

# YoMoApp: a Tool for Analyzing QoE of YouTube HTTP Adaptive Streaming in Mobile Networks

Florian Wamser\*, Michael Seufert\*, Pedro Casas<sup>†</sup>, Ralf Irmer<sup>‡</sup>, Phuoc Tran-Gia\*, Raimund Schatz<sup>†</sup>

\*University of Würzburg

{florian.wamser | seufert | trangia}  
@informatik.uni-wuerzburg.de

<sup>†</sup>FTW

Telecommunication Research Center Vienna  
{casas | schatz}@ftw.at

<sup>‡</sup>Vodafone Group

Research and Development  
ralf.irmer@vodafone.com

**Abstract**—The performance of YouTube in mobile networks is crucial to network operators, who try to find a trade-off between cost-efficient handling of the huge traffic amounts and high perceived end-user Quality of Experience (QoE). This paper introduces YoMoApp (YouTube Performance Monitoring Application), an Android application, which passively monitors key performance indicators (KPIs) of YouTube adaptive video streaming on end-user smartphones. The monitored KPIs (i.e., player state/events, buffer, and video quality level) can be used to analyze the QoE of mobile YouTube video sessions. YoMoApp is a valuable tool to assess the performance of mobile networks with respect to YouTube traffic, as well as to develop optimizations and QoE models for mobile HTTP adaptive streaming. We test YoMoApp through real subjective QoE tests showing that the tool is accurate to capture the experience of end-users watching YouTube on smartphones.

## I. INTRODUCTION

YouTube is one of the most popular services in today's Internet. It has more than 1 billion users and every day people watch hundreds of millions of hours of YouTube videos. Half of those YouTube views are on mobile devices [1]. On the one hand, mobile operators want to handle the huge amount of video traffic as efficiently as possible (high revenue per bit), on the other hand, they want to deliver a high Quality of Experience (QoE) to satisfy their customers. Therefore, it is very important for mobile operators to understand the performance of their networks with respect to YouTube traffic.

In order to measure the network performance in terms of QoE, different concepts are proposed in literature. First, operators can conduct subjective studies and ask the users about the perceived service quality. Subjective studies can directly assess the QoE in terms of mean opinion scores (MOS), but their design and execution is complex and costly. Second, operators can perform active measurements with client devices to probe the network. However, these samples can only provide an estimation of the network performance and cannot cover all contingencies. Finally, passive measurements can be conducted either in the network or at the client device. In-network measurements (e.g., YOUQMON [2]) have a more global scope and cover more users, but the resulting QoE has to be estimated from traffic characteristics and/or deep-packet

inspection. Moreover, the recent trend towards HTTPS (e.g., YouTube, Vimeo) is about to inhibit its applicability. Client-side measurements (e.g., YoMo [3], [4]), on the other hand, are more extensive as end users are directly involved, but they can provide an individual and accurate view on objective key performance indicators (KPIs). Those KPIs provide more elaborate information about the perceived service quality and can be mapped to QoE by means of QoE models [5].

In this paper, we introduce YoMoApp (YouTube Performance Monitoring Application), a measurement application for client-side measurement of YouTube video streaming on mobile Android devices. The application uses the YouTube mobile website and the YouTube HTML5 API to exactly replicate the well-known YouTube service, which employs HTTP adaptive streaming (HAS) technology based on resolution adaptation. However, it additionally monitors and stores multiple KPIs of the video streaming via the YouTube API (i.e., player state/events, buffer, and video quality level), which allow to analyze the QoE of adaptive video streaming sessions.

YoMoApp is a highly valuable application for passive client-side measurements of the YouTube performance in mobile networks, an approach, which is becoming highly popular among cellular operators. In addition, the tool can be used for developing new QoE models for HAS in mobile devices, enabling multiple QoE-based monitoring applications for YouTube traffic, such as dynamic traffic engineering [6], troubleshooting, load balancing and caching, and many more.

This paper describes the measurement concept and the implementation of the application in detail. Moreover, a subjective study was conducted in which YoMoApp was used to monitor the performance of YouTube video streaming sessions under different network conditions. The measured KPIs for HTTP adaptive video streaming and the quality ratings of the participants are investigated in order to validate the measurement application through subjective QoE tests.

The remainder of the paper is structured as follows. First, related papers on QoE for HTTP video streaming in mobile devices are summarized in Section II. Second, the design and implementation of YoMoApp is described in detail in Section III. The subjective lab study conducted with YoMoApp and the analysis of the measured KPIs and the QoE feedback of the participants are presented in Section IV. Finally, Section V concludes the paper by discussing future applications of YoMoApp.

This work was partly funded in the framework of the EU ICT projects SmartenIT (FP7-2012-ICT-317846), mPlane (FP7-2012-ICT-318627), and IN-PUT (H2020-2014-ICT-644672), and was partly performed within the project ACE 3.0 at the Telecommunications Research Center Vienna (FTW).

## II. RELATED WORK

Due to its paramount relevance, QoE in HTTP video streaming is a well-known and largely investigated topic. Previous papers [7], [8] have shown that stallings (i.e., stops of the video playback) and initial delays on the video playback are the most relevant KPIs for QoE in HTTP video streaming. While authors in [9] have shown that initial delays are generally tolerated by most users, stallings have a huge impact on user experience; indeed, a small number of stalling events severely degrades the QoE [7].

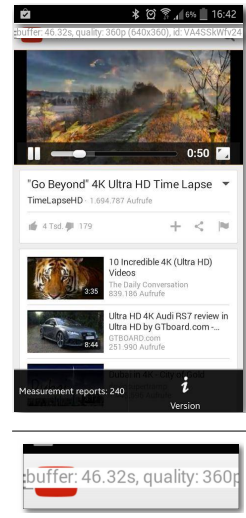
In the case of adaptive streaming, a new KPI becomes relevant in terms of QoE: quality switches. While authors in [10] found that quality adaptation could dramatically reduce stalling when bandwidth decreases in a mobile environment, authors in [11] have shown that quality switches have an important impact on QoE, as they increase or decrease the video quality during the playback. Authors in [12] found that it is sufficient to consider the time spent on each quality layer to properly estimate the QoE of a video session, and not the number of quality switches as one might expect. A more comprehensive survey of the QoE of adaptive streaming can be found in [5].

When it comes to our specific analysis of YouTube QoE in mobile networks and mobile devices, references become scarcer, showing that there is still an important gap to fill. In [13], authors study the characteristics of YouTube traffic on smartphones connected to a cellular network, showing that these devices have a non-negligible impact on the characteristics of the downloaded traffic. Closer to the subject of this paper, authors in [14] describe a subjective QoE evaluation framework for mobile Android devices in a lab environment. Authors conduct a basic QoE-based study on the non-adaptive YouTube streaming using very low bit rate videos and without considering the impact of the download throughput. In [15], authors study the QoE of YouTube in mobile devices through a field trial, exclusively considering the non-adaptive version of the YouTube player. In [2], we introduced an on-line measurement system to monitor the QoE of YouTube in cellular networks relying exclusively on network-layer measurements. Similarly, in [16], authors introduce Prometheus, an approach to estimate QoE of mobile apps, using both passive in-network measurements and in-device measurements, applying machine learning techniques to obtain mappings between QoS and QoE.

## III. YOMOAPP - MEASUREMENT CONCEPT AND DESCRIPTION OF THE MONITORING APPLICATION

The goal is to provide a methodology for monitoring application-layer KPIs of YouTube that have a high correlation with the actual QoE of mobile app users. According to [7], [12], the main influence parameters of the YouTube QoE are *stallings* and *video quality*. In order to obtain these parameters, we monitor the buffer filling levels and the resolution of the YouTube videos.

The approach is as follows. The original YouTube app is fully replicated in functionality and design, see Fig. 1. To this end, existing libraries from YouTube are used that are available for YouTube developers. An Android web view browser element was embedded for the YouTube video playback, such that HTML5 video playback is possible including



Name	Description
<i>buffered</i>	List of time ranges of the media content that have been buffered
<i>height/width</i>	Height and width of the video's display area in CSS pixels
<i>played</i>	Object indicating all the ranges of the video that have been played
<i>currentTime</i>	Current video playtime
<i>youtubeld</i>	Object indicating YouTube identifier of the video content
<i>totalVideo-Frames</i>	Total number of frames that would have been displayed if no frames are dropped
<i>dropped-Video-Frames</i>	Total number of frames dropped predecode or dropped because the frame missed its display deadline
<i>corrupted-Video-Frames</i>	Total number of corrupted frames that have been detected
<i>Timestamp</i>	Timestamp of the data query
<i>Name</i>	A pre-defined device name
<i>Session</i>	A session timestamp to identify the YouTube session.

**Fig. 1 & TABLE I:** Screenshot of the app and selected parameters from the HTML5 `<video>` object [18], Media Source Extensions [19], and device, which can be investigated by the app.

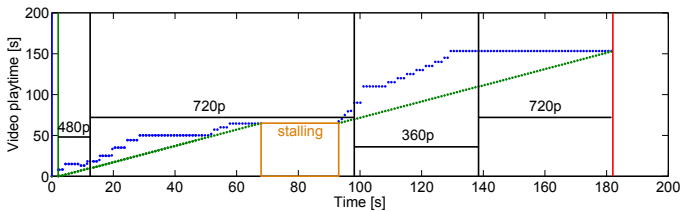
adaptive streaming according to the MPEG Dynamic Adaptive Streaming over HTTP (DASH) approach [17] of YouTube. Then, functions are injected, which ultimately perform the monitoring of the application parameters in the newly created app. The monitoring is done at runtime via JavaScript, which queries the embedded HTML5 `<video>` object. In Table I, the utilized parameters are listed. Note that the obtained parameters can be displayed in YoMoApp for validation (see Fig. 1), but are usually hidden. The measurement methodology in the app follows four consecutive steps:

**Step #1: HTML Detection** - In this step, the YouTube web page is detected and the HTML video player element is identified. YouTube consists of many web pages and elements. Thus, the name and id of the relevant video elements must be determined. The detection is done via the injection of JavaScript code to the running Android WebView browser element of the app. The JavaScript code analyzes the HTML Document Object Model (DOM) tree for the `<video>` element.

**Step #2: Request the Data** - Here, the specified parameters are queried such as the current playtime or the current available video content. This is realized by injecting JavaScript code that requests the application parameters from the detected video player element in Step #1 every second.

**Step #3: Calculation** - On the basis of the retrieved parameters, the buffer filling level can be calculated. The parameter *buffered* is subtracted from *currentTime*, which results in the current buffer level. Likewise, the video resolution is obtained based on *height* and *width* of the video player.

**Step #4: Data Transfer** - The last step, which is implemented in the app, is to transfer the data to an external database located in the Internet. For this purpose an external library is used, which compresses the data and stores them as structured objects. The data can be transmitted at different time instances: a) when closing the app, b) manually by the user, c) at predetermined intervals. The time of transmission is always stored together with the data. The data must also be cached locally, since the connection from the phone to the Internet server is often not reliable.



**Fig. 2:** Illustration of monitored parameters for an exemplary video streaming: current video playtime (green) and buffered video playtime (blue). Events are displayed as vertical lines: page load (blue), playback start (green), quality switch (black), playback end (red). Stalling is depicted as orange box. The horizontal black lines indicate the currently played out video quality/resolution.

Fig. 2 shows the data of an exemplary run in their processed form. Postprocessing of the data is recommended because the usage of JavaScript can sometimes introduce inconsistencies and obvious errors, e.g., missed player events, non-equidistant data queries, missing/incorrect values. However, after removing unusable runs and the recovery of missing events (e.g., stalling events can be estimated from buffer filling level), YoMoApp proved to perform accurate measurements on a sufficiently small time scale ( $\sim 1$  s).

#### IV. USING YOMOAPP IN SUBJECTIVE QOE TESTS

To demonstrate the applicability of YoMoApp to analyze the QoE of YouTube in mobile devices, we conducted a subjective study in which participants watched YouTube videos in smartphones instrumented with the YoMoApp tool and rated the resulting watching experience. To induce different QoE levels, the downlink traffic from YouTube servers to the mobile devices was throttled through an intermediate instrumented router imposing different bandwidth profiles.

##### A. Study Setup and Simple Traffic Characterization

The subjective study was performed in a dedicated lab for subjective analysis, compliant with the recommendations provided by the QoE subjective studies standards [20], [21], [22]. 52 people participated in the study (29 female, 23 male), the average age was 32 years old, with 40 participants being less than 30 years old. Around half of the participants were students and almost 43% were employees, and 70% of the participants completed university or baccalaureate studies. Participants were compensated with vouchers, which proved to be sufficient for achieving reliable and thoughtful involvement.

Android smartphone devices (Samsung Galaxy S4, OS Android 4.4 KitKat) were used in the tests. The devices were connected to the Internet through independent WiFi access points. The downlink traffic was routed through a modified version of the NetEm network emulator [23] to impose different access network bandwidth profiles to the video streaming. Three different bandwidth profiles were used: (i) constant downlink bandwidth: 1 Mbps, 2 Mbps, and 4 Mbps; (ii) fluctuating downlink with variable bandwidth (“var”): downlink bandwidth is periodically increased from 1 Mbps to 3 Mbps for periods of 5 seconds, 3 times per minute. The resulting average downlink bandwidth is 1.5 Mbps; (iii) downlink bandwidth outages (“out”): downlink bandwidth drops from 4 Mbps to 0 Mbps for periods of 10 seconds, twice per minute. In this case, the resulting average downlink bandwidth is 2.7 Mbps.

All traffic flows were captured and exported to standard `pcap` traces for off-line analysis and traffic characterization using high-performance Endace DAG cards.

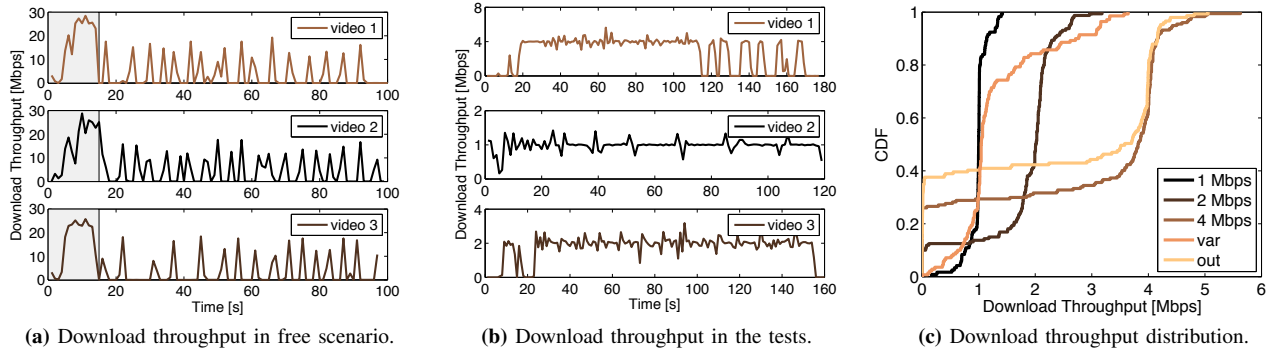
The specific task imposed to participants was to watch two minute long YouTube adaptive streaming videos using YoMoApp for the five different downlink bandwidth conditions, and rate the resulting watching experience (i.e., overall experience, impact of initial playback delay and stalling, video image quality) afterwards on continuous ACR MOS scales [20]. Each of the five conditions was linked to one of the five videos indicated in Table II, which included four mainly nature-themed clips and a movie trailer. All videos are available as 4K ultra-HD videos (i.e., 2160p), but the maximum video quality observed in the tests was HD (i.e., 720p) due to the devices’ display capabilities (i.e., screen size and resolution).

To better understand the impact of the traffic shaping on the resulting YouTube traffic flows, Fig. 3 depicts the download throughput as measured at the network level using the collected `pcap` traces. Fig. 3a depicts the observed throughputs when no shaping is done, which we shall refer to as the “free” scenario (in fact, download bandwidth is limited to 25 Mbps due to technical constraints, i.e., the physical connection maximum speed). For the sake of brevity, only the first three videos are depicted in Fig. 3a and 3b (see Table II). A very similar traffic pattern is observed for the three videos in the free scenario: there is an initial big video block which is downloaded as fast as possible to fill-in the playout buffer (see the average maximum throughput at about 25 Mbps), and then subsequent smaller blocks downloaded at periodic intervals. The observed throughputs when throttling the traffic follow the expected shapes with a constant download throughput at 4 Mbps, 1 Mbps, and 2 Mbps, respectively. Interesting is the fact that the 4 Mbps condition is high enough to free the traffic-shaper queue, as observed for video 1 in Fig. 3b after about two minutes download. The measured average download throughputs for both the free scenario and the subjective tests are included in Table II.

Fig. 3c depicts the cumulative distribution function of the instantaneous download throughput measured for each video at each condition using a time window of one second to compute each throughput sample. As expected, the distribution is centered at 1 Mbps for both the 1 Mbps condition and the variable condition (which also shows a mode at 3 Mbps due to the bandwidth increases), and no 0-throughput samples are observed, showing that the throttling was very aggressive in these conditions. The 2 Mbps condition shows some empty gaps in the throttling (i.e., 0-throughput samples), which are caused by an initial quality switch and a subsequent download delay as observed in Fig. 3b. Finally, the 4 Mbps condition and the outage condition have both a clear mode at 4 Mbps, with a large number of 0-throughput samples, caused by enough bandwidth allocation in the former condition, and by the bandwidth outages in the latter.

##### B. Analysis of Stallings and their Impacts on User Experience

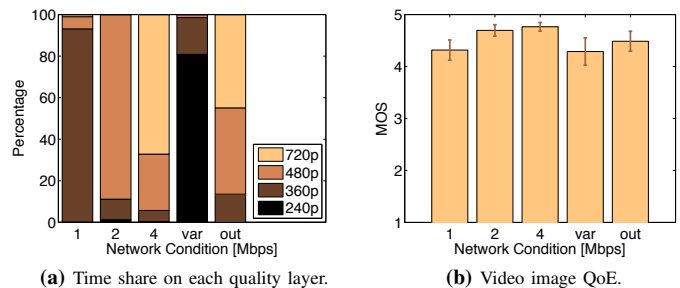
Stalling, i.e., the interruption of playback due to a playout buffer under-run, is considered the worst quality degradation of video streaming [7]. Stalling occurs when the available



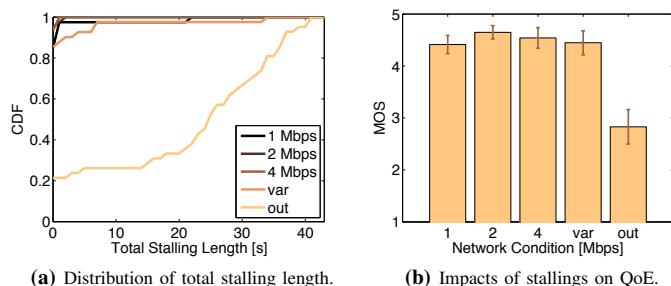
**Fig. 3:** Download throughput as measured at the network level. The plots depict the throughput for a free scenario in which traffic is not shaped, as well as the throughput observed for three of the tested traffic-shaping conditions (4 Mbps, 1 Mbps, and 2 Mbps).

**TABLE II:** Video content and download throughputs as measured in the lab experiment.

Video Nr.	YouTube Video ID	Avg. Th. (condition)	Avg. Th. (free)
1	6pxRHBw-k8M	2.8 Mbps (4 Mbps)	4.6 Mbps
2	iNJdPyoqt8U	1.0 Mbps (1 Mbps)	5.8 Mbps
3	kObNpTFPV5c	1.8 Mbps (2 Mbps)	5.0 Mbps
4	QS7IN7giXXc	2.3 Mbps (out)	5.0 Mbps
5	suWsd372pQE	1.3 Mbps (var)	3.9 Mbps



**Fig. 5:** Monitoring of video quality switches and the resulting image QoE. In (b), participants rated the image quality on a continuous scale ranging from 1 (*bad*) to 5 (*excellent*).



**Fig. 4:** Monitoring of stallings and their impact on user experience. In (b), participants rated the disturbance of the stallings on a continuous scale ranging from 1 (*very disturbing*) to 5 (*not disturbing at all*).

network bandwidth is lower than the video bit rate. The playout buffer is filled slower by the download than it is emptied by the playback, which will eventually lead to stalling. During a stalling event, the playback is paused until the buffer is filled with a sufficient amount of video data to continue the playback. [7] found an exponential relationship between stalling parameters and MOS and that users tolerated at most one stalling event of up to three seconds length. Thus, it is important to monitor the stalling during the video playback.

Fig. 4a shows the cumulative distribution function of the total stalling length for each tested condition. It can be observed that for the constant bandwidth conditions, almost no stalling occurs. For 4 Mbps, 93.48% of the streaming sessions have no stalling, the remaining sessions have less than 1 s of stalling. A maximum of 2 s of stalling was measured for the 2 Mbps condition, with 95.12% of the sessions showing no stalling at all. For the 1 Mbps condition, 85.00% of the streaming

sessions do not include stalling. However, one outlier with 22 s stalling was observed. For the variable condition (“var”), stalling occurs in 14.63% of the conditions ranging up to a total stalling length of 34 s. 78.57% of the outage condition (“out”) streams contained stalling. The average total stalling length in this condition is 25 s, and the maximum total stalling length is 41 s.

In Fig. 4b, the MOS and 95% confidence intervals of the respective stalling ratings are presented. Participants were asked to which extend they perceived the interruptions caused by stalling as disturbing. The MOS values for the disturbance of stalling are ranging from 1 (very disturbing) to 5 (not disturbing at all). For 1 Mbps, 2 Mbps, 4 Mbps and variable condition, stalling is not considered disturbing having mean opinion scores of at least 4.41. This corresponds to the measured total stalling time, which indicated very short stalling events, if any at all. Only for the outage condition, stalling is perceived as disturbing having a MOS of 2.83. Again, this corresponds to the frequently long total stalling times measured by YoMoApp.

### C. Analysis of Quality Switches

HTTP adaptive streaming trades off stalling for video quality. It adapts the downloaded video quality (i.e., video bit rate) to the currently available network conditions, thereby avoiding stalling to the greatest possible extend. In YouTube, which is considered in this work, this is implemented by

adaptively changing the resolution of the streamed video. However, quality switches, i.e., the change of the video bit rate, occur during streaming and can be perceived by the users. In the following, the monitored quality of the video streaming will be investigated.

Fig. 5a shows the percentage of time on each quality layer per condition, i.e., the percentage of time which each video resolution was played out during the streaming. In the 1 Mbps condition, all available resolutions were used with the following overall shares: 240p (0.23%), 360p (92.89%), 480p (5.85%), 720p (1.02%). The 2 Mbps condition shows a larger percentage of 480p quality (88.84%), and in the 4 Mbps condition a significant share of 720p quality (67.15%) can be played out. Additionally, the outage condition has similar quality shares to the 4 Mbps conditions, which is not surprising considering that it is a 4 Mbps on/off pattern. The variable condition contains a large percentage of the lowest resolution (240p, 80.86%), which indicates that the YouTube adaptation is very conservative when the network conditions fluctuate considerably.

Fig. 5b shows how the image quality of each condition was rated by the participants (MOS and 95% confidence intervals). It can be observed that the image quality is rated good for all conditions having a MOS of at least 4.17 for all conditions. This means that resolution adaptation does not have a big impact on the subjectively perceived image quality, which could be due to the small display size of the used smartphone. Similar findings can also be found in [12]. Nevertheless, the increasing levels of quality played out for 1 Mbps, 2 Mbps, and 4 Mbps correspond to the increasing image quality ratings by the participants.

## V. CONCLUSION

In this paper, we presented YoMoApp, an Android application, which exactly replicates the behavior of the well-known YouTube service including its design and its HAS technology. YoMoApp passively monitors and stores QoE-relevant KPIs of the YouTube adaptive video streaming, such as player state/events, buffer, and video quality level (resolution). Thus, the monitored data can be used to analyze the QoE of mobile YouTube, which is of special importance to mobile operators as it is among the most popular and most volume-demanding services in today's Internet.

A subjective study was conducted to test the implementation and associate its monitored data with the subjective ratings of the participants. The measured total stalling lengths and time on each quality layer could explain the subjective ratings, which indicates that the application works as expected. As such, YoMoApp proves to be a valuable tool for analyzing the QoE of YouTube streaming sessions on mobile devices. This passive client-side measurement approach can be applied to assess the momentarily performance of mobile networks as well as the impact of traffic management policies/mechanisms on mobile YouTube QoE.

In future work, YoMoApp will be used to investigate the streaming sessions in more technical detail. The monitored video sessions will be analyzed for initial delays, number and lengths of stalling events, and quality switches. These data will provide more insights into how the network conditions affect mobile YouTube streaming. Moreover, further subjective

studies on mobile devices will be conducted to correlate the monitored streaming and adaptation parameters to subjective quality ratings. This allows to quantify the impact of the streaming and adaptation parameters on the subjective quality and to develop improved QoE models for mobile HAS.

## REFERENCES

- [1] [Online]. Available: <https://www.youtube.com/yt/press/statistics.html>
- [2] P. Casas, M. Seufert, and R. Schatz, "YOUQMON: A System for On-line Monitoring of YouTube QoE in Operational 3G Networks," *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, 2013.
- [3] B. Staehle, M. Hirth, R. Pries, F. Wamser, and D. Staehle, "YoMo: A YouTube Application Comfort Monitoring Tool," in *QoEMCS*, 2010.
- [4] F. Wamser, T. Zinner, L. Iffländer, and P. Tran-Gia, "Demonstrating the Prospects of Dynamic Application-aware Networking in a Home Environment," in *ACM SIGCOMM*, 2014.
- [5] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hößfeld, and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys & Tutorials*, vol. PP, 2014.
- [6] F. Wamser, D. Hock, M. Seufert, B. Staehle, R. Pries, and P. Tran-Gia, "Using Buffered Playtime for QoE-Oriented Resource Management of YouTube Video Streaming," *Transactions on Emerging Telecommunications Technologies*, vol. 24, 2013.
- [7] T. Hößfeld, R. Schatz, M. Seufert, M. Hirth, T. Zinner, and P. Tran-Gia, "Quantification of YouTube QoE via Crowdsourcing," in *MQoE*, 2011.
- [8] R. K. P. Mok, E. W. W. Chan, X. Luo, and R. K. C. Chan, "Inferring the QoE of HTTP Video Streaming from User-Viewing Activities," in *W-MUST*, 2011.
- [9] T. Hößfeld, S. Egger, R. Schatz, M. Fiedler, K. Masuch, and C. Lorentzen, "Initial Delay vs. Interruptions: Between the Devil and the Deep Blue Sea," in *QoMEX*, 2012.
- [10] J. Yao, S. S. Kanhere, I. Hossain, and M. Hassan, "Empirical Evaluation of HTTP Adaptive Streaming Under Vehicular Mobility," in *Networking*, 2011.
- [11] B. Lewcio, B. Belmudez, A. Mehmood, M. Wältermann, and S. Möller, "Video Quality in Next Generation Mobile Networks – Perception of Time-varying Transmission," in *CQR*, 2011.
- [12] T. Hößfeld, M. Seufert, C. Sieber, and T. Zinner, "Assessing Effect Sizes of Influence Factors Towards a QoE Model for HTTP Adaptive Streaming," in *QoMEX*, 2014.
- [13] J. J. Ramos-Muñoz, J. Prados-Garzon, P. Ameigeiras, J. Navarro-Ortiz, and J. M. López-Soler, "Characteristics of Mobile YouTube Traffic," *IEEE Wireless Communications*, vol. 21, 2014.
- [14] I. Ketykó, K. De Moor, T. De Pessemier, A. J. Verdejo, K. Vanhecke, W. Joseph, L. Martens, and L. De Marez, "QoE Measurement of Mobile YouTube Video Streaming," in *MoViD*, 2010.
- [15] G. Gómez, L. Hortigüela, Q. Pérez, J. Lorca, R. García, and M. C. Aguayo-Torres, "YouTube QoE Evaluation Tool for Android Wireless Terminals," *EURASIP Journal on Wireless Communications and Networking*, vol. 164, 2014.
- [16] V. Aggarwal, E. Halepovic, J. Pang, S. Venkataraman, and H. Yan, "Prometheus: Toward Quality-of-Experience Estimation for Mobile Apps from Passive Network Measurements," in *HotMobile*, 2014.
- [17] ISO/IEC, "23009-1:2012 Information Technology – Dynamic Adaptive Streaming over HTTP (DASH) – Part 1: Media Presentation Description and Segment Formats," 2012.
- [18] [Online]. Available: <http://developer.mozilla.org/en-US/docs/Web/HTML/Element/video>
- [19] [Online]. Available: <http://dvcs.w3.org/hg/html-media/raw-file/tip/media-source/media-source.html>
- [20] ITU-T, "Recommendation P.800: Methods for Subjective Determination of Transmission Quality," 1996.
- [21] —, "Recommendation P.910: Subjective Video Quality Assessment Methods for Multimedia Applications," 2008.
- [22] —, "Recommendation P.1501: Subjective Testing Methodology for Web Browsing," 2013.
- [23] S. Hemminger, "Network Emulation with NetEm," in *LCA*, 2005.