Pers Ubiquit Comput
DOI 10.1007/s00779-013-0701-z

ORIGINAL ARTICLE

Context-dependent software solutions to handle video synchronization and delay in collaborative live mobile video production

Mudassar Ahmad Mughal · Oskar Juhlin

Received: 21 November 2011 / Accepted: 14 August 2012

Abstract The advent of modern mobile phones, 3G networks, and live video streaming has made it possible to broadcast live video from mobile devices. This is now giving rise to a new class of applications which enable mobile collaborative live video production, in which groups of amateurs work together to provide a rich broadcast of events. We focus on new and expected synchronization problems that arise in these more complex systems when broadcasting live events because of the delays that often occur in streaming over internet and mobile networks. The problem has been investigated by acquiring initial user feedback, as well as conducting technical delay measurements of two examples of such systems and relating them to existing literature. We identified two types of technical problems which affect the mixing of the streams, namely the difference in delay in multiple streams, a.k.a. asynchrony among streams, and the delay between the event itself and its presentation in the mixer. These problems affect the mixing in various ways depending on whether or not the director has visual access to the unmediated event. This knowledge has then been used to inform the conceptualization of identifiable ways of handling delays and synchronization. We suggest the introduction of a software feature providing context-dependent delay, in which these requirements can be balanced differently to fit specific contexts of use. We specifically address the different types of mixing which occurs when the director, or mixer, only has access to the topic through the mobile media ("out of view"), as well as mixing in a context in which the topic also is physically present ("in-view") in front of the mixer.

Keywords Webcasting · Live video · Synchronization · Asynchrony delay · Mixing · Collaboration

Abbreviations

HD	High definition
IBS	Instant broadcasting system
MVM	Mobile vision mixer
NTP	Network time protocol

M. A. Mughal (✉)
Mobile Life @ Stockholm University, Box 1197,
164 26 Kista, Sweden
e-mail: mamughal@dsv.su.se; mudassir@mobilelifecentre.org

O. Juhlin
Mobile Life @ Interactive Institute, Box 1197,
164 26 Kista, Sweden
e-mail: oskar@mobilelifecentre.org

1 Introduction

In recent years, the availability of high-speed mobile networks together with advanced mobile phones with cameras has given rise to a new generation of mobile live video streaming services that, in turn, has opened a new avenue for live mobile video production. Most such services and applications today are limited to a single mobile camera as a source for video production. Lately, the demand for more extended resources for amateur storytelling, which resemble professional TV production technology, has been discussed [1, 2]. To fill this gap, there is an emerging class of applications that focus on enabling the use of collaborative resources in live video production. These applications allow users to produce videos collaboratively using multiple mobile cameras, in a manner similar to how professional live TV production teams work. Previously, the most critical issues have concerned video quality aspects of the mobile systems, such as frame rate and resolution. However, these problems will diminish as mobile internet with higher bandwidths, such as 4G, becomes more established. However, as this first and most obvious level of problems with regard to quality in these services is overcome, we are likely to face a new set of challenges. We argue for a focus on challenges relating to expected delays in video transmission.

Delay is an inherent feature of all forms of signal transmission, but some forms of delay influence the perceived quality of a transmission more than others. In professional live TV production, there is a delay of a couple of seconds between the event and when it reaches the viewers in their homes. This divergence is almost never experienced as a problem. However, in the actual production situation, i.e. when the video systems are collaboratively tied together, the demands for short delays and adequate synchronization are very high. In this article, we turn our attention to how this later problem should be addressed in the domain of mobile collaborative live video production systems.

These systems involve three user roles: camerapersons, a director (or producer, both terms being used interchangeably in this text), and viewers. Camerapersons carry mobile phones and film the object of interest. Currently, mobile collaborative live video production systems support up to four different live feeds. The director sits at the control location viewing these live feeds on a mixer console. This typically shows all the live feeds at the same time in separate windows allowing the director to "multi-view" all available content. The task is then to decide, on a moment by moment basis, which camera to select for the live broadcast. The viewer consumes the final video output in real time, based on the director's selection.

In professional live TV production environments, delays are minimized by using high-speed dedicated media for video transmission and specialized hardware to synchronize multiple cameras. The new generation of applications we are investigating faces similar challenges of synchronization among multiple camera feeds and delays in the transmission of video from one point to another. We can expect these challenges for two reasons. First, since customized professional production technology is not available for these systems, we expect large delays to occur in mobile collaborative live video production systems, which will affect the "liveness" of the video transmission. This, in turn, will negatively affect the video production process. By liveness, we refer to qualities related to the perceived immediacy of the event and its presentation to viewers. Second, due to the architecture of the Internet, the delays from different cameras will potentially be different, resulting in asynchrony in the live feeds presented to the mixer. This asynchrony will affect the multi-viewing and lead to problems for producers.

We identify two problems generated by end-to-end video delays. First, end-to-end delays that in professional systems are of no consequence, because of the separation between the event and the production environment, turn out to be a source of confusion for mobile systems, since the producer often can choose between looking at the event itself and at the video feeds of it, when making broadcast selections. The time for the actual selection of a cut, as decided by looking at the event itself, is not aligned with the video stream in the system. Second, if all the cameras are filming the same event from different angles, which is likely in collaborative production, inter-camera asynchrony also becomes a serious issue.

Our initial user feedback study identifies a new type of problem, which has not been present in professional TV production. The mobile character of the mixing technology makes it easy to carry around, which enables the director to be at the same site as the camerapersons. In professional production, the director is always off-site, often sitting in a so-called OB-bus, which we term "out-of-view mixing." He only has access to the mediatized event as it appears when captured by the camera persons and communicated through the network. The mobile technology provides an opportunity for the director to be at the scene, which gives him direct access to the event in an unmediated and non-delayed way, as well as access to the mediated version of the production at the same time. We term this way of conducting multi-camera production "in-view mixing." When the director has the possibility to move around, a new form of problem arises when the delay between what is happening in real life and in the preview window becomes visible.

The paper discusses the difference in between these two modes of mixing and how systems could best provide problems demands on high synchronization and low delay, emerging out of the two different contexts. In mobile collaborative live video production, synchronization can be achieved following two steps. First, we need to ensure that there is a way to temporally compare the feeds. This could be done by marking them with a common point of reference, such as by identifying common audio features (e.g., a clapping noise) or a visual signature (e.g., a flash), or by means of synchronized time stamps. The next step is to introduce techniques to align the feeds, either by buffering at the receiving mixer side or by dropping frames of early streams at the receiver. The first approach provides high synchronization and smooth video but with a higher delay because of the extra buffering. The second approach ensures less delay in achieving synchronization, but at the cost of less smoothness. The first approach provides better video quality, but the added delay between the event and its presentation at the receiving mixer makes it difficult for the director to utilize the resource of looking directly at the event itself. Therefore, our solution suggests a dual-mode approach, which is adapted to the degree of non-mediated visual availability of the event itself.

The paper is structured as follows. In the next section, we present relevant research on video transmission. We then provide a description of two mobile collaborative live video production systems, followed by an analysis drawing on a field trial with one of the systems. We then present a delay measurement study of the existing prototypes. In the Sect. 8, we present a theoretical discussion on how to identify the problems and suggest possible approaches for finding solutions.

2 Related work

The study is influenced by research on four different topics. First, we present existing studies on mobile collaborative applications in the literature. Second, we present the research that focuses on studying the effect of delays on video streaming. We then present work that focuses on the reduction of these delays, followed by research on video stream synchronization in general as well as inter-camera synchronization.

Mobile collaboration is very well-studied field of research; however, there are not many examples of studies closely related to our topic. Kaheel et al. [3] presented a system for mobile collaborative "eventcasting" called Mobicast. This system allows users to create a better viewing experience of an event by selecting a suitable angle of view from among multiple live mobile streams, or stitching more than one video stream together to provide a wider field of view. To accomplish synchronization among multiple video streams, Mobicast uses time stamps generated by mobile clocks that are precisely synchronized using network time protocol (NTP). Engstrom et al. [4] presented a collaborative mobile video mixing application called SwarmCam, which allows multiple mobile cameras to stream video over 3G to a mixing station where a video mixer enables the director to select one of the streams and mix it with pre-recorded material in the system's video bank and broadcast the final product. Wang et al. [5] present another example of mobile collaborative system called Mobile Audio Wiki that enables audio-mediated collaboration on the move. As this application focuses on asynchronous audio-based collaborations, delays and synchronization are not an issue.

Video transmission delay generates different types of effects. Ito et al. [6] studied the effect of the mean end-to-end delay and jitter (variations in delays) on user-perceived quality of service (QoS) of live audio–video. They found that the standard deviation of delay affected the user experience more than a constant delay. Baldi et al. [7] presented a study that focuses on the question of how end-to-end delay in video conferencing in packet switched networks can be minimized. They analyzed end-to-end delay with six different settings combining three generic network architectures (circuit switching, synchronous packet switching, and asynchronous packet switching) with two video encoding mechanisms (raw video and variable bit rate [VBR] MPEG video encoding). They showed that VBR MPEG video encoding is a better choice for delay sensitive systems. Endoh et al. [8] propose a new live video streaming system featuring low end-to-end delay and jitter. The system does not incorporate audio data, however.

In all, many other researchers have focused on the problem of delay in end-to-end video transmission [9–11], but no one has performed delay analysis in collaborative settings.

Video stream synchronization is defined as maintaining the same temporal relation between different video steams at the time and place of reception as they had at the time of acquisition [12]. In networked video streaming services, the synchronization of live continuous media is a critical issue that has been extensively studied, and researchers have advanced several clever solutions for media synchronization in different scenarios. Media synchronization can be divided into three categories: intra-stream synchronization, inter-stream synchronization, and group synchronization [12].

2.1 Intra-stream synchronization

In multimedia streaming at the time of multimedia acquisition, the analog data (video or audio) are digitized and converted into an ordered list of samples. The intra-stream synchronization recovers the time-based ordered list of samples so that the original analog signal can be precisely reconstructed at the receiving end [13].

2.2 Inter-stream synchronization

In inter-stream synchronization, it is desirable to reconstruct the original timing relationship that exists between two or more media streams at the time of acquisition. Inter-stream synchronization depends on intra-stream synchronization being achieved.

2.3 Group synchronization

Group synchronization, or inter-destination synchronization, ensures the retention of the temporal relationship between different multimedia streams being displayed at different destinations (receivers).

Many studies (see, e.g., [9, 13, 14]) have analyzed the effects of multimedia synchronization. Most have proposed new systems that do not fit the requirement of our heterogeneous collaborative mobile setting. Others [15–18] have explored the possibility of using common features such as audio signatures and sequences of camera flashes in videos as reference points for calculating the synchronization offset. However, with every new application and service, the problem reemerges due to new intricacies and limitations.

Summing up, there is a large body of research that addresses the topic of synchronization and delays. However, it does not focus on delays in the mixing of live video streams, which is an essential feature of both professional live TV systems and the upcoming new types of mobile collaborative systems.

3 Background

A new generation of video production applications is emerging with the advent of 3G and 4G networks, powerful mobile devices with cameras, and live streaming technology. We take two such systems, the instant broadcasting system (IBS) and the mobile vision mixer (MVM), as examples of this technology. A brief description of these systems follows.

3.1 The instant broadcasting system

The instant broadcasting system (IBS) [19] is an example of a mobile collaborative live video production system. It is a further development of SwarmCam [4], which we have mentioned earlier in Sect. 2. IBS allows amateurs to produce their own live video broadcasts in real time using video feeds from multiple mobile cameras streaming over a 3G network. The system can support up to four mobile cameras. A mobile application captures video using each mobile phone's built-in camera and streams it live over the 3G network to the IBS node. The mobile streaming application is based on the open source mobile live streaming framework Movino [20] that uses transmission control protocol (TCP). The node is a desktop computer running a vision mixing application that allows the director to view all the camera feeds at the same time so that he/she can select the suitable one to air (see Fig. 1). The director, who mixes the live camera feeds at the IBS node (see Fig. 5), has the possibility to instruct camerapersons through a text-based back channel. The back channel also includes an automated red "tally light," alerting the cameraperson that his/her feed is being broadcast. Figure 1 shows the graphical user interface of the IBS mixing console. The system is capable of mixing live feeds from mobile cameras with pre-recorded videos. It is also capable of applying various image processing filters and blend modes to the mixed videos and broadcasting final video output to the web in real time.

3.2 Mobile vision mixer

The mobile vision mixer (MVM) [21] provides a minimalistic collaborative production environment, where the mixing resources are provided on a mobile handset. This enables both the camerapersons and the director to be mobile.

The MVM piggybacks on the commercial Bambuser [22] live mobile video broadcasting service to capture the individual streams and then to output the finalized broadcast. The video is transmitted using the TCP-based Real-Time Messaging Protocol (RTMP). Figure 2 shows how four camera feeds and the final video output are streamed into Bambuser using the 3G network and the Internet, respectively. The local MVM server fetches these camera feeds from Bambuser, combines them, and presents them to the mobile mixer application through Bambuser.

The director can select any of the four camera feeds to be broadcast as final output, using the mobile mixer application on a mobile phone. Figure 3 shows four camera views in the mobile mixer application. When the director selects a certain camera feed, this selection command is sent to the local MVM server, which in turn broadcasts the selection via Bambuser as a final output. The result is a sequence mixed according to the director's temporally unfolding mixing decisions.

4 Method

To investigate our hypothesis of the existence of delay problems in mobile collaborative production, we have carried out two types of studies. First, we performed an initial user feedback study by having a group of teenagers (aged 11–17 years) producing live videos using the MVM system at a skateboarding park. The field test took place in Stapelbädd Park (Stapelbäddsparken) in Malmö, June 2010. We decided to employ an ethnographic method in which we video recorded and observed the participants using the system.

The participants were told that they were going to be video recoded during their test of the system, and accepted it. The participants were selected by the managers of these facilities and consisted of three different groups with teenagers in between the size of 5 and 7 persons. The groups consisted of four camerapersons and a director. At their request, the directors were allowed to assistants.

The participants were instructed on the system functionalities. We briefly introduced the system to the participants and showed them how it works. But they were given

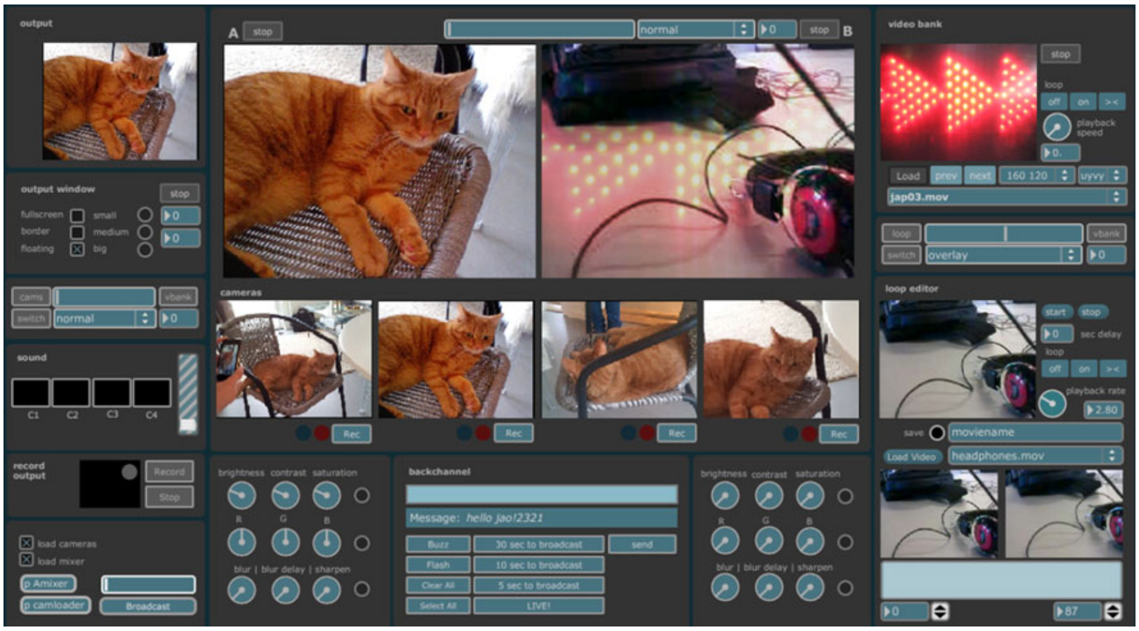

Fig. 1 Graphical user interface of the IBS mixer node

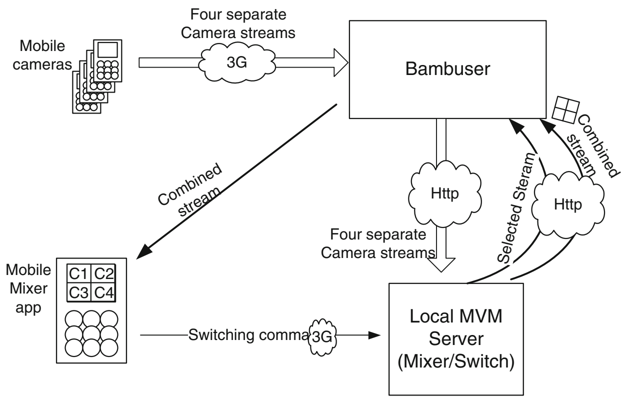

Fig. 2 Architecture of MVM

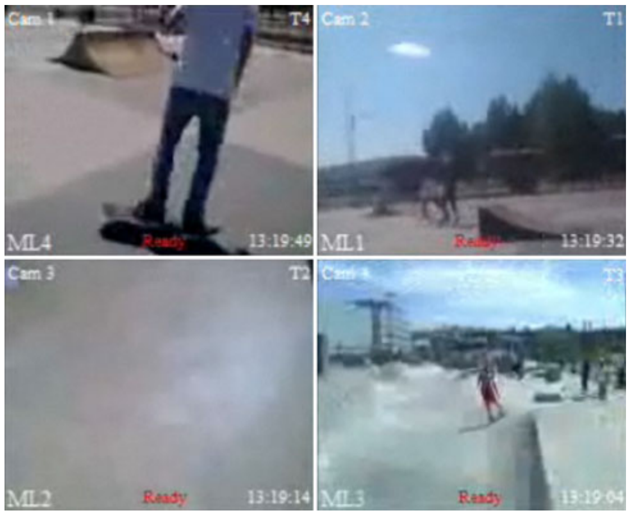

Fig. 3 Four-camera view in mobile mixer interface

no request as to what topic to film or how to make use of the system, other than filming a topic that interested them during the skateboarding activities. They were then gathered for a focus group interview directly after the test. The focus group approach supports a free and ad hoc commenting of the system performance than formalized interviews. This interview was also video recorded. This method enabled us to see how participants confronted and dealt with different situations [23].

The video recording of the test was then transcribed, which allowed us to make additional indications on how users oriented themselves with regard to delays and synchronization issues in collaborative production.

Second, we conducted an experiment to measure the delays in the two systems (IBS and MVM). We recorded a high definition (HD) video of both IBS and MVM in operation on separate occasions. We used three Nokia N86 8MP mobile phones as cameras in IBS and an additional HTC Google Nexus One phone as a fourth camera in the MVM test setup. We filmed a screen displaying changing colors in separate tests for IBS and MVM. The changing colors on the display served as an "event" in this test. The occurrence of the event and its presentation on the mixer console and in the final video output were recorded in the HD video. We measured the different delays in the given systems by studying this HD video recording frame by frame.

5 Analysis

5.1 Initial user feedback study

The study provided feedback on how the multi-camera setup was used to tell stories and what the participants focused on, as well as on general impressions regarding the concept itself. Here, we will present aspects of the study of relevance for the discussions on delay and synchronization.

It was generally apparent to the observers that the producers were often looking back and forth between the park and the display showing the camera feeds of the event (see Fig. 4). This is a way of mixing that is not available to professional producers who most of the time are confined to a bus, where the only access to the event is through the screens displaying the camera captures [24]. Utilizing the mobile character of the system in this way makes the delay between the event and its presentation on the mixing screen visible. The producers commented on this as causing various problems. One producer claimed that the lag made the task more demanding in general:

> It was lagging a bit behind as well, so it was…it was a bit hard. You saw a person do something, and then it turned up on the camera afterward, so you didn't know how to…you had to wait and see when it turned up on the display.

We asked another producer whether it was possible to decide which stream came from which cameraperson:

> It was a bit difficult since it was lagging behind.

Thus this lag gets in the way of connecting a specific feed to a cameraperson, thus making it difficult to get an overall grasp of the available material. We also observed how the producers managed to work around the problem by mixing while looking at the actual scene rather than viewing it through the mobile mixer application. They were able to do this because they were physically present at the filming location, and could make out the relative positions of the camerapersons at the scene, as shown in Fig. 4. This reduced the confusion concerning delays, but such a method of mixing requires them to take decisions without previewing what the camerapersons are filming.

5.2 Technical delay measurement

In order to further investigate the question, we performed technical tests to measure delays occurring in both the IBS and the MVM systems.

5.2.1 Delay test of the IBS

We recorded a high definition video of IBS in operation to observe the delays in the system. The experimental setup was as follows. We used three *Nokia N86* mobile phones as cameras (*Cam1, Cam2, and Cam3*) aimed at a color-changing display, as shown in Fig. 5. The changing colors on the display served as an event that was filmed and streamed from each of the mobile cameras to the *IBS node*. *Cam1* input was selected for the final output of the IBS, displayed at the *Local Output* display, and it was also broadcast to a *Remote Machine* (web).

We measured different delays in IBS by studying the video recording frame by frame. As the video's frame rate was 25 frames per second, the precision in the delay measurements is up to 40 ms. The delays that we measured during the experiment are presented in Fig. 5. *D1* is the overall delay between the event and the final remote machine output. *Dc1, Dc2,* and *Dc3* are the delays between cameras *Cam1, Cam2, Cam3* and their corresponding displays *C1, C2,* and *C3* in the *IBS node*.

5.2.1.1 Results The line chart in Fig. 6 shows delay values plotted against time. The delays *Dc1*, *Dc2,* and *Dc3* tend to follow almost the same trend. The average value of these delays in our study turned out to be 900 ms. The overall delay *D1* has an average value of 9.08 s. The small fluctuation is caused by the nature of 3G networks, in

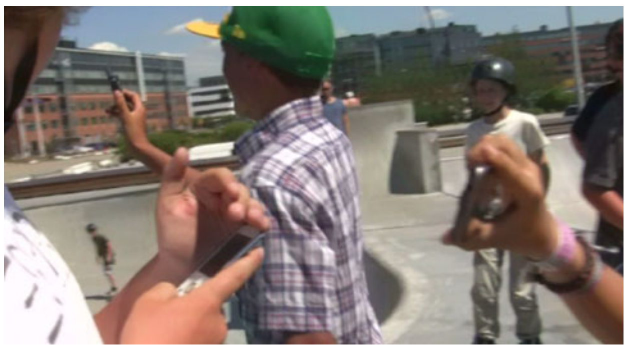
Fig. 4 Director (*left*) alternating between looking at the event and the MVM mixer on a mobile phone

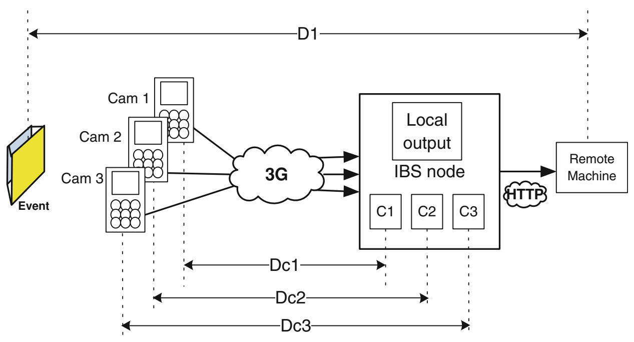

Fig. 5 Overview of IBS and delays involved

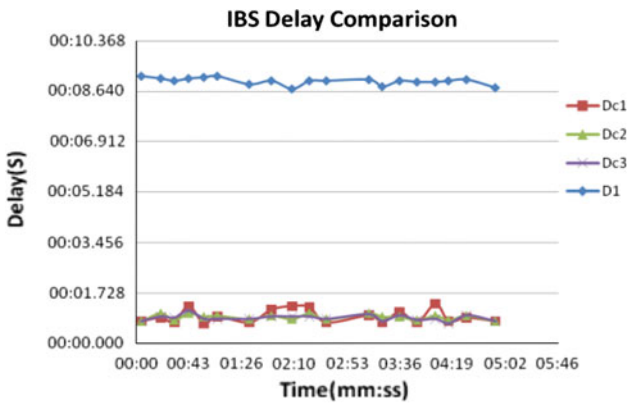

Fig. 6 Delays in instant broadcasting system

which we cannot be sure that all the streams follow the same route.

5.2.2 *Delay test of the MVM*

Figure 7 shows the delays measured over time in the MVM system. The Bambuser server and the local MVM server are presented as a single MVM node in order to simplify the description. *Dcm1* is the delay between the camera *Cam1* and its corresponding presentation *C1* in mobile mixer application. *Dcm2, Dcm3,* and *Dcm4* are the same kinds of delays for *Cam2, Cam3,* and *Cam4,* respectively.

Ddir is the delay between the camera selected for final output (in this case *Cam1*) and the *Final Output*. The time that elapses between the instant when the director changes the camera selection at the *Mobile Mixer Application* and when the director's output changes to the newly selected camera is called *Dswitching*.

We repeated the same experiment as with IBS to measure the live video transmission delays in MVM. Three *Nokia N86* and one *Google Nexus One* phones were used as mobile cameras. The mixer application was running on another *Nokia N86* mobile phone.

5.2.2.1 Results We also measured Dcm1, Dcm2, Dcm3, Dcm4, Ddir, and Dswitching. Figure 8 shows delay values in MVM plotted against time. From the line chart, it is clear that the delays *Dcm1, Dcm2, Dcm3,* and *Dcm4* increase over time, which is caused by an implementation flaw in the MVM (to be fixed in the next version).

However, even the minimum value, 11.120 s, is enough to severely affect the liveness of the feed and hence the director's performance, since he/she instructs the camerapersons and makes mixing decisions based on his/her view in the *Mobile Mixer App* (Mixer console).

The delay from the *Google Nexus One Phone*, *Dcm4,* follows a completely different pattern possibly due to its

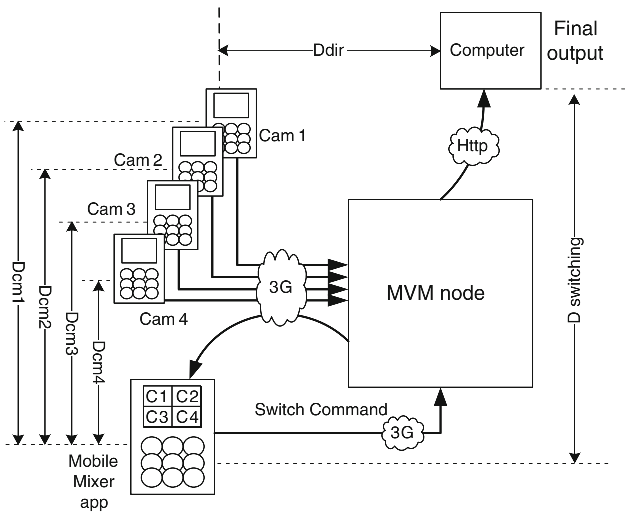

Fig. 7 Overview of MVM and delays involved

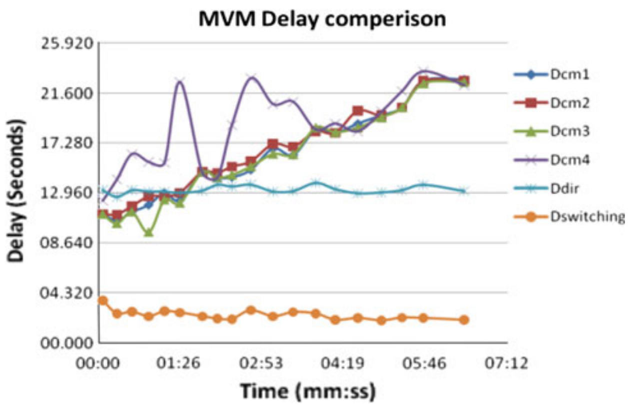

Fig. 8 Delays in mobile vision mixer

transmission properties differing from those of the other three camera phones, hence causing a synchronization problem. We will address these issues in the following section.

Ddir, the final output delay, has an average value of 13.20 s. This value is also high, but a remote viewer cannot perceive the delay as long as the delay is constant. The average switching delay, *Dswitching,* is 2.38 s. *Dswitching* does not affect the director's performance.

6 Discussion

In this section, we will bring together the results of the technical tests and the initial user feedback study. The technical tests indicate that delays will affect different user roles in collaborative live video production. We will here further discuss the test results, their effects on users, the causes and effects of asynchrony in video feeds, and related issues pertaining to video synchronization in a collaborative mobile video production scenario.

6.1 Problems with delays in IBS and MVM when mixing at the event

In this section, we will focus on the new types of delay problems that the initial user feedback study identified. These delays will occur with mobile vision mixers when the producers are able to watch the event both live and on the preview screen. In video streaming applications, end-to-end video delay is generally composed of three major components.

- *Sending delay:* The time consumed in capturing and processing the video data units before dispatching them over the network.
- *Transmission delay:* The time consumed in transmitting a video data unit over the network.
- *Receiver delay:* The time consumed in processing the video packet received from the network and displaying the data in the final output.

The sender and receiver delays will be reduced with the arrival of more powerful mobile devices, while the network delay can be reduced by choosing favorable transmission protocols in the video streaming applications. As we have mentioned earlier, both IBS and MVM use TCP as the transport protocol for video streaming. TCP is not well suited for live video streaming as it can introduce undesirable delay because of its retransmission mechanisms [25]. Streaming protocols based on User Datagram Protocol (UDP), such as Real-Time Transport Protocol (RTP), are more favorable in this regard.

In our tests, we have measured two types of delays that occur in both IBS and MVM, delays between cameras and the mixer console, and the overall delay between an event and its presentation to the "viewer" (see Figs. 5, 7). The delay from camera to mixer console affects the collaboration between director and cameraperson. When the director instructs a cameraperson through a feedback channel, he/she does so according to his/her current view of that particular camera feed. The overall delay, on the other hand, is not significant as the viewer cannot perceive it. However, if a real-time feedback channel is introduced between the viewer and the director, the overall delay can become a problem as the viewer and the director would not share the same view of the event at any given moment. But we learned in the user study that the delay also confused the producer and made it harder to mix in general and in particular made it more difficult to link specific feeds to individual camerapersons.

Furthermore, different devices may have different delays, as was evident in the result of the MVM delay measurements. This gives rise to the problem of asynchrony in the presentation of the different previews to the producer.

To further clarify the identified problems, let us consider a scenario where amateur users are producing a collaborative video using MVM, as shown in Fig. 9. Four camerapersons (c1, c2, c3, and c4) are filming the same event from different angles. The director holds a mobile phone running a mobile vision mixer application. The continuous unfolding of the event is presented in terms of instants in time, or frames (f0, f1, f2…), in the video context in Fig. 9.

Each camera feed has a different delay, which means that the director sees differently timed framings of the event in the different camera feeds at the same time on his mobile mixer application, despite all the camerapersons filming the same event at the same place and time from different angles (see Fig. 9). This delay and asynchrony in the video presentation at the director's end cause inaccurate and bad mixing decisions in the video production, thus severely impairing the final broadcast quality.

From the above discussion, we can infer that video streaming delays negatively affect timing decisions in production when the viewer (in this case the director) is able to observe the event directly or through some other real-time medium. It is also clear that asynchrony between video feeds negatively affects the multi-viewing.

6.2 Causes of asynchrony

The following factors are traditionally seen as causing asynchrony in a networked environment [12]:

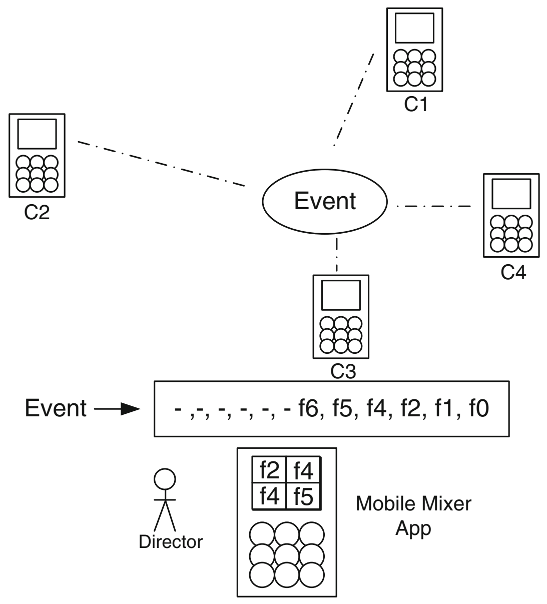

Fig. 9 Amateur user practices

- *Network delays:* Delays for stream packets in the network to reach their receiver, which vary according to network load.
- *Network jitter:* Variation in network delay caused by the variation in network load and other network properties.
- *Receiver system delay:* Delay caused by the processing time needed by the receiving system. This is the time that elapses between the reception and presentation of stream data.
- *Receiver system jitter:* Variation in the receiver system delay caused by varying system load and processing delays.
- *Clock skew:* Difference in the clocks of sender and receiver.
- *Clock drift:* Variation in clock skew caused by variations in temperature and other imperfections in the clock.

6.3 Synchronization in collaborative mobile video production scenarios

We have learned from our technical delay measurement tests that the systems have delays between camerapersons and the director, as well as between the director and the viewers. We have also shown that different mobile devices can exhibit different delays depending on many factors ranging from device capabilities to the current network load. This further adds to the asynchrony among multiple camera feeds.

Because in this paper we are talking about collaborative mobile videos, the mobile collaborative sessions are not likely to be of long enough duration for significant clock skew and drift effects to be experienced. And in any case, they can be handled using clock synchronization protocols like Network Time Protocol (NTP). The receiving system's delay and jitter are problems that depend solely on the receiver's processing capability. Network delay and jitter are important factors to consider and are normally equalized by employing elastic buffers at the receiving end using different buffering techniques [12].

In professional live video production, the synchronization among multiple camera feeds, also called inter-camera synchronization, is achieved by physically connecting the cameras to external synchronization hardware [18]. This process is called "jam-sync" [26]. This device keeps the cameras' internal clocks synchronized. When recording with multiple cameras, a "clap" sound is also commonly used at the start of each take. The clap sound causes a loud audio signal that is recorded by all the cameras filming the shot. This audio feature is later used to synchronize videos manually while editing [26]. In our case, however, as the cameras are mobile, jam-sync or similar clock synchronization techniques are not suitable.

7 Design suggestions

Based on our study and analysis, we present the following design suggestions to cope with the delay and synchronization issues in mobile collaborative live video production systems. We see this as a two-step process, as indicated in the introduction. The first is to ensure that the different feeds' temporal variations vis-à-vis the event can be compared, i.e., to calculate synchronization offset. Second, we need to align or synchronize the feeds, i.e. equalize the asynchronies with buffering/sync techniques. The techniques that we have considered (time stamps, buffering, frame drops, etc.) already exist; in this text, we argue for their suitability for our particular scenario. Possible techniques are discussed with reference to the delay requirements.

Unfortunately, due to the mobility of such systems, we cannot go with solutions that employ additional synchronization hardware. Instead, we need to follow an approach that does not require any special changes at the cameraphone end.

7.1 Temporal comparison or synchronization offset

Synchronization offset is the temporal difference between two continuous media streams, as shown in Fig. 10. In mobile video collaboration environments, the following two techniques can be good candidates for calculating synchronization offset.

7.1.1 Audio signature method

When multiple cameras are filming the same event, we can extract audio signatures from corresponding streams and

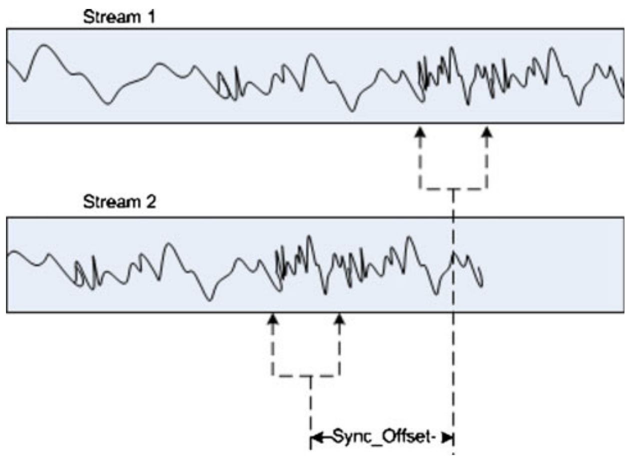

Fig. 10 Calculating synchronization offset

calculate synchronization offset [16] by comparing the occurrence of similar features in the audio of both streams, as in shown in Fig. 10.

The advantages of using this approach to calculate synchronization offset is that we do not have to worry about clock drift and skew, as we are not relying on the timestamps in the stream. But on the other hand, this approach requires more processing resources, thus introducing an extra processing delay at the receiving end. Also, this approach requires all the cameras to be present at the same location, which is not always the case in mobile collaboration.

7.1.2 *Timestamp method*

Using timestamps generated by the cameras' internal clocks to calculate synchronization offset would be more efficient in terms of processing resources. However, when depending solely on timestamps generated by cameras for this purpose, the inaccuracies caused by clock drift and skew can get in the way. However, in most practical scenarios, the mobile live video production time will not exceed a few hours. Therefore, clock drift and skew will not have a significant effect on the final synchronization calculation. Thus, we can safely choose the timestamp method for offset calculation.

If higher precision in clock synchronization should nevertheless be required, it is possible to use the network time protocol (NTP) to keep the mobile devices' clocks synchronized. This protocol offers precision on the order of 10 ms [27].

7.2 Temporal alignment and synchronization techniques

It is very important that the director receives all streams of an event at the same time, and with as little delay as possible, in order to be able to cut between different camera angles [26]. Synchronization issues are usually handled by buffering and/or frame dropping [12, 28], after calculating the synchronization offset. On the other hand, stream quality is also of importance, since the producer also needs to be able to see what is going on in the event by looking at the screen. The following two buffering schemes/techniques balance these requirements differently. These techniques presented here are very abstract, high-level ideas from the existing literature and should not be taken as finished technical solutions.

7.2.1 *Pre-mixer buffering technique*

Figure 11 shows three mobile cameras streaming live to the IBS mixer console (IBS node). A, B, and C are live streams from camera cam1, cam2, and cam3, respectively. Stream C is the most delayed and B the least delayed stream. The vertical blocks along the streams represent individual video frames. The gray frame shown represents a certain event captured at the same time by all cameras.

The positioning of the gray frames in the different streams shows that the streams are experiencing different degrees of delay and are thus out of synch. In this technique, we propose buffering the least delayed stream (B) until the most delayed stream's (C) buffer starts filling up. Simply put, we shall buffer stream B until the buffers for stream A and stream C also receive a gray frame so that we can present them at the same time in the display. In this way, the asynchrony among the live streams can be equalized before presentation to the mixer console in a similar way as presented by Shepherd et al. [29] and Escobar et al. [30]. In all, this technique ensures the synchronization of the streams with high visual quality, but at the expense of a longer delay between the event and the previews on the mixer console.

7.2.2 *Frame dropping technique*

There is another scheme available to ensure synchronization [31, 32], which here can be used to minimize the delay at the mixer console. In this scheme, as shown in Fig. 12, the early arriving frames in streams A and C are chopped off and only those frames that arrive in synch with the delayed stream B are presented to the mixer console. This scheme ensures synchronized playback of all the streams at the mixer console with a shorter delay, but at the cost of less smooth video playback because of the frame dropping. This technique minimizes the delay to the director, but reduces the available video quality and the possibility to make decisions about which feed to broadcast.

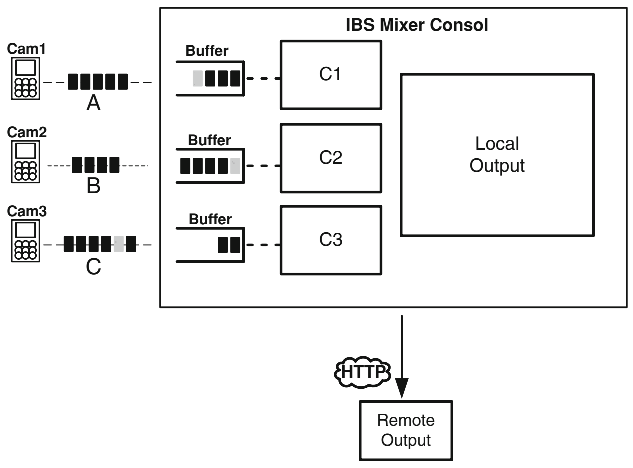

Fig. 11 Buffering before presentation to mixer

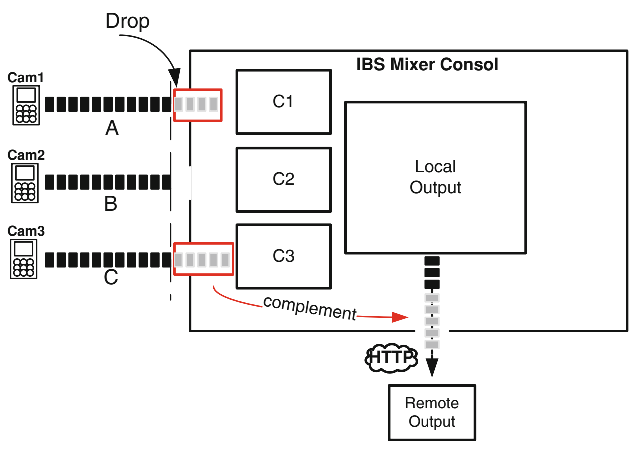

Fig. 12 Data after presentation to mixer

7.3 A context approach to mixing

Considering how the approaches described above balance delay and synchronization in different, but not altogether satisfactory ways, we suggest a context approach that is adapted to the new types of applications discussed in this article. From our user practices study and analysis of delays and video synchronization issues, we can conclude that live video production in mobile collaborative scenarios will occur in two major production settings: “in-view mixing” and “out-of-view mixing.” Both settings have their own specific requirements regarding delays and synchronization. Thus, our suggested solution adapts the way the balance is accomplished to the context of the producer.

7.3.1 In-view mixing

In this setting, the director is present at the site of the event being filmed and can directly see and observe the event, as well as look at the mixer console with the preview streams. In this situation, delays and asynchrony in the mixer console are both highly noticeable. In an “in-view” mixing scenario, delays and asynchrony are quite intolerable as they confuse the director and affect his/her production decisions. The two temporal alignment techniques above represent two different priorities in the trade-off between *delay* and *smoothness*. In in-view mixing, smoothness may be compromised in this case, as the director can also see and observe the event itself. As the frame dropping technique ensures a shorter delay in the streams on the mixer console, this is quite suitable for scenarios where the director is mixing and producing live videos while looking directly at the event.

7.3.2 Out-of-view mixing

In this setting, the director is producing/mixing the live streams at a location separate from the actual event being filmed, and he/she can only see the event through the camera feeds that are presented to him/her in the mixer console. In this case, the director cannot notice delays, as the actual event is not available for comparison. The synchronization among streams and smoothness of the video presentation is of great importance here because they affect the *multi-viewing* and thus influence the director’s mixing decisions.

Here, we suggest that the pre-mixer buffer technique is more suitable. It can be useful for improving the synchronization among video streams with smooth presentation, however, due to extensive buffering it also causes increased delays. In the case of “out-of-view” mixing, the delay to the mixer console does not matter and can thus be tolerated.

The close analysis of video streaming delays, jitter, and synchronization brings up an interesting relationship among the three. When the video jitter effect is covered up (by buffering), the delay builds up. Similarly, when we try to synchronize camera feeds with different delays, and sometimes visible jitter, as in the case of C4 in MVM, the delay builds up further because when we synchronize video feeds, we again need to rely on buffering. Ideally speaking, the camera feeds presented to the mixer console should have negligible delay, high synchronization, and high smoothness while being played back. However, in reality, there is always trade-off between these parameters.

In our pre-mixer buffering technique, the focus is to achieve synchronization while keeping the video playback smooth. In such techniques, the more we improve synchronization (using buffers), the longer the delay, as is shown in the Fig. 13. The ideal/professional systems have higher synchronization and lower delay, as indicated by the cross in the fourth quadrant in Fig. 13. This technique is suitable for “out-of-view” mixing where synchronization and smoothness are more important than immediacy.

In the case of the frame dropping technique, we are maintaining a low delay. We achieve synchronization by dropping early frames at the cost of smoothness. In this technique, the more synchronous the streams, the jerkier the individual streams will be, given differences in transmission quality. The dotted line in Fig. 14 shows this relationship.

This kind of technique is suitable for an “in-view” mixing scenario as it ensures low delay with synchronization, when a decrease in smoothness can be tolerated. In the ideal case, the system should have highly smooth video playback with high synchrony, as indicated in the second quadrant of the graph in Fig. 14.

8 Conclusion

With the advent of live streaming in mobile phones, made possible by high-speed mobile networks, a new class of

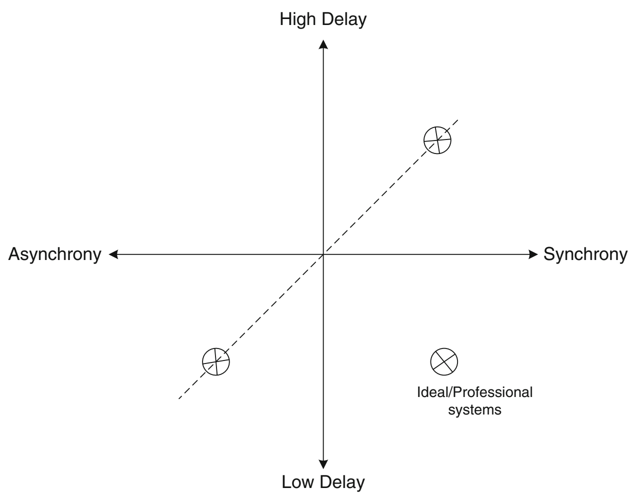

Fig. 13 Synchronization versus delay

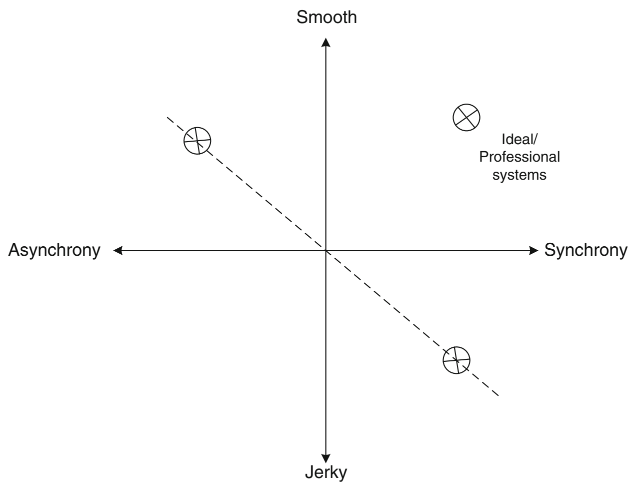

Fig. 14 Synchronization versus smoothness

applications, mobile collaborative live video production systems, are coming into being. Previously, the focus in research has been on image quality and the generation of new conceptual features. We argue that since video collaboration is based on mobile networks (3G, 4G), the next important issue will concern the handling of delays and asynchronies in such applications.

We have investigated these problems in mobile collaborative live video production by both studying the user practices and performing technical tests with prototypes.

We have analyzed the effects of different delays in two examples of collaborative mobile video production systems. The delays in IBS affect both camerapersons and the director, as it has a feedback channel. More advanced systems are likely to incorporate a viewer's feedback channel as well [19], where viewers will have the possibility to influence the production as well. This means that the final delay, which in the current implementation does not have any effect, would more visibly affect the production process. The situation would be even more complex with two-way feedback channels between cameraperson and director or/and between a director and viewer. From this, we can conclude that the higher the level of collaboration, the more complex the effect of delays.

From our user practices study, we found that streaming delay impairs the liveness of the video and that asynchrony affects the viewing of multiple feeds (multi-viewing) by the director. In mobile collaborative live video production, there are two important mixing modes or settings: "in-view" mixing and "out-of-view" mixing. These two mixing settings have slightly different requirements, which lead us to propose a context-dependent approach which balances the requirements for delays, synchronization, and image quality differently. For "in-view" mixing, we need a technique that ensures small delays and good synchronization; for "out-of-view" mixing, on the other hand, synchronization and smoothness are more important than short delay time. We recommend the pre-mixer buffer technique for "out-of-view" mixing and the frame dropping technique for "in-view" mixing.

References

1. Engstrom A, Juhlin O, Reponem E (2010) Mobile broadcasting—the whats and hows of live video as a social medium, In proceedings of mobile HCI 2010, Sept 7–10, Lisbon, Portugal
2. Engström A, Perry M, Juhlin O (2012) Amateur vision and recreational orientation: creating live video together. In proceedings of CSCW 2012 Seattle
3. Khaleel A et al (2009) A system for collaborative event casting using mobile phones. In proceeding MUM '09 proceedings of the 8th international conference on mobile and ubiquitous multimedia
4. Engström A, Esbjörnsson M, Juhlin O (2008) Mobile collaborative live video mixing. Proc. Mobile HCI 2008:157–166
5. Wang L et al (2008) An audio wiki supporting mobile collaboration. In proceedings of SAC '08 Proceedings of the 2008 ACM symposium on applied computing
6. Ito Y, Tasaka S, Fukuta Y Psychometric analysis of the effect of end-to-end delay on user level QoS in live audio–video transmission. In 2004 IEEE International Conference on Communications
7. Baldi M, Ofek Y (2000) End-to-End delay analysis of videoconferencing over packet-switched networks. IEEE/ACM Trans Netw 8(4):479–492
8. Endoh K, Yoshida K, Yakoh T (2008) Low delay live video streaming system for interactive use. The IEEE international conference on industrial informatics (INDIN2008) DDC, Daejeon, Korea, July 13–16
9. Sareenan CJ, Narendaran B, Agarwal P (1996) Internet stream synchronization using concord. Proceedings of IS&T/SPIE International Conference on Multimedia Computing and Networking (MMCN)

10. Gualdi G, Cucchiara R, Prati A (2006) Low-latency live video streaming over low-capacity networks. Proceedings of the eighth IEEE international symposium on multimedia
11. Weber R et al (2006) Measurement and analysis of video streaming performance in live UMTS networks. Proceedings of WPMC 2006 conference, San Diego, CA, USA, Sept 17–20, page 1 of 5
12. Boronat F et al (2009) Multimedia group and inter-stream synchronization techniques: a comparative study. Inf Syst 34: 108–131
13. Blum C Practical method for the synchronization of live continuous media streams, Institut Eurécom
14. Rautiainen M et al (2009) Swarm synchronization for multi-recipient multimedia streaming, multimedia and expo, 2009. ICME 2009. IEEE international conference on page(s):786–789
15. Cremer M, Cook R (2009) Machine-assisted editing of user generated content. Proceedings of SPIE-IS&T Electronic Imaging SPIE vol 7254. Article ID 725404, p 10
16. Haitsma J, Kalker T (2002) A highly robust audio fingerprinting system. In Proceedings of the international symposium on music information retrieval
17. Kennedy L, Naaman M (2009) Less talk, more rock: automated organization of community-contributed collections of concert videos, WWW'09: Proceedings of the 18th international conference on World Wide Web, ACM, pp 311–320
18. Shrestha P (2009) Automatic mashup generation of multiple-camera videos, a Ph.D. dissertation, Technische Universiteit Eindhoven
19. Engstrom A, Juhlin O, Esbjörnsson M (2009) Instant broadcasting system: mobile collaborative live video mixing. In proceedings SIGGRAPH ASIA '09 ACM SIGGRAPH ASIA 2009 Art Gallery and Emerging Technologies: adaptation
20. http://movino.org. Accessed 20 Dec 2011
21. Toussi R (2011) Mobile vision mixer, a system for collaborative live mobile video production, a master thesis, Royal Institute of Technology (KTH), Stockholm, Sweden
22. http://bambuser.com/. Accessed 9 Dec 2011
23. Weilenmann A (2001) Negotiating use: make sense of mobile technology. Pers Ubiquitous Comput 5:137–145
24. Perry M, Juhlin O et al (2009) Lean collaboration through video gestures: co-ordinating the production of live televised sport. In CHI '09 Proceedings of the 27th international conference on Human factors in computing systems
25. Wang B (2008) Multimedia streaming via TCP: an analytic performance study. ACM Trans Multimed Comput Commun Appl 4(2):1–22
26. http://robgwilson.com/2009/04/14/jam-sync-your-damn-cameras. Accessed on 05 Nov 2011
27. http://www.bytefusion.com/products/ntm/ptnt/timeprotocolsaccuracy.htm/. Accessed at 10 Oct 2011
28. Lynnae E, Borko F, Mohammad I (1994) Evaluation of multimedia synchronization techniques. In proceedings of multimedia computing and systems (MMCS1994)
29. Shepherd D, Salmony M (1990) Extending OSI to support synchronization required by multimedia applications. Comput commun 13(7):399–409
30. Escobar et al (1994) Flow synchronization protocol. IEEE/ACM Trans Netw 2(2):111–121
31. Huang CM et al (2000). PARK: a paused-and-run k-stream multimedia synchronization control scheme. In Proceedings of the 20th International conference on distributed computing systems, Taipei, Taiwan
32. Manvi SS, Venkataram P (2006) An agent based synchronization scheme for multimedia applications. J Syst Software (JSS) 79(5): 701–713