Chapter 3
Resource-Aware Fuzzy Logic Control of Video Streaming over IP and Wireless Networks

M. Fleury, E. Jammeh, R. Razavi, and M. Ghanbari

Abstract Congestion control of real-time streaming a video clip or film across the Internet is vital, as network traffic volatility requires constant adjustment of the bit rate in order to reduce packet loss. Traditional solutions to congestion control are prone to delivery rate fluctuations and may respond only when packet loss has already occurred, while both fluctuations and packet loss seriously affect the end user's appreciation of the delivered video. In this chapter, fuzzy logic control (FLC) is newly applied to control of video streaming in fixed and wireless networks. In a fixed network, by way of congestion control the encoded video bitstream's rate is adjusted according to the available bandwidth. Compared to existing controllers, FLC's sending rate is significantly smoother, allowing it to closely track available bandwidth at a bottleneck on the video stream's path across a network. The chapter also shows that when multiple video streams are congestion controlled through FLC, the result is a fairer and more efficient sharing of the bandwidth capacity. Also considered is a pioneering application of FLC to wireless networks, where other resources, apart from available bandwidth, come into play. An FLC system has been designed that provides a modular solution to control of latency and energy consumption, which is important for battery-powered devices, but must be balanced against the quality of delivered video. The chapter concludes by presenting the potential of emerging type-2 fuzzy logic as a way of significantly improving the robustness of classical type-1 fuzzy logic.

Keywords Congestion control · Fuzzy logic · Networks · Video streaming

3.1 Introduction

Real-time video streaming applications, such as IPTV (TV distributed over an IP network), video-on-demand, video-clip-Web-click, and network-based video recorder, have high bit rates that risk overwhelming traditional networks if it is not

M. Fleury (✉)
School of CSEE, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK
e-mail: fleum@essex.ac.uk

A.-E. Hassanien et al. (eds.), *Pervasive Computing: Innovations in Intelligent Multimedia and Applications*, Computer Communications and Networks,
DOI 10.1007/978-1-84882-599-4_3,

possible to control their flows. In video streaming, a compressed video bitstream is transmitted across a network to the end user's decoder (prior to display) without the need for storage other than in temporary buffering. Such video services are attractive to end users because of their variety and flexibility compared to broadcast TV over the airwaves and they interest telecommunications companies because of their high bit rates. These companies are also interested in seamlessly extending video services across wireless networks because these networks have the advantages of mobility and convenience. However, wireless networks reintroduce many impediments to high-quality video delivery such as adverse channel conditions due to noise and interference, while mobility brings a need to preserve a device's battery power.

We have applied fuzzy logic control (FLC) of unicast video streaming across packet-switched internet protocol (IP) networks and wireless networks. Prior to our research, analytical methods have been almost exclusively applied to congestion control of video on the Internet. For example, the industry standard approach models the video stream's rate with a complex equation designed to replicate the average flow of other competing flows across network bottlenecks. In that way it is hoped that further network congestion will be avoided. Unfortunately, this approach may lead to unnecessary packet loss because of the need to probe the network using the video stream itself. Inevitably as the video rate is increased, packet loss occurs once the available bandwidth or capacity limit is reached. Based on fuzzy logic, we have devised a more flexible method that estimates traffic conditions without the need for probing. Though fuzzy logic has been applied to access control for prior networking technologies such as the asynchronous transfer mode (ATM), to the best of the authors' knowledge it has not or rarely been applied to congestion control over IP networks. In recent years, interval type-2 logic has grown in prominence [1] as an improvement on traditional type-1 FLC. We examine the possibility of employing interval type-2 logic to improve estimates of the network conditions into which the video stream is injected. Certainly, no other research has extended FLC to type-2 fuzzy logic congestion control for video streaming.

Within fixed networks, an FLC must compete with traditional solutions to congestion control. The main transport protocol within the Internet is the transmission control protocol (TCP). However, as TCP transport in order to ensure reliability may result in unbounded delay, the user datagram protocol (UDP) is normally preferred for video communication, while congestion control is done by a variety of application layer controllers. Both protocols are routed through the Internet by IP. IP is a best-effort protocol in the sense that packets may be lost if it proves impossible to deliver them through a variety of causes. Typically, buffer overflow occurs at a bottleneck link, but mis-routing may also occur and very occasionally in today's fixed networks, errors are detected within the packet. The traditional method of gauging congestion in the wired Internet is through packet loss, as is the case for TCP, though round-trip time plays its part. Packet losses certainly indicate congestion, but do not provide direct information on the level of network congestion or the available network bandwidth. In fact, packet loss does not signal the onset of congestion, but rather the presence of already full-blown congestion. While packet loss is acceptable in file transfer, as lost packets are simply retransmitted through TCP, playout and decode deadlines must be met when streaming video.

A compressed video stream must compete for buffer space and bandwidth across the fixed Internet, as other traffic flows continuously cross the video stream's path. Without a fixed Internet core, the mobile or pervasive Internet cannot exist. At the periphery of the fixed core network, access networks directly deliver the Internet over its final hop to the consumer. In a wireless access link, added competition for resources occurs, as energy and latency become important commodities and the wireless channel's capacity fluctuates according to the channel error conditions. Uncertainty exists in various forms: in the measured delay across a network path; in variations in the cross-traffic; and in volatile wireless channel conditions. With uncertainty comes the need to employ fuzzy logic rather than precise analytical methods.

We have found that fuzzy control is able to function in low packet loss environments, which is certainly necessary to preserve fragile compressed video streams, but is unavailable to traditional solutions as they mostly rely on packet loss to predict network conditions. Because of the predictive nature of video compression, the effect of packet loss is not confined to a single video frame but may have a repercussion across a group of pictures (GOP) within a video stream, which, in the standard codecs [2], normally consists of 12 or 15 frames. (Notice that the terms "picture" and "frame" are used interchangeably within this chapter, as they are equivalent in respect to progressive video, video without interlacing.) If packets forming an intra-coded frame (I-frame) are lost then all other frames within a GOP are affected, as all depend on motion estimation from this reference frame. If predictive frame (P-frame) packets are lost then all frames following that frame are affected until the next I-frame is transferred.

For us, FLC is a convenient tool for handling un-anticipated or unmodeled network congestion states. In respect to FLC over a fixed network, a sender-based system for unicast flows [3]. The receiver returns a feedback message that indicates time-smoothed and normalized changes to packet inter-arrival times. These allow the sender to compute the network congestion level. The sender subsequently applies a control signal either directly to an encoder in the case of live video or to an intermediate bit rate transcoder's [4] quantization level in the case of stored video. (A transcoder converts the bit rate of an encoded bitstream from the pre-encoded rate to a lower rate. To reduce delay this process may be performed in frequency space, as all encoded videos undergo a frequency transform to reduce spatial correlations in the contents.)

For reasons of cost, Bluetooth or IEEE 802.15.1 (Sect. 3.4.1) is widely available on cellular phones and laptops as a cable replacement technology. In British Telecom's plans for a next generation IP network, Bluetooth is a way of seamlessly moving between a cellular phone network and indoor wireless access network (Sect. 3.4). We take Bluetooth as an example of a centrally scheduled time division multiple access wireless system, which is the norm for multimedia traffic delivery as it avoids unbounded latencies. Effective automatic repeat request (ARQ) management is the key to both ensuring acceptable video quality at the receiver device in the event of packet error over the error-prone wireless channel and power management of the transmission.

Our pioneering development of FLC of ARQ [5] is a way of combining three factors: (1) channel state, (2) display/decode deadline, and (3) power budget. We have adopted a modular scheme whereby a two-input FLC stage with a single output is concatenated with a second FLC stage, with the output from the original FLC and an additional "remaining power" input. A modular scheme reduces the construction complexity of the design and enables future enhancements. Assuming a fixed power budget for the duration of a video clip streaming session, the declining power budget as the stream progresses has the effect of modulating the ARQ retransmission count.

Real-time delivery of video is delay-sensitive, as a frame cannot be displayed if its data arrive after their display or decode deadline [6]. In practice, a play-out 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. Error concealment at the decoder is implementation dependent, but to reduce decoder complexity, it often only consists of replacing the missing portion of the frame with a matching portion from the previous frame. The net result is poor quality video. Not only do packets arrive after their display deadline, but while retransmission takes place, other packets may either wait too long in the send buffer, or in the extreme case, arriving packets may find the send buffer full. ARQ adds to delay and, therefore, the number of retransmissions should be minimized even before taking into account their impact on the power budget.

In general, a fuzzy scheme is easily tuned by adjustment of its membership functions. A fuzzy scheme is also 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. There is also a range of hardware designs [7] for FLC to aid real-time operation. Therefore, many practical reasons exist why of all the natural algorithms, fuzzy logic is most suited to real-time control.

3.2 Related Work on Fuzzy Logic in Telecommunications

Fuzzy logic is of course a form of computational intelligence. In a survey of congestion control through computational intelligence, the authors of [8] observe that little work has been reported on deploying natural algorithms including fuzzy logic within the Internet. ATM networks, which employ access control to virtual switched circuits, are one domain to which fuzzy logic has been more extensively applied [9, 10], but for the purpose of access control. In fact, [11] reports a type-2 FLC used for that purpose. However, it should be remarked that access control does not involve dynamically controlling a video stream once it is admitted to the network, unlike congestion control. This is because once a virtual circuit is established, it is assumed that sufficient capacity already exists within the network. Moreover, there is currently strong pressure from telecommunications manufacturers to move away from ATM and towards Ethernet framing on fixed networks. Consequently, ATM is losing its relevance to contemporary networks. Because of Bluetooth (IEEE 802.15.1)'s centralized scheduling, which resembles ATM admission control, fuzzy logic video bit rate control was applied in a similar manner to a Bluetooth wireless link [12]. An interesting modular design was employed, but the main input to the fuzzy models was the Bluetooth input buffer fullness, which does not account for a number of important factors in wireless transmission including energy consumption.

Again because the problem resembles ATM admission control, in a number of papers, the authors of [13] have explored fuzzy logic to improve the performance of the random early discard (RED) router queue algorithm, and in [14] fuzzy logic was applied to DiffServ buffer occupancy for each class of layered video packets. Both these systems control the quality-of-service of video streams at routers in the face of other competing traffic. These are not end-to-end solutions to the problem of network congestion, but rely on deployment of quality-of-service strategies that in practice are confined to particular internet service providers. In the case of RED, packets are dropped to signal to TCP controlled flows that congestion is about to occur. Even if TCP could be employed for video streaming, this implies packet loss with its destructive effect on video quality.

Within video coding, fuzzy logic has found an application [15, 16] in maintaining a constant video rate by varying the encoder quantization parameter according to the output buffer state, which is a complex control problem without an analytical solution. This is an open loop solution to the problem of controlling access to the network and should be compared to the ATM-based solutions previously discussed. In our work, we have preferred a closed loop solution in which feedback for the network state from the receiver serves as input to the FLC. This allows greater awareness of traffic conditions experienced by the video stream within the network itself. In [17], fuzzy logic determines the size of different video frame type sizes and classifies the video genre. The intention is to allow modeling of variable bit rate (VBR) video traffic without the need for video sources. However, there is no attempt at real-time control.

Wireless networks represent a promising application of fuzzy logic as not only are their uncertainties inherent in network traffic, but also the wireless channel is more error prone and takes a wider variety of forms than a wired link. Additionally, the need to conserve battery energy brings into play another set of factors. In [18], fuzzy logic was applied to modeling the lifetime of a wireless sensor network, though again no real-time control took place. In [19], the problem of fading wireless channels (ones in which multi-path interference causes mutual interference between simultaneously received versions of the signal) was tackled with fuzzy filters to equalize the signal response according to variation of the wireless channel. This is a physical layer technique, whereas our work with FLC is cross-layer between the application and data link layers of the standard model for the protocol stack.

Recent applications of fuzzy logic have been with type-2 logic, which are further discussed in Sect. 3.5. Though the possibility of type-2 fuzzy systems has been known for some time [20], only recently have algorithms become available [21] allowing calculation of an interval type-2 output control value at video frame rates. The first work on interval type-2 sets, which simplifies the calculation of type-2 logic, is due to Gorzalczany [22] and other pioneering work is given credit in [23]. A growing number of type-2 controllers for robotic and industrial applications are already surveyed in [1], demonstrating the effectiveness of the new algorithms for real-time control.

3.3 Fuzzy Logic Control on the Fixed Internet

Figure 3.1 shows a video streaming architecture in which fuzzy logic is utilized to control the bit rate. A video transcoder at the server is necessary for adaptation of stored pre-encoded video-rate adaptation, while a video decoder at the client decodes the received video stream in real time. (For live video, directly changing the quantization parameter of the encoder has the same effect.) The client-side timer unit monitors the dispersion of incoming packets and relays this information to the congestion level determination (CLD) unit. The CLD unit monitors the outgoing packet stream, especially the packet sizes, and combines this information with the receiver device feedback as a basis for determining the network congestion level, C_L. This unit also computes the congestion-level rate of change, δC_L.

Principally, the timer unit measures the arriving packet inter-packet gaps (IPGs) before finding a time-smoothed and normalized estimate of the packet dispersion. An IPG is the time duration between the receipt of the end of one packet and the arrival of the next. The FLC takes C_L and δC_L as inputs and computes a sending rate that reflects the network's state. The appropriate change in the transcoder

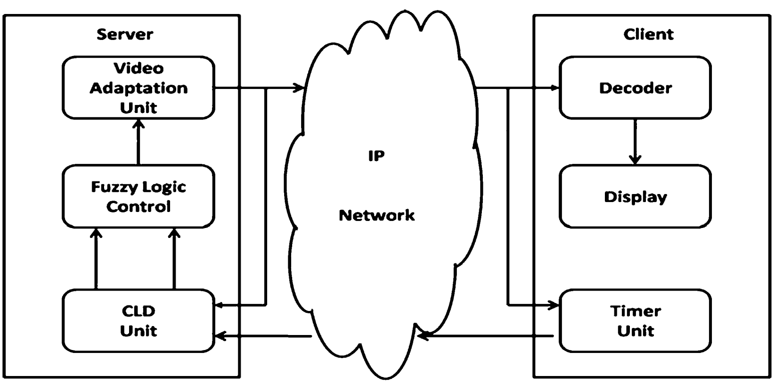


Fig. 3.1 Fuzzy logic control (FLC) video streaming architecture

quantization level is then calculated. Transported packets are received by the client, depacketized, decoded and displayed at video rate.

At the server, the video transcoder inputs the encoded video and reduces its bit rate in response to a control signal from the FLC. A lower bound to the sending rate is set to be 10% of the input sending rate. For an average input sending rate of 2 Mb/s, a lower limit of 200 kb/s is sufficient for an acceptable video quality. The transcoded video is packetized, with one slice per packet, and sent across the network within a UDP packet. European format source input format (SIF) at 25 frame/s Motion Picture Experts Group (MPEG)-2 encoded video can be partitioned into eighteen slices per frame, each slice consisting of a row of 22 macroblocks (a macroblock is a unit of motion estimation in standard codecs [2]). Apart from error resilience due to decoder synchronization markers, per-slice packetization also reduces delay at the server. Transcoded video packets are subsequently output with a constant IPG at the point of transmission. Ensuring a constant IPG reduces packet inter-arrival jitter at the client and also renders the streamed video more robust to error bursts. At 18 packets per frame, for 25 frame/s the IPG at the server before dispersion is 2.2 ms. The MPEG-2 codec was employed in this work as it has been adopted by the Digital Video Broadcasting consortium and is widely deployed by other major commercial broadcasters.

3.3.1 Calculation of the Congestion Level

Let the IPG of the packets be T_S and T_C for the packet entering the network and exiting the network, respectively. T_C will equal T_S when the available network bandwidth is equal to or more than the sending rate of the packets. In other words, as far as the application sending the packets is concerned, equality will apply when the network is not congested. A congested network will generally have a dispersive effect on the IPG, resulting in T_C being greater than T_S. The difference between T_C and T_S is, therefore, a measure of network congestion. The more congested the network, the more is the difference. The IPGs, T_C and T_S, apart from being dependent on the sending rate of the packets, are dependent on two variables: (1) the level of network congestion and (2) the size of the packets. Normalizing these variables by the packet sizes makes them only dependent on the network congestion level. Knowing the normalized values of T_C and T_S will then enable computation of the level of network congestion.

Thus, congestion level is determined without relying on packet loss, which is vital for video because, as previously mentioned in Sect. 3.1, any packet loss from an anchor or reference frame endures until the GOP completes. A mean packet transfer time G_S is found at the server and similarly G_C is found after measurements made at the client. G_S and G_C are calculated over the duration of one frame's transmission and subsequently act to estimate the level of network congestion. The sending rate is changed when and only when a feedback message arrives at the receiver. To ensure consistent quality within a frame, the change itself is only made at the beginning of a video frame.

The sending rate of the application into the network, and the receiving rate of the client from the network are calculated as R_S and R_C, respectively, in (3.1).

$$R_S = \frac{1}{G_S}; \quad R_C = \frac{1}{G_C}. \tag{3.1}$$

The difference between the two rates, *Rd*, can then be calculated as:

$$R_D = R_S - R_C. \tag{3.2}$$

The network congestion level, C_L, can subsequently be calculated as:

$$C_L = \frac{R_D}{R_S}, \tag{3.3}$$

$$C_L = 1 - \frac{G_S}{G_C}. \tag{3.4}$$

Finally, δC_L is also calculated as simply the difference between the present and previous value of C_L.

3.3.2 Fuzzy Logic Controller

The input variables were fuzzified by means of triangular-shaped membership functions, being the usual compromise between reduced computation time at the expense of a sharper transition from one state to another. Choosing the number of membership functions is important, since it determines the smoothness of the bit rate granularity. However, the number of membership functions is directly proportional to the computation time. The congestion level, the rate at which it changes and the control signals of the sample set were each partitioned into nine triangular membership functions.

Table 3.1 defines the linguistic variables for inputs C_L and δC_L. The defuzzified output has the same linguistic variables as δC_L in Table 3.2, but abbreviated with a prefix of S, e.g., SZ (zero). All the inference rules of a complete set used in simulations for a fixed Internet are given in Table 3.2.

The centre of gravity method was used for defuzzification [24]. From Table 3.2, a typical number of rules is 45 (total number of outputs). A few different rules map to the same outputs. The control signal resulting from defuzzification, *CT*, is normalized to the range (0, 1], subject to a minimum lower bound. For input bit rate R_{in}, the target output bit rate is $Rout = CT.Rin$ through multiplication. In steady state, to achieve sending rate *Rout*, the quantization scale of the transcoder is directly proportional to *CT* and the dynamic range of available quantizers. In order to

Table 3.1 Fuzzy logic control (FLC) linguistic variables for C_L and δC_L

C_L		δC_L	
Value	Meaning	Value	Meaning
CL	Low	NVH	Negative very high
CM	Medium	NH	Negative high
CH	High	NM	Negative medium
CVH	Very high	NL	Negative low
CEH	Extremely high	Z	Zero
		PL	Positive low
		PM	Positive medium
		PH	Positive high
		PVH	Positive very high

Table 3.2 FLC inference rules

$\delta C_L/C_L$	CL	CH	CH	CVH	CEH
NVH	SPH	SPM	SPL	SZ	SNL
NH	SPM	SPL	SZ	SNL	SNM
NM	SPL	SZ	SZ	SNM	SNM
NL	SPL	SZ	SNL	SNM	SNH
Z	SZ	SNL	SNM	SNH	SNH
PL	SNL	SNL	SNM	SNH	SNH
PM	SNL	SNM	SNH	SNH	SNVH
PH	SNM	SNH	SNH	SNVH	SNVH
PVH	SNM	SNH	SNVH	SNVH	SNVH

manage the combined transcoder and target decoder buffer occupancy [25] without increasing delay, a correction factor is applied in a picture dependent manner. As *CT* equates to α in Algorithm A of [26], the interested reader is referred to that paper for further details.

3.3.3 Testing FLC on a Fixed Network

The algorithm was simulated with the well-known ns-2 network simulator (v. 2.31 used). The simulated network, with a typical "dumbbell" topology, Fig. 3.2, had a tight link between two routers and all side link bandwidths were provisioned such that congestion would only occur at the tight link. A tight link is the link that instantaneously represents the link with minimum available bandwidth on a network path. In Fig. 3.2, the tight link's total bandwidth capacity can also be altered. The one-way delay of the tight link was set to 5 ms and the side links' delays were set to 1 ms. The tight link's queueing policy was defaulted to be drop-tail or FIFO and the queue

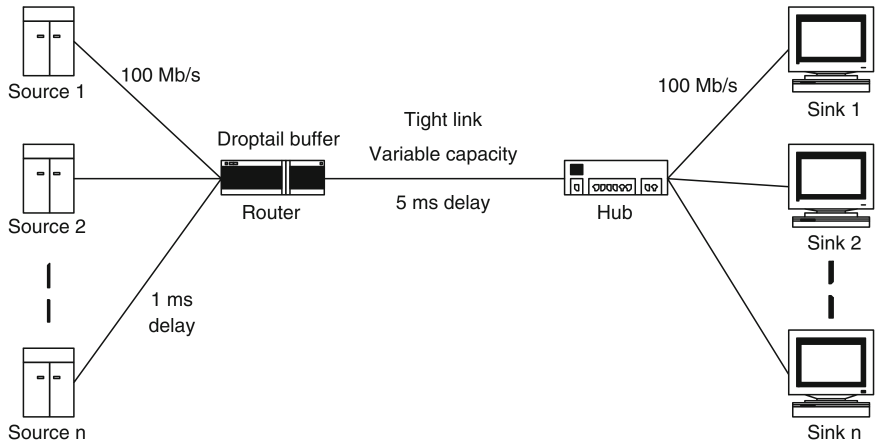


Fig. 3.2 Simulated network topology

size was set to twice the bandwidth-delay product, as is normal in such experiments to avoid packet losses from too small a buffer. An MPEG-2 encoded video "news clip" with GOP size of 12 pictures was selected as a generic input to the simulations. In live tests of 4,000 video clips [25], over 50% of clips were less than 200 s with a median of 180 s. Consequently, though the chosen news clip had a duration of 40 s, it was preferable for the experiments than a still shorter standard clip so as to provide information about the behaviour across more frames. The newsclip shows a newsreader against a moving backdrop, with moderate motion and by implication with moderate encoding complexity.

3.3.4 Calibration Experiments

Figures 3.3a, b illustrate a set of calibration experiments: simulating streaming of the MPEG-2 "news clip" encoded at a VBR with an average bit rate of 2.7 Mb/s against a constant bit rate (CBR) background cross traffic, with packet size 700 B. The bottleneck link was set to 2 Mb/s and then 3 Mb/s., which are typical of constrictions at the core network boundary before entering an edge local area network (LAN). The goal of the controller is to detect the available channel rate and make this the target bit rate of the video transcoder. The results in Fig. 3.3 show that the FLC clearly detects and adapts to the available network bandwidth. For example, in Fig. 3.3a for a 2 Mb/s capacity link, the average background traffic is initially 1.5 Mb/s and the FLC sending rate is at the 0.5 Mb/s level and when the average CBR rate steps down to 0.5 Mb/s, the FLC stream is at the 1.5 Mb/s level. The FLC rate retains some of the inherent "burstiness" in the instantaneous rate of the news

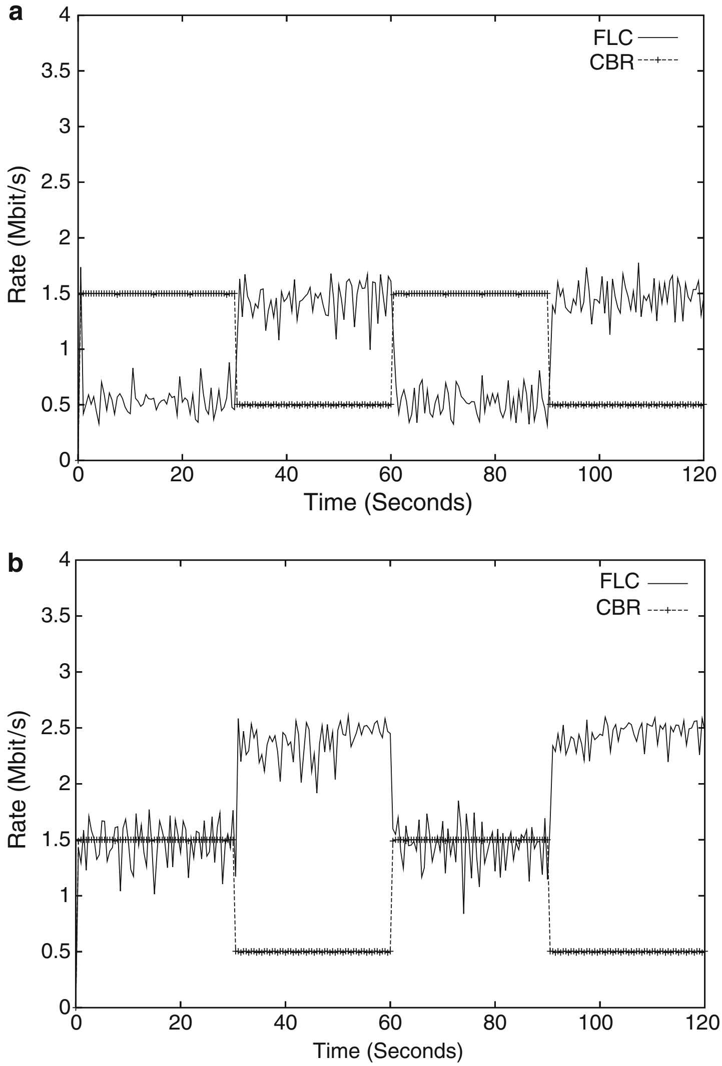


Fig. 3.3 FLC instantaneous sending rate of an MPEG-2 video stream in the presence of step-wise varying Constant Bit Rate cross traffic for a (**a**) 2.0 Mb/s and (**b**) 3.0 Mb/s bottleneck

clip, especially when a scene contains motion. Without adequate buffering, the FLC (or CBR) traffic may suffer packet loss. However, the intention of this set of experiments was primarily to demonstrate the ability to track the changing available bandwidth.

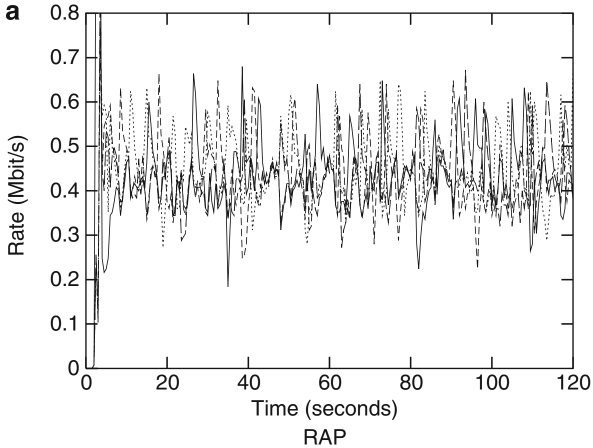
3.3.5 Comparison with Traditional Congestion Controllers

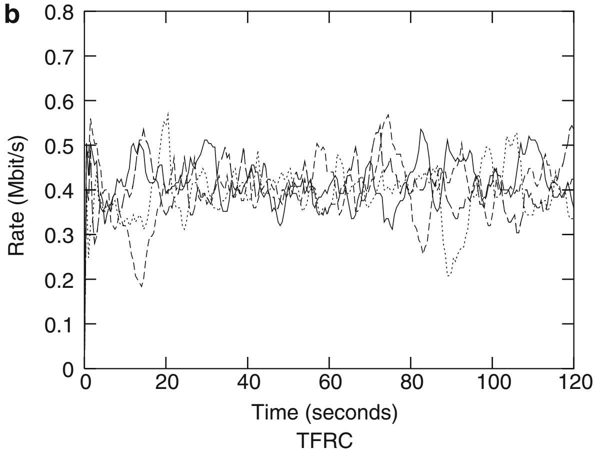
This Section examines the ability of an FLC to coexist with traffic either controlled by a different congestion controller or by another FLC. Rate adaptation protocol (RAP) [27] is a well-known controller that varies the IPG between fixed-size packets to allow its average sending rate to approach TCP's for a given available bandwidth. Every smoothed round trip time (RTT), RAP implements an arithmetic increase multiplicative decrease (AIMD)-like algorithm [28] with the same thresholds and increments as TCP. Because this would otherwise result in TCP's "sawtooth"-like sending rate curve with obvious disruption to video streams, RAP introduces fine-grained smoothing (turned on in our tests), which takes into account short- and long-term RTT trends. RAP has a known weakness in heavy congestion, as it does not employ timeouts and, consequently, is likely to be more aggressive than TCP. Because of its pioneering role and its close resemblance to TCP, RAP has frequently served as a point of comparison for congestion controllers. To ensure fairness to RAP, publicly-available ns-2 models were employed.

Comparison was also made with the TCP-friendly rate control (TFRC) protocol, the subject of an RFC [29] and a prominent method of congestion control from the originators of the "TCP-friendly" concept. To ensure fairness, the publicly available TFRC ns-2 simulator model (in the form of object tcl scripts to drive the simulator) was availed of from http://www.icir.org/tfrc/. In TFRC, the sending rate is made a function of the measured packet loss rate during a single RTT duration measured at the receiver. The sender then calculates the sending rate according to the TCP throughput equation given in [30].

A potential weakness of TFRC is the response to short-term TCP flows, typically hyper text transfer protocol (HTTP) traffic, which never develop long-term TCP flow behaviour. TFRC was designed to produce smooth multimedia flows, but assumes constant-sized large (or maximum transport unit (MTU)) packet sizes, which are not suited to MPEG-2 encoded VBR video. The TCP throughput equation is not necessarily designed for conditions in which there are just a few flows. This is because changes to the sending rate may alter the conditions at the bottleneck link, which affect the feedback, resulting in a non-linear system. However, TFRC's reliance on a packet loss measure is its principle weakness in respect to congestion control of transcoded video. Notice that unlike TCP, and in common with FLC, TFRC does *not* guarantee delivery of packets. Another weakness of TFRC is that rate change decisions are made every RTT. It appears that less frequent decisions would lead to a tendency for a TFRC controlled flow to dominate the bandwidth across a tight link [31]. Unfortunately, too frequent rate changes, as remarked earlier, can have a disturbing subjective effect upon the viewer of decoded video.

In intra-protocol fairness tests, a number of flows were simultaneously streamed over the shared tight link. The simulator was unable to set the feedback at exactly the same time for all the controllers and, therefore, as for live Internet sources, the output is not identical, even though the congestion controllers are. Figure 3.4 records the intra-protocol allocation of bandwidth for the three congestion controllers for a 2.0 Mb/s bottleneck over 120 s. Visual inspection indicates that the FLC flow is





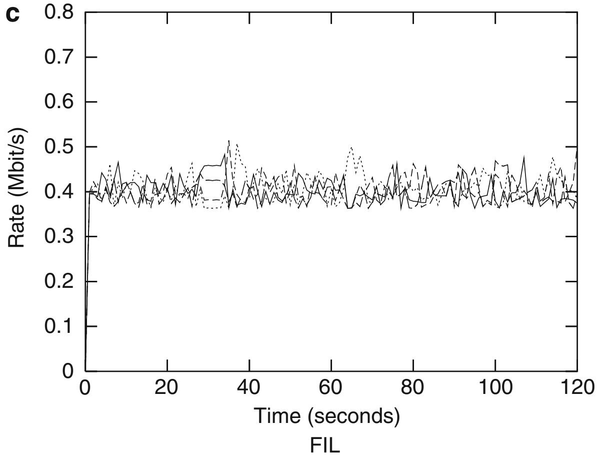


Fig. 3.4 Five simultaneous flows each with identical congestion controllers for a 2 Mb/s bottleneck with (**a**) rate adaptation protocol controllers, (**b**) TCP-friendly rate control (TFRC) controllers, and (**c**) with FLC

Table 3.3 Standard deviations (s.d.'s) for a sample of two flows over a 120 s test

Congestion controller	Bottleneck b/w (Mb/s)	Flow 1 (F1) s.d. (Mb/s)	Flow 2 (F2) s.d. (Mb/s)	Abs(F1–F2) s.d. (Mb/s)
TFRC	2.0	0.44849	0.63709	0.18860
RAP	2.0	1.35377	0.99761	0.37616
FLC	2.0	0.27979	0.28815	0.00836

certainly smoother than RAP and often smoother than TFRC, as is also shown quantitatively in Table 3.3. Though all flows, for each type of controller, cluster around the correct level, given the available bandwidth, RAP is visibly more "bursty" in its response. The RAP plot includes a large initial sending rate, which was discarded in subsequent analysis.

The issue is less easy to resolve between TFRC and the FLC and to do so standard deviations (s.d.'s) were taken. Table 3.3 tabulates the s.d.'s for a sample of two flows (the other flows were similar) taken at random from the five, for each of the congestion controllers in Fig. 3.4 (with bit rate around 0.4 Mb/s for the 2 Mb/s bottleneck). The FLC's s.d.'s are less than those of TFRC (and RAP), showing that in these circumstances the FLC's output is smoother. Moreover, there is more consistency in the FLC's behaviour for tight bottlenecks, as the absolute difference column in Table 3.3 indicates. It should be remarked that for less constricted bottlenecks, TFRC fares better in respect to smoothness, but these situations are not so critical.

Turning to inter-protocol allocation of bandwidth, when multiple flows compete, Fig. 3.5, then the average bandwidth of the TFRC flows is always above that of the FLC flows and above the ideal or equally-shared bottleneck bandwidth. This behaviour of TFRC, when exposed to a tight link without TCP traffic, would work to the disadvantage of other traffic. In fact, converged IP networks are just such networks, as these networks are under construction with multimedia traffic in mind [32]. If replicated in such a network, it is also likely that this behaviour would result in increased packet loss for the TFRC flows. It is postulated that the cause is oversensitivity on the part of TFRC's response to variation in packet loss and RTT.

3.4 Fuzzy Logic Control on the Wireless Internet

This Section considers using Bluetooth (IEEE 802.15.1) [33] for video streaming across an access network. Though Bluetooth was originally conceived as a Personal Area Network with a role in cable replacement, its centralized control means that it is potentially suitable for real-time applications such as video streaming. The master node responsible for scheduling acts as a wireless access point, connecting to the fixed Internet. Consequently, Bluetooth can act as an access network in wireless hot-spots such as at airports and on trains or coaches. The Enhanced Data Rate (EDR) of Bluetooth version 2.1 [34] now has a peak user payload of 2.2 Mb/s (gross air rate 3.0 Mb/s), which is the same average rate offered by some implementations

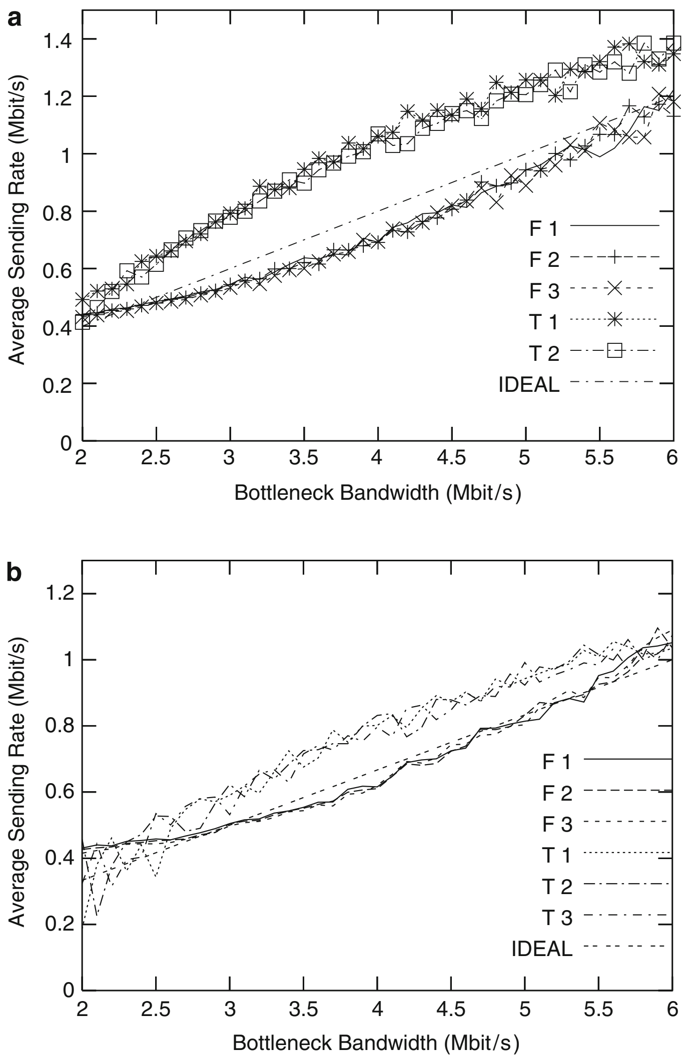


Fig. 3.5 Multiple competing flows with different controllers across a range of bottlenecks, compared to an ideal per-flow allocation of bandwidth (**a**) 3 FLC and 2 TFRC flows (**b**) 3 FLC and 3 TFRC flows

of IP-TV. Compared to IEEE 802.11 (Wi-Fi)'s [35] typical current usage of 100–350 mA, Bluetooth's consumption is 1–35 mA, implying that for mobile multimedia applications with higher bandwidth capacity requirements, Bluetooth is a preferred solution. Many cellular phones are also equipped with a Bluetooth transceiver and larger resolution screens of CIF (352 × 288) and QCIF (176 × 144) pixel size. Consequently, Bluetooth is part of the "Bluephone" solution to connectivity, whereby a user may wander seamlessly between a home access network and a telephony network [36].

3.4.1 Bluetooth Background

Bluetooth is a short-range (less than 10 m for class 2 devices), radio frequency interconnect. Bluetooth's short range, robust frequency hopping spread spectrum (FHSS) and centralized medium access control through time division multiple access and time division duplex (TDD) means it is less prone to interference from other Bluetooth networks. Bluetooth employs variable-sized packets up to a maximum of five frequency-hopping time-slots of 625 μs in duration. Every Bluetooth frame consists of a packet transmitted from a sender node over 1, 3 or 5 timeslots, while a receiver replies with a packet occupying at least one slot, so that each frame has an even number of slots. Therefore, in master to slave transmission, a single slot packet serves for a link layer stop-and-go ARQ message, whenever a corrupted packet payload is detected.

The timeout or retransmission limit value by default is set to an infinite number of retransmissions. On general grounds, this is unwise in conditions of fast fading caused by multi-path echoes, as error bursts occur. Another source of error bursts is co-channel interference by other wireless sources, including other Bluetooth piconets, IEEE 802.11b,g networks, cordless phones, and even microwave ovens. Though this has been alleviated to some extent in version 1.2 of Bluetooth by Adaptive Frequency Hopping [37], this is only effective if interference is not across all or most of the 2.402–2.480 GHz unlicensed band. However, both IEEE 802.11b and g may occupy a 22 MHz sub-channel (with 30 dB energy attenuation over the central frequency at $\pm$11 MHz) within the 2.4 GHz band. Issues of interference might arise in apartment blocks with multiple sources occupying the 2.4 GHz band or when higher-power transmission occurs such as at WiFi hot-spots. For Bluetooth, an ARQ may occur in various circumstances but the main cause of packet error [38] is payload corruption, which is the simplified assumption of this work.

3.4.1.1 Fuzzy Logic Control of ARQ

Figure 3.6 shows a complete two-stage FLC system for adaptive ARQ. In the first stage, there are two inputs: buffer fullness and the normalized delay of the head of the queue packet. Bluetooth buffer fullness is a preferable measure (compared to delay or packet loss) of channel conditions and of buffer congestion, as was established in [39]. 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. As an FLC input, buffer fullness is normalized to the size of the send buffer. The retransmission count of the packet at the head of the Bluetooth send queue will affect the delay of packets still to be transmitted. Retransmissions not only overcome the effect of noise and interference, but also cause the send buffer queue to grow, with the possibility of packet loss from send buffer overflow, which is why it is necessary to also introduce an active buffer, as discussed in the next Section. The second FLC input modulates the buffer fullness input by the already experienced delay of the head of queue packet.

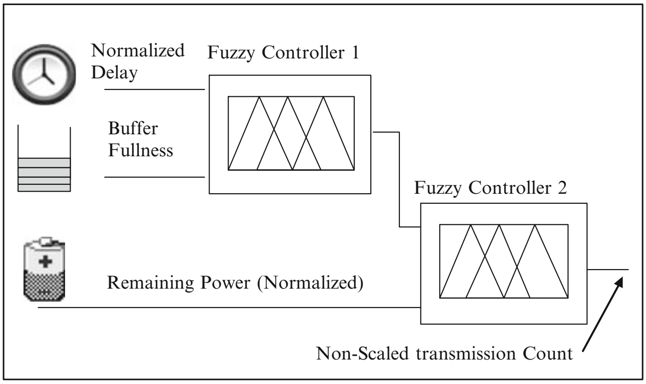


Fig. 3.6 Overview of the FLC of ARQ system

Table 3.4 FLC stage 1 If. . .then rules used to identify output fuzzy subsets from inputs

	Delay/Deadline				
Buffer fullness	Too low	Low	Normal	High	Too high
High	Normal	Normal	Low	Too low	Too low
Normal	Too high	High	Normal	Low	Too low
Low	Too high	Too high	High	Low	Too low

The output of the first stage FLC forms the input to the second stage FLC. Though it might be possible to modify the first stage output by non-fuzzy logic means, the arrangement adopted neatly provides for an all FLC solution. The other input to the second stage is normalized remaining power, assuming a predetermined power budget for streaming of a particular video clip, which diminishes with time and retransmissions. The output of the second stage is a transmission count, which is subsequently scaled according to picture type importance. In other words, reference picture packets are allocated a greater allocation of power and of the reference frame packets I-frame packets are favoured over P-frame packets.

It is important to note that any packet in the send buffer is discarded if its deadline has expired. However, this takes place after fuzzy evaluation of the desired ARQ retransmission count. In practice, the inputs to the FLC were sampled versions of buffer fullness and packet delay deadline, to avoid excessive ARQ retransmission count oscillations. The sampling interval was every 20 packets. Table 3.4 shows the "if. . .then" rules that allow input fuzzy subsets to be combined to form an output from stage one and an input to stage two. Notice more than one rule may apply because of the fuzzy nature of subset membership. The output of stage one is combined with a fuzzy input for "remaining power," and the "if. . .then" rules resulting in the final non-scaled transmission count are in Table 3.5. Triangular membership functions were employed to model all inputs and outputs.

Table 3.5 FLC stage 2 If. . .then rules used to identify output fuzzy subsets from inputs

	Output 1				
Remaining power	Too low	Low	Normal	High	Too high
High	Too low	Low	High	Too high	Too high
Normal	Too low	Low	Normal	High	High
Low	Too low	Too low	Low	Low	Normal

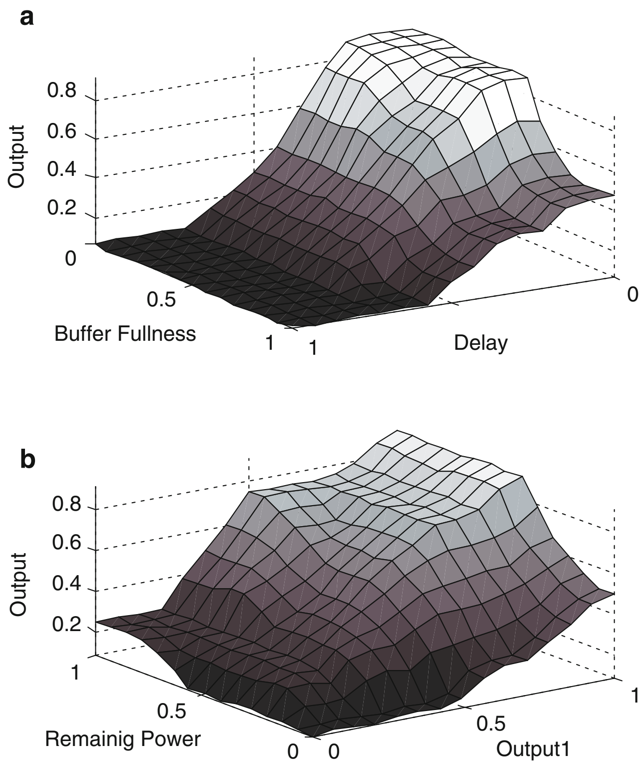


Fig. 3.7 (**a**) Stage 1 FLC control surface resulting from FLC ARQ (**b**) Stage two control surface giving the transmission count output (before subsequent scaling)

The inputs were combined according to the Mamdani model [24] to produce the output values for each stage. The standard centre of gravity method was employed to resolve a crisp output value. The fuzzy control surfaces are represented in Fig. 3.7, as derived from the Matlab Fuzzy Toolbox v. 2.2.4.

Clearly a packet can only be transmitted an integer number of times, but the final 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 the fractional part of the crisp output value, then that value was rounded up to the nearest integer, otherwise it was rounded down. The advantage of the randomization procedure over simple quantization is that, in the long term, the mean value of the output numbers of transmissions will converge more closely to a desired output level.

The crisp output after defuzzification of the FLC is scaled according to picture type. A simple scaling of 5:3:2 was applied respectively for I-, P-, B-pictures, given up to five maximum transmissions. (B-pictures are bi-predicted but themselves have no predictive role). For example, if the crisp output value was 0.7 and a P-picture packet was involved then the value after scaling is $0.7 \times 3.0 = 2.10$. Then, the random number-based resolution results in three transmissions, if the random number is less than or equal to 0.10 and two transmissions otherwise. It should be mentioned that a maximum value of five retransmissions was also adopted in the priority queueing tests in [40], albeit for an IEEE 802.11 wireless network.

3.4.2 *Testing FLC on a Wireless Network*

3.4.2.1 Buffer Specification

An important but sometimes neglected aspect of simulating a wireless link when multimedia is involved is the specification of the buffer. ARQ adds to delay and, therefore, the number of retransmissions should be minimized. To reduce the number of retransmissions, an active buffer or deadline-aware buffer (DAB) is possible. In a conservative send buffer discard policy, 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, the deadline is set as the maximum time that the playout buffer can delay the need for a packet. In simulations, the display deadline was set to 0.10 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. For a 12-picture GOP, this is the time to display 11 frames, i.e., 0.44 s at 25 frame/s. For P-picture packets, the decode deadline will vary depending on the number of frames to the end of the GOP. For B-pictures, the decode deadline is set to zero (as B-pictures have no value to the decoding of future frames). The decode deadline is added to the display deadline and a packet is discarded from the send buffer after its total deadline expires.

3.4.2.2 Channel Specification

Unlike a fixed network, for which optical transmission at the Internet core makes channel errors very unlikely, the wireless channel characteristics are important, as errors are very likely to be introduced from various sources of noise and interference. A Gilbert-Elliott [41, 42] two state discrete-time, ergodic Markov chain is a standard model for the wireless channel error characteristics between a Bluetooth master and slave node. By adopting this model, it is 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), $T_g = 3{,}200$ and $T_b = 400$, which implies from:

$$T_g = \frac{1}{1 - Pgg}, \qquad T_b = \frac{1}{1 - Pbb} \tag{3.5}$$

that, given the current state is good (g), Pgg, the probability that the next state is also g, is 0.9996875 and Pbb, and 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 BER during a good state was set to $a \times 10^{-5}$ and during a bad state to $a \times 10^{-4}$, where a is a scaling factor (of course, not the fuzzy output scaling factor) and is subsequently referred to as the channel parameter.

3.4.2.3 Simulations

The same "news clip," as was used for the fixed network, was employed in tests for the wireless access network. For these tests, the clip was encoded at a mean rate of 1.5 Mb/s. However, the packetization policy differed from the one slice per packet policy for the fixed network discussed previously in this chapter. In [43], fully filled Bluetooth packets were formed using maximal bandwidth five time-slot packets, regardless of slice boundaries. These packets carry a 1,021 B payload. While this results in some loss in error resilience (and added latency), as each MPEG-2 slice contains a decoder synchronization marker, in [43] it is shown that the overall delivered video quality is superior to choice of smaller packet sizes.

Figure 3.8 shows the output of stage 1 of the FLC as the video clip was passed across a Bluetooth link with channel parameter a set to two. The high variability of the output is due to the repeated onset of bad states occasioned by the Gilbert-Elliott channel model.

The normalized power budget for the clip declines with the number of bits passed across the link and the loss is exacerbated by repeated retransmissions during bad states. As the power budget changes linearly, this has the effect of modulating the original input, as illustrated by Fig. 3.9 for the output of FLC stage 2, again with channel parameter set to two. As the power budget varies linearly over time, the envelope of the output from stage 2 also varies in a linear fashion over time. This is as required because as the sending device's power budget declines it must reduce the number of retransmissions in order to reduce its power usage.

After the removal of deadline expired packets, through operation of the DAB, the buffer fullness input to stage one of the send buffer oscillates around a level well-below the 50 packet maximum. Consequently, head-of-line packet delay acts as a typical trimming input to the FLC stage one unit, as its pattern resembles that of buffer fullness over time.

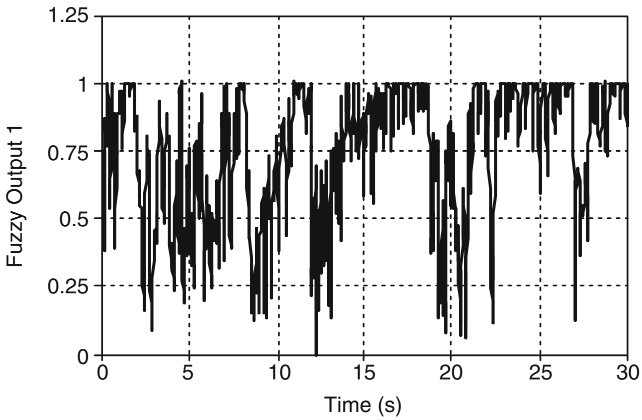


Fig. 3.8 Output from stage one of the FLC, with $a = 2$

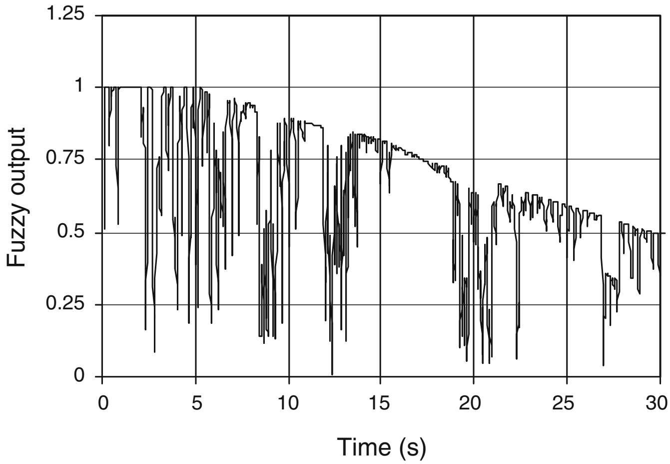


Fig. 3.9 Output from stage two of the FLC, with $a = 2$

3.4.2.4 Comparison with Default ARQ

A comparison was made between the default Bluetooth scheme of infinite ARQ and no power control with the FLC scheme with power control. That is the default scheme was allocated an infinite power budget. For fairness both schemes were compared with a DAB in place, though, of course, a DAB is not a feature of the Bluetooth standard. The channel parameter, a, was varied in the tests to show the impact of differing channel conditions. As an aid to comparison, the FLC scheme was also tested with an infinite power budget.

Figure 3.10 compares the ratio of packets lost to total packets arriving in the send buffer. Even though the FLC scheme is compensating for a diminishing power

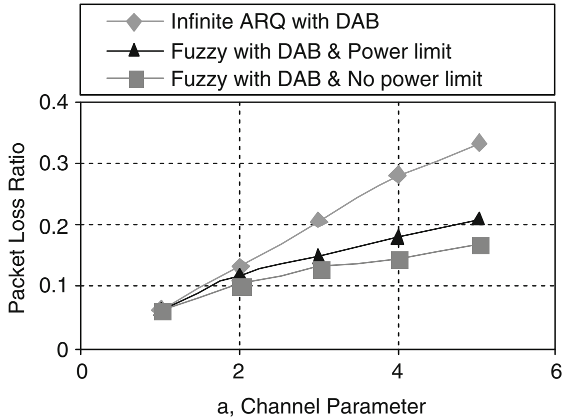


Fig. 3.10 Packet loss during transmission of the News video clip, with the default scheme and the FLC power-aware scheme

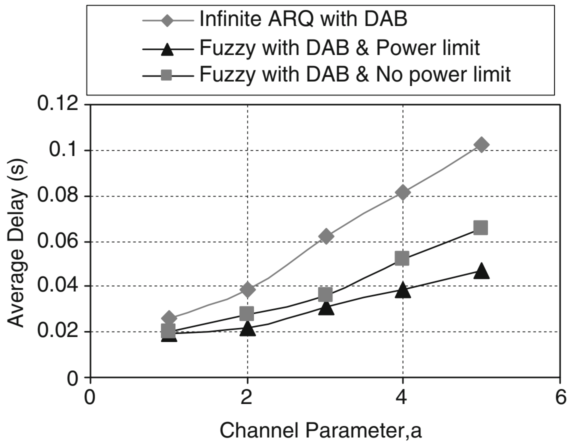


Fig. 3.11 Average packet delay during transmission of the News video clip, with the default scheme and the FLC power-aware scheme

budget, it shows a clear superiority. The effect is more pronounced with worsening channel conditions. The average delay of successfully transmitted packets was also considerably reduced under the FLC scheme, Fig. 3.11, while the default ARQ scheme results in a more rapid climb to its peak average value. Larger average delay will impact start-up time in one-way streaming and add to overall delay in a two-way interactive video exchange, such as for a videophone connection. Notice that removing the power budget results in more delay for the FLC scheme than with a power budget because the scheme is not handicapped by the need to reduce transmissions for power budget considerations. Either way the scheme is superior to default ARQ in delay (and also in reduced packet loss). Crucially, the FLC is able to save power over the non-power-aware default ARQ scheme, Fig. 3.12. The impact is clearly greater as channel conditions worsen.

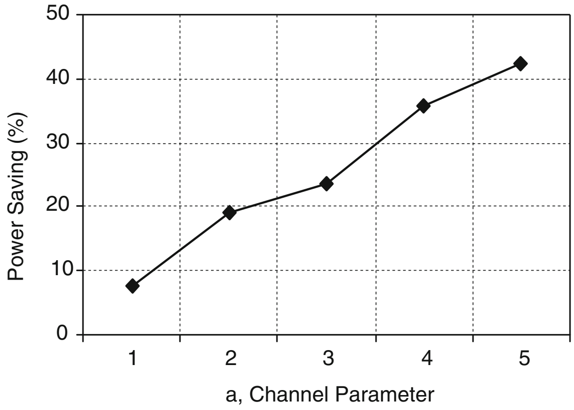


Fig. 3.12 Relative power saving of the FLC power-aware ARQ scheme compared to that of the default ARQ scheme

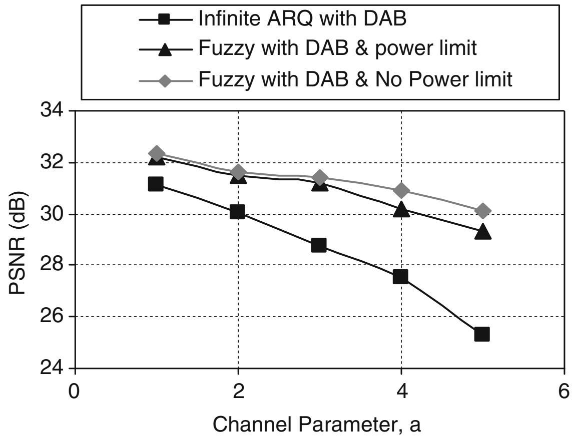


Fig. 3.13 Comparison of PSNR for the News video clip between the Bluetooth default and FLC ARQ schemes with DAB

Video quality is objectively measured by the peak signal-to-noise ratio (PSNR), which has logarithmic units of measurement, the decibel (dB). Provided PSNR comparisons are made across the same video [2], PSNR is a reliable measure of video quality, though other subjective effects also have a role. Considering the packet loss statistics of Fig. 3.10, it is not surprising, Fig. 3.13, that the mean PSNR of FLC ARQ is better than that of the default scheme and the relative advantage becomes more so as the channel conditions worsen. A significant part of that advantage is also due to the superiority of FLC ARQ and there is little difference between FLC ARQ with and without a power budget in better channel conditions. Notice that for power-aware control, averaged PSNR figures do not "show the whole story," as the achievable PSNR will deteriorate over time as the available power reduces.

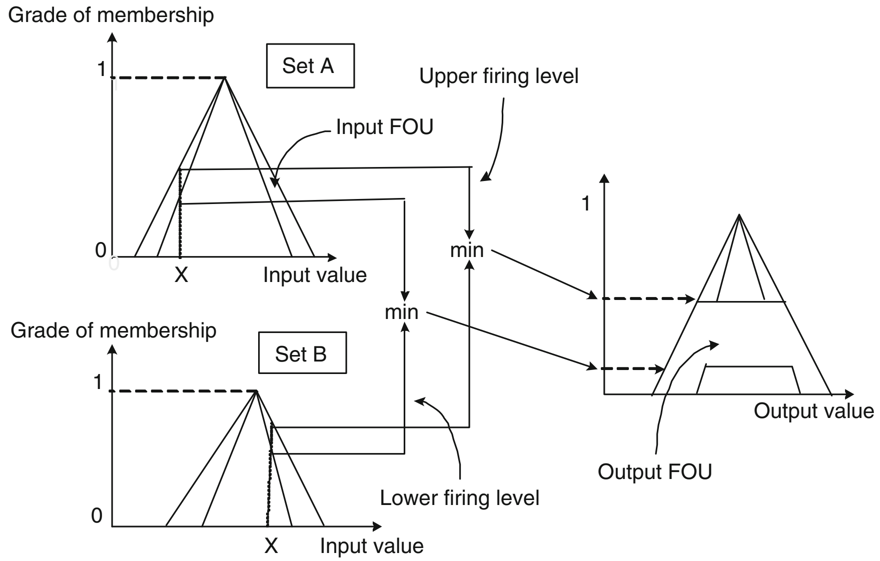


Fig. 3.14 Interval type-2 fuzzy logic calculation of an output FOU

This confirms previous experience [44] that for the very worst channel conditions shown in Fig. 3.14, i.e., $a = 5$, then the mean PSNR is improved by around 3 dB if an infinite power budget is assumed. Thus, power conservation comes at a cost to the receiver in reduced video quality and is a trade-off that might be open to user configuration. The end result is that power-awareness has been realistically factored in, resulting in over 40% saving in power, Fig. 3.12, in the same conditions.

3.5 Future Development: Type-2 Fuzzy Logic

A traditional, type-1 FLC is not completely fuzzy, as the boundaries of its membership functions are fixed. This implies that there may be unforeseen traffic scenarios for which the existing membership functions do not suffice to model the uncertainties in the video stream congestion control task. An interval type-2 FLC can address this problem by extending a footprint-of-uncertainty (FOU) on either side of an existing type-1or traditional membership function. In interval type-2 fuzzy logic, the variation is assumed to be constant across the FOU, and hence, the designation "interval."

Consider two type-2 sets A and B, then strictly an infinite number of membership functions (not all necessarily triangular) can exist within the FOUs of sets A and B. However, interval type-2 sets allow the upper and outer firing levels to be taken rather than the complete range of firing levels, as shown in Fig. 3.14. The minimum operator (min) acts as a t-norm on the upper and lower firing levels to produce a firing interval. The firing interval serves to bound the FOU in the output triangular membership function shown to the right in Fig. 3.14. The lower trapezium outlines the FOU, which itself consists of an inner trapezoidal region that is fixed in extent. The minimum operator, used by us as a t-norm, has the advantage that its implementation cost is less than a product t-norm. (A t-norm or triangular norm is a generalization of the intersection operation in classical logic.) Once the FOU firing interval is established, Centre-of-Sets type reduction was applied by means of the Karnik-Mendel algorithm, which is summarized in [21]. Type reduction involves mapping the interval output set to a type-1 set. In practice, defuzzification of this type-1 output fuzzy set simply consists of averaging maximum and minimum values. The result of defuzzification is a crisp value that determines the change in the video rate.

Results adapting an FLC for congestion control over the fixed Internet have been encouraging [45]. For example in Fig. 3.15, (a) News and (b) Football VBR sources

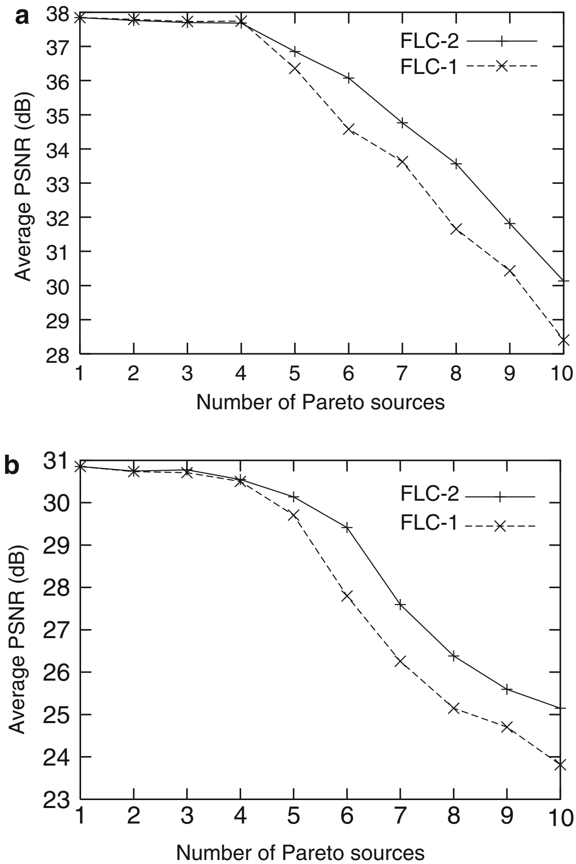


Fig. 3.15 Interval type-2 and type-1 FLCC showing resulting PSNR when tracking against an increasing number of Pareto sources for (**a**) news and (**b**) football video clips

are controlled against the same cross-traffic on the tight link of a dumbbell topology network. "Football" is a video clip with the same encoding characteristics as the News clip employed elsewhere in this chapter, with a mean rate of 1 Mbit/s. Where it differs is that there is considerable motion within the video sequence, increasing the encoded bitstream's complexity owing to motion estimation and compensation. A number of long-tailed, Pareto probability distribution function (pdf) with shape factor of 1.3 sources were configured. The heavy-tailed nature of Internet file sizes and the synchronization of TCP connections are two factors that give rise to this type of traffic source on the Internet. Each 500 kbit/s source was set with 500 ms for both idle and burst time. The resulting video quality varied from moderately good to weak as the number of sources sharing the tight link increased. For "Football," at lower background traffic intensities, the type-1 FLC at a few data points gave a little better quality owing to the pattern of losses' impact on the compressed stream. However, as the number of cross-traffic sources increased, the relative gain from interval type-2 FLC became apparent.

3.6 Conclusion and Discussion

The anticipated growth of IPTV makes selection of suitable congestion controllers for video stream traffic of vital concern. Traditional congestion controllers are strongly motivated by the need to avoid congestion collapse on the fixed Internet and, as such, mimic with UDP transport the average throughput behaviour of TCP, which is still responsible for transport of the great majority of fixed Internet traffic. It is an open question whether this form of congestion control is suited to video streaming because of the need for packet loss feedback, when no loss is preferred for video transport, and because of residual TCP-like rate fluctuations that persist when these controllers are applied. Compared to TCP-emulators, such as TFRC and RAP, a fuzzy logic trained system's sending rate is significantly smoother when multiple video-bearing sources share a tight link. Using the FLC system in this chapter similarly results in a fairer allocation of bandwidth than TFRC and RAP. It is also more suited to video dominated networks such as the all-IP networks now being constructed.

Fuzzy logic may be even more suited to resource control on wireless access networks, which now frequently form the final network hop to the end user. Transmission of higher quality video over a wireless interconnect has been long sought. However, it is important to factor in power usage and not simply regard a wireless channel as a fixed channel with the addition of errors, as mobile devices are typically battery powered. In this chapter, FLC of ARQ (in a case study concerned with the Bluetooth wireless interconnect) is able to respond to a fixed power budget that subsequently diminishes over time. Other factors included are packet delay deadlines (both display and, for anchor picture packets, decode deadlines) and send buffer congestion. FLC, which varies its transmission policy with packet picture type, still outperforms a default fully-reliable ARQ scheme, resulting in superior delivered video quality and reduced delay.

One problem with FLC has always been that compared to mathematically based analytic methods, it has always been possible to object that the response to unmodeled network states is difficult to predict. However, type-2 logic can now model uncertainty in network conditions to a still greater extent, and therefore, should increasingly find applications in video streaming over fixed and wireless networks.

Acknowledgments The authors gratefully acknowledge assistance from C. Wagner and H. Hagras in applying interval-type 2 fuzzy logic to an original type-1 controller.

This work was supported by the EPSRC, UK under grant no. EP/C538692/1.

References

1. Hagras, H.: Type-2 FLCs: A new generation of fuzzy controllers. IEEE Comput. Intell. **2(1)**, 30–43 (2007)
2. Ghanbari, M.: Standard codecs: Image compression to advanced video coding. The Institute of Electrical Engineering Press, London, (2003)
3. Jammeh, E., Fleury, M., Ghanbari, M.: Fuzzy logic congestion control of transcoded video streaming without packet loss feedback. IEEE Trans. Circuits Syst. Video Technol. **18(3)**, 387–393 (2008)
4. Assunção, A. A., Ghanbari, M.: A frequency domain video transcoder for dynamic bit-rate reduction of MPEG-2 bit streams. IEEE Trans. Circuits Syst. Video Technol. **8(8)**, 953–967 (1998)
5. Razavi, R., Fleury, M., Ghanbari, M.: Power-constrained fuzzy logic control of video streaming over a wireless interconnect. EURASIP J. Adv. Signal Process. 14 (2008). Available online at http://www.hindawi.com/journals/asp/2008/560749.html
6. Kalman, M., Ramanathan, P., Girod, B.: Rate-distortion optimized video streaming with multiple deadlines. Int. Conf. on Image Processing, 662–664, Singapore, Sept. (2003)
7. Baturone, I., Barriga, A., Sánchez-Solano, S., Jiménez, C., López, C.: Microelectronic design of fuzzy logic-based systems. CRC Press, Baton Rouge, FO, (2000)
8. Pitsillides A., Sekercioglu, A.: Congestion control. In W. Pedrycz and A. Vasiliakos, (eds.) Computational Intelligence in Telecommunications Networks, CRC Press, Boca Raton, FL, pp. 109–158 (2000)
9. Ghosh, S., Razouki, Q., Schumacher, H. J., Celmins, A.: A survey of recent advances in fuzzy logic in telecommunications networks and new challenges. IEEE Trans. Fuzzy Syst. **6(3)**, 443–447 (1998)
10. Şekercioglu, A., Pitsillides, A., Vasilakos, A.: Computional intelligence in management of ATM networks: A survey of current state of research. Soft Comput. J. **5(4)**, 257–263 (2001)
11. Liang, Q., Karnik, N., Mendel, J. M.: Connection admission control in ATM networks using survey-based type-2 fuzzy logic system. IEEE Trans. Syst. Man Cybern. C Appl. Rev. **30(3)**, 329–339 (2000)
12. Kazemian, H. B., Meng, L.: An adaptive control for video transmission over Bluetooth. IEEE Trans. Fuzzy Syst. **14(2)**, 263–274 (2006)
13. Rossides, L., Chrysostemou, C., Pitsillides, A., Şekercioglu, A.: Overview of Fuzzy-RED in Diff-Serv networks. Soft-Ware 2002, 2–14, Coleraine, April (2002)
14. Wang, X., D. Ye, D., Wu, Q.: Using fuzzy logic controller to implement scalable quality adaptation for stored video in DiffServ networks. 12th Int. PacketVideo workshop. Pittsburgh, PA, April (2002)
15. Leone, A., Bellini, A., Guerrieri, R.: An H.261 fuzzy-controlled coder for videophone sequences. IEEE World Conference on Computational Intelligence, 244–248 June (1994)

16. Grant, P. M., Saw, Y.-S., Hannah., J. M.: Fuzzy rule based MPEG video rate prediction and control. Eurasip ECASP Conference, 211–214 (1997)
17. Liang, Q., Mendel, J. M.: MPEG VBR video traffic modeling and classification using fuzzy techniques. IEEE Trans. Fuzzy Syst. **9(1)**, 183–193 (2001)
18. Shu, H., Liang, Q., Gao, J.: Wireless sensor network lifetime analysis using interval type-2 fuzzy logic systems. IEEE Trans. Fuzzy Syst. **16(2)**, 416–427 (2008)
19. Liang, Q., Mendel, J. M.: Equalization of time-varying nonlinear channels using type-2 fuzzy adaptive filters. IEEE Trans. Fuzzy Syst. **8(5)**, 551–563 (2000)
20. Zaddeh, L. A.: The concept of linguistic variable and its application to approximate reasoning. Inform. Sci. **8**, 199–249 (1975)
21. Mendel, J. M.: Type-2 fuzzy sets and systems: An overview. IEEE Comput. Intell. **2(1)**, 20–29 (2007)
22. Gorzalczany, M. B.: A method of inference in approximate reasoning based on interval-valued fuzzy sets. Fuzzy Sets and Systems **21**, 1–17 (1987)
23. John, R., Coupland, S.: Type-2 fuzzy logic: A historical view. IEEE Comput. Intell. **2(1)**, 57–62 (2007)
24. Jang, J.-S. R., Sun, C.-T., Mitzutani, E.: Neuro-fuzzy and softcomputing, Prentice Hall, Upper Saddle River, NJ, (1997)
25. Chun, J., Zhu, Y., Claypool, M.: FairPlayer or foulPlayer? – head to head performance of RealPlayer streaming video over UDP versus TCP. Worcester Polytechnic Institution, Worcester, MA, Tech. Rep. May (2002)
26. Assunção, P. A. A., Ghanbari, M.: Buffer analysis and control in CBR video transcoding. IEEE Trans. Circuits Syst. Video Technol. **10(1)**, 83–92 (2000)
27. Rejaie, R., Handley, M., Estrin, D.: RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet. IEEE INFOCOM, 1337–1345, New York, Mar. (1999)
28. Cai, L., Shen, X., Pan, J., Mark, J. W.: Performance analysis of TCP-friendly AIMD algorithms for multimedia applications. IEEE Trans. Multimed. **7(2)**, 339–335 (2005)
29. Handley, M., Floyd, S., Padyhe, S. J., Widmer, J.: TCP friendly rate control (TFRC): Protocol specification. IETF RFC 3448 (2003). Available online at http://www.ietf.org/rfc/rfc3448.txt
30. Padyhe, J., Firoiu, V., Towsley, D., Krusoe, J.: Modeling TCP throughput: A simple model and its empirical validation, ACM SIGCOMM'98, 303–314, Vancouver, Sept. (1998)
31. Rhee I., Xu, L.: Limitations of equation-based congestion control. IEEE/ACM Trans. on Networking **15(4)**, 852–865 (2007)
32. Greer, D.: Building converged networks with IMS technology. IEEE Comput. **38(11)**, 14–16 (2005)
33. Haartsen, J.: The Bluetooth radio system. IEEE Personal Comms. **7(1)**, 28–36 (2000)
34. Specification of the Bluetooth System – 2.1 + EDR. Nov. (2007) Available online at http://www.bluetooth.com
35. Ferro E., Potorì, F.: Bluetooth and Wi-Fi wireless protocols: A survey and a comparison. IEEE Wireless Communications **12(1)**, 12–26 (2005)
36. Reeve, M., Bilton, C. E., Holmes, M., Bross, M.: 21CN. IEEE Comms. Eng. Oct. (2005)
37. Golmie, N., Chevrolier, N., Rebala, O.: Bluetooth and WLAN Coexistence: Challenges and solutions. IEEE Wireless Commun. **10(6)**, 22–29 (2003)
38. Valenti, M. C., Robert, M., Reed, J. H.: On the throughput of Bluetooth data transmissions. IEEE Wireless Communication and Networking Conference, 119–123, Orlando, Florida, Mar. (2002)
39. Razavi, R., Fleury, M., Ghanbari, M.: Detecting congestion within a Bluetooth piconet: Video streaming response. London Comms. Symposium, 181–184 Sept. (2006)
40. Li, Q., van der Schaar, M.: Providing QoS to layered video over wireless local area networks through real-time retry limit adaptation. IEEE Trans. on Multimed. **6(2)**, 278–290 (2004)
41. Gilbert, E. N.: Capacity of burst-noise channel. Bell System Technical J. **39**, 1253–1265 (1960)
42. Elliott, E. O.: Estimates of error rates for codes on burst noise channels. Bell System Technical J. **42**, 1977–1997 (1963)

43. Razavi, R., Fleury, M., Ghanbari, M.: An efficient packetization scheme for Bluetooth video transmission. Electron. Lett. **42(20)**, 1143–1145 (2006)
44. Razavi, R., Fleury, M., Ghanbari, M.: Fuzzy control of adaptive timeout for video streaming over a Bluetooth interconnect. 2nd mediaWin Workshop at IEEE 12th Int. Symposium on Computers and Communications, Lisbon, Portugal, July (2007)
45. Jammeh, E. A., Fleury, M., Wagner, C., Hagras, H., Ghanbari, M.: Interval type-2 fuzzy logic congestion control of video streaming. IET Intelligent Environments Conference, Seattle, July (2008)