Arun Raj et al. EURASIP Journal on Image and Video Processing (2017) 2017:41
DOI 10.1186/s13640-017-0191-4

EURASIP Journal on Image and Video Processing

RESEARCH Open Access

# Adaptive video streaming over HTTP through 4G wireless networks based on buffer analysis

L. Arun Raj*, Dhananjay Kumar, H. Iswarya, S. Aparna and A. Srinivasan

**Abstract**

One of the main challenges in video transmission is understanding and adapting to the varying network bandwidth. The traditional approach of bandwidth estimation is not accurate as there are many factors like congestion that can delay the arrival rate of the ping packet which may lead to a misconception that the bandwidth was low. Thus, the better approach to this problem will be to estimate the link conditions based on the buffer fullness. In this paper, a new system to support streaming of live and stored video through wireless network is proposed which is based on adaptive playback buffer management on the top of HTTP at the client. The buffer fullness is treated as a direct state variable that reflects the fluctuation of the network bandwidth. The buffer fullness estimation predicts the buffer status at a point in the future based on observations of the buffer over a stipulated period of time. The proposed algorithm uses non-linear exponential non-parametric regression for computing the decision parameter. A feedback message is then sent to the server in order to change the quality of the video stream for smoother video play at the client side. The synchronized update and feedback between the server and clients is achieved using HTTP protocol. During the experimentation with live video streaming, the proposed algorithm shows an improvement of 24.48% in average peak signal-to-noise ratio and 6.63% in average structural similarity index against the buffer underflow probability algorithm.

**Keywords:** Video streaming, Adaptation, Client-server, Buffer analysis, PSNR, SSIM, VQM, Regression

## 1 Introduction

The demand for video streaming over a wireless network is rapidly increasing day by day. As per prediction of Cisco Visual Networking Index, 4G wireless networks will have 40.5% of the total mobile connections worldwide, out of which 75% of the mobile data traffic will be the video by 2020 [1].

In streaming application, the video content is sent in compressed form over the Internet, and at the receiver, it is decoded and played in real-time [2]. A number of parameters need to be considered during live video streaming in wireless environment. Some of the factors that affect video streaming are bandwidth, delay, jitter, packet loss rate, and compression. Due to the time-dependent nature of the Internet traffic, maintaining the quality of service (QoS) in heterogeneous network remains a challenge. However, QoS is an end-to-end issue of communication quality concerning link level parameters [3].

The fluctuating bandwidth in wireless channel conditions requires adaptive video streaming for efficient and higher quality of experience (QoE). Further, it may create video freeze at display in the user equipment due to more wait for the video frame to buffer [4]. The link capacity estimation based on the arrival of ping packets on server side leads to a low bandwidth misconception for estimating the channel bandwidth on the server side. When the server sends a video stream of low quality, buffering of stream with user's available bandwidth could take place and degradation of the user's QoE still may occur during high bandwidth conditions [5]. Thus, a better solution to this problem will be to estimate the channel conditions based on the buffer fullness in the client side itself and to send a feedback message to the server, so that the server will change the video quality accordingly.

* Correspondence: arun4u85mit@gmail.com
Department of Information Technology, Anna University, MIT Campus, Chennai, India

In the existing MPEG-DASH system, different video segments are downloaded using the TCP protocol [6]. It may reduce packet loss compared to the UDP-based stream, but because of retransmission policy of TCP, extra delay gets added to the streaming packets which degrades the video quality. Additionally, DASH methods have difficulty in handling live video stream as it converts the video data into several chunks and encodes at different quality level to be downloaded by the receiver judiciously [7].

The play back disruptions like flicker and video freeze resulting from buffer underflow degrade the QoE. The probability of buffer underflow can be linked to the streaming bit rate and channel throughput [8, 9]. However, the estimation of underflow probability requires theory of large deviation with application of the queuing theory and channel conditions. This limits the performance of the system based on buffer underflow probability (BUP) in live streaming of video.

The proposed algorithm consists of a regression theory to estimate the buffer filling rate and predict the near future state of buffer to determine whether any action is to be taken to the video stream by the server. During buffer underflow/overflow conditions, the video quality parameter is adjusted to match the link condition, thereby facilitating an uninterrupted reception of video stream on the client side. Hence, buffer fullness is used as a measure for the mismatch between the video bit rate and channel throughput.

Investigating the relationship between dependent (target) and one or more independent variables (predictor) is referred to as regression analysis which is a form of predictive modeling technique [10] and can be used to compare the effects of variables measured on different scales that also evaluate the best set of variables for predictive model building. It also allows comparing the effects of variables measured on different scales. There are various kinds of regression techniques available to make predictions. These techniques are mostly driven by three metrics: number of independent variables, type of dependent variables, and shape of regression line. The most commonly used types of regression are linear, logistic, polynomial, stepwise, ridge and lasso regression. The proposed method employs exponential non-linear regression to estimate and predict buffer state used in decision-making of the client.

The rest of this paper is organized as follows: Section 2 presents a case study and solution methodology. Section 3 introduces the related work on adaptive video streaming. Section 4 consists of system model, problem formulation, and performance metrics. Section 5 presents an overview of the proposed algorithm, and Section 6 deals with the implementation environment for video streaming system. Results and discussion are presented in Section 7, and Section 8 consists of conclusion and future works.

## 2 Case study and solution methodology

The streaming video needs to be adapted to the available bit rate at client to provide seamless delivery of the media contents. This poses several challenges on system design as the underlying network, i.e., the Internet, which works on best effort service model, does not guarantee the quality of service. Further, if the link connectivity is supported on wireless channel, the system does not provide a constant throughput. Figure 1 shows an observed bit rate at client machine connected to the Internet through 4G hotspot (offered by Airtel Limited) at the workplace in a given time during streaming of a live video. Clearly, the available bit rate not only fluctuates in uplink but also in download, which often results in video freeze at display device during streaming. The solution to these problems will be highly rewarding if the proposed method can be easily deployed in application space without any modifications in the underlying setup which include network protocols and physical layer hardware.

In the proposed method, a suitable mechanism is devised which works on the top of TCP/IP network stack, i.e., above HTTP protocol where the underlying physical

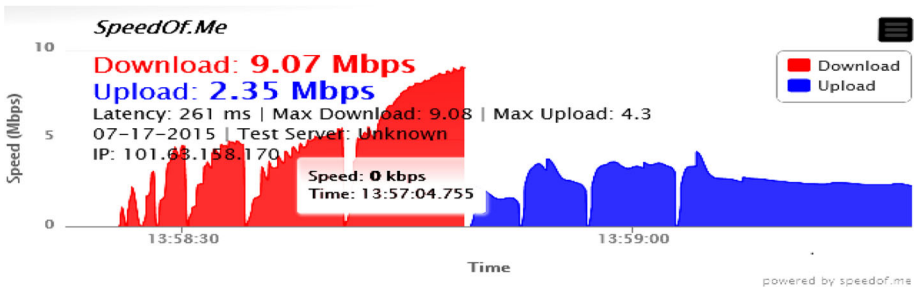


**Fig. 1** Bit rate measurement on 4G hotspot devices. The bit rates were observed at client machine connected to the Internet using LTE 4G hotspot device of Airtel Limited

link is supported on a 4G cellular mobile network. The adoption of client-server architecture simplifies the constraints of implementation at the end systems. The client is allowed to monitor the flow of user content continuously by observing the fullness of media playout buffer. Now, the paramount interest lays in the effective management of playout buffer which results in increase of quality of video at client.

The dynamic analysis of buffer fullness can be carried out with on-the-fly study of filling rate of the buffer with some prior knowledge of filling pattern. This approach requires a development of statistical prediction method which accounts not only for the present condition but also for the series of past data. The non-parametric regression techniques provide a suitable mechanism in predicting the state of buffer over a defined interval of time. This technique is called recursively for decision-making and implemented in the system loop consisting of client, server, network, and the source of contents.

## 3 Related work

Adaptive video streaming uses HTTP segments of video based on different resolutions and frame rate and stores each segment. On the client side, an adaption algorithm is implemented that request a segment of the required and most relevant quality based on the current network quality level. When the bandwidth fluctuates or becomes very low abruptly, present algorithms do not perform very well [11]. This leads to buffering on the client side, and this severely affects the user's quality of experience. Any framework that has the ability to increase the quality of experience on the client side by mainly decreasing the video freezes is desirable [12]. A way of prioritization will further help to deliver video segments based on server as well as client feedback.

The best decision needed to be taken is that which increases video quality on the client side. Also, during poor bandwidth conditions or when there is a sudden drop in bandwidth, the network congestion is reduced and, at the same time, there is minimal risk of buffering.

An adaptive layer switching system that uses scalable video coding at the server side and buffer underflow probability on the client side is very efficient for wireless networks that support on demand streaming [8]. The fullness of the client side buffer can be used as a measure of the current network condition. If the buffer is full, then it means that the network conditions are excellent and vice versa. Hence, buffer fullness can be used efficiently for dynamic and adaptive streaming systems. The video layers are "switched" based on the buffer underflow probability. Here, probability is used in order to improve the user's quality of experience without any knowledge of the network conditions. Further, the size of playback blocks varies among different layers and priority may be assigned as per the importance of the block [13].

Decisions at the application, transport, network, data link, and physical layers influence the received quality of video transmitted over a wireless network. Finding a suitable bit rate to send each coded block video stream is a challenge [14]. Many streaming issues can be tackled by taking video-specific information into account when making lower layer decisions. A cross-layer approach [15] considers the frequency transformed quantization, entropy encoding functions, and predictive video encoders to take each image block and compare it to other macroblocks either within the same frame (intra-prediction) or in a previous frame (inter-prediction). By identifying the difference between two macroblocks and encoding only that, the encoder can significantly reduce the amount of information necessary to represent a video for a desired quality. The compressed sensing (CS) can offer an alternative to traditional video encoders by enabling imaging systems that sense and compress data simultaneously at very low computational complexity for the encoder. CS images and videos are also resilient to channel errors. CS takes advantage of the fact that the difference between two correlated frames is sparse and uses this information to again compressively sample the difference between two encoded frames. The video encoder performance can be improved by applying optimal tiling during the transformation stage [16, 17].

Layer switching algorithm when implemented for a dynamic system streaming videos is very difficult to design and implement in real-time video applications. This is because the network conditions are not accurately available in a real-time system. When there is an underflow on the client buffer, there is playback interruption on the client side while video is played. Hence, it is a good idea to reduce the bit rate when there is buffer overflow and to increase the bit rate and hence the quality when there is buffer underflow and maintain the quality when the buffer is moderately filled. Also, buffer fullness does not depend on knowing the video statistics, so it is very useful in live streaming as well as on demand streaming. A quality-driven cross-optimized system [18] needs to consider the client buffer fullness measure represented as a mixed binary integer programming problem and solution using a sub-gradient method which thereby reduces the client side interruptions. Further, the system also needs to understand the tradeoff between downloading enough video content before playing it and the use of the channel bandwidth for downloading and storing it in buffer.

## 4 System model, problem formulation, and performance metrics

### 4.1 Client-server model

The proposed architecture (Fig. 2) is modeled after client-server system, where the server captures the input video through a camera or stored content in database.

On the expense of complexity at client, the server is relived from the major work, i.e., flow of stream analysis. The media content is encoded initially in a standard quality and then streamed to the client. As soon as the streaming starts, the client starts the buffering of media content and applies the proposed algorithm to estimate the buffer filling rate. While the video stream is being played, the buffer fullness rate is sampled periodically, and this chronological data acts as an input to the proposed algorithm. If the buffer fullness is too high or too low, a feedback message is sent to the server to adjust the outgoing stream property. The server concurrently listens to the message from the client and adjusts the video quality (video resolution, frame rate, and etc.) so that it matches the maximum achievable quality.

### 4.2 System modules at client

The client side contains three modules.

1. The first module establishes a TCP/IP connection with the server. It receives the data stream and decodes the content using the VLCJ framework for playing the video. The data stream consists of the encoded video and different metadata to help the decoder.
2. The second module monitors the data rate of the stream by packet capture using JPCAP (a Java network packet capture library). This information is used to estimate the buffer fullness at a regular interval. The analysis module basically employs a regression algorithm to analyze and predict the buffer state.
3. The third module implements the feedback decision and formats the message as per the server requirements. Based on the predicted value by module 2 and the channel conditions, an appropriate decision is sent as a feedback message to the server. Figure 3 provides the illustration of the client side modular implementation.

### 4.3 Server side modules

The server side contains two main modules as shown in Fig. 4.

1. The first module uses the VLCJ framework for streaming the video data continuously to the client. It involves the socket-binding to establish connection between the server and the client. The VLCJ provides in-built encoders and options to stream video with preferred parameters.
2. The second module listens to the feedback messages send by the client for adjusting the video bit rate by changing the quality parameters (resolution, frame per second, and etc.).

### 4.4 Problem formulation

In adaptive video streaming over HTTP, the playout buffer fullness need to be monitored and analyzed, and

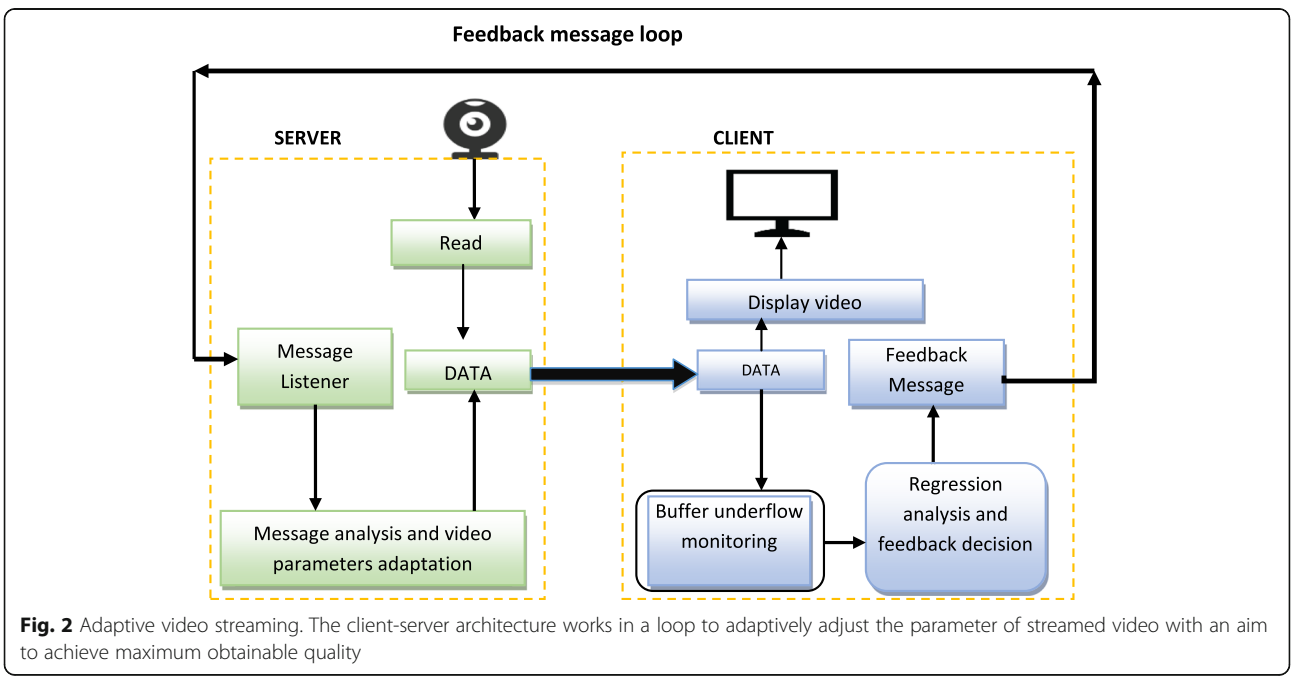


**Fig. 2** Adaptive video streaming. The client-server architecture works in a loop to adaptively adjust the parameter of streamed video with an aim to achieve maximum obtainable quality

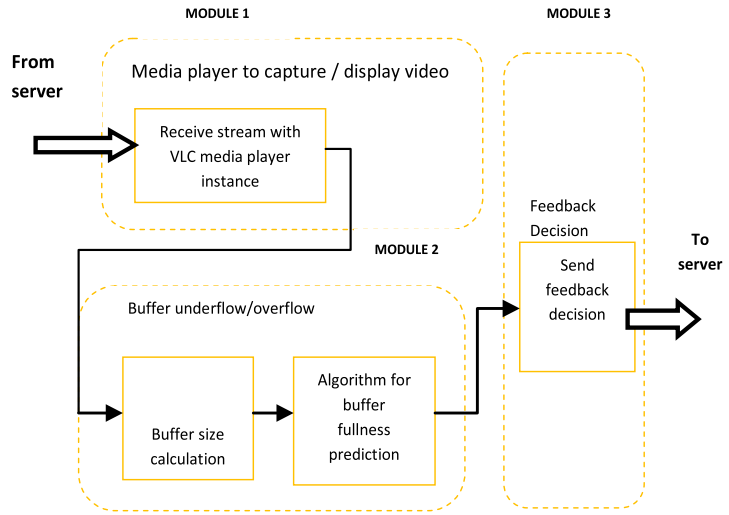


**Fig. 3** Client side modules. The main task is to monitor the receiving bit stream and apply prediction algorithm to estimate buffer fullness

remedial actions for seamless viewing requirements, i.e., to sustain the streaming, need to be defined. The buffer fullness state depends on the throughput of the underlying networks. Further, the network throughput is governed by the prevailing Internet traffic and wireless channel conditions.

The video playout starts when the filling of media buffer exceeds a threshold. The number of frames $\{Q(n)\}$ in a playout buffer provided that the $n$th frame of video has been played can be related to the content arrival rate and playout rate. The buffer occupancy in a given instance can be expressed as [19].

$$Q(n+1) = \max\left\{0,\ Q(n) - 1 + \frac{A_r(n)}{A_p}\right\},\ n = 1, 2, \ldots \quad (1)$$

where, $Ar(n)$ is the frame arrival rate and $Ap$ is the playout rate in frame per second.

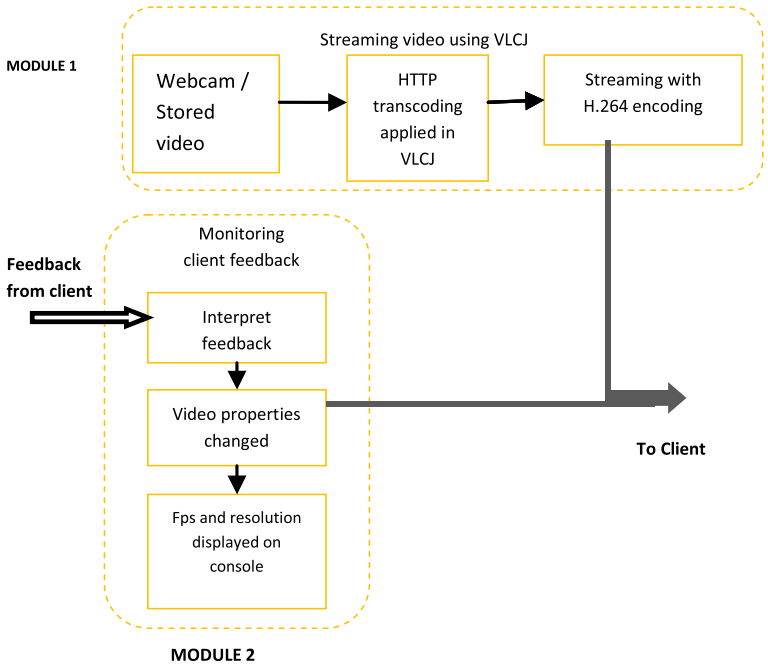


**Fig. 4** Server side modules. The server's main function includes capturing and transcoding of video frames at a different quality based on client feedback

In a statistical approach, the state of buffer occupancy over time can be treated as set of random variables ($x \in X_n$, $n = 1, 2, 3, \ldots N$), where $x_1, x_2, x_3, \ldots\ldots x_n$ represents the sampled value of the buffer state. A suitable regression technique can be applied to predict the buffer fullness based on these sampled data.

#### 4.4.1 Non-linear regression

In the given scenario, the values representing the buffer state are found to be non-linear in nature. Hence, the non-linear regression model is used to fit to the data. A non-linear regression can be modeled in exponential relationship given by

$$y_n = A_1 e^{B_1 x_n} \tag{2}$$

where $A_1$ and $B_1$ are constants that define the nature of curve while representing the trend of buffer filling over the time interval for $n$ samples $x_1, x_2, \ldots, x_n$.

Finding solution to non-linear model is always problematic, but converting it to a linear model with suitable transformation can simplify the process. The linearization of (2) is carried out by applying natural logarithm which can be expressed as

$$ln\, y = \alpha + \beta x \tag{3}$$

where $\beta$ represents the slope and $\alpha$ represents the intercept of the regression line.

The process of linearization of (2) and transformation back to the exponential non-linear regression is depicted in Fig. 5.

Now, (3) can be solved using linear regression methods while taking input in natural logarithmic form. Defining a straight line representing linear regression, where $Y$ replaces $lny$, the input-output relation can be expressed as

$$Y = \alpha + \beta x \tag{4}$$

With a view to model the equation on the straight line, from which the slope $\beta$ and intercept $\alpha$ values can be obtained, and which would provide the "best fit" for the data points, $Y$ is transformed to the non-linear domain given as

$$y = ln\, A_1 + B_1 x \tag{5}$$

where

$$A_1 = e^a \tag{6}$$

and

$$B_1 = \beta \tag{7}$$

#### 4.4.2 Non-parametric regression

Another approach for predicting the buffer conditions uses non-parametric regression called regressogram. The Nadaraya-Watson equation [8] is used as the regression function for the time interval bandwidth of a positive value of $h > 0$. The buffer state $f_n(x)$ in a given instance $n$ is represented by

$$f_n(x) = \frac{\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) y_i}{\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)} \tag{8}$$

where the Gaussian kernel, $k(.)$ for the time sample is given by

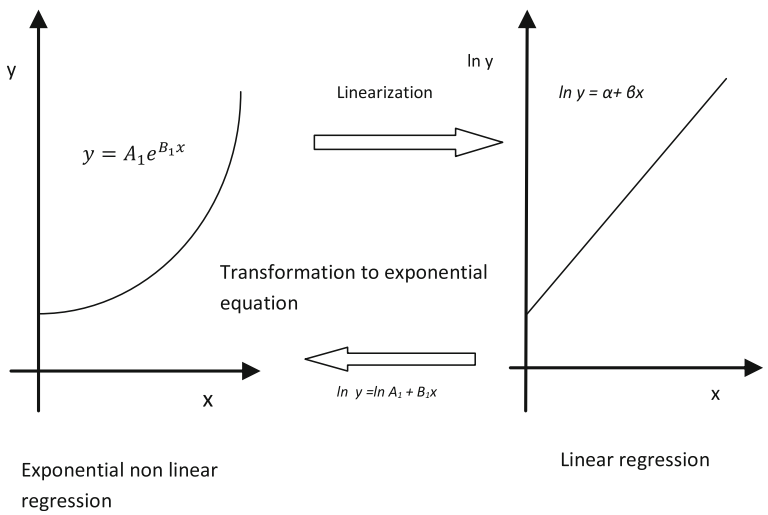


**Fig. 5** Transformation of regression domain. The non-linear regression is converted to linear using log function and after processing converted back to non-linear

$$k(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}} \tag{9}$$

An appropriate value of time interval ($h$) is chosen such that $R(h)$ given by (10) is minimum.

$$R(h) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{r}_n(X_i))^2 \tag{10}$$

where $\hat{r}_n(X_i)$ is the time sample estimator defined as

$$\hat{r}_n(x) = \sum_{i=1}^{n} l_i(x) Y_i \tag{11}$$

Here,

$$l_i(x) = \begin{cases} \frac{1}{k_j}, if\ X_j \epsilon B_j \\ 0, \quad \text{otherwise} \end{cases} \tag{12}$$

where $X_i$ is confined for interval $[a, b]$ defined by $b$ ins $B_1$, ….. $B_m$, and $k_j$ is the number of observations in b in $B_j$.

### 4.5 Video quality evaluation parameters

Objective models for determining video quality using mathematical models can be used for ascertaining the efficiency of the algorithm which performs the transmission of the video packets with minimum delay and optimal quality. The full-reference (FR) metrics helps to compute the quality difference by comparing the original video signal against the received video signal. Every pixel from the source is compared against the corresponding pixel at the received video, with no knowledge about the encoding or transmission process in between. FR metrics are usually the most accurate at the expense of higher computational effort.

#### *4.5.1 Peak signal-to-noise ratio*

The peak signal-to-noise ratio (PSNR) is the ratio between the maximum possible power of a signal and the power of noise that corrupts the accuracy of its representation. PSNR is the most widely used objective image quality metric, and the average PSNR over all frames can be considered a video quality metric [20].

An average PSNR given a noise-free $p \times q$ monochromatic image $I$ and its noisy approximation $J$ is defined using mean square error as

$$\text{MSE} = \frac{1}{pq}\sum_{m=0}^{p-1}\sum_{n=0}^{q-1}[I(p,q) - J(p,q)] \tag{13}$$

Now, PSNR is formulated as

$$\text{PSNR} = 10 \times \log_{10}\left(\frac{\text{MAX}_I^2}{\text{MSE}}\right) \tag{14}$$

where $\text{MAX}_I$ is the maximum possible pixel value of the image.

#### *4.5.2 Structural similarity index*

The structural similarity (SSIM) index is another full-reference metric used for measuring the similarity between two frames based on an initial uncompressed or distortion-free image as reference [21]. The SSIM improves on traditional metrics like PSNR and mean-squared error, which have proven to be inconsistent with human eye perception. The SSIM is expressed as

$$\text{SSIM}(x,y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{xy} + c_2\right)}{\left(\mu_x^2 + \mu_y^2 + c_1\right)\left(\sigma_x^2 + \sigma_y^2 + c_2\right)} \tag{15}$$

where, $\mu_x$ and $\mu_y$ are the average, $\sigma_x^2$ and $\sigma_y^2$ are the variance, and $\sigma_{xy}$ is the covariance of $x$ and $y$, respectively. $c_1$ and $c_2$ are the variables to stabilize the division with weak denominator.

#### *4.5.3 Video quality measurement (VQM)*

The VQM is a standardized method of objectively measuring video quality. A human viewer panel is used to predict the subjective quality rating [22]. End-users and service providers can use VQM tools for a number of purposes like specifying or verifying system performance, comparing service offerings, maintaining and monitoring quality of networks, and optimizing network parameters like bitrate. VQM uses DCT (discrete cosine transform) to correspond to human perception. Brighter blocks correspond to greater difference.

Mean distance between blocks is given by

$$\text{Mean\_Dist} = 1000 * \text{mean}(|\text{diff}|) \tag{16}$$

where 1000 is the standardization ratio.

The maximum distance between the blocks in DCT transformation Max_dist and the VQM score is expressed as

$$\text{Max\_dist} = 1000 * \text{maximum}(|\text{diff}|) \tag{17}$$

$$\text{VQM} = (\text{Mean\_dist} + 0.005 * \text{Max\_dist}) \tag{18}$$

where 0.005 is the maximum distortion weight chosen based on several primitive psychophysics experiments.

## 5 Proposed algorithm

The main proposed algorithm which does the analysis and prediction runs on the client, giving the feedback to the server about the buffer status and indicating the channel conditions. The other algorithm implemented

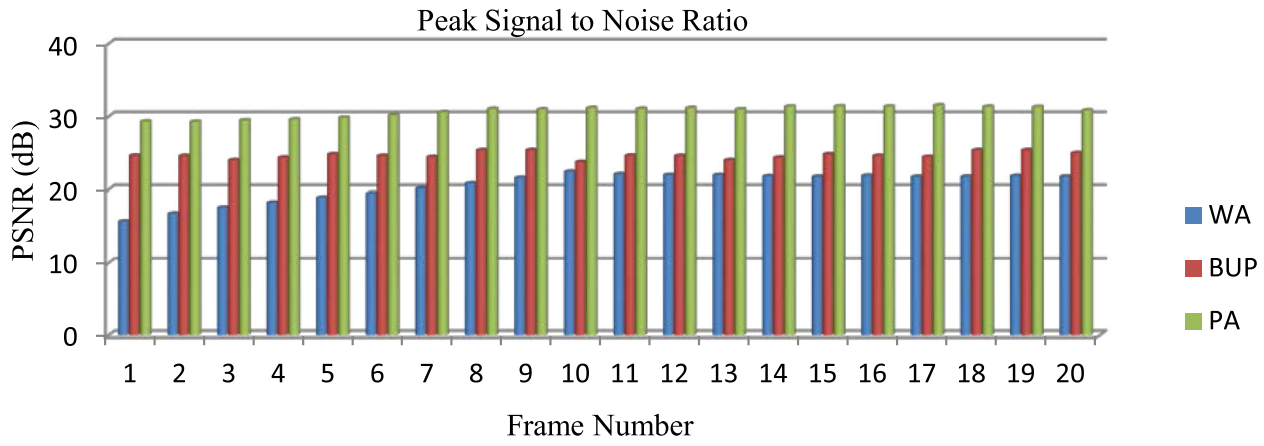


**Fig. 6** The PSNR measurement. The PSNR were measured across selected frames corresponding to the proposed method, and the result is plotted against the existing BUP and WA method

on the server side receives the message from the client and adjusts the video parameters.

### 5.1 Client side algorithm

When the video streaming starts, the buffer fullness state is sampled at a fixed time interval and stored in a temporary memory. The proposed algorithm is applied for every $N$ sample taken chronologically to predict buffer state for a given time interval.

## 6 Buffer analysis algorithm at client

The following are buffer parameters used in the proposed client side algorithm.

***Avg*** is the average size of the buffer taken over $N$ samples
***Lst*** is the buffer size during the $N$th second
***Max*** is the maximum size of the buffer
***Half_Buff*** is half of maximum size of the buffer
***ThreeFour_Buff*** is three fourths of maximum buffer size
***y_diff*** is the array with percentage difference between consecutive buffer size values in an interval
$\boldsymbol{\Delta_i}$ is the percentage increase in $y$_diff
$\boldsymbol{\Delta_d}$ is the percentage decrease in $y$_diff

### 6.1 Client side algorithm

1. Compute *lny* for all $y_i$ values//$y_i$ denotes buffer state at $i$th interval.
2. Calculate slope $\beta$ and intercept $\alpha$, using (3).//Finding regression line
3. Find $A_1$and $B_1$,defined in (6) and (7).
4. Find $y_{(n+1)}$ applying(2).
5. Assign $y_{(n+1)}$ to predicted value $p$.
6. Calculate $\hat{r}_n(x)$ using (11).
7. Find $h$ such that $R(h)$ is minimized, using (10).
8. Find sum of $K\left(\frac{x-x_i}{h}\right)$ values based on (9).

**Table 1** The PSNR measurements

| No. | WA (dB) | BUP (dB) | PA (dB) | No. | WA (dB) | BUP (dB) | PA (dB) |
|---|---|---|---|---|---|---|---|
| 1 | 15.70 | 24.63 | 29.31 | 11 | 22.07 | 24.62 | 31.06 |
| 2 | 16.77 | 24.59 | 29.28 | 12 | 21.96 | 24.59 | 31.16 |
| 3 | 17.59 | 24.00 | 29.48 | 13 | 21.95 | 24.00 | 30.99 |
| 4 | 18.28 | 24.36 | 29.59 | 14 | 21.79 | 24.36 | 31.38 |
| 5 | 18.96 | 24.83 | 29.84 | 15 | 21.73 | 24.83 | 31.40 |
| 6 | 19.53 | 24.59 | 30.19 | 16 | 21.87 | 24.59 | 31.37 |
| 7 | 20.16 | 24.45 | 30.59 | 17 | 21.72 | 24.45 | 31.52 |
| 8 | 20.82 | 25.37 | 31.05 | 18 | 21.73 | 25.37 | 31.34 |
| 9 | 21.58 | 25.38 | 30.97 | 19 | 21.83 | 25.38 | 31.31 |
| 10 | 22.43 | 23.75 | 31.16 | 20 | 21.75 | 24.97 | 30.85 |
| Average (dB) | | | | | *20.44* | *24.65* | *30.69* |

**Table 2** The SSIM index

| No. | WA | BUP | PA | No. | WA | BUP | PA |
|---|---|---|---|---|---|---|---|
| 1 | 0.8772 | 0.8889 | 0.948 | 11 | 0.866 | 0.871 | 0.9543 |
| 2 | 0.8764 | 0.8897 | 0.954 | 12 | 0.875 | 0.879 | 0.9514 |
| 3 | 0.8763 | 0.8825 | 0.957 | 13 | 0.884 | 0.890 | 0.9414 |
| 4 | 0.8705 | 0.8885 | 0.952 | 14 | 0.877 | 0.901 | 0.9536 |
| 5 | 0.8737 | 0.8881 | 0.953 | 15 | 0.878 | 0.930 | 0.9551 |
| 6 | 0.8759 | 0.8877 | 0.958 | 16 | 0.882 | 0.927 | 0.9472 |
| 7 | 0.8733 | 0.8903 | 0.956 | 17 | 0.814 | 0.902 | 0.9516 |
| 8 | 0.8758 | 0.8891 | 0.950 | 18 | 0.826 | 0.895 | 0.9566 |
| 9 | 0.8751 | 0.8899 | 0.958 | 19 | 0.836 | 0.891 | 0.9529 |
| 10 | 0.8762 | 0.8908 | 0.956 | 20 | 0.842 | 0.889 | 0.9475 |
| Average | | | | | *0.867* | *0.893* | *0.953* |

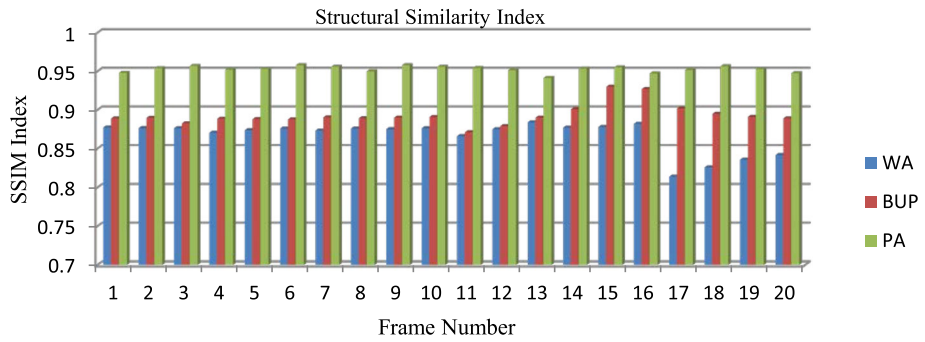


**Fig. 7** The SSIM comparsion. The structural similarity index corresponing to selected consecutive frames is normalized between 0 and 1 for the three techniques discussed here

9. Compute $f(n+1)$ and assign to predicted value $q$, based on(8).//$p$ and $q$ are used for setting decision parameter.
10. Compute average of $p$ and $q$ and assign to average predicated value $P$.
11. If$\Delta_i > \Delta_d$ or $\Delta_i > L_{th}$//Progressive increase buffer
12. If $P \approx Lst$ and $P < ThreeFour\_Buff$
13. Set feedback *message_1*
14. Else,
15. Set feedback *message_2*
16. Else if, $\Delta_d > \Delta_i$ or $\Delta_d > L_{th}$//Progressive decrease buffer
17. If $P \approx Lst$ and $P > ThreeFour\_Buff$
18. Set feedback *message_3*
19. Else,
20. Set feedback *message_4*
21. Else if, $\Delta_i \approx \Delta_d$//Fluctuating buffer
22. If $P \approx Avg$,
23. Set feedback *message_2*
24. Else,
25. Set feedback *message_4*
26. Else if, $\Delta_i < L_{th}/2$ and $\Delta_d < L_{th}/2$//*Constant buffer*
27. If $P > ThreeFour\_Buff$,
28. Set feedback *message_3*
29. Else,
30. Set feedback *message_1*
31. Else,
32. Set feedback *message_4*//no changes to be made.
33. Repeat Step 1 **until** connection terminates.

**Table 3** The VQM scores

| # | WA | BUP | PA | # | WA | BUP | PA |
|---|---|---|---|---|---|---|---|
| 1 | 18.70 | 17.017 | 13.835 | 11 | 17.977 | 17.074 | 9.458 |
| 2 | 18.68 | 17.047 | 13.349 | 12 | 17.994 | 17.105 | 9.174 |
| 3 | 18.71 | 17.187 | 12.627 | 13 | 18.012 | 17.157 | 8.947 |
| 4 | 18.74 | 17.215 | 12.028 | 14 | 18.019 | 17.232 | 8.872 |
| 5 | 18.72 | 17.279 | 11.436 | 15 | 18.088 | 17.234 | 9.015 |
| 6 | 18.58 | 17.330 | 11.041 | 16 | 18.018 | 17.254 | 9.077 |
| 7 | 18.82 | 17.366 | 10.590 | 17 | 18.035 | 17.225 | 9.080 |
| 8 | 18.57 | 17.403 | 9.9642 | 18 | 18.011 | 17.221 | 9.129 |
| 9 | 18.54 | 17.351 | 9.909 | 19 | 17.971 | 17.196 | 9.120 |
| 10 | 18.50 | 17.29 | 9.718 | 20 | 17.206 | 16.418 | 8.673 |
| Average | | | | | *18.294* | *17.180* | *10.25* |

### 6.2 Stream adaptation at server

At the server, a program thread is created for receiving the feedback message from the client and choosing corresponding matching steps. The live action corresponds to the adjustment of outgoing video stream parameters. The most commonly used session parameters are spatial and temporal resolution (frames per second). Initially, streaming is carried out in QCIF mode at 30 frames per second (fps).

$S$ = spatial resolution; $T$ = temporal resolution
$S$ = {SQCIF, QCIF, CIF, QVGA}, $T = \{T_1, T_2, T_3, T_4\}$

### 6.3 Server side algorithm

1. Initially set the configuration to stream in QCIF at $T_2$
2. **Repeat**
3. Receive status from the client
4. If feedback *message_1*
5. Increase both spatial and temporal resolution
6. Else if feedback *message_2*
7. Increase temporal resolution
8. Else if feedback *message_3*
9. Increase spatial resolution
10. Else if feedback *message_4* Continue with the existing spatial and temporal resolution
11. **Do** until connection with the client is terminated

## 7 Implementation environment

The proposed algorithm was implemented using VLCJ and JPCAP in Java environment. Airtel 4G LTE category

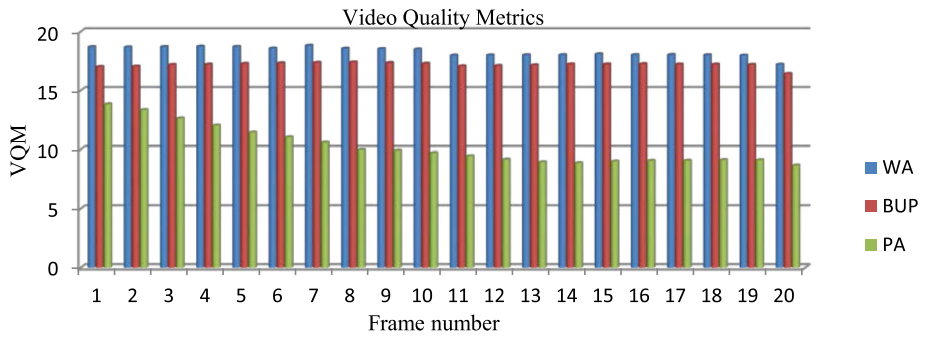


**Fig. 8** The VQM score. The VQM value may be normalized, but an absolute value corresponding to the three techniques is depicted here, where lower value signifies less distortion in the decoded video

4 hotspot device was used as last mile connectivity in Internet to the client. Performance measure using standard full-reference metrics were carried out offline at the end of real-time experimentation.

### 7.1 Video streaming using VLCJ

A native VLC media player instance can be embedded into a Java AWT window or Java Swing JFrame using VLCJ project which is a complete Java framework. A higher level of framework hiding a lot of working complexities can be provided using LibVLC in addition to the simple bindings that it provides. Various types of rich media applications, like audio players, embedding native video output of movie players in a Java component, network streaming clients and servers, webcam clients, and video processing applications, can be created using VLCJ [23].

### 7.2 Buffer implementation using JPCAP

JPCAP [24] is a network packet capture library for applications written in Java. It is used to analyze the real-time network data analysis. It is used in the proposed system to create a buffer of a given size and continuously perform read and write process and store the buffer size for every second in an array and further perform analysis on this array for predicting the buffer capacity using our algorithm.

## 8 Results and discussion

The proposed system uses a non-linear regression algorithm based on exponential non-parametric adaptation (PA) method, which was implemented on Airtel 4G LTE category 4 wireless networks in client-server environment.

Another algorithm based on BUP was implemented to compare with the proposed work. Additionally, system performances were also observed with default mechanism (RTP-RTCP, RTSP) for streaming in the client-server system, i.e., without any adaptation (WA) method.

### 8.1 Peak signal-to-noise ratio

The PSNR was measured offline by the comparing the received live webcam video on client side and the original captured video on the server side. This is carried out by splitting the videos into frames and by comparing each frame on the server side with its corresponding frame on the client side. An average PSNR of 30.6923 dB was observed using the proposed algorithm with non-parametric exponential non-linear regression (PA), 20.44 dB without adaptation (WA), and 24.65456 dB using buffer algorithm based on BUP approach.

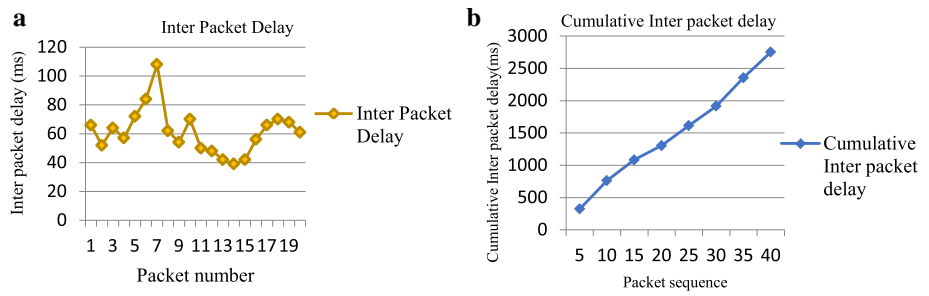


**Fig. 9** Packet arrival delay. **a** The inter-packet delay. **b** The cumulative inter-packet delay


**Fig. 10** Selected frames of live video streams. **a** Original frames. **b** Decoded frames

The average PSNR of the proposed algorithm is 24.48% higher (Fig. 6) than the buffer underflow probability. The average PSNR of the proposed algorithm is 24.48% higher (Fig. 6) than the buffer underflow probability algorithm and 33.39% more than without adaptation. The PSNR values obtained for the different frames are listed in Table 1.

### 8.2 Structural similarity index

The SSIM index is used to measure of how much the transmitted video is structurally similar with the video received on the client side. It computes the SSIM by splitting the server and client side live webcam video into frames. An average SSIM index of 0.953 was obtained using the proposed algorithm based on non-parametric regression, 0.867 without adaptation (WA), and 0.893 using buffer algorithm based on buffer underflow probability. The average SSIM of the proposed algorithm is 6.63% higher than the BUP algorithm and 9.91% greater than without adaptation. The SSIM index values for the different frames are listed in Table 2 along with its values plotted in Fig. 7.

### 8.3 Video quality measurement score

An average VQM score of 10.252 is obtained using the proposed algorithm based on non-parametric regression and exponential non-linear regression (PA), 18.294 without adaptation (WA), and 17.180 using buffer algorithm based on BUP.

The average VQM value of the proposed algorithm is 40.32% lesser than that of the buffer underflow probability algorithm and 43.95% lesser than that without adaptation. The VQM values for the different frames are listed in Table 3 along with its corresponding line graph plotted in Fig. 8.

### 8.4 Inter-packet delay

The time difference in selected packets in a flow with the lost ignored packets at end-to-end one-way delay is referred to as packet delay variation/jitter. It is a an

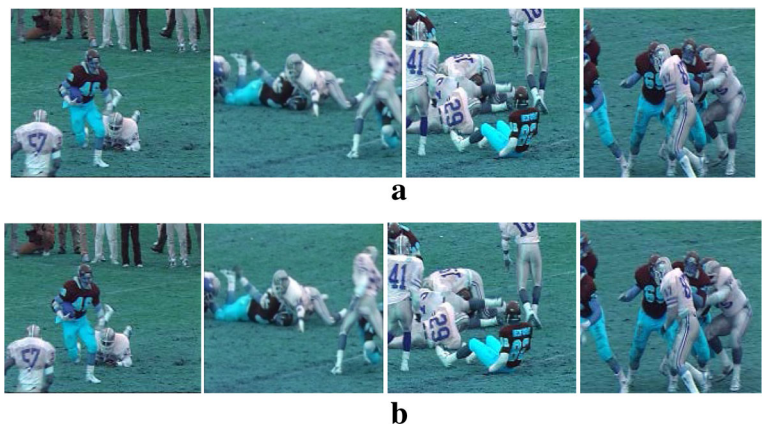


**Fig. 11** Selected frames of stored video football.yuv (https://media.xiph.org/video/derf/). **a** original frames. **b** Decoded frames

important component of quality of service in live video streaming because a high jitter may lead to problems like lip sync and can cause loss of many packets. Thus, it is important to maintain a constant inter-packet delay. The proposed algorithm causes an average inter-packet delay of 61.55 ms. The inter-packet delay graph is given by Fig. 9(a) and a cumulative inter-packet delay is plotted in Fig. 9(b).

### 8.5 Some selected visual frames

Figure 10 shows the original and decoded frames of live video. For 8 bit depth video system, the desirable values of PSNR lies between 30 and 50 dB [25, 26]. Since the proposed streaming system maintains an average PSNR of 30.69 dB at the client, there is no noticeable distortion or damage in the processed frames.

The sample of the live video presented here corresponds to random capturing of frame during streaming. Since the system dynamically adjusts spatial and temporal resolution, its value is not specified here. The experiment results were also validated with stored video: football_qcif_15fps.y4m [21]. This video is in YUV 4 Mpeg raw, color-sensitive format which consists of sequence of uncompressed YCbCr images frame by frame. Figure 11 shows few frames of the original video (football_qcif_15fps.y4m) of frame size $176 \times 144$ at 15 fps corresponding to Quarter Common Interchange Format (QCIF) and corresponding decoded sequences.

## 9 Conclusions

The proposed algorithm on the client side uses the regression concept for predicting the future buffer fullness for a fixed interval of time. The feedback sent to the server depends upon the average buffer fullness state and their trend in the given time interval. The videos were transcoded with varying quality levels to match maximum achievable quality depending upon the available bandwidth and the feedback from the client for every stipulated time interval. Separate sockets for binding packets were created separately for sending/receiving feedback messages. The test result on different video quality metrics validated the practical usefulness of the proposed algorithm. Although proposed system is ready to support entertainment videos, it is extended to accommodate the telemedicine applications.

Future plan includes extending live adaptive video streaming using the buffer overflow/underflow adaptation for multicast delivery. The proposed work can be optimized further in client-server loop using principles of control theory. Future work also includes implementation and testing algorithm in two-way video communications. Applications like telemedicine requires better performance from such video streaming systems.

**Acknowledgements**
We like to thank the University Grant Commission (UGC), Government of India, New Delhi, for providing funds to create infrastructure needed to implement the proposed research work and also giving permission to publish the research work.

**Authors' contributions**
AR contributed to the main part of this manuscript. DK supervised the whole work, offered useful suggestions, and helped to modify the manuscript. IH and AS participated in the code development of the proposed system. SA participated in the discussion of this work. All authors read and approved the final manuscript.

**Competing interests**
The authors declare that they have no competing interests.

## Publisher's Note



Received: 17 August 2016 Accepted: 6 June 2017
Published online: 20 June 2017

**References**

1. Cisco, "Cisco Visual Networking Index: global mobile data traffic forecast update, 2014–2019," (2016)
2. A Chadagorn, I Khalil, C Cameron, Z Tari, PileCast: multiple bit rate live video streaming over bit torrent". J Netw Comput Appl **39**, 167–178 (2014)
3. B Cheng, J Yang, S Wang, J Chen, Adaptive video transmission control system based on reinforcement learning approach over heterogeneous networks. IEEE Transaction on Science and Engineering **12**(3), 1104–1113 (2015)
4. M Zhao, X Gong, J Liang, W Wang, X Que, S Cheng, QoE-Driven cross-layer optimization for wireless dynamic adaptive streaming of scalable videos over HTTP". IEEE Transactions on Circuits and Systems for Video Technology **25**(3), 451–465 (2015)
5. O Oyman, S Singh, "Quality of experience for HTTP adaptive streaming services". IEEE Commun Mag **50**, 20–27 (2012)
6. C Zhou, C-W Lin, Senior Member, IEEE, Z Guo, "mDASH: A Markov decision-based rate adaptation approach for dynamic HTTP streaming," IEEE Transactions on Multimedia **18**(4), 738–751 (2016)
7. S Hammer, MPEG DASH: the enabler standard for video delivery over the Internet". Motion Imaging Journal, SMPTE **121**(5), 40–46 (2015)
8. S Chen, J Yang, Y Ran, E Yang, Adaptive layer switching algorithm based on buffer underflow probability for scalable video streaming over wireless networks. IEEE Transactions on Circuits and Systems for Video Technology **26**(6), 1146–1160 (2015)
9. Y Jian, H Han, X Hongsheng, H Lajos, "Online buffer fullness estimation aided adaptive media playout for video streaming". IEEE Transactions on Multimedia **13**(5), 1141–1153 (2011)
10. S Yong Tao, P Goran, H Thorsten, *"Server-driven rate control for adaptive video streaming using virtual client buffers," IEEE Fourth International Conference on Consumer Electronics Berlin*, 2014
11. S Fernandes, J Kelner, D Sadok, An adaptive predictive architecture for video streaming servers". J Netw Comput Appl **34**, 1683–1694 (2011)
12. S Petrangeli, T Wauters, R Huysegems, T Bostoen, F De Turck, *Network-based dynamic prioritization of HTTP adaptive streams to avoid video freezes" IEEE International Symposium on Integrated Network Management*, 2015
13. K-L Hua, G-M Chiu, H-K Pao, Y-C Cheng, "An efficient scheduling algorithm for scalable video streaming over P2P networks," Elsevier Computer Networks **57**(14), 2856–2868 (2013)
14. Han-Chiang Li1, Kate Ching-Ju Lin, Kai-Lung Hua, Ge-Ming Chiu, Yu-Chin Tsai, and Shan Chin, "Dependency-aware quality-differentiated wireless video multicast," IEEE Wireless Communications and Networking Conference (WCNC): Networks (2013)
15. S Pudlewski, N Cen, Z Guan, T Melodia, "Video transmission over lossy wireless networks: a cross-layer perspective" in IEEE Journal of Selected Topics in Signal Processing **9**(1), 6–21 (2015)
16. KL Hua, I Pollak, M Comer, *"optimal tilings for image and video compression" IEEE Fortieth Asilomar Conference on Signals, Systems and Computers*, 2006

17. K-L Hua, R Zhang; M Comer, I Pollak, "Inter frame video compression with large dictionaries of tilings: algorithms for tiling selection and entropy coding," IEEE Transactions on Circuits and Systems for Video Technology **22**(8), 1136–1149 (2012)
18. N-S Vo, TQ Duong, H-J Zepernick, M Fiedler, "A cross-layer optimized scheme and its application in mobile multimedia networks with QoS provision," IEEE System Journal **10**(2), 817–830 (2016)
19. M Hassan, M Krunz "Video streaming over wireless packet networks: an occupancy-based rate adaptation perspective," IEEE Transactions on Circuits and Systems for Video Technology **17**(8), 1017–1027 (2005)
20. Luca De Cicco and Saverio Mascolo "An Adaptive Video Streaming Control System: Modeling, Validation, and Performance Evaluation" in IEEE Transactions on Networking. **22**(2) (2014)
21. M Mrak, M Grgic, S Grgic, *"Scaleable video coding in network applications" International Symposium on Video/Image processing and Multimedia Communications*, 2002
22. Y-M Hsiao, C-H Chen, J-F Lee, Y-s Chu, "Designing and implementing a scalable video-streaming system using an adaptive control scheme" in IEEE Transactions on Consumer Electronics **58**(4), 1314–1322 (2012)
23. VLCJ framework, https://wiki.videolan.org/Documentation:Streaming_HowTo/
24. Java framework for VLC media player, https://github.com/caprica/vlcj
25. ST Welstead, "Fractal and wavelet image compression techniques," SPIE Publication, vol. TT40, pp. 155–156, ISBN 978-0-8194-3503-3, (1999)
26. J up, R Hamzaoui, DS Barni, M Barni, ed, "Document and image compression," CRC Press. 968, pp. 168–169, ISBN 9780849335563, (2006)