

A HANDOVER SCHEME FOR VIDEO STREAMING OVER HETEROGENEOUS MULTICAST/BROADCAST AND UNICAST NETWORKS

YIHAN XU, YUAN MA, WENLI WU AND YUE GAO

College of Information Science and Technology

Nanjing Forestry University

No. 159, Longpan Road, Nanjing 210037, P. R. China

xuyihan@njfu.edu.cn; mayuan_njfu@163.com; {1053670916; 351710242}@qq.com

Received July 2016; revised November 2016

ABSTRACT. *With the deployment of heterogeneous networks, mobile users are expecting ubiquitous connectivity when using applications. For bandwidth-intensive applications such as Internet Protocol Television (IPTV), multimedia contents are usually transmitted using multicast delivery method for reason of efficiency. However, not all networks support multicast delivery. Hence, multicast delivery could lead to service disruption when the users move from a multicast network to non-multicast network. In this paper, we propose a connection handover subsystem called Application Layer Seamless Switching (ALSS) to provide smooth multimedia delivery across multicast and unicast networks. Attention is paid on analyzing and fine-tuning the various stages within the third phase of vertical handoff process, i.e., handover execution in order to reduce handover latency needed by Mobile Terminals (MTs) to achieve seamless playback. A prototype has been implemented to study the overall handover delay, particularly on the overlapping period where both network interfaces were activated. Results showed that the overlapping period for Multicast-to-Unicast (M2U) and Unicast-to-Multicast (U2M) handover took a minimum of 56 and 4 milliseconds respectively. The measurement of Peak Signal-to-Noise Ratio (PSNR) further showed that the quality of received video frame during handover was at the minimum of 33dB which is categorized as good based on ITU-T recommendation.*

Keywords: Multimedia service management, Experimental approach, Multimedia session continuity, Seamless multimedia session handover, Unicast/multicast switching handover

1. Introduction. Mobile Terminals (MTs) are rapidly evolving towards supporting multi-mode operations with the adoption of multiple air interface technologies within a single mobile device. In combination with the omnipresence of heterogeneous access networks such as Wi-Fi, WiMAX and LTE, users can access multimedia services anywhere at any time through any network. Internet Protocol Television (IPTV) is one of the popular applications, in which multimedia contents are transmitted to its subscribers using either unicast or multicast delivery method.

Multicast seems to be the best way to deliver multimedia services to a large number of users due to its bandwidth efficiency [1]. While multimedia content delivery could gain performance benefits with multicast, such capability is not consistently available across the entire network infrastructure [2,3]. For example, although UMTS network supports the Multimedia Broadcast Multicast Services (MBMS) for Mobile TV (MTV) services, it is not activated in certain geographical areas or cells [4,5]. This is due to the costing issue resulting network operator to provide MBMS only in areas with high subscriber density [6-8]. This limitation leads to service disruption when a user moves from an area with

multicast delivery to another area without multicast support. Hence, a handover scheme that takes account of different delivery methods while guaranteeing seamless playback is needed.

The closest work related to handover between multicast and unicast networks is in [9] where the various signaling flows for session joining and session handover across both network with and without multicast support were described. However, it lacked detailed description and performance analysis on the proposed mechanism. Both [10,11] focused on seamless handover and playback of streaming contents. However, simply emulating IPTV/video traffic with RTP streams is insufficient since IPTV traffic consists of both synchronized audio and video streams. Either one of these streams experiencing buffer underflow at the MT will result in an interrupted multimedia stream to the user. Unfortunately, this aspect is not studied in both [10,11].

In this paper, we propose a handover scheme called application layer seamless switching (ALSS) to provide smooth multimedia delivery across unicast and multicast networks in Figure 1. As unicast streaming has a direct relationship between a server and a client whereas multicast streaming is a one-to-many connection, a dedicated multimedia stream should be delivered to the user who is moving from a multicast network to a non-multicast area. During the handover period, ALSS aims to preserve the ongoing multimedia session, i.e., seamless playback. To this end, soft handover forms the basis of ALSS as hard handover is inadequate to support seamless handover of multimedia streams [12]. Here, soft handover refers to one in which the same multimedia stream (but a different delivery method) is sent to the MT via two Access Points (APs) simultaneously. The multimedia stream from the old AP is terminated only after a specific overlapping period has elapsed. It is, however, not clear how long the delay should be to achieve a seamless playback. This paper is devoted to answering the above question with the consideration of all practical constraints, especially the audio and video buffer behaviors. To demonstrate the realistic effectiveness of ALSS, we built a real-time streaming testbed where a standardized streaming method is adopted. The measurements from such an experimental approach are an important complement to previous analytical and simulation studies.

The remainder of this paper is organized as follows. Section 2 presents the ALSS system describing both the system architecture and switching interaction between the client and the server. Section 3 describes both the prototype implementation and testbed setup. The descriptions of test methodology, performance metrics and comparison are provided in Section 4. Section 5 discusses the performance evaluation and results of the prototype implementation. Section 6 summarizes the paper and highlights some future work.

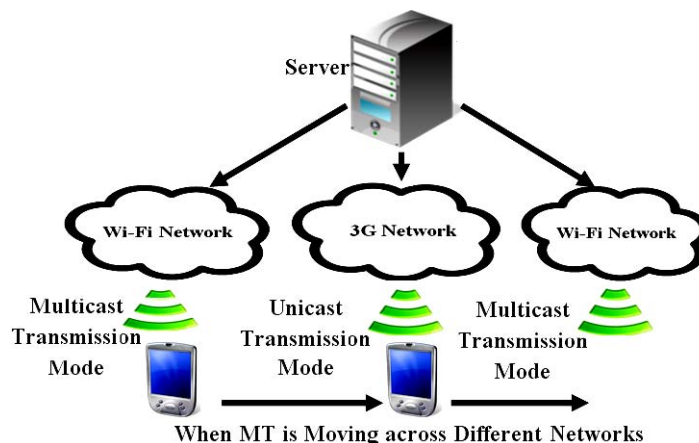


FIGURE 1. Switching between multicast and unicast networks

2. Application Layer Seamless Switching (ALSS) System.

2.1. System architecture and components. Figure 2 shows the architectural overview of both the client (MT) and server of ALSS. Basically, it consists of two layers. The top layer consists of the proposed modules (shaded blocks) that we have built to manage seamless switching. The underlying modules are existing modules that perform video encoding, decoding and streaming of multimedia data.

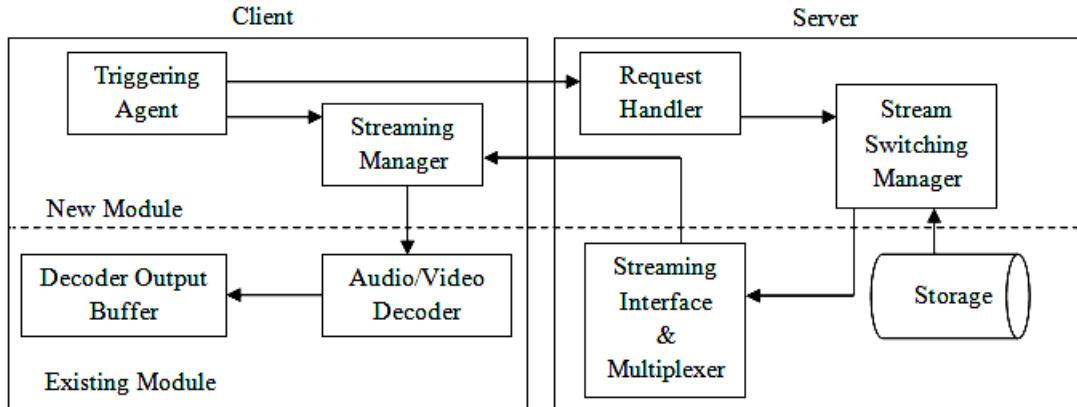


FIGURE 2. The client and server architecture for ALSS

There are four components in the client, namely the triggering agent, streaming manager, A/V decoder and decoder output buffer. Triggering agent is responsible for initiating the connection switching and is hence in charge of creating an alternative connection to the server during handover. Thus, the proposed handover scheme belongs to the terminal-controlled handover. Depending on the type of alternative connection (unicast/multicast), the triggering agent will either pass over the current streaming Uniform Resource Locator (URL) or issue a multicast join request. Various triggering algorithms for handover have been reported in [13,14], and they are not our intention in this paper to propose yet another triggering algorithm. Instead we focus on the experimental aspect of the handover execution phase for audio/video delivery. For this reason, the switching process is manually triggered with a button pressed on a multimedia player user interface. The functionality of streaming manager is to receive streaming contents (both current and new streams) from the server. For the new stream, it extracts payload from the incoming packets in order to classify and forward the packets to the A/V decoder. The A/V decoder is responsible for decoding audio and video packets, i.e., to assemble defragmented packets into audio and video frames, and later store them into the decoder output buffer for playback purpose.

Corresponding to the client, there are four modules at the server: request handler, Stream Switching Manager (SSM), storage, and the streaming interface and multiplexer. Request handler is an active module that is always listening to any incoming client request. Its main task is to extract the IP address of any incoming client and forward it together with the streaming URL to the SSM. SSM utilizes the streaming URL to look up the resource that is currently being delivered to the requesting client and then retrieves the current playback time of the sending stream. After that, it accesses the same multimedia from the storage, jumps to the specific playback time frame, and informs the streaming interface to stream the multimedia back to the client using an alternative delivery method. The multiplexer is responsible for multiplexing audio and video streams to form a single stream before transmitting over network.

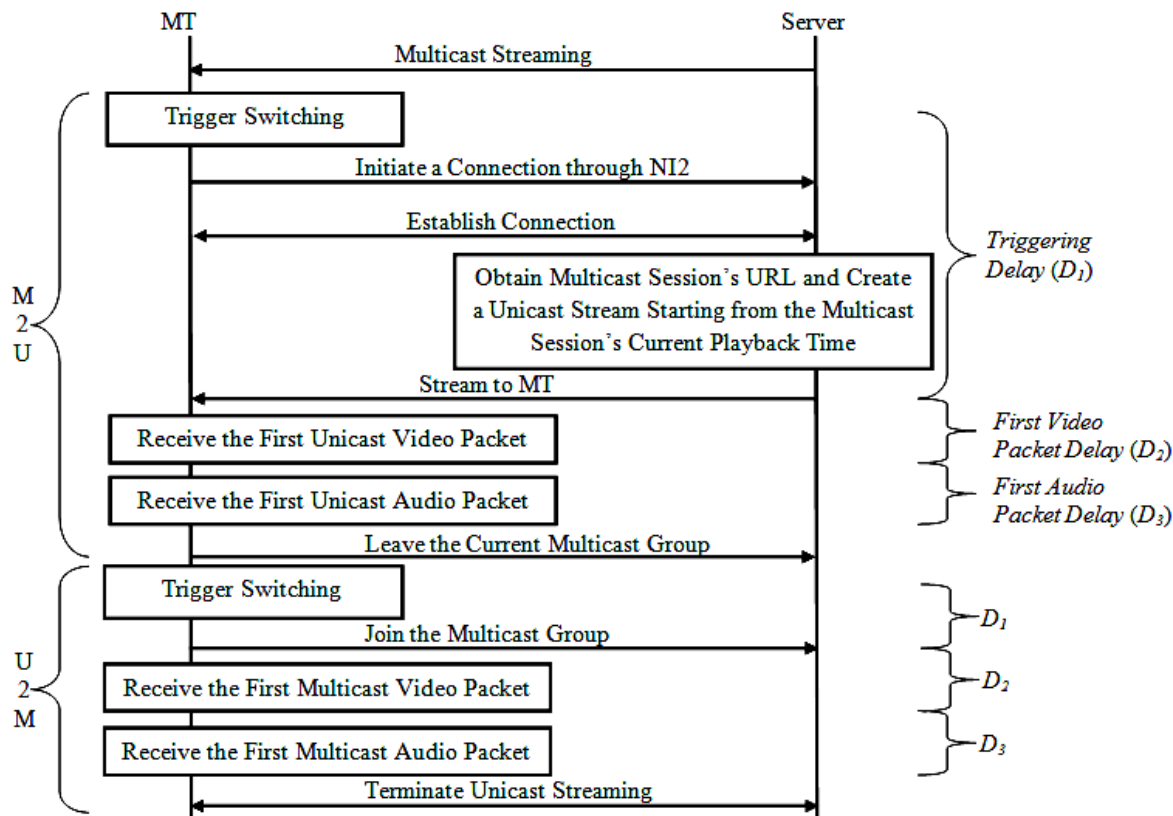


FIGURE 3. Interaction between the ALSS client and server

2.2. Switching interaction. Figure 3 describes the flow of switching connection. Two phases were considered, namely the Multicast-to-Unicast (M2U) and the Unicast-to-Multicast (U2M) phase, with a scenario as follows. First, the MT connects to a network via its first Network Interface (NI1) to watch a streaming multimedia with multicast delivery. Playback commences after a sufficient amount of multimedia data (as defined by the target buffer level) has been buffered at the decoder output buffer. Whenever the switching is triggered, the MT will send a message to the server through its second Network Interface (NI2) for requesting a unicast stream. The server then retrieves the Uniform Resource Locators (URL) of the multicast session currently being played and creates a unicast stream for the same multimedia back to the client. In order to synchronize both the multicast and unicast sessions, the server accesses certain parts of the multimedia file and streams a unicast session starting from the current playback time of the multicast session.

When the unicast data packets arrive at NI2, the client will examine whether it is an audio or video stream. The duration from the switching trigger till the receiving of the first packet from NI2 is known as the *Triggering Delay* (D_1). The examination process shall continue until the first video unicast packet is captured. This is because in the process of packetizing and multiplexing the audio and video frames into a single data stream by the server, it is possible to pack several audio packets much earlier than any video packet since video packet requires longer time to process. In the absence of video packet, the client video decoder will malfunction due to video buffer underflow, resulting in the occurrence of blank screens during playback. In the worst case, re-buffering will take place until the decoder output buffer reaches its target buffer level again before any playback. The duration from the first received audio packet until the first received video packet is known as the *First Video Packet Delay* (D_2).

Upon detection of the first unicast video packet, the first and subsequent data packets are forwarded to the A/V decoder. At the same time, the client detects the first audio packet. It is important for the client to stay in the current multicast group until the next (first) unicast audio packet is received. Otherwise, the audio decoder will malfunction due to audio buffer underflow. Under such a circumstance, the video/image may freeze and will not recuperate. For this reason, the current multicast stream is still required to supply audio packets until the next unicast audio packet arrives. Such duration is defined as the *First Audio Packet Delay* (D_3). For a seamless handover experience, the handover *overlapping period* must be at least equal to the sum of D_2 and D_3 where both NI1 and NI2 are receiving data packets. Continuing our scenario, the next handover is from unicast back to multicast delivery. The client shall first connect to the alternative network by subscribing to the specific multicast group before terminating the existing unicast stream. Similar to the M2U handover, the client shall wait for the first video and first audio packet before terminating the unicast stream. The three different delays described above are labeled as in Figure 3 (on the right). These delays applied to both the M2U and U2M scenarios. The *overall handover delay* for connection handover can be defined as the sum of D_1 , D_2 and D_3 .

3. Testbed Setup. We have built a real-time streaming testbed as shown in Figure 4 to experiment with our handover approach. The testbed is generally divided into three network segments, with Network A as the server segment, and Network B and C as the client segments.

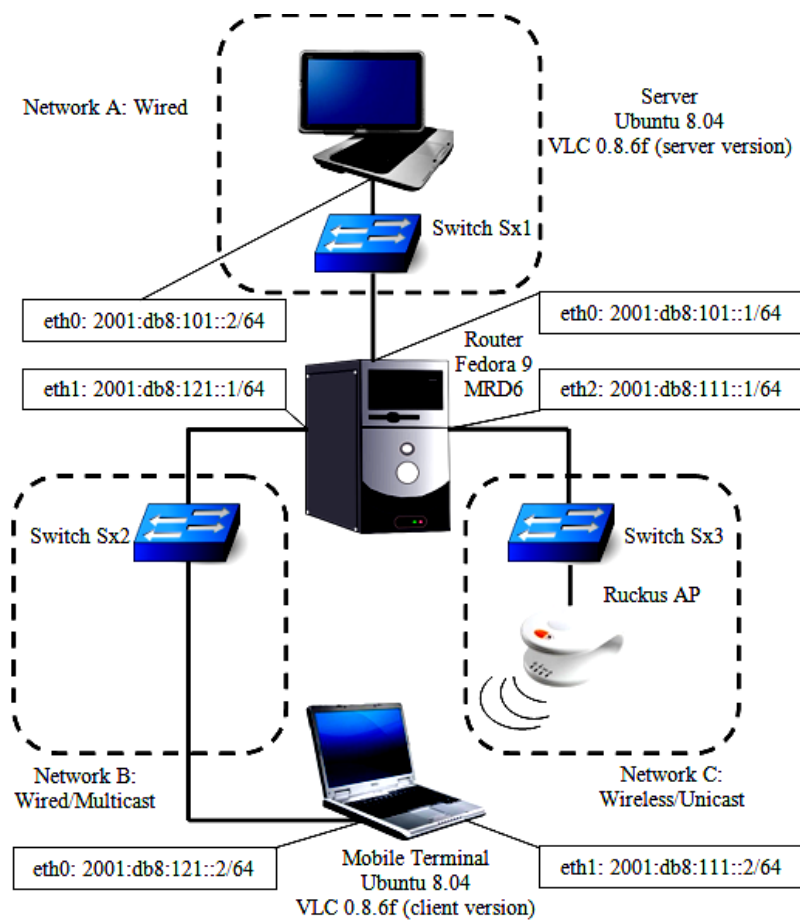


FIGURE 4. Testbed for seamless handover

Two laptops (one for server and the other for client with two network interfaces) and one desktop computer are used, where each was assigned a static IPv6 address. The Operating Systems (OS) of the two laptops are both Ubuntu 8.04. The desktop computer acting as the router is installed with MRD6 in Fedora 9 OS with three network interfaces linking all three network segments. MRD6 is an IPv6 routing daemon with multicast forwarding capabilities. The laptop serving as client is equipped with both Ethernet (802.3) and WLAN (802.11) interfaces, connecting to both Network B and C respectively. The Ethernet is chosen to mimic other cellular networks as we do not have base station equipment. Network C serves the Wi-Fi access with multicast-capable Access Point (AP) [16].

The three proposed ALSS modules (except the streaming manager) are integrated into the VLC media player (VLC) [17] of version vlc-0.8.6f to play the role of a streaming server and client. Streaming manager is implemented with tcpdump [18], which is a network monitoring tool that captures and analyzes the contents of packets on a specific network interface. This empowers us to identify audio and video packets from the new multimedia stream.

We used MPEG-2 Transport Stream (TS) for delivering multimedia contents in our prototype. MPEG-2 TS is a popular format for transmission of multimedia stream over network [19]. In MPEG, a multimedia stream typically consists of two elementary streams (audio and video streams). An MPEG encoder converts each elementary stream into its corresponding Packetized Elementary Stream (PES) packets, each carrying either an audio or a video frame. Each PES packet is further split into multiple fixed-length TS packets for transmission over the IP based network. To this aim, audio and video TS packets are multiplexed and encapsulated in the UDP packets, each carrying seven 188-byte TS packets as shown in Figure 5. The number of audio and video TS packets for a particular UDP packet is allocated in such a way that both the audio and video streams are played back in synchronization with each other.

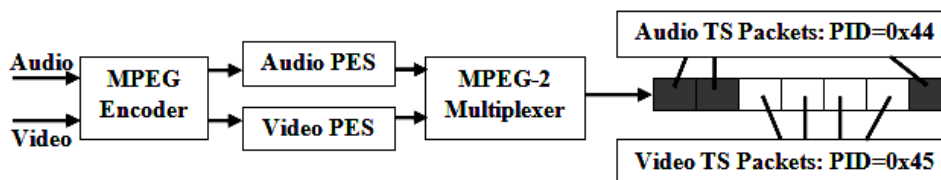


FIGURE 5. Process of MPEG-2 transport stream

At the receiver's side, the UDP payload is de-multiplexed into individual PES streams by the Packet Identifier (PID) values before forwarding them to the appropriate A/V decoder and buffer. If either of the audio and video buffers does not have enough buffered data, a buffer starvation arises at the receiver. Because a single video frame typically consumes more data packets to be displayed as compared to a single audio frame (at least in the multimedia files we experiment with), we privilege the arrival of the video packet to avoid re-buffering. Thus, it is of the utmost importance to recognize the values of D_2 and D_3 .

4. Test and Performance Evaluation.

4.1. Overview. The connection handover experiment was carried out on the testbed as shown in Figure 4 with three multimedia (A/V) files of different variable bit-rate video and constant bit-rate audio as tabulated in Table 1. In the experiment, the target buffer level was set to 300 milliseconds.

TABLE 1. Properties of three test multimedia files

Multimedia	Video Frame Rate (f/s)	Average Video Bitrates (kb/s)	Audio Frame Rate (f/s)	Audio Bitrates (kb/s)
I	15	411	38	128
II	25	770	38	112
III	30	4025	38	160

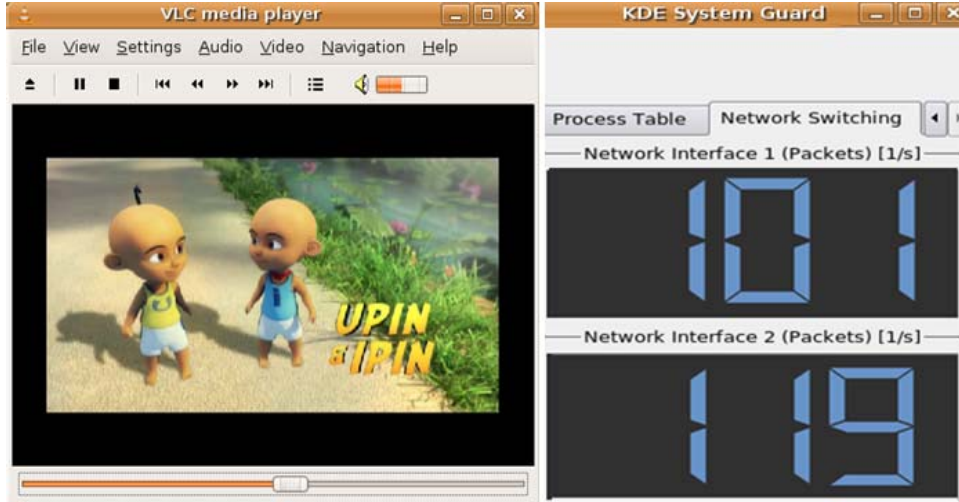


FIGURE 6. Overlapping of incoming packets during handover

We conducted visual verification as well as detailed performance measurement. Figure 6 shows the number of incoming packets from both the wired and wireless interfaces during the overlapping period in KSysGuard [20]. By monitoring the number of packets, it was verified that the multimedia stream has successfully switched from one network to the other, without any interruption on the visual playback.

The performance measurement was conducted in two phases. First, the handover delay of M2U and U2M was studied. Second, the streamed video quality of the M2U handover phase was further examined with the PSNR analysis.

4.2. Handover delay. Two approaches were adopted to verify and examine the efficiency of performing handover for the delivery of three different multimedia as shown in Table 1. The first approach is by putting several timestamps at different parts of the VLC source code to capture the three delay values during connection handover as explained earlier. In the second approach, a lower (network) layer approach using Wireshark [21] was adopted to further capture and verify packets that flow in and out through both the wired (eth0) and wireless (eth1) network interfaces. Different test points using these two approaches are shown in Figure 7, which is an extended version of Figure 3 with the additions of network interface and router.

In short, the various intervals between the neighboring test points defined the three different delays as explained earlier. The definition for each time interval is as follows:

- A→B: D_1
- B→C: D_2
- C→D: D_3
- F→G: D_1
- G→H: D_2
- H→I: D_3

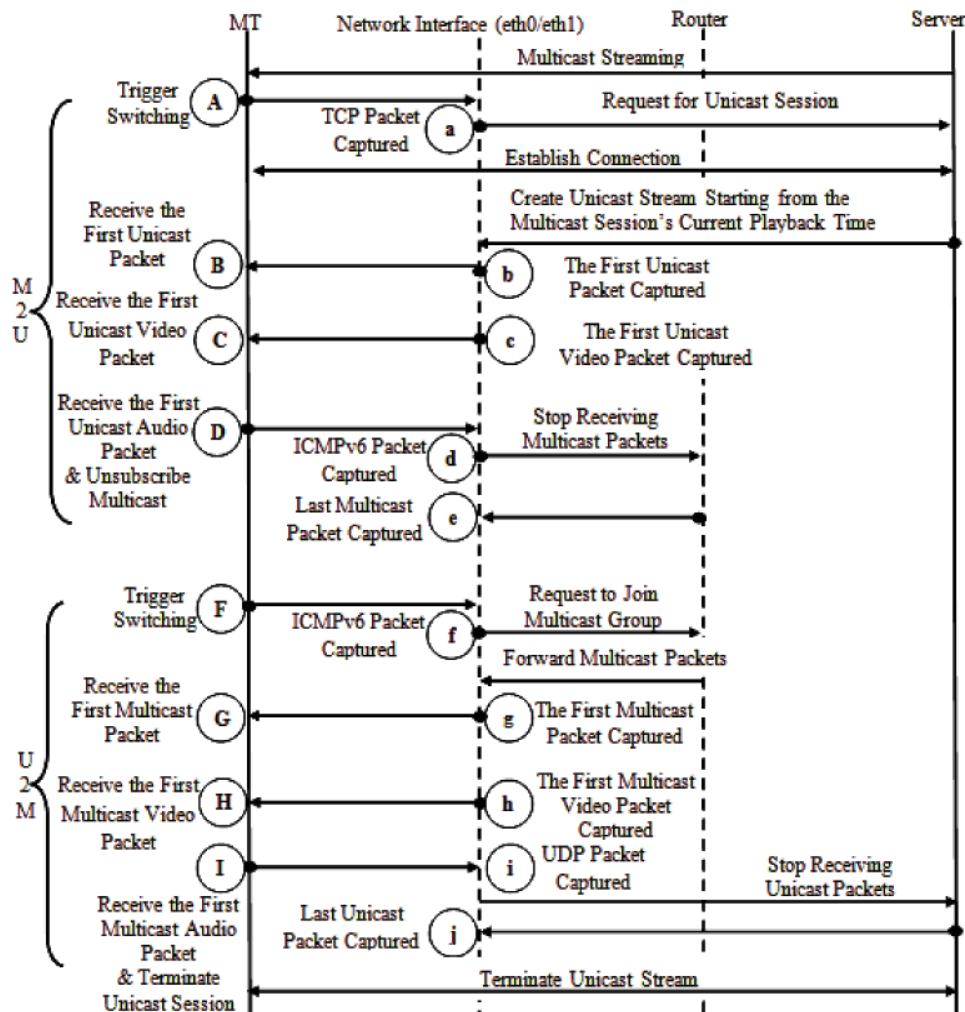


FIGURE 7. Sequence diagram for handover

In a similar fashion, we also have various test points measured with Wireshark, with addition of test points d , e , i and j . The interval of $(d \rightarrow e)$ and $(i \rightarrow j)$ refer to current stream's leave latency. In the context of M2U, the delay $(d \rightarrow e)$ is called group leave latency, which is defined as the time from the last listening nodes on a subnet to leave the group to the time when no more multicast traffic is forwarded to that subnet [22]. On the other hand, delay $(i \rightarrow j)$ is the current stream's leave latency for the U2M, which is the duration when the client requests the server to stop sending unicast stream till the last unicast packet is received.

In addition to studying the various delay values during connection handover execution, we further compared the *overall handover delay* of M2U for system with and without the use of ALSS. The execution flow for M2U without ALSS is as follows.

- 1) The server streams a multicast session to the client over the Wi-Fi AP.
- 2) After a while, the Wi-Fi AP is turned off to simulate connection breakdown.
- 3) Once the client detected that there is no more incoming packet, it then makes a unicast connection to the server over an alternative network interface.
- 4) The duration for waiting the first unicast packet (both audio and video) is then recorded.

4.3. Handover effect on streamed video. PSNR analysis was chosen to study the handover impact on the streamed video as it is often used in the literature to study video

quality [23]. In short, PSNR is calculated by comparing every pixel in the first frame of the streamed video with the corresponding pixel in the first frame of pre-encoded video, and similarly to the subsequent frames. The PSNR equation is defined as below:

$$\text{PSNR} = 10 \log_{10} \frac{(\text{MAX}_I)^2}{\text{MSE}} \quad (1)$$

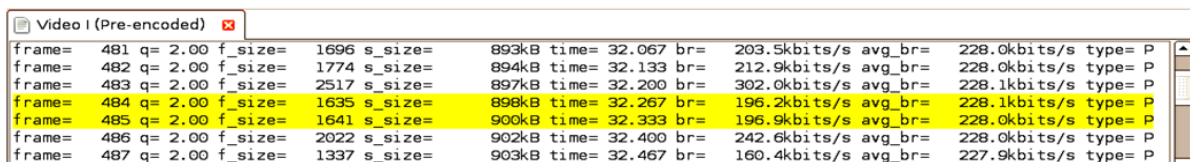
where MAX_I is the maximum luminance with the value of 255 for picture coded with 8-bit resolution; and MSE is the mean squared error. MSE is null if two compared frames are equally the same. Note that if there is no distortion, the PSNR value should be infinity according to Equation (1). For reason of simplicity, we used the same approach as in [24] to define the highest value of PSNR as 100 dB. The higher the PSNR value is, the higher the received frame quality is and the higher level of viewing satisfaction experienced by the users is.

Before studying the handover effect, we first examined whether video streaming without handover operation causes any distortion. For this purpose, experiments without handover for all three videos were conducted. The video sent from the server and the video received by the client were extracted into frames using ffmpeg [25]. Then, the PSNR was computed in sequence using ImageMagick [26]. Such PSNR results also serve as the baseline for comparison with video quality after handover that will be further described in Section 5.

For the M2U experiment, the PSNR calculation is slightly different from the method just described. The reason is that the above method assumes no skipping or redundant frames in the streamed video. However, frame losses and frame redundancy are prevalent in the case of connection handover due to imperfect synchronization between current and new multimedia streams.

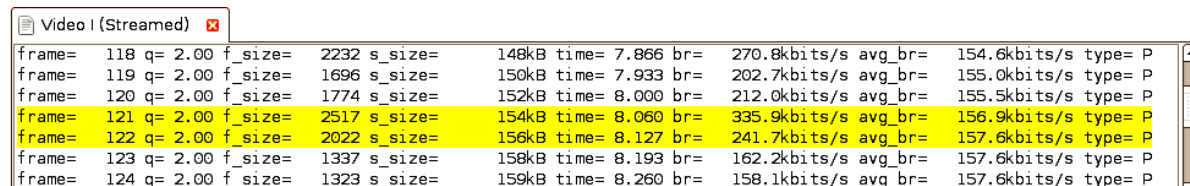
A missing or extra frame would cause frames to be in altered positions when compared with the pre-encoded video. Subsequently, the altered frame position will cause inaccurate frames to be compared in the PSNR analysis. As a result, this would cause serious degradation to the average PSNR value of the streamed video. For this reason, we have adopted the frame matching process in [27] to locate the correct frame for comparison. The key idea is to match each frame in a streamed video to a frame in the pre-encoded video so that the sums of PSNR of all frame pairs are maximized. However, the possibility of redundant frames further complicates the matching process.

We have thus applied another strategy for computing PSNR as shown in Figure 8. First, ffmpeg was used to extract video coding statistics from both the pre-encoded and



frame	q	f_size	s_size	time	br	avg_br	type
481	2.00	1696	893kB	32.067	203.5kbits/s	228.0kbits/s	P
482	2.00	1774	894kB	32.133	212.9kbits/s	228.0kbits/s	P
483	2.00	2517	897kB	32.200	302.0kbits/s	228.1kbits/s	P
484	2.00	1635	898kB	32.267	196.2kbits/s	228.1kbits/s	P
485	2.00	1641	900kB	32.333	196.9kbits/s	228.0kbits/s	P
486	2.00	2022	902kB	32.400	242.6kbits/s	228.0kbits/s	P
487	2.00	1337	903kB	32.467	160.4kbits/s	227.9kbits/s	P

(a) Pre-encoded video I



frame	q	f_size	s_size	time	br	avg_br	type
118	2.00	2232	148kB	7.866	270.8kbits/s	154.6kbits/s	P
119	2.00	1696	150kB	7.933	202.7kbits/s	155.0kbits/s	P
120	2.00	1774	152kB	8.000	212.0kbits/s	155.5kbits/s	P
121	2.00	2517	154kB	8.060	335.9kbits/s	156.9kbits/s	P
122	2.00	2022	156kB	8.127	241.7kbits/s	157.6kbits/s	P
123	2.00	1337	158kB	8.193	162.2kbits/s	157.6kbits/s	P
124	2.00	1323	159kB	8.260	158.1kbits/s	157.6kbits/s	P

(b) Streamed video I

FIGURE 8. Identifying extra and lost frame

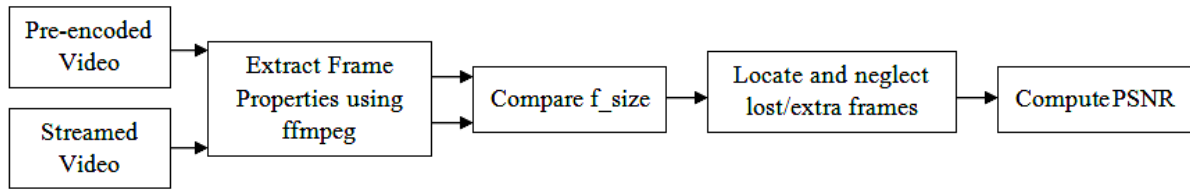


FIGURE 9. PSNR analysis for the streamed video

streamed videos and later store them into separate file. Figure 8 shows the detailed video coding statistics of each frame that consist of eight properties: frame number (frame), video quantizer scale (q), frame size (f_size), accumulated frame size (s_size), presentation time stamp (time), bitrates (br), average bitrates (avg_br) and picture type from a Group of Picture (GOP) structure (type). From the figure, it can be observed that the highlighted frames in Figure 8(a) are the lost frames as these frames were missing from Figure 8(b) by checking on the frame size. By identifying, extracting and filtering both the lost and redundant frames, the PSNR analysis can be done properly as shown in Figure 9.

5. Results and Discussions.

5.1. **Handover delay.** We have conducted ten rounds of tests to tackle the issues of deviation, and the average handover delay is reported in Figures 10, 11, 12, 13 and 14.

5.1.1. *Connection handover at application layer.* Figures 10 and 11 show the time taken to perform M2U and U2M respectively with internal timestamps in VLC source code. From

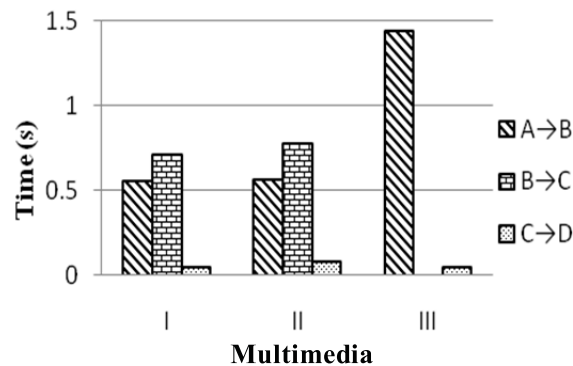


FIGURE 10. M2U at application layer

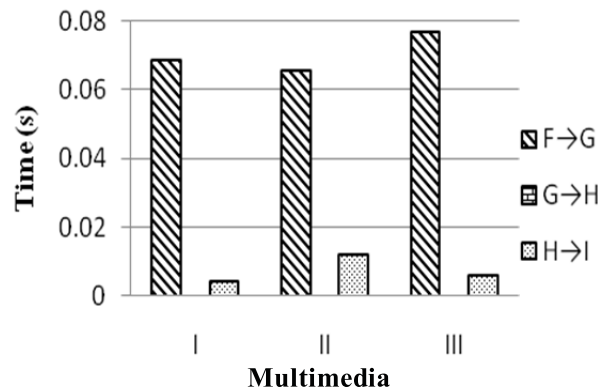


FIGURE 11. U2M at application layer

Figure 10, it can be observed that delay (A→B) is almost the same for Multimedia I and II, whereas for Multimedia III, it is one second higher. Two factors may lead to this disparity, which are the network conditions and processing capacity of the server. To pinpoint the particular reason, the Round-Trip Time (RTT) of the TCP request was determined. Test result indicated that the RTT is relatively consistent in all three scenarios, which is 0.01 second. We could then exclude the network condition as the influential factor here. The only reason is the time spent on creating unicast stream, which includes reading of multimedia file from the storage, jumping to the specific playback time frame and streaming the data to the client. In general, the larger the content size is, the longer it will take for the server to establish a unicast stream.

Figure 10 also shows a significant difference in delay (B→C) of Multimedia I and II as compared to Multimedia III. For Multimedia I and II, the unicast audio packets reach earlier than the unicast video packets by approximately 0.75 second while it is 0.003 second for Multimedia III. This may be due to the variation between audio and video frame rates. For example, based on Table 1, one second of Multimedia I contents requires 38 audio frames and 15 video frames, which implies that to display one video frame, the decoder output buffer needs $[(1/15)/(1/38)] = 2.5$ audio frames to be ready. Whereas for Multimedia III, it only needs $[(1/30)/(1/38)] = 1.3$ audio frame to be ready together with one video frame. Hence, the encoder and multiplexer could exhibit synchronization issue at the beginning when it receives a request to create a unicast session for Multimedia I and II. The consequence of this issue is that there is a silence gap for audio during playback. This is due to the design as explained in Section 2.2 that starts accepting a new multimedia stream only after the arrival of the first video packet in order to prevent re-buffering. Besides that, delay (C→D) is relatively consistent for all three multimedia, which is approximately 60 milliseconds. Apparently, Multimedia III with no synchronization issue has the least *overlapping period* ($D_2 + D_3$) which is 56 milliseconds.

As shown in Figure 11, all three multimedia have approximately the same delay (F→G) values with an average value of 0.07 second. Meanwhile, delay (G→H) is zero and delay (H→I) is consistently small for all three multimedia since the existing multicast session is a well-synchronized multimedia stream, which means the first UDP packet received by the client carries both the audio and video packets. In addition, Multimedia I has the least *overlapping period* of only 4 milliseconds.

5.1.2. *Connection handover at network layer.* Figures 12 and 13 show the time taken to perform M2U and U2M respectively with packet analysis done using Wireshark. As shown in Figure 12, it is observed that delay (a→b), (b→c) and (c→d) are consistently similar to delay (A→B), (B→C) and (C→D) respectively due to the same measured parameters. However, it should be emphasized that although delay (c→d) is almost the

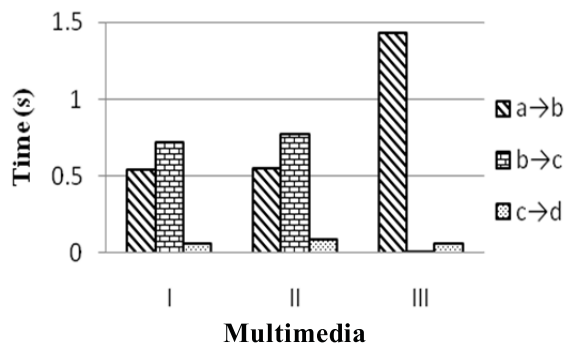


FIGURE 12. M2U at network & transport layers

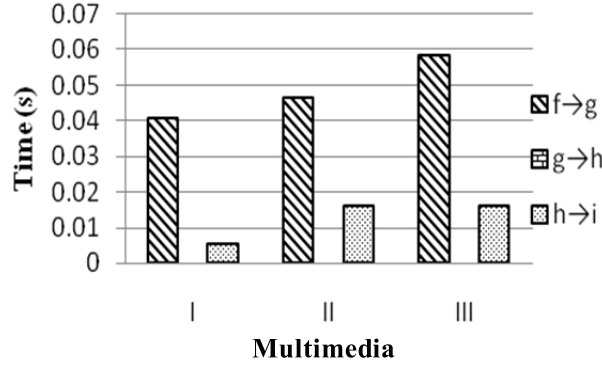


FIGURE 13. U2M at network & transport layers

TABLE 2. Total received packets during delay (c→d)

Multimedia	Number of Received UDP Packets during delay (c→d)
I	135
II	155
III	25

same for all three multimedia, there is a significant difference in the number of received packets during that interval as tabulated in Table 2. For Multimedia I and II, the number of received packets is five times more than the number of received packets for Multimedia III, which is unusual since Multimedia III contains the highest bit rate data. This is due to unsynchronized multimedia stream. As discussed earlier, the audio packets are sent earlier than the video packets for Multimedia I and II. This could lead to unsynchronized playback where only audio frame is available while the video packet is absence. In response to this, we believed that the streaming interface and multiplexer transmit the video packets of Multimedia I and II at higher data rate in order to ensure that there are no skipping of video frames due to late arrival of video packets. After that, it will resume to normal streaming rate to form a synchronized stream.

As shown in Figure 13, it is observed that delay (f→g), (g→h) and (h→i) are almost similar to delay (F→G), (H→I) and (I→J) respectively due to the same measured parameters. (f→g) is called multicast join latency which is too small compared to (a→b).

5.1.3. *Overall discussion.* As shown in Figure 14, there is only one MT in our testbed. Therefore, the MT is considered as the only (last) member on its associated subnet. When the router receives a multicast leave request, it needs 4 seconds of delay (d→e) to check if there is still any multicast subscriber in the network before it stops forwarding the multicast traffic to the subnet. On the other hand, for the U2M, the server needs 3.75 seconds of delay (i→j) to confirm whether its recipients are still accepting the unicast data packets for playback. It is important to know that packets received by the MT at these intervals (d→e) and (i→j), will not be buffered for playback. Hence, such group/session leaving delay will not contribute to the entire handover delay.

Figure 14 shows the *overall handover delay* of U2M and M2U. From the figure, it can be observed that the U2M handover is faster than the M2U handover by 1.34 seconds. This is due to the readily available and well-synchronized multicast stream. More specifically, the MT joins an already existing multicast group by requesting the router to forward data packets to it. On the contrary, a unicast stream is created upon the request from the MT.

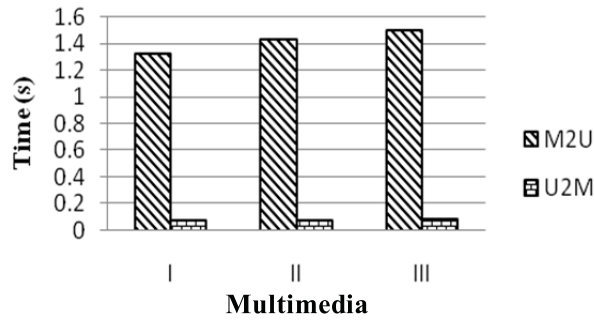
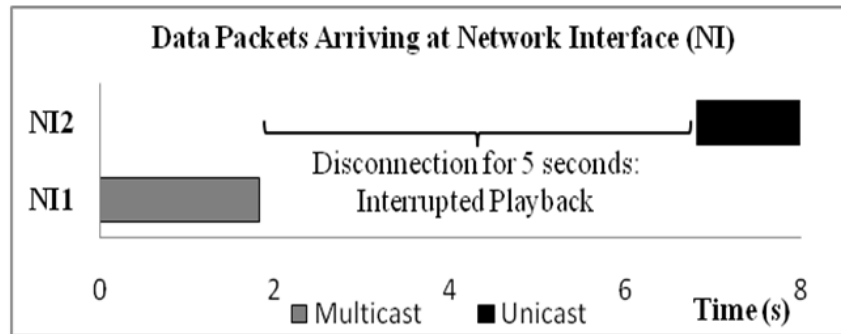
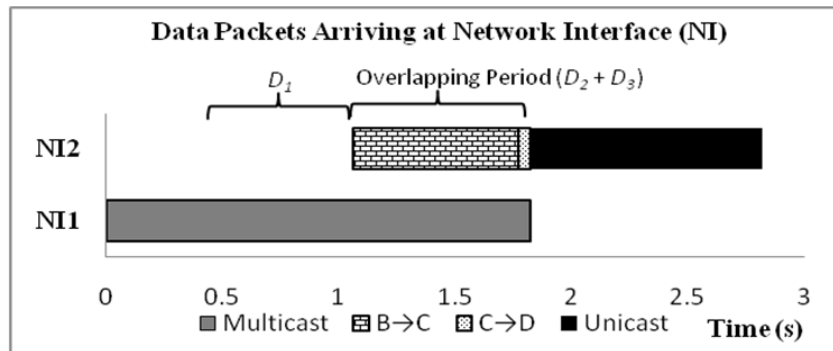


FIGURE 14. Overall handover delay for M2U and U2M



(a) Without ALSS

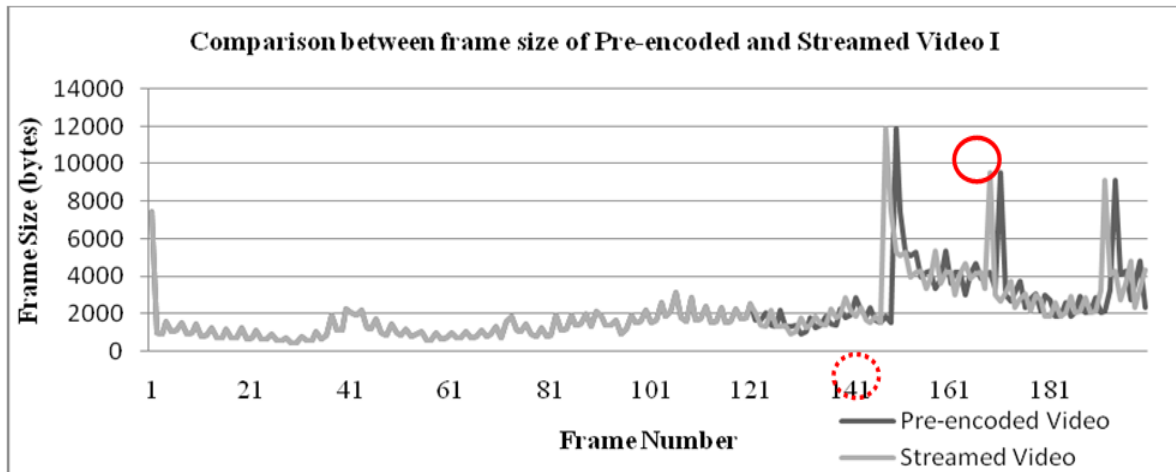


(b) With ALSS

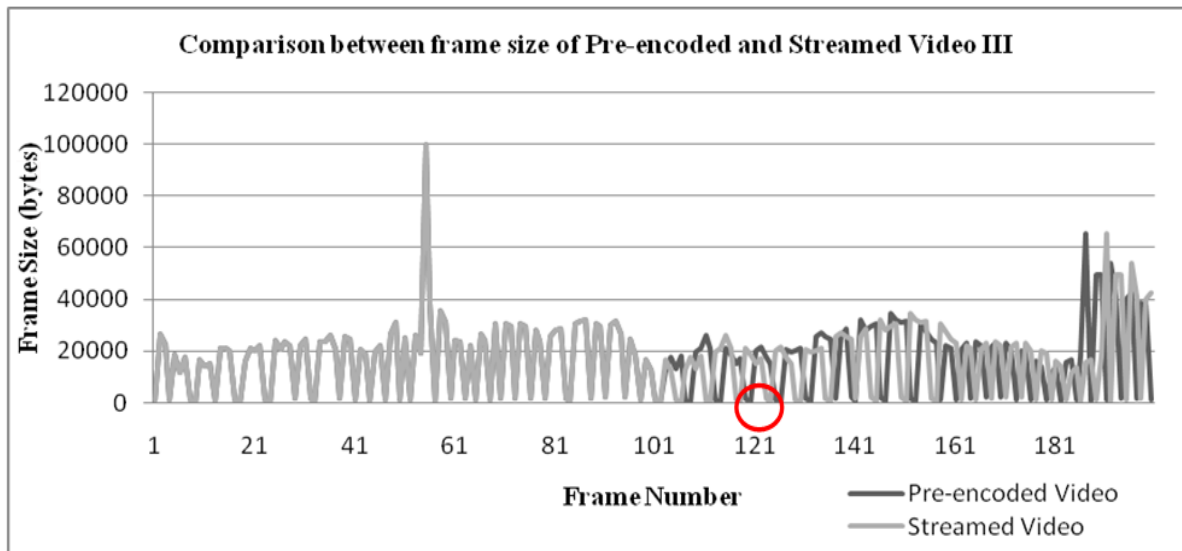
FIGURE 15. Performance comparison of M2U without and with ALSS for Multimedia I

Figure 15 compares the *overall handover delay* of M2U for system without and with the use of ALSS for the delivery of Multimedia I. As shown in Figure 15(a), a disconnection period of 5 seconds for M2U without ALSS was observed, causing underflow at the decoder output buffer, which leads to interrupted playback. With a target buffer level set to 300 milliseconds, the total waiting time is around 5.3 seconds before playback is resumed. On the contrary, with the use of ALSS, smooth playback is observed as shown in Figure 15(b) because the decoder always has sufficient audio and video frames for playback. It is important to observe that both D_2 and D_3 must take place during the overlapping period to ensure seamless handover experience.

5.2. **Handover effect on streamed video quality.** Figure 16 compares the frame size of pre-encoded and streamed videos for Video I and III. The frame number plotted here



(a) Video I



(b) Video III

FIGURE 16. Comparison between frame size of pre-encoded and streamed video

TABLE 3. Lost/duplicated frame

Video	Lost Frame	Duplicated Frame
I	2	0
II	0	1
III	0	4

refers to the frame number of the streamed video (Refer to the first column of the table in Figure 8(b)). From Figure 16(a), visual inspection of the solid circle suggests that two frames of the streamed Video I appear earlier than the pre-encoded Video I due to lost frames. By tracing backward, we found that the issue started off at frame number 122 as indicated by the dotted circle. In a similar fashion, from Figure 16(b), it can be observed that the frames of streamed Video III appear later than the pre-encoded Video III due to duplicated frames. The number of lost or duplicated frames is tabulated in Table 3.

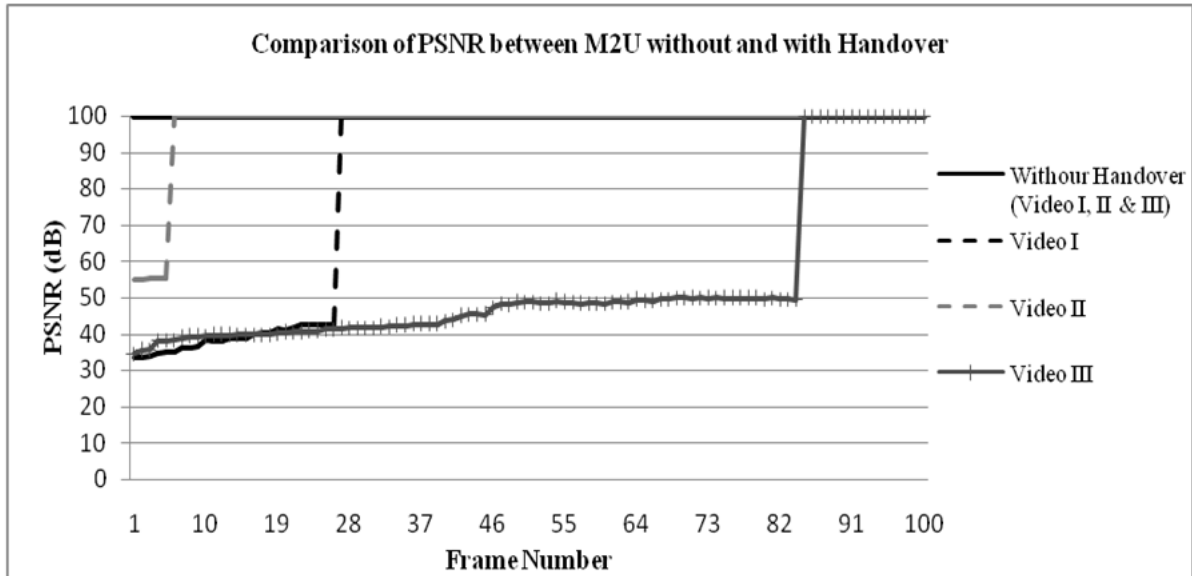


FIGURE 17. PSNR for streamed video with and without handover

Figure 17 displays the PSNR value of all three videos with and without M2U handover for selected 100 frames, which shows significant PSNR differences. As expected, for all three videos without handover, the resulting average PSNR value is 100 dB. This means that there are no distortions in any frame of these videos. On the contrary, if there is any frame pair returning a non 100 dB, it is the distortion caused by the handover.

For each streamed video, the first frame pair returning PSNR of non 100 dB has been selected as the starting point of the graph. From the figure, it can be observed that Video II obtains the highest average PSNR among all videos at 97.76 dB with 5 distorted frames. The average PSNR for Video I is 84.02 dB with 26 distorted frames while the average PSNR for Video III declines by 36.3% as compared to Video I with the highest number of distorted frames. Obviously, this indicates that Video III has the lowest QoS during connection handover. The reason lies in the nature of predictive video coding technique, more specifically, the Group of Pictures (GOP) structure which results in error propagation [28]. A GOP is a group of successive frames reflecting spatial motion activities in video shots. It always begins with I-frame which does not require any additional information to reconstruct it and then P-frame which requires the prior decoding of preceding I- or P-frame in order to be decoded. Therefore, if an error occurs within a GOP (e.g., due to frame lost), the error will propagate till the next reference picture as shown in Figure 18.

To verify the stated reason, ffmpeg was again used to find the GOP length for each streamed video as tabulated in Table 4. In this paper, the GOP length refers to the distance (in number of frames) between the first frame after lost or duplicated frames and the next I-frame. For example, by observing Figures 8(b) and 19, the first frames after lost frames and the next I-frame for Video I are frame 122 and frame 148 respectively. Hence, the GOP length will be $148 - 122 = 26$, which is the same as the number of frame returning non 100 dB as shown in Figure 17. This means that the MT must wait for 26 future frames to arrive before a possible correction with the next I-frame is received.

The above experiments and results have shown that ALSS helps to maintain QoS experience during connection handover. By mapping the observed PSNR (33.5 dB) to the Mean Opinion Score (MOS) as recommended by ITU-T in Table 5 [29], it is further

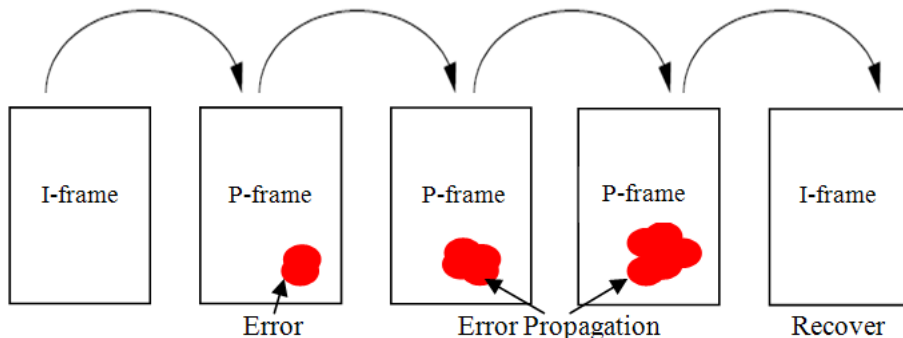


FIGURE 18. Error propagation at video decoder

TABLE 4. GOP length of three test videos

Video	GOP Length
I	26
II	5
III	84

frame=	q=	f_size=	s_size=	time=	br=	avg_br=	type=
146	2.00	1823	195kB	9.726	217.9kbits/s	164.4kbits/s	P
147	2.00	1506	197kB	9.793	180.0kbits/s	164.5kbits/s	P
148	2.00	11886	208kB	9.860	1420.7kbits/s	173.0kbits/s	I
149	2.00	7424	215kB	9.926	900.8kbits/s	177.8kbits/s	P

FIGURE 19. Next I-frame for streamed Video I

TABLE 5. Possible PSNR to MOS mappings

MOS	PSNR (dB)	Comments
1	$\text{dB} < 20$	Bad
2	$20 \leq \text{dB} < 25$	Poor
3	$25 \leq \text{dB} < 31$	Fair
4	$31 \leq \text{dB} < 37$	Good
5	$\text{dB} \geq 37$	Excellent

indicated that the quality of the streamed video is in fact at the good level during the handover process.

6. Conclusions. This paper has presented a connection handover subsystem called ALSS that provide smooth multimedia delivery across multicast and unicast-enabled network. We have described the ALSS architecture, the switching interaction as well as the prototype implementation details and testbed of our work. The results showed that the overlapping period for M2U and U2M handover took a minimum of 56 and 4 milliseconds respectively, and the quality of received video frame during handover was categorized as good based on ITU-T recommendation.

Some of the future work includes managing the stream multiplexer so as to allocate a well-mixed multimedia stream consisting of both video and audio packets for immediate playback at MT, with the objective of minimizing the time for waiting a mixed multimedia stream. Next is to get the server to properly estimate the current video playback time of the MT so that the new connection to be established would start streaming at the right playback time with fewer overlapping packets for processing at the MT. We

are also interested in integrating our work with smart triggering algorithm and network selection approaches for a complete handover system. Our work continues, and we invite collaboration towards our system development effort.

Acknowledgements. This project was supported by Introduction of High-Level Talents and Overseas Returnees Scientific Fund in Nanjing Forestry University (No.: GXL015.) and Natural Science Fund for Colleges and Universities in Jiangsu Province (No.: 16KJB510016).

REFERENCES

- [1] B. Hilt, M. Sarni and P. Lorenz, Enhancement of channel switching scenario and IGMPv3 protocol implementation in multicast IPTV networks, *International Journal on Advances in Networks and Services*, vol.2, nos.2-3, pp.167-181, 2009.
- [2] P. Namburi, K. Sarac and K. Almeroth, Practical utilities for monitoring multicast service availability, *Computer Communications*, vol.29, no.10, pp.1675-1686, 2006.
- [3] C. Lee, S. Kim and S. Kang, Design of overlay multicast protocol supporting mobility and its implementation, *The 9th International Conference on Optical Internet*, Shilla Jeju, Korea, pp.1-3, 2010.
- [4] A. Pitsillides and C. Christophorou, MBMS handover control: A new approach for efficient handover in MBMS enabled 3G cellular networks, *Computer Networks*, vol.51, no.18, pp.4897-4918, 2007.
- [5] Y.-H. Xu, C.-O. Chow, G.-X. Kok and M.-L. Tham, A load-aware scheme for providing heterogeneous multimedia broadcast/multicast service (Het-MBMS), *Wireless Personal Communications*, vol.77, no.3, pp.1-24, 2014.
- [6] B. Åström and P. Edlund, *Switching between Delivery Methods in an IPTV Communication Network*, United States Patent WO/2009/140994, 2014.
- [7] T. Lohmar, Z. Peng and P. Mahonen, Performance evaluation of a file repair procedure based on a combination of MBMS and unicast bearers, *International Symposium on a World of Wireless Mobile and Multimedia Networks*, pp.1-9, 2006.
- [8] Y. Xu, J. Song, J. Li, G.-X. Kok and K. Utsu, Load-aware based independent information services for heterogeneous handover, *International Journal of Innovative Computing, Information and Control*, vol.12, no.1, pp.243-262, 2016.
- [9] C. K. Lee, S. H. Kim and S. G. Kang, Mobility management of mobile node in relay-based overlay multicast, *Proc. of the 11th International Conference on Advanced Communication Technology*, pp.391-394, 2009.
- [10] G. Cunningham, P. Perry, J. Murphy and L. Murphy, Seamless handover of IPTV streams in a wireless LAN network, *IEEE Trans. Broadcasting*, vol.55, no.4, pp.796-801, 2009.
- [11] G. H. Liaw, C. F. Lee and J. L. Liu, A seamless playback mechanism for mobile multicast streaming video in IEEE 802.11-based wireless mesh networks, *Proc. of the 24th International Conference on Advanced Information Networking and Applications Workshops*, pp.926-931, 2010.
- [12] G. Cunningham, P. Perry, J. Murphy et al., Seamless handover of IPTV streams in a wireless LAN network, *IEEE Trans. Broadcast*, vol.55, no.4, pp.796-801, 2009.
- [13] E. Gustafsson and A. Jonsson, Always best connected, *IEEE Wireless Communications*, vol.10, no.1, pp.49-55, 2003.
- [14] S. Balasubramaniam and J. Indulska, Vertical handover supporting pervasive computing in future wireless networks, *Computer Communications*, vol.27, no.8, pp.708-709, 2004.
- [15] *MRD6*, <http://fivebits.net/proj/mrd6/>.
- [16] *Ruckus Security*, <http://www.ruckussecurity.com/ZoneFlex-2925.asp>.
- [17] *VideoLAN*, <http://www.videolan.org/vlc/>.
- [18] *Tcpdump*, <http://www.tcpdump.org/>.
- [19] *Information Technology – Generic Coding of Moving Pictures and Associated Audio Information: Systems*, ISO/IEC 13 818-1, 1996.
- [20] *Klik*, <http://ksysguard.klik.atekon.de/>.
- [21] *Wireshark*, <http://www.wireshark.org/>.
- [22] *Telecommunications and Internet Converged Services and Protocol for Advanced Networking*, http://docbox.etsi.org/TISPAN/Open/NGN_LATEST_DRAFTS/RELEASE3/03208-ngn-r3v320.pdf.

- [23] A. Belhouli, Y. A. Sekercioglu and N. Mani, Mobility-aware RSVP: A framework for improving the performance of multimedia services over wireless IP-based mobile networks, *Computer Communications*, vol.32, no.4, pp.569-582, 2009.
- [24] U. Engelke, T. M. Kusuma and H. J. Zepernick, Perceptual quality assessment of wireless video applications, *Proc. of the 6th International ITG-Conference on Source and Channel Coding*, 2006.
- [25] *FFmpeg*, <http://www.ffmpeg.org/>.
- [26] *ImageMagick*, <http://www.imagemagick.org/script/index.php>.
- [27] A. Chan, K. Zeng, P. Mohapatra, S. J. Lee and S. Banerjee, Metrics for evaluating video streaming quality in lossy IEEE 802.11 wireless networks, *Proc. of the IEEE International Conference on Computer Communications*, pp.1-9, 2010.
- [28] C. Kiraly, L. Abeni and R. L. Cigno, Effects of P2P streaming on video quality, *Proc. of the IEEE International Conference on Communications*, pp.1-5, 2010.
- [29] *ITU-T Recommendation P.910*, <http://www.videoclarity.com/PDF/T-REC-P.910-199909-I!!PDF-E%5B1%5D.pdf>.