360EAVP: THE EDITION-AWARE 360-DEGREE VIDEO PLAYER

GABRIEL DE CASTRO ARAÚJO

DISSERTAÇÃO DE MESTRADO
EM ENGENHARIA ELÉTRICA

DEPARTAMENTO DE ENGENHARIA ELÉTRICA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASÍLIA

Universidade de Brasília
Faculdade de Tecnologia
Departamento de Engenharia Elétrica

360EAVP: The Edition-Aware 360-degree Video Player

360EAVP: Um Reprodutor de Vídeo em 360 Graus Sensível à Edição

Gabriel de Castro Araújo

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA DA UNIVERSIDADE DE BRASÍLIA COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE.

APROVADA POR:

Marcelo Menezes de Carvalho, Doutor (UnB)
(Orientador)

Paulo Roberto de Lira Gondim, Doutor (UnB)
(Examinador Interno)

Debora Christina Muchaluat Saade, Doutora (UFF)
(Examinador Externo)

Brasília/DF, julho de 2023.

FICHA CATALOGRÁFICA

DE CASTRO ARAÚJO, GABRIEL

360EAVP: The Edition-Aware 360-degree Video Player. [Brasília/DF] 2023.

81p., 210 x 297 mm (ENE/FT/UnB, Mestre, Dissertação de Mestrado, 2023).

Universidade de Brasília, Faculdade de Tecnologia, Departamento de Engenharia Elétrica.

Departamento de Engenharia Elétrica

1. 360-Degree Videos
2. Video Processing
3. Streaming
4. Virtual Reality

I. ENE/FT/UnB
II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

DE CASTRO ARAÚJO, GABRIEL (2023). 360EAVP: The Edition-Aware 360-degree Video Player. Dissertação de Mestrado, Publicação PPGEE.XXXXX/2023, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 81p.

CESSÃO DE DIREITOS

AUTOR: Gabriel de Castro Araújo

TÍTULO: 360EAVP: The Edition-Aware 360-degree Video Player.

GRAU: Mestre ANO: 2023

Gabriel de Castro Araújo

Universidade de Brasília (UnB)

Campus Darcy Ribeiro

Faculdade de Tecnologia - FT

Departamento de Engenharia Elétrica(ENE)

Brasília - DF CEP 70919-970

I dedicate this work to my parents, Luís Cláudio and Orcilene, whose unconditional support and encouragement were crucial in every step of this journey.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to all the people who made the accomplishment of this work possible.

To Professor Marcelo, I owe immense gratitude for his guidance, support, and availability throughout the entire journey of the master's program. His presence was crucial in the decisions made. The teachings provided by him are of great value, and I will make sure to apply them when I find myself in similar positions in the future. I appreciate him for being such a helpful person.

To Professor Mylène, my special recognition for the support, especially for the precious advice and the generously dedicated time, allowing the subjective experiments to be conducted properly.

To both professors, I have great admiration for their work and the way they see the human side of relationships.

To my family, I thank you from the bottom of my heart for being unconditionally by my side throughout this journey. In moments of doubt, your presence and support were the solid foundation that sustained and encouraged me to move forward.

To all the colleagues from the GPDS Laboratory, I want to express my recognition. The companionship and knowledge exchange were fundamental to my academic and personal growth. I hope to have contributed equally to their good development.

To all those mentioned and also to those who, in some way, were present along this path, I leave my profound gratitude here. Without the support and collaboration of each one, this work would not have been possible.

ABSTRACT

Compared to traditional videos (2D), which have been studied for over a century, 360-degree videos (omnidirectional videos) represent a relatively new type of media. The exploration of creating more complex videos with storytelling in the 360-degree format is still ongoing. However, due to the fact that 360-degree videos have their own unique characteristics, filmmakers are still establishing their own identity and methods for telling complex narratives. In such cases, they face the challenge of guiding users' attention towards key scenes and ensuring that the intended message is not missed, without the ability to rely on traditional editing techniques due to their lack of control over the camera.

To address this problem, this work introduces 360EAVP, an open-source Web application for streaming and visualization of 360-degree *edited* videos on head-mounted displays (HMD). The platform enables guiding users' attention to the main key scenes, being aware of these scenes before the video starts. This work presents the main features introduced by 360EAVP, which are: 1) operation on HMDs based on real-time user's viewport; 2) dynamic editing via "snap-change" or "fade-rotation"; 3) visibility evaluation of user's Field of View with respect to the player's cubic projection; 4) incorporation of editing timing information into the operation of the ABR algorithm; 5) viewport prediction module based on either linear regression or ridge regression algorithms; and 6) data collection and log module during video playback. The introduced application can be freely used to support research on many topics such as optimization of tile-based 360-degree edited video streaming, psycho-physical experiments, dataset generation, and ABR algorithm development, to name a few. To evaluate the platform capabilities, some proof of concept experiments were made to show how editing techniques can impact the user's experience. Our findings reveal that the implementation of editing techniques did not reduce the overall QoE and comfort. In fact, in certain scenarios, we observed an improvement in both aspects.

Keywords: 360-degree videos, video processing, streaming, virtual reality

RESUMO

Comparados aos vídeos tradicionais estudados há mais de um século, vídeos em 360 graus (vídeos omnidirecionais) representam um tipo de mídia relativamente novo. A exploração da criação de vídeos mais complexos com narrativas no formato de 360 graus ainda está em andamento. No entanto, devido às características únicas dos vídeos em 360 graus, os cineastas ainda estão estabelecendo sua própria identidade e métodos para contar narrativas complexas. Nesses casos, eles enfrentam o desafio de direcionar a atenção dos usuários para as principais cenas e garantir que a mensagem pretendida não seja perdida, sem a capacidade de contar com as técnicas de edição tradicionais devido à falta de controle da câmera.

Para enfrentar esse problema, este trabalho apresenta o 360EAVP, uma aplicação web de código aberto para transmissão e visualização de vídeos em 360 graus editados em "Head-mounted Displays" (HMDs). O 360EAVP permite direcionar a atenção dos usuários para as principais cenas-chave, estando ciente dessas cenas antes do início do vídeo. As principais características introduzidas pelo 360EAVP são: 1) operação em HMDs com base no *viewport* do usuário em tempo real; 2) edição dinâmica por meio das técnicas de *snap-change* ou *fade-rotation*; 3) avaliação da visibilidade do *viewport* do usuário em relação à projeção cúbica do player; 4) incorporação das informações das edições na operação do algoritmo ABR; 5) módulo de previsão do *viewport* do usuário com base em algoritmos de regressão de ridge ou linear; e 6) coleta de dados durante a reprodução do vídeo. O 360EAVP pode ser usado em pesquisas sobre diversos temas, como otimização da transmissão de vídeo em 360 graus editado com base em segmentos, experimentos psicofísicos, geração de bases de dados, desenvolvimento de algoritmos ABR, etc. Para avaliar o potencial da plataforma, experimentos subjetivos foram realizados como prova de conceito para exemplificar como as técnicas de edição impactam a experiência do usuário. Os resultados mostram que, em sua maioria, as técnicas de edição não reduziram a qualidade de experiência e conforto do usuário. De fato, em alguns cenários, foi observado uma melhora em ambos aspectos.

Palavras-chave: vídeos em 360 graus, processamento de vídeo, streaming, realidade virtual

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

ω	Angular speed	[radians/second]
t_{init}	Time editing starts	[second]
t_{end}	Time editing ends	[second]
t_{fade}	Time fade animation starts	[second]
d	Size of a cube	[p.u.]
P	Mapped points on the face of a cube	[p.u.]
$P^{up}_{m,n}$	Positions on $\mathbb{R}^3$ of the mapped points of the upper face of a cube with a $m \times n$ map grid	[p.u.]
V_{pw}	Predict window	[second]
λ	Ridge constant	[p.u.]
Cvp	Coordinates of the center of the viewport (yaw, pitch)	[radians]
Cvp_k	Coordinates of the center of the viewport (yaw, pitch) in the k_{th} position of the list of predict window	[radians]
f_k	frame number in the k_{th} position of the list of predict window	[p.u.]
Q	Quality index selected on the ABR algortihm	[p.u.]
Q_{min}	Minimum quality index available	[p.u.]
b	Byte	[byte]
T_{fin}	Time instant the download finishes	[second]
T_{req}	Time instant the request is sent to the server	[second]
T_{tot}	Total period of a request	[second]
Q_{LIST}	List of qualities available	[p.u.]
Q_f	Quality selected by the Edition Aware ABR algorithm	[p.u.]
VP	Viewport position (yaw,pitch) on the Edition Aware ABR algorithm	[radians]
P_f	Request priority on the Edition Aware ABR algorithm	[p.u.]
TP_{AVG}	Average throughput calculated	[bits/second]

T_{pw}	Viewport prediction threshold	[second]
S_C	Current segment number being downloaded	[p.u.]
S_{Play}	Segment number being played	[p.u.]
S_E	Edition segment Number	[p.u.]
F	Face index	[p.u.]

GLOSSARY

VR	Virtual Reality
AR	Augmented Reality
HMD	Head-Mounted Displays
FoV	Field of View
RoI	Region of Interest
ABR	Adaptive Bit Rate
QoE	Quality of Experience
VDTP	DASH Tile-Based Player
360EAVP	Edition-Aware 360-degree Video Player
ERP	Equirectangular Projection
CMP	Cube Map Projection
CDN	Content Distribution Networks
DASH	Dynamic Adaptive Streaming over HTTP
SRD	Spatial Representation Description
MPD	Media Presentation Description
SVC	Scalable Video Coding
SVR	Support Vector Regression
SVM	Support Vector Machine
OMAF	Omnidirectional Media Format
LR	Linear Regression

RR	Ridge Regression
FpS	Frames per Second
MOS	Mean Opinion Scores
DMOS	Difference Mean Opinion Score
DSCQS	Double Stimulus Continuous Quality Scale

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION

In recent years, Virtual and Augmented Reality (VR/AR) technologies have become increasingly popular due to the immersive and interactive experience delivered to users through special devices such as Head-Mounted Displays (HMD). The HMD is a display device worn on the head or as part of a helmet that has a small display optic in front of one (monocular HMD) or each eye (binocular HMD) (SHIBATA, 2002). In fact, the global AR/VR market is projected to reach USD 571.42 billion by 2025 (REPORTS, 2023), and the entertainment sector is experiencing significant growth with the emergence of 360-degree videos (omnidirectional videos). However, significant challenges lie in the development of content that truly exploits the potential of VR/AR technologies since creating compelling and engaging experiences requires a deep understanding of both the technology and the target audience.

To date, the best way to have an immersive experience while watching a 360-degree video is by wearing an HMD. When wearing it, the users are transported into a virtual world that enables them to explore and interact with panoramic videos in a way that can replicate real-life scenarios. Sometimes the user can be introduced into a completely virtual world with no need to replicate real-life scenarios. In such cases, the content creator might even have more room to create new experiences because they are not attached to real-life physical phenomena or structures. The point is, when someone is watching a 360-degree video, he/she can look at any direction and observe the environment from different perspectives, having a sense of presence when they become part of the action. However, unlike traditional videos, the user will have access to only a portion of the whole 360-degree video while watching it. In fact, most of the content delivered to users will not be watched by them. The content the user can actually see is limited by the *Field of View* (FoV) of the device, which is defined as the observable area a person can see through a display device in a given moment. It is affected by the horizontal and

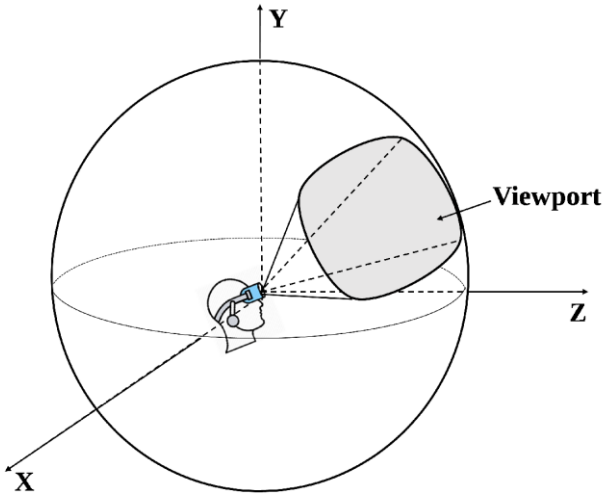

Figure 1.1. *Viewport* visualization when an user is watching a video rendered inside a sphere
Source: (NGUYEN *et al.*, 2020)

vertical angles of a view of the device (e.g., approximately 89° horizontally and 93° vertically for the Meta Quest 2). The FoV is responsible for determining how much content will be displayed on the *user's viewport*, which is the actual portion of the video that is visible to the user. Figure 1.1 illustrates the portion of the video the users will actually see in their viewport.

On the network side, streaming 360-degree videos is not an easy task. As already mentioned, the user is in the center of the scene and only sees a portion of the video at every time instant. This means that, because HMD displays are very close to the eyes, they require a higher resolution than traditional displays. Therefore, 360-degree videos contain significantly more data than traditional videos. For example, to extract a 4K video viewport (3840 × 2160 pixels) from the entire 360-degree video, at least a 12K video resolution (11520 × 6480 pixels) is required (CORBILLON *et al.*, 2017). However, since only a fraction of the video is viewed by the user at any given time instant, the rest of the video may consume memory and network resources without being watched.

To work around this problem, the common streaming strategies of 360-degree videos over the Internet not only divides the video temporally, according to "chunks," but also spatially, by dividing video frames into smaller parts called *tiles*. In this context, it is very popular the adoption of the Dynamic Adaptive Streaming over HTTP (DASH) protocol because it enables seamlessly video playback and dynamic quality adaptation over the intensively used HTTP protocol for Web applications. Moreover, DASH incorporates an extension called the Spatial

Figure 1.2. Visualization of a 2x3 tile grid for a Cube Mapping Projection.

Representation Description (SRD), which further enhances streaming efficiency by allowing the content to be divided not only temporally, into "chunks," but also spatially into "tiles."

When using tiles, one can divide the video into different tile grids and stores them all on a video server. For instance, when we map a 360-degree video into a Cube Mapping Projection (CMP), we are actually doing a 2×3 tile grid, as shown in Figure 1.2, where each face represents a tile, and only the visible faces will show up on the user's viewport. By doing so, the client can reduce the use of network resources by making better decisions, giving less priority to request the tiles that are not visible on the user's viewport.

This revolutionary form of watching 360-degree videos over the Internet opens new possibilities for storytelling, gaming, education, and numerous other fields, as it immerses users in a complex and interactive environment. As the technology continues to advance and filmmakers start understanding how to make better use of it, the potential for creating inspiring and engaging 360-degree video experiences on HMDs is boundless. In this work, we focus on studying videos with complex context. More specifically, the videos presented here are promotional/advertising videos that always have a product/service that they want to sell. Compared with what we have in traditional videos in this field, it is still in its early stages and there is a lot to be learned yet.

Lately, 360-degree content storytelling has attracted significant attention (HAAKE; MÜLLER, 2019) and, due to its nature, new challenges are being posed to film directors, as viewers have control of the camera and the freedom to explore the scene (KROMA, 2022). Consequently, creating a coherent narrative in 360-degree videos is a difficult task due to the likely spatial displacement between the user's viewport and a given *Region of Interest* (RoI) within

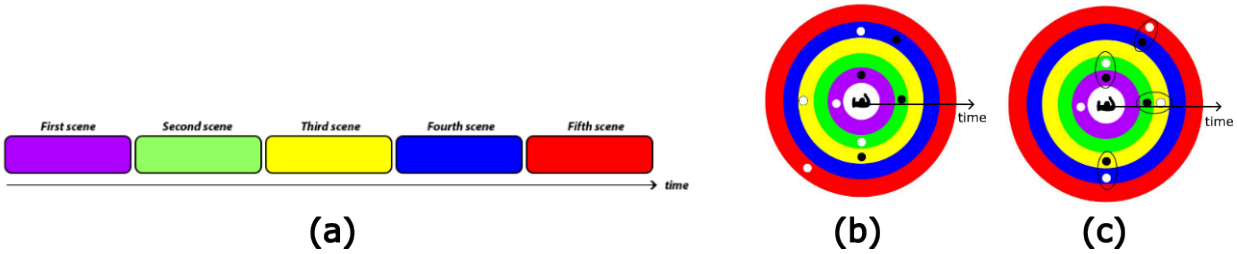

Figure 1.3. Representation of video editing. (a) Traditional 2D video editing: color and length of each rectangle represent a distinct scene and its duration, respectively. (b) 360° video editing: color and width of each concentric track indicate a distinct scene and its duration, respectively. Temporal evolution represented on the radial axis: inner tracks presented earlier than outer tracks. White and black dots represent RoIs at the beginning and end of each scene, respectively. (c) RoI alignment between scenes of the 360° video.
Source:(DAMBRA *et al.*, 2018).

the scene, as intended by a director.

The operation of editing is really a common tool in traditional videos to create this coherent narrative. It consists of ordering a linear sequence of scene cuts in time, as illustrated in Figure 1.3(a). In the case of 360-degree videos, one scene can be represented as a circular track of a certain color, with the radial axis representing the temporal dimension, as presented in Figure1.3(b). In this figure, each colored track represents a different scene, with the width of the track proportional to the duration of the scene. Additionally, the white and black dots indicate the position of a given RoI at beginning and end of each scene, respectively. Therefore, the editing process of 360-degree videos involves not only arranging the sequences of tracks (scenes) on time, but also positioning them in relation to each other to establish a desired visual direction for the viewer, as intended by the filmmaker. Ideally, in any storytelling scenario, the black dot at the end of one scene should be aligned with the white dot of the next scene, as depicted in Figure 1.3(c), in order to guide the viewer through the video's narrative context.

The content alignment that is made during video creation is called "static" editing. If the viewer is looking at the right RoI at the end of a scene, he/she will be correctly realigned to the next RoI without having to explore the next scene to understand its storytelling context. However, since viewers can look anywhere they want, some of them might not be correctly realigned to the next RoI because they were not looking in the desired direction. In such cases, they will miss the RoI in the next scene. Additionally, if the filmmaker wants to change a scene alignment, a new video generation must be done.

On the other hand, when the realignment of the content is done at runtime, we call it

"dynamic" editing. The content rendered on a 3D object (sphere or cube) is rotated in real time, taking into account where the viewer is currently looking at, so that we can ensure that the viewer will always follow the RoIs that compose the video narrative. Such an operation requires knowledge of the user's viewport to rotate, if necessary, the content at the appropriate angle at the time of a scene cut. Therefore, depending on the current user's viewport, content rotation might not be necessary because the next RoI will already be within the user's field of view. Thus, in addition to tracking the user's viewport, accurate *prediction* of the user's future direction is also required in order to determine whether or not to trigger the dynamic editing.

Note that, in the context of tile-based video streaming, having prior knowledge of where the RoIs will be located in the future is a powerful piece of information that can significantly enhance the decision-making process of Adaptive Bit Rate (ABR) algorithms, if used correctly. Therefore, incorporating the knowledge of scene cuts and RoIs into tile-based 360-degree video ABR algorithms can facilitate the prediction of the user's viewport and improve the selection of video quality. Furthermore, in addition to predicting the user's viewport, ABR algorithms need to determine which tiles are visible within the user's viewport in order to make better decisions. In particular, the successful integration of tile-based operation with DASH is key to effective bandwidth usage and timely presentation of the intended RoIs across the scene cuts.

With all the information presented above, it becomes evident that editing techniques in 360-degree videos have the potential to enhance the utilization of network resources during the streaming of such content. By providing prior knowledge of the user's viewing focus, these techniques enable ABR algorithms to make better decisions. Additionally, editing techniques serve as a powerful tool to guide the user's attention towards key RoIs within the video. This ensures that the user does not miss the intended message, particularly in more complex video scenarios.

1.2 OBJECTIVES

The main objective of this dissertation is to introduce an open-source platform that facilitates the study of various approaches on the field of streaming 360-degree videos, such as viewport prediction and ABR algorithms. We also want to expose how the use of dynamic edi-

ting techniques can impact the user's QoE, comfort and overall head movement. Specifically, our aim is to enable the exploration of different editing techniques for more complex videos, unlocking new possibilities for storytelling. By providing this platform, we want to encourage the research community to collaborate and promote advancements in the field of 360-degree video streaming and enhancing the overall understanding of immersive storytelling.

1.3 WORK CONTRIBUTION

In this work, we introduce 360EAVP (ARAÚJO *et al.*, 2023), an open-source web application for tile-based DASH streaming and visualization of 360-degree *edited* videos on HMDs. The proposed application is built on top of the VR DASH Tile-Based Player (VDTP)(RAYSEE, 2023) and introduces the following functionalities:

- Operation on HMDs: The only FoV-based algorithm available on VDTP is actually based on mouse movements. Hence, we have added the capability to operate based on the user's FoV on an HMD;
- Dynamic editing module that implements the editing techniques of "snap-change" and "fade-rotation";
- An algorithm to identify visible faces of the cubic projection used by the player, with corresponding visibility evaluation within the user's FoV (i.e. fractions of the visible cubic faces). This is important for streaming of requested tiles;
- Edition-aware ABR algorithm: implementation of editing timing information into the operation of the ABR algorithm;
- Viewport prediction module based on linear regression and ridge regression algorithms;
- Data collection and log generation module.

To evaluate the impact of the dynamic editing techniques over the user's QoE, comfort, and head movement, we conducted a subjective experiment with 15 participants. Our findings reveal that the implementation of editing techniques did not reduce the overall QoE and comfort. In fact, in certain scenarios, we observed an improvement in both aspects.

1.4 MANUSCRIPT PRESENTATION

This dissertation is structured as follows: Chapter 2 provides an introduction to the fundamental topics discussed in this work. Chapter 3 presents an overview of existing research in the field and situates the current work within this context. Chapter 4 focuses on the presentation of the 360EAVP platform itself, detailing its modules and functionalities. In Chapter 5, a subjective experiment is conducted to showcase the capabilities of the platform. Finally, Chapter 6 summarizes the key findings of this work, and outlines what we have as future works.

CHAPTER 2

FUNDAMENTALS

This chapter introduces the main topics that make part of what is discussed in this work so that the reader can easily follow the next discussions. First, we present an introduction to filmmaking on 360-degree videos. Then we present an overview of 360-degree video projections. After this, we discuss video streaming over the Internet and what are the main challenges faced when streaming 360-degree videos over the internet. Finally, it is discussed the type of editing used in this work, based on a 360-degree video perspective, that can be used as a tool to enhance the use of network resources while maintaining a good quality of experience (QoE) to the user.

2.1 FILMMAKING ON 360-DEGREE VIDEOS

360-degree videos are a new way to tell stories. However, it still sticks to the traditional cinematic rules due to the fact that its own rules are yet to have more studies and explorations. Traditional videos filmmaking rules have been intensively studied and developed for over hundred years. Therefore, filmmakers that aim to create contents for VR will unconsciously follow those rules to find a point of reference when telling immersive stories. The traditional rules tend to be the base-line for the content creators and know them first is really important before starting to break them, and when this is mastered, one can review the lessons already learned and experiment with alternative approaches regarding immersive storytelling.

As described in (MURCH, 2001), traditional videos have used cuts intensively, i.e., the joining of many different pieces of film together into one single part, and it is actually intriguing that this displacement works seamlessly for the human perspective because nothing in our day-to-day seems to prepare us for such a thing. However, our brain tends to make a connections from one scene to another using the content context as a guide. The discovery that such technique works so smoothly in the cinematic field was compared by (MURCH, 2001) as the

equivalent to the discovery of flight.

With this in hand, the filmmakers were no longer stuck to work with a film within the same continuous time and space and were able to play with it. Before the use of cuts, the number of subjects were really restricted because it could become a lot complex to organize everything in the same take. Moreover, it was really challenging to do long takes because, the longer it gets the greater are the chances of mistakes. So, there is a considerable logistical problem of getting everything together at the same time and, for practical reasons, we do not see a lot of these types of films after the filmmakers discovered the benefits of using cuts.

When it comes to immersive video contents, we see that a majority of videos are still being made like old traditional videos, i.e., a single continuous shot or just not focusing to create storytelling in this new type of media. One explanation for this is that this type of media have to be concerned about a different problem called *motion sickness*, which can happen when the movement one sees is different from what one feels, so one can feel dizzy and/or nauseous. However, drawing a parallel with the development of traditional video rules and techniques, it can be argued that, although 360-degree videos have this characteristics today this does not mean that it has to be so in the future. As filmmakers continue to study and master the unique rules and techniques of working with immersive videos, they will likely become more comfortable and adept at creating complex and immersive experiences that mitigate the limitations and challenges currently associated with 360-degree videos.

As aforementioned, there are already some lessons learned for 360 videos filmmaking. We will describe some of them in the next lines. All lessons were described by (HOU, 2020) and are a combination of all the knowledge that the author learned in his 360 filmmaker/videographer career.

- *Extreme wide shot* or *Establishing shot* is a common technique used to tell the users where they are. These shots are often used as the first shot of a new scene to establish where the story is taking place. It is known as one of the most important shots on the 360-degree video because it is responsible to introduce the users to the new environment and make them less disorientated. Some filmmakers will give the user 30 seconds to look around to re-orient themselves before introduce any character or action.

- *Medium long shot* shows the character form the knees up. It is used to deliver information and is very common on VR documentaries. The reason to use this type of shot comes from the fact that the character might be in the scene for a longer period of time and if it is too close it can get uncomfortable for the user.

- Other technique really used on the immersive experience is *breaking the fourth wall*. This happens when the users become part of the story and have a role for themselves on the progression of the story. It is a very important factor when aiming for total immersion and presence. However, the filmmakers are also aware that it is important to create some kind of social relationship from the character to the users so that they can feel more comfortable when characters step into their comfort zone.

From the list above, it is evident that all of these lessons learned prioritize user comfort and QoE. With this in mind, there is still a lot to be explored and studied in this new way of storytelling, and we are only scratching the surface of what we can accomplish. As time goes by, similar to what we have witnessed with traditional videos, we will discover more effective methods of creating complex narratives without being constrained by long continuous shots. Just as in traditional videos, the strategic use of cuts can serve as a valuable tool for guiding the story and directing the user's attention to key moments. In the following sections, we will go deeper into some of these editing techniques as potential resources available for filmmakers.

2.2 360-DEGREE VIDEO PROJECTIONS

Unlike traditional videos, 360-degree videos lack specialized video coding algorithms specifically designed for encoding spherical domain videos. To overcome this challenge and improve the efficiency of encoding 360-degree videos, a suggested solution is to project the original spherical information onto a 2D plane before encoding. This allows the encoded video to be decoded by a majority of video decoders.

Although video decoders can handle 360-degree videos when they are projected onto a 2D plane, this process is not without flaws. The transformation from a sphere to a 2D plane introduces artifacts that vary depending on the chosen projection technique. These artifacts include redundant samples, shape distortion, and discontinuous boundaries.

Redundant samples refer to pixels that are unnecessarily decoded due to the projection. In the case of the Equirectangular Projection (ERP), which is the most commonly used projection, redundant samples are mainly found at the poles where the sphere video had to be stretched horizontally. This stretching introduces distortion and increases the number of pixels that need to be decoded.

Shape distortion in the projection affects the efficiency of Motion Estimation and Motion Compensation in video coding. The distortion makes it challenging to accurately predict and compensate for motion between frames, reducing the coding efficiency.

Discontinuous boundaries in the projection impact prediction performance. The abrupt transitions at the boundaries of the projected video can lead to inaccuracies in predicting the content, affecting the overall quality.

The process of creating a projection is illustrated in Figure 2.1. First, the 360-degree video is captured using special cameras capable of stitching the entire image into one panoramic view. Next, the spherical video needs to be projected onto a 2D plane, enabling coders and decoders to work with it. Currently, the most commonly used projections for 360-degree videos are the Equirectangular Projection (ERP) and the Cube Map Projection (CMP).

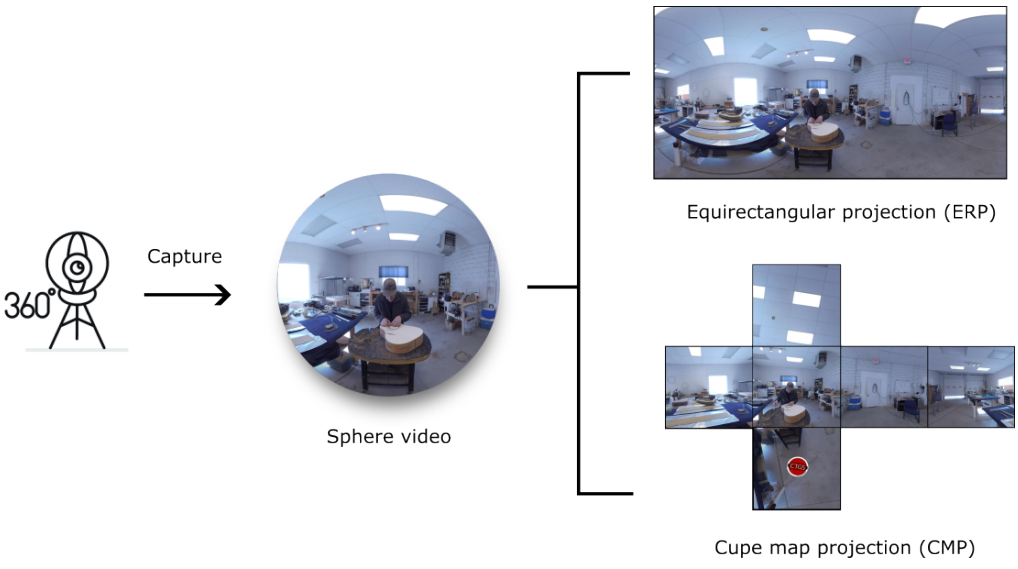

Figure 2.1. 360-degree video projection creation.

The ERP is a simple and widely used method that maps the spherical video onto a rectangle, preserving the entire 360-degree view. However, it suffers from redundant samples, particularly at the poles. It achieves this by stretching the spherical video horizontally, however, it suffers

from redundant samples, particularly at the poles causing distortions on that regions. Despite this distortion, the ERP is commonly used due to its simplicity and compatibility with existing video codecs.

On the other hand, the CMP divides the sphere video into six faces of a cube. It simulates a cube that encloses the sphere and projects each face of the cube with the video content. The cube is then unfolded and rearranged for display. CMP avoids the shape distortion present in the pole region of ERP. However, it causes a suboptimal distribution of resolution, with the content in the center of the cube having better resolution than the content in the corners. Additionally, CMP can introduce discontinuous boundaries, which can be problematic for decoders trying to predict the video content accurately. CMP is more suitable for graphics library rendering (CHEN *et al.*, 2018) , such as A-frame, which is used in this work.

In summary, both ERP and CMP have their advantages and disadvantages. The choice between them depends on factors such as the specific application, compatibility with rendering libraries, resolution distribution requirements, and trade-offs between distortion and decoding efficiency.

In addition to the aforementioned projections, various approaches have been proposed in other studies to enhance the coding performance. Some of these approaches aim to build upon the CMP by utilizing different polyhedrons, such as dodecahedrons and octahedrons, to achieve an optimal sampling rate. While these approaches can reduce the sampling rate, they often introduce significant more discontinuous boundaries.

To address the issue of oversampling, Google and Qualcomm Inc. have introduced two similar techniques: the Equi-Angular Cubemap Projection and the Adjusted Cubemap Projection (GOOGLE, 2017) (COBAN *et al.*, 2017). These methods involve applying a nonlinear transformation to the resulting CMP, which ensures an equal distribution of resolution across each face, overcoming the resolution problem per face of the CMP.

In the presented work, the CMP has been chosen as the primary projection for the player that will be explored in a subsequent chapter. The primary objective is to ensure that the projection aligns seamlessly with the A-frame framework used by the player to handle the rendering process. In this approach, each face of the cube will be transmitted as an individual video to be rendered. Consequently, during video playback, there will be a minimum of six

rendering processes occurring simultaneously. It is worth noting that the study by (CHEN *et al.*, 2018) discovered that the sampling rate does not necessarily correlate with coding performance. Therefore, even though CMP may have a higher sampling rate compared to the Equirectangular Projection (ERP), this does not necessarily lead to a decrease in performance on the client side. This is because both projections, ERP and CMP, will have almost the same processing time for rendering and decoding the video.

2.3 VIDEO STREAMING OVER THE INTERNET

Nowadays, ABR algorithms are commonly used in conjunction DASH protocol. The DASH protocol has been widely adopted in video streaming applications (STOCKHAMMER, 2011). It is based on HTTP, which allows for its widespread deployment on Content Distribution Networks (CDNs) for regular web pages. The DASH protocol divides the input video into fragments of equal time length, also referred to as chunks or segments. Each chunk is decoded in multiple video qualities, enabling the client to request the best chunk quality based on the current network condition. An extension of the DASH protocol is the Spatial Representation Description (SRD) feature. This feature spatially divides the video into parts called tiles, allowing users to selectively retrieve video streams at resolutions that are relevant to their viewing experience (NIAMUT *et al.*, 2016).

To initiate video playback, the client needs to request what is known as the video manifest from the server. The video manifest, also referred to as the *Media Presentation Description* (MPD), is a file that provides a detailed description of the video. It includes information about the available video qualities and the approximate bit rates required to download each chunk with a specific quality. When utilizing DASH-SRD, the MPD file also maintains the available qualities for each tile. The bit rate availability is typically estimated on the client side, serving as an estimation of the network conditions at the time the quality was requested.

ABR algorithms have the capability to utilize various information to determine the optimal quality to request at any given time. Two key aspects that are extensively investigated in this regard are throughput estimation and buffer size. For example, dash.js, which serves as the reference player for the DASH protocol, employs a sophisticated approach that dynamically

switches between rules based on throughput and the BOLA algorithm (SPITERI *et al.*, 2020), a buffer-based algorithm. Additionally, dash.js incorporates secondary rules to address specific scenarios (SPITERI *et al.*, 2019).

2.4 360-DEGREE VIDEO STREAMING OVER THE INTERNET

360-degree videos offer fully immersive experiences that enclose the user, resulting in significantly higher data volume compared to traditional videos. However, as only a fraction of the video is visible to the user at any given time, the remaining portions of the video consume memory and network resources without being seen. Unfortunately, completely eliminating unnecessary resource consumption is challenging, but ABR algorithms primarily aim to minimize its impact.

While optimizations for streaming traditional videos often rely on throughput estimation and buffer size to achieve satisfactory results, optimizations for streaming 360-degree videos require different approaches to minimize the utilization of network resources. One promising approach involves leveraging editing techniques to direct the users' attention while providing valuable information about their future viewing direction. By employing the DASH-SRD protocol, ABR algorithms can prioritize higher quality for the tiles within the user's FoV based on prior knowledge of the users' head movements and estimations of their upcoming RoIs. Consequently, when the RoI changes during playback, the ABR algorithm can request better quality for the tiles that make up that region, especially if the client employs dynamic editing techniques to ensure content alignment, as will be explained in the next section.

2.5 DYNAMIC EDITING IN 360-DEGREE VIDEOS

In this section, we present an overview of the two types of dynamic editing for 360-degree videos implemented in this work. When watching 360-degree videos, the viewer's FoV and the scene being watched (i.e., the sequence of frames in the same time and space presenting a continuous act) change based on head movements. This means that, at any particular moment, two viewers might watch different scenes within the same virtual environment. This presents

a significant challenge in creating a coherent narrative for 360-degree videos, as certain scenes may be critical to understanding the outcome of a story. If a viewer misses these essential scenes, he/she may not fully grasp the intended message of the video. To guide the viewer's attention to the next RoI, traditional videos usually use editing, e.g., transitions from one scene to another. For 360-degree videos, the concept of editing changes as the RoI position may change between scenes. To illustrate this, imagine a scene that the user is inside a room looking outside from a window when suddenly a new character enters the room talking with him/her. In this example, there was not a change of environment, however, the RoI for the user may changed due to the new element introduced in the scene.

Therefore, editing is more related to content alignment than to scene transitions. One way to mitigate these issues is to introduce dynamic editing in the 360-degree video with the goal of drawing the viewer's attention to a specific RoI. These editing techniques can also be used to help ABR algorithms to make more efficient use of network resources and mitigate factors that can negatively impact user experience. This section discusses two types of dynamic editing: "snap-change" and "fade-rotation."

2.5.1 Snap-Change

The "snap-change" editing technique involves a sudden cut that immediately takes the viewer from one view to another. This technique implements an almost instantaneous rotation of the scene from time t_0 to $t_0 + \epsilon$, aligning the viewer's FoV with the editor's intended RoI. As a result, the viewer experiences a more focused and engaging visual experience, which is likely to maintain the viewer's attention on intended RoI. The "snap-change" editing technique has two variants: "static" and "dynamic." Static editing assumes where the viewer will be looking at $t_0 + \epsilon$ so that the edition can realign the complete scene. This technique poses a challenge when working with just one user profile, as viewers who do not fit the assumed viewing profile may have their FoV rotated to an unwanted scene. In contrast, dynamic editing considers the viewer's current FoV at time t_0 and calculate the necessary rotation required to align the viewer's FoV with the editor's intended RoI at time $t_0+\epsilon$. This technique takes into account the center of the user's viewport, making it more dynamic and adaptive for each viewing situation. Therefore, the rotation performed using dynamic editing is not constant, unlike static editing.

In "snap-change" editing, viewers may experience a loss of spatial orientation, particularly when cuts occur within the same scene. This can cause discomfort and disorientation, making it difficult for viewers to follow the flow of narrative. Using dynamic "snap-change" editing reduces some of this discomfort, especially when used together with other editing techniques, such as the fade-rotation discussed in the next section.

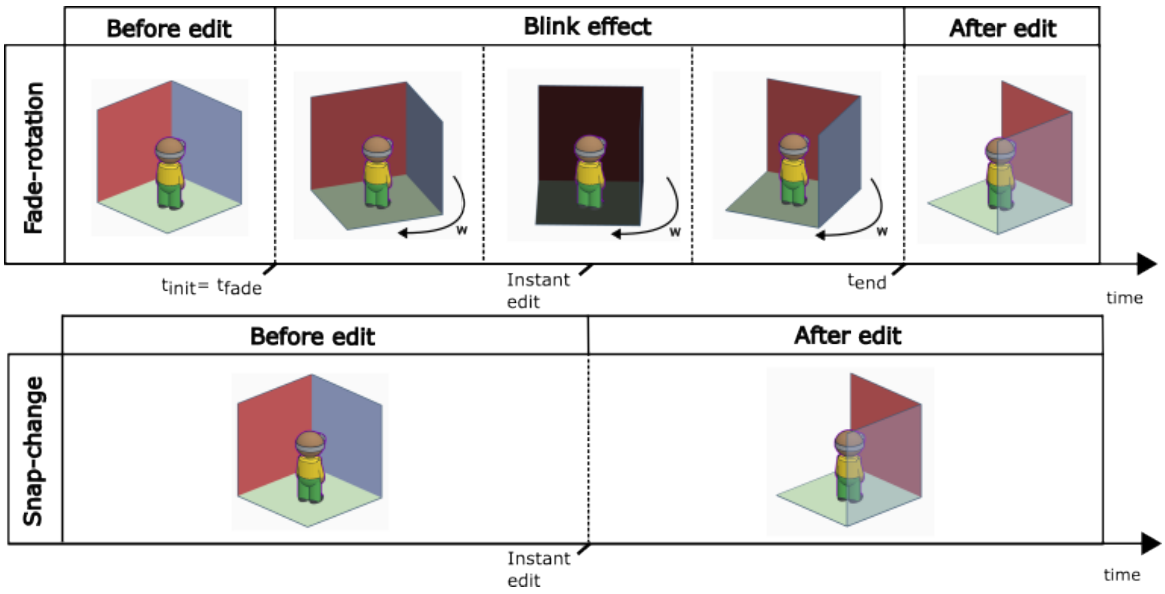

Figure 2.2. Time comparison between the two types of editing: Fade-rotation and Snap-change. In both cases the user is looking at the blue face, and the next RoI is in the red face. The Fade-rotation editing starts before the snap-change effect (blink), and finishes after the instant the editing occurs for a smooth viewing transition.

2.5.2 Fade-Rotation

As mentioned previously, snap-change can lead to spatial disorientation, particularly when multiple cuts occur within the same scene. Another promising technique for video editing is *fade-rotation*, which aims to create a smoother transition in 360-degree videos that can guide the user's attention to the next point of interest without causing spatial disorientation. This is achieved by gradually rotating the scene towards the next point of interest while simultaneously incorporating a "blinking" effect using a combination of fade-in and fade-out techniques. By seamlessly blending scenes, the viewer is better able to follow the narrative flow and maintain spatial orientation, resulting in a more engaging and immersive experience. Figure 2.2 provides a visual comparison of the fade-rotation and snap-change techniques. In both cases, the user is initially viewing the blue face on his/her front, with the RoI being the red face on his/her

left. To enable viewing of the RoI, dynamic editing can be implemented. As illustrated in this figure, the fade-rotation technique begins by gradually rotating the scene and applying a blink effect well before the actual cut. It is necessary to estimate the user's gaze direction at the time of the cut and decide whether the edition is necessary. If the user is already looking towards the next RoI, the edition may not be needed. We have provided a playlist on Youtube with all use cases to exemplify the use of editing with different user's behaviors[1].

[1] <https://youtube.com/playlist?list=PLZEkGMafEvJNUKQqyH7vhCv5IpXNbUvY->

CHAPTER 3

RELATED WORKS

This chapter introduces the main works that already have been done in the field and shows where the presented work fits in it. First, it is shown a summary of the most important strategies already proposed regarding ABR algorithms. Then, it covers the main open-source tools available to analyze and reproduce new ABR approaches and how the application introduced in this works can help other researches in the area.

3.1 ABR ALGORITHMS FOR 360-DEGREE VIDEOS

As discussed in previous chapter, when trying to optimize the streaming of 360-degree videos, ABR algorithms cannot rely only on the throughput estimation and the buffer size in order to increase the QoE. To do so, the user behavior needs to be integrated as part of the equations. To meet this need, the *viewport-based* algorithm approach tries to maximize the quality of the predicted viewport, while minimizing the quality for the rest scene not seen by the user. However, as the player prefetches segments several seconds in advance, the performance of such approach can drastically decrease because unwatched regions can have high quality selected. In addition, bad viewport predictions can lead to regions with low qualities on the user's viewport. Therefore, when employing this method, it is crucial to be careful when determining the buffer size of the player. Previous studies have demonstrated that maintaining accurate viewport prediction becomes increasingly challenging for time lapses longer than 2 seconds. (BAO *et al.*, 2016; QIAN *et al.*, 2016).

The main viewport-based adaptation schemes, including the one used in this work, rely on tile-based streaming. This approach allows for the selection of different quality levels as each tile is independently streamed. The main downsides come from the artifacts that can be seen in the tile borders when they have very different qualities. (CHOPRA *et al.*, 2021) used the

viewport prediction in addition to the video content and the user's preferences on the playback to build their solution. They employed a pyramid-based bit rate allocation scheme and was able to increase significantly the QoE. They claim that the head movement of a user is hardly related to the object's movements in the scene. To evaluate their results, they simulated using two available datasets of 360-degree videos with head movement logs and, therefore, does not have subjective responses of the user's experience. However, their findings serve as a stimulus for further studies on how proper content alignment can reduce head movement, as it helps users stay focused on important objects within the scene. The work presented here utilizes this artifact to enhance the user's QoE.

Using a different approach, Reinforcement learning-based (RLB) algorithms try to optimize the expected long-term QoE by applying its learning strategy. For instance, (KAN *et al.*, 2021) uses a Markov decision process to model the rate adaptation logic and implements a deep reinforcement learning based algortihm to learn the optimal quality selection for each tile. As the other works exposed, the (KAN *et al.*, 2021) opted to make use of simulations, based on viewport datasets and/or a pre-stablished network conditions. (NASRABADI *et al.*, 2017) introduce the use of Scalable Video Coding (SVC) in the purpose of reducing the occurrences of rebuferring, an event that reduces signicantly the users QoE. The idea is to alleviate the restriction mentioned above of the buffer size for viewport-based 360-degree videos player. The SVC is an extension of the H.264 video compression standard and enables the streaming of high-quality videos over layers that can be decoded separately (SCHWARZ *et al.*, 2007). While the results show that there was indeed a rebuffering reduction, the approach does take into account the user's behavior on such events. In occasions where a low quality is shown on the users' viewport, they tend to have an exploratory behavior, as presented in (LINDSKOG; CARLSSON, 2021). This reinforces how the human behavior can affect the results of such methods.

The (GUAN *et al.*, 2019) proposed that the quality of 360-degree videos is perceived differently from the traditional videos due to the presence of moving objects in the scene. In their approach, the adaptation scheme used different tiling schemes to achieve a more precisely quality selection of tiles that compose the object moving. Moreover, their proposal also works with the information of change of brightness from one region to another and with the

so-called Depth-of-Field changes. In a nutshell, the Pano approach tries to assimilate the user's attention with the video content itself to guide their algorithm scheme. Pano's downside is its deployment not be an easy task because it needs changes on the server and the client side.

Some works are more interested in improving the streaming of 360-degree videos on mobile devices, the ones that are much more accessible for the majority of the population than the HMDs available on the market. (ZHANG *et al.*, 2021) studied the impacts of tiling schemes and decoding time over mobile 360-degree video streaming. They proposed that adaptative tiling schemes should be integrated into the bitrate adaptation stream, making the ABR algortihm select not only the next chunk quality but also the tile scheme. To analyze their results they also used simulations over the available head movement logs. In addition to this approach, they analyzed the most common viewport prediction algorithms and showed that simple methods like Linear Regression (LR) can have basically the same result as more complex methods such as Support Vector Regression (SVR). This reassert what (QIAN *et al.*, 2018) did previously. More specific, (QIAN *et al.*, 2018) also tackled the mobile 360-degree video streaming but created a completed system. The system integrates a viewport prediction module that changes dynamically the history window according to the prediction window. They used two prediction techniques: Ridge Regression and Linear Regression selecting then dynamically according to the current prediction window. To measure their results they created a QoE method that minimize the stall duration, the average bitrate consumed by the user, the average quality level and the quality switches. Although it has good results for the mobile streaming, the Flare system itself it is not available for the community and did not take into account the impact of the video content and scenes in the viewport prediction.

Although extensive research has been conducted in the field to find better solutions for 360-degree video streaming, it is understood that only a few studies are exploring 360-degree videos as a new storytelling format, where scenes are carefully assembled by filmmakers to pass a message. The work presented here aims to have a better understanding of how editing techniques can be utilized as a tool to improve the streaming of these videos.

3.2 OPEN-SOURCE 360-DEGREE VIDEO PLAYERS

As far as it is understood, there is a lack of open-source 360-degree video players that can be used to study the impact of editing, ABR algorithms, and viewport prediction approaches on user QoE when content is played on HMDs. Current players are often too specific in their formats or use cases, only supporting certain devices. For example, omaf.js is a proof-of-concept implementation of the Omnidirectional Media Format (OMAF), a system standard developed by MPEG for 360-degree multimedia content (PODBORSKI *et al.*, 2019). Although omaf.js is a web browser-based application that can be used on HMDs, it was not designed to enable other approaches, being a demonstration of the HEVC-based viewport-dependent OMAF video profile. Another popular player is GPAC, which is one of the most renowned media content analysis tools with great modularity capabilities (FEUVRE, 2020). Although it has 360-degree support for H.265/HEVC tile-based DASH streaming, it lacks the ability to analyze the impact of editing, ABR algorithms, and viewport prediction on HMDs. Furthermore, GPAC is not compatible with most HMDs, including the Android version used by current devices. TOUCAN (DAMBRA *et al.*, 2018) is another player available that is based on the H.264 encoder, which is capable of playing locally stored DASH-SRD 360-degree videos while benefiting from editing strategies. TOUCAN is able to manipulate 360-degree contents using different editing strategies, including snap-change used in both static and dynamic scenarios.

One major limitation of using TOUCAN is that it was designed to be used on Samsung devices compatible with Gear VR, a framework for rendering 360-degree videos, which was discontinued at the end of 2020. This limits its use to older devices.

TAPAS-360° is a tool for designing and experimenting viewport-adaptive algorithms for 360-degree videos. It emulates HMD behavior by using head movement data sets. Unfortunately, it does not work with tile-based streaming (RIBEZZO *et al.*, 2020). REEFT-360° provides a framework to evaluate the well-known frame-based QoE metrics of traditional videos in 360-degree videos while considering head movements and network conditions (LINDSKOG; CARLSSON, 2021). It emphasizes the importance of human involvement in ABR algorithm performance and the need for a tool to collect subjective information. This result motivates even more the need for a tool that enables the collection of such subjective information.

Table 3.1. Comparison of the open-source 360-degree video players presented.

Player	Use on HMD	Viewport Pred.	Custom ABR	Editing Support	Data Collection	Projection	Impl.	Coder
OMAF.JS	Yes	No	No	No	No	CMP	Web	HEVC
GPAC	No	No	Yes	No	Yes	ERP	PC-App	HEVC
TOUCAN	Yes	No	No	No	Yes	ERP	Android	AVC
TAPAS-360°	No	No	Yes	No	No	ERP	PC-App	N/A
REEFT-360°	No	No	No	No	No	ERP	PC-App	N/A
VDTP	Yes	No	Yes	No	No	CMP	Web	VP9
360EAVP	Yes	Yes	Yes	Yes	Yes	CMP	Web	VP9

The VR DASH Tile-Based Player (VDTP) is a 360-degree video player that offers users the ability to add custom adaptive bitrate (ABR) algorithms and tile support techniques, and is compatible with different Head-Mounted Display (HMD) models. By using a cube-map projection, VDTP provides several benefits over the more common equirectangular projection, including more efficient loading and rendering, overcoming the pixel redundancy in the poles of the equirectangular projection while having a greater flexibility for post-processing due to separate processing of the faces. During the video tiling process, a spatial division is implemented on the faces of the cubic projection, with each face of the cube having its own video instance. The player handles the synchronization of the video instances for live streaming and locally stored videos. However, as the number of tiles increases, the decentralization issue becomes more pronounced, which can result in a decrease in performance. The 360EAVP inherits this issue from the VDTP, but the problem will be addressed in future works.

VDTP offers three types of ABR algorithms: the lowest-bitrate rule, the FoV-rule, and the default rule. As their names imply, the first algorithm requests the lowest quality level for all faces, while the second algorithm requests the best quality for the visible faces without considering network conditions. However, this algorithm cannot be used on HMDs as it requires mouse grabs and clicks. The default rule, being the standard rule of the DASH reference player, *dash.js*, is a more sophisticated approach that switches dynamically between rules based on throughput and the BOLA algorithm (SPITERI *et al.*, 2020), while also implementing secondary rules to treat special cases (SPITERI *et al.*, 2019). These rules, including the new Edition-aware algorithm, are also available on the 360EAVP. Table 3.1 summarizes the main contributions and features of all the mentioned works, as well as how our platform combines

these key features into a single application.

CHAPTER 4

360EAVP: EDITION-AWARE 360-DEGREE VIDEO PLAYER

In this chapter, we describe the main components of 360EAVP, the edition-aware 360-degree video player that is built on top of the basic structure provided by the VDTP player. We have implemented the player using the Firefox Reality version 12.2 web browser. Compared to existing tools and players in the literature, our player offers significant additions and improvements such as the capability of studying the streaming of videos on HMDs. Figure 4.1 shows a block diagram of the 360EAVP, which summarizes its different modules and their relationships.

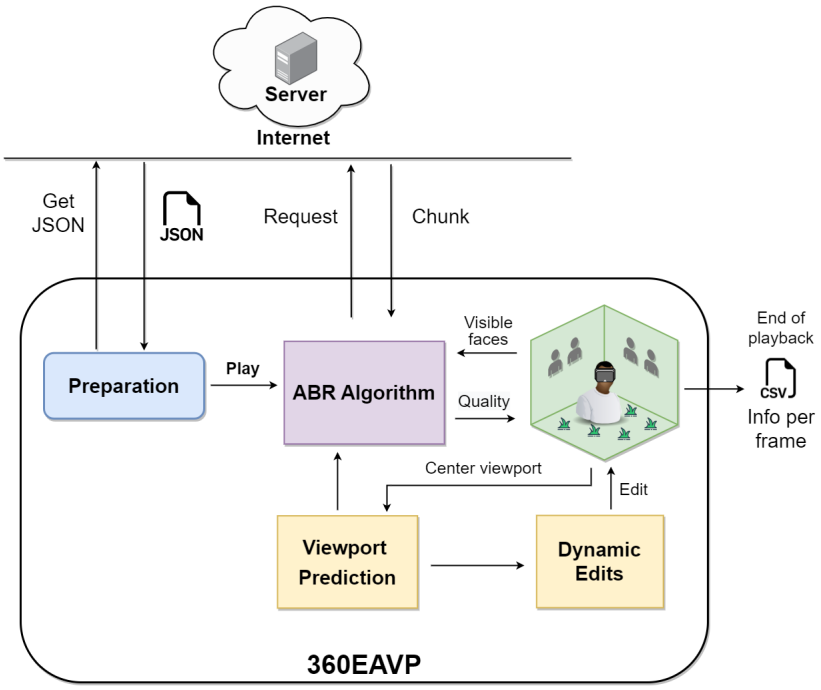

Figure 4.1. *360EAVP* modules: Preparation, ABR algorithm, Viewport Prediction, Visible faces estimation, and Dynamic Edits.

More specifically, 360EAVP includes the following modules: (1) preparation module; (2) dynamic editing module; (3) visible-face identification module; (4) viewport prediction module; (5) edition-aware ABR module; and (6) video playback data tracking module. In this chapter

we describe each of the 360EAVP modules. In addition to this modules, a preprocessing step needs to be performed to make the video conform to the 360EAVP's predetermined format.

4.1 PREPROCESSING STEP

As discussed before, the implemented player requires the input video to be in cubic projection. In addition, the faces of the cubic projection need to be separated into different files and individually compressed with different quality levels. So, to ensure that the 360-degree content is in the correct format, we first check whether the video is in cubic projection format. If not, we perform a conversion using the FFMPEG library. The 360-degree video is sliced into six smaller videos, each representing one face of the cube. These videos, called "mini-videos," are then compressed with different CRFs, creating visual contents with different quality levels. Using the Bento4 library (BENTO4, 2023), the compressed mini-videos are converted into segments that will be used by the DASH protocol. A DASH manifest file is created for each face, thus grouping the content information. Finally, a JSON file that contains the different manifests for the different compressed mini-videos is created, allowing communication between the server and the player. The output JSON file contains information about the location of the manifests for the audio and video content for each face to be reproduced. In addition, it is also necessary to have information on the types and locations of the video edits. One can find the JSON files used in this work in Annex III, where both videos are without audio.

4.2 PREPARATION MODULE

As shown in Figure 4.2, the preparation module consists of three steps: 1) requesting the video JSON from the server, which enables the application to know where to request the videos for each face and fill the corresponding scope variables; 2) rendering the 3D environment to create the objects for each face on which the video will be played; and 3) loading the videos on each face, starting the playback once each face has its buffer completely loaded.

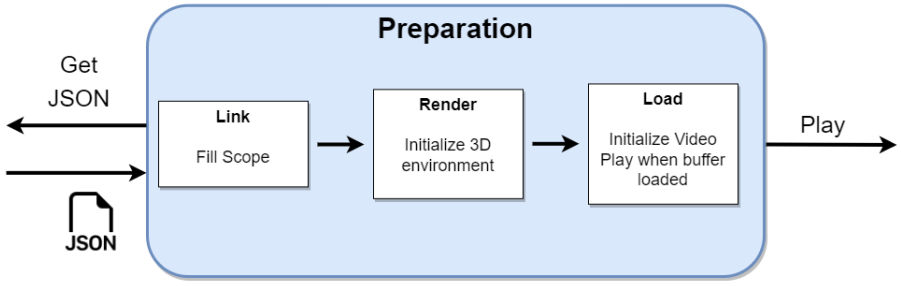

Figure 4.2. Preparation module: First requesting the JSON information for the server, then initializing the 3D environment and then loading the video into the 3D Objects.

4.3 DYNAMIC EDITING MODULE

As mentioned earlier, in this work, we have implemented a dynamic implementation of fade-rotation, which takes into account both the RoI for the cut and the predicted viewport. Edits need to be initiated a few moments before the next RoI to allow for the rotation and blinking effect to occur. The editing method uses several parameters, including the angular speed ω, the start time t_{init}, the blink time t_{fade}, and the end time t_{end}. An important consideration is that gradually rotating the scene at a speed of ω degrees/second may not always be sufficient to reach the intended RoI within the time period of $t_{end} - t_{init}$. To address this issue, a snap-change is added when the "blinking" effect is completely dark. This ensures that the user does not see the instant cut and that the smooth editing effect is maintained. It is important to note that the implemented snap-change does not perfectly align the user's center of viewport with the RoI in the following scene, but the final alignment is achieved through the end of the scene's rotation.

Moreover, we have implemented a feature to determine when dynamic editing is necessary. In certain situations, the users may already be looking at the desired region or, at the very least, be close enough to observe it within their viewport. Therefore, to avoid this micro realignment, we have established that dynamic editing will only be triggered if the user's center of viewport is more than 30 degrees away from the next RoI. This value is configurable, but we chose it based on the "30 Degree Rule," which stipulates that the change in camera angle between two shots of the same subject should exceed 30 degrees (LEXICON, 2000).

4.4 VISIBLE FACE IDENTIFICATION MODULE

Efficient 360-degree video transmission relies on identifying the visible faces in the user's viewport to prioritize their quality, minimize network waste, and optimize transmission. Understanding the user's FoV is essential to achieve this goal. The player projects the user's FoV as a Frustum, which has the shape of a truncated pyramid with the narrow end at the camera and the wider end at the edge of the viewport, as illustrated in Figure 4.3. In other words, the frustum is the 3D shape that represents the user's FoV in a virtual scene and can be used to determine the visible scene objects, excluding objects that do not intercept the camera's view to enhance rendering performance.

To determine which objects fall within the user's FoV, a computer graphic technique called the "Bounding Sphere"(RITTER, 1990) is widely employed. This technique is the standard way on Three.js to check if an object is in the user's viewport. It creates a sphere that encloses a 3D object and uses it to determine whether the object is visible in the user's viewport by checking whether the object's sphere intersects the camera's Frustum. While the Bounding Sphere method is a useful technique for determining which objects are within the user's FoV, it has limitations. In situations where the user is positioned at the center of an object, such as a cube, all of the object's Bounding Spheres intersect the camera's Frustum, meaning that all faces are listed as visible faces. Figure 4.3 shows a simplified scenario in which only the bounding sphere of the face is shown on the back of the user. This highlights the limitations of this method, which results in a waste of resources. The issue arises because even faces that are not visible to the user remain on the list of visible faces. Therefore, the standard function available does not meet the requirements of our project.

To determine which faces are visible in the user's FoV in an efficient manner, an alternative method is necessary. The solution employed in this work to tackle this problem involves using a function that can identify if a point in the 3D space is within the user's FoV. This approach views the faces as a set of predetermined points in the 3D space. To illustrate this method, consider a cube of size d centered on the origin of $\mathbb{R}^3$. For each face of the cube, we map P points that are regularly spaced on the surface of the face. In this work, we consider $P = 25$ and use a 5×5 sampling matrix for each face, with each point consisting of 4% of the total face. These values were chosen empirically to balance the computational cost and precision of

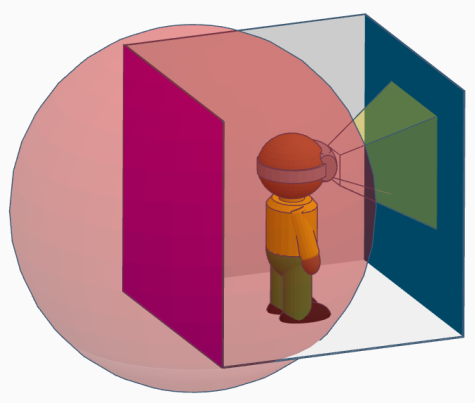

Figure 4.3. User looking at the blue face in his/her front while the Bounding Sphere of the pink face in his/her back is intercepting the user's Frustum in yellow light. The method returns the pink face as visible, which is incorrect.

the method. We found that using a smaller matrix size leads to a lower precision, especially when the user is looking at one of the cube's edges, where the number of visible points is well divided between the faces.

To demonstrate the mapping process, we illustrate the procedure for the upper face, since the other faces are mapped similarly. The upper face of the cube in $\mathbb{R}^3$ has a support plane described by equation $z = \frac{d}{2}$, with its points defined by

$$P(m,n) = P^{up}_{m,n} = \left(\frac{(m-2)d}{4}, \frac{(n-2)d}{4}, \frac{d}{2} \right), \tag{4.1}$$

where $m,n \in \mathbb{N}$, $0 \leq m,n \leq 4$, and the values in parentheses represent the coordinates (x,y,z) in the 3D space. To identify whether the face $P^{up}_{m,n}$ is visible in the user's FoV, the function verifies whether any of the points mapped to the face are within the user's FoV. This approach also provides information on the degree to which the face is visible. For example, if a face has nine points mapped on it and only three of these points are visible, it can be inferred that approximately 33% of the face is visible. This information is important when choosing the quality of a face of the 360-degree projection, as it allows a prioritization of those faces that are more visible.

Figure 4.4 illustrates the proposed approach to identify visible faces. In this example, only two faces are visible to the user, each of which has a total of 25 mapped points. One of the faces has six visible points (24%), while the other has eight visible points (32%). In this particular scenario, since only two faces are visible, quality selection can be more lenient and the percentage of the face that is visible does not need to be considered. However, in a situation

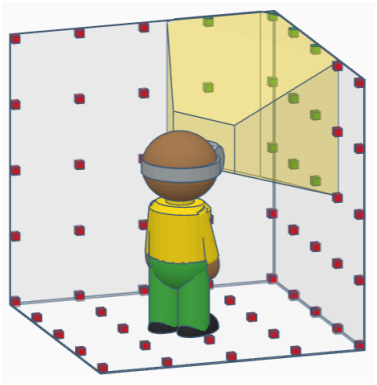

Figure 4.4. Proposed mapped points approach where each face has a 5×5 matrix of points. If at least one of the face mapped points are within the Frustum, the face is visible (green dots - visible, red dots - not visible).

where more faces are visible (e.g., four faces), this information on how much of each face is visible becomes useful in prioritizing the quality of the visible faces.

On the player, the method needs as input the center of the user's viewport (yaw and pitch in radians) to return the list of visible faces and their respective visibility. In our work, this method is used on two occasions. First, by the ABR algorithm, to get the visible faces at the center of the viewport predicted by the viewport prediction module. Second, by the video playback data tracking module, to store the current visible faces.

4.5 VIEWPORT PREDICTION MODULE

In the original VDTP, the center of the user's viewport is identified during video playback using the "click and drag" function of a mouse, which is not suitable for use with an HMD. Since identifying the user's center of the viewport is crucial for effective video playback and action, we developed a solution that works with HMDs and uses the visible face identification module. In other words, our solution tracks the user's gaze in real time, identifying the current visible faces, and providing information on how much of the faces are visible. Instead of using only the current visible faces, the predictor analyzes the user's past head movements to predict his/her future viewport V_{pw} seconds in the future. We keep track of the user's head movement from the last playback second. Since we store the information per frame, the sample size used to predict the viewport is equal to the video frame rate, for example, 60 samples for a 60fps 360-degree video. This approach enables better analysis of user behavior and improved video playback performance.

Previous studies have shown that it is very difficult to maintain good viewport prediction for time lapses longer than 2 seconds (BAO *et al.*, 2016; QIAN *et al.*, 2016). For predictions of up to 1 second, Linear Regression (LR) techniques have similar results to more complex methods (QIAN *et al.*, 2018), such as the Support Vector Machine (SVM). For predictions longer than 1 second, a significant decline in LR efficiency is observed, and more complex methods have been shown to provide better results. For these cases, Ridge Regression (RR) provides a good compromise between computational cost and performance (QIAN *et al.*, 2018), while at the same time making the prediction less biased. 360EAVP has both the LR and RR methods implemented, allowing it to use the best algorithm according to the application.

The RR technique differs from the LR technique because it aims to, not only minimize the root mean squared error, but also to ensure that the regression resulted is not so biased with the input data used. By using a small data sample, the results become more susceptible to *overfitting* problems. To work around this problem, *RR* adds an element in minimizing the root mean squared error as a way to penalize bias. This element is called the Ridge constant λ and its value can be adjusted as needed by the project. In this way, Ridge Regression can provide better performance for predictions longer than 1 second while maintaining a low computational cost.

To ensure that the prediction module always has the most recent inputs to work with, we have implemented a list that stores the last frame numbers and the corresponding centers of the viewport Cvp in terms of yaw and pitch. This list is updated each time a video is rendered, and its size remains constant as specified during player initialization.

The resulted slope of the RR can be described by the equations below:

$$LR_{slope} = \frac{(n\sum_{k=0}^{n} Cvp_k \cdot f_k) - (\sum_{k=0}^{n} f_k \cdot \sum_{k=0}^{n} Cvp_k)}{[n(\sum_{k=0}^{n} f_k^2 + \lambda)] - (\sum_{k=0}^{n} f_k \cdot \sum_{k=0}^{n} f_k)} \tag{4.2}$$

The next equation describes the intercept of the RR technique.

$$LR_{intercept} = \frac{[(\sum_{k=0}^{n} f_k^2 + \lambda) \cdot \sum_{k=0}^{n} Cvp_k] - (\sum_{k=0}^{n} f_k \cdot \sum_{k=0}^{n} Cvp_k \cdot f_k)}{[n \cdot (\sum_{k=0}^{n} f_k^2 + \lambda)] - (\sum_{k=0}^{n} f_k \cdot \sum_{k=0}^{n} f_k)} \tag{4.3}$$

Where n,$k \in \mathbb{N}$, n is the number of samples used to do the prediction , Cvp_k is the k-th center of the viewport value (yaw or pitch) on the list, f_k is the k-th frame number of the video on the list and λ is the Ridge constant. One can notice that the Ridge constant always

follows the $\sum f_k^2$ expression. To ensure that the Ridge constant stays relevant as the frame number increases during run-time, we empirically chose to update its value in real time using $\lambda = 0.05 \cdot \sum f_k^2$. One can also observe that the slope and intercept of the RR only differs from LR by the Ridge constant. Therefore, if $\lambda = 0$, the RR is equal to the LR.

Following the concept above, we focus on optimizing the results and the reliability of the viewport predictor. More specifically, the Edition-Aware ABR algorithm presented in the next section is designed with the most appropriate method chosen based on the prediction time according to a time threshold T_{pw} of 1 second (QIAN *et al.*, 2018). Longer predictions ($T_{pw} > 1$) use the RR method, while shorter predictions ($T_{pw} \leq 1$) use the LR.

4.6 CUSTOM ABR MODULE

To enable the study of different approaches for ABR algorithms, we utilized the existing *Custom Rules* feature of the *dash.js* framework to integrate them into the 360EAVP.

In the *dash.js* platform, a rule is a criteria that returns the highest index corresponding to the most suitable quality. An ABR algorithm can consist of a combination of various rules, and the ultimate decision will be based on the lowest index quality returned by each rule. Within the *dash.js* platform, exists a file named `ABRRulesCollection`, which contains a list of rules.

To make a custom rule, it is necessary to override the `getMaxIndex()` function, in which the logic needs to be encoded. Figure 4.5 shows the *dash.js* default rule flow to exemplify how these rules work together to create an ABR algorithm. The purpose of this explanation is to provide an insight into the underlying processes, enabling new users to better understand the decision-making involved in this rule available on the platform. Additional information can be found on the *dash.js* GitHub page (DASH.JS, 2023). In this approach, the quality selection follows a list of rules before each segment is downloaded. The new quality selection starts calling the function `getMaxQuality()` when a new segment must be downloaded, and then each rule follows determining the maximum possible quality to be selected. The final decision is based on selecting the minimum quality (among the maximum possible quality values) suggested by all the rules.

The algorithm is divided into two parts: primary and secondary rules. The primary rules

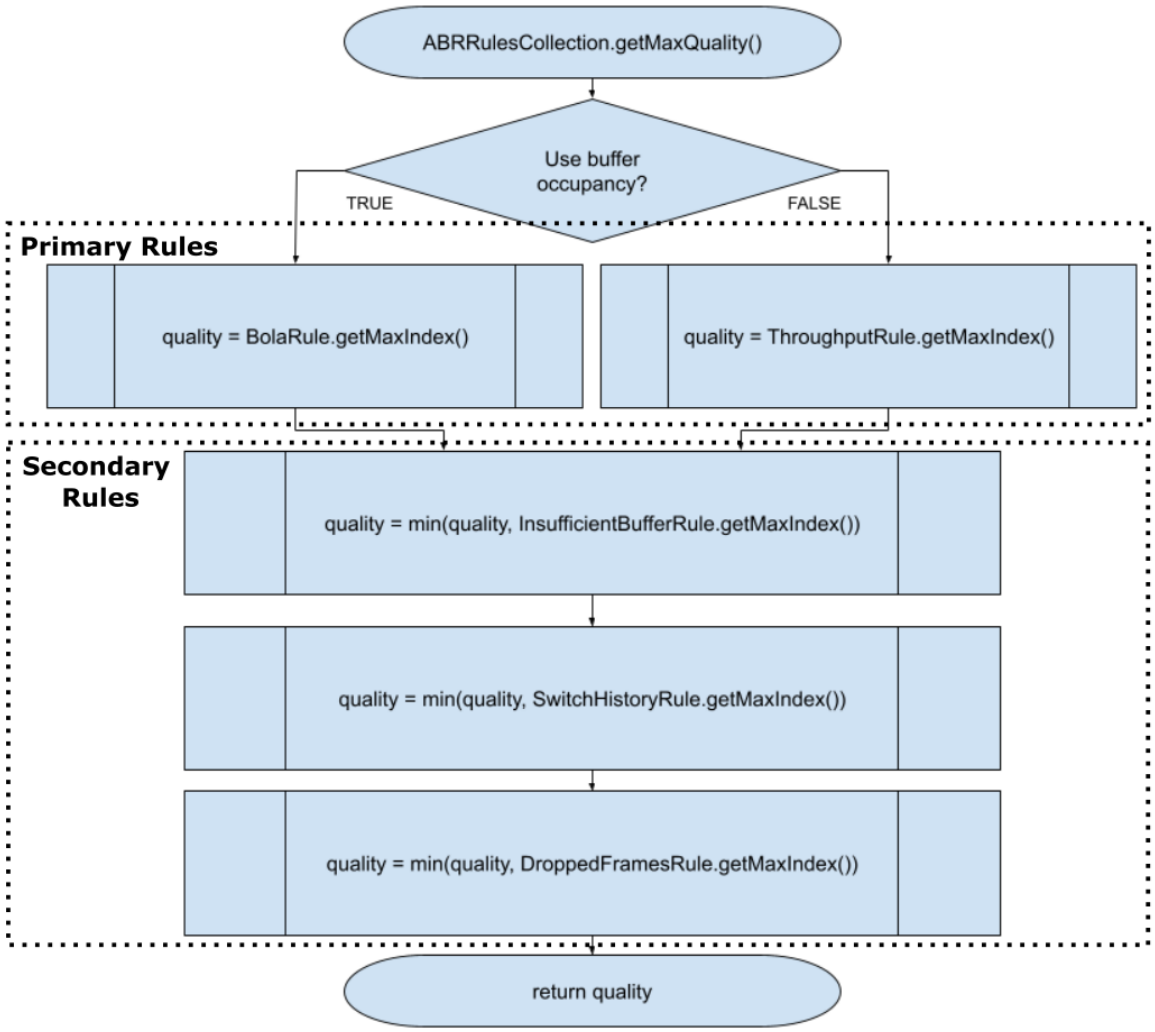

Figure 4.5. Simplified diagram outlining the ABR management of the *default rule* implemented.
Source: Adapted from (DASH.JS, 2023)

serve as the main determining factor for the majority of the video playback. They play a crucial role in the decision-making process. On the other hand, the secondary rules are designed to handle specific cases and aim to enhance the overall user experience.

The primary rules consist of two components: the throughput rule and the BOLA rule. The standard strategy involves dynamically switching between these two rules, leveraging their respective strengths. The throughput rule utilizes the historical throughput data to estimate future conditions. It applies a safety factor of 90% and selects the highest quality available within this safe throughput range. Additionally, it incorporates a *rich buffer threshold* when it selects the highest quality available to ensure that a lower quality is not chosen until the buffer level drops below this threshold. On the other hand, the BOLA rule employs the BOLA

algorithm, as described in (SPITERI *et al.*, 2020), to determine the optimal quality based on the current buffer level. The BOLA rule takes the buffer occupancy into account when making quality selection decisions.

The secondary rules encompass the following aspects: the insufficient buffer rule, switch history rule, and dropped frame rule. The insufficient buffer rule ensures that when a rebuffering event happens, the quality selection is limited to the lowest available option to prevent it to happen again. The switch history rule forces that a specific quality level be maintained for a minimum of eight segments when another rule drops the current quality, therefore, preventing frequent oscillations. The dropped frame rule addresses CPU limitations by monitoring the video's rendering capability. If the dropped frames exceed 15%, this rule prevents the selection of that specific quality level for a certain duration, allowing the player to recover and optimize performance.

On the 360EAVP platform all the custom ABR algorithms are refereed as rules. This comes from the fact that, with the exception of the *dash.js* default rule, all algorithms rely on their custom rule to determine the quality selection. Therefore, from now on, we will refer to the ABR algorithms implemented in the platform as *rules*, aligning with the nomenclature used within the platform.

It is important to note that there is one ABR algorithm associated with each video instance, and each video instance is associated with a face of the 360-degree video. Therefore, each ABR algorithm will select a quality independently of the other ABR algorithms (instances) associated to the other faces. This means that the faces are not aware of the overall context of the entire 360-degree video. Hence, they only consider their respective playback. This principle also applies to the default rule, treating each face as an independent video instance.

However, when it comes to the 360-degree video playing on the platform, we aim for decisions to be made while considering the entire video playback, rather than just its individual faces. To address this challenge, we establish a connection between the video scope and each instance of the ABR algorithm, enabling them to have a comprehensive view of the entire video playback in order to allow informed quality decisions. The scope contains the most important tools and information about the whole video, such as:

- Current playback time;
- Viewport prediction module;
- Visible faces identification module;
- List of editions on the video;
- Total estimated bandwidth.

By incorporating the custom rules of *dash.js* with the video scope into 360EAVP, we provide the capability to analyze different approaches. Additionally, users have the option to utilize the default rule provided by *dash.js*. This default rule, as mentioned above, is highly complex and serves as an invaluable reference for future ABR algorithms development on the platform. It is worth noting that when using the default rule, each ABR algorithm associated with a face independently selects the quality level for their face (instance). The platform already includes several custom ABR algorithms as examples to showcase the range of possibilities.

Some of them are simple enough to be used as ground-truth, such as the Highest/Lowest birate Rule, which, as the name suggests, selects the highest/lowest quality available for the video regardless of network conditions. Another ABR algorithm already implemented, named Field of View Rule, selects the highest quality available only for the faces that are in the user's viewport. The faces that are not visible will have the lowest quality available. This rule was created to demonstrate how we can use the visible face identification module in the design of ABR algorithms. It does not use any viewport prediction.

To study the impact of video editing on the user's experience and in the video streaming, we have implemented an ABR algorithm called *Field of View Edit Rule* that makes use of the editing information to make better quality selections. However, the performance evaluation of this algorithm was not conducted in this work, and is listed as future work. In the following section, we describe the the *Edition-aware* ABR algorithm implemented on the platform to enable the use of prior knowledge of the video editions for a better quality selection.

4.6.1 Edition-Aware ABR algorithm

The proposed ABR algorithm is divided into two parts. Algorithm 1 expresses every step in detail. The algorithm is called every time a new segment has to be downloaded and it is implemented inside the function `getMaxIndex()`. Therefore, it returns the maximum index corresponding to the most suitable quality for the given algorithm criteria. The algorithm takes as input the DASH playback information and the *scope*. The output of the algorithm is the selected quality index Q_f with its corresponding priority P_f. The algorithm starts by initializing several variables and retrieving the list of editions from the scope. It also retrieves the lowest available quality index Q_{min} and defines a list of available qualities Q_{LIST}.

The first part uses the function `computeAverageThroughput()` to get the average throughput TP_{AVG} from the last six requests, then it gets the highest quality index Q from the list of qualities Q_{LIST} available that best fits the calculated throughput.

The throughput calculation is based on the information provided by DASH, and it consists of computing the number of successfully downloaded bytes, indicated by b, during the total time period T_{tot} observed from the start of the request to completion of the download, i.e., the total time T_{tot} is computed as

$$T_{tot} = T_{fin} - T_{req}, \tag{4.4}$$

where T_{fin} indicates the time instant the download finishes, and T_{req} is the time instant the request is sent to the server. The number of bytes downloaded during T_{tot} may not accurately reflect the number of bytes in the requested segment due to network instabilities. To account for fluctuations and inaccuracies in the throughput measurement, we have implemented a "safety factor" in the `getMaxQualityIndex()` function. To select a video quality higher than the current one (i.e., video bitrate), the measured average throughput must be 50% higher than the bitrate value, so that the quality increase is implemented. This is to ensure a more reliable quality selection process and avoid constant quality changes.

Following Algorithm 1, the next part of it gets the current segment number S_C to be downloaded using the function `getCurrentSegmentNumber()`, and the segment number being played S_{Play} using the function `getSegmentBeingPlayed()`.

The algorithm then calculates the predict window V_{pw} based on the difference between S_C

Algorithm 1: Edition-Aware ABR Algorithm

Input : The *scope*, DASH playback information
Output: Quality Q_f selected with priority P_f

Initialization
  Retrieve list of *editions* from *scope*;
  Retrieve the lowest quality index Q_{min} available;
  List of qualities available $Q_{LIST} = [q_0, q_1, \ldots, q_k]$;
  $nextEditionIndex = 0$;
end

`/* Compute the average throughput of the last 6 requests */`
$TP_{AVG} = computeAverageThroughput()$;
`/* Get the maximum quality index possible for the current throughput estimation */`
$Q = getMaxQualityIndex(Q_{LIST}, TP_{AVG})$
`/* Get the current segment number to be downloaded */`
$S_C = getCurrentSegmentNumber()$;
`/* Get the segment number being played */`
$S_{Play} = getSegmentBeingPlayed()$;
$V_{pw} = S_C - S_{Play}$;
if $V_{pw} \geq T_{pw}$ **then**
  `/* Ridge Regression */`
  $VP =$ `predictWithRR(`$V_{pw} \times frameRate$`)`
else
  `/* Linear Regression */`
  $VP =$ `predictWithLR(`$V_{pw} \times frameRate$`)`
end
`/* Get the next edition segment */`
$S_E = getEditionSegment(editions[nextEditionIndex] \rightarrow frame)$;
if $S_C = S_E$ **then**
  $VP = editions[nextEditionIndex] \rightarrow RoI$;
end
$F = getFaceIdFromInfo()$;
`/* Check if the face F is on the user's viewport */`
if $isFaceVisibleOnVP(F, VP)$ **then**
  if $F \rightarrow visibility < 12\%$ **then**
    $Q_f = Q - 1$;
  else
    $Q_f = Q$;
  end
  $P_f = strong$;
else
  $Q_f = Q_{min}$;
  $P_f = weak$;
end
$nextEditionIndex = nextEditionIndex + 1$;
return $[Q_f, P_f]$

and S_{Play}. If the predict window is greater than or equal to a threshold T_{pw}, the algorithm performs the Ridge Regression prediction using the function `predictWithRR()`. Otherwise, it uses the Linear Regression prediction using the function `predictWithLR()`. The result of the predict viewport, regardless of the regression technique used, will be the viewport position VP.

However, if the current segment number being downloaded S_C matches the segment number on the upcoming edition S_E, the algorithm adjusts the VP to the RoI of the next edition. This adjustment is based on the assumption that the user will be looking at the next RoI either because he/she is already looking to that region or because a content realignment will be triggered by the dynamic edition module.

The algorithm then retrieves the next edition segment S_E using the function `getEditionSegment()`. Within this function, the approach involves examining the segment of the video to which the next scheduled edition frame is part of. This is possible by using information about the segment size from DASH playback details and the edition frame found in the list of editions.

It then obtains from the function `getFaceIdFromInfo()` the current face F having the segment downloaded. This information comes from the player *scope* and it will be used further to check if the current face F is visible on the user's viewport using the function `isFaceVisibleOnVP()`. If it is visible, it checks the visibility of the face using the Visible Face Identification Module described on Section 4.4. If the visibility is less than 12%, the algorithm decreases the quality index by 1. Otherwise, it keeps the selected quality index the same. We chose the value of 12% by taking into account that, when using the 25 mapped points, each point corresponds to 4% of the matrix. Therefore, 12% corresponds to 3 visible points. The quality of faces that are completely outside the user's viewport is set to the minimum quality Q_{min}.

The priority ranking system is used by the player to efficiently handle parallel requests from multiple faces to the server. Since we are handling nearly six requests simultaneously, it is crucial to know which among them should have the preference. In the proposed algorithm, the priority P_f is set as "strong" for visible faces and "weak" otherwise.

Finally, the algorithm updates the *nextEditionIndex* variable, so that it points to the next edition scheduled. The return of it will be the selected quality level Q_f and priority P_f as the

output.

As point of attention, although the viewport predictor can sometimes produce suboptimal video quality, we believe that the proposed ABR algorithm is based on sound assumptions, particularly due to the effective use of the cube mapping projection. In the worst case that we encountered, only 2 of the 4 visible faces were less than 12% visible. Therefore, for the user to observe the two faces with the worst available quality, the viewport predictor would have to nearly make a 180° mistake, which is unlikely. However, it is important to note that if subtiles are used (other than the cube faces), the logic may not be robust enough to maintain a consistent high-quality user experience. Even a small error in viewport prediction could cause the user to see a face with the worst video quality. Therefore, more studies are needed to compare the proposed ABR algorithm with other methods and perform subjective experiments to determine whether the model accurately reflects user feedback.

4.7 VIDEO PLAYBACK DATA COLLECTION MODULE

The last module tracks the playback information and generate a `csv` format file as output. In the output file, each line refers to a frame. The file is updated each render loop of the A-frame framework. The function named `tick` is responsible to do this additional actions, i.e., actions that are not in the main render loop process and it is triggered about 60 to 120 times per second. Because the sample size is greater than the number of frames per second of the video, we can find duplicate frame information in the output file. Although this is not ideal, the duplicate information can be handled in the post-processing.

The captured information can be changed depending on the need. However, in the default version, the captured information includes the frame number, throughput, head position (yaw and pitch), a list of face qualities, a list of visible faces, a list of percentages indicating the visibility of each face, any scheduled edits for the frames, occurrences of dynamic edits, the degrees rotated (in radians) for each edition, the type of scheduled edition (fade-rotation and snap-change), the number of rebuffering events that happened, and the total stall time in seconds. Next, we describe how each piece of information is gathered during video playback. The scope, as described before, is our main variable, and is responsible to share most of the

videos' metadata.

- Frame number: A simple framework called *VideoFrame* calculates the current frame from the current video time;
- Throughput: Calculated as the average value of the past 6 requests to the server. The requests can be from any face;
- Head position (yaw and pitch): Gathered from the Ray casting technique;
- List of face qualities: Gathered from each face video instance object;
- List of visible faces and its percentage of visibility: Gathered from the Visible Face Identification Module
- All editing information: Gathered from the JSON file and the Dynamic Edit Module: editing scheduled, rotation, and if editing was triggered;
- Number of rebuffering events: Count of the number of times the "Buffer Empty Event" triggers for a face. Counting covers the whole video, not for each face.
- Stall time: Amount of time the video stalled. Timer starts when the video pauses due to a "Buffer Empty Event." Timer stops when the video starts to play again.

4.8 360EAVP INTERFACE AND FRAMEWORKS

The 360EAVP is a solution that uses essentially three frameworks. The *AngularJS* is the framework responsible for handling the user interface and the data storage in the video reproduction (ANGULARJS, 2023). All video playback information is shared within the playback scope, which is an AngularJS object that can propagate events and watch expressions. In our project, it is used to centralize information about playbacks and retrieves information about scene objects and throughput. Furthermore, the scope serves as an excellent tool for sharing functions that are used by every video instance, including the one used to identify every visible face in the current frame.

To render the 3D environment and utilize editing techniques, 360EAVP employs the *A-Frame* framework (A-FRAME, 2023). The A-Frame is a so-called entity component system framework for the well-known Three.js framework (THREE.JS, 2023). Hence, it is behind the scenes so much that we make directly use of Three.js by calling its native functions. Each video is rendered on a face that is positioned on the right spot on the 3D environment initialization. The editing techniques are applied by rotating each face of the cube and the fade effect is deployed by changing the opacity of an invisible black sphere that encloses the user. This invisible sphere is also used to track the user's head movement using a technique called Ray casting.

Finally, 360EAVP uses the *dash.js* framework to enable adaptive streaming via DASH. Since one video instance is used for each face/tile, an instance of an ABR algorithm is required for each face. Additionally, the framework was slightly modified to allow synchronous playback of each face/tile. Dash.js includes a functionality known as *catchup*, which is designed to synchronize the playback of different videos. However, in the current version of the framework, this feature is limited to live streaming content. As a result, an optimization was required to enable the use of the *catchup* mechanism for on-demand and recorded videos. This optimization involves using the current playback time, shared within the scope, instead of relying on individual time measurements, to calculate the existing latency. By adopting this approach, all videos can maintain the same latency, ensuring synchronization of all faces. During the video initialization, each face/tile is identified and rendered on a specific face of the cube. This location is specified on the JSON file retrieved by the preparation module, which will be discussed further.

The 360EAVP Interface is depicted in Figure 4.6. It closely resembles the interface of the *dash.js* reference player[1] as it was developed based on that interface. To focus on what is important for this application, many resources were removed from the DASH reference player.

[1] <https://reference.dashif.org/dash.js/nightly/samples/dash-if-reference-player/index.html>

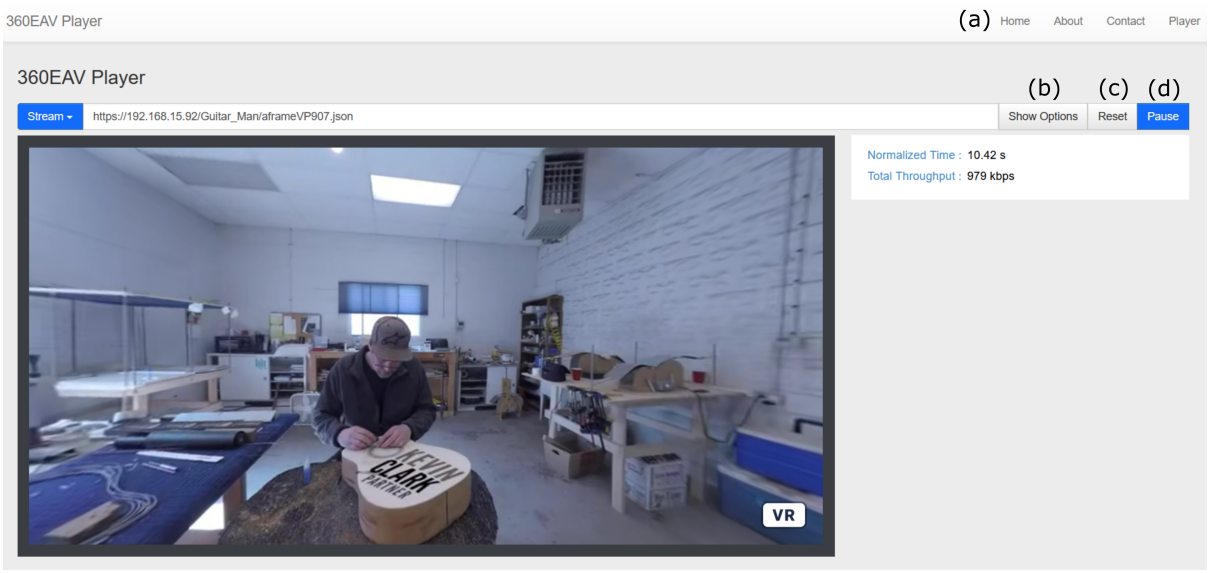

Figure 4.6. *360EAVP* Interface: (a) header with basics application information, (b) ABR algorithm selection, (c) Reset button to refresh the page and (d) Play/Pause Button after initialization

CHAPTER 5

PLATFORM EVALUATION

In this chapter, we present and discuss the key findings of a subjective experiment conducted using the 360EAVP platform. The objective was to examine the impact of dynamic editing on the user's QoE and feeling of immersion on edited videos, specifically those featuring various scenes cuts that contribute to storytelling. Moreover, in order to understand if the use of dynamic editing techniques can improve the overall viewport prediction and/or quality selection, we want to analyze how it can impact the user's head movement. These experiments serve as proof of concept to exemplify and demonstrate how the platform can be utilized for deeper analyses of editing techniques or other approaches implemented with the platform. It is worth mentioning that these experiments were not created to evaluate the impact of editing information on the streaming side. As a result, none of the custom rules were applied, and the player would always select the best quality available.

5.1 SUBJECTIVE EXPERIMENT DESIGN

This experiment aims to study how the use of dynamic editing techniques can impact the user's overall Quality of Experience (QoE) and comfort compared to the original video with static editing. Additionally, we want to examine the effects of editing techniques on the user's head movement. The goal is to determine whether the use of editing techniques can reduce head movement and make it more predictable, enabling users to focus their attention on the presented scenes. This can potentially enhance viewport prediction accuracy, allowing Adaptive Bit Rate (ABR) algorithms to make better quality decisions for tile-based video streaming. We are specifically focusing on studying two dynamic editing techniques: snap-change and fade-rotation. Both techniques involve triggering a realignment of the content, although they employ differently. We define realignment of the content as the rotation of the content to ensure that the next RoI is visible on the user's viewport. By comparing their results, we aim to understand if

there are differences in the user's gaze fixation when using these different editing techniques.To capture the user's gaze, we utilize the technique called Ray casting, as mentioned in Section 4.8.

The main questions that we want to answer are:

- Can the use of dynamic editing techniques improve the user's QoE?
- How does the dynamic editing techniques affect the user's comfort?
- Does the realignment of the content reduce the user's head movement?
- How is the user's head movement affected with different editing techniques?

To answer the questions raised above, we had to describe what we understand as comfort and Quality of Experience in the experiment.

- **Comfort** - Any feeling of sickness or physical discomfort.
- **Quality of Experience** - It is the degree of satisfaction experienced by a user of an application or service. It arises from the fulfillment or non-fulfillment of the user's expectations regarding what they anticipated from the application or service, taking into account the user's personality and current state (BRUNNSTRÖM *et al.*, 2013). More specific, we asked the subjects to evaluate their level of immersion and the ease of following the story.

During the experiment, the subjects went through 4 *random* sessions to evaluate their Quality of Experience (QoE) and comfort while watching two different videos, each with two different dynamic editing techniques: snap-change and fade rotation. we have implemented the Double Stimulus Continuous Quality Scale (DSCQS) following (UNION, 2002) to evaluate the comfort and the QoE. This subjective assessment methodology was originally designed to evaluate the quality of television pictures. However, (GUTIERREZ *et al.*, 2021) demonstrates its validity for subjective tests with 360-degree videos.

The idea is to display and evaluate the techniques in pairs (from the same source content). The subjects rate both videos at the same time using a continuous five-points scale that will be

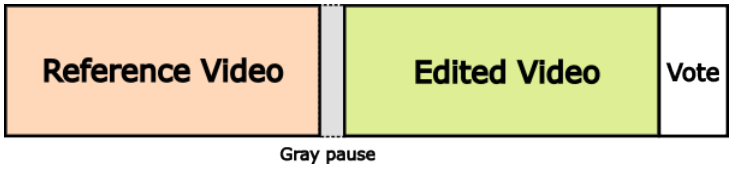

Figure 5.1. Experiment session flow.

presented further below. By collecting the data like this, we aim to have information to do the Difference Mean Opinion Score (DMOS) from the reference video, i.e., static editing (presenting the video as filmed by the filmmakers), to the dynamic editing technique implemented.

The session flow is illustrated in Figure 5.1. On each session, the subjects watched a reference video (static editing video). Then, they watched the same video again with a dynamic editing technique implemented. After watching both versions, the subjects had to answer the following questions for each video:

- On a continuous scale from 1 to 5, please evaluate your comfort level.
- On a continuous scale from 1 to 5, please evaluate your quality of experience;

The Comfort followed the scale below:

- 5 - No problem - No perceptible effect, natural feeling
- 4 - Light Effects - Slight discomfort, but not sickness
- 3 - Uncomfortable- Moderate discomfort, but tolerable for a while
- 2 - Unpleasant - Strong discomfort or sickness, but can continue the test
- 1 - Unbearable - Strong discomfort or sickness, and want to stop the test

The Quality of Experience followed the scale: (5) Excellent, (4) Good, (3) Fair, (2) Poor and (1) Bad.

After all sessions finished the subject was invited to answer some general questions about the experiment and the videos. The questions were the following:

- Did you enjoy being guided to different regions of the video?

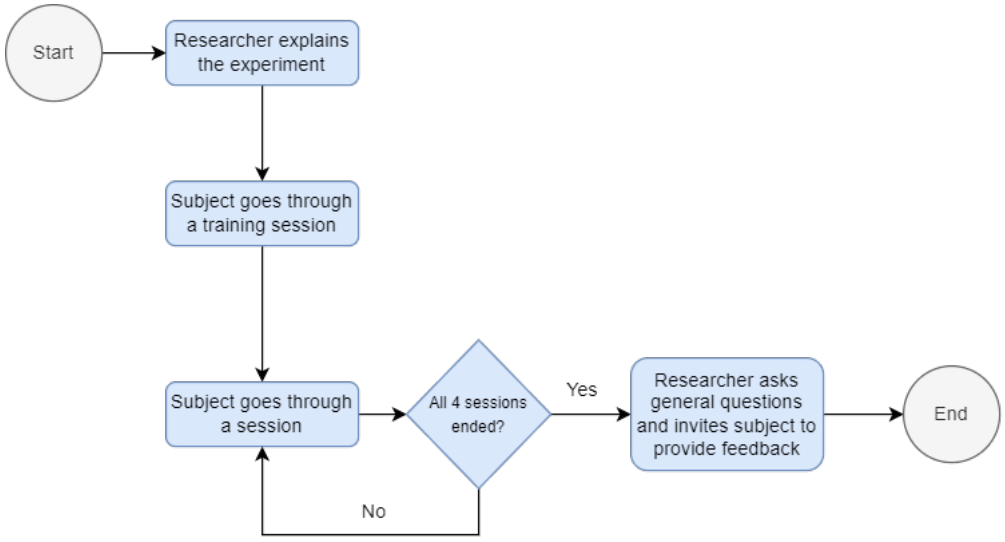

Figure 5.2. Experiment flowchart: Firstly, an introduction is provided to explain the purpose and scope of the experiment. Subsequently, each subject goes through a training session. Once the training is completed, the experiment session begins. After all sessions are concluded, some general questions are asked and the subject is given the opportunity to provide feedback.

- Were you able to perceive the moments when the edits occurred?
- Were there any elements/objects in the videos that you only noticed when you were guided to them?
- How much did the faces "artifacts" of the cube bother you in terms of overall quality of experience and comfort?

The last question was posed because, prior to the start of the experiment, there were noticeable "artifacts" on the vertices and edges of the cube's faces when viewing the video through an HMD. This had the potential to reduce the overall immersive experience, thereby potentially impacting the experiment scores. Hence, we sought to understand how this aspect affected the user's experience. The subject comments will be discussed on Section 5.5.

5.2 EXPERIMENT METHODOLOGY

The Figure 5.2 illustrates the experiment flowchart to provide clarity on each step of the experiment. Firstly, we presented an introduction to the experiment, explaining what will be studied, showcasing all editing techniques, demonstrating the platform interface, and presenting the concept of comfort and QoE as mentioned before.

Figure 5.3. Photography of a subject performing the experiment.

Next, the subjects went through a training session where they experienced, for the first time with an HMD , the three editing techniques under study: static editing, snap-change, and fade-rotation. During the training session, the subjects watched the same scene 3 times, each time with a different editing technique. To restart the scene, the content would fade out to start again with a different editing technique. This preparation helped the subjects become familiar with the platform interface and the steps they needed to follow when starting a video playback. Once the training is completed, the experiment session begins. The order of the sessions was *randomized*. After all sessions are concluded, the general questions listed on the Section 5.1 are asked and the subject is given the opportunity to provide feedback.

We conducted the experiment with 15 subjects ranging in age from 12 to 56 (1 subject aged 12, 8 subjects aged between 20 and 30, 3 subjects aged between 30 and 40, 3 subjects aged over 50). 46% of the subjects already had some familiarity with immersive experiences in VR. We are aware that the number of subjects in our study may be limited to have conclusive results. However, we recognize that with a small sample size, only substantial effects can be detected with sufficient statistical meaning. Therefore, even with a small sample, the impact of editing techniques on the user's comfort and QoE may still be observed. Moreover, it can be used as a proof of concept on how the editing techniques can impact the user's QoE and what the 360EAVP platform can offer to the research community. For future work, we plan to carry out experiments with more subjects. Figure 5.3 shows a photography of a subject performing the experiment. Every subject participated in the experiment while seated on a chair that could rotate 360-degrees, allowing them to easily view the content all around them.

The videos were streamed wirelessly locally from a PC on the same network. We utilized Wi-Fi 5 (802.11ac) with a 5GHz network bandwidth. The network was private, with only the PC server and the HMD connected to it. The 360EAVP would always request the best quality possible from the server and playback data collection is possible by enabling the video playback data collection module from the 360EAVP platform.

The experiment itself was carried out on an HMD (Meta Quest 2 (META, 2023)) using its native web browser, using Chromium version 91, and took approximately 50 minutes to be done (30 minutes for the sessions and 20 minutes for the presentation and training session), and consisted of a total of 4 sessions (2 different editing techniques to each of the 2 video contents). Each session followed the flow introduced by Figure 5.1, where the subject first watched the reference video, which was always the version with static editing, i.e., the original version created by filmmakers. After that, there was a brief break called the "Gray pause", followed by re-watching the video with a dynamic editing technique implemented. For instance, the subjects would watch the vaude video with static editing first. After a brief pause, they would then watch the vaude video again, but this time with the snap-change editing technique applied. The subject was not informed about the difference between the reference video and the edited video. Each session lasted approximately 8 minutes, resulting in a total duration of 32 minutes for the 4 sessions. After each session, the subject was asked to answer some questions verbally.

5.3 VIDEO CONTENT SELECTION

The two videos used in this experiment, titled *vaude* [1] and *bank*[2], are promotional/advertising videos. Figure 5.4 and Figure 5.5 show the video cuts that appear in the videos “vaude”, and “bank”. On both Figures, the scenes presented are the RoIs that the user have to watch in order to follow the video message. In the “vaude” video, the first scene merely depicts an outdoor scene that is not necessarily essential to the video’s storytelling; it serves to situate the viewer in the world. The second scene introduces the main character and the bag factory. The subsequent scene shows the main character with her friends using the bags in a square.

[1] https://www.youtube.com/watch?v=c9NpT88d22Q
[2] https://www.youtube.com/watch?v=sZVxsYA9HIQ

Figure 5.4. Sample video frames of "vaude" video presented at each scene cut. There is a total of 9 scene cuts. Each frame shows the first RoI of the scene.

Figure 5.5. Sample video frames of "bank" video presented at each scene cut. There is a total of 6 scene cuts. Each frame shows the first RoI of the scene.

Afterward, the video returns to the factory, where the main character explains that their bags are "PVC-free". Then, in a mountain biking scene, the bikers are seen using the mentioned bag. The following scene features the main character interacting with the user. Returning to the mountain biking segment, the characters observe a man wearing a panda costume holding a sign that says "Reduce PVC". In the next scene, back at the factory, the main character presents a new bag. Then, in a city scene, the characters are once again seen using the bags on their bicycles. Finally, in the last scene, the main character addresses the user once more before departing.

In the bank video, the first scene presents the character who will guide us through the story, introducing her family members. The next scene shows one of the characters using a bank feature to perform a money transaction. In another scene, another character uses a personal assistant to help furnish her house. Afterwards, the next scene shows another character enjoying time with her kids while their money is being well invested on the bank platform. Then, there

Table 5.1. Videos content and information

Content	Duration	N° scenes	Scenes per min	Resolution
vaude	2 min 25 sec	9	3.75	4096×2048
bank	3 min 58 sec	6	1.5	4096×1024

is a brief scene that does not entirely contribute to the video storytelling. Next, the main character appears in a meeting room, concluding a conversation. Finally, the main character is seen skydiving.

As one can observe, they have very different content with different narratives paces, that is, a fast-paced and a slow-paced video. In this context, we selected the "vaude" video as our fast-paced video because it features frequent changes between indoor and outdoor scenes, as well as different action scenes of mountain biking. For the slow-paced video, we chose the "bank" video because it has more indoor scenes than outdoor scenes, and most of the scenes are static with minimal action. Table 5.1 summarizes the information about each videos. Every scene cut is almost completely made of instant cuts, which is really similar to the snap-change technique. As one can notice, the "vaude" video has more than double the number of scenes per minute compared to the bank video. This indicates that, in the "bank" video, subjects have more time to explore the environment before the next scene cut, in comparison to the "vaude" video.

They were first downloaded in the Equirectangular Projection format, as it was the available format. Therefore, to make them compatible with the 360EAVP platform, they had to go through a preprocessing, following the steps described in Section 4.1. This involved converting them into the Cube Map Projection, and then tiling each face into different video instances and DASH Manifests.

Both videos were played without audio and had a frame rate of 30 frames per second (fps). We decided to omit the audio because it was originally in German, and none of the subjects would understand it. Additionally, we aimed to minimize variables and focus only on the impact of visual content on the user's comfort and Quality of Experience. However, for future studies, an examination of the audio's impact should be conducted.

5.4 EXPERIMENT RESULTS

Before going deeper into the questions raised above, let us give some general statistical of the use of dynamic editing techniques to clarify some of the results found.

As mentioned in Section 4.3, dynamic editing is only triggered if the difference between the user's current center of the viewport and the next region of interest exceeds 30 degrees. It is important to notice that, in the experiment, every RoI was manually selected to perform the dynamic editing. However, this selection followed the regions that were pre-selected by the filmmakers during the realignment process when creating the video. Then, if the difference between the user's current center of the viewport and the next RoI does not exceed 30 degrees, the realignment is not unnecessary because the user is already looking close enough. In those cases, the need for no dynamic editing comes from the fact that the content itself guides the user to a specific area of interest. Additionally, it is worth noting that the subjects always watched the dynamic editing version of the video on the second viewing. This means that they already had prior knowledge of the content of the video, and therefore, they might have wanted to focus on different regions instead. In this context, it is not only important to track how many times dynamic editing was triggered during their viewing, but also to consider how many RoIs they missed on the first time that they watched the content.

To remind the reader, both videos were watched 4 times in 2 session each. On each session, the subject had to watch the static editing as their reference video. Therefore, there were 2 times that the reference video had to be watched. On the first one, the subject did not have any prior knowledge of the video content. On the second time, he/she already had some information of the video content. We want to get the average number of missing RoIs in the first time that they watched the content because everything was new for them. This missing RoIs would actually be realigned if the dynamic editing was enabled. Then, we want to compare the result with the average number of times that the dynamic editing was triggered for each technique to understand if the prior knowledge of the video content can impact the number of content realignments. This information can provide valuable insights.

Figure 5.6 illustrates the normalized number of missing RoIs during the first viewing of the video and the normalized realignments made by each dynamic editing technique on each video.

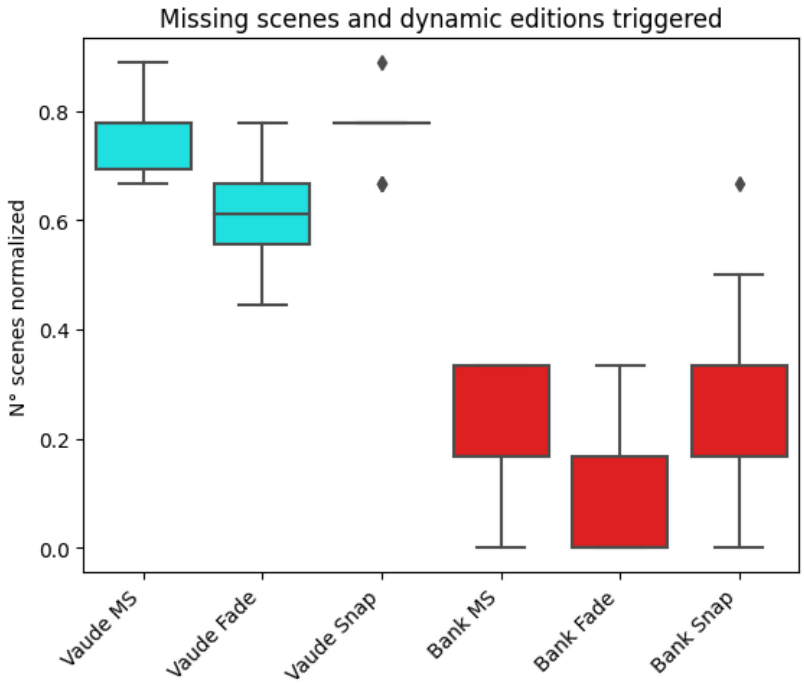

Figure 5.6. Comparison of the normalized number of Missing Scenes (MS) when watching the video for the first time and the normalized number of scenes realigned by a dynamic editing (Snap-change and Fade-rotation).

It is interesting to observe that the number of missing RoIs when the subjects watched the video for the first time is quite similar to the number of dynamic editing triggered when the subject had some information about the video content. This might suggests that the subjects were following a similar trajectory.

.Importantly, this implies that prior knowledge of the content may not be of significant importance for studying scene transitions.

Another notable observation is the discrepancy in the number of missing RoIs/dynamic editing triggers between the "vaude" and 'bank" videos. Approximately 70% of the scene cuts in the "vaude" video required realignment, whereas only 20% of the scene cuts had to be realigned in the 'bank" video. This can be attributed to the inherent characteristics of the content itself. As mentioned previously, the "vaude" video is a fast-paced content with more action scenes, while the "bank" video is a slow-paced video with fewer actions and regions of interest within each scene. Consequently, it is easier to capture and maintain the user's attention on a single element when watching a slow-paced video.

One can also observe that the fade-rotation technique resulted in a lower number of realignments. However, there does not appear to be a clear explanation for this finding. After a

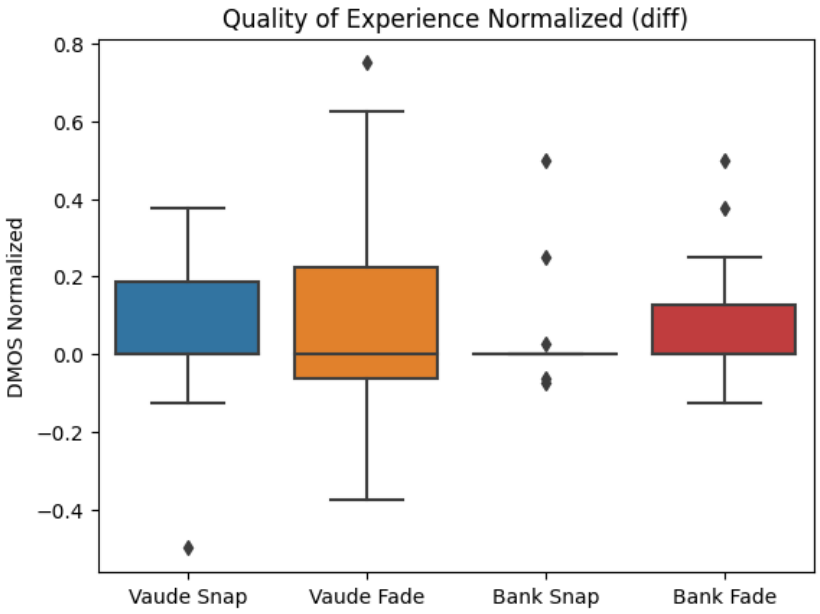

Figure 5.7. Difference Mean Opinion Score (DMOS) normalized between QoE values given by subjects considering the reference video (static editing) and the corresponding video with dynamic editing (snap-change or fade-rotation). The labels in the X axis indicate the video type and the respective dynamic editing applied.

content realignment, the users has the freedom to look in any direction they choose. Since the result represents an average number, it encompasses subjects with varying exploratory behaviors. Consequently, while some subjects may not have observed any dynamic editing, others may have experienced a greater number of realignments.

These preliminary results will help us understand the next findings. Let us then analyze the first question: can the use of dynamic editing techniques improve the user's QoE?

Figure 5.7 shows a box-plot graph of the results of the DMOS normalized from the dynamic edition video compared with the reference video (static video). We can see that the two videos have different scores. This shows that the video content indeed was different enough to have statistical differences. The first thing that is highlighted is how the Bank video with Snap-change had almost the same QoE for most part of the subjects. This can be explained by 2 factors: the first is that, being our slow-paced video, we had less changes when compared with the vaude video, therefore, it was less noticeable the differences from the reference video to the edited video. The other thing is that the snap-change, being an instant cut technique from one scene to another, ends up blending in as a regular video cut and goes unnoticeable. On the vaude video, the user might notice more times the use of snap-change technique because they might have different regions of interest on the same scene and they were realigned to a

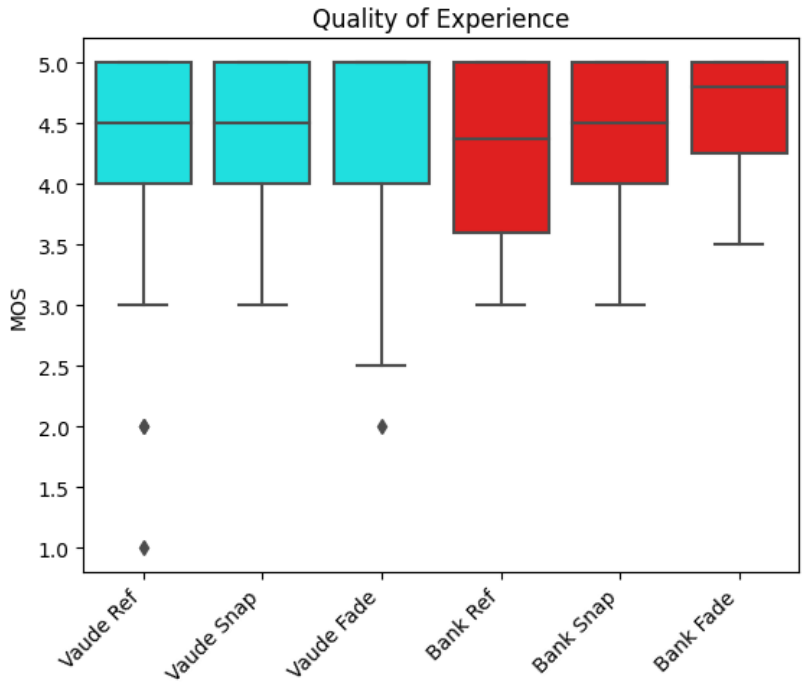

Figure 5.8. Mean Opinion Scores (MOS) for the QoE of different videos, editing technique (Snap-change or Fade-Rotation), and the reference video (Static editing).

different one. Overall, we can see that the use of dynamic editing increased or maintained the same level of QoE for the majority of subjects. One thing to notice is that the vaude video with the fade-rotation technique was the one with the most variable scores. In this sense, while not being the majority, a significant number of subjects preferred the reference video over the fade-rotation technique video.

Figure 5.8 illustrates the Mean Opinion Scores (MOS) for the QoE for each reference video and dynamic editing technique. We can observe that, in most scenarios, the dynamic editing technique either maintained or reduced the variability of the scores compared to the static editing technique. This suggests that the use of dynamic editing techniques has the potential to enhance the overall QoE for the users.

In general, we can answer that the use of dynamic editing techniques can indeed improve the user's QoE. We see a more significant improvement for videos with more action, but there is a drawback that it can actually reduce the QoE for this type of content for some users.

The next question to be explored is: how does the dynamic editing techniques affect the user's comfort?

To address this question, we present Figure 5.9, which illustrates the results of the DMOS

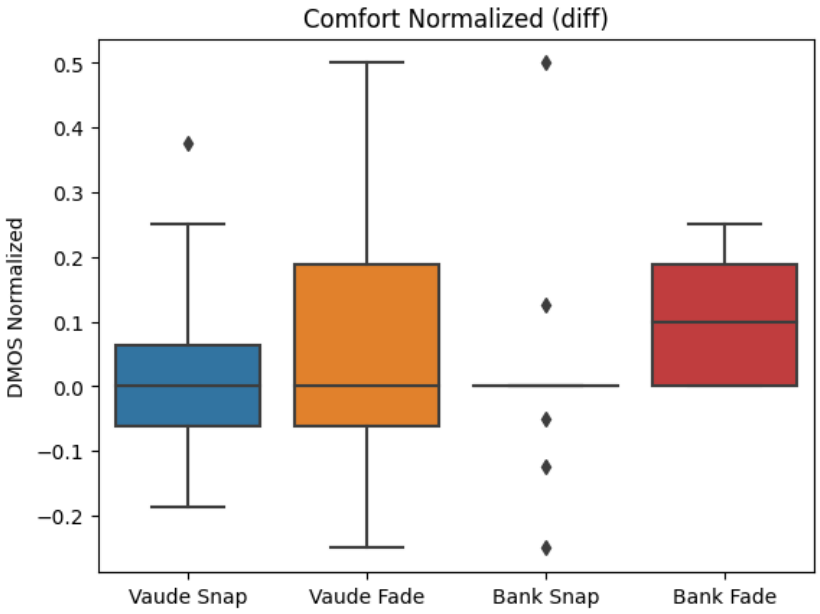

Figure 5.9. Difference Mean Opinion Score (DMOS) normalized between comfort values given by subjects considering the reference video (static editing) and the corresponding video with dynamic editing (snap-change or fade-rotation). The labels in the X axis indicate the video type and the respective dynamic editing applied.

normalized for comfort between the dynamic edited video and the reference video (static video). We observe that the results are similar to those from the QoE. However, one noteworthy aspect is that every subject reported the same or higher level of comfort when watching the bank video with fade-rotation. Some subjects described that they felt more comfortable watching the video with this technique as it allowed them to anticipate scene transitions when the fade-rotation occurred. In the case of the other videos and editing techniques, we observe a similar trend in the QoE. However, for the vaude video with fade-rotation, there were varying perspectives, although a higher number of subjects expressed a preference for this technique.

Figure 5.10 illustrates the Mean Opinion Scores (MOS) for the comfort for each video reference and dynamic editing technique. We can observe that, for every dynamic editing technique, we either maintained or reduced the variability of the scores compared to the static editing technique. This suggests that the use of dynamic editing techniques has the potential to increase the overall comfort of the users. However, it is important to acknowledge that there are additional variables that need to be taken into account to draw more conclusive results. For instance, since subjects always watch the dynamic editing version as the second video of a session, they might have experienced a lower sense of discomfort because they had already watched the content at least once when viewing the video with dynamic editing. Nonetheless,

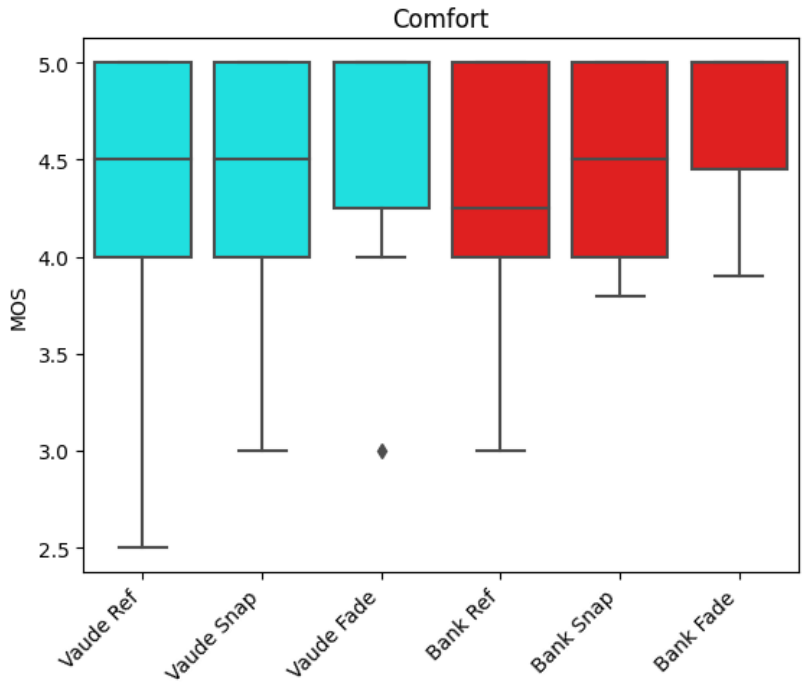

Figure 5.10. Mean Opinion Scores (MOS) for the Comfort of different videos, editing technique (Snap-change or Fade-Rotation), and the reference video (Static editing).

one key takeaway from analyzing Figure 5.10 is that, the lowest comfort scores for the reference videos were 2.5 and 3 for the "vaude" and "bank", respectively. On the other hand, every dynamic editing technique had higher scores, indicating an improvement in overall comfort. This provide a positive indication in response to the raised question.

The next two questions can be addressed by analyzing the subjects' head movements while watching the same video with different editing techniques. It is important to note that positive results in reducing the subjects' head movements can contribute to improved performance of Adaptive Bitrate (ABR) algorithms that utilize this prior knowledge, as described in Section 4.6.

- Does the realignment of the content reduce the user's head movement?
- How is the user's head movement affected with different editing techniques?

Figure 5.11 displays the head movement trace of a subject for each editing technique in the vaude video. The Y-axis represents the normalized yaw values, which correspond to horizontal head movements, which is the direction in which changes in scene content are more noticeable. As the subjects had to watch the reference video twice during the evaluation of the editing

techniques, we chose to analyze the subject's first experience of watching the video when analyzing their head movement for the reference video. This was done to ensure that there was no prior knowledge of the content on the reference head movement graph.

Upon examining the head movement trace for the reference video, it is evident that this particular subject missed at least three regions of interest during scene transitions. While he/she may have managed to catch up with the ongoing scene, the initial part was lost. However, during the fade-rotation and snap-change techniques, these previously missed RoIs were realigned, allowing the subject to view the content he/she had missed. Notably, during scenes 4, 5, and 6, the subject significantly reduced hir/her head movement as he/she followed the newly presented content. This behavior was observed across different subjects and their point of attention stayed focused until the next scene transition, where the subject shifted his/her gaze to the newly realigned scenes. These three scenes transitions happened in an outdoor scene, where the characters were mountain biking. The head movement traces of all subjects, for each video, can be found in Annexes I and II. Unfortunately, we lost the head movement trace of one subject because we failed to check if the data collection module was online.

Figure 5.12 presents the head movement trace of another subject for each editing technique in the bank video, following the same structure as described in Figure 5.11. As indicated in Figure 5.6, the number of realignments in this video is considerably lower due to its slow-paced nature, with fewer events occurring within the same scene. Upon analyzing the head movement trace for the reference video, we observe that the subject was able to follow every scene transition without missing any important regions of interest. This can be attributed to the effective content guidance provided by the video itself. Some subjects even mistakenly believed they were being guided during certain scenes, when in reality it was the content's inherent flow. In the subsequent two graphs, we can observe that only two scenes required realignment, and the subject continued focusing on the same region in the subsequent scene. However, we do not see a significant reduction in the subject's head movement within the scene. This behavior was consistent across different subjects.

The results from the fast-paced and the slow-paced video shows that the use of dynamic editing techniques can reduce the user's head movement, but this results is more visible when the scene itself has more actions happening on it. In such a situation, the content realignment,

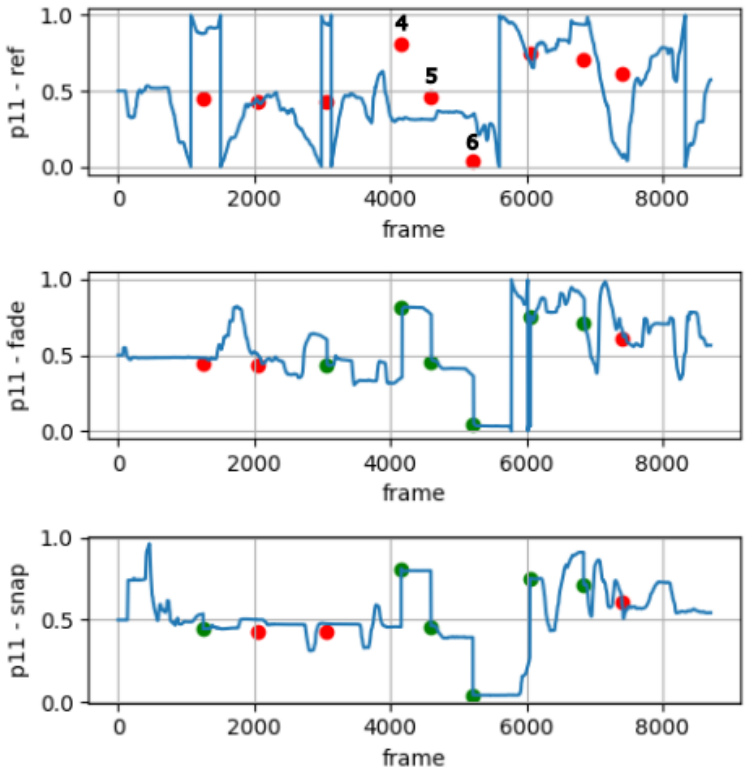

Figure 5.11. Yaw head movement normalized per frame on video vaude - Subject 11. Red dots indicate the points of scene transitions, while green dots represent the points where dynamic editing was triggered. From top to bottom: Reference video, Fade-rotation, and Snap-change.

i.e., the rotation of the content to ensure that the next RoI is visible on the users' viewport, made them focus more on the content that was presented to them. Regarding the impact of different editing techniques on the user's head movement, we did not observe any significant indication that this might be a significant variable. As discussed in the previous results, the dynamic editing techniques will have a greater impact on the user's comfort and QoE rather than reducing the user's head movement.

5.5 GENERAL COMMENTS

In this section, we will discuss the general comments and questions after the completion of the four sessions for each subject. While not all subjects chose to provide feedback, those who did were given the opportunity to share their thoughts. We will first dive into the general questions asked to the subjects, and then we will explore their overall comments.

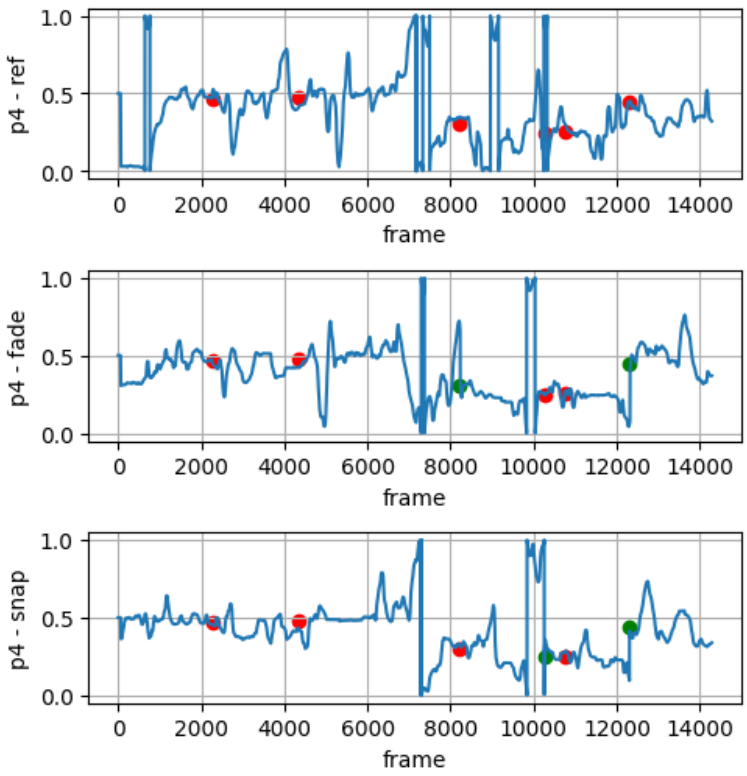

Figure 5.12. Yaw head movement normalized per frame on video vaude - Subject 4. Red dots indicate the points of scene transitions, while green dots represent the points where dynamic editing was triggered. From top to bottom: Reference video, Fade-rotation, and Snap-change.

5.5.1 Did you enjoy being guided to different regions of the video?

In response to this question, we observed three different types of subject responses: 60% of the subjects expressed that they enjoyed being guided, 13.33% did not like it, and 26.6% did not even realize that they were being guided. Among those who enjoyed being guided, one subject mentioned a preference for being guided in the "bank" video rather than the "vaude" video because they wanted the freedom to explore and not focus on a specific area. Another subject initially thought he/she would not like being guided but found that he/she felt more comfortable knowing he/she was directed to the right spot. This same subject also mentioned feeling better about the guidance in the "vaude" video. These comments, which may seem contradictory, demonstrate the subjective nature of the user's QoE.

5.5.2 Were you able to perceive the moments when the edits occurred?

In response to this question, we specifically inquired about the subjects' perception of the dynamic editing techniques. We found that 53.3% of the subjects were able to identify these editing techniques, with almost everyone mentioning that it was easier to recognize the fade-rotation technique. As previously mentioned, the snap-change technique can sometimes blend in with regular static cuts, particularly because the original video already employs this technique. Therefore, subjects would only notice these dynamic editing techniques if there were slight discrepancies in the timing of the cuts or if they were unsure of what to expect in the next scene.

5.5.3 Were there any elements/objects in the videos that you only noticed when you were guided to them?

On this question, if the subject confirms that they could notice this different element/object, we asked what was this element/object they missed on the first time. 40% of the subjects said that they could see some things only when they were guided and every scene was reported to the vaude video. From those subjects 50% reported the same scene of a man wearing a panda custom holding a sign. The other 50% reported different scenes such as different backpacks on the bicycle, or the main character lifting a bag when the camera was put on a trolley. This last example could be followed on the first time the subject watched the video if the subject knew that someone was talking with them, therefore, it is an example of how the audio can interfere the user's behavior. It is interesting to notice that every scene lost was reported on the vaude video. The reason for that is related to the number of realignments and regions of interest on the same video, because the bank video did not have a variety of regions of interest on the same scene.

This question exposed the vision of the user to the utilization of the content realignment. Although less than half of the subject described that could see different scenes, the number is not insignificant.

5.5.4 How much did the faces "artifacts" of the cube bother you in terms of overall quality of experience and comfort?

This question can help in a future work that has already been established for the 360EAVP platform. During the development process, it was discovered that when using the platform on the native web browser of the Meta Quest 2, certain "artifacts" would appear. This issue was not initially recognized because the development was initially conducted on the Firefox VR web browser, which did not exhibit this problem. As the development progressed, it became apparent that the Firefox VR browser was not fully compatible with the data collection module, prompting a switch to the native web browser of the Meta Quest 2.

With this context, the objective is to understand how these "artifacts" impact the user's QoE and Comfort. It was discovered that every subject noticed these "artifacts", and 33.33% of the subjects reported that they had some level of interference in their overall experiences. Many subjects mentioned that they couldn't fully immerse themselves in the content because they were aware of these artifacts. One subject specifically mentioned that the artifacts were more noticeable in brighter scenes and caused discomfort when the region of interest coincided with the middle of these artifacts. This particular occurrence happened only once in the vaude video.

However, the majority of subjects reported that the presence of these "artifacts" did not bother them significantly. As one subject expressed, "you have to concentrate to see these problems."Overall, it can be concluded that subjects' scores might have been higher if they were not exposed to these artifacts. Resolving this issue would likely result in an improvement in the overall immersion of users when using the platform.

5.5.5 Subjects' general comments

In this section, we will present the subjects' general comments about the experiment, focusing on the points that were reported by more than one subject.

Regarding the video content itself, multiple subjects expressed their dislike for scene transitions that were too close to each other. This issue was observed in the bank video when the

main character suddenly moved from inside a house to a meeting room, as well as in the vaude video where some scenes were relatively short. Unfortunately, this discomfort was beyond our control as it was related to the way the scenes were designed. The problem comes from the fact that the filmmakers did not used the "Extreme wide shot"technique, as described in the Section 2.1, to help situate the users in the new environment that was being presented to them.

It seems that some subjects experienced discomfort due to camera movements and positions during the experiment. For example, in the bank video, there were frequent small camera movements, which may have contributed to the discomfort. In the vaude video, when the camera was on a trolley, subjects reported a disproportionate feeling because they felt shorter, which could have also contributed to their discomfort.

Additionally, some subjects mentioned that having audio could have improved their QoE. However, for the purpose of isolating the experimental variables, audio was intentionally excluded from the videos.

Furthermore, some subjects reported feeling that they were losing some information within the same scene. More specifically, in the bank video where the scenes were longer, some subjects felt that they wanted to explore the environment more to capture different regions of interest, even though the content itself did not offer much to explore. This opens an opportunity to study the use of dynamic editing techniques within the same scene, particularly when new elements or objects are introduced.

Another aspect that was frequently mentioned is the video quality. Unfortunately, the availability of high-quality videos with storytelling is currently limited. Most of the videos that are currently available to the community do not possess the necessary level of quality to be presented on an HMD (Head-Mounted Display). Even though these videos may have a high resolution (up to 4K), the HMD users perceive a lower resolution due to only a portion of that resolution being visible within their viewport. In the future, a more careful curation process will be required to select videos with both storytelling elements and higher overall quality. Another aspect related to the video content that was mentioned is the length of the videos. Some subjects reported feeling that the videos were too long. To address this, it may be necessary to implement a pre-processing step on the videos themselves to make them more focused on the specific aspects we wanted to study. In this particular experiment, the videos were used as

they were, without any modifications.

Overall, subjects expressed positive feedback regarding the experience. The majority of them had not participated in a similar experiment before. They also reported that they started paying closer attention to the details and information when a new scene was presented to them.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

6.1 CONCLUSIONS

In this work, we introduced 360EAVP - The Edition-Aware 360-degree Video Player, which is an open-source, web browser-based application for streaming and visualizing *edited* 360-degree videos on an HMD (Head-Mounted Display). Our focus was on studying videos of promotional/advertising videos, specifically complex videos with different scenes that aim to convey a message to the user.

The proposed application builds upon VDTP, another web application that had limitations when used on Head Mounted-Displays (HMDs) and did not support the use of any editing techniques. The platform utilizes the "dash.js"framework for video streaming, the "A-Frame"framework for 3D rendering and environment handling, and the "angular.js"framework for user interface and data sharing. The platform incorporates cube mapping projection (CMP) to enable the streaming of tile videos. Each face of the cube represents a dash video instance that independently requests data from the server.

We have introduced specific functionalities, such as the ability to work with dynamic editing techniques, which involve realignments of content in real-time. We have studied two types of dynamic editing techniques: snap-change and fade-rotation. The snap-change editing technique involves instant cuts from one scene to another, providing a direct transition. On the other hand, the fade-rotation editing technique is more sophisticated. It involves slight rotation of the content and utilizes a "blink effect"to create a smooth transition between scenes.

The platform serves as a research tool for analyzing the effects of editing on the user experience and exploring new viewport prediction strategies and ABR (Adaptive Bit Rate) algorithms. It consists of various modules designed to facilitate these studies.

A preprocessing step is required to enable video playback on the platform. The platform

includes a visible face identification module that identifies the faces visible within the users' viewport and determines the extent to which they are visible. Additionally, the platform incorporates a viewport prediction module with two implemented regression techniques: Linear Regression and Ridge Regression.

To study Adaptive Bit Rate approaches, the platform features a Custom ABR module. Two available algorithms are emphasized: the Edition-Aware algorithm, which leverages prior knowledge of scene transitions to enhance quality selection, and the Default rule from the "dash.js"reference player, which represents the standard rule. However, the performance evaluation of the algorithms was not conducted in this work and is listed as a future work. Moreover, the platform includes a video playback data collection module that gathers pertinent information, including the current user's viewport details and edition information such as scheduled frames and their types.

Furthermore, we conducted an experiment to evaluate the impact of dynamic editing techniques on the user's Quality of Experience (QoE) and comfort. Two videos were used for this evaluation: one with fast-paced content (vaude) and another with slow-paced content (bank). Our findings revealed that the implementation of editing techniques did not reduced the overall QoE and comfort, in fact, in certain scenarios, we observed an increase in both aspects.

In the same experiment, we analyzed the head movement track data to assess how the use of dynamic editing techniques could influence the overall head movement speed and whether different techniques could affect user head movement. Interestingly, we observed a significant reduction in head movement during certain scene transitions, particularly in content featuring more actions within the same scene. However, we did not obtain conclusive results regarding whether the use of different dynamic editing techniques could impact head movement.

These results highlight the potential benefits of incorporating prior knowledge of scene transitions into ABR algorithms, enabling them to make better decisions. By leveraging dynamic editing techniques effectively, it is possible to enhance the overall user experience by optimizing QoE, comfort, and head movement in specific scenarios.

Moreover, we have collected general comments about the experiment and their experience. To summarize, the subjects' feedback provided valuable insights into their experience during the experiment. While some subjects reported discomfort related to camera movements, positions,

and scene transitions, others expressed a desire for more exploration within the scenes. The absence of audio was noted as a potential factor that could have improved the overall Quality of Experience, although it was deliberately excluded to isolate the experimental variables.

The findings highlighted the importance of considering dynamic editing techniques within the same scene, especially when introducing new elements or objects. Additionally, subjects' feedback emphasized the need for a careful curation process to select videos with storytelling and higher quality for future experiments.

Overall, the subjects appreciated the experience, and for many, it was their first time participating in such an experiment. They demonstrated an increased awareness and attention to the details and information presented within the scenes. The feedback gathered provides valuable insights for further enhancing the experimental setup and improving the overall Quality of Experience for future subjects.

6.2 FUTURE WORK

Future work will focus on enhancing the performance of 360EAVP. Currently, the platform does not support working with different tile grids due to the potential overload of parallelism on the client side. Addressing this limitation will enable more flexible configurations and improved scalability.

Furthermore, as we have observed the positive impact of dynamic editing techniques on user quality of experience, comfort, and reduce head movement, the next step is to have more subjects on the study and to integrate this knowledge with optimization efforts for ABR algorithms. By leveraging this prior information effectively, ABR algorithms can make more informed decisions and enhance video streaming of storytelling content over the internet. These future developments can improve the overall performance and user experience of the 360EAVP platform, opening possibilities for more versatile and optimized video streaming in the context of immersive storytelling.

REFERENCES

A-FRAME. *A-Frame*. 2023. Accessed: May 11, 2023. Disponível em: <https://aframe.io/>. Cited in page 40.

ANGULARJS. *AngularJS*. 2023. Accessed: May 11, 2023. Disponível em: <https://angularjs.org/>. Cited in page 39.

ARAÚJO, G. D. C.; GARCIA, H. D.; FARIAS, M.; PRAKASH, R.; CARVALHO, M. 360eavp: A 360-degree edition-aware video player. In: *Proceedings of the 15th International Workshop on Immersive Mixed and Virtual Environment Systems*. [S.l.: s.n.], 2023. p. 18–23. Cited in page 6.

BAO, Y.; WU, H.; ZHANG, T.; RAMLI, A. A.; LIU, X. Shooting a moving target: Motion-prediction-based transmission for 360-degree videos. In: *2016 IEEE International Conference on Big Data (Big Data)*. [S.l.: s.n.], 2016. p. 1161–1170. Cited 2 times in pages 18 and 30.

BENTO4. *Bento4*. 2023. Accessed: March 14, 2023. Disponível em: <https://www.bento4.com/>. Cited in page 25.

BRUNNSTRÖM, K.; BEKER, S. A.; MOOR, K. D.; DOOMS, A.; EGGER, S.; GARCIA, M.-N.; HOSSFELD, T.; JUMISKO-PYYKKÖ, S.; KEIMEL, C.; LARABI, M.-C. *et al.* Qualinet white paper on definitions of quality of experience. 2013. Cited in page 43.

CHEN, Z.; LI, Y.; ZHANG, Y. Recent advances in omnidirectional video coding for virtual reality: Projection and evaluation. *Signal Processing*, Elsevier, v. 146, p. 66–78, 2018. Cited 2 times in pages 12 and 13.

CHOPRA, L.; CHAKRABORTY, S.; MONDAL, A.; CHAKRABORTY, S. Parima: Viewport adaptive 360-degree video streaming. In: *Proceedings of the Web Conference 2021*. [S.l.: s.n.], 2021. p. 2379–2391. Cited in page 18.

COBAN, M.; AUWERA, G. Van der; KARCZEWICZ, M. Ahg8: Adjusted cubemap projection for 360-degree video. *Joint Video Exploration Team of ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, JVET-F0025*, 2017. Cited in page 12.

CORBILLON, X.; DEVLIC, A.; SIMON, G.; CHAKARESKI, J. Optimal set of 360-degree videos for viewport-adaptive streaming. In: *Proceedings of the 25th ACM International Conference on Multimedia*. New York, NY, USA: Association for Computing Machinery, 2017. (MM '17), p. 943–951. ISBN 9781450349062. Cited in page 2.

DAMBRA, S.; SAMELA, G.; SASSATELLI, L.; PIGHETTI, R.; APARICIO-PARDO, R.; PINNA-DéRY, A.-M. Film editing: New levers to improve vr streaming. In: *Proceedings of the 9th ACM Multimedia Systems Conference*. New York, NY, USA: Association for Computing Machinery, 2018. (MMSys '18), p. 27–39. ISBN 9781450351928. Cited 2 times in pages 4 and 21.

DASH.JS. *dash.js*. 2023. Accessed: May 11, 2023. Disponível em: <https://github.com/Dash-Industry-Forum/dash.js/wiki/ABR-Logic#custom-rules>. Cited 2 times in pages 31 and 32.

FEUVRE, J. L. Gpac filters. In: *Proceedings of the 11th ACM Multimedia Systems Conference*. New York, NY, USA: Association for Computing Machinery, 2020. (MMSys '20), p. 249–254. ISBN 9781450368452. Cited in page 21.

GOOGLE. *Bringing pixels front and center in VR video*. 2017. Accessed: May 29, 2023. Disponível em: <https://blog.google/products/google-ar-vr/bringing-pixels-front-and-center-vr-video/>. Cited in page 12.

GUAN, Y.; ZHENG, C.; ZHANG, X.; GUO, Z.; JIANG, J. Pano: Optimizing 360 video streaming with a better understanding of quality perception. In: *Proceedings of the ACM Special Interest Group on Data Communication*. [S.l.: s.n.], 2019. p. 394–407. Cited in page 19.

GUTIERREZ, J.; PEREZ, P.; ORDUNA, M.; SINGLA, A.; CORTES, C.; MAZUMDAR, P.; VIOLA, I.; BRUNNSTRÖM, K.; BATTISTI, F.; CIEPLIŃSKA, N. *et al.* Subjective evaluation of visual quality and simulator sickness of short 360 circ videos : Itu − trec.p.919.IEEE transactions on multimedia, IEEE, v. 24, p.3087 − −3100, 2021. Citedinpage43.

HAAKE, S.; MÜLLER, W. Cyberella–design issues for interactive 360 degree film. In: SPRINGER. *Interactivity, Game Creation, Design, Learning, and Innovation: 7th EAI International Conference, ArtsIT 2018, and 3rd EAI International Conference, DLI 2018, ICTCC 2018, Braga, Portugal, October 24–26, 2018, Proceedings 7*. [S.l.], 2019. p. 152–162. Cited in page 3.

HOU, H. *Learn 360 VR Filmmaking - Cinematic Rules: Framing, Angles of Shots, Camera Movements in 360°*. 2020. Accessed: July 30, 2023. Disponível em: <https://www.youtube.com/watch?v=DJUrHXiZImU>. Cited in page 9.

KAN, N.; ZOU, J.; LI, C.; DAI, W.; XIONG, H. Rapt360: Reinforcement learning-based rate adaptation for 360-degree video streaming with adaptive prediction and tiling. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 32, n. 3, p. 1607–1623, 2021. Cited in page 19.

KROMA, A. The technical dilemmas of creative design and rapid prototyping for immersive storytelling. *Creativity and Cognition*, 2022. Cited in page 3.

LEXICON, H. *30 Degree Rule*. 2000. Accessed: July 13, 2023. Disponível em: <http://www.hollywoodlexicon.com/thirtydegree.html>. Cited in page 26.

LINDSKOG, E.; CARLSSON, N. Reeft-360: Real-time emulation and evaluation framework for tile-based 360 streaming under time-varying conditions. In: *Proceedings of the 12th ACM Multimedia Systems Conference*. [S.l.: s.n.], 2021. p. 307–313. Cited 2 times in pages 19 and 21.

META. *Meta Quest 2*. 2023. Accessed: July 18, 2023. Disponível em: <https://www.meta.com/quest/products/quest-2/>. Cited in page 47.

MURCH, W. *In the Blink of an Eye*. [S.l.]: Silman-James Press Los Angeles, 2001. v. 995. Cited in page 8.

NASRABADI, A. T.; MAHZARI, A.; BESHAY, J. D.; PRAKASH, R. Adaptive 360-degree video streaming using scalable video coding. In: *Proceedings of the 25th ACM international conference on Multimedia*. [S.l.: s.n.], 2017. p. 1689–1697. Cited in page 19.

NGUYEN, D. V.; TRAN, H. T.; THANG, T. C. An evaluation of tile selection methods for viewport-adaptive streaming of 360-degree video. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, ACM New York, NY, USA, v. 16, n. 1, p. 1–24, 2020. Cited in page 2.

NIAMUT, O. A.; THOMAS, E.; D'ACUNTO, L.; CONCOLATO, C.; DENOUAL, F.; LIM, S. Y. Mpeg dash srd: spatial relationship description. In: *Proceedings of the 7th International Conference on Multimedia Systems*. [S.l.: s.n.], 2016. p. 1–8. Cited in page 13.

PODBORSKI, D.; SON, J.; BHULLAR, G. S.; SKUPIN, R.; SANCHEZ, Y.; HELLGE, C.; SCHIERL, T. Html5 mse playback of mpeg 360 vr tiled streaming: Javascript implementation of mpeg-omaf viewport-dependent video profile with hevc tiles. In: *Proceedings of the 10th ACM Multimedia Systems Conference*. [S.l.: s.n.], 2019. p. 324–327. Cited in page 21.

QIAN, F.; HAN, B.; XIAO, Q.; GOPALAKRISHNAN, V. Flare: Practical viewport-adaptive 360-degree video streaming for mobile devices. In: *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: Association for Computing Machinery, 2018. (MobiCom '18), p. 99–114. ISBN 9781450359030. Cited 3 times in pages 20, 30, and 31.

QIAN, F.; JI, L.; HAN, B.; GOPALAKRISHNAN, V. Optimizing 360 video delivery over cellular networks. In: *Proceedings of the 5th Workshop on All Things Cellular: Operations, Applications and Challenges*. New York, NY, USA: Association for Computing Machinery, 2016. (ATC '16), p. 1–6. ISBN 9781450342490. Cited 2 times in pages 18 and 30.

RAYSEE. *vr-dash-tile-player*. 2023. Accessed: May 11, 2023. Disponível em: <https://github.com/ForgetMe17/vr-dash-tile-player>. Cited in page 6.

REPORTS, V. *Augmented Reality and Virtual Reality (AR & VR) Market Size is Expected to Reach USD 571.42 Billion by 2025 | Valuates Reports*. [S.l.], 2023. Accessed: March 13, 2023. Disponível em: <https://www.prnewswire.com/in/news-releases/augmented-reality-and-virtual-reality-ar-amp-vr-market-size-is-expected-to-reach-usd-571-42-billion-b
html>. Cited in page 1.

RIBEZZO, G.; CICCO, L. D.; PALMISANO, V.; MASCOLO, S. Tapas-360: A tool for the design and experimental evaluation of 360 video streaming systems. In: *Proceedings of the 28th ACM International Conference on Multimedia*. [S.l.: s.n.], 2020. p. 4477–4480. Cited in page 21.

RITTER, J. An efficient bounding sphere. *Graphics gems*, Academic Press Boston, MA, v. 1, p. 301–303, 1990. Cited in page 27.

SCHWARZ, H.; MARPE, D.; WIEGAND, T. Overview of the scalable video coding extension of the h. 264/avc standard. *IEEE Transactions on circuits and systems for video technology*, IEEE, v. 17, n. 9, p. 1103–1120, 2007. Cited in page 19.

SHIBATA, T. Head mounted display. *Displays*, Elsevier, v. 23, n. 1-2, p. 57–64, 2002. Cited in page 1.

SPITERI, K.; SITARAMAN, R.; SPARACIO, D. From theory to practice: Improving bitrate adaptation in the dash reference player. *ACM Trans. Multimedia Comput. Commun. Appl.*, Association for Computing Machinery, New York, NY, USA, v. 15, n. 2s, jul 2019. ISSN 1551-6857. Cited 2 times in pages 14 and 22.

SPITERI, K.; URGAONKAR, R.; SITARAMAN, R. K. Bola: Near-optimal bitrate adaptation for online videos. *IEEE/ACM Trans. Netw.*, IEEE Press, v. 28, n. 4, p. 1698–1711, aug 2020. ISSN 1063-6692. Cited 3 times in pages 14, 22, and 33.

STOCKHAMMER, T. Dynamic adaptive streaming over http– standards and design principles. In: *Proceedings of the second annual ACM conference on Multimedia systems*. [S.l.: s.n.], 2011. p. 133–144. Cited in page 13.

THREE.JS. *Three.js*. 2023. Accessed: May 11, 2023. Disponível em: <https://threejs.org/>. Cited in page 40.

UNION, I. T. Methodology for the subjective assessment of the quality of television pictures. tech. rep. bt.500-11, itu-r. In: . [S.l.: s.n.], 2002. Cited in page 43.

ZHANG, L.; SUO, Y.; WU, X.; WANG, F.; CHEN, Y.; CUI, L.; LIU, J.; MING, Z. Tbra: Tiling and bitrate adaptation for mobile 360-degree video streaming. In: *Proceedings of the 29th ACM International Conference on Multimedia*. [S.l.: s.n.], 2021. p. 4007–4015. Cited in page 20.

ANNEX

PARTICIPANTS HEAD MOVEMENT PER EDITING TECHNIQUE - VAUDE

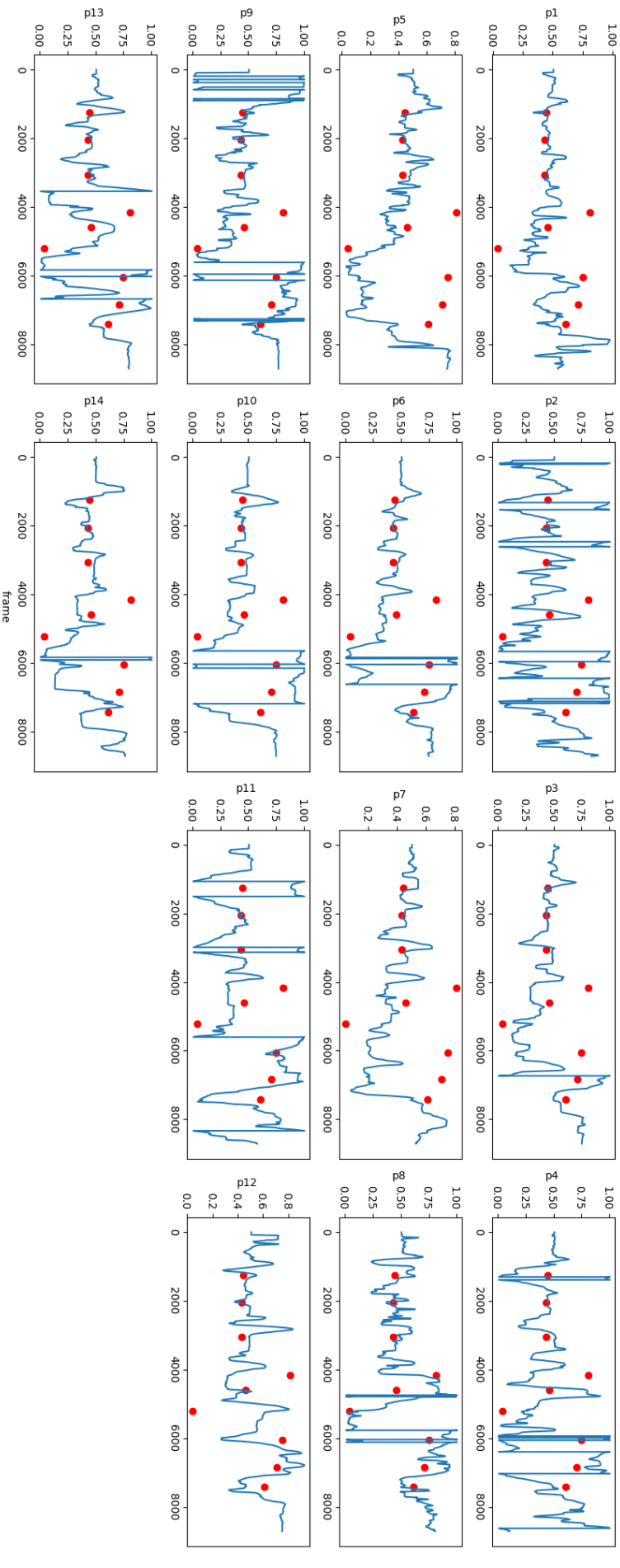

Figure I.1. Reference video - Static editing. Red dots indicate the points of scene transitions.

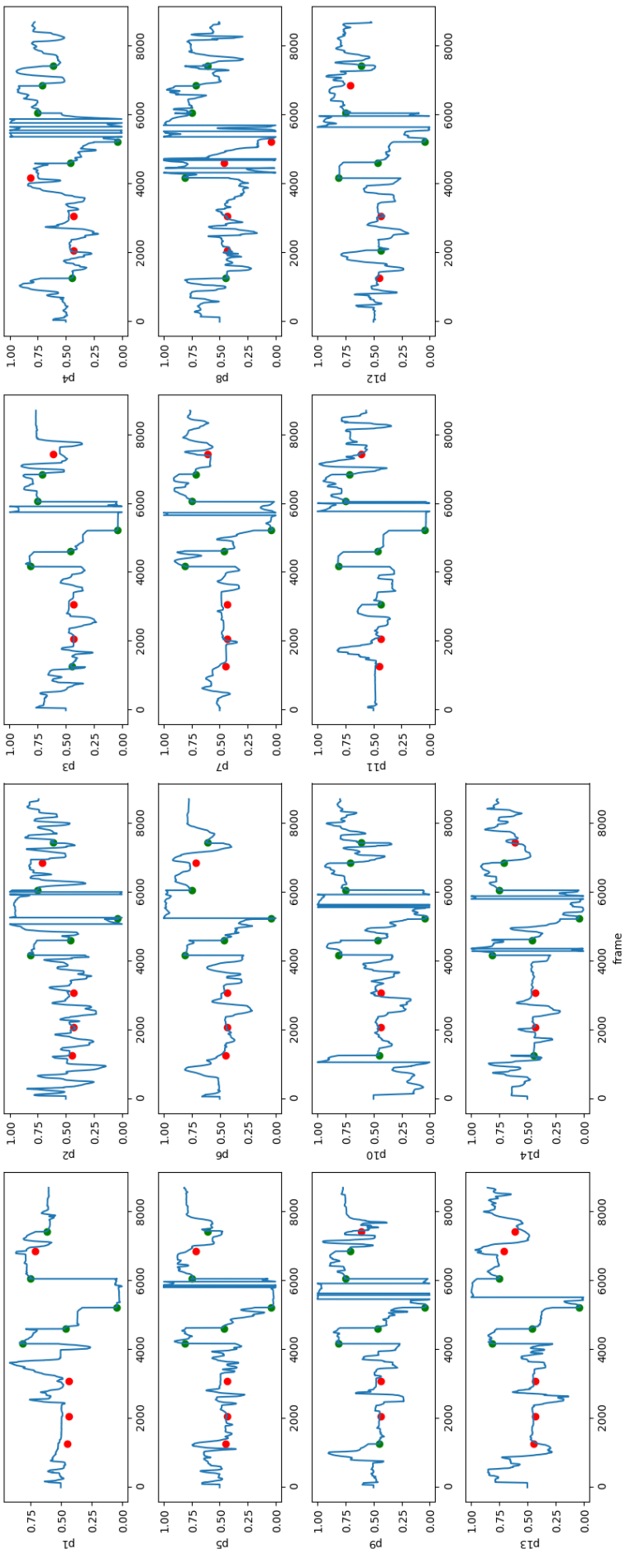

Figure I.2. Fade-rotation editing technique. Red dots indicate the points of scene transitions, while green dots represent the points where dynamic editing was triggered

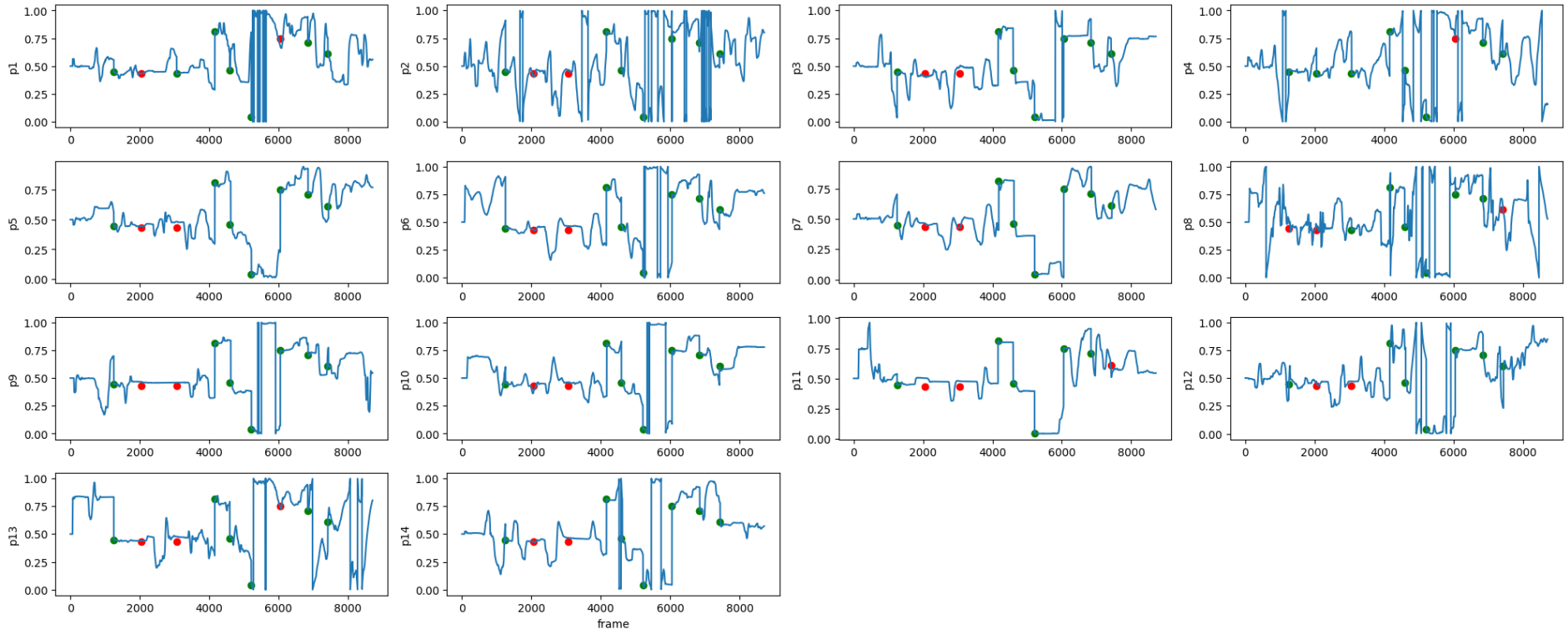

Figure I.3. Snap-change editing technique. Red dots indicate the points of scene transitions, while green dots represent the points where dynamic editing was triggered

II

PARTICIPANTS HEAD MOVEMENT PER EDITING TECHNIQUE - BANK

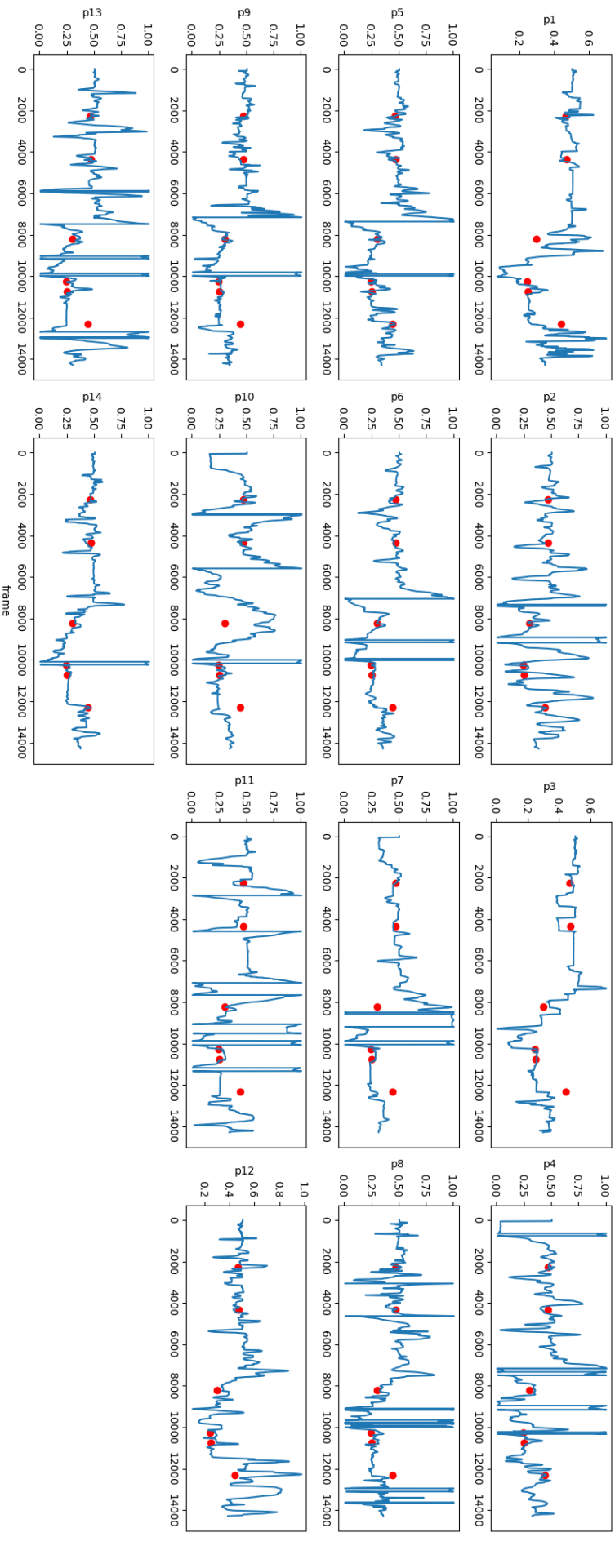

Figure II.1. Reference video - Static editing. Red dots indicate the points of scene transitions.

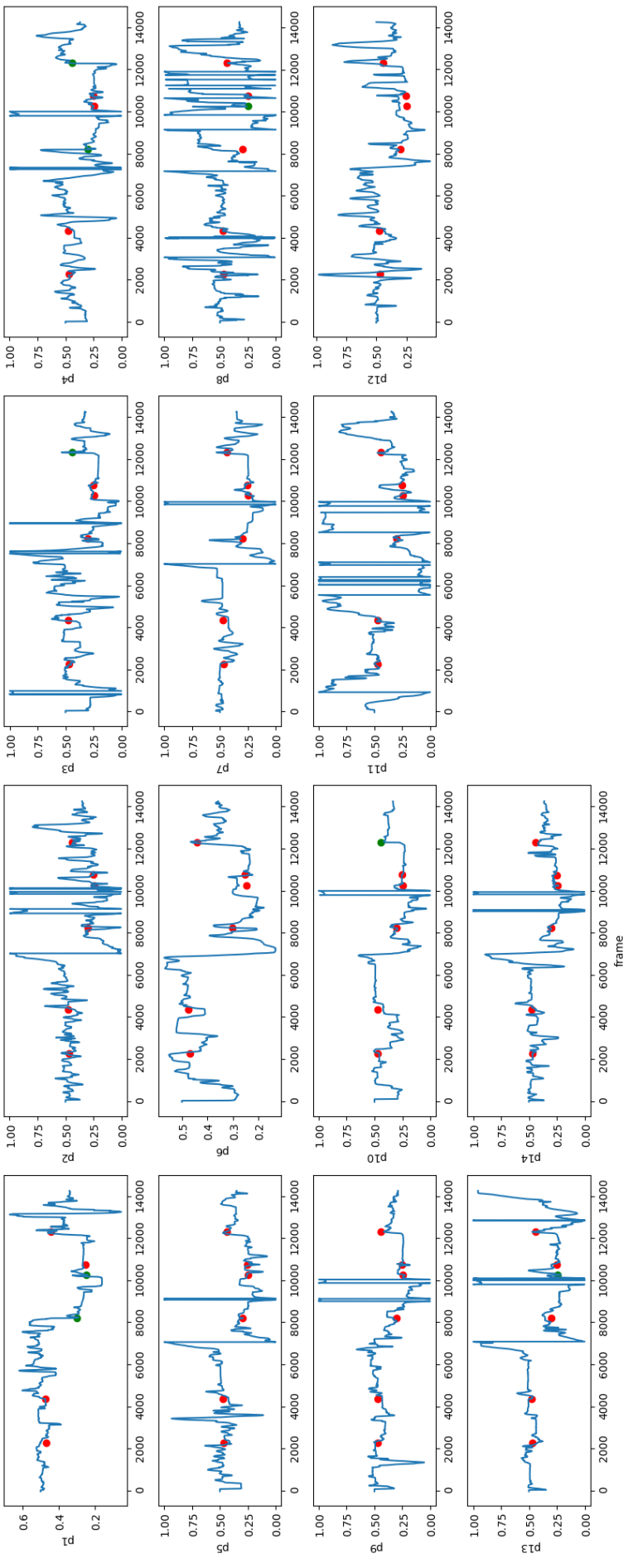

Figure II.2. Fade-rotation editing technique. Red dots indicate the points of scene transitions, while green dots represent the points where dynamic editing was triggered

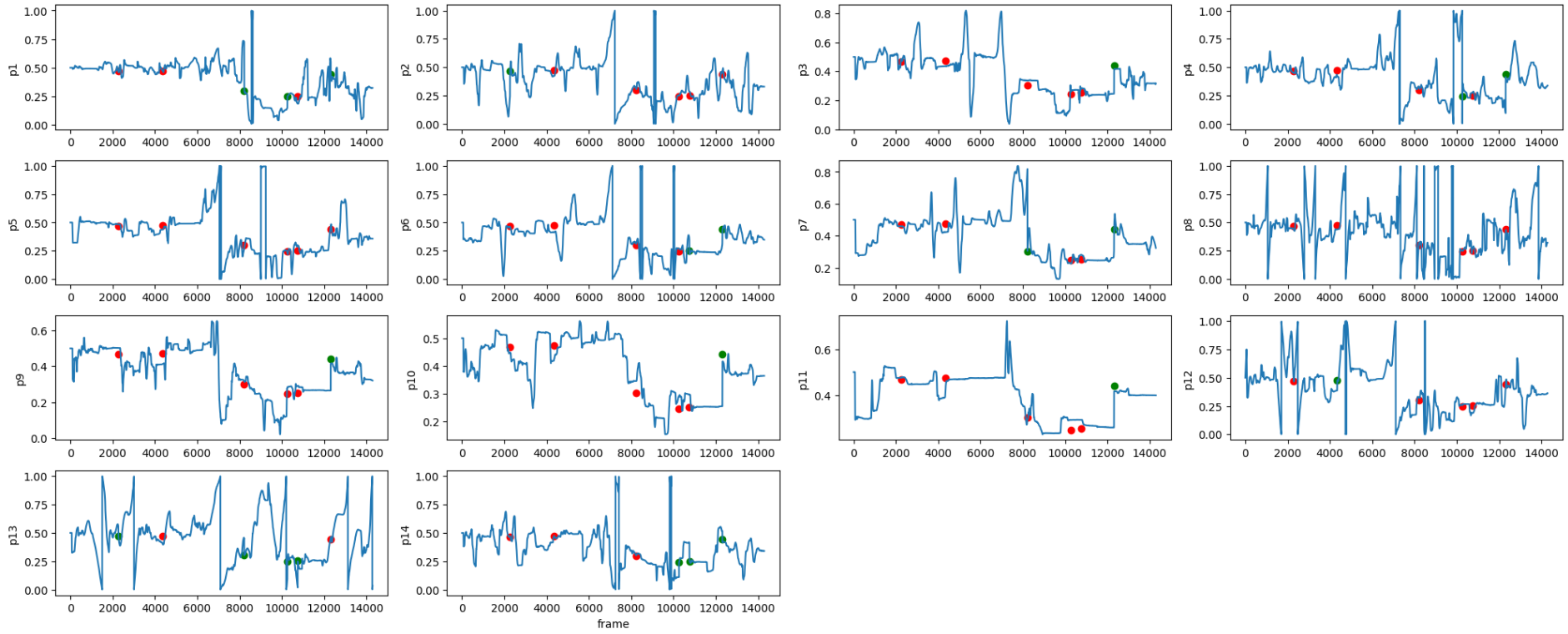

Figure II.3. Snap-change editing technique. Red dots indicate the points of scene transitions, while green dots represent the points where dynamic editing was triggered

III

JSON FILE CONTAINING EDITING TECHNIQUES SCHEDULED.

Listing III.1. Video *vaude* without audio with snap-change editing scheduled.

```
{
  "baseUrl": "https://192.168.15.182/vaude/",
  "face": 6,
  "row": 1,
  "col": 1,
  "duration": 1,
  "tiles": [
    [[{"src": "face0/output/face0.mpd", "width": "1440", "height": "1440"}]],
    [[{"src": "face1/output/face1.mpd", "width": "1440", "height": "1440"}]],
    [[{"src": "face2/output/face2.mpd", "width": "1440", "height": "1440"}]],
    [[{"src": "face3/output/face3.mpd", "width": "1440", "height": "1440"}]],
    [[{"src": "face4/output/face4.mpd", "width": "1440", "height": "1440"}]],
    [[{"src": "face5/output/face5.mpd", "width": "1440", "height": "1440"}]]
  ],
  "edits":
      {
      "edit" : [
        {
        "frame" : 1251,
        "type": "instant",
        "region_of_interest": [
          {
            "rank": 1,
            "ROI_theta": 0.446

          }
          ]
        },
        {
          "frame" : 2052,
```

```
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.432
              }
              ]
            },
          {
            "frame" : 3060,
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.432
              }
              ]
            },
          {
            "frame" : 4160,
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.813
              }
              ]
            },
          {
            "frame" : 4591,
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.464
              }
              ]
            }
            ,
          {
            "frame" : 5212,
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.042
              }
              ]
            }
```

```
            ,
          {
            "frame" : 6050,
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.750
              }
              ]
            },
          {
            "frame" : 6840,
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.718
              }
              ]
            },
          {
            "frame" : 7417,
            "type": "instant",
            "region_of_interest": [
              {
                "rank": 1,
                "ROI_theta": 0.611
              }
              ]
            }
        ]
    }
}
```