QoE-aware Inter-stream Synchronization in Open N-Screens Cloud

Mu Mu*, Steven Simpson†, Hans Stokking ‡ and Nicholas Race†
*Department of Computing and Immersive Technologies, The University of Northampton, Northampton, UK
mu.mu@northampton.ac.uk
†School of Computing and Communications, Lancaster University, Lancaster, UK
f.lastname@lancaster.ac.uk
‡TNO, The Hague, The Netherlands
hans.stokking@tno.nl

***Abstract*—The growing popularity and increasing performance of mobile devices is transforming the way in which media can be consumed, from single device playback to orchestrated multi-stream experiences across multiple devices. One of the biggest challenges in realizing such immersive media experience is the dynamic management of synchronicity between associated media streams. This is further complicated by the faceted aspects of user perception and heterogeneity of user devices and networks. This paper introduces a QoE-aware open inter-stream media synchronization framework (IMSync). IMSync employs efficient monitoring and control mechanisms, as well as a bespoke QoE impact model derived from subjective user experiments. Given a current lag, IMSync's aim is to use the impact model to determine a good catch-up strategy that minimizes detrimental impact on QoE. The impact model balances the accumulative impact of re-synchronization processes and the degree of non-synchronicity to ensure the QoE. Experimental results verify the run-time performance of the framework as a foundation for immersive media experience in open N-Screens cloud.**

I. Introduction

The explosion in digital and mobile device ownership has greatly changed the focus of social and interactive TV from adding sophisticated features to the TV set to enhancing user experiences with tablets and smartphones as second screen devices [7], [2]. With the help of emerging hardware and software technologies, mobile devices are now regularly used to carry multimedia streams. Market research shows that 40%-65% of tablet devices are used to stream movie and TV programmes at least once a day [6]. In the U.S. mobile users are now spending 59% more time watching video on their mobile devices compared to 5 years ago [17]. However, the media experience offered by a single user device is limited by the capabilities of the equipped hardware. Meanwhile, the role of second screen devices has also evolved: from a companion device offering non-essential functions, to a key player in a *device cloud* (N-Screens), transforming media applications and the user experience. We have also seen recent developments around semantic video applications that adapt existing single-screen applications to multi-screen environments based on author or user choices [20] and multi-screen orchestration that connects TV programs with "social sense" using mobile devices [8]. An example is the IllumiRoom project in which Microsoft looked into augmenting the area surrounding a television with projected visualizations to enhance traditional gaming experience [9]. The BBC took a similar approach in its Surround Video, an immersive video technology in a domestic-scale viewing environment. A short film Broken is also commissioned specifically for Surround Video [24]. There have also been psychological studies on attention split, cognitive load, perceived comfort, and the maximum number of screens that could be watched simultaneously [25], [1].

This paper focuses on the orchestration of multiple user devices as an open device cloud as the foundation to enable new and immersive media experiences. Its contributions lie in the design and realization of an open synchronization framework using web technologies as well as an adaptive synchronization model derived from the deep understanding and modelling of relevant human factors to ensure the user experience of collective and interactive media.

II. Use Case and Related Work

We picture a group of friends sitting outside a cafe when one of them decides to show an online video story she made in an international musical event on her tablet computer. She interacts with the video library as well as the playback to navigate between highlights of the event. The sound coming out of the tablet feels flat outdoors. Three of her friends take their smartphones out and join the application to contribute with background sound track whilst adding ambient light and vibrations as directed by the composite media stream to help recreate the immersive experience of the musical event. As a result, the user devices, connected via different networks such as WiFi and LTE, form an N-Screens device cloud and collaboratively create a vibrant and immersive media experience. Similar use cases have also been explored recently where mobile device clouds are constructed to offer advanced sound features such as multi-channel surround sound [12] and directional sound [3]. More such examples are also seen in the field of Internet of Things (IoT) [5]. The main challenge of ensuring the quality of user experience in immersive and interactive N-Screen applications is the real-time measurement, QoE evaluation, and control of the synchronicity between media objects in an ensemble of user devices over heterogeneous networks. Even a small degree of media non-synchronicity can be detrimental to user experience. Many external and internal factors such as clock drift and intermittent CPU overload at user devices, and user interactions such as pause and skip often cause linked media objects to fall out of sync.

Research on the topic of media synchronization is conventionally categorized into *intra-stream synchronization*, *inter-stream synchronization* and *inter-destination synchronization* (IDMS). Intra-stream synchronization addresses the fidelity of media playback with respect to temporal relationships between adjacent media units (MUs) within the same stream. Inter-stream synchronization refers to the preservation of temporal dependencies between the playout processes of correlated media streams [16], an example of which is lip-sync [21], [4]. With the increasing demand of simultaneous media streaming to geographically distributed end systems, the level of synchronicity between media streams has become a deterministic factor in assuring quality of user experience and fairness. Recently, Rainer et al. introduced a self-organizing control scheme with temporal distortion metrics based on the buffer level for peer synchronization in an IDMS session [18]. Montagud et al. extensively reviewed 19 emerging media applications that require inter-destination synchronization from the level of "very high" ($10\mu s - 10ms$) to "low" ($500ms - 2000ms$) [16]. Existing studies do not systematically model the quantitative joint impact of non-synchronicity and re-synchronization in the emerging scenario of multi-stream synchronization at the same physical location. It is then not possible to orchestrate media objects for the best overall user experience.

Examples of multi-device media applications have also emerged from industry, with the innovations in multi-room wireless audio being a particularly recent example (Sonos[1], Play-Fi[2], and Caskeid[3]). Most of the products in this area make use of customized chipsets or a proprietary mesh-network for synchronicity. Our work aims at providing an open, portable, and QoE-aware application-level synchronization framework, which can be enabled on different types of user devices with minimal configuration and user intervention. We take audio-visual media streaming as the first reference application. Through user experiments and data modelling, we also investigate and model user perception related to inter-stream synchronicity. Such modelling is valuable to minimise the impact of non-synchronicity and create new applications.

III. INTER-STREAM MEDIA SYNCHRONIZATION

The framework, shown in Figure 1, defines three reference device types: *master device*, *auxiliary device*, and *sync server* with each representing a specific role within an N-Screens device cloud. The master device is usually the first user device to start the media application and the last one to depart from (and terminate) the application. Auxiliary devices may join (and leave) at any point to contribute to the media experience and take the master as the reference device for synchronized playback. The master device does not maintain session information for auxiliary devices. The playback statistics of the master device are multicast to all auxiliary devices periodically, and also at crucial service events via a heartbeat mechanism. The auxiliary devices then work out whether they are out of sync and catch up using the best re-synchronization strategy with respect to user perception and device capacity. Any participating device can be elected as the master device during the course of the application. The sync server is a central point where measurements related to playhead position and player statistics are gathered and dispatched.

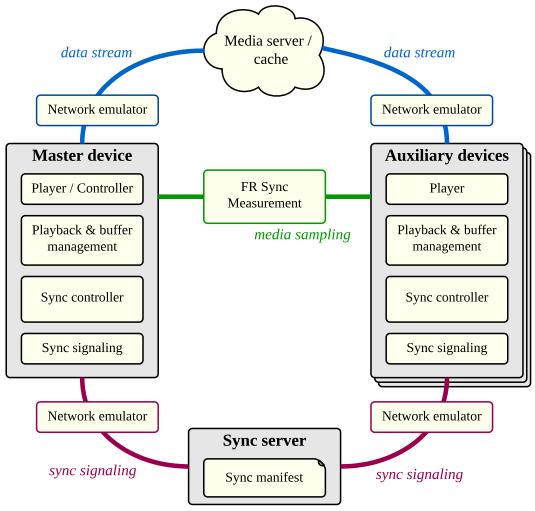


Fig. 1. Inter-stream synchronization framework and testing environment

The IMSync framework is designed with network impact and device capability in mind. We set up a testbed with controllable network emulators, a bespoke full-reference (FR) sync measurement device, and media servers (Figure 1). The network emulators allow us to evaluate the effectiveness of the framework in the context of best-effort delivery networks. The FR objective measurement device directly samples and comparatively measures the rendered outputs from media players in order to accurately evaluate the level of non-synchronicity between devices. The FR device is only used for building the impact model and evaluating the framework.

A. Master and auxiliary devices

Master and auxiliary devices share three functional modules: *playback and buffer management*, *sync signaling*, and *sync controller*.

1) Playback and buffer management: The playback and buffer management module directly interacts with media applications on the same device. The module also intervenes in activities of the player including playout rate adjustment and pre-emptive buffering according to decisions made by the sync controller. Modern web browsers provide detailed runtime statistics and control interfaces of the native audio and video playback engine. Monitoring the buffer level also provides insights into buffering delay, which is one of the main causes of non-synchronicity between user devices.

2) Sync signaling: In order to measure the discrepancy between the playhead positions of media streams, sync messages carrying information such as the playback statistics from participating devices are exchanged periodically and efficiently by the sync signaling module via the sync server.

The IMSync framework departs from the conventional designs with dependencies on the Network Time Protocol (NTP) and employs web technologies such as WebSockets to enable efficient full-duplex communication channels for the

[1]http://www.sonos.com
[2]http://www.playfiaudio.com
[3]http://www.imgtec.com/ensigma/caskeid/

exchange of timing information directly and synchronously. It also has specially designed mechanisms to compensate for any signalling or playback delay. In practice, sync messages carrying the current playhead position of the master device p_{master} may take the time of Δt_0 to arrive at an auxiliary device, by which time the master has already a new playhead position of $p_{master} + \Delta t_0$. Moreover, when the sync controller and the playback management function instruct the media player to re-synchronize by adjusting its playhead position, the media player must request new data range from the content server which often leads to an additional buffering delay of Δt_1 determined by the available bandwidth and the buffer size/buffering strategy at the end device. Without the help of a synchronization framework, the streams at auxiliary devices may lag behind the master for $\Delta t_0 + \Delta t_1$, which could be in the scale of hundreds of milliseconds to tens of seconds. To mitigate such delay impact, the sync signaling module monitors the round trip time of the sync messages exchanged between user devices and the sync server and estimates the network delay Δt_0. This is similar to the design principle behind NTP but executed and maintained natively. The measurements are carried out by "piggybacking" probing signals on top of the sync heartbeat signals. Because the measurement is conducted on sync messages directly (rather than using separated NTP probing messages), the mechanism is more efficient for interactive media applications, having very little overhead. Together with the playback and buffer management module, the sync signaling function also statistically measures Δt_1, the delay between an order being sent from the playback management function and the media player completing the execution.

3) Sync controller: The sync controller is ultimately the decision maker of the framework. Being active on each auxiliary device, it monitors the level of playback non-synchronicity with the master device, and derives from that the timing and strategy for the re-synchronization process. The sync controller currently employs two re-synchronization approaches, namely Adaptive Media Playout (AMP) and Predictive Playhead Projection (PPP), which are selectively enabled for the best results as perceived by human. Table I defines the metrics and functions used by the sync controller. Given a current lag, the controller chooses to increase playback speed temporarily (AMP), and balances the choice of speed against the duration of the adjustment, such that the QoE impact is minimized. Only under extreme conditions does it perform a discrete jump (PPP) to perform the bulk of the work, with AMP for a final correction.

The impact of non-synchronicity is denoted as $I_{\text{non-sync}}$, a function of the non-synchronicity s measured by $p_{Master} + \Delta t_0 - p_{Aux}$. To reduce s by Δs, the AMP approach temporarily changes the original playback rate v of the auxiliary media stream to a new v'. The change of playback rate G is defined as $\frac{v'}{v}$. It would take the duration of $T = \frac{\Delta s}{|G-1|}$ for the auxiliary media stream to be perceptually in-sync with the master stream. Given Δs and v, T is inversely proportional to $|G-1|$. Therefore, a more radical change in playback rate (i.e., a higher value of $|G-1|$) could reduce the non-synchronicity quicker and therefore result in lower accumulative impact (i.e., $C_{\text{non-sync}}$) to the user experience. However, the change made on the playback rate can be noticeable or even annoying to the user. The accumulative re-synchronization impact $C_{\text{re-sync}}$ is contributed by G and T. Given Δs, different combinations of G and T can be selected. Applying $G = 1.2$ for $T = 8$ seconds and $G = 1.8$ for $T = 2$ seconds would both help reducing the non-synchronicity by 1.6 seconds though their impact to the QoE can be significantly different. Finding an optimal solution for a given Δs requires quantitative modelling of the impact from G and T which are believed to be non-linear in psychological scales. In practice, there might also be constraints on G. The execution of G by the user device is determined by the buffer occupancy and the network bandwidth. G_{limit} defines the upper limit of G that the device can possibly perform. The goal of the sync controller is to name a re-synchronization strategy that leads to minimal total impact between $C_{\text{non-sync}}$ and $C_{\text{re-sync}}$ (denoted as J).

Symbol	Description
s	Non-synchronicity between an auxiliary device and the master device.
S_L	The level of s when the non-synchronicity becomes perceivable by human.
S_H	The level of s when the non-synchronicity is too severe for the AMP approach to rectify without taking too much time or causing highly detrimental distortions.
Δs	The amount of non-synchronicity to reduce. Often it's equal to the non-synchronicity.
v	The original (native) playback rate.
v'	The adjusted playback rate during AMP.
G	The gain of the playback rate. $G = \frac{v'}{v}$.
T	The duration of the AMP re-synchronization process with G in effect. $T = \frac{\Delta s}{\lvert G-1 \rvert}$.
G_{limit}	The maximum playback gain that can be supported by the device and network.
T_{limit}	The maximum use of time for re-synchronization.
$I_{\text{non-sync}}$	The perceptual impact of non-synchronicity.
$C_{\text{non-sync}}$	The accumulative impact of non-synchronicity.
$I_{\text{re-sync}}$	The perceptual impact of re-sync process.
$C_{\text{re-sync}}$	The accumulative impact of re-sync process.
J	The overall impact of non-synchronicity and re-synchronization to the user.

TABLE I. SYNC METRICS AND FUNCTIONS

While the AMP approach can be exploited to smoothly re-synchronize media streams, it might not be suitable when Δs reaches a certain threshold. Predictive playhead projection (PPP) is an approach that directly manipulates the playhead position of auxiliary streams (i.e., skipping). Although frame skipping is perceptually detrimental to the user experience, it is more efficient to rectify severe Δs. When a media player skips to a non-buffered point in the media stream, a further buffering delay Δt_1 is introduced. This will then cause the non-synchronicity of Δt_1 following the skipping. By monitoring the available bandwidth and buffer status, PPP estimates the Δt_1 (as $E(\Delta t_1)$) and pre-emptively appends this as an additional adjustment to the new playhead position. Any small residual (the difference between the observed and expected Δt_1 (i.e., $|O(\Delta t_1) - E(\Delta t_1)|$) will be subsequently corrected by AMP. The IMSync re-synchronization algorithm is summarized in Algorithm 1.

B. Sync server

The sync server bridges the associated devices in an N-Screens cloud so that application configurations and sync timing information can be efficiently exchanged. An alternative design is to use a self-organizing overlay to carry the function of a sync server [18]. We recognize the distinctive benefits of each design and favour the presence of a sync server function

if $s < S_L$ **then**
  Do nothing; /* s not perceivable. */
else if $S_L \leqslant s < S_H$ **then**
  /* Use AMP approach. */
  Derive G_{limit} through bandwidth and buffer monitoring. [G,T]=$\mathcal{J}_{QoE\_IMPACT}(\Delta s, G_{\text{limit}})$;
  /* Find the optimal AMP re-synchronization solution using a QoE impact function. */
  Adjust playback rate using the new G and T;
else
  /* Use PPP approach. */
  Estimate the buffering delay Δt_1;
  Instruct media player to skip to the playhead position of $p_{Master} + \Delta t_0 + \Delta t_1$;
  AMP eliminates any residual non-synchronicity;
end

Algorithm 1: IMSync re-synchronization algorithm

because of its relatively minimal network and application level run-time overheads and software requirements at user clients. The media application may specify a manifest that defines the media tracks (such as video, audio, ambient light, etc.) available for baseline and enhancements to the media experience. The sync server may host such a manifest to orchestrate the playback of the media tracks. The timing information is carried by the periodic heartbeat messages (as part of the sync signaling) initiated by the sync server and forwarded to the sync controller of all connected devices using the most efficient connection type possible. The framework also allows the sync server to run on a user device so private device clouds can be established in a local environment.

C. Full-reference sync measurement

One of the challenges of designing and evaluating a media synchronization framework is to accurately measure the absolute non-synchronicity between media streams under the influence of network latency, bandwidth constraint, and device capacity. We choose audio as the reference signal and customized a small FR measurement device that simultaneously captures the rendered audio outputs from two user devices. We take the audio sampling in the rate of 10,000 samples per second from both sources and measure the cross-correlation between samples. Conventionally, cross-correlation is calculated based on the entire range of data from the sampling process, and the time offset from 0 that gives the peak of cross-correlation defines the inter-stream "lag" (i.e., non-synchronicity). However, the granularity of the results from such measurements is too coarse to capture the change of non-synchronicity influenced by the re-sychronization methods. Hence, we designed an *expandable moving slice* algorithm to better capture the intensity and variation of non-synchronicity.

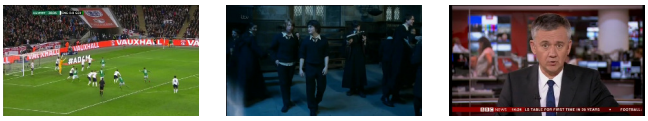
The algorithm begins by taking a slice in the size of 100 samples from both audio sources to calculate cross-correlation. A small slice size gives finer measurement, but no matches between two slices can be found if the slice size is smaller than the non-synchronicity. With 10,000 samples per second and a slice size of 100 samples, each slice covers a duration of 10 ms. Therefore, an analysis based on 100-sample slices will detect non-synchronicity below 10 ms. If a correlation over a pre-defined threshold is found, a measurement is registered and the calculation will move on to the next slice. Otherwise, we increase the slice size by 100 samples to expand the search range, until a result is found.

IV. Modeling the Human Factor

The change in playback speed yields two perceived effects: re-synchronization (change of speed) and non-synchronicity. This section introduces experiments that serve to measure these two effects independently, and allow us to produce a combined model to capture the overall human perception.

A. Perception of non-synchronicity

Existing studies on non-synchronicity focus on the measurements between tracks of a single media stream or the lags between media streams at different locations [16]. We focus on studying synchronously played multiple streams at a shared location, which reflects our use-case scenario. The modeling of human perception helps us determine the optimal timing and strategy of the re-synchronization process. The ultimate means to construct the impact model is through subjective user experiments. Our test environment is built with one video source (playing video only) accompanied by two audio sources. Both audio sources have nearly identical distance to the test participants, therefore any latency caused by the speed of sound is negligible.



(a) Soccer (b) Film (c) News

Fig. 2. Audio-visual clips for user experiments

We selected three representative video clips of 20 seconds long for the experiment (Figure 2). The *soccer* clip is sliced from a *FIFA Worldcup 2014* match with audio commentary. The *news* clip shows a short news item on *BBC NEWS*. The *film* clip is a scene of a *Harry Potter* film with multiple characters engaging in a conversation. We prepared test materials from all three clips with audio source 2 lagging behind audio source 1 (which is in perfect sync with the video source) for 20 ms, 40 ms, 60 ms, 80 ms, 100 ms and 160 ms. 16 participants rated the non-synchronicity in the form of ACR-HR (absolute category rating with hidden reference) using the ITU 5-point rating in the impairment scale (*5 - Imperceptible*; *4 - Perceptible but not annoying*; *3 - Slightly annoying*; *2 - Annoying*; *1 - Very annoying*).

Scores given by each user are re-scaled to range $[1, 5]$. Figures 3(a), 3(b) and 3(c) show the mean opinion score (MOS) for each clip. While they all exhibit polynomial-like distribution, the non-synchronicity on the soccer clip seems to be far more tolerable compared with the same test conditions applied to film and news clips. The MOS of the experiments on soccer does not drop below 2 ("annoying") even when the audio source 2 manifests a 160 ms lag, which is considered as "very annoying" by many participants on other clips. From a short post-experiment user interview, we learned that the echo-like effect caused by the non-synchronicity between multiple

audio sources resonates with the experience in a large stadium. Because of this specific context, non-synchronicity on soccer clip becomes more acceptable. Some users also suggested that their attention was more on the actions in the soccer scene rather than what the commentator had to say. Accommodating the influence of content characteristics in modeling is an interesting research topic to be part of our future work, though the feasibility of the model could be affected by increased run-time complexity. Judging from the overall experimental results in Figure 3(d), a non-synchronicity of 20 ms is barely noticeable. When the level reaches 40 ms, it is perceivable by some users though not considered as annoying. From 60 ms, the non-synchronicity becomes annoying.

To generalize our findings, we derived the overall fitting function from user scores of all three clips as below:

$$U_{\text{non-sync}}(s) = a_U s^2 - b_U s + c_U \tag{1}$$

With $a_U = 0.0002$, $b_U = -0.0495$, and $c_U = 5.5174$, the fitting has a goodness-of-fit of $R^2 = 0.93$ to the observed data. It is also noticed that, when the impact function is modelled on either of the three clips separately, R^2 yields 0.97, 0.99 and 0.96, which reflects our previous conclusion that adopting content characteristics could potentially further improve the modeling of non-synchronicty.

The corresponding impact function (how much the user scores deviate from $5 - Imperceptible$) is given below with $a_I = -0.0002$, $b_I = 0.0495$, and $c_I = -0.5174$

$$I_{\text{non-sync}}(s) = 5 - U_{\text{non-sync}}(s) = a_I s^2 - b_I s + c_I \tag{2}$$

In practice, when s reaches a certain level s_0 that is perceivable by the user, re-synchronization mechanisms reduce non-synchronicity to a level s_1 that is unnoticeable by the user. We define the amount to catch up as $\Delta s = s_0 - s_1$:

Assuming the catch-up process will linearly reduce the non-synchronicity and it takes a certain amount of time T for the process to complete, the instantaneous non-synchronicity during the catch-up from time $t = 0$ to $t = T$ is $s(t) = s_0 - \frac{t}{T}\Delta s$. First, we expand (2) by substituting $s(t)$:

$$\begin{aligned} I_{\text{non-sync}}(t) &= a_I \left(s_0 - \frac{t}{T}\Delta s\right)^2 + b_I \left(s_0 - \frac{t}{T}\Delta s\right) + c_I \\ &= a_I \left(\frac{\Delta s}{T}\right)^2 t^2 - \left(2as_0 \frac{\Delta s}{T} + b_I \frac{\Delta s}{T}\right) t + a_I s_0^2 + b_I s_0 + c_I \end{aligned}$$

The non-synchronicity experienced by the user is then an accumulative effect of $I_{\text{non-sync}}(t)$ characterized by s_0, s_1, and T. We consider the accumulation linear in time scale and yield the accumulative impact factor $C_{\text{non-sync}}$.

$$\begin{aligned} &\int I_{\text{non-sync}}(t)\mathrm{d}t = a_I \left(\frac{\Delta s}{T}\right)^2 \frac{t^3}{3} \\ &- \left(2a_I s_0 \frac{\Delta s}{T} + b_I \frac{\Delta s}{T}\right) \frac{t^2}{2} + (a_I s_0^2 + b_I s_0 + c_I)t + K \end{aligned} \tag{3}$$

We now integrate $I_{\text{non-sync}}(t)$ with our specified limits:

$$C_{\text{non-sync}} = \int_{t=0}^{T} I_{\text{non-sync}}(t)\mathrm{d}t \tag{4}$$

$$= \frac{\left[2a_I \Delta s^3 - 6a_I s_0 \Delta s^2 - 3b_I \Delta s^2 + 6a_I s_0^2 \Delta s + 6b_I s_0 \Delta s + 6c_I \Delta s\right]}{6|G-1|} \tag{5}$$

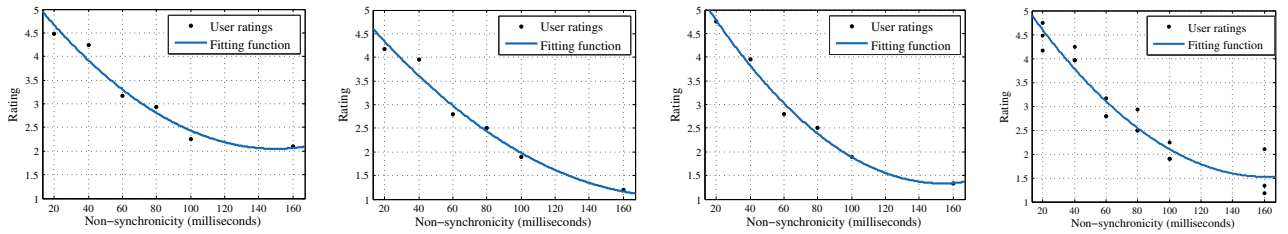
With s_0 and s_1 defined, $C_{\text{non-sync}}$ is directly proportional to T which suggests that the quicker we bring media streams back in sync, the less perceivable impact there will be to the user. However the processes of re-synchronization can also lead to new distortions to the media, which sometimes can be more detrimental than the non-synchronicity itself.

B. Perception of re-synchronization

AMP is a rate control mechanism that has been widely used to achieve smooth media playback or to harmonize buffer level via the dynamic adjustments of the media playout rate to mitigate the perceptual impact of network impairments. Li et al. defined multiple thresholds for the playout controller to start playback and dynamically adjust the playout rate based on the "buffer fullness" [13]. Learned from "informal tests", Kalman et al. concludes that the change of playback rate of up to 25% is often unnoticeable and a change of up to 50% is sometimes acceptable [11]. The threshold of 25% has been adopted by a number of previous works as the guidance for the maximum playback rate variation [15], [22]. Li et al. uses a "simple linear function" to model the "slowdown cost" due to playing slower than the original playback rate [14]. A number of studies (e.g., [23]) also exploit a quadratic impact function initially proposed in [10], though the function does not seem to have been derived from subjective experiment. It is then uncertain whether the values given by the impact function are in psychological scales for QoE optimization. Li et al. [14] also recognized the influence of content characteristics (visual and acoustic features) on the perception of AMP. Rainer et al. evaluated the impact of playout variations on the QoE by adopting a crowdsourcing approach [19].

There are three main issues with such abstract rules and functions found in existing work. Firstly, they do not quantitatively capture the impact of AMP as perceived by users. Hence the re-sync process would not be able to optimize for the user experience. Secondly, the modeling on the impact of the duration of AMP, which is unlikely to be linear, is missing. A 30% increment in playback rate for 1 second can be imperceptible, while it may simply take users a bit longer to start noticing the playback distortion or even find it annoying. Finally, there is a lack of systematic study on the joint perceptual impact of non-synchronicity and re-synchronization to optimize the balance between the two.

To fill the gap in this research field, we carried out further user experiments to quantitatively model the impact of AMP-based re-synchronization by the change of playback rate $G = \frac{v'}{v}$ and the effective duration T of AMP. This effectively contours the operational range of AMP. We reuse the three representative clips shown in Figure 2 to generate test videos.



(a) Soccer (b) Film (c) News (d) Overall

Fig. 3. Aggregated ratings on non-synchronicity

Each test video has one test condition applied to it which is a combination of G and T. The selection of G is 1.1, 1.2, 1.4, 1.8, and 2.0 which maps to the playback rate v' of 33, 36, 42, 54, and 60 fps for our test videos with the same native rate v of 30fps. The durations T of 1, 2, 4, and 8 seconds are selected. The test conditions are then applied to the reference videos. The playback starts at its native rate v; switches to v' at t_0; and finally goes back to v at t_1. We avoid the first 5 seconds of the clip, so t_0>5. Overall, 60 test videos are generated and rated by 16 participants.

$$C_{\text{re-sync}}(G,T) = p_{00} + p_{10}T + p_{01}G + p_{20}T^2 + p_{11}TG + p_{02}G^2 + p_{21}T^2G + p_{12}TG^2 + p_{03}G^3 \quad (6)$$

$$C_{\text{re-sync}}(G) = p_{00} + p_{10}\frac{\Delta s}{G-1} + p_{01}\frac{\Delta s}{G-1} + p_{20}(\frac{\Delta s}{G-1})^2 + p_{11}\frac{\Delta s}{G-1}G + p_{02}G^2 + p_{21}(\frac{\Delta s}{G-1})^2G + p_{12}\frac{\Delta s}{G-1}G^2 + p_{03}G^3 \quad (7)$$

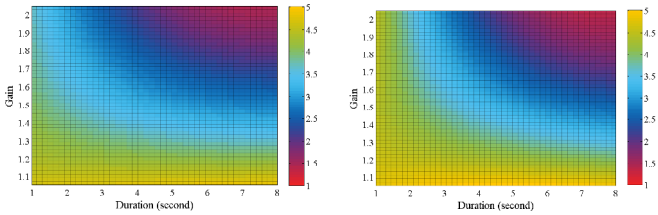
The impact metric derived from user scores is modelled using a two-variable polynomial function (Equation 6). We use a second-order fitting option for the duration T and a third-order fitting option for the gain G to achieve the optimal balance between the performance and the complexity of the fitting function. We also investigated models with higher order coefficients. However they prove to be overly complex and generally cause over-fitting.

Since $T = \frac{\Delta s}{G-1}$, the function can be simplified into a single-variable polynomial in G (Equation 7).

The fitted coefficients are shown in Table II. Overall, function $C_{\text{re-sync}}(G)$ exhibits the goodness-of-fit of $R^2 = 0.96$ and $RMSE = 0.24$. The fitting process is also carried out on test results of three clips separately which exhibit very similar measures of the goodness-of-fit.

Coefficient	Fitted value	Coefficient	Fitted value
p_{00}	-3.542	p_{02}	-3.806
p_{10}	-1.259	p_{21}	-0.053
p_{01}	6.995	p_{12}	-0.1694
p_{20}	0.05653	p_{03}	0.6828
p_{11}	1.341		

TABLE II. FITTED VALUES OF COEFFICIENTS



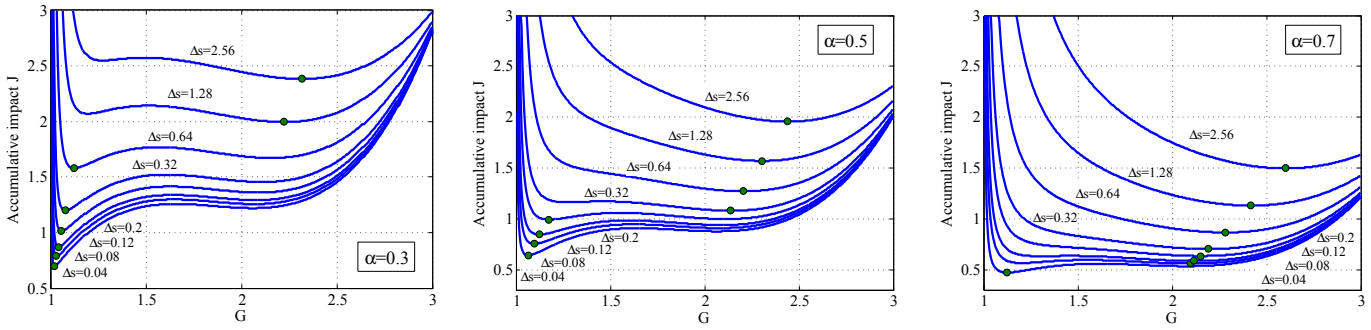
(a) Main (b) Soccer

Fig. 4. Colormap of user scores

We also plotted the colormap to demonstrate the user opinion scores of AMP-based re-synchronization with respect to the combinations of G and T (Figure 4(a)). Note that both the intensity and the duration of the playback rate adjustment have non-linear impact to the perception of re-synchronization. Overall, when G is below 1.2, users are unlikely to notice any anomaly even when the duration of it is as high as 8 seconds. In fact, the combination of 1.2 gain and 8 seconds duration results in 48 additional frames being played for a 30 fps content, allowing any auxiliary stream to catch up by 1.6 seconds of playback time. Using a higher G such as 1.8 could also yield the same results, though its impact starts to become annoying when the duration T exceeds 2 seconds. The re-synchronization impact on soccer clip is shown in Figure 4(b) as a comparison to the figure based on all experimental results. We learn that content characteristics do influence the perceptual impact of AMP re-synchronization. Temporal change of playback rate is less noticeable on soccer than on other clips. Users find a 2-second long doubling of playback rate "perceivable but not annoying". The user interviews suggest that the high motion and complexity of the soccer scene lead to a "masking effect", which affects the perception of the playback rate change.

C. Optimization

The modelling of the non-synchronicity impact $C_{\text{non-sync}}$ (Equation 5) and the re-synchronization impact $C_{\text{re-sync}}$ (Equation 6) enables us to identify the optimal solutions to adjust playback rate with the minimal overall impact J to user experience. We normalize and rescale both impact functions into $[0, 4]$ before combining them using the weighted-sum method for the global optimization (Equation 8). The weight coefficient α defines the balance between non-synchronicity impact and resynchronization impact when searching for the optimal solution using function J. The IMSync framework is flexible in tuning the AMP solution for applications/users that are more affected by non-synchronicity (with $\alpha > 0.5$) or more



(a) $\alpha = 0.3$ (b) $\alpha = 0.5$ (c) $\alpha = 0.7$

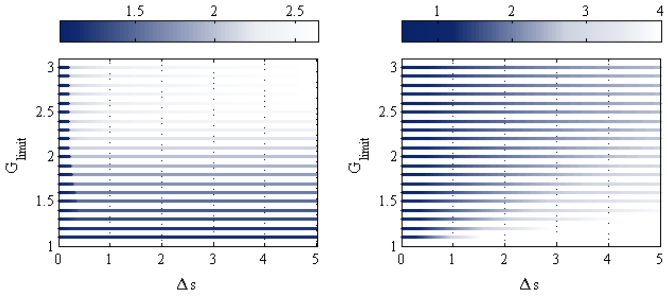
Fig. 5. Impact curve J configured using different values of weight coefficient

susceptible to the change of playback rate (with $\alpha < 0.5$). Given Δs, G_{limit}, and α, J is a function of G.

$$J = \alpha\, C'_{\text{non-sync}} + (1-\alpha)\, C'_{\text{re-sync}}, with\ 0 \leq \alpha \leq 1 \quad (8)$$

Figure 5 shows the overall impact functions of AMP re-synchronization for different levels of non-synchronicity and device/network capabilities G_{limit}. Figure 5(b) represents the case when $C_{\text{non-sync}}$ and $C_{\text{re-sync}}$ are valued equally ($\alpha = 0.5$). The figure clearly manifests the joint impact of $C_{\text{non-sync}}$ and $C_{\text{re-sync}}$. When the non-synchronicity is relatively low (less than 0.2 seconds), the best solution with minimum total accumulative perceptual impact can be found using a small playback gain G allowing a mild Δs to be rectified without causing high re-synchronization distortion to the application. However, when Δs is above 0.2 seconds, $C_{\text{non-sync}}$ may accumulate large impact over time. In this case, a more intensive adjustment to the playback rate is required to greatly reduce the non-synchronicity quickly with a small cost in re-synchronization distortion.

The weight coefficient α has great influence on the impact function J as well as the optimal configurations for the AMP re-synchronization process. As depicted in Figure 5(a), with more weight on the re-synchronization impact ($\alpha = 0.3$), the framework favours mild playback gain G until the non-synchronicity to catch up reaches the level of 1.28 seconds (compared with 0.32 seconds when $\alpha = 0.5$). For applications that are more prone to the level of non-synchronicity than the change of playback rate, α can be set above 0.5 to trigger the framework to use more radical approach. Figure 5(c) gives an example of α being set to 0.7 where the framework uses a high G of greater than 2 for 80ms of Δs.



(a) Optimal G (b) Impact of the optimal G

Fig. 6. Optimal G for given Δs and G_{limit}

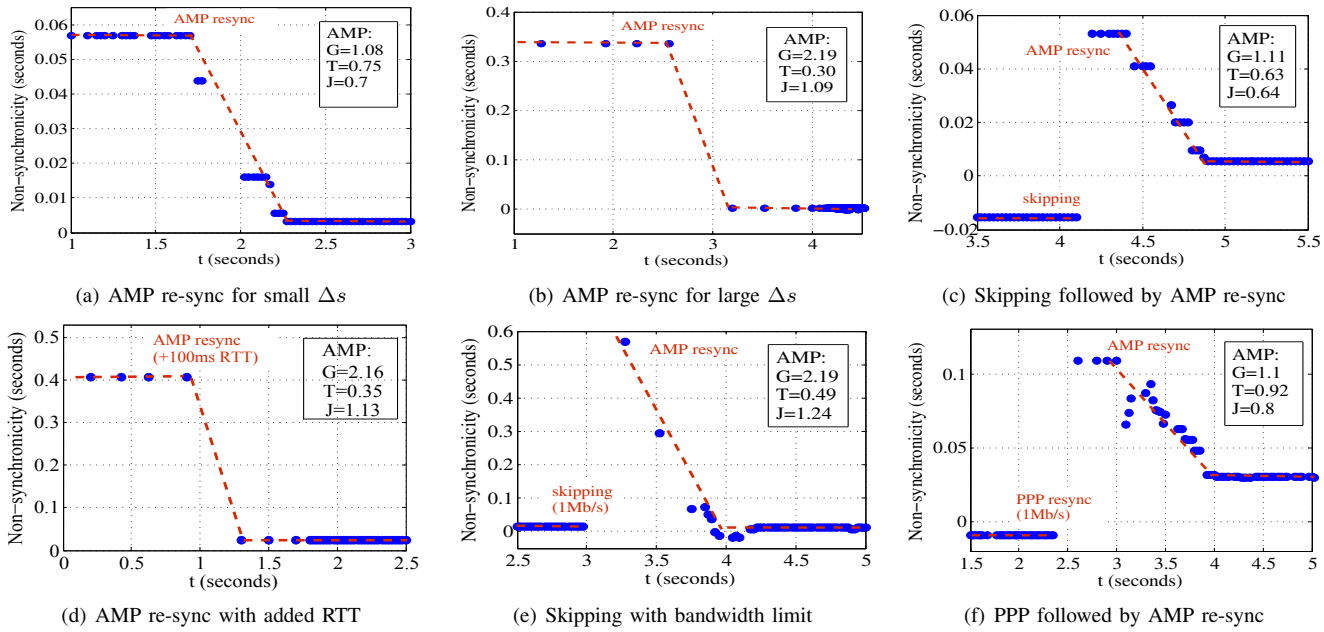
In order to automate the optimization process to derive the optimal AMP solution for a given Δs and G_{limit}, the sync controller dynamically calculates the value of G that minimizes Equation 8. The mathematical approaches to search for the minimal value on our impact function are not limited by the capabilities of the playback device. In production environments, the optimal Gs can be pre-computed based on intervals of Δs and G_{limit}. This would greatly improve the run-time efficiency of the synchronization process.

The optimal Gs on the impact curve J for different Δs and a G_{limit} are marked in Figure 5. We also take samples of Δs in the range of (0,3] and G_{limit} in the range of (1,3] to study the performance of the framework when the non-synchronicity and device/network limit varies. Figure 6(a) gives the optimal G while their corresponding total impact is shown in Figure 6(b). The visible leap around $\Delta s = 0.2$ when $G_{\text{limit}} > 2$ in Figure 6(a) reflects the shift of minimal impact point in Figure 5. When $G_{\text{limit}} < 2$, IMSync is forced to opt for a lower G which leads to higher impact. Figure 6(b) gives an overview of the effective range of the AMP re-synchronization. In general, AMP is most suitable for non-synchronicity of low degree when the overall impact is below 2 (at which point the users find it "perceptible" or "slightly annoying"). This is also determined by the user device and the network. Re-synchronization can be less detrimental to user experience on devices connected via broadband networks. Figure 6(b) also suggests the points when the AMP-based approach is comparable to the more straightforward PPP-based re-synchronization. For instance, when $\Delta s = 2$ and $G_{\text{limit}} = 1.5$, skipping may be preferable to having a 4-second long annoying playback rate change.

V. Implementation and Experiments

The IMSync framework has been implemented using open web technologies such as Javascript. Any user device that supports Javascript can join an N-Screens device cloud without additional applications. A customized Node.js server operates as the sync server handling device discovery and sync signalling as specified in the framework design.

In order to evaluate the framework, we set up a testbed environment with multiple user devices, a sync server that



(a) AMP re-sync for small Δs

(b) AMP re-sync for large Δs

(c) Skipping followed by AMP re-sync

(d) AMP re-sync with added RTT

(e) Skipping with bandwidth limit

(f) PPP followed by AMP re-sync

Fig. 7. Experimental results

also hosts the Javascript libraries, a media server which serves media content, a full-reference sync measurement device, and emulators for networks of different properties. We also use a web admin interface to monitor sync messages exchanged between devices and their player status (such as playhead position, playback rate and buffer level). The interface provides real-time measurements of network statistics on all devices and control interfaces for experimentation. The framework is configured to weight the impact of non-synchronicity and re-synchronization equally ($\alpha = 0.5$).

We used the full-reference sync measurement device to capture the operations of the framework (Figure 7). Every marker presents a point of valid measurement. A positive value of non-synchronicity denotes the auxiliary stream being behind the master stream. We also use dash lines to plot the trends of the measurements. Due to the nature of the sampling method, the measurement tool will yield fewer results when the non-synchronicity is high, though the accuracy of measurements is not affected. The synchronicity during the change of playback rate and skipping is very difficult to capture. The results given during these transition periods are, however, still valuable in understanding the operations of the IMSync framework. Based on the user study results shown in Figure 3, we define the threshold of 30 ms (just below the display time of one video frame for a 30 fps video content) as the measure of whether a pair of media streams in the same location are "in-sync".

The first group of tests are performed with no network emulation. The available bandwidth is 100 Mb/s and the round trip time between user devices and the sync server is less than 10 ms. This resembles the scenario when an application is running locally with all devices joining a local network. We start the playback of a media stream on all devices with the synchronization framework turned off and control the playhead positions of the auxiliary stream to be around 57 ms behind the master stream. We then activate the framework which immediately detects the non-synchronicity on the auxiliary device and uses the AMP method to re-synchronize the media streams by slightly increasing the playback rate ($G = 1.08$) for 0.75 seconds. With the impact J of just 0.7, users are very unlikely to notice any distortion from the time the framework is enabled. Figure 7(a) suggests that the media streams are less than 5 ms apart after the re-synchronization process.

In the second test, we greatly increase the initial non-synchronicity to around 350ms. Using the impact function, the framework instructs a short surge ($T = 0.3$) of high playback gain ($G = 2.19$) which brings the non-synchronicity back to around 1 ms (Figure 7(b)). The impact of operation is increased to $J = 1.09$, which suggests that statistically users will experience a very short perceivable but not annoying distortion. The third test studies how the IMSync framework reacts to media events. We start the test with all stream in-sync ($s \cong 18ms$), then commit a skip operation to a point around 15 minutes further into the video on the master stream (which is a common user operation). Because the non-synchronicity (around 15 minutes) is beyond the range of AMP, IMSync instructs the auxiliary stream to skip (Figure 7(c)) whilst factoring in the signaling delay. Due to the small buffering delay, the auxiliary stream ends up to be over 50 ms behind the master stream. This is immediately followed by AMP which closes up the gap with minimal impact ($J = 0.64$) in less than a second (Figure 7(c)).

The second group of tests evaluates the framework's performance when network delays and bandwidth affect sync signaling and media buffering. We apply a 100ms round-trip delay to the link of the auxiliary device and enable the AMP on 400ms of non-synchronicity. The results demonstrate that the framework detects the additional network latency and adjusts the playback rate change to close up the lag between media streams to merely 3 ms. We then limit the available bandwidth of the auxiliary device to 1 Mb/s and repeat the skipping test. As a result, the limit on the buffering throughput increases the non-synchronicity after the skipping tenfold to around 600

ms. It then takes AMP to apply a high playback gain of $G = 2.19$ with impact of $J = 1.24$ to adjust the media stream (Figure 7(e)). The PPP approach is brought in to estimate the buffer delay based on 1) the moving average of previous skip events, and 2) out-of-band bandwidth monitoring using probing packets. The estimated buffer delay is then employed to skip the auxiliary stream to a projected playhead position further into the future so that the playback deficit can be greatly reduced when the skip event completes. Figure 7(f) gives an example of how the bandwidth/buffering delay measurement could improve the synchronization. With the same set-up used for Figure 7(e), the PPP-based approach takes the measurement of around 500 ms of buffering delay based on statistics from previous events, and reduces the non-synchronicity after the skip to just above 100 ms, which has much less impact ($J = 0.8$) to catch up further by AMP.

VI. Conclusion

Orchestrating multiple media streams on heterogeneous user devices to facilitate new media experiences is a very challenging task. The paper contributes to this topic with the design and implementation of an open inter-stream synchronization framework: IMSync. The framework is unique in providing the optimized re-sychronization strategies with minimal perceptual impact to the user using a comprehensive QoE impact model, while incorporating an efficient sync-signaling mechanism and functional modules to interact with media engines. We implement the framework using web technologies and evaluate its performance using a tailor-made testbed. The purpose of IMSync is not only achieving absolute inter-stream synchronicity, but also offering a foundation for new media applications and user experiences by programming the temporal attributes of associated media objects over multiple devices.

Acknowledgement

This work was supported in part by European Commission FP7 grants 318343 (STEER) and 603662 (FI-Content2). This work would not have been possible without the technical contribution of Jamie Jellicoe, Lyndon Fawcett, and Jamie Bird.

References

[1] A. Brown, M. Evans, C. Jay, M. Glancy, R. Jones, and S. Harper. Hci over multiple screens. In *CHI'14 Extended Abstracts on Human Factors in Computing Systems*, pages 665–674. ACM, 2014.

[2] P. Cesar, D. C. Bulterman, and A. Jansen. Usages of the secondary screen in an interactive television environment: Control, enrich, share, and transfer television content. In *Changing television environments*, pages 168–177. Springer, 2008.

[3] J. Cheer, S. J. Elliott, Y. Kim, and J.-W. Choi. Practical implementation of personal audio in a mobile device. *Journal of the Audio Engineering Society*, 61(5):290–300, 2013.

[4] M. Chen. A low-latency lip-synchronized videoconferencing system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 465–471, New York, NY, USA, 2003. ACM.

[5] J. Chmielewski. Device-independent architecture for ubiquitous applications. *Personal and Ubiquitous Computing*, 18(2):481–488, 2014.

[6] Exacttarget. 2014 mobile behavior report - combining mobile device tracking and consumer survey data to build a powerful mobile strategy. 2014.

[7] D. Geerts, R. Leenheer, and D. De Grooff. In front of and behind the second screen: Viewer and producer perspectives on a companion app. In *Proceedings of the ACM International Conference on Interactive Experience of Television and Online Video (TVX2014)*, 2014.

[8] H. Hu, J. Huang, H. Zhao, Y. Wen, C. W. Chen, and T.-S. Chua. Social tv analytics: a novel paradigm to transform tv watching experience. In *Proceedings of the 5th ACM Multimedia Systems Conference*, pages 172–175. ACM, 2014.

[9] B. R. Jones, H. Benko, E. Ofek, and A. D. Wilson. Illumiroom: peripheral projected illusions for interactive experiences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 869–878. ACM, 2013.

[10] M. Kalman, E. Steinbach, and B. Girod. Rate-distortion optimized video streaming with adaptive playout. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, volume 3, pages III–189. IEEE, 2002.

[11] M. Kalman, E. Steinbach, and B. Girod. Adaptive media playout for low-delay video streaming over error-prone channels. 14(6):841–851, 2004.

[12] H. Kim, S. Lee, J.-W. Choi, H. Bae, J. Lee, J. Song, and I. Shin. Mobile maestro: enabling immersive multi-speaker audio applications on commodity mobile devices. In *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pages 277–288. ACM, 2014.

[13] M. Li, T.-W. Lin, and S.-H. Cheng. Arrival process-controlled adaptive media playout with multiple thresholds for video streaming. *Multimedia systems*, 18(5):391–407, 2012.

[14] Y. Li, A. Markopoulou, J. Apostolopoulos, and N. Bambos. Content-aware playout and packet scheduling for video streaming over wireless links. 10(5):885–895, 2008.

[15] M. Montagud, F. Boronat, and H. Stokking. Design and simulation of a distributed control scheme for inter-destination media synchronization. In *Advanced Information Networking and Applications (AINA), 2013 IEEE 27th International Conference on*, pages 937–944, 2013.

[16] M. Montagud, F. Boronat, H. Stokking, and R. van Brandenburg. Inter-destination multimedia synchronization: schemes, use cases and standardization. *Multimedia Systems*, 2012.

[17] Nielsen. The u.s. digital consumer report. *http://www.nielsen.com/us/en/insights/reports/2014/the-us-digital-consumer-report.html*.

[18] B. Rainer and C. Timmerer. Self-Organized Inter-Destination Multimedia Synchronization for Adaptive Media Streaming. In *Proceedings of the ACM International Conference on Multimedia*, pages 327–336. ACM, 2014.

[19] B. Rainer and C. Timmerer. A subjective evaluation using crowdsourcing of adaptive media playout utilizing audio-visual content features. In *Network Operations and Management Symposium (NOMS), 2014 IEEE*, pages 1–7. IEEE, 2014.

[20] M. Sarkis, C. Concolato, and J.-C. Dufourd. The virtual splitter: refactoring web applications for themultiscreen environment. In *Proceedings of the 2014 ACM symposium on Document engineering*, pages 139–142. ACM, 2014.

[21] R. Steinmetz. Human perception of jitter and media synchronization. 14(1):61–72, 1996.

[22] Y.-F. Su, Y.-H. Yang, M.-T. Lu, and H.-H. Chen. Smooth control of adaptive media playout for video streaming. *Multimedia, IEEE Transactions on*, 11(7):1331–1339, 2009.

[23] E. Tan and C. T. Chou. A frame rate optimization framework for improving continuity in video streaming. *Multimedia, IEEE Transactions on*, 14(3):910–922, 2012.

[24] G. Thomas, P. Mills, P. Debenham, and A. Sheikh. Surround Video. *White Paper WHP 208, http://downloads.bbc.co.uk/rd/pubs/whp/whp-pdf-files/WHP208.pdf*.

[25] R.-D. Vatavu and M. Mancas. Visual attention measures for multi-screen tv. In *Proceedings of the 2014 ACM international conference on Interactive experiences for TV and online video*. ACM, 2014.