with the expected throughput for the next x seconds (*i.e.*, x being the segment length). The server replies with the requested video representation, from which the client extracts the viewport displayed on the HMD, as illustrated in Figure 3.3.

Rate-adaptive streaming systems are based on the assumption that the selected representation will match the network conditions for the next x seconds. Rate adaptation algorithms are developed [81, 131] to reduce the mismatch between the requested bit-rate and the throughput. In our proposal, the adaptation algorithm should also ensure that the viewport centers will be as close as possible to the QEC of the chosen QER during the x next seconds. In this paper, we implement a simple algorithm for QEC selection: we select the QEC that has the smallest angular distance to the viewport center at the time the client runs the adaptation algorithm. Similarly as for bit-rate adaptation, we expect new viewport-adaptive algorithms to be developed in the future to better predict the head movement and select the QEC accordingly. In their recent paper, Quan et al. [103] have made a first study where they show that a simple linear regression algorithm enables an accurate prediction of head movements for short segment size (see Section 2.7 for more detail on viewport prediction related-work).

Video segment length. A video segment length determines how often requests can be sent to the server. It typically ranges from 1 s to 10 s. Short segments enable quick adaptation to head movement

```
<?xml version="1.0"?>
<MPD>
  <Representation id="1" qec="90,60" bandwidth="9876" width="1920" height="1080" frameRate="30">
   <EssentialProperty schemeIdUri="urn:mpeg:dash:vrd:2017" value="0,0">
   <SegmentList timescale="1000" duration="2000">
   ...
  </Representation>
 </AdaptationSet>
</MPD>
</xml>
```

Listing 1: Extensions of MPD file

and bandwidth changes, but they increase the overall number of segments and results in larger manifest files. Shorter segments also increase the network overhead due to frequent requests, as well as the network delay because of the round trip time for establishing a TCP connection. Longer segments improve the encoding efficiency and quality relative to shorter ones, however they reduce the flexibility to adapt the video stream to changes. We discuss segment length and head movement in Section 3.4.2 based on a dataset.

Extending the MPD file. To implement the proposed viewport-adaptive video streaming, we extended a DASH MPD file with new information, as illustrated in Listing 1. Each representation contains the `coordinates` of its QEC in degrees, besides the parameters that are already defined in the standard [60]. Those coordinates are the two angles of the spherical coordinates of the QEC (latitude and elevation), ranging respectively from $-180°$ to $180°$ and from $-90°$ to $90°$. All representations from the same adaptation set should have the same reference coordinate system. The `@schemeIdUri` is used to indicate some extra information on the video such as the video source id and the projection type. The projection type is used by the client to determine if it knows how to extract viewports from this layout. The OMAF video format, newly introduce by MPEG-I, contains all the metadata needed to extract viewports from the video downloaded by the client. It contains description of the projection used, how the frame was packed (i. e. how faces of the geometrical object used for the projection are mapped on the planar picture), and the rotation applied on the sphere before the projection. It also introduce the notion of region-wise quality ranking which provides a superset of the functionality needed to advertise QERs.

3.4 SYSTEM SETTINGS

The preparation of omnidirectional videos for viewport-adaptive streaming relies on multiple parameters. We distinguish between global parameters (the number of QERs, the number of representations, the segment length and the geometric layout i. e. the projection map) and local (*per representation*) parameters (the target bit-rate, the number of different qualities in a representation, the quality arrangement of different faces of a geometric layout). We will not be comprehensive regarding the selection of all these parameters here. Some of them require a deeper study related to signal processing, while others depend on business considerations and infrastructure investment. In this paper, we restrict our attention to three key questions: What is the best geometric layout to support quality-differentiated omnidirectional video? What is the best segment length to support head movements, while maintaining low management overhead? What is the best number of QERs n to reduce the induced storage requirements, while offering a good QoE? To answer these three questions, we have developed a software tool and used a dataset from a real VR system.

Dataset. We graciously received from *Jaunt, Inc* a dataset recording the head movements of real users watching omnidirectional videos. The dataset is the same as the one used by Yu, Lakshman, and Girod [152]. It comprises eleven omnidirectional videos that are ten seconds long. These videos represent the typical content for such VR systems. The dataset contains the head

movements of eleven people who were asked to watch the videos on a state-of-the-art HMD (Occulus Rift DK2). The subjects were standing and they were given the freedom to turn around, so the head movements are of wider importance than if they were asked to watch the video while sitting. Given the length of the video and the experimental conditions, we believe that the head movements thus correspond to a configuration of wide head movements, which is the most challenging case for our viewport-adaptive system. Yu, Lakshman, and Girod [152] studied the most frequent head positions of users. We are interested here in head movements during the length of a segment.

Software. We have developed our own tool to manipulate the main concepts of viewport-adaptive streaming. Since the code is publicly available,[1] the software can be used to make further studies and to develop real systems. The main features include projection from one sphere-to-plane projection onto any others, face-based quality adjustment for geometric based projection, and viewport extraction, and are further detailed in Chapter 7. In this Chapter we used the version of the software of September 2016. The encoding is based on *ffmpeg* use the default *libx265* rate control algorithm with default configuration.

3.4.1 *Geometric Layout*

We report now the experiment of measuring the video quality of viewports, extracted from omnidirectional videos projected onto various geometric layouts and with various face quality arrangements. We used two reference videos.

- *The original equirectangular video at full quality (highest bit-rate offered by youtube for the tested video)*:[2] We extract viewports at 1920×1080 pixels resolution from this 4K equirectangular video. Those viewports are the reference (original) video used to assess the objective video quality.
- *The same equirectangular video re-encoded at a target bit-rate*. It is what a regular delivery system would deliver for the same bit-rate budget (here 6 Mbit s^{-1} being 75 % of the original video bit-rate). We re-encoded the original full-quality video with HEVC by specifying this bit-rate target. We call it *uniEqui* to state that, in this video, the quality is uniform.

The videos are encoded using the 360Transformation software introduced previously in its version of September 2016 and described in Chapter 7.

The performance of the layout can be studied with regards to two aspects: (*i*) *the best viewport quality*, which is the quality of the extracted viewport when the viewport center and the QEC perfectly matches, (*ii*) and the *sensitivity to head movements*, which is the degradation of the viewport quality when the angular distance between the viewport center and the QEC increases. To examine both aspects, we select one QEC on the spherical video. We chose one angular distance d that will vary from 0 rad to π rad. We extract a ten seconds long viewport video, without any tilt, at distance d from the QEC, with viewports at constant orientation during the ten seconds, at the same spherical position on the original equirectangular video and on the tested video. We used two objective video quality metrics to measure the quality of the extracted viewport compared to the original full quality viewport: MS-SSIM [143] and PSNR (see Section 2.4.2 for more detail about those metrics). Since we compare several encoded versions of the *same* viewport against the original, these well-known tools provide a fair performance evaluation of viewport distortion. We perform multiple quality assessment (typically forty) at the same distance d, but at different positions selected randomly with a uniform distribution, and average the result.

We represent in Figure 3.4 the video quality (measured by MS-SSIM) of the viewport that is extracted from our quality-differentiated layouts (equirectangular panorama with 8×8 tiles, cube-map, pyramid, and dodecahedron). We also represent by a thin horizontal line the video quality of the same viewports

1 https://github.com/xmar/360Transformations/tree/master/transformation
2 https://youtu.be/yarcdW91djQ

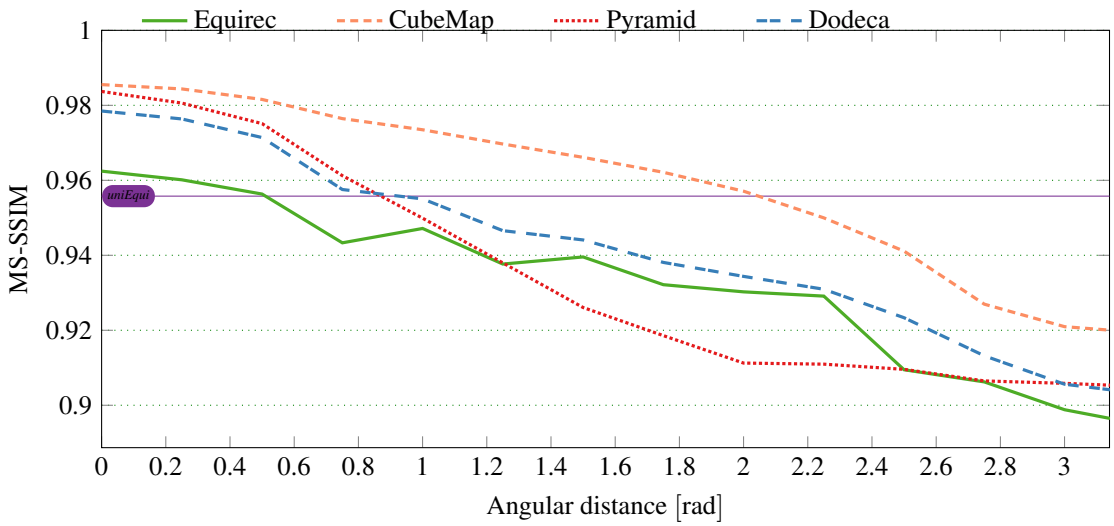


Figure 3.4: Average MS-SSIM depending on the distance to the QEC for the four geometric layouts. Global bit-rate budget 6 Mbit s^{-1}

extracted from the *uniEqui* layout (it does not depend on the distance since the quality is uniform). For each geometric layout, we have tested numerous quality arrangements with respect to the overall bit-rate budget. We selected here the "best" arrangement for each layout out of the tested combinaisons. The bit-rate allocation per faces was performed as follow: (*i*) we set a weight br_i for each face of the projection. Here the weight is equal to $br_i = 1$ for faces in the QER and $br_i = 0.25$ for the others; (*ii*) we fixe a total bit-rate target B. Here $B = 6\,\text{Mbit}\,\text{s}^{-1}$; (*iii*) we compute the bit-rate target B_i for each face i with the following formula: $B_i = \frac{B \cdot br_i}{\sum_i br_i}$. For the cube-map, the QEC is located at the center of a face and the QER contains only this face. It means that the front face bit-rate target is $2.667\,\text{Mbit}\,\text{s}^{-1}$ and the bit-rate target for the other faces is $0.667\,\text{Mbit}\,\text{s}^{-1}$. The resolution of each cube face is 960×960. For the pyramid the the QEC is at the center of the base face and the QER contains only this face. The resolution of each face is 960×960. The equirectangular tiled representation has 5×5 tiles in the QER. Each tile has the resolution 480×240. Finaly the dodecahedron has three face in the QER and each face has resolution 480×480.

The projection on a cube-map appears to be the best choice for the VR provider. The quality of the viewport when the QEC and the viewport center matches ($d = 0$ rad) is above 0.98, which corresponds to imperceptible distortion relative to the full quality video. For all layouts, the quality decreases when the distance d increases but the quality for the cube-map layout is always the highest. Note that the pyramid projection (the layout chosen by Facebook [74]) is especially sensitive to head movements. The viewports extracted from a cube-map projection has a better quality than those extracted from the *uniEqui* when their distance to the QEC is lower than 2 rad, while the other layouts viewports have a better video quality only for distance to the QEC lower than 1 rad. We study next the interplay between this distance, the segment length and the number of QECs.

3.4.2 *Segment Length*

The segment length is a key aspect of viewport-adaptive streaming. Long segments are easier to manage and better for video encoding compression efficiency, but short segments enable fast re-synchronisation to head movement. With respect to Figure 3.4, the segment length should be chosen such that the distance between the viewport center and the QEC are rarely higher than $\frac{\pi}{2}$ rad. Indeed, if this angular

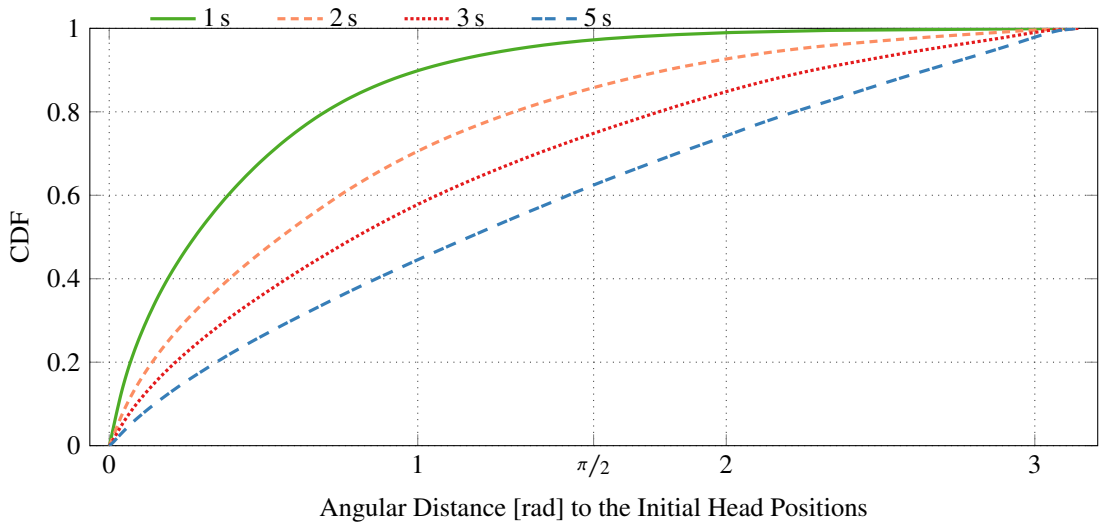


Figure 3.5: CDF of the time spent at distance d from the head position on the beginning of the segment, for various segment lengths.

distance is rarely higher than $\frac{\pi}{2}$ rad, with the tested cube-map scenario users would rarely observe an average viewport quality lower than for the *uniEqui* representation.

Given the dataset, we show the distribution of head movements for various segment lengths in Figure 3.5. For each video and person watching it, we iterate over all timestamps in the datatset and consider consecutively those timestamps as the starting time of a video segment, *i.e.,* the time at which the client select the next QER. Then, for each iterated timestamp we measure the angular distance between this initial head position and every viewport center during the next x seconds, where x is the segment length. In Figure 3.5, we show the Cumulative Density Function (CDF) of the time spent at a angular distance d from the initial head position. For instance, for $x = 5$ s, the blue curve cross the point $(1.5, 0.6)$, which means that, on average, users spend 60 % of their time with a viewport center at less than 1.5 rad from the viewport center on the beginning of the segment.

Our main observation is that viewport-adaptive streaming requires short segment lengths, typically smaller than 3 s. Indeed, for a segment length of 5 s, users spend on average half of their time watching at a position that is more than 1.3 rad away from the initial head position, which results in a degraded video quality. A segment length of 2 s appears to be a good trade-off: 92 % of users never diverged to a head position that is further than 2 rad away from the initial head position, and users can experience the full video quality three quarters of the time (head distance lesser than 0.7 rad). Please recall that our dataset captures a challenging experiment for our system. We can expect narrower head movements, and thus longer possible segment lengths, for sitting users and longer videos. Note also that these results are consistent with the head movement prediction from Quan et al. [103], who showed that prediction accuracy drops for time periods greater than 2 s.

3.4.3 *Number of QERs*

The number n of QERs represents another key trade-off. The more QERs there are, the better the coverage of the spherical video is, and thus the better the viewport quality will be due to a better match between the QEC and the viewport center. However, increasing the number of QERs also means increased storage and management requirements at the server (and a longer MPD file).

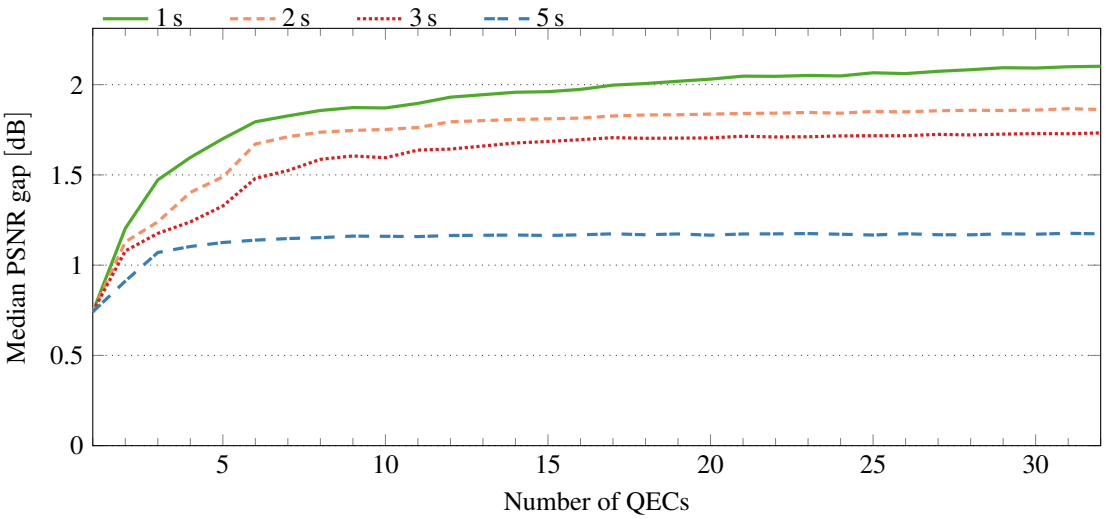


Figure 3.6: Median PSNR gap between the viewports of the cube-map layout and the *uniEqui* depending on the number of QERs. Bit-rate: 6 Mbps

We represent in Figure 3.6 the median PSNR difference between the viewport extracted from the cube-map layout and the same viewport extracted from the *uniEqui* layout with the same overall bit-rate budget. In other words, we measure for each viewport position of a user the PSNR in the viewport extracted in selected cube-map representation and the PSNR in the viewport extracted in the *uniEqui* representation, and we compute the gap (the difference) between the two PSNR. Then we compute the median value of all those PSNR gaps to get on point on the figure. To modify the number of QERs, we set a number n, then we determined the position of the n QECs using the Thomson positioning problem [106]. For each head position in the dataset, we computed the distance between the viewport center and the QEC that was chosen at the beginning of the segment and we computed the viewport quality accordingly.

The best number of QERs in this configuration is between 5 and 7. The gains that are obtained for higher number of QERs are not significant enough to justify the induced storage requirements (in particular not 30 QERs as in the Facebook system [74]). Having multiple QERs provides higher quality gains for short segments, due to the better re-synchronization between the QERs and the viewport centers. Note that a significant part of these gains stems from the cube-map layout.

3.5 CONCLUSION

We have introduced in this Chapter a viewport-adaptive streaming architecture for navigable omnidirectional videos. Our system aims at offering both interactive highs-quality service to HMD users with low management for VR providers. We studied the main system settings of our framework, and validated its relevance. We emphasize that, with current encoding techniques, the cube-map projection for two seconds long segment and six QERs offers the best performance. This Chapter opens various research questions and some of them will be studied in next Chapters: Chapters 5 and 6 explore how QER can be generated and how bit-rate should be allocated within the spherical picture. The following Chapter, Chapter 4, proposes and evaluates an upgrade of this viewport-adaptive streaming architecture to stream multi-viewpoint omnidirectional videos.

4

MULTI-VIEWPOINT OMNIDIRECTIONAL VIDEO STREAMING

4.1 INTRODUCTION

The current immersive multimedia services offering omnidirectional video are typically designed to provide VR experience with three Degrees of Freedom (3DoF). A HMD can choose the portion of the spherical content to view by rotating the head to a specific direction. Yet, to enable full immersion inside a VR scene, head rotation alone is not sufficient. The ability to perform translational movements inside the content is also required [44]. The user can then fully navigate the scene by virtually moving (walking) within it. VR applications allowing rotations and translations inside a virtual scene are referred to as six Degrees of Freedom (6DoF) applications. Yet the implementation of a continuous 6DoF VR application, i.e., *free-viewpoint*, without restriction in translations is still an open challenge [55].[1]

A first implementation of 6DoF VR applications consists in restricting the navigation to only predefined positions in space. We denote by *Multi-ViewPoint (MVP) omnidirectional video* a 6DoF VR application where the scene is offered at some finite number of predefined *viewpoints*, *i.e.* positions from which the scene can be viewed. To allow changes of viewpoint, the same scene has to be simultaneously captured by multiple omnidirectional cameras located at different positions in space.[2] *Google Earth VR* [57] is an example of such a MVP 6DoF VR application allowing the navigation into a virtual scene by changing the viewpoint (see Figure 4.1). However, it is to date restricted to static pictures instead of videos.

When a MVP omnidirectional video acquisition system is available, streaming MVP omnidirectional videos to the end-user brings new challenges. We highlight in particular two challenges: (*iv*) determining the service implementation design that enables a good immersion experience without explosion of the resource requirements in the delivery chain (server, network, client); (*v*) designing the best data downloading strategy at the client side to maximize the QoE. Exploring these challenges and their possible solutions is the scope of this Chapter.

1 6DoF VR will typically be addressed during the phase 2 of the MPEG-I
2 Cameras can be either real cameras or virtual cameras based on view synthesis

Figure 4.1: Example of viewpoint switching options appearing in a viewport in the *Google Earth VR* [57] interface, i.e., *walk-through* functionality. Each white sphere appearing in the viewport represents a viewpoint that the user can switch to. The viewpoint switch can be selected using a controller, whose virtual representation is appearing in the viewport as well.

The first challenge is related to the implementation of the server. In the recent years, researchers have sought for solutions to stream *single-viewpoint* omnidirectional videos. For obvious reasons of re-using existing delivery architectures, proposals aim to patch the technologies of dynamic adaptive streaming, typically MPEG DASH [120, 125]. This is for instance the case of the viewport-adaptive streaming solution introduced in the previous Chapters. Multiple version of the same video is prepared once at the server side, and the client, based on a prediction of the future available bandwidth and the future head orientation (viewport position), decides which video segments to download to maximize the quality in the displayed area. By multiplying the number of viewpoints, the service provider should provision a large amount of resources to prepare, store, and deliver the data. The choices at the server side, which are both on the implemented technologies and the design of the application, are bound to the resource limitations on the whole delivery chain. Identifying the best design choices in this regard is thus an open research topic.

The second challenge is related to the selection of representations at the client side. Now, in addition to the two criteria that the client has to take into account to select the representation (bit-rate and head orientation), the translation movements in the 6DoF scene should be anticipated to enable fast viewpoint switch. To the best of our knowledge, no previous work has studied *MVP* omnidirectional video streaming. Nevertheless, it is reasonable to assume errors in translational movement predictions, which would affect the interactivity of the application and could significantly impact the user's QoE. The design of an accurate client adaptation strategy, including a 6DoF movement prediction module, is thus another open research topic.

In this Chapter, we analyze some MVP omnidirectional adaptive streaming scenarios, we describe the general layout of the omnidirectional MVP acquisition configuration as well as the content navigation at user side. We define the adaptation strategy of the dynamic adaptive streaming client as an optimization problem and demonstrate that exploiting the knowledge upon the way the user is navigating the content can maximize the QoE, while minimizing the bandwidth consumption. More precisely, we propose four contributions in this Chapter:

- We discuss possible implementations of a MVP omnidirectional adaptive streaming system from the server perspective, reviewing state of the art encoding solutions that could be used in this scenario.
- For a given adaptation set, we formulate the client adaptation logic, i.e., the algorithm that selects which representation to download for each segment, as a Mixed Integer Linear Programming (MILP) optimization problem to jointly (*i*) maximize the visual quality in the user's viewports, (*ii*) minimize the viewpoint switching delay, and (*iii*) minimize the frequency and duration of video playout stalls, subject to bandwidth constraints. We consider the ideal scenario, where the client performs a perfect prediction of both the available bandwidth as well as the user's navigation patterns, as an upper bound scenario for the optimization.
- We then study two extreme strategies for the realistic scenario in which the client does not predict the viewpoint switch. The first strategy is a *proactive* algorithm where the client systematically downloads other viewpoints to anticipate viewpoint switches and enable minimum switching delay. The second strategy is a *reactive* algorithm where the client downloads viewpoints only when the user commands a translational movement, such that viewpoint switch requires some tolerance to delay. Both strategies bound the spectrum of all viewpoint-switching adaptation algorithms that the service providers could design.
- We acquired a four-minutes long omnidirectional MVP video sequence and developed an interface for a 6DoF VR application to collect the navigation patterns of some users watching our video sequence. An analysis of the navigation patterns of our set of users is presented. We gathered multiple existing tools together to emulate the MVP omnidirectional DASH streaming, to extract users' viewports from the reconstruct bit-stream with non-homogeneous quality, and to compute some objective video quality metrics.

Figure 4.2: Example of omnidirectional video frame in *cube-map baseball layout*: the spherical frame is mapped to the plane via the cube-map projection and the faces of the cube are arranged into a rectangular frame with 3:2 aspect ratio by minimizing discontinuities between the cube faces.

The Chapter is organized as follow. Section 4.2 presents the related work on the topic of omnidirectional video streaming and multi-view adaptive video streaming. Section 4.3 describes the layout of a general MVP omnidirectional content acquisition and navigation framework, introducing the notation and terminology used in the Chapter, and exhibits the options considered in this Chapter to encode a MVP omnidirectional at the server side. Section 4.4 presents the client of a MVP omnidirectional video streaming system, as well as the metrics used to model the user's QoE in such a system. Section 4.5 introduces the proposed formulation of the client adaptation logic as an optimization algorithm. Section 4.6 describes the MVP video content used for the evaluation, as well as the test conditions used to simulate the streaming session, and discusses the results. Finally, Section 4.7 concludes the Chapter.

4.2 RELATED WORK

We are not aware of any paper dealing with MVP omnidirectional video streaming for 6DoF VR. The bibliography that we report in the following is related to multi-view perspective (as opposed to omnidirectional) videos. The bibliography related to the streaming of single-camera omnidirectional videos (such as heterogeneous spatial video encoding, user behavior prediction and representation request) is elaborated in Chapter 2. We focus on those topics because MVP omnidirectional videos can be seen as a mix between single-camera omnidirectional videos and multi-view perspective videos and so innovation in one of those fields may also be beneficial to MVP omnidirectional video technology.

Multi-view video streaming considers content acquired by multiple cameras having limited disparity. Often, depth information is also available, which can be used to synthesise additional views. The studies addressing multi-view video streaming over HTTP can be clustered according to their focus on (*i*) the strategy used to create the adaptation set, thus, the multi-view video encoding strategy used to optimize the storage space at server side and allow rate-adaptive video transmissions and (*ii*) the optimal request or delivery scheme, to account for the priority of the view that clients request as well as the view-switching probabilities.

Multi-View Adaptation set. To exploit the redundancy that is present in multiple views of the same scene, Su et al. [126] propose an HEVC multi-view streaming system where the multi-view video and depth content is encoded using the scalable extension of HEVC. The scalable coding introduces dependencies between representations of different views. Scalable video coding of multi-view video including depth content is also used by Zhao et al. [161], who propose a cloud-assisted streaming system where virtual views can be synthesized at server or client side depending on the network

conditions and the cost of the cloud-based server. Recently, Toni and Frossard [134] have proposed to formulate the adaptation set design at server side as an integer linear programming optimization problem, in order to optimize storage constraints while offering a good navigation quality to the different users.

Optimal Request or Delivery Scheme. Focusing on the adaptation logic at the client side, Hamza and Hefeeda [50, 51] propose a quality-aware rate adaptation method for free-viewpoint streaming, based on a rate-distortion model that relates the distortion of the texture and depth components of reference views and target virtual views. This model enables the client to find the best set of representations to request from the server. The adaptation strategy takes into consideration the user interaction with the scene, assuming that the navigation trajectory of the user is predicted at client side. The problem of minimising the latency of the view-switching at client side has been addressed by multiple works in literature. Xiao et al. [147] propose two view switching approaches which exploit data buffering at client side, in order to reduce the view switching delay. Carlsson et al. [18] propose a prefetching policy implemented at client side, so that the requested view and rate are adapted based on the stream switching probabilities and the current bandwidth constraints. Yun and Chung [153] use a buffer occupancy controller at client side, as well as parallel streaming and server push policy to minimize the view-switching delay. Finally, Zhang et al. [160] propose a priority-based adaptive scheduling algorithm implemented at the server side when multiple viewpoints are simultaneously transmitted over bandwidth constrained network to multiple clients.

In this paper, we study adaptive streaming for MVP omnidirectional videos. To the best of our knowledge, neither the single-camera omnidirectional video, nor the multi-view perspective video delivery are directly related to the system we address here. Regarding single-camera omnidirectional video, we adopt most of the techniques that have been developed in the recent years to match the delivered content to the displayed viewport. But no previous work has dealt with multiple viewpoints. Regarding multi-view delivery, the cameras in our case exhibit a high variation of the disparity and only a fraction of a viewpoint is displayed. Note however that multi-view techniques can be used to generate *virtual* cameras, which may become new viewpoints in the MVP omnidirectional video system. But virtual viewpoint synthesis is out of the scope of this paper.

4.3 MULTI-VIEWPOINT OMNIDIRECTIONAL FRAMEWORK AND NOTATION

In this section we introduce the geometry of a single viewpoint, generalize the framework to a MVP system focusing on the viewpoint switching conditions, and present the options available to a service provider to encode MVP omnidirectional videos. We describe the general layout of content acquisition and navigation of MVP omnidirectional content, introducing the notation and terminology used in the paper.

In this Chapter we consider the same Euclidean space as introduced in Section 2.2. This space is associated with the reference frame $\left(O, \vec{\imath}, \vec{\jmath}, \vec{\mathbf{k}}\right)$.

4.3.1 *One Viewpoint*

In this Chapter we define a omnidirectional camera as in Section 2.3.2 except that its center of projection can now be any point $v \in \mathbb{R}^3$. The camera projects any point in $\mathbb{R}^3$ to a point on the spherical imaging surface of radius r, usually considered unitary, *i.e.*, the *viewing sphere* S_v centered at v. Formally, $S_v \coloneqq \left\{p \in \mathbb{R}^3 : \|p - v\| = r\right\}$. The omnidirectional video signal is defined on S_v. The viewing sphere

is associated to the orthonormal frame $\left(v, \vec{\imath}, \vec{\jmath}, \vec{\mathbf{k}}\right)$: a translation $T \in \mathbb{R}^3$ transforms $\left(O, \vec{\imath}, \vec{\jmath}, \vec{\mathbf{k}}\right)$ into $\left(v, \vec{\imath}, \vec{\jmath}, \vec{\mathbf{k}}\right)$.

A user watching the omnidirectional video is assumed to be at the center of the viewing sphere. At each instant in time, the user visualizes only a portion of the spherical surface, depending on his *viewing orientation*. During the video duration, the user can change his viewing orientation to navigate the video. We associate the user's head to the orthonormal frame $\left(v, \vec{\imath'}, \vec{\jmath'}, \vec{\mathbf{k}'}\right)$, where $\vec{\imath'}$ goes through the front of the user's head, $\vec{\jmath'}$ through his left ear, and $\vec{\mathbf{k}'}$ through the top of his head. A rotation $R \in \mathrm{SO}(\mathbb{R}^3)$ transforms $\left(v, \vec{\imath}, \vec{\jmath}, \vec{\mathbf{k}}\right)$ into $\left(v, \vec{\imath'}, \vec{\jmath'}, \vec{\mathbf{k}'}\right)$. Thus, at a given instant in time, the user's viewing orientation within the viewing sphere centered at v is uniquely identified by R. $\vec{\imath'} = R(\vec{\imath})$ is then the user viewing direction.

The viewport attended to the user at a given instant, as defined in Section 2.3.2, and extracted by the client, is then uniquely defined by (v, R) (i. e. by the user viewing position and viewing orientation).

4.3.2 *Multiple Viewpoints*

An omnidirectional MVP video content corresponds to a finite set of L omnidirectional video sequences of the same scene, captured by omnidirectional cameras located at different positions in space, and synchronised in time, at frame precision. We refer to each omnidirectional video, corresponding to a camera at a particular position in space, as a *viewpoint*. We denote by $\mathcal{V} = \{v_j\}$, with $j \in \{1, \ldots, L\}$, the finite set of all viewpoints in the MVP omnidirectional content.

Since the MVP omnidirectional streaming applications are still in their early stages of development, the interfaces for viewpoints switching that will become the most popular are not known yet. We present hereafter several options. We denote by $\mathcal{N}_{j,R} \subseteq \mathcal{V}$ the set of viewpoints accessible from a given viewport (v_j, R).

TELEPORTATION WITHOUT RESTRICTION Users can switch to any viewpoint without restriction, formally $\mathcal{N}_{j,R} = \mathcal{V}$. It means the 6DoF VR application authorizes *teleportation*, without regards to the physics nor the visibility of the viewpoint.

STEP BY STEP MOVES Users can only switch to neighbor viewpoints. Formally, $\mathcal{N}_{j,R}$ contains only the neighbors of v_j that are at most at distance d from v_j: $\mathcal{N}_{j,R} \coloneqq \left\{v_k \in \mathcal{V} : 0 < \|v_k - v_j\| \leqslant d\right\}$. This option respects the law of physics since it binds the user to the most immediate moves in the Euclidean space. The user can decide to switch to a viewpoint that is not in her current viewport: the use can move backward or sideways.

TELEPORTATION WITHIN THE VIEWPORT Users can switch to any viewpoint in their current viewport. Teleportation is still possible, but users can only switch to visible viewpoints. Formally, $\mathcal{N}_{j,R}$ contains all viewpoints v_k within a certain angular distance from $\vec{\imath'}$, where $\vec{\imath'}$ denotes the the user's viewing direction (as introduced in Section 4.3.1).

STEP BY STEP MOVES WITHIN THE VIEWPORT $\mathcal{N}_{j,R}$ is not only restricted to the nearest viewpoints to v_j, but also to the viewpoints that are located in the current viewport. This implementation is the most natural since it restricts the user to moving as in the real world. Formally, viewpoint v_k belongs to $\mathcal{N}_{j,R}$ if and only if the *switching direction* v_k, is within a given angular distance from the user's viewing direction $\vec{\imath'}$ and if the distance between v_k and v_j is smaller than a given value d.

The temporal aspects of the switch are discussed in Section 4.4.

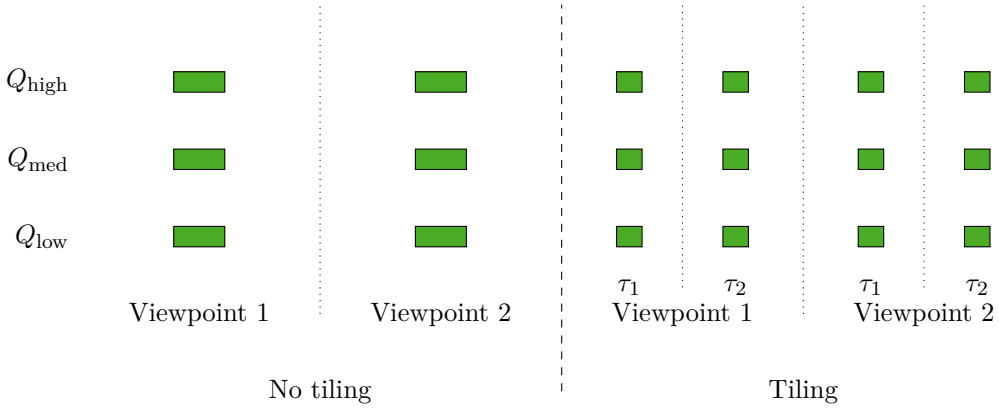


Figure 4.3: Adaptation Sets for a scenario without tile and an other with two tiles. In green, directly decodable base layers.

4.3.3 *Multi-viewpoint Omnidirectional Encoding Options*

We consider the case of a content provider, which implements dynamic adaptive streaming technologies, such as MPEG DASH, to stream omnidirectional MVP video content to its clients. We do not consider any storage limitation at server side, i. e. the server has unlimited storage capacity. The adaptation logic is implemented at client side, i. e. the client selects the video segments to request, since most of the adaptive streaming technologies implemented nowadays use this solution. The content provider has to choose how to encode the MVP omnidirectional data.

We consider a scenario where the representation bit-rate varies but the resolution remains constant. The choice of the encoding solution results in different adaptation sets stored on the streaming server. In this Chapter, we consider two encoding options that result in corresponding adaptation sets (Figure 4.3):

No Tiling The simplest encoding option to create the adaptation set is to encode each viewpoint as one independent spatial entity, at multiple bit-rates. The adaptation set, as illustrated in Figure 4.3, contains one representation per bitrate (i. e., quality level) and per viewpoint. Each representation can be downloaded independently of the others. To select the representation to download, the client needs a prediction algorithm for both available bandwidth and viewpoint switching. When decoding the stream, the client uses only one representation per viewpoint. All other downloaded representations are dropped and the bandwidth used to download them is wasted.

Tiling Each viewpoint can be spatially divided into non-overlapping, independently decodable portions, i. e. *tiles*. The tile-based adaptive streaming optimization that has been proposed for mono-view omnidirectional video streaming can be extended to the MVP case. Each video corresponding to a viewpoint is split into the same number $\mathcal{T}$ of tiles, according to the same tilling pattern. The adaptation set is then composed of $\mathcal{T}$ representations per viewpoint and per bitrate (i. e. quality level), as illustrated in Figure 4.3 with $\mathcal{T} = 2$. Each tile can be downloaded and decoded independently of the others. Only one quality can be used by the decoder for a given tile and viewpoint. Downloading tiles at different position in a given viewpoint with different quality level allows the decoder to generate an output omnidirectional video with variable quality within the omnidirectional frame. If the user watches a viewpoint v_0, all downloaded representations of other viewpoints are wasted.

More complex encoding scenarios, exploiting the inter-view redundancy by using scalable video coding, have been successfully used to create adaptation sets for adaptive streaming of perspective videos [56, 71, 112], even in the case of multi-view perspective videos, as discussed in Section 4.2. While such scalable encoding solutions could be suitable for the MVP omnidirectional scenario, eventually in combination with tiling, these options are out of the scope of this dissertation: we only consider encoding scenarios that do not use scalability.

4.4 CLIENT SIDE

The goal of the client is to download video segments so that it can display with a high quality the viewport requested by the user in the right viewpoint. The client has a limited downloading capacity for each video segment. We denote by B_i the average available downloading bandwidth during video segment i.

4.4.1 *Switching Decision and Timing*

We first focus on viewpoint switching, which is a key novel feature of the MVP 6DoF video streaming system. We suppose a user can at most switch viewpoint once during a segment. We consider that any representation (whether it is the full omnidirectional video or a tile) has frequent RAPs, *i. e.*, frames without any dependency to any prior frame. To display a given video frame, a client should have downloaded and decoded all the frames since the last RAP before the said frame. For the sake of simplicity and in conformance with the recommendations for implementation of dynamic adaptive streaming systems, we suppose that each segment in the adaptation set starts with a RAP and contains no other RAP.

Let us consider a user switching at instant t from a viewpoint v_j to another viewpoint $v_k \in \mathcal{N}_j$. To enable an *immediate* switch, the MVP omnidirectional streaming system should both (*i*) have in buffer the frames of the omnidirectional video of the viewpoint v_k, and (*ii*) have decoded all the frames of viewpoint v_k since the latest RAP. In practice, a standard user device does not concurrently decode multiple videos (multiple viewpoints of the same scene) due to memory and computing resource consumption. We thus consider two more realistic cases, whether the translation movement command results in a viewpoint switch to the immediately following RAP of viewpoint v_k or to the next one. Let denote by i the identifier of the currently displayed segment and by t_r the starting time of the next segment, i. e. segment $i+1$.

- If the client, at instant t, has already downloaded tiles of segment $i+1$ for viewpoint v_k or if the client has still time to download to download those tile before t_r, then the switching can be done at t_r.
- If the client has not downloaded tiles of segment $i+1$ for v_k and it has no time to download them until the deadline t_r, then the switch cannot be done at the next RAP.

In the graphical interface, typically in *Google Earth VR*, the implementation of the translation movement includes a transition effect based on a "tunneling" animation, so that the switch appears as if it was instantaneous: upon the movement command, a blurred animation of the content (basically simulating motion) is shown until the time at which the display of the new viewpoint can be done.

4.4.2 *Download Decision and Scheduling*

To fuel the video decoder buffer in input, we propose a serialized iterative request process with a unique buffer, as illustrated in Figure 4.4. The client issues a request for one representation at one segment in one viewpoint. We suppose, for modeling simplification, that once a segment is requested, the download cannot be canceled and has to be completed. Not that in practice representation downloads can be canceled by the client. Once the download is complete, the downloaded segment is stored in the buffer, and then the client can issue another request for another representation.

At the beginning of a video segment i, the client selects, among the video representations that are buffered for this segment i, the representation to decode. We suppose that (*i*) the client can only select fully downloaded and fully decodable representations; (*ii*) the client cannot start decoding

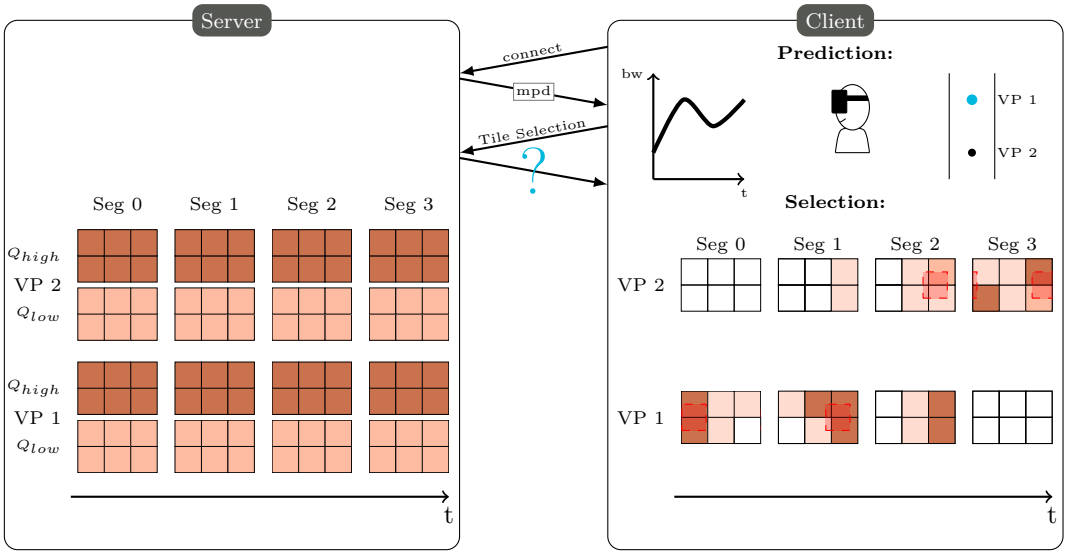


Figure 4.4: Illustration of the proposed DASH-like streaming architecture. The server has prepared an adaptation set, and the client uses prediction of the future available bandwidth, future user's head orientation and future selected viewpoint to download representation for some tiles and fill its download buffer. The red block illustrate the viewport positions and show that many representation are downloaded but never actually displayed. See Figure 3.3 to compare with the mono-viewpoint streaming architecture proposition and Figure 2.8 for the traditional DASH architecture.

another representations afterwards during the course of the segment; (*iii*) for each tile, at most one representation can be selected for decoding ; and (*iv*) the client can only select representations of the same viewpoint (i. e. cannot decode tiles from different viewpoints in parallel).

4.4.3 *Assessing the Quality of Experience*

Three objective metrics can be considered to evaluate the QoE felt by a user watching a MVP omnidirection video. Evaluating the correlation between those metrics and the subjective evaluation of the QoE is out of scope for this paper. Those metrics are:

Distortion The objective quality of the displayed viewports is related to the quality of the selected representations. For a given displayed representation, and for a given viewport during a segment, we extract the same viewport from the original full-quality omnidirectional video of this viewpoint. This latter is the reference video. We can then run an objective video quality metric to compare the displayed viewport with the viewport from the original content. Video quality metrics include the PSNR and the MS-SSIM. See Section 2.4.2 for more details on objective quality metrics.

Stalls This event, sometime denoted as rebuffering event, happen when the client pauses the display of the content to wait for the next frames to be downloaded. A stall happens when no representation has been received on time. In the MVP scenario, a stall is likely to occur after unanticipated viewpoint switch.

Viewpoint Switching Lag It is the duration between the time the user commands a translation move and the time the first viewport of the requested viewpoint is displayed. The impact of this lag on the QoE is an open question. Even if not supported by any subjective testing, the shorter is the transition, the better is the feeling of immersion. It requires however the MVP omnidirectional system to anticipate the switch by downloading neighboring viewpoints, which may be not watched.

4.5 ALGORITHMS

We start this Section by a discussion about the requirements of the algorithms that need to be implemented at the client side to enable good performance for 6DoF video streaming systems. Then, we study optimal algorithms, which correspond to some specific conditions. These algorithms aim to provide a theoretical framework for the study of MVP omnidirectional video systems and to highlight some radical implementation choices.

4.5.1 *Toward Practical Algorithms*

The system performance depends on the capacity of the client to predict (*i*) the network conditions in the next seconds, in particular the available bandwidth; (*ii*) the next head orientation rotations, which correspond to the interactive feature of traditional 3DoF video systems: yaw, pitch, and roll; (*iii*) the next translation movements, which are the three novel interaction features of 6DoF systems, here restricted to viewpoint switch. We would like to emphasize again that the accuracy of the prediction algorithms on those three parameters is critical to the success of the implementation. The ongoing efforts to develop prediction strategies, for both traditional adaptive streaming and single-camera omnidirectional videos can be reused here.

The prediction of the client can then be used when the client has to take a decision on which representation to download next. This decision contains up to four choices:

- **The segment**. The request can be for a representation that will be played in the immediate next segment (typically if a viewpoint switch just happened or if the latest head rotations reveal that previous head rotation predictions were wrong), or can be for longer term segments to anticipate the movements and have the time to download high-quality representations.
- **The bit-rate**. The higher the bit-rate the lower is the distortion. However the client should also take into consideration that requesting a high-quality representation will monopolize the bandwidth for a long time, at the expense of the next requests. Furthermore the client has to request representation such that the downloading can be completed before the decoding and displaying time.
- **The viewpoint**. The prediction of the next translation movements enable the client to decide to request representations in the current viewpoint (if the client does not anticipate any move in the near future) or in another viewpoint (if the client anticipates a move soon). The client can also request representations in other viewpoint as a proactive strategy in case of unanticipated moves.
- **The tile** (if implemented). It is here the prediction of the next head orientation rotations that make the client request tiles in various locations of the frame in the given viewpoint. As for the viewpoint, the request depends on the estimated accuracy of the prediction.

Multiple parameters have to be considered for the implementation of efficient algorithms in practice. In this Chapter, we do not design practical algorithms in this regard. We focus instead on analyzing optimal algorithms for two extreme strategies: a proactive strategy where the client always proactively downloads all other viewpoints to anticipate possible viewpoint switches, and, on the contrary, a reactive strategy where the client reacts only to movement commands. Our analysis provides some bounds, which will hopefully serve a comparison basis for future work on practical algorithms.

4.5.2 *Optimal Decision with Perfect Predictions*

We first consider an omniscient client, which is able to perfectly predict the future available bandwidth, the future head orientation rotations, and the future switching decisions. The goal is thus to schedule the

representation requests so that every video that is displayed in the viewport (*i. e.*, every displayed tile in the right viewport) is downloaded on time at the highest possible quality. For the sake of simplicity, we will not consider the case of scalable coding. The model can easily be upgraded by introducing a decoding dependency constraint to include this case. We present the model in the context of the tiling encoding scenario because the non tiling scenario can be seen as a tiling scenario with only one tile per viewpoint.

The objective is to maximize the QoE for the user, with respect to the three metrics described in Section 4.4.3: distortion, stalls, and switch lag. We combine these three metrics into one by applying two float weight parameters α and β so that the function to maximize is a traditional image distortion metric, plus α times the stall metrics, plus β times the switch lag. Subjective quality assessment campaigns will be necessary to set these weights based on the relative impact of those metrics on the subjective QoE.

The notation is as follows. We denote by $r_{i,q,v,\tau}$ the representation for tile τ of viewpoint v with quality level q for the segment i. To model this decision problem, we introduce a set of binary variables $x_{i,q,v,\tau}$ such that:

$$x_{i,q,v,\tau} = \begin{cases} 1, & \text{if the client selects the representation } r_{i,q,v,\tau} \\ & \text{to generate the viewports of segment } i \\ 0, & \text{otherwise} \end{cases}$$

We introduce the binary variables $y_{i,v}$ such that:

$$y_{i,v} = \begin{cases} 1, & \text{if viewpoint } v \text{ is selected for display} \\ & \text{during segment } i \\ 0, & \text{otherwise} \end{cases}$$

The video distortion observed by the user for a segment i is computed as follows. We denote by $Q_{i,q,v,\tau}$ the average distortion observed by a user having its viewports totally inside the representation $r_{i,q,v,\tau}$ during the segment i. We suppose the distortion observed by a user having its viewports inside multiple tiles is equal to the weighted average of the average distortion of each tile representations, weighted by the ratio of the viewports surface within each tile. This is for instance the property of the MSE when the distortion is evenly spatially distributed within each representation. We denote by $V_{i,\tau}$ the average ratio of the surface of the viewports of the user within the tile τ during the segment i. Then, the distortion observed by the user during the segment i is

$$Q_i = \sum_{v,\tau} V_{i,\tau} \cdot \left(\sum_{q} (x_{i,q,v,\tau} \cdot Q_{i,q,v,\tau})\right)$$

Other notations include $\ell_{i,v}$, which is a constant equal to 0 if the user wants to display the viewpoint v during the video segment i, and 1 otherwise. The set of float variables $d_{i,j,q,v,\tau}$ is the downloading ratio of representation $r_{i,q,v,\tau}$ at the time of segment j. The integer variable s_j represents the duration (expressed in number of segment duration) of a stall during the downloading segment j, and S_j is a binary variable equal to 1 if and only if s_j is greater than 0.

The optimal model can be formulated as follow:

$$\min_{\{x_{i,q,v,\tau}\}} \sum_i (Q_i + \beta \cdot l_{i,v} \cdot y_{i,v}) + \alpha \sum_j (s_j + S_j)$$

s.t.

$$\sum_{i,q,v,\tau} d_{i,j,q,v,\tau} \cdot b_{i,q,v,\tau} \leqslant (1 + s_j) B_j \qquad \forall j \tag{4.1a}$$

$$\sum_j d_{i,j,q,v,\tau} \leqslant 1 \qquad \forall i, q, v, \tau \tag{4.1b}$$

$$\sum_{j=i+1}^{N} d_{i,j,q,v,\tau} = 0 \qquad \forall i, q, v, \tau \tag{4.1c}$$

$$x_{i,q,v,\tau} = \sum_j d_{i,j,q,v,\tau} \qquad \forall i, q, v, \tau \tag{4.1d}$$

$$V_{i,\tau} \leqslant \sum_v \sum_{q=0}^{L} x_{i,q,v,\tau} \qquad \forall i, \tau \tag{4.1e}$$

$$\sum_v y_{i,v} = 1 \qquad \forall i \tag{4.1f}$$

$$x_{i,q,v,\tau} \leqslant y_{i,v} \qquad \forall i, q, v, \tau \tag{4.1g}$$

$$y_{i,v} \leqslant y_{i-1,v} + (1 - \ell_{i,v}) \qquad \forall i, v \tag{4.1h}$$

The set of equations (4.1) formally defines the optimization problem into an MILP problem. Equation (4.1a) defines the bandwidth constraint for each segment, including the extra bandwidth obtained when the client pause the video display. Equation (4.1b) limits each representation to be downloaded at most once. Equation (4.1c) forbids the download of a representation after its display deadline. Equation (4.1d) indicates that a representation can only be selected for decoding if it was fully downloaded, and forbids the optimal solution to download not displayed representations. Equation (4.1e) enforces the model to select one available representation when the tile is visible during the segment. The minimization problem implies that if this constraint is active, only one representation is selected. Equation (4.1f) guarantees that one and only one viewpoint is displayed during a segment. Equation (4.1g) states that a representation of segment i can only be selected for display if it belongs to the displayed viewpoint during segment i. Equation (4.1h) indicates that a viewpoint can be selected for display either if the user requested to watch it or if it was displayed during the previous segment.

4.5.3 *Optimal Proactive Strategy for Fast Switch*

We consider now the case of a content provider that guarantees that the viewpoint switching lag is minimal, *i. e.* whenever the user commands a translation move, the new viewpoint is displayed at the next segment. This strategy can typically be implemented by a content provider that implements accurate bandwidth and head orientation prediction algorithms, but has no clue on the user behavior regarding translation movements.

The implementation of this strategy imposes to download, for every segment, at least one representation for the predicted tiles in every viewpoint among all the possible switchable viewpoints. For instance, if the client is in viewpoint v_j at segment i, then the requests for segment $i + 1$ proactively include the tiles corresponding to the predicted head rotation R for *all* viewpoints in $\mathcal{N}_{j,R}$. To ensure a smooth transition, we also force the quality to be the same for all downloaded tiles of the same segment.

This strategy guarantees a minimal switching delay at the expense of the displayed video quality, wasted download resources, and stall if the bandwidth is too low. Indeed, at the start of a new video

segment, representations for all viewpoints have been downloaded but only one viewpoint is decoded and displayed. The price to pay for guaranteed interactivity is thus entire viewpoint downloading waste. Note that the design choices of the content provider regarding the switching conditions (see Section 4.3.2) can have a significant impact on the performance.

To model this strategy and to allow fair comparison with the other tested strategies, we suppose a perfect bit-rate and head orientation prediction. The Equation (4.1d) is updated to additionally enforce the download of a representation if a representation with the same quality, at the same tile and segment was selected at any viewpoint. We consider the worst case in Section 4.3.2: teleportation without restriction.

$$\sum_{v'} x_{i,q,v',\tau} = \sum_{j} d_{i,j,q,v,\tau} \quad \forall i,q,v,\tau \tag{2d}$$

4.5.4 *Optimal Reactive Strategy*

We consider finally the opposite case of the previous strategy. The content provider now agrees that a viewpoint switch can be postponed. It can typically be the case if the graphical animation for viewpoint switching can be elegantly implemented. The prediction of bit-rate and head orientation is still perfect and, again, no prediction for translation movement is available. The client does not anticipate any viewpoint switch, and thus implement a *reactive* strategy.

In this strategy, the client requests only representations in the current displayed viewpoint. The strategy is said reactive because representations in the new viewpoint are requested only when the user commands a translation moves. If the client has no time to download any representation until the start of the next segment, then the switch is postponed to the following segment. In this strategy, the waste of representation downloading is minimized, and thus the video quality is maximized.

To model this strategy, we insert the following constraints in the omniscient model:

$$d_{i,j,q,v,\tau} \leqslant y_{\max(j,0),v} + y_{\max(j+1,0),v} \forall i,j,q,v,\tau \tag{3h}$$

If user switch to viewpoint v at segment $j+1$:

$$\sum_{i,q,\tau} d_{i,j,q,v,\tau} \leqslant (1 - t_j + s_j) B_j \quad \forall v,j \tag{3j}$$

Equation (3h) allows the client to download only representations that will be displayed during the next video segment or during the current video segment. Constraint (3j) indicates that if the user decides to switch viewpoint t_j seconds after the beginning of the current downloading segment, the client cannot download segment for the new viewpoint before the decision was made. We additionally add a constraint to enforce representations to be downloaded in display order (i. e. if a representation of video segment i is downloaded during segment j then representations of segment $i-k$ cannot be downloaded during segment $j+k'$ with k,k' integers greater than 1).

4.6 EVALUATION

In this Section we evaluate the two extreme strategies introduced in Sections 4.5.3 and 4.5.4 and compare them to the optimal omniscient client defined in Section 4.5.2.

Videos	Quality 0		Quality 1		Quality 2	
	PSNR [dB]	rate [Mbit s^{-1}]	PSNR [dB]	rate [Mbit s^{-1}]	PSNR [dB]	rate [Mbit s^{-1}]
no tile	46.25	5.01	47.12	8.01	48.45	16.02
3×2 tiles	46.22	5.02	47.10	8.02	48.44	16.02
6×4 tiles	46.19	5.08	47.06	8.08	48.42	16.08

Table 4.1: Bit-rate and distortion expressed in PSNR for the encoded video, averaged over all segments, over all tiles, and over all three cameras, for the three tiling scenarios and the three quality levels.

4.6.1 *Test-bed*

The evaluation test-bed is made of four main components.

- A four-minute long multi-view omnidirectional video, captured by three *Orah Live Spherical VR Camera 4i* omnidirectional cameras. The cameras were aligned in a hallway, inside a building. Each camera showed in different offices with people working/moving inside. Each camera records a 4K equirectangular video at 30 fps. Cameras are positioned such that their local reference frames are aligned.
- An Android application to navigate into the MVP omnidirectional video and record users' trajectories. The application is based on *Google Daydream*[3] framework. The neighboring viewpoint are represented by small white sphere, as in Figure 4.1; users switch by using a controller. The video is streamed over Wi-Fi, with 2K resolution, at a constant quality level using DASH. The video is split into one-second long segments. The user can switch at any time but the switch is only effective at the beginning of the next segment. Around 60 times per seconds, the system logs a timestamp, the user's viewports orientation, and the current displayed viewpoint. The switching decision times are also recorded in the log file.
- An offline optimization software, used to solve the three optimization problems introduced in Section 4.5. The input are the distortion and the bit-rate of each encoded video segment, the user trajectories inside the MVP video, and the average download bandwidth available during each download chunk. The outputs are the list of downloaded segments, the viewpoint that was displayed to the user, and indication of the necessary stalls. The software is implemented in C++, and use the *IBM ILOG CPLEX Optimization Studio*. We released it open-source, and made it available on *Github*[4] to reproduce the results of this Chapter (see Appendix B.1).
- A Python3 script which combines *Gpac MP4Box* [76] and *ffmpeg*,[5] to generate the decodable bit-stream that the client would have obtained. The script then extracts, for each video frame, the viewport inside the generated video, with the orientation indicated by the user navigation trace, and in parallel inside the original video. This extraction is performed using an open-source software named *360Transformations*.[6]

We used the following procedure to encode the MVP omnidirectional video. First we project the omnidirectional video of each viewpoint onto a 3×2 compacted cube-map baseball as illustrated in Figure 4.2. For each tile set-up (no tile, 3×2, and 6×4 tiles), we used the open-source *Kvazaar* [69] software to encode each projected viewpoint into three representations, encoded with an average bit-rate target: 5 Mbit s^{-1}, 8 Mbit s^{-1} or 16 Mbit s^{-1}. When tiles are used, we used the MCTS configureation of *Kvazaar*, and we enforced each tile to have the same dimensions, so that tiles are restricted to the

3 https://developers.google.com/vr/
4 https://github.com/xmar/MultiViewpoint360_MMSys18
5 https://www.ffmpeg.org/
6 https://github.com/xmar/360Transformations/tree/master/transformation

	User A	User B	User C	User D
Sex	F	F	F	F
Age	34	31	27	27
Nb. switch event	18	10	4	13
Ang. Vel. [Deg./s]:				
global	28.81	20.88	28.44	20.00
2 s *before a switch*	8.48	3.42	3.24	7.53
2 s *after a switch*	59.57	46.88	54.25	51.21
between two switches	29.38	20.45	28.79	20.42

Table 4.2: Information about the users: sex, age, number of switching event, median angular velocity. The median angular velocity is ether computed globally, two seconds before a switch event, two seconds after a switch event or in the period between two switch events.

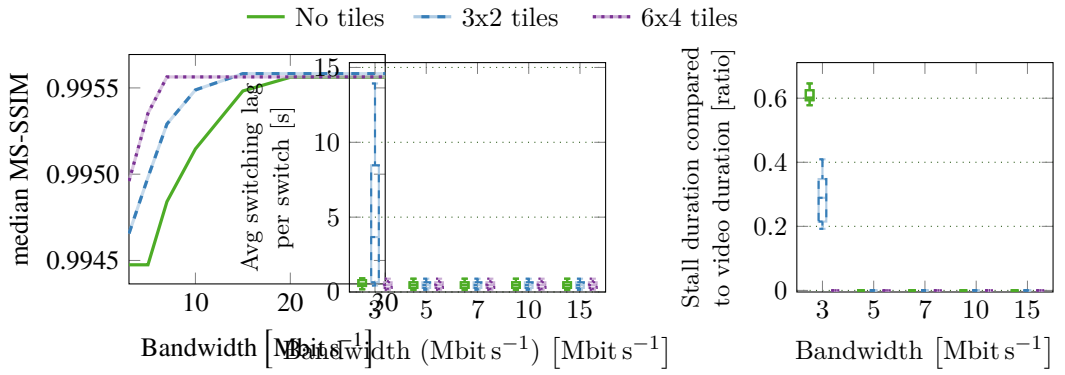
faces of the cube-map projection. The resulting video were split into one-second DASH segments using *GPAC MP4Box*. We also used *MP4Box* to extract each tile of each video segment into a different bit-stream file. Table 4.1 summarizes the PSNR measured by *Kvazaar* during the encoding process, and the average bit-rate of each representation. The PSNR is measured in comparison to the original video, on the whole frame (even for the tiled videos), in the cube-map domain, on the luma component of the pixels. The bit-rate is computed on the DASH-ed files by adding the size of each file (including the initialisation files) and dividing by the video duration.

The objective function of the three optimization models uses the parameters α and β. The parameter α can be interpreted as a weight for the dissatisfaction on the user for every stall events; we set it to 100 times the maximal MSE in our adaptation set. The parameter β is the weight of the dissatisfaction for every second of lag; we set it to $\frac{\alpha}{10}$.[7]

4.6.2 *User Behavior in Multi-viewpoint (MVP) omnidirectional Video*

Table 4.2 shows some statistics about the navigation traces that we captured on four users (all female), whose age ranges from 27 to 34. All users had already watched single-viewpoint omnidirectional videos before. They switched to new viewpoints in average 11 times during the four-minute long MVP video but the variance is here significant (more than four times more switches between user A and user D).

We extracted the median angular velocity from head movements, and we identified a *switch preparation time* and a *discovery time* respectively before and after switching events. Indeed, the median angular velocity is significantly lower than the median velocity two seconds before a switching event (around $20\,^\circ\,s^{-1}$ lower), while it is significantly higher two seconds after a switching event (around $30\,^\circ\,s^{-1}$ higher). Should these first observations be confirmed by more measurements, the prediction of viewpoint switching events could become an easy process in standard behavior.



(a) Median MS-SSIM within the displayed viewports **(b)** Box plot switching lag **(c)** Box plot stall ratio

Figure 4.5: QoE metrics for the omniscient scenario

4.6.3 *Results: Optimal with Perfect Prediction*

We first analyse the choices of the omniscient optimal client, illustrated in Figure 4.5. This theoretical client uses its perfect knowledge of the future head orientation of the client to minimize the objective function. The objective function is split into three sub-objectives: minimizing the distortion in the displayed viewports, minimizing the time required to switch to a new viewpoint, and minimizing the duration the video display was paused by the client. Our results show the interplay between these sub-objectives, with respect to our weight parameters α and β.

Figure 4.5a represents the median MS-SSIM [143] inside the video displayed in the viewports for various available bandwidth (see Section 2.4.2). We observe that the distortion on the user viewports is reduced when the available download bandwidth increases. Moreover, the encoding with tiles allows the client to select only the displayed tiles in high quality, which results, for a given download bandwidth budget, in a viewport video with less distortions.

Figure 4.5b shows the average duration of a switching lag per requested viewpoint switching, for different average available bandwidth budget, and different number of tiles. The results are represented inside a box plot (from bottom to top, the horizontal lines represent respectively the minimum, the 25$^{\text{th}}$, the median, the 75$^{\text{th}}$, and the maximum value of the average switching lags for a given scenario). When the budget is higher or equal to 5 Mbit s^{-1}, the client does not decide to delay the switch.[8] When the bandwidth is less than 5 Mbit s^{-1} the client does not have enough bandwidth to download a full viewpoint at the lower quality level (encoded at 5 Mbit s^{-1}) and it has sometimes to delay the switch to allow the display of viewports with better quality.

Figure 4.5c depicts the cumulated stall duration relative to the duration of the video. The results are also represented in box plots, which are defined as the box plots in Figure 4.5b. When the bandwidth is higher than 5 Mbit s^{-1} (at least the bit-rate of the video with lowest quality), the client decides to never pause the video. However, when the bandwidth is not enough, the client decides to pause the video to get extra time to download better quality segments.

Figure 4.6 depicts the optimal value of the objective function depending on the available bandwidth for the three encoding scenarios and the omniscient scheduler. The lower the objective value the better the quality of the experience. The vertical axis uses a logarithmic scale. We observe that, taking into

7 Reader should note that the value of α and β have been chosen to represent the bias of the authors on how the stall, the switching lag, the image distortion respectively impact the quality of experience of the users. Further study should be perform to accurately select the value of those parameter to better represent the user QoE

8 The small variations in the switching delay come from the fact the users switches at any time in the course of a segment.

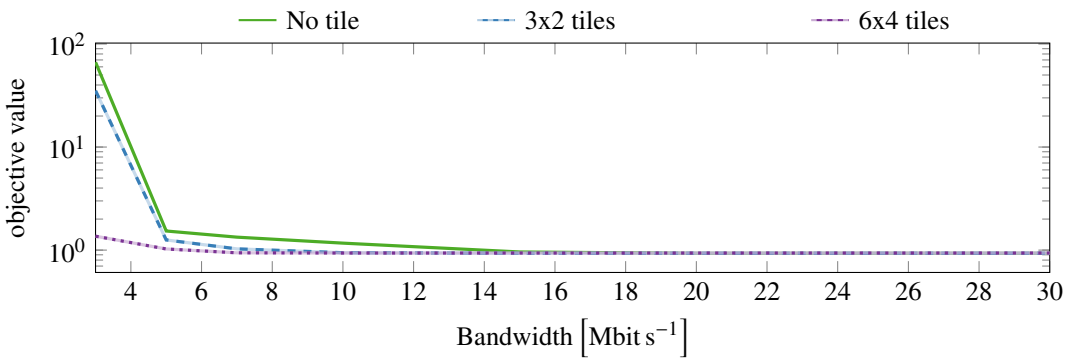
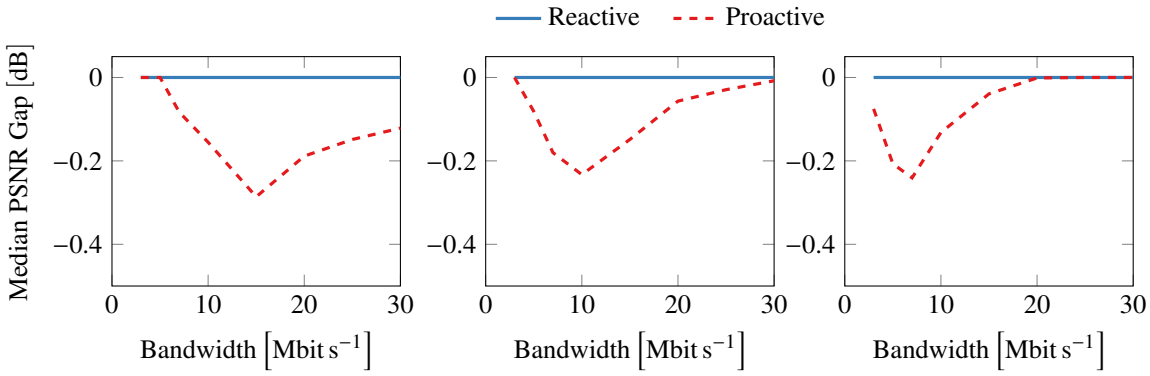


Figure 4.6: Raw objective function value for the omniscient scenario



(a) No tiles **(b)** 3x2 tiles **(c)** 6x4 tiles

Figure 4.7: Median of the gap between the displayed viewports PSNR of reactive or guaranteed scenario with the omniscient scenario.

account all metrics, using tiles improve the the quality of experience. In this scenario, after 20 Mbit s^{-1} there are enough bandwidth and all tiling scenarios become equivalent from the quality outlook.

We highlight the complex interplay between the three options in case of bandwidth shortage: reduce visual quality by switching to lower quality representations, or pause, or delay a switch, or combinations of them. Here, the omniscient optimal client picks in every of these behaviors to optimize the objective function with regards to the set weight parameters. This complex behavior calls for a campaign of subjective tests to model the parameters that reflect the interplay with better accuracy.

4.6.4 *Results: Proactive and Reactive Strategies*

We now compare the performance and the behavior of both client strategies against the optimal performance of the omniscient client. These two strategies are extremum of the possible downloading strategies that the client may implement. The *proactive* strategy does not accept any compromise on the viewpoint switching. The excessive anticipative viewpoint downloading generates bandwidth shortage, which can only be mitigated by stalls and distortion. On the contrary, the *reactive* strategy tolerates switch lags, with expected gains regarding video quality and stalls.

Figure 4.7 represents the median of the difference between the PSNR in the viewports for both strategies and the PSNR in the viewports of the omniscient client. The *reactive* client manages to obtain viewport videos with distortion close to the optimal, although the viewport distortion of the

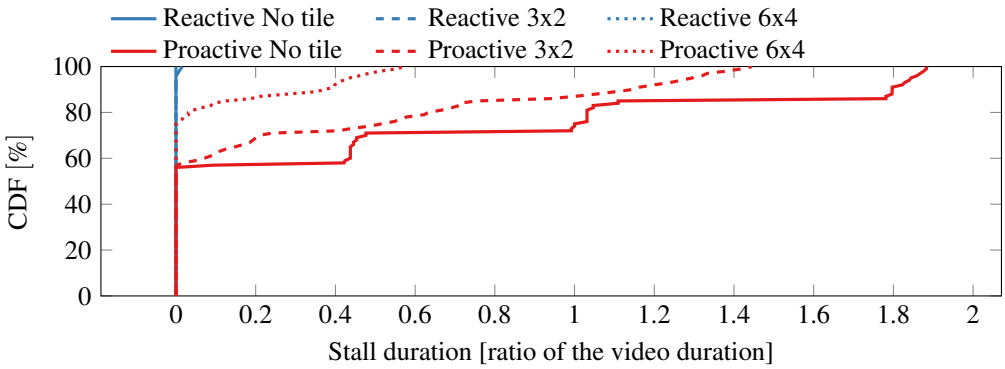


Figure 4.8: CDF stall duration compared to video duration, aggregated for bandwidth between 5 Mbit s^{-1} to 20 Mbit s^{-1}

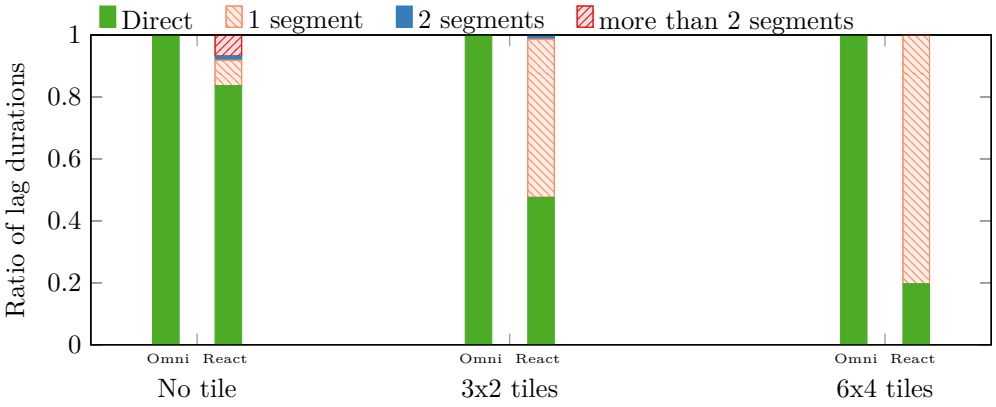


Figure 4.9: Ratios of lag durations in segment duration unit, aggregated for bandwidth between 5 Mbit s^{-1} to 20 Mbit s^{-1}, for the different scenarios and for different number of tiles

proactive client is higher. Note that the proactive strategy compensates the bandwidth shortage in different ways. For very high shortage (3 Mbit s^{-1} bandwidth), the client prefers to pause the video long enough to get a good quality video. However, the quality increases less quickly with the increase of the bandwidth than for the omniscient client. We also observe the advantage of using tiles: the client can more efficiently use the available bandwidth to get high-quality images in the user viewing direction. This effect is amplified by our assumption of perfect head orientation prediction.

Figure 4.8 depicts the CDF of the stall duration, relatively to the video duration, for both strategies. The CDF is generated using the stall duration for each user, aggregated for available bandwidth ranging from 5 Mbit s^{-1} to 20 Mbit s^{-1}. The proactive client has to pause the video display for a duration at least equal to the video duration in 25 % of the cases when no tile are used, and in 20 % of the cases when 3×2 tiles are used. There is at least one stall of 250 ms in 20 % of the case for the 6×4 tiles scenario. This long stalls are the price to pay for downloading the tiles in all the viewpoints (which guarantees minimum-lag viewpoint switch). The client requires around three times more download bandwidth than the two other clients to download a given displayed tile at a given quality. On the contrary, the reactive client never pauses the video for the 6×4 and 3×2 tiles scenarios and pause in only 5 % of the cases for the no tile scenario. Increasing the number of tiles decreases the needs for both strategies to pause the video.

Figure 4.9 represents the distribution of the delay between the switch command and the actual display of the new viewpoint. The delay is measured in terms of number of segments. The results are aggregated for the download bandwidth ranging from 5 Mbit s^{-1} to 20 Mbit s^{-1}. We do not represent the *proactive* strategy since the switch is by definition always performed immediately after the command.

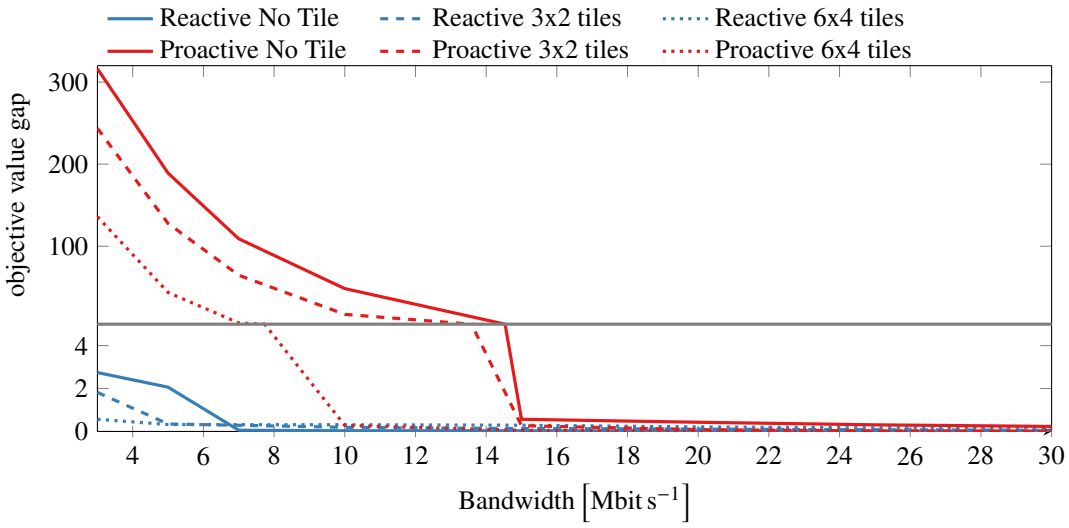


Figure 4.10: Objective function value difference with the omniscient scheduler for the reactive and proactive schedulers. Warning: The y-scale change after 5. The gray line indicates the scale where the scale changes.

The optimal omniscient does not need any delay to switch viewpoint for this bandwidth, which means that a perfect prediction of viewpoint switches enables high-quality interactive QoE. On the contrary, the *reactive* strategy needs to compensate the unanticipated switch events by delaying viewpoint switch. Depending on the setting of the tile encoder, the strategy differs. When no tile is implemented, the strategy manages to switch immediately in 84 % of cases, but it compensates by a few more stalls and, more importantly, switch delayed by two or more segments in 9 % of cases. The flexibility offered by the tiling encoding enables the reactive strategy to implement more consistent switches in the 6×4 and 3×2 tiles scenarios. Then, no switch is delayed by more than one segment.

Figure 4.10 indicates the optimal objective value, for the reactive scheduler and the proactive scheduler, compared with the omniscient scheduler. The plot display the difference between the measured objective values and the objective value of the omniscience scheduler measured for the same bandwidth budget and encoding scenario. We observe, as for the omniscient scenario, that the more bandwidth the better that total quality. As expected no scheduler performs better than the omniscient scheduler (the objective value gaps are greater than 0). When the bandwidth is greater than 16 Mbit s^{-1} the objective value of each scheduler become almost equivalent to the objective value of the omniscient scheduler.

4.7 CONCLUSION

In this Chapter, we discuss the new challenges that are brought by 6DoF VR applications. We focus on a restricted version of 6DoF applications where the scene is simultaneously captured by several synchronized omnidirectional cameras. This Chapter shows that such an application permits different implementations of viewpoint switching. It also describes different options to encode the MVP videos into segments friendly to existing encoders and adaptive streaming technologies. The Chapter introduces the key trade-off the client has to consider when scheduling the video segment downloading to maximize the user experience. We identify the main objective metrics that correlate with the user's feeling of immersion and QoE. We design an optimization model, which bounds the achievable performance, and two client strategies: a reactive client and a proactive client with guaranteed fast viewpoint switch. A real MVP omnidirectional video is used, and real user navigation traces are exploited to evaluate

the performance of these algorithms. Our main observations include that (*i*) tiling improves service performance and is thus a key technology for 6DoF VR applications, and (*ii*) proactive strategies based on anticipated systematic neighboring viewpoint downloads represent an excessive price to pay. A compromise between the reactive strategy (which tends to delay switches) and the proactive strategy (which has to pause the videos to get a decent image quality) has to be found, but to date, the reactive strategy seems a better option.

Being a first step in the field of MVP omnidirectional video systems, this Chapter reveals many open research questions, including (*i*) viewpoint switching prediction (the early results show that it should be possible to exploit the change in user behavior before switching); (*ii*) subjective tests to better understand the interplay between the various options when unanticipated events happen; (*iii*) graphical transition effects during switch, which may allow for delayed viewpoint switches by increasing image quality and reduce stalls; (*iv*) exploiting at the client side multiple decoders in parallel at the client to allow near frame delay viewpoint switching; (*v*) novel encoding strategies based on multi-view correlations, to increase the number of virtual cameras, and thus to get closer to continuous 6DoF VR applications.

Part III

THEORETICAL STUDY & OPTIMIZATION

5

SPHERICAL SAMPLING: AN OFFSET PROJECTION STUDY

5.1 INTRODUCTION

To deliver omnidirectional videos, the content providers implement *viewport-adaptive streaming* solutions [33, 98, 123], where the delivered video is characterized by *heterogeneous spatial quality*: some regions of the frame have a better quality than others [32] (see Section 2.6.1). The motivation is twofold: (*i*) display a high-quality video in the *viewport* of the client, and (*ii*) reduce the delivered bit-rate by encoding less information in the regions that are unlikely to be watched.

In a video with a spatially heterogeneous quality, each pixel is associated with a target quality, which ranges in a given ordered set. The rationale behind the mapping between quality and pixels, whether it comes from a statistical analysis of previous sessions [34, 82, 145] or from a content analysis [78], is out of the scope of this Chapter. We assume that the content provider has defined a small set of qualities and a set of *regions* consisting of contiguous pixels with the same quality. An ideal implementation of spatially heterogeneous quality in an omnidirectional video is characterized by:

PRECISION The *visual quality* of a region in the frame reflects the specifications, both in terms of region boundaries and relative quality with respect to the quality in other regions.

SMOOTHNESS The pixels at the boundary of two contiguous regions enable a smooth transition between both regions.

ENCODING EFFICIENCY The bit-rate budget that is necessary to implement the spatially heterogeneous quality is low with respect to the obtained visual quality of the regions.

UNIVERSALITY The process of preparing the video can be implemented and widely deployed.

REQUIREMENT The resources (in particular, computing power and memory) that have to be provisioned to prepare the video are available in standard media servers.

To prepare heterogeneous spatial quality in videos, the concept of *tiling* has received the most attention. Tiling is offered by the MCTS feature in the HEVC encoder [52, 75, 113, 157] (see Section 2.6.1). The drawbacks of tiling become evident when analyzed with regards to the aforementioned fundamental characteristics. First, the precision of tiling depends on the number of tiles. A large number of tiles comes at the price of a lower encoding efficiency [26] and an increased resource requirement (high signalling overhead and heavier CDN management). Moreover, the visual quality changes abruptly at the boundary of two contiguous tiles. Finally, tiling is not yet widely supported in open source fast encoders. So far, the only fast open-source HEVC encoder to implement tile encoding is *Kvazaar* [69]. Hence, despite being the most widely used method, tiling is not an ultimate solution, and the preparation of heterogeneous spatial quality in omnidirectional videos is still an open research question.

In this Chapter we study how sphere-to-plane projection could be used to overcome the limitations of tiling. We propose a new metric to evaluate the heterogeneous quality of a given projection: the spherical pixel sampling. We propose a formal study of the offset projections [162, 163]. The idea is for the content provider to patch a "classic" sphere-to-plane projection to map more pixel of the planar image onto a given area on the sphere (i. e. increase the spherical sampling of this area). Despite the interest in this approach, the impact of the offset parameters on the visual quality after video encoding has never been formally studied.

In our MMSP'18 paper [54], we study a second approach named *Gaussian Pyramid Composition*. The idea is to process the input video pixel-wise using multiple decreasing qualities, arranged in a *Gaussian pyramid*. For each pixel in the output video, the content provider picks a pixel from a Gaussian level with respect to the expected quality at this pixel. The results is that low quality part of the content is blurred and become easier to compress. Such a pixel-wise approach has never been explored for preparing heterogeneous spatial quality in omnidirectional videos but we will not discuss about this proposition in this Chapter.

This Chapter is organized as follow. Section 5.3 introduces the mathematical principles of the spherical sampling. Section 5.4 describes the formally the offset projection and analyse the correlation between the spherical pixel sampling and the visual quality of uncompressed content. Finally Section 5.5 compare the offset projection (unequal pixel sampling approach) with the tiling (leveraging encoder distortion approach) and Section 5.6 conclude the Chapter.

5.2 DEFINITIONS

The concept of heterogeneous spatial quality is defined by two input parameters. First, we define $\mathcal{Q}$ as an ordered set of qualities. This set is not formally associated with any precise measurable scale. Following the definition from MPEG [59], the quality is a rather vague indicator. We thus consider that there exist Q qualities in $\mathcal{Q}$, denoted q_i for $i \in \{0, 1, \ldots, Q-1\}$, where q_0 is the lowest quality, and q_{Q-1} is the best quality.

Second, we define a set of *regions*, which are sets of contiguous pixels in a frame. The idea behind the concept of quality-region is the following. The content provider considers that the pixels in a given region have approximately the same probability to be displayed in the viewport, so they must be encoded at the same quality, i. e. the higher the probability to be displayed in the viewport, the higher the quality. Here, we do not restrict the regions to spherical rectangles. On the contrary, we assume that the regions may have any shape. We denote the set of regions by $\mathcal{R}$. The regions do not overlap and cover the whole frame. The quality of a region $R \in \mathcal{R}$ is a spatial function denoted $q(R) \in \mathcal{Q}$. The qualities of any two regions in $\mathcal{R}$ can be either different or the same. In Chapter 6 we model the quality allocation for each regions based on regions viewing probabilities and on the number of representation the content provider is ready to generate.

5.3 SPHERICAL PIXEL DENSITY

5.3.1 *Theory*

To study how plane-to-sphere projection, and especially the offset projections introduced in Section 5.4, continuously degrades the quality of the spherical image, we measure the sampling density of the projection on the sphere. The sampling density, sometimes denoted only as sampling, is a number that represents the number of pixels per surface unit on the sphere, it is measured in m^{-2}. It can be computed for any point on the sphere. If we consider only a finite number of pixels $W \times H$ on the plane, a point p on the sphere is projected inside a unique pixel s_p on the plane. The sampling density σ at point p on the sphere can be approximated by the inverse of the surface of the pixel s_p, once s_p is projected back on the sphere. If each pixel have exactly the same surface on the sphere, the sampling is uniform. For traditional planar pictures, each pixel has the same surface on the plan so the planar sampling density is uniform.

We are interested in studying how a plane-to-sphere projection affect the spherical sampling. The planar picture contains a uniform grid of rectangular pixel, but once projected on the sphere this grid is not uniform anymore. We denote by $f : \mathbb{R}^3 \to \mathbb{R}^2$ a piecewise continuously differentiable sphere-

to-plane projection and by $f^{-1} : \mathbb{R}^2 \to \mathbb{R}^3$ the corresponding piecewise continuously differentiable plane-to-sphere projection. The piecewise continuity is needed to allow a study of pixel neighborhood on the sphere from the planar picture. In traditional projection used with omnidirectional video, this continuity is wanted to allow good compression efficiency with traditional planar video encoder.

Metric tensors are mathematical tools of *tensor geometry* introduced, among other things, to study the deformations generated by this kind of projections [43]. In this Section we will present a simplified overview of the mathematical tools needed to compute the spherical sampling from the plane-to-sphere projection. Reader interested in more details may for instance read the book "*Tensor Geometry*" written by Dodson and Poston [43].

$\vec{\mathbf{v}_\mathbf{u}} = \frac{\partial f^{-1}}{\partial u}(u,v)$ is a vector in $\mathbb{R}^3$ corresponding to the variation of the function f^{-1} at the position (u,v) on the planar picture when applied an infinitesimal variation du on u: $\frac{\partial f^{-1}}{\partial u}(u,v) \approx \frac{f^{-1}(u+du,v)-f^{-1}(u,v)}{du}$. Similarly $\vec{\mathbf{v}_\mathbf{v}} = \frac{\partial f^{-1}}{\partial v}(u,v)$ represents the variation of f^{-1} at position (u,v) for an infinitesimal variation of v.

On the planar picture, pixels are aligned on the grid formed by $\vec{\mathbf{u}}$ and $\vec{\mathbf{v}}$. If there were an infinite number of pixel, the pixel would be spaced horizontally (respectively vertically) with an infinitesimal distance du (respectively dv). So on the sphere, neighbor pixels of $f^{-1}(u,v)$ are $f^{-1}(u,v) + \vec{\mathbf{v}_\mathbf{u}}du$ and $f^{-1}(u,v) + \vec{\mathbf{v}_\mathbf{v}}dv$. The surface of the parallelogram generated by $\vec{\mathbf{v}_\mathbf{u}}$ and $\vec{\mathbf{v}_\mathbf{v}}$ is then a good approximation of the infinitesimal surface s of the pixel at position (u,v) when there are an infinite number of pixel on the planar picture: i. e. $s = \|\vec{\mathbf{v}_\mathbf{u}} \times \vec{\mathbf{v}_\mathbf{v}}\|$ and $\sigma = \frac{1}{s}$.

5.3.2 *Equirectangular Projection Example*

In this Section, we will illustrate the theory on spherical sampling introduced in the previous Section by studying the Equirectangular projection already introduced in Section 2.3.3.

We know that

$$f(u\vec{\mathbf{u}} + v\vec{\mathbf{v}}) \triangleq \cos(\frac{2u\pi}{W})\cos(\frac{v\pi}{H})\vec{\imath} \quad - \sin(\frac{2u\pi}{W})\cos(\frac{v\pi}{H})\vec{\jmath} \quad - \sin(\frac{v\pi}{H})\vec{\mathbf{k}}$$

So, by applying partial derivation formula, we get

$$\vec{\mathbf{v}_\mathbf{u}}(u,v) = \frac{2\pi}{W}(-\sin(\frac{2u\pi}{W})\cos(\frac{v\pi}{2H})\vec{\imath} \quad - \cos(\frac{2u\pi}{W})\cos(\frac{v\pi}{2H})\vec{\jmath} \quad)$$

and

$$\vec{\mathbf{v}_\mathbf{v}}(u,v) = \frac{\pi}{H}(-\cos(\frac{2u\pi}{W})\sin(\frac{v\pi}{2H})\vec{\imath} \quad + \sin(\frac{2u\pi}{W})\sin(\frac{v\pi}{2H})\vec{\jmath} \quad - \cos(\frac{v\pi}{2H})\vec{\mathbf{k}})$$

Hence $s(u,v) = \frac{2\pi^2}{HW}|\cos(\frac{v\pi}{H})|$ and $\sigma(u,v) = \frac{HW}{2\pi^2\cos(\frac{v\pi}{H})}$

It confirms that the sampling is constant at a given elevation (i. e. for constant v) and increases the closer we get to the poles. The sampling diverges to infinity at the pole (i. e. when $v = \pm\frac{H}{2}$). See Figure 2.11a for a representation of the normalized pixel density of the equirectangular projection: ${}^{\sigma(u,v)}/_{\sigma(0,0)}$.

5.4 OFFSET PROJECTION

The *offset transformation* [73] is a bijective sphere-to-sphere transformation, which increases the quality of an omnidirectional video near an emphasized direction $\vec{\mathbf{b}}$. When two pixels on the sphere are close to (respectively far from) the emphasized direction $\vec{\mathbf{b}}$, the offset transformation decreases (respectively increases) the angular distance between the two pixels. The *offset projection* is a composition of a

sphere-to-plane projection with an offset transformation. First, the spherical image is distorted using the offset transformation, then the distorted spherical image is projected to a plane using a sphere-to-plane projection. The offset projection maps on the plane more pixels close to $\vec{\mathbf{b}}$ and fewer pixels far from $\vec{\mathbf{b}}$ compared to the original sphere-to-plane projection. The spatial sampling (thus, the quality) of the video decreases when the angular distance to $\vec{\mathbf{b}}$ increases.

5.4.1 *Theory*

The offset projection has been experimentally studied by Zhou, Li, and Liu [162] and Zhou et al. [163] in the case of the cube-map projection. It is characterized by the following parameters: a projection function $f : \mathbb{R}^3 \to \mathbb{R}^2$, an emphasized direction $\vec{\mathbf{b}}$, and an offset *amplitude* α. The projection function f can be any sphere-to-plane projection. The emphasized direction $\vec{\mathbf{b}}$ is a unit vector, pointing to the direction of space emphasized by the offset projection, and α is a real value in $[0, 1)$.

A sphere-to-plane projection maps a spherical pixel, characterized by a unit vector $\vec{\mathbf{a}}$ pointing from the origin of the sphere to the pixel, to a point (u, v) on the plane.

The offset transformation $F : \mathbb{R}^3 \to \mathbb{R}^3$ distorts the sphere by mapping the vector $\vec{\mathbf{a}}$ to the vector $(\vec{\mathbf{a}} + \alpha\vec{\mathbf{b}})/\|\vec{\mathbf{a}} + \alpha\vec{\mathbf{b}}\|$. The offset projection $g : \mathbb{R}^3 \to \mathbb{R}^2$ is then given as follows:

$$g(\vec{\mathbf{a}}) \triangleq f \circ F(\vec{\mathbf{a}}) = f\left((\vec{\mathbf{a}} + \alpha\vec{\mathbf{b}})/\|\vec{\mathbf{a}} + \alpha\vec{\mathbf{b}}\|\right) \tag{5.1}$$

In Equation (5.1), both projection functions f and g transform a given viewing direction $\vec{\mathbf{a}}$ into planar coordinates (u, v). The inverse offset projection g^{-1}, transforming coordinates (u, v) on the planar picture into a viewing direction $\vec{\mathbf{a}}$, is defined as follows:

$$g^{-1}(u, v) = F^{-1} \circ f^{-1}(u, v) \tag{5.2}$$

where f^{-1} is the plane-to-sphere projection corresponding to f, and F^{-1} is:

$$F^{-1}(\vec{\mathbf{a}}) = \left(\vec{\mathbf{a}} \cdot \alpha\vec{\mathbf{b}} + \sqrt{(\vec{\mathbf{a}} \cdot \alpha\vec{\mathbf{b}})^2 - \alpha + 1}\right)\vec{\mathbf{a}} - \alpha\vec{\mathbf{b}} \tag{5.3}$$

with $\cdot$ being the inner vector product defined in Section 2.2.

In what follows, $\sigma_{\alpha,r}$ denotes the sampling density function for the equirectangular offset projection with an amplitude α, an emphasized direction $\vec{\mathbf{b}} = (1, 0, 0)$ and a resolution $r(W \times H)$. Hereafter, $r \in [0, 1]$ refers to the resolution ratio.

Increasing the amplitude α for a constant r increases the sampling near the emphasized direction and decreases it near the direction opposite $\vec{\mathbf{b}}$. Put differently, when α increases, more pixels from the plane are assigned to spherical angles near the emphasized direction and less in the opposite direction. Figure 5.1 depicts the offset sampling density compared to the sampling density at the same position in the original video, *i.e.* it depicts $\sigma_{\alpha,r}/\sigma_{0,1}$. When $\alpha = 0$, the sampling density ratio is constantly equal to r. As illustrated by the curve for $r = 0.36$ in Figure 5.1, when α tends towards 1, the sampling density ratio tends toward 0 for points farther than 90° from $\vec{\mathbf{b}}$, and tends towards $4r$ in the emphasized direction $\vec{\mathbf{b}}$ (*i.e.* tends towards 1.44 for $r = 0.36$). The latter means that the offset projection with an amplitude $\alpha > 0$ cannot increase the sampling $\sigma_{0,r}$ in the emphasized direction more than four times the sampling for $\alpha = 0$. Indeed, if we apply the computation step introduced in Section 5.3 on g^{-1} with f^{-1} being the equirectangular projection, we get in the emphasized direction:

$$\sigma_{\alpha,r}(0, 0) = \sqrt{\frac{8\alpha^6 - 16\alpha^5 + 25\alpha^4 - 19\alpha^3 + 12\alpha^2 + 4(2\alpha^5 - 3\alpha^4 + 4\alpha^3 - 2\alpha^2 + \alpha)\sqrt{\alpha^2 - \alpha + 1} - 3\alpha + 1}{\alpha^2 - \alpha + 1}}\, r$$

which is equal to $4r$ when *alpha* tends towards 1.

Furthermore, when $\alpha > 0$, the sampling decreases with the distance from the $\vec{\mathbf{b}}$ and the transition between the emphasized and non-emphasized regions is smooth. Hereafter, we define the QER for the

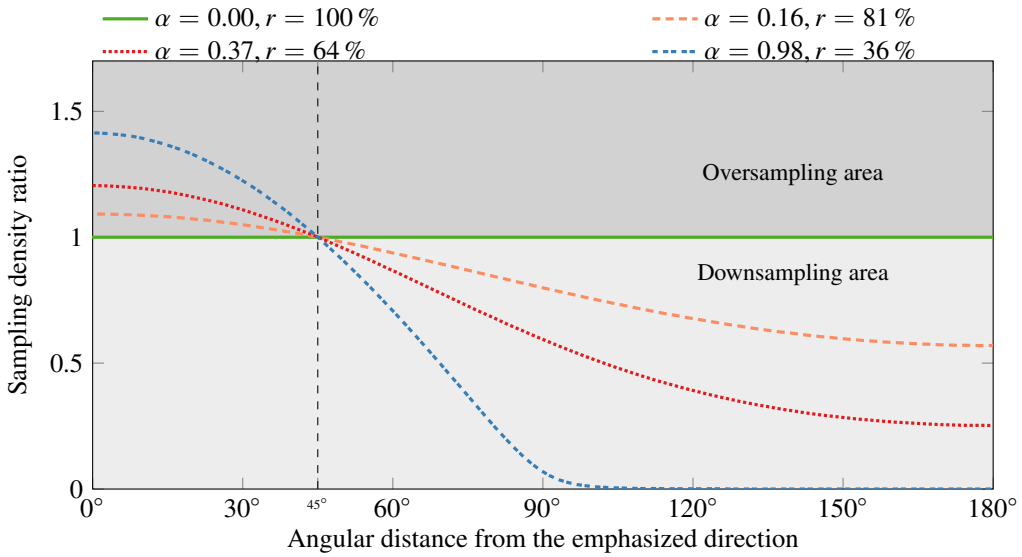


Figure 5.1: Sampling ratio $\sigma_{\alpha,r}/\sigma_{0,1}$ for the equirectangular offset projection for $\vec{\mathbf{b}} = (1,0,0)$ at constant latitude $\frac{\pi}{2}$.

offset projection as being the zone of the video for which the sampling density ratio is at least equal to the original sampling ratio (in Figure 5.1, the original sampling ratio is constant and equal to 1). For a given resolution ratio r, there exists at most one value of α such that the angular size of the QER is equal to a given value. For instance, Figure 5.1 shows that for $r = 0.64$, a region of angular size 90° (i. e. 45° from $\vec{\mathbf{b}}$) is a QER when $\alpha = 0.37$ (resp. $\alpha = 0.98$ for $r = 0.36$ and $\alpha = 0.16$ for $r = 0.81$). Figure 5.2 illustrates, for the *rollercoaster* video used in Chapter 8, how an equirectangular picture is distorted when using the offset transformation with those parameters (the offset amplitude α and the downsampling ratio r).

5.4.2 *Experiments and analysis*

We now evaluate the impact of r and α on the visual quality of the videos. So far, only Zhou et al. [163] have studied the quality distortion caused by the offset approach. The authors compare the video quality degradation of the offset cube-map projection to the quality of the original equirectangular video. Their experiments show that, for a given α, the offset cube-map projection produces videos with similar quality as a video at a higher resolution within a certain angular distance from the offset center. However, Zhou et al. do not elaborate on the choice of α and, more importantly, how this choice influences the quality of the offset videos. Hereafter, we present a more thorough analysis of the quality distortion, caused by the offset (equirectangular) projection given various video resolutions and various values of the offset amplitude.

We applied the offset equirectangular projection on three omnidirectional videos using three resolution ratios, *i.e.* $r \in \{0.81, 0.64, 0.36\}$, and five amplitude values, i. e. $\alpha \in \{0, 0.25, 0.5, 0.75, 0.9\}$, for each resolution ratio r. The three videos are *rollercoaster*, *venice*, and *timelapse* from our public dataset [34] described in Chapter 8. To reduce the content-related bias in the quality measurement, we computed the offset projection *eight* times per video by applying different sphere rotations (yaw rotations of 0 rad, $\pi/4$ rad, $\pi/2$ rad, $3\pi/4$ rad, π rad, $5\pi/4$ rad, $3\pi/2$ rad and $7\pi/4$ rad). To measure the distortion, introduced by the offset projection, we computed the PSNR on the luma components of the pictures between viewports from both the original equirectangular video and the offset videos. We extracted viewports with FoV 110°×90° and resolution 1920×1080 at varying angular distances from

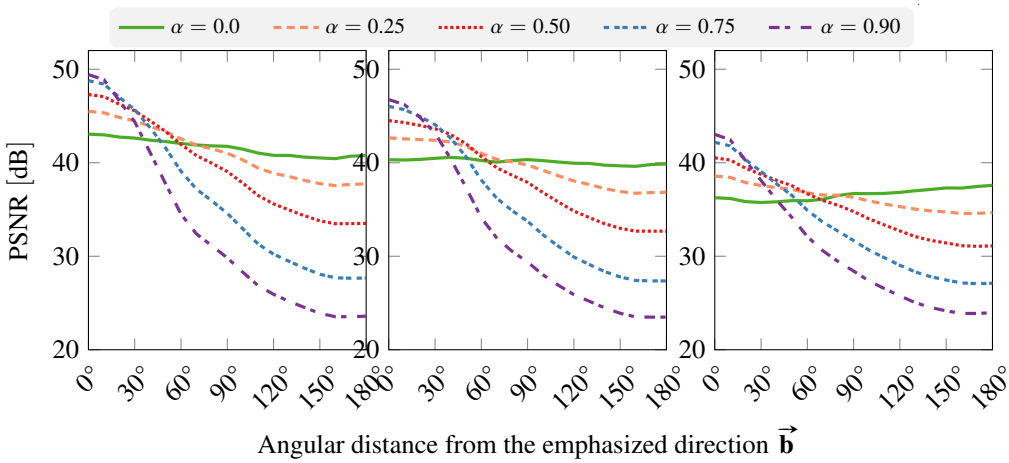
(a) Resolution: 100 % (3840 × 2048) $\alpha = 0$

(b) Resolution: 81 % (3456 × 1844) $\alpha = 0.16$

(c) Resolution: 64 % (3072 × 1638) $\alpha = 0.37$

(d) Resolution: 36 % (2304 × 1228) $\alpha = 0.98$

Figure 5.2: Examples of offset projection applied on equirectangular pictures. The resolution ratio between the different version is respected.



(a) Resolution: 81 % **(b)** Resolution: 64 % **(c)** Resolution: 36 %

Figure 5.3: PSNR between viewports, extracted from the original equirectangular video, and viewports, extracted from the equirectangular offset projection for various values of the amplitude α and three resolution ratios r.

the emphasized direction $\vec{\mathbf{b}}$ to illustrate the relation between quality degradation and distance to the offset center.

Figure 5.3 shows the PSNR quality curves for each considered resolution ratio r and each amplitude α. The curves for $\alpha = 0$, i. e. when no offset is applied, are referred to as baselines. Each baseline measures the distortion caused by the resolution decrease (by r). The PSNR remains constant regardless of the distance to the spherical center. In contrast, for $\alpha > 0$, the video quality varies depending on the angular distance to the emphasized direction $\vec{\mathbf{b}}$. The PSNR curves show that, for $\alpha > 0$, the quality of the viewports close to $\vec{\mathbf{b}}$ is higher than the baseline quality. The farther we get from the center of emphasis, the higher the distortion.

Furthermore, for all resolution ratios r, the more we increase the amplitude α, the more we improve the video quality near the emphasized direction $\vec{\mathbf{b}}$ and the more we degrade the quality near the opposite direction to $\vec{\mathbf{b}}$. The plots in Figure 5.3 show that the quality curves intersect the baseline at smaller angles when α increases. For instance, the curve for $\alpha = 0.5$ intersects the baseline at 60°, whereas the curve for $\alpha = 0.9$ intersects the baseline at 40°. The latter means that the higher the amplitude α, the smaller the emphasized region. As shown in Figure 5.1, for high amplitude values, e. g. $\alpha = 0.98$, most of the samples are concentrated in a small region, centered at the emphasized direction $\vec{\mathbf{b}}$, and there are less samples for regions away from the center of emphasis. This explains the rapid decrease in the quality when the distance from $\vec{\mathbf{b}}$ increases.

Figure 5.4 depicts the relation between the PSNR measured experimentally inside extracted viewports and the spherical pixel density computed using the equations introduced in Section 5.3. In this Figure we only consider the density variation due to the offset transformation, ignoring the extra oversampling introduced by the equirectangular projection. Here a pixel density equal to 1 m^{-2} means same pixel density at the same position on the sphere for the equirectangular projection with the original resolution ($r = 1$). Point with the same color uses the same value for α and points with the same shape use the same value for the resolution ration r. We observe that for a given set of parameter α and r, then spherical pixel density and the PSNR are highly correlated (Pearson Correlation Coefficient (PCC) between 0.986 and 0.998). For a given *ampha* (i. e. points with the same color), we also observe a very high linear correlation (PCC between 0.965 and 0.990). For a fixed resolution r, the linear correlation is lower but still significant (PCC between 0.811 and 0.813). Globally the PCC is equal to 0.814 showing a high linear correlation between the PSNR and the pixel density on the sphere. We observe that when the pixel density is close to zero (for high offset amplitude α at opposite direction to $\vec{\mathbf{b}}$) the PSNR stop

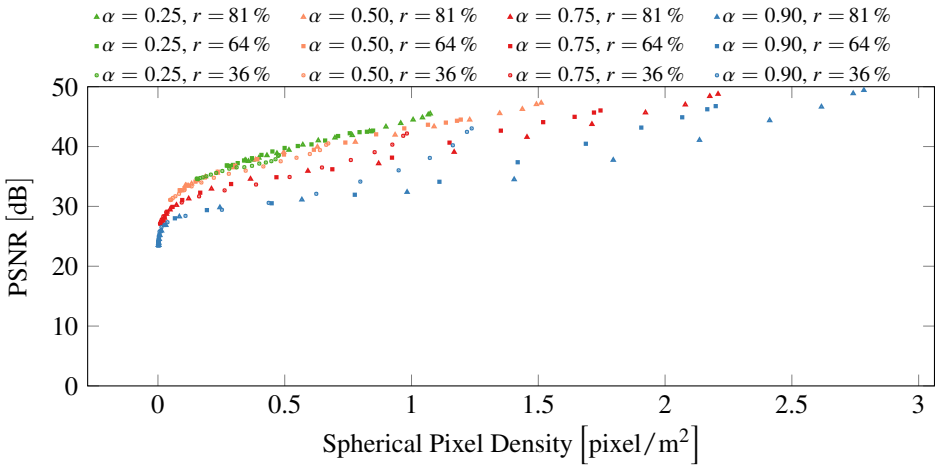


Figure 5.4: Correlation between the experimentally measured PSNR and the theoretical computed spherical pixel density

being linearly correlated with the spherical pixel density. Future work should study how this correlation stand with better quality metric and with other projections.

5.5 EVALUATION

In Table 5.1, we summarize the main differences between the two approaches tested in this Chapter with regards to four out of the five main characteristics introduced in Section 5.1. The analysis of the computing and storage requirements for composing omnidirectional videos as well as the user reaction to abrupt quality changes are left for future work. Here, we focus on the encoding efficiency by measuring the visual quality of the videos created using the three approaches, for a similar overall bit-rate.

	Tile	**Offset**
Precision	depends on the nb. of tiles	low
Smoothness	no	yes
Universality	require specific encoder	require metadata to undo the projection
Requirements	depends on the nb. of tiles	low

Table 5.1: Summary of main characteristics

Visual Quality Metric. We lack a metric to capture the heterogeneity of quality in a omnidirectional video. To fill this gap, we created a new metric based on the Spherical Peak Signal to Noise Ratio (S-PSNR) introduced by Yu, Lakshman, and Girod [152]. The S-PSNR first maps some predefined points on the sphere to their corresponding location on the original and the encoded video to interpolate their color. Then, the color values are used to compute the distortion between the original video and the encoded video. This distortion can be computed even if the original and the encoded planar picture have not the same resolution or do not use the same sphere-to-plane projection (because the color errors are computed on the sphere). Finally, the errors for all spherical points are averaged to compute the S-PSNR. However, the concept of heterogeneous spatial quality in omnidirectional videos takes its root from the fact that not all spherical pixels are equal (as some pixels have higher probability to be in the clients' viewports). To capture this heterogeneity, we propose to weigh the error of each pixel with respect to its expected quality. We thus use our quality function $q(R)$ to weigh the S-PSNR and obtain a new *weighted* metric, denoted Weighted Region of Interest - Spherical - Peak Signal Noise to

Ratio (WRoI-S-PSNR). The highest weight is assigned to spherical points which, after projection, lie in the region with the highest quality. The weights of the spherical points lying within the other quality regions decrease exponentially.

To sum up, the S-PSNR measures the average quality of the entire encoded video, whereas the WRoI-S-PSNR measures the quality with respect to the given quality regions. We combine both metrics to obtain an *interpolated* quality estimation, denoted IS-PSNR.

Quality Regions. We consider two configurations of quality regions (Figure 5.5), which represent common viewport movements [32]. The □-*shape* is common for a video where viewers stably focus on a single location, whereas the *V-shape* may appear when the object of interest is moving.

Video Preparation. We use three equirectangular omnidirectional videos from our public dataset [34] (*rollercoaster*, *venice*, and *timelapse*). Their resolution is 3840×2048 (4K). We use the Kvazaar encoder [69] with three bit-rate targets: $6\,\text{Mbit}\,\text{s}^{-1}$, $9\,\text{Mbit}\,\text{s}^{-1}$ and $14\,\text{Mbit}\,\text{s}^{-1}$. Specific settings include:

TILES We set 8×8 tiles. We encode four tiled videos with four bit-rate targets ($3\,\text{Mbit}\,\text{s}^{-1}$, $7\,\text{Mbit}\,\text{s}^{-1}$, $13\,\text{Mbit}\,\text{s}^{-1}$ and $21\,\text{Mbit}\,\text{s}^{-1}$). We extract each tile from each tiled video into independent files using *GPAC MP4Box* toolkit. To obtain a quality-variable video matching the quality regions and the bit-rate target, we select for each tile one of the four qualities and merge the bit-stream together so that the overall bit-rate is close to the target.

OFFSET We chose two resolutions: 3456×1844 ($r = 81\,\%$ of the original video) and 2304×1228 ($r = 36\,\%$). The offset intensity α is set so that a 90° of the QER (resp. 50°) is oversampled for the □-shape (resp. V-shape).

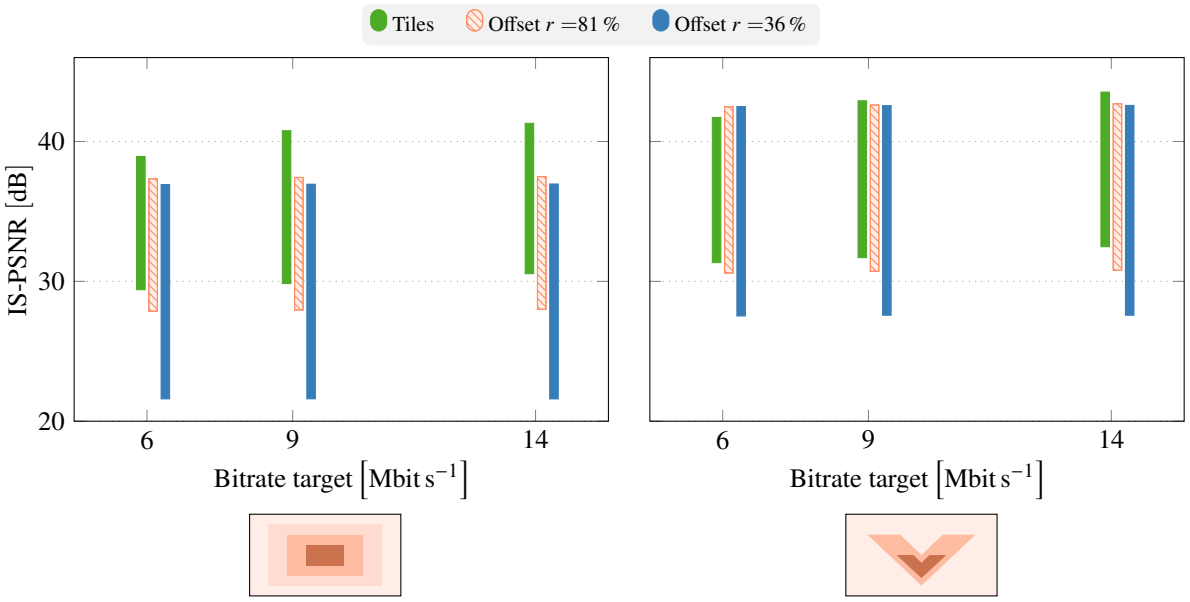
Result Analysis. In Figure 5.5a (resp. in Figure 5.5b), we show the IS-PSNR for the three bit-rate targets and for the □-shape (resp. V-shape) quality regions. The lowest and the highest values of the bars correspond to the S-PSNR and the WRoI-S-PSNR respectively. In Figure 5.6, we show snapshots of the back viewports and thus highlight the different approaches (distorting, and reducing bit-rate by encoding).

First, we observe that all approaches manage to prepare omnidirectional videos with heterogeneous qualities. In each configuration, the visual quality in the emphasized regions, measured by the WRoI-S-PSNR, is higher than the visual quality, measured by the S-PSNR. The difference between both metrics is greater or equal to 10 dB, which is a significant gap.

The offset approach is relevant for low encoding bit-rates, but it does not benefit from extra bit-rates. Also, the offset is sensitive to the viewport location, with large quality range between the S-PSNR and the WRoI-S-PSNR, especially at low resolutions. To obtain a high quality in a large QER despite the low resolution, the price we pay is a severely degraded quality in the opposite direction to $\vec{\mathbf{b}}$, as epitomized in Figure 5.6. Finally, the tiling approach appears more stable. It offers a consistent good quality in the best cases and the quality never reaches very low levels (unlike the offset). The back viewport in Figure 5.6 has also a good visual quality. However, tiling requires specific encoders, and the abrupt changes between tiles can degrade user's experience.

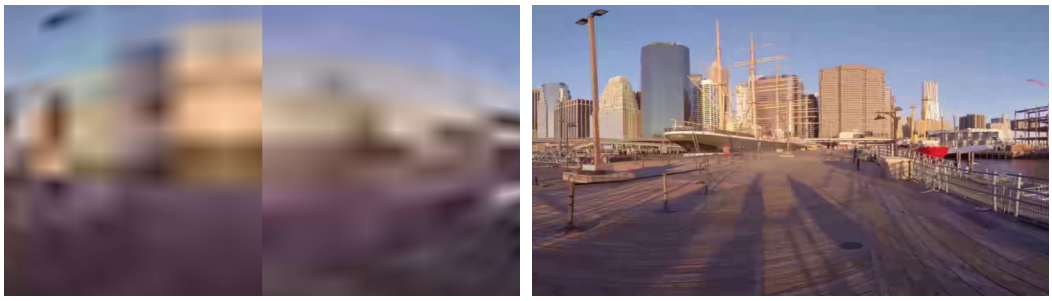
5.6 CONCLUSION

In this Chapter, we study the preparation of heterogeneous spatial quality in omnidirectional videos. We propose a novel metric named spherical pixel density and present the theoretical framework one can used to computed this metric from the plane-to-sphere projection. We then analyze the theoretical principles of the offset projection. We identify two families of approaches to generate heterogeneous spatial quality video: a quality degradation by the encoder (tiling), and an unequal projection (offset



(a) □-shape **(b)** V-shape

Figure 5.5: IS-PSNR bars for two quality regions



(a) Back Offset 36 % **(b)** Back Tiles

Figure 5.6: Viewport in the non-QER for the 6 Mbit s^{-1} videos

projection). We compare the two approaches in terms of encoding efficiency. A third approach (a blurring approach) is introduced in our MMSP paper [54].

This Chapter presents one of the first work that formally study heterogeneous spatial quality in omnidirectional videos. Each of the proposed approaches deserves a deeper analysis to better understand the impact of the settings on the overall performance. We show a high correlation between the spherical pixel density metric and the PSNR measured in user's viewports in the context of equirectangular offset projection. This correlation could be used to predict the impact of different projections on the users QoE. The integration of the propositions in the global delivery chain of viewport-adaptive streaming solutions also deserves a deeper analysis with respect to the constraints of the services, including definition of quality regions, live requirements, and implementation in CDN.

6

OPTIMAL QUALITY EMPHASIZED REGION FOR VIEWPORT ADAPTIVE STREAMING

6.1 INTRODUCTION

Previous Chapters have introduced the viewport-adaptive streaming as a solution to stream, with little modifications to existing streaming technologies, omnidirectional videos with a limited wast of bandwidth and a higher average quality withing users' viewports [33, 52, 93, 123]. Chapter 3 presents an example of viewport-adaptive streaming architecture. We also presented the concepts of *heterogeneous spatial quality* encoding. Different implementations of heterogeneous spatial quality are possible which can be based on tiling [157] or offset projections [162] as presented in Chapter 5, or based on scalable coding [16, 151].

The design of efficient viewport-adaptive streaming systems requires the understanding of the complex interplay between the most probable viewport positions, the coding efficiency, and the resulting QoE with respect to the traditional constraints of delivery systems such as bandwidth and latency. MPEG experts have proposed the concept of *quality region*, which is a rectangular region defined on a sphere, characterized by a quality level ranging from 1 to 100. The main idea is that the content provider determines some quality regions based on offline external information (e. g., content analysis and statistics about viewport positions), and then prepares multiple quality-variable versions of the same omnidirectional video based on these quality regions.

We provide in this Chapter a theoretical analysis of this concept of quality regions for omnidirectional videos. We present an optimization model to determine the optimal quality distributions, subject to a population of clients, the number of quality-variable video versions to generate, and the bandwidth. We aim at maximizing the video quality displayed in the client viewports by identifying (*i*) the location of the quality region, (*ii*) their dimensions (or area size), and (*iii*) the quality inside and outside the regions. Our model enables content providers to prepare omnidirectional videos based on the analytics of the head movements collected from the first content consumers. Using a dataset of real head movements captured on an HMD, and further described in Chapter 8, we study an optimal set of video versions that are generated by our algorithms and evaluate the performance of such optimal viewport-adaptive streaming. We demonstrate that, for a given overall bit-rate, the bit-rate spent on covering the viewports can be increased by 102 %, thus significantly improving the quality perceived by the user.

6.2 RELATED WORK

In this Section we present the related work on viewport-adaptive streaming and on RoI. Related work on heterogeneous spatial quality video can be find in Section 2.6.1 and chapter 5.

In this Chapter, we focus on an optimization model to generate heterogeneous spatial quality video versions for viewport-adaptive streaming that maximize the quality inside users' viewports when number of video versions available to the user is limited. To the best of our knowledge, nobody studied before us optimal parameters to generate limited number of heterogeneous spatial quality versions for omnidirectional videos.

6.2.1 *Regions of Interest*

Our work has also some common roots with the literature on RoI in video delivery. The human vision system can only extract information at high resolution near the *fovea*, where the gaze focuses its attention; the vision resolution decreases with eccentricity. Within the same video picture, it is common that most users focus their gaze on some specific regions of the picture, named RoI. Researchers have studied saliency map, which measures the gaze location of multiple users watching the same video. The goal is to extract RoI and, if possible, to corroborate RoI with picture structures to enable automatic RoI prediction [15, 42]. However, the concept of saliency map should be revisited with omnidirectional videos, because the head movement is the prevailing factor to determine the attention of users. To the best of our knowledge, the relation between gaze-based saliency map and head movements in HMD has not been demonstrated.

The attention-based video coding [13, 62, 78, 110] is a coding strategy, which takes advantage of the gaze saliency prediction. The quantization parameters of the encoder are adjusted to allocate more bits near the different RoI and less bits farther away. A live encoder can perform attention-based video coding by using either feedback from a set of specific users or predicted RoI.

We revisit this approach to omnidirectional videos in this Chapter. Our work is both to study per-segment RoI localization based on head movement information and to generate RoI-based encoded video representations. The creation of spherical heterogeneous spatial quality video versions based on head movement analysis enables viewport-adaptive streaming in the same manner that saliency map and attention-based video coding enable efficient video delivery on regular planar videos [42].

6.3 HETEROGENEOUS SPATIAL QUALITY VIDEOS

We first introduce a model for heterogeneous spatial quality omnidirectional videos and then provide some illustrations of this model on some implementation proposals.

6.3.1 *Generic Model*

Spherical videos. The unit sphere that underlies the omnidirectional video is split into N non-overlapping *areas* that cover the full sphere. The set of areas is denoted by $\mathcal{A}$. In essence, each area corresponds to a part of the video signal projected on the sphere. Let us denote by s_a the surface of an area a on the sphere and observe that the smallest possible surface s_a is the pixel (in which case the set $\mathcal{A}$ is the full signal decomposition and N is the video resolution). However, video preparation processes are generally based on a video decomposition $\mathcal{A}$ with larger surface s_a, such as the concept of *tiles* in HEVC [85]. For the preparation of omnidirectional videos, any decomposition of the video into $\mathcal{A}$ can be considered if it respects that it covers the whole sphere, formally $\sum_{a\in\mathcal{A}} s_a = 4\pi$.

Area Quality. The goal of a video encoder is to compress the information of the video signal corresponding to a given area a into a decodable byte-stream (lossy compression generating distortion when the video is eventually played). An encoder uses a compression algorithm with various parameter settings to encode the video. For a given encoder, the more compression due to the encoding settings, the more distortion in the decoded and played video. Using MPEG terminology, we use the generic term *quality* to express the settings of the encoding scheme on a given area, regardless of the used area encoding process. The number of different ways to encode areas is finite, which results in a set of available qualities Q for this encoder (typically this set of quality ranges can be advertised by the *region-wise quality ranking* from MPEG-OMAF). The set Q is totally ordered with a transitive comparison function, noted with $>$.

We provide some natural notations: $q_{\min}$ (respectively $q_{\max}$) is the lowest (respectively highest) possible quality for areas. The encoder processes an area $a \in \mathcal{A}$ with a quality q to generate a byte-stream of size $b_{a,q}$. Given the usual strictly increasing feature of the rate-distortion performance of video encoders, we get that if a quality $q_1 \in Q$ is better than a quality $q_2 \in Q$ (formally $q_1 > q_2$), then we have $b_{a,q_1} > b_{a,q_2}$, $\forall a \in \mathcal{A}$.

Video Version. We use the term *version* to represent the transportable full video signal byte-stream. It is the video as it can be delivered to clients. Based on the definitions of areas and qualities, a version is a function that associates with every area $a \in \mathcal{A}$ a unique quality $q \in Q$, which corresponds to the encoding quality of a. Let us denote by $\mathcal{R}$ the set of all possible versions. Please note that the number of possible versions is finite since both the set of areas $\mathcal{A}$ and the set of qualities Q are finite. However, the number of different versions is $N^{|Q|}$. We use the notation $r(a)$ to denote the quality q that corresponds to the quality at which the area $a \in \mathcal{A}$ is encoded in the version $r \in \mathcal{R}$.

Let B be a positive real number. We denote by $\mathcal{R}_B$ the subset of versions in $\mathcal{R}$ such that $r \in \mathcal{R}_B$ satisfies that the sum of the byte-stream sizes for every area $a \in \mathcal{A}$ is equal to B. Formally, we have :

$$\forall r \in \mathcal{R}_B, \quad \sum_{a \in \mathcal{A}} b_{a,r(a)} = B$$

Viewport. One of the peculiarities of omnidirectional videos is that at a given time t a user u watches only a fraction of the whole video, which is generally called the *viewport*. The viewport displays only a subset of all the areas of the sphere. Let $v_{u,t,a}$ be a real number equal to the ratio of the surface of area a that is inside the viewport of user u at time t and let $v_{u,a}$ be the average value of $v_{u,t,a}$ during all time t in a video segment: $v_{u,a} = \sum_t v_{u,t,a}/T$, with T the duration of the segment. With respect to the same definition of quality, we have that the average viewport quality during a video segment can be defined as being the sum of the qualities of all the areas that are visible in the viewports, formally $\sum_a v_{u,a} \cdot r(a)$. In practice, the satisfaction of the user watching a viewport is more complex since it depends not only on the visible distortion of the different areas in the viewport but also on the possible effects that different levels of distortion on contiguous areas can produce. Nevertheless, for the sake of simplicity, and with regards to the lack of formal studies dealing with subjective satisfaction evaluation of multi-encoded videos, we consider here that the satisfaction grows with the sum of qualities of the visible areas.

6.3.2 *Illustration: Offset Projections & Tiling*

Offset Projections. We introduced the offset projection in Section 5.4, and saw that this projection is characterized by four parameters: the original plane-to-sphere projection f^{-1}, the offset direction $\vec{\mathbf{b}}$, the offset amplitude α, and the resolution ratio ∇ (denoted r in the previous Chapter). We show how to compute the spherical pixel density variation σ introduced by the offset transformation and indicates its linear correlation with the viewport distortion. It is then possible to model the offset projection by the set of version $r \in \mathcal{R}$ such that $\forall a \in \mathcal{A}$,

$$r(a) = c_1 \frac{\sigma(a)}{\sigma(\vec{\mathbf{b}})} r(a_{offset})$$

with $\sigma(a)$ the spherical pixel density in the direction of the center of a, and c_1 a real constant.

Tiling. To model a tiled version, we consider only versions $r \in \mathcal{R}$ such that for all areas a and a' belonging to the same tile, $r(a) = r(a')$. In other word, we consider only versions that assigns a unique quality within each tile.

6.4 VIEWPORT-ADAPTIVE STREAMING

An adaptive streaming system is modeled as being one client and one server, where the server offers J different versions of the video, and the client periodically selects one of these versions based on a *version selection algorithm*.

Server. The main question is to prepare J versions in $\mathcal{R}$ among all the possible combinations of qualities and areas. In the practical omnidirectional video streaming system described by Zhou, Li, and Liu [162], the number of versions J is equal to 30, while the solution that is promoted by Niamut et al. [93] is to offer all the combinations of tiles (typically 8×4) and qualities (typically 3). In practice, a low number of versions J is suitable since it means less files to manage at the server side (96 files in the latter case) and less complexity in the choice of the version at the client side (more than 32 thousand combinations in the aforementioned case). The main variable of our problem is the boolean x_r, which indicates whether the server decides to offer the version $r \in \mathcal{R}$. Formally, we have:

$$x_r = \begin{cases} 1, & \text{if the server offers } r \in \mathcal{R} \\ 0, & \text{otherwise} \end{cases}$$

Since the server offers only J different versions, we have $\sum_{r \in \mathcal{R}} x_r = J$. In the following, we restrict our focus on the case of a given overall bit-rate budget B, which is a real number. The main idea is to offer several versions of the video meeting the same bandwidth requirement but with different quality distributions. All the versions have thus the same overall bit-rate "budget" but they differ by the quality of the video, which is better at some directions in the sphere than others.

To allow bandwidth adaptation in addition to viewport adaptation, the server shall solve the optimization problem for multiple bit-rate budget to offer the client choice between multiple bit-rate. In the following we will only consider one bit-rate value B for the sake of simplicity.

Client. The version selection algorithm first determines the most suitable bit-rate, here B, and then selects one and only one versions among the J offered versions for every segment of the videos, ideally the version that is the best match to user viewport. To simplify notations, we omit in the following the subscripts related to temporal segments, and we thus denote by $y_{u,r}$ the binary variable that indicates that user u selects $r \in \mathcal{R}$ for the video. Formally:

$$y_{u,r} = \begin{cases} 1, & \text{if the client } u \text{ selects } r \in \mathcal{R} \\ 0, & \text{otherwise} \end{cases}$$

Since the user selects only one offered versions, we have $\sum_{r \in \mathcal{R}} y_{u,r} \cdot x_r = 1$. We consider an ideal version selection algorithm and we thus assume that the client always selects the version that maximizes the viewport quality as previously defined, which is r such that $\sum_a v_{u,a} \cdot r(a)$ is maximum.

6.4.1 *Model Formulation*

Our objective is to determine, for a given set of users who request the video at bit-rate B, the J versions that should be prepared at the server side so that the quality of the viewports is maximum. In its most generic form, the problem can thus be formulated as follows.

$$\max_{y_{u,r}} \sum_{u} \sum_{r \in \mathcal{R}} y_{u,r} \cdot \sum_{a} v_{u,a} \cdot r(a)$$

Such that:

$$\sum_{a} b_{a,r(a)} = B \qquad \forall r \in \mathcal{R} \tag{6.1a}$$

$$\sum_{r} x_r \leqslant J \tag{6.1b}$$

$$\sum_{r} y_{u,r} = 1 \qquad \forall u \tag{6.1c}$$

$$y_{u,r} \leqslant x_r \qquad \forall r, u \tag{6.1d}$$

Note that with this formulation the problem is tractable.

6.5 PRACTICAL OPTIMIZATION MODEL

We take into account some practical additional constraints and some further hypothesis to formulate a tractable optimization problem, which meets key questions from content providers.

6.5.1 *Practical Hypothesis*

We first suppose that each area $a \in \mathcal{A}$ in the whole spherical video has the same coding complexity. This means we suppose that for a given quality, the byte-stream size of a area is proportional to its size. We derive the concept of *surface bit-rate*, which expresses in bit s^{-1} m^{-2} the amount of data that is required to encode an area at a given quality. We obtain that $b_{\max}$ (respectively $b_{\min}$) corresponds to the surface bit-rate for the maximum (respectively minimum) quality.

Second, we restrict our study to only two qualities per version. We follow in that spirit the MPEG OMAF testing conditions of viewport-dependent coding schemes [130] for the implementation of quality-variable 360-degree video versions. Let s_r be the overall surface of the areas that are in the QER for a given version $r \in \mathcal{R}$. The bit-rate constraints (6.1a) can thus be expressed as follow:

$$s_r \cdot b_{qer} + (4\pi - s_r) \cdot b_{out} = B \tag{6.2}$$

Third, we introduce a maximum gap between both qualities. The motivation is to prevent the video to have too visible quality changes between areas. This *quality gap ratio*, denoted by r_b, can be defined as the maximum ratio that relate the qualities b_{qer} and b_{out}:

$$\frac{b_{qer}}{b_{out}} < r_b \tag{6.3}$$

This hypothesis supposes that the quality differences are linearly proportional to the bit-rate differences.

Finally, we define the QER as a rectangular region defined on the sphere as shown in Figure 6.1. We thus adopt the restriction that has been introduced in the MPEG OMAF [59] to delimit a so-called *rectangular region on the sphere*. We also adopt the same way to define the region by delimiting two

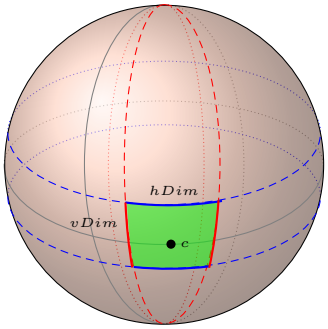


Figure 6.1: A rectangular region of the sphere: in blue the two small circle that delimit the region and in red the two great circles that delimit the region.

small circles (angular distance *vDim*), two great circles (angular distance *hDim*) and the spherical coordinates of the region center is $(1, \theta, \varphi)$.

In the following, we consider only video versions $r \in \mathcal{R}$ such that there exists $-\pi \leqslant \theta \leqslant \pi$, $0 \leqslant \varphi \leqslant \pi$, $-\pi \leqslant hDim \leqslant \pi$, and $0 \leqslant vDim \leqslant \pi$ such that for all area $a \in \mathcal{A}$, if a is inside the rectangle characterized by $(\theta, \varphi, hDim, vDim)$, the bit-rate of a is b_{qer} otherwise it is b_{out}. We denote such a version by $r_{\theta,\varphi,hDim,vDim}$.

6.5.2 *Bit-Rate Computation*

The objective function (6.1) imply that if two versions have a QER containing the same areas, the optimal set of offered video versions can only contains the version that maximize the b_{qer} subject to the bit-rate constraint (6.2) and the ratio constraint (6.3).

In order to simplify the complexity of the model, we pre-computed the value of b_{qer} and b_{out} depending on the size of the QER s_r. We identify four different cases depending on the size of the QER s_r. For simplicity, we provide in the following the main ideas of the algorithm and put the details of the mathematical model in the Appendix C.

We first combine the constraints given by the overall bit-rate budget with Equation (6.2) and the knowledge that $b_{min} \leqslant b_{out} < b_{qer} \leqslant b_{max}$. There are two cases, depending on whether the QER is small or not:

- When the surface of the QER is small, i.e., $s_r \leqslant \frac{B-4\pi b_{min}}{b_{max}-b_{min}}$ (see in Appendix) , the constraint on the maximum surface bit-rate prevails for b_{qer}. The surface bit-rate inside the QER can be maximum. The bit-rate budget that remains after deducing the bit-rate in the QER is $B - (s_r \cdot b_{\max})$. This remaining bit-rate budget is large enough to ensure that the surface bit-rate for the areas outside the QER is greater than $b_{\min}$. We obtain that b_{qer} is equal to $b_{\max}$ and b_{out} is derived as:

$$b_{out} = \frac{B - (b_{max} \cdot s_r)}{4\pi - s_r} \tag{6.4}$$

- When the surface of the QER is large, i.e., $s_r \geqslant \frac{B-4\pi b_{min}}{b_{max}-b_{min}}$, the constraint on the minimum surface bit-rate prevails. The surface bit-rate inside the QER cannot be $b_{\max}$, otherwise the remaining bit-rate that can be assigned to the video area outside the QER would not be large enough to ensure that b_{out} is greater than b_{min}. Here, we first have to set b_{out} to $b_{\min}$ and then assign the remaining budget $B - (b_{\min} \cdot (4\pi - s_r))$ to the QER area.

$$b_{qer} = \frac{B - (b_{min} \cdot (4\pi - s_r))}{s_r} \tag{6.5}$$

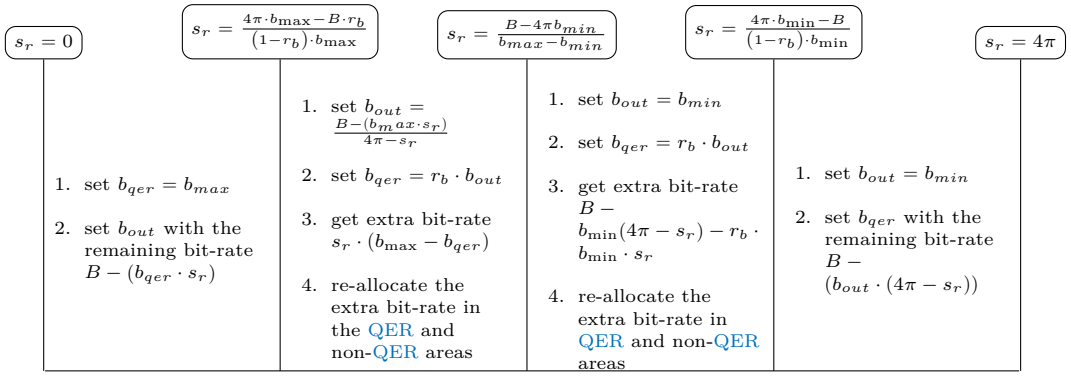


Figure 6.2: Algorithm for surface bit-rates in and out of the QER. The algorithm depends on the surface of the QER s_r. We show here the four different cases, for various surfaces (smallest to largest from left to right).

Next, we consider the quality gap ratio, which applies to both previously discussed cases:

- When the QER is small, setting $b_{qer} = b_{\max}$ and $b_{r,out}$ to (6.4) can lead to not respect Equation (6.3). It occurs for any QER such that (see in Appendix) :

$$s_r \geq \frac{4\pi \cdot b_{\max} - B \cdot r_b}{(1 - r_b) \cdot b_{\max}}$$

  The surface bit-rate b_{qer} should be instead reset as $b_{qer} = r_b \cdot b_{out}$. This constraint makes that some *extra* bit-rate are not assigned: $s_r \cdot (b_{max} - r_b \cdot b_{out})$. These extra bit-rates can thus be re-assigned to both b_{qer} and $b_{r,out}$ (see in Appendix) .

- When the QER is large, setting $b_{out} = b_{\min}$ and $b_{r,qer}$ with Equation (6.5) can also lead to not respect Equation (6.3). It occurs for any QER such that:

$$s_r \leqslant \frac{4\pi \cdot b_{\min} - B}{(1 - r_b) \cdot b_{\min}}$$

  Similarly as in the previous case, resetting b_{qer} with respect to the quality gap ratio leads to release of some extra bit-rates, which can be re-assigned to both b_{out} and b_{qer}.

We represent in Figure 6.2 the algorithm with the four cases when it applies to standard settings[1] of the overall bit-rate B, the maximum surface bit-rate $b_{\max}$, the minimum surface bit-rate $b_{\min}$, and the quality gap ratio r_b. Finally, we show in Figure 6.3 how the surface bit-rates are assigned depending on the surface s_r for a given parameter configuration (see in caption and in Section 6.6). Here the thin gray vertical lines correspond to the threshold at which the algorithm runs a different case.

6.6 EVALUATION – CASE STUDY

6.6.1 *Settings*

We used a custom-made C++ software publicly available on github.[2] This software uses the *IBM Cplex* library to solve our optimization problem.

1 In some configurations, it is possible that some of the presented cases do not hold since the threshold for the cases can be negative, greater than 4π, or interfering with a prevailing constraint. This however does not occur for the most common configuration parameters such that a quality gap ratio not too large and consistent values for both $b_{\min}$ and $b_{\max}$.

2 https://github.com/xmar/optimal-set-representation-viewport-adaptive-streaming

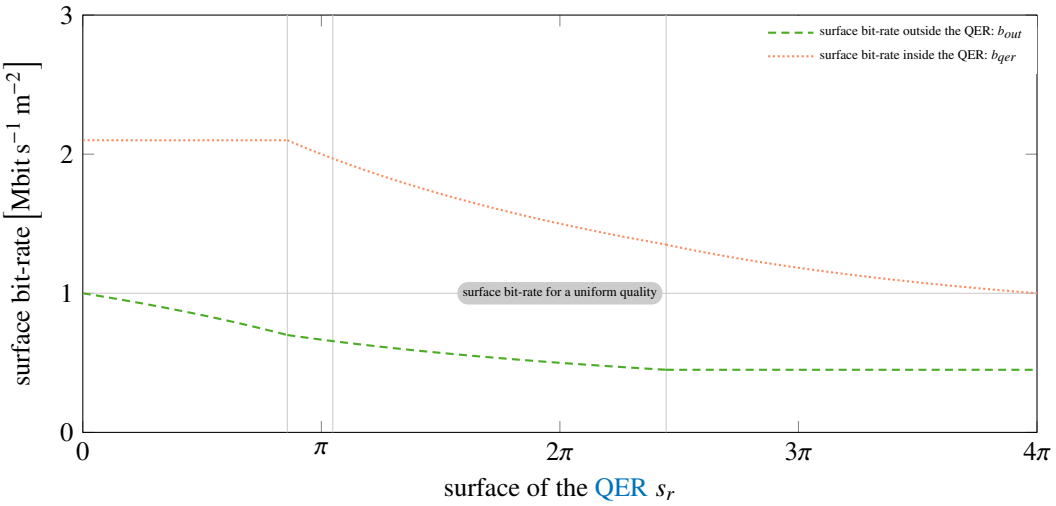


Figure 6.3: Surface bit-rates as a function of the QER surface. The overall video bit-rate B is 12.56 Mbit s^{-1}, so the surface bit-rate for a uniform quality is 1 Mbit s^{-1} m^{-2}. The maximum surface bit-rate $b_{\max}$ is 2.1 Mbit s^{-1} m^{-2} while the minimum $b_{\min}$ is 0.45 Mbit s^{-1} m^{-2}. Finally, the quality gap ratio r_b is 3.

Dataset of Head Movements. We used the public head movement dataset that we recently extracted and shared with the community [34].[3] This dataset contains the head orientation of 59 persons watching, with a HMD, five 70-second-long omnidirectional videos. In this paper we used the results from only two out of the five videos available: *roller-coaster* and *diving*. We selected those videos because users exhibit different behaviors while watching them: most users focus on a single RoI in the *roller-coaster* video while people move their heads to explore the scene in the *diving* video. The dataset is further presented in Chapter 8.

number of offered versions J	4
overall bit-rate B	12.56 Mbit s^{-1}
maximum surface bit-rate b_{max}	2.1 Mbit s^{-1} m^{-2}
minimum surface bit-rate b_{min}	0.45 Mbit s^{-1} m^{-2}
quality gap ratio r_b	3.5
number of areas N	400
video segment size	2 s

Table 6.1: Default evaluation settings

Content Provider Case Study. The default parameters are summarized in Table 6.1. The content provider generates up to $K = 4$ video versions and solves the optimization problem for every video segment (*i.e.,* each video segment has its own set of versions). The parameters related to the bit-rates are similar as in Figure 6.3: a total bit-rate budget B of 12.56 Mbit s^{-1}, a maximal surface bit-rate b_{max} of 2.1 Mbit s^{-1} m^{-2} and a minimal surface bit-rate b_{min} of 0.45 Mbit s^{-1} m^{-2}. We restricted the positions of the center of the QER on the sphere to 17 possible latitudes and 17 possible longitudes. Moreover the angular distance *hDim* and the angular distance *vDim* can take 12 different values. We split the sphere into a total of $N = 400$ areas. We cut the videos of the dataset into 2 s long segments. We solved the optimization model independently for each video segment, using the *IBM Cplex* solver[2].

3 http://dash.ipv6.enstb.fr/headMovements/

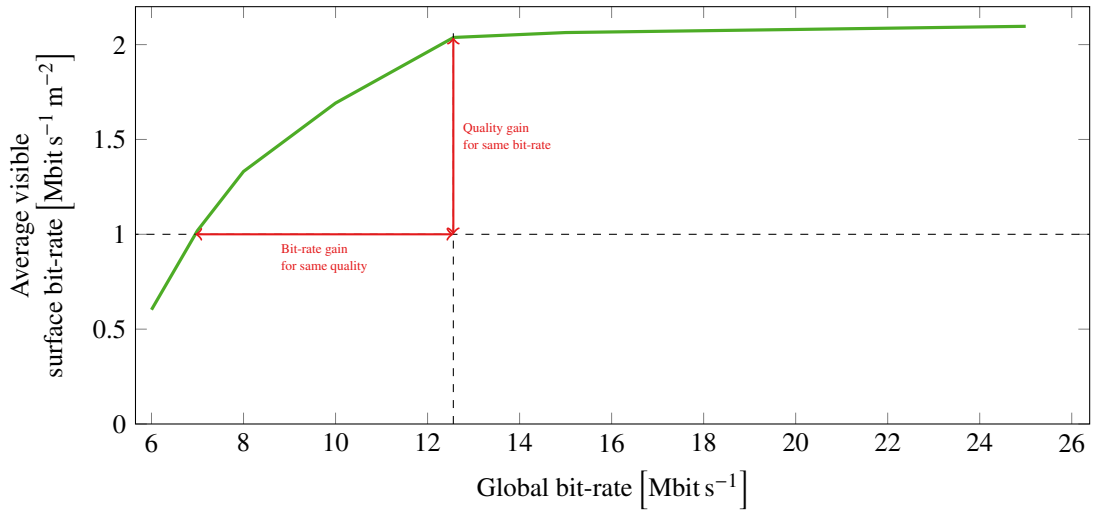


Figure 6.4: Visible surface bit-rate depending on the global bit-rate B. The horizontal red arrow shows the difference in total bit-rate to deliver viewports with the same average quality as a user would observe with a video encoded with a uniform quality. The vertical red arrow indicates the gain in quality (measure in surface bit-rate) compared to viewports extracted at the same position on a video with uniform quality with the same total bit-rate.

6.6.2 *Theoretical Gains of Viewport-Adaptive Streaming*

Our first goal is to evaluate the possible (theoretical) gains that the implementation of viewport-adaptive streaming can offer to the content providers. The gains can be evaluated from two perspectives: either the opportunity to save bandwidth while offering the video at the same level of quality as if the video was sent with uniform quality, or the opportunity to improve the quality of the video that is displayed at the client side for the same bit-rate as for a standard delivery. We computed the average surface bit-rate inside the viewport of the users (named *visible surface bit-rate* in the following) for different bit-rate budgets. The average visible surface bit-rate b_{vqer} in the viewport during a segment can be formally written as follow, with N_u the number of user:

$$b_{vqer} = \sum_{r,u} y_{u,r} \cdot \left(\frac{\sum_a v_{u,a} \cdot b_{r(a)} \cdot s_a}{N_u \cdot \sum_a v_{u,a} \cdot s_a} \right) \tag{6.6}$$

Figure 6.4 represents the mean average visible surface bit-rate for all segments of the two selected videos. The horizontal dashed line shows the average visible surface bit-rate for the bit-rate budget of 12.56 Mbit s^{-1} that is uniformly spread on the sphere, while the vertical dashed line indicates the quality for a constant bit-rate of 12.56 Mbit s^{-1}. We also represent the gains from the two aforementioned perspectives (either bit-rate savings or quality).

For a constant average quality inside the user viewports, the delivery of optimally generated QER versions enables 45% bandwidth savings. For a constant bit-rate budget, the optimal viewport-adaptive delivery enables an average increase of visible surface bit-rate of 102%.

6.6.3 *Video Content vs. Delivery Settings*

We now study the settings of the viewport-adaptive streaming systems, especially the parameters related to the number of different versions (J) and the segment size (T). We compare the set of versions that

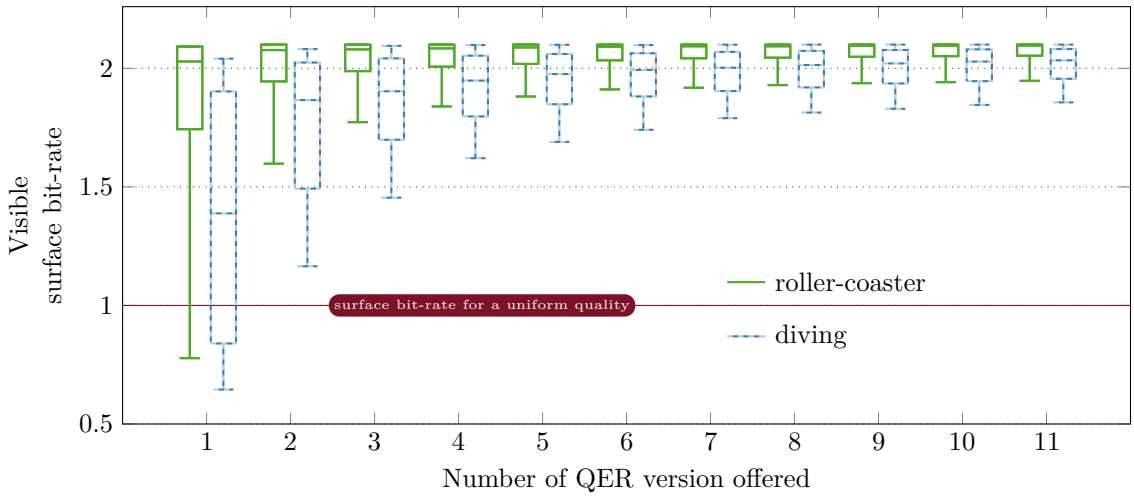


Figure 6.5: Visible surface bit-rate depending on the number of offered QER versions. The dark red line represents the visible surface bit-rate of a video encoded with the same overall bit-rate but with uniform quality.

are generated by the optimal solver for both selected videos. We are interested in studying whether there exists a common *best-practice* setting to generate versions, regardless of the video content, or whether each video should be prepared with respect to the content by a dedicated process with its own setting. We show the results computed separately for the *roller-coaster* and the *diving* video. Recall that the roller-coaster video has a single static RoI and most of the 59 users focus on it. On the contrary, the diving video has multiple moving RoI, which most users alternatively watch.

Figure 6.5 represents the average visible surface bit-rate b_{vqer} of the optimal QER versions for each user and each video segment for both videos: the *roller-coaster* video is in plain-green lines while the *diving* video is in dashed-blue lines. The results are shown with a box plot, with the 10^{th}, 25^{th}, 50^{th}, 75^{th} and 90^{th} percentiles for the 30 segments watched by the 59 users of each video in an optimal viewport-adaptive delivery system.

The viewport-adaptive streaming systems make that the higher the number of QER versions offered by the content provider, the better the average quality watched by the users because the set of versions covers more user behaviors. However, we notice that there exists a threshold value after which increasing the number of versions does not significantly improve the quality of the viewport of the users. This threshold depends on the video content. For the *roller-coaster* video, the limit is four QER versions while this limit is eight for the *diving* video. Please note that both threshold are significantly lower than the thirty versions that are generated by state-of-the-art viewport-adaptive delivery systems [73].

In Figure 6.6 we fix the number of QER versions to four and we evaluate the impact of the segment size on the generated QER versions. Like for Figure 6.5 the results are displayed with a box plot, which follows the same color code.

The median quality decreases while the size of the segments increases. Indeed, the higher the segments size, the wider are the head movements of the users. But, similarly as in the number of video versions, we notice that the median average displayed quality for the *diving* video is more sensitive to the segment size than for the *roller-coaster* video. For the latter, the quality decreases for segments longer than 2 s while for the *diving*, the quality decreases for segment longer than 1 s.

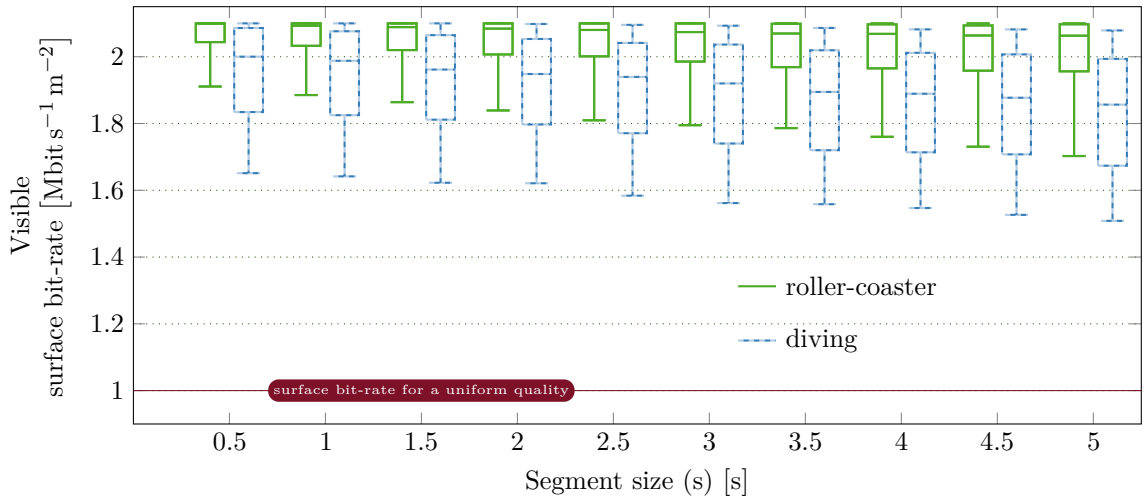


Figure 6.6: Visible surface bit-rate depending on the size of the segment. The dark red line represents the visible surface bit-rate of a video encoded with the same overall bit-rate but with uniform quality.

6.6.4 *QER Dimensions vs. Overall Bit-rate*

We study the main characteristics of the generated QER versions with a focus on the impact of the global bit-rate budget on the dimensions. We evaluate both the size of the QER inside each video version and the shape of the QERs.

Figure 6.7 represents the CDF of the surface of the QER inside each generated optimal version, for different global bit-rate budget, for both video. The dashed vertical black line represents the surface of the viewports of the users as it is seen in a HMD with a FoV of 110°×90°.

The size of the QERs increases with the overall bit-rate budget. If the bit-rate budget is small, the size of each QERs is smaller than the surface of the viewports. It means that no user has a viewport with full quality everywhere. The optimal solver prefers here to keep a high quality on an area that is common to the viewport of many users. If we increase the available bit-rate budget, the surface of the optimal QERs increases and is now wider than the viewport, so when a user who moves the head can nevertheless still have a viewport within the QER.

Figure 6.8 represents the probability density function (PDF) of the difference between the horizontal and vertical dimensions of the generated QERs. For instance, Figure 6.8a indicates that 21 % of the QERs have a horizontal size $hDim$ that is within the range $[-1 + vDim, -0.5 + vDim)$. The more occurrences of QER on the right, the more horizontal QERs are generated by the optimal solver.

QERs have often a squared shape (the horizontal dimension is close to the vertical dimension), and are mostly more horizontal than vertical. The horizontal shape can be explained by the fact that users move more often horizontally than vertically (they often stay close to the horizon). Moreover, when the bit-rate budget is limited, shapes are less often squared. Our interpretation is that, given that the generated QERs are narrower, the optimal solver generates QERs that cover various positions, corresponding to more users whose attention is on various positions around the horizon.

6.7 CONCLUSION

This Chapter investigates some theoretical models for the preparation of omnidirection video for viewport-adaptive streaming systems. Viewport-adaptive streaming has recently received a growing

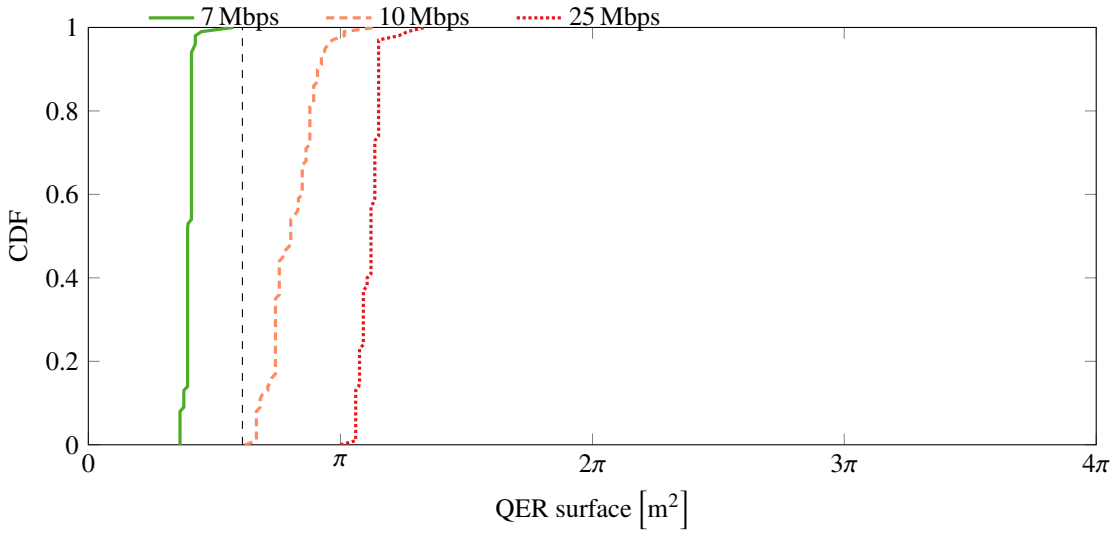
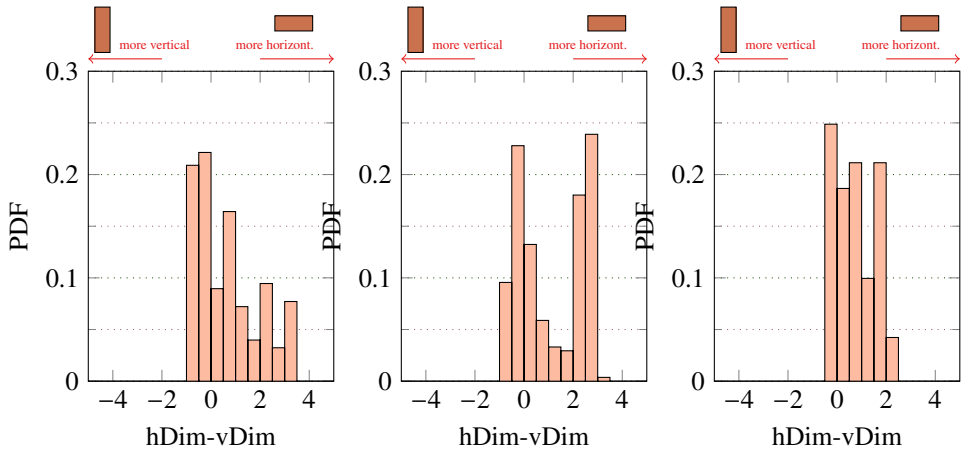


Figure 6.7: CDF of the surface of the QER of the offered version for different bit-rate budget.



(a) Bit-rate budget of 7.0 Mbits **(b)** Bit-rate budget of 10.0 Mbits **(c)** Bit-rate budget of 25.0 Mbits

Figure 6.8: Difference between the horizontal and vertical dimension of the QERs

attention from both academic [33, 94, 103] and industrial [7, 41, 129] communities. Despite some promising proposal, no previous work has explored the interplay between the parameters that characterize the video area in which the quality should be better. We denote this special video area a QER. In this Chapter, we address, on a simplified version of our theoretical model, the fundamental trade-off between spatial size of the QERs and the aggregate video bit-rate. We show that some new concepts, such as the surface bit-rate, can be introduced to let the content provider efficiently prepare the content to be delivered. Finally, we demonstrate the potential benefits of viewport-adaptive streaming: the gains compared to streaming of a video version with a uniform quality are greater than 102 % in terms of displayed quality to a user given a constant bit-rate budget, and a bit-rate budget reduction for more than 45 % for the same displayed video quality.

In this Chapter, we assumed that content provider already has some user head movement statistics. In future work we will study the generic QERs parameters that the provider can use to generate initial video versions of a omnidirection video, without video specific statistics. When the provider receives enough analytic, he will be able to generate versions adapted to real user behavior on each video segment. Such functionality would be required in both the processed and the live video viewport-adaptive streaming. Additionally, in this Chapter we studied only a simplified version of the theoretical model with only two different levels of quality per versions.

Part IV

PRACTICAL EVALUATION

7

OPEN SOFTWARE: 360TRANSFORMATIONS

7.1 INTRODUCTION

Omnidirectional videos are by essence spherical content, but in order to keep using existing technologies related to video compression and video streaming, omnidirectional videos are usually stored as planer video using sphere-to-plane projections (see Section 2.3). In 2015, when we started to work on this thesis, very few open software existed to manipulate projected omnidirectional videos.

In this Chapter we present *360Transformations*, an open-source software we developed to manipulate omnidirectional video: project, pack and unpack them. The software is openly available on Github [28]. *360Transformations* was design to read omnidirectional videos with any projection and packing formats, to project the videos from any format into any other format, to extract viewports at static location, to replay viewports extraction from head orientation datasets, and to compute some omnidirectional objective quality metrics. The software architecture allows easy addition of new projection and packing.

This software was used in most of our work to evaluate the proposed solutions. In Chapter 3, the software is used to generate the representations with a projection based QER, to extract viewports and to compute the objective quality metrics inside the viewports. In Chapter 4 the software is used to extract viewports from the original video and from the recomposed bit-stream generated after selecting tiles with different quality. Extracted viewports follow the recorded user's trajectory available in the MVP dataset. In Chapter 5, the software is used to generate the offset projection videos with different offset parameters (amplitudes and directions), to extract the viewports, to computes the S-PSNR and the Weighted Spherical - Peak Signal Noise to Ratio (WS-PSNR). Finally the software is compatible with the dataset presented in Chapter 8 and a part of the dataset evaluation is done using this software.

The rest of the Chapter is organized as follow. Section 7.2 discusses similar existing software, Section 7.3 presents an overview of *360Transformations*, describes its usage and its architecture, Section 7.4 shows an example of configuration file, Section 7.5 emphasized the scientific interests related this software, Section 7.6 explains how to compile the software and emphasize the different licenses and finally Section 7.7 concludes.

7.2 EXISTING SOFTWARE

The goal of *360Transformations* is to allow the community to easily manipulate omnidirectional content by changing the projection and the mapping of the omnidirectional signal on planar videos. The principle of sphere-to-plane mapping was introduced in Section 2.3.3. In this Section we will discuss the different existing software with a similar purpose as *360Transformations*.

The Joint Video Exploration Team (JVET)[1] and MPEG[2] have developed a reference software named *360Lib* [65] to project and pack omnidirectional videos, and to perform omnidirectional objective quality evaluation. The first version of *360Lib* was released in December 2016. Ye, Alshina, and Boyce [149] provide in a technical report a deep description of the software and of the different projections and quality metrics available. This software integrates the latest version of the HEVC reference software named *HM*. This software is mono-threaded and can only manipulate raw YUV videos or HEVC

1 ITU-T VCEG (Q6/16)
2 JTC 1/SC 29/WG 11

binary stream. It support nine projections including the equirectangular, the cube-map and the viewport projections. Our software perform multi-threaded computation when possible and is based on *ffmpeg* library to support all formats and codecs supported by ffmpeg.

Other applications exist but do not have the same functionality or the same purpose as *360Transformations* or are not openly available. *Facebook Transform360* is a *ffmpeg* filter that can transform a omnidirectional video from an equirectangular projection into a cube-map projection with flexible configurations. At the time this Chapter was written, only the equirectangular to cube-map transformation is supported. *Google Spatial Media* is a software used to inject and/or read metadata for spherical video and audio into a MPEG-4 Part 14 (MP4) container. It manipulates only the metadata. with a FoV of 110°×90°

7.3 SOFTWARE OVERVIEW

360Transformations is a C++ application, developed from scratch to take advantages of latest features from the C++11 and C++14 standards [64].

360Transformations can read omnidirectional videos encoded with any codec supported by *ffmpeg*. The video bitstream or the raw video should be encapsulated into a MP4 or a Matroska (MKV) container so that enough metadata is available for ffmpeg to decode the video stream. Multiple videos can be processed at the same time. Frames with the same id in each video are processed synchronously. The frame format of the input videos can be any of the implemented projections and packing (such as 4×3 cube-map or equirectangular).

The software takes for input an *INI* configuration file. This configuration file indicates the path to each input video and describes their packed format: the sphere-to-plane pixel mapping used. The *INI* can define multiple sphere-to-plane mapping and sphere-to-sphere transformations. Each sphere-to-plane mappings, such as equirectangular or cube-map projections, are characterized by multiple parameters:

PROJECTION TYPE A type indicating which projection to used. The type should correspond to one the of projection implemented in the software

PLANAR PICTURE RESOLUTION the spatial resolution of the planar picture

LOCAL REFERENCE FRAME ORIENTATION the orientation of the local reference frame used to perform the projection. The orientation is measured compared to the world reference frame which is supposed immutable. The orientation is the rotation that transform the world reference frame into the local reference frame.

PROJECTION SPECIFIC PARAMETERS some projection type specific parameters used to modify the behaviours of some projection. For instance, with the cube-map projection it is possible to indicate the position and orientation of each face of the cube on the planar picture.

A SPHERE-TO-SPHERE TRANSFORMATIONS an optional transformation, such as the offset transformation studied in Chapter 5, applied on the sphere before the projection to modify the behavior of the transformation. It is characterized by a transformation type, a local reference frame orientation and some transformation specific parameters.

Then the *INI* file defines a *transformation flow* for each input video. A transformation flow is a set of consecutive sphere-to-plane transformation applied, frame by frame, on a video.

Each frame of the input videos goes through its transformation flow, as specified in the configuration file. The frames outputted from a transformation flow can be displayed on the screen and/or encoded into a new videos. The format of the output video have to be a format supported by *ffmpeg*. One should notes that *ffmpeg* support single image output or input, so do *360Transformations*. It is not possible

to generate output videos with HEVC's MCTS yet because *libx265* [146], the open-source HEVC encoder used by default by *ffmpeg* does not implement tile encoding. To generate MCTS a user would have to pipe the output video to an encoder that supports MCTS, such as *Kvazaar* [69].

Omnidirectional Objective Quality Metrics. The software implements some omnidirectional full-reference objective quality metrics. The metrics are computed frame by frame and use the output of the first transformation flow as reference. Five metrics are currently implemented:

PSNR, SSIM AND MS-SSIM The PSNR , Structural Similarity (SSIM) and MS-SSIM are the traditional objective quality assessment metric used with planar videos (see Section 2.4.2). They do not take into account any properties related to projected omnidirectional videos. They require the two compared videos to have the same resolution and to use the same projection with the same parameters.

S-PSNR The Spherical - Peak Signal Noise to Ratio (S-PSNR) projects the two compared video on the sphere and get color samples on the sphere at 655 362 predefined positions. The color samples are interpolated when they do not match a projected pixel center. Usually the interpolation used is a bilinear interpolation performed on the original projected picture, but other interpolation can be used such as the nearest neighbor, the bi-cubic, or the lanczos interpolations. Then a classic PSNR computation is performed on the luminance value of those 655 362 colors for the two pictures. The advantages of the S-PSNR is that it does not requires the two compared video to use the same projection nor the two compared video to have the same resolution. This metrics was used in Chapter 5.

WS-PSNR The WS-PSNR computes a regular PSNR between two pictures but weight the errors in the MSE computation using the size on the sphere of the projection of each pixel of the planar picture. Thereby, a pixel that take up a big portion of the sphere has more influence on the WS-PSNR value than a smaller pixel on the sphere. This metric, like the traditional PSNR requires the two compared pictures to use the same projection and to have the same resolution.

7.3.1 *Software Architecture*

The code of *360Transformations* can be separated in five parts.

The main function. This function initialize all the sphere-to-plane projections and the sphere-to-sphere transformation used in the *INI* configuration file, and run the main loop that read pictures one by one in all input videos and apply the transformation flow on them.

Configuration Parser. This function reads a section of the *INI* configuration file and transform it into an instance of a concrete class that defines a sphere-to-plane projections or a sphere-to-sphere transformation.

Layouts. Inside the *360Transformations* a *Layout* is the name used to define a sphere-to-plane projections. A virtual class named *Layout* is the based class to any sphere-to-plane projections. This virtual class defines the basic interface to manipulate any projections. For instance, the main function manipulate only *Layout* objects with the need to understand which actual projection is used. This structure allow ease addition to new projections inside the software. The interface define two publics functions: *From2dTo3d* and *From3dTo2d*. *From2dTo3d* transforms the coordinates (u, v) of a pixel in the planar picture into a viewing direction $\vec{\mathbf{a}}$ indicating the position of this pixel on the sphere. *From3dTo2d* does the inverse transformation: it takes a viewing direction $\vec{\mathbf{a}}$ and return the coordinate (u, v) of the corresponding pixel on the planar picture. Here the pixel coordinates may be fractional and an interpolation should be used to compute the color of the direction pointed out by $\vec{\mathbf{a}}$. Other functions are defined to initialized the layout, to update dynamic layout (projection with properties that change with time) or to get the surface of a specific pixel on the unit sphere.

VectorialTrans. As the *Layout* class, *VectorialTrans* is a virtual class that defines a unique interface to manipulate sphere-to-sphere transformations. Any sphere-to-sphere transformations should inherit from *VectorialTrans*. The *VectorialTrans* defines two functions: one to perform the sphere-to-sphere transformation and one to perform the inverse transformation.

LibAvWrapper. This is the part of the code used to interface with *ffmpeg* library.

In the version of *360Transformations* available at the time this dissertation is written, seven types of sphere-to-plane projections are implemented. Figure 7.1 displays an example of each projection available. Those formats are: (*i*) The standard Equirectangular projection. (*ii*) The Equirectangular projection splits into tiles. (*iii*) The Cube-Map projection with a compact packing or with the classic 4×3 packing. (*iv*) A square based pyramid projection with a compact or not compact packing. (*v*) A Rhombicdodeca projection with a compact packing. Finally (*vi*), viewport can be extracted using the FlatFixed or the gnomonic projection.

Two sphere-to-sphere transformations are currently implemented. The offset transformation, introduced in Chapter 5, which generalizes the offset cube-map projection used by *Facebook* for the Oculus HMD as described by Zhou, Li, and Liu [162] and Zhou et al. [163]. The second transformation implemented is the horizontal plane offset transformation. This transformation is very similar to the offset transformation except that it preserves horizontal lines. It was also introduced by *Facebook* to be more "encoder friendly".

For each layout and transformation, it is possible to apply a rotation on the spherical signal. This means that the center of the projection can be moved to any direction. We denote by center of the projection, the direction of $\vec{\imath}$ (as defined in Section 2.2), which has for Cartesian coordinate $(1, 0, 0)$ before applying the rotation. For instance for the Cube-Map projection, the front face of the cube can be rotated to any direction.

Viewports can be extracted using the FlatFixed projection or the gnomonic projection. For each, two options are available. (*i*) extract the viewports at a static location: for each frame of the video the viewport is extracted at the same position. (*ii*) replay a recorded head-movement trajectory. The trajectory should have the same format as trajectories in the dataset of Corbillon, Simone, and Simon [34], presented in Chapter 8. The relative timestamps of the frame is used to select which position of the trajectory to use to extract the viewport.

7.3.2 *Software Behaviors*

360Transformations always starts by reading the INI configuration file. It reads the "LayoutFlow" field, initializes a video decoder for each video listed, and initializes a *Layout* for each packed format needed by the "LayoutFlow".

To transform a frame from one projection to another, the software generates an empty picture with the same resolution as required by the output projection. Then, as illustrated in Figure 7.2, it iterates in parallel over all pixels of the output picture, asks the output *Layout* for the coordinates of the viewing direction vector corresponding to the selected pixel, then asks the input *Layout* for the coordinates of the input pixel corresponding to this viewing direction vector, and then interpolates the color of the output pixel. This operation is done consecutively for all projection present in transformation flow of an input videos.

7.4 CONFIGURATION EXAMPLES

Equirectangular

4 × 3 cubemap

Equirectangular with 8 × 8 tiles packed with different resolutions

Compact cubemap

Pyramid

Compact rhombicdodecahedron

Compact pyramid

FlatFixed: viewport

Figure 7.1: Illustration of the currently available projection and frame packing: without rotations, without offset projections, and, except for the Equirectangular with tiles paking, with the same resolution on each faces

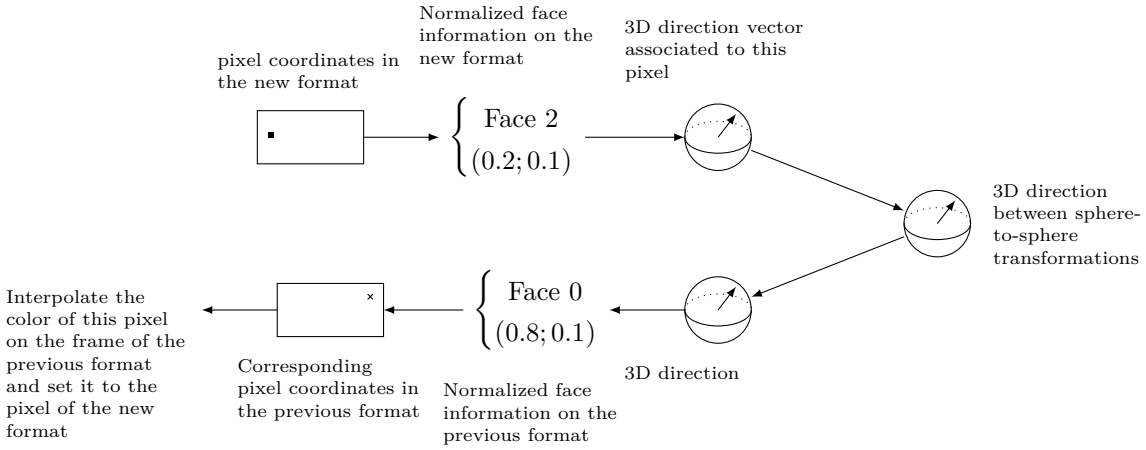


Figure 7.2: Projection transformation steps. Those steps are executed, for each projection transformation in a transformation flow, for each pixel in the new video frame.

```
[Global]
fps=24
layoutFlow= [["./input1.mp4", "Equirectangular1"],
             ["./input1.mp4", "Equirectangular1", "Cubemap1"]]
displayFinalPict=true
videoOutputName= outputVideo.mp4
videoOutputBitRate=0
qualityOutputName = quality.txt
qualityToComputeList = [S-PSNR-I]
nbFrames= 5
startFrame= 0

[Equirectangular1]
type=equirectangular
upscale=false
refWidth=3840
refHeight=2048
rotation= {"type":"euler", "yaw":0.0, "pitch":0.0, "roll":0.0}
relativeResolution=true
width=1
height=1
bitrate=1
offsetRatio=0

[CubeMap1]
type=cubeMap2
refWidth=3840
refHeight=2048
relativeResolution=true
rotation= {"type":"angleAxis", "angle":90, "x":0.0, "y":0, "z":1}
```

Listing 2: Example of Configuration File

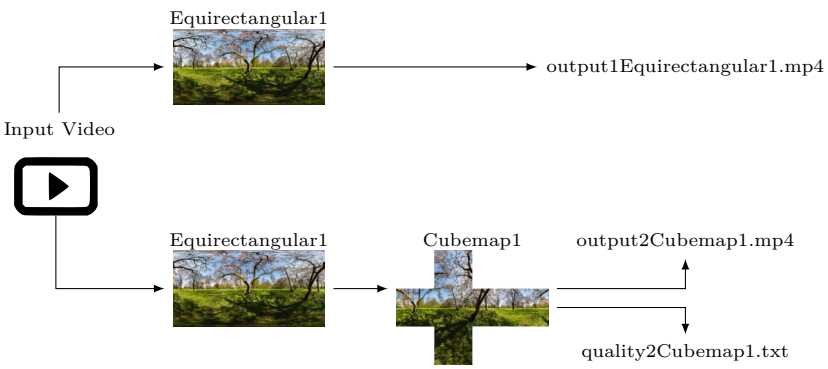


Figure 7.3: Processing flow described in the configuration file

In this Section we illustrate how *360Transformations* can be used to manipulate projected omnidirectional video.

Listing 2 presents a simple configuration file that takes one Equirectangular omnidirectional video, projects it into a Cube-Map and packs it into a 4×3 Cubemap layout. Then, it computes the S-PSNR objective metric to compare this Cube-Map projection with the original Equirectangular projection. It also re-encode the output frames into new HEVC videos. Figure 7.3 illustrates this transformation flow. Here the computed quality metric is stored into a ".txt" file, and the output videos are stored into ".mp4" files.

The *Global.layoutFlow* key contains a list of transformation flow. In each transformation flow we first indicate the path to the input video and then the name of the section that describe the projection used by this input video (here *Equirectangular1*). We stop there for the first flow because we want to keep the reference frame with original projection of the video. For the second flow, we add *Cubemap1* after the *Equirectangular1* format in order to transform the frame into a 4×3 Cube-Map projection.

In the *Equirectangular1* section and the *Cubemap1* section, we describe the two different packed frame formats used. The *Equirectangular1* section indicates the resolution of the Equirectangular picture and indicates that no rotation was performed. In the *Cubemap1* section, we indicate the resolution of the output cube-map frame, indicate that we want all faces to use their standard resolution (no upscaling nor downscaling) and then indicate that we want to rotate the cube 90° around the vertical axis (axis $\vec{\mathbf{k}}$). This rotation means that the front face of the cube will face the original "right" direction.

7.5 SCIENTIFIC INTEREST

We designed and implemented *360Transformation* to manipulate omnidirectional videos. At the time we started to work on this software, only *Facebook ffmpeg* filter to extract flat-fixed viewports or to generate cube-map from equirectangular and equirectangular from cube-map was available. The use of this filter was limited as only the three projections implemented by *Facebook* was available and extension with now projections was quite challenging. Moreover, interfacing this filter with scripts to allow reproducible scientific experiment was hard.

360Transformation was designed to allow the community to easily add new projections and new spherical transformations. This is possible by taking advantage of polymorphism to manipulate only abstract base classes inside the body of the software. A user who want to add a new projection or a new sphere-to-sphere transformation only need to implement a new class that inherit from the right base class, and then needs to describe what fields from the INI configuration file should be used to initialized the new projection/transformation.

360Transformation can be used only by manipulating the INI configuration file. If no new projection or transformation are needed, there is no need to manipulate the C++code. Then it is easy to use for instance *Python3* to generates configuration files with different set of parameters, running the software to generate videos with new projection and/or to extract viewports, and to compute objective quality metrics. The quality metrics output can then be parsed, and statistics can be computed. To reproduce the results, only the configuration file (our the script used to generate the configuration files), the input videos are needed.

In order to fairly compare different viewport-adaptive streaming strategies (server content preparation strategies or client segment download strategies) it is needed to be able to replay head orientation dataset with different input omnidirectional videos (different versions with QERs, different bit-streams generated with tiles selected with different qualities) to extract viewports in the same positions in all tested scenario. *360Transformation* is able to read head orientation datasets and to extract viewports at the same positions as the user did.

7.6 INSTALLATION AND LICENSE

360Transformations is an open source software, release under the MIT License, and provided as a tool for researchers. *360Transformations* uses three external libraries: ffmpeg libav (LGPL v2.1) to decode and encode the videos, OpenCV (3-clause BSD) to manipulate the frames, and Boost (Boost license) to parse the input configuration file.

To install *360Transformations*, you need a C++11 ready compiler (for instance *gcc* version 5 or greater). You also need *cmake* in order to generate the *Makefile* that will be used to compile the software. Compilation instruction are available on the Github homepage [28].

A *Docker*[3] container named *xmar/trans360* and available on *Docker Hub*[4] allows the community to use the software directly without the need to compile it. The container can also be used to test new projections/transformations without the need to install the dependency of *360Transformations* first.

We did all our testing on Archlinux OS and on Debian OS.

7.7 CONCLUSION

In this Chapter, we described the *360Transformations* software made to manipulate omnidirectional videos. This software is able to read omnidirectional videos encoded with standard codecs, projected and packed in any implemented formats. It is able to sequentially change the projection and the packing of the frames of multiple input videos. It supports sphere-to-sphere transformations, such as the offset transformation introduced by Facebook, to apply modification on any projection. For each input video, the final projected and packed frame can be encoded into a new output video, and/or an objective quality metric can be computed. *360Transformations* was designed to allow easy addition of new projection and new frame packing and is able to replay recorded head orientation datasets. Next Chapter describes the head orientation format accepted as input of *360Transformations*, and will describe the head-orientation dataset we recorded.

Future work on this software may focus on adding support of stereoscopic omnidirectional videos, and adding new projections and packing formats such as the barrel projection or the truncated pyramid projection. Adding the possibility to perform pixel color interpolation on the sphere and not only on the planar picture as it is today. Indeed, in current version of the software, to interpolate the color of the pixel in the projected picture (using nearest neighbor, bi-linear, or the bicubic interpolation) we project the pixel on the original planar picture and perform the interpolation on this planar picture. But pixels near on the planar picture are not mandatory near (or at least not with the same ratio of distance) on the sphere. The interpolation might introduce less noise if performed directly with neighboring pixels on the sphere instead of neighboring pixels on the original planar picture.

3 https://www.docker.com/
4 https://hub.docker.com/r/xmar/trans360/

8

A DATASET OF USERS' HEAD MOVEMENTS IN OMNIDIRECTIONAL VIDEOS

8.1 INTRODUCTION

The adoption of viewport-adaptive streaming raises many open questions regarding the navigation patterns of users. For instance: how fast do people move their heads when wearing an HMD? Do different people focus on the same parts of the sphere? Does the user's behaviour depend on the type of video? Does this behaviour depend on the user's characteristics? In order to answer these questions, the availability of data collected while users are watching omnidirectional videos via HMDs is critical.

For this reason, in this Chapter, we present an omnidirectional video head movement dataset gathered by recording the navigation patterns of 59 users watching five 70 s long omnidirectional videos. The dataset is available on our website [31].[1] In order to help the community to reproduce our results and to upgrade this dataset, we release the open source software developed for our data collection campaign and allow new contributors to share their datasets on our website [31].

The rest of the Chapter is organized as follow. Section 8.2 reviews the existing works that have reported an analysis of the navigation patterns of viewers consuming omnidirectional content via HMDs. Section 8.3 describes the choices made to setup our data collection, the software and the experimental conditions. Section 8.4 details the structure of our dataset so that it can be reused by the community. Section 8.5 portrays a first analysis of the collected data to illustrate the kind of information that can be extracted from the dataset. Finally, Section 8.6 concludes this Chapter.

8.2 RELATED WORK

Yu, Lakshman, and Girod [152] use the average viewport-based head motion trajectory of ten users viewing ten omnidirectional videos to compute a weighted version of the PSNR. The authors observed that the users tend to look around the equator more than the poles. The average viewing probability map is reported in the paper, but the navigation patterns per user and content are not publicly available. Also, the viewing conditions and test material used to collect the data are not disclosed.

Upenik, Rerabek, and Ebrahimi [137] describe a test-bed to perform controlled quality assessment experiments on omnidirectional images via HMDs. The software allows to capture, among other data, the viewing direction of the user at a chosen sampling rate. Examples of average viewing probability maps obtained during a quality assessment subjective test are reported for three test images. The method used to process the information on user's head movements and derive saliency maps is described by Upenik and Ebrahimi [136].

De Abreu, Ozcinar, and Smolic [40] describe a study of navigation patterns collected during static omnidirectional image viewing on a HMD. The test-bed and collected dataset are publicly available.

Since the work presented in this Chapter was done, multiple head-orientation datasets of subjects watching omnidirectional video was publicly released by the community. Wu et al. [145] performed two subjective experiments: a task-free scenario and a scenario where user were asked to pay attention to the scene. In both experiments, 48 young subjects (with sex parity) watched 9 omnidirectional

1 http://dash.ipv6.enstb.fr/headMovements/

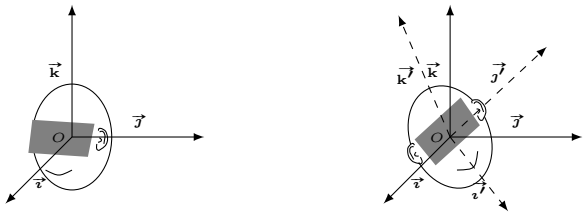


Figure 8.1: Illustration of the choice for the stationary reference frame $(O, \vec{\imath}, \vec{\jmath}, \vec{\mathrm{k}})$ and for the rotating reference frame $(O, \vec{\imath'}, \vec{\jmath'}, \vec{\mathrm{k}'})$ linked to the user head.

videos selected from YouTube (different set of videos for each experiment). Subjects used a *HTC Vive* HMD. They recorded the HMD orientation (using unit quaternion representation) and the HMD spatial position (storing the coordinates of its centroid). They observe that in the task-free scenario users head trajectories are more diverse than in the scenario with a task. Our dataset record only the head orientation and represents a task-free scenario. Song et al. [121] recorded a dataset of head movement of 58 users watching 76 omnidirectional videos. The videos were a few second long (10 s to 80 s). The dataset is, at the date this dissertation is written, not publicly available anymore. Lo et al. [82] recorded 50 users watching 10 one minute long omnidirectional videos extracted from *YouTube*. The users wear a *Oculus Rift DK2 HMD*. They store the users' head orientation using Euler angles representation (yaw, pitch, and roll). Fremerey et al. [46] presents a dataset of 48 users watching 20 omnidirectional videos, 30 s long, on an *HTC Vive* HMD in a task-free scenario. The content was 4K video downloaded from *YouTube* and *ARTE*. A simulator sickness questionnaire was filled after each viewing session. The questionnaire answers show a significant difference in spatial disorientation between female and male subjects showing that female users are more sensitive to spatial disorientation or answer to the questionnaire more honestly. David et al. [39] generated the first dataset that records not only the head orientation by also the gazing position of 57 users watching 19 omnidirectional videos downloaded from *YouTube* during a free-task experiment. Previews datasets recording gazing position in omnidirectional content exists only for static omnidirectional images [105].

8.3 EXPERIMENTAL SETTINGS

In this Section we describe the notation chosen to describe user's head movements, the software implemented to capture these movements during omnidirectional video consumption on a HMD, as well as the test material and conditions considered during the viewing sessions.

8.3.1 *Head Orientations*

Since current delivery platforms and HMD technologies are restricted to 3DoF, we captured rotational head movements and ignored the translational movements of users. To measure the head position we chose the following conventions, illustrated in Figure 8.1:

- We consider the same Euclidean space and direct orthonormal basis $(O, \vec{\imath}, \vec{\jmath}, \vec{\mathrm{k}})$ as introduced in Section 2.2.
- We denote by $(O, \vec{\imath'}, \vec{\jmath'}, \vec{\mathrm{k}'})$ the direct orthonormal basis linked to the user's head position. The $\vec{\imath'}$ axis goes through the center of the HMD, the $\vec{\jmath'}$ axis goes through the viewer's left ear and the $\vec{\mathrm{k}'}$ axis goes through the top of the head. This basis is linked to the viewport reference frame

$(O_p, \vec{\mathbf{u}}, \vec{\mathbf{v}})$, introduced in Section 2.3.1, by the following properties: $\vec{\mathbf{u}}$ (respectively $\vec{\mathbf{v}}$) is collinear to $\vec{\boldsymbol{\jmath}'}$ (respectively $\vec{\mathbf{k}'}$) with the same direction.

- The reference head position (i.e. the position without any rotation) is the position where $(O, \vec{\boldsymbol{\imath}}, \vec{\boldsymbol{\jmath}}, \vec{\mathbf{k}})$ and $(O, \vec{\boldsymbol{\imath}'}, \vec{\boldsymbol{\jmath}'}, \vec{\mathbf{k}'})$ coincide. The reference position (i.e. the $(O, \vec{\boldsymbol{\imath}}, \vec{\boldsymbol{\jmath}}, \vec{\mathbf{k}})$ basis) is set at boot time by the HMD.

Note that in practice, the world reference frame is fixed by the HMD after each reboot. The HMD always choose $\vec{\mathbf{k}}$ vertical but $\vec{\boldsymbol{\imath}}$ and $\vec{\boldsymbol{\jmath}}$ can change each time the HMD restarts (but are always horizontal). Between two reboots the world reference frame (and so the reference viewport) never change.

Using the software described at Section 8.3.2, we captured any variation of the head position during a viewing session, with respect to the reference position. This is described by the rotation $\mathscr{R}$ that transforms $(O, \vec{\boldsymbol{\imath}}, \vec{\boldsymbol{\jmath}}, \vec{\mathbf{k}})$ into $(O, \vec{\boldsymbol{\imath}'}, \vec{\boldsymbol{\jmath}'}, \vec{\mathbf{k}'})$. There are many ways to characterize a rotation in $\mathbb{R}^3$ [23]: we use the unit Hamiltons quaternions representation. According to Euler's rotation theorem, any rotation or sequence of rotations of a three-dimensional coordinate system with fixed origin is equivalent to a single rotation around an axis, represented by a unit vector $\vec{\mathbf{v}} = (x, y, z) = x\vec{\boldsymbol{\imath}} + y\vec{\boldsymbol{\jmath}} + z\vec{\mathbf{k}}$ in $\mathbb{R}^3$, and by a given angle θ, using the right hand rule. This axis-angle representation of $\mathscr{R}$ can be expressed by four scalars defining the unit quaternion [23]:

$$q = (q_0, q_1, q_2, q_3) = (q_0, q_1\,\vec{\boldsymbol{\imath}} + q_2\,\vec{\boldsymbol{\jmath}} + q_3\,\vec{\mathbf{k}}) = (cos(\theta/2), sin(\theta/2)\,\vec{\mathbf{v}})$$

.

We chose the quaternion representation because (*i*) it has the advantage of being a compact representation (four scalars instead of the nine required by the 3×3 matrix representation), (*ii*) quaternion are not subject to the *gimbal lock* [139], which is a well-known issue of the Euler angles representation, and (*iii*) quaternion representation of rotations is less sensitive than matrix representation to rounding errors occurring when scalars are represented at floating point precision. Note that if q is a unit quaternion representing the rotation $\mathscr{R}$, then $-q$ represents the same rotation $\mathscr{R}$.

8.3.2 *Software*

We developed a omnidirectional video player to capture and save to a log file the user's head orientation at each frame or whenever a head movement (*i.e.* a motion event) occurs during the visualization of a video on an HMD. The main purpose of the software is to accurately associate each motion event to a timestamp corresponding to a video timestamp at frame level, thus, to the id of the video frame displayed by the HMD when the motion event occurred.

Please note that we share the software in a public repository [31] with the MIT open source license [86]. Therefore, it can now be used by the scientific community, potentially to extend the dataset with more viewers, more videos, and different viewing conditions.

The software has been implemented in C++, based on the Open-Source Virtual Reality (OSVR) Application Programming Interface (API) [14], the Open Graphics Library (OpenGL) [95], and the *ffmpeg* library [80]. We performed all tests and the data collection campaign on Linux OS with the Razer OSVR Hacker Development Kit 2 (HDK2) HMD [104] but the software is expected to be compatible with any HMD and Windows OS.[2] Figure 8.2 illustrates a high-level diagram of the input and output interfaces of our software.

The experimenter can set some input parameters by using a configuration file. This file specifies: (*i*) the sphere to plane projection used to produce the planar video file ; (*ii*) the time offset in second starting from which the video file is displayed; (*iii*) the number of video frames, i.e. the duration of the

2 The OSVR API is available on Linux and Windows OS and is agnostic to the HMD used.

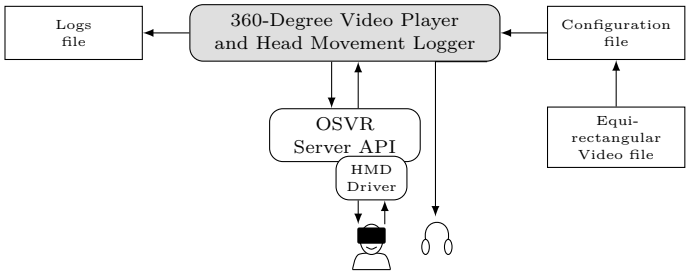


Figure 8.2: High level diagram of components and interfaces of our OSVR Video Player and head movement logger. The gray block is the components we developed.

YouTube Id	Name	Content Description & Expected Focus of Attention	Spatial Resolution	Frame Rate	Bit Rate	Start Offset
2bpICIClAIg	Elephants	Elephants along a river side. Fixed camera, main content along the equator line.	3840 × 2048 pixels	30 fps	16 522 kbitps	15 s
		One main azimuthal focus expected along the equator line.				
7IWp875pCxQ	Rhinos	Rhinos in the nature. Fixed camera, main content along the equator line.	3840 × 2048 pixels	30 fps	13 462 kbitps	15 s
		Focus expected along the equator line.				
2OzlksZBTiA	Diving	Diving scene. Slowly moving camera, no clear horizon.	3840 × 2048 pixels	30 fps	19 604 kbitps	40 s
		No main focus expected within the sphere.				
8lsB-P8nGSM	Rollercoaster	Rollercoaster. Fast moving camera fixed in front of a moving roller-coaster.	3840 × 2048 pixels	30 fps	16 075 kbitps	65 s
		Strong main focus following the rollercoaster trail.				
CIw8R8thnm8	Timelapse	Timelapse of city streets. Fixed camera, clear horizon with a lot of fast moving	3840 × 2048 pixels	30 fps	15 581 kbitps	0 s
		people/cars, many scene cuts. Focus expected along the equator line.				
s-AJRFQuAtE	Venice	Virtual aerial reconstruction of Venice. Slowly moving camera.	3840 × 2048 pixels	25 fps	16 101 kbitps	0 s
		No main focus expected within the sphere.				
sJxiPiAaB4k	Paris	Guided tour of Paris. Static camera with some smooth scene cuts.	3840 × 2048 pixels	60 fps	14 268 kbitps	0 s
		Focus expected along the equator line.				

Table 8.1: Description of the *YouTube* omnidirectional videos used. The entries with grey background in the table identify the videos used for training.

segment, to be displayed. The time offset and duration are specified because the input omnidirectional video can be a video of several minutes of duration: the configuration file allows to specify what portion of the video to display to the user, so that the navigation patterns are measured over a fixed limited duration.

The player communicates with the OSVR Server API to get the last known position of the user's head. With this information, the player extracts from the input video the viewports to be displayed (one for each eye) and sends them to the HMD using the OSVR API. In parallel, the audio signal is sent to a headphone. Each time a new head position is measured, the quaternion that represents the rotation of the head with respect to the reference head position is stored in a log file alongside the current timestamp and the picture id of the displayed video frame.

8.3.3 *Test Material*

The navigation patterns in the dataset have been collected on five omnidirectional *YouTube* videos, described in Table 8.1. We limited the test material to few videos to be able to perform a viewing

Number of users	Minimum age	Average age	Maximum age	Ratio of women	Ratio of users using a HMD for the first time
59	6	34.15	62	20%	61%

Table 8.2: Statistics on the users who took part to the experiment

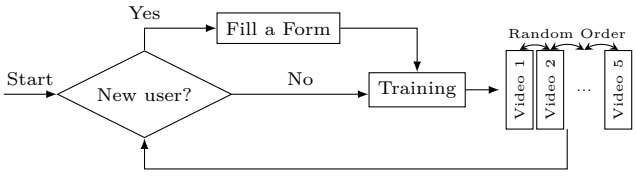


Figure 8.3: Flow for a new user evaluation

session of reasonable duration and collect data from many users. Therefore, the test videos have been chosen to span a wide range of omnidirectional content, including tourism, thrill, and discovery, for which different viewer's involvement, thus navigation patterns, could be expected. Each video file has been downloaded in equirectangular format, at the maximum resolution and bit-rate available on *YouTube* (reported in Table 8.1), The videos do not have the same frame rate but this does not impact the data collection since the head position is captured for each motion event, rather than at a fixed rate. A 70 s-long portion of each video (starting at the offset indicated in Table 8.1) has been selected and used in the viewing session. Two additional videos (with grey background in Table 8.1) have been used as training material to familiarise the users with the viewing set-up.

8.3.4 *Viewing Session*

All participants to our study performed one viewing session of a total duration of seven minutes. Before the beginning of the session, oral instructions were provided to describe the main steps of the viewing session and explain that the user's navigation patterns were going to be recorded. Each user was informed about the presence of a training session, to familiarize with the omnidirectional viewing experience and adjust the HMD calibration, if needed, followed by a viewing session of seven minutes, consisting of the sequential display of videos, separated by a grey screen, displayed for 5 s to 10 s between two consecutive videos.

The exact flow (Figure 8.3), followed by every user, is described hereafter:

1. Before the user puts the HMD on, he/she is asked to fill in a questionnaire, displayed on a computer via a Graphical User Interface (GUI), concerning the user's gender, age, vision impairments if any, and the level of familiarity with HMDs.
2. The training session takes place, during which a 70 s training video, randomly selected for each user by the GUI, is displayed. Viewers were orally instructed to adjust the HMD vision correction settings, if needed, by using the wrench adjuster under the HMD and familiarise with the omnidirectional viewing experience by moving their heads.
3. All five test videos are consecutively displayed in a random order.
4. At the end of the viewing session, the user is asked to remove the HMD.

We invited people to stand during the entire viewing session but some asked to sit for some videos (often for the *Rollercoaster*). When seated, people sat on a rolling chair and were still able to turn easily in any direction. An operator always stood next to the user to hold cables out of user's range and guarantee free movements.

8.3.5 *User Sample*

At the time this paper was written, 59 users took part to our data collection. Most people in the sample group are students or staffs from the IMT Atlantique school in France. Some are children from staff members and some are employees from IMT Atlantique's startups. Table 8.2 shows some statistics

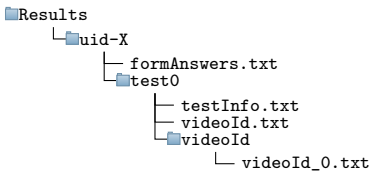


Figure 8.4: Folder structure

about this sample group. Users are aged from 6 to 62 with an average age equal to 34 years. 80 % of the sample is composed by men and 61 % of the sample was using a HMD for the first time. Half of the users who had already used a HMD, did it for less than 12 minutes.

8.4 DATASET STRUCTURE

The dataset was created and structured to allow other research teams to use it and add new traces, while protecting the privacy of the users. The folder structure, data format and meaning of the results collected by using the software described at Section 8.3 are detailed hereafter.

8.4.1 *Result Folder Structure*

Figure 8.4 pictures the structure of the result folder. When a new user participates to the data capture, he receives a unique identifier [77] that guarantees there will be no naming collision. The data collected for each user is stored in a dedicated folder, named "uid-X", with *X* being the user's identifier. This folder contains:

- one file named "formAnswers.txt", containing the answers of the user to the questionnaire;
- one folder per viewing session , named "testN", with *N* being the viewing sessions id associated to the viewing session, generated by the GUI , useful to structure the results when the same user participates to multiple viewing sessions. This folder contains: (*i*) a file named "testInfo.txt", reporting on each line a video id followed by the MD5 sum [107] of the video file. These identify the videos displayed to the user during the corresponding viewing session test, according to the order of presentation, the first video being the training video; for each video id in the file "testInfo.txt", (*ii*) a corresponding file named "videoid.txt", reporting the configuration file of the C++ OSVR video player and head movement logger used to displayed this video to the user (see Section 8.3.2 for more details about the software), and (*iii*) a folder named "videoid" containing the user's head movements for this specific video, according to the format detailed in the next subsection.

8.4.2 *Head Position Log Structure*

Each log file of the user's head position has the same structure. There is one sample per line. Values are separated by spaces, according to the following format :

```
timestamp frameId q0 q1 q2 q3
```

The first value, at floating-point precision, is the timestamp in second relative to timestamp 0: timestamp 0 is the time when the video player started to display the first frame and is the first timestamp in the file. The second value, an integer, is the picture order counts (POC) of the video frame displayed at time equal the timestamp. Here, POC zero corresponds to the first picture displayed after the player

seeks to the start offset of the video. The next four values, at floating-point precision, are the q_0, q_1, q_2 and q_3 values of the unit quaternion q used to identify the head position of the user (*cf.* Section 8.3.1): $q = (q_0, q_1 \vec{\boldsymbol{\imath}} + q_2 \vec{\boldsymbol{\jmath}} + q_3 \vec{\mathbf{k}})$.

In the log files, head position samples are recorded each time the video player renders a new picture for the HMD. This means the sampling rate is not constant and may vary within each test. Note that rendering refresh rate can be higher than the video frame rate.

8.5 A TYPICAL USAGE OF THE DATASET

In this Section we illustrate a possible use of the dataset by focusing on the viewport adaptive streaming scenario presented in Chapter 3. This analysis does not aim to be exhaustive of all possible usages of the dataset. The evaluation in presented in Chapter 6 is an other example on how the dataset can be used.

8.5.1 *Pre-Processing*

The population of users from which we collected the dataset is close to the typical audience expected for viewport adaptive streaming, therefore there is no need to filter the users.

In Section 8.4.2, we mentioned that the head position sampling rate is not constant. This means that some sample are very close in time and some are farther away. In order to compute head movement statistics, we resampled the collected data, i.e. the quaternions, using a sampling frequency of 30 Hz. We chose 30 Hz because most of the videos used to generate the dataset have a frame rate of 30 fps. There are multiple ways to interpolate quaternions based on existing samples: we chose to use the spherical linear interpolation (SLERP) [89], using the two samples that are the closest in time to the timestamp we want to extract. The SLERP formula assumes that the user moves on the shortest great-circle arc between the two measured positions, with a constant velocity.

8.5.2 *Head movements*

With adaptive streaming, it is often not possible to switch from video representations once a specific representation started to be displayed to the user, so it is important for the video segment size to be short enough to allow frequent representation switching. On the other hand, if the video segment is too short, video codecs are less efficient and the service provider needs to store and transmit more metadata to describe the segments.

To estimate the maximum duration a video segment should have, we compute "how static the user is". Particularly, for each user, we compute the angular distance between the center of the viewport at the beginning of a segment and the center of each viewport attended by the user during the duration of a segment. Figure 8.5 shows the CDF of the maximum angular distance traveled during a fixed duration inside each segment by each user and for each video. We used segments of length 1 s, 2 s, 3 s, and 5 s. It can be noticed that, for instance, within a segment duration of 2 s, 95% of the users move less that $\pi/2$ radians. This means that within a 2 s length segment, 95% of the users stay inside the hemisphere centered on the head position of the user at the beginning of the segment. Therefore, we conclude that 2 s is probably a good compromise for the duration of the segment.

Figure 8.6 shows the CDF of the maximum angular distance per video for a segment length of 2 s. The data shows that, in this dataset, there are three categories of videos: a video triggering very few head movements (Roller-Coaster), a video triggering many head movements (Timelapse), and

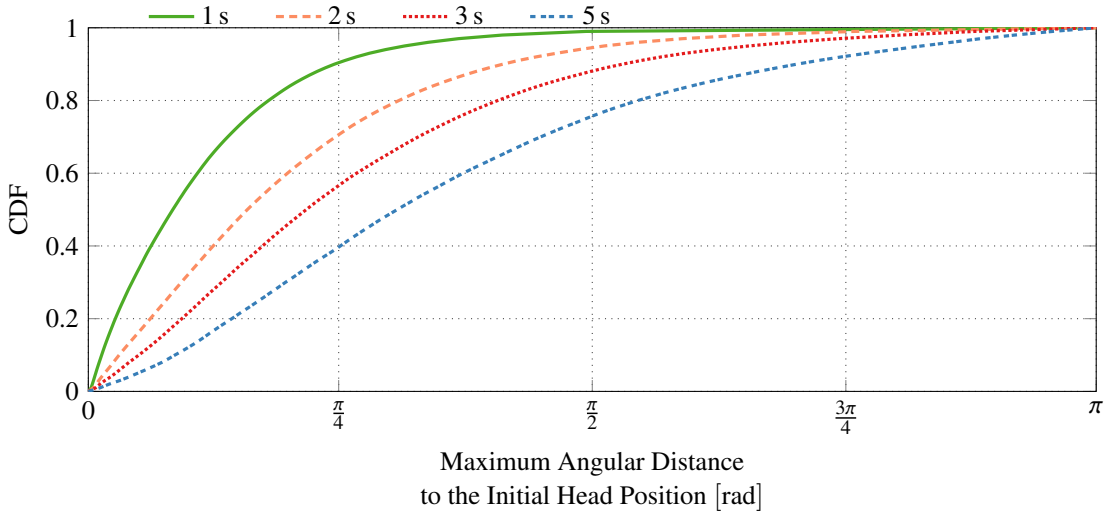


Figure 8.5: CDF of the maximum angular distance from the head position at the start of the segment for different segment lengths.

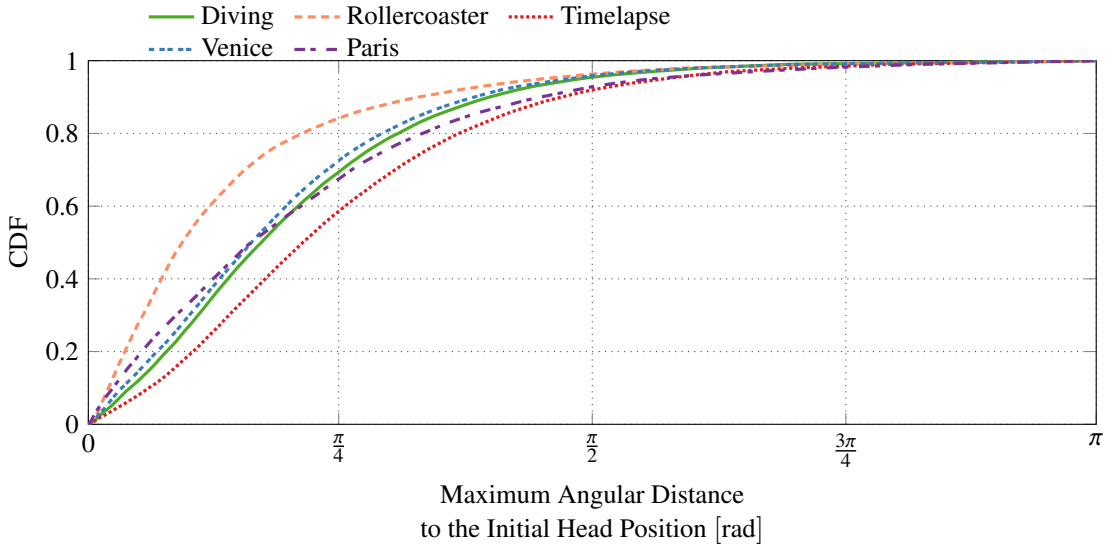
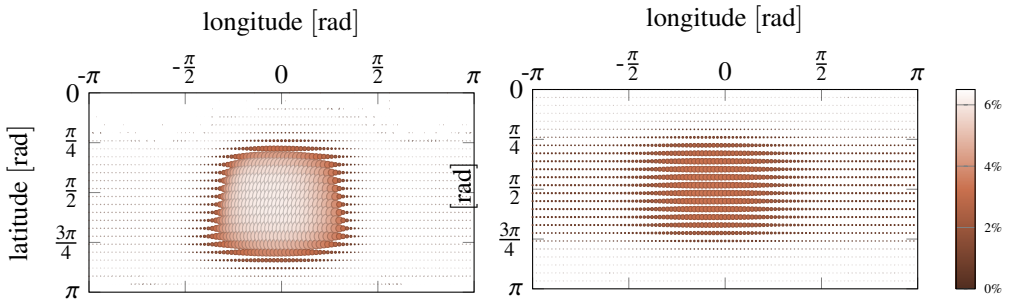


Figure 8.6: CDF of the maximum angular distance from the head position at the start of the segment, per video, for a 2 s long segment.



(a) Roller-Coaster **(b)** Timelapse

Figure 8.7: Probability for a pixel to be be attended by a user during a whole video

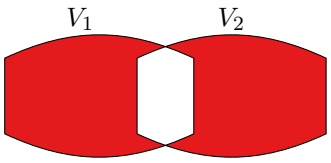


Figure 8.8: In red the area of the symmetric difference between viewport V_1 and viewport V_2 represented in the equirectangular domain.

intermediary videos (Venice and Diving). The video Paris is an hybrid, triggering very few head movements at the beginning and many more towards the end.

8.5.3 *Viewing probability*

Statistics on where users looked at in a specific omnidirectional video can be used to (*i*) re-encode representations with QERs, adapted to a majority of users, or (*ii*) send information to the client to help its representation selection decision.

Figure 8.7 shows the probability for a pixel in the equirectangular domain to be inside the viewport of the user, aggregated for the whole video duration. Figure 8.7a shows the statistics for the Roller-Coaster video and Figure 8.7b for the Timelapse video. We observe that for the Roller-coaster video there is a very well defined RoI at the center of the equirectangular picture. This is in the direction of the rails. For the Timelapse video, there is no prominent viewing direction, but most viewports stay near the horizon.

To have a better understanding of the time variation of the concentration of the viewports in the videos and similarity across users navigation patterns, we compute for each video frame the area of the symmetric difference between all possible couple of viewports attended by each user during the navigation. Figure 8.8 depicts the symmetric difference of two viewports. The area of the symmetric difference is a pseudo-distance that is equal to zero when the two viewports are identical and is equal to two time the area of a viewport when the intersection of the two viewports is empty. Figure 8.9 represents the median distance between all couple of viewports attended by each user inside the same frame during video segments of 2 s. We observe that for the Roller-coaster video, the users focus their gaze in the same direction after 10 s (5 segments of 2 s). For the Timelapse, Venice, and Diving videos, user viewports are spread across multiple directions. Regarding the Paris video, at the beginning of the video, most people look in the same direction (at the tourist guide) and then most users look in different directions.

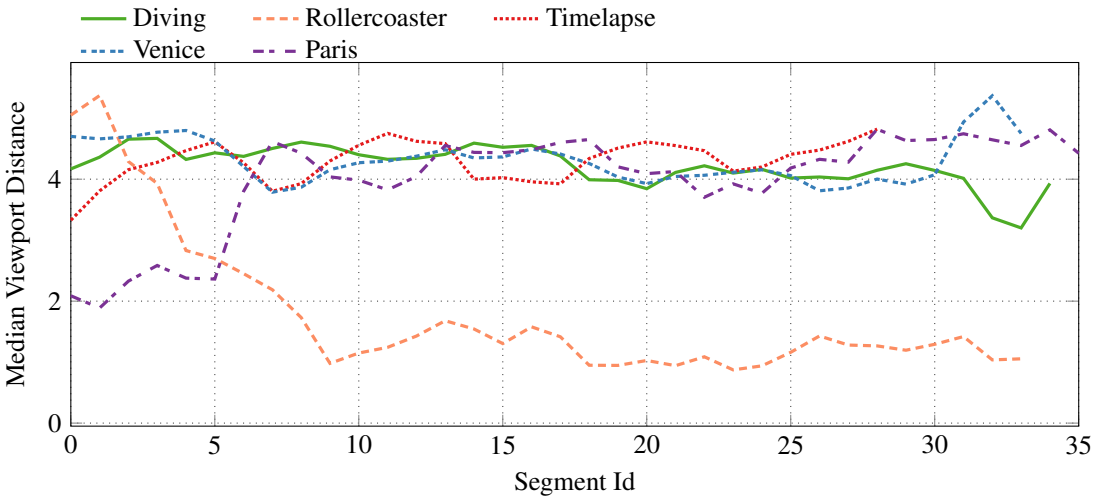


Figure 8.9: Vision distance for a 2 s long segment on a the Roller-coaster video

8.6 CONCLUSION

In this Chapter we presented a dataset including the head positions of 59 users recorded while they were watching five 70 s-long omnidirectional videos using the Razer OSVR HDK2 HMD [104]. The dataset is available on our website [31] alongside the used videos and the open-source software that we developed to collect the dataset. We described our settings, the test material and how we performed the data collection. We also detailed the structure of the dataset. Finally we introduced examples of statistics that can be extracted from the dataset to provide an overview of the users' behaviour and the videos characteristics, focusing on the viewport adaptive streaming scenario.

We expect that this dataset will help researchers to study and understand omnidirectional video consumption. The prediction of navigation patterns is a cornerstone of the new generation viewport-adaptive streaming systems for omnidirectional content. The dataset will hopefully enable researchers to test new prediction algorithms.

Part V

CONCLUSION AND FUTURE WORK

9

CONCLUSION & FUTURE WORK

9.1 DISSERTATION CONCLUSION

In this dissertation we presented the work accomplished since 2015 on the adaptive streaming of omnidirectional video. We split this work into six main contributions, each presented in its own Chapter.

Chapter 3 presents a viewport-adaptive streaming architecture. This architecture is compatible with Dynamic Adaptive Streaming over HTTP (DASH), the main deployed adaptive streaming protocol, with Content Delivery Networks (CDNs) and with content provider architectures. We propose to generate, at the server side, video representation characterised by a predefined Quality Emphasized Region (QER). We evaluated this architecture with video versions prepared with a QER following the shape of the face of Three Dimensional (3D) geometrical objects used in the projection process. We showed, based on a small head movement dataset, that (*i*) the cube-map projection is the most efficient projection compared to the equirectangular, the square base pyramid and the dodecahedron projections, based on the the encoding technique and parameters used in this experiment, (*ii*) it is not needed to generate more than six representation per omnidirectional video, and (*iii*) two-seconds is a good compromise for the segments duration.

Chapter 4 proposes an extension of the viewport-adaptive streaming architecture to stream Multi-ViewPoint (MVP) omnidirectional videos. MVP omnidirectional videos offer the user the possibility to perform a finite number of predefined translational movement inside the scene in addition to the rotational movement available in omnidirectional videos. We evaluated some optimal download client strategies based on the proposed architecture. We introduce the key trade-off the client has to consider when scheduling the downloading of such videos to maximize the user experience. We emphasize the improvement of service performance introduced by the tiling and show that proactive strategies in this scenario introduces a too expensive cost on the Quality of Experience (QoE). With the current state of the research a reactive download strategy seems the best option.

In Chapter 5 we study the offset projection introduced by Facebook, generalize it to any projection with the offset transformation, and propose to use the spherical pixel density to estimate the distortion inside viewports extracted from those videos. We showed that the spherical pixel density metric, in the context of offset transformation, is highly correlated to the distortion introduced inside the viewports. Our evaluation points out that tilling is globally more efficient than offset transformation regarding the distortion but the offset transformation allows the use of planar projected video with lower resolution which is compatible with most of the available decoding hardware. It is possible to generate bit-streams with lower resolution using tilling as demonstrated by Sreedhar et al. [123], but we did not evaluate this scenario.

In Chapter 6 we propose a model to generate version with heterogeneous spatial quality. The model uses viewing statistics (measured or predicted) to allocate a fixed bit-rate budget inside the omnidirectional video in order to maximize the overall users satisfaction using a viewport-adaptive streaming architecture as described in Chapter 3. We demonstrate the potential gains of viewport-adaptive streaming compared to the streaming of omnidirectional video with uniform surface bit-rate in term of increase of the bit-rate inside the generated viewports for constant bandwidth usage (+102 %) or in term of decrease of bandwidth usage for constant bit-rate inside the viewports (-45 %).

Chapter 7 describes the open-source software we developed to manipulate projected omnidirectional videos. This open-source software was used to evaluate most of our proposed solution.

Finally Chapter 8 presents the head-movement dataset we collected in order to better understand users behaviors inside omnidirectional content and to better evaluate viewport-adaptive streaming. The dataset contains 59 users watching five 70 s-long videos in free-task scenario. Analysis of this dataset confirms that segment duration of two seconds is a good trade-off between encoding efficiency and user head movements velocity in the context of representations with a QER. We emphasize that users head movements velocity vary depending on the video content type and that group of users can have similar behaviors. This dataset is now part of Moving Picture Experts Group (MPEG) official test for conformance datasets under the name "IMT Atlantique dataset".

9.2 PERSPECTIVE AND FUTURE WORKS

Omnidirectional video streaming has become, since 2015, a trending subject with a growing involvement of the multimedia community. MPEG experts have already release a first version of the Omnidirectional MediA Format (OMAF) tool kit and are actively working on the second version. They are pushing, through MPEG – Immersive (MPEG-I), toward full six Degrees of Freedom (6DoF) content, opening a new era of convergence between videos and video-games.

Even with the growing interest of the community and of standardization groups, many questions related to the streaming of omnidirectional content are still open.

How subjective evaluation should be performed with omnidirectional content? The experimental conditions to evaluate the user subjective QoE when watching a traditional planar content are strictly defined by international standards, but those experimental conditions do not consider user watching content through a Head-Mounted Display (HMD). Even if the experimental conditions are defined properly, it is still very hard to aggregate the QoE measured by different users. Indeed, the inherent freedom offered by omnidirectional videos makes it highly unlikely that two users watch exactly the same content (i. e. that for all given display timestamps, the same viewport is displayed to each users). Comparison of Mean Opinion Scores (MOSs) in this condition is not trivial.

We have already seen in the related work Chapter that assessing the QoE with real users is a challenging, time consuming, and not a cost efficient task. Objective quality metrics have been developed for traditional video to automatically assess the QoE of users. They are not always highly correlated with the MOS but are still very useful for automatic evaluation. It is still not clear how quality of omnidirectional video should be objectively assess. Some objective metrics such as the Spherical - Peak Signal Noise to Ratio (S-PSNR), and the Weighted Spherical - Peak Signal Noise to Ratio (WS-PSNR) compute objective metrics on the whole projected omnidirectional video but are not adapted to representations with heterogeneous quality. Others extract viewports and compute classic objective metrics inside. Some methodologies have been proposed to evaluate the quality of omnidirectional video with heterogeneous quality [7] but they are usually not compared to subjective QoE and no methodology exist to predict users subjective QoE from those metrics.

Viewport-adaptive streaming requires accurate medium-term user's head orientation prediction to allow clients to prefetch video segments which offer high quality in the viewports the user will attend. Wrong prediction results in lower quality viewports and too short prediction period makes the system more sensitive to bandwidth variations. Studies of omnidirectional saliency maps and omnidirectional sound correlation with head movements may improve the accuracy and the predictions.

New compression and storage of omnidirectional data could be investigated to improve the performance of viewport-adaptive streaming. Indeed, new omnidirectional encoding techniques should investigate how fast representation switching can be done without introducing high bit-rate or high

computation overhead. The goal is to offer the lowest motion-to-high-quality delay to the users with the lowest computational cost and the highest possible compression efficiency.

Overall, we start to observe a paradigm change in the multimedia world by slowly moving from sender-centric to *receiver-centric* architectures. Previous architectures were designed to enable servers to stream, in a cost efficient way, the same content to a great number of users. The switch from broadcast content to adaptive streaming was already a first step toward the receiver, as its bandwidth and characteristics are now considered in the streaming process. With immersive content and higher degrees of freedom, users receive more and more unique data as two users are unlikely to attend exactly the same parts of the content. On the one hand the content size is drastically increasing, and on the other hand the part of the content attended by the user is decreasing. For low latency interactivity it is becoming critical to offer smaller elementary retrievable and decodable data units. With a full receiver-centric architecture, those elementary data units could be distributed in a cloud fashion, and the client could retrieve just the required data units, compose a decodable bit-stream and extract the viewport to display to the user.

There are still plenty of open questions and existing scientific problems to be solved to provide future users highly immersive experiences at the minimum cost.

REFERENCES

[1] ITU-T Recommendation H.264 ISO/IEC 14496-10. *Advanced Video Coding for Generic Audiovisual Services*. Standard. Geneva, CH: International Organization for Standardization, May 2003.

[2] ITU-T Recommendation H.265 ISO/IEC 23008-2. *High efficiency video coding*. Standard. Geneva, CH: International Organization for Standardization, Apr. 2013.

[3] Hamed Ahmadi, Omar Eltobgy, and Mohamed Hefeeda. "Adaptive Multicast Streaming of Virtual Reality Content to Mobile Users." In: *Proceedings of the on Thematic Workshops of ACM Multimedia 2017 - Thematic Workshops '17*. 2017, pp. 170–178. DOI: 10.1145/3126686.3126743.

[4] Akamai via Statista. *Average global internet connection speed from 1st quarter 2011 to 1st quarter 2017*. 2017. URL: https://www.statista.com/statistics/204954/average-internet-connection-speed-worldwide/ (visited on 05/04/2018).

[5] Saamer Akhshabi, Ali C Begen, and Constantine Dovrolis. "An experimental evaluation of rate-adaptation algorithms in adaptive streaming over HTTP." In: *Proceedings of the second annual ACM conference on Multimedia systems*. ACM. 2011, pp. 157–168. DOI: 10.1145/1943552.1943574.

[6] Patrice Rondao Alface, Jean-François Macq, and Nico Verzijp. "Interactive omnidirectional video delivery: A bandwidth-effective approach." In: *Bell Labs Technical Journal* 16.4 (2012), pp. 135–147. DOI: 10.1002/bltj.20538.

[7] Alireza Aminlou, KKashyap Kammachi Sreedhar, Alireza Zare, and Miska Hannuksela. *Testing methodology for viewport-dependent encoding and streaming*. MPEG meeting. m39081. Oct. 2016.

[8] Jérôme Ardouin, Anatole Lécuyer, Maud Marchal, and Eric Marchand. "Navigating in virtual environments with 360 omnidirectional rendering." In: *3D User Interfaces (3DUI), 2013 IEEE Symposium on*. IEEE. 2013, pp. 95–98. DOI: 10.1109/3DUI.2013.6550203.

[9] Jérôme Ardouin, Anatole Lécuyer, Maud Marchal, and Eric Marchand. "Stereoscopic rendering of virtual environments with wide Field-of-Views up to 360°." In: *Virtual Reality (VR), 2014 iEEE*. IEEE. 2014, pp. 3–8. DOI: 10.1109/VR.2014.6802042.

[10] Yanan Bao, Huasen Wu, Tianxiao Zhang, Albara Ah Ramli, and Xin Liu. "Shooting a moving target: Motion-prediction-based transmission for 360-degree videos." In: *2016 IEEE International Conference on Big Data (Big Data)*. IEEE, Dec. 2016, pp. 1161–1170. DOI: 10.1109/BigData.2016.7840720.

[11] Yanan Bao, Tianxiao Zhang, Amit Pande, Huasen Wu, and Xin Liu. "Motion-Prediction-Based Multicast for 360-Degree Video Transmissions." In: *2017 14th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, June 2017, pp. 1–9. DOI: 10.1109/SAHCN.2017.7964928.

[12] Mariem Ben Yahia, Yannick Le Louedec, Gwendal Simon, Loutfi Nuaymi, and Xavier Corbillon. "HTTP/2-Based Frame Discarding for Low-Latency Adaptive Video Streaming." In: *IEEE Transactions on Multimedia Computing Communications and Applications* (2018). submitted.

[13] Giuseppe Boccignone, Angelo Marcelli, Paolo Napoletano, Gianluca Di Fiore, Giovanni Iacovoni, and Salvatore Morsa. "Bayesian integration of face and low-level cues for foveated video coding." In: *IEEE Transactions on Circuits and Systems for Video Technology* 18.12 (2008), pp. 1727–1740. DOI: 10.1109/TCSVT.2008.2005798.

[14] Yuval S Boger, Ryan A Pavlik, and Russell M Taylor. "OSVR: An open-source virtual reality platform for both industry and academia." In: *Proceeding of IEEE Virtual Reality (VR)*. 2015, pp. 383–384. DOI: 10.1109/VR.2015.7223456.

[15] Ali Borji and Laurent Itti. "State-of-the-art in visual attention modeling." In: *IEEE Transactions on pattern analysis and machine intelligence* 35.1 (2013), pp. 185–207. DOI: 10.1109/TPAMI.2012.89.

[16] Jill M. Boyce, Yan Ye, Jianle Chen, and Adarsh K. Ramasubramonian. "Overview of SHVC: Scalable Extensions of the High Efficiency Video Coding Standard." In: *IEEE Transactions on Circuits and Systems for Video Technology* 26.1 (2016), pp. 20–34. DOI: 10.1109/TCSVT.2015.2461951.

[17] Matthew Brown and David G Lowe. "Automatic panoramic image stitching using invariant features." In: *International journal of computer vision* 74.1 (2007), pp. 59–73. DOI: 10.1007/s11263-006-0002-3.

[18] Niklas Carlsson, Derek Eager, Vengatanathan Krishnamoorthi, and Tatiana Polishchuk. "Optimized Adaptive Streaming of Multi-video Stream Bundles." In: *IEEE Transactions on Multimedia* 19.7 (July 2017), pp. 1637–1653. DOI: 10.1109/TMM.2017.2673412.

[19] Andrea Cavallaro, Olivier Steiger, and Touradj Ebrahimi. "Semantic video analysis for adaptive content delivery and automatic description." In: *IEEE Transactions on Circuits and Systems for Video Technology* 15.10 (2005), pp. 1200–1209. DOI: 10.1109/TCSVT.2005.854240.

[20] Jacob Chakareski, Ridvan Aksu, Xavier Corbillon, and Gwendal Simon. "Viewport-Driven Rate-Distortion Optimized 360-degree Video Streaming." In: *Proceedings of IEEE International Conference on Communications (ICC)*. 2018, pp. 1–7. DOI: 10.1109/ICC.2018.8422859.

[21] Yue Chen, Debargha Murherjee, Jingning Han, Adrian Grange, Yaowu Xu, Zoe Liu, Sarah Parker, Cheng Chen, Hui Su, Urvang Joshi, et al. "An Overview of Core Coding Tools in the AV1 Video Codec." In: *In proceeding of Picture Coding Symposium (PCS)*. 2018, pp. 24–27.

[22] Shyamprasad Chikkerur, Vijay Sundaram, Martin Reisslein, and Lina J. Karam. "Objective Video Quality Assessment Methods: A Classification, Review, and Performance Comparison." In: *IEEE Transactions on Broadcasting* 57.2 (June 2011), pp. 165–182. DOI: 10.1109/TBC.2011.2104671.

[23] Su Bang Choe and Julian J Faraway. *Modeling head and hand orientation during motion using quaternions*. Tech. rep. SAE Technical Paper, 2004, p. 9. DOI: 10.4271/2004-01-2179.

[24] Sharon Choy, Bernard Wong, Gwendal Simon, and Catherine Rosenberg. "A hybrid edge-cloud architecture for reducing on-demand gaming latency." In: *Multimedia Systems* 20.5 (2014), pp. 503–519. DOI: 10.1007/s00530-014-0367-z.

[25] Cisco. *Cisco Visual Networking Index: Forecast and Methodology, 2016–2021*. Sept. 15, 2017. URL: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html (visited on 05/05/2018).

[26] Cyril Concolato, Jean Le Feuvre, Franck Denoual, Eric Nassor, Nael Ouedraogo, and Jonathan Taquet. "Adaptive Streaming of HEVC Tiled Videos using MPEG-DASH." In: *IEEE Transactions on Circuits and Systems for Video Technology* 8215.c (2017), pp. 1–1. DOI: 10.1109/TCSVT.2017.2688491.

[27] Conviva. *Conviva's All-Screen Streaming TV Census Report Q1 2018 - Measuring the quantity and quality of streaming video*. Tech. rep. 2018. URL: https://www.conviva.com/research/convivas-screen-streaming-tv-census-report-q1-2018/ (visited on 04/26/2018).

[28] Xavier Corbillon. *360Transformations*. 2016. URL: https://github.com/xmar/360Transformations/tree/master/transformation (visited on 04/26/2018).

[29] Xavier Corbillon, Ramon Aparicio-Pardo, Nicolas Kuhn, Géraldine Texier, and Gwendal Simon. "Cross-layer scheduler for video streaming over MPTCP." In: *Proceedings of the 7th International Conference on Multimedia Systems, MMSys 2016, Klagenfurt, Austria, May 10-13, 2016*. 2016, 7:1–7:12. DOI: 10.1145/2910017.2910594.

[30] Xavier Corbillon, Florian Boyrivent, Gregoire Asselin De Williencourt, Gwendal Simon, Géraldine Texier, and Jacob Chakareski. "Efficient lightweight video packet filtering for large-scale video data delivery." In: *IEEE International Conference on Multimedia & Expo Workshops, ICME Workshops*. 2016, pp. 1–6. DOI: 10.1109/ICMEW.2016.7574700.

[31] Xavier Corbillon, Francesca De Simone, and Gwendal Simon. *360-Degree Videos Head Movements Dataset*. 2017. URL: http://dash.ipv6.enstb.fr/headMovements (visited on 06/27/2018).

[32] Xavier Corbillon, Alisa Devlic, Gwendal Simon, and Jacob Chakareski. "Optimal Set of 360-Degree Videos for Viewport-Adaptive Streaming." In: *Proceedings of ACM on Multimedia Conference, MM*. 2017, pp. 943–951. DOI: 10.1145/3123266.3123372.

[33] Xavier Corbillon, Alisa Devlic, Gwendal Simon, and Jacob Chakareski. "Viewport-Adaptive Navigable 360-Degree Video Delivery." In: *IEEE International Conference on Communications (ICC)*. 2017, pp. 1–7. DOI: 10.1109/ICC.2017.7996611.

[34] Xavier Corbillon, Francesca De Simone, and Gwendal Simon. "360-Degree Video Head Movement Dataset." In: *Proceedings of the 8th ACM on Multimedia Systems Conference (MMSys'17)*. 2017, pp. 199–204. DOI: 10.1145/3083187.3083215.

[35] Xavier Corbillon, Francesca De Simone, Gwendal Simon, and Pascal Frossard. "Dynamic Adaptive Streaming for Multi-Viewpoint Omnidirectional Videos." In: *Proceedings of the 9th ACM Multimedia Systems (MMSys)*. 2018, pp. 237–249. DOI: 10.1145/3204949.3204968.

[36] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. "Surround-screen Projection-based Virtual Reality: The Design and Implementation of the CAVE." In: *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '93. Anaheim, CA, 1993, pp. 135–142. DOI: 10.1145/166117.166134. URL: http://doi.acm.org/10.1145/166117.166134.

[37] Lucia D'Acunto, Jorrit van den Berg, Emmanuel Thomas, and Omar Niamut. "Using MPEG DASH SRD for zoomable and navigable video." In: *ACM MMSys*. 2016.

[38] ISO/IEC 23009-1:2014/Amd 2:2015 DASH-SRD. *Spatial Relationship Description, Generalized URL Parameters and Other Extensions*. Standard. Geneva, CH: International Organization for Standardization, 2015. URL: https://www.iso.org/standard/66486.html.

[39] Erwan J. David, Jesús Gutiérrez, Antoine Coutrot, Matthieu Perreira Da Silva, and Patrick Le Callet. "A Dataset of Head and Eye Movements for 360 Degree Images." In: *Proceedings of the 9th ACM on Multimedia Systems Conference - MMSys'18*. ACM Press, 2018, pp. 432–437. DOI: 10.1145/3204949.3208139.

[40] Ana De Abreu, Cagri Ozcinar, and Aljosa Smolic. "Look Around You: Saliency Maps for Omnidirectional Images in VR Applications." In: *Proceedings of IEEE Quality of Multimedia Experience (QoMEX)*. 2017, pp. 1–6. DOI: 10.1109/QoMEX.2017.7965634.

[41] Peiyun Di, Qingpeng Xie, and Jose Alvarez. *Adaptive streaming for FOV switching*. MPEG meeting. m39207. Oct. 2016.

[42] Samuel F. Dodge and Lina J. Karam. "Visual Saliency Prediction Using a Mixture of Deep Neural Networks." In: *IEEE Transactions on Image Processing* 27.8 (2018), pp. 4080–4090. DOI: 10.1109/TIP.2018.2834826.

[43] Christopher Terence John Dodson and Timothy Poston. *Tensor Geometry*. Springer Berlin Heidelberg, 1991. DOI: 10.1007/978-3-642-10514-2.

[44] Marek Domański, Olgierd Stankiewicz, Krzysztof Wegner, and Tomasz Grajek. "Immersive visual media-MPEG-I: 360 video, virtual navigation and beyond." In: *Proceeding of IEEE International Conference on Systems, Signals and Image Processing (IWSSIP)*. 2017, pp. 1–9. DOI: 10.1109/IWSSIP.2017.7965623.

[45] Zhengfang Duanmu, Kai Zeng, Kede Ma, Abdul Rehman, and Zhou Wang. "A Quality-of-Experience Index for Streaming Video." In: *IEEE Journal of Selected Topics in Signal Processing* 11.1 (Feb. 2017), pp. 154–166. ISSN: 1932-4553. DOI: 10.1109/JSTSP.2016.2608329.

[46] Stephan Fremerey, Ashutosh Singla, Kay Meseberg, and Alexander Raake. "AVtrack360: An Open Dataset and Software Recording People's Head Rotations Watching 360° Videos on an HMD." In: *Proceedings of the 9th ACM Multimedia Systems Conference*. MMSys '18. ACM, 2018, pp. 403–408. DOI: 10.1145/3204949.3208134.

[47] Chi-Wing Fu, Liang Wan, Tien-Tsin Wong, and Chi-Sing Leung. "The Rhombic Dodecahedron Map: An Efficient Scheme for Encoding Panoramic Video." In: *IEEE Transactions on Multimedia* 11.4 (2009), pp. 634–644. DOI: 10.1109/TMM.2009.2017626.

[48] Vamsidhar Gaddam, Hoang Ngo, Ragnar Langseth, Carsten Griwodz, Dag Johansen, and Pål Halvorsen. "Tiling of Panorama Video for Interactive Virtual Cameras: Overheads and Potential Bandwidth Requirement Reduction." In: *Picture Coding Symposium (PCS)*. 2015. DOI: 10.1109/PCS.2015.7170076.

[49] Mario Graf, Christian Timmerer, and Christopher Mueller. "Towards Bandwidth Efficient Adaptive Streaming of Omnidirectional Video over HTTP: Design, Implementation, and Evaluation." In: *Proceeding of ACM Multimedia Systems (MMSys)*. 2017, pp. 261–271. DOI: 10.1145/3083187.3084016.

[50] Ahmed Hamza and Mohamed Hefeeda. "A DASH-based Free Viewpoint Video Streaming System." In: *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop (NOSSDAV)*. ACM. 2014, p. 55. DOI: 10.1145/2578260.2578276.

[51] Ahmed Hamza and Mohamed Hefeeda. "Adaptive Streaming of Interactive Free Viewpoint Videos to Heterogeneous Clients." In: *Proceeding of ACM Multimedia Systems (MMSys)*. 2016, p. 10. DOI: 10.1145/2910017.2910610.

[52] Mohammad Hosseini and Viswanathan Swaminathan. "Adaptive 360 VR Video Streaming Based on MPEG-DASH SRD." In: *Proceeding of IEEE International Symposium on Multimedia (ISM)*. 2016, pp. 407–408. DOI: 10.1109/ISM.2016.0093.

[53] Mohammad Hosseini and Viswanathan Swaminathan. "Adaptive 360 VR video streaming: Divide and conquer!" In: *IEEE International Symposium on Multimedia (ISM)*. IEEE. 2016, pp. 107–110. DOI: 10.1109/ISM.2016.0028.

[54] Hristina Hristova, Xavier Corbillon, Gwendal Simon, Viswanathan Swaminathan, and Alisa Devlic. "Heterogeneous Spatial Quality for Omnidirectional Video." In: *Proceeding of IEEE International Workshop on Multimedia Signal Processing (MMSP)*. 2018.

[55] Jingwei Huang, Zhili Chen, Duygu Ceylan, and Hailin Jin. "6-DOF VR videos with a single 360-camera." In: *IEEE Virtual Reality (VR)*. 2017, pp. 37–44. DOI: 10.1109/VR.2017.7892229.

[56] S. Ibrahim, A. H. Zahran, and M. H. Ismail. "SVC-DASH-M: Scalable video coding dynamic adaptive streaming over HTTP using multiple connections." In: *2014 21st International Conference on Telecommunications (ICT)*. May 2014, pp. 400–404. DOI: 10.1109/ICT.2014.6845147.

[57] Google Inc. *Google Earth VR*. URL: https://vr.google.com/earth/ (visited on 04/26/2018).

[58] Institute for Telecommunication Sciences. *Video Quality Metric (VQM) software*. URL: http://www.its.bldrdoc.gov/vqm (visited on 06/29/2018).

[59] *Omnidirectional Media Application Format (OMAF) Committe Draft*. •. ISO/IEC JTC1/SC29/W11. Jan. 2017.

[60] ISO/IEC 23009-1:2014. *Information tech.: Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats*. 2014.

[61] *Dynamic adaptive streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats*. ISO/IEC JTC1/SC29. https://www.iso.org/standard/65274.html. May 2014.

[62] Laurent Itti. "Automatic foveation for video compression using a neurobiological model of visual attention." In: *Transactions on Image Processing* 13.10 (2004), pp. 1304–1318. DOI: 10.1109/TIP.2004.834657.

[63] ITU-T Recommendation P.910. *Subjective Video Quality Assessment Methods for Multimedia Applications*. Standard. Geneva, CH: International Organization for Standardization, 1999.

[64] JTC1/SC22/WG21. *The C++Standards Committee – ISOCPP*. URL: http://www.open-std.org/jtc1/sc22/wg21/ (visited on 04/26/2018).

[65] JVET. *360Lib*. URL: https://jvet.hhi.fraunhofer.de/svn/svn_360Lib/tags/ (visited on 04/26/2018).

[66] Juho Kannala and Sami S. Brandt. "A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses." In: *transactions on pattern analysis and machine intelligence* 28.8 (2006), pp. 1335–1340. DOI: 10.1109/TPAMI.2006.153.

[67] Michael Kazhdan and Hugues Hoppe. "Metric-aware Processing of Spherical Imagery." In: *Proceedings of ACM SIGGRAPH Asia Conference*. 2010, pp. 1–10. DOI: 10.1145/1866158.1866175.

[68] Ingo Kofler, Robert Kuschnig, and Hermann Hellwagner. "Implications of the ISO base media file format on adaptive HTTP streaming of H. 264/SVC." In: *Consumer Communications and Networking Conference (CCNC), 2012 IEEE*. IEEE. 2012, pp. 549–553. DOI: 10.1109/CCNC.2012.6180986.

[69] Ari Koivula, Marko Viitanen, Ari Lemmetti, Jarno Vanne, and Timo D. Hämäläinen. "Performance evaluation of Kvazaar HEVC intra encoder on Xeon Phi many-core processor." In: *IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. 2015, pp. 1250–1254. DOI: 10.1109/GlobalSIP.2015.7418398.

[70] Jari Korhonen and Junyong You. "Improving objective video quality assessment with content analysis." In: *Proceeding of the International Workshop on Video Processing and Quality Metrics for Consumer Electronics*. 2010, pp. 1–6.

[71] Christian Kreuzberger, Daniel Posch, and Hermann Hellwagner. "A Scalable Video Coding Dataset and Toolchain for Dynamic Adaptive Streaming over HTTP." In: *Proceedings of the 6th ACM Multimedia Systems Conference*. MMSys '15. ACM, 2015, pp. 213–218. DOI: 10.1145/2713168.2713193.

[72] Jonathan Kua, Grenville Armitage, and Philip Branch. "A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP." In: *IEEE Communications Surveys & Tutorials* 19.3 (2017), pp. 1842–1866. DOI: 10.1109/COMST.2017.2685630.

[73] Evgeny Kuzyakov. *End-to-end optimizations for dynamic streaming*. Feb. 2017. URL: https://code.facebook.com/posts/637561796428084 (visited on 04/26/2018).

[74] Evgeny Kuzyakov and David Pio. *Next-generation video encoding techniques for 360 video and VR*. Jan. 2016. URL: https://code.facebook.com/posts/1126354007399553 (visited on 04/26/2018).

[75] Jean Le Feuvre and Cyril Concolato. "Tiled-based adaptive streaming using MPEG-DASH." In: *Proceeding of the 7th International Conference on Multimedia Systems (MMSys'16)*. ACM. 2016, 41:1–41:3. DOI: 10.1145/2910017.2910641.

[76] Jean Le Feuvre, Cyril Concolato, and Jean-Claude Moissinac. "GPAC: open source multimedia framework." In: *Proceeding of ACM Multimedia Conference (MM)*. 2007, pp. 1009–1012. DOI: 10.1145/1291233.1291452.

[77] Paul J. Leach, Rich Salz, and Michael H. Mealling. *A Universally Unique IDentifier (UUID) URN Namespace*. RFC 4122. July 2005. DOI: 10.17487/rfc4122. URL: https://rfc-editor.org/rfc/rfc4122.txt.

[78] Jong-Seok Lee, Francesca De Simone, and Touradj Ebrahimi. "Efficient video coding based on audio-visual focus of attention." In: *Journal of Visual Communication and Image Representation* 22.8 (2011), pp. 704–711. DOI: 10.1016/j.jvcir.2010.11.002.

[79] Maxim Levkov. "Video encoding and transcoding recommendations for HTTP Dynamic Streaming on the Adobe® Flash® Platform." In: *White Paper, Adobe Systems Inc* (2010).

[80] *Libav*. https://libav.org/. Accessed: 2015-11-12.

[81] Chenghao Liu, Imed Bouazizi, and Moncef Gabbouj. "Rate Adaptation for Adaptive HTTP Streaming." In: *Proceeding of ACM Multimedia Systems Conference (MMSys)*. 2011, pp. 169–174. DOI: 10.1145/1943552.1943575.

[82] Wen-Chih Lo, Ching-Ling Fan, Jean Lee, Chun-Ying Huang, Kuan-Ta Chen, and Cheng-Hsin Hsu. "360° Video Viewing Dataset in Head-Mounted Virtual Reality." In: *Proceedings of the 8th ACM on Multimedia Systems Conference*. 2017, pp. 211–216. DOI: 10.1145/3083187.3083219.

[83] William Martin McClain. In: *Symmetry Theory in Molecular Physics with Mathematica. A new Kind of Tutorial Book*. New York, NY: Springer New York, 2008. Chap. 11. Euler rotation matrices, pp. 137–143. DOI: 10.1007/b13137.

[84] Branislav Micušık. "Two-view geometry of omnidirectional cameras." PhD thesis. Czech Technical University, 2004.

[85] Kiran Misra, Andrew Segall, Michael Horowitz, Shilin Xu, Arild Fuldseth, and Minhua Zhou. "An Overview of Tiles in HEVC." In: *IEEE Journal of Selected Topics in Signal Processing* 7.6 (Dec. 2013), pp. 969–977. DOI: 10.1109/JSTSP.2013.2271451.

[86] *MIT Open Source License*. https://opensource.org/licenses/MIT. Mar. 2017.

[87] Jason D. Moss and Eric R. Muth. "Characteristics of head-mounted displays and their effects on simulator sickness." In: *Human Factors: The Journal of the Human Factors and Ergonomics Society* 53.3 (2011), pp. 308–319. DOI: 10.1177/0018720811405196.

[88] Debargha Mukherjee, Jim Bankoski, Adrian Grange, Jingning Han, John Koleszar, Paul Wilkins, Yaowu Xu, and Ronald Bultje. "The latest open-source video codec VP9-an overview and preliminary results." In: *Picture Coding Symposium (PCS), 2013*. IEEE. 2013, pp. 390–393. DOI: 10.1109/PCS.2013.6737765.

[89] Ramakrishnan Mukundan. "Quaternions: From classical mechanics to computer graphics, and beyond." In: *Proc. of the Asian Technology conference in Mathematics*. 2002.

[90] Ulrich Neumann, Thomas Pintaric, and Albert Rizzo. "Immersive Panoramic Video." In: *Proceedings of the* 8^{th} *ACM international conference on Multimedia*. 2000, pp. 493–494. DOI: 10.1145/354384.376408.

[91] King-To Ng, Shing-Chow Chan, and Heung-Yeung Shum. "Data Compression and Transmission Aspects of Panoramic Videos." In: *IEEE Transactions on Circuits and Systems for Video Technology CSVT* 15.1 (Jan. 2005), pp. 1–15. DOI: 10.1109/TCSVT.2004.839989.

[92] Omar A. Niamut, Axel Kochale, Javier Ruiz Hidalgo, Rene Kaiser, Jens Spille, Jean-Francois Macq, Gert Kienast, Oliver Schreer, and Ben Shirley. "Towards a format-agnostic approach for production, delivery and rendering of immersive media." In: *Proceedings of the 4th ACM Multimedia Systems Conference on - MMSys '13*. 2013, pp. 249–260. DOI: 10.1145/2483977.2484007.

[93] Omar Aziz Niamut, Emmanuel Thomas, Lucia D'Acunto, Cyril Concolato, Franck Denoual, and Seong Yong Lim. "MPEG DASH SRD: spatial relationship description." In: *Proceedings of the 7th International Conference on Multimedia Systems (MMSys'16)*. 2016, p. 34. DOI: 10.1145/2910017.2910634.

[94] Daisuke Ochi, Yutaka Kunita, Akio Kameda, Akira Kojima, and Shinnosuke Iwaki. "Live streaming system for omnidirectional video." In: *IEEE Virtual Reality (VR)*. 2015, pp. 349–350. DOI: 10.1109/VR.2015.7223439.

[95] *OpenGL home page*. https://www.opengl.org/. Mar. 2017.

[96] Urvashi Pal and Horace King. "Effect of UHD High Frame Rates (HFR) on DVB-S2 Bit Error Rate (BER)." In: *SMPTE15: Persistence of Vision - Defining the Future*. 2015, pp. 1–11. DOI: 10.5594/M001606.

[97] Roger Pantos and William May. *HTTP Live Streaming*. RFC 8216. RFC Editor, Aug. 2017.

[98] Stefano Petrangeli, Viswanathan Swaminathan, Mohammad Hosseini, and Filip De Turck. "An HTTP / 2-Based Adaptive Streaming Framework for 360 ° Virtual Reality Videos." In: *In Proceeding of ACM Multimedia (MM)*. 2017, pp. 306–314.

[99] Benjamin Petry and Jochen Huber. "Towards Effective Interaction with Omnidirectional Videos Using Immersive Virtual Reality Headsets." In: *Proceedings of the 6th Augmented Human International Conference*. AH '15. 2015, pp. 217–218. DOI: 10.1145/2735711.2735785.

[100] Dimitri Podborski, Emmanuel Thomas, Miska M. Hannuksela, Sejin Oh, Thomas Stockhammer, and S. Pham. *Virtual Reality and DASH*. Oct. 2017. URL: https://www.ibc.org/delivery/virtual-reality-and-dash-/2445.article (visited on 07/10/2018).

[101] Andrew Pressley. In: *Elementary Differential Geometry*. London: Springer London, 2001. Chap. Gauss's Theorema Egregium, pp. 229–246. DOI: 10.1007/978-1-4471-3696-5_10.

[102] *QoE parameters and metrics relevant to the Virtual Reality user experience (Release 16)*. Technical Report TR 26.929. 3GPP, Mar. 2018.

[103] Feng Quan, Bo Han, Lusheng Ji, and Vijay Gopalakrishnan. "Optimizing 360 Video Delivery Over Cellular Networks." In: *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. 2016, pp. 1–6. DOI: 10.1145/2980055.2980056.

[104] Caramel Quin. "The teardown: Razer OSVR HDK2 virtual reality headset." In: *Engineering & Technology* 11.10 (2016), pp. 80–81. DOI: 10.1049/et.2016.1027.

[105] Yashas Rai, Jesús Gutiérrez, and Patrick Le Callet. "A Dataset of Head and Eye Movements for 360 Degree Images." In: *Proceedings of the 8th ACM on Multimedia Systems Conference - MMSys'17*. ACM Press, 2017, pp. 205–210. DOI: 10.1145/3083187.3083218.

[106] Evguenii A Rakhmanov, EB Saff, and YM Zhou. "Electrons on the sphere." In: *Series in Approximations and Decompositions* 5 (1994), pp. 293–310. DOI: 10.1142/9789814533232_0022.

[107] Ronald Rivest. "The MD5 message-digest algorithm." In: (1992).

[108] Werner Robitza, Marie Neige Garcia, and Alexander Raake. "A modular HTTP adaptive streaming QoE model - Candidate for ITU-T P.1203 ("P.NATS")." In: *2017 9th International Conference on Quality of Multimedia Experience, QoMEX 2017* 1203 (2017). DOI: 10.1109/QoMEX.2017.7965689.

[109] Demostenes Zegarra Rodriguez, Renata Lopes Rosa, and Graca Bressan. "No-reference video quality metric for streaming service using DASH standard." In: *2015 IEEE International Conference on Consumer Electronics (ICCE)*. Vol. 62. 3. IEEE, Jan. 2015, pp. 106–107. DOI: 10.1109/ICCE.2015.7066339.

[110] Miguel Fabián Romero Rondón, Lucile Sassatelli, Frédéric Precioso, and Ramon Aparicio-Pardo. "Foveated Streaming of Virtual Reality Videos." In: *ACM International Conference on Multimedia Systems (MMSys)*. 2018, pp. 494–497. DOI: 10.1145/3204949.3208114.

[111] Gustavo Alberto Rovelo Ruiz, Davy Vanacken, Kris Luyten, Francisco Abad, and Emilio Camahort. "Multi-viewer gesture-based interaction for omni-directional video." In: *Proceedings of the 32nd annual ACM conference on Human factors in computing systems - CHI '14*. 2014, pp. 4077–4086. DOI: 10.1145/2556288.2557113.

[112] Yago Sanchez, Thomas Schierl, Cornelius Hellge, Thomas Wiegand, Dohy Hong, Danny De Vleeschauwer, Werner Van Leekwijck, and Yannick Le Louédec. "Efficient HTTP-based Streaming Using Scalable Video Coding." In: *Image Commun.* 27.4 (Apr. 2012), pp. 329–342. ISSN: 0923-5965. DOI: 10.1016/j.image.2011.10.002. URL: http://dx.doi.org/10.1016/j.image.2011.10.002.

[113] Yago Sánchez, Robert Skupin, and Thomas Schierl. "Compressed domain video processing for tile based panoramic streaming using HEVC." In: *IEEE International Conference on Image Processing (ICIP)*. 2015, pp. 2244–2248. DOI: 10.1109/ICIP.2015.7351200.

[114] Ashutosh Singla, Stephan Fremerey, Werner Robitza, and Alexander Raake. "Measuring and comparing QoE and simulator sickness of omnidirectional videos in different head mounted displays." In: *Quality of Multimedia Experience (QoMEX), 2017 Ninth International Conference on*. IEEE. 2017, pp. 1–6. DOI: 10.1109/QoMEX.2017.7965658.

[115] Robert Skupin, Yago Sanchez, Dimitri Podborski, Cornelius Hellge, and Thomas Schier. "HEVC Tile Based Streaming to Head Mounted Displays." In: *14th IEEE Annual Consumer Communications & Networking Conference (CCNC)* (2017), pp. 613–615. DOI: 10.1109/CCNC.2017.7983191.

[116] Robert Skupin, Yago Sanchez, Dimitri Podborski, Cornelius Hellge, and Thomas Schierl. "Viewport-dependent 360 degree video streaming based on the emerging Omnidirectional Media Format (OMAF) standard." In: *Image Processing (ICIP), 2017 IEEE International Conference on*. IEEE. 2017, pp. 4592–4592. DOI: 10.1109/ICIP.2017.8297155.

[117] Robert Skupin, Yago Sanchez, Y.-K. Wang, Miska M. Hannuksela, J. Boyce, and Mathias Wien. "Standardization status of 360 degree video coding and delivery." In: *2017 IEEE Visual Communications and Image Processing, VCIP 2017, St. Petersburg, FL, USA, December 10-13, 2017*. 2017, pp. 1–4. DOI: 10.1109/VCIP.2017.8305083.

[118] John P. Snyder. In: *Map Projections: A Working Manual (U.S. Geological Survey Professional Paper 1395)*. Books Express Publishing, Oct. 2012. Chap. Gnomonic Projection, pp. 164–167. ISBN: 978-1782662228.

[119] John P. Snyder. *Map Projections: A Working Manual (U.S. Geological Survey Professional Paper 1395)*. Books Express Publishing, Oct. 2012. ISBN: 978-1782662228.

[120] Iraj Sodagar. "The MPEG-DASH Standard for Multimedia Streaming Over the Internet." In: *IEEE MultiMedia* 18.4 (Apr. 2011), pp. 62–67. DOI: 10.1109/MMUL.2011.71.

[121] Yuhang Song, Mai Xu, Minglang Qiao, Jianyi Wang, Liangyu Huo, and Zulin Wang. "Modeling Attention in Panoramic Video: A Deep Reinforcement Learning Approach." In: 14.8 (Oct. 2017). arXiv: 1710.10755. URL: http://arxiv.org/abs/1710.10755.

[122] speedtest.net. *Speedtest.net report based on Q1-Q2 2017 data for the USA*. 2017. URL: http://www.speedtest.net/reports/united-states/ (visited on 04/29/2018).

[123] Kashyap Kammachi Sreedhar, Alireza Aminlou, Miska M Hannuksela, and Moncef Gabbouj. "Viewport-Adaptive Encoding and Streaming of 360-Degree Video for Virtual Reality Applications." In: *IEEE International Symposium on Multimedia (ISM)*. IEEE. 2016, pp. 583–586. DOI: 10.1109/ISM.2016.0126.

[124] James Alfred Steers. In: *An introduction to the study of map projections*. University of London Press, 15th edition, 1970, pp. 135–136.

[125] Thomas Stockhammer. "Dynamic Adaptive Streaming over HTTP –: Standards and Design Principles." In: *Proceedings of ACM Multimedia Systems Conference (MMSys)*. 2011, pp. 133–144. DOI: 10.1145/1943552.1943572.

[126] Tianyu Su, Ashkan Sobhani, Abdulsalam Yassine, Shervin Shirmohammadi, and Abbas Javadtalab. "A DASH-based HEVC multi-view video streaming system." In: *Journal of Real-Time Image Processing* 12.2 (Aug. 2016), pp. 329–342. DOI: 10.1007/s11554-015-0504-8.

[127] Gary J Sullivan, Jens Ohm, Woo-Jin Han, and Thomas Wiegand. "Overview of the high efficiency video coding (HEVC) standard." In: *IEEE Transactions on circuits and systems for video technology* 22.12 (2012), pp. 1649–1668. DOI: 10.1109/TCSVT.2012.2221191.

[128] David Taubman and Michael Marcellin. *JPEG2000 image compression fundamentals, standards and practice: image compression fundamentals, standards and practice*. Vol. 642. Springer Science & Business Media, 2012. DOI: 10.1007/978-1-4615-0799-4.

[129] Emmanuel Thomas. *Draft for VE on region and point description in omnidirectional content*. MPEG meeting. m39576. Oct. 2016.

[130] Emmanuel Thomas. *Descriptions of Comparison Experiments for Omnidirectional Media Application Format*. MPEG meeting. N16639. Jan. 2017.

[131] Guibin Tian and Yong Liu. "Towards Agile and Smooth Video Adaptation in Dynamic HTTP Streaming." In: *Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies*. CoNEXT '12. 2012, pp. 109–120. DOI: 10.1145/2413176.2413190.

[132] Tiledmedia. *The VR company*. 2018. URL: http://tiledmedia.com/ (visited on 07/10/2018).

[133] Marko Tkalcic and Jurij F. Tasic. "Colour spaces: perceptual, historical and applicational background." In: *The IEEE Region 8 EUROCON 2003. Computer as a Tool.* Vol. 1. Sept. 2003, 304–308 vol.1. DOI: 10.1109/EURCON.2003.1248032.

[134] Laura Toni and Pascal Frossard. "Optimal Representations for Adaptive Streaming in Interactive Multiview Video Systems." In: *IEEE Transactions on Multimedia* 19.12 (Dec. 2017), pp. 2775–2787. DOI: 10.1109/TMM.2017.2713644.

[135] Ivana Tosic and Pascal Frossard. "Low bit-rate compression of omnidirectional images." In: *Proceedings of Picture Coding Symposium (PCS)*. 2009, pp. 1–4. DOI: 10.1109/PCS.2009.5167400.

[136] Evgeniy Upenik and Touradj Ebrahimi. "A Simple Method to Obtain Visual Attention Data in Head Mounted Virtual Reality." In: *IEEE Int. Conf. on Multimedia and Expo*. 2017. DOI: 10.1109/ICMEW.2017.8026231.

[137] Evgeniy Upenik, Martin Rerabek, and Touradj Ebrahimi. "A Testbed for Subjective Evaluation of Omnidirectional Visual Content." In: *32nd Picture Coding Symposium*. EPFL-CONF-221560. 2016. DOI: 10.1109/PCS.2016.7906378.

[138] JD Van Ouwerkerk. "Image super-resolution survey." In: *Image and vision Computing* 24.10 (2006), pp. 1039–1052. DOI: 10.1016/j.imavis.2006.02.026.

[139] John Vince. *Rotation transforms for computer graphics*. Springer Science & Business Media, 2011.

[140] *Virtual Reality (VR) media services over 3GPP (Release 15)*. Technical Report TR 26.918. 3GPP, Mar. 2018.

[141] VRTimes. *Comparison Chart of FOV (Field of View) of VR Headsets*. 2015. URL: http://www.virtualrealitytimes.com/2015/05/24/chart-fov-field-of-view-vr-headsets/ (visited on 05/22/2015).

[142] Hui Wang, Vu-Thanh Nguyen, Wei Tsang Ooi, and Mun Choon Chan. "Mixing Tile Resolutions in Tiled Video: A Perceptual Quality Assessment." In: *Proceedings of Network and Operating System Support on Digital Audio and Video Workshop*. NOSSDAV '14. 2014, 25:25–25:30. DOI: 10.1145/2578260.2578267.

[143] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. "Multiscale structural similarity for image quality assessment." In: *Internationnal Conference on Signals, Systems and Computers*. 2003, pp. 1398–1402. DOI: 10.1109/ACSSC.2003.1292216.

[144] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. "Overview of the H.264/AVC video coding standard." In: *IEEE Transactions on circuits and systems for video technology* 13.7 (2003), pp. 560–576. DOI: 10.1109/TCSVT.2003.815165.

[145] Chenglei Wu, Zhihao Tan, Zhi Wang, and Shiqiang Yang. "A Dataset for Exploring User Behaviors in VR Spherical Video Streaming." In: *Proceedings of the 8th ACM on Multimedia Systems Conference*. MMSys'17. 2017. DOI: 10.1145/3083187.3083210.

[146] x265. *Home Page*. URL: https://bitbucket.org/multicoreware/x265/wiki/Home (visited on 05/18/2018).

[147] Jimin Xiao, Miska Hannuksela, Tammam Tillo, and Moncef Gabbouj. "A Paradigm for Dynamic Adaptive Streaming over HTTP for Multi-view Video." In: *Advances in Multimedia Information Processing – PCM 2015*. Ed. by Yo-Sung Ho, Jitao Sang, Yong Man Ro, Junmo Kim, and Fei Wu. Cham: Springer International Publishing, 2015, pp. 410–418. DOI: 10.1007/978-3-319-24078-7_41.

[148] Zhimin Xu, Xinggong Zhang, Kai Zhang, and Zongming Guo. "Probabilistic Viewport Adaptive Streaming for 360-degree Videos." In: *2018 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, May 2018, pp. 1–5. DOI: 10.1109/ISCAS.2018.8351404.

[149] Yan Ye, Elena Alshina, and Jill Boyce. *Algorithm descriptions of projection format conversion and video quality metrics in 360Lib*. Tech. rep. N16699. International Organization for Standardization ISO/IEC JTC 1/SC 29/WG 11 Coding of Moving Pictures and Audio, Jan. 2017. URL: https://mpeg.chiariglione.org/standards/exploration/future-video-coding/n16699-algorithm-descriptions-projection-format-conversion (visited on 06/25/2018).

[150] Yan Ye and Jill Boyce. *Algorithm descriptions of projection format conversion and video quality metrics in 360Lib (JVET-K1004)*. ISO/IEC JTC 1/SC 29//WG 11. July 2018.

[151] Ramin Ghaznavi Youvalari, Alireza Zare, Huameng Fang, Alireza Aminlou, Qingpeng Xie, Miska M. Hannuksela, and Moncef Gabbouj. "Comparison of HEVC coding schemes for tile-based viewport-adaptive streaming of omnidirectional video." In: *19th IEEE International Workshop on Multimedia Signal Processing, MMSP 2017*. 2017, pp. 1–6. DOI: 10.1109/MMSP.2017.8122227.

[152] Matt Yu, Haricharan Lakshman, and Bernd Girod. "A Framework to Evaluate Omnidirectional Video Coding Schemes." In: *Proceeding of IEEE Mixed and Augmented Reality (ISMAR)*. 2015, pp. 31–36. DOI: 10.1109/ISMAR.2015.12.

[153] Dooyeol Yun and Kwangsue Chung. "DASH-based Multi-view Video Streaming System." In: *IEEE Trans. on Circuits and Systems for Video Technology* PP.99 (2017), pp. 1–1. DOI: 10.1109/TCSVT.2017.2690958.

[154] Alex Zambelli. "IIS smooth streaming technical overview." In: *Microsoft Corporation* 3 (2009), p. 40.

[155] Alireza Zare, Alireza Aminlou, and Miska M Hannuksela. "6K Effective Resolution with 4K HEVC Decoding Capability for OMAF-compliant 360° Video Streaming." In: *Proceedings of the 23rd Packet Video Workshop*. ACM. 2018, pp. 72–77. DOI: 10.1145/3210424.3210425.

[156] Alireza Zare, Alireza Aminlou, and Miska M. Hannuksela. "Virtual reality content streaming: Viewport-dependent projection and tile-based techniques." In: *2017 IEEE International Conference on Image Processing, ICIP 2017, Beijing, China, September 17-20, 2017*. 2017, pp. 1432–1436. DOI: 10.1109/ICIP.2017.8296518.

[157] Alireza Zare, Alireza Aminlou, Miska Hannuksela, and Moncef Gabbouj. "HEVC-compliant Tile-based Streaming of Panoramic Video for Virtual Reality Applications." In: *In Proceeding of ACM Multimedia (MM)*. 2016, pp. 601–605. DOI: 10.1145/2964284.2967292.

[158] Alireza Zare, Kashyap Kammachi Sreedhar, Vinod Kumar Malamal Vadakital, Alireza Aminlou, Miska M Hannuksela, and Moncef Gabbouj. "HEVC-compliant viewport-adaptive streaming of stereoscopic panoramic video." In: *Picture Coding Symposium (PCS)*. IEEE. 2016. DOI: 10.1109/PCS.2016.7906401.

[159] Zenith via Statista. *Average daily media consumption per person worldwide*. June 2017. URL: https://www.statista.com/chart/9761/daily-tv-and-internet-consumption-worldwide/ (visited on 05/04/2018).

[160] Weizhan Zhang, Shuyan Ye, Bin Li, Hui Zhao, and Qinghua Zheng. "A priority-based adaptive scheme for multi-view live streaming over HTTP." In: *Computer Communications* 85 (2016), pp. 89–97. DOI: 10.1016/j.comcom.2016.04.001.

[161] Mincheng Zhao, Xiangyang Gong, Jie Liang, Jia Guo, Wendong Wang, Xirong Que, and Shiduan Cheng. "A cloud-assisted DASH-based Scalable Interactive Multiview Video Streaming framework." In: *Picture Coding Symposium (PCS)*. 2015, pp. 221–226. DOI: 10.1109/PCS.2015.7170079.

[162] Chao Zhou, Zhenhua Li, and Yao Liu. "A Measurement Study of Oculus 360 Degree Video Streaming." In: *Proceeding of ACM Multimedia Systems Conference (MMSys)*. 2017, pp. 27–37. DOI: 10.1145/3083187.3083190.

[163] Chao Zhou, Zhenhua Li, Joe Osgood, and Yao Liu. "On the Effectiveness of Offset Projections for 360-Degree Video Streaming." In: *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)* 14.3s (2018), p. 62. DOI: 10.1145/3209660.

APPENDIX

A

RÉSUMÉ EN FRANÇAIS : RENDRE POSSIBLE LA TRANSMISSION VIA L'INTERNET DES PROCHAINES GÉNÉRATIONS DE VIDÉOS INTERACTIVES

A.1 CONTEXTE GÉNÉRAL

Au cours de la dernière décennie, le temps passé par les hommes à visionner les médias sur l'Internet a explosé, et les analystes prédisent qu'Internet deviendra la principale source de divertissement au cours de la prochaine décennie [159]. Cette tendance, illustrée par la Figure 1.1a, est principalement due à la possibilité d'accéder de plus en plus facilement à des médias de haute qualité. Cette facilitation est majoritairement influencée par trois facteurs : (*iv*) Comme le montre la Figure 1.1b, la vitesse moyenne de l'accès à Internet a fortement augmentée au cours de la dernière décennie, ce qui permet aux utilisateurs de consommer, sans délai, du contenu de meilleure qualité. Le déploiement à grande échelle des réseaux d'accès sans fil, tels que le Wi-Fi et la 4G, permet aux utilisateurs de consommer du contenu où et quand ils le souhaitent. (*v*) Les dispositifs capables de décoder du contenu en haute qualité et capables de se connecter à un réseau à haut débit sont devenus courants. Enfin, (*vi*) le nombre de contenus disponibles sur Internet a explosé, grâce aux fournisseurs de contenus, tels que YouTube, Dailymotion, Netflix, Hulu et Twitch, qui permettent aux producteurs de partager toujours plus de contenu multimédias sur Internet. La diversité des types de contenu, tels que les vidéos à la demande (VoD), les rediffusions de programmes de télévision (TV) ou la diffusion en direct d'événements sportifs en ligne, contribue à attirer un large éventail de la population vers ces services.

Parmi tous les médias consommables sur Internet, les utilisateurs passent la majeur partie de leur temps à visionner des vidéos, consommant ainsi une grand partie des ressources de l'Internet. Par exemple, au premier trimestre 2018, plus de 4.80 milliards d'heures de contenu vidéo ont été diffusées sur Internet (ce qui équivaut à 6 siècles de contenu diffusé chaque jour), avec une augmentation de 114 % par rapport à la même période en 2017 [27]. De plus, le streaming de vidéos représentait jusqu'à 73 % du trafic Internet mondial en 2016 et devrait atteindre 82 % d'ici 2021 [25].

Par rapport aux programmes TV traditionnels, les services de streaming de vidéos sur Internet offrent plus de liberté aux utilisateurs et donnent un véritable sentiment de services personnalisés. L'utilisateur peut décider quand et où regarder un contenu, il peut faire une pause, passer une partie et continuer à regarder le contenu plus tard. Cette flexibilité est l'une des principales forces des services de streaming de vidéos. Le sentiment de services personnalisés vient de la possibilité pour le fournisseur d'adapter la qualité de la vidéo transmise selon comment l'utilisateur consomme le média (sur son téléphone portable ou sur un téléviseur, avec une connexion Internet haut ou bas débit), mais surtout grâce aux services très efficaces de recommandation de contenus proposés par la plupart des fournisseurs. L'utilisateur est toujours en mesure de trouver le contenu qu'il aime et de continuer à utiliser la plate-forme du fournisseur de contenu. Ces fournisseurs de contenu sont des sociétés Over-The-Top (OTT) : ils diffusent le contenu multimédia directement sur les appareils de leurs clients sans intermédiaire, ce qui leur permet de collecter des informations précieuses sur leurs utilisateurs.

Les fournisseurs OTT sont capables de diffuser autant de contenu sur Internet grâce aux technologies de streaming utilisant le principe de débit binaire adaptatif (ABR), principe mis en œuvre dans les protocoles du type HTTP Adaptive Streaming (HAS). Le seul HAS disposant d'un standard international est Dynamic Adaptive Streaming over HTTP (DASH). Avec HAS, les fournisseurs de contenu transmettent les vidéos à leurs clients comme les diffuseurs TV classique (c'est-à-dire le même contenu pour plusieurs utilisateurs) tout en leur donnant le sentiment de services personnalisés. L'idée

principale est d'encoder les vidéos en différentes représentations ayant chacune des résolutions et/ou des débits binaires différents. Chaque représentation est divisée en segments de quelques secondes (généralement entre 2 s et 3 s). L'appareil des utilisateurs, généralement nommé client, peut décider du segment à télécharger, en fonction de la bande passante de téléchargement disponible et des ressources de l'appareil, en envoyant des requêtes HTTP. Le fournisseur de services n'a besoin d'envoyer que le segment demandé au client avec un serveur HTTP standard et sans aucun traitement dédié. Toutes les décisions sont prises par le client. Cette technologie peut facilement passer à l'échelle, en évoluant en fonction du nombre de clients et du nombre de vidéos pour s'adapter aux besoins des utilisateurs.

La dernière décennie a aussi été marquée par la montée de l'intérêt des utilisateurs pour les médias interactifs, principalement parce que la technologie nécessaire est de plus en plus abordable. Les médias interactifs sont des contenus multimédias qui peuvent être modifiés activement par un utilisateur pendant la visualisation. Ils utilisent de nouvelles générations de lecteurs vidéos qui exploitent les capacités de calcul croissantes des appareils des utilisateurs pour permettre de nouvelles formes d'interactions. Par exemple, l'utilisateur peut changer la position de la caméra, zoomer à l'intérieur de la vidéo ou même interagir avec des objets à l'intérieur de la vidéo. De nombreux services OTT utilisent des vidéos interactives comme par exemple les jeux vidéos dans les nuages (cloud-gaming), les vidéos multi-vues et les vidéos omnidirectionnelles.

Une vidéo omnidirectionnelle, aussi appelée vidéo sphérique ou vidéo 360°, est une vidéo avec des pixels capturés dans toutes les directions de l'espace. Lorsqu'un utilisateur regarde une telle vidéo, seule une petite partie de la vidéo, nommée viewport (ou fenêtre de visualisation en français), est affichée à l'écran de l'utilisateur. On peut représenter une vidéo omnidirectionnelle par une vidéo sur la surface intérieure d'une sphère. Pour générer le viewport, une « caméra virtuelle » est positionnée au centre de la sphère et n'extrait que la partie de la vidéo dans son Champ de Vision (CdV). L'orientation de la caméra virtuelle est contrôlée par un retour de l'utilisateur : si la vidéo est visionnée à l'aide d'un Casque de Réalité Virtuel (CRV), le lecteur de vidéo omnidirectionnelle utilise l'orientation de la tête de l'utilisateur, sinon il utilise les entrées d'un clavier ou d'une télécommande.

Le contenu multimédia en Réalité Virtuelle (RV) vise à offrir aux utilisateurs une sensation d'immersion élevée. Les utilisateurs doivent avoir l'impression de vivre une expérience réelle : la barrière entre réalité et virtualité disparaît. Un tel niveau d'immersion ne peut être atteint qu'avec un haut niveau d'interactivité entre les utilisateurs et le contenu multimédia.

Pour assurer une bonne immersion dans le contenu, la résolution spatiale de la viewport doit être d'au moins 4096×2048 pixels (4K), sinon l'utilisateur peut voir le bord des pixels lorsqu'il utilise un CRV. Les vendeurs de CRV recommandent que l'ensemble du système puisse réagir au mouvement de la tête aussi vite que la période de rafraîchissement du CRV pour éviter le mal des simulateurs [87] : 11 ms pour les CRV à 90 Hz. En d'autres termes, le temps de réaction mouvement/photon, c'est à dire le délai entre un mouvement de la tête et l'affichage de la première vue correspondant à ce mouvement de tête, doit être inférieur à 11 ms.

Streamer de tels contenus omnidirectionnels sur Internet est un défi. Pour obtenir des viewport avec une résolution d'affichage 4K, l'ensemble de la vidéo omnidirectionnel doit être équivalente à une vidéo avec une résolution d'au moins $16\,384 \times 8192$ pixels (16K). La diffusion en continu de l'ensemble du contenu (une vidéo 16K par œil) à 90 images par seconde avec les technologies de streaming existantes nécessiterait plus de 150 Mbit s^{-1} [96], ce qui est bien plus élevé que la vitesse médiane des connexions d'accès à Internet pour les particuliers (64 Mbit s^{-1} sur une connexion fixe aux États-Unis et 22 Mbit s^{-1} sur une connexion mobile aux États-Unis [122]).

L'objectif de cette thèse est de proposer et d'évaluer les modifications de l'architecture existante de diffusion OTT et de proposer de nouvelles façons de représenter les vidéos omnidirectionnelles afin de permettre le streaming de ce type de contenu interactif et immersif sur Internet.

A.2 MOTIVATIONS

Depuis que Neumann, Pintaric et Rizzo [90] ont publié en 2000 le premier article introduisant un système de capture de vidéos « omnidirectionnelles », de nombreux chercheurs ont travaillé sur des vidéos/images omnidirectionnelles. Mais ce que l'on appelle souvent « vidéo omnidirectionnelle » ou « vidéo panoramique » dans la littérature n'est pas exactement la même chose que ce que nous dénotons par vidéo omnidirectionnelle dans cette thèse. Souvent, il s'agit d'un contenu cylindrique, avec des pixels proches de l'équateur, capturés à 360°, sans pixel près des pôles. Il s'agit d'un contenu où les utilisateurs ne peuvent tourner que sur eux-mêmes (i. e. rotation en lacet seulement). Dans cette thèse, on désigne par vidéos panoramiques les vidéos cylindriques à un degré de liberté (1DoF) (voir Figure 1.2a), et par vidéos omnidirectionnelles les vidéos sphériques à trois degrés de liberté (3DoF) (voir Figure 1.2b), et nous nous concentrons principalement sur ces dernières.

Avant le début de cette thèse, peu de publications ont étudié le streaming de vidéos omnidirectionnelles dans le cadre de HAS pour les entreprises OTT, ciblant les CRV. Les travaux existants peuvent être classés en trois groupes : (*i*) la diffusion en continu de vidéos panoramiques sur des écrans de télévision traditionnels, sur des tablettes ou sur d'immenses écrans « omnidirectionnels » appelés écrans de type CAVE [36, 111], (*ii*) l'étude des moyens efficaces de calculer les images à afficher (usuellement appelé faire le rendu du contenu) [8, 9], et (*iii*) l'étude des nouvelles interactions possibles entre l'utilisateur et le contenu [99]. Alface, Macq et Verzijp [6] sont les premiers, en 2012, à étudier le découpage de vidéos omnidirectionnelles en tuiles. Ils modélisent la sélection de la qualité des tuiles, dans le contexte d'un streaming avec bande passante limitée, en un problème d'optimisation linéaire en nombres entiers (OLNE). Ils n'ont pas réalisé l'étude avec le codec vidéo à haute efficacité (HEVC) du Moving Picture Experts Group (MPEG), qui dispose d'une option de tuilage, mais avec le codec pour image JPEG 2000, qui ne permet pas de faire du tuilage pour le streaming. De plus ils n'ont pas discuté de l'intégration du système dans des architectures de streaming de type HAS. Niamut et al. [92] ont été les premiers en 2013 à proposer une architecture de streaming complète pour diffuser des vidéos panoramiques en streaming. Leur solution ne se concentre pas sur les vidéos omnidirectionnelles et n'exploite pas les caractéristiques de ces vidéos. Il n'y avait pas vraiment de consensus au sein de la communauté sur la meilleure façon de préparer le contenu omnidirectionnel avant la diffusion en continu, et sur la façon dont les technologies existantes peuvent être mises à niveau pour soutenir un streaming efficace de vidéos omnidirectionnelles en haute qualité. Le but de cette thèse est de contribuer à combler cette lacune.

Dans cette thèse, nous nous sommes concentrés sur le streaming de vidéos omnidirectionnelles vers des CRV, dans le contexte des fournisseurs de contenu OTT : Architectures HAS, faible temps de réaction mouvement/photon et bande passante variable dans le temps. Dans l'état de l'art, on constate que peu de travaux ont été réalisés sur ce domaine avant 2015. De plus aucun enregistrement publique de trajectoires d'utilisateurs regardant des vidéos omnidirectionnelles avec un CRV n'existait, rendant l'étude des comportements des utilisateurs à l'intérieur d'un contenu omnidirectionnel difficile. Nous discuterons plus en détail de l'état de l'art et des travaux connexe dans le Chapitre 2.

Les objectifs de la thèse sont alors triples :

i. **Architecture de streaming** : Proposer et évaluer une architecture de streaming compatible avec HAS et en particulier avec le protocole DASH largement déployé.

ii. **Théorique** : Proposer une analyse théorique de l'architecture proposée, de la façon de générer les représentations des vidéos pour cette architecture et de la façon d'évaluer objectivement la qualité de chaque représentation.

iii. **Outils pratiques** : Proposer des outils pratiques pour évaluer les différentes propositions et permettre à la communauté de reproduire les résultats de nos travaux.

A.3 CONTRIBUTIONS ET ORGANISATION DU MANUSCRIT

Nous présentons dans ce manuscrit les contributions liées au streaming de vidéos omnidirectionnelles. Nous avons exploité les caractéristiques du contenu omnidirectionnel pour proposer des solutions de streaming adaptatives compatibles avec les architectures de streaming OTT actuelles : principalement avec les architectures de type DASH. Nos contributions dans ce domaine peuvent être classées en trois groupes, présentés dans la Section précédente et détaillés ci-dessous : (*i*) Architecture, (*ii*) Études théoriques, et (*iii*) Outils pratiques.

La reproductibilité des travaux de recherche est la pierre angulaire de la science. Pour aider la communauté à reproduire nos travaux, nous avons publié, avec toutes nos publications, les logiciels, les scripts et les datasets (ensembles de données) utilisés pour produire les résultats. La seule exception est lorsque le titulaire des droits d'auteur des données utilisées interdit la publication des données brutes.

Nos contributions liées au streaming de vidéos omnidirectionnels présentées dans ce manuscrit ne représentent pas tout l'étendu du travail effectué pendant la thèse. La figure 1.3 représente une chronologie des contributions de la thèse commencée en mars 2015 et terminée en mai 2018. Nous avons choisi de ne pas parler des contributions liées au filtrage de trame (avec un fond gris clair dans la Figure) dans le corps de cette thèse pour garder un document cohérent et concis. Les lecteurs de vidéos modernes sont capables de décoder le flux binaire d'une vidéo encodée même s'il manque des trames. Les trames manquantes introduisent de la distorsion dans les images de la vidéo décodée (artefacts de décodage dus à des informations manquantes), mais certaines trames manquantes introduisent plus de distorsions que d'autres. Dans le contexte du streaming de vidéos avec une faible latence, nous avons étudié la possibilité de ne pas transmettre certaines des trames qui introduisent peu de distorsion afin d'atténuer les effets d'une pénurie temporaire de bande passante sur la qualité de la vidéo reçu et afin d'éviter les mises en pauses. Nous avons appliqué cette idée sur un ordonnanceur multicouche de trames vidéos pour Multi-Path Transmission Control Protocol (MPTCP), et sur un ordonnanceur applicatif de trames vidéos utilisant les nouvelles fonctionnalités de HTTP/2. Les lecteurs intéressés pour en savoir plus sur ces contributions peuvent lire CORBILLON et al. [29, 30].

Le manuscrit est structuré comme suit : La partie I contient cette introduction et présente l'état de l'art, la partie II présente et évalue une architecture de streaming qui s'adapte aux viewports et une extension possible à la diffusion vidéo à six degrés de liberté (6DoF), la partie III présente quelques études théoriques sur les vidéos omnidirectionnelles, la partie IV présente quelques outils pratiques développés pour mieux étudier le streaming qui s'adapte aux viewports, les projections des vidéos omnidirectionnelles et les comportements des utilisateurs à l'intérieur du contenu 360°, et enfin la partie V conclut.

Les sous-sections suivantes présentent les contributions qui seront discutées plus en détail dans le corps du manuscrit.

A.3.1 *Architecture de Streaming*

Nous proposons une nouvelle architecture de streaming adaptatif, compatible avec les protocoles de type DASH, pour diffuser des vidéos omnidirectionnelles sur Internet. Nous introduisons le concept de représentations à qualité spatiale hétérogènes, qui sont des représentations vidéo où la qualité n'est pas la même partout. Si une vidéo omnidirectionnelle est encodée avec une qualité spatiale hétérogène, les viewports extraits dans des zones de haute qualité auront moins de distorsion que les viewports extraits dans des zones de qualité inférieure. Nous avons introduit le concept de Région de Qualité plus Élevée (RQE), qui est un sous-ensemble connexe de la sphère avec une qualité supérieure à celle des autres régions sphériques, et le concept de Centre de la Qualité plus Élevée (CQE), qui est le barycentre de la RQE, pour proposer une architecture de streaming viewport adaptatif, qui s'adapte non seulement

à la bande passante disponible mais aussi à l'orientation de la tête de l'utilisateur. Ce travail a été publié dans les actes de la conférence IEEE ICC 2017 [33] et est présenté dans le Chapitre 3.

Cette architecture ne peut qu'offrir 3DoF. En effet, les utilisateurs ne peuvent que tourner leur tête à l'intérieur du contenu. Ils ne peuvent pas faire de translation. Offrir, à la demande, du contenu avec 6DoF reste un défi parce que ça nécessite soit une énorme ressource informatique dédiée pour chaque utilisateur (comme ce qui est fait avec les jeux vidéos dans le cloud), soit nécessite l'utilisation de technologies qui ne sont pas encore matures comme par exemple l'utilisation d'images du champ de lumière (lightfield images en anglais).

Nous avons étudié une prochaine étape possible pour atteindre une expérience utilisateur avec 6DoF : les vidéos omnidirectionnelles multi-vues. Il s'agit d'un ensemble de vidéos omnidirectionnelles synchronisées qui filme la même scène à partir de différents points de vue. L'utilisateur peut se téléporter d'un point de vue à un autre. Il ne s'agit pas d'un scénario 6DoF complet mais plutôt d'un scénario 6DoF discret où l'utilisateur peut choisir librement l'orientation de sa tête mais ne peut se déplacer que vers un sous ensemble fini de position prédéfinie de l'espace. Notre étude discute différentes possibilités d'implémentation pour streamer de telles vidéos omnidirectionnelle multi-vues, modélise la stratégie de téléchargement optimale et évalue deux stratégies radicales de téléchargement. Ce travail a été publié dans les actes de la conférence ACM MMSys'18 [35] et est présenté au Chapitre 4.

A.3.2 *Étude Théorique*

Une projection de la sphère vers le plan est généralement utilisée pour permettre aux vidéos omnidirectionnelles d'être encodées avec des encodeurs conçus pour les vidéos rectangulaires traditionnelles à deux dimensions (2D). Pour mieux comprendre l'impact de la projection de la sphère vers le plan sur la qualité des viewports extraits, nous avons étudié la relation entre la densité de pixels sur la sphère et la distorsion introduite à l'intérieur des viewports. Nous dénotons par échantillonnage sphérique, ou densité de pixels sur la sphère, le nombre de pixels par unité de surface sur la sphère. L'approche de l'échantillonnage sphérique est utile pour étudier les projections qui dégradent de manière continue la qualité de la vidéo. C'est le cas par exemple de « l'offset projection », proposée par Facebook pour générer une représentation avec un RQE et utilisée dans le cas particulier de la projection cube-map [73, 162]. Nous généralisons cette transformation à décalage à n'importe quelle projection. Ce travail est publié dans les actes de la conférence IEEE MMSP'18 [54] et présenté au Chapitre 5.

Savoir générer des représentations avec une RQE n'est pas suffisant, nous avons ausi besoin d'outils pour décider automatiquement de la forme et de la position de la RQE. Notre objectif est de modéliser l'allocation de la RQE indépendamment de la projection de la sphère vers le plan utilisée, en utilisant les statistiques de visualisation des utilisateurs spécifiques au contenu. Nous modélisons ce problème en un problème d'allocation de débit binaire d'encodage dans la vidéo sphérique. L'objectif est de générer n représentations qui respectent une contrainte de débit binaire total, et qui maximisent la qualité à l'intérieur des viewports générés par les trajectoires des mouvements de tête enregistrées. Pour réaliser cette allocation, nous avons introduit la notion de débit binaire surfacique. Ce travail a été publié dans les actes de la conférence ACM MM'17 [54] et est présenté au Chapitre 5.

A.3.3 *Outils Pratiques*

Afin d'évaluer la performance du streaming viewport adaptatif avec différentes projections sphère vers plan, de prédire les futures orientations de tête, nous avons développé un logiciel open source capable de convertir des vidéos omnidirectionnelles d'une projection à l'autre, d'extraire des viewports, de

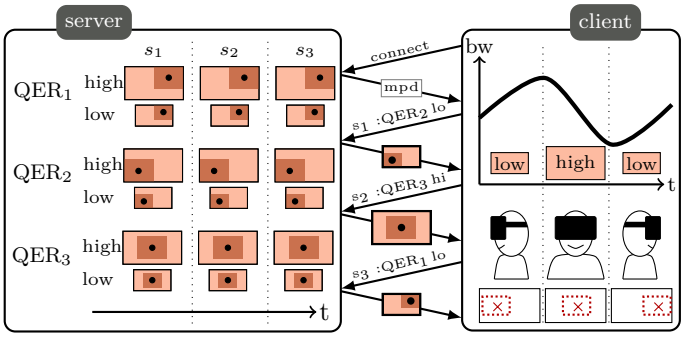


FIGURE A.1 : Illustration de l'architecture de streaming viewport adaptatif. Le serveur prépare différentes représentations de la même vidéo, avec chacune un région avec une qualité plus élevée que le reste et avec un débit binaire différent. Un client reçoit une description des segments disponibles et télécharge ceux, en respectant la contrainte de débit disponible, qui maximisent la qualité dans les viewport prédit.

rejouer des enregistrements de trajectoires de mouvements de tête et de calculer différentes métriques objectives de distorsion. Le logiciel est conçu pour permettre l'insertion facile de nouvelles projections sphère vers plan. Ce logiciel a été utilisé dans la plupart de nos travaux pour évaluer notre proposition et est disponible pour la communauté sur GitHub [28]. Le chapitre 7 décrit sa conception.

Des enregistrements de la trajectoire du mouvement de la tête d'utilisateurs regardant des vidéos omnidirectionnelles sont nécessaires pour permettre à la communauté d'évaluer les différentes propositions de streaming adaptatif, pour comprendre les comportements globaux des utilisateurs dans le contenu omnidirectionnel et pour développer des algorithmes permettant de prédire les positions futures de la tête des utilisateurs. Par exemple, le streaming viewport adaptatif suppose que les mouvements de la tête des utilisateurs peuvent être prédits quelques secondes à l'avance. La définition d'un petit ensemble de RQE suppose qu'il existe des régions d'intérêt (RdI) bien définies et que la plupart des utilisateurs se concentrent sur ces régions.

Pour pallier à ce manque, nous avons mesuré les mouvements de tête de 59 utilisateurs regardant cinq vidéos de 70 s et nous avons publié en libre accès cet ensemble de données. Cet enregistrement est, avec WU et al. [145] et Lo et al. [82], l'un des trois premiers datasets publiques de mouvements de têtes dans les vidéos omnidirectionnelles. Ce dataset a été publié dans les actes de la conférence ACM MMSys'17 [34] et est décrit au chapitre 8.

A.4 RÉSULTATS

Dans le Chapitre 3, nous présentons une architecture de streaming viewport adaptatif compatible avec le protocole de streaming standardisé Dynamic Adaptive Streaming over HTTP (DASH) et avec les architectures de streaming existantes des fournisseurs de contenu. Notre solution, illustrée par la Figure A.1, considère que le fournisseur de service est capable de générer plusieurs représentations d'une vidéo avec les propriétés suivantes : le client peut sélectionner des segments de vidéo de façon à obtenir une vidéo omnidirectionnelle avec une qualité plus élevée dans une direction que dans les autres ; le client peut sélectionner le débit binaire du flux téléchargé. Nous avons étudié la possibilité pour le serveur de générer une RQE en utilisant les faces des objets géométriques à trois dimensions (3D) utilisées pour projeter la vidéo sphérique en une vidéo plate. Nous avons montré, grâce à un petit jeu de traces de mouvements de tête dans des vidéos omnidirectionnelles, que (*iv*) la projection cube-map est la projection la plus efficace dans notre contexte d'étude, (*v*) qu'il est suffisant de générer six représentations au plus, et que (*vi*) que deux secondes est un bon compromis pour la durée des segments de la vidéo.

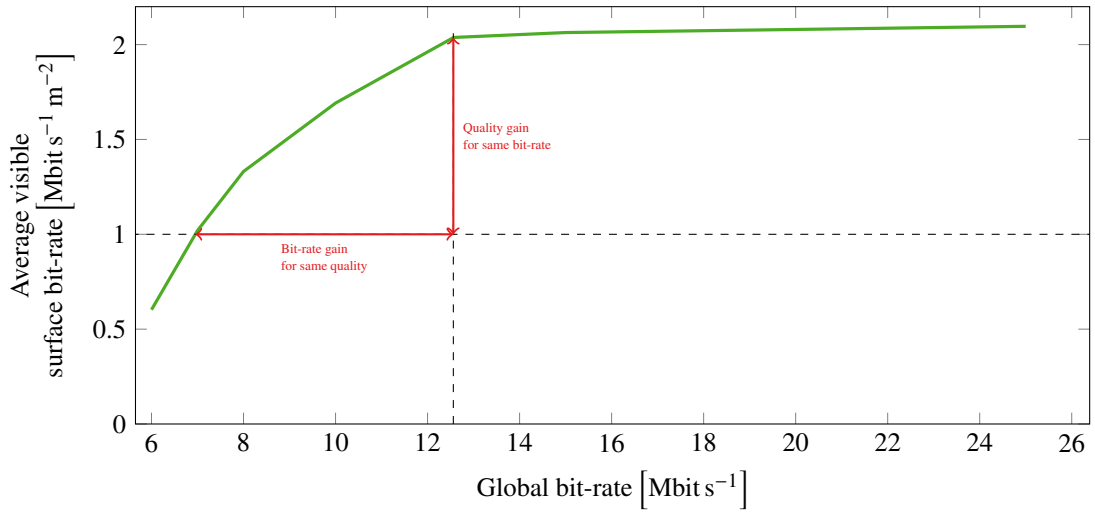


FIGURE A.2 : Moyenne du débit binaire surfacique moyen visible dans les viewport des utilisateurs en fonction dans la quantité totale de bande passante disponible. Les flèches rouges indiquent les gains comparés à la solution de streaming utilisant une vidéo encodée avec une qualité uniforme.

Nous avons proposé dans le Chapitre 4 une extension à notre architecture de streaming viewport adaptatif pour transmettre des vidéos omnidirectionnelles à multiple points de vue (MVP). Une vidéo omnidirectionnelle MVP est une scène enregistrée à l'aide de plusieurs caméras omnidirectionnelles synchronisées. Un utilisateurs peut alors choisir de se téléporter d'un point de vue à l'autre en plus de choisir l'orientation de son viewport. Les téléportations possibles sont définies à l'avance. Nous avons évalué les solutions optimales de trois stratégies de téléchargement côté client utilisant l'architecture proposée. Nous avons mis en évidence le compromis fondamental que tout client doit prendre en considération lors de la planification des segments à télécharger pour maximiser la qualité d'expérience utilisateur. Notre étude met en évidence les gains obtenus lors de l'utilisation de l'encodage par tuilage avec contrainte de mouvement (MCTS), montre qu'une stratégie de téléchargement purement proactive est trop couteuse et que, à l'heure actuel, une stratégie réactive est la meilleure option.

Dans le Chapitre 5 nous étudions l'offset projection introduite par Facebook pour la projection cube-map. Nous la généralisons à toutes projections et nous introduisons la notion de densité sphérique de pixel pour estimer la distorsion introduite par la projection dans les viewports. Nous montrons que la densité sphérique de pixel, dans le contexte de l'offset transformation, est fortement corrélée à la quantité de distorsion introduite dans les viewports. Notre évaluation montre que le tuilage MCTS est globalement plus efficace que l'offset transformation en terme de distorsion mais l'offset transformation permet l'utilisation de vidéos plates avec une faible résolution, autorisant l'usage du codec Advanced Video Coding (AVC) mieux déployer que HEVC.

Le Chapitre 6 présente un modèle permettant la génération de RQE en générant des vidéos à qualité spatialement non homogène. Le modèle utilise des statistiques de visionnage des vidéos pour allouer le débit binaire de manière optimale entre chaque zone de la vidéo. L'objectif est de maximiser les QoE perçu par un utilisateur utilisant une solution de streaming viewport adaptatif comme décrit dans le Chapitre 4 tout en minimisant le nombre de représentations nécessaires. Nous démontrons le gain théorique obtenu en résolvant optimalement ce problème comparé au cas où une vidéo préparée avec une qualité uniforme est streamée. Les viewports affichés à l'écran des utilisateurs ont un débit binaire moyen supérieur de 102 % par rapport au débit binaire moyen des viewports extraient depuis la vidéo à qualité uniforme. Si on stream le contenu avec un débit binaire moyen constant dans les viewport alors le streaming viewport adaptatif permet de réduire de 45 % la bande passante nécessaire pour le streaming. La Figure A.2 illustre ces résultats.

Le Chapitre 7 présente notre logiciel open-source développé pour manipuler les vidéos omnidirectionnelles. Ce logiciel a été conçu pour faciliter l'ajout par la communauté de nouvelles projections sphère-vers-plan et pour calculer des métriques de qualité objective sur les vidéos omnidirectionnelles. Ce logiciel permet d'extraire des viewports et permet de rejouer des enregistrements de déplacements d'utilisateurs qui regardent des vidéos omnidirectionnelles.

Finalement le Chapitre 8 présente le jeu d'enregistrements d'orientations de tête de 59 utilisateurs regardant 5 vidéos omnidirectionnelle d'une durée de 70 s. Nous présentons des statistiques associées à ce dataset. Nous montrons que les statistiques de vitesse de déplacement de tête des utilisateurs varient selon le contenu de la vidéo, mais que globalement un segment de deux secondes reste un bon compromis entre l'efficacité de l'encodage et la distance parcouru par l'utilisateur. Notre jeu de données fait maintenant partie des données officielles utilisées par MPEG, sous le nom de « IMT Atlantique dataset » pour effectuer les tests de conformité des nouveaux standards.

B

REPRODUCIBILITY OF RESULTS ON MVP OMNIDIRECTIONAL VIDEO STREAMING

B.1 OPEN SOFTWARE

We release a part of the software and of the dataset used to generate the results on multi-viewpoint omnidirectional video streaming presented in Chapter 4. In this Annexe, we describe briefly this software and where you can find it.

B.1.1 *Description*

The open software is available on *Github* at the following web address: https://github.com/xmar/MultiViewpoint360_MMSys18. Readme files and docker containers are available to help the reader to run the software.

The software is split into two main pieces:

MILP_MULTIVIEW contains the C++ implementation of the Mixed Integer Linear Programming (MILP) presented in Section 4.5, using the *IBM ILOG CPLEX Optimization Studio*.

RECONSTRUCT contains a *Python3* [1] script that read the output results from the MILP to construct the bitstream as the client would have received it, and use the user head movement records to extract the viewport in the original videos and in the reconstruct bitstream to compute distortion metrics.

Docker[2] containers are available to run the software, and bash script are available to start the docker containers with the right shared resources. The docker images are available on *docker hub*[3] with the tag *mmsys18*, but the reader can build the images from scratch using the *Dockerfile*s available in the repository. The docker images were compiled and tested using docker version *18.03.0-ce* on an Archlinux (4.15.12-1-ARCH x86_64) machine.

The *reconstruct* script requires docker to run the 360Transformations[4] software.

The video dataset is downloaded when running the bash script inside the *reconstruct* folder. Only the 34 s of the video are currently available.

The raw head movement and viewpoint switching dataset is available in the folder *rawNavigationTrace*.

B.1.2 *Docker Installation*

It is not mandatory to use docker to run the different pieces of software but we only provide instructions to use the docker containers.

1 https://docs.python.org/3/
2 https://www.docker.com/
3 https://hub.docker.com
4 https://github.com/xmar/360Transformations/tree/master/transformation

If docker is not installed on your machine, please follow the instructions to install the docker Community Edition available on docker official website[5]. Docker website provides installation instructions for all major operating systems.

Do not forget to start the docker server before trying to run the containers.

B.1.3 *License*

The software is released under the *MIT* license[6].

B.1.4 *Usage Examples*

In this Section we indicates how to run the two pieces of software with docker. It may take a few hours to run each pieces of software. If you want to build the docker images yourself, please read the README files in the github repository. In the following we suppose docker is already installed on running on your machine.

To run the MILP program, open a terminal inside the folder named *MILP_Multiview*, and run the bash script named *./runDockerContainer.sh*.

To run the viewport extraction program, open a terminal inside the folder named *reconstruct*, run the bash script *./buildDockerContainer.sh* to build the last layer of the docker images and then run the bash script *./runDockerContainer.sh*.

5 https://docs.docker.com/install/
6 https://opensource.org/licenses/mit-license.php

C

OPTIMAL QUALITY EMPHASIZED REGION FOR VIEWPORT ADAPTIVE STREAMING: BIT-RATE ALLOCATION

C.1 LIMITS IN THE OPTIMAL BIT-RATE ALGORITHM

Constraint on maximum and minimum bit-rate. Let set $b_{qer} = b_{\max}$, which makes that $s_r \cdot b_{\max}$ bit-rate are used for the QER. The remaining bit-rate can be used to the non-QER: $b_{out} = \frac{B-(s_r \cdot b_{\max})}{4\pi - s_r}$. We know that $b_{\min} \leqslant b_{out}$. So:

$$b_{\min} \leqslant \frac{B-(s_r \cdot b_{\max})}{4\pi - s_r}$$
$$s_r \leqslant \frac{B - 4\pi b_{min}}{b_{max} - b_{min}}$$

Constraint on the quality gap ratio. Let set $b_{qer} = b_{\max}$ and b_{out} be computed from Equation (6.4). However, for some s_r, it can happen that $r_b \cdot b_{out}$ is lower than $b_{\max}$:

$$r_b \cdot \frac{B - b_{\max} \cdot s_r}{4\pi - s_r} \leqslant b_{\max}$$
$$s_r \geqslant \frac{4\pi b_{\max} - r_b \cdot B}{(1-r_b) b_{\max}}$$

Extra bit-rate assignment. In some cases, the algorithm obtains (at the step 3 in Figure 6.2) some so-called *extra bit-rate*, which comes from the quality gap ratio. This extra bit-rate must be assigned to both the QER and non-QER areas while still maintaining the constraints. Let E be the extra-bit-rate. Let y be the ratio of the extra bit-rate that is assigned to the non-QER areas. Let b_{int} be an intermediate surface bit-rate computed as in the step 1 in Figure 6.2. We have:

$$b_{out} = b_{int} + y \cdot \frac{E}{4\pi - s_r}$$
$$b_{qer} = r_b \cdot b_{int} + (1-y) \cdot \frac{E}{s_r}$$

Given that the quality gap ratio is the prevailing constraint in the considered cases, $b_{qer} = r_b \cdot b_{out}$. We thus obtain:

$$r_b \cdot b_{int} + (1-y) \cdot \frac{E}{s_r} = r_b \cdot \left(b_{int} + y \cdot \frac{E}{4\pi - s_r} \right)$$
$$y = \frac{4\pi - s_r}{4\pi + s_r \cdot (r_b - 1)}$$

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Titre : Rendre Possible la Transmission via l'Internet des Prochaines Générations de Vidéos Interactives

Mot clés : Vidéo Omnidirectionelle, Vidéo 360°, Streaming Viewport Adaptif, DASH, HEVC

Résumé : Les vidéos omnidirectionnelles, également appelées vidéos sphériques ou vidéos 360°, sont des vidéos avec des pixels enregistrés dans toutes les directions de l'espace. Un utilisateur qui regarde un tel contenu avec un Casques de Réalité Virtuelle (CRV) peut sélectionner la partie de la vidéo à afficher, usuellement nommée *viewport*, en bougeant la tête. Pour se sentir totalement immergé à l'intérieur du contenu, l'utilisateur a besoin de voir au moins 90 viewports par seconde en 4K. Avec les technologies de streaming traditionnelles, fournir une telle qualité nécessiterait un débit de plus de 100 Mbit s^{-1}, ce qui est bien trop élevé.

Dans cette thèse, je présente mes contributions pour rendre possible le streaming de vidéos omnidirectionnelles hautement immersives sur l'Internet. On peut distinguer six contributions : une proposition d'architecture de streaming viewport adaptatif réutilisant une partie des technologies existantes ; une extension de cette architecture pour des vidéos à six degrés de liberté ; deux études théoriques des vidéos à qualité spatiale non-homogène ; un logiciel open-source de manipulation des vidéos 360° ; et un jeu d'enregistrements de déplacements d'utilisateurs regardant des vidéos 360°.

Title : Enable the Next Generation Interactive Video Streaming

Keywords : Omnidirectional Video, 360° Video, Viewport-Adaptive Streaming, DASH, HEVC

Abstract : Omnidirectional videos, also denoted as spherical videos or 360° videos, are videos with pixels recorded from a given viewpoint in every direction of space. A user watching such an omnidirectional content with a Head Mounted Display (HMD) can select the portion of the video to display, usually denoted as viewport, by moving her head. To feel high immersion inside the content a user needs to see viewport with 4K resolution and 90 Hz frame rate. With traditional streaming technologies, providing such quality would require a data rate of more than 100 Mbit s^{-1}, which is far too high compared to the median Internet access bandwidth.

In this dissertation, I present my contributions to enable the streaming of highly immersive omnidirectional videos on the Internet. We can distinguish six contributions : a viewport-adaptive streaming architecture proposal reusing a part of existing technologies ; an extension of this architecture for videos with six degrees of freedom ; two theoretical studies of videos with non-homogeneous spatial quality ; an open-source software for handling 360° videos ; and a dataset of recorded users' trajectories while watching 360° videos.