Results for a statistically optimal algorithm for multimedia receiver buffers

B.H. Hafskjold Gade

Abstract: For interactive multimedia and multimedia streams, receiver playout buffers are required to smooth network delay variations. Instead of using a constant playout speed, newer receiver buffer algorithms control the playout speed, which can give a lower end-to-end delay and fewer packets that are lost because of late arrivals. This paper presents a statistically optimal algorithm to control playout speed. The most significant difference to other published playout speed adjusting algorithms is the thorough mathematical approach that this work is based on. A stringent notation and stringent mathematical models of the media receiver system have been developed, which are generic and independent of the networks and protocols used. This has enabled us to deduce the statistically optimal controller for the playout speed, which is also independent of the networks and protocols used. Three deviations from perfect playout have been identified: (i) buffering delay (ii) a playout rate different from the sender rate and (iii) a change of playout rate. Our approach is statistically optimal by minimising the three deviations, based on their relative importance. The importance will vary for different user and application requirements, and is thus freely tunable by means of three weight factors. The optimal control algorithm is easy to implement and has demonstrated very good results when evaluated by perceptual evaluation of speech quality, an objective technique for measuring voice quality, and degradation mean opinion score, a subjective listening test, for both simulated and real network measurement traces.

1 Introduction

For a user of a media stream, the perceived quality consists of the delay (which is especially important for interactivity) and the listening-only or viewing-only audio or video quality.

When a stream of media is sent through a network, the packets in the stream will be individually delayed. Therefore, a reception buffer at the receiver machine is necessary to protect against playout interruptions because of variations in the data arrival rate. While the amount of protection offered grows with the size of the client's buffer, so does the extra delay that is introduced. A playout buffer algorithm is used to find a compromise between the delay and the listening-only or viewing-only media quality. The most commonly used playout buffer algorithms for voice are fixed playout delay and adaptive playout delay.

Fixed playout delay gives every packet a constant end-to-end delay d, and thus uses a constant playout speed. Packets arriving after their deadline are considered lost. This algorithm does not take into consideration the delay change that most networks experience. If d is set to a value close to the mean network delay in a network with varying delay, a conversation may be impossible to make, since almost half of the packets may arrive too late, and therefore considered lost. If the delay on the other hand is set much larger than the mean network delay, unnecessary delay is introduced.

Adaptive playout delay is an improvement, valid for speech only, where each talkspurt, numbered i, gets its own end-to-end delay d_i. Much research has been done on between-talk-spurt-adjustment to find a good value of d_i [1–12]. All these versions of between-talk-spurt-adjustment have the same problems as fixed playout delay for long talkspurts and other media without pauses, like music.

By modifying synchronised overlap-and-add to scale individual voice packets [13] has adapted the adaptive playout delay algorithm, to enable it to handle delay spikes in the middle of a talkspurt.

To solve the problems related to long talkspurts and other media without pauses, the most recent playout buffer algorithms control the playout speed of the media to be able to find a better compromise between playout interruptions and added delay [14–17].

One of the best ways to change the playout speed of sound may be [18] where a time-domain interpolation waveform similarity overlap-add (WSOLA) is modified to scale individual packets, and where the playout speed can be changed without changing the pitch ([18] indicates good voice quality with a stretch or compression of 25% of the inter-packet-time). For video, the playout speed can be controlled by changing the holding time of each picture.

The algorithms presented in [14, 15] (which are meant for packet video receivers) are both reported as having buffering delays above 0.8 s, and are thus not suited for interactive communication. The algorithm presented in [16], which uses fuzzy networks, is compared to the optimal control algorithm introduced in this paper, in Section 6. The algorithm presented in [17] calculates the scaling of each packet based on the network delay during the last w packets, where w is a parameter that is used as a trade-off between accuracy

doi:10.1049/iet-com:20060583

Paper first received 20th October 2006 and in revised form 9th May 2007

The authors are with Norwegian Defence Research Establishment, PO Box 25, Kjeller N-2027, Norway

E-mail: brita.gade@ffi.no

and responsiveness. The optimal control algorithm presented in this paper is more general, in that it gives the user or application the ability to control the playout quality by setting three different weight factors.

A perfect playout that is a playout with no buffering delay and with a perfect listening-only or viewing-only quality (where the playout rate is equal to the sender rate at all times), cannot be obtained as long as the network introduces jitter. However, the deviations from the perfect playout can be minimised. The deviations from the perfect playout that may be experienced by a user are: (i) buffering delay, and listening-only or viewing-only quality deviations, consisting of (ii) a playout rate different from the sender rate and (iii) a change of playout rate. Using a thorough mathematical approach, we aim at finding the statistically optimal control of the playout speed that minimises the three deviations from the perfect playout, based on their relative importance. The two main steps towards the statistically optimal controller is the development of a strict notation and strict mathematical models, which are independent of the network and protocols, and general enough to fit any kind of playout buffer algorithm. The next step is to deduce the optimal controller.

Much work has been performed on packet loss concealment techniques [19–24]. However, the optimal control algorithm is not a packet loss concealment technique, but a statistically optimal control of buffering delay by controlling playout speed, and is normally without packet loss.

This paper presents results for voice and music, but the mathematics presented is independent of the medium.

2 Mathematical modelling

We cannot use the terms bits or bytes to express the amount of media in a flow, since two equal time intervals in a flow can contain a very different number of bytes (for instance, because of a different level of compression). Therefore, we introduce the term media-unit to define the amount of media corresponding to a constant period of time when playing the media at the correct media speed. One example is a 50 pictures/s–video, where it would be most intuitive to define a media-unit as 20 ms of media.

Fig. 1 contains an illustration of the stringent mathematical models of the media receiver system that were needed to deduce the statistically optimal control of playout speed. For a more thorough motivation and description of the model [25]. The notation used in this paper is summarised in Section 11. As illustrated in Fig. 1, a media-unit is first sent from the sender to the transport segment (consisting of all networks and protocols between the sender application and the receiver playout buffer) with the correct media speed r_{SNDR}. The media stream through the transport

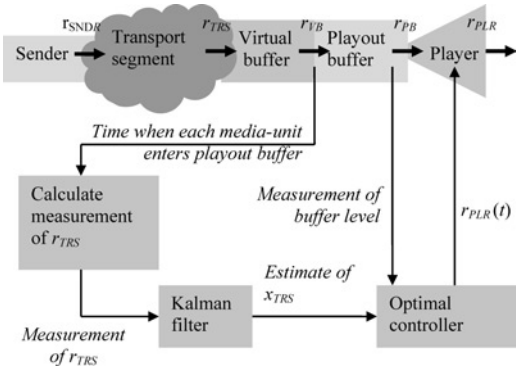

Fig. 1 *Total system with optimal controller*

segment is modelled as a continuous stream. We therefore introduce a virtual buffer as a mathematical converter from the continuous stream out of the transport segment (with the rate $r_{TRS}(t)$) to the whole packets into the playout buffer. The function of the playout buffer is to smooth the jitter, and feed the player at the rate $r_{PB}(t)$. The player plays the media at the rate $r_{PLR}(t)$. The virtual buffer (which does not represent any physical entity) and the player work as counterparts; the virtual buffer converts the continuous rate from the transport segment to whole packets, and the player converts the whole packets from the playout buffer to a continuous playout rate.

The number of media-units in the receiver buffers is $M_{VB}(t)$ for the virtual buffer, $M_{PB}(t)$ for the playout buffer and $M_{PLR}(t)$ for the player buffer. The total number of media-units in the receiver buffers is

$$M_{RCV}(t) = M_{VB}(t) + M_{PB}(t) + M_{PLR}(t).$$

The total state-space model (A state space model is a mathematical model of a system as a set of input, output and state variables related by first-order differential equations. The state variables are expressed as vectors and the differential and algebraic equations are written in matrix form. The state space model is a convenient and compact way to model and analyse general systems with multiple inputs and outputs. A good textbook on vectors and matrixes is [26]. Two textbooks on state space modelling are [27] and [28].) for our system is (for a detailed derivation, see [25] or [30])

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}u + \boldsymbol{C}\boldsymbol{v}_{TRS},$$

where $\dot{\boldsymbol{x}} = \mathrm{d}/\mathrm{d}t(\boldsymbol{x})$,

$$\boldsymbol{x} = \begin{bmatrix} M_{RCV} \\ r_{PLR} - r_{SNDR} \\ \boldsymbol{x}_{TRS} \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} 0 \\ 1 \\ \boldsymbol{0}_{n_{TRS}\times 1} \end{bmatrix}, \boldsymbol{C} = \begin{bmatrix} 0 \\ 0 \\ \boldsymbol{C}_{TRS} \end{bmatrix}$$

$$\text{and } \boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{1,RCV} & \boldsymbol{A}_{2,RCV} \\ \boldsymbol{0}_{n_{TRS}\times 2} & \boldsymbol{A}_{TRS} \end{bmatrix}$$

$$\text{where } \boldsymbol{A}_{1,RCV} = \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix}, \boldsymbol{A}_{2,RCV} = \begin{bmatrix} 1 & \boldsymbol{0}_{1\times(n_{TRS}-1)} \\ 0 & \boldsymbol{0}_{1\times(n_{TRS}-1)} \end{bmatrix}$$

and n_{TRS} is the number of states in the state-space equation for the transport segment (x_{TRS}), $\boldsymbol{0}_{a\times b}$ is a zero matrix with dimension a times b, and the control variable u is $u = r_{PLR}$. The time derivative of the first state $M_{RCV}(t)$ (i.e. the time derivative of the number of media-units in the receiver buffers) is equal to the difference between $r_{TRS}(t)$ (the rate from the transport segment into the receiver buffers) and $r_{PLR}(t)$ (the rate out of the receiver buffers). This is equal to the difference between state 2, $(r_{PLR} - r_{SNDR})$ and state 3, $(r_{TRS} - r_{SNDR}$, explained below). Thus, the first row of the system matrix $\boldsymbol{A}$ is $\begin{bmatrix} 0 & -1 & 1 & \boldsymbol{0}_{1\times(n_{TRS}-1)} \end{bmatrix}$. Since r_{SNDR} is constant, the time derivative of state 2 is equal to the control variable $u = r_{PLR}$. Thus, the second row of $\boldsymbol{A}$ contains zeros and the second row of $\boldsymbol{B}$ is 1.

The state-space model for the transport segment is $\dot{\boldsymbol{x}}_{TRS} = \boldsymbol{A}_{TRS}\boldsymbol{x}_{TRS} + \boldsymbol{C}_{TRS}\boldsymbol{v}_{TRS}$, where $\boldsymbol{x}_{TRS}$ is the state vector, $\boldsymbol{A}_{TRS}$ is the system matrix and $\boldsymbol{C}_{TRS}$ $\boldsymbol{v}_{TRS}$ expresses the system noise, where $\boldsymbol{v}_{TRS}$ is a vector of uncorrelated Gaussian white noise with zero mean and unit variance. The first state of the state vector $\boldsymbol{x}_{TRS}$ is $(r_{TRS} - r_{SNDR})$, and the rest of the states are given by the specific model used: $x_{TRS} = \begin{bmatrix} r_{TRS} & -r_{SNDR} & \dots \end{bmatrix}^T$. One can usually

obtain a good transport segment model even with a low number of states in $\boldsymbol{x}_{\mathrm{TRS}}$. The transport segment state-space model can either be a general model (in this paper, we have used a simple model with $n_{\mathrm{TRS}} = 2$), or given by the user of the optimal algorithm (e.g. the application programmer) who may use guidelines from [25] to find the model. As stated in Section 8, an automatic detection algorithm could be used to find the transport segment state-space model.

3 Optimal control

Mathematically, the three deviations from perfect playout (mentioned in Section 1) can be minimised by minimising $|M_{\mathrm{RCV}}(t) - M_{\mathrm{RCV},d}|$ (where $M_{\mathrm{RCV},d}$ is the desired receiver buffer level), $|r_{\mathrm{PLR}}(t) - r_{\mathrm{SNDR}}|$ and $|\dot{r}_{\mathrm{PLR}}(t)|$. Since these minimisations are conflicting, we introduce the weight factors:

w_1: the importance of minimising $|M_{\mathrm{RCV}}(t) - M_{\mathrm{RCV},d}|$
w_2: the importance of minimising $|r_{\mathrm{PLR}}(t) - r_{\mathrm{SNDR}}|$
w_3: the importance of minimising $|\dot{r}_{\mathrm{PLR}}(t)|$

The user of the system, or the application programmer, will feed these weight factors to the optimal controller to get the desired playout quality. The optimal control algorithm will find the optimal compromise between playout buffering delay (with the weight w_1) and listening-only or viewing-only quality (with the weights w_2 and w_3). An optimal control guideline from [29] is to give the weight factors a magnitude relative to the expected (or nominally acceptable) value of the variable to be minimised. We have used $w_i = 1/(\Delta x_i)^2$, where Δx_i is the nominally acceptable value of $|M_{\mathrm{RCV}}(t) - M_{\mathrm{RCV},d}|$ for $i = 1$, of $|r_{\mathrm{PLR}}(t) - r_{\mathrm{SNDR}}|$ for $i = 2$ and of $|\dot{r}_{\mathrm{PLR}}(t)|$ for $i = 3$. The desired buffer level can be set by the algorithm described in [31] or by the guidelines in [25].

Note that the words 'statistically optimal' in the title does not refer to the results presented, but to the statistically optimal control algorithm presented in this section. Statistical optimality means that no other algorithm will have smaller deviations from the perfect playout that is the output of the algorithm is the playout speed that will give the statistically optimal results based on the three weight factors given by the user.

The statistically optimal controller is given by (for a detailed derivation, see [25] or [30]) $\boldsymbol{u}(t) = \boldsymbol{G}\boldsymbol{x}(t)$ where

$$\boldsymbol{G} = \begin{bmatrix} -r_{12}/w_3 & -r_{22}/w_3 & \boldsymbol{r}_{2\mathrm{B}}/w_3 \end{bmatrix},$$

where

$$r_{12} = -\sqrt{w_3 w_1},\ r_{22} = \sqrt{w_3(w_2 + 2\sqrt{w_3 w_1})}$$

and

$$\boldsymbol{r}_{2\mathrm{B}} = w_3\left(\begin{bmatrix} r_{11} & \boldsymbol{0}_{1\times(n_{\mathrm{TRS}}-1)} \end{bmatrix} + \begin{bmatrix} r_{12} & \boldsymbol{0}_{1\times(n_{\mathrm{TRS}}-1)} \end{bmatrix}\boldsymbol{A}_{\mathrm{TRS}}\right) \cdot \left(r_{12}\boldsymbol{I}_{n_{\mathrm{TRS}}\times n_{\mathrm{TRS}}} - w_3\boldsymbol{A}_{\mathrm{TRS}}^2 + r_{22}\boldsymbol{A}_{\mathrm{TRS}}\right)^{-1}$$

where $r_{11} = -r_{12}r_{22}/w_3$.

As shown in Fig. 1, a Kalman filter [32] is used to find the estimate of $\boldsymbol{x}_{\mathrm{TRS}}(t)$, needed by the optimal controller. The input to the Kalman filter is a calculated measurement of r_{TRS}, obtained by dividing the number of media-units arriving during a short time interval by the length of the time interval.

We have developed a network simulator and an implementation of the total system, shown in Fig. 1, in Matlab (www.mathworks.com), to produce results for both simulated and real network data.

4 Quality metrics

Subjective methods for measuring listening-only sound quality are the mean opinion score (MOS) [33] and degradation MOS (DMOS), described in Section 4.1. Objective methods include perceptual evaluation of speech quality (PESQ) [34], described in Section 4.2 and late packet loss rate, described in Section 4.3. This paper also uses a dissimilarity measure, described in Section 4.4.

Listening-only tests should be combined with the receiver buffer level when used to compare different algorithms, for example by assuring that the mean buffer levels of all algorithms are equal during the tests.

For algorithms that use constant playout speed, existing quality metrics [35, 36] and performance bounds [37] that combine the effect of buffering delay and late packet loss rate, can be used.

4.1 MOS and DMOS

For MOS (defined by Annex B of [33]), subjects rate the voice quality as 'excellent', 'good', 'fair', 'poor' or 'bad', on a scale from 5 to 1.

The MOS score tends to lead to low sensitivity in distinguishing among good quality circuits. DMOS is a modified version, defined by Annex D of [33], which affords higher sensitivity. Here, the test persons hear the correct sound, followed by a short period of silence, and then the output sound from the system to be tested. The test subjects rate the degradation of the output sound as 'inaudible', 'audible but not annoying', 'slightly annoying', 'annoying' or 'very annoying', on a scale from 5 to 1.

We collected two male and two female voice samples from [38], and two music samples; one from Beethoven's 9th symphony and one from David Byrne's 'Like humans do'.

The samples were scaled (using WSOLA) according to the playout speed output of different algorithms. For incidents of packet loss or run-dry (where the playout speed is zero) silence was replaced by a low amplitude white noise.

The test was performed by 14 test persons according to the DMOS standard described in Annex D of [33]. All sound files used 16-bit mono PCM encoding, sampled at 44 100 Hz (CD quality is 44 100 Hz, 16-bit stereo PCM encoding and regular telephone quality is 8000 Hz 8-bit mono PCM encoding.), because we would like our algorithms to work for all quality levels. VoIP and other sound transmitted over networks may also have higher quality in the future.

4.2 Perceptual evaluation of speech quality

ITU-T Recommendation P.862 [34] describes PESQ as an objective alternative to MOS for measuring voice quality. PESQ is a computer program that compares an original sound signal $X(t)$ with a degraded signal $Y(t)$. The output of PESQ is a prediction of the MOS score that test persons would give to $Y(t)$.

The PESQ scores in this paper were obtained by using 16 000 Hz sound files, since we use the reference implementation of PESQ that works for 8000 and 16 000 Hz sampling frequencies. The 16 000 Hz sound files were obtained from the 44 100 Hz files from [38] by using the Matlab (www.mathworks.com) command 'resample'. We did not use PESQ for music samples, since it is defined only for voice.

According to Liu *et al.* [39], PESQ is very sensitive to stretching and compression of the sound signal. For a

speech signal with much stretching and compression, where WSOLA was used to change the playout speed without changing the pitch [39], reports that subjective listening tests showed very good hearing results, but that PESQ gave an average score of 3.2.

4.3 Packet loss and run-dry incidents

Many of the published playout buffer algorithms discard packets that arrive after a deadline. The rate of discarded packets to the total number of packets is called the late packet loss rate. The optimal control algorithm normally does not lose packets, but may experience incidents where the buffer runs dry. The corresponding run-dry rate is used as a quality metric in this paper.

A run-dry incident with the duration of one media-unit will affect the sound quality less than the loss of a packet containing one media-unit, since no information is lost during a run-dry incident. Thus, with otherwise equal quality, a speech signal with x% run-dry rate will probably have a higher quality than a speech signal with x% late packet loss.

4.4 Arentz dissimilarity measure

Content-based retrieval is an active research area, where methods are developed for searching for contents contained in digital text, sound, music, image and video and so on. One of the research areas within content based musical retrieval is Query-by-Humming systems. Arentz *et al.* [40] have developed the following dissimilarity measure (for Query-by-Humming systems) between two pieces of music a and b

$$d(a, b) = \sum_{j=1}^{i} \omega(a_{j-1}, a_j, b_{j-1}, b_j)^2 \quad (1)$$

where i is the number of notes in the tune and $\omega(a_k, a_l, b_m, b_n)$ represents the cost of pairing up the note pair a_k, a_l in tune a with the note pair b_m, b_n in tune b. The cost function is defined as (Arentz *et al.* [41] used a constant scaling factor to compensate for tempo differences between the two tunes. Since we compare two traces with identical long term tempo, the scaling factor is not used (i.e. it is set equal to 1) in this paper.) $\omega(a_k, a_l, b_m, b_n) = (t(a_l) - t(a_k)) - (t(b_n) - t(b_m))$, where $t(s_i)$ is the timestamp for the given note $s_i \in s$.

We use this measure to calculate the dissimilarity between the original sound played at the correct media speed r_{SNDR}, and the resulting sound with playout speed $r_{\mathrm{PLR}}(t)$. We calculate the cost function for an integer (i) number of media-units, as the time difference between the correct playout time period and the actual time period used to play the i media-units. Equation (1) is used to calculate the total dissimilarity measure for the playout period as the sum of these costs. The dissimilarity measure given by (1) is dependent upon the length of the two tunes a and b. Therefore, in this paper, the dissimilarity per second will be used as the quality measure.

5 DMOS, PESQ and Arentz tests

In this section, we have run the listening-only tests DMOS, PESQ and Arentz on three different algorithms. To be able to rightfully compare the three algorithms, both the buffering delay and the listening-only quality must be taken into consideration. We have adjusted the parameters of all three algorithms to make their mean buffer levels equal, to be able to compare the algorithms by comparing the results of the listening-only tests.

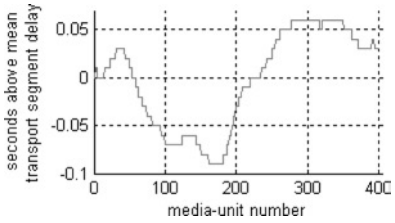

Fig. 2 *Transport segment delay for simulated transport segment*

5.1 DMOS and PESQ tests

This section compares the results from three different algorithms.

Algorithm 1: is one of the most commonly used algorithms (fixed playout delay), with a constant playout speed, which may drop to zero if packets arrive after their deadline.

Algorithm 2: was published in a 'to be submitted' version of [17], and is chosen here because it is the only playout speed adjusting algorithm we have found that is documented well enough to be implemented. For playout buffer levels above a target level, the inter packet time (IPT) is set to f*normal_IPT, where $f < 1$, and for buffer levels below the target level, the IPT is set to s*normal_IPT, where $s > 1$. We use the suggested values $s = 1.25$ and $f = 0.75$.

Algorithm 3 is the optimal control of playout speed.

One simulated and one real transport segment trace are used. The transport segment delay for the simulated transport segment is shown in Fig. 2. The real trace will be presented as trace 1 in Section 6.

Fig. 3 shows the DMOS and PESQ results for algorithms 1, 2 and 3, where all algorithms have the same mean buffer level. DMOS results are presented for two music samples, four speech samples and the mean of the speech samples.

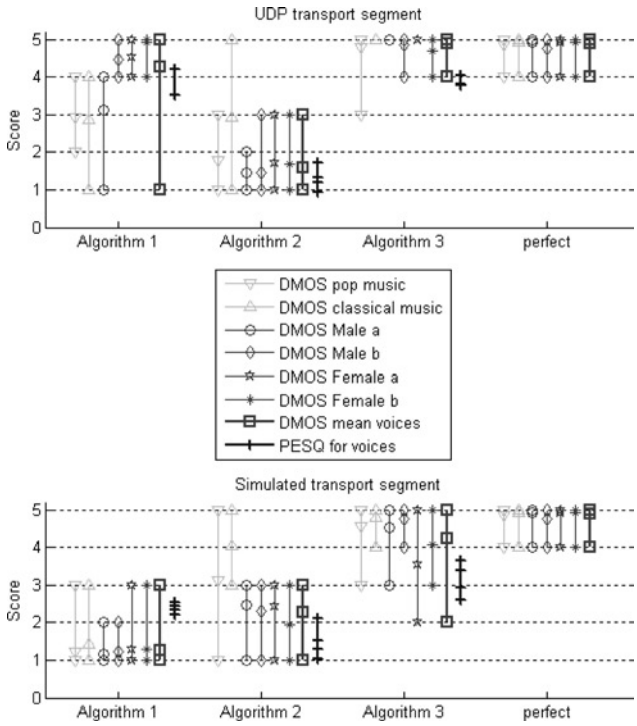

Fig. 3 *DMOS and PESQ results*

PESQ results are calculated for each of the four voice samples.

The DMOS results are presented using markers, connected by lines, at the minimum value, the mean value and the maximum value. The PESQ results include only four calculated scores, and are therefore presented by markers for each value.

Since the rating of the perfect sound is not dependent upon the transport segment used, the results shown for perfect sound in the two graphs of Fig. 3 are equal. Note that DMOS and PESQ use different quality scales (Sections 4.1 and 4.2).

Algorithm 1 discards packets that arrive after their deadline. For the simulated transport segment, one second of sound was lost due to late packet arrivals, but for the User Datagram Protocol (UDP) transport segment, only short periods of sound were lost. These short periods happened during periods of no sound (between talkspurts) or low sound for three of the voice samples, which therefore received high DMOS scores. The music samples did not contain any low-sound periods, and thus the information loss was easily heard, giving lower DMOS scores.

The one-second period where algorithm 1 discarded all packets from the simulated transport segment resulted in low DMOS scores for both speech and music. PESQ gave a higher score than DMOS, thus it seems that PESQ is less sensitive to loss of sound or information than DMOS.

For algorithm 2, the rate change caused by the transport segment is small compared to the rate change caused by the algorithm, since algorithm 2 switches the playout speed very frequently between 20% below and 33% above the correct media speed. Thus, as shown by Fig. 3, algorithm 2 received low DMOS scores for the voice samples for both transport segments. The music samples received higher DMOS scores than the voice samples (with large variations), thus for most test subjects, the frequent speed changes were less disturbing for music than for voice. As expected (since PESQ is sensitive to stretching and compression, as explained in Section 4.2), PESQ gave a lower score than DMOS for algorithm 2 for both transport segments.

The optimal control algorithm (algorithm 3) received high DMOS scores for both voice and music, which for the UDP transport segment were comparable to the scores of the perfect sound. The average scores of the voice samples are also equal to the corresponding score of the perfect sound. Algorithm 3 uses stretching and compression, but without the frequent changes of playout speed that are present in algorithm 2. As expected, PESQ gave a lower score than DMOS also for algorithm 3, because changes in playout speed are still present.

For the UDP transport segment, algorithm 1 received a slightly higher PESQ score than algorithm 3. Algorithm 1 received a high PESQ score because the late packet loss happened during periods of low sound or no sound for the voice samples, and algorithm 3 received a lower PESQ score because the PESQ algorithm is very sensitive to the stretching and compression of algorithm 3. In the DMOS test, however, algorithm 3 received a 0.6 point higher score than algorithm 1.

5.2 Results for Arentz dissimilarity measure

Since, as described in Section 4.4, Arentz dissimilarity measure costs are calculated based on the output results from running the different algorithms, and not based on sound files, only one cost is calculated for each combination of transport segment and algorithm.

To be able to roughly compare the scores from DMOS, PESQ and Arentz dissimilarity measure, we have used a common scale from 0 to 1, where 1 represents the best quality. The PESQ and DMOS scores are divided by 5, and the following equation is used for the Arentz dissimilarity measure

$$\text{newscore} = 1 - \frac{\text{dissimilarity measure}}{\text{max dissimilarity measure}} \quad (2)$$

The maximum dissimilarity measure was approximately 0.2.

Fig. 4 shows scaled DMOS and PESQ scores for the mean of the voice samples (equal to the mean values shown in Fig. 3) and for Arentz dissimilarity measure.

Fig. 4 shows that Arentz dissimilarity measure is relatively close to the DMOS score for algorithms 1 and 3, but for algorithm 2, the closeness between DMOS and Arentz dissimilarity measure is very dependent upon the cost period. This is because algorithm 2 changes the playout speed very frequently. Short cost periods lead to high dissimilarity values, since many such periods will have a shorter or longer duration than the perfect duration. Long cost periods lead to low dissimilarity values since many shorter periods of stretching and compression occurs within a long cost period, which will thus have a duration that is relatively close to the perfect duration.

For the 60 ms cost period, the Arentz dissimilarity measure is relatively close to the DMOS score for all algorithms, even closer than the PESQ score. Thus it seems that Arentz dissimilarity cost with a 60 ms cost period may be a good prediction for the DMOS score. Fig. 4 shows only six different combinations of algorithms and networks, thus to draw a better conclusion regarding the use of Arentz dissimilarity measure to predict the DMOS score, more algorithms and transport segments need to be tested with both DMOS and Arentz dissimilarity.

6 Comparison with fuzzy network results

This section uses the same measurement traces as Ranganathan and Kilmartin [16]. They measured the Internet packet delays by transmitting packet streams from

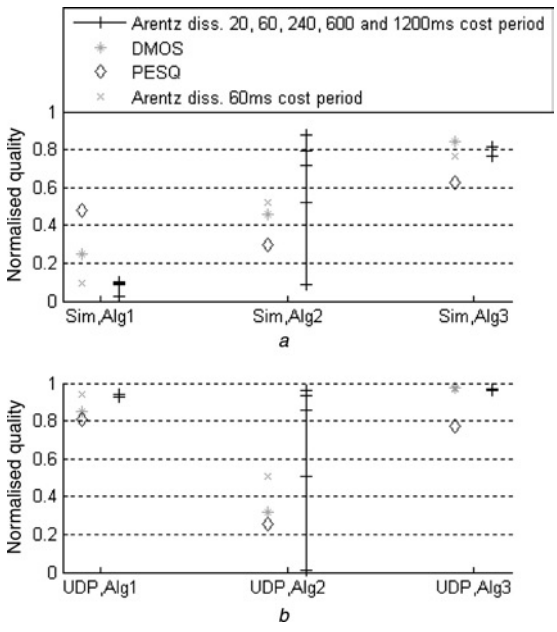

Fig. 4 *Comparison of DMOS, PESQ and Arentz dissimilarity measure*

Table 1: Internet delay traces from Ranganathan and Kilmartin [16]

Trace no.	Internet path	Inter packet interval	Trace date
trace 1	NUIG – DCU	20 ms	28 April 2003
trace 2	NUIG – UNSW	20 ms	30 April 2003
trace 3	NUIG – DCU	40 ms	7 May 2003
trace 4	NUIG – UNSW	40 ms	28 April 2003

a host located at National University of Ireland, Galway (NUIG), Ireland to two other hosts, the first located at University of New South Wales (UNSW), Sydney, Australia, and the other at Dublin City University (DCU), Ireland. The trace details are given in Table 1. The media-unit size was chosen equal to the inter packet interval.

For each of these four traces, Ranganathan and Kilmartin [16] evaluated their fuzzy network with PESQ. To compare results, we have run the optimal control algorithm on the same four traces, and evaluated the resulting voice files with PESQ.

Ranganathan and Kilmartin [16] let the user or application choose a 'history size' to be used by the fuzzy network and a sensitivity parameter λ used to control the responsiveness of the system for decreasing network delays. Their results are reproduced in Figs. 5*a*–*c*. They consist of 3-D graphs with the 'history size' and λ along the horizontal axes and the results that we want to compare our algorithm to, on the vertical axis.

Since PESQ is sensitive to stretching and compression of the sound signal (and thus also to the changes made by WSOLA), we can think of PESQ as a user and application that requires the player rate to be close to the correct media speed. In this section, we have therefore used a relatively high value for Δx_1 and lower values for Δx_2 and Δx_3 (Section 3).

6.1 Results for trace 1

The results shown in Fig. 5*d* and *e* are obtained by running the optimal controller with $\Delta x_1 = 1$ media-unit, $\Delta x_2 = 5$ media-units/s and $\Delta x_3 = 0.1$ media-units/s^2.

Figs. 5*a* and *b* use the same range of λ and history size. Thus, for each combination of λ and 'history size', the PESQ score shown in Fig. 5*a* and the additional buffering delay (i.e. the delay introduced by receiver buffering) shown in Fig. 5*b* belongs to the same run. Each such combination of PESQ score and buffering delay can be compared to Fig. 5*d*, which shows the results from the optimal control algorithm with the mean playout buffer level along the *x*-axis and the PESQ score along the *y*-axis. This comparison shows that the optimal control algorithm has a higher PESQ score for most receiver buffer levels. At the highest buffer level (at $\lambda = 10$ and 'history size' = 100 in Figs. 5*a* and *b*), the PESQ score is close to equal, and at the lowest buffer level (at $\lambda = 50$ and 'history size' = 20 in Figs. 5*a* and *b*) the optimal control algorithm has one point higher PESQ score than [16].

Fig. 5*e* shows the additional buffering delay on the *x*-axis and the run-dry rate on the *y*-axis. This can be compared to Fig. 5*b*, where the additional buffering delay is on the vertical axis, and Fig. 5*c*, where the late packet loss rate is on the vertical axis, in the same way as explained above.

Fig. 5*e* shows that the maximum run-dry rate of the optimal control algorithm is less than 0.002. Figs. 5*b* and *c* show a late packet loss rate with a minimum value of 0.005, which is more than double the run-dry rate of the optimal control algorithm.

The optimal control algorithm has a higher PESQ score (close to one point better for most buffer levels) and a much lower run-dry rate than the corresponding numbers from [16], and can thus be said to be a considerably better playout algorithm for trace 1.

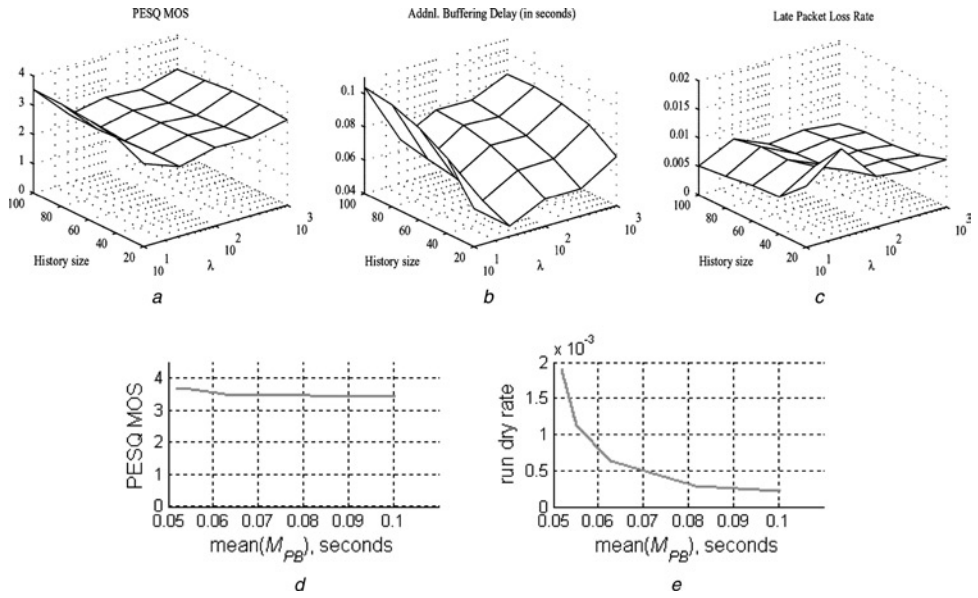

Fig. 5 *Results from the optimal controller and from [16] for the NUIG-DCU trace with 20 ms packetisation interval*

a Results from [16] with the PESQ score on the vertical axis
b Results from [16] with the additional buffering delay on the vertical axis
c Results from [16] with the late packet loss rate on the vertical axis
d Results from the optimal control algorithm with the mean playout buffer level along the *x*-axis and the PESQ score along the *y*-axis
e Results from the optimal control algorithm with the mean playout buffer level on the *x*-axis and the run-dry rate on the *y*-axis

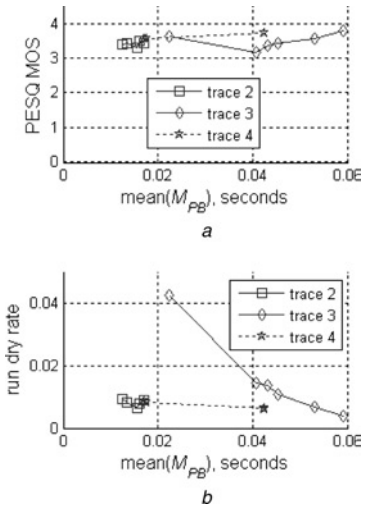

Fig. 6 *PESQ and run-dry rate for traces 2, 3 and 4*

a PESQ
b Run-dry rate

6.2 Results for traces 2, 3 and 4

The results shown in Fig. 6 are obtained by running the optimal controller with $\Delta x_1 = 1$ media-unit, $\Delta x_2 = 5$ media-units/s and $\Delta x_3 = 0.1$ media-units/s^2 for trace 2 and 3 (equal to the weight factors used for trace 1), and $\Delta x_1 = 10$ media-units, $\Delta x_2 = 1$ media-unit/s and $\Delta x_3 = 0.1$ media-units/s^2 for trace 4.

For trace 2, the optimal control algorithm has a PESQ score that is equal to or higher than the PESQ score from [16], while the run-dry rate of the optimal control algorithm is lower than the packet loss rate from [16]. Thus, the optimal control algorithm is a better playout algorithm for trace 2.

For trace 3, the optimal control algorithm has a PESQ score that is on average one point higher than the PESQ score from [16] for equal buffer levels, and a run-dry-rate that is slightly below the late packet loss rate of [16] for equal buffer levels. Thus, the optimal control algorithm is a considerably better algorithm for trace 3.

For trace 4, the PESQ score is on the same level and the run-dry rate is comparable to the late packet loss rate from [16] and will thus (as explained in Section 4.3) have a lower impact than the late packet loss rate. Thus, the optimal control algorithm is slightly better than [16] for trace 4.

7 Summary and conclusion

The optimal controller is based on a stringent notation and stringent mathematical models of the media receiver system. The notation and mathematical models are network and protocol independent, and can also be used as a basis for developing any kind of playout buffer algorithms.

Our approach is statistically optimal by minimising three deviations from the perfect playout, based on their relative importance: (i) buffering delay (ii) a playout rate different from the sender rate and (iii) a change of playout rate. The importance will vary for different user and application requirements, and are thus freely tunable by means of weight factors.

The optimal control algorithm has demonstrated very good results when compared to other algorithms in an objective technique for measuring voice quality (PESQ) and in a subjective listening test (DMOS), for both simulated and real network measurement traces. A comparison with an advanced fuzzy network algorithm [16] on real network data showed that the optimal control algorithm gave clearly better results.

8 Open problems

It is shown in [25, 30] that the optimal control algorithm works very well even when it uses a wrong transport segment model that is it is very robust. Section 6 demonstrated very good results for the optimal controller with a general model of the transport segment. However, an improved transport segment model could give even better results.

An automatic real-time identification or detection of the transport segment state space model (to combine it with the optimal control algorithm) could lead to even better results than shown in this paper. This identification procedure could have parts similar to a subset of Matlab's System Identification Toolbox (www.mathworks.com).

The presented mathematics is independent of the medium, but this paper has validated only the audio case. The effects on video need to be investigated and validated.

A new quality metric for variable playout speed is needed, that combines the effects of buffering delay and listening-only or viewing-only media quality. Today, such metrics [35, 36] and performance bounds [37] exist only for constant playout speed.

9 Acknowledgments

I would like to thank M.K. Ranganathan at Sasken Communication Technologies Limited, Bangalore, India and L. Kilmartin at NUIG, Ireland, for letting me use their measurement traces, used in [16], and for giving me permission to reproduce their results.

I would also like to thank the anonymous reviewers for a number of critical and fruitful remarks that led to substantial improvements, and the editor for letting me revise the manuscript twice.

10 References

1 Ramjee, R., Kurose, J., Towsley, D., and Schulzrinne, H.: 'Adaptive playout mechanisms for packetized audio applications in wide-area networks'. 13th IEEE Proc. INFOCOM '94, networking for global communications, Toronto, Canada, June 1994, vol. 2, pp. 680–688

2 DeLeon, P., and Sreenan, C.J.: 'An adaptive predictor for media playout buffering'. Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP), Phonenix, Arizona, March 1999, vol. 6, pp. 3097–3100

3 Pinto, J., and Christensen, K.J.: 'An algorithm for playout of packet voice based on adaptive adjustment of talkspurt silence periods'. Proc. IEEE Conf. Local Computer Networks, Lowell, Massachusetts, October 1999, pp. 224–231

4 Atzori, L., and Lobina, M.L.: 'Speech playout buffering based on a simplified version of the ITU-T E-model', *IEEE Signal Process. Lett.*, 2004, **11**, (3), pp. 382–385

5 Atzori, L., Lobina, M.L., and Isola, M.: 'Playout buffering in IP telephony: a quality maximization approach'. 1st Int. Conf. Multimedia Services Access Networks, Orlando, Florida, June 2005, pp. 49–53

6 Jung, Y., and Atwood, J.W.: 'β-adaptive playout scheme for voice over IP applications', *IEICE Trans. Commun*, 2005, **E88-B**, (5), pp. 2189–2192

7 Jung, Y., and Atwood, J.W.: 'Dynamic adaptive playout algorithm using interarrival jitter and dual use of $^{\alpha}$', *IEE Proc., Commun.*, 2006, **153**, (2), pp. 279–287

8 Narbutt, M., and Murphy, L.: 'Adaptive playout buffering for audio/video transmission over the internet'. Proc. IEE 17th UK Teletraffic Symp., Dublin, Ireland, May 2001, pp. 27/1–27/6

9 Narbutt, M., and Murphy, L.: 'VoIP playout buffer adjustment using adaptive estimation of network delays'. Proc. 18th Int. Teletraffic Congress – ITC-18, Berlin, Germany, September 2003, pp. 1171–1180
10 Narbutt, M., and Murphy, L.: 'A new VoIP adaptive playout algorithm'. IEE Telecommunications Quality of Services: The Business of Success (QoS 2004), London, March 2004, pp. 99–103
11 Narbutt, M., and Murphy, L.: 'Improving voice over IP subjective call quality', *IEEE Commun. Lett.*, 2004, **8**, (5), pp. 308–310
12 Narbutt, M., and Davis, M.: 'An assessment of the audio codec performance in voice over WLAN (VoWLAN) systems'. Proc. 2nd Annual Int. Conf. Mobile and Ubiquitous Systems: Networking and Services (MobiQuitous'05), San Diego, California, July 2005, pp. 461–470
13 Liu, F., Kim, J., and Kuo, C.-C.J.: 'Adaptive delay concealment for internet voice applications with packet-based time-scale modification'. Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing, Salt Lake City, Utah, May 2001, pp. 1461–1464
14 Laoutaris, N., and Stavrakakis, I.: 'An analytical design of optimal playout schedulers for packet video receivers', *Comput. Commun.*, 2003, **26**, (4), pp. 294–303, available at: www.elsevier.com\locate\comcom
15 Laoutaris, N., Van Houdt, B., and Stavrakakis, I.: 'Optimization of a packet video receiver under different levels of delay jitter: an analytical approach', *Perform. Eval.*, 2004, **55**, (3–4), pp. 251–275
16 Ranganathan, M.K., and Kilmartin, L.: 'Neural and fuzzy computation techniques for playout delay adaptation in VoIP networks', *IEEE Trans. Neural Netw.*, 2005, **16**, (5), pp. 1174–1194
17 Liang, Y.L., Färber, N., and Girod, B.: 'Adaptive playout scheduling and loss concealment for voice communication over IP networks', *IEEE Trans. Multimed.*, 2003, **5**, (4), pp. 532–543
18 Liang, Y.L., Färber, N., and Girod, B.: 'Adaptive playout scheduling using time-scale modification in packet voice communications'. Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP), Salt Lake City, Utah, May 2001, vol. 3, pp. 1445–1448
19 ITU-T Recommendation G.711 Appendix I, 'A high quality low complexity algorithm for packet loss concealment with G.711' (ITU-T, 1999)
20 ANSI Recommendation T1.521a-2000 (Annex B), 'Packet loss concealment for use with ITU-T recommendation G.711' (ANSI, 2000)
21 Gündüzhan, E., and Momtahan, K.: 'A linear prediction based packet loss concealment algorithm for PCM coded speech', *IEEE Trans. Speech Audio Process.*, 2001, **9**, (8), pp. 778–785
22 Rodbro, C.A., Murthi, M.N., Andersen, S.V., and Jensen, S.H.: 'Hidden Markov model-based packet loss concealment for voice over IP', *IEEE Trans. Audio Speech Lang. Process.*, 2006, **14**, (5), pp. 1609–1623
23 Rodbro, C.A., Christensen, M.G., Andersen, S.V., and Jensen, S.H.: 'Compressed domain packet loss concealment of sinusoidally coded speech'. Proc. 2003 IEEE Int. Conf. Acoustics, Speech, and Signal Processing, 6–10 April 2003, vol. 1, pp. 104–107
24 Sanneck, H., Stenger, A., Younes, K., and Girod, B.: 'A new technique for audio packet loss concealment'. IEEE Proc. Global Internet, November 1996, pp. 48–52
25 Gade, B.H.H.: 'A statistically optimal algorithm for multimedia buffers', PhD thesis, University of Oslo, 2007
26 Strang, G.: 'Linear algebra and its applications' (Brooks/Cole Thomson learning, 1988, 3rd edn.)
27 Chen, C.-T.: 'Linear system theory and design' (Oxford University Press, 1999, 3rd edn.)
28 Hinrichsen, D., and Pritchard, A.J.: 'Mathematical systems theory I, modelling, state space analysis, stability and robustness' (Springer, 2005)
29 Balchen, J.G., and Mummé, K.I.: 'Process control – structures and applications' (Van Nostrand Reinhold, New York, 1988), pp. 60–66
30 Hafskjold, B.: 'Optimal control of playoutbuffers'. Proc. Int. Conf. Computer, Communication and Control Technologies (CCCT '03), Orlando, Florida, USA, July/August 2003, vol. VI, pp. 175–181
31 Hafskjold, B.: 'Anti-run-dry algorithm for optimal control of playoutbuffers'. Proc. Int. Symp. on Information and Communication Technologies (ISICT03), Dublin, Ireland, 24–26 September 2003, pp. 410–417
32 Gelb, A.: 'Applied optimal estimation' (The MIT Press, Cambridge, Massachusetts and London, England, 1974, 16th printing, 2001)
33 ITU-T Recommendation P.800: 'Methods for subjective determination of transmission quality', in series P: Telephone transmission quality, Methods for objective and subjective assessment of quality (ITU-T, 1996)
34 ITU-T Recommendation P.862: 'Perceptual evaluation of speech quality (PESQ): an objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs', in series P: Telephone transmission quality, telephone installations, local line networks, Methods for objective and subjective assessment of quality (ITU-T, 2001)
35 Narbutt, M., Kelly, A., Murphy, L., and Perry, P.: 'Adaptive VoIP playout scheduling: assessing user satisfaction', *IEEE Internet Comput.*, 2005, **9**, (4), pp. 28–34
36 Cole, R.G., and Rosenbluth, J.H.: 'Voice over IP performance monitoring', *ACM SIGCOMM Comput. Commun. Rev.*, 2001, **31**, (2), pp. 9–24
37 Moon, S., Kurose, J., and Towsley, D.: 'Packet audio playout delay adjustment: performance bounds and algorithms', *Multimedia Syst.*, 1998, **6**, (1), pp. 17–28
38 Nordavinden og sola, Norwegian dialect samples A database of identical text read by different people. Available at: http://www.ling.hf.ntnu.no/nos, accessed January 2006
39 Liu, F., Kim, J., and Kuo, C.-C.J.: 'Quality enhancement of packet audio with time-scale modification'. Proc. SPIE: ITCOM 2002: Multimedia Systems and Applications V, Boston, Massachusetts, July 2002, vol. 4861, pp. 163–173
40 Arentz, W.A., Hetland, M.L., and Olstad, B.: 'Retrieving musical information based on rhythm and pitch correlations', in Arentz, W.A. (Eds.), 'Searching and classifying non-textual information'. PhD thesis, Norwegian University of Science and Technology, 2004

11 Appendix

Table 2 gives an overview of the notation rules used in the paper, and Table 3 gives an overview of the specific symbols used.

Table 2: Notation rules

Symbol	Description	Example from paper
lowercase letter	scalar variable	r_{SNDR}
right subscript	specification of the value	
bold lowercase letter	Vector	$\boldsymbol{x}$ (state vector)
bold uppercase letter	matrix	$\boldsymbol{A}$ (system matrix)
dot above a variable	the time derivative of the variable	$\dot{\boldsymbol{x}} = \mathrm{d}/\mathrm{d}t(\boldsymbol{x})$
vertical lines on each side of a variable	$\lvert x\rvert = \sqrt{x^2}$, that is the absolute value of the variable x	$\lvert \dot{r}_{\mathrm{PLR}}(t)\rvert$
right superscript T	matrix transpose, $[\,a\ b\,]^{\mathrm{T}} = \begin{bmatrix} a \\ b \end{bmatrix}$	$\boldsymbol{x}_{\mathrm{TRS}} = [r_{\mathrm{TRS}} - r_{\mathrm{SNDR}} \ldots]^{\mathrm{T}}$
$\boldsymbol{0}_{a\times b}$	A zero matrix with a rows and b columns, $\boldsymbol{0}_{1\times 2} = [0 \quad 0]$	$\boldsymbol{0}_{n_{\mathrm{TRS}}\times 1}$

Table 3: Specific symbols used

Symbol	Description
r_{SNDR}	constant media-unit rate out of the sender, equal to the correct media speed
$r_{TRS}(t)$	media-unit rate out of the transport segment
$r_{VB}(t)$	media-unit rate out of the virtual buffer
$r_{PB}(t)$	media-unit rate out of the playout buffer
$r_{PLR}(t)$	media-unit rate out of the player
$M_{VB}(t)$	number of media-units in the virtual buffer at time t
$M_{PB}(t)$	number of media-units in the playout buffer at time t
$M_{PLR}(t)$	number of media-units in the player at time t
$M_{RCV}(t)$	number of media-units in the receiver buffers at time t, $M_{RCV}(t) = M_{VB}(t) + M_{PB}(t) + M_{PLR}(t)$
$M_{RCV,d}(t)$	desired number of media-units in the receiver buffers
$\boldsymbol{x}$	state vector for the total state space model
$\boldsymbol{x}_{TRS}$	state vector for the transport segment state space model
n_{TRS}	number of states in $\boldsymbol{x}_{TRS}$
$\boldsymbol{A}$	system matrix for the total state space model
$\boldsymbol{A}_{1,RCV}$	a sub-matrix of $\boldsymbol{A}$
$\boldsymbol{A}_{2,RCV}$	a sub-matrix of $\boldsymbol{A}$
$\boldsymbol{A}_{TRS}$	system matrix for the transport segment state space model
$\boldsymbol{B}$	control matrix for the total state space model
u	control variable for the total state space model
$\boldsymbol{C}$	process noise matrix for the total state space model
$\boldsymbol{C}_{TRS}$	process noise matrix for the transport segment state space model
$\boldsymbol{v}_{TRS}$	noise vector for the transport segment state space model
w_1	weight factor 1: the importance of minimising additional latency
w_2	weight factor 2: the importance of minimising difference between playout speed and correct media speed
w_3	weight factor 3: the importance of minimising the time derivative of the playout speed
$\boldsymbol{G}$	optimal control gain matrix
r_{12}, r_{22}, r_{22B}	variables used to simplify the presentation of $\boldsymbol{G}$