Light Weight Content Fingerprinting for Video Playback Verification in MPEG DASH

Zhu Li and Imed Bouazizi
Multimedia Core Standards Research
Samsung Research America - Dallas
{zhu1.li, i.bouazizi}@samsung.com

Abstract — **Adaptive HTTP Streaming solutions are phasing out traditional online video distribution solutions such as progressive download. MPEG DASH is an open standardized solution that has been developed to minimize solution fragmentation and to ensure quick market adoption. However, the openness of the standard loosens the grip of content providers on the client behavior and may threaten the success of the whole ecosystem. This paper proposes a content fingerprinting and verification mechanism for restricting the client's playback behavior in such an open environment by using a very light weight Eigen thumbnail appearance differential fingerprinting. Simulation results demonstrate the effectiveness of the proposed solution.**

Keywords—MPEG DASH; Video Fingerprinting; Playback Verification; Eigen Appearance;

I. INTRODUCTION

Online video ads are becoming the main revenue channel for content providers due to the exponential growth in online video consumption. A larger portion of advertising budgets is now being allocated to online video. In return to watching free content on the Internet, the user is "forced" to watch a short ad. The ad may be inserted at the start (pre-roll), in the middle (mid-roll), or towards the end (post-roll) of the original content. While the mid-roll option is very popular in traditional linear TV, pre-roll has been very popular in online video. The ads are typically 15 seconds spots and thus much shorter than the classical ads on TV.

This business model of sponsoring online video through online video advertisements has established itself in the media distribution industry. Several players contribute to building this eco-system. Those include content delivery networks (CDNs), analytic data providers, ad networks and ad exchange platforms. Impressions are sold via ad-exchange platforms and the selected ad is delivered by the CDN. Verification and analytics tools verify the completion rate of the ads and report this information to the advertisers.

DASH defines an open standard for adaptive media streaming over HTTP. DASH uses open standards such as XML, HTTP, and MPEG ISO-Base Media File Format for building the streaming function. Contrary to classical streaming approaches, DASH is client-driven, which means that the client is in full control of the content it receives. The service provider offers to the client a set of variants to choose from and combine, in order to optimize the delivery experience. The variants are described in the MPD, which is an XML formatted document.

Recently, W3C has published an API for web browsers to feed content segments received from multiple media sources to an integrated media player. This API integrates seamlessly with the HTML 5 media tags and enables the support of DASH and other adaptive media streaming solutions over HTTP.

As a consequence, a large variety of client implementations, most of which will be open-source, will be offered to the clients. For instance, web sites may offer Javascript DASH implementations as part of their web pages. Users may also use their own players or modify existing player implementations to play content offered via DASH. The Interactive Advertising Bureau (IAB) has developed a set of recommendations such as the Video Ad Impression Measurement Guidelines (VAST)[1], which defines procedures to report about ad impressions. However this also relies on client player certification.

Given these facts, it is difficult to establish a trust relationship between the service provider and the DASH client. This fact jeopardizes the existing online video delivery eco-system, which requires a trusted client to display the ad to the viewer at a given time point and for a given period of time.

So far, ad insertion in DASH may occur in two different ways: 1) ad splicing where content is pre-inserted as part of the original media content 2) provided separately, e.g. as a new Period in the content. While the former option offers better reliability, it limits the flexibility of the ad insertion, e.g. ad customization and dynamic decision about the ad to be inserted. The latter, however, and in the absence of trusted DASH clients will mark pieces of content as ads and thus, literally invite implementations to bypass those ads completely.

In this paper, we propose a new approach for playback verification for DASH, which can be used to enforce playback of ads. In section 2, we give a brief introduction to the MPEG DASH standard. In section 3, we present the approach for content playback verification. In section 4, we discuss content finger printing as a tool to verify playback. In section 6, we show simulation results to analyze the accuracy of fingerprints in identifying content. Section 7 provides a conclusion and potential future research directions.

II. MPEG DASH

MPEG has standardized a solution for adaptive HTTP streaming under the name MPEG DASH [2]. DASH enables delivering content from any regular web server.

In order to enable client rate adaptation, DASH encodes content in multiple variants at different bitrates, named representations. The authors in [3] discuss different ways to perform rate adaptation in DASH. In addition, content may consist of multiple components, which may be offered for delivery separately. Representations of the same media content are offered as adaptation sets among which the DASH clients may select the most appropriate one to match their available bandwidth.

The available adaptation sets and representations are described in an XML formatted manifest file, the Media Presentation Description (MPD), which is provided to the DASH clients to start streaming. The MPD also divides the media presentation into multiple consecutive time intervals, called periods. To facilitate switching between representations at the DASH clients, a representation may be segmented into one or more segments. After downloading one media segment, the DASH client will typically assess the actual throughput of its channel and decide whether to keep the current representation or switch to another representation with a higher or lower bandwidth requirement. The structure of the MPD is depicted byFigure 1.

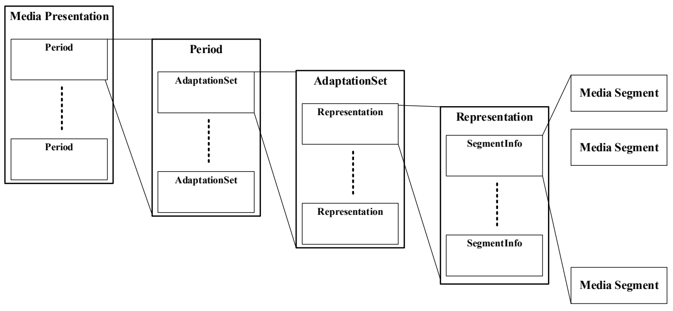


Figure 1 MPD Structure

DASH supports two different segment formats, one based on the ISO-Base Media File Format and one based on the MPEG-2 Transport Stream format. Each segment is supposed to contain a random access point (RAP) to enable successful switching. A segment is also identified by a segment index and is assigned a segment URL. The segment itself contains more information to assist the DASH client in performing seamless switching. This information is provided as part of a segment index box. The design principles behind MPEG DASH are discussed in more detail in [4].

III. AD PLAYBACK VERIFICATION

As described previously, DASH enables two ways of ad insertion, either as ad splicing as part of the same representation or through starting a new period in the presentation. Ad periods are susceptible to manual and automatic skipping. A client implementation may automatically identify short periods (having the duration of a one or a few ad clips) and skip them automatically. In both approaches, users may also skip the ad content by seeking forward.

In order to enforce DASH clients to consume the ad content, a playback verification approach is proposed in this paper. The playback verification mechanism is based on creating dependencies among the content. A dependency implies that a piece of content cannot be played back before the playback of another piece of content. This approach is depicted in Figure 2.

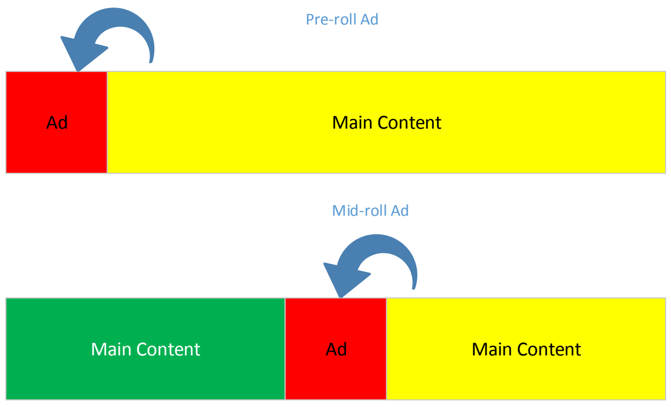


Figure 2 Content Dependency for Playback Verification

An indication in the MPD informs the DASH client about the existing dependencies between the different content pieces. Dependent content becomes conditionally accessible based on a playback verification mechanism. The DASH client calculates a token from the Ad content and uses that token to gain access to the main content. Figure 3 shows the client interactions to access to the main content after playing the Ad content.

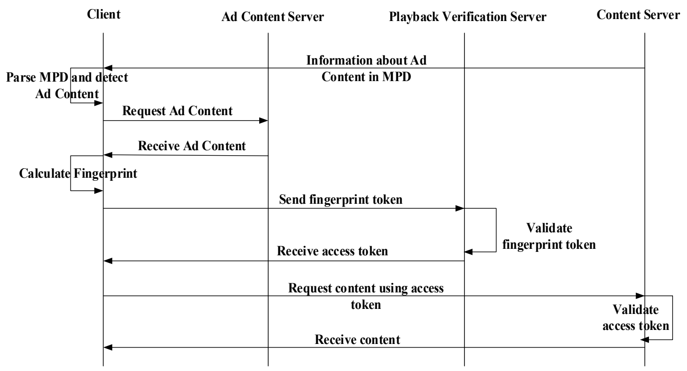


Figure 3 Client Interactions for Playback Verification

After identifying and playing the Ad content at the scheduled time, the DASH client will calculate a verification token that it will use to verify that the Ad has been played back. It sends that token to the server together with information about the Ad clip that was played to the playback verification server.

The playback verification server will match the received verification token against the one it has already calculated for the same Ad content and in case of a match will provide the DASH client with an access token. The access token is then used to access the main content.

In this paper, we propose the use of codec and quality invariant fingerprinting mechanism to calculate the verification token in the spatial domain. This forces the DASH client to at least receive and decode all the content, in order to be able to generate the verification token and access the depending content. The details of this fingerprinting approach are described in section 4.

IV. Differential Fingerpinting of VIdeo Segments for Playback Verification

For ad video playback verification, we need to find a video feature that is very light weight in computing and communication overhead, while robust to coding and communication losses in playback. For this purpose we investigated a video fingerprint scheme [6] that was originally devised for the task of real-time visual surveillance [7]. An 1-dimensional signature is computed for ad video clips as follows: the frames are first down-sampled to thumbnail size of *w*x*h* pixels, then an offline thumbnail Eigen appearance modeling is performed over a data set $\{f_k\}$ in R^{wxh}, randomly sampled from a large video repository [8]. The Eigen appearance model of video thumbnails, *A* is obtained by,

$$A^* = \arg\max_A \sum (x_k - m)' A' A (x_k - m) \qquad (1)$$

which is solved using PCA. For thumbnail sizes of *w*=16, *h*=12, the Eigen values of PCA are plotted in Figure 4 below,

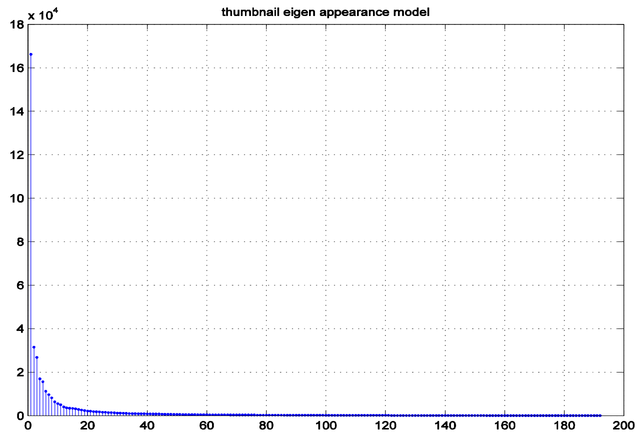


Figure 4 Thumbnail appearance model Eigen values

As shown in Figure 4, the thumbnail itself even at the size of 16x12 pixels still has a lot of redundancy inside, by selecting only a limited number of *d* PCA components, we can further reduce the video sequence to a low *d*-dimensional signature,

$$x = Af \qquad (2)$$

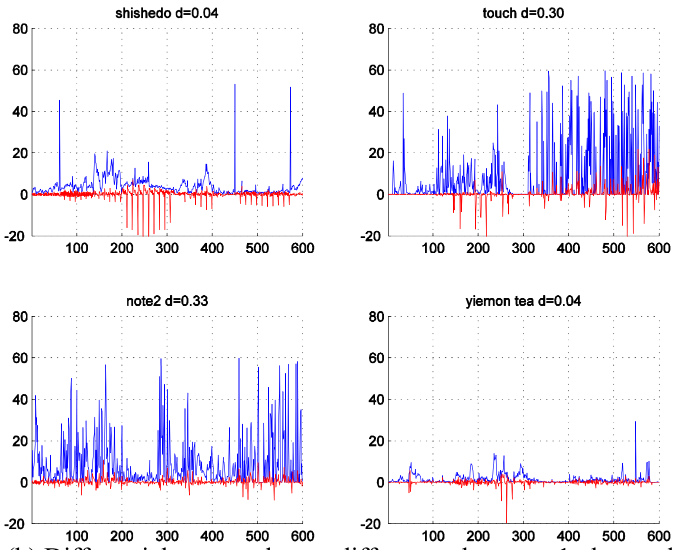
where *A* is *d* x (*w* x *h*). In [6] we have demonstrated that this can be an effective video segment de-duplication solution over a large repository. But for the playback verification problem, which is much less demanding than the identification problem in de-duplication, an even more compact signature can be derived.

An Eigen appearance differential trace is therefore computed for this purpose. For a video segment of *n*-frames, and its thumbnails, $\{f_1, f_2, \ldots, f_n\}$ in $\boldsymbol{R}^{wxh}$, its differential 1-dimensional signature is computed as,

$$dx(k) = \begin{cases} 0, & if\ k = 1 \\ Af_{k+1} - Af_k, & else \end{cases} \qquad (3)$$

This differential feature is very compact, and requires only 8 bits per frame to describe, this translate into approximately 200 bps communication overhead for video sequences frame rate at 25 *fps*. Some examples of ads video sequences and their differential signatures are plotted in Fig. 5.

(a) Ads sequences: shisheido, touch, note 2, and yiemon



(b) Differential trace and trace differences between 1mbps and 400kbps coded sequences

Figure 5 Differential Thumbnail Eigen Appearance Examples

The playback verification is performed as follows, on the client side, after video is decoded, a thumbnail is computed for each frame, and its differential trace signature is computed according to Eq. 3, and communicated back to the server for verification. A threshold θ is tested to determine positive or negative verification of two video sequences of *m*-frames, and their differential signature, dx^1 and dx^2,

$$v(dx^1, dx^2) = \begin{cases} 1, & if\ \frac{1}{m}\sum_{k=1}^{m}|dx_k^1 - dx_k^2| < \theta \\ -1, & else \end{cases} \quad (4)$$

Notice that, at different coding rates and potential stream switching, it could result in a sequence that is not exactly the same as the single rate stream that is stored at the server.

Examples of ad sequences and their differential Eigen thumbnail fingerprints, are illustrated in Figure 5. In Figure 5a, thumbnails from 4 ad sequences obtained from Youtube are illustrated, their differential eigen thumbnail appearances are plotted in Figure 5b, in blue lines. Notice that ad sequences are typically very dynamic, with many scene cuts and actions, reflected by the 3 sequences, “shishedo”, “touch”, and “note 2”, while the 4th one, “yiemon”, was selected because of its less dynamic content, which is more similar to regular program content, as indicated by its differential traces. The average differences between the original sequences coded at 1mbps, and their 400kpbs coded alternative stream, are plotted in red, and their average differences are summarized in the Table I below.

Table 1 Average Differences of Selected Ad Sequences Encoded at Different Bitrates

Seq.	“shishedo”	“touch”	“note 2”	“yiemon”
Distances	0.04	0.30	0.33	0.04

The average differences are very small compared with the dynamic range of the differential trace, which points to a high SNR of this signature to coding variations. As discussed, the thumbnail Eigen appearance modeling processes, has a strong de-noising effect that can smooth out these differences, and still offer robust verification performance. This is illustrated by these 4 examples and shall be examined more thoroughly in the simulation section over a larger data set and more realistic test set up.

To further improve the performance, a noise suppression scheme is applied at the differential Eigen appearance computing phase. A max difference threshold is applied, i.e, if $dx(k) > d_{max}$, then $dx(k)$ is set to value d_{max}.

$$dx(k) = \begin{cases} 0, & if\ k = 1 \\ d_{max}, & else\ if\ A(f_{k+1} - f_k) > d_{max} \\ A(f_{k+1} - f_k), & else \end{cases} \quad (5)$$

The resulting signature is only 1-dimensional and can be quantized at 8bits per frame sample, which results in a signature that is only 200bps for video sequences at 25fps.

V. Simulation Results

To verify the effectiveness of the proposed light weight video fingerprinting system in playback verification, a test data set is collected from various sources consisting mostly of commercial videos and movie trailers. There are n=4000 video clips of max length of t=60s in total. The test data set videos are all 720x480 pixel resolution videos, and coded at 3 rates, R=[480kbps, 640kbps, 800kbps].

A distractor data set is collected from TRECVID video data set [8], and random collections from Youtube, total about 120 hours. These sequences covered a variety of content types and activity intensity levels, and are coded at various rates ranging from 300kpbs to 1mbps. The purpose of the distractor data set is to demonstrate that a false signature will be rejected by the playback verification system.

To compute the differential signature, we choose the thumbnail size [w=16, h=12], and the dimension of the Eigen thumbnail appearance space is set as d = 6. The 6 basis functions that the thumbnails are projected, are illustrated in Figure 6 below. The 6 basis functions a_1, a_2, …, a_6, correspond to the 6 largest Eigen value basis in solving Eq. 1). The choice of the dimensionality in computing the differential reflects the trade-off between signature resolution and robustness to coding at different quality levels.

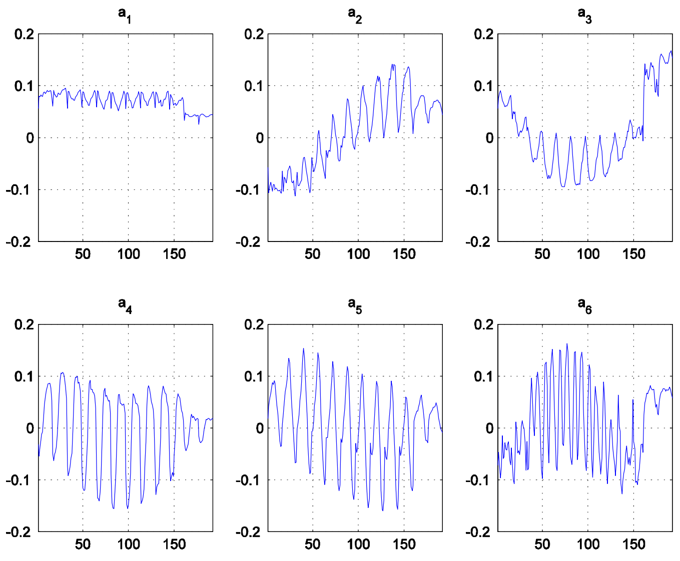


Figure 6 Thumbnail Eigen Appearance Basis Functions

The *positive probe* tests are conducted by computing the differential 1-d signatures of test data set at lower bit rates, 640 and 480kbps, and compute their distance with the original signature extracted from the 800kbps version. The *false positive* probe tests are conducted by randomly selecting m=10 clips from the distractor data set, and compute their differential signature and distances to the differential signature of that of the test data set. For each ads segment in the testing dataset, there are total 2 true positive probes, and 10 false positive probes. The distance histograms for true positive and false positive probes are plotted in Figure 7.

The false positive pair distances are distributed over a wide range, with a mean of 12.37 and standard deviation of 9.89, while the true positive pair distances are tightly distributed around a mean of 0.77, and standard deviation of only 0.25.

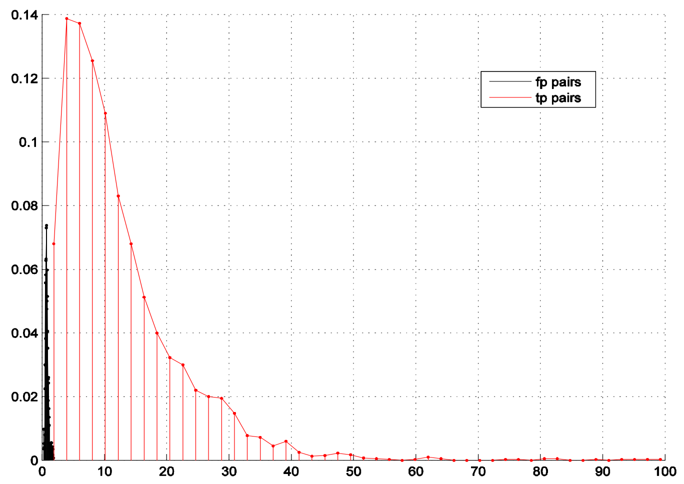


Figure 7 True positive and false positive pair distances histogram

A distance threshold θ is applied, to guarantee 100% true positive rate, i.e, never rejecting a correct verification request, and the resulting false positive rates, i.e, how many times a bogus signature was mistaken for a true played back sequence, are shown in the Table 2, for probing video clips of length t=[60, 30, 15] seconds.

Table 2 False positive rates at 100% True positive rate

	t=60s	t=30s	t=15s
R=640kbps	0.15%	0.52%	1.30%
R=480kbps	0.15%	0.60%	3.15%

It is noted that as the video clips become shorter, the false positive rates go up, but for typical commercials of 30 sec or more, the verification accuracy is good. At no false negatives in verification, the false positive rate is less than 1%.

The cost of computing the differential signature is small, accounting for less than 0.5% of the total complexity of FFMPEG [9] decoding process. The communication overhead is approximately 8bit per frame that is approximately 200bps for a typical 25fps video, regardless of its bit rate and frame size.

VI. Conclusion & Future Work

In this paper we presented a very light weight video fingerprint and verification solution for video playback verification in modern open multimedia transport systems like DASH. The computational and communication overhead incurred are minimal, and the accuracy of the verification is good, and quite suitable for ad playback verification for MPEG DASH streaming system. In the future we will introduce some filtering to the differential signature to make it more robust to transcoding induced variations, and further improve the accuracy performance and bandwidth requirements. A matched filtering solution will also be investigated for ad playback statistics and localization in sequences.

References

[1] IAB, Digital Video Ad Impression Measurement Guidelines, December 2009

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

[3] C. Liu, I. Bouazizi, M. Gabbouj, "Rate Adaptation for Adaptive HTTP Streaming", ACM Multimedia Systems 2011, San Jose, CA, USA, Feb. 2011.

[4] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP- Standards and Design Principles", ACM Multimedia Systems, San Jose, CA, USA, Feb. 2011.

[5] G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*

[6] Bo Liu, Zhu Li, Linjun Yang, Meng Wang, Xinmei Tian: Real-Time Video Copy-Location Detection in Large-Scale Repositories. *IEEE Multimedia* 18(3): 22-31 (2011).

[7] Zhu Li, Yun Fu, Thomas S. Huang, Shuicheng Yan: Real-time human action recognition by luminance field trajectory analysis. *ACM Multimedia* 2008: 671-676.

[8] NIST TRECVID Data Set: http://www-nlpir.nist.gov/projects/trecvid/trecvid.data.html

[9] http://www.ffmpeg.org/