TEM Journal. Volume 11, Issue 2, pages 611-619, ISSN 2217-8309, DOI: 10.18421/TEM112-15, May 2022.

Content Aware Video Streaming with MPEG DASH Technology

Ecem İren [1], Aylin Kantarci [2]

[1] *Computer Programming Department, İzmir Kavram Vocational School, İzmir, Turkey*
[2] *Department of Computer Engineering, Ege University, İzmir, Turkey*

***Abstract* – Streaming has become one of the most significant topics in recent years. Network conditions may change from time to time and this affects quality of communication negatively. MPEG DASH, which means dynamic adaptive streaming over HTTP has been developed to adapt communication to environmental conditions and maintain quality. In this study, a system is developed with MPEG DASH to reduce bandwidth usage of network. Video is separated to parts as detailed and non-detailed visually. While detailed scenes are streamed in high quality, less detailed ones are streamed in low quality. Encoding rates are decided by considering SSIM, and suitable bitrate values are selected. Streaming of content is performed with Bitmovin Player. Performance of model is analyzed through bandwidth metric in Wireshark Packet Tracer with a conclusion that there is 35% reduction in bandwidth utilization.**

***Keywords* – Content Aware Streaming, MPEG DASH, Adaptive Streaming, Bandwidth Consumption, Video Compression.**

DOI: 10.18421/TEM112-15
https://doi.org/10.18421/TEM112-15

Corresponding author: Ecem İren,
Computer Programming Department, İzmir Kavram Vocational School, İzmir, Turkey.
Email: ecem.iren@kavram.edu.tr

Received: 28 January 2022.
Revised: 15 April 2022.
Accepted: 20 April 2022.
Published: 27 May 2022.

1. Introduction

According to the CISCO Visual Networking Index, video traffic, which constitutes 75% of the total Internet traffic in 2017, is expected to reach 82% by 2022. Based on this numerical data efficient distribution of video content to the viewers has a great importance [1]. For this reason, video streaming technologies have become one of the most important issues of the Internet today. Companies working in this field strive to enhance and improve video transmission by paying attention to some metrics such as performance, cost and usefulness [2]. Basic HTTP streaming systems are not considered suitable for mobile environments including severe bandwidth fluctuations since users cannot adapt to changing network conditions. As a consequence, video quality affects, from this situation, negatively causing some stallings from time to time [3]. In order to prevent these events, traditional HTTP streaming systems have been replaced by HTTP adaptive systems. In the HTTP adaptive streaming mechanism, users have full control over the stream and can adjust the quality of transmitted media particles within the differing bandwidth. Video interruptions are prevented by sending media particles in a quality level compatible with the state of the current network [4].

In adaptive video streaming techniques, the source video in the content provider is compressed with different quality values and these qualities are represented with different display layers. When the video traffic is added to the traditional data traffic during the playback, the network capacity may become insufficient and a congestion may occur. Although traditional streaming systems provide adaptation by reducing video quality, this method is not a recommended solution in terms of viewers because this kind of adaptation can cause the loss of important details which disrupts the quality of the video leading the annoying of the watchers. Therefore, high bandwidth providing high quality has become mandatory to protect video details.

Limited user systems and network resources bring many problems. Operating systems may fail to manage large amounts of data or delays can affect the quality. Limited bandwidth increases the risk of network congestion and affects video quality. In order to prevent this, it is necessary to isolate both traffic from each other and a special channel capacity should be allocated to the flow. However, there is no mechanism providing bandwidth allocation on the Internet. In this context, the streaming application can control the bandwidth consumption by fixing the transmission rate at a certain value. There are some mechanisms to facilitate bandwidth management in local area networks (LAN). For example, multiple traffics in virtual LANs. can be separated from each other [5].

Zach and Slanina encoded different kinds of videos with fixed and adaptive group of pictures (GOP) lengths which can be adapted depending on the scene changes. Instead of compressing videos with a constant bitrate, CRF (Constant Rate Factor) is used and video is encoded with a bitrate decided by the encoder according to the content detail. After streaming experiments, longer sized segments behave efficiently by offering 11% bitrate savings in adaptive GOP length and maintain high quality compared to short sized segments [6]. Kantarcı proposed a system that provides content aware streaming of medical video for bandwidth management. The system prevents possible congestion by keeping the bandwidth consumption below the capacity allocated to the network. Segmentation is done according to detail density. Streaming is achieved with Real Time (RTP) and the system helped reducing both the congestion in the network and the risk of packet loss [5]. Curcio et al. conducted a study to reduce bandwidth usage in video experiences acquired using virtual reality technology. When watching video with virtual reality, some parts of the video can be perceived and seen by the user. While detectable parts are categorized as foreground, imperceptible parts are labelled as background. The non-detectable parts were encoded at the lowest level quality and high resolution according to the SNR quality metric, and the flow was obtained. Bandwidth usage is greatly reduced with this method [7]. Adzic et al. developed an algorithm which detects the scene changes and inserts minimum number of I frames at boundaries of each GOP sequence of media for equalizing segment duration and bitrate savings. Algorithm works based on the statistical information of current GOP including bit spent and number of frames. If those values are greater than determined threshold values, it is supposed to occur a scene change and I frame is placed to the beginning of that GOP. When video sequences are encoded with H.264 encoder with this manner, savings in bitrate could be observed from 10% to 15% with reduced network complexity (bandwidth usage) [8]. Hassan et al. examined the performance of varying segment sizes in terms of user download efficiency and processor usage with MPEG DASH. As a result of the tests, it has been determined that the short-term segments are more stable in terms of output efficiency and can continue the flow compared to the long-term segments. When controlled by the CPU consumption percentage, it is understood that long-term segments occupy the processor less than short-term segments. The main source of this result is based on the lower frequency of long-dimensional segments being requested from the server and transferred over HTTP [9]. Kreuzberger et al. compared the performance of the bitrates recommended by YouTube, Apple and Netflix and the optimal bitrates offered in the study in terms of metrics such as user satisfaction and video size transmitted per second. Compression processes were carried out by considering the SSIM values of different screen resolutions of the videos. According to the SSIM results for each resolution, the most appropriate bitrate value was decided. As a result, when the transmission is made with optimized bitrates, user satisfaction is at the top and the amount of bits transferred per second to the receiver is higher. Beside this, active users can benefit more from the communication channel when the bitrate is low [10].

In this research, a server based streaming design has been proposed as a solution to the problem mentioned above with the purpose of not to increase bandwidth consumption of the communication channel. By using MPEG DASH streaming technology, while high detailed video scenes are transmitted in high bitrate, low detailed scenes are sent in low quality level which is tolerable and not recognized by human eye. Thus, during low quality video transmission, less bandwidth is consumed in the network reducing the possibility of network congestion. Finally, a performance analysis is carried out and the findings have shown that the system is useful for decreasing bandwidth usage.

The main contribution and difference of this study unlike others is that we take into account SSIM quality metric while encoding the video and quality is not compromised in terms of human sensation while reducing the encoding rate in non-detailed parts. In addition, we analyze the results in terms of bandwidth usage by examining the performance of the system.

In this article, Section 2 introduces MPEG DASH technology. In Section 3, image quality techniques are described. Section 4 gives details about the method and material. Section 5 includes experimental results of experiments. The last section summarizes the study and concludes the paper with future research.

2. MPEG DASH Technology

In the past, many difficulties have arisen at maintaining video quality with existing technologies in the video industry depending on the transmission conditions. For this reason, MPEG DASH (MPEG Dynamic Adaptive Standard), which is an adaptive and dynamic video streaming method over HTTP, is developed and this technology is referred by almost all companies due to the benefits it has. Advances in multimedia compression and communication technologies have enabled video to be delivered in high quality format on various platforms. For example, high quality video transmission has a great importance especially in the field of telemedicine since medical video carries sensitive information. The MPEG-DASH standard is defined as a framework or structure that realizes adaptive video streaming on different transmission platforms. This standard is designed to distribute high quality multimedia data under varying bandwidth conditions. Therefore, it is considered as a suitable solution especially for mobile communications [11]. It has HTTP and transmission control protocol (TCP) support and this protocol is a widely used in Internet networks for web browsers. Therefore, there is no need to make any extra changes in network nodes and video content can be accessed through any web browser. Moreover, the firewalls allow HTTP/TCP traffics to pass through. Since the video has different versions of representations, the adaptation logic is allowed to be applied by the user. Figure 1 shows the simple flow between the HTTP server and the DASH user. In this mechanism, the multimedia content is prepared and stored on an HTTP server and then the same content is distributed using HTTP [11], [12].

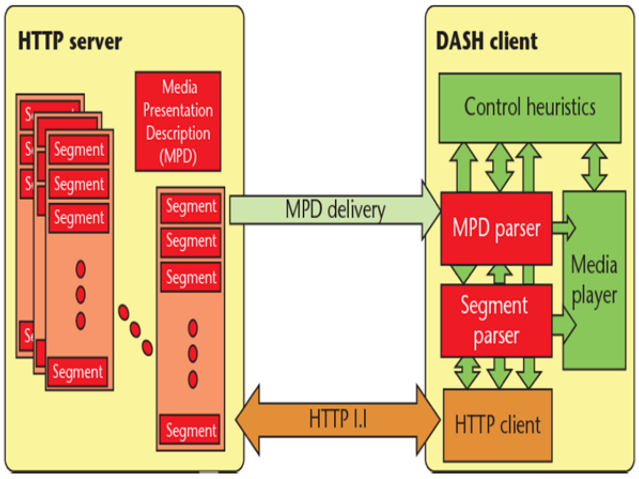


Figure 1. MPEG DASH Mechanism [13]

This content is available on the server in 2 parts: Media Presentation Description (MPD) and segments (parts). Video is fragmented into multiple segments and different alternatives of the content are described in the media presentation description file. At first, DASH user has to access MPD in order to play the media content and MPD can be distributed via HTTP, email and other transportation methods. MPD consists of adaptation sets which can be seen in Figure 2.

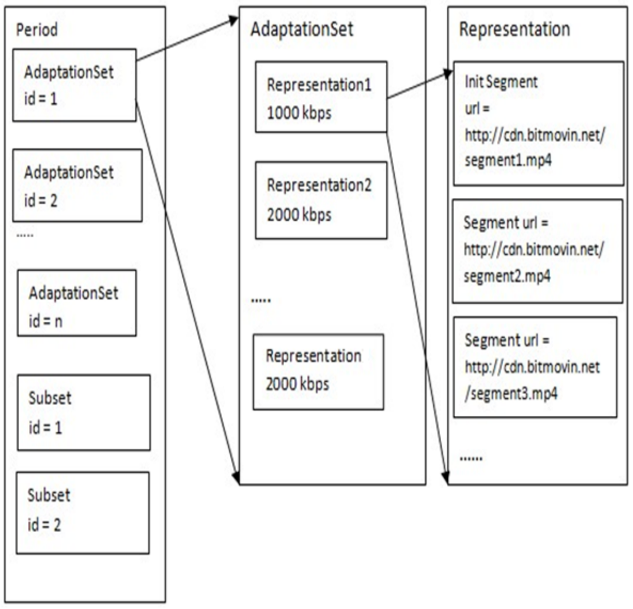

Period
AdaptationSet id = 1
AdaptationSet id = 2
.....
AdaptationSet id = n
Subset id = 1
Subset id = 2
AdaptationSet
Representation1 1000 kbps
Representation2 2000 kbps
.....
Representation 2000 kbps
Representation
Init Segment url = http://cdn.bitmovin.net/segment1.mp4
Segment url = http://cdn.bitmovin.net/segment2.mp4
Segment url = http://cdn.bitmovin.net/segment3.mp4
......

Figure 2. MPD Structure [12]

Streaming can be structured as in the form of one or more period in MPD file. Generally, media components such as video, audio or subtitles are arranged in sets of adaptations. Each period may contain one or more sets of adaptations that allow grouping of different multimedia components which are logically similar. For example, components with the same resolution, language or audio channel format can be collected in the same adaptation set. In addition, this mechanism gives clients the ability to eliminate a number of multimedia elements which are not compatible with their requirements. Adaptation sets includes some groups containing different versions of the respective video content in terms of different bit and resolution. This structure enables users to adapt the media stream under multiple conditions. In addition, it guarantees smooth playback under network variations and bandwidth requirements. By parsing MPD, the user can obtain information about the video content, resolution, media formats and encoded alternatives of multimedia components shown in Figure 3. Based on this information, the DASH user chooses the most appropriate encoding alternative and uses HTTP GET requests and fetches segments to stream the content. The user also observes network bandwidth fluctuations while continuing to fetch [11], [12], [13]. Segments allow for changes between views during video playback and they are shown with specific URLs. They usually have same length in terms of time and are arranged according to the media representation timeline which is essential for smooth transition and synchronization [14], [15].

```
<!-- MPD file Generated with GPAC version 0.7.0-rev0-
gbd5c9af-master at 2020-05-02T17:12:56.406Z-->
<MPD xmlns="urn:mpeg:dash:schema:mpd:2011"
minBufferTime="PT1.500S" type="static"
mediaPresentationDuration="PT0H18M"
maxSegmentDuration="PT0H0M2.000S"
profiles="urn:mpeg:dash:profile:full:2011">
<Period id="1" duration="PT0H3M56.125S" >
<AdaptationSet   segmentAlignment="true"   maxWidth="640"
maxHeight="360" maxFrameRate="24" par="16:9" lang="und">
<Representation id="1" mimeType="video/mp4"
codecs="avc3.64081E" width="640" height="360"
frameRate="24" sar="1:1" startWithSAP="1"
bandwidth="3200000">
<SegmentList timescale="24000" duration="48000">
<Initialization sourceURL="segment_3200_init.mp4"/>
<SegmentURL media="segment_3200_1.m4s"/>
<SegmentURL media="segment_3200_2.m4s"/>
<SegmentURL media="segment_3200_3.m4s"/>
</SegmentList>
</Representation>
</AdaptationSet>

<AdaptationSet segmentAlignment="true" lang="eng">
<Representation id="1" mimeType="audio/mp4"
codecs="mp4a.40.2" audioSamplingRate="44100"
startWithSAP="1" bandwidth="130436">
<AudioChannelConfiguration
schemeIdUri="urn:mpeg:dash:23003:3:audio_channel_configur
ation:2011" value="2"/>
<SegmentList timescale="44100" duration="176400">
<Initialization sourceURL="segment_2_init.mp4"/>
<SegmentURL media="segment_1_1.m4s"/>
<SegmentURL media="segment_1_2.m4s"/>
<SegmentURL media="segment_1_3.m4s"/>
</Representation>
</AdaptationSet>
</Period>

<Period id="2" duration="PT0H13M37.750S" > <AdaptationSet
segmentAlignment="true" maxWidth="1280" maxHeight="720"
maxFrameRate="24" par="16:9" lang="und"> <Representation
id="1" mimeType="video/mp4" codecs="avc3.64081F"
width="1280" height="720" frameRate="24" sar="1:1"
startWithSAP="1" bandwidth="6400000">
<SegmentList timescale="24000" duration="48000">
<Initialization sourceURL="segment_init.mp4"/>
<SegmentURL media="segment_6400_1.m4s"/> <SegmentURL
media="segment_6400_2.m4s"/>
<SegmentURL media="segment_6400_3.m4s"/><SegmentURL
media="segment_6400_4.m4s"/><SegmentURL
media="segment_6400_5.m4s"/>
</SegmentList>
</Representation>
</AdaptationSet>

<AdaptationSet segmentAlignment="true" lang="und">
<Representation id="1" mimeType="audio/mp4"
codecs="mp4a.40.2" audioSamplingRate="44100"
startWithSAP="1" bandwidth="131511">
<AudioChannelConfiguration schemeIdUri="urn:
mpeg:dash:23003:3:audio_channel_configuration:2011"
value="2"/>
<SegmentList timescale="44100" duration="88200">
<Initialization sourceURL="segment_1_init.mp4"/>
<SegmentURL media="segment_2_1.m4s"/>
<SegmentURL media="segment_2_2.m4s"/>
<SegmentURL media="segment_2_3.m4s"/>
</SegmentList>
</Representation>
</AdaptationSet>
</Period>
</MPD>
```

Figure 3. Extended Markup Language (XML) Representation of a sample MPD [14]

3. Image Quality Measurement Techniques

Image quality can be categorized as two main classes as objective and subjective quality. While subjective quality methods are based on human perception, objective quality methods are related with some mathematical calculations. Objective quality methods are associated with the concepts in the vision system and designed according to principles of it. Frequently used objective quality methods can be described as PSNR and SSIM.

MSE (Mean Square Error): It is a widely used image quality metric like PSNR and calculated as seen in the Equation (1). The values of MSE close to 0 indicate that the quality of the image is in a good level [16]. While I and I0 show the input and output images, N and M symbolizes the number of rows and columns of the pixel matrix. MSE value is obtained by calculating the squares of the differences of luma values, adding each difference of column and again adding each sum of row in the pixel matrix.

$$MSE = \frac{1}{NM}\sum_1^N \sum_1^M (I - I_0)^2 \qquad (1)$$

PSNR (Peak Signal to Noise Ratio): One of the most important factors that directly affect image quality can be said as noise. PSNR is a quality measure which is created to measure how the noises occurring after lossy compression of the images are recognized by human perception. Moreover, when the PSNR value is higher, it means image quality becomes better and consequently, we get lower noise level in the image. While calculating PSNR, an input image is taken as a reference and compared with an output image by getting the square root of MSE to measure the amount of the noise. The formula representing the PSNR is shown in the Equation (2). While I0 represents the input image (original), I represents the distorted image [17].

$$PSNR = 20\; x\; log_{10}(255\sqrt{MSE(I, I_0)} \qquad (2)$$

SSIM (Structural Similarity Index Measurement): Frame based structural similarity between two images is measured by structural similarity index. SSIM compares three different properties called as brightness, contrast and structure of both original and the distorted image. SSIM can be qualified as an alternative method used in image quality analysis. Since the structural information differences between the frames are calculated this quality metric takes into account quality deterioration directly. On the other hand, there is a weighting manner in PSNR and MSE. The changes in luminance and or contrast in pixels which are not perceived by the human eye may lead high weighting result while deciding deterioration of the image. Since SSIM considers the structural change of a pixel pair, it is more possible to asses them correctly. Thus, this is the reason why SSIM is chosen as quality metric in the study. The SSIM value is expressed functionally as in the Equation (3).

$$\mathrm{SSIM}(I_R, I_D) = f(l(I_R, I_D), c(I_R, I_D), s(I_R, I_D)) \qquad (3)$$

Based on the equation, while $l(I_R,I_D)$ represents the luminance function, $c(I_R, I_D)$ and $s(I_R, I_D)$ show contrast and structure functions respectively.

$$l(I_R, I_D) = \frac{2\mu_{IR}\mu_{ID} + C_1}{\mu_{IR}^2 + \mu_{ID}^2 + C_1} \qquad (4)$$

$$c(I_R, I_D) = \frac{2\alpha_{IR}\alpha_{ID} + C_2}{\alpha_{IR}^2 + \alpha_{ID}^2 + C_2} \qquad (5)$$

$$s(I_R, I_D) = \frac{\sigma_{\mathrm{IRID}} + C_3}{\sigma_{IR}\,\sigma_{ID} + C_3}$$

(6)

When the 3 equations above are multiplied with each other finally we get SSIM result in the form of the equation as below.

$$\frac{[2\mu_{\mathrm{IR}}(n)\mu_{\mathrm{ID}}(n) + C_1][2\sigma_{\mathrm{IR}}\mu_{\mathrm{ID}}(n) + C_2]}{[\mu_{\mathrm{IR}}^2(n) + \mu_{\mathrm{ID}}^2(n) + C_1][\sigma_{\mathrm{IR}}^2(n) + \sigma_{\mathrm{ID}}^2(n) + C_2]} \qquad (7)$$

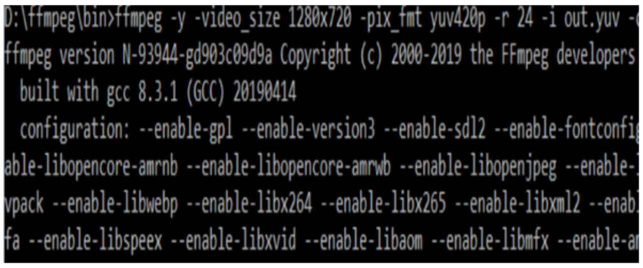
The pixel density averages of the n_{th} frame in the reference (I_R) and distorted (I_D) are expressed with the symbols (pixel) $\mu I_R\,(n)$ and μI_D (n) respectively. On the other hand, $\sigma I_R\,(n)$ and (n) indicate the pixel density standard deviation. Besides, c_1, c_2 and c_3 are used when denominator is close to zero. As a result; mean structural similarity index measurement is also calculated with the following formula.

$$MSSIM = \frac{1}{N}\sum_{1}^{N} SSIM\,(n) \qquad (8)$$

As it can be understood from the formula first the SSIM value for a frame is found then the average structural index is revealed based on the SSIM value of each frame [18], [19], [20], [21].

4. Method and Material

In the scope of this study, it is aimed to stream different detailed video scenes with suitable qualities proportional to their details to minimize the bandwidth allocation. Moreover, since less detailed scenes are streamed with lower bitrate, the bandwidth usage will have decreased. In the first stage, a sample video with different detailed parts is searched for encoding. We selected a video which has averagely three seconds duration. Afterwards, video is examined and divided it into detailed and non-detailed parts based on some criteria such as color changes and object density by visually. During the encoding phase, video is encoded with several bitrate alternatives according to the SSIM quality metric. At the end of the encoding, segments and manifest (MPD) files are created for the experiments with MP4Box tool. Then an up-to-date player has been selected for MPD file to be read and video segments to be played properly. The server where the video contents will be stored is set up and the video contents on the server are streamed by the player at the last stage. Finally, statistical information observed in Wireshark Packet Capture program for traditional and proposed streaming architectures is analyzed. The FFMPEG tool has been selected for the compression process and converts the video file to the H.264 extension file by encoding according to the user parameters. Additionally, video is primarily converted from mp4 format to YUV model before encoding process shown in Figure 4. YUV is a colour model and coding system used as a part of colourful image processing systems. While coding an image or video colours in the YUV model, the bandwidth of these colours is reduced by taking into account the human perception [22]. All phases of the study can be seen in Figure 5.



```
D:\ffmpeg\bin>ffmpeg -y -video_size 1280x720 -pix_fmt yuv420p -r 24 -i out.yuv -
ffmpeg version N-93944-gd903c09d9a Copyright (c) 2000-2019 the FFmpeg developers
  built with gcc 8.3.1 (GCC) 20190414
  configuration: --enable-gpl --enable-version3 --enable-sdl2 --enable-fontconfi
able-libopencore-amrnb --enable-libopencore-amrwb --enable-libopenjpeg --enable-
vpack --enable-libwebp --enable-libx264 --enable-libx265 --enable-libxml2 --enab
fa --enable-libspeex --enable-libxvid --enable-libaom --enable-libmfx --enable-a
```

Figure 4. Converting to YUV Model

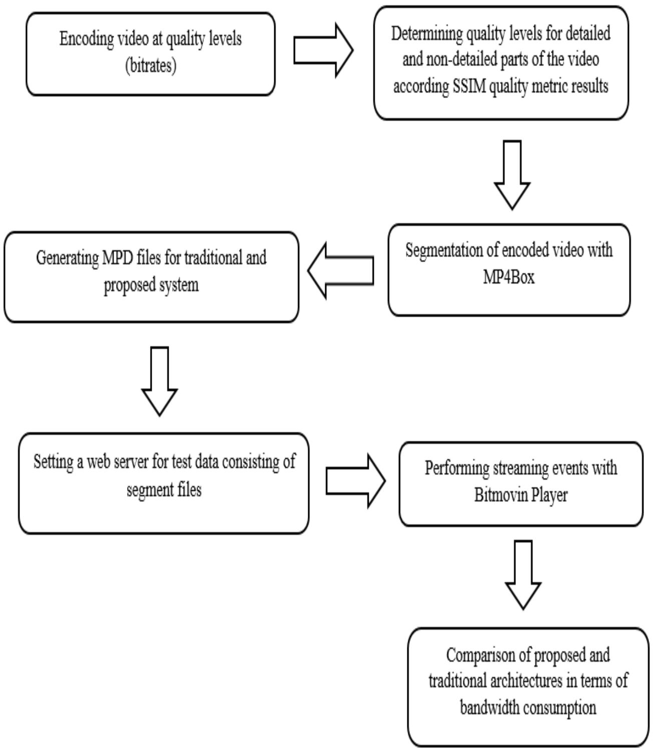

Encoding video at quality levels (bitrates)
Determining quality levels for detailed and non-detailed parts of the video according SSIM quality metric results
Segmentation of encoded video with MP4Box
Generating MPD files for traditional and proposed system
Setting a web server for test data consisting of segment files
Performing streaming events with Bitmovin Player
Comparison of proposed and traditional architectures in terms of bandwidth consumption

Figure 5. Phases of the Proposed System

Table 1. Compression Information for the video

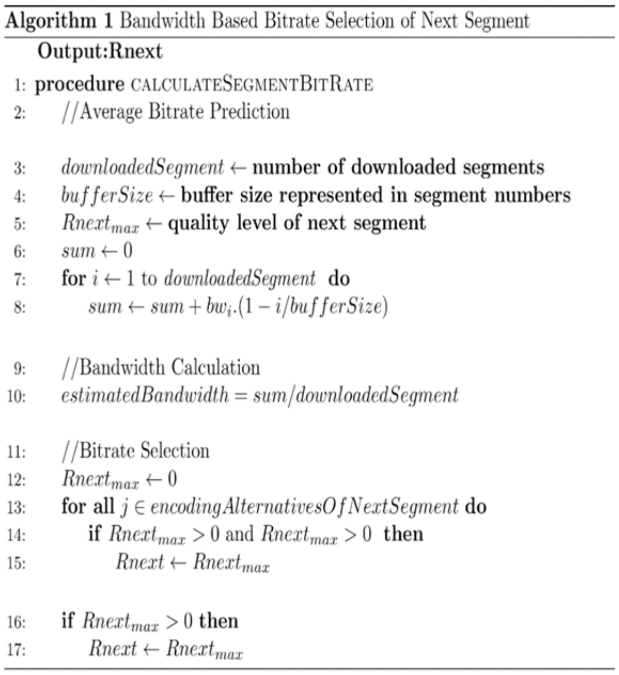
Compression Bitrate (Kbps)	SSIM Value	Resolution
400	9737868	1280x720
500	9767528	1280x720
800	9790472	1280x720
1200	9848024	1280x720
1500	9864244	1280x720
2500	9916218	1280x720
3500	9968426	1280x720
4200	9986218	1280x720

According to the SSIM values, while 4200 Kbps bitrate is selected for detailed parts of the video, for undetailed parts, 1500 Kbps bitrate is seen as suitable for the proposed streaming process, which is shown in Table 1. Therefore, SSIM values of these bitrates could be acceptable in terms of human perception. Finally, MPD file is created including both detailed and non-detailed segments.

Table 2. Preferred Quality Levels in Each Streaming

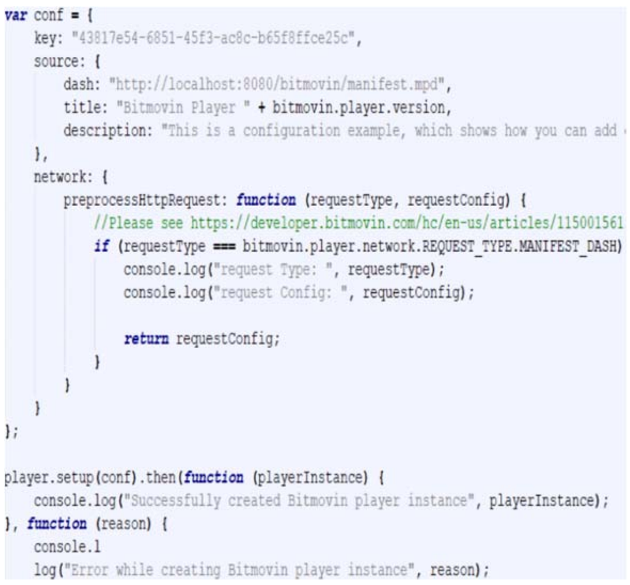
Streaming Information		
	Traditional Streaming Design (Kbps)	Proposed Streaming Design (Kbps)
Detailed Video Scenes	4200	4200
Non – Detailed Video Scenes	4200	1500

Since we aim to reduce the burden of non-detailed part of the whole media, we only modify the bitrate of those segments when carrying out our experiments with traditional and proposed architectures. After generating segments and MPD file, Bitmovin player is applied for the streaming experiments and it works based on the algorithm given in Figure 6. Firstly, the bandwidth is estimated by averaging the measured bandwidths of recently received segments into the buffer of the player. Then, the rate-base switching method selects the segment with the maximum bitrate by examining encoding alternatives that is smaller than the estimated bandwidth [23]. Therefore, segment quality cannot exceed bandwidth of the network. A client web page is prepared for this purpose by configuring Bitmovin player settings which are represented in Figure 7. These settings include some parameters such as defining path of manifest file, description and title of the streaming and player key value which is unique for each user. Furthermore, we preferred the segment size as 2 seconds since long sized segments cannot adapt quickly and flexibly to changing network conditions compared to short sized segments. In [3] it is proposed that the segment size should be between 2-4 seconds. Therefore, we set the segment size at 2 seconds.



```
Algorithm 1 Bandwidth Based Bitrate Selection of Next Segment
  Output:Rnext
procedure CALCULATESEGMENTBITRATE
    //Average Bitrate Prediction

    $downloadedSegment \leftarrow$ number of downloaded segments
    $bufferSize \leftarrow$ buffer size represented in segment numbers
    $Rnext_{max} \leftarrow$ quality level of next segment
    $sum \leftarrow 0$
    for $i \leftarrow 1$ to $downloadedSegment$ do
        $sum \leftarrow sum + bw_i.(1 - i/bufferSize)$

    //Bandwidth Calculation
    $estimatedBandwidth = sum/downloadedSegment$

    //Bitrate Selection
    $Rnext_{max} \leftarrow 0$
    for all $j \in encodingAlternativesOfNextSegment$ do
        if $Rnext_{max} > 0$ and $Rnext_{max} > 0$ then
            $Rnext \leftarrow Rnext_{max}$

    if $Rnext_{max} > 0$ then
        $Rnext \leftarrow Rnext_{max}$
```

Figure 6. Algorithm of Bitmovin Player

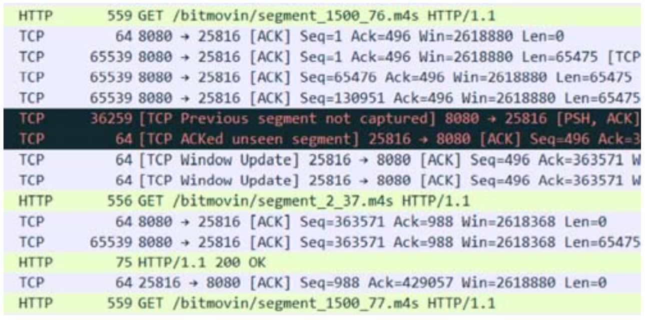


```
var conf = {
    key: "43817e54-6851-45f3-ac8c-b65f8ffce25c",
    source: {
        dash: "http://localhost:8080/bitmovin/manifest.mpd",
        title: "Bitmovin Player " + bitmovin.player.version,
        description: "This is a configuration example, which shows how you can add
    },
    network: {
        preprocessHttpRequest: function (requestType, requestConfig) {
            //Please see https://developer.bitmovin.com/hc/en-us/articles/115001561
            if (requestType === bitmovin.player.network.REQUEST_TYPE.MANIFEST_DASH)
                console.log("request Type: ", requestType);
                console.log("request Config: ", requestConfig);

                return requestConfig;
            }
        }
    }
};

player.setup(conf).then(function (playerInstance) {
    console.log("Successfully created Bitmovin player instance", playerInstance);
}, function (reason) {
    console.l
    log("Error while creating Bitmovin player instance", reason);
```

Figure 7. Configuring Settings

5. Experimental Results and Discussion

Experiments are accomplished with both traditional and proposed streaming architectures successfully. HTTP GET requests of the segments can be seen in Figure 8. From Figure 10 and Figure 12, measured average bandwidth consumptions (amount of transferred data bit per second) are reached for both model through TCP layer. Bandwidth utilization is measured with Wireshark Packet Tracer and behaviour of both streaming models are shared with the graphs represented with Figure 9 and Figure 11 illustrating usage of bandwidth per second during 200 seconds. As given in the Table 3, experimental results show that in traditional streaming, bandwidth usage is observed as 2114 Kbps averagely. It has been noticed that video buffer is quickly filled with the media segments which is above 6.10^7 and closer to 8.10^7 bits during the initial seconds of streaming which is called as initial buffering. Consequently, in the next phases of streaming, it has been observed that streaming proceeded with bitrates smaller than the compression rate. It is because high amount of data is used from buffer and only remaining part of the media data is transmitted over the network during the playback.



```
HTTP      559 GET /bitmovin/segment_1500_76.m4s HTTP/1.1
TCP        64 8080 → 25816 [ACK] Seq=1 Ack=496 Win=2618880 Len=0
TCP     65539 8080 → 25816 [ACK] Seq=1 Ack=496 Win=2618880 Len=65475 [TCP
TCP     65539 8080 → 25816 [ACK] Seq=65476 Ack=496 Win=2618880 Len=65475
TCP     65539 8080 → 25816 [ACK] Seq=130951 Ack=496 Win=2618880 Len=65475
TCP     36259 [TCP Previous segment not captured] 8080 → 25816 [PSH, ACK]
TCP        64 [TCP ACKed unseen segment] 25816 → 8080 [ACK] Seq=496 Ack=3
TCP        64 [TCP Window Update] 25816 → 8080 [ACK] Seq=496 Ack=363571 W
TCP        64 [TCP Window Update] 25816 → 8080 [ACK] Seq=496 Ack=363571 W
HTTP      556 GET /bitmovin/segment_2_37.m4s HTTP/1.1
TCP        64 8080 → 25816 [ACK] Seq=363571 Ack=988 Win=2618368 Len=0
TCP     65539 8080 → 25816 [ACK] Seq=363571 Ack=988 Win=2618368 Len=65475
HTTP       75 HTTP/1.1 200 OK
TCP        64 25816 → 8080 [ACK] Seq=988 Ack=429057 Win=2618880 Len=0
HTTP      559 GET /bitmovin/segment_1500_77.m4s HTTP/1.1
```

Figure 8. HTTP GET Requests of Segments

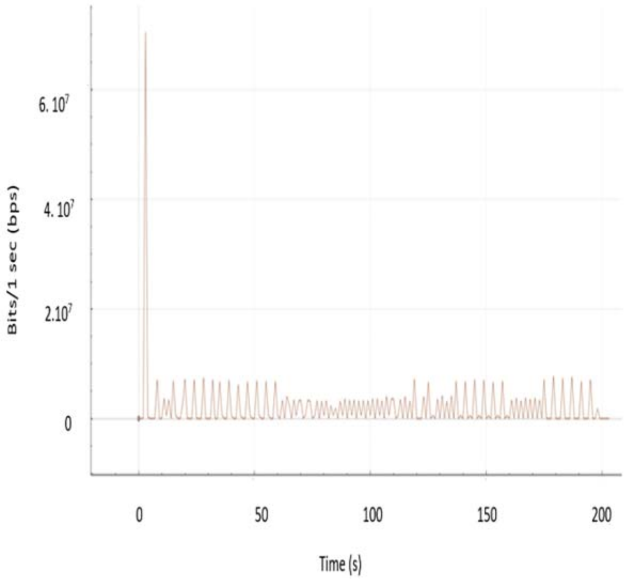
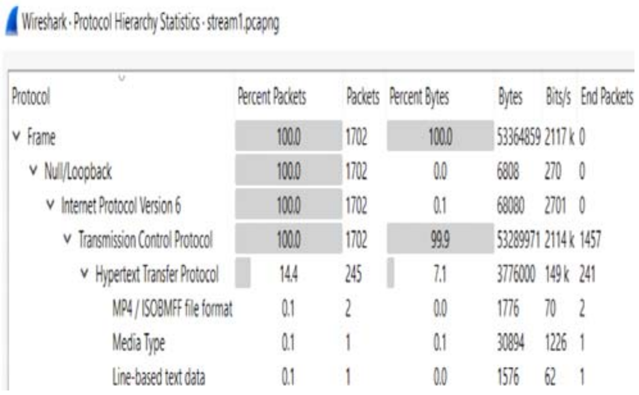


Figure 9. Bandwidth Usage Graph of Traditional Streaming



Wireshark · Protocol Hierarchy Statistics · stream1.pcapng

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets
∨ Frame	100.0	1702	100.0	53364859	2117 k	0
∨ Null/Loopback	100.0	1702	0.0	6808	270	0
∨ Internet Protocol Version 6	100.0	1702	0.1	68080	2701	0
∨ Transmission Control Protocol	100.0	1702	99.9	53289971	2114 k	1457
∨ Hypertext Transfer Protocol	14.4	245	7.1	3776000	149 k	241
MP4 / ISOBMFF file format	0.1	2	0.0	1776	70	2
Media Type	0.1	1	0.1	30894	1226	1
Line-based text data	0.1	1	0.0	1576	62	1

Figure 10. Bandwidth Usage Graph of Traditional Streaming

The same situation has also been noticed in the proposed system and again an amount of media data, which is closer to 6.10^7 bits is filled into the buffer average bandwidth consumption is found as 1380 Kbps. Also, it is seen that the behaviour of bandwidth usage becomes smoother than the traditional one when they are compared.

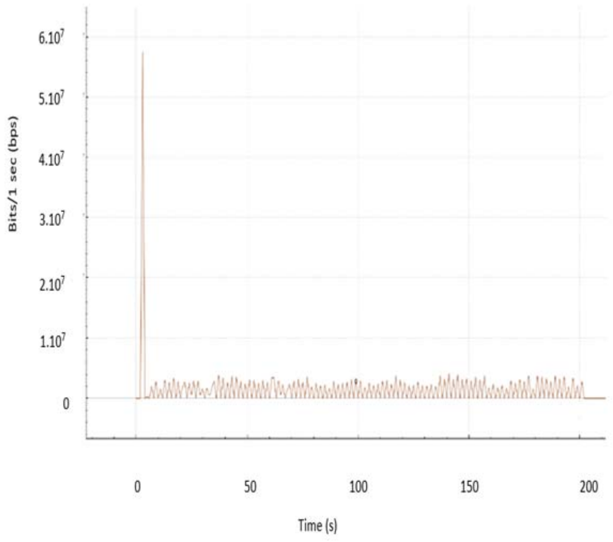
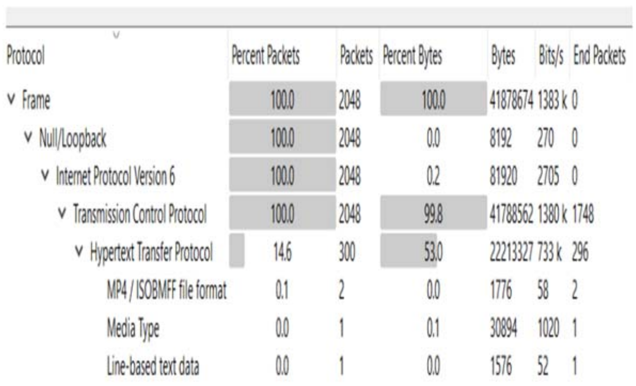


Figure 11. Bandwidth Usage Graph of Proposed Streaming



Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets
∨ Frame	100.0	2048	100.0	41878674	1383 k	0
∨ Null/Loopback	100.0	2048	0.0	8192	270	0
∨ Internet Protocol Version 6	100.0	2048	0.2	81920	2705	0
∨ Transmission Control Protocol	100.0	2048	99.8	41788562	1380 k	1748
∨ Hypertext Transfer Protocol	14.6	300	53.0	22213327	733 k	296
MP4 / ISOBMFF file format	0.1	2	0.0	1776	58	2
Media Type	0.0	1	0.1	30894	1020	1
Line-based text data	0.0	1	0.0	1576	52	1

Figure 12. Bandwidth Utilization of Proposed Streaming

As a result of the experiments, the flow of the proposed system where non-detailed scenes are encoded with low bitrate and the traditional one where all scenes are compressed with high bitrate are compared. We observed that the bandwidth usage is reduced by a rate of 35%. For this reason, it is possible to create efficient video streams through encodings with high SSIM results leading contents of high quality and also reduce the burden of data passing along the channel.

Table 3. Average Results of Both Streaming Events

Traditional Streaming (Kbps)	Proposed Streaming (Kbps)
2114	1380

6. Conclusion

When the study is summarized, a streaming architecture based on MPEG DASH has been developed that alleviate bandwidth allocation and satisfy video users. In the results of experiments, it is noticed that very large amount of media data is stored in the player buffer at the beginning of the video playback. Therefore, lower level of media data is transmitted in terms of bitrate per second in the later periods of the flow. Also, our proposed system has been found to give more efficient outcome which supplies 35% reduction at bandwidth utilization in the network. On the other hand, developed architecture also enables video files to be stored in a way those take up less space on the web server allowing bitrate savings. Another important issue is that different algorithms are used to determine the quality level (bitrate) of the next segment to be sent over the network during streaming [24]. For example, these algorithms work based on some parameters such as amount of the media remaining in the buffer of the player or how much of the last segment is downloaded per second. However, generally, such algorithms cannot intervene quickly when sudden changes occur in network conditions and can delay in taking actions. Within the scope of the future study, as a solution to this problem historical statistical results can be used to estimate the quality level of the next segment with neural networks-based algorithms. The segment output efficiencies observed during the flow can be given as parameters to these algorithms and the relationships between them can be calculated to decide the behaviour of the network at a certain rate. Accordingly, the quality level of the next segment can be chosen according to the result of the optimization algorithm.

References

[1]. Barnett, T., Jain, S., Andra, U., & Khurana, T. (2018). Cisco visual networking index (vni) complete forecast update, 2017–2022. *Americas/EMEAR Cisco Knowledge Network (CKN) Presentation*.

[2]. Mogollón Rodríguez, J. F., (2018). Improving http adaptive streaming systems, the latency and content distribution questions. Universitat Oberta de Catalunya.

[3]. Lederer, S., Müller, C., & Timmerer, C. (2012, February). Dynamic adaptive streaming over HTTP dataset. In *Proceedings of the 3rd multimedia systems conference* (pp. 89-94).

[4]. Stockhammer, T. (2011, February). Dynamic adaptive streaming over HTTP-- standards and design principles. In *Proceedings of the second annual ACM conference on Multimedia systems* (pp. 133-144).

[5]. Kantarcı, A. (2010). Bandwidth-effective streaming of educational medical videos. *Multimedia systems*, *16*(6), 381-397.

[6]. Zach, O., & Slanina, M. (2018). Content aware segment length optimization for adaptive streaming over HTTP. *Radioengineering*, *27*(3), 819.

[7]. Curcio, I. D., Toukomaa, H., & Naik, D. (2017, October). Bandwidth reduction of omnidirectional viewport-dependent video streaming via subjective quality assessment. In *Proceedings of the 2nd International Workshop on Multimedia Alternate Realities* (pp. 9-14).

[8]. Adzic, V., Kalva, H., & Furht, B. (2012, January). Content aware video encoding for adaptive HTTP streaming. In *2012 IEEE International Conference on Consumer Electronics (ICCE)* (pp. 92-93). IEEE.

[9]. Hassan, Y. M., Helmy, A., & Rehan, M. M. (2014, April). Effect of varying segment size on DASH streaming quality for mobile user. In *2014 International Conference on Engineering and Technology (ICET)* (pp. 1-4). IEEE.

[10]. Kreuzberger, C., Rainer, B., Hellwagner, H., Toni, L., & Frossard, P. (2016, May). A comparative study of DASH representation sets using real user characteristics. In *Proceedings of the 26th International Workshop on Network and Operating Systems Support for Digital Audio and Video* (pp. 1-6).

[11]. Ognenoski, O., Razaak, M., Martini, M. G., & Amon, P. (2013, October). Medical video streaming utilizing MPEG-DASH. In *2013 IEEE 15th International conference on e-health networking, applications and services (Healthcom 2013)* (pp. 54-59). IEEE.

[12]. Mueller, C. (2019). MPEG-DASH (Dynamic Adaptive Streaming over HTTP, ISO/IEC 23009-1). Retrieved from: https://bitmovin.com/dynamic-adaptive-streaming-http-mpeg-dash/ [accessed: 02 December 2021].

[13]. Sodagar, I. (2011). The mpeg-dash standard for multimedia streaming over the internet. *IEEE multimedia*, *18*(4), 62-67.

[14]. Long, B. (2015). The structure of an MPEG-DASH MPD. *Brendan Long Website, Mar*, *20*. Retrieved from: https://www.brendanlong.com/the-structure-of-an-mpeg-dash-mpd.html [accessed: 02 January 2022].

[15]. Ramani, K. (2015). *Media Presentation Description over MPEG-DASH*. LAP LAMBERT Academic Publishing.

[16]. Sara, U., Akter, M., & Uddin, M. S. (2019). Image quality assessment through FSIM, SSIM, MSE and PSNR—a comparative study. *Journal of Computer and Communications*, *7*(3), 8-18.

[17]. Ümit, Ka Ş., & Tanyildizi, E. (2017). Analiza performansi Eulerove boje i metode povećanja kretanja. *Afyon Kocatepe University Journal of Science and Engineering* , *17* (2), 506-515.

[18]. Søgaard, J., Krasula, L., Shahid, M., Temel, D., Brunnström, K., & Razaak, M. (2016). Applicability of existing objective metrics of perceptual quality for adaptive video streaming. *Electronic Imaging*, *2016*(13), 1-7.

[19]. Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, *13*(4), 600-612.

[20]. Kaya, Y., & Kayci, L. (2014). Kelebek görüntülerin sınıflandırılması için bir içerik bazlı görüntü erişim sistemi. *Akademik Bilişim, Mersin, Türkiye, Şubat*.

[21]. A Video Clarity White Paper. (2021). White Paper – Advancing to Multi-Scale SSIM. Retrieved from: https://videoclarity.com/PDF/Advancing-To-Multi-Scale-SSIM-Final.pdf [accessed: 11 December 2021].

[22]. Opteamx. (2021). YUV. Retrieved from: https://www.opteamx.com/YUV. [accessed: 04 November 2021].

[23]. Ayad, I., Im, Y., Keller, E., & Ha, S. (2018). A practical evaluation of rate adaptation algorithms in http-based adaptive streaming. *Computer Networks*, *133*, 90-103.

[24]. Chung, K. (2015). Buffer-based adaptive bitrate algorithm for streaming over HTTP. *KSII Transactions on Internet and Information Systems (TIIS)*, *9*(11), 4585-4603.