Fuzzy Control of Adaptive Timeout for Video Streaming over a Bluetooth Interconnect

R. Razavi, M. Fleury, and M. Ghanbari
University of Essex, ESE Dept.
Wivenhoe Park
Colchester, CO4 3SQ, UK

Abstract- **Bluetooth's default ARQ scheme is not well suited to real-time video streaming, resulting in missed display and decode deadlines. Therefore, adaptive ARQ combined with a deadline aware buffer is an alternative approach. However, even with the addition of adaptation to picture type content importance, ARQ is not ideal in conditions of deteriorating RF channel. The paper presents fuzzy logic control of ARQ, based on send buffer fullness and the head of line packet's deadline. Tests show an improvement of over 4 dB improvement in video quality compared to a non-fuzzy scheme. The scheme can be applied to any codec with I-, P- and (possibly) B-slices by inspection of packet headers without encoder intervention.**

I. INTRODUCTION

The Enhanced Data Rate (EDR) of IEEE 802.15.1, Bluetooth [1] version 2.0 [2] now has a peak user payload of 2.2 Mb/s, which is the same rate offered by current implementations of interactive IPTV. Therefore, a bottleneck free way exists of streaming encoded video from a server across an IP network to a Bluetooth master node, and, thence, over a Bluetooth interconnect. The default Bluetooth setting of the Automatic Repeat Request (ARQ) facility (an infinite re-transmission limit) leads to poor quality video at the receiving device, as streaming is delay-sensitive. Work in [3] appears to have first introduced priority-based re-transmission for video frames, though that paper went no further than a *static* scheme favouring Intra-coded pictures (I-pictures) at the link layer, based on similar application layer techniques such as those reported in [4]. The need to adjust *adaptively* the default re-transmission timeout (RTO) for multimedia applications was established in [5].

Previous research by us [6] introduced an adaptive ARQ scheme according to channel conditions, as reported by buffer fullness, with priority given to according to picture type and not just to I-pictures. However, the RTO values, though resulting in better received video quality than the default option, were stepped in nature rather than smoothly varying. This led to an investigation of a more flexible approach. This paper introduces fuzzy logic control (FLC) of adaptive ARQ. It also employs a deadline aware buffer, whereby deadline expired packets are pruned from the send buffer. The aim of the current work is to retransmit a packet as many times as needed to ensure error-free reception but without delaying that packet beyond its deadline and without leading to send buffer overflow. The ARQ mode but not the timeout in IEEE 802.11e is adapted in [7]. In [8], a general scheme, which is also deadline aware, but with bitrate control at the encoder, uses selective-repeat ARQ. A similar burst-error model to the one in Section II.D was employed. In [9] the error propagation impact of packets is found at the encoder and a retry limit with active discard is set for IEEE 802.11 DCF retries.

In general, a fuzzy scheme is more easily tuned by adjustment of the membership functions. By introducing two control inputs, a fuzzy scheme can trim its response. The two inputs in our scheme were buffer fullness *and* the deadline margin of the packet at the head of the Bluetooth send queue. A fuzzy scheme is well-suited to implementation on a mobile device, because not only are the decision calculations inherently simple (and can be made more so by adoption of triangular membership functions) but also by forming a Look-up-Table (LUT) from the fuzzy control surface, its operation can be reduced to simple LUT access. Because of Bluetooth's resemblance to Asynchronous Transfer Mode admission control, fuzzy logic bitrate control has been applied to Bluetooth wireless links [10]. However, this application of fuzzy control was for a quite different purpose, flow control, to the work herein.

Real-time delivery of video is delay-sensitive as a frame cannot be displayed if its data arrive after their decode deadline. A further deadline exists for some picture types if their presence contributes to decoding of future frames. In practice, a playout buffer exists on a mobile device to account for start-up delay and also absorbs delay jitter (variation of delay). Therefore, the maximum delay permissible corresponds to the start-up delay deemed tolerable to the user. ARQ adds to delay and, therefore, the number of retransmissions should be minimized. Equally, simply dropping all packets corrupted by RF noise or interference is not an option, as the display quality will be inferior according to channel conditions [6], though for encoded video the actual impact is dependent on the type of picture dropped and scene content characteristics. Therefore, some form of ARQ is advisable in order to account for the wireless channel condition and FLC ARQ seeks to optimally adapt the ARQ RTO to track channel conditions and send buffer congestion.

II. METHODOLOGY

A. Bluetooth ARQ

For Bluetooth, an ARQ may occur in the following circumstances [11]: a) failure to synchronize on the access header code; b) header corruption detected by a triple redundancy code; c) payload corruption detected by CRC; d)

failure to synchronize with the return packet header; e) header corruption of the return packet. Notice that a faulty ARQ packet can itself cause retransmission. As mentioned in Section I, the default value of the ARQ RTO in most Bluetooth chipsets [5] is set to infinity. In [5], a fixed RTO and an adaptive RTO were considered. The disadvantage of a fixed RTO is that it is difficult to arrive at a value that avoids either excessive delay or excessive packet drops in *all* circumstances. The adaptive RTO, which was upper and lower- bounded, was based in [5] on a smoothed round-trip time. The RTO was adapted downwards or upwards if the new smoothed round trip time respectively is less than or more than the previous smoothed round trip time.

B. Fuzzy logic control of ARQ

This Section briefly introduces FLC before introducing FLC ARQ. In a fuzzy subset, each member is an ordered pair, with the first element of the pair being a member of a set S and the second element being the possibility, in the interval [0, 1], that the member is in the fuzzy subset. This should be compared with a Boolean subset in which every member of a set S is a member of the subset with probability taken from the set {0, 1}, in which a probability of 1 represents certain membership and 0 represents non-membership. In a fuzzy subset of (say) 'buffer fullness', the possibility that a buffer with a given fullness taken from the set S of fullness may be called high is modeled by a membership function, which is the mapping between a data value and possible membership of the subset. Notice that a member of one fuzzy subset can be a member of another fuzzy subset with the same or a different possibility. Membership functions may be combined in fuzzy `if then rules` to make inferences such as `if x is high and y is low then z is normal`, in which `high`, `low`, and `normal` are membership functions of the matching fuzzy subsets and `x`, `y`, `z` are linguistic variables (names for known data values). In practice, the membership functions are applied to the data values to find the possibility of membership of a fuzzy subset and the possibilities are subsequently combined through defuzzification, which results in a crisp (non-fuzzy) value.

For the adaptive ARQ FLC, there are two inputs: buffer fullness and the time remaining to the playout buffer deadline for the packet about to be transmitted. Bluetooth buffer fullness is a preferable measure (compared to delay or packet loss) of channel conditions and of buffer congestion, as was established in [12]. Buffer fullness is available to an application via the Host Controller Interface (HCI) presented by a Bluetooth hardware module to the upper layer software protocol stack. Retransmissions avoid the effect of noise and interference but also cause the master's send buffer queue to grow, with the possibility of packet loss from send buffer overflow. (For simplicity, possible buffer overflow at the receiver is neglected, though in practice a matching receive buffer preceding the playout buffer [13] would avoid this possibility.) In the simulations of Section III, the size of the send buffer was set to 50 packets. In all experiments, the buffer queue discipline was set as FIFO.

The RTO of the packet at the head of the Bluetooth send queue will affect the deadlines of packets still to be transmitted. Therefore, the second FLC input moderates the buffer fullness input. The assigned membership functions, which were arrived at heuristically, are shown In Fig. 1 a) and b), and once found were fixed. The buffer fullness range in Fig. 1 a) is [0,1] corresponding to a percentage fullness. In Fig. 1 b), the horizontal axis represents the delay time of the packet at the head of the queue divided by the display deadline. In Fig. 1 b), unit delay/deadline corresponds to expiration of playout deadline. It is important to note that any packet in the send buffer is discarded if its deadline has expired (Section II.C). However, this takes place after the fuzzy evaluation of the desired ARQ RTO. In practice, the inputs to the FLC were sampled versions of buffer fullness and packet delay/deadline to avoid excessive ARQ RTO oscillations over time. The sampling interval was every 50 packets. Table 1 shows the `if then rules` that allow `input fuzzy` subsets to be combined to form an output. Notice more than one rule may apply because of the fuzzy nature of subset membership.

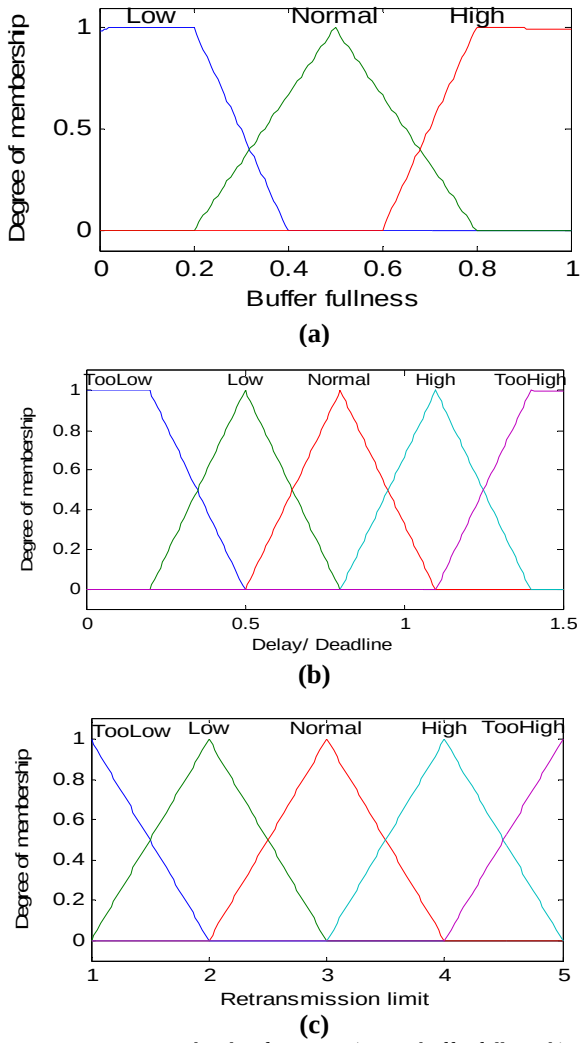

Figure 1. Fuzzy membership functions a) Input buffer fullness b) Input delay/deadline c) Output retransmission limit

The inputs were combined according to the well-known Mamdani model [14] to produce a single output value. The standard center of gravity method was employed to resolve to a crisp output value according to the output membership functions shown in Fig. 1 c). In Fig. 1 c) the retransmission limits correspond to the RTO of the current packet. Clearly a packet can only be retransmitted an integer number of times but the crisp output may result in a real-valued number. This difficulty was resolved by generating a random number from a uniform distribution. If the random number was less than fractional part of the crisp output value then that value was rounded up to the nearest integer, otherwise it was rounded down. The output value was subsequently scaled according to the priority of the packet's picture type.

Table 1 FLC `If..then` rules used to identify output fuzzy subsets from inputs.

	Delay/Deadline					
Buffer Fullness		*TooLow*	*Low*	*Normal*	*High*	*TooHigh*
	High	Normal	Normal	Low	TooLow	TooLow
	Normal	TooHigh	High	Normal	Low	TooLow
	Low	TooHigh	High	Normal	Low	TooLow

For reasons of error resilience, encoded video is transmitted as a repeating sequence of Group of Pictures (GOP) [15], with the start of each GOP formed by an I-picture. An I-picture is the basis for prediction of all other pictures in the GOP (usually 12 to 15 pictures in all) and, hence, its loss has drastic consequences for all other pictures. P-pictures also form the basis for predictions but are not essential for the reconstruction of other pictures within the GOP (as other I- or P- anchor pictures retained in a decode buffer can be applied). Lastly, the third type of picture, the bi-predictive B-picture, has no predictive value.

A simple scaling of 3:2:1 was applied respectively for I-, P-, B-pictures. For example, if the crisp output value was 2.3 and the random number based resolution resulted in 2 retransmissions, an I-picture packet would be retransmitted 6 times.

The fuzzy control surface is represented in Fig.2, as derived from the Matlab Fuzzy Toolbox v. 2.2.4. As mentioned in Section I, by means of an LUT derived from the surface, a simple implementation becomes possible.

C. Deadline aware buffer

In the conservative send buffer discard policy of this paper, all packets of whatever picture type have a display deadline which is the size of the playout buffer expressed as a time beyond which buffer underflow will occur. In a conservative policy, in which there is no need for playout buffer fullness updates, the deadline is set as the maximum time that the playout buffer can delay the need for a packet. Playout buffers are normally present to smooth out jitter across a network path (if the Bluetooth master was also an access point) and in this paper the size is assumed to be constant. In the simulations of Section III, the display deadline was set to 0.15 s.

In addition to the display deadline, all I-picture packets have a decode deadline, which is the display time remaining to the end of the GOP. Thus, for a 12 frame GOP, this is the time to display 11 frames, i.e. 0.44 s at 25 frame/s. For P-picture packets, the time will vary depending on the number of frames to the end of the GOP. For B-pictures the decode deadline is set to zero.

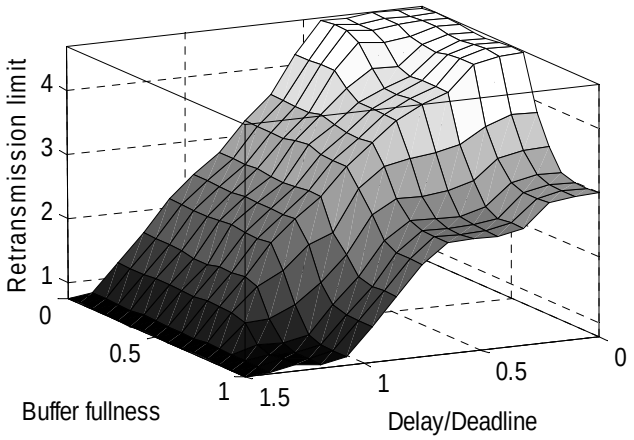

Figure 2. Control surface resulting for FLC ARQ.

The decode deadline is added to the display deadline and a packet is discarded from the send buffer after its total deadline expires. By storing the GOP end time, an implementation performs one subtraction to find each decode deadline. Account has been taken of I- B- P-picture reordering at encode and send buffer output, which has an effect on buffer fullness. Reordering is introduced to ensure that reference pictures arrive and can be decoded before the dependent B-pictures. In the discard policy, propagation delay is assumed (optimistically) to be constant.

D. Simulation setup

A Gilbert-Elliott [16] two state discrete-time, ergodic Markov chain modeled the wireless channel error characteristics between a Bluetooth master and slave node. By adopting this model it was possible to simulate burst errors of the kind that cause problems to an ARQ mechanism. The mean duration of a good state, T_g, was set at 2 s and in a bad state, T_b, was set to 0.25 s. In units of 625 μs (the Bluetooth time slot duration), T_g= 3200 and T_b= 400, which implies from:

$$T_g = \frac{1}{1-Pgg}, T_b = \frac{1}{1-Pbb} \quad (1)$$

that, given the current state is good (g), Pgg, the probability that the next state is also g, is 0.9996875 and Pbb, given the current state is bad (b), the probability that the next state is also b, is 0.9975. At 3.0 Mb/s, the Bit Error Rate (BER) during a good state was set to $5x10^{-5}$ and during a bad state to $5x10^{-4}$. The transition probabilities, Pgg and Pbb, as well as the BER, are approximately similar to those in [17], but the

mean state durations are adapted to Bluetooth. The two states result in CNRs of respectively 15.13 and 13.60 dB.

This research employed the University of Cincinatti Bluetooth (UCBT) extension (download is available from `http://www.ececs.uc.edu/~cdmc/UCBT`) to the well-known ns-2 network simulator (v. 2.28 used). The UCBT extension supports Bluetooth EDR but is also built on the air models of previous Bluetooth extensions such as BlueHoc from IBM and Blueware. All links were set at the maximum EDR 3.0 Mbps gross air rate. Simulation runs were each repeated ten times and the results averaged to produce summary statistics.

The simulations were carried out with input from an MPEG-2 encoded bitstream at a mean rate of 1.5 Mbit/s for a 30 s video clip with moderate motion. Peak Signal-to-Noise Ratio (PSNR) was found by reconstructing with a reference MPEG-2 decoder. The display rate was 25 frame/s resulting in all in 750 frames in each run. The source video was CIF-sized (366×288 pixels) with a GOP structure of N = 12, and M = 3 (M is the number of pictures from the I-picture to the first P-picture, i.e. including two B-pictures). In [18], fully filled Bluetooth packets were formed using maximal bandwidth five time slot packets, regardless of slice boundaries. While this results in some loss in error resilience, as each MPEG-2 slice contains a decoder synchronization marker, in [18] it is shown that the overall video performance is superior.

III. Simulation results

As a point of comparison, the default Bluetooth scheme with infinite ARQ was tested with the input video from Section II.D. This resulted in send buffer fullness recorded in Fig. 3. Buffer fullness is frequently above the 50 packet capacity, leading to buffer overflow. The other input source to the FLC (if it were applied) is recorded in Fig. 4. Packet delay quickly exceeds the 0.15 s playout buffer deadline.

As an alternative [6], the ARQ RTO can be adaptively selected in terms of number of retransmissions allowed, to avoid further delay after the packet enters the tail of the send buffer. A threshold is set that is the maximum number of retransmissions allowed when the buffer is empty. The maximum number of retransmissions is subsequently changed by a factor depending on the buffer fullness reported by the Bluetooth HCI. The formula employed is summarized as

$$N = round\left(\frac{m.(c-f)}{c}\right), \qquad (2)$$

where N is the maximum number of retransmissions allowed -- the RTO, m is the maximum number of retransmissions allowed when the buffer is empty, f is the number of packets buffered in the send buffer (buffer fullness), and c is the buffer capacity. The operator *round* returns the nearest integer. According to (2), the maximum number of retransmissions allowed is a function of buffer fullness. When the buffer is empty, $f = 0$, then the maximum number of retransmissions occurs, whereas when the buffer approaches full occupation then no retransmissions may occur. The smaller the value of m the sooner this latter event occurs.

Fig. 5 shows the buffer fullness resulting from application of this scheme for the channel conditions of Section II.D. Fig. 6 shows the corresponding packet delay over time, as the video clip is transmitted over the Bluetooth interconnect. While this scheme certainly improves upon the Bluetooth scheme under these channel conditions it is also apparent that delay continues to be high and buffer overflow will frequently occur.

Now consider the adaptive ARQ scheme of the previous paragraph with the addition of the same deadline-aware policy discussed in Section II.C. Deadline awareness removes packets from the buffer if their usefulness has passed and adjusts the RTO according to their picture-type-based deadline. Fig. 7 shows that this policy immediately leads to a reduction in buffer fullness, so that there is no buffer overflow. Packet delay is also reduced, Fig. 8 but there are still packets that miss their display deadline. This is because I- and P- picture packets are still transmitted if their presence at the receiver can be used to decode subsequent pictures. However, their absence from the display results in the deterioration in video quality shown through PSNR over time in Fig. 9.

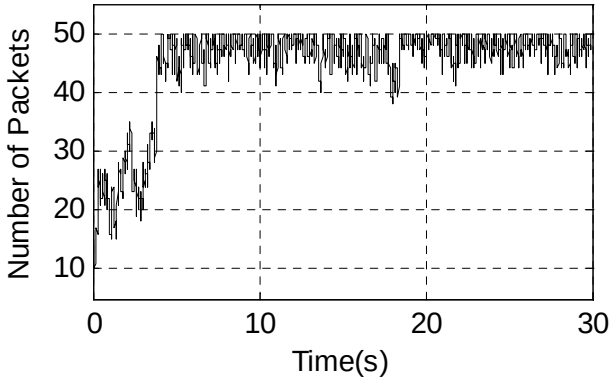

Figure3. Default Bluetooth infinite ARQ buffer fullness.

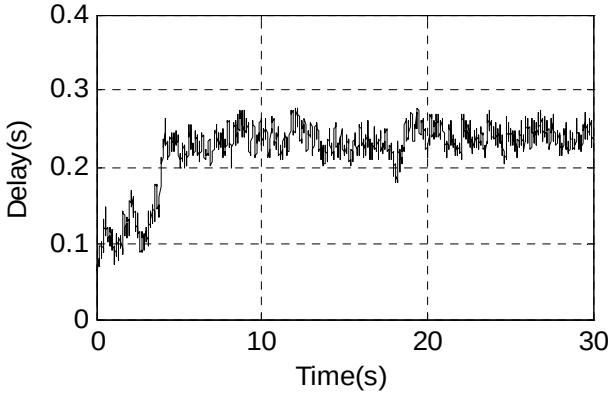

Figure 4. Default Bluetooth infinite ARQ packet delay.

Turning to FLC ARQ with deadline awareness, Fig. 10 represents the first input to the FLC (the actual sampled version of the buffer fullness plot is very similar).Compared to Fig. 7 buffer fullness on average is reduced. There is a similar

improvement in packet delay, as evidenced by Fig. 11, though this is not strongly indicated by the Figure. n Fig. 12, the video quality resulting from the FLC ARQ scheme is shown, with matching improvement in PSNR. For reference, Fig. 13 shows the output of the FLC before priority adaptation.

Table 2 provides summary statistics (mean of 10 runs) for the different schemes, confirming the behavior over time already reported. There is a progressive improvement across the schemes. (The PSNR is not recorded for the default and simple adaptive scheme as it is clearly inferior.) Comparing the traditional adaptive ARQ scheme with deadline awareness to that of the FLC ARQ there is over 4 dB improvement from employing the FLC scheme.

IV. CONCLUSION

This paper compared various link layer schemes for control of ARQ timeouts and found that fuzzy logic control results in a quite considerable improvement in received video quality over a traditional scheme. Though adaptive control of ARQ at the link layer is known in the literature, mainly for other than Bluetooth, the identification of a near optimal scheme is not. Fuzzy logic control can readily be tuned but once the operating parameters are established no further modifications are required. Though the detailed experiments in this paper are specific to Bluetooth, there is no reason why the same approach should not be applied to other wireless technologies that employ ARQ as a form of error control. Equally, though the scheme was tested with the widely deployed MPEG-2 codec, I and P slices are present in the more recent H.264 and B slices occur in all but H.264's baseline profile.

REFERENCES

[1] J. Haartsen, "The Bluetooth Radio System", *IEEE Personal Comms.,*vol. 7, no. 1, pp. 28-26, 2000.

[2] Specification of the Bluetooth System -- 2.0 + EDR, 2004, Available online at http://www.bluetooth.com.

[3] R. Kapoor et al., "Link Layer Support for Streaming MPEG Video over Wireless Links", *Int. Conf. on Computer Comms. and Networks*, pp. 477-482, 2003.

[4] W. Tan and A. Zakhor, "Packet Classification Schemes for Streaming MPEG Video over Delay and Loss Differentiated Networks", *PacketVideo Workshop*, May, 2001.

[5] L.-J. Chen, R. Kapoor, S. Jung, M. Y. Sadadidi, and M. Gerla, "Audio Streaming over Bluetooth: An Adaptive ARQ Timeout Approach", *24th Conf. on Distributed Computing Systems Workshops*, vol. 7, pp. 196-201, March 2004.

[6] R. Razavi, M. Fleury, and M. Ghanbari, "Adaptive Timeout for Video Delivery over a Bluetooth Wireless Network", *Int. Conf. on Applied Computing*, pp. 245-254, Feb. 2007.

[7] J. Wall and J. Y. Khan, "An Adaptive ARQ Enhancement to Support Multimedia Traffic using 802.11 Wireless LANs", *IEEE GLOBECOM*, pp. 3037-3041, Nov. 2004.

[8] C.-H. Hsu, A. Ortega, and M. Khansari, "Rate Control for Robust Video Transmission over Burst-Error Wireless Channels", *IEEE J. on Selected Areas in Comms.*, vol. 17, no. 5, pp. 756-773, 1999.

[9] C.-M. Cheng, C.-W. Lin, and Y.-C. Chen, "Packet-by-Packet Scheduling for Video Streaming over Wireless with Content-Aware Retry Limit", IEEE Int. Workshop on Multimedia Signal Processing, Oct. 2006.

[10] H. B. Kazemian and L. Meng, "An adaptive control for video transmission over Bluetooth," *IEEE Trans. on Fuzzy Systems*, vol. 14, no. 2, pp. 263–274, April 2006.

[11] M. C. Valenti, M. Robert, and J. H. Reed, "On the Throughput of Bluetooth Data Transmissions", *IEEE Wireless Commun. and Networking Conf.,* pp. 119-123. 2002.

[12] R. Razavi, M. Fleury, and M. Ghanbari, "Detecting Congestion within a Bluetooth Piconet: Video Streaming Response*", London Comms. Symposium*, pp. 181-184, Sept. 2006.

[13] M. Yokotsuka, M., "Memory Motivates Cell-phone Growth", *Electronic Design*, vol. 9, no. 3, pp. 27-30, 2004.

[14] J. S. Jang, C. T. Sun, and E. Mizutani, *Neuro-fuzzy and Soft Computing*. Prentice Hall, Upper Saddle River, NJ, 1997.

[15] M. Ghanbari, *Standard Codecs: Image Compression to Advanced Video Coding*, IEE Press, Stevenage, UK, 2003.

[16] J.-P. Ebert and A. Willig, "A Gilbert-Elliot Bit-Error Model and the Efficient Use in Packet Level Simulation", Technical report TKN-99-2002, Tech. Univ. Berlin, 1999.

[17] R. Fantacci, R. and M. Scardi, "Performance Evaluation of Preemptive Polling Schemes and ARQ Techniques for Indoor Wireless Networks", IEEE Trans. on Vehicular Tech., vol. 45, no. 2, pp. 248-257, 2003.

[18] R. Razavi, M. Fleury, and M. Ghanbari, "An Efficient Packetization Scheme for Bluetooth Video Transmission", *Electronic Letters*, vol. 42, no. 20, pp. 1143-1145, 2006.

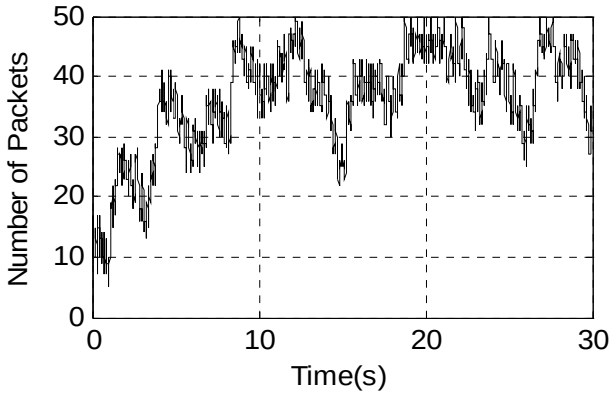

Figure 5. Adaptive ARQ scheme buffer fullness.

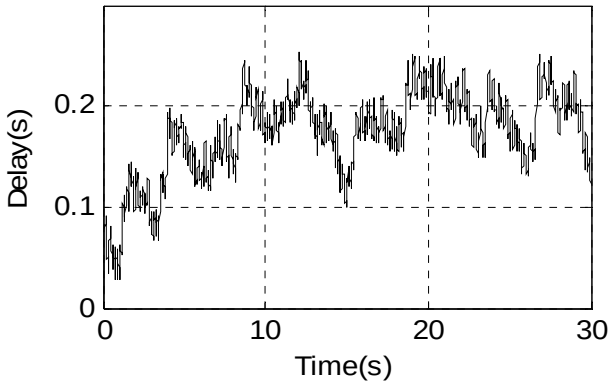

Figure 6. Adaptive ARQ scheme packet delay.

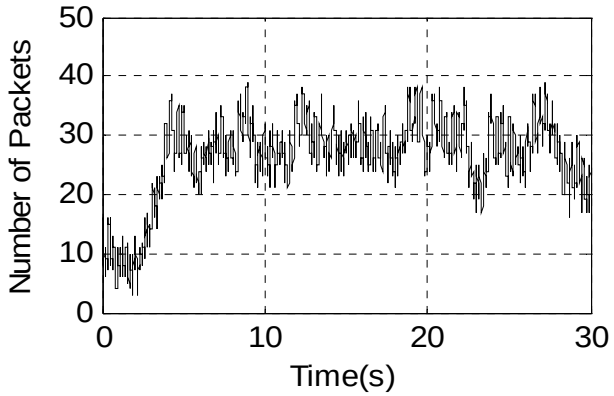

Figure 7. Adaptive ARQ scheme with deadline awareness buffer fullness.

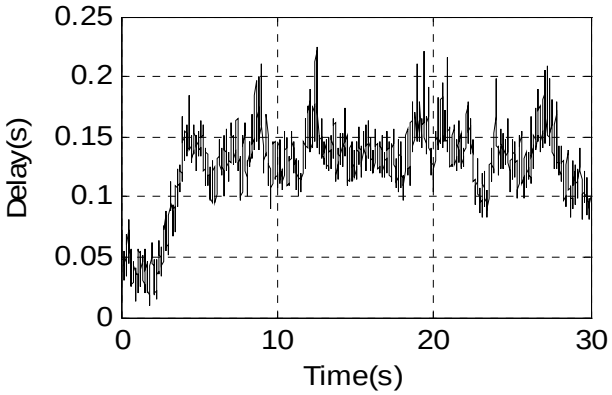

Figure 8. Adaptive ARQ scheme with deadline awareness packet delay

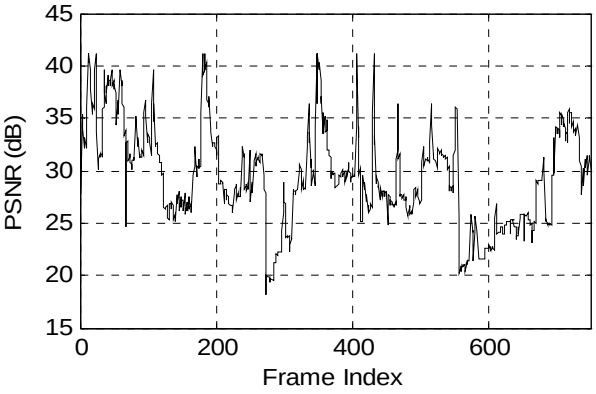

Figure 9. Adaptive ARQ scheme with deadline awareness video quality.

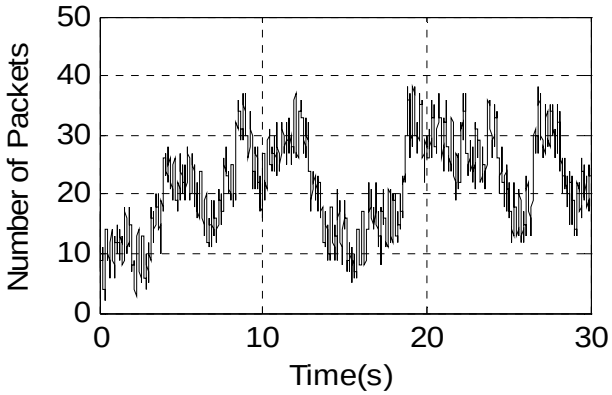

Figure 10. FLC ARQ scheme with deadline awareness: buffer fullness.

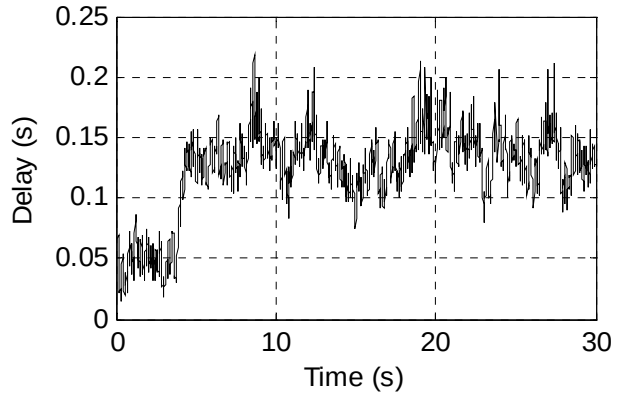

Figure 11. FLC ARQ scheme with deadline awareness packet delay.

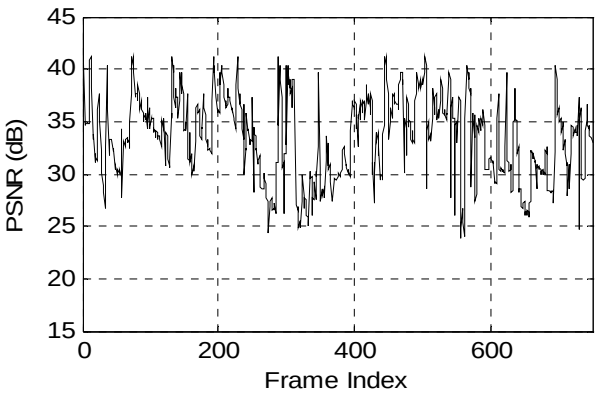

Figure 12. FLC ARQ scheme with deadline awareness: video quality.

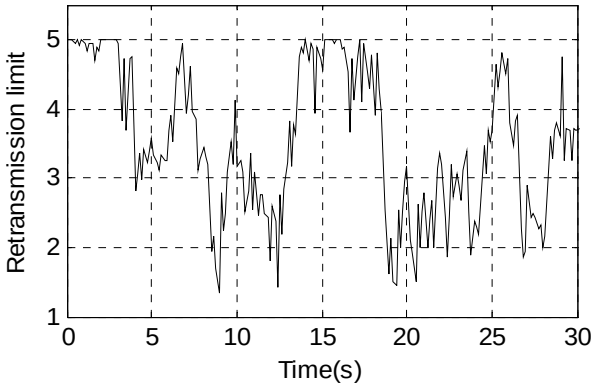

Figure 13. FLC ARQ output before picture type priority adaptation.

Table 2. Summary statistics for various ARQ schemes.

FLC ARQ with deadline awareness	
Mean buffer fullness (packets)	**21.4263**
Mean packet delay (s)	**0.0998**
Packet loss (%)	**12.45**
Mean PSNR (dB)	**33.4159**
Adaptive ARQ with deadline awareness	
Mean buffer fullness (packets)	26.3028
Mean packet delay (s)	0.1232
Packet loss (%)	17.87
Mean PSNR (dB)	29.4662
Adaptive ARQ	
Mean buffer fullness (packets)	36.1225
Mean packet delay (s)	0.1725
Default infinite ARQ	
Mean buffer fullness (packets)	44.3984
Mean packet delay (s)	0.2203