Design and Optimization considerations for real-time video conferencing using IMS in 4G/LTE Networks

Rajeta Meshram, Rituja Srivastava, Aswathy A, Punugu Anjanadri, Suja S, Charumati P
Centre for Development of Telematics, India
rajetam@cdot.in

Abstract—**IP Multimedia Subsystem (IMS) offers a plethora of functionalities in 4G/LTE networks. One such functionality is the video conference where three or more User Equipments (UEs) communicate with each other using VoLTE. This research paper presents optimizations in the design and implementation of an IMS video conference server for reducing the video display delay and improving the performance thereof. Both application as well as network level parameters are found to impact the video display delay at the UE, hence performance improvement of both are considered in this paper. Optimization techniques adopted in this implementation resulted in enhanced performance of video conference calls over the LTE network.**

Index Terms*—*IMS, LTE, 4G, video conference, optimization

I. INTRODUCTION

IMS is defined by 3GPP, comprising core network elements to provide multimedia services [1]. In order to provide the functionalities, a leveled and layered architecture approach is followed in the design of IMS. The architecture can be divided into three major layers: (1) Access Layer (2) Core Layer (3) Service Layer [2] as shown in Fig 1.

Access layer contains the networks that allow access of services to UE. Core layer includes the entities responsible for regulating communication flow. Main elements of the core layer include : 1) Call Session Control Functions (CSCF) can be categorized as Proxy-CSCF (P-CSCF), InterrogatingCSCF (I-CSCF) and Serving-CSCF (S-CSCF). 2) Multimedia Resource Function (MRF) performs various media processing functions for real-time communications. Service layer contains the Application Servers (AS) responsible for delivering services to the end users.

Our design proposes a ConfMedia server as a part of the service layer of IMS architecture, which provides video conferencing service. HyperText Transfer Protocol (HTTP/2) is used to receive requests related to create, delete or modify the conference call. These messages are also used to forward the session details required for the media processing using Session Description Protocol (SDP). Real Time Protocol (RTP) is used for media communication between UEs and the ConfMedia server.

Every video conferencing solution architecture is based on sending and receiving video streams of all the participants. There are three methods in which this can be implemented : 1) Peer-to-peer (P2P) also referred as mesh architecture in which each user is broadcasting its media to all the peers through a direct link, 2) Selective Forwarding Unit (SFU) is a centralized solution in which all the participants send media to a SFU server, which then redistributes these media streams to all participants unaltered and 3) Multipoint Control Unit (MCU) in which all the participants send media to a MCU server which then mixes all media streams into one stream and send back to the participants.

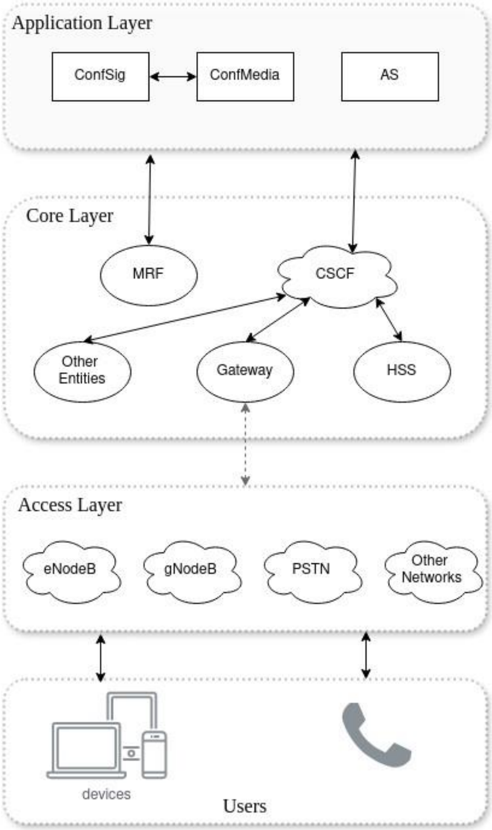


Fig. 1. IMS Architecture

Our paper differs from the existing works by focusing on the design of the ConfMedia server, built on a centralized architecture, specifically tailored for video conferencing in IMS-LTE networks. While prior research explored distributed multimedia conferencing solution for IMS-based LTE networks using eNodeB as a proxy server solutions [3], centralized IMS architecture focusing on SIP signaling between key components like CSCF, MRF, and UE [4] and use of the ”E-Model” (ITU-T G.107) as an optimization tool to select network and voice parameters like coding scheme, packet loss limitations, and link utilization level to maximize QoS for VoIP calls [5], our approach addresses real-time video conferencing challenges with an emphasis on optimizing the encoding process, reducing delay, and ensuring efficient media synchronization. This distinct focus on video conferencing sets our work apart from the existing studies.

Rest of the paper is divided into following sections : Section II gives a detailed design of the ConfMedia server. Section III explains the issues faced with the design and how the optimization is done to handle live media streams. Section IV presents test results before and after optimization of ConfMedia server. These tests are performed on a live 4G network. Section V concludes the paper with scope for future improvements.

II. DESIGN OF VIDEO CONFERENCE SERVER

ConfMedia implementation, hosted on a general purpose Linux based server, comprises four modules as shown in Fig 2 1) Request Handling Module receives Create, Add, Modify and Delete requests from the signaling module and extracts SDP message from the request, 2) SDP Handling Module (SdpMod) receives the SDP message, parses and stores the conference related information, 3) Conference Participant DataBase is where the conference related information is stored for active conference calls and 4) Media Handling Module (MediaMod) combines received media and sends it to all participants.

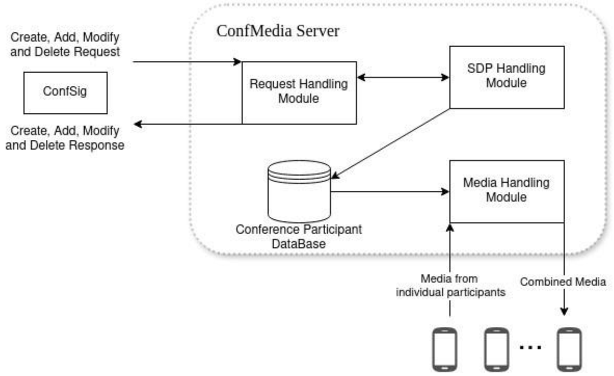

ConfMedia Server
Create, Add, Modify and Delete Request
ConfSig
Create, Add, Modify and Delete Response
Request Handling Module
SDP Handling Module
Media Handling Module
Conference Participant DataBase
Media from individual participants
Combined Media

Fig. 2. ConfMedia Design

This paper focuses on implementation and optimization of the MediaMod module using Gstreamer, a versatile and comprehensive open source multimedia framework designed to handle various media processing tasks. It supports a wide range of multimedia formats and provides a pipeline-based architecture for constructing complex media processing workflows useful for media streaming, and conferencing[6]. Media travels from the source element to the sink element passing through different elements performing tasks.

MediaMod implements several functionalities to support video conference calls. Fig 3 shows the essential elements required in a pipeline to process audio and video from the UE. Codec information (eg; AMR-NB for audio and H264 for video) are negotiated with the UE using SDP messages and are processed by SdpMod. *Audio-n* and *Video-n* are the source elements for audio and video to receive the incoming UDP data from *User-n*. *RTP Depayload* elements extract the RTP packets from the UDP data. *Audio Decoder* and *Video Decoder* elements decode the audio and video RTP packets to raw data. *Audio Mixer* and *Video Compositor* are the elements that perform the task of mixing individual audio and video streams to make combined audio and video streams. The combined audio and video are encoded using *Audio Encoder* and *Video Encoder* respectively. *RTP Payload* element adds the RTP header to the encoded streams. Finally the combined RTP audio and video is sent to *User-n* through *AudioSink-n* and *VideoSink-n* sink elements respectively.

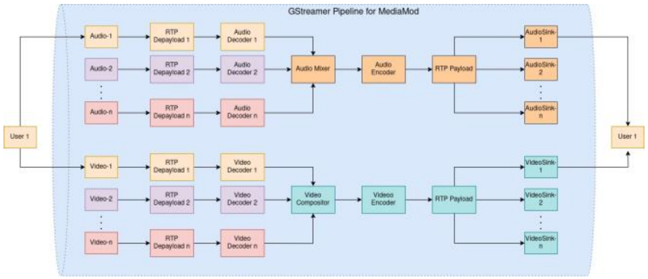


Fig. 3. MediaMod Gstreamer Pipeline Structure for Video Conferencing

III. OPTIMIZATION CONSIDERATIONS

The proposed design and implementation of MediaMod resulted in an unsatisfactory video quality for UEs when tested with LTE network, leading to a suboptimal user experience. Fine tuning of the pipeline elements is crucial for achieving the expected quality output, given that the LTE network is very sensitive towards bandwidth and latency.

In this section, we present the optimization performed on various elements for a smooth video conferencing experience at the UE. We consider the factors contributing to the overall performance of a video conference call in a 4G network. Performance of a video conference call is found to depend on the type of UE used, eNodeB capabilities, properties of encoder, handling of different signals like call hold/resume from UE and the platform on which ConfMedia server is hosted.

Following are the main challenges and the optimizations performed on MediaMod:

1. Inherent delay added by the *Video Encoder* (Encoding Delay) causes an overall delay in combined video display at the UE. In a live media, delay refers to the difference between the time at which an event occurs and when it is displayed at the UE. To address this, the *speed preset* property of Video Encoder which affects the encoding capabilities is set to a value optimal for the LTE network.

2. Audio and video RTP streams from UE arrive separately and are processed separately. After processing the mixed audio stream and the composited video stream are sent separately to the UEs. Due to this separation, RTP packets for audio and video may become unsynchronized. To address this, clock synchronization is applied to the media pipeline, ensuring that both audio and video streams remain synchronized.

3. In an ongoing conference call with N participants, adding (N+1)th participant should be seamless. However a perceivable delay in video display is consistently observed in that participant. Study revealed that the parameter set packets, which are a fundamental part of the video codec and important for the decoding process, are missed in the network due to timing and synchronization mismatch [7]. MediaMod sends these parameter sets periodically and if lost or erroneously transmitted, the

(N+1)th UE has to wait for the next to decode the video streams leading to the delay in display at (N+1)th participant. Adjusting the parameter set property of the video encoder to match the 4G network conditions reduces this delay and improves video synchronization in new participants.

4. During a conference when any participant is on hold, it does not process the incoming conference data sent by MediaMod. Later, as the participant resumes the conference call there is bound to be a discontinuity between the last RTP packet sequence number and the current RTP packet sequence number. Also, sending data to the participant in the HOLD state causes load on the network resulting in performance degradation. As an optimization, MediaMod stops sending output media to the participant for the hold duration. After the call is resumed by the participant, output media is resumed from the MediaMod with the next sequence number.
5. Video encoders have settings that speed up video processing. However, when these settings are applied, the number of packets sent from MediaMod increases. This puts extra strain on the eNodeB, which leads to potential network instability. The additional load can make it harder to establish new calls or maintain a stable connection, affecting the overall performance of the network. To address network instability, speed up video processing setting is disabled in Video Encoder.

IV. TESTS AND RESULTS

To evaluate proposed optimization a test setup is created as shown in Fig 4. This test setup includes eNodeB, EPC, IMS, ConfMedia Server and UEs. For all the test cases the procedure followed is : Originator referred as *party1* initiates the conference call and adds multiple participants like *party2, party3...partyN* to the call. At any given instant, the total participants in a call is considered as *N*. The new participant *party(N+1)* is added to the call by *party1*.

We have established several test cases to assess the optimizations applied to the base design. Each test case is conducted with multiple trials to ensure consistent and reliable results across varying scenarios.

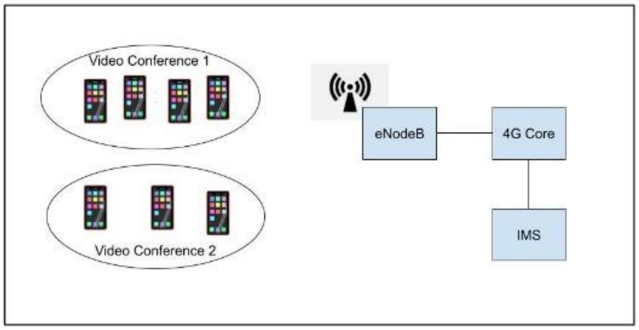


Fig. 4. Test Setup

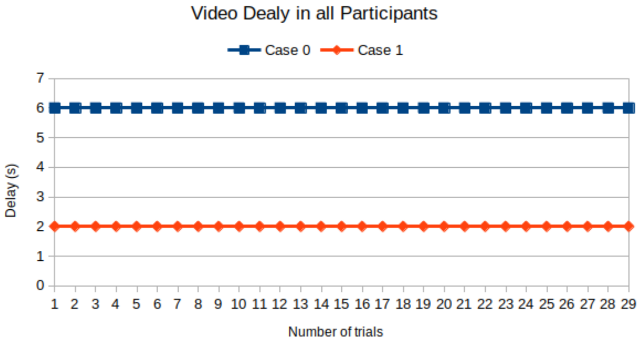


Fig. 5. Video delay observed in Case 0 and Case 1

Case 0: ConfMedia implementation without the proposed optimization.

Case 1: Speed preset property set in *Video Encoder*. Fig 5 shows the encoding delay in Case 0 and Case 1. The result shows that the delay is less for Case 1 as compared to Case 0. Reducing the delay further causes degradation in the quality of the combined video.

Case 2: Clock Sync for the pipeline.

Our primary goal is to ensure that audio and video buffers have matching timestamps for the same event. To achieve proper synchronization between composite audio and video streams when adding a new participant to the conference, a clock synchronization API is applied. Without this, mixed audio is heard before the composited video due to the inherent encoding delay, as observed in Case 1. The sync API ensures that both streams maintain matching timestamps, keeping them synchronized throughout the session.

Case 3: Parameter set value of *Video Encoder*.

Fig 6 shows the time taken to view the composited video of the conference at party(N+1) for Case 0. It ranges from 9-50 seconds with an exception of one trial. Fig 7 shows the time taken to view the combined video on party(N+1) for Case 3.

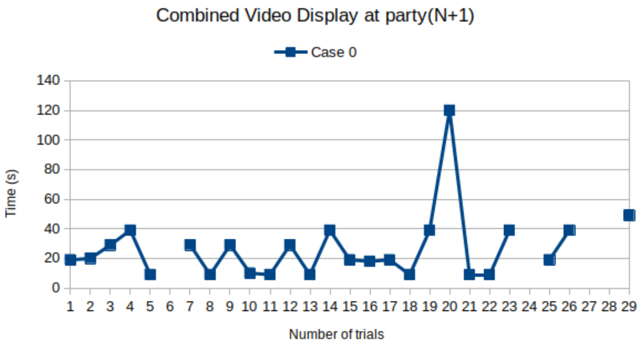


Fig. 6. Combined video display at *party(N+1)* for Case 0

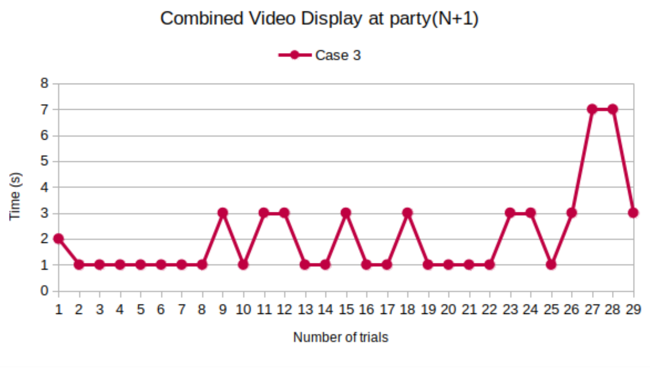


Fig. 7. Combined video display at *party(N+1)* for Case 3

It ranges from 1-3 seconds, with an exception of two trials, which is an improvement from Case 0.

Fig 8 shows the time taken to view the composited video of the conference at party1 for Case 0. It ranges from 9-29 seconds with an exception of one trial. Fig 9 shows the time taken to view the composited video of the conference at party1 for Case 3. It ranges from 1-3 seconds, with an exception of one trial, which is an improvement from Case 0.

Case 4: Stop data to participant when it is in HOLD state, send data when participant resumes.

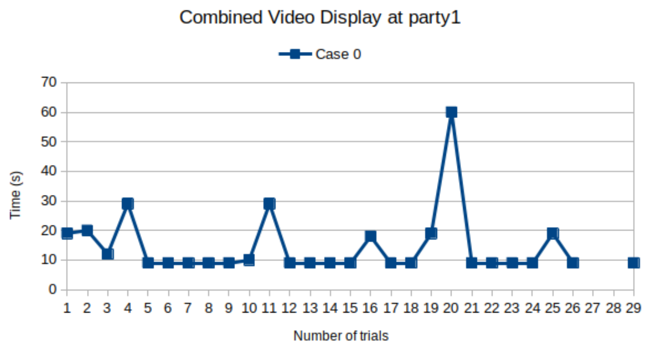


Fig. 8. Combined video display at *party1* for Case 0

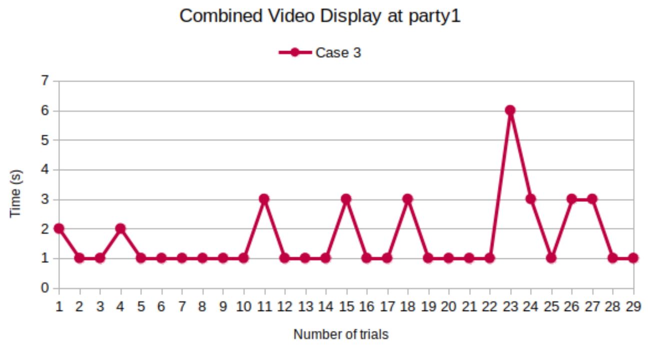
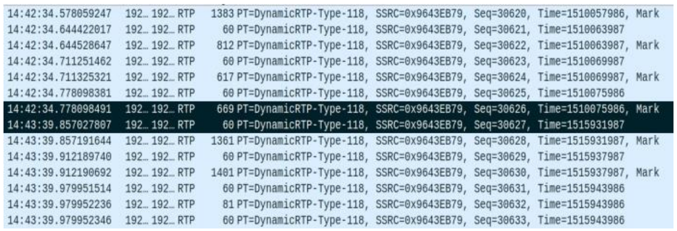
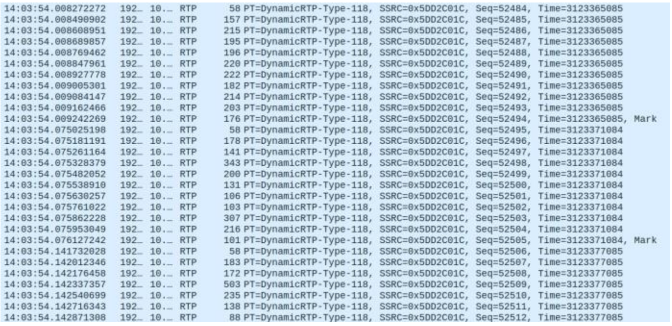


Fig. 9. Combined video display at *party1* for Case 3



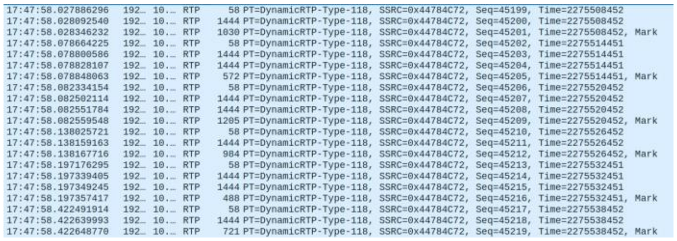
```
14:42:34.578059247   192… 192… RTP   1383 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30620, Time=1510057986, Mark
14:42:34.644422017   192… 192… RTP     60 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30621, Time=1510063987
14:42:34.644528647   192… 192… RTP    812 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30622, Time=1510063987, Mark
14:42:34.711251462   192… 192… RTP     60 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30623, Time=1510069987
14:42:34.711325321   192… 192… RTP    617 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30624, Time=1510069987, Mark
14:42:34.778098381   192… 192… RTP     60 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30625, Time=1510075986
14:42:34.778098491   192… 192… RTP    669 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30626, Time=1510075986, Mark
14:43:39.857027807   192… 192… RTP     60 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30627, Time=1515931987
14:43:39.857191644   192… 192… RTP   1361 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30628, Time=1515931987, Mark
14:43:39.912189740   192… 192… RTP     60 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30629, Time=1515937987
14:43:39.912190692   192… 192… RTP   1401 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30630, Time=1515937987, Mark
14:43:39.979951514   192… 192… RTP     60 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30631, Time=1515943986
14:43:39.979952236   192… 192… RTP     81 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30632, Time=1515943986
14:43:39.979952346   192… 192… RTP     60 PT=DynamicRTP-Type-118, SSRC=0x9643EB79, Seq=30633, Time=1515943986
```

Fig. 10. Combined media stream sent to UE for Case 4



```
14:03:54.008272272   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52484, Time=3123365085
14:03:54.008490902   192…  10.…  RTP    157 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52485, Time=3123365085
14:03:54.008608951   192…  10.…  RTP    215 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52486, Time=3123365085
14:03:54.008689857   192…  10.…  RTP    195 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52487, Time=3123365085
14:03:54.008769462   192…  10.…  RTP    196 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52488, Time=3123365085
14:03:54.008847961   192…  10.…  RTP    220 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52489, Time=3123365085
14:03:54.008927778   192…  10.…  RTP    222 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52490, Time=3123365085
14:03:54.009005301   192…  10.…  RTP    182 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52491, Time=3123365085
14:03:54.009084147   192…  10.…  RTP    214 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52492, Time=3123365085
14:03:54.009162466   192…  10.…  RTP    203 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52493, Time=3123365085
14:03:54.009242269   192…  10.…  RTP    176 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52494, Time=3123365085, Mark
14:03:54.075025198   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52495, Time=3123371084
14:03:54.075181191   192…  10.…  RTP    178 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52496, Time=3123371084
14:03:54.075261164   192…  10.…  RTP    141 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52497, Time=3123371084
14:03:54.075328379   192…  10.…  RTP    343 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52498, Time=3123371084
14:03:54.075482052   192…  10.…  RTP    200 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52499, Time=3123371084
14:03:54.075538910   192…  10.…  RTP    131 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52500, Time=3123371084
14:03:54.075630257   192…  10.…  RTP    106 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52501, Time=3123371084
14:03:54.075761022   192…  10.…  RTP    103 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52502, Time=3123371084
14:03:54.075862228   192…  10.…  RTP    307 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52503, Time=3123371084
14:03:54.075953049   192…  10.…  RTP    216 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52504, Time=3123371084
14:03:54.076127242   192…  10.…  RTP    101 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52505, Time=3123371084, Mark
14:03:54.141732028   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52506, Time=3123377085
14:03:54.142012346   192…  10.…  RTP    183 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52507, Time=3123377085
14:03:54.142176458   192…  10.…  RTP    172 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52508, Time=3123377085
14:03:54.142337357   192…  10.…  RTP    503 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52509, Time=3123377085
14:03:54.142540699   192…  10.…  RTP    235 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52510, Time=3123377085
14:03:54.142716343   192…  10.…  RTP    138 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52511, Time=3123377085
14:03:54.142871308   192…  10.…  RTP     88 PT=DynamicRTP-Type-118, SSRC=0x5DD2C01C, Seq=52512, Time=3123377085
```

Fig. 11. Combined Video stream sent to UE with video processing speed up enabled (Case 5)



```
17:47:58.027886296   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45199, Time=2275508452
17:47:58.028092540   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45200, Time=2275508452
17:47:58.028346232   192…  10.…  RTP   1030 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45201, Time=2275508452, Mark
17:47:58.078664225   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45202, Time=2275514451
17:47:58.078800586   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45203, Time=2275514451
17:47:58.078828107   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45204, Time=2275514451
17:47:58.078848063   192…  10.…  RTP    572 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45205, Time=2275514451, Mark
17:47:58.082334154   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45206, Time=2275520452
17:47:58.082502114   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45207, Time=2275520452
17:47:58.082551784   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45208, Time=2275520452
17:47:58.082559548   192…  10.…  RTP   1205 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45209, Time=2275520452, Mark
17:47:58.138025721   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45210, Time=2275526452
17:47:58.138159163   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45211, Time=2275526452
17:47:58.138167716   192…  10.…  RTP    984 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45212, Time=2275526452, Mark
17:47:58.197176295   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45213, Time=2275532451
17:47:58.197339405   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45214, Time=2275532451
17:47:58.197349245   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45215, Time=2275532451
17:47:58.197357417   192…  10.…  RTP    488 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45216, Time=2275532451, Mark
17:47:58.422491914   192…  10.…  RTP     58 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45217, Time=2275538452
17:47:58.422639993   192…  10.…  RTP   1444 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45218, Time=2275538452
17:47:58.422648770   192…  10.…  RTP    721 PT=DynamicRTP-Type-118, SSRC=0x44784C72, Seq=45219, Time=2275538452, Mark
```

Fig. 12. Combined Video stream sent to UE with video processing speed up

Fig 10 shows the combined media stream sent to UE. UE paused the conference call at 14:42:34 and resumed the call at 14:43:39. During this time MediaMod did not send output media to the UE in HOLD state as mentioned in Case 4. After the call is resumed by the UE, output media is resumed from the MediaMod with the next sequence number 30627.Case 5: Video processing speed up setting disabled in *Video Encoder*. Fig 11 and Fig 12 refers to Case 5 and is summarized in Table I. With the *speed up video processing* setting enabled, 165 packets are transmitted in one minute, with each packet ranging in size from 103 to 503 bytes. When the setting is disabled, only 61 packets are sent in one minute, with packet sizes ranging from 572 to 1444 bytes. This indicates disabling *speed up video processing* reduces the number of packets to 1/3rd thereby reducing load on the network.

TABLE I. RESULTSFOR CASE 5

Parameters observed	Speed up enabled	Speed up disabled
Number of packets in 1 min	165	61
Length of each packet	103-503	572-1444

V. CONCLUSION

The research led to significant improvement in video conference efficiency in LTE networks. Encoding delay reduced from 6 seconds to 2 seconds achieved without compromising video quality. Video delay on (N+1)th participant decreased from 9 seconds to within 3 seconds. By disabling the *speed up video processing* reduced the load on the network by one third. These findings indicate that the algorithm is highly suitable for bandwidth-constrained video conferencing scenarios in 4G networks.

Future improvements to the system could involve the implementation of a feedback mechanism that dynamically adjusts encoding settings based on real-time network conditions. Additionally, integrating advanced machine learning algorithms could further refine the system's ability to predict and respond to network congestion, ensuring smooth media quality across diverse network environments, especially in 4G and 5G networks. To further enhance the performance of the proposed design, ConfMedia server can be hosted on platforms optimized for video processing.

REFERENCES

[1] 3rd Generation Partnership Project, ”IP Multimedia Subsystem (IMS), Stage 2”.

[2] Shih-Wen Hsu, Chi-Yuan Chen, Kai-Di Chang, Han-Chieh Chao, and Jiann-Liang Chen, ”Towards Service-oriented Cognitive Networks over IP Multimedia Subsystems, ” The 17th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2011), Tainan, Taiwan, December 7-9, 2011.

[3] Tien Anh Le, Hang Nguyen, Noel Crespi, “IMS-based distributed multimedia conferencing service for LTE” in 2012 IEEE Wireless Communications and Networking Conference (WCNC).

[4] Weiwei Lai, Jian Guo, Yuliang Tang, “The design of multimedia conference system based on IMS” in 2011 IEEE 3rd International Conference on Communication Software and Networks.

[5] Wagdy A. Aziz, Salwa H. Elramly, Magdy M. Ibrahim, “VoIP Quality Optimization in IP-Multimedia Subsystem (IMS)” in 2010 Second International Conference on Computational Intelligence, Modelling and Simulation.

[6] Gstreamer open source multimedia framework. https://gstreamer.freedesktop.org/

[7] RFC 6184 - RTP Payload Format for H.264 Video - Parameter Set

[8] Considerations. https://datatracker.ietf.org/doc/html/rfc6184