# Gaze-Based Personalized Multi-View Experiences

Maria Teresa Andrade
Faculty of Engineering, University of Porto, Porto, Portugal
Email: mandrade@fe.up.pt

Tiago Soares da Costa
INESC TEC/Center for Telecommunications and Multimedia, Porto, Portugal
Email: tacosta@inescporto.pt

***Abstract*—This paper describes a solution for delivering and presenting stereoscopic video content to users in an innovative way. It adopts the multi-view paradigm of the H.264-MVC video coding standard and the emergent MPEG DASH specification to provide users in heterogeneous network environments multiple and varying perspectives of stereoscopic video sequences. Unlike existing 3D systems based on multi-view technology, which require high transmission bandwidth and high processing power on the terminal device to achieve the same objective, the proposed solution is able to make an efficient use of network resources whilst being cost-effective. It offers users a higher quality of experience by seamlessly adapting the quality of the delivered video content according to the network conditions, whilst providing a more realistic sense of immersion by offering stereoscopic views of the scene, dynamically switching the perspective to match the interests of the user. A non-intrusive head-tracking system using an off-the-shelf Web camera detects the focus of attention of the user, transmitting this information to the server that selects the most appropriate view to send to the client. Additionally, the system is able to generate the multiple perspective stereoscopic scenes using 2D cameras.**

***Index Terms*—multi-view, eye tracking, personalized, immersive, context, multi-media adaptation**

## I. Introduction

Multi-view video can be obtained by using an array of 2D cameras to capture the video scenes. Cameras should be placed alongside in arch, observing some rules such as a maximum spacing of 65 mm or 1/30 of the distance to the object being filmed so that the user may enjoy smooth evolving panoramic visualization. The acquired multiple perspectives can then be stored in a video repository, sharing semantic metadata with additional information concerning their displacement in relation to a central view. If this type of content is to be presented in stereoscopic format then, one single perspective will be formed by the sequence of images captured by a pair of cameras. The pair of 2D streams should be further combined in a format suitable for integrated transmission, such as the side-by-side format, before being stored in a repository. This kind of multi view content can then be presented to the user in a dynamic and adapted way, providing the sense of immersion in the scene in a seamless way. By automatically detecting the focus of attention of the user through eye tracking at the terminal side, it is possible to dynamically adapt the perspective of the scene being presented to the user; eventually the user could have a 360 º view of the scene without the need to move or explicitly provide indications to the system. This kind of perspective-adapted visualization can be very interesting when watching live events, such as cultural or sports events. But it can also be useful in industrial environments, such as remotely monitoring the installation of equipment in difficult-to-reach places. The work described in the paper aims at delivering a cost-effective client-server solution to this challenge by implementing efficient ways of delivering multi-view video to diverse terminal devices in networked environments applying the free viewpoint paradigm, yet not requiring highly sophisticated terminals. Assisted by knowledge derived from contextual data sources (physical and software sensors, namely video camera and network probes), as well as content metadata the server dynamically select the most suitable video content and perspective/view to send to the client. The video camera pointed at the user enables to extract contextual data used by the system to infer the focus of attention of the user in the scene being visualized; the contextual data collected by the network probe enables the system to verify network conditions and consequently to automatically adapt the bit rate of the delivered 3D video sequence, thus providing a sustained quality of service. Although the solution has already been conceived and functionally tested, it is still an on-going work, where usability and performance tests are yet to be conducted and technical aspects related essentially to latency need to be solved.

This paper is organized as follows. After the brief introduction to the work as here given, Section II provides some background information on relevant topics to this work, namely eye tracking techniques, stereoscopic video coding and adaptive video streaming. Section III describes the expected functionality of the

Manuscript received September 1, 2015; revised November 23, 2015.



doi: 10.12720/jmmc.1.2.43-49

client-server system and it illustrates its corresponding functional architecture, indicating the main technological options made. Sections IV and V provide details on, respectively, the client application and MPEG DASH framework, highlighting the main challenges that were overcome. Section VI describes the results obtained from the initial functional tests and Section VII concludes the paper by discussing future developments.

## II. BACKGROUND

The sense of immersion can be obtained by presenting stereoscopic or panoramic video to users. The current main approaches for 3D video coding are: 1) conventional stereo; 2) video plus depth; and 3) multi-view [1]. Combination of the three can be used aiming at enhancing the basic technique to optimize the rate-distortion ratio [2]. MPEG-C Part 3 [3] is a video-plus-depth representation of 3D video, which requires reduced transmission payload comparing to conventional stereo video (L-R). Multi-view Video Coding (MVC) [4] is one of the state-of-the-art methods for encoding multiple camera views using inter-view correspondences for advanced compression. Multi-View-plus-Depth (MVD) encodes multi-view depth map together with the MVC signal. MVD enables the synthetization of different viewpoints suitable to auto-stereoscopic and multi-view displays [5]. This concept is being explored by MPEG in the development of a new 3D video coding framework. It enables to overcome MVC limitations in terms of bit-rate, which requires sending many views when using advanced multi-view displays, by sending a limited amount of data, yet sufficient to allow the user terminal to render arbitrarily many views. HEVC (High Efficiency Video Coding [6]), the most advanced video compression standard, has also a 3D/multi-view extension [7] and [8].

In traditional 3D systems, even when using the most advanced compression algorithms, the amount of data to be transmitted already exceeds largely that of 2D systems. When considering 3D immersive environments, where multiple views of the same scene are normally captured using many cameras, to enable wide field of view (FOV) and 3D reconstruction, then the involved resources explode. Accordingly, given the best effort nature of the Internet, when delivering 3D video over networked environments, adaptation should be considered [9]. Two different approaches can be adopted: 1) acting directly on the video sources in real time, by altering the encoding parameters; and 2) having pre-prepared different versions of the same video source and then dynamically switching between versions according to network conditions.

In multi-view systems, the former approach can lead to the implementation of generic Free-viewpoint TV (FVT) applications [10]. It provides view scalability, using the scalable extension of H.264/AVC [11], which allows truncating a number of viewpoints from the complete set of views. Visual and audio attention modeling approaches are also interesting alternatives for designing adaptable multi-view 3D video encoding systems [12].

The latter solution can resort to recently developed transport protocols such as HTTP Live Streaming (HLS) [13], or the recent MPEG DASH (Dynamic Adaptive Streaming over HTTP) specification [14]. These media delivery specifications are based on a segmentation of the video sources at the server side. They define a set of messages to be exchanged between clients and server alongside specific metadata, enabling the client to request the version of the video source that suits better the current bandwidth availability, on a video segment basis. These specifications thus aim the adaptable transmission of audio visual flows in IP networks using the HTTP protocol.

Although there are more than one such specifications, they are based on the same paradigm: the content to be transmitted is divided in segments of short duration in time (normally from 2 to 10 seconds), which are sent at the request of the user. These requests are made in a manner such that the reproduction of the audio visual content is achieved in real-time and matching the network constraints. If the server creates several versions of the video segments, each one with different qualities, formats and bit rates, it is possible to satisfy the needs of different clients. Accordingly such specifications provide the tools to implement a transmission of audiovisual content adapted to the instantaneous conditions of networks and initial constraints of terminals. To indicate the client the versions that have been created, the server generates a metadata file with information indicating the location of the segments, their duration and associated quality/bit rates, amongst other data. This file is called "manifest" and locations of the segments is signaled using URIs.

From the available solutions, the MPEG-DASH specification is the most recent one and yet with more penetration and support. DASH is a technology to stream multimedia content with bit rates that smoothly adapt to the network and reception conditions. It accepts different media compression formats, defining how the compressed data are sectioned and packed to be streamed via HTTP to clients. It also defines the type and format of the metadata to be included in the manifest file, made available to the client that uses it to decide in each instant which segment to request from the server.

Visual attention can be detected by collecting eye gaze data. This approach started to be used to study user attention patterns when processing information [15]. It was then adopted within the human-computer interaction to gain insights of user behavior and preferences [16]. Whereas initially the aim was to generate models for off-line use, it has recently expanded to real-time systems, using eye tracking as a mean to directly control a system. In this context, solutions are starting to appear also in audiovisual commercial applications [17]. In TV for example, a solution was recently launched to allow users to replace the remote control with their own eyes [18]. Users are able to activate a user-interface on the TV set by staring at a certain area of the screen and then they can change the volume or switch channel just like they do with the remote control. These solutions attempt to provide a more user-friendly interface for already existing functionality. They thus basically provide to the user a different way of explicitly interacting with the application or device.

Our system uses eye tracking to implement an implicit and seamless interaction of the user with the audiovisual content itself, thus providing the user a more realistic sense of immersion in the scene. Additionally, in relation to existing immersive 3D content solutions, our approach is based on low-cost equipment, does not consume high network resource nor it requires sophisticated auto-stereoscopic displays. In fact, as we have just seen, presenting 3D content to remote users, enabling them to move around the screen to have different perspectives of the scene (wide FOV as in FVTV) requires extremely large bandwidth channels and/or highly sophisticated terminals. If many views are sent simultaneously to the terminal, then network resources consumption is high. If only some views are sent, then the terminal must have high processing power to be able to regenerate the skipped views. In either case, the display must have very high refresh rates to be able to present all the received or regenerated views to the user. The motivation for our work is precisely to try to overcome these limitations. The developed system has also the ability to automatically adapt the bit rate of the 3D video stream to suit varying bandwidth availability, typical of best-effort networks such as the Internet. Accordingly, our solution distinguishes itself from its ability to be aware of user preferences and of network conditions, 0shown to the user and the bit rate, whilst being cost-effective.

## III. Functional Model of the Multi-View Personalized Solution

The aim of the work described in this paper is to obtain an efficient and low-cost solution to offer personalized immersive experiences to remote users. The idea is to present 3D multi view audiovisual content to distinct publics using simple tools whilst overcoming obstacles such as limitations in network bandwidth availability and terminal constraints. The developed prototype comprehends modules located at both the client and server side which communicate with each other via HTTP and UDP. The concept is based on: the availability at the server side of multiple perspectives of varying qualities of the same audiovisual sequence; and on the intelligence capability at the client side to identify at any given instant the focus of attention of the user, as well as network conditions, to be able to control the transmission of video data. Based on an initial request from the user, the server sends a pair of views from a selected video source stored at the server. Video content is encoded according to the Multi View Coding (MVC) concept of the H.264 standard. The pair of views that is sent at a given instant to the client is selected according to eye tracking data sent by the client to the server, translating the user focus of attention in the scene. In this way the server is able to seamlessly adapt the perspective of the scene being presented to the user according to the user visual attention preferences.

By dynamically monitoring network conditions, the client is able to adapt the bit-rate of the streamed pair of views, without the need to actively involve the server in this decision to adapt. The server simply receives HTTP requests from the client and accordingly streams the requested video data. Fig. 1 illustrates the high level concept of the developed system.

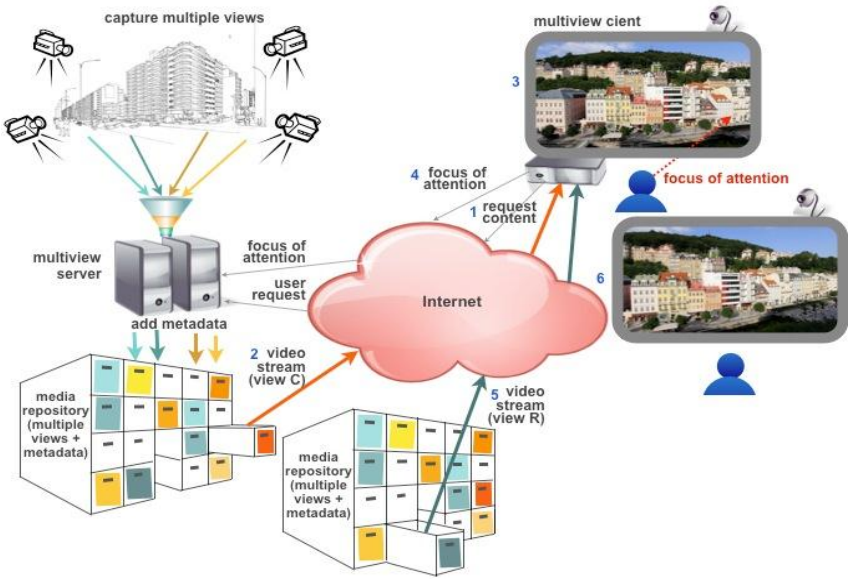


Figure 1. Representation of the multi-view personalized solution.

The client application uses an off-the-shelf Web camera and the FaceAPI recognition software [19], to track head movement in real-time. Periodically, x, y, z, yaw, pitch and roll data are captured, interpreted and compared to previously acquired data. If significant changes are detected, an indication is sent to the server, which selects the new pair of views to send, corresponding to the point of interest of the user in the scene. This control data is sent via an UDP channel, whereas the video data is transmitted using HTTP. More precisely, the MPEG DASH protocol is used to control the transmission of video data from the server to the

client. In this way the system is also able to offer a sustained quality of service, even when network conditions deteriorate.

Using a software network agent, the client continuously monitors the network conditions to decide whether it is necessary to switch to a lower quality or if it is possible to switch to a higher quality. The indication of network conditions is used by the DASH client to decide which segment to request from the server, in a transparent way to the user.

## IV. Client Application

The development of a real-time client application to collect data to detect the focus of attention of the user, analyze it and consequently send requests to the server, posed a number of unforeseen challenges. Since several areas would need to be addressed, the client application was divided into four separate modules, each one with specific requirements and outputs which contribute to the final outcome of the application. Fig. 2 graphically represents the modules present in the client application.

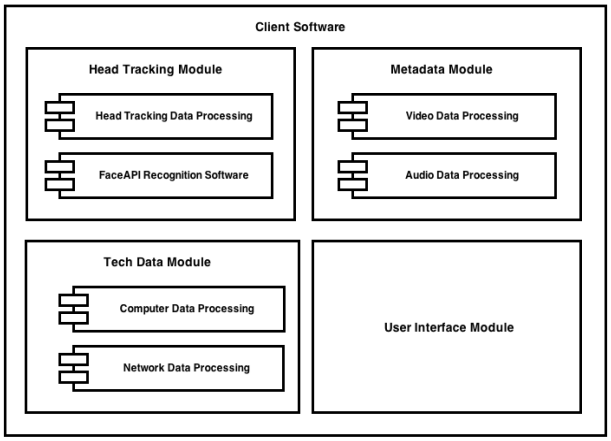


Figure 2. Structure of the software modules in the client application.

### A. Head-Tracking Module

Reliable head tracking software was required so that any kind of trustworthy eye gaze results could be obtained within the client application. Initial tests with the OpenCV suite delivered unsatisfactory results and for that reason, investigation began on the availability of other face recognition packages. FaceAPI, a commercial and educational head tracking solution, demonstrated to be a suitable solution [4]. Due to the fact that FaceAPI is only available on Windows platform, a hybrid C++/C# application was then devised, mixing low level code available through FaceAPI with an attractive and powerful graphical user interface, capable of showing multiple data streams in real-time.

### B. Tech Data Module

Aside from the head tracking data which is processed by the client application, further crucial data was needed to select the optimal stream quality. CPU usage, RAM occupation and network bandwidth are required for a correct decision on the stream which will be selected by the client application on a given moment. This data is collected in real-time by the client application and presented to the user through the graphical user interface. Based on that data, a network agent automatically selects the appropriate video stream quality, which then sends an UDP view switching message to the MPEG DASH Video Player. On the arrival of that message, the MPEG DASH Video player outputs the request for a new stream to the server, which then selects the best view for that situation.

### C. Metadata Module

In order to correctly identify the video and audio content being sent from the server to the MPEG DASH Video Player, two separate modules located on the client application are used to process video and audio packets. Information such as sample rate, number of channels, bit-rate and video resolution is presented through the graphical user interface of the client application, allowing the users to identify possible problems with the incoming content, at any given time.

### D. Graphical User Interface

The graphical user interface is responsible for the gathering all the data from the underlying modules and presenting it to the end users. Being a custom built interface, this GUI also integrates the FaceAPI video feed, allowing the user to identify the head tracking mechanism response to the current environment and lightning conditions, factors which condition the performance obtained with this module. Output control messages, shown in several textboxes within the GUI, are used to identify the current status of the client application. Right next to these control messages, the GUI also allows the manual override of the selected pair of views and its quality, besides the default automatic selection mechanisms. Network configuration is possible through this user interface, allowing the user to change network settings according to the scenario where the application is being used.

## V. MPEG DASH Server Framework

While the client application is responsible for the view switching requests, the MPEG DASH Video Player is responsible for the processing of those packets and respective server stream requests. Based on an open-source GPAC framework, developed with MPEG DASH support and made available for Windows and Linux platform, the MPEG DASH Video Player (MP4Client) supports the latest specification of this particular multimedia format. In order to receive the messages sent by the client application, support for UDP raw sockets messages was added to the MPEG DASH Video player. This leads to further development and refinement of the internal mechanisms used in this application, allowing it to change any MPEG DASH stream on the fly. For this to happen, the MPEG DASH coding software provided by the GPAC framework (MP4Box) required specific parameters while coding input MP4 files. This way, the software would provide specific M4S and MPD index files capable of being recognized and used on the custom

version of MPEG DASH Video player developed for this project. The coded contents can then be stored on a common Apache web server and are sent to the MPEG DASH Video Player via HTTP, allowing them to be made available and used on any conventional scenario or situation. Fig. 3 illustrates the high level concept of the server structure.

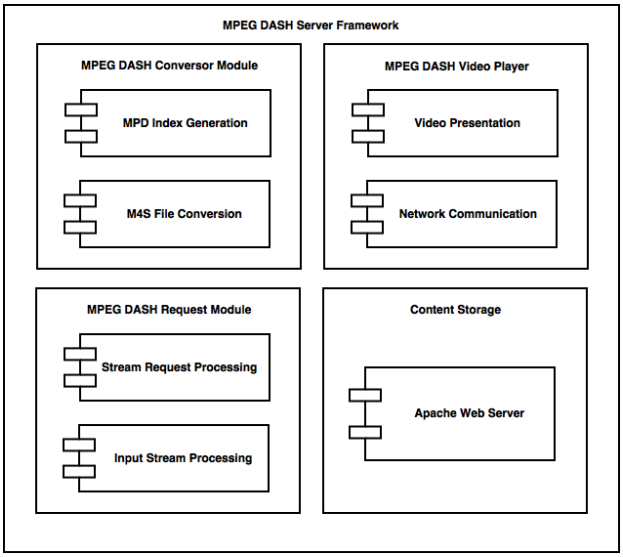


Figure 3. Structure of the modules in the MPEG DASH server framework.

## VI. TESTS

To evaluate the performance of the proposed solution in a real world environment, a testbed was used to recover data such as lag between requests from the client application to the server and video player and visual artifacts perceived in real time viewing. For this test, while one desktop computer was used as video player and content holder, other laptop was used for the client application with integrated head-tracking recognition software. Results are shown in Table I.

TABLE I. TEST DEVICES SPECIFICATIONS

| Device | Desktop Computer | Laptop |
|---|---|---|
| OS | Windows 7 Pro Service Pack 1 x64 | Windows 7 Enterprise Service Pack 1 x86 |
| Processor | Intel Core 2 Duo 2.2 GHz | Intel Core i7-3770 3.4 GHz |
| Memory | 16 GB | 4 GB |

In order to test the effect of the quality and view switching on the solution, two types of tests were carefully planned. On the first test, side-by-side view switching was tested and the time lag between the moment of the client ś request and the actual view switching on the video player was monitored. The side-by-side videos used on this test was previously coded using the specifications shown on Table II.

The two devices were connected together via LAN, created specifically for this testbed. One Cisco WRT610N v2 router was used for this purpose, without any specific configuration besides the common setup needed by this device.

TABLE II. SIDE BY SIDE VIEWS SPECIFICATIONS

| Views | Resolution | Bitrate | Video Codec | Frame rate |
|---|---|---|---|---|
| Left | 1920x1080 | 3739 kbps | mp4a.40.2 | 25 fps |
| Center | 1920x1080 | 3739 kbps | mp4a.40.2 | 25 fps |
| Right | 1920x1080 | 3739 kbps | mp4a.40.2 | 25 fps |

The first test, which would evaluate the performance of this solution based in MPEG-DASH while switching between side-by-side views, was repeated three times with the same conditions and the actual times obtained are shown on Table III, Table IV and Table V.

TABLE III. 1ST VIEW SWITCHING TEST

| View Switch | Laptop | Desktop Computer | Video Player |
|---|---|---|---|
| Left > Right | 10:52:40 | 10:52:41 | 10:52:42 |
| Right > Center | 10:53:45 | 10:53:46 | 10:53:47 |
| Center > Left | 10:54:05 | 10:54:06 | 10:54:07 |

TABLE IV. 2ND VIEW SWITCHING TEST

| View Switch | Laptop | Desktop Computer | Video Player |
|---|---|---|---|
| Left > Right | 11:04:01 | 11:04:02 | 11:04:03 |
| Right > Center | 11:04:10 | 11:04:11 | 11:04:12 |
| Center > Left | 11:04:20 | 11:04:21 | 11:04:22 |

TABLE V. 3RD VIEW SWITCHING TEST

| View Switch | Laptop | Desktop Computer | Video Player |
|---|---|---|---|
| Left > Right | 11:08:20 | 11:08:21 | 11:08:22 |
| Right > Center | 11:08:40 | 11:08:41 | 11:08:42 |
| Center > Left | 11:08:51 | 11:08:52 | 11:08:53 |

Since head-tracking conditions are difficult to match on repeated tests, even in this specific testbed, a manual view switch was used on the client application. This way, all data can easily and reliably be compared throughout these various tests.

On this controlled scenario, the delay between the moment when the user request a view switch and the actual moment when that view is presented on the MPEG DASH Video Player is practically the same throughout these tests. While the desktop computer receives the request from the laptop one second after it was done, the video player only shows the new view one second after that request as arrived to the desktop computer.

The second test would allow us to evaluate the effect of the bitrate on this solution. The views used on this test were previously coded with the specifications presented on Table VI.

 

TABLE VI. QUALITY VIEWS SPECIFICATIONS

| Views | Resolution | Bitrate | Video Codec | Frame rate |
|---|---|---|---|---|
| Very Low | 1920x1080 | 1112 kbps | mp4a.40.2 | 25 fps |
| Low | 1920x1080 | 1613 kbps | mp4a.40.2 | 25 fps |
| Medium | 1920x1080 | 2680 kbps | mp4a.40.2 | 25 fps |
| Highest | 1920x1080 | 3739 kbps | mp4a.40.2 | 25 fps |

As it was the case on the first round of tests, manual quality switching was used for these tests. This way, reliable and comparable data would be achieved, allowing its comparison with other results collected on further experiments. The results of these tests are presented on Table VII, Table VIII and Table IX.

TABLE VII. 1ST QUALITY SWITCHING TEST

| View Switch | Laptop | Desktop Computer | Video Player |
|---|---|---|---|
| Very Low > Low | 11:18:10 | 11:18:11 | 11:18:12 |
| Low > Normal | 11:18:40 | 11:18:41 | 11:18:42 |
| Normal > Highest | 11:19:00 | 11:19:01 | 11:19:02 |
| Highest > Very Low | 11:19:30 | 11:19:31 | 11:19:32 |

TABLE VIII. 2ND QUALITY SWITCHING TEST

| View Switch | Laptop | Desktop Computer | Video Player |
|---|---|---|---|
| Very Low > Low | 11:20:15 | 11:20:16 | 11:20:17 |
| Low > Normal | 11:20:45 | 11:20:46 | 11:20:47 |
| Normal > Highest | 11:21:25 | 11:21:26 | 11:21:27 |
| Highest > Very Low | 11:22:45 | 11:22:46 | 11:22:47 |

TABLE IX. 3RD QUALITY SWITCHING TEST

| View Switch | Laptop | Desktop Computer | Video Player |
|---|---|---|---|
| Very Low > Low | 11:24:01 | 11:24:02 | 11:24:03 |
| Low > Normal | 11:24:20 | 11:24:21 | 11:24:22 |
| Normal > Highest | 11:24:35 | 11:24:36 | 11:24:37 |
| Highest > Very Low | 11:24:50 | 11:24:51 | 11:24:52 |

Regardless of the resolution or bitrate used on each of the views, the solution achieved the same results when switching from one view to another. The desktop computer took approximately one second to process the request sent by the client application and the MPEG-DASH Video Player took another second to show the new view. These results show that, regardless of the video specifications of the contents used and the remote location of the MPEG DASH Video Player and the contents, this solution delivered low delays in real time usage. To complement the data collected during these tests, the head tracking automatic switch was tested. Corresponding results are shown in Table X, quantifying the user perception of the view switching process.

TABLE X. HEAD TRACKING SWITCH TEST

| Head Rotation | Head | Client Application |
|---|---|---|
| Center > Right | 15:20:45 | 15:20:47 |
| Center > Left | 15:21:04 | 15:21:07 |
| Left > Right | 15:21:43 | 15:21:45 |
| Right > Center | 15:22:25 | 15:22:27 |
| Left > Center | 15:23:00 | 15:23:03 |

With the head tracking software correctly following the user, the worst case scenario for view switching in the client application was 3 seconds. If all variables are accounted for, the worst result for the complete process (head tracking recognition, client application request, desktop computer packet processing and MPEG DASH Video Player presentation) is 5 seconds, from the moment when the user rotates its head to the actual view switching.

## VII. CONCLUSIONS AND FUTURE WORK

The results obtained during test sessions show the true potential of this solution regarding the usage of head tracking recognition along with adaptive streaming with MPEG DASH and 3D contents, while delivering low latency timings. Even with high bit-rate, high resolution videos, the solution was capable of achieving good latency values on a controlled local network environment. The consistency of the results also prove that, regardless of the load put on the network, view switching control and quality selection mechanism, the output achieved is practically identical on real world situations.

Given the high interest on public presentations of this solution, an increasing demand for the usage of the prototype on other real life scenarios was presented. One research area which is currently being worked on adapts this solution for public monitoring situations, where live feeds from multiple cameras are being broadcasted to the MPEG DASH Video Player. The end user is then able to control the cameras via head tracking recognition, presenting a complete 360º view of the monitored environment. Along with this work, further refinement of the current prototype needs to be addressed. Allow multiple and simultaneous user connections, reduce latency times and optimize communication timings between sub-modules are top development priorities which are scheduled for the near future. Further research on HEVC and its inclusion on this particular prototype is also planned.

## ACKNOWLEDGMENT

The work presented in this paper was partially supported by the North Portugal Regional Operational Programme (ON.2– O Novo Norte), under the National Strategic Reference Framework (NSRF), through the European Regional Development Fund (ERDF) and by the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) within project UID/EEA/50014/2013.

## REFERENCES


[1] K. Mueller, A. Smolic, K. Dix, P. Merkle, and T. Wiegand, "Coding and intermediate view synthesis of multi-view video plus depth," in *Proc. IEEE International Conference on Image Processing*, November 2009, pp. 741-744.

[2] P. Merkle, K. Müller, and T. Wiegand, "3D video: Acquisition, coding, and display," *IEEE Transactions on Consumer Electronics*, vol. 56, no. 2, pp. 946-950, May 2010.

[3] ISO/IEC 23002-3:2007, Information technology–MPEG video technologies–Part 3: Representation of auxiliary video supplemental information.

[4] ISO/IEC 14496-10:2008, Amendment 1- Multiview Video Coding (MVC).

[5] M. M. Hannuksela, D. Rusanovskyy, W. Su, L. Chen, R. Li, P. Aflaki, D. Lan, M. Joachimiak, H. Li, and M. Gabbouj, "Multiview-video-plus-depth coding based on the advanced video coding standard," *IEEE Transactions on Image Processing*, vol. 22, no. 9, pp. 3449-3458, September 2013.

[6] K. Muller, "3D high-efficiency video coding for multi-view video and depth data," *IEEE Transactions on Image Processing*, vol. 22, no. 9, pp. 3366-3378, May 2013.

[7] A. Vetro and K. Müller, "Depth-based 3D video formats and coding technology," in *Emerging Technologies for 3D Video: Creation, Coding, Transmission and Rendering*, 1st ed. Chichester, ch. 8, UK: Wiley, 2013, pp. 139-161.

[8] K. Müller, "3D extensions for high-efficiency video coding," *IEEE Multimedia Communications Technical Committee E-Letter*, vol. 9, no. 1, pp. 20-22, January 2014.

[9] G. Saygili, C. G. Gurler, and A. M. Tekalp, "Evaluation of asymmetric stereo video coding and rate scaling for adaptive 3D video streaming," *IEEE Transactions on Broadcasting*, vol. 57, pp. 593-601, April 2011.

[10] Y. Mori, N. Fukushima, T. Yendo, T. Fujii, and M. Tanimoto, "View generation with 3D warping using depth information for FTV," *Image Communication*, vol. 24, no. 1-2, pp. 65-72, January 2009.

[11] N. Ozbek, A. M. Tekalp, and E. T. Tunali, "Rate allocation between views in scalable stereo video coding using an objective stereo video quality measure," in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 1, pp. 1045-1048, April 2007.

[12] J. S. Lee, F. Simone, and T. Ebrahimi, "Video coding based on audio-visual attention," in *Proc. IEEE International Conference on Multimedia and Expo*, 2009, pp. 57-60.

[13] E. R. P. Pantos. (April 2015). HTTP Live Streaming. [Online]. Available: https://tools.ietf.org/id/draft-pantos-http-live-streaming-16.txt

[14] ISO/IEC 23009-1:2012, Information technology–Dynamic adaptive streaming over HTTP (DASH)–Part 1: Media presentation description and segment formats.

[15] G. E. Raney, K. Rayner, and S. C. Sereno, "Eye movement control in reading: A comparison of two types of models," *Journal of Experimental Psychology: Human Perception and Performance*, vol. 22, no. 5, pp. 1188-1200, 1996.

[16] Y. Nakano, C. Conati, and T. Bader, *Eye Gaze in Intelligent User Interfaces-Gaze-based Analyses, Models and Applications*, Springer 2013.

[17] BBC News. Samsung Galaxy S4 eye-tracking smartphone unveiled. [Online]. Available: http://www.bbc.co.uk/news/technology-21791023

[18] BBC News. Eye-controlled Gaze TV unveiled by Haier and Tobii. [Online]. Available: http://www.bbc.co.uk/news/technology-19441860

[19] FaceAPI. New commercial API license in 3D visualization. [Online]. Available: http://www.seeingmachines.com/wp-content/uploads/2014/10/sm11024-0-API-DiOMatic.pdf


**Maria Teresa Andrade** gained her PhD in Telecommunications, Electrical and Computing Engineering, from the University of Porto, Portugal in 2008. Her areas of expertise are multimedia networking, context-aware multimedia and content adaptation. She has authored or co-authored more than 50 scientific articles.

**Tiago Soares da Costa**, born in Porto on the 17th of June 1985, received his Master in Informatics Engineering from ISEP – Instituto Superior de Engenharia do Porto, Portugal. He is currently working as a researcher at INESC TEC, located in Porto. His main research interests include areas such as 3D applications, MPEG-DASH and Android/Java development.