Master's Thesis

Title

A rate control method for QoE improvement in video streaming services based on a human cognitive model

Supervisor

Professor Masayuki Murata

Author

Masayoshi Iwamoto

February 8th, 2019

Department of Information Networking

Graduate School of Information Science and Technology

Osaka University

Master's Thesis

A rate control method for QoE improvement
in video streaming services based on a human cognitive model

Masayoshi Iwamoto

Abstract

Recently, over-the-top video service provider has appeared, and the number of video application users using mobile devices is rapidly increasing. Network providers aim to provide stable network communication quality (Quality of Service; QoS) to users, but the proliferation in mobile traffic makes it difficult. Under such circumstances, QoE (Quality of Experience), which is a measure of the degree of user satisfaction with a service, is attracting attention as an important factor when evaluating the quality of video application service. Currently, most video streaming service providers such as YouTube and Netflix adopt HTTP Adaptive Streaming (HAS) which is one of adaptive bitrate control techniques according to the user and QoS context and, especially Dynamic Adaptive Streaming over HTTP (MPEG-DASH) is widely being spread. In DASH systems, video content is encoded into multiple versions at different bitrates, and the video player can switch to video with different bitrates even during the playback.

The selection of video bitrate by the video player is performed by the adaptive bitrate algorithm (ABR algorithm) which is implemented in the client, and in recent years, many ABR algorithms aiming at improving the user QoE have been proposed. General ABR algorithms estimate QoS between the video client and the video server and collect information available to the client. Based on these information, the algorithms select a video bitrate of the segment to be downloaded next. However, under the mobile network environment, QoS fluctuates due to various factors destabilizing the QoS, such as the inherent variability in signal strength, interference, noise, and user mobility in addition to the increase in mobile traffic. Also, in the application layer where the ABR algorithm operates, the means for estimating QoS is limited. Then, as an error of the estimated QoS becomes larger, frequent switching of the video bitrate and selection of an inappropriate bitrate for the actual QoS may occur. This causes a great decrease of the user QoE.

Note that since the preference for video quality differs user by user, factors for improving the user QoE also differ user by user. For example, some may prefer high video quality or stable video quality, and others may place more emphasis on not stopping video playback. Therefore, it is desirable to select a bitrate in consideration of the user preference, but this issue has not been sufficiently discussed in existing research.

In this research, in order to maximize the QoE of individual users even in the environment where the QoS fluctuates, we propose a method to properly recognize observation information including QoS and select a bitrate suitable for user preference. Here, we assume that a user preference information for video quality is given. Then, we use a QoE model that reflects the user preference with some QoE metrics used in an existing research. Our proposed method first recognizes the condition of the network and application in the client device by using a human cognitive model, the Bayesian attractor model (BAM), which models cognition and decision making of the human brain, as the name suggests, according to the Bayesian inference. And then, based on the cognitive result and the user preference model, our method selects a video bitrate during video playback.

Through computer simulation, even in situations where the available bandwidth of the network fluctuates, bitrate selection correctly recognizes the current situation of the network and the client application and improves QoE for each user with different preference. Simulation results showed that our proposed method increased an average bitrate by up to 16% compared with BOLA-O. This improves QoE by 18%–36% in our user preference model where user prefers high image quality. For a user preference model where user prefers stable image quality, QoE is improved by 52%–121% compared with BOLA-O, where an average bitrate variation is reduced by 98%. We also implemented our proposed method into an MPEG-DASH application service, and showed that our bitrate selection algorithm could provide an appropriate bitrate for each user with different preference.

Keywords

adaptive bitrate

QoE

video streaming

human brain

decision making

Contents

List of Figures

List of Tables

1 introduction

Most people nowadays carry mobile devices to access information on the Internet and use various services. Also, the amount of video traffic is increasing at a drastic pace. Cisco VNI [1] forecast that global mobile data traffic will grow seven-fold over five years from 2016 to 2021, and video traffic will account for 78% of the world's mobile data traffic by 2021. This increase in mobile traffic intensifies the degree of fluctuation in mobile traffic, and the range of fluctuation in the quality of service (QoS) level, which can be represented by the throughput, delay time, and packet loss rate, is thus increasing.Although a QoS guarantee is an objective of network service providers, it faces many challenges because there are various factors destabilizing the QoS, such as the inherent variability in signal strength, interference, noise, and user mobility [2] in addition to the increase in mobile traffic. These factors make it harder to guarantee the QoS of mobile devices.

From the viewpoint of over-the-top video service providers, the quality of experience (QoE) is attracting attention as an important factor when they provide video content. There are several reasons for this. One reason is the diversification of user context in the use of mobile devices; i.e., many types of devices, services, and communications. The QoE is a concept of subjectively perceived quality that was introduced in [3], and techniques that maximize user QoE are essential.

Today, most video streaming service providers, such as YouTube and Netflix, provide video content to users with adaptive bitrate control techniques according to the user and QoS context. DASH (Dynamic Adaptive Streaming over HTTP, also known as MPEG-DASH) [4] is one of the standards of HTTP Adaptive Streaming (HAS). Using DASH, the video player can dynamically switch among quality levels/representations, which means different bitrate levels, of the user's watching video while viewing in accordance with the QoS and the current quality of video. In DASH systems, an original video content is encoded into multiple encoded videos at different bitrates, and each encoded video is then partitioned into videos of a fixed length (generally a few seconds), which are called *chunks* or *segments* (where we use the term *segments*). Every finishing download of a segment, a client selects a next segment to download according to an adaptive bitrate (ABR) algorithm that is implemented generally in an application layer of the client.

Recent research has proposed various ABR algorithms for increasing the user QoE. General ABR algorithms estimate the instantaneous network quality and use it as a decision criterion. However, as mentioned above, network conditions can fluctuate over time and are unstable for mobile devices, and the accurate estimation of network conditions is therefore difficult. This results in degrading the user QoE because client applications (1) cannot fully utilize network resources through ABR algorithms, (2) frequently switch the bitrate in response to fluctuating decisions made by an ABR algorithm, and (3) request a higher bitrate than the network bandwidth, which leads to video rebuffering.

Many research focusing on improving video user QoE have been studied [2, 5], but most of them have not been sufficiently considered on the difference of user preference. Since the preference for video quality differs user by user, factors for improving the user QoE also differ user by user. For example, some users prefer higher video quality, some users place more emphasis on not stopping video playback, and some users prefer more stable video quality. Therefore, in considering improvement of the QoE of different users, a bitrate selection algorithm according to each user's preference type should be different.

In this thesis, we propose a bitrate control method that maximizes the QoE of individual users even in the environment where the QoS fluctuates. There are three problems to realize the method. The first problem is how to obtain a correct user preference model, the second problem is how to deal with the fluctuating QoS, and the third is how to choose the bitrate. On the first problem, there are some research aiming at estimating the user QoE and clarifying factors that affects QoE in video viewing. In order to obtain the *real* QoE model of video viewing users, their degrees of satisfaction have to be measured in a subjective manner. It is expected that such real QoE can be acquired by several methods, such as user's answers by using a good/bad buttons or estimation using user's Electroencephalogram. In this thesis we assume that the model of user preference on its QoE is given and under the assumption, we solve the second and third problems.

We propose a method to properly recognize observation information including QoS and select a bitrate suitable for user preference. Our proposed method recognizes the condition of the network and application in the client device by using a human cognitive model, the Bayesian attractor model (BAM [6]), which models cognition and decision making of the human brain, as the name suggests, according to the Bayesian inference. Based on the cognitive result and user preference, our method selects a video bitrate during video reproduction.

In our method, the BAM is implemented in the client MPEG-DASH video streaming application, and it perceives information available in the application layer and recognizes the network and application conditions of the client. Our method selects a video bitrate according to the BAM's cognitive result. Then we prepare a bitrate selection algorithm suitable for each user preference. In this thesis, we use a QoE model where the user QoE is calculated by “average bitrate,” “average bitrate variations, and “rebuffering time.” User preferences to the video quality can be represented by coefficients in the model. We propose bit rate selection algorithms according to some user preference types, and by providing a suitable algorithm for individual users, our method improves the QoE of the individual users.

The remainder of the paper is organized as follows. Section 2 provides existing research on QoE metrics, HAS techniques, and ABR algorithms. This section also gives a detailed description of the BAM. Section 3 explains how to apply the BAM to an ABR. We present our proposed method and evaluate the performance of it with computer simulation in this section. Section 4 describes a implementation of our proposed method into an MPEG-DASH streaming application service, and evaluates its performance in a real video player. Finally in Section 5, we offer concluding remarks and refer to future challenges.

2 Related work

2.1 Video QoE

The QoE is a measure of the degree of user satisfaction with a service. Past studies on the QoE of a video streaming service show that the QoE is strongly correlated with video player events (e.g., rebuffering, a change in video quality, and start-up delay). Some papers describe that the QoE relies on the start-up delay (e.g., [7, 8]) while other papers show that the QoE relies on rebuffering [7–9], the played bitrate [10, 11], and the bitrate change ratio [9, 10].

There are also studies that estimate the user QoE using important factors of the QoE. Reference [11], for example, presents a user experience model that can quantitatively measure the QoE of the ABR video streaming service and designs the model with three factors of the QoE, the initial (start-up) delay, stalling (rebuffering), and variation of video quality. As a wide survey of the QoE for video streaming in real society, the authors of [9] developed a browser plug-in for YouTube, named YouSlow, and collected and analyzed information on video player events and the user's video abandonment. The results of YouSlow analysis show that the bitrate changes ratio (average amplitude of bitrate changes over playback time) and rebuffering ratio (average rebuffering time over playback time) are correlated to the user's video abandonment. Regarding the bitrate change ratio, it is reported that even when the bitrate was improved, a high bitrate change ratio led to the user abandoning the video. Although the reasons are not clarified in [9], this may be because users prefer the stability of the bitrate to higher video quality.

2.2 MPEG-DASH

HAS is widely used for video streaming services. For instance, it is implemented in Microsoft Silverlight Smooth Streaming (MSS) by Microsoft, HTTP Live Streaming (HLS) by Apple, and Adobe HTTP Dynamic Streaming (HDS) by Adobe Systems. As a standard for HAS, DASH [4] was issued by MPEG in 2012 (MPEG-DASH). DASH aims to provide a smooth video streaming service to users corresponding to network conditions and types of client device. An overview of the MPEG-DASH system is shown in Fig. 1.

In a DASH system, video content is encoded into multiple versions at different bitrates, and each encoded video is then partitioned into videos of fixed length $segments$. Segments are stored on the DASH server. When a DASH streaming session starts, the DASH server provides the Media

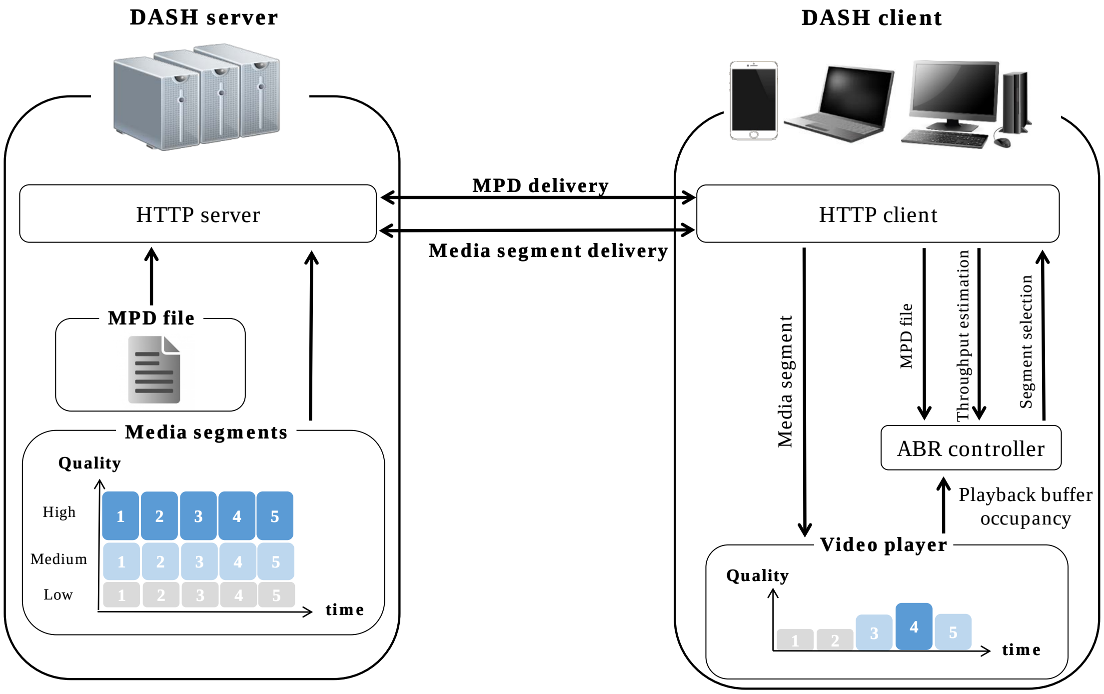


Figure 1: MPEG-DASH system

Presentation Description (MPD) to the DASH client. The MPD is an index file that describes media metadata of the different audio and video bitrates available to the client. To play video content, the client first obtains the MPD and then requests segments in the desired bitrate according to MPD information, network conditions, and types of the client device. The MPD and segments are delivered using HTTP. Because the client sends HTTP requests for each segment, the video player can switch to video with different bitrates for each segment. In this way, ABR streaming is realized in DASH.

2.3 ABR algorithms

Various ABR algorithms have been proposed and they can be broadly classified into three categories according to the feedback information they use [12]: *throughput-based* [13, 14], *buffer-based* [15, 16], and *hybrid/control theory- based* [17, 18]. Because ABR algorithms work in the application layer of the client device, they generally decide the appropriate video bitrate for the next segment to be downloaded, according to information available to the application layer of the client (e.g., playback buffer occupancy, and TCP throughput estimated by the application layer).

Here, it is difficult to estimate accurate network conditions because network conditions can fluctuate over time and vary across environments. Inaccurate estimation can lead to inappropriate bitrate selections, resulting in lower video quality or frequent bitrate switching or rebuffering.

Each time the client sends an HTTP request, it has to select an appropriate video bitrate according to information available to it. This selection of bitrates is made by an ABR algorithm implemented in the client device. The general goals of the ABR algorithm are as follows [19].

1. Avoid playback interruptions due to buffer underruns (rebuffering).
2. Maximize the video quality.
3. Minimize the number of video quality shifts.
4. Minimize the time between the request for a new video by the user and starting to play the video.

However, there are trade-off relationships among these goals as the authors of [19] mentioned. For instance, it is always possible to minimize the number of interruptions by selecting the lowest video bitrate to achieve goal 1, but goal 2 then cannot be achieved. To achieve goal 2, the ABR algorithm can switch video bitrate by reacting to the smallest changes in the network bandwidth. This causes frequent video quality shifts, and goal 3 cannot be achieved. Goal 4 is also a trade-off with goal 2 because selecting the lowest video bitrate at the start minimizes the start-up time but degrades the video quality. It is therefore necessary for the ABR algorithm to maximize a multi-objective function for these multiple goals. However, factors for maximizing the user QoE differ among people. It is significant to provide appropriate ABR algorithms for person by person.

2.4 Bayesian attractor model

This section explains the Bayesian attractor model (BAM) proposed in [6] and our extension of the BAM. The BAM models a human's brain, which accumulates sensing information of the external field and makes a decision using the Bayesian inference framework.

The BAM has a decision state $\mathbf{z}$ as its internal state and updates $\mathbf{z}$ according to an internal generative model that has stable fixed points (*attractors*). Note that the authors of [6] used winner-takes-all dynamics for the generative model of the BAM. Internally, the BAM has several decision alternatives, and each alternative i corresponds to each attractor ϕ_i. Since $\mathbf{z}$ is a hidden variable, in the cognitive process model, the BAM estimates the posterior density function of $\mathbf{z}$ by using the Bayesian inference. In the decision-making process model, the BAM checks whether a probability density when $\mathbf{z} = \phi_i$ exceeds a threshold value.

The cognitive process model discriminates attractors by comparing the perceived information with past experience and memory. Past experience and memory are linked to K attractors. For more detail, the state vector of ϕ_i ($i = 1 \cdots K$), is associated with past experience and memory by a feature vector μ_i. As mentioned above, the generative model of the BAM uses a nonlinear dynamics with these K attractors ($\phi_1 \cdots \phi_K$). In the BAM, decision state $\mathbf{z}$ is updated by the following equation.

$$\mathbf{z}_t = \mathbf{z}_{t-\Delta} + \Delta g(\mathbf{z}_{t-\Delta}) + \sqrt{\Delta}\mathbf{w}_t, \tag{1}$$

where $\mathbf{z}$ is updated from one time step to the next and $g(*)$ denotes the attractor dynamics [20], Δ means the update interval of the dynamics, $\mathbf{w}_t$ is a white noise following the normal distribution $\mathcal{N}(0, \mathbf{Q})$, where $\mathbf{Q} = (q^2/\Delta)\cdot\mathbf{I}$ is the variance–covariance matrix of the noise, and q is a parameter representing dynamics uncertainty. If there is no noise in the dynamics (namely, $q = 0$), $\mathbf{z}$ is drawn into one of the fixed points ϕ_i by repeating the update. The dynamics uncertainty represents the amount of noise with which the decision maker expects the state variable to be changed, which is interpreted as the tendency for state variables to switch between fixed points.

In the BAM, it is assumed that an observation, denoted by a vector $\mathbf{x}_t$, are generated corresponding to one of the attractors, which is represented by Eq. (2).

$$\mathbf{x_t} = \mathbf{M} \cdot \sigma(\mathbf{z_t}) + \mathbf{v_t}, \tag{2}$$

where $\mathbf{M}$ is a feature matrix of $[\mu_1, \mu_2, ..., \mu_K]$, and a feature vector μ_i links ϕ_i and memory. $\sigma(*)$ is a sigmoid function that maps all values $z_j \in \mathbf{z}$ to values between 0 and 1. Owing to the winner-takes-all dynamics of $\mathbf{z}$, the fixed point ϕ_i is mapped to a vector $\sigma(\phi_i)$, where one element is approximately 1 and the other elements are approximately zero. The linear combination $\mathbf{M}\cdot\sigma(\phi_i)$ thus becomes almost μ_i. Note that μ_i is a feature vector of the same dimension as an observation values $\mathbf{x}$. v_t is a white noise following the normal distribution $\mathcal{N}(0, \mathbf{R})$, where $\mathbf{R} = r^2 \cdot \mathbf{I}$ is the variance–covariance matrix of the noise and r is a parameter representing sensory uncertainty. The sensory uncertainty represents the amount of noise in observations that the decision maker expects.

The BAM estimates the posterior density function of $\mathbf{z}$ from input sequences of $\mathbf{x}_t$. In the decision-making process model, the estimation of the decision state $\mathbf{z}$ according to the observation value $\mathbf{x}$ involves estimating $\mathbf{z}_t$ that gives the minimum variance of $\mathbf{x}_t$ in the Eq. (2). In [6], the unscented Kalman filter (UKF), one of a Bayesian filters, is used for this estimation. Although the UKF is developed for estimating a nonlinear generative model, due to the generative model of the BAM with strong nonlinearity such that a sigmoid function is included, it loses the accuracy of the estimation. Another algorithm that can handle a nonlinear/non-Gaussian system and can estimate the state with higher precision is therefore desirable. In this paper, the particle filter (PF) is adopted as an algorithm satisfying this condition.

Unlike the UKF, the PF supports a non-Gaussian state space model, such that a more accurate estimation can be expected in the BAM's internal model. Using the PF, the probability density function of $\mathbf{z}_t$ at time t, $P(\mathbf{z}_t|\mathbf{x}_t)$ is estimated and the probability density $P(\mathbf{z}_t = \phi_i|\mathbf{x}_t)$ for each attractor ϕ_i is referred to as *confidence*. In the decision-making process model, when the confidence for the attractor ϕ_i, $P(\mathbf{z}_t = \phi_i|\mathbf{x}_t)$, exceeds the threshold λ, the attractor ϕ_i is finally adopted as the result of estimation. Additionally, if such ϕ_i does not exist, we will not do anything. If this threshold value is higher, estimation is more accurate but its speed is lower, and vice versa.

3 Rate control method based on a human cognitive model

3.1 Overview

The goal of the proposed method is to maximize the QoE of individual users in consideration of network and application conditions that change dynamically and the user preference for video quality, by selecting appropriate bitrates of a video segments. For that purpose, it is important to properly process observation information of network and application that can be available to the client device, and to correctly recognize the current conditions of the client device. In this thesis we adopt the BAM to recognize them. Based on the cognitive result, our method selects a bitrate according to the user preference to video quality. An overview of our proposal is shown in Fig. 2.

As described in Sec. 1, it is assumed that the user preference to video quality can be represented by a QoE model with "average bitrate," "average bitrate variations," and "rebuffering time." In our method, bitrate selection algorithms suit for improving the QoE of different users considering their preferences are prepared in advance, and according to the given QoE model of a user, one of the bitrate selection algorithm is chosen.

In this section, we first explain how the BAM recognizes network and application conditions. Next, we give a detailed description of a bitrate selection algorithm corresponding to the BAM's cognitive result for each user preference. Finally, we perform computer simulation of bitrate selection with our method and evaluate it quantitatively from the viewpoint of QoE.

3.2 Cognition of network and application conditions

In our method, the BAM runs in the client application and observes the network communication quality and video quality in the application layer. According to the observation, the BAM estimates which feature vector is closest to the current observation among feature vectors designed in advance, and chooses the video bitrate of the next segment to be downloaded according to the estimation result.

3.2.1 Observation information

As the network communication quality and application conditions to be considered, we focus on the available bandwidth and the buffer occupancy. These are widely adopted metrics in ABR

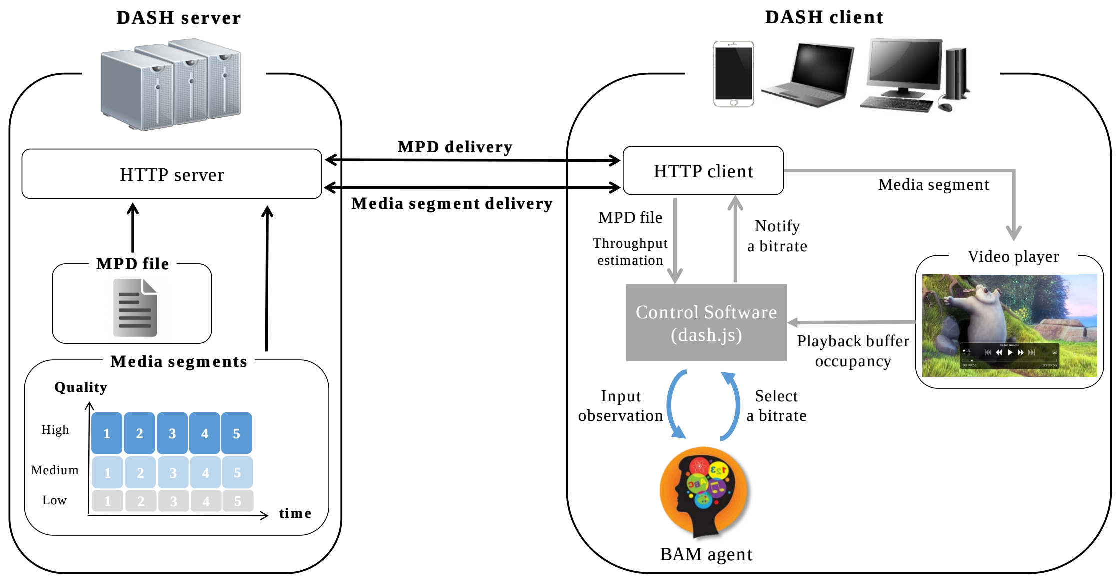


Figure 2: Overview of our proposed method

algorithms for DASH. An observation is performed every time the download of a segment is completed. dash.js [21] can acquire the playback buffer occupancy at the present moment. On the available bandwidth, our method measures instantaneous network throughput, as used in dash.js, with using a passive measurement method where an network throughput is calculated by dividing the segment size by the download time for it. We define the throughput as the estimated available bandwidth and use it as a part of input to BAM.

In our method, we prepare K sets of the playback buffer occupancy and the available bandwidth as feature vectors in advance, each of which equals μ_i. The observation information $\mathbf{x}_t$ input to the BAM at t is also a set of the available bandwidth and the buffer occupancy, and these pieces of information are acquired on the client device. From $\mathbf{x}_t$, the BAM estimates the current decision state $\mathbf{z}_t$. When $\mathbf{z}_t$ is identified as one of the pre-specified attractor, which is represented by ϕ_i, the BAM outputs μ_i as a result of decision making. our method selects an appropriate video bitrate according to the decision result.

Table 1: Example of BAM attractors and feature vectors

Attractor	Available bandwidth	Buffer occupancy
ϕ_1	T_3	B_{safe}
ϕ_2	T_2	B_{safe}
ϕ_3	T_1	B_{safe}
ϕ_4	T_3	$B_{transient}$
ϕ_5	T_2	$B_{transient}$
ϕ_6	T_1	$B_{transient}$
ϕ_7	T_3	B_{risky}
ϕ_8	T_2	B_{risky}
ϕ_9	T_1	B_{risky}

3.2.2 Attractor and feature vector design

In this section, we explain how to design the attractor and feature vector of the BAM. The attractor design means to decide how many attractors are prepared, namely to decide the value of K. Since K is the number of network and application conditions we want to discriminate, we determine feature vectors. On the available bandwidth, we want to know whether it can accommodate bitrates that a client application can choose from a MPD file. Then, the number of the network communication quality condition is set to that of available encoded videos. On the buffer occupancy, we want to know if the current buffer is abundant or depleted. Then, the buffer occupancy is classified into three types, safe, transient, and risky, and the value of the buffer occupancy is represented by B_{safe}, $B_{transient}$, and B_{risky}, respectively. Thus, the number of the application conditions is three. Finally, K is calculated by multiplying the number of the network conditions and the application conditions.

An example of the attractors and feature vectors when the number of selectable bitrate is three is shown in Table 1. In the table, T_1, T_2, and T_3 represent the available bandwidth (assuming $T_1 < T_2 < T_3$) corresponding to the bitrate of three encoded videos, respectively. In Table 1, $\mu_1 = (T_3,\ B_{safe})$, $\mu_2 = (T_2,\ B_{safe})$, ..., $\mu_9 = (T_1,\ B_{risky})$

3.3 Rate selection considering user preference

When $P(\mathbf{z}_t = \phi_i|\mathbf{x}_t)$ exceeds a threshold, the BAM refers μ_i as a current condition. Then, our method decides which bitrate of a next segment is to be selected and downloaded. Aiming at improving the user QoE in video streaming services, we consider the difference in user preferences for video quality. Although, as mentioned in Sec. 2.1, there are various factors that affect the user QoE, in this thesis, we focus on "average video bitrate," "bitrate variations," and "rebuffering time," which are taken up in many research. Thus, we use a QoE model consisting of these three factors as shown in Eq. (3).

$$QoE(\mu, \lambda) = \sum_{n=1}^{N} q(R_n) - \mu \sum_{n=1}^{N} T_n - \lambda \sum_{n=1}^{N} |q(R_{n+1}) - q(R_n)| \tag{3}$$

where λ and μ are non-negative weighting parameters for rebuffering time and bitrate variations, respectively. Here, we assume that occurrence of rebuffering greatly affects the user QoE compared to the other factors in video streaming services as pointed in [9]. In the QoE model, as a premise of avoiding rebuffering, user preference for these factors is classified into two types which are "prefer high image quality" and "prefer stable image quality." Our method provides a simple bitrate selection algorithm for each user preference type.

3.3.1 Bitrate selection algorithm for the preference type: "Prefer high image quality"

For users who prefer high image quality, bitrate selection algorithm tolerates the risk of occurrence of rebuffering and positively selects a higher bitrate. Algorithm 1 provides the pseudo-code of the algorithm. In case the buffer occupancy is abundant, a higher bitrate than the estimated available bandwidth is selected. Even if it is not abundant, unless it becomes near exhausted, this algorithm keeps a last bitrate or choose the highest bitrate that can be accommodated in the estimated available bandwidth.

Algorithm 1: Bitrate selection algorithm for the preference type: "prefer high image quality"

input : Estimated available bandwidth T_{est}, Estimated buffer occupancy B_{est}
Current bitrate $R_{current}$

output: Next bitrate R_{next}

$T^{-}_{est} \leftarrow T_{est-1}$
$T^{+}_{est} \leftarrow T_{est+1}$
if $B_{est} == B_{safe}$ **then**
  if $R_{current} < T^{+}_{est}$ **then**
    $R_{next} \leftarrow T^{+}_{est}$
  else
    $R_{next} \leftarrow R_{current}$
  end
else if $B_{est} == B_{transient}$ **then**
  if $R_{current} <= T_{est}$ **then**
    $R_{next} \leftarrow T_{est}$
  else
    $R_{next} \leftarrow R_{current}$
  end
else
  if $R_{current} < T^{-}_{est}$ **then**
    $R_{next} \leftarrow R_{current}$
  else
    $R_{next} \leftarrow R_{current-2}$
  end
end
return R_{next}

3.3.2 Bitrate algorithm for the preference type: "prefer stable image quality"

For users who prefer less average bitrate variations, a bitrate selection algorithm suppresses frequency of bitrate switching and magnitude of the bitrate changes. Algorithm 2 provides the pseudo-code of the algorithm. In order to suppress the bitrate variations, the algorithm basically keeps a last bitrate. Even when changing the bitrate, only one or two higher/lower bitrate than the current one is selected. Note that in case the buffer occupancy is abundant, the algorithm selects a bitrate higher than current one in order to avoid buffer overflow.

Algorithm 2: Bitrate algorithm for the preference type: "prefer stable image quality"

---

input : Estimated available bandwidth T_{est}, Estimated buffer occupancy B_{est}

Current bitrate $R_{current}$

output: Next bitrate R_{next}

```
T_est^- ← T_{est-1}
if B_est == B_safe then
    if R_current < T_est^- then
        R_next ← R_{current+1}
    else
        R_next ← R_current
    end
else if B_est == B_transient then
    if R_current <= T_est then
        R_next ← R_current
    else
        R_next ← R_{current-1}
    end
else
    if R_current < T_est^- then
        R_next ← R_current
    else
        R_next ← R_{current-2}
    end
end
return R_next
```

3.4 Simulation Evaluation

We evaluate our proposed method assuming a video streaming service with it in a situation where the available bandwidth changes dynamically. In following section, we explain a QoE model used in the evaluation, and evaluation results.

3.4.1 Simulation settings

Video parameters The 5-minute movie was encoded at five bitrates (0.5, 1.0, 1.5, 3.0, and 5.0 Mbps) and partitioned into 1-second segments.

Network environment For the network bandwidth to be observed in the simulation, referring to the benchmark provided by the DASH Industry Forum, we prepare two observation sequences of network bandwidth. In the first observation sequence (we call it "Network profile 1"), the average value of available bandwidth is changed every 30 s from the start time and the average value thereof is switched to 5.0, 4.0, 3.0, 2.0, 1.5, 2.0, 3.0, 4.0, and 5.0 Mbps in order from the start time. In the second observation sequences (we call it "Network profile 2"), the average value of available bandwidth is changed every 30 s from the start time and the average value thereof is switched to 9.0, 4.0, 2.0, 1.0, 2.0, 4.0, and 9.0 Mbps in order from the start time.

Additionally, we add a noise to each average value of available bandwidths. Each noise follows a normal distribution having an average of zero and standard deviation of $l_{noise}(\%)$ of each average value of the available bandwidth, where l_{noise} is defined as *noise level* hereafter. We change the value of the noise every second according to the distribution. For example, we use the normal distribution where the standard deviation is $2.0 \cdot l_{noise}/100$ for a 2.0 Mbps bandwidth. We set $l_{noise} = 10$ (we call it "noise level 1") or $l_{noise} = 30$ (we call it "noise level 2"). Then, note that the observation information of the network bandwidth actually input to the BAM is network throughput calculated by dividing a segment size by the download time for the segment as we explain in Sec.3.2.1.

BAM parameters The set of the buffer occupancy embedded in each attractor, B_{risky}, $B_{transient}$,and B_{safe}, is 10, 30, and 50 s, respectively, and a set of the available bandwidth embedded in each attractor $\mathbf{T}$ corresponds to the set of bitrates available to the client; i.e., $T_1 = 0.5$, $T_2 = 1.0$, $T_3 = 1.5$, $T_4 = 3.0$, and $T_5 = 5.0$ (Mbps). Therefore, the number of the BAM's attractor K is equal to 15. For parameters of the BAM, we set sensory uncertainty r to 0.5, dynamics uncertainty q to 0.5, and a threshold of confidence λ to 0.01.

Benchmark methods In this evaluation, we compare not only the performance of the bitrate selections for each user preference type, but we also compare the performance of them with a ABR algorithm which is proposed in existing research, BOLA-O [16] as benchmarks. BOLA is an algorithm used in dash.js [21] that is a client-side reference implementation of MPEG-DASH, and a method expected to be widely used. We compare the performance of our proposed method with that of BOLA-O as BOLA-O is one of practical ABR algorithms.

3.4.2 Metrics

In this evaluation, we investigate the performance of our method in terms of played video quality, and evaluate its performance under QoE metrics. For played video quality, we measure an average bitrate, average bitrate variations, and rebuffering time in overall video playback. The average bitrate is calculated by dividing the total size of all segments by the overall video playback time. The average bitrate variations is calculated by dividing the sum of the absolute values of difference in bitrate between itself and it's previous segment by the overall video playback time. For the QoE model, in order to evaluate from the viewpoint of difference in user preferences, two sets of weighting parameters of the QoE model shown in Eq. (3) are used, that is, $\lambda = 1$ and $\mu = 10$ for "prefer high image quality" type and $\lambda = 3$ and $\mu = 10$ for "prefer stable image quality" type.

3.4.3 Simulation results

Recognition result of BAM We first verify the recognition process of BAM described in Section 3 by computer simulation. Figs. 3–5 show the result of the BAM's recognition of the available bandwidth and the buffer occupancy, and the transition of *confidence* for all attractors (ϕ), where observation information of network bandwidth is generated according to "Network profile 2" with $l_{noise} = 10$ and the bitrate selection algorithm is for the "prefer high image qualty" type of users. In Fig. 5, the BAM adopts the most confident attractor among the attractors whose own confidence exceeds the threshold.

The recognition results of BAM in Figs. 3 and 4 correspond to the attractors (Fig. 5) whose confidence exceed the threshold. For example, when the confidence of ϕ_{13} exceeds the threshold at about time 60 s, a set of the available bandwidth and the buffer occupancy is recognized as the feature vector μ_{13} (in this situation, available bandwidth is estimated to 1.5 Mbps, buffer occupancy is estimated to 50 s) by the BAM.

Figure 6 presents that the bitrate selection of the proposed method is based on the estimated available bandwidth and the buffer occupancy which are shown in Figs. 3 and 4. We can see that the bitrate selection algorithm realizes stable image quality even in an environment where the network bandwidth largely fluctuates.

These results confirm that the recognized values are not unstable or not affected by a fluctuation in observations. Meanwhile, the BAM's recognition tracks large changes in observation. It is confirmed that the state estimation of the BAM appropriately performs.

Played video quality Figure. 7 shows the average bitrate of proposed method and that of BOLA. In Fig. 7(a), our method for "prefer high image quality" is slightly inferior to BOLA-O at noise level 1, but slightly better at noise level 2 where the variance of network bandwidth is greater than that at noise level 1. In Fig. 7(b), our bitrate selection algorithm for "prefer high image quality" in both noise level 1 and noise level 2 realizes a high average bitrate. This is because our method for "prefer high image quality" adopts an algorithm that positively selects a higher bitrate according to the set of the buffer occupancy and the estimated available bandwidth, described in Sec. 3.3.1.

The result of the average variations of bitrate is shown in Fig. 8. For each network profile and noise level, our bitrate selection algorithm for "prefer stable image quality" achieves a greatly lower average variations of bitrate than BOLA-O as shown in Fig. 7(a) and 8(b). The average variations of bitrate in our algorithm for "prefer high image quality" is also much lower than that of BOLA-O. The reason why the average bitrate variations of the selection algorithm for "prefer stable image quality" is lower than those of others is that the method takes a policy to positively keep the current bitrate according to the set of buffer occupancy and estimated available bandwidth, which is described in Sec. 3.3.2. In addition to the characteristics of the bitrate selection algorithm, less fluctuated recognition of the BAM makes it possible to realize the performance intended by the algorithm with a high accuracy. Note that the bitrate variations of BOLA was reported in the paper that proposed BOLA [16]. This paper proposed BOLA-O as an improved algorithm that overcomes this problem (Hereafter, the BOLA which does not mitigate the oscillations is called BOLA-U and it is distinguished from BOLA-O). Our simulation results, however, despite mimicking the evaluation environment of the Ref. [16], BOLA-O is not taken advantage of comparing with BOLA-U in our preliminary simulation.

The result of rebuffering time is shown in Fig. 9. While BOLA-O causes rebuffering in some situations, our bitrate selection algorithms for both “prefer high image quality” and “prefer stable image quality” do not lead rebuffering for each network profile and each noise level.

QoE evaluation We compare the performance of our method and BOLA-O in terms of the user QoE. The results of QoE values are normalized with dividing by the QoE of BOLA-O (therefore the QoE of BOLA-O is always 1). In Fig. 10, we compare our selection algorithm for “prefer high image quality” with BOLA-O in terms of QoE defined by Eq. (3) for the preference type “prefer high image quality” ($\lambda = 1$, $\mu = 10$).

The QoE of our bitrate selection algorithm for “prefer high image quality” is higher than BOLA-O. Although there is a less difference between our method and BOLA-O in the average bitrate at network profile 1, our algorithm for “prefer high image quality” is greatly superior to BOLA-O in terms of the average bitrate variation. Therefore, although the QoE model is for “prefer high image quality”, the QoE of our method is larger than BOLA-O.

The result of the QoE for the preference type “prefer stable image quality” ($\lambda = 3$, $\mu = 10$) is shown in Fig.11. Since the difference in bitrate variations between our selection algorithm for “prefer stable image quality” and BOLA-O is so large, which imposes a large penalty on this QoE model, the QoE of our bitrate selection algorithm for “prefer stable image quality” is higher than that of BOLA-O as the figure shows.

Thus, through computer simulation, we can conclude that the bitrate selection algorithm for suppression of the switching frequency of the bitrate can be realized under the condition where observation information greatly fluctuates. Our proposed method can improve the QoE for each user preference type by using an appropriate bitrate selection algorithm according to user preference. In the following section, we show our proposed method works as intended in a real video application.

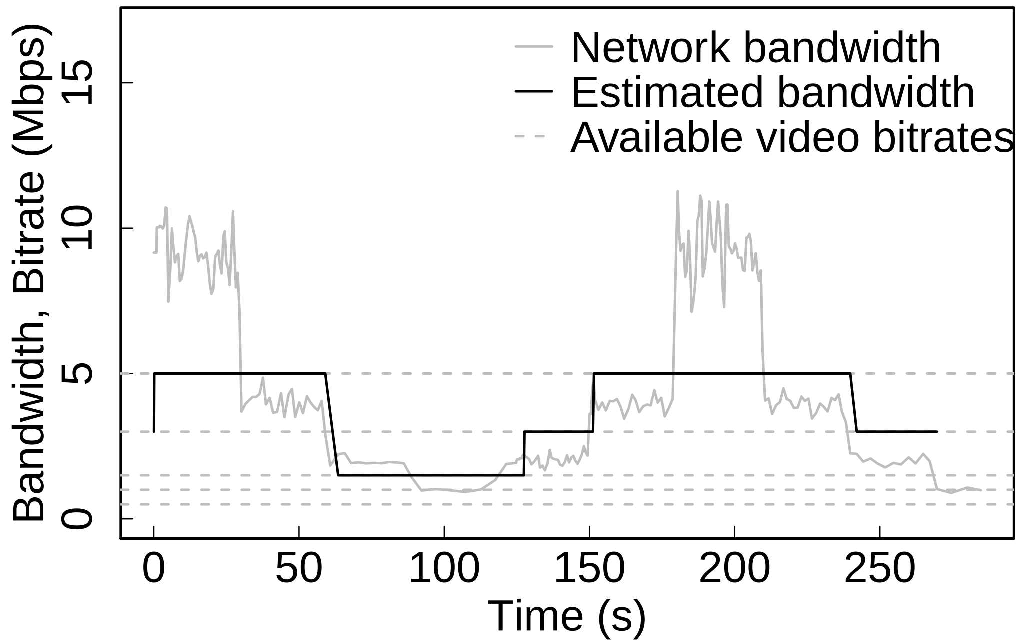


Figure 3: Estimation result of available bandwidth using the BAM

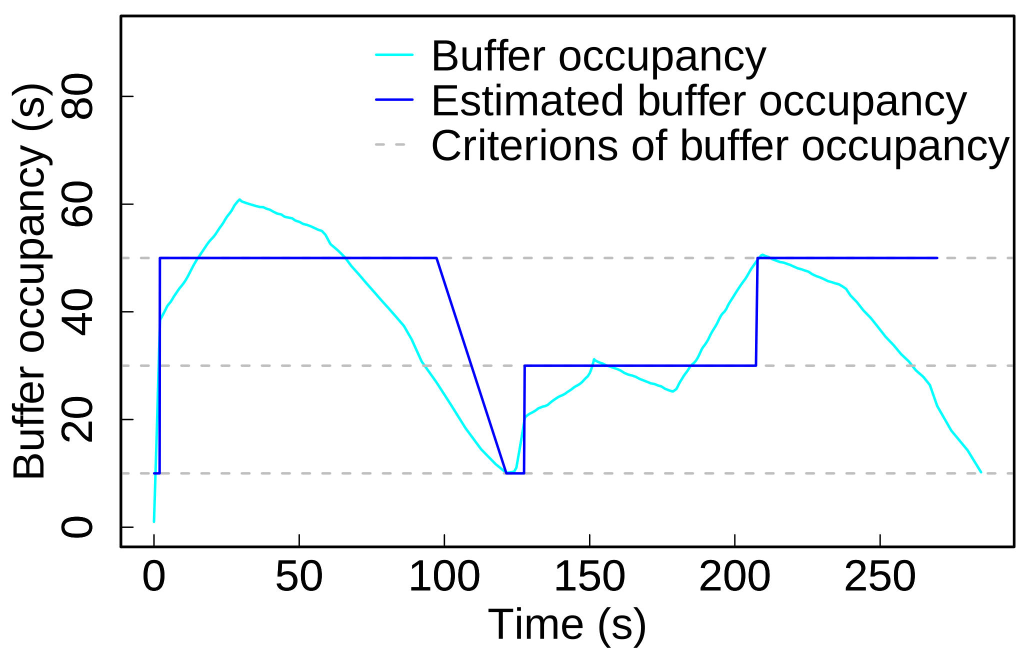


Figure 4: Estimation result of buffer occupancy using the BAM

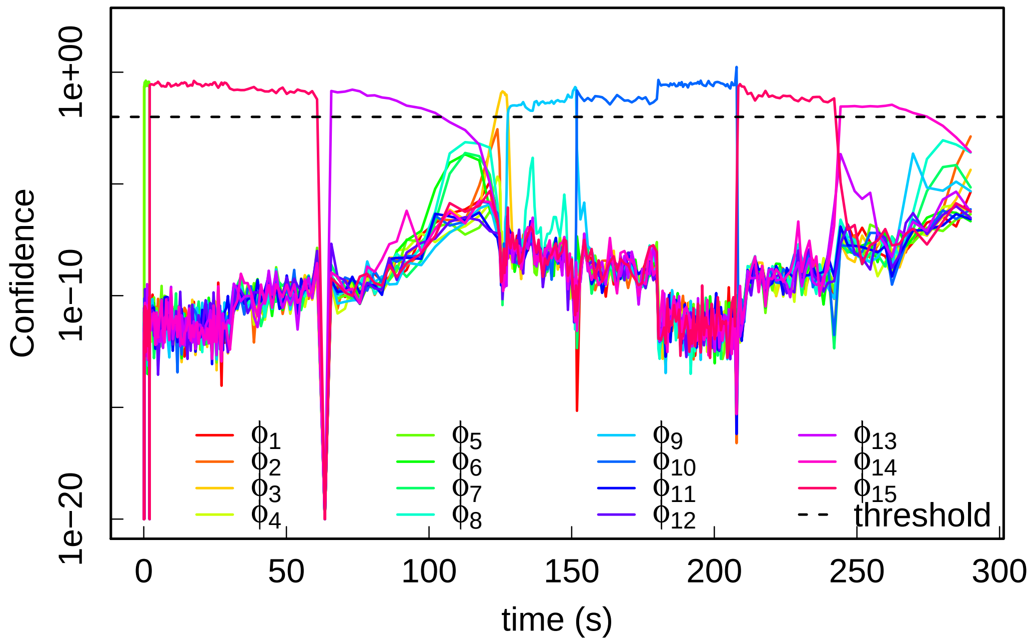


Figure 5: Transition of the BAM confidence

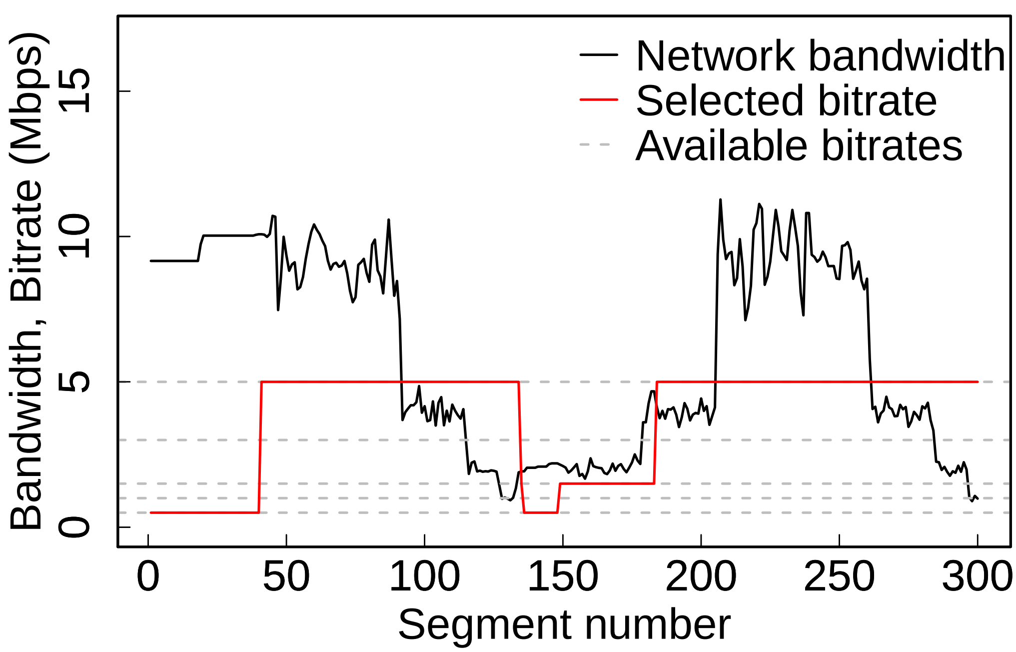
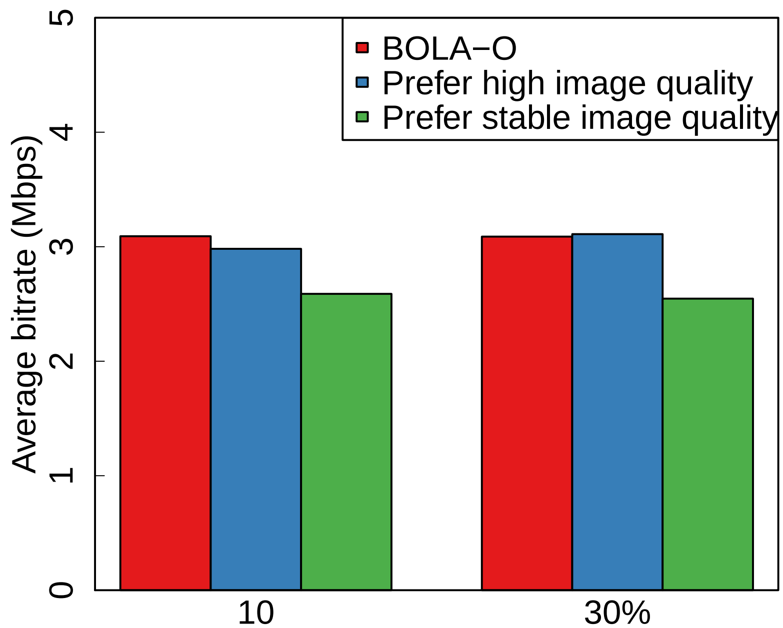
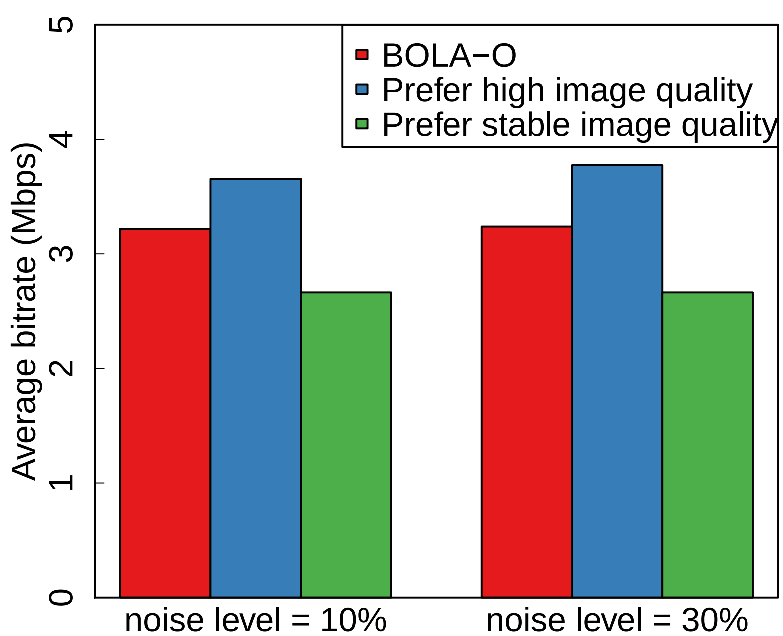


Figure 6: A example of bitrate selection using the BAM

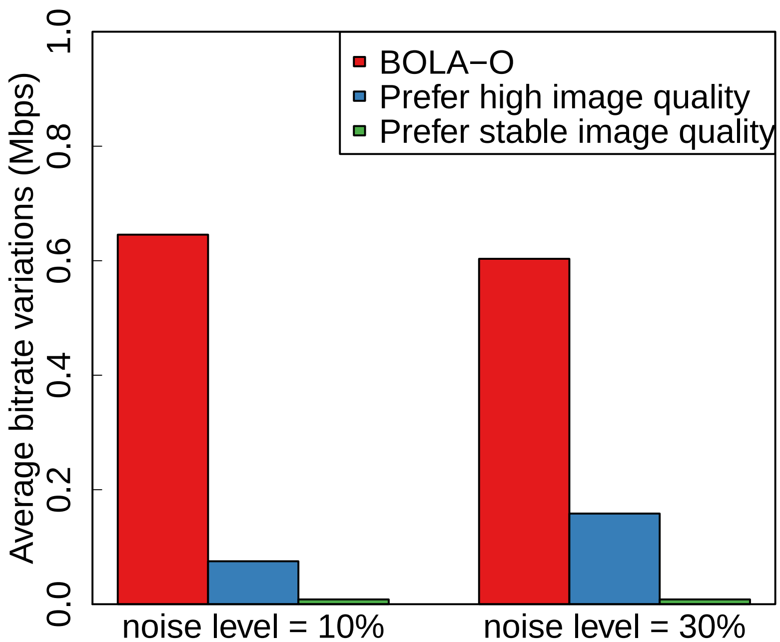


(a) Network profile 1

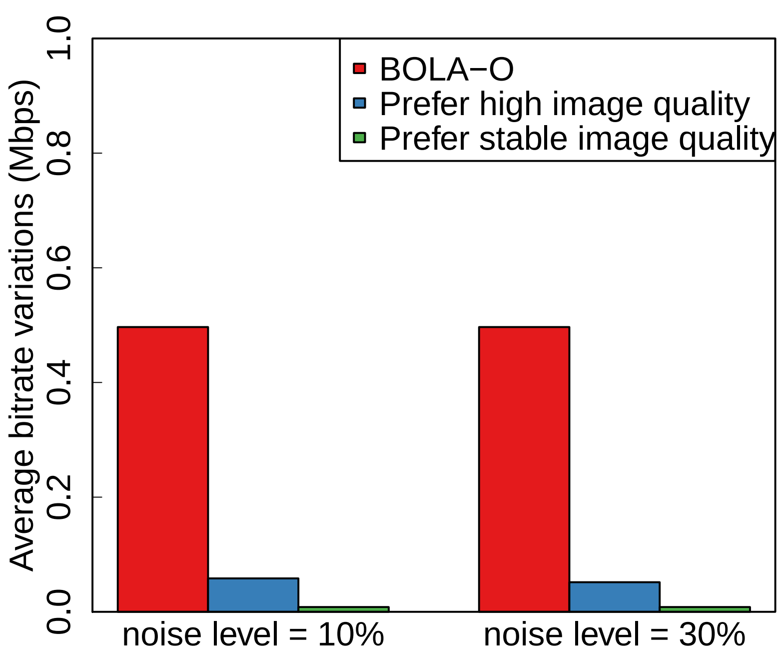


(b) Network profile 2

Figure 7: Simulation result of average bitrate

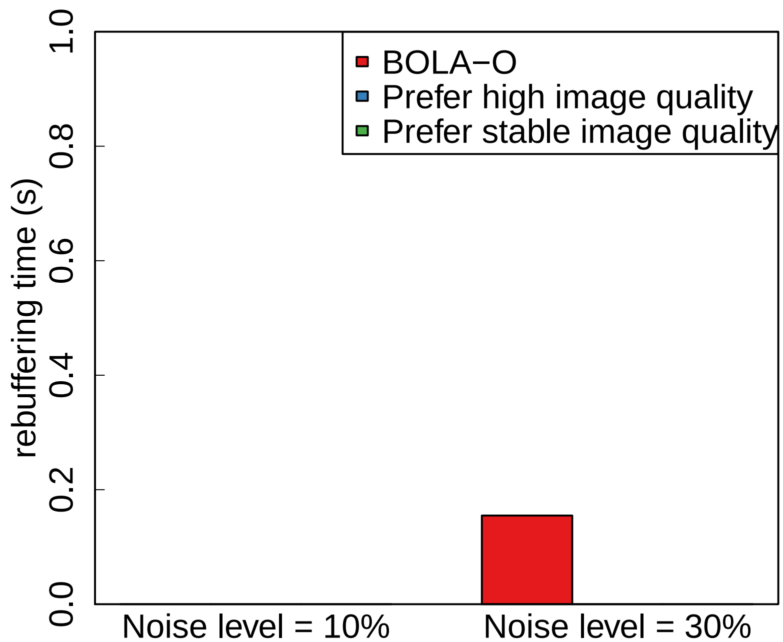


(a) Network profile 1

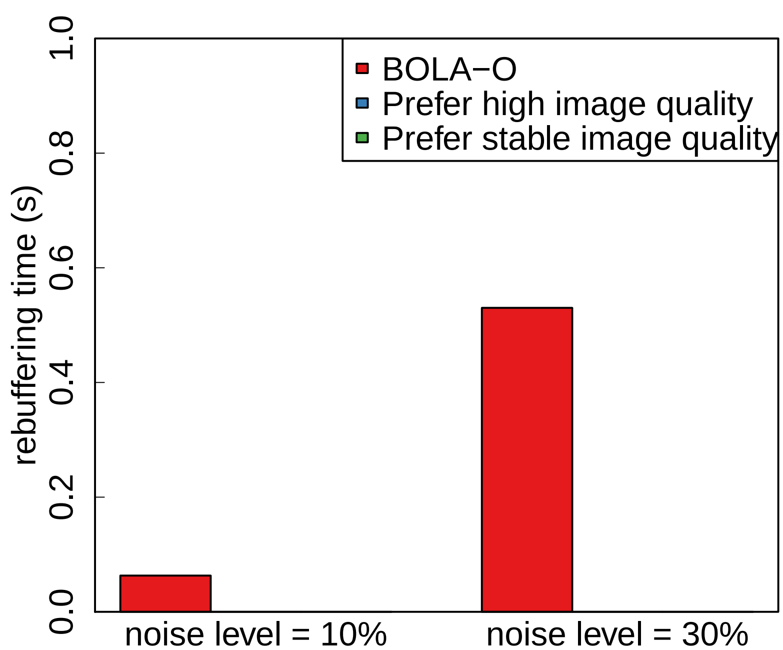


(b) Network profile 2

Figure 8: Simulation result of average bitrate variations

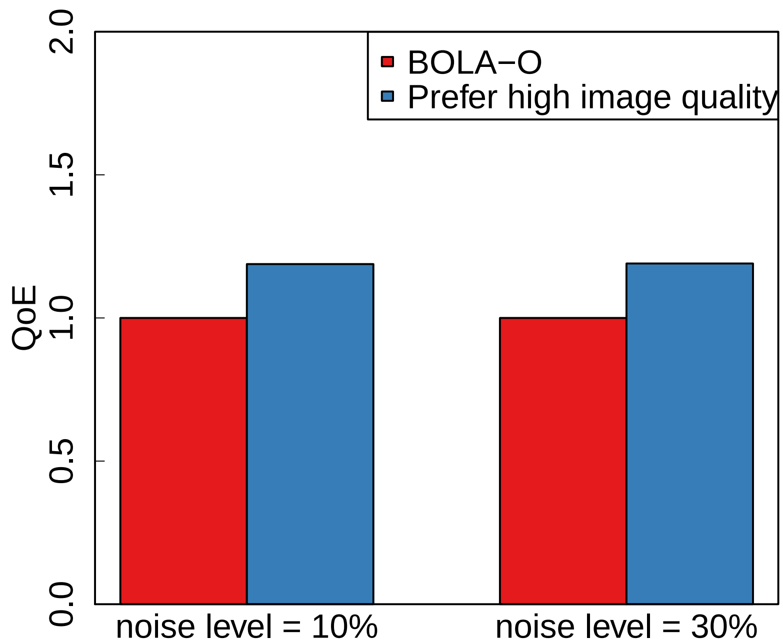


(a) Network profile 1

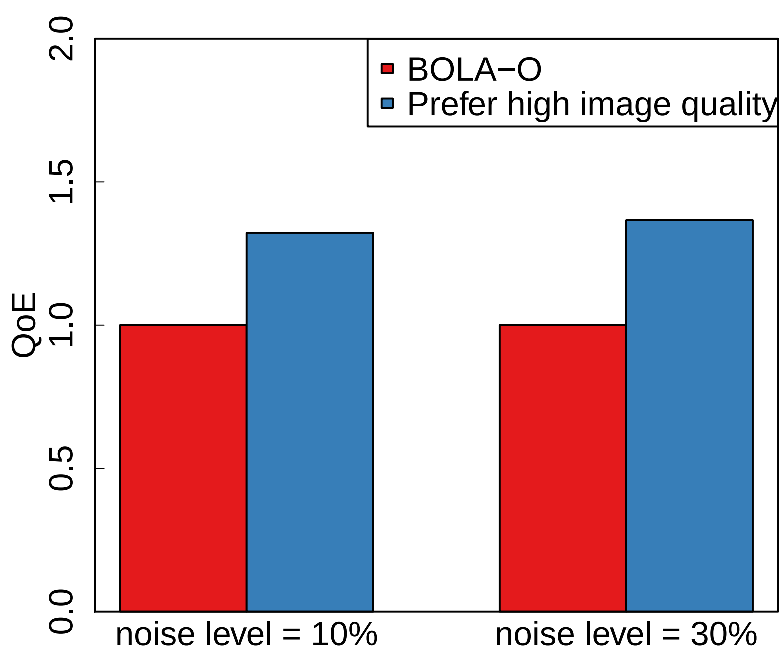


(b) Network profile 2

Figure 9: Simulation reslult of rebuffering time

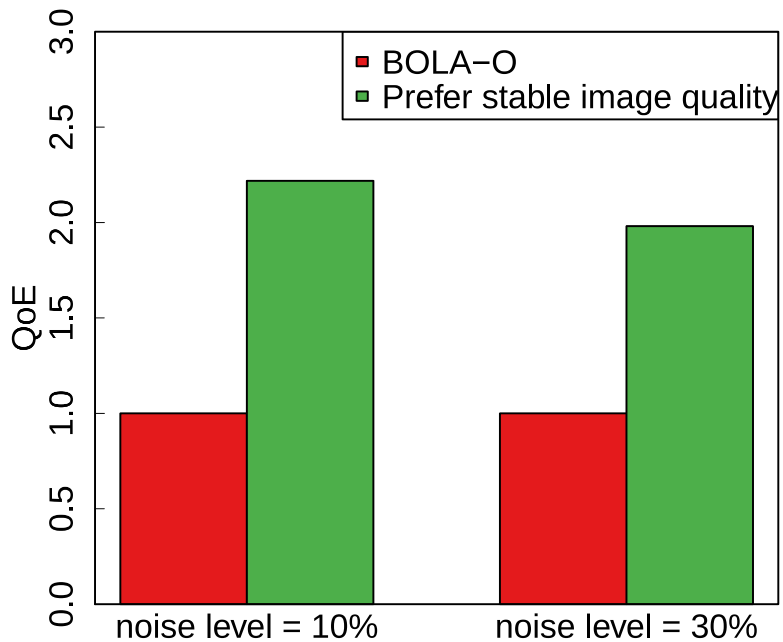


(a) Network profile 1

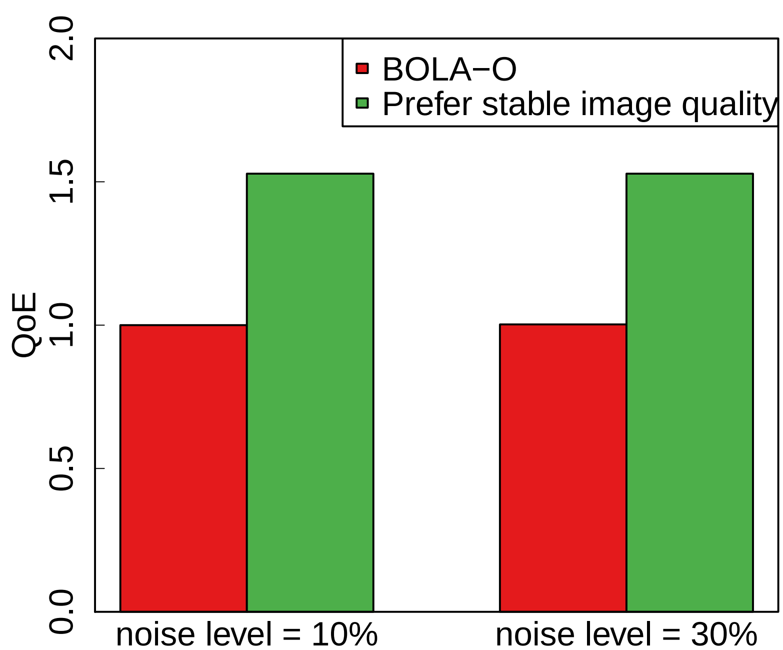


(b) Network profile 2

Figure 10: Result of QoE for “Prefer high image quality”



(a) Network profile 1



(b) Network profile 2

Figure 11: Result of QoE for “Prefer stable image quality”

4 Implementation of rate control method in video streaming application

In this thesis, we implement our method into an MPEG-DASH application and evaluate its performance. For the implementation, we build a video server that provides a video streaming service in MPEG-DASH framework, and a video client that plays a video through a video player on a web browser.

We implemented our proposed method in the client side, which receives observation information from the video player and notifies a video bitrate to the video player for the next segment to be downloaded. Observation information obtained in dash.js is sent to our method through a web-socket. Within our method, the BAM updates the internal state according to the received observation information, and recognizes the situation of the current client. Our method determines a bitrate of next segment with using the cognition result and a bitrate selection algorithm for each type of user preference described in Sec. 3.3, and sends it to dash.js.

4.1 Evaluation in a real video player

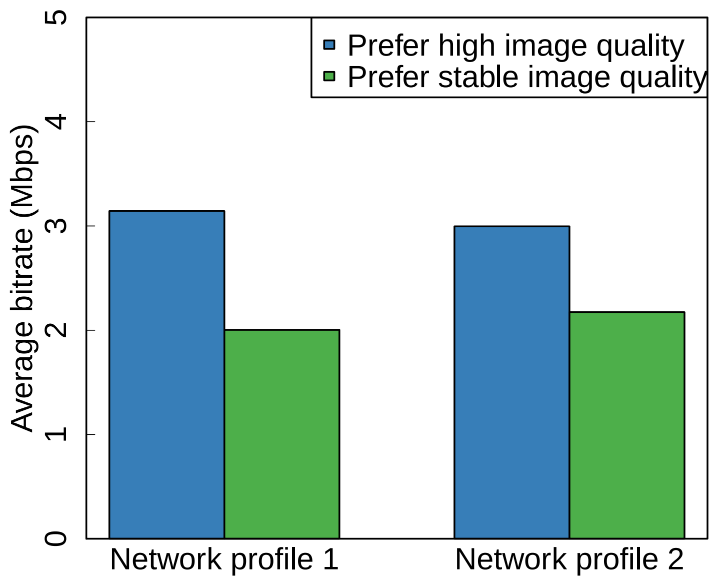
In this subsection, we verify the performance of the implemented our method and evaluate its performance.

First, we explain the settings of the video player. Our implemented system is composed of two computers which are a video server (Ubuntu 16.04 LTS) and a video client (Windows 10). They are connected with a 1000 Mbps direct Ethernet connection. In the client, the video player runs on a Google Chrome web browser for Windows (version 71.0.3578.98) with JavaScript engine (version 8). In the server, Apache HTTP server (version 2.4) is launched as a video streaming service provider. For network emulation, we use the Linux traffic control tool (tc tool) to throttle the available bandwidth of the link between two computers according to the Network profiles which is described in Sec. 3.4.1. Note that in the tc tool, we can only set an upper bound of the network bandwidth. Therefore, an average value of network bandwidth in the evaluation of a real video player are lower than that in the simulation evaluation.

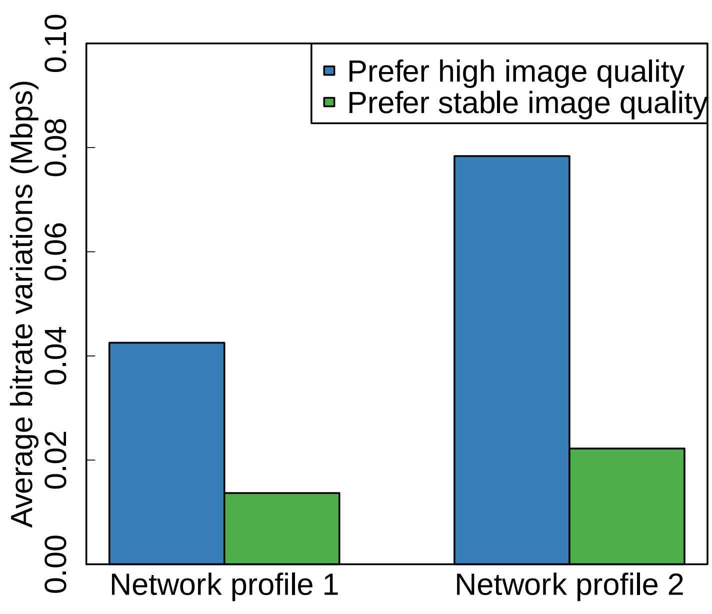
As for the evaluation metrics, in this evaluation, we investigate the performance of our method in terms of played video quality. For played video quality, we measure the average bitrate and average bitrate variations. These metrics are same in the simulation evaluation.

4.2 Evaluation Results

Experimental results of the average bitrate and the average bitrate variation are shown in Fig. 12(a) and Fig. 12(b). Since rebuffering did not occur in this evaluation, we do not show any figure about the performance of rebuffering time. Our bitrate selection algorithm for "prefer high image quality" realizes a higher average bitrate with larger bitrate variations than the algorithm for "prefer stable image quality" in each network profile. Conversely, the algorithm for "prefer stable image quality" suppresses bitrate variations although average bitrate is lower than our method for "prefer high image quality" in each network profile. These results confirm that our method can improve QoE of each user preference type by using an appropriate bitrate selection algorithm according to user preference type even in a real video player.



(a) Average bitrate



(b) Average bitrate variations

Figure 12: Played video quality in a evaluation of a real video player

5 Conclusion

In this paper, we proposed a rate control method that selects the appropriate video bitrate according to user preference, aiming at improving the QoE of each user by selecting bitrate according to the type of user preference to video quality. In order to select an appropriate bitrate according to the user preference type, it is essential to recognize information of user device and its network communication quality. In our proposed method, for the cognition of such information, we focused on the cognitive model of a human's brain, the Bayesian attractor model and we associate simple bitrate algorithms with the cognitive result according the user preference type, "prefer high image quality" and "prefer stable image quality."

In our computer simulation, we compare the performance of our proposed method with that of BOLA-O algorithm [16] adopted in dash.js [21] as a benchmark algorithm. Through the simulation, we demonstrated that our proposed method can perform appropriate bitrate control according the user preference type, that is, it can control a bitrate with less bitrate variations for the user preference type "prefer stable image quality" compared to the BOLA-O, and our method can control a bitrate with higher bitrate for the user preference type "prefer high image quality" compared to the BOLA-O even in the situation where network bandwidth greatly fluctuates. Simulation results showed that our proposed method for the user preference type "prefer high image quality" improved average bitrate by up to 16% and a QoE by 18%–36% for the user preference type "prefer high image quality" compared with BOLA-O. Also our proposed method for the user preference type "prefer stable image quality" reduced average bitrate variations by 98% and a QoE by 52%–121% compared with BOLA-O.

In addition to the simulation, we implemented our proposed method and evaluated it in a real video player. The evaluation result confirms that our proposed method realizes a bitrate control according to the user preference type also in a real video player.

Our future work includes to evaluate our proposed method in real mobile network environment and to implementation a cognitive revision mechanism by using a meta-cognitive algorithm that can adapt to situations where the model of environmental variation itself changes.

Acknowledgements

We cannot accomplish this thesis without a lot of great supports of many people. First, I would like to express my deepest gratitude to Professor Masayuki Murata of Osaka University, for his valuable comments, insights and continuous encouragement. I especially wish to express my gratitude to Assistant Professor Daichi Kominami of Osaka University. His appropriate guidance and firsthand advices have definitely been indispensable for my research. Furthermore, I would like to show my greatest appreciation to Specially Appointed Assistant Professor Tatsuya Otoshi of Osaka University. He devoted a great deal of time for me and gave me a lot of advices about my research. Furthermore, I am indebted to Associate Professor Shinichi Arakawa and Associate Professor Yuichi Ohshita. Their insightful comments and feedback developed my research. Finally, I would like to thank all the members of Advanced Network Architecture Research Laboratory at the Graduate School of Information Science and Technology, Osaka University, for their support and advices.

References

[1] Cisco, “Cisco visual networking index: Global mobile data traffic forecast update, 20162021,” tech. rep., Cisco Systems, Inc., April 2017.

[2] X. K. Zou, J. Erman, V. Gopalakrishnan, E. Halepovic, R. Jana, X. Jin, J. Rexford, and R. K. Sinha, “Can accurate predictions improve video streaming in cellular networks?,” in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications - HotMobile ’15*, pp. 57–62, February 2015.

[3] K. Brunnström, S. A. Beker, K. De Moor, A. Dooms, S. Egger, M.-N. Garcia, T. Hossfeld, S. Jumisko-Pyykkö, C. Keimel, M.-C. Larabi, B. Lawlor, P. Le Callet, S. Möller, F. Pereira, M. Pereira, A. Perkis, J. Pibernik, A. Pinheiro, A. Raake, P. Reichl, U. Reiter, R. Schatz, P. Schelkens, L. Skorin-Kapov, D. Strohmeier, C. Timmerer, M. Varela, I. Wechsung, J. You, and A. Zgank, “Qualinet white paper on definitions of quality of experience,” March 2013. Qualinet White Paper on Definitions of Quality of Experience.

[4] “ISO/IEC 23009-1:2014 – information technology – dynamic adaptive streaming over HTTP (DASH) – part 1: Media presentation description and segment formats.”

[5] C. Wang, A. Rizk, and M. Zink, “SQUAD: a spectrum-based quality adaptation for dynamic adaptive streaming over HTTP,” *Proceedings of the 7th International Conference on Multimedia Systems, MMSys 2016*, pp. 1 – 12, 2016.

[6] S. Bitzer, J. Bruineberg, and S. J. Kiebel, “A bayesian attractor model for perceptual decision making,” *PLoS computational biology*, vol. 11, p. e1004442, August 2015.

[7] R. K. Mok, E. W. Chan, X. Luo, and R. K. Chang, “Inferring the QoE of HTTP video streaming from user-viewing activities,” in *Proceedings of the first ACM SIGCOMM workshop on Measurements up the stack – W-MUST ’11*, p. 31, 2011.

[8] T. Pessemier, K. Moor, W.Joseph, L. Marez, and L. Martens, “Quantifying the influence of rebuffering interruptions on the user’s quality of experience during mobile video watching,” *IEEE Transactions on Broadcasting*, vol. 59, pp. 47–61, March 2013.

[9] H. Nam, K. Kim, and H. Schulzrinne, "QoE matters more than QoS: Why people stop watching cat videos," in *Proceedings of IEEE INFOCOM 2016 – The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, April 2016.

[10] M. Z. Shafiq, J. Erman, L. Ji, A. X. Liu, J. Pang, and J. Wang, "Understanding the impact of video quality on user engagement florin," *ACM SIGMETRICS Performance Evaluation Review*, vol. 42, pp. 367–379, July 2014.

[11] Y. Liu, S. Dey, F. Ulupinar, M. Luby, and Y. Mao, "Deriving and validating user experience model for DASH video streaming," *IEEE Transactions on Broadcasting*, vol. 61, pp. 651–665, August 2015.

[12] J. Kua, G. Armitage, and P. Branch, "A survey of rate adaptation techniques for dynamic adaptive streaming over HTTP," *IEEE Communications Surveys Tutorials*, vol. 19, pp. 1842–1866, March 2017.

[13] I. B. C. Liu and M. Gabbouj, "Rate adaptation for adaptive HTTP streaming," in *Proceedings of the second annual ACM Multimedia Systems conference*, pp. 169–174, January 2011.

[14] J. Jiang, V. Sekar, and H. Zhang, "Improving fairness, efficiency, and stability in HTTP-based adaptive video streaming with festive," *IEEE/ACM Transactions on Networking*, vol. 22, pp. 326–340, February 2014.

[15] T. Huang, R. Johari, N. McKeown, M. Trunnell, and M. Watson, "A buffer-based approach to rate adaptation: Evidence from a large video streaming service," *ACM SIGCOMM Computer Communication Review*, August 2014.

[16] K. Spiteri, R. Urgaonkar, and R. K. Sitaraman, "BOLA: Near-optimal bitrate adaptation for online videos," in *Proceedings of IEEE INFOCOM*, July 2016.

[17] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. Begen, and D. Oran, "Probe and adapt: Rate adaptation for HTTP video streaming at scale," *IEEE Journal on Selected Areas in Communications*, vol. 32, May 2013.

[18] A. Mansy, B. Versteeg, and M. Ammar, "SABRE: A client based technique for mitigating the buffer bloat effect of adaptive video flows," in *Proceedings of the 4th ACM Multimedia Systems Conference, MMSys 2013*, pp. 214–225, February 2013.

[19] S. Varma, "Chapter 6 – flow control for video applications," in *Internet Congestion Control* (S. Varma, ed.), pp. 173–203, Morgan Kaufmann, August 2015.

[20] C. Zhang, G. Dangelmayr, and I. Oprea, "Storing cycles in hopfield-type networks with pseudoinverse learning rule: Admissibility and network topology," *Neural networks: the official journal of the International Neural Network Society*, vol. 46C, pp. 283–298, June 2013.

[21] DASH Industry Forum, "Dash-Industry-Forum/dash.js." https://github.com/Dash-Industry-Forum/dash.js. accessed 2018-10-25.