Deliverable

Project Acronym:	**ImmersiaTV**
Grant Agreement number:	688619
Project Title:	*Immersive Experiences around TV, an integrated toolset for the production and distribution of immersive and interactive content across devices.*

D4.3 Technical Evaluation

Revision: 0.6

Author:

Saeed Mahmoudpour (imec)

Delivery date: M25

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement 688619		
Dissemination Level		
P	Public	x
C	Confidential, only for members of the consortium and the Commission Services	

Abstract: This deliverable will detail the technical aspects of the outcome of the pilots, what worked and what didn't, what should be improved from a technical performance perspective. The technical evaluation will be focused on the performance monitoring of different components delivered in the iterations of the project, the lessons learned in each iteration, and guidelines to improve performance.

REVISION HISTORY

Revision	Date	Author	Organisation	Description
0.1	13/12/2016	Saeed Mahmoudpour/ Adriaan Barri	imec	First draft version of the deliverable
0.2	30/01/2016	Saeed Mahmoudpour/ Adriaan Barri	imec	First review
0.3	08/03/2017	Saeed Mahmoudpour/ Adriaan Barri	imec	Second review
0.4	31/05/2017	Saeed Mahmoudpour	imec	Third revision – addressing reviewers comments
0.5	22/12/2017	Saeed Mahmoudpour	imec	Pilot 2 technical evaluation
0.6	22/01/2018	Saeed Mahmoudpour	Imec	Third Review

EXECUTIVE SUMMARY

This deliverable describes the detailed technical evaluation of the outcome of Pilot 1 and Pilot 2 in the ImmersiaTV project. The aim of the document is to report the technical specifications and performance of the project components, describe the current issues and provide guidelines for better performance.

After the introduction, section 2 provides an overview of the ImmersiaTV platform and components. In section 3, we go through the most noteworthy technical problems and limitations faced in Pilot 1 at both the production side and the end-user side. In section 4 we explain how in video capturing and visualization we observed optical distortions, which are mentioned and described. Next, the video stitching software used in Pilot 1 is introduced and the stitching errors are further explained. In the video editing part, we report the improvements in the Adobe Premiere plug-in, portals and scene transitions. The distortion types observed in Pilot 1 (including Blocking, Ringing, blur and colour artifacts) are described and shown by examples. The issues related to distribution, reception and interaction are also described in detail.

According to the observed issues and lessons learned in Pilot 1, a number of guidelines are provided in section 4 to improve the performance of pre-production and post-production. Moreover, some desirable features are suggested that could be added to the current ImmersiaTV platform. Section 5 summarizes the current status of user requirements mentioned in deliverable D 2.3.

The architecture of the ImmersiaTV platform for Pilot2 is briefly described in section 6. The technical specifications of the tools used in Pilot2 and the technical experiments are provided in section 7. Different modules including production tool, codec, projection and remapping, reception tools and QoE module go through substantial evaluations and the results are summarized and discussed in this section. Finally, section 8 addresses the conclusion.

CONTRIBUTORS

First Name	Last Name	Company	e-Mail
Sergi	Fernandez	i2CAT	Sergi.fernandez@i2cat.net
Joan	Llobera	i2CAT	Joan.llobera@i2cat.net
Isaac	Fraile	i2CAT	Isaac.fraile@i2cat.net
David	Gomez	i2CAT	David.gomez@i2cat.net
Stephane	Valente	VideoStitch	Stephane.valente@video-stitch.com
Szymon	Malewski	PSNC	szymonm@man.poznan.pl
Touradj	Ebrahimi	EPFL	touradj.ebrahimi@epfl.ch
Evgeniy	Upenik	EPFL	Evgeniy.upenik@epfl.ch
Evangelos	Alexiou	EPFL	Evangelos.alexiou@epfl.ch
Alexander	Kelembet	Cinegy	kelembet@cinegy.com
Mikołaj	Węgrzynowski	PSNC	mwegrzyn@man.poznan.pl

CONTENTS

TABLE OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

Acronym	Description
VR	Virtual Reality
HMD	Head Mounted Display
QoE	Quality of Experience
DASH	Dynamic Adaptive Streaming over HTTP
UHD	Ultra-High Definition
HDMI	High-Definition Multimedia Interface
GoP	Group of Pictures
DoP	Director of Photography
HDRI	High Dynamic Range Imaging
FPS	Frame Per Second
RC	Rate Control
PTS	Presentation Time-Stamp
CDN	Content Delivery Network

1. INTRODUCTION

1.1. Purpose of this document

This report will detail, iteratively, the technical aspects of the outcome of the pilots: what worked and what didn't, what should be improved from a technical performance perspective. The technical evaluation will be centred on the performance monitoring of the components delivered in the various iterations of the project.

1.2. Scope of this document

This document is divided into main three sections. It starts with a brief overview of the ImmersiaTV platform and the technical specifications of video capture, stitching, editing, compression, distribution, reception & interaction, and QoE feedback. A more elaborate description can be found in deliverable D3.1 (Architecture design). Section 3 goes through the most noteworthy technical problems and limitations faced in Pilot 1 at both the production side and the end-user side. The impact of the occurred technical problems on the user experience will be reported in deliverable D4.4 (User Evaluation). Section 4 lists the most important lessons learned and some guidelines gathered from ImmersiaTV partners.

1.3. Relation with other ImmersiaTV activities

This deliverable is part of task T4.3 (User and Technical Evaluation) in WP4 (Demonstration Pilots). The relationship between this task and the other WP tasks is shown below.

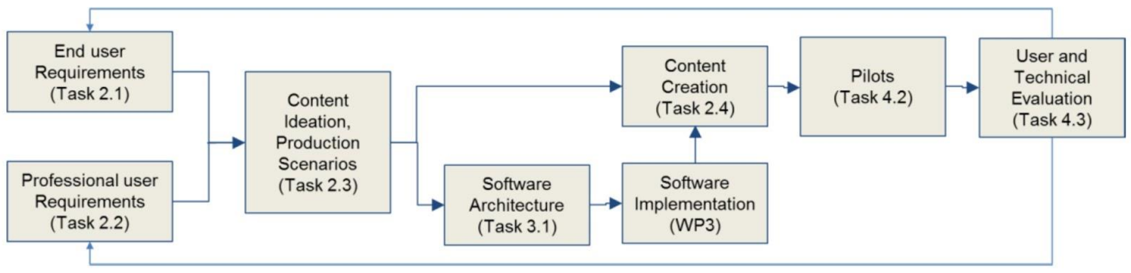

End user Requirements (Task 2.1)
Professional user Requirements (Task 2.2)
Content Ideation, Production Scenarios (Task 2.3)
Software Architecture (Task 3.1)
Software Implementation (WP3)
Content Creation (Task 2.4)
Pilots (Task 4.2)
User and Technical Evaluation (Task 4.3)

Figure 1: Role of T4.3 within the ImmersiaTV platform

2. IMMERSIATV PLATFORM OVERVIEW

ImmersiaTV aims to distribute omnidirectional and directive audiovisual content simultaneously to head mounted displays (HMD), companion screens and the traditional TV. The content distributed is constituted of one or more omnidirectional videos, complemented with several directive shots, and metadata detailing how to merge these streams in an immersive display, in coordination with directive and omnidirectional videos also shown in traditional TVs and tablets.

As detailed in Deliverable D3.1 - Architecture Design[1], the ImmersiaTV platform involves the following processes:

2.1. Video Capture

The acquisition of the video streams coming from 360 omnidirectional camera systems as well as other sources such as high-resolution directive cameras, video clips, textual information and other metadata required for generating omnidirectional video enriched with audiovisual and auxiliary information in further stages.

2.2. Video Stitching

The combination of video streams from the omnidirectional camera systems (constructed using multiple physical cameras) into a 360-degree video.

2.3. Post-Production (Offline or Live)

A set of tools and plugins for the montage, colour grading, and processing/enhancement of the 360-degree video material. The outcome of the post-production phase is a set of synchronized video signals to be played on the each of the display devices of the end user (HMD, companion screen, and traditional TV).

2.4. Compression (Encoding)

Reduction of the bitrate of the video signals (produced offline or streamed in real-time) by removing unnecessary or less important information.

2.5. Distribution

Transmission of the encoded video signals from the server at the production side to the centralized computing unit at the end-user side. The transmission channel encapsulates the selected video streams into network protocols.

2.6. Reception & Interaction

Reception of the encoded video signals at the end-user side. The player communicates with different display devices to make sure the video signals are properly received and synchronized.

2.7. QoE feedback

The QoE module estimates the perceptual quality of the visualization on the display devices at the end-user side by means of subjective and objective metrics and suggests adjustments of the network/encoding/rendering parameters at regular time intervals that will optimize the overall quality of experience of the end user.

[1] http://www.immersiatv.eu/wp-content/uploads/2015/11/D3.1-Architecture-design-version-0.9.pdf

3. TECHNICAL SPECIFICATIONS OF PILOT 1 AND THE OBSERVED ISSUES

Pilot 1 focuses on offline video production scenarios. The stitching of the omnidirectional scenes captured by several cameras was performed using AutoPano. Omnidirectional and directive streams are processed and aligned using off the shelf stitching tools and an Adobe Premiere Plugin specifically designed to edit multi-platform videos, combining omnidirectional and traditional videos, both standalone and embedded as video inserts within the omnidirectional scene.

After offline production, the video signals are compressed using the H.264 codec, transferred to a DASH server, and transmitted to central computing unit at the end user's side. This computing unit runs a dedicated player that communicates with different display devices to make sure the video signals are properly received and synchronized on each of the display devices. All components required to achieve the goals of Pilot 1 are depicted in Figure 2.

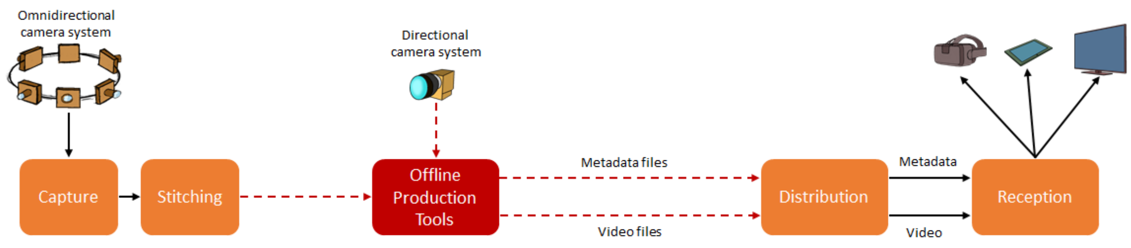


Figure 2: Architecture design of Pilot 1 (cf. D3.1)

This section describes the technical specifications of the different components in the architecture design and sums up the most noticeable issues that were observed during Pilot 1 (see table 1). The next section lists important lessons learned from these technical issues.

Design Components	Technical issues
1. Video Capture & Visualization	Some optical distortions in the 360 video material may be resolved with a better calibration and smarter camera setup.
2. Video Stitching	Some stitching errors would have been less visible if they had appeared in less sensitive parts of the scene (e.g. objects in the foreground or through smooth curves).
3. Post Production	Slow Adobe premiere plugin makes video editing more cumbersome.
4. Compression	Compression ratio is too high: distortion artifacts of different types were clearly visible in foreground objects (blocking, blurring, ringing, etc.).
5. Distribution	High-resolution content could not yet be handled by the DASH player.
6. Reception & Interaction	Synchronization between HMD, tablet and TV is sometimes disrupted when WiFi network is unstable.

Table 1: Technical issues per design component

3.1. Video capture

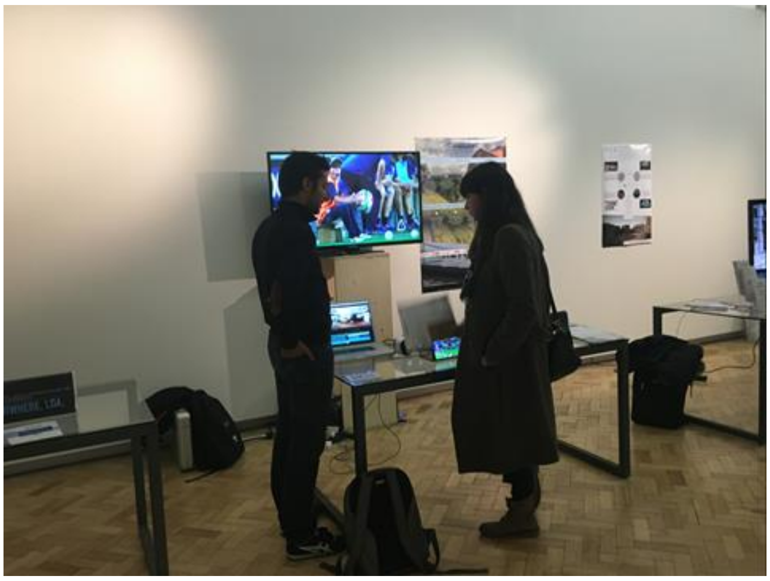


Figure 3: Display devices available to the end-user in Pilot 1

3.1.1. Camera specifications and setup

Pilot 1 deploys the one 6 Camera Rig with GoPro3 and one 3 Camera GoPro4 rig with a set of Entaniya 220 lenses[2] for off-line omnidirectional content capturing. These capture devices store several video streams on SD cards, which are stitched and made available to off-line processing tools.

The H3PRO6 rig enables to combine 6 GoPro Hero 3 Black cameras together for capturing omnidirectional video streams. Each piece of the camera has a 12MPix CMOS sensor and produces H.264 encoded stream or provides HDMI live output. The cameras support storing on microSD/microSDHC cards in resolution up to 4K, although the frame rate in UHD resolutions is rather poor (12 or 15 fps). Each camera handles Full HD resolution in 60 fps (recording) or 30 fps (HDMI output). In this specific case, each camera recorded 2,7K at 23,98 fps. By using Pro Tune, we managed to balance the exposure between cameras.

The QBiC Panorama X camera rig enables to combine 4 QBiC cameras for capturing omnidirectional video streams. The camera is equipped with CMOS sensor and supports resolution up to Full HD in 60p. (recording) or 30 fps (HDMI output). The camera has WiFi output.

The most important parameters of the described camera systems with output capabilities are described and compared in Table 2 (cf. D3.1 - Architecture design).

3.1.2. Optical distortions

The spherical lenses in HMDs and in the capture devices induce optical distortions of which geometric distortions are the most common and important type. Geometric distortions in the 360-degree video material result in straight lines being perceived as skewed curves. Several scenes in the Pilot 1 demo suffered from clearly visible geometric distortions in certain angular

[2] https://www.entapano.com/en/l/panoramic_camera_panorama.html

perspectives (Figure 4) on both the tablet and the HMD. Some distortions at the capture side could have been avoided by improving the calibration quality of the cameras.

Camera Rig	# sensors	Output resolution
H3PRO6 Rig	6	**Max resolution/frame rate** 6x 1920x1080p/60 fps when recording on SD card 6x 1920x1080p/30 fps on HDMI output
Elmo QBIC Panorama X rig	4	**Max resolution/frame rate** 4x 1920x1080p/60 fps when recording on SD card 4x 1920x1080p/30 fps on HDMI output

Table 2: Pilot 1 camera specifications

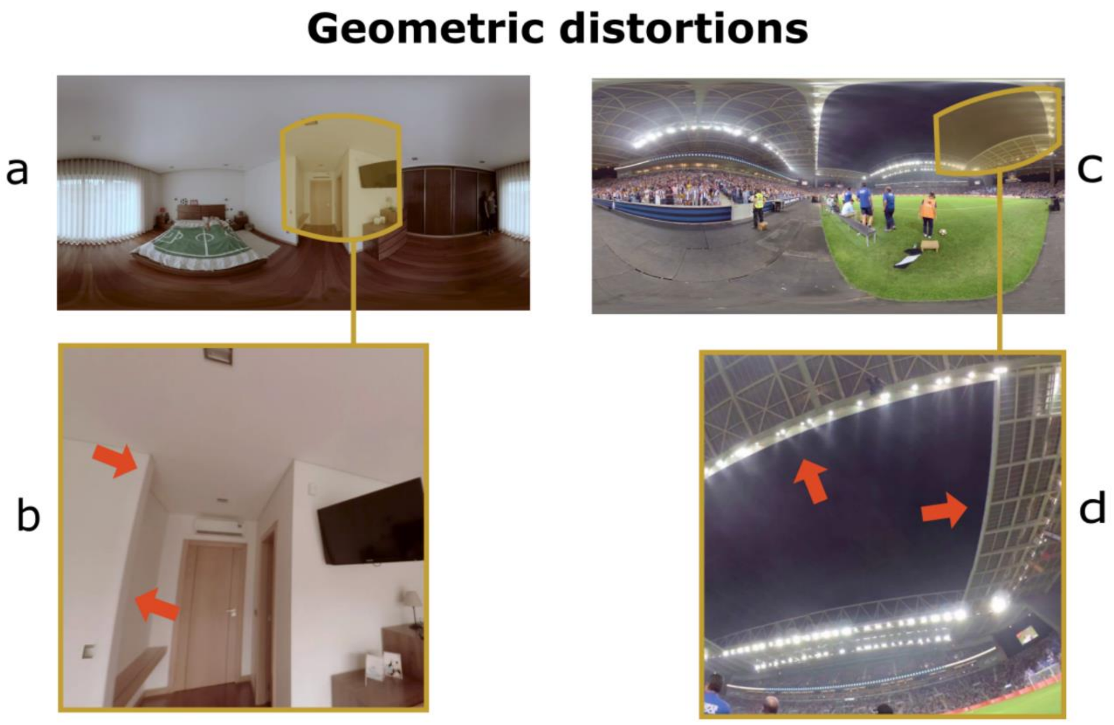


Figure 4: Geometric distortions observed in Pilot 1. (a) A video frame with equirectangular projection. (b) Zoom-in patch of the frame_(a) for better visualization of geometric distortion. (c) A video frame with equirectangular projection and (d) Zoom-in patch of the frame_(c) for better visualization of geometric distortion.

3.2. Video stitching

3.2.1. Stitching software

The original plan for Pilot 1 was to use VideoStitch to merge the footages of all cameras in the rig to a 360-degree video. However, Videostitch studio software was only available for Windows and could not be integrated into the MacOS-based pre and post-production framework of Lightbox. As a result, Lightbox resorted to a software from outside the ImmersiaTV consortium, named Kolor Autopano (www.kolor.com/autopano/).

3.2.2. Stitching errors

While stitching errors occurred with approximately the same frequency in every scene of pilot 1, they were far more visible in sensitive areas such as foreground objects and complex edge patterns. Since not all scenes are as easy to stitch and render, there should be given clear guidelines from the post-production side to the pre-production side about the scene composition, camera calibration, camera position, etc. For example, the camera operator should be aware that the parallax effect influences the stitching performance and can create artifacts (ghosting). Parallax is defined as the difference in alignment between two objects, when observed from different viewpoints. The parallax effect is happening when the images are not taken from the same place. The parallax can introduce stitching errors (by making seams more visible), especially when the objects are close to camera.

In case of static scenes, it is important to think about the location of camera to get less stitching errors. For example, in Figure 5, the stitching errors are visible in the pattern of the goal net. Moreover, this part of the scene has a large depth of field (the net, the ball and the background are in different depths), which complicates the stitching process even more. The stitching errors would have been less visible if the cameras were reoriented so that the main parts to be stitched were grass patches.

Another solution is to protect certain areas in a scene using a mask, such as the face of the football player in Figure 5b.

Unfortunately, relocating the camera rig or applying masks may not solve stitching problems when an object is moving or the scene is dynamic due to camera movements. This is, for example, the case for the crisp bridge discontinuity in Figure 5d. However, as the bridge is a part of the background, the stitching error could have been made less visible by applying a local blurring filter or another blending technique.

Besides a smoother blending of the different segments of the scene, one should always check that the colour information of the stitched fragments match. In Figure 5c, there is an abrupt change of colour saturation which was caused by different exposures of the cameras. The differences in exposures should have been compensated for. The VideoStitch Studio[3] software enables automatic exposure compensation/correction and it can be a solution.

[3] http://www.video-stitch.com/studio/

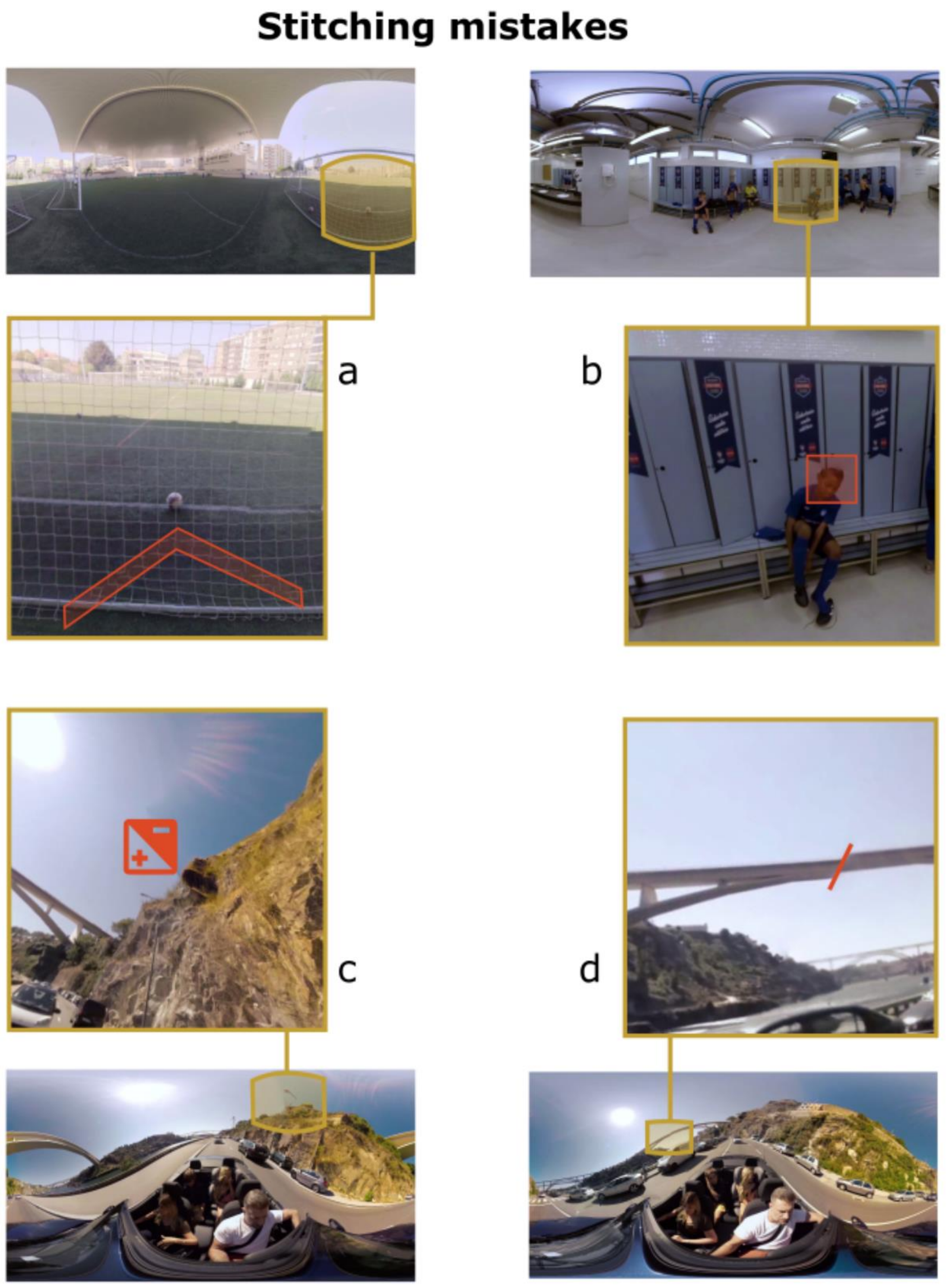


Figure 5: Stitching mistakes. Video frames and zoom-in patches for better visualization of the stitching errors. (a) Error in football net. (b) Error in person face. (c) Colour mismatch of stitched fragments. (d) Bridge discontinuity.

3.3. Video editing

3.3.1. Workflow

After video capture and stitching, offline video editing is performed. The video editing consists of a set of tools and plugins with the functionality of synchronization of multiple 2D videos, omnidirectional videos, and auxiliary data. These data come from the Video Acquisition, and Stitching blocks. Video editing, in general, is a complex process with many stages. In the project, typical media creation is extended by adding new technical possibilities, however they also impose some restrictions on the content creation process. All of the required additional functionalities are implemented and added to Adobe Premiere Pro as a set of plugins.

There are three main stages added to the standard editing workflow:

- synchronization of media for different output destinations
- defining portals/transitions/interactive points
- exporting to different output formats

By adding three stages to the typical content edition workflow, separate processes for each device (TV, tablet, HMD) are merged into one bigger process.

3.3.2. User friendliness of the Adobe Premiere plugin

The user friendliness issues of the plugin can be summarized as follows:

- Plugin Adobe Premiere is slow which makes rendering slow for post-production

Plug-in drains too much power from the machines. Small edits take huge time.

- Design of scene transitions not yet intuitive enough

The Adobe Premiere plug-in has improved significantly from its inception. At first, the plug-in only allowed to add static portals on an omnidirectional content. These portals offered the possibility of combining and synchronizing 2D contents with 360 videos. Subsequently, interactivity functionality was added to the portals offering, for instance, the possibility of making a transition from one scene to another or making a portal appear or disappear.

In addition to the features discussed above, currently, this plugin offers the possibility of adding input and output transitions to the different scenes, making the effect much friendlier.

Finally, this plugin allows the content creator to select what content they want to export from the project and for what devices they want to be available, therefore speeding up the edition of multi-platform content.

A preview of an edited scene can be observed in the Program Monitor window. Selecting the Portal effect in the Effect Control window enables overlay in the preview that visualizes parameters of a portal and allows their direct modification.

3.4. Compression

Compression is required in order to allow efficient streaming of the content after stitching and video editing. In the first iteration (offline encoding with minimal restrictions), ImmersiaTV adopted H.264 as the choice codec due, in part, to the ubiquitous availability of compatible decoders.

In pilot 1, many compression artifacts were noticeable in the 360 videos played on both HMD and Tablet. These artifacts include blocking, colour artifacts, ringing, and blurring.

Blocking artifacts are rectangular blocks in highly compressed video material. They occur mostly in fast motion sequences or quick scene changes. Figure 6a shows clearly visible blocking artifacts in the classroom scene of pilot 1. The artifacts on the right side are marked in red using an edge detector.

Compression artifacts in the chroma color channels lead to distorted colours. In the metro station scene, the footballer's arm appears to be partly green (Figure 6b).

Ringing artifacts are distortions near sharp transitions in a signal. They are caused by a loss of high-frequency information. Visually, they appear as bands or "ghosts" near edges. The breakfast scene in Figure 6c contains many ringing artifacts, which are marked in red on the right side using an edge detector.

The high compression ratio also causes sharpness defects and loss of details. The foreground objects in Figure 6d are blurred and some details only become more visible after sharpening, as shown on the right side.

The decrease in resolution leads to a loss of important details in the video. One of the problems is that the equirectangular projection magnifies less important parts of the scene such as the floor and ceiling and reduces the size of objects in the center of the scene. By reducing the resolution in such a projection, many foreground details will be removed. A region-aware projection and remapping (e.g. pyramid or half-back cubic) before encoding can mitigate this problem. To preserve more details, a region-aware remapping can be used to put more emphasis on important parts of the scene. However, the region-aware method impacts perceived visual quality. So, it was decided to use equirectangular projection which is not region-aware.

3.5. Distribution

In Pilot 1, due to Dash server limitations, it was necessary to use limited resolution for the content delivered to HMD and Tablet.

The IBC Congress (September 2016 Amsterdam) demonstration contained a 360 video (1080p, 8Mbps), TV scene (1080p, 3Mbps) and portals (640x360, 1.5Mbps). The videos are encoded at 25 FPS (frames per second), 25 GOP (Group of pictures). The segment length of the MPEG-DASH content is 3 seconds.

The next demonstration was shown in the NEM Summit (Nov. 2016 Porto). The demo contained 360 video (1080p, 3.7Mbps), TV scene (1080p, 2.7Mbps) and only one portal (340x180, 400Kbps) in the tablet and the HMD (Head-mounted display). The contents were encoded using the same parameters used at the IBC congress.

The latest, more recent demonstration setups tested in the laboratory aimed to improve the quality of the videos. We have already run a 360 video (2560x1440, 4Mbps), TV scene (1080p, 2.7Mbps) and only one portal (640x360, 600Kbps) in the tablet and the HMD (Head-mounted display). The technical test videos are encoded using the same parameters used at the IBC congress. The recent test setup delivers a better experience because the spherical videos are UHD. These latest technical tests have not been evaluated with end-users.

In Figure 7a the faces of the main actors in the scene are not well recognizable and details are lost. In Figure 7b the texts on the board in the center of the class are not readable.

To improve the end-user experience, the resolution should be increased in the next pilot.

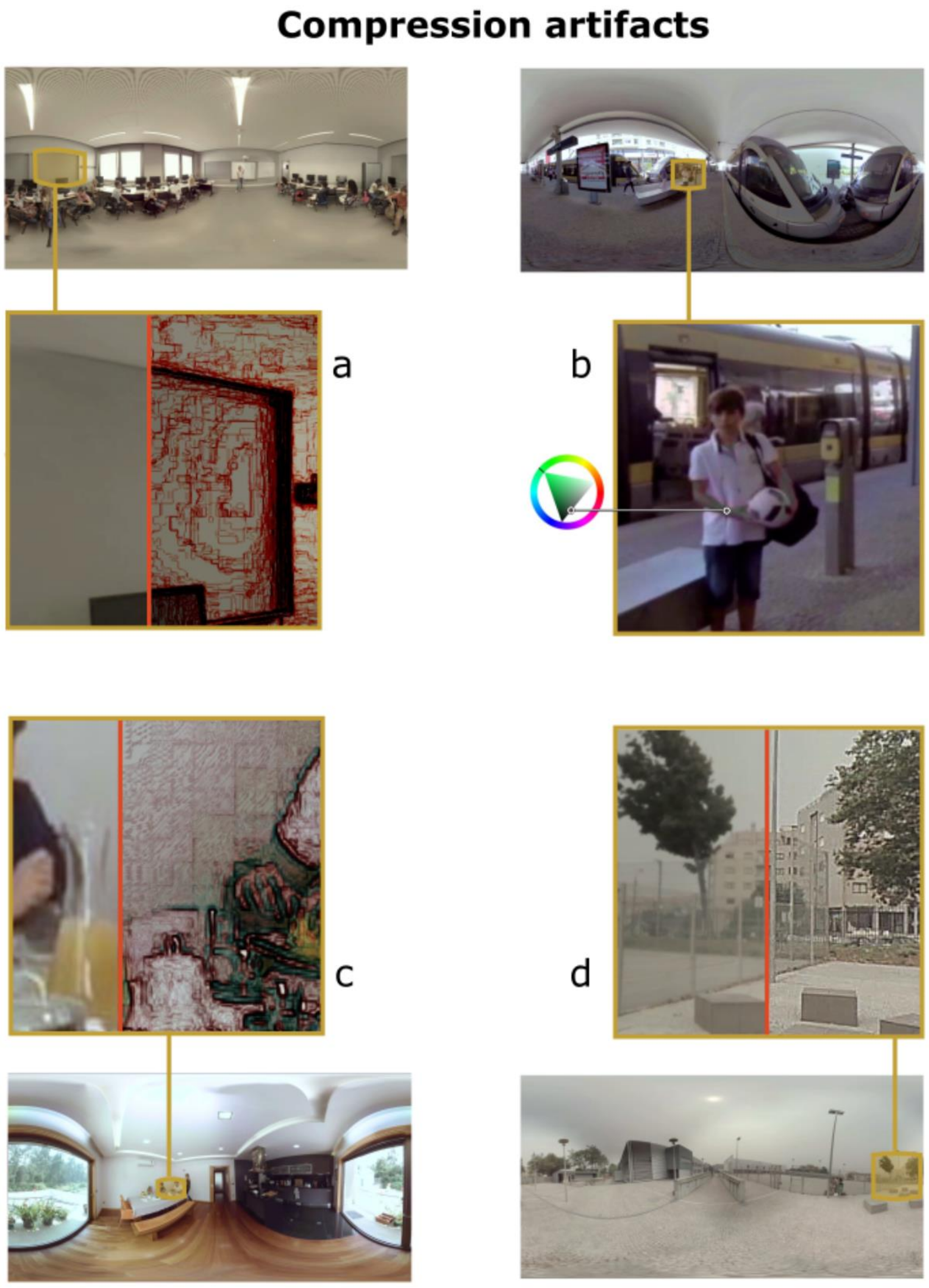


Figure 6: Compression artifacts: (a) A video frame and zoom-in patch for better visualization of the blocking artifact. (b) A video frame and zoom-in patch for better visualization of the colour distortion. (c) A video frame and zoom-in patch for better visualization of the ringing artifact. (d) A video frame and zoom-in patch for better visualization of the blurring artifact.

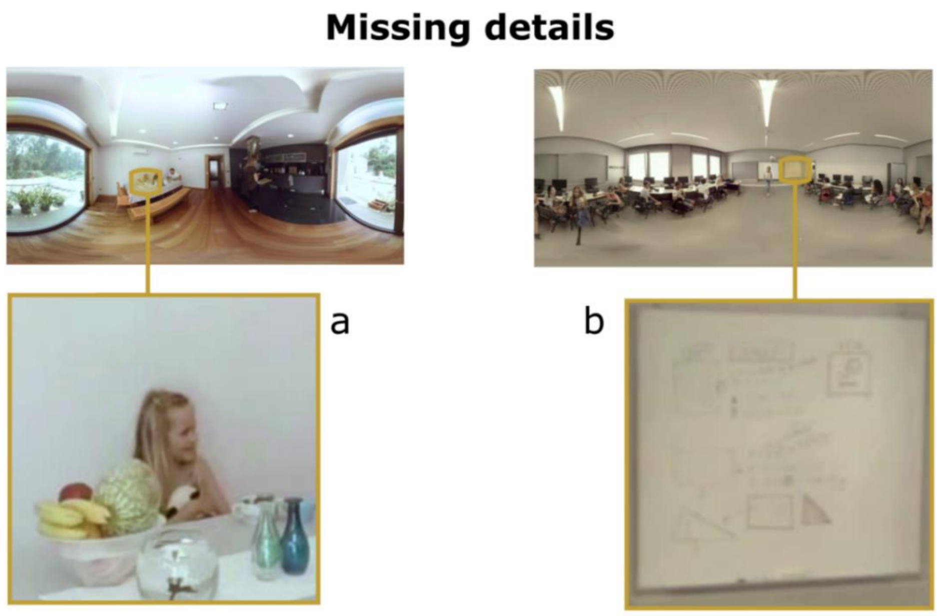


Figure 7: Missing details: The face details in (a) and. the texts in (b) are missing.

Tests: different encoding settings, different frame rate

The table 3 shows the characteristics of the videos that have been used to perform the tests.

	360 Video			TV			Portal		
Test vectors	**Resolution**	**FPS**	**Bitrate (Mbps)**	**Resolution**	**FPS**	**Bitrate (Mbps)**	**Resolution**	**FPS**	**Bitrate (Mbps)**
1SP_1080p_TV_1080_1PT_360p	1024x512	25	2.3	1024x512	25	2.9	426x240	25	0.638
1SP_2K_TV_2k_1PT_360p	2560x1440	25	3.7	2560x1440	25	3.5	426x240	25	0.638
1SP_4K_TV_4k_1PT_360p	3840x2160	25	3.7	3840x2160	25	4.3	426x240	25	0.638

Table 3: The characteristics of the test videos

Below we present the table containing the refresh rate of the graphics card for different devices and for different playback modes (tablet, TV or HMD). The results are presented in frames per second. They clearly show how the HMD mode requires considerably more demanding resources, as well as which devices can reasonably handle the test videos, and which cannot handle properly.

	SamsungGalaxy S7 (Tablet mode)	**SamsungGalaxy S7 (HMD mode)**	**SamsungGalaxy S6 (Tablet mode)**	**SamsungGalaxy S6 (HMD mode)**
1SP_4K_TV_4k_1PT_360p	29.9-33.1	18.2-24.7	15.2-20.0	12.1-13.5
1SP_2K_TV_2k_1PT_360p	38.9-57.9	27.3-33.4	25.4-31.2	16.5-20.7
1SP_1080p_TV_1080_1PT_360p	56.7-61.8	35.6-45.7	26.6-35.1	22.2-30.8
	Nexus 7 (Tablet mode)	**Nexus 7 (TV mode)**	**SamsungGalaxy S6 (Tablet mode)**	**SamsungGalaxy S6 (HMD mode)**
1SP_4K_TV_4k_1PT_360p	Not supported	Not supported	Not supported	Not supported
1SP_2K_TV_2k_1PT_360p	Not supported	Not supported	7.9-10.2	12.1-16.8
1SP_1080p_TV_1080_1PT_360p	9.0-12.2	10.1-13.5	14.4-17.7	24.0-29.3
	Android TV	**Pixel C (Tablet mode)**	**Pixel C (TV mode)**	
1SP_4K_TV_4k_1PT_360p	7.1-10.6	3.2-4.2	3.9-4.6	
1SP_2K_TV_2k_1PT_360p	24.5-28.7	7.2-8.7	8.8-9.6	
1SP_1080p_TV_1080_1PT_360p	44.8-48.3	8.9-12.0	12.1-15.2	

Table 4: measurements per second

3.6. Reception and interaction

In the on-demand scenario of pilot 1, videos were published on an HTTP server. After connecting to the server the player presented a list of available content. Each content included versions of video dedicated for different devices – directive video for TV, omnidirectional videos for tablet and HMD. Users could start playout by selecting one position from the list. All the connected devices (TV, tablet, HMD) were presenting the same selected content (the same story), but each of them in a version dedicated for that device. On HMD users could freely select a direction to look at by movement of the head. On tablets they could do this moving tablet around or on the touch screen.

The main concern for pilot 1 was synchronization between HMD, TV and Tablet. In the unstable network environment encountered during the demonstration in IBC (Amsterdam), the streams sent to HMD and tablet were delayed and easily got out of sync with the TV. The synchronization problem could not be further investigated when it occurred and the temporary solution was to reduce the bitrate of the transmitted signal and number of receivers. Later tests and developments have helped to improve this aspect, resulting in proper synchronization during the pilot in NEM Summit (Porto). For pilot 1 initial synchronization between HMD, TV and Tablet took several seconds, but as soon as it was achieved no further synchronization issues were observed. It is now possible to synchronize much faster – in less than a second.

Solving synchronization issues allowed focusing on increasing resolution of content playout. It was also a key point for presenting multiple video streams on the same device e.g. additional directive view composed into omnidirectional view. Lack of synchronization between videos would be even more disturbing than on separate devices. During pilot 1 it was presented in a limited way - only one video insert at a time was visible on Tablet and HMD. The interaction was limited to switching video inserts on and off.

For the next Pilot, we need a more durable solution based on adaptively changing the bitrate of the signal based on the quality of network connection using the DASH protocol. Currently, the DASH player is not yet configured to switch between different quality presets.

4. LESSONS LEARNED FROM PILOT 1

4.1. Guidelines

According to the observed technical issues in Pilot 1, a number of guidelines are provided to improve performance of different stages of ImmersiaTV platform.

4.1.1. Shooting

The guidelines for shooting are summarized as follows:

Never shoot separately omnidirectional and directional scenes meant to be synchronized.

Shooting the scenes independently is problematic (for instance, first with the 360 Rig and next with the directional cameras). The problem is that adequate synchronization will never be possible because the acting (movements, moving objects, timings, etc.) will never be exactly the same between takes. As well, it is important to use same frame rate for all cameras during the capturing process.

The best way to shoot a documentary with omnidirectional and directional cameras simultaneously would be to have some micro 2k cameras around the set, hidden by props in the set design. This approach still necessitates some rotoscoping of the micro cameras to remove them from shot but this will be easier than removing an entire crew with filming equipment.

Put the directional cameras and crew in an area where there will be no interaction.

Another good approach is to put the directional cameras and crew in an area where there will be no interaction of characters or moving objects. In this case, another take of the same scene without the crew and directional cameras suffices to be able to remove them in post-production.

Pick up the tripod holding the 360 rig at the beginning of each take and rotate it 180 to 360 degrees, making regular pauses.

A good method to apply on the shooting set is to pick up the tripod holding the 360 rig at the end of each take and rotate it in 180 to 360 degrees, making regular pauses. Later in post-production, this movement of the cameras on the rig will give a clear idea if the cameras are synchronized correctly (by aligning the motion of all the captured videos). In addition, the pauses between rotations will image the same scene by each camera sensor, and will give more opportunities to correctly calibrate the rig.

Use a live preview system to review the scene composition before shooting.

It is recommended to use a quick or live preview system, such as an inexpensive camera like the Ricoh Theta S, or a better rig connected to Vahana VR, to let the director have a preview of the scene composition and the placement of actors: spherical video has no concept of “framing” or “zooming on” a subject since the entire sphere is covered at once. The director no longer has this liberty to guide the users’ attention. If stitching process in a post-production step does not made on location, and the composition of the scene is not suit the director’s intent, the take has to be repeated, however, this is not always possible, due to time and resource constraints.

4.1.2. Video calibration and stitching

- **Carefully pick up a calibration scene:** For automatic calibration (using VideoStitch Studio or Vahana VR for instance), it is better to choose a footage sequence with suitable content. This means: Choose a sequence that includes enough details (if a camera is shooting 100% the sky, it will be hard to calibrate). Moreover, check a sequence where there are no close objects in the video.
- **Calibration on several frames:** VideoStitch Studio enables calibration on several frames, if the calibration on a sequence failed.
- **Rotate your rig and launch the calibration again:** By rotating the camera, new details can be obtained on the overlapping part of the images, which will enable VideoStitch Studio or Vahana VR to detect new control points. The newly created control points will accumulate with the previous ones providing a better chance to get a successful calibration.
- **Using an external calibration software such as PTGui:** If the automatic calibration is still producing unsatisfactory results, one solution is to export individual frames from the input video and use an external calibration software, where users can interactively select and review control points. PTGUI is a helpful software tool to make a more perfect stitch of the camera rig output. Especially for parallax issues between cameras.
- **Move camera rig at the beginning of the take:** By moving the camera rig at the end of the take, the VideoStitch software gives a more accurate result on the synchronization, by aligning the motion of all the captured video. By changing the position of each video on the timeline the synchronization can be further optimized before starting the stitching process. This way, and finally, we will have the perfect synchronization to finally move to the stitching.
- **Keep the actors at least three meters from the camera rig:** Stitching solutions are still in their infancy, e.g. minor parallax problems are difficult to avoid for now. Keeping the actors at least three meters from the camera rig can leave these stitching and parallax problems largely unnoticed, by making a perfect stitch for the actors and ending up with some bad stitching in unimportant areas with no character interaction in the scene. Minor parallax stitching issues can be resolved in AfterEffects and Photoshop. Such issues can be corrected by painting out what we don’t want to be seen and fixing the parallax problems.
- Stitching issues are more severe on certain locations (such as faces, patterns, etc), so it is more efficient to protect such areas by performing a conscious stitching.

4.1.3. Post-production

A good communication between preproduction and postproduction sides is needed because preproduction setups (camera adjustments, scene details and object distance from the lens, etc) can influence the complexity of postproduction.

4.1.4. Video compression, transmission, and reception

- Adaptive streaming can help to mitigate synchronization issues in case of an unstable network.
- A conscious remapping of the equirectangular projection needs to be done to avoid missing details. Some regions of the scene contains visually important details (for example the face details of an actor or texts, etc), while they occupy a very small portion

of the scene in an equirectangular projection. Therefore, it is important to detect such areas and perform an adaptive projection to preserve such areas from sever distortion.
- Higher resolution is needed particularly in omnidirectional videos to improve visual quality. Frame rate increase from 25Hz to 50Hz is recommended on HMD and Tablets.

4.2. Desirable features

In addition to the aforementioned guidelines and suggestions, some new features can be included in the ImmersiaTV platform to increase the end-user satisfaction:

4.2.1. 3D immersive audio

The 3D immersive audio recording is meant to reflect the way we receive sound in real life, creating rich soundscapes you would experience if you were actually there.

In past decades, 3D audio recording was a novelty which was utilized for less technically demanding methods. But with the rise of virtual reality hardware like the Oculus Rift, Sony's Morpheus, and Samsung's Gear — systems dependent on realistic 3D audio to fully immerse their users — 3D audio can have significant influence to increase the immersive experience. 3D immersive audio is becoming an important tool in virtual reality development.

4.2.2. Stereoscopic content

By generating stereoscopic content, the illusion of depth is created which helps for better immersive experience. However, there may be some challenges to use Stereoscopic 3D in VR. An important issue is to have a proper geometric arrangement of stereo cameras. When moving a camera rig, the left and right camera must be horizontal to each other. Any small stitching misalignment (error) magnifies in 3D which leads to visual discomfort for end users. Inaccurate implemented 3D immersive video footage can cause a great discomfort to the viewer including eye strain or nausea.

The stereoscopic omnidirectional video can be achieved by providing two projected views for left and right eyes. The perspective projections will carry horizontal parallax effects that the human brains will analyze as depth cues, similar to conventional stereoscopic videos. However, 3D omnidirectional content introduces critical issues:

The camera rigs that capture such omnidirectional scenes use several sensors. Due to the physical occupancy of the sensors and lenses, the sensors do not share the same centers of projection, and view themselves the scene under some parallax. This parallax results in stitching errors (resulting in broken objects and ghosting effects). In monocular omnidirectional videos, such stitching errors are just seen as visible impairments impacting the quality, but without further consequence. In stereo omnidirectional videos, the stitching errors may not be identical and coherent between the left and right eyes. These incoherencies cannot be analyzed properly by our human brain, and result in visual discomfort and fatigue. Remedying to these inconsistencies requires a lot of effort in postproduction and manually editing the stitching masks to avoid artefacts. This kind of postproduction is beyond the scope of the project for the offline pilot, and impossible for live pilots.

4.2.3. High Dynamic Range

High Dynamic Range Imaging (HDRI) has recently gained attention and it is affecting almost all fields of digital imaging. HDRI overcomes the limitation of traditional imaging by performing operations on color data with much higher precision. HDRI can represent all colors of the real world close to what can be perceived by the human eye. Today many state-of-the art video game engines perform rendering using HDR precision to provide more believable virtual reality imagery.

Enabling HDR needs provisioning new capturing and visualization devices on the ImmersiaTV platform. It also needs to redesign the encoding system to support for HDR content compression. However, the main bottleneck is that the HMD and mobile phones/tablets are not able yet to visualize HDR content and tone mapping methods are required to convert HDR to SDR. The HMD and tablets needs to support higher brightness and contrast ratio in future to enable HDR omnidirectional video visualization.

5. STATUS OF USER REQUIREMENTS IN PILOT1

In the Deliverable D 2.3[4] - Content ideation, production scenarios and requirement analysis, four general user scenarios (US) of pilot1 with their specific requirements have been identified. This section summarizes the status of the requirements mentioned in D 2.3 (section 4).

Each requirement in the table is coloured by one of colour codes (green, yellow, and red) indicating how well the requirement is addressed:

The *green* colour implies that the requirement is fully addressed, *yellow* means the item is addressed but it can be improved and *red* shows that the item is not addressed.

As it is shown in the table, most of the requirements are addressed. The relevant requirements which are essential for operation of the ImmersiaTV platform are already enabled, however, there are some additional features that could not be addressed or they will be fulfilled in pilot 3. Among the requirements that are not fully addressed, R-EDIT-10 and R-EDIT-11 are the most relevant issues, as they limit possible visualizations.

US1. Story Preparation

Requirement Description	Current Status and Future Steps
🟩 **R-STORY-1** The content creator can create the main storyline.	It is possible to create the main storyline. However, this new narrative paradigm requires experimentation and reflection in order to improve this particular point.
🟩 **R-STORY-2** The content creator can define the main and side characters.	This feature depends mainly on the last point and how one makes use of Immersia's unique features.
🟨 **R-STORY-3** The content creator can define the detailed story structure.	Due to the lack of information and knowledge in terms of narrative creation with this specific medium, this is still up for debate. Further experimentation will show how detailed one can craft a story within these constructs.
🟥 **R-STORY-4** The content creator can define the sub-storylines for TV, tablet and HMD.	At this moment, this features shows the highest potential. Crafting nonlinear and time continuum disrupting narratives may be the key to help us understand this.
🟨 **R-STORY-5** The content creator can define the user interaction design.	In terms of the plugin, this is still being worked. Hopefully in the near future we will be able to test it with test audiences to better understand how to craft a better interaction.
🟩 **R-STORY-6** The content creator can define the multi-platform logic.	It is possible to define the multi-platform logic, when it is correctly implemented with the story.

[4] http://www.immersiatv.eu/wp-content/uploads/2015/11/D2.3-Content-ideation-production-scenarios -version-0.9.pdf

R-STORY-7 The content creator can define the viewer perspective(s).	It is possible to define the viewer perspective(s). One of the best working aspects at the moment.
R-STORY-8 The content creator can define the detailed script.	The script must contemplate all aspects beforehand, otherwise, the risk of lack of narrative material is substantial. This might create a somewhat “dry” experience.
R-STORY-9 The content creator can define if it uses omnidirectional and/or directive content.	The feature is working perfectly at the present moment. Some technological improvement is still required.
R-STORY-10 The content creator can define the viewing angle.	The feature is working well. Good exercise on points of view for any given narrative.
R-STORY-11 The content creator can define the interaction points: portal, AR object, caption, graphics, …etc.	Currently, the main issue is how counter-intuitive the organization of this material has to be conducted within premiere pro. Looking forward to, through collaboration, getting this point as optimized as possible.
R-STORY-12 The content creator can define transition between scenes.	At the moment, we can’t give exact feedback whether this feature is correctly working or not.
R-STORY-13 The content creator can specify use of audio for guidance and transitions.	The feature is working correctly. It plays a very important role. However, it’s currently not viable.
R-STORY-14 The content creator can indicate use of camera movements.	Presently, camera movement on the omnidirectional is still something we doubt it works correctly. Further testing will be conducted during Pilot 3.
R-STORY-15 The content creator can indicate forced exploration mode where applicable.	Not available, it is in the scope of pilot 3.
R-STORY-16 The content creator can save resulting format script as master template.	Not available.

Table 5: Status of the requirements for story preparation (US1)

US2. Production Preparation

Requirement Description	Current Status and Future Steps
R-PREPROD-1 The content creator can add on-location research material as placeholder in format script.	The feature is possible now.
R-PREPROD-2 The content creator can perform first VR preview of relevant scenes.	It is possible to perform first VR preview. It allows the content creator to understand if the location is viable for 360º shooting.
R-PREPROD-3 The content creator can define the shooting plan.	It depends on the ability/possibility of articulating how to tell a story with those three platforms.
R-PREPROD-4 The content creator can define the VR/directive capturing strategy.	The content creator can define the VR/directive capturing strategy. However, some field testing has yet to be done to fully understand the extent of such strategies and their specific applications.

Table 6: Status of the requirements for production preparation (US2)

US3. Edition and Compositing

Requirement Description	Current Status and Future Steps
R-EDIT-1 The content creator can visualize the raw material across the different end-user devices.	The raw material can be visualized across end-user devices.
R-EDIT-2 The content creator can use a standard edition software (Adobe Premiere, Final Cut, or other), and avoid, for simple projects, using advance compositing software.	Using Adobe Premiere is enough for standard edition.
R-EDIT-3 In the editor software, the content creator can edit content for TV and for omnidirectional video in such a way that the timings of the content for the 2 targeted devices is visible constantly	TV and omnidirectional videos are separate tracks on a common timeline.
R-EDIT-4 The content creator can use Windows and OS X	The software versions for Windows and OS X are available.
R-EDIT-5 The content creator can use of an advanced mode in a compositing software (Nuke, Adobe After Effects).	Advanced mode in the editing software is not addressed directly, but it does not prevent user from enhancing content with tools other than Adobe Premiere.
R-EDIT-6 The content creator can introduce interactivity within the editor timeline through conditional transitions between shots and scenes.	The portal effect plugin allows interactivity.

R-EDIT-7 The content creator can select, within the editor timeline, which video assets are visible within the TV, the tablet and the HMD.	Creator selects the target device for video assets, by naming the video tracks.
R-EDIT-8 The content creator can also create ImmersiaTV scene typologies, i.e., interaction between devices, through conditional transitions within the editor timeline.	There are no interaction between devices currently, different scene typologies can be created.
R-EDIT-9 In pilot 1, the end user will experience the content with a common timing between devices (HMD, TV, tablet), it will be continuous and have no jumps.	The content for pilot 1 is continuous, with common timing for all devices.
R-EDIT-10 The content editor, using either a classic video editor or an advanced one, will easily define transitions between omnidirectional videos using black and white video MATTE.	Transitions are implemented as Adobe Premiere plugins with corresponding shaders on player side. Defining transition as black and white video matte is not available.
R-EDIT-11 The content editor will be able to add a beauty layer to the interactive transition which, unfolding synchronously with the black and white video matte, will add borders and eventually other visual content needed for the transition.	Not available.
R-EDIT-12 The content creator will allow seeing omnidirectional content both in projected and non-projected views by using Previsualisation tools integrated in the content editor.	Previewing in projected and non-projected mode is a built-in feature of latest Adobe Premiere Pro.
R-EDIT-13 The content creator will be able to visualize transitions and interactive transitions will be visible within the editing software.	Transitions are rendered in preview window.
R-EDIT-14 The content creator will be able to visualize synchronized playout between 2 devices, for example, to see how TV and HMD content fit in timing.	The feature is not addressed directly, but should be possible using built-in Adobe Premiere features.
R-EDIT-15 An export button will generate a set of videos and metadata that is ready to distribute content across devices.	Simple export panel allows to generate videos and metadata ready to be played in the player application.
R-EDIT-16 The export functionality will accept sequences involving different aspect ratios, due to differences in omnidirectional and traditional video formats (most likely solved through nested sequences).	The export functionality takes into account aspect ratios.
R-EDIT-17 The common cutting points between devices will be visualized putting the content for the different devices in 2 sequences, one on top of one another.	Content for different devices is edited in a common sequence, on different tracks, which allows to visualize common cutting points.

R-EDIT-18 It will be possible to define a label specifying the destination for each sequence.	Creator selects the target device for a sequence, by naming the video tracks.
R-EDIT-19 The outcome should be: 1) A set of videos in the highest resolution possible. The videos should have a shared timestamp. This means that the timestamp introduced at the frame level is common to all the different fluxes. For example, the first frame of a video introduced exactly at second 12 of the broadcast should have its first frame with a timestamp set at 12. 2) A metadata file detailing how the different videos have to be organised to compose an omnidirectional scene. This file should be compatible with broadband distribution standards.	The outcome of export is: - video files in highest possible resolution, with shared timestamp - metadata describing relations between videos, compatible with DASH

Table 7: Status of the requirements for editing and compositing (US3)

US4. Content Playback

Requirement Description	Current Status and Future Steps
R-PLAY-1 Basic controls. The basic controls of the player will be: Select media source, Play, Stop, Select tablet or HMD mode.	The media source can be selected from the player, current situation is that the user has to stop every device in order to playback a new content. The HMD or tablet behavior can be selected by going back to device selection screen.
R-PLAY-2 The player will process metadata to describe and define the scene: The information regarding how the scene is composed must be distributed to the player. It must include information like which videos are visible and where are they placed or how are they composed.	The player can process the metadata and is distributed from the same place the contents are originated. The metadata indicates the size, position, start time and duration of the scenes.
R-PLAY-3 The scene is device dependent. Each type of device will have to render a different scene, as the interaction with the user will be different.	There are three types of scenes per type of device (HMD, tablet and TV).
R-PLAY-4 Render multimedia content over textures and 3D objects. One or several videos will be displayed in different positions over the 3D scene (over a spherical surface, as a regular 360° video, or over plain surface in a mirror or portal like effect).	The videos are rendered in different parts of the scene, managed by Unity3D game engine.

R-PLAY-5 Apply video masks in videos. A mask is needed to overlay more than on video over the same texture forming an overlay of an arbitrary shape (i.e. to render a portal as a circle over the 360° sphere).	The masks are now rendered from luma matte videos. For next pilot they will be generated from predefined transitions and applied from the player.
R-PLAY-6 Interaction management. There needs to be a systematic way to define interaction mechanisms in the end-user devices, and the methods implementing such interaction mechanisms need to be made available to the content creator.	The interactions are defined in the Premiere Pro plugin with the portal effect plugin.
R-PLAY-7 Achieve a frame level precision: This is relevant as devices can display different omnidirectional and directional contents that were shot together, so any sort of desynchronization is going to be noticeable by the user.	Frame accurate synchronization has been achieved.
R-PLAY-8 The devices may need to synchronize to any base media time at start up: A device can be turned on when there is already the reproduction going on in another device, so the one joining the group must get synchronized without affecting the other ongoing reproductions.	When a device joins the common session it synchronizes to the playback point of the others (previously synchronized).
R-PLAY-9 Basic audio control in the end-user devices	Stereo audio is available and can be muted from the device.
R-PLAY-10 Real time communication channel between devices: It will be needed to send messages from one device to another.	There is no mechanism for inter-device messaging currently.
R-PLAY-11 Second screen scene definition: The definition of the second screen view (mosaic layout) in the tablet must be defined within the content production process.	The mosaic layout is not available yet, it will be implemented for pilot 2.
R-PLAY-12 The end-user can capture screen casts and share them with other devices.	Not available, it will be implemented for pilot 3.
R-PLAY-13 The end-user can capture screen casts and share them through social media.	Not available, it will be implemented for pilot 3.

Table 8: Status of the requirements for content playback (US4)

6. IMMERSIATV PLATFORM OVERVIEW IN PILOT 2

The Phase 2 of ImmersiaTV development aims to implement and employ tools that demonstrate Pilot 2 and adapt the framework for running the live scenario. The architecture of the Pilot 2 is shown in Figure 8.

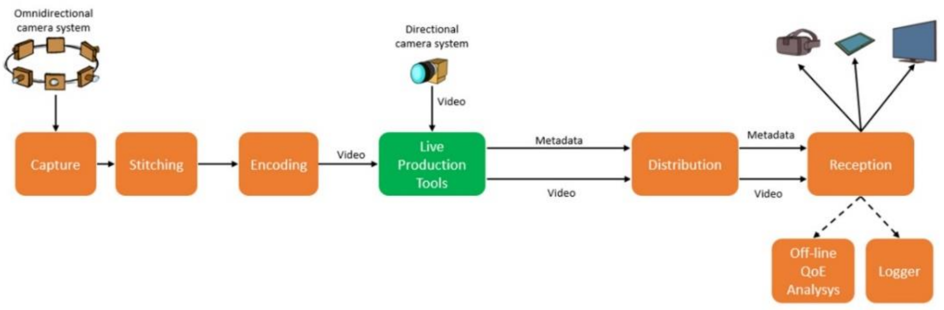


Figure 8: ImmersiaTV architecture in Pilot 2**.**

In Pilot2, several cameras with ability to capture in real-time as well as processing tools for live stitching are employed. The cameras used in this pilot include: Go Pro/Elmo rigs combined with Vahana VR, Integrated "Orah 4i" cameras developed by VideoStitch, iMinds/EDM camera, commercialized in their AZilPix spin-off under the name Studio.One and directional broadcast cameras: Grass Valley LDK 8000. The architecture of Capture modules are depicted in Figure 9 and the technical specifications of cameras and stitching tools are elaborated in D3.1[5] and D3.2[6]. The stitched omnidirectional and directional streams are combined in the live production tool. The tool provides real-time editing and merging of several video files. The outputs of the live production will be transferred to the DASH server, transcoded, and served in MPEG-DASH streamable format to be delivered to end user's devices.

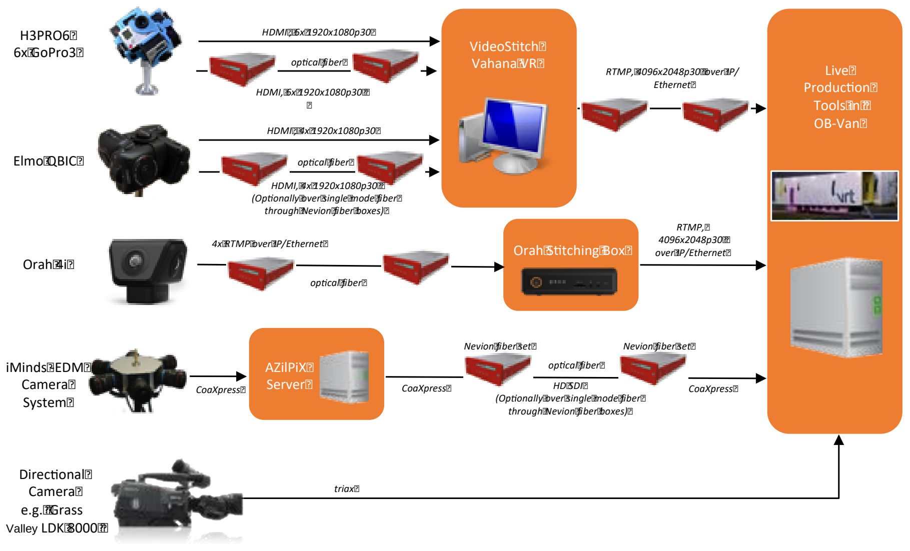


Figure 9: Architecture of the capture modules for Pilot 2

[5] http://www.immersiatv.eu/wp-content/uploads/2015/11/D3.1-Design-Architecture.pdf
[6] http://www.immersiatv.eu/wp-content/uploads/2015/11/D3.2-Capture-components.pdf

7. TECHNICAL SPECIFICATIONS OF TOOLS IN PILOT2 AND PERFORMANCE EVALUATION

This section describes the technical specifications of different tools developed and deployed for Pilot2. The performance is evaluated based on several technical experiments.

The architecture of the system suggests that a dynamic scenery is captured from multiple cameras. Every frame is stitched and edited in order to form 360-omnidirectional video content, which is fed into a multimedia server. The 360-omnidirectional content is mapped to a rectangular frame and encoded. The encoded bitstream is then passed on to the content distribution network (CDN), which re-transmits the content to the receiving devices for decoding, re-mapping and consumption.

The deployed system targets to host both content delivery and live streaming events. In the former case, low end-to-end time delay is a main requirement. Thus, the time allocation of the entire processing timeline should be kept at the minimum. In addition, the computational cost of each module should be evaluated to ensure the flawless functionality of the system. The technical specifications and performance evaluation of modules developed in Pilot2 are explained in the following subsections.

7.1. Production Tool Performance

The ImmersiaTV platform offers a set of tools to extend the production process and enable new forms of storytelling. Cinegy production software package is developed for live editing and broadcasting of omnidirectional content in Pilot 2. The software provides user interface for live VR production and dynamic manipulation of omnidirectional scenes.

The overview of the Cinegy production tool is presented in Figure 10. The application and functionality of the tool are elaborated in deliverables D 3.1[5] and D 3.3[7].

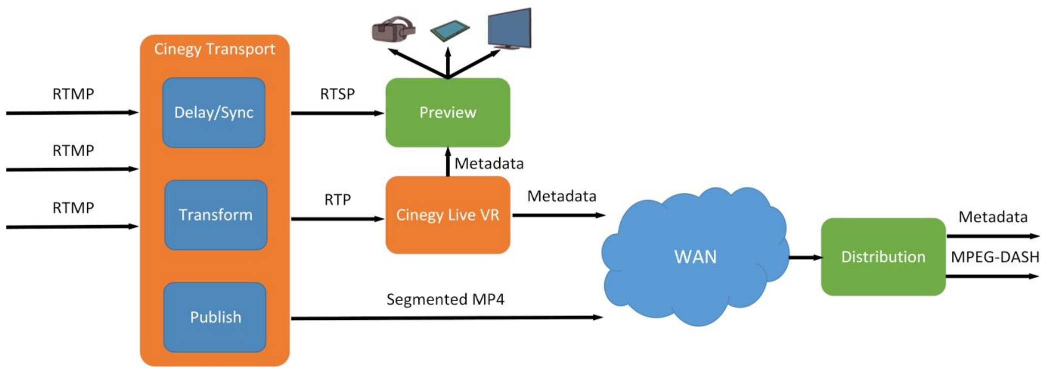


Figure 10: Overview of the live production tool

The performance of the production tool is evaluated through several experiments. The most complex part at production side is related to the encoding of live streams from cameras in order to conform to MPEG-DASH specifications and align to segments boundaries. Therefore, the encoding process in production tool is testified as it consumes the most computational power. Other functionalities do not require complex technical processing and mostly related to the user interface. Benchmarking were done with special internal tools made by Cinegy for encoding scores calculation:

[7] http://www.immersiatv.eu/wp-content/uploads/2015/11/D3.3-Production-Tools.pdf

- Cinescore[8] (available from Cinegy Web site)
- Parallel GPU encodings benchmark (under development)

7.1.1. Benchmarking the performance of H.264 encoding on different graphic cards

The maximum possible performance of the NVIDIA-based H.264 encoding is evaluated on different graphic cards. Four graphic cards with different architectures and price levels were used in the experiment. The specifications of the cards are summarized in Table 9.

		Quadro P2000	GTX 1080	GTX 960	Quadro M3000M
Processor	**GPU Name:**	GP106	GP104	GM206	GM204
	Process Size	16 nm	16 nm	28 nm	28 nm
	Transistors	4,400 million	7,200 million	2,940 million	5,200 million
Memory	**Memory Size**	5120 MB	8192 MB	2048 MB	4096 MB
	Memory Type	GDDR5	GDDR5X	GDDR5	GDDR5
	Memory Bus	160 bit	256 bit	128 bit	256 bit
	Bandwidth	160.2 GB/s	320.3 GB/s	112.2 GB/s	160.4 GB/s
Clock Speed	**GPU Clock**	1370 MHz	1607 MHz	1127 MHz	1050 MHz
	Memory Clock	2002 MHz 8008 MHz effective	1251 MHz 10008 MHz effective	1753 MHz 7012 MHz effective	1253 MHz 5012 MHz effective
Render Config	**Shading Units**	1024	2560	1024	1024
	Pixel Rate	58.80 GPixel/s	110.9 GPixel/s	37.70 GPixel/s	33.60 GPixel/s
	Texture Rate	94.08 GTexel/s	277.3 GTexel/s	75.39 GTexel/s	67.20 GTexel/s
	Floating-point performance	3,011 GFLOPS	8,873 GFLOPS	2,413 GFLOPS	2,150 GFLOPS
Price (EUR)		450	600	200	850

Table 9: Specifications of the graphic cards used in the experiment.

In the first test, the maximum possible frame-per-second (FPS) of H.264 encoding with High Quality profile enabled were collected for three resolutions running on four different graphic cards. Next, the corresponding FPS results were projected to the max possible real-time encodings for the live streams with 30 fps. The test results are shown in Table 10. As it is

[8] https://open.cinegy.com/products/cinescore/10.4/

expected, the FPS is decreased when encoding higher resolutions, the FPS is still sufficient for real-time 4K encoding. The Quadro P2000 and GTX 1080 have delivered higher FPS than the other graphic cards.

GPU Card	**FPS**			**Streams, 30 fps**		
	UHD	**1/2 UHD**	**1/4 UHD**	**UHD**	**1/2 UHD**	**1/4 UHD**
Quadro P2000	135	298	1270	4	9	41
GTX 1080	125	279	999	**2**	**2**	**2**
GTX 960	96	205	807	**2**	**2**	**2**
Quadro M3000M	86	186	742	2	6	24

Table 10: The maximum FPS and possible real-time encodings for different video resolutions using four different graphic cards.

As it is observed in Table 10, due to hardware architectural limitations, GTX cards cannot perform more than 2 parallel encodings even if there is enough processing power available. Quadro cards do not have such limitation thus allowing more than 2 parallel encodings at once. Therefore, the most cost effective is usage of Quadro P2000 type cards that can perform up to 4 x UHD@30 fps encodings on a single unit. The GTX 960 / 1060 cards can be of the same price range but will require 2 slots on motherboard.

7.1.2. Benchmarking of GPU load based on stream processing

The second experiment aims to measure the GPU load when different stream resolutions are encoded at production side. Single card Quadro M3000M was used for tests. Input stream 4k@30 fps was passed to H.264 encoder unaltered (1:1) and downscaled (1:2 and 1:4). Additionally, parallel encodings of the same stream were done to check how load is scaled with the number of streams. Table 11 presents the GPU load for encoding of different stream resolutions.

Resolution/# streams	**1**	**2**	**3**	**4**
1:1	46	92	-	-
1:2	19	39	48	67
1:4	5	9	15	21

Table 11: GPU load of encoding process in production tool.

As indicated in Table, the dependency of GPU load to the number of streams is quite linear and predictable. GPU load dependency based on frame size is also close to linear with some additional overhead added by downscaling of the original frame before encoding. Thus, maximum capacity of the encodings and their configuration can be estimated for the given card type.

7.1.3. Benchmarking of parallel encodings for MPEG-DASH alternate resolutions

To enable MPEG-DASH delivery of streams for different devices with different decoding capabilities, the system generates several stream versions in parallel. Here, we report the GPU load when Input stream 4k@30 fps was encoded and at the same time the alternate stream versions were generated (1:2, 1:4). Single card Quadro M3000M was used for tests. Table 12 summarizes the results.

		Number of streams					
Resolution	**1:1**	1	1	1	1	1	1
	1:2	1	2	0	0	1	1
	1:4	0	0	1	2	1	2
GPU load		67	85	51	57	71	78

Table 12: GPU load for parallel encoding of different MPEG-DASH stream resolutions

The results confirm that it is possible to generate several stream versions at once with small additional overhead for MPEG-DASH delivery to different devices with different decoding capabilities. The graphic card load is quite predictable so recommendations on configuration can be done for a given card type.

7.2. Codec System Performance

Considering that a significant amount of time is usually spent during the encoding of the broadcasting content, the trade-off between time delay, target bitrate and visual quality has to be efficiently addressed. Here, the real-time processing constraints imposed by the selected hardware encoder are reported. Furthermore, based on our experimentation, the resource usage of the coding module for various complexity settings is demonstrated.

7.2.1. HEVC encoder hardware specifications and parameters

In this turn of the project, HEVC technology is integrated in the system. Considering that HEVC brings higher visual quality at the expense of complexity, given the bitrate, the performance of the encoder computing platform is of crucial importance. In this section, an overview of the selected coding module is provided along with its encoding capabilities and features.

To reach quasi real-time encoding, a high-performance hardware-based codec was integrated to the multimedia server who receives the stitched content. In particular, the selected hardware is a Pascal generation GPU of NVIDIA, which provides fully-accelerated video encoding for both H.264 and HEVC. Its performance is independent of the graphics performance and, thus, neither the graphics engine nor the CPU module are loaded with encoding operations. This is particularly useful, considering that both mapping and encoding are performed on the same machine (See Deliverable 3.4[9] – “Encoding and Decoding”).

Two coding engines are physically present on the silicon, as illustrated in Figure 11, and multiple hardware encoding contexts are natively supported with negligible context-switching costs. Thus, an application can encode multiple videos on the same system simultaneously, subject to the hardware performance limit and available memory.

[9] http://www.immersiatv.eu/wp-content/uploads/2015/11/D3.4-Coding-and-Decoding-modules.pdf

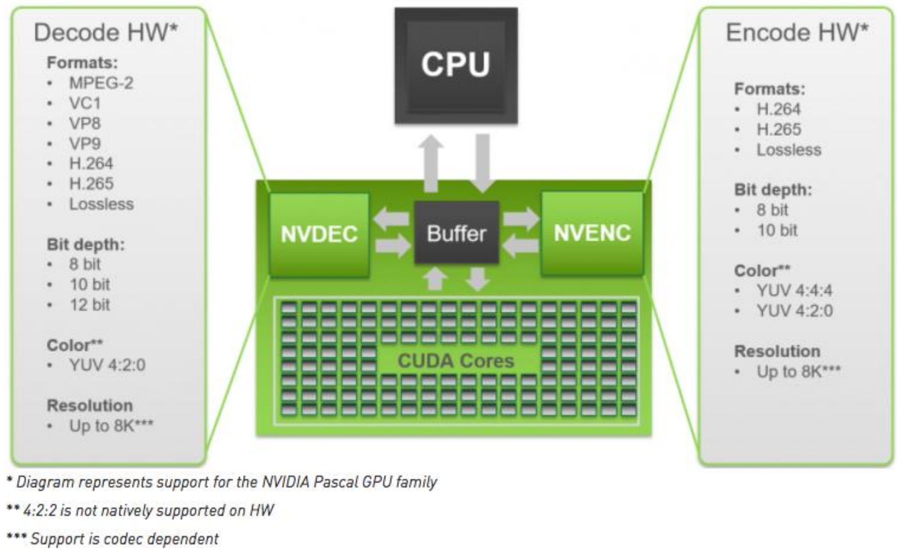


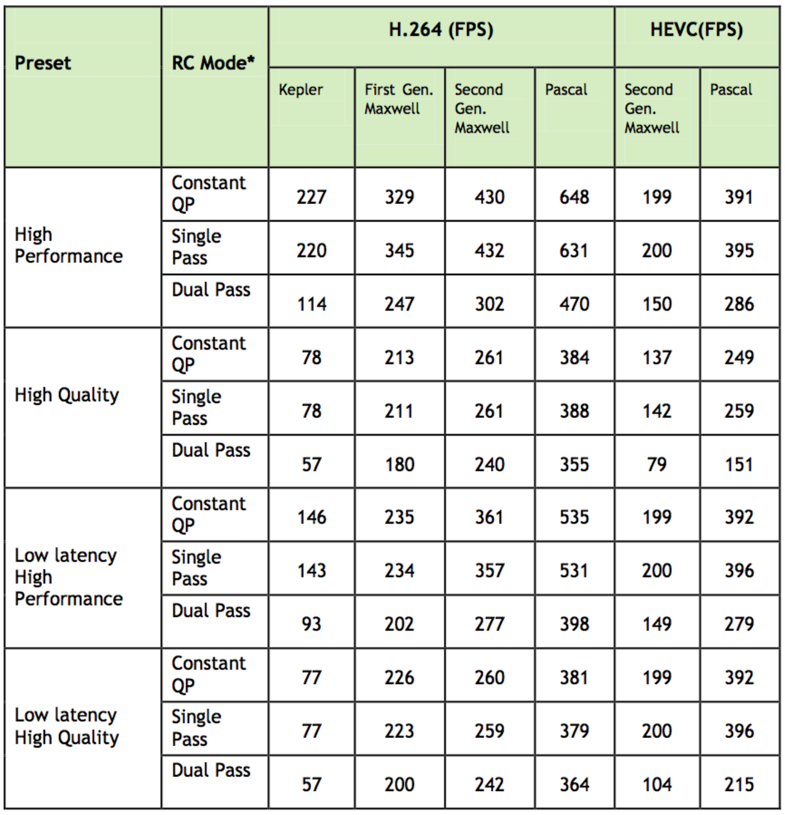
Figure 11: Hardware architecture[10]

The capabilities of the selected hardware encoder are outlined below:

- **H264 Base, Main, High Profiles:** encode YUV 4:2:0 sequence and generate H.264 bit stream
- **H264 4:4:4 Encoding**: encode YUV 4:4:4 sequence and generate H.264 bit stream
- **H.264 Lossless Encoding**: lossless encoding
- **H.264 Motion Estimation only Mode**: provide Macro-block level motion vectors and intra/inter modes
- **Support for ARGB input**: encode RGB input
- **HEVC Main Profile**: encode YUV 4:2:0 sequence and generate HEVC bit stream
- **HEVC Main10 Profile**: support for 10-bit content and generate HEVC bit stream
- **HEVC Lossless Encoding**: lossless encoding
- **HEVC Sample adaptive Offset**: significant improvement of encoded video quality in HEVC
- **HEVC 4:4:4 Encoding**: encode YUV 4:4:4 sequence and generate HEVC bit stream
- **HEVC Motion Estimation only Mode**: provide Coding-Tree-Block level motion vectors and intra/inter modes
- **HEVC 8K Encoding**: support for 8192x8192 resolution content

Various Rate Control (RC) Modes and flags can be set through the SDK that is provided for the configuration of the hardware. A combination of these parameters enables video encoding at varying quality and performance levels. Indicatively, in Table 13, the performance of the hardware encoder is demonstrated under different RC Modes. The input is a YUV video sequence of 1920x1080 resolution and 4:2:0 chroma sub-sampling with 8-bit. The frames per second (FPS) indicates the encoding speed.

[10] This figure is taken by: https://developer.nvidia.com/nvidia-video-codec-sdk#collapseOne



Preset	RC Mode*	H.264 (FPS)				HEVC(FPS)	
		Kepler	First Gen. Maxwell	Second Gen. Maxwell	Pascal	Second Gen. Maxwell	Pascal
High Performance	Constant QP	227	329	430	648	199	391
	Single Pass	220	345	432	631	200	395
	Dual Pass	114	247	302	470	150	286
High Quality	Constant QP	78	213	261	384	137	249
	Single Pass	78	211	261	388	142	259
	Dual Pass	57	180	240	355	79	151
Low latency High Performance	Constant QP	146	235	361	535	199	392
	Single Pass	143	234	357	531	200	396
	Dual Pass	93	202	277	398	149	279
Low latency High Quality	Constant QP	77	226	260	381	199	392
	Single Pass	77	223	259	379	200	396
	Dual Pass	57	200	242	364	104	215

Table 13: Performance of the hardware encoder[11]

7.2.2. Encoder Performance Evaluation and Efficiency

This sub-section provides the results of the performance evaluation of the encoder system. Resolution of the uncompressed 360 video sequences used in the experiments is 3840x1920 (4k). Length is 10 seconds.

The numbers in the tables below were obtained from the tests on EPFL encoding server with Linux operating system and nVidia GPU GeForce GTX 1080 (Pascal architecture). The software system is described in Deliverable 3.4 – "Encoding and Decoding".

Preset	Encoding AVC	Encoding HEVC	Default AVC->HEVC	Lossless AVC->HEVC
nvenc HQ	89.39	88.82	94.00	55.59
nvenc HP	89.34	89.02	131.54	55.72
nvenc low latency HP	91.20	88.76	131.54	55.61
nvenc low latency HQ	82.78	88.88	131.54	55.71
nvenc Lossless HP	79.20	62.20	114.57	55.53

Table 14: Encoding and transcoding performance in FPS for different presets

¡Error! No se encuentra el origen de la referencia. presents performance numbers in frames per second of encoding and transcoding test 360 degree video sequences with different preset parameters. "Encoding AVC" column contains FPSs for encoding a raw YUV file to AVC. "Encoding HEVC" column contains FPSs for encoding a raw YUV file to HEVC. "Default AVC-

[11] NVENC_DA-06209-001_v08.pdf

>HEVC" column contains FPSs for transcoding AVC bitstream compressed with the "default" preset to HEVC. "Lossless AVC->HEVC" column contains FPSs for transcoding AVC bitstream compressed with the "lossless" preset to HEVC.

Preset	Encoding AVC	Encoding HEVC	Default AVC->HEVC	Lossless AVC->HEVC
nvenc HQ	13.842	13.283	10.854	13.283
nvenc HP	16.920	13.313	10.922	13.313
nvenc low latency HP	14.334	13.313	10.922	13.313
nvenc low latency HQ	13.874	13.313	10.922	13.313
nvenc Lossless HP	731.672	673.389	213.866	673.389

Table 15: Bandwidth of encoded and transcoded streams in Mbps

Preset	Name	Bandwidth, Mbps
Default	DrivingInCity_3840x1920_30fps_8bit_420_erp.h264	13.832
HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset1.h264	13.842
HP	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset2.h264	16.920
low latency HP	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset3.h264	14.334
low latency HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset4.h264	13.874
Lossless HP	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset5.h264	731.672
Default	DrivingInCity_3840x1920_30fps_8bit_420_erp.h265	13.168
HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset1.h265	13.283
HP	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset2.h265	13.313
low latency HP	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset3.h265	13.313
low latency HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset4.h265	13.313
Lossless HP	DrivingInCity_3840x1920_30fps_8bit_420_erp_preset5.h265	673.389
	With default h264 input	
Default	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans.h265	10.770
HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset1.h265	10.854
HP	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset2.h265	10.922
low latency HP	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset3.h265	10.922
low latency HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset4.h265	10.922
Lossless HP	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset5.h265	213.866
	With lossless h264 input	
HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset1.h265	13.283
HP	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset2.h265	13.313
low latency HP	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset3.h265	13.313
low latency HQ	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset4.h265	13.313
Lossless HP	DrivingInCity_3840x1920_30fps_8bit_420_erp.trans_preset5.h265	673.389

Table 16: Bandwidth per preset and per file in Mbps

¡Error! No se encuentra el origen de la referencia. presents the bandwidth of the encoded bitstreams for different presets. The names of the columns correspond to those in **¡Error! No se encuentra el origen de la referencia.**.

In the **¡Error! No se encuentra el origen de la referencia.** one can find the detailed information of the bandwidth for each encoded of transcoded bitstream.

7.3. Projection Types Evaluation

7.3.1. Projections and Remapping

Omnidirectional images and video must be represented in one of the panoramic mappings in the form of rectangular picture/frame in order to be fed to a base encoder (AVC, HEVC, etc.) Equirectangular projection today is the most widely used representation supported by all capture devices and stitching tools. Moreover, it allows one to perceive and assess visually a spherical picture encoded in this format. However, this representation contains pixels' redundancy, especially near the poles. To optimize the geometrical representation in the current scenario, before feeding a frame to a base encoder re-mapping must be performed.

There exist other mappings which utilize different geometric transformations in order to reduce number of pixels per frame, notably variations of cubic mapping (cubic 3x4, cubic 3x2, cubic 2x3 rotated), truncated square pyramid mapping, and more complex icosahedron and dodecahedron mappings. Some of the listed hereinabove prevent virtually the same visual quality of rendered viewport (cubic); others decrease the quality by downscaling parts of the spherical image (truncated square pyramid). See Figure 12.

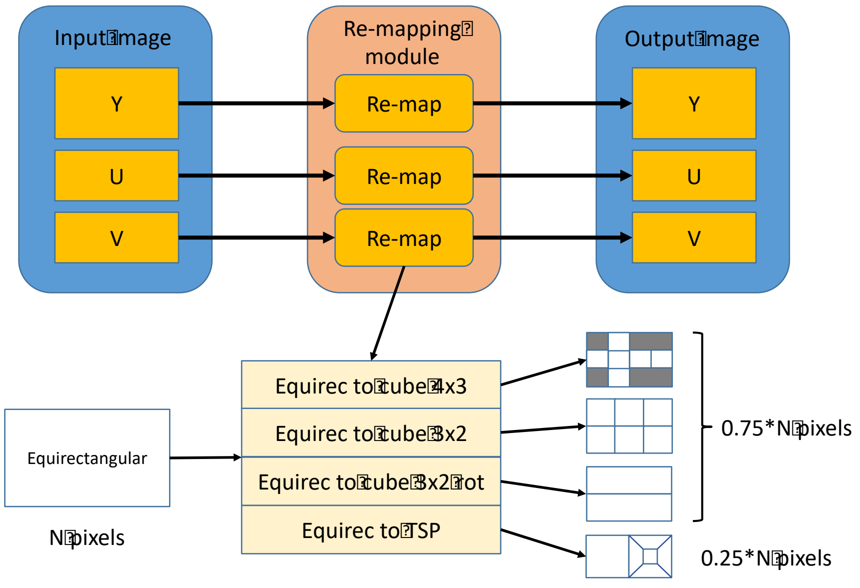


Figure 12: Remapping process diagram

During transformation from one mapping/projection to another not every pixel of the target image has a correspondent pixel in the source image. Figure 13 illustrates pixel loss during remapping from equirectangular to cubic projection. Pixels which do not exist in the source equirectangular picture are marked with red color.

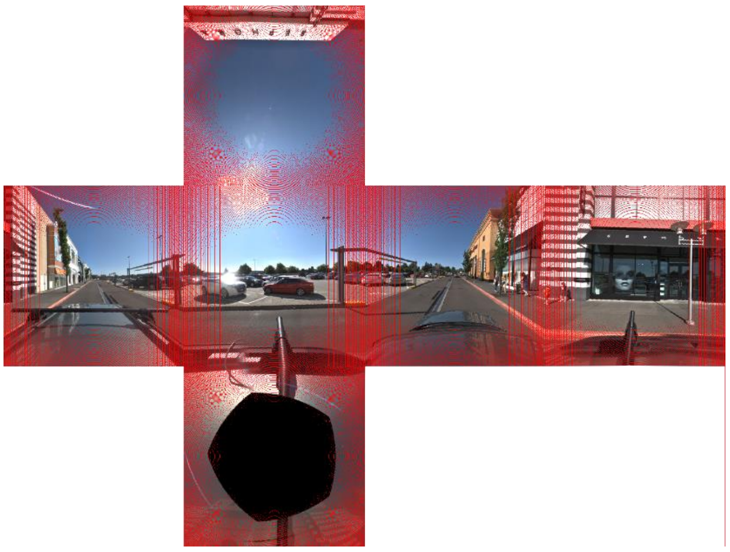


Figure 13: Visualisation of losses in cubic projection

In order to fill the gaps, interpolation (known also as re-sampling) of the source image must be performed. Another purpose of re-sampling is to prevent aliasing. It appears in details where a Nyquist frequency is higher than the sampling frequency of the signal (sampling theorem). Re-sampling is used in computer graphics for decades; and the most common algorithms are nearest neighbor, bi-linear, bi-cubic, and Lanczos interpolations. In **¡Error! No se encuentra el origen de la referencia.** one can see the interpolation rationale and a diagram of the re-mapping algorithm.

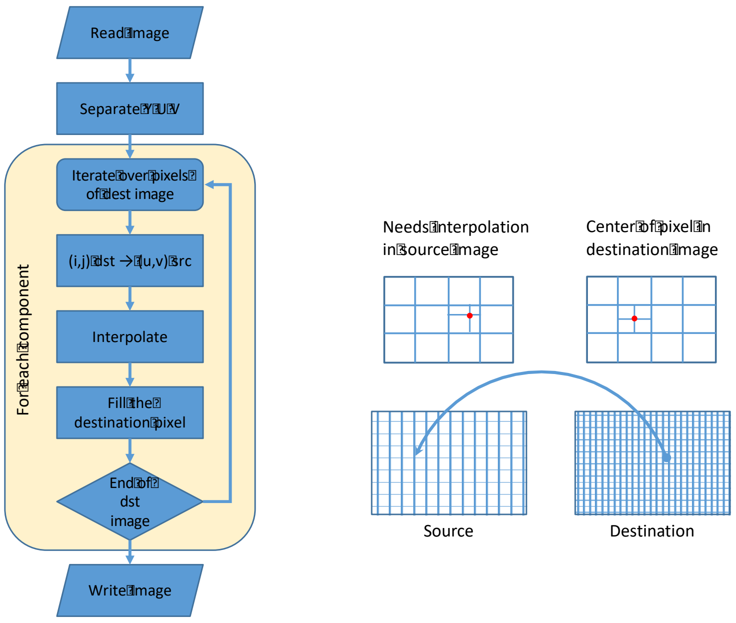

Read image
Separate Y U V
For each component
Iterate over pixels of dest image
(i,j) dst → (u,v) src
Interpolate
Fill the destination pixel
End of dst image
Write image
Needs interpolation in source image
Center of pixel in destination image
Source
Destination

Figure 14: Remapping flow chart

In the current scenario of re-mapping + encoding with base codec we must choose an optimal mapping. The following main criteria should be considered: representation efficiency (pixel redundancy), visual quality of rendered image, and re-mapping complexity. Cubic mapping 3x2 with rotated faces is recommended to use. It preserves virtually the same quality as equirectangular picture, reduce pixel redundancy, and not complex computationally. Moreover, rotating particular cube faces reduces spatial complexity of the picture which leads to higher compression efficiency.

7.3.2. Projection Types in terms of pixel redundancy

Decrease in overall resolution during consequent remapping is depicted in Figure 15. After transforming equirectangular projection to cubic one we save 16% of the image area (red hatched rectangular). Further remapping to half resolution back part cube projection subtracts 32% more pixels (blue hatched area) from original making the picture to contain only 52% of equirectangular representation pixels. Note that keeping resolution the same is not reasonable because it would be a simple upscaling of picture.

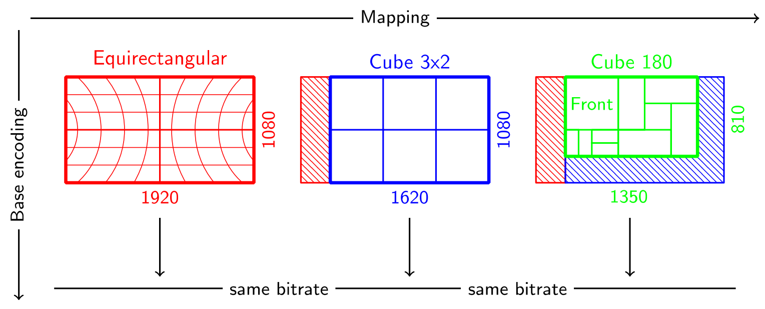


Figure 15: Remapping impact on overall resolution**.**

Maintaining **quasi-constant overall resolution** is another approach to perform rempping. In **¡Error! No se encuentra el origen de la referencia.** one can see that, when we apply remapping filters directly to the original ultra-high resolution image, pixel losses are 16% and 6% for cubic and half-back cubic projections respectively.

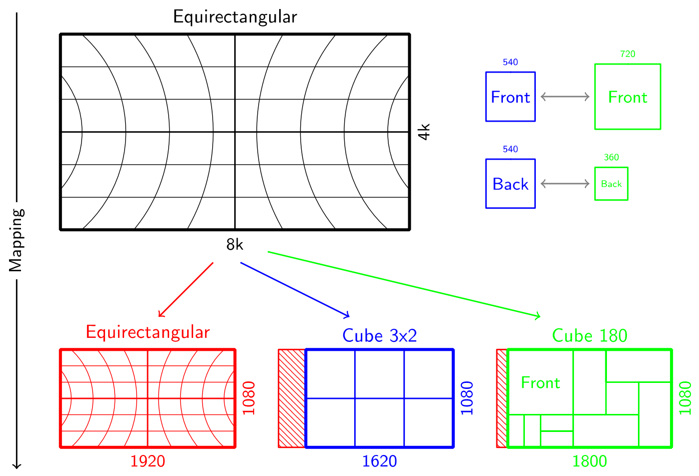


Figure 16: Quasi-constant resolution over different mappings**.**

Moreover, in the top right part of Figure 16 one can notice visualization of resolution differences for front and back faces between cubic and half-back cubic projections.

7.3.3. Subjective quality evaluation of different projections

In **¡Error! No se encuentra el origen de la referencia.**, Mean Opinion Score (MOS) vs. different Bitrates are plotted for different contents. MOSs obtained from naive subjects were shown to be highly correlated with expert subjects results. Pearson Linear Correlation (PLCC), Spearman Rank Order Correlation Coefficient, and Root Mean Square Error (RMSE) indices are used to verify the validity of experiment. The standard correlation indexes between naive and expert scores are *PLCC = 0.95*, *SROCC = 0.87*, *RMSE = 0.40*. Certainly, this allows us to consider the subjective evaluation results being reliable and consistent.

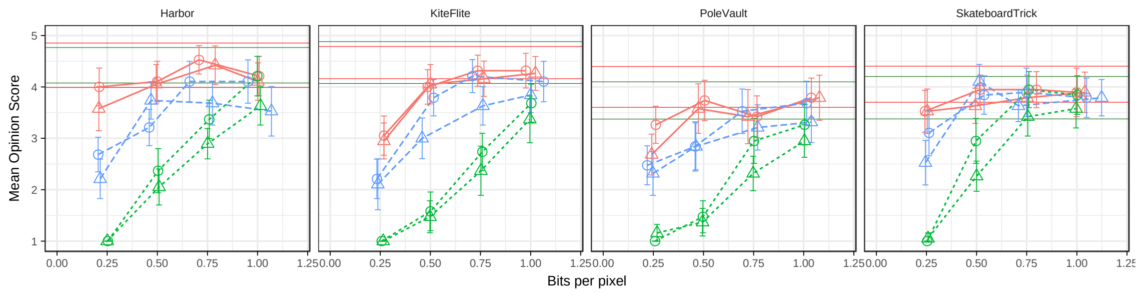


Figure 17: MOSs vs. bpp (bit-per-pixel) obtained using ACR-HR method for compressed omnidirectional images. Red (solid) line represents HEVC encoded content, blue (long-dashed) - JPEG 2000, and green (short-dashed) - JPEG. Equirectangular projection is depicted with circles and cubic mapping - with triangles. Filled area between two horizontal lines corresponds to the 95% confidence interval of the hidden reference for each projection (red for equirectangular, cyan for cubic).

When compressing images represented in a cubic projection, edges of continuous parts of the frame are distorted non-uniformly with different intensity. This makes cube-face borders distinguishable for some stimuli in the rendered viewport when observed using an HMD. Experimental results, indeed, show lower scores for cubic mapping at medium bitrates and the same scores as for equirectangular mapping at high and low bitrates. This may occur for the reason that at high bitrates there are no impairments, and at low bitrates the entire image is distorted, thus only at medium bitrates the cube-face borders are distinguishable due to compression artefacts.

7.4. Evaluation of Distribution and Reception Tools

The distribution and reception of multimedia content within the ImmersiaTV platform is provided through MPEG-DASH (Moving Picture Expert Group - Dynamic Adaptive Streaming over HTTP). Using MPEG-DASH, different video resolutions are generated (HD (1080p), 2K, and 4K for the 360-degree and TV videos, and 420p for the portal videos). The video segments with different resolutions are selected to be streamed based on the available bandwidth.

In order to provide a coherent experience through TV, Tablets and HMDs, the contents should be synchronized in all devices, however, it is possible to lose synchrony due to network problems and capability of different display devices. ImmersiaTV distribution and reception module developed mechanisms to overcome the delay problems and achieve synchronization of multimedia streams. Two versions of the player is developed in the context of ImmersaTV platform: (a) web player and (b) player software based on the Unity3D engine. The web player is based on web technologies such as Javascript and HTML5 and enables playing multimedia content on web browsers. The functionality of two players are evaluated in ImmersiaTV platform.

To achieve the synchronization of multimedia streams in web environments, a master-slave architecture is used in which the master is a server in charge of managing both Wall Clock time and media times. The clients are web browsers that connect to the server to synchronize the media assets they want to play.

The synchronization accuracy of web-based system is testified by several experiments. In the first test, we start playback of a streaming DASH with 720p resolution and 30fps frame-rate (4Mbps) from different devices reproduced for 30 minutes. In order to observe the synchronization algorithm effect, a cut is applied to the transmission every 10 minutes. The cuts force a temporary jump and let us observe the act of synchronization algorithm.

Figure 18(a) shows the synchronization performance. As it is observed, the delay is in between the values of +20ms and –20ms, which is imperceptible for human eyes and when a temporary jump is forced, the system reacts to re-synchronize the stream.

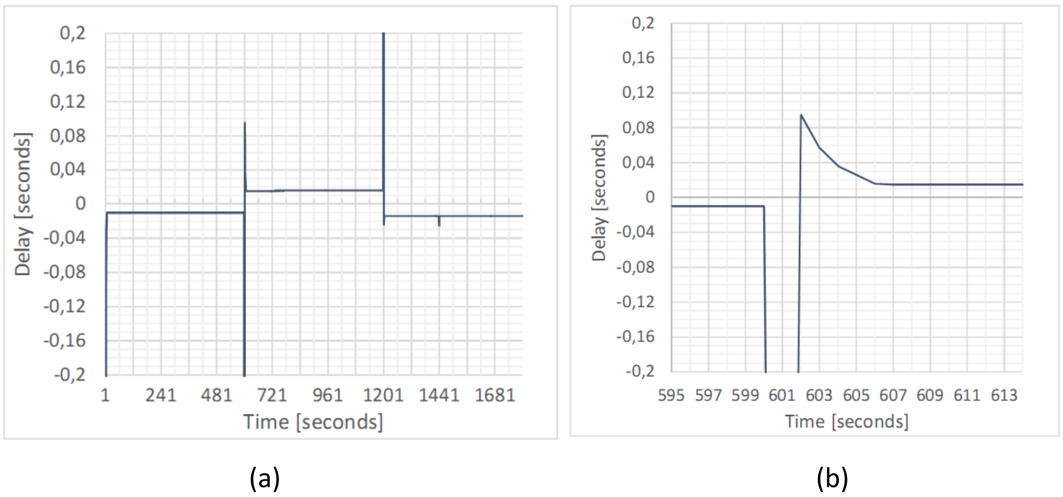
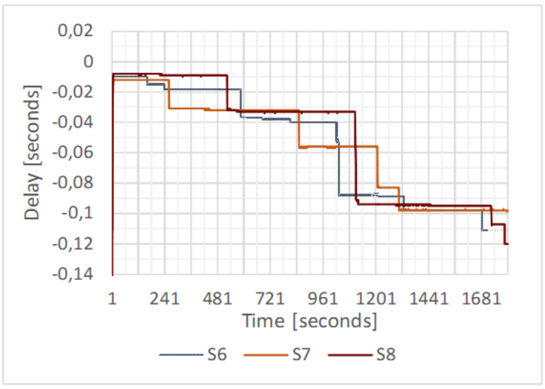


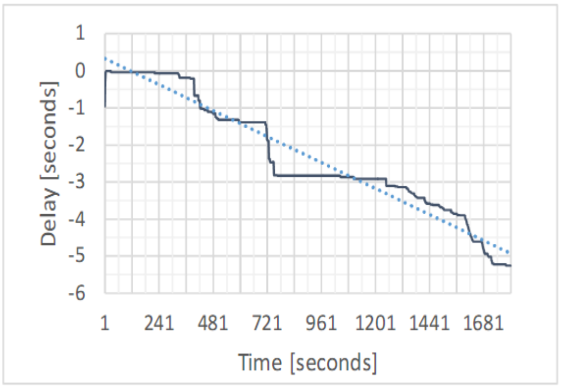
Figure 18: Synchronization evaluation. (a) Stability of media playback in the platform. (b) Synchronization jump detail between 595 and 614 seconds of playback

Figure 18(b) illustrates in more detail that the video stream can be synchronized with the main stream in a short time when an abrupt jump is happened. It takes less than one second for the player to detect the delay event and activate the algorithm, synchronizing the stream with 95ms of delay between the both streams. Then, the delay is decreased in just 3 seconds until reaching a frame accurate synchronization. The system can synchronize the streams at frame level in about 5 seconds.

The second experiment presents the importance of synchronization for multi-device playback. The goal is to realize how the system performs without having a synchronization mechanism. In this test, a streaming DASH with a single resolution of 720p at 30fps and 4Mbps is started to playback from different devices with activated synchronization. After 5 seconds, the synchronization mechanism has been deactivated and the playback is continued for 30 minutes. The Samsung Galaxy S6, S7, and S8 smartphone devices and Galaxy Tab S were used in the experiment. As it is depicted in Figure 19, the smartphones begin to lose synchrony after few seconds. This desynchronization effect is even more evident in case of Tablet (Figure 19 (b)) which have difficulties to reproduce the content due to its lower processing capacity than smartphone devices.

In the synchronization tests, it has been shown that the delay in a device with the synchronization mechanism is not significant, which allows for the delivery of a coherent playback between multiple devices. In case of deactivating the synchronization, once established, the delay observed on smartphones after 30 minutes is approximately 110ms, that is, about 3 frames apart. In the case of the Galaxy Tab S, the delay is much more apparent reaching more than 5 seconds of delay after 30 minutes of playback.





(a) (b)

Figure 19: Delay over time caused by the lack of synchronization in (a) Samsung Galaxy S6, S7 and S8 and (b) Samsung Galaxy Tab S.

In addition to a web player, we developed a player that integrates with the Unity3D game engine. The Unity3D player is available for Windows and Android devices and uses the Gstreamer open-source framework to process media streams.

We now report on the performance of the Unity3D player in terms of decoding delay, synchronization accuracy, CPU load and memory usage.

First, we compared the required time for video decoding on a Unity3D player running three video resolutions on three different devices (PC, smartphone, and tablet). To measure the decoding delay, we report the timestamps of frame position, start of frame decoding and end of frame decoding during a period of 10 minutes. Using the timestamps, the average decoding delay during a 10 minutes playback is measured. The hardware specifications of the test devices are summarized in Table 17. Three video resolutions (1024x512, 2048x1024, 3840x1920) are considered in the experiment to capture the system performance changes due to different resolutions of the video content. All three devices were playing same content.

PC	Smartphone	Tablet
OS Name: Microsoft Windows 10 Education **System Manufacturer**: GIGABYTE **System Model**: GB-BSi7A-6500 **System Type**: x64-based PC **Processor(s)**: Intel64 Family 6 Model 78 Stepping 3 Genuine Intel ~2592 MHz **Total Physical Memory**: 8062 MB	**Model**: Samsung Galaxy S7 SM-G930F **SYSTEM:** Android Version: 7.0 **Display**: Size: 1920x1080, Refresh Rate: 59 Hz **Processor**: Exyons8 Cores: 8 **Max Frequency**: 1586 MHz **Memory**: 3533 MB **Video Frame Format**: Nv21	**Model**: Samsung Galaxy Tab SM-T800 **SYSTEM**: Android 6.0.1 **Display**: Size: 1920x1200, Refresh Rate: 60 Hz **Processor**: ARMv7 , Cores: 4 **Max Frequency**: 1900 MHz **Memory:** 2773 MB **Video Frame Format**: Yuv420sp

Table 17: Specification of PC, Smartphone, and Tablet used in the experiment

Table 18 indicates the average time consumed for decoding the video streams of different resolutions in each of three devices. The amount of decoding time varies based on the type of device and video resolution, therefore, a synchronization strategy is necessary to provide a coherent experience between all devices. The Tablet needs more time for video decoding compared to PC and phone due to its lower processing capacity.

Resolution	1024x512			2048x1024			3840x1920		
Device Type	**PC**	**Phone**	**Tablet**	**PC**	**Phone**	**Tablet**	**PC**	**Phone**	**Tablet**
Mean Delay (s)	0.124	0.132	0.246	0.201	0.164	0.314	0.213	0.1953	0.333

Table 18: The average decoding delay of different devices playing different video resolutions.

To evaluate the synchronization performance between three devices, a video with resolution 2048x1024p is played for 10 minutes and the display timestamps of PC, phone, and tablet are gathered during this period. Next, at each presentation timestamp (PTS), the maximum latency between display times of three devices is computed. Figure 20 shows the maximum delay observed between three devices within 10 minutes playback. The average delay is about 30ms and the results confirm that delay is small enough to provide a synchronized playback.

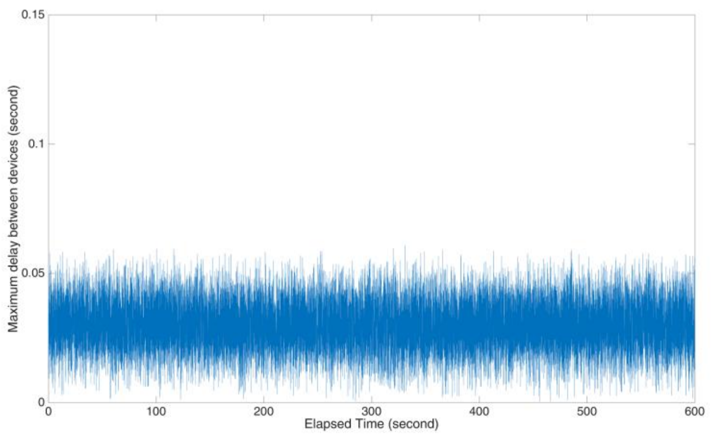


Figure 20: Maximum delay observed at each presentation timestamps during 10 minutes playback.

In order to examine the player performance, we report the CPU and memory load of the player running on PC. The amount of CPU load of Unity3D player on PC in different resolutions is determined and compared. The *% of Processor* is reported during 10 minutes of playing which indicates the percentage of time the processor is busy by measuring the percentage of time the thread of the Idle process is running and then subtracting that from 100 percent. This measurement is the amount of processor utilization. The PC has four logical processor, so the maximum value is 400%. Figure 21 presents the % of Processor time during 10 minutes playback. As shown in the figure, the CPU load is increasing when playing higher resolution videos and the installed CPU is sufficient to run 4K videos using Unity3D player.

For performance monitoring of the player in case of the amount of memory usage, *Private Bytes* counter is used. The counter indicates the total amount of memory that a process has allocated, not including memory shared with other processes. Figure 22 presents the memory request by Unity3D player when playing 10-min videos in three different resolutions. In Table 19, we summarized the minimum, maximum, and mean CPU load and memory usage considering different video resolutions.

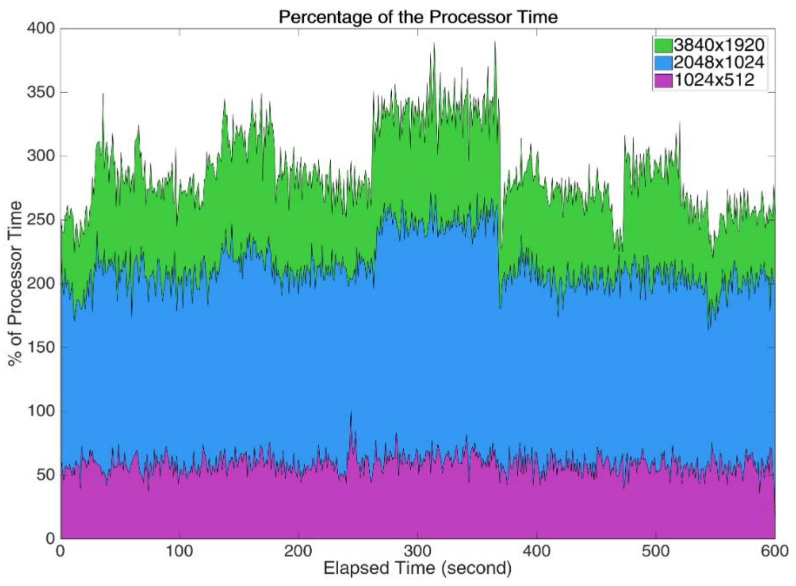


Figure 21: Comparison of the % of Processor time when playing videos with three different resolutions on PC.

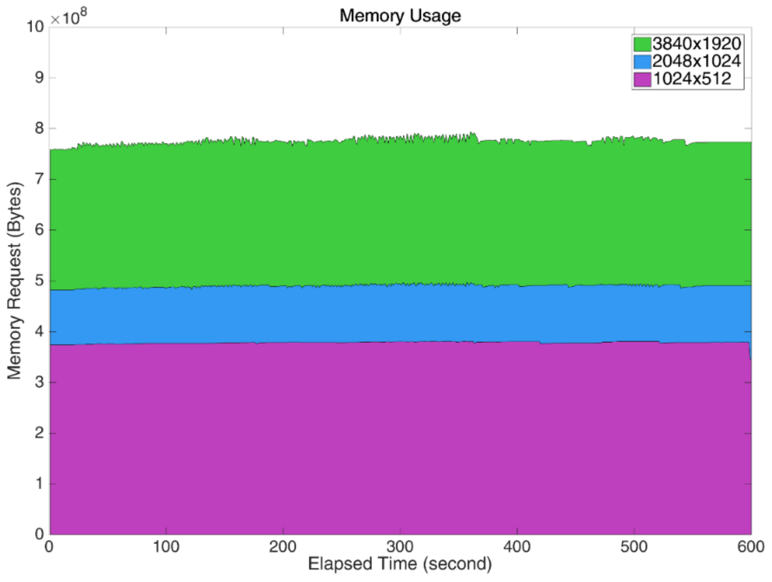


Figure 22: Comparison of the memory usage when playing videos with three different resolutions on PC.

	% of Processing Time			Private Bytes		
Resolutions	min	max	mean	min	max	mean
1024x512	14.089	101.044	59.112	3.45×10^8	3.81×10^8	3.78×10^8
2048x1024	163.582	271.685	213.991	4.82×10^8	4.97×10^8	4.90×10^8
3840x1920	219.788	390.222	289.293	7.58×10^8	7.93×10^8	7.75×10^8

Table 19: The minimum, maximum and average CPU load (% of processing time) and memory usage (Private Bytes) of Unity3D player when playing 10-min videos of different resolutions.

7.5. The Quality of Experience Tool Evaluation

The Quality of Experience (QoE) module evaluates the perceived quality as experienced by end-user and considers the parameters important for satisfactory video content delivery. The QoE module aims to steer parameters inside the codec and to guarantee the highest quality. According to the Pilot 2 plan, the QoE software is devised to receive logging data from client side and return a number of quality analysis outputs that can further be used in Pilot 3 to help codec for more concise encoding.

Parameters	Detailed sub-items	Description
Content	Content id	Video identification data. It points to an xml file in ftp server which contain basic information of the video
Session	Session id	Player session identification data
Constant parameters	Device id	This is an ID that specifies which type of device is on operation (TV, Tablet or HMD)
	FoV	Horizontal and vertical field of view in degree
	resolution	Device screen resolution in pixels
Session events	Center of looking information	The information about the clients looking direction (specified by three parameters: yaw, pitch and roll)
Stream events	Frame rate	The nominal and actual frame rate of streams
	Dash Segment ID	The address of the video segment that is currently playing on the ftp server
	Delay information	The amount of delay in seconds and the presentation time stamps (PTS)

Table 20: List of logging data gathered for QoE analysis

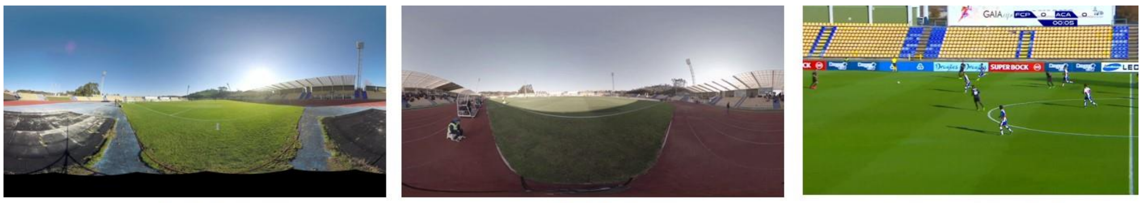
The client is in fact a player software that can provide the logging data for QoE module from different devices. The QoE communication module receives information from clients in JSON format through a tcp socket. The gathered logging information is presented in Table 20.

The quality of videos in terms of visual artifact is evaluated using No-Reference methods since they do not need to access the reference frames for quality evaluation. Several NR quality assessment methods are implemented which can be selected based on the complexity constraints. The blocking metric, sharpness and blur metrics are selected for current platform as they are running faster. The final score is derived by fusing three metrics using our LAF[12] machine learning system.

To verify the functionality of the designed objective metric, an experiment has been conducted. The metric is used to evaluate the quality of several videos captured in the ImmersiaTV platform. Three videos are selected for the experiment and four different bitrates (6Mbps, 3Mbps, 1.5

[12] http://www.locally-adaptive-fusion.com/

Mbps, and 750 Kbps)) are generated for each video. A frame of each of the three videos are shown in Figure 23.



sp_cam1_360_0_Trim sp_cam1_360_2_Trim tv_tv_maintrack_3

Figure 23: Frames of three videos used in the experiment.

The videos were played back for 5 minutes and the mean quality scores were reported. Table 21 shows the mean quality scores obtained for each video and bitrate. The higher score means the better quality. Figure 24 better depicts the quality changes of videos in different bitrates.

	6Mbps	**3Mbps**	**1.5Mbps**	**750Kbps**
sp_base_360_2	84.93	79.15	73.88	64.78
sp_cam1_360_0	81.41	76.95	72.60	64.57
tv_tv_maintrack_3	73.46	64.14	52.66	36.77

Table 21: Average quality score of three videos in four selected bitrates. (The score range is between 0-100 and higher score means better quality)

As it is observed, the designed metric can measure the extent of quality changes in different bitrates. The quality drop is faster for “tv_tv_maintrack_3” video than the two others. Two 360-degree videos (sp-base-360-2 and sp-base-360-0) mostly contain static scenes without any abrupt changes while the third video (tv_tv_maintrack_3) is a football game with many motion and frame changes. The compression, and the quality, can vary considerably depending on the amount of motion in the scene. The relatively "static" scenes can maintain high quality in low bitrates and "active" scenes lead to lower quality in higher compression. Therefore, when the bitrate is decreased, it is expected to observe lower quality in the “tv_tv_maintrack_3” video which have many motions while the static scenes (which use less bits) can better maintain quality in low bitrates.

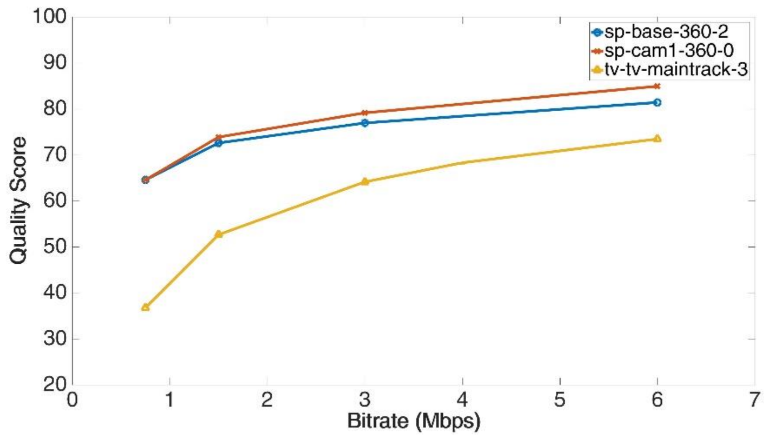
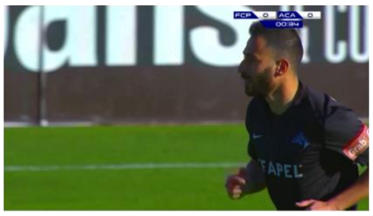


Figure 24: Average quality vs. Bitrate for three videos in four bitrates.

As it can be seen in the Figure 24, the designed quality metric can correctly indicate faster quality drop of the “tv_tv_maintrack_3” video compared to other videos. In other words, lower bitrates can be used for two 360 videos without significant effect on the perceived visual quality while the low bitrates of “tv_tv_maintrack_3” can result in noticeable quality degradation.

Figure 25 shows frames of the selected videos and the quality scores at different bitrates. When looking at the frames, it is clear that the quality of “tv_tv_maintrack_3” is significantly decreased at 1.5 Mbps while the “sp-base-360-2” has descent quality even at 750 Kbps. The derived quality scores can also correctly confirm that the quality of “tv_tv_maintrack_3” video at 1.5 Mbps is lower than “sp-base-360-2” at 750 Kbps. The quality evaluation can be used within ImmersiaTV platform to steer parameters for concise and efficient compression.

sp_base_360_2 (Bitrate 750Kbps, Quality score: 64,57)


tv_tv_maintrack_3 (Bitrate 3Mbps, Quality score: 64,14)

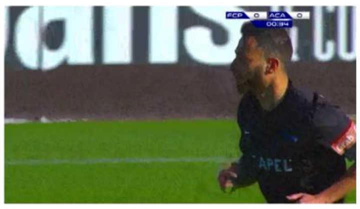

tv_tv_maintrack_3 (Bitrate 1,5Mbps, Quality score: 52,66)

Figure 25: The video frames at different bitrates and the corresponding quality scores

In addition to the designed quality metric, the QoE module reports the amount of delay and fixation point information. The delay information represents a stream of moving averaged values of delay in seconds which are used to analyse stalling events and jerkiness. The fixation point information represents the spots that are more focused by users and will be used in Pilot3 for adaptive resource allocation in Encoder. An example of the QoE outputs is presented in Figure 26.

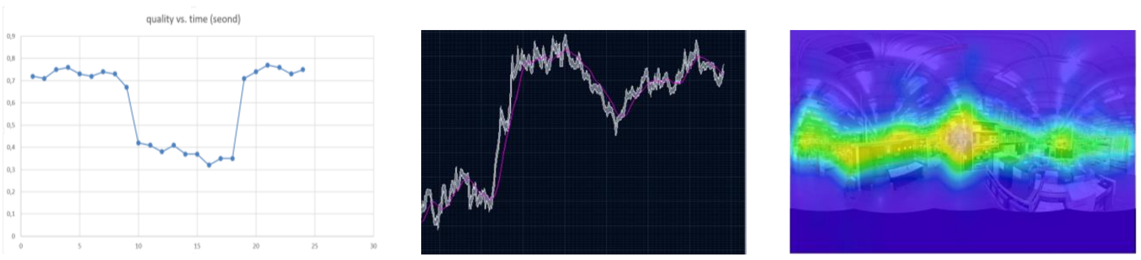


Figure 26: Graphical representation of the output data. a) Video quality changes in time b) Moving averaged delay which shows the trend of delay changes (red line) c) Fixation map shows the locations focused by users.

The QoE module consists of several processing stages. In order to examine the performance of QoE module, the required time of each processing stage is measured when analysing two video resolutions (Full-HD and 4K). The videos are played for 10 minutes and the average required time to process each stage is reported. The experiment is conducted on a PC with 2.7 GHz CPU, Quad Core i7, 8M cache, and 16GB memory. The processing stages include: Receiving log data from tcp socket, Parsing the logging data, Accessing video from server, Concatenating videos and extract frames, Computing delay and focusing data, and Computing visual quality scores. Table 22 presents the processing time for two video resolutions.

Task	Elapsed Time in Second (FHD)	Elapsed Time in Second (4K)
Receive log data from tcp socket	0.13	0.15
Parsing the logging data	0.037	0.036
Access video segments from server	0.32	0.98
Concatenate video and extract frames	0.13	0.18
Compute delay and gathering focus data	0.017	0.02
Compute quality score	0.12	0.391
Overall processing	0.803	1.92

Table 22: Average computational time of different components of the QoE module for FHD and 4K resolutions

The results show that the QoE module is able to report the quality parameters in reasonable time intervals. In analysing the 4K video, most of the time is consumed to access the video segments from ftp server. The processing time can be further improved by establishing a connection to a DASH server or by accessing videos directly at the QoE server.

8. CONCLUSION

This document presented an overview of ImmersiaTV platform, technical specifications of components and the related evaluations. The document was updated iteratively for each Pilot.

Regarding Pilot 1, the technical components as well as the technical issues observed in Pilot 1 are demonstrated. Moreover, a number of guidelines are provided to improve the pre-production and post-production performance. Some important guidelines are summarized as follows: In shooting (video capture), the 360 rig and directional camera should operate simultaneously to avoid synchronization issues. It is required to carefully pick up a calibration scene. A number of recommendations are provided in the document for better calibration which can also lead to improved stitching performance. A good production plan is needed because preproduction setups (such as camera adjustments, and scene details) can influence the complexity of postproduction (stitching). In video compression, a region-aware remapping can avoid missing detail because of low resolution, however, it affects the visual quality. Higher resolution and frame rate are recommended to improve the quality of experience.

For Pilot2, the developed modules are evaluated through several substantial experiments to validate the system functionality. For production tools, the tests are focused on evaluating the GPU performance for parallel content encoding since the encoding is the most complex task at the production side. The results show that the GPU usage is almost linearly proportional to the number of streams and frame resolution. This can help to predict the number of required resources (GPU) when increasing the number of streams and users. The parallel stream generation is possible using the current developed production tool. The focus point data of users can also be used to optimize media delivery and make it adaptive to the user view ports.

The HEVC encoder hardware specifications are also demonstrated in the deliverable and the efficiency of codec is evaluated with different preset parameters. The Encoder/decoder should be able to compress 4K omnidirectional video at up to 30 frames per second. A H.264/AVC and H.265/HEVC mixed MPEG-DASH distribution can also be enabled to allow both codecs per one MPEG-DASH stream in which the web player will reproduce the H.264/AVC stream and the native player can get advantage of the more optimized compression of H.265/HEVC. The performance of the projection types used in the ImmersiaTV platform is testified in terms of pixel redundancy and visual quality. Since both equirectangular and cubemap projections are useful in the system, an automatic conversion framework from equirectangular to cubemap representation is recommended. Furthermore, the encoder will perform more efficiently by adjusting the encoder settings based on the QoE module data. At the distribution and reception side, several experiments considering synchronization issues, CPU load and memory usage have been conducted to validate the functionality of web player and Unity3D player. The experiments indicates a suitable synchronization between different devices. Finally, the performance of the QoE module is examined and the results are reported. The QoE module will be integrated with encoder to improve compression efficiency.