Multimedia Tools and Applications (2020) 79:25143–25170
https://doi.org/10.1007/s11042-020-09214-9

Look ahead to improve QoE in DASH streaming

Román Belda[1] · Ismael de Fez[1] · Pau Arce[1] · Juan Carlos Guerri[1]

Received: 2 July 2019 / Revised: 26 March 2020 / Accepted: 12 June 2020 /
Published online: 30 June 2020

Abstract
When a video is encoded with constant quality, the resulting bitstream will have variable bitrate due to the inherent nature of the video encoding process. This paper proposes a video Adaptive Bitrate Streaming (ABR) algorithm, called Look Ahead, which takes into account this bitrate variability in order to calculate, in real time, the appropriate quality level that minimizes the number of interruptions during the playback. The algorithm is based on the Dynamic Adaptive Streaming over HTTP (DASH) standard for on-demand video services. In fact, it has been implemented and integrated into ExoPlayer v2, the latest version of the library developed by Google to play DASH contents. The proposed algorithm is compared to the Müller and Segment Aware Rate Adaptation (SARA) algorithms as well as to the default ABR algorithm integrated into ExoPlayer. The comparison is carried out by using the most relevant parameters that affect the Quality of Experience (QoE) in video playback services, that is, number and duration of stalls, average quality of the video playback and number of representation switches. These parameters can be combined to define a QoE model. In this sense, this paper also proposes two new QoE models for the evaluation of ABR algorithms. One of them considers the bitrate of every segment of each representation, and the second is based on VMAF (Video Multimethod Assessment Fusion), a Video Quality Assessment (VQA) method developed by Netflix. The evaluations presented in the paper reflect: first, that Look Ahead outperforms the Müller, SARA and the ExoPlayer ABR algorithms in terms of number and duration of video playback stalls, with hardly decreasing the average video quality; and second, that the two QoE models proposed are more accurate than other similar models existing in the literature.

Keywords Adaptive bitrate streaming (ABR) · Dynamic adaptive streaming over HTTP (DASH) · Quality of experience (QoE) · Video multimethod assessment fusion (VMAF) · ExoPlayer

✉ Román Belda
robelor@iteam.upv.es

Extended author information available on the last page of the article

1 Introduction

Providing the best quality at any time depending on the particular context of each client is the main objective of adaptive streaming. To that extent, adaptive streaming works by detecting client device capabilities, such as available network bandwidth, playback buffer size, throughput or video decoding capacity, in order to adapt the video flow to those constraints.

Nowadays, one the most important examples of adaptive streaming is DASH [17], an ISO standard for the transmission of live and on demand content. DASH is based on the segmentation of multimedia content. Hence, media files are encoded with different qualities (called representations), which are then split into small parts called segments. All the information about media segments (such as video resolution or average bitrates) is contained in the Media Presentation Description (MPD). Clients download the segments in a sequential way using the HTTP protocol, and may select different representations for each content segment. Segments are displayed seamlessly in order so, while no problem arises, the result is an uninterrupted video playback. Adaptation process is managed by the client, which implies that the client carries out the corresponding calculations to decide the convenience of a representation switch.

To perform the adaptation, ABR algorithms generally use the average video bitrate to be compared to the estimated bandwidth [23] as well as the playback buffer [26]. However, using average bitrate as a prediction of the needed bandwidth is only barely precise when videos are encoded with a constant bitrate. This type of encoding is indeed creating representations that have quality changes due to the inherent characteristics of different video scenes. Although some works have considered this problem, such as [15, 19, 30], in general, those quality changes are not usually taken into account when ABR algorithms are assessed.

Even though this approach usually works rather well for constant bitrate encoded content, it may cause stalls when the size of video segments changes abruptly. In fact, when a video is encoded with constant quality, the resulting bitstream usually has very variable bitrate due to the different scene types. The reason is that bitrate changes over time, so every single segment of a representation will have, almost inevitably, a slightly variation (if not a huge one). This is true even for constant bitrate encoded videos, as Fig. 1 shows. The figure depicts a video ("Elephants Dream") encoded with constant quality (a target Constant Rate Factor –CRF–) and with constant bitrate. Although both encodings provide similar average bitrate, encoding with a target CRF causes more variation in terms of segment size.

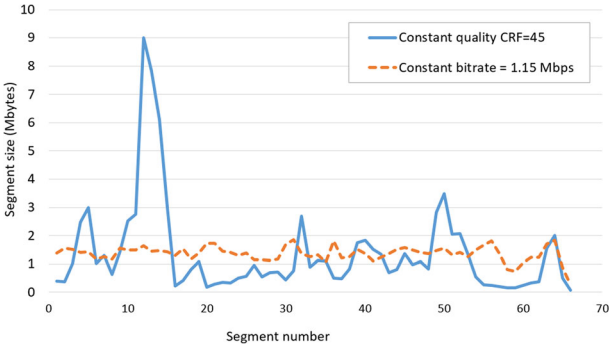

Fig. 1 Segment size comparison for a video encoded with a target quality (CRF) and with fixed bitrate for the video "Elephants Dream"

In order to address these difficulties, this paper proposes an ABR algorithm that takes into account the bitrate of forthcoming segments when choosing the next video representation in order to avoid interruptions during the video playback, which worsen the Quality of Experience (QoE) [6].

Apart from evaluating the performance of the proposed ABR algorithm with objective measures, the algorithm is evaluated using two new QoE models proposed in this work.

The rest of the paper is organized as follows: Section 2 presents the state of the art; Section 3 explains relevant ABR algorithms in the literature and details the proposed ABR algorithm; Section 4 analyzes different models for calculating the Quality of Experience, presenting two new QoE models; Section 5 explains the methodology used to carry out the evaluation presented in Section 6; Finally, Section 7 summarizes the main conclusions.

1.1 Contribution

The main contributions of this work are:

- A new ABR algorithm for DASH, called Look Ahead, which takes into account the bitrate of forthcoming segments so as to reduce considerably the number and duration of stalls, at the expense of hardly decreasing the average bitrate displayed.
- The development of Look Ahead and its integration into ExoPlayer, the library developed by Google to play DASH content on the Android platform. Apart from Look Ahead, other existing ABR algorithms proposed in the literature have been integrated into ExoPlayer, in order to carry out a fair comparison based on a real implementation. It is important to highlight that: 1) Look Ahead is rather simple to be implemented; 2) Look Ahead eliminates the need to include all segment sizes in the MPD file; 3) the evaluation has not been carried out by simulations but using a real performance; 4) it is possible to check the performance of the Look Ahead by accessing a dedicated server set up by the authors [16], which includes a publicly available App that contains the developed Look Ahead algorithm integrated into ExoPlayer.
- Two new models to measure objectively the QoE perceived by the users: one QoE model is based on the bitrate; and the other is based on the calculation of VMAF, which is one of the most popular metrics for video assessment nowadays.

2 State of the art

In recent years, many publications related to DASH have been published, several of them focused on the optimization of the standard and the combination with other solutions to improve the QoE of users [14, 21].

In the same way, multiple implementations of the standard have been released. Regarding these implementations of DASH, it should be mentioned DASH Industry Forum (DASH-IF) [9]. DASH-IF, among other features, elaborates interoperability guidelines and provides a reference DASH player implementation. Among the most remarkable DASH player implementations nowadays, we underline Shaka Player and, especially, ExoPlayer [2], the open source media library developed by Google for the Android platform. The importance of this library is reflected considering that in May 2018 more than 0.2 million apps in Google Play used ExoPlayer [38]. This player library provides

modularity, so users are able to develop and inject new implementations of the different modules. The work presented in this paper has been carried out using the latest version of the aforementioned library: ExoPlayer v2.

With regard to the underlying ABR algorithms, it is worth highlighting some relevant papers that propose new ABR algorithms for DASH. For instance, the authors of [28] provide an own implementation of a DASH ABR algorithm and present an evaluation of the proposed solution compared to popular implementations such as Apple HTTP Live Streaming or Adobe HTTP Dynamic Streaming. That implementation is based on the use of an adaptive algorithm that measures the download time of each segment and builds an adaptation decision out of this download time, the average bitrate of the representations and the buffer level. On the other hand, apart from evaluating some commercial and open source DASH players, the authors of [1] propose an ABR algorithm with the aim of detecting persistent and short-term bandwidth variations in a timely manner to provide smooth bitrate transitions and avoid video freezes. Also, the authors of [40] present a generic dynamic ABR algorithm to be used in both bandwidth and buffer-based approach. A complete study about the state of the art of DASH can be found in [20, 42]. In this context, it is interesting to mention [31], where the authors analyze several adaptive bitrate video proposals by classifying them into rate-based and buffer-based adaptation logics.

In contrast to most of ABR algorithms existing in the literature, which usually consider the average video bitrate to decide the representation to display, our proposed ABR algorithm takes into consideration the variability of the instantaneous bitrate, which is reflected in the segment size. This idea was initially considered by the authors of this paper in [5]. Likewise, the concept has been also considered in [18, 35], where it is proposed an ABR algorithm called SARA that knows the segments size of the whole video in advance, at the expense of modifying the MPD, instead of getting the size in runtime. Also, the authors of [39] use the extension part of the MPD to include information of instant bitrates of each segment to perform a proposed QoE-based video adaptation method. However, the modification of the MPD can lead to a meaningful increase in terms of the size of the MPD, as we will see in the next section. Our proposal, prior to the first segment representation selection, initializes all available qualities by downloading and parsing the *SegmentBase-indexRange* of each representation as defined in the MPD. Therefore, the Look Ahead algorithm does not need any modification in the MPD file.

Moreover, in [41] the dynamics of bandwidth and segment bitrate are considered, but in this case it is used a partial-linear trend prediction to estimate the trend of client buffer level variation. In [30] it is proposed the CAVA ABR algorithm, which takes into account the sizes of upcoming segments as well as its complexity (based on the segment size) and prioritizes the playback of the most complex segments in order to maximize video quality. Finally, in [29] it is presented an ABR algorithm for VBR (variable bitrate) videos, specifically a network-based solution.

It is important to highlight that the ABR algorithm presented in this paper, unlike some aforementioned theoretical ABR algorithms which have been evaluated in simulation scenarios, has been implemented and tested in a real environment. In this sense, among the several ABR algorithms existing in the literature, in this paper, apart from ExoPlayer, we have selected Müller [28] and SARA [18, 35] to carry out the evaluation. The reason is that the papers in which these algorithms are described, unlike most papers, provide enough detail to implement and integrate these ABR algorithms into a real player.

3 ABR algorithms

In this section, the proposed representation adaptation algorithm, called Look Ahead, is detailed. First, different ABR algorithms existing in the literature are explained, such as the ABR algorithm used by ExoPlayer or the Müller algorithm.

3.1 ExoPlayer adaptive algorithm

The ExoPlayer library contains a built-in adaptation algorithm in charge of managing the representation changes by default if no additional implementation is provided. The input parameters of the algorithm are tracks, buffer size and an estimation of the available bandwidth.

When deciding the representation of the next segment, the algorithm first calculates the best representation that fits in the current available bandwidth. This is done by checking all available qualities and selecting the representation with the highest average bitrate lower than or equal to the estimated bandwidth ($\widehat{bw}$), which is weighted by a factor λ, as shown in (1).

$$q[i+1] = \max\left\{Q : bw_{q_j}[i+1] \leq \lambda \cdot \widehat{bw}\right\}, \tag{1}$$

where $Q = \{q_0, q_1, \ldots, q_{k-1}\}$ is the vector with the k available qualities. With the best average bandwidth fitting representation, if it differs from the current representation, the algorithm will switch to the new representation, excluding the following scenarios (2):

- If the new selected representation bandwidth ($bw[i+1]$) is higher than the previous ($bw[i]$) and the buffer size (b) is lower than a minimum buffer (β_{min}) –by default 10 s–.
- If the new selected representation bandwidth is lower than the previous and the buffer size is higher than a maximum buffer level (β_{max}) –by default 25 s–.

$$\text{if } (bw[i+1] > bw[i] \text{ and } b < \beta_{min}) \text{ or } (bw[i+1] < bw[i] \text{ and } b > \beta_{max}) \text{then } q[i+1] = q[i]\ . \tag{2}$$

This algorithm uses the default implementation of the bandwidth estimator and the buffer manager offered by the ExoPlayer library.

3.2 Müller algorithm

The Müller algorithm, proposed in [28], provides its own adaptation method for DASH, and it is widely used in open source software. The algorithm uses the available bandwidth and the buffer level to calculate a new maximum bandwidth: the lower the buffer level, the lower the maximum bandwidth of the next segment, and vice versa. This maximum is compared to the average bitrate of each representation to select the highest representation that accomplishes that the bitrate is lower than the maximum bandwidth of the next segment, as shown in (3).

$$max\_bw(s_i) = \begin{cases} bw(s_{i-1}) * 0.3 & if \quad 0 \leq bl_i < 0.15 \\ bw(s_{i-1}) * 0.5 & if \quad 0.15 \leq bl_i < 0.35 \\ bw(s_{i-1}) & if \quad 0.35 \leq bl_i < 0.50 \\ bw(s_{i-1}) * (1 + 0.5 * bl_i) & if \quad 0.50 \leq bl_i \leq 1 \end{cases} \tag{3}$$

where $i \in [1, n]$ is the segment index, n is the number of segments that compose the video, bl_i is the buffer level when downloading segment i, and $bw(s_i)$ is the function that returns the bandwidth measured during the download of segment i. The default implementation is used for bandwidth estimation and buffer management.

3.3 Segment aware rate adaptation algorithm (SARA)

The SARA algorithm, proposed in [18], considers the segment size variation in addition to the estimated path bandwidth and the current buffer occupancy to predict the time required to download the next segment. The solution is based on a modified MPD file that contains the information about segment sizes.

Also, the algorithm uses a throughput estimation with weighted harmonic mean. Specifically, the weighted harmonic mean download rate for n downloaded segments is given by:

$$H_n = \frac{\sum_{i=1}^{n} \omega_i}{\sum_{i=1}^{n} \frac{\omega_i}{d_i}}, \tag{4}$$

where ω_i is the weight proportional to the size of segment i, and d_i is the download rate of segment i. The time to download the next segment is predicted by ω_{n+1}/H_n, which is compared to different buffer thresholds to decide the next representation. As shown in the evaluation section, this method of estimating the bandwidth provides smooth variations of video representations but reacts slowly to sudden throughput changes which can lead to playback stalls.

The defined strategies to choose the next representation are: fast start, representation decrease, additive increase, increasing by one level, aggressive switching, delayed download or keeping the current representation.

One of the main drawbacks of SARA algorithm is the need of modifying the MPD, since this modification could increase the size of the MPD considerably. For instance, as provided by the authors of the algorithm [12], a video of 77 min with a segment size of 4 s has a MPD with a size of 1.6 MB. This size corresponds to the aggregated size of the first 76 segments of the lowest quality representation which, in turn, represents around 5 min of video. In contrast, the same MPD but without SARA modifications has a size of 9.2 kB.

3.4 Look ahead algorithm

With the aim of avoiding stalls and rebufferings during the playback, this paper proposes an ABR algorithm called Look Ahead that takes into account the bitrate variability of the available representations. The main objective of Look Ahead is to provide a continuous playback while maximizing video quality. In this way, when calculating the representation chosen for the next segment $i+1$, it is intended to provide the maximum quality, as long as no stalls occur, among the k available representations $Q = \{q_0, q_1, \ldots, q_{k-1}\}$, where q_j is the representation of the segment j (note that $q_j < q_{j+1}$).

Look Ahead is an iterative process consisting on computing the average bandwidth of the forthcoming z segments for all representations from $z = 1$ to θ, where θ is the maximum number of forthcoming segments into consideration to calculate the average rate. The algorithm selects, on each iteration, the highest representation of which average bitrate of the next z segments, τ_z, is lower than the estimated bandwidth, according to (5).

$$\tau_z(i+1,j) = \frac{\sum_{m=i+1}^{i+z} S_{m,q_j}}{\sum_{m=i+1}^{i+z} t_m}, \quad \tau(i+1,j) < \widehat{bw}, \tag{5}$$

where i is the current segment, S_{m,q_j} is the size of segment m for the representation q_j, t_m is the duration of segment m, and $\widehat{bw}$ is the estimated bandwidth. Note that t_m will usually be equal in every segment, although it depends on the encoding process.

The selected representation for the next segment, $\xi(i+1)$, corresponds to the lowest representation obtained from the θ iterations, as shown in (6):

$$\xi(i+1) = min\ \{\tau_z\}, \quad z = 1 \ldots \Theta. \tag{6}$$

Considering different iterations when calculating the representation of forthcoming segments is a conservative process that avoids stalls during the playback, since future segments with higher bitrate can make the algorithm to select lower representations than the bandwidth may allow, thus keeping or increasing the buffer depending on forthcoming segments. In this sense, the parameter θ could have a great impact on the QoE of users. In fact, when choosing θ to maximize the QoE, there is a trade-off in terms of stalls, video representation displayed and noticeable representation switches.

To see an example, in the particular case of $\theta = 3$, when calculating the representation of segment u, the available rates for the k representations under consideration are first calculated. This means calculating $\tau(u,j)$, where $j \in [0, k\text{-}1]$, and generating vectors $T(u)$ as shown in (7)–(9):

$$\mathrm{T}(u)_{z=1} = \left[\frac{S_{u,0}}{t_u} \quad \frac{S_{u,1}}{t_u} \quad \ldots \quad \frac{S_{u,k-1}}{t_u} \right], \tag{7}$$

$$\mathrm{T}(u)_{z=2} = \left[\frac{S_{u,0} + S_{u+1,0}}{t_u + t_{u+1}} \quad \frac{S_{u,1} + S_{u+1,1}}{t_u + t_{u+1}} \quad \ldots \quad \frac{S_{u,k-1} + S_{u+1,k-1}}{t_u + t_{u+1}} \right], \tag{8}$$

$$\mathrm{T}(u)_{z=3} = \left[\frac{S_{u,0} + S_{u+1,0} + S_{u+2,0}}{t_u + t_{u+1} + t_{u+2}} \quad \ldots \quad \frac{S_{u,k-1} + S_{u+1,k-1} + S_{u+2,k-1}}{t_u + t_{u+1} + t_{u+2}} \right]. \tag{9}$$

In each vector $T(u)$, the chosen representation is the highest j (the highest column in the $T(u)$ vector) that fulfills the condition $\tau(u,j) < \widehat{bw}$, i.e., the necessary rate for downloading that segment must be lower than the estimated bandwidth. When all vectors $T(u)_{z=1..\theta}$ are calculated generating a vector $T_q(u) = \{q_{(z=1)}, q_{(z=2)}, \ldots, q_{(z=\theta)}\}$, the chosen representation of segment u corresponds to the lowest representation of the vector $T_q(u)$, i.e. $\xi(u) = min\ \{T_q(u)\}$.

Note that, when calculating the representation of the last segment, the value of z will be 1. In the case of the penultimate segment, the value of z will be $min\{2, \theta\}$, and so on.

It is important to highlight that the proposed algorithm is only intended for on-demand video services, where it is possible to know the sizes of the ahead segments in runtime by parsing the *SegmentBase-indexRange* byte range of each representation.

4 Quality of experience models

The Quality of Experience is a subjective evaluation parameter to measure the user experience regarding a service. The importance of this parameter in video evaluation has grown

considerably in the last decade, as the amount of related work recently published proves (a complete survey can be found in [4]).

In the literature, there are different proposals to measure analytically the QoE. In the following subsections we first present the proposal explained in [36], then we suggest a new model based on a brief modification of this algorithm and, finally, we propose a new model based on VMAF for measuring the QoE.

4.1 Normalized QoE model

Yin et al. [36] propose a formula where the QoE is calculated through the sum of the QoE of each segment. Thus, Yin et al. define the QoE of video segment 1 through K by a weighted sum of three components: video quality, quality variations and total rebuffering time, as (10) shows.

$$QoE_1^K = \sum_{k=1}^{K} q(R_k) - \lambda \sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)| - \mu \sum_{k=1}^{K} \left(\frac{LR_k}{C_k} - B_k \right), R_k \varepsilon \Re \tag{10}$$

where K is the number of segments of the video, $R_k \in \Re$ (where $\Re$ is the set of all available bitrate levels) is the bandwidth of the selected representation of segment k, $q(\cdot)$ is an increasing function which maps selected bitrate R_k to video quality perceived by user $q(R_k)$, L is the duration (in seconds) of each segment, C_k is the average download speed of segment k, B_k is the buffer occupancy at the instant of time when the segment k is being downloaded, and finally, λ and μ are positive weighting parameters corresponding to video quality variations and rebuffering time, respectively.

With regard to these latest parameters, a small λ implies that the user is not particularly concerned about video quality variability, whereas a large μ indicates that the user is deeply concerned about rebuffering. As stalls, generally, disturb users much more than video quality changes do, the value of μ is usually much higher than λ.

Yin et al. define a normalized QoE model to compare the performance of algorithms to the theoretical optimum, calculated assuming that the future bandwidth is known, as (11) shows:

$$nQoE_1^K = \frac{QoE_1^K}{QoE_{opt}} \tag{11}$$

4.2 QoE model modified

We propose an initial modification regarding the QoE model proposed by Yin et al. The proposed model is shown in (12):

$$QoE_1^K = \sum_{k=1}^{K} q(R_k) - \lambda \sum_{k=1}^{K-1} |q(R_{k+1}) - q(R_k)| - \mu \sum_{k=1}^{K} \left(\frac{LR_k}{C_k} - B_k \right), \quad R_k \varepsilon \Re_S, \tag{12}$$

Apparently, the formula is the same as (10). However, there is a meaningful difference. In this case $R_k \in \Re_S$, where $\Re_S$ has a different set of values for each segment compared to $\Re$. That is, R_k does not belong to a set of available bitrates specified in the MPD, since the bitrate of each representation, generally, changes in every segment. To see an example, supposing that a video is encoded with only one quality, for instance with a bitrate of 1 Mbps, the value of $\Re$ will always be 1 Mbps for each segment, whereas $\Re_S$

could have a different value in each segment around the average bitrate (e.g. 0.77, 0.95, 1.12 Mbps…).

The idea of taking into consideration the specific bitrate in each segment instead of the average bitrate of every representation makes the proposed Yin et al.-based QoE model more accurate since, as explained in the introduction, as the bitrate changes over time, every single segment of a representation has, almost inevitably, some variation.

4.3 VMAF-based QoE model

VMAF is an objective measure that usually has a strong correlation with the QoE: the higher the VMAF, the better the QoE. Specifically, VMAF [3, 24] is a Video Quality Assessment (VQA) method developed by Netflix and used by many tools like FFmpeg and Elecard StreamEye. It uses Visual Information Fidelity (VIF) [32], Detail Loss Metric (DLM) [22] and Temporal Impairment Feature (TI) metrics fused by Support Vector Machine (SVM) regression [8] with a built-in machine-learning trained model. The model has been trained using the opinion scores obtained through a subjective experiment. Figure 2. outlines the VMAF process to calculate frame scores.

VMAF adopts a modified version of VIF that uses each one of the values of the four scales used by VIF, while VIF combines them into a single value. The SVR (Support Vector Regression) uses the six features to generate per-frame value. The final VMAF value is the arithmetic mean of the per-frame values.

It is worth noting that, even encoding with a target quality and variable bitrate, the resulting segment VMAF changes over time. The variation of VMAF in a representation is particularly meaningful when videos are encoded using a target mean bitrate. As an example, Fig. 3 shows the VMAF obtained for each segment for the same video encoded with a target quality (CRF) and a fixed bitrate, both having the same average bitrate (around 1.13 Mbps). Although both provide similar average VMAF (around 87), a video encoded with a target bitrate has more variations in terms of VMAF (a maximum fluctuation of 43.16 against 23.69 with a fixed CRF).

Moreover, since the bitrate and the VMAF have an increasing logarithmic relationship, bitrate variations do not affect equally VMAF depending on the value of bitrate. For example, a slight bitrate variation can imply a high VMAF variation for low bitrates, as we can check in Fig. 4, where an increase of 500 kbps to 1 Mbps implies a VMAF rise of 12, whereas the same increase, but this time between 4 Mbps and 4.5 Mbps, causes a VMAF rise of hardly 0.4.

Both bitrate and VMAF are important objective measures, however VMAF offers a more representative relationship regarding the QoE. Also, due the nonlinearity of bitrate regarding VMAF depicted in Fig. 4, the QoE model proposed by Yin et al., which is based on the bitrate, can be improved. Although it is true that the model is based on an increasing $q(\cdot)$ function that

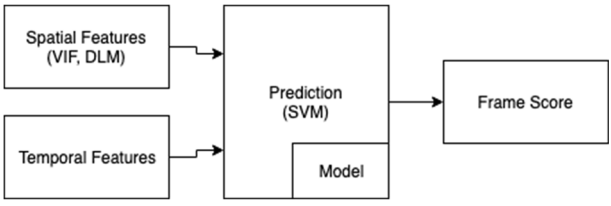

Fig. 2 Outline of the VMAF system

Fig. 3 Segment VMAF comparison for a video encoded with constant CRF and with constant bitrate for the video "Elephants Dream" encoded with VP9

affects the bitrate, and therefore this could be a linear or logarithmic function (among others), this function is not specified in the proposal by Yin et al. [36].

For the abovementioned reasons, we propose a new QoE model based on such an important parameter as the VMAF is. Formula (13) shows the VMAF-based QoE model proposed:

$$QoE'_{VMAF} = \frac{1}{K}\sum_{k=1}^{K} VMAF(\xi_k) - \lambda \frac{1}{K-1}\sum_{k=1}^{K-1} \left|VMAF(\xi_{k+1}) - VMAF(\xi_k)\right| -$$

$$\gamma \cdot \frac{1}{d}\sum_{k=1}^{K}\left[\frac{LR_k}{C_k} - B_k\right] - \delta \cdot T_s, \quad R_k \varepsilon \mathfrak{R}_S, \tag{13}$$

where K is the number of segments of the video, ξ_k is the selected representation of segment k, $VMAF(\xi_k)$ is the VMAF of the selected representation of segment k, d is the total duration of the video (in seconds), L is the duration (in seconds) of each segment, $R_k \in \mathfrak{R}_S$ is the bandwidth of the selected representation of segment k, C_k is the average download speed of segment k, B_k is the buffer occupancy at the instant of time when the segment k is being download, T_s is the start-up delay, and finally λ, γ and δ are positive weighting parameters corresponding to video

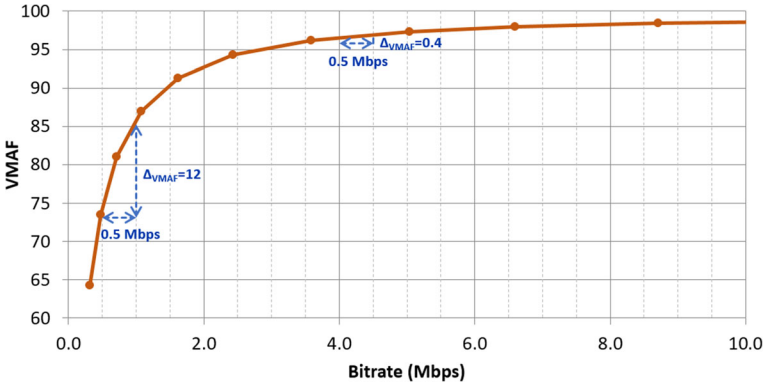

Fig. 4 Relation between bitrate and VMAF for the video "Elephants Dream" encoded with VP9

representation switches, rebuffering time and start-up delay, respectively. As in the previous case, R_k does not belong to a set of available bitrate levels specified in the MPD.

In order to establish a lower bound in case there are many stalls (so as to avoid negative values of the model), $\mathrm{QoE}_{\mathrm{VMAF}}$ is defined as follows:

$$QoE_{VMAF} = \max\left(QoE'_{VMAF}, 0\right) \tag{14}$$

The structure and idea of the proposed formula is similar to the QoE model by Yin, explained in Eq. (10), that is: the VMAF increases the value of the QoE model, whereas both quality changes and the rebuffering duration penalize the QoE. The stalling ratio is the amount of time spent so that video playback is stalled (rebuffering time) divided by the total duration of the video. The proposed formula also includes the effect of the start-up delay. Conceptually, (13) can be expressed as shown in Eq. (15):

$$QoE'_{VMAF} = Average\ VMAF - \lambda * Average\ VMAF\ switches - \gamma \cdot stalling\ ratio - \delta \cdot start_{up} delay. \tag{15}$$

Note that, when rebuffering time is zero, the third term of the formula (15) will also be. It is important to highlight that, the $\mathrm{QoE}_{\mathrm{VMAF}}$ model can provide information by itself about the Quality of Experience of the video playback, without the need of comparison with other values, in contrast to the model proposed by Yin et al., which is normalized with an ideal case, shown in Eq. (11). Thus, the proposed formula has the same scale of VMAF, that is, the maximum value is 100 (an excellent QoE) and the minimum value is 0 (very bad QoE).

In practice, the main difficulty of using the formula is calculating the VMAF of each segment for each representation. This could imply a meaningful processing time, which grows as the number of representations increases. To ease this procedure, we have made available a program in GitHub that calculates VMAF [13], as explained in the next section.

To see an example, making use of Eq. (13), Fig. 5 shows the $\mathrm{QoE}_{\mathrm{VMAF}}$ for different values of stalling ratio and different values of γ. In the figure, the parameter of average VMAF has been fixed to 95, the average VMAF variations have been set to 5, $\lambda = 1$ and $\delta = 0$ (the start-up delay is not considered) so, in case of no stalls, the $\mathrm{QoE}_{\mathrm{VMAF}}$ obtained is 90, as the figure shows. As it can be seen, the parameter γ has a high impact on the $\mathrm{QoE}_{\mathrm{VMAF}}$ model. For example, when $\gamma = 1800$, if the duration of the stalls is 4% of the video playback, we obtain a

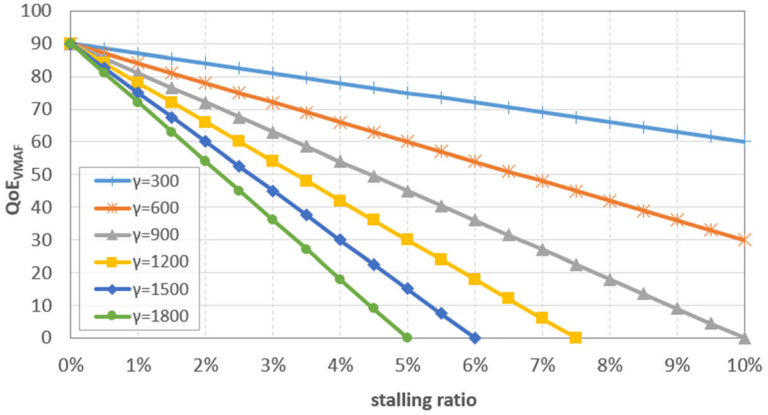

Fig. 5 $\mathrm{QoE}_{\mathrm{VMAF}}$ for different values of stalling ratio and γ

very poor $QoE_{VMAF} = 18$. On the contrary, the same percentage of stalls duration causes an acceptable value of $QoE_{VMAF} = 66$ when $\gamma = 600$. In the studies presented in this paper we have used $\gamma = 900$ since, according to [25], stalling ratio of 1% is considered to be noticeable for users, while values higher than 10% are considered to be not acceptable. Taking into consideration the results shown in Fig. 5, the γ that best accomplishes the previous condition is $\gamma = 900$, since it provides a value of $QoE_{VMAF} = 0$ when the stalling ratio is 10%.

5 Methodology

5.1 Implementation and integration with ExoPlayer

The proposed Look Ahead algorithm and the Müller and SARA algorithms have been developed and integrated into the ExoPlayer v2 library. The use of a real player, instead of emulations, has several advantages for gathering precise data. For example, when using a real implementation, buffer occupancy is updated as soon as a frame is parsed from the HTTP connection and not just once the segment transmission ends. Emulations that do not have this feature cannot be used for detecting video stalls accurately as they may find stalls where there are not.

In order to play video in real devices, ExoPlayer has defined a set of different modules and implementations. The logic of the ABR algorithm is split into three different elements defined by its interfaces: *BandwidthMeter*, *LoadControl* and *TrackSelection*. The first receives periodic updates on transferred bytes and computes a bandwidth estimation that other modules can request. *LoadControl* handles the allocated buffer and instructs the player whether to keep filling the buffer and if the playback should start. Finally, *TrackSelection* may use the data provided by the other two elements to choose the next representation to download.

Concerning the implementation of the algorithms analyzed in this paper, with regard to the default adaptive algorithm of ExoPlayer, the bandwidth estimator and the buffer manager of this algorithm correspond to the default implementation offered by the ExoPlayer library. The default bandwidth estimator uses the percentile 0.5 of a sliding window of weighted values while the buffer manager basically instructs the upstream layers to start downloading segments of the representation selected by the *TrackSelection* implementation when the buffer becomes emptier than 25 s and up to it reaches 30s. On the other hand, due to the lack of information regarding the bandwidth estimation and the buffer manager used by the Müller algorithm, in this work we have used the same default implementations used by the ExoPlayer ABR algorithm. Finally, regarding SARA, the algorithm uses a throughput estimation with weighted harmonic mean, which we have implemented for the ExoPlayer library. As regards to the buffer management, for similarity to the other analyzed studied, we have used the following values of these buffer thresholds for SARA: $I = 5$, $B = 12.5$, $B_\beta = 25$ and $B_{max} = 30$.

With respect to the algorithm proposed, we have only modified the *TrackSelection* interface of ExoPlayer in order to add the new functionality. With the modifications carried out, the implementation of that interface now receives callbacks when a new representation has been initialized. Thanks to an intermediate abstract implementation provided by the library itself (*BaseTrackSelection*), there is no need to update other implementations of the *TrackSelection* interface but only the aforementioned intermediate abstract class.

Also, in order to carry out the measurements, we have developed an ExoPlayer module that limits the HTTP connection bandwidth based on pre-configured tables of time-bitrate values. The bandwidth limiter is in charge of managing the channel bandwidth by limiting the

perceived bandwidth of the downstream elements: *SegmentDownloader*, *LoadControl*, etc. Anyway, our proposal is open to be evaluated with different implementations of the bandwidth estimation and buffer manager.

It is worth noting that Look Ahead does not require any modification (e. g. extra headers nor MPD modifications) to the DASH standard. Specifically, with the objective of having all the information regarding the segment sizes of the representations, the Look Ahead algorithm instructs the underlying player to initialize all available representations before taking any decision. This procedure generates a request of the *Initialization-range* and the *SegmentBase-indexRange* of each representation file [17]. With the information provided by the *indexRange*, the algorithm is able to compute the initial and final byte index of each segment and, thus, the segment sizes. This process generates a little increase in the initial delay due to the network request. For instance, when initializing a representation of a 45-min video with a segment size of 10 s, the size of the initial download would be lower than 6 kB.

Finally, all video playbacks have been carried out using an instance of the official Android 8 emulator running on HP Pavilion dv6 (i7/6GB) with the Ubuntu 18.04 Linux distribution. Also, on server-side, we have used a local instance of Apache 2.4 in order to avoid undesired bandwidth limitations.

5.2 Testbed and evaluation parameters

The videos chosen to perform the evaluation have been created by the Blender Foundation [7]: "Elephants Dream" and "Tears of Steel". Also, it has been used a longer video, whose duration is about 46 min. The video is made up of 4 open source videos: the aforementioned videos "Elephants Dream" and "Tears of Steel" as well as the videos "Sintel" and "Big Buck Bunny". All representations have a Full HD resolution (1080p24) and a segment size of 10 s. The videos have been encoded with VP9, the latest free video coding format developed by Google, and one of the most used video codecs nowadays. Table 1 summarizes the main characteristics of the videos used for the evaluation.

The videos have been encoded using CRF values between 5 (better quality) and 60 (lower quality) in intervals of 5, that is, a total of 12 video qualities. We have encoded videos with CRFs from 5 to 60 in steps of 5 in order to be systematic and to better evaluate the performance of the algorithms in terms of representation switches even though some representations could have never been selected by the player. Fig. 6 shows the bitrate over time of the video "Elephants Dream" for different CRF values (note that, for the sake of clarity, only 4 qualities are shown in the figure).

It is worth highlighting the great variability of bitrate over play time, especially when the sequence is encoded at high qualities. This is one of the key elements of the proposal, since many of existing algorithms do not take into consideration bitrate variability but average bitrate. For

Table 1 Characteristics of the evaluation videos

Video	Duration (s)	Number of segments	Frame size	Codec
Elephants Dream	654	66	1920 × 1080	VP9
Tears of Steel	734	74	1920 × 1080	VP9
Mix (Sintel - Big Buck Bunny - Elephants Dream - Tears of Steel)	2757	276	1920 × 1080	VP9

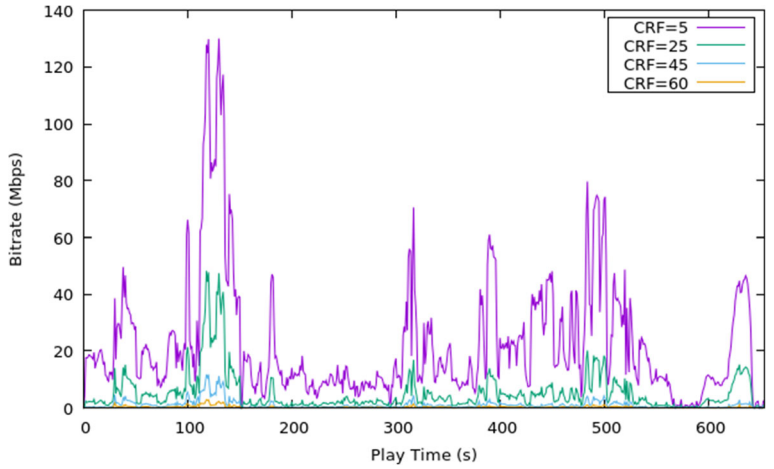

Fig. 6 Bitrate over time for different CRF values of the video "Elephants Dream"

instance, note the peak at 120 s with CRF = 5 (130 Mbps), which quintuplicates the average bitrate of the video (22.75 Mbps) for that quality. When CRF = 45, the value of the bitrate in the aforementioned peak (11.74 Mbps) is ten times higher than the average bitrate (1.13 Mbps).

In the adaptation process, when choosing the representation of the next segment, ABR algorithms will decide to keep the same representation or to change it, either increasing or decreasing the representation. Note that users could perceive a video quality change, which could lead to worsen their QoE, especially if quality decreases, although the more fine-grained video representations, the more unlikely users could detect a quality change between adjacent representations. In this sense, in the evaluation of the Look Ahead algorithm we have fixed the value of $\theta = 1$, meaning that the forthcoming segment (that is, 10 s of the video) will be considered. We have used this value because it is the worst scenario for the proposed algorithm regarding the number of stalls, as shown in the evaluation section. However, a specific evaluation of θ is carried out in one of the studies to check how θ affects video playback.

On the other hand, the adaptation algorithms have been tested on different scenarios: 4 channels with constant bandwidth (1, 2, 5 and 10 Mbps); and 5 channels with variable bandwidth. In particular, the first variable channel (staircase) switches between 2, 4, 8 and 4 Mbps, in loop, every 100 s, whereas the second switches between 2 and 8 Mbps in loop every 100 s. The other three are 4G scenarios obtained from traces of field measurements carried out by the Ghent University, specifically a bus (two scenarios) and a car in motion, publicly available in [11]. In that regard, Fig. 7 shows the throughput for two of these 4G channels. As the figure depicts, although the throughput is high on average (higher than 10 Mbps), there are instants of time where there is a sudden decrease of the throughput, which can lead to a buffer emptying, thus causing stalls in video playback.

Regarding the evaluation parameters, as the ultimate goal of bitrate adaptation is to improve the Quality of Experience of users [36], it is important to analyze the main parameters that affect the QoE. Generally, it is considered that the three key elements of QoE in a video streaming service are: total rebuffering time, average video quality and average quality variations (we will be using representations instead of quality as stated above). These will be the main parameters to analyze in the studies presented in the following section.

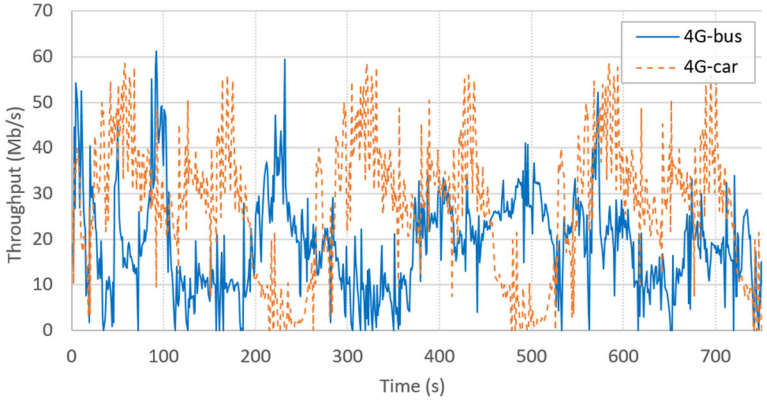

Fig. 7 Throughput for the 4G-bus and 4G-car channels

The algorithms under study have been also evaluated by using three QoE models: the Yin et al. QoE model, the modified Yin et al. QoE model, and the VMAF-based QoE model. In this regard, as no details about the $q(\cdot)$ function are shown in the proposal by Yin et al., we have assumed, for simplicity, that $q(x) = x$, which accomplishes the only requirement of being an increasing function.

Also, to obtain the data shown in the evaluation section, 5 iterations have been carried out for each algorithm, channel and video under study, providing narrow confidence intervals. Specifically, a total of 111 h and 25 min (that is, about 4.5 days) of video have been displayed for the evaluation.

Finally, the authors have set up a web server where the most relevant information used to carry out the evaluations presented in this paper is publicly available. The server can be found in [16]. Among the information, an app with ExoPlayer using the Look Ahead algorithm is available to download and test, as well as the MPD and different representations of the videos used for the evaluation. In this way, interested users can prove the performance of the proposed Look Ahead algorithm for the videos and channels presented in this paper. Furthermore, the authors have developed a program in charge of encoding a video according to input parameters such as target quality or bitrate. This program, publicly available in GitHub [13], calculates the VMAF score of each segment for each representation. This is rather useful when calculating the VMAF-based QoE model proposed in this paper.

6 Evaluation

6.1 Evaluation of look ahead

This section presents the evaluation carried out to compare the performance of the algorithms under study. Specifically, the following parameters are considered when evaluating the four algorithms into seven bandwidth environments for two videos (“Elephants Dream” and “Tears of Steel”): number and duration of stalls, average representation, and number of representation switches.

To summarize the results obtained in different scenarios, as an example, Fig. 8(a-d) show the evaluations of a particular iteration in a 4G channel of a car in motion using the ExoPlayer,

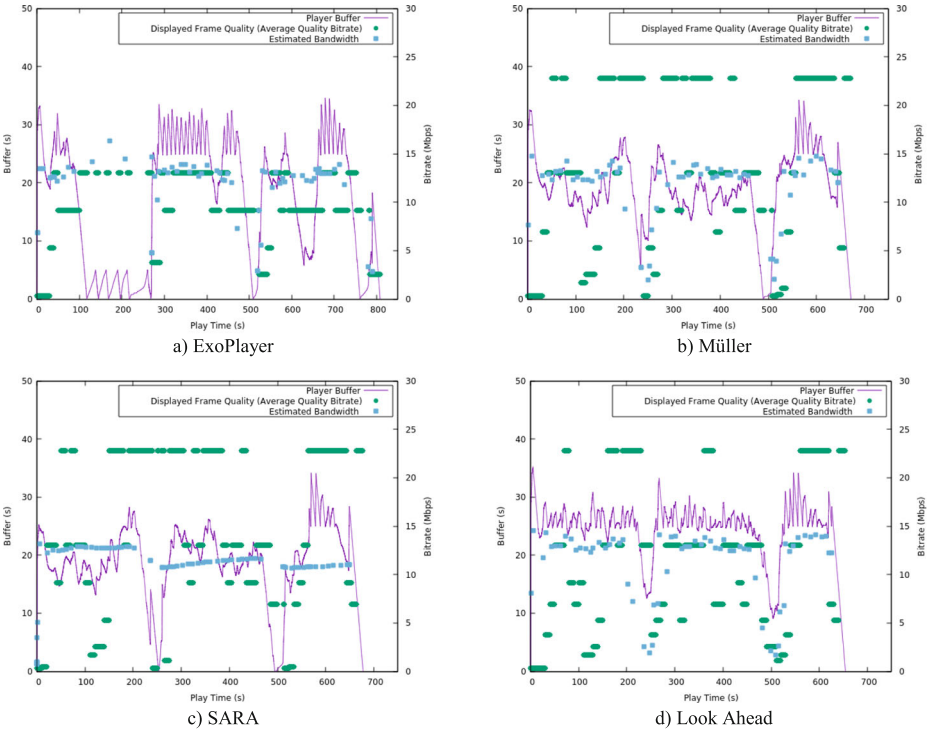

Fig. 8 Evaluation of different algorithms in 4G-car channel for the video "Elephants Dream"

Müller, SARA and Look Ahead algorithms, respectively. As the figures depict, three parameters are analyzed: player buffer level (in seconds); displayed representation according to its average bitrate; and estimation of available bandwidth (in Mbps).

Analyzing the figures, we first see that the adaptive algorithm of ExoPlayer causes several stalls during the playback, whereas Müller causes one stall and SARA two. In contrast, no stalls occur in the Look Ahead algorithm. The playback time when the buffer is empty corresponds, generally, to those instants of time in which the average bitrate of the video is higher or when there is a sudden fall of the throughput. As an example, as shown in Fig. 6, the increase of video bitrate during the play time between 110 and 140 s causes several stalls using the ExoPlayer algorithm (Fig. 8a). Likewise, analyzing Fig. 6 and Fig. 7 we can conclude that, the peak of the video bitrate at the instant of playback around 500 s together with the fall in throughput in the 4G-car channel also at that instant of time lead to one stall both in the Müller and the SARA algorithms, as Fig. 8b and Fig. 8c show, respectively.

The situations of buffer emptiness cause an accumulative delay of the video displayed. In fact, we can see that the video ends 163 s later than it should for the ExoPlayer algorithm (Fig. 8a), 21 s for the Müller algorithm (Fig. 8b) and 24 s for SARA (Fig. 8c). Regarding the buffer, when using the Look Ahead algorithm, the buffer always keeps a stable level (the minimum buffer level during the playback is 9 s) and, consequently, no playback stalls occur. For this purpose, the algorithm selects lower representations when it detects a considerable increase in the size of the following segments (e.g. at instant around 230 s), whereas it selects higher representations when the following video segments have lower bitrates. We also see that most algorithms cause several bitrate switches.

In order to perform a thorough evaluation of the algorithms under study, Table 2 and Table 3 show a comparison for different scenarios in terms of number and duration of interruptions, average representation (which range is between 0 and 11) and average number of representation switches. The best value in each scenario is highlighted in bold in both tables.

Results are in line with those shown in the previous figures, that is, whereas the ExoPlayer, Müller and SARA algorithms suffer from video playback stalls, Look Ahead avoids rebufferings during the playback. In the ExoPlayer algorithm, the fact that a bandwidth increase does not always involve a decrease in the number and duration of interruptions is because the system detects a higher average channel bandwidth and chooses segments of better qualities. Analyzing particular results, when bandwidth is 1 Mbps, the use of the ExoPlayer adaptive algorithm causes up to 5 interruptions for a total time of 94 s, whereas Müller and SARA have 1 stall which duration is about 6 s. In contrast, no playback stall occurs when Look Ahead is used. Although it could seem that having, on average, 1 stall for 6 s for a video of about 10 min is not very meaningful, this could imply an accumulated stalling of 9.17 millions of hours every day in YouTube, taking into account that, according to YouTube [37], the total number of hours of video watched on YouTube every day is about 1 billion.

Moreover, Table 3 reflects that the lack of stalls does not imply a meaningful decrease in the average representation using the Look Ahead algorithm. In fact, the average representation obtained by Look Ahead for all the scenarios under consideration is slightly lower than the average representation of the best case (SARA or Müller, depending on the scenario): 7.33% lower for the video "Elephants Dream" and 9.56% for "Tears of Steel". Also, we see that the average representation for Look Ahead is higher than that obtained by ExoPlayer. Finally, Look Ahead, SARA and Müller algorithms cause many representation switches in comparison to ExoPlayer. The low number of representation switches of the adaptive ExoPlayer algorithm can explain the high value of average number and duration of stalls of this algorithm.

In conclusion, Look Ahead causes less stalls (with less duration) than the other ABR algorithms at the expense of hardly decreasing the average representation. Also, the tables

Table 2 Number and duration of stalls comparison for different adaptation algorithms and for videos 'Elephants Dream' and 'Tears of Steel'

		Number of stalls				Duration of stalls (s)			
	Channel	Look Ahead	SARA	Müller	ExoP.	Look Ahead	SARA	Müller	ExoP.
Elephants Dream	1 Mbps	**0.00**	1.00	1.00	5.00	**0.00**	5.98	5.81	93.76
	2 Mbps	**0.00**	1.00	1.00	5.00	**0.00**	5.95	5.48	93.24
	5 Mbps	**0.00**	**0.00**	**0.00**	6.00	**0.00**	**0.00**	**0.00**	101.18
	10 Mbps	**0.00**	**0.00**	**0.00**	5.20	**0.00**	**0.00**	**0.00**	77.81
	2–4–8-4 Mbps	**0.00**	**0.00**	**0.00**	3.00	**0.00**	**0.00**	**0.00**	47.86
	4G-bus	**0.00**	0.40	0.60	4.00	**0.00**	2.58	9.49	82.56
	4G-car	**0.00**	2.00	0.80	7.80	**0.00**	21.46	13.00	156.73
Tears of Steel	1 Mbps	**0.00**	**0.00**	**0.00**	3.00	**0.00**	**0.00**	**0.00**	34.32
	2 Mbps	**0.00**	**0.00**	**0.00**	3.00	**0.00**	**0.00**	**0.00**	34.20
	5 Mbps	**0.00**	**0.00**	**0.00**	4.00	**0.00**	**0.00**	**0.00**	57.91
	10 Mbps	**0.00**	**0.00**	**0.00**	5.40	**0.00**	**0.00**	**0.00**	51.27
	2–4–8-4 Mbps	**0.00**	**0.00**	**0.00**	4.00	**0.00**	**0.00**	**0.00**	28.91
	4G-bus	**0.00**	**0.00**	**0.00**	4.00	**0.00**	**0.00**	**0.00**	45.35
	4G-car	**0.00**	2.60	0.80	7.80	**0.00**	30.04	7.43	94.66

Table 3 Average representation and number of representation switches comparison for different adaptation algorithms and for videos 'Elephants Dream' and 'Tears of Steel'

		Average representation [0–11]				Number of representation switches			
	Channel	Look Ahead	SARA	Müller	ExoP.	Look Ahead	SARA	Müller	ExoP.
Elephants Dream	1 Mbps	3.27	**3.66**	3.64	1.94	44.80	49.20	49.00	**2.00**
	2 Mbps	3.30	**3.66**	3.64	1.94	45.20	49.20	49.20	**2.00**
	5 Mbps	6.56	**7.41**	7.25	5.71	47.40	48.20	45.20	**2.80**
	10 Mbps	8.17	**8.79**	8.66	6.66	37.20	40.40	42.40	**3.00**
	2–4–8-4 Mbps	6.07	6.17	**6.48**	4.90	46.40	51.60	44.60	**10.80**
	4G-bus	**8.74**	8.06	7.93	8.28	35.00	45.60	43.80	**28.40**
	4G-car	8.53	**9.10**	8.74	8.67	38.00	38.60	37.80	**27.60**
Tears of Steel	1 Mbps	2.78	**3.28**	**3.28**	1.95	52.00	52.40	54.00	**2.00**
	2 Mbps	2.77	**3.29**	3.28	1.95	52.60	53.80	54.60	**2.00**
	5 Mbps	6.07	6.61	**6.62**	5.84	43.60	54.40	50.80	**2.00**
	10 Mbps	7.09	**7.95**	7.94	6.81	48.60	48.20	46.20	**2.20**
	2–4–8-4 Mbps	5.52	5.78	**6.06**	5.05	45.60	53.40	54.20	**11.40**
	4G-bus	6.73	**8.45**	8.20	8.02	43.60	44.20	51.80	**37.40**
	4G-car	7.99	**8.23**	8.17	8.26	50.20	43.80	43.20	**18.80**

reflect that in the most demanding channels (such as 1 Mbps, 4G-bus or 4G-car) there are more stalls than in the least demanding channels. Also, the higher the average channel bandwidth, the higher the average representation.

6.2 Evaluation of QoE models

This section shows the evaluation of the QoE models proposed using the four algorithms, eight bandwidth channels and three videos under consideration. Table 4 shows an evaluation of the algorithms in terms of Quality of Experience. Specifically, three different models are used: the QoE proposed by Yin et al., the QoE by Yin et al. modified and the VMAF-based QoE model. We first see that ExoPlayer provides negative values of the QoE in most scenarios due, mainly, to the duration of the stalls, as we saw in Table 2.

Taking the values of Table 4, Fig. 9 shows the evaluation of the Look Ahead, SARA and Müller algorithms in terms of the relationship of the formula of QoE by Yin et al. divided by the maximum QoE for different channels (highlighted in bold in Table 4). We have considered the maximum QoE as the maximum value of the QoE for the four algorithms under consideration in each particular case. For example, in a constant bandwidth channel of 2 Mbps for the video "Elephants Dream", the best QoE value is provided by Look Ahead, whereas in a constant 5 Mbps channel for the video "Elephants Dream" the best QoE value is provided by SARA. In the figure, we have omitted the ExoPlayer algorithm for the sake of clarity since, due to the high value of stalls duration in some channels, it provides negative values in many cases (and in no case it provides the best value). The values shown in the figure have been obtained by fixing $\lambda = 1$ and $\mu = 6000$. That is, 1 s of rebuffering has the same penalty as the bitrate reduction of a chunk by 6000 kbps. We have used these values as suggested in [33]. Also, each algorithm, for each channel, contains a lower error bar that represents the value obtained when $\mu = 9000$ (a higher penalty for stalls) and an upper error bar that represents the value obtained when $\mu = 3000$. This value of μ is suggested by Yin et al. in

Table 4 Quality of Experience comparison for different adaptation algorithms and for videos 'Elephants Dream' and 'Tears of Steel' ($\lambda = 1$, $\mu = 6000$, $\gamma = 900$, $\delta = 0$)

		QoE representation (Millions)				QoE segment (Millions)				QoE VMAF			
	Channel	Look Ahead	SARA	Müller	ExoP.	Look Ahead	SARA	Müller	ExoP.	Look Ahead	SARA	Müller	ExoP.
Elephants Dream	1 Mbps	**68.15**	37.69	40.03	−527.77	**42.17**	9.94	11.01	−550.48	**76.26**	70.95	70.43	0.00
	2 Mbps	**69.74**	36.27	40.10	−524.55	**42.71**	9.80	12.62	−547.26	**76.55**	70.32	70.88	0.00
	5 Mbps	241.16	**352.39**	340.19	−388.08	162.47	**199.75**	186.74	−507.57	90.68	**92.98**	91.99	0.00
	10 Mbps	464.00	**580.38**	560.56	−150.85	289.87	**341.43**	337.25	−317.70	93.92	**95.11**	94.27	0.00
	2–4–8-4 Mbps	249.27	217.97	**288.99**	−119.11	**131.46**	117.41	122.93	−225.93	**87.04**	85.71	86.99	23.67
	4G-bus	**617.04**	402.49	377.47	−57.12	**363.60**	225.37	174.44	−257.52	**93.24**	89.13	78.04	0.00
	4G-car	**548.60**	499.57	548.47	−442.72	**363.07**	274.16	311.10	−680.17	**93.97**	61.95	72.95	0.00
Tears of Steel	1 Mbps	42.88	**53.94**	52.15	−161.81	47.84	**54.93**	53.97	−183.58	79.47	**83.19**	82.23	37.60
	2 Mbps	42.37	**51.96**	51.19	−161.09	47.76	**54.03**	54.01	−182.85	79.47	82.23	**82.63**	37.75
	5 Mbps	197.01	216.40	**220.29**	−98.01	203.41	**226.83**	226.07	−203.61	94.06	**94.76**	94.68	23.34
	10 Mbps	298.47	427.21	**442.57**	75.03	320.08	**404.53**	398.68	−78.41	95.33	**96.48**	95.97	32.57
	2–4–8-4 Mbps	154.88	153.78	**171.80**	12.24	172.30	183.75	**199.42**	−48.84	91.57	90.59	**92.69**	56.84
	4G-bus	375.31	**545.28**	441.90	229.54	226.49	**507.49**	410.85	79.94	95.94	**96.87**	95.96	40.32
	4G-car	453.31	399.78	**550.97**	77.95	**513.02**	384.93	508.80	−182.86	**95.67**	55.10	83.03	0.00

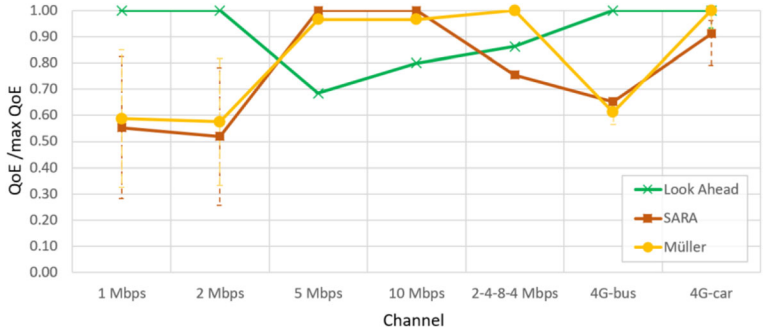

Fig. 9 Evaluation of the QoE model by Yin et al. ($\lambda = 1$, $\mu = 6000$) for the video "Elephants Dream"

[36]. The figure shows that, in the most demanding scenarios (that is, in the 4G scenarios in mobility or in the constant bandwidth channel of 1 and 2 Mbps), the Look Ahead algorithm provides values of the QoE much better than SARA and Müller. The worst case for Look Ahead is the case of a fixed 5 Mbps channel, in which it provides a QoE a 30% lower than SARA. Considering the case of $\mu = 9000$, as the penalty for stalls duration is lower, both Müller and SARA are closer to Look Ahead, especially in the constant 1–2 Mbps channels. On the other hand, when $\mu = 3000$, the differences regarding Look Ahead are even higher in the constant 1–2 Mbps channels.

Figure 10 shows the results obtained when the QoE model by Yin et al. modified is used. It is worth remembering that this proposed model, in contrast to the original QoE model by Yin et al., considers the specific bitrate in each segment instead of the average bitrate of the representations. Results, compared to those shown in Fig. 9, are rather favorable for Look Ahead in comparison to Müller and SARA. In those scenarios where Look Ahead behaved better than the other two algorithms (e. g. 4G scenarios or 1 Mbps), the difference between Look Ahead regarding Müller and SARA increases. On the other hand, in the 5 and 10 Mbps bandwidth channels, the difference of Look Ahead regarding the best algorithm for these scenarios is reduced, whereas in the staircase scenario Look Ahead is the algorithm that provides the maximum QoE considering the modified QoE model. As in the previous figure, the lower and upper bounds represent the QoE value obtained when $\mu = 9000$ and $\mu = 3000$, respectively. In this way, we can conclude that, considering the QoE model by Yin et al. modified, the benefits of the Look Ahead algorithm are more evident.

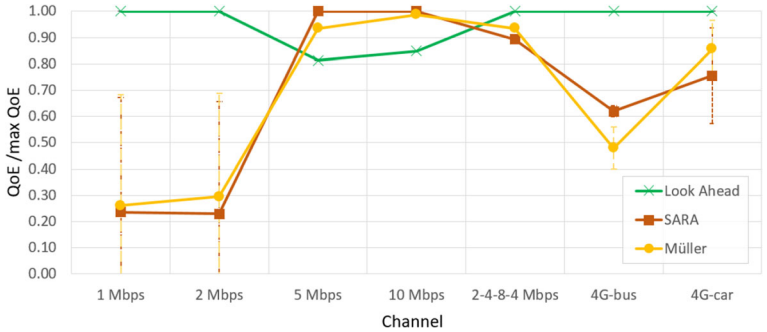

Fig. 10 Evaluation of the modified QoE model by Yin et al. modified ($\lambda = 1$, $\mu = 6000$) for the video "Elephants Dream"

The evaluation of the proposed QoE_{VMAF} model for the video "Elephants Dream" is shown in Fig. 11 (fixing $\lambda = 1$ and $\gamma = 900$). To make an accurate comparison regarding the Yin et al. QoE model shown in Fig. 9, throughout this section we do not consider the start-up delay, since the formula of Yin et al. does not consider it, so $\delta = 0$. As in the previous case, in order to see the behavior of the QoE_{VMAF} model under different conditions, we have also considered the case of $\gamma = 1500$ (lower bound in Fig. 11) and $\gamma = 300$ (upper bound in Fig. 11). In this case, Look Ahead provides the best results for the most demanding scenarios, whereas the difference in the rest of channels compared to the best algorithm is now insignificant. Using this QoE model it is possible to analyze each algorithm independently of the other algorithms. The three algorithms provide good values of the QoE_{VMAF} in all scenarios (the minimum value is 61.95 in SARA for the 4G-car channel). As expected, the lowest values are obtained in the most demanding channels. We can also see the great importance of the parameter γ when there are stalls. For instance, in the 4G-car channel for SARA, when $\gamma = 300$ then $QoE_{VMAF} = 81.64$, whereas this value gets worse considerably ($QoE_{VMAF} = 42.26$) when $\gamma = 1500$.

In order to make a comparison among the analyzed QoE models, we are going to consider a particular case. Specifically, the 4G-car scenario for the mixed video. Analyzing the parameters shown in Table 5, we see that whereas the SARA algorithm has, on average, 9.2 stalls whose duration is 100.44 s, and Müller has 5.6 stalls of average duration 64.37 s, the Look Ahead algorithm has only 4 stalls with a total duration of 24.37 s. The average representation is slightly higher in SARA (8.87) than in Look Ahead (8.81) and Müller (8.66), and the number of representation switches is rather similar in both algorithms (between 161.20 and 168.20). Since the average representation and the number of representation switches are almost equal in the three algorithms, it seems that Look Ahead behaves better than SARA and Müller because of the difference in terms of number of stalls and stalls duration. However, the QoE model proposed by Yin et al. is better in Müller (2.23 M) and SARA (2.12 M) than in Look Ahead (2.10 M). In a real scenario, it is difficult to believe that users perceive a better experience watching a video playback that uses an algorithm that causes more stalls than another that has almost the same average quality and much less stalls. In contrast, the QoE model of Yin et al. modified (proposed in Section 4.2) shows a completely different result since, in this case, the QoE of Look Ahead (1.61 M) is better than the QoE of the SARA algorithm (1.11 M) and the Müller algorithm (1.21 M). Likewise, this result is coherent with the VMAF-based QoE model, which reflects a good value for the Look Ahead algorithm (88.37), a worse value for

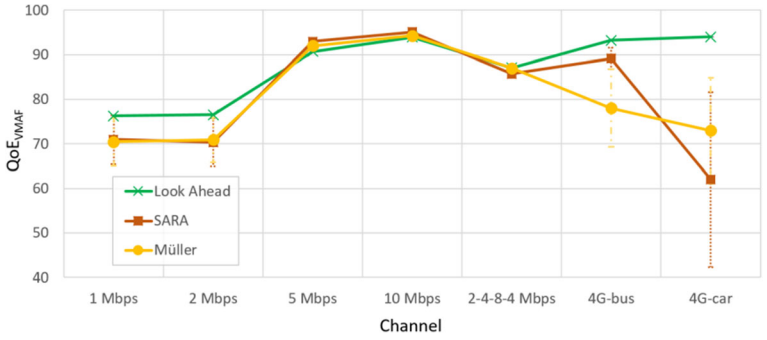

Fig. 11 Evaluation of the proposed VMAF-based QoE model ($\lambda = 1$, $\gamma = 900$, $\delta = 0$) for the video "Elephants Dream"

Table 5 Evaluation of Look Ahead, SARA and Müller for the mixed video ($\lambda = 1$, $\mu = 6000$, $\gamma = 900$, $\delta = 0$)

	Channel	Number of Stalls	Stalls durat. (s)	Initial buff. (s)	Total time stopped (s)	Average repres.	Repres. switches	QoE repres. (M)	QoE segment (M)	QoE VMAF(dB)
Look Ahead	1 Mbps	**0.00**	**0.00**	4.38	**4.38**	3.17	**192.20**	176.27	**193.43**	80.58
	10 Mbps	**0.00**	**0.00**	1.20	1.20	7.87	174.80	1443.09	1212.40	95.91
	8–2 Mbps	**1.00**	**3.71**	1.16	**4.87**	7.87	174.60	1432.60	1201.11	**94.20**
	4G-bus	**0.00**	**0.00**	**0.87**	**0.87**	8.69	174.20	2010.71	**1558.94**	**96.74**
	4G-car	**4.00**	**24.37**	**1.00**	**25.37**	8.81	163.40	2100.26	**1610.78**	**88.37**
SARA	1 Mbps	1.00	9.05	4.24	13.29	3.55	208.40	152.46	144.06	79.74
	10 Mbps	**0.00**	**0.00**	**1.07**	**1.07**	8.13	192.00	1559.89	1152.67	96.11
	8–2 Mbps	4.60	46.91	1.03	47.94	8.23	185.60	1403.60	916.80	79.17
	4G-bus	2.40	42.44	0.88	43.32	**9.03**	**173.20**	**2154.50**	1354.25	81.74
	4G-car	9.20	100.44	1.16	101.60	**8.87**	**161.20**	2119.87	1113.45	60.74
Müller	1 Mbps	1.00	6.86	**3.26**	10.11	**3.62**	194.80	**192.07**	166.72	**80.67**
	10 Mbps	**0.00**	**0.00**	1.28	1.28	**8.74**	**166.20**	**2133.48**	**1473.66**	**96.81**
	8–2 Mbps	1.80	22.48	**0.74**	23.21	**9.16**	**173.40**	**2481.90**	**1650.38**	88.65
	4G-bus	**0.00**	**0.00**	0.93	0.93	8.74	197.80	1956.69	1229.98	95.67
	4G-car	5.60	63.19	1.17	64.37	8.66	168.20	**2227.83**	1214.99	72.72

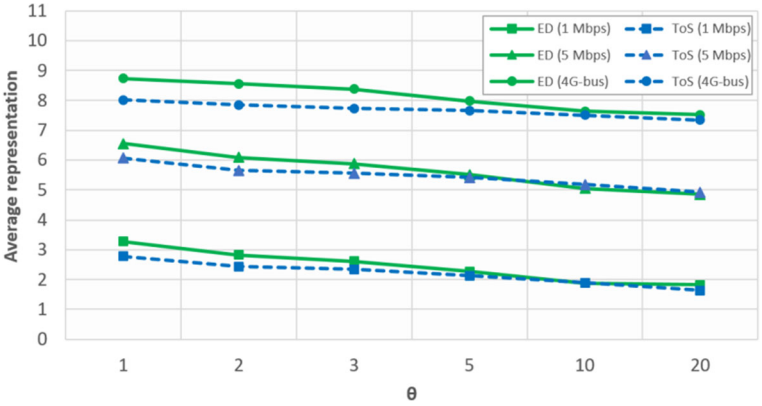

Fig. 12 Average representation in Look Ahead for different

Müller (72.72) and a bad QoE_{VMAF} for SARA (60.74). Similar conclusions arise when we analyze other scenarios, for example, the staircase 8–2 Mbps channel using the mixed video.

In conclusion, we can say that both the QoE model by Yin et al. modified and the proposed VMAF-based QoE model offer more realistic results in terms of Quality of Experience than the QoE model proposed by Yin et al. [36].

6.3 Evaluation of θ in look ahead

Finally, we are going to analyze how the parameter θ affects the video playback. In this case, we evaluate the number of stalls, average representation, number of representation switches and QoE_{VMAF}. We consider two different scenarios: one scenario where no stalls occur, and other where there are stalls during video playback.

Regarding the first scenario, Fig. 12, Fig. 13 and Fig. 14 show the behavior of the Look Ahead algorithm for different values of θ: 1, 2, 3, 5, 10, and 20, using the videos "Elephants Dream" (ED) and "Tears of Steel" (ToS) in constant bandwidth channels of 1 and 5 Mbps, and in the 4G-bus channel. In all cases, no stalls have occurred. Also, the initial buffering is not very meaningful in none of the above cases, with values between 1.4 and 2 s for the 1 Mbps

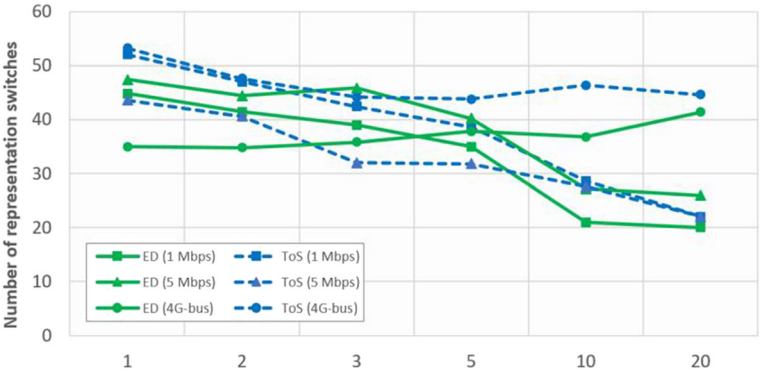

Fig. 13 Number of representation switches in Look Ahead for different

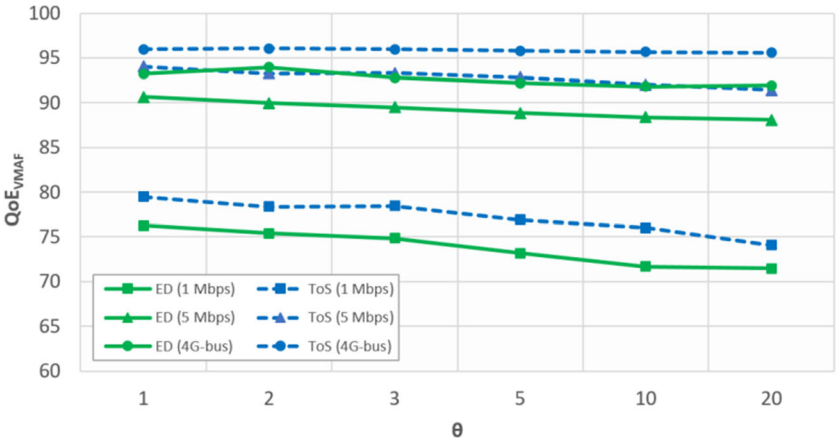

Fig. 14 QoE_{VMAF} in Look Ahead for different $(\lambda = 1, \gamma = 900, \delta = 0)$

channel, values between 0.7 and 1 s for the 5 Mbps channel and between 0.4 and 0.6 for the 4G-bus channel for both videos.

Fig. 12 shows the average representation for different values of θ. We can see how the algorithm is more conservative as θ increases, causing that the average representation decreases. This occurs in both videos. Regarding the number of representation switches, Fig. 13 reflects that, in general, the higher the θ the lower the number of representation switches. Respecting the QoE, Fig. 14 shows the VMAF-based QoE model in the scenarios under study, with $\lambda = 1$, $\gamma = 900$ and $\delta = 0$. As expected from the results shown in Fig. 12, low values of θ provide the best QoE_{VMAF} values, since no stalls occur.

We now analyze the second scenario (where stalls occur) in order to check how high values of θ make the algorithm more conservative, thus reducing the number and duration of stalls. For that, we use the mixed video in a highly variable channel, in this case another 4G scenario for a bus in motion, which we call "4G-bus2."

Fig. 15 shows the number of stalls, average representation (right axis) and QoE_{VMAF} (left axis) for different values of θ. First, we see that, in this case, as θ increases, the number of stalls is reduced (from 3 stalls when $\theta = 1$, to 0 stalls when $\theta = 4$). The number of stalls affects

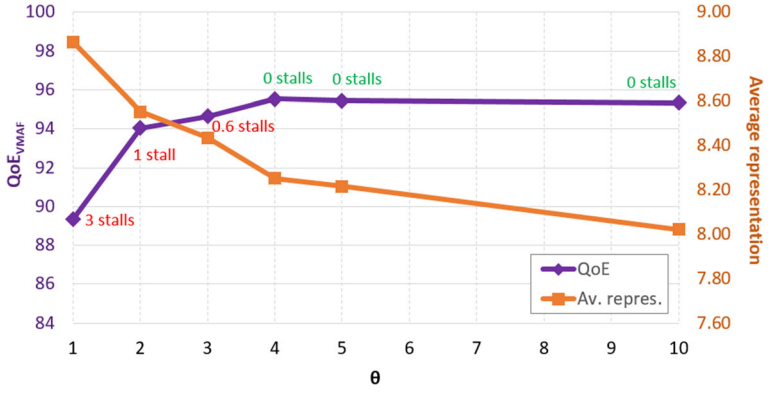

Fig. 15 Evaluation of for Look Ahead in a 4G-bus2 channel using the mixed video

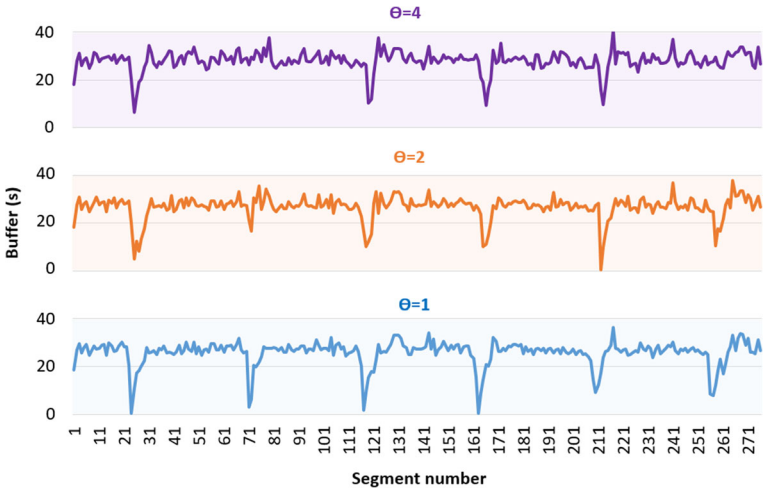

Fig. 16 Evaluation of the state of the buffer for different in a 4G-bus2 channel using the mixed video

directly the QoE_{VMAF}. It can be seen that this parameter increases until there are not stalls (when $\theta = 4$), and then starts decreasing slightly because the average representation decreases as θ increases, as we can see in the figure.

Following the previous study, Fig. 16 reflects the state of the buffer for different values of θ for a specific iteration. Obviously, when the buffer empties stalls occur. In this way, we can see that when $\theta = 1$ there are three instants of time (in segments number 22, 116 and 162) where the buffer gets empty, thus causing 3 stalls, as we saw in Fig. 15. When $\theta = 2$ there is one stall. Finally, the use of a higher θ (e. g. $\theta = 4$) causes a softer fluctuation of the buffer state, not causing any stall.

As conclusion, we can affirm that the optimum θ that provides the maximum QoE_{VMAF} will be the lowest θ that does not cause stalls.

7 Conclusion

This paper has shown that the instantaneous bitrate variability of video contents is a key factor for DASH ABR algorithms. Even in constant bandwidth environments, which would provide enough bandwidth for a continuous playback at the average video bitrate, not considering this information about instantaneous bitrate can lead to stalls during video playback.

The algorithm proposed in this paper, called Look Ahead, takes into account the variability of the bitrate to choose the best representation in order to avoid interruptions. Results prove that both the number and duration of video playback stalls (rebuffering) are highly reduced, compared to the adaptive algorithm used by ExoPlayer, and to the SARA and Müller algorithms. This turns out in good values of the QoE model proposed by Yin et al. [36].

In this sense, the modified QoE model by Yin et al. as well as the VMAF-based QoE model proposed in this work, have been proved to be more accurate models compared to the QoE model originally proposed by Yin et al., according to the evaluations presented in this paper. These two proposed QoE models, although provide a good performance according to the

results shown, have limitations. So, as part of the future work, it is intended to analyze other metrics to propose new QoE models, for example SSIM, PSNR or VQM.

Also, as future work, the proposed VMAF-based QoE model can be improved by considering other parameters that affect the user experience, as the number of stalls. In general, it is more annoying for users having many but short stalls than having few although long stalls [10]. For instance, in video playback, users usually prefer having 1 stall of 10 s than 10 stalls of 1 s. Moreover, it is worth analyzing how the number of representation switches affects the QoE perceived by the users, a topic deeply analyzed in [27, 34]. In this regard, it is interesting to perform different subjective studies that complete the objective evaluation presented and validate the QoE models proposed.

Finally, it is important to emphasize that it is possible to check the performance of Look Ahead by accessing a dedicated server set up by the authors [16], which includes a publicly available App that contains the developed Look Ahead algorithm integrated into ExoPlayer. The App allows to redo the evaluations presented in this paper using the videos and scenarios hereby analyzed. Also, authors have made available a program in GitHub [13] to encode videos and calculate the VMAF of each segment for each representation.

Acknowledgements This work is supported by the PAID-10-18 Program of the Universitat Politècnica de València (Ayudas para contratos de acceso al sistema español de Ciencia, Tecnología e Innovación, en estructuras de investigación de la Universitat Politècnica de València) and by the Project 20180810 from the Universitat Politècnica de València ("Tecnologías de distribución y procesado de información multimedia y QoE").

References

1. Akhshabi S, Narayanaswamy S, Begen AC, Dovrolis C (2012) An experimental evaluation of rate-adaptive video players over HTTP. Signal process. Image Commun 27(4):271–287. https://doi.org/10.1016/j.image.2011.10.003
2. Android Developers webpage, ExoPlayer. Available online at: https://developer.android.com/guide/topics/media/exoplayer.html. Accessed: Jun. (2019)
3. Bampis CG, Li Z, Bovik AC (2018) SpatioTemporal feature integration and model fusion for full reference video quality assessment. IEEE Trans on Circuits and Syst for Video Tech 29:2256–2270. https://doi.org/10.1109/TCSVT.2018.2868262
4. Barman N, Martini MG (2019) QoE modeling for HTTP adaptive video streaming - a survey and open challenges. IEEE Access 7:30831–30859. https://doi.org/10.1109/ACCESS.2019.2901778
5. Belda R (2013) Algoritmo de adaptación DASH: Look Ahead. Master Thesis. Universitat Politècnica de València. http://hdl.handle.net/10251/33359.
6. Belda R, de Fez I, Arce P, Guerri J C (2018) Look ahead: a DASH adaptation algorithm. Proc. of the IEEE Int. Symp. On broadband multimed. Syst. And broadcast., Valencia, Spain: article no. 158. https://doi.org/10.1109/BMSB.2018.8436718.
7. Blender Foundation webpage. Available online at: https://www.blender.org/foundation. Accessed: Jun. (2019).
8. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20-3:273–297. https://doi.org/10.1023/A:1022627411411
9. DASH Industry forum webpage. Available online at: http://dashif.org. Accessed: Jun. (2019)
10. Ghadiyaram D, Pan J, Bovik AC (2019) A subjective and objective study of stalling events in mobile streaming videos. IEEE Trans on Circuits and Syst for Video Technol 29(1):183–197. https://doi.org/10.1109/TCSVT.2017.2768542
11. Ghent University. 4G/LTE bandwidth logs. Available online at: http://users.ugent.be/~jvdrhoof/dataset-4g. Accessed: Jun. (2019).
12. Github webpage. A DASH segment size aware rate adaptation model for DASH. Available online at: https://github.com/pari685/AStream. Accessed: Jun. (2019)

13. GitHub website. Dashgen, Multimedia Communications Group. Available online at: https://github.com/comm-iteam/dashgen. Accessed: Jun. (2019).
14. van der Hooft J, Petrangeli S, Wauters T, Huysegems R, Alface PR, Bostoen T, De Turck F (2016) HTTP/2-based adaptive streaming of HEVC video over 4G/LTE networks. IEEE Commun Lett 20(1):2177–2180. https://doi.org/10.1109/LCOMM.2016.2601087
15. Huang TY, Johari R, McKeown N, Trunnell M, Watson M (2014) A buffer-based approach to rate adaptation: evidence from a large video streaming service. Proc. of the 2014 ACM Conf. On SIGCOMM, Chicago, IL, USA: 187-198. https://doi.org/10.1145/2619239.2626296
16. Institute of Telecommunications and Multimedia Applications website. Look Ahead Demo. Available online at: https://lookahead.iteam.upv.es. Accessed: Jun. (2019)
17. ISO/IEC 23009–1:2014 (2014) Dynamic adaptive streaming over HTTP (DASH) - Part 1: media presentation description and segment formats.
18. Juluri P, Tamarapalli V, Medhi D (2015) SARA: segment aware rate adaptation algorithm for dynamic adaptive streaming over HTTP. Proc. of the IEEE Int. Conf. On Commun. Workshop (ICCW), London, UK: 1765-1770. https://doi.org/10.1109/ICCW.2015.7247436.
19. Juluri P, Tamarapalli V, Medhi D (2016) QoE management in DASH systems using the segment aware rate adaptation algorithm. Proc. of the IEEE/IFIP Netw. Oper. And Manag. Symp. (NOMS), Istanbul, Turkey: 129-136. https://doi.org/10.1109/NOMS.2016.7502805.
20. Kua J, Armitage G, Branch P (2017) A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP. IEEE Commun Surv & Tutor 19(3):1842–1866. https://doi.org/10.1109/COMST.2017.2685630
21. Lee S, Youn K, Chung K (2015) Adaptive video quality control scheme to improve QoE of MPEG DASH. Proc. of IEEE Int. Conf. On Consum. Electron. (ICCE), Las Vegas, NV, USA: 126-127. https://doi.org/10.1109/ICCE.2015.7066348.
22. Li S, Zhang F, Ma L, Ngan K (2011) Image quality assessment by separately evaluating detail losses and additive impairments. IEEE Trans. on Multimed. 13-5:935–949. https://doi.org/10.1109/TMM.2011.2152382
23. Liu C, Bouazizi I, Gabbouj M (2011) Rate adaptation for adaptive HTTP streaming. Proc. of the second annual ACM Conf. On multimed. Syst. (MMSys), San Jose, CA, USA: 169-174. https://doi.org/10.1145/1943552.1943575.
24. Medium webpage (2016) Toward a practical perceptual video quality metric. Available online at: https://medium.com/netflix-techblog/toward-a-practical-perceptual-video-quality-metric-653f208b9652. Accessed: Jun. 2019.
25. Mobile Video Service Performance Study (2015) HUAWEI white paper. Available online at: http://www.ctiforum.com/uploadfile/2015/0701/20150701091255294.pdf.
26. Mok RKP, Luo X, Chan EWW, Chang RKC (2012) QDASH: a QoE-aware DASH system. Proc. of multim. Syst. Conf. (MMSys), Chapel Hill, NC, USA: 11-22. https://doi.org/10.1145/2155555.2155558
27. Moldovan C, Hagn K, Sieber C, Kellerer W, Hoßfeld T (2017) Keep calm and don't switch: about the relationship between switches and quality in HAS. Proc. of the Int. Teletraffic Congr. (ITC), Genoa, Italy: pp. 1-6. https://doi.org/10.23919/ITC.2017.8065802
28. Müller C, Lederer S, Timmerer C (2012) An evaluation of dynamic adaptive streaming over HTTP in vehicular environments. Proc. of the 4th workshop on mob. Video (MoVid), Chapel Hill, NC, USA: 37-42. https://doi.org/10.1145/2151677.2151686
29. Nguyen T, Vu T, Nguyen DV, Ngoc NP, and Thang TC (2015) QoE optimization for adaptive streaming with multiple VBR videos. Proc. of the Int. Conf. On comp., Manag. And Telecommun. (ComManTel), DaNang, Vietnam: 189-193. https://doi.org/10.1109/ComManTel.2015.7394285.
30. Qin Y, H. Shuai, Pattipati K R, Qian F, Sen S, Wang B, Yue C (2018) ABR Streaming of VBR-encoded videos: characterization, challenges, and solutions. Proc. of ACM CoNext 2018, Heraklion, Greece: 366–378. https://doi.org/10.1145/3281411.3281439.
31. Samain J, Carofiglio G, Muscariello L, Papalini M, Sardara M, Tortelli M, Rossi D (2017) Dynamic adaptive video streaming: towards a systematic comparison of ICN and TCP/IP. IEEE Trans on Multimed 19(10):2166–2181. https://doi.org/10.1109/TMM.2017.2733340
32. Sheikh H, Bovik A (2006) Image information and visual quality. IEEE Trans on Image Process 15(2):430–444. https://doi.org/10.1109/TIP.2005.859378
33. Shuai Y, Herfet T (2016). A buffer dynamic stabilizer for low-latency adaptive video streaming. Proc. of the Int. Conf. on Consum. Electron., Berlin: 1–5. https://doi.org/10.1109/ICCE-Berlin.2016.7684742.
34. Tavakoli S, Egger S, Seufert M, Schatz R, Brunnström K, García N (2016) Perceptual quality of HTTP adaptive streaming strategies: cross-experimental analysis of multi-laboratory and crowdsourced subjective studies. IEEE Journal on Select Areas in Commun 34-8:2141–2153. https://doi.org/10.1109/JSAC.2016.2577361

35. Yarnagula H K, Juluri P, Mehr S K, Tamarapalli V, Medhi D (2019) QoE for Mobile clients with segment-aware rate adaptation algorithm (SARA) for DASH video streaming. ACM trans. On multimed. Comput., Commun., and Appl. (TOMM) 15(2):article no. 36 https://doi.org/10.1145/3311749.
36. Yin X, Sekar V, Sinopoli B (2014) Toward a principled framework to design dynamic adaptive streaming algorithms over HTTP. Proc. of the 13th ACM workshop on hot topics in Netw. (HotNets), Los Angeles, CA, USA: 1-7. https://doi.org/10.1145/2670518.2673877.
37. YouTube webpage (2019) Youtube press. Available online at: https://www.youtube.com/yt/about/press. Accessed: Jun. 2019.
38. Youtube webpage, Google I/O '18: Building feature-rich media apps with ExoPlayer. Available online at: https://youtu.be/svdq1BWl4r8?t=2m. Published: May (2018)
39. Yu L, Tillo T, Xiao J (2017) QoE-driven dynamic adaptive video streaming strategy with future information. IEEE Trans on Broadcast 63-3:523–534. https://doi.org/10.1109/TBC.2017.2687698
40. Zhao S, Li Z, Medhi D, Lai P, Liu S (2017) Study of user QoE improvement for dynamic adaptive streaming over HTTP (MPEG-DASH). Proc. of the Int. Conf. On Comput., network. And Commun. (ICNC): multimed. Comput. And Commun., Santa Clara, CA, USA: 566-570. https://doi.org/10.1109/ICCNC.2017.7876191.
41. Zhou Y, Duan Y, Sun J, Guo Z (2014) Towards a simple and smooth rate adaption for VBR video in DASH. Proc. of the IEEE Vis. Commun. and Image Process. Conf, Valletta, pp 9–12. https://doi.org/10.1109/VCIP.2014.7051491
42. Zhou C, Lin C-W, Guo Z (2016) mDASH: a Markov decision-based rate adaptation approach for dynamic HTTP streaming. IEEE Trans. on Multimed 18(4):738–751. https://doi.org/10.1109/TMM.2016.2522650

Affiliations

Román Belda[1] · Ismael de Fez[1] · Pau Arce[1] · Juan Carlos Guerri[1]

Ismael de Fez
isdefez@iteam.upv.es

Pau Arce
paarvi@iteam.upv.es

Juan Carlos Guerri
jcguerri@dcom.upv.es

[1] Institute of Telecommunications and Multimedia Applications (iTEAM), Universitat Politècnica de València, Camino de Vera, 46022 Valencia, Spain