A DASH-based Mulsemedia Adaptive Delivery Solution

Ting Bi
Dublin City University
Dublin, Ireland
ting.bi2@mail.dcu.ie

Antoine Pichon
Dublin City University
Dublin, Ireland
antoine.pichon2@mail.dcu.ie

Longhao Zou
Shenzhen University
Shenzhen, China
zoulonghao@gmail.com

Shengyang Chen
Dublin City University
Dublin, Ireland
shengyang.chen5@mail.dcu.ie

Gheorghita Ghinea
Brunel University
London, United Kingdom
george.ghinea@brunel.ac.uk

Gabriel-Miro Muntean
Dublin City University
Dublin, Ireland
gabriel.muntean@dcu.ie

ABSTRACT

The Dynamic Adaptive Streaming over HTTP (DASH) is a protocol designed for adaptive multimedia delivery. This paper describes an adaptive multimedia and mulsemedia delivery system which employs an extension of DASH to support multiple sensorial media. Testing using a real-life test-bed shows how user satisfaction increases when employing the mulseplayer and users are exposed to multi-sensorial content.

CCS CONCEPTS

• **Human-centered computing** → **Graphical user interfaces**; **User interface design**;

KEYWORDS

DASH, Mulsemedia, Adaptation, Synchronization

1 INTRODUCTION

There are five human senses: visual (sight), auditory (sound), tactile/haptic (touch), olfactory (smell), and gastronomy (taste). Multimedia applications are omnipresent in our day-to-day activities. However, the information they provide is only composed of audio and video, which means they only use two human senses. "Mulsemedia" refers to the combination of multimedia components (video and audio) and objects to target other human senses (e.g. touch, smell and taste). As mulsemedia is essentially about using these multiple media objects to communicate information to users, achieving synchronization between the component media objects that make up the mulsemedia communication systems is essential to the success of these systems. [13].

In order to create an immersive multimedia environment that stimulates user experience with multiple media elements engaging three or more human senses and allow content adjustment in existing cloud-based content delivery networks, a novel Dynamic Adaptive Streaming over HTTP (DASH)-based MulSemedia delivery solution (DASH-MS) has been proposed.

This paper is organized as follows. Section II introduces the state-of-the-art related works in mulsemedia communications. Section III presents the DASH-based mulsemedia delivery system. Section IV described the multi-sensory effect synchronization mechanism, Section V analyzes the subjective test results. Finally, Section VI concludes the paper.

2 RELATED WORKS

Mulsemedia term was first coined in 2010 in [4] and refers to media elements which engage three or more human senses. Current mulsemedia applications can use two standards designed by The Moving Picture Experts Group (MPEG): MPEG-7 [5] and MPEG-V [12]. MPEG-7 is an standard of the beginning of the 2000s designed to describe multimedia content data, whereas MPEG-V is designed to interface with virtual worlds.

Adaptation techniques have been widely exploited in content delivery system, especially for the video delivery over the Internet. Moreover, various industry and academic research proposed relevant solutions to address the problems during video streaming while satisfying the different requirements of both transmission environment and users' quality of experience. In terms of the different requirements for video streaming, the proposed adaptation schemes from the literature are divided into five categories [1, 3, 9, 11, 15, 17]: 1) Content-aware adaptive solutions which relate to content-aware encoding and content classification algorithms; 2) Quality-oriented/QoS-aware solutions which consider the transmission channel conditions and QoS requirements; 3) User Experience-aware solutions which mainly consider the users' Quality of Experience when adapting the quality levels of the video stream by getting the clients' feedbacks; 4) Energy-efficiency/saving solutions which adopt green algorithms for saving more power on server side or client side during streaming; 5) Other solutions that perform video adaptation.

Dynamic Adaptive Streaming over HTTP (DASH), referred to also as MPEG-DASH, is a standard which enables adaptive bitrate video delivery over the Internet. The media content is partitioned into one or more segments and is delivered from conventional HTTP web servers to the client using HTTP [10]. However MEPG-DASH does not indicate the actual adaptation mechanism that is employed during the media content delivery. This adaptation mechanism is designed separately and many solutions have already been proposed, including Quality-Oriented Adaptive Scheme (QOAS) [7, 8], Prioritized Adaptive Mechanism (PAM) [16], Region of Interests Adaptive Scheme (ROIAS) [2], Enhanced Energy-aware Device Oriented Adaptive Scheme (E2DOAS) [18], etc. Additionally MPEG-DASH was designed for adaptive multimedia delivery, and does not consider adaptive mulsemedia content delivery.

Adaptive multimedia delivery methods can be divided into different categories, depending on the object to be adapted. Indeed, in order to maintain a high QoE, some methods can use usersâĂŹ

preferences or device energy/power levels. Other methods adapt the quality of the video/audio component to network conditions. Similarly, adaptive mulsemedia solutions could use some network delivery-related characteristics, user profile information or multi-sensory device-related data. However to the best of authors knowledge, they are the only researchers who have proposed an adaptive mulsemedia delivery solution to date [14].

ADAptive MulSemedia delivery solution (ADAMS) proposed in [14] employs an algorithm which extends the video quality adjustment process described in QOAS [6] for multimedia to suit mulsemedia data. The server receives feedback regarding both network conditions and user preferences on sensory effects from client and makes an adaptive decision. However, ADAMS is not based on the DASH standard, cannot be employed in conjunction with existing DASH videos and therefore cannot be easily embedded in future streaming systems.

It is particularly relevant to design and build a DASH-based adaptive application capable to deliver mulsemedia content. This is based on an extension of the DASH protocol which was designed for multimedia content only. The proposed extension to the existing DASH protocol is meant to support adaptive mulsemedia content delivery.

3 SYSTEM ARCHITECTURE

The architecture of the DASH-based adaptive multimedia and mulsemedia delivery system is illustrated in Figure 1. The system is deployed at both client and server sides. An the client side, a mulse-player renders various sense segments into video/audio streaming and multi-sensorial effects (haptic, olfaction, etc.) which will be deployed via different devices. At the server side, the cloud-based platform annotates different sense information and encodes all information into an adaptive multimedia and multi-sensory stream and associates it with two MPD files: "Multimedia MPD" and "Multisensory MPD".

3.1 Encoding

Figure 2 illustrates the encoding architecture for Adaptive Multimedia and Multisensorial Delivery System. Multimedia content delivery is very well developed. In this system, the open source tools "mp4box"[1] and "bento4"[2] are used to encode the original video/audio content in MPEG DASH. For all other senses content/information, an annotation tool is proposed to generate "mulsemedia MPD", which follows the data structure designed for this system and described in this paper.

3.2 Delivery

After encoding, the sender buffer will deliver the "Adaptive Multimedia and Multi-sensory Stream" to the cloud network based on the network condition and user preference, and it will be received in the receiver buffer on the client side. Then, by decoding and rendering, a mulsemedia player on the client side will enable the video/audio content playout, as well as the control of the multi-sensory devices to generate the multi-sensory user experience.

[1] mp4box: https://gpac.wp.imt.fr/mp4box/
[2] bento4: https://www.bento4.com/developers/dash/

4 SYNCHRONIZATION MECHANISM

4.1 Mulsemedia Player

The mulsemedia player is illustrated in Fig. 4. The mulsemedia player contains two components: the dash multimedia player "multi.js" and the mulsemedia effect synchronizer "mulse.js". Both "multi.js" and the "mulse.js" are JavaScript application on server side which executed by a web browser. "multi.js" is a dash multimedia player based on "dash.js"[3] with user profile. When the web page is load, the "multi.js" downloads video elements from cloud-based platform and plays it on web page. The "mulse.js" downloads a "mulsemedia MPD (Multimedia Presentation Description)", which contains all the URL addresses of the mulsemedia segment files. Mulsemedia segment files are progressively downloaded by the "mulse.js" which is synchronized with the multimedia DASH player "multi.js".

4.2 Mulsemedia MPD and Segment

One of the major feature introduced by the DASH-MS is the concept of "mulsemedia MPD". It is inspired by the concept of MPD in DASH. A MPD in DASH is a manifest file describing the structure of a multimedia video/audio and the URL addresses of Initialization segments. The "mulse.js" is using a similar (and simplified) manifest file called "mulsemedia MPD". Both "multimedia MPD" and "mulsemedia MPD" are XML documents. The "mulsemedia MPD" contains all the URL addresses of the multi-sensorial effect-related segment files.

After downloading the "mulsemedia MPD", the "mulse.js" can download segment files. They are three different types of segment files: haptic, olfaction and wind. Each segment file corresponds to a duration of 10 seconds in the video. For the same duration of 10 seconds there can be several types of effects. Thus, there can be up to 3 segments (one haptic, one olfaction and one wind) for the same 10 seconds duration. All mulsemedia segment files are using the JSON format.

Once the mulsemedia player downloads the mulsemedia segment files, the mulsemedia player fills the mulsemedia buffers, which is a table of mulsemedia segments. The buffer can contain 4 different kinds of segments which differ in terms of the "type" as follows:

- "real", for "real segments", are mulsemedia segments that contain multi-sensorial effects. These segments are referred to as "full" into the "mulsemedia MPD",
- "empty segment", for segments that do not contain anything. These segments are referred as "empty" into the "mulsemedia MPD",
- "download in progress", for "full" segments that have not been downloaded yet,
- "end segment", for the segments whose starting times are after the end of the media.

When the mulsemedia player needs to download a mulsemedia segment file (place a "real segment" in the buffer), it first puts a "download in progress segment". Then, it sends an HTTP GET request to obtain the corresponding segment files. When the first segment file is received, the "download in progress segment" is replaced by a "real segment", which contains the mulsemedia effects.

[3] dash.js: https://github.com/Dash-Industry-Forum/dash.js

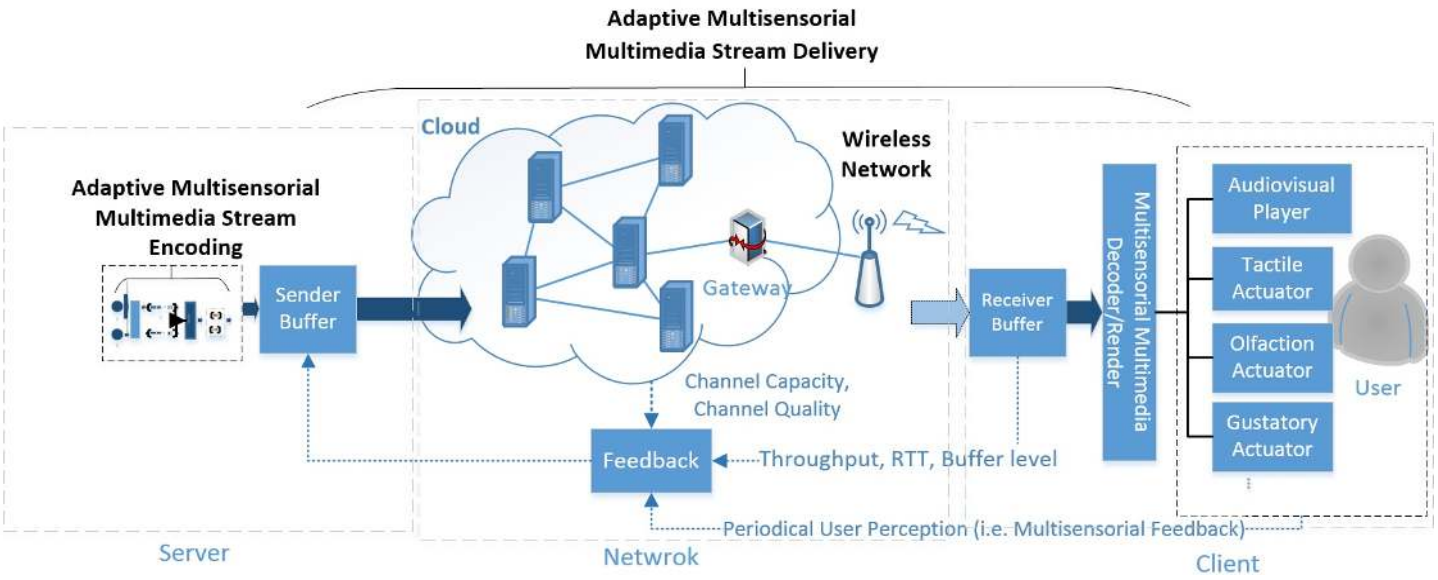

Figure 1: DASH-based Adaptive Multimedia and Mulsemedia Delivery System

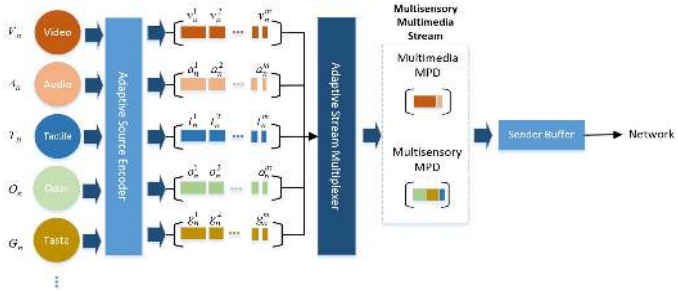

Figure 2: Encoding Architecture

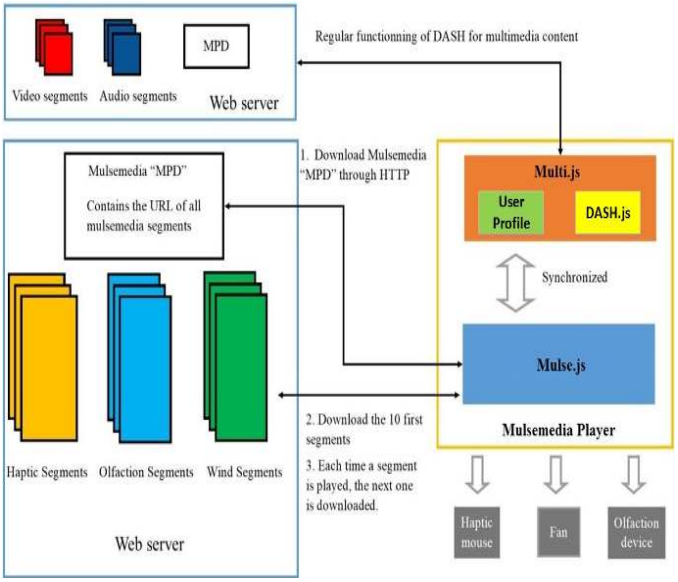

Figure 3: block level architecture of the mulsemedia player

In case another segment file (of another type) arrives, the "real segment" is updated to include information corresponding to the second type of effect.

Each time the application requests a segment whose starting time is after the end time of the video, an "end segment" is put in the buffer. When the mulsemedia player reads an "end segment", it indicates that the end of the video was reached and that the mulsemedia player needs to stop. The purpose of these different kinds of segments is to never have empty positions into the buffer whatever happens. The buffer always contains the 9 following segments (+ the one which is played). If the media stops at 100 s and that player time is at 80 s, the buffer will contain the segment that starts at 80s, the one that starts at 90s and 8 "end segments", which would correspond to segments starting after 100s.

Once a segment is played, another one is downloaded and takes the position in the buffer of the segment that has just been played. Then, the segment situated in the next position in the buffer is played. Therefore, the segment that hast just been downloaded will be played 90 seconds later (assuming a buffer length of 10 and a segment duration of 10 seconds). In case the user goes to a specific moment into the video, the buffer is automatically filled again according to the new time of "multi.js" player.

4.3 Playing and Synchronizing

Several functions are designed in the mulse.js and used to play the effects contained into the buffer: "play-mulsemedia", "synchronize-and-play and "play-segment".

The functioning of the function "play-mulsemedia" is begins by playing segment situated at buffer 0 into the buffer by calling the function "synchronize-and-play". This function returns the current

delay between multi.js player actual time and the starting time of the current mulsemedia segment. This delay can be either positive or negative. Then, "play-mulsemedia" waits for 10 seconds minus the synchronization delay (the result is inferior to 10 seconds if the delay was positive, and superior to 10 seconds if the delay was negative), and plays the segment contained in buffer 1. It also replaces the segment contained in buffer 0 (the one that has just been played), by the one that will be played in 9*10 = 90 seconds later. "play-mulsemedia" is a recursive function. After playing segment contained in buffer 10, it plays segment in buffer 0.

The function "synchronize-and-play" is the one that handles synchronization with the multimedia player multi.js, by computing the delay between the time of multi.js and the starting time of a mulsemedia segment. In case the delay (called "synchro-ms" for "synchronization in milliseconds") is positive, it means the mulsemedia effects are late compared to multi.js, so the segment is played with a starting time equal to "syncho-ms". In case the delay is negative it means the effect are in advance compared to multi.js, so the program waits for a time of "syncho-ms" before playing the segment with a starting point of 0.

The function "play-segment" is called by "synchronize-and-play". It is the function that plays a segment by calling "bind-game-event", "send-game-event", "play-olfaction" and "play-wind" (these are the functions used to play a mulsemedia effect.) A "real" segment is composed of 3 different tables containing all of the effect descriptions ("haptic-effects", "olfaction-effects", and "wind-effects"). The function "play-segment" launch a timer for every effect contained into these tables, and play them when the corresponding timer expires. The timer depends on the starting point of the effect inside the segment and on the starting point of the segment. Indeed, "play-segment" is able to start to play the segment at different starting time. If the segment is started at 3 seconds, effects before 3 seconds are not played and an effect at 5 seconds would be started after 2 seconds. In case of a haptic effects, there is a delay of 250 ms between the binding time and the sending time.A haptic effect is played if the segment starts before the binding time of the effect. In case the segment starts between the binding time and the sending time, the effect is not played.

5 EXPERIMENTAL SETUP

Satisfactory synchronization is essential to guarantee a temporal ordering of multiple sensorial effects in a mulsemedia delivery system [13].In this section, the perceived experience of haptic, olfaction and air-flow effects with low synchronization delay in DASH-based Mulsemedia delivery system is studied. In order to measure the user's satisfaction with the DASH-MS, a subjective experiment is implemented with a sample of individuals due to the human perception varies depending on individual user preference. Also, the synchronization delay is measured.

37 users from different backgrounds (e.g. students, staff, researchers, engineers) participated in the subjective tests. The study was promoted via institutional email and through a specially created "doodle" on-line scheduling tools to arrange and filled with the preset scheduling table. All the participants accepted body haptic effects and were screened against anosmia and color blindness. It took around 10 minutes for each participant to complete the whole test.

All video clips were selected from the video: "Big buck bunny". All video clips are synchronized with haptic, air-flow and olfaction events. They had the same codec settings: a resolution of 1920x1080 pixels, 2 minute long and a frame rate of 30 fps. The video clips 1 with no effects. Multi-snsory effects were correctly synchronised with the video clips 2. Haptic effects had a synchronization delay of +1 seconds whereas wind and olfaction effects had synchronization delays of + 5 seconds in video clips 3. Haptic effects had a synchronization delay of -1 seconds whereas wind and olfaction effects had synchronization delays of -5 seconds in video clips 4.

Compare to the subjective testing of the traditional multimedia content, additional multi-sensory actuator equipments are required to present the mulsemedia effect to the end users with mulseplayer in the DASH-MS. A gaming haptic mouse, a Exhalia scent diffuser and an Arduino-based programmable CPU fan were used to simulate haptic, olfaction and air-flow effects in sync with multimedia content. According to the content scenario, the multi-sensory effects were manually synchronized with the corresponding sensorial content in the multimedia clips by setting the start and end timestamps and generate the "mulsemedia segment file" by a multi-sensory data annotation tools in DASH-MS.

Fig.4 shows the interface of the mulseplayer developed for DASH-MS. On the top of the page we have the DCU, NEWTON and European Union logos. In the video play area. Mulseplayer have these control buttons for play/pause, volume control, full screen functions. The Effects Control Zone is a manual control approach for user to select the mulsemedia effect on their preference. The Mulseplayer will detective the user profile in "multi.js" and enable/disable the multi-sensory effect accordingly. The default settings are enable all multi-sensory effects and shows with green color, once it's be click with "disable" button, the color will change to red and that mulsemedia effect will not occur anymore. "Current Representation" corresponds to the "mulsemedia representation" currently used by the mulseplayer (default setting is "0").

Fig. 5 illustrates the equipment and the test-bed employed in the tests. Fig. 4(a) shows an Arduino-based programmable controlled fan that provides the air flow. The fan's on/off state and strength was controlled by a C++ program using the Arduino board. The gaming haptic mouse in Fig. 4(b) was available from Rival and supports full control of the haptic effect in terms of frequency and duration. Fig. 4(c) presents the Exhalia scent diffuser that enables to diffuse scents from each of its four small fans. The subjective test-bed shown in Fig. 2(d) was built in a separate room in the Performance Engineering Lab at Dublin City University, Ireland.

The testing environment was such setup in order to avoid any potential disturbances during the tests, obeying all the recommendations of ITU-T R.P.910, ITU-T R.P.911 and ITU-T R.P.913.

6 TEST RESULT AND ANALYSIS

6.1 Synchronization Delay

The way effects are synchronized with the video is described in section IV. A function is designed and called "time-checker" in "mulse.js" which measures the difference between the moment an effect is played and the moment it theoretically should have been

Figure 4: Interface of the mulseplayer

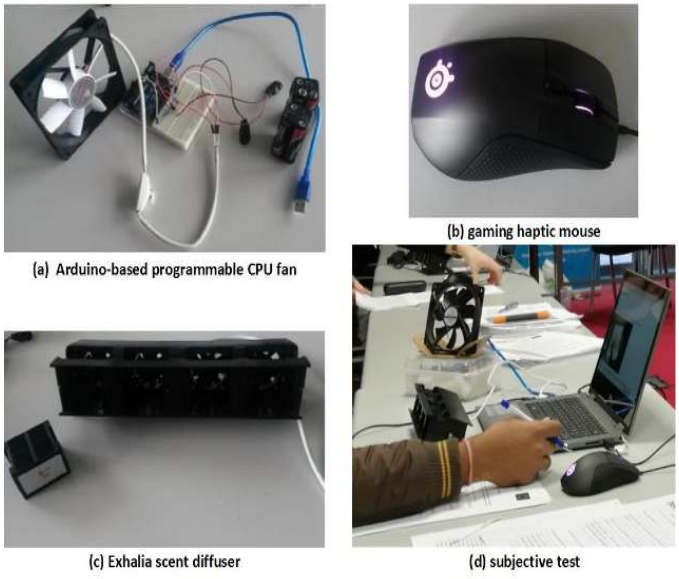

Figure 5: Mulsemedia Equipment and Test-bed

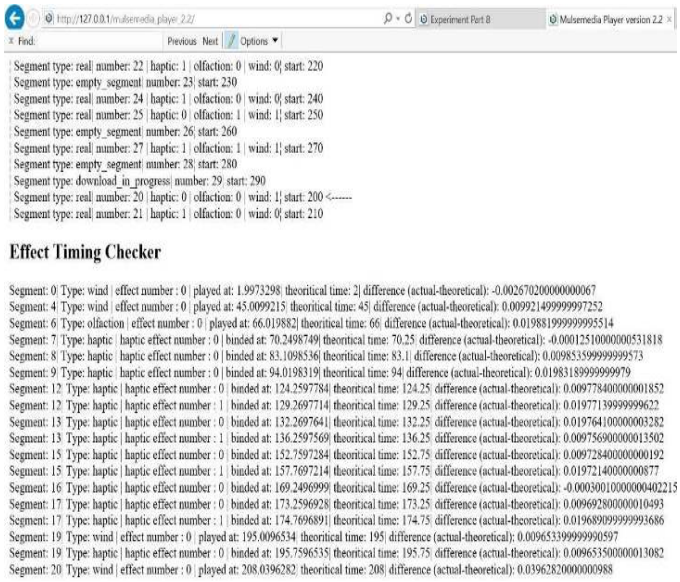

Figure 6: Effect Timing Checker Result

played. A screenshot of results given by this function is provided in fig.6. The evolution of the delay between mulsemedia effects and the video for a 5-min clip with 33 effects (without any pause or seek operation) is present in fig.7. The synchronization mechanism of the mulseplayer is working on mulsemedia segments, whose duration is 10 seconds. Thus, effects contained in the same segments have close synchronization delays. The synchronization mechanism is able to adapt the starting point of mulsemedia segments to the actual time of the player "multi.js". Therefore, delays rarely exceed ±0.07 seconds. The importance of synchronization between effects and video is showed in [13]. The delay for a haptic effect should not exceed 1 second, but wind effects can have delays up to -5 seconds or +3 seconds.

6.2 User Satisfaction

Each participant is asked to answer a set of questions about their perceptual experience of the video in presence of a subset of these multi-sensorial stimuli (i.e. give marks between 1 to 10, from 1 representing "bad" satisfaction level to 10 representing "excellent" satisfaction level). Fig 8 illustrates the results in terms of average

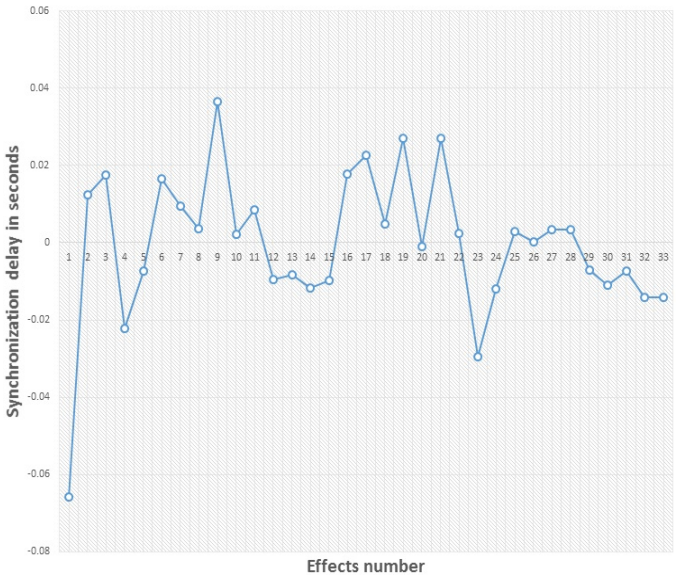

Figure 7: Evolution of the delay between effects and video for a 5-min clip without pause or seek operation

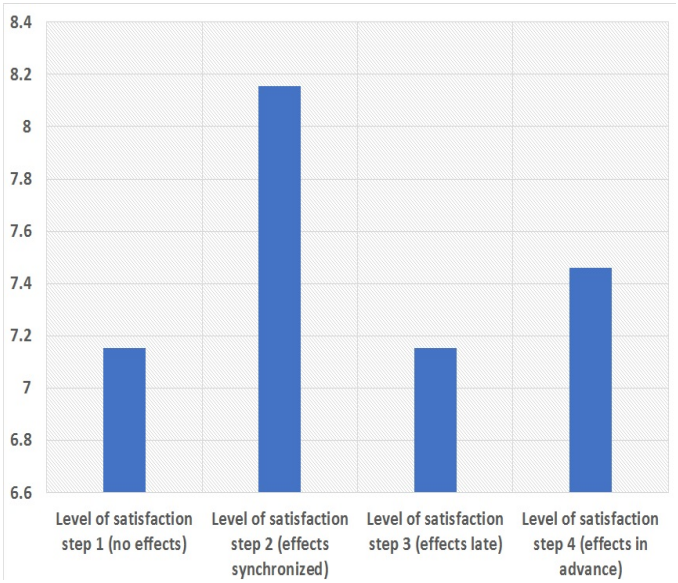

Figure 8: Average Levels of Satisfaction

levels of satisfaction for different synchronization delays. It is interesting to note that the average levels of satisfaction with the mulsemedia effects are always superior or equal to the average level of satisfaction with no effects. As expected, the highest value of the level of satisfaction corresponds to the step when the effects were correctly synchronized. Users seem to have preferred effects in advance than late effects.

7 CONCLUSIONS AND FUTURE WORK

This paper describes, illustrated and tests an adaptive mulsemedia delivery system, based on a multi-sensorial player (mulseplayer). The mulseplayer introduces some important concepts such as the decomposition of effects in segments of different types and the "Mulsemedia MPD". All of these new concepts can be used by future adaptive mulsemedia streaming application. Another interest of the mulseplayer is built based on an existing DASH-based multimedia player (dash.js), so it will work with any further updates of dash.js with minor modification of the code.

The future of mulsemedia depends on the release of new mulsemedia devices and their adoption by the general public. If these devices need to transmit significant amount of data(i.e. VR and tactile gloves based on-line shopping/ real-time gaming, etc), it will be necessary to perform adaptive mulsemedia data delivery. The recent release of virtual reality headsets might consider the expectation of the general public for more immersion. Therefore, the need for mulsemedia devices might sharply increase in the coming years.

ACKNOWLEDGMENTS

This work is supported by the European Union's Horizon 2020 Research and Innovation programme via Grant Agreement no. 688503 for NEWTON (http://newtonproject.eu).

REFERENCES

[1] Velibor Adzic, Hari Kalva, and Borko Furht. 2012. Content aware video encoding for adaptive HTTP streaming. In *Consumer Electronics (ICCE), 2012 IEEE International Conference on*. IEEE, 92–93.

[2] B. Ciubotaru, G. Ghinea, and G. M. Muntean. 2014. Subjective Assessment of Region of Interest-Aware Adaptive Multimedia Streaming Quality. *IEEE Transactions on Broadcasting* 60, 1 (March 2014), 50–60. https://doi.org/10.1109/TBC.2013.2290238

[3] Jen-Wen Ding, Der-Jiunn Deng, Tin-Yu Wu, and Hsiao-Hwa Chen. 2010. Quality-aware bandwidth allocation for scalable on-demand streaming in wireless networks. *IEEE Journal on Selected Areas in Communications* 28, 3 (2010).

[4] Gheorghita Ghinea and Oluwakemi Ademoye. 2010. A user perspective of olfaction-enhanced mulsemedia. In *Proceedings of the International Conference on Management of Emergent Digital EcoSystems*. ACM, 277–280.

[5] Sylvie Jeannin, Leszek Cieplinski, Jens Rainer Ohm, and Munchurl Kim. 2001. Mpeg-7 visual part of experimentation model version 9.0. *ISO/IEC JTC1/SC29/WG11 N* 3914 (2001).

[6] G-M Muntean, Philip Perry, and Liam Murphy. 2004. A new adaptive multimedia streaming system for all-IP multi-service networks. *IEEE Transactions on Broadcasting* 50, 1 (2004), 1–10.

[7] G. M. Muntean, P. Perry, and L. Murphy. 2005. Objective and subjective evaluation of QOAS video streaming over broadband networks. *IEEE Transactions on Network and Service Management* 2, 1 (Nov 2005), 19–28. https://doi.org/10.1109/TNSM.2005.4798298

[8] G. M. Muntean, P. Perry, and L. Murphy. 2007. A Comparison-Based Study of Quality-Oriented Video on Demand. *IEEE Transactions on Broadcasting* 53, 1 (March 2007), 92–102. https://doi.org/10.1109/TBC.2006.886451

[9] Vishwanath Ramamurthi and Ozgur Oyman. 2014. Video-QoE aware radio resource allocation for HTTP adaptive streaming. In *Communications (ICC), 2014 IEEE International Conference on*. IEEE, 1076–1081.

[10] Thomas Stockhammer. 2011. Dynamic adaptive streaming over HTTP–: standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems*. ACM, 133–144.

[11] Hrishikesh Venkataraman, Ting Bi, Ting Wu, Ioana Ghergulescu, and Gabriel-Miro Muntean. [n. d.]. DE-BAR: Device Energy-Centric Backlight and Adaptive Region of Interest Mechanism for Wireless Mobile Devices. *Wireless Personal Communications* ([n. d.]), 1–27.

[12] Kyoungro Yoon, Sang-Kyun Kim, Jae Joon Han, Seungju Han, and Marius Preda. 2015. *MPEG-V: bridging the virtual and real world*. Academic Press.

[13] Zhenhui Yuan, Ting Bi, Gabriel-Miro Muntean, and Gheorghita Ghinea. 2015. Perceived synchronization of mulsemedia services. *IEEE Transactions on Multimedia* 17, 7 (2015), 957–966.

[14] Zhenhui Yuan, Gheorghita Ghinea, and Gabriel-Miro Muntean. 2015. Beyond multimedia adaptation: Quality of experience-aware multi-sensorial media delivery. *IEEE Transactions on Multimedia* 17, 1 (2015), 104–117.

[15] Zhenhui Yuan and Gabriel-Miro Muntean. 2013. A prioritized adaptive scheme for multimedia services over IEEE 802.11 WLANs. *IEEE Transactions on Network and Service Management* 10, 4 (2013), 340–355.

[16] Z. Yuan and G. M. Muntean. 2013. A Prioritized Adaptive Scheme for Multimedia Services over IEEE 802.11 WLANs. *IEEE Transactions on Network and Service Management* 10, 4 (December 2013), 340–355. https://doi.org/10.1109/TNSM.2013.110513.130490

[17] Longhao Zou, Ramona Trestian, and Gabriel-Miro Muntean. 2015. E 2 DOAS: user experience meets energy saving for multi-device adaptive video delivery. In *Computer Communications Workshops (INFOCOM WKSHPS), 2015 IEEE Conference on*. IEEE, 444–449.

[18] L. Zou, R. Trestian, and G.-M.. Muntean. 2015. E2DOAS: user experience meets energy saving for multi-device adaptive video delivery. In *Computer Communications Workshops (INFOCOM WKSHPS), 2015 IEEE Conference on*.