CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION:

The title of our project is ***"PERFORMANCE ANALYSIS OF IMAGE QUALITY OF MPEG VIDEO WITH NEURO-FUZZY METHOD OVER BLUETOOTH".*** Here the MPEG video is transferred over the BLUETOOTH network with a Neuro-Fuzzy approach and the quality of the video is not degraded that is shown by a comparison of Neuro-Fuzzy and Non-Neuro-Fuzzy method.

1.2 Overview of the Project:

For the transmission over a network, digital video is compressed beforehand since it is data-hungry and requires a huge bandwidth for the extremely high bit rate. From the service providers' point of view, the Moving Picture Expert Group (MPEG) Variable Bit Rate (VBR) encoding scheme is favored to ensure a constant image quality. The level of image quality can be determined by the level of compression specified by the quantization parameter (QP) of the encoder. However, the burstiness of a VBR stream, i.e. the existence of sudden changes and big variance in its bit rate over time, will inevitably cause serious network congestion and data dropping and thus result in image quality degradation at the receiving end. To tackle such a problem, Constant Bit Rate (CBR) encoding scheme has been widely adopted in wired networks. CBR encoding scheme was first introduced to allow transmission of video signals over narrow-band integrated services digital network (ISDN). CBR uses a close loop to regulate the output bit rate from an MPEG encoder such that it conforms to a pre-set value. However, in a wireless network, due to mobility of the hosts and the interference in a radio-frequency (RF) environment, the actual transmission speed of a wireless link is unpredictable and highly variable and may become much lower than the contracted value. Subsequently, when transmitted over a wireless network a CBR encoded stream would face the same network congestion problem and suffer image degradation, just like VBR streams. This paper presents a control scheme developed specially for video transmission over wireless networks, which regulates the output bit rate from the MPEG encoder according to the current condition of the wireless channel. Fuzzy logic exploits the pervasive imprecision, uncertainty and partial truth of the real world using simple linguistic statements and thereby achieves tractability, robustness, and low solution cost. There have been comprehensive and successful studies into the application of fuzzy logic to many traffic control problems in wireless networks. In this research, the control procedure is carried out at the Host Controller Interface (HCI) of the wireless host at the sending end. The control scheme takes the current condition of the wireless channel into consideration. However, the idea of rate control according to the current link situation may be applied to any wireless networks.

The project is organized as follows. First section describes the structure of the Bluetooth model for video transmission and the configuration of my simulation system. Second section outlines the structure and development of the fuzzy control scheme. Next section explains and compares the Matlab-Simulink computer simulation results for the real-time MPEG video clips. Last section will be the conclusion and the comparison of the results of the proposed scheme with the conventional VBR and CBR video schemes.

1.3 EXISTING SYSTEM

In the existing system, the problem with the image quality degradation due to several deficiencies and old algorithm is solved and developed. In that existing system there is no application of Neuro-Fuzzy algorithm or the Rule Based system. The encoded MPEG video is directly sent to the air through a buffer.

The below figure illustrates the existing system.

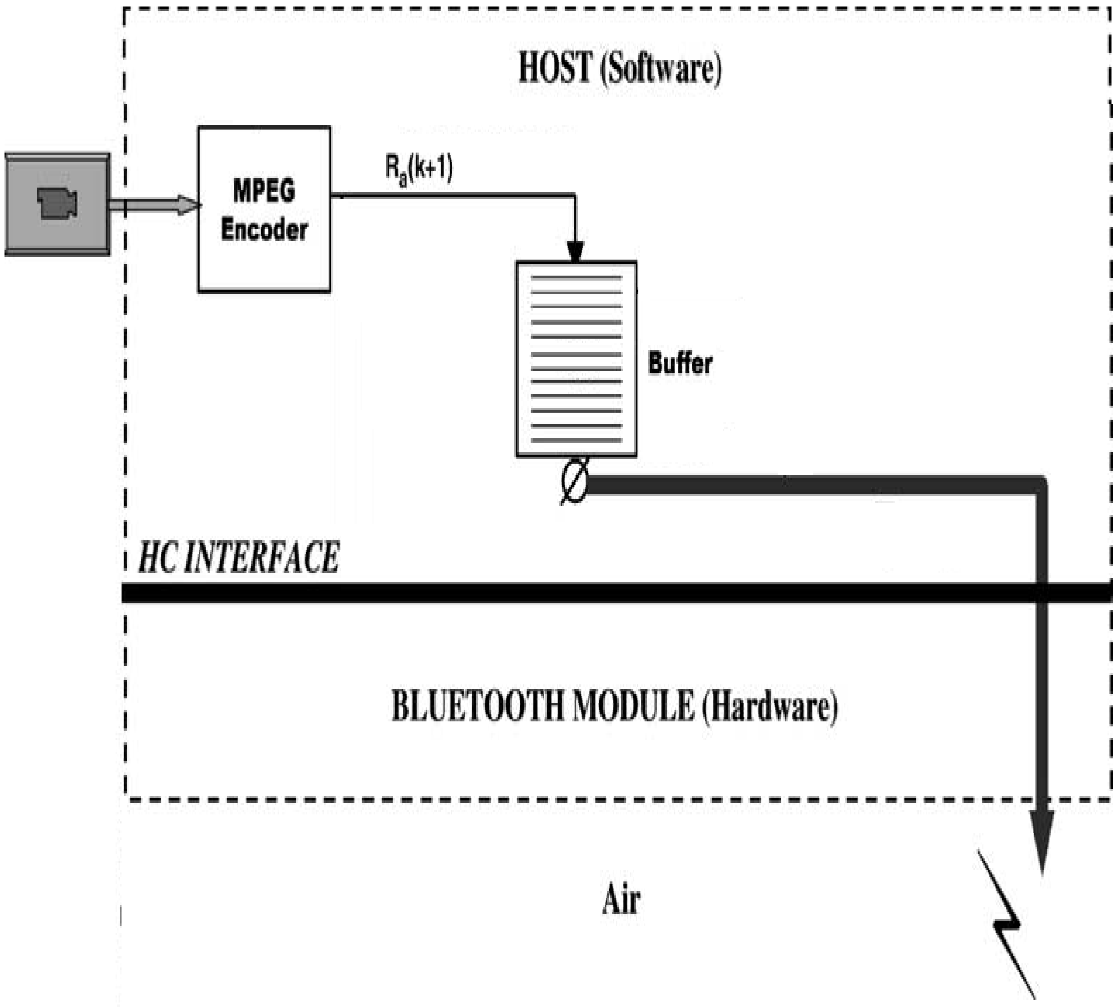


Fig.1.3.1 Existing System

1.3.1 Frame size of the MPEG video:

The frame size is not set properly before it is delivered over BLUETOOTH network. The bigger frame size causes the degradation of the image quality when it is received on the other end or the intended receiver. This causes the variation in the image of the MPEG video.

1.3.2 UNRELIABLE UNPREDICTABLE TRANSMISSION RATE:

In a Bluetooth network, transmission rate is unpredictable due to interferences by other wireless devices or general Bluetooth channel noises. MPEG Variable Bit Rate (VBR) video transmission is also unreliable and presents long delay and excessive data loss, due to variations in bit rate. It is therefore almost impossible to transmit MPEG VBR video over a Bluetooth channel, without data loss, excessive time delay or image quality degradation.

1.4 Proposed system:

The different deficiencies are overcome in the proposed model in this project. The proposed model defines some algorithm to obtain better quality when the MPEG video is transmitted over BLUETOOTH network.

1.4.1 Altering the Frame size:

The frame size of the video can be altered so that the range of GOP also varies for some desired inputs. A GOP has 12 frames and there are two GOPs in one second. If this ratio is altered on the available to the leaky bucket, also called Generic Cell Rate Algorithm (GCRA), then a better video quality will be available.

Advantages:

A video clip is used to implement the proposed NF control scheme and the first 350 GOPs of the clip will be utilized for the computer simulation. The frame size of the clip will be 240 (width) x 180 (height) pixels. In MPEG the GOP is a group of pictures or frames between successive I- (Intra) frames, and the others being P- (Predicted) and/or B- (Bi-directional) frames. I- holds data to construct a whole picture, P- considers at the difference between the present and the previous frames, and B- measures the difference between the previous and the next frames. Quantization parameters therefore can be determined independently for each frame. High quantization parameters mean inferior quality of picture. If the Group of Pictures is altered in the project then the change will be obvious and the image quality with be far better than the existing system. The image quality will not be degraded in this way.

After setting all the parameters properly a comparison is shown in the MATLAB simulation to have a look of the performance of the proposed one. The parameters are be taken is such a way that the video image data will be kept unchanged.

1.4.2. Determining the Departure Rate:

The Standard Deviation of the output bit rate and the number of dropped data is varied through the novel NF scheme by correspondingly altering the departure bit rate between the arrival bit rate and actual token rate.

Bluetooth uses polling-based packet transmission. All communication between devices takes place between a master and a slave, using time-division duplex (TDD), with no direct slave-to-slave communication. The master will poll each active slave to determine if it has data to transmit. The slave may only transmit data when it has been polled. Also, it must send its data in the time slot immediately following the one in which it was polled. The master transmits only in even numbered time slots, while the slaves transmit only in odd-numbered time slots.

Taking care of the above information my project undergoes in such a way that the image data, which is transmitted over the BLUETOOTH, is sent in a well fashioned manner with the help of the GCRA. First the available bandwidth is measured and depending on the available of the space of the 'token bucket' the data frames is transmitted to get a continuous flow of the video image on the receiver end.

1.4.3. Developing Traffic Shaping Buffer:

A traffic shaping buffer is introduced after the MPEG encoder, which is used to store the frames coming from the encoder temporarily before it is sent to Rule Based System. This buffer regulates or confirms that neither the discharge of the packets is overloaded or empty.

1.4.4. Developing Rule Based System:

The Rule-Based system is an application of Fuzzy logic. This system actually takes care of the packet of the Traffic Shaping Buffer. This function takes the frames or the packet as an input and discharge continuous flow of packet of packets or frames.

1.4.5. Introducing Leaky Bucket:

Leaky bucket is developed to regulate the frames just before the discharge to the air or Bluetooth. The leaky-bucket is located prior to the HCI and measures the departure rate against the contracted mean rate. The architecture and functionality is described later in the detailed section.

CHAPTER 2

Technical Description of the Bluetooth Technology

In 1994 Ericsson Mobile Communications AB in Lund, Sweden, initiated a study of a new radio communication technology, today known as the Bluetooth technology. The technology was from the beginning intended to be a low-power and cheap radio interface between mobile phones and mobile accessories. The main idea was to replace cables. As the work progressed, the people involved started to realize the true potential of this new technology. Not only could it be used as a simple cable replacement, it could also, with the ability to connect multiples of units, be used in an ad-hoc manner. Ericsson understood that to make something out of this they would need help. Therefore, four other companies were invited to cooperate and promote the Bluetooth technology. Ericsson together with IBM, Intel, Nokia and Toshiba created the Special Interest Group (SIG). These five promoters envisioned Bluetooth to be an open, global standard. Later, four more promoters joined the SIG, Lucent, 3Com, Motorola and Microsoft. Besides the promoters several adopters have signed the SIG adopters' agreement. Today there are about 2500 adopters/associates.

2.1 DATA FLOW THROUGH THE BLUETOOTH PROTOCOL STACK

2.1.1 BLUETOOTH ARCHITECTURE:

Bluetooth provides two types of links-Synchronous Connection Oriented (SCO) link and Asynchronous Connection-Less (ACL) link. The SCO link uses a slot reservation mechanism to guarantee the periodic transmission of data and is typically suitable to carry real-time traffic such as voice and video. However, the bandwidth of the SCO link is at most 288 kb/s. The ACL link may support up to 723.2 kb/s downlink using DH5 packets with 57.6 kb/s uplink or symmetrically 433.9 kb/s in both directions. However, the ACL link is typically suitable to support data applications rather than real-time video traffic.

Voice/Video over IP (VoIP) is a set of technologies that enables real-time voice/video to be sent over a data network. Studies have been conducted on the protocol concepts of mobile IP and Cellular IP so as to provide Internet services over wireless networks such as Bluetooth. Since VoIP protocol sends voice or video as IP packets, it is regarded as data and therefore enables the use of ACL link when transmitting video over a Bluetooth channel.

Fig. 2.1.1 represents the Bluetooth protocol stack and the video data flow through it. The fuzzy scheme controls the flow of the video data sources at the Host Controller Interface (HCI). In this Fig, above the HCI is the software running on a HOST processor and below the HCI is the Bluetooth module—the hardware. Again in Fig. through the VoIP layer, the compressed video is sent as IP packets to Bluetooth Network Encapsulation (BNEP) layer.

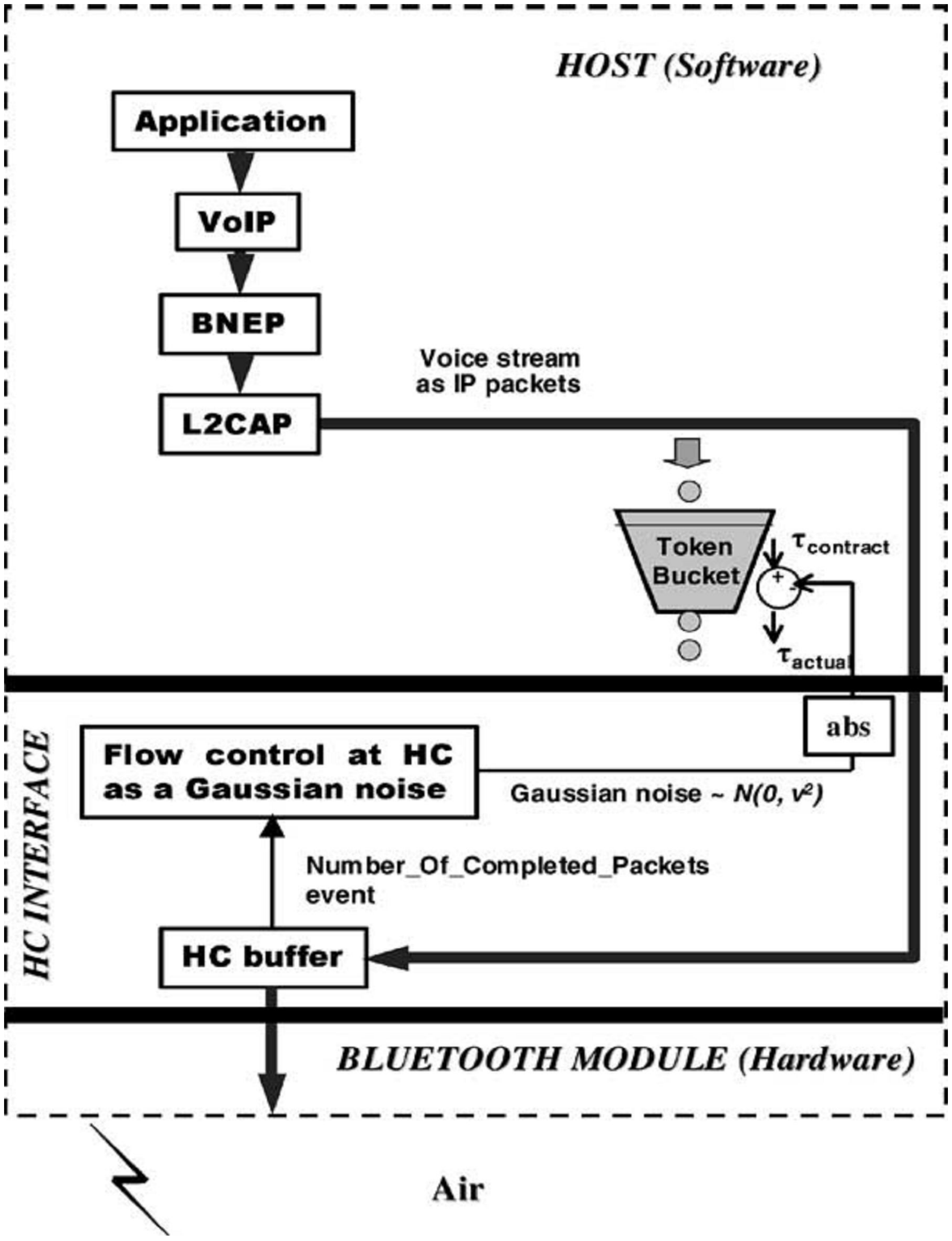

HOST (Software)
Application
VoIP
BNEP
L2CAP
Voice stream as IP packets
Token Bucket
$\tau_{contract}$
+
-
τ_{actual}
abs
HC INTERFACE
Flow control at HC as a Gaussian noise
Gaussian noise ~ $N(0, v^2)$
Number_Of_Completed_Packets event
HC buffer
BLUETOOTH MODULE (Hardware)
Air

Fig. 2.1.1The dataflow through the Bluetooth protocol stack

The BNEP protocol emulates an Ethernet type of broadcast medium on top of Bluetooth, to encapsulate IP packets. Through the BNEP layer, VoIP packets are sent to Logical Link Control and Adaptation Protocol (L2CAP) layer and transformed into L2CA packets there. The three lower layers of the Bluetooth protocol stack-Link Manager, Baseband and Radio-are typically integrated into a single chip called Bluetooth module. The HCI specification defines a standard interface independent method of communicating with the Bluetooth module. The higher layer software stack on the host processor communicates with the Bluetooth hardware using HCI commands.

2.1.2 DESCRIPTION OF QOS AND HCI

2.1.2.1. QUALITY OF SERVICE (QOS)

The Bluetooth specification provides QoS configuration in the traffic contract to allow the properties of links to be configured according to the requirements of higher layer applications or protocols. The task of transmitting video streams is bandwidth demanding and time critical. Keep in mind that wireless links are subject to interference and thereby cannot guarantee QoS. A Bluetooth channel can only be configured so that it does not compromise QoS by accepting more traffic than it can handle. Even if Bluetooth could guarantee the contracted bandwidth, network congestion would occur when traffic sources exceed their contracted rates.

To avoid QoS degradation caused by misbehaving sources, Bluetooth carries out rate policing by means of the token bucket algorithm. Each packet arriving at the ingress point generates a token, which is then passed to the token bucket. Token bucket leaks tokens at a set rate called token rate. Token rate equals the rate at which the data may be sent on a link, i.e. the transmission rate of a link. If a traffic source exceeds its contracted rate, the token bucket would overflow. And when this happens, any incoming packet would be marked as non-conforming and then dropped. The bandwidth of 723.2 kb/s mentioned earlier is for the maximum amount of data that a Bluetooth ACL link can transfer on air per second. Higher protocol layers such as L2CAP layer and VoIP layer (Fig. 1) will use some of the channels capacity with headers and framing information. At the application level, the maximum data rate is around 650 kb/s. The token rate is taken in all of my experiments to be 650 kb/s in total with 32 kb/s for sound and 618 kb/s for video images. Bursty data sources such as video streams will most likely need more storage than sources that flow evenly. However a token bucket with a large buffer will lead to long potential delay. Considering this, a bucket size of 10 kb will be used here. With a token rate of 650 kb/s it gives rise to a maximum delay of 123 ms, which amounts to the duration of less than three video frames. Frame rate in our experiments is 24 frame/s.

2.1.2.2. INTERFERENCE AND FLOW CONTROL OF THE HCI

As stated, the Bluetooth HCI transport layer passes HCI packets from the host to the Bluetooth module. Some HCI transport interfaces offer data rates much higher than the data rate across the Bluetooth radio and air interface. To handle the mismatch between the data rates across the transport layer and on air, the Bluetooth specification provides flow control of the HCI such that data transport across the HCI slows down when the host controller (HC) buffer (Fig. 1) is overloaded while runs at full speed the rest of time. The host can only send data when there is space available in the HC buffer. That is to say although a Bluetooth link may have been set up with a certain token rate, the actual data rate at which the host sends data varies according the space left in the HC buffer at that time instance; while the space left in the HC buffer is determined by the current situation of the Bluetooth link. As stated, the quality of a link can be expected to vary for many reasons: the mobile devices can move around and vary affecting signal strength; objects can get in the way of the link and absorb power; and sources of radio interference can reduce the channel quality. Moreover, losses due to retransmission, signaling packets interfering with data flow, and the capabilities of the device at the far end of the Bluetooth link will also affect the data rate. Effect of different interference sources on the performance of wireless networks has been investigated before.

To reflect interference in the simulation system, a Gaussian noise representing the cumulative interference will be introduced to the contracted token rate to simulate the actual data rate at which the host sends IP packets to the Bluetooth module.

τ_{actual} and $\tau_{contract}$ denote the actual token rate and the contracted token rate, respectively. The “abs”—absolute value—block in the system is used to make sure the actual token rate is always lower than the contracted value. With the “number of completed packets” event, the Baseband layer in the Bluetooth module informs the L2CAP layer in the host about the number of data packets that have completed transmission. That is either successfully transmitted or being flushed by the HC buffer.

2.2 Architecture for Video Streaming over Bluetooth

Traditional video streaming over wired/wireless networks typically has band-width, delay and loss requirements due to its real-time nature. More over, there are many potential reasons including time-varying features, out-of-range devices, and interference with other devices or external sources that make Bluetooth links more challenging for video streaming. To address these challenges for video streaming over Bluetooth links, recent research has been conducted. To present various issues and give a clear picture of the field of video streaming over Bluetooth, three major areas are discussed, namely video compression, QoS control and intermediate protocols. Each of the areas is one of the basic components in building a complete architecture for streaming video over Bluetooth. The relations among them can be illustrated in Fig.2.2.1

Figure 2.2.1 shows functional components for video streaming over Bluetooth links. Moreover, the layer/layers over which a component works is also indicated. The aim of video compression is to remove redundant information form a digitized video sequence. Raw data must be compressed before transmission to achieve efficiency. This is critical for wireless video streaming since the bandwidth of wireless links is limited to 732Kbps. Upon the client’s request, the media sever retrieves compressed video and the QoS control modules adapts the media bit-streams, or adjusts transmission parameters of intermediate layer based on the current link status and QoS requirements. After the adaptation, compressed video stream are partitioned into packets of the chosen intermediate layer (e.g., L2CAP, HCI, IP), where packets are packetized and segmented. It then sends the segmented packets to Bluetooth module for transmission. On the receiving side, the Bluetooth module receives media packets from air, reassembles them in the intermediate protocols, and sends them to decoder for decompression.

As shown in figure 2.1.1, QoS control can be further categorized into congestion control and error control. Congestion control in Bluetooth is employed to prevent packet loss and reduce delay by regulating transmission rate or reserving bandwidth according to changing link status and QoS requirements. Error control, on the other hand, is to improve video quality in the presence of packet loss.

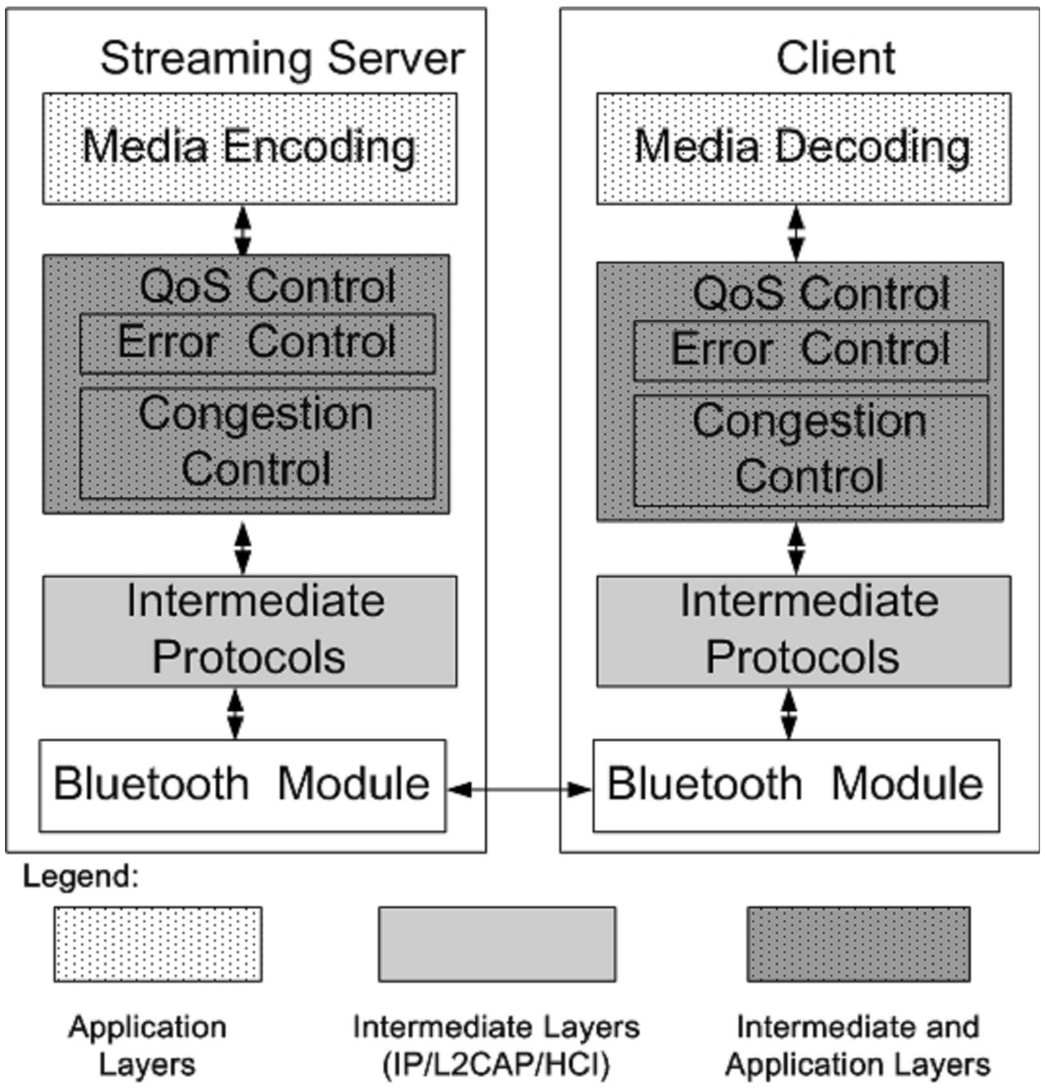


Figure 2.2.1 Architecture for streaming over Bluetooth

2.2.1 Intermediate Protocols

Bluetooth protocols deal with physical and link layer operations, while also provide flexible interfaces for interacting with upper layers. Among multiple layers in the Bluetooth protocol stack, it is very important to choose a proper intermediate protocol for packetizing and segmenting media streams. M. H. Chia and M. Salim Beg proposed and compared HCI, L2CAP and IP as alternative intermediate protocols for video streaming over Bluetooth.

The proposed implementations are named as MPEG-4 over Bluetooth (MPEG4BT) via HCI, MPEG4BT via L2CAP, and MPEG4BT via IP over Bluetooth. A qualitative comparison of the three intermediate layers is made based on the size of the overheads, the efficiency of segmentation and reassembly processes, and hardware compatibility. Implementation issues of streaming video via different layers over Bluetooth. It is suggested that video streaming via IP and L2CAP can be achieved using three Bluetooth specifications – Local Area Network Access Point Profile (LAP[), Bluetooth Network Encapsulation Specification (BNEP) and Audio/Video Distribution Transport Protocol (AVDTP).

2.2.2 Streaming via HCI

For streaming via HCI, video bit-streams are directly packetized into HCI packets to send down to baseband for transmitting over air. This approach can maximize the bandwidth usage and minimize overhead by exposing the internal operation to lower layer. In HCI layer, the size of packets depends on the buffer size. Since HCI does not have segmentation and reassembly (*SAR*) function, application-layer software need to perform SAR based on HCI buffer size. This loads the host system resources and therefore impacts the overall performance, as MPEG-4 compression needs very high processing power. Another disadvantage is that using HCI without L2CAP is not allowed in current Bluetooth protocol, which asks for the change of hardware and software specification.

2.2.3 Streaming via L2CAP

Streaming via L2CAP hides the peculiarities of the Bluetooth low layers, thus making it possible for existing applications to run over Bluetooth links without too much modification. L2CAP can facilitate the SAR of larger-size, higher-layer packets to and from the smaller baseband packets. Nevertheless, this method produces more overheads compared to HCI because of extra bits needed for L2CAP packet encapsulation. Bluetooth streaming via L2CAP is defined by three draft specifications covering the protocols and profiles: Audio/Video Distribution Transport Protocol (AVDTP), Audio Video Control Transport Protocol (AVCTP) and Generic Audio/Video Distribution Profile (GAVDP). Following these A/V specifications, a sender could stream Real Time Protocol (RTP) packets to a receiver across L2CAP channels at high quality independent from video codec's. In AVDTP specification, it is suggested that L2CAP channels are best suited for the support of A/V stream data distribution links, because L2CAP can be flexibly configured to enable bandwidth to be shared between multiple A/V content streams. AVDTP applies point-to-point signaling over a pseudo-isochronous, connection oriented L2CAP ACL channel. Both A/V streams and signaling messages are transported via the same physical L2CAP channels. AVDTP Signaling provides stream discovery, configuration, establishment, and transfer control. When A/V applications transport audio and/or video streams over Bluetooth links, AVDTP performs A/V parameter negotiation. Based on the result of this negotiation, A/V applications transport audio and/or video content.

2.2.4 Streaming via IP:

Streaming via IP over Bluetooth relies on the bridging of TCP/IP and Bluetooth. Its main advantage is that IP-based video streaming mechanisms such as RTP can be transparently used without modification. The method is actually a duplication of the functionality provided directly by IP-based video streaming. However, this convenience is achieved at the cost of overhead added by upper layers, which may penalize performance due to the limited bandwidth provided by Bluetooth. Bluetooth streaming via IP can be achieved by using LAP or BNEP:

LAP defines general procedures to set up a Point-to-Point Protocol (PPP) link over RFCOMM and thus allow IP packets to flow across the link. The profile specifies two components on being a client accessing the services of a network by the means of using a gateway known as a Bluetooth access point. This is normally hard wired to the network.

BNEP describes the process to send TCP or UDP over L2CAP. Following BNEP, the packets traverse each TCP/IP stack layer and in turn a header is added onto the original packet , then it is channeled through the Ethernet Frame layer and finally through L2CAP. The application on the receiving side processes the packet and converts it back into video streams. Compared to streaming via L2CAP, LAP and BNEP enables streaming via IP over Bluetooth by processing upper layer packet headers across L2CAP links. As the video stream is packetized in IP layer, it adds an additional encapsulation layer between the L2CAP encapsulation layer, and the encapsulation provided by IP, such that there are at least three encapsulation layers for an IP Bluetooth solution: L2CAP, BNEP /LAP, and IP.

CHAPTER 3

THE FUZZY CONTROL SCHEME

To avoid network congestion and hence reduce data dropping at the HCI, a fuzzy control scheme will be introduced at the HCI. The control scheme involves two rule-based fuzzy (RBF) controllers and a traffic-shaping buffer. The traffic-shaping buffer, as a temporary storage, will bring more flexibility into the system. The two RBF controllers both will be based on Mamdanis min implication function. One RBF controller I will monitor the output rate from the traffic shaping buffer according to the number of tokens available and the occupancy of the shaping buffer so that the output rate conforms to the current conditions of the Bluetooth channel. And the other RBF controller II regulates the data rate of the encoded video stream so as to avoid a sudden burst of data by applying a close-loop encoding scheme. This control scheme will be successfully applied to video transmission over BLUETOOTH networks. Therefore, the rules and fuzzy sets are mainly based on the experience gained from the fuzzy logic application to video transmission in BLUETOOTH, and indeed from trial and error methods, taking into account the capacity of the traffic-shaping buffer, the Bluetooth bandwidth and the Bluetooth channel noises.

3.1 Fuzzy Logic Application for MPEG Encoding:

3.1.1 Introduction:

Fuzzy logic and fuzzy set theory have been extensively used for industrial and commercial control applications since its introduction. Recently, fuzzy rule-based control (FRC) has been applied to discrete cosine transform (DCT)-based video sequence coding algorithms such as PEG and H.261 by adaptively controlling the MPEG quantizer and the buffer occupancy. The first step in designing FRC system requires transforming expert knowledge into a set of rules comprising linguistic expressions so that a decision can be made on the desired output value. **A** fuzzy set is expressed in a non-numerical form carrying linguistic meanings, e.g. very big, big, small or very small. These FRC techniques thus follow cascaded processes; fuzzification, decision making and defuzzification.

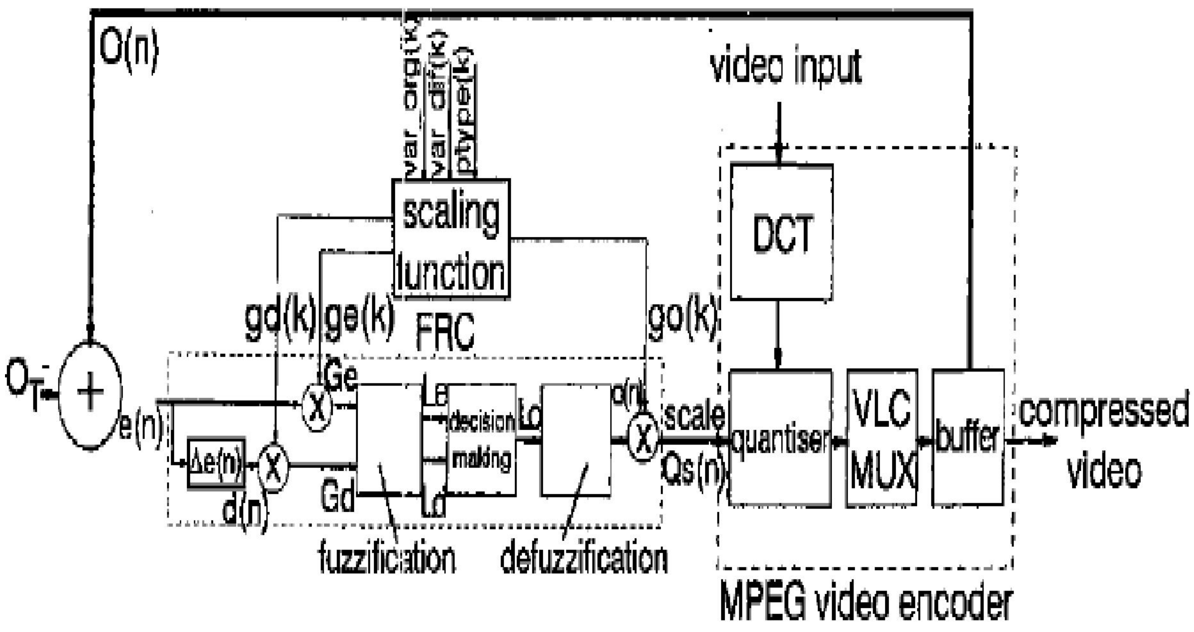


Fig. 3.1.1 Configuration of FRC bused MPEG video encoder

3.1.2 Enhanced fuzzy rule-based video rate control:

To achieve accurate estimates of incoming video rate we have identified the need to use additional information. Scene change feature provide vital information about the complexity of an incoming picture in advance of the encoding operation. Here, they are incorporated with scaling factors *(ge, gd* and *go)* to adaptively scale the inputs to the fuzzification process *e(n)* and *d(n)* and the output (Fig. 3.1.1). Three scene change features, *var-org(k), vur-dzxk)* and *ptype(k),* are supplied to the scaling factor calculation block. *vurorg(k)* and *var-dgk)* are variances of the input picture and the difference picture, between the current and the previous frames. *ptype(k)* is the picture type information which has an integer value for the picture frame type, i.e. I, P and B. The time index *k* corresponds to the frame number.

The MPEG encoder quantizes a discrete cosine transformed video and inputs this into a variable (run) length coder multiplexer (VLC MUX). The scale control, *Qs(n),* sets the MPEG encoder quantization step size. 'The FRC time index, *n,* is incremented once for each 16 x 16 video macro block, to provide the required quantizer update rate. The buffer occupancy ***O(n)*** is provided as a second output from the encoder, in addition to the compressed video sequence. The error signal *e(n)* which is the difference between 0, (target occupancy) and ***qn),*** forms the input to the FRC process. The process begins by calculating the macro block error value *e(n)* and the differential (error value *d(n)* which is the difference between the current error value *e(n)* and the previous macro block error value *e(n* - 1). The entire process of generating the FRC output proceeds with the two inputs, *Ge* and *Gd,* which are translated into linguistic expression ***Le*** and *Ld.* In the decision making process, a linguistic judgment ***Lo*** is determined, based on a predetermined set of rules. The defuzzification process calculates *o(n)* by combining the membership function of *Lo* and those of ***Le*** and *Ld* in a set theoretic way; *o(n)* is then scaled to yield the updated video quantizer scale value *Qs(n).*

3.2 RBF Neural Networks

A Radial Basis Function (RBF) neural network has an input layer, a hidden layer and an output layer. The neurons in the hidden layer contain Gaussian transfer functions whose outputs are inversely proportional to the distance from the center of the neuron.

3.2.1 How RBF networks work:

Although the implementation is very different, RBF neural networks are conceptually similar to K-Nearest Neighbor (k-NN) models. The basic idea is that a predicted target value of an item is likely to be about the same as other items that have close values of the predictor variables.

An RBF network positions one or more RBF neurons in the space described by the predictor variables (x,y in this example). This space has as many dimensions as there are predictor variables. The Euclidean distance is computed from the point being evaluated (e.g., the triangle in this figure) to the center of each neuron, and a radial basis function (RBF) (also called a kernel function) is applied to the distance to compute the weight (influence) for each neuron. The radial basis function is so named because the radius distance is the argument to the function.

$$\boldsymbol{\mathit{Weight}} = \mathbf{RBF}(\boldsymbol{\mathit{distance}})$$

The further a neuron is from the point being evaluated, the less influence it has.

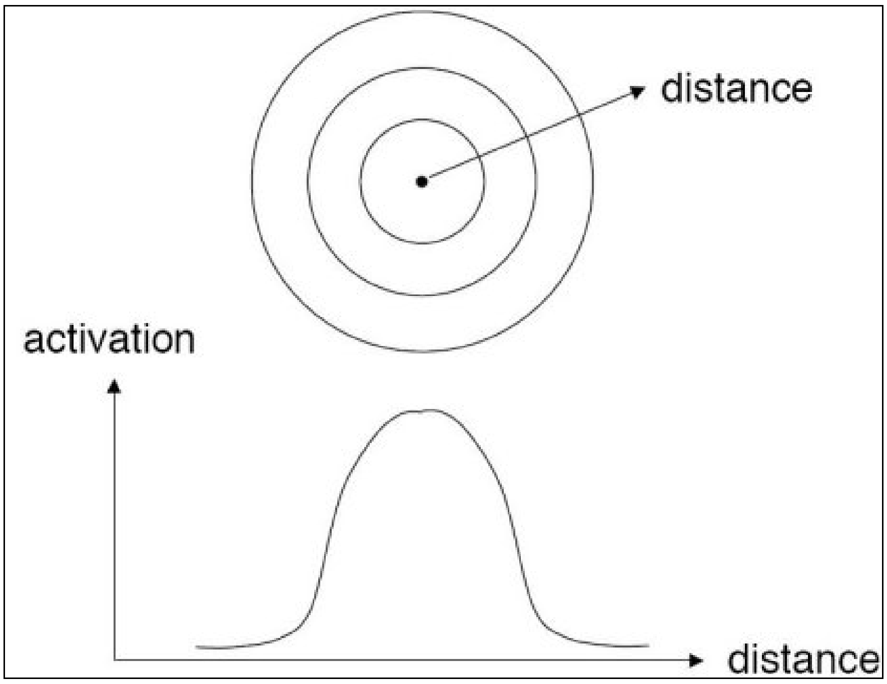


Fig 3.2.1.1 Activation vs Distance Graph

3.2.2 Radial Basis Function:

Different types of radial basis functions could be used, but the most common is the Gaussian function:

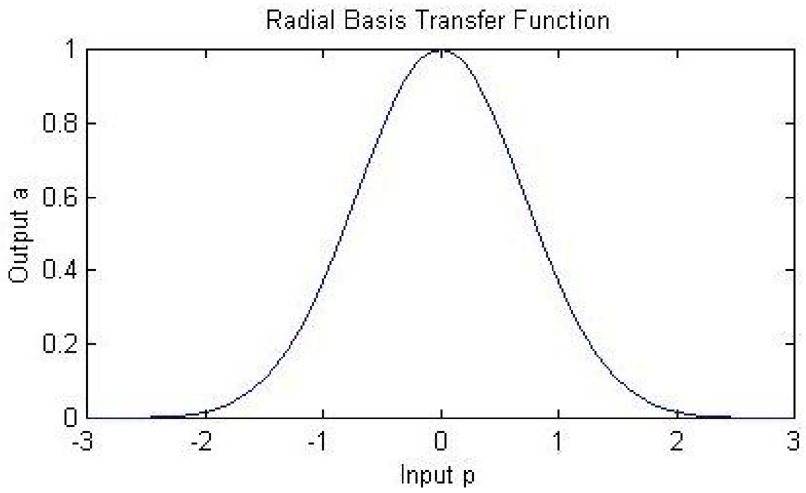


Fig. 3.2.2.2 Input b vs Output a

If there is more than one predictor variable, then the RBF function has as many dimensions as there are variables. The following picture illustrates three neurons in a space with two predictor variables, *X* and *Y*. *Z* is the value coming out of the RBF functions:

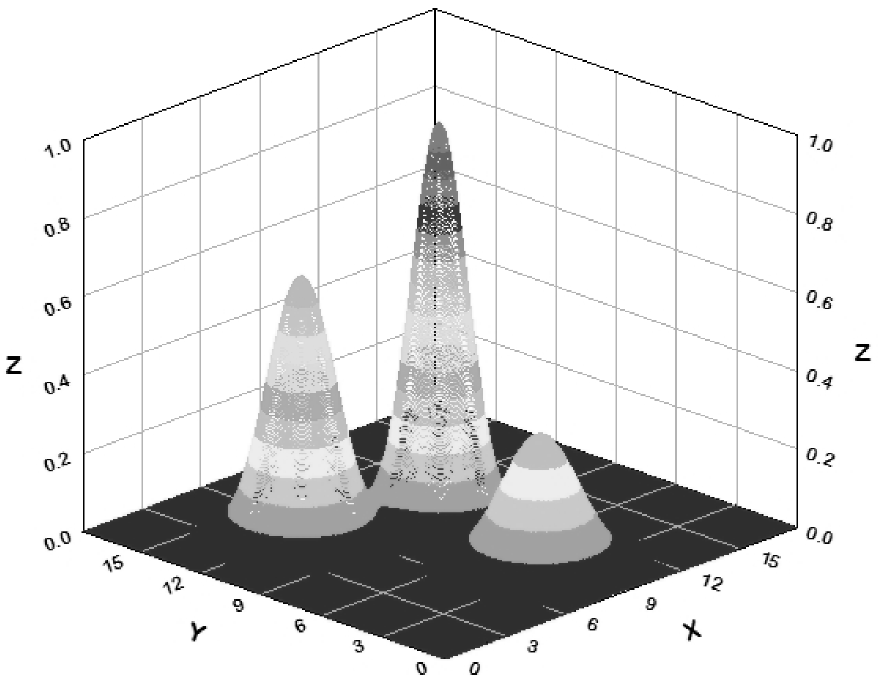


Fig. 3.2.2.3   3D figure illustrating X, Y, Z

The best predicted value for the new point is found by summing the output values of the RBF functions multiplied by weights computed for each neuron.

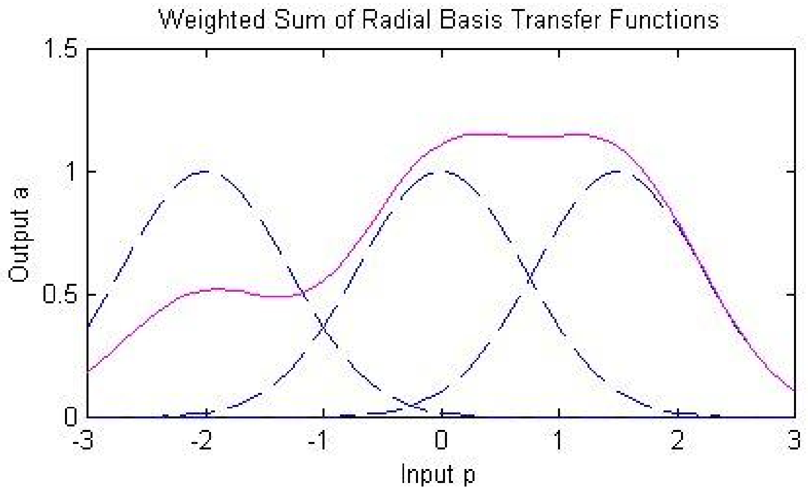


Fig. 3.2.2.4 Weighted Sum of Radial Transfer Functions

The radial basis function for a neuron has a center and a radius (also called a spread). The radius may be different for each neuron, and, in RBF networks generated by DTREG, the radius may be different in each dimension.

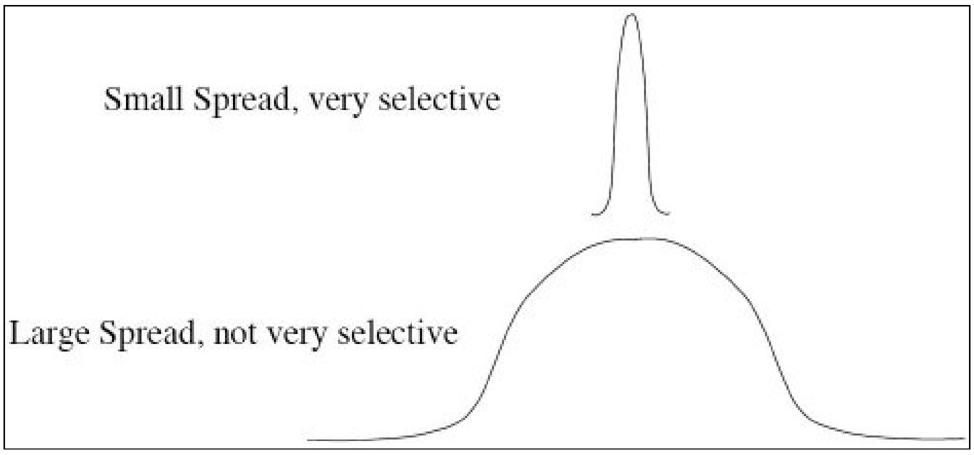

Small Spread, very selective
Large Spread, not very selective

Fig. 3.2.2.5 Figure illustrating Spreading

With larger spread, neurons at a distance from a point have a greater influence.

3.3 Application of a fuzzy rule based system:

I have applied a fuzzy rule based system to describe the behavior (the lengthening) of a sample of Cu-Zn-Al under the effects of changes of stress. In general, these systems can approximate arbitrary functions by means of a set of fuzzy rules of the following form:

R-i: If X_1 is A_1,s(1,i) and... and X_n is A_n,s(n,i) then Y is Ys(i)

Fig. 3.3.1 outlines the whole control scheme. At any time instant, the kth GOP in the model is the group that has completely entered the network; the (k + 1)th GOP is the picture group currently passing through the model so as to reach the token bucket and enter the network; and the (k + 2)th GOP is the group being encoded by the MPEG encoder.

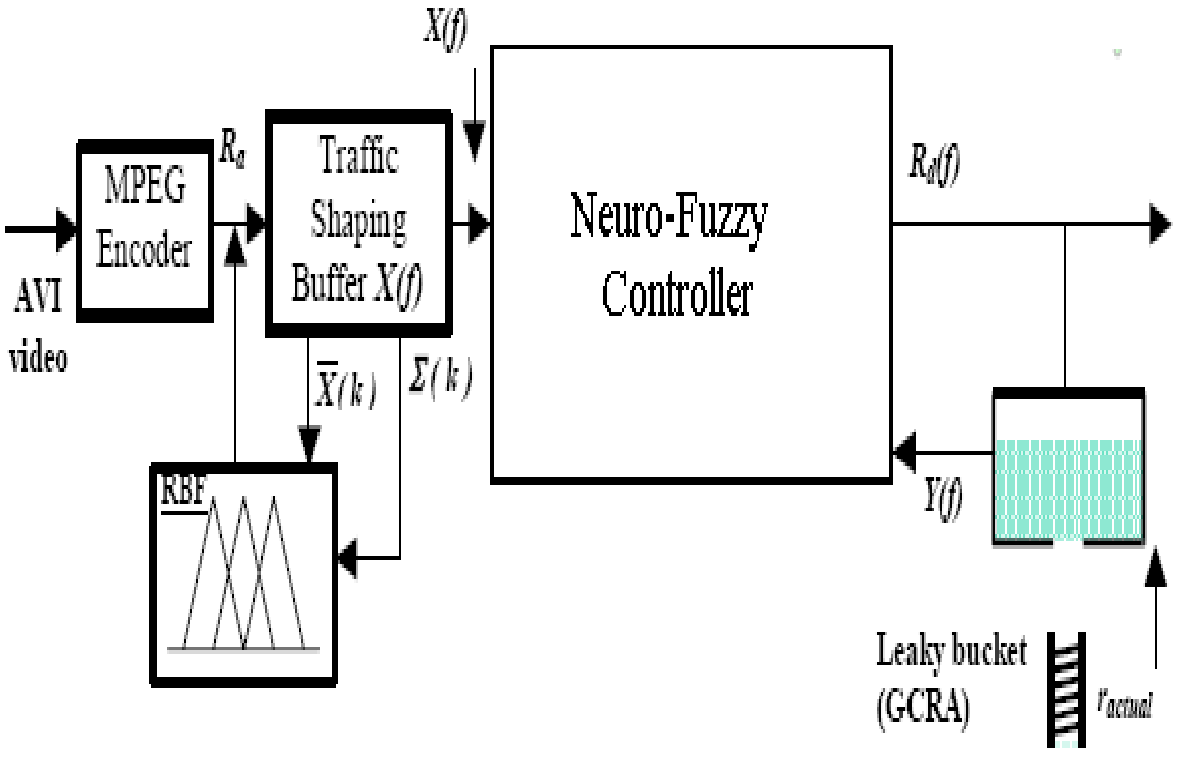


Fig.3.3.1The fuzzy control scheme for video transmission over a Bluetooth ACL link.

RBF controller I takes the queue length in the traffic-shaping buffer X(f) and the available memory space in the token bucket Y(f) as inputs and generates the cell inter-departure time from the traffic-shaping buffer id(f). f is denoted as a frame. The departure rate $R_d(f)$ will be calculated from id(f) using the following equation:

$$R_d(f) = 1/[i_d(f) \times (I_{d\_\max} - I_{d\_\min}) + I_{d\_\min}], \qquad \text{Eq}\ldots\ldots\ldots\ldots\ldots\ldots(3.3.1)$$

where $I_{d\_\min} = \min\{1/R_a(k+1), 1/r_{\text{actual}}\}$ and $I_{d\_\max} = \max\{1/R_a(k+1), 1/r_{\text{actual}}\}$.

By keeping R_d between R_a and r_{actual}, we ensure that there will be always video stream flowing through the system as long as $R_a \neq 0$.

$Y(f)$ and $X(f)$ are calculated using Eqs. (2) and (3), respectively

$$Y(f) = 1/b \times \sum_{f_i=1}^{f_i=f-1} [r_{\text{actual}} - R_d(f_i)] \times T_f, \quad Y(1) = 1, \quad 0 < Y(f) \leqslant 1 \qquad \text{Eq}\ldots\ldots\ldots\ldots\ldots\ldots(3.3.2)$$

$$X(f) = 1/K \times \sum_{k_i=0, f_i=1}^{k_i=k, f_i=f-1} [R_a(k_i+1) - R_d(f_i)] \times T_f, \qquad \text{Eq}\ldots\ldots\ldots\ldots\ldots\ldots(3.3.3)$$

$$X(1) = 0, \quad 0 < X(f) \leqslant 1,$$

Where b is the buffer size of the token bucket, K is the buffer size of the traffic shaper, and T_f is the time period of a video frame. It is assumed that the traffic shaper is empty and the token bucket is full at the beginning of a video transmission.

CHAPTER 4

DESCRIPTION OF THE PROJECT MODULES

4. PROJECT MODULE

The project will be divided into several into several phases mentioned below:

1. MPEG Encoder
2. Traffic Shaping Buffer
3. Rule-based-fuzzy implementation.
4. Neuro-Fuzzy implementation
5. Leaky Bucket(GCRA)

Descriptions of the above modules are as follows:

Complete Architecture Describing the Modules of this Project:

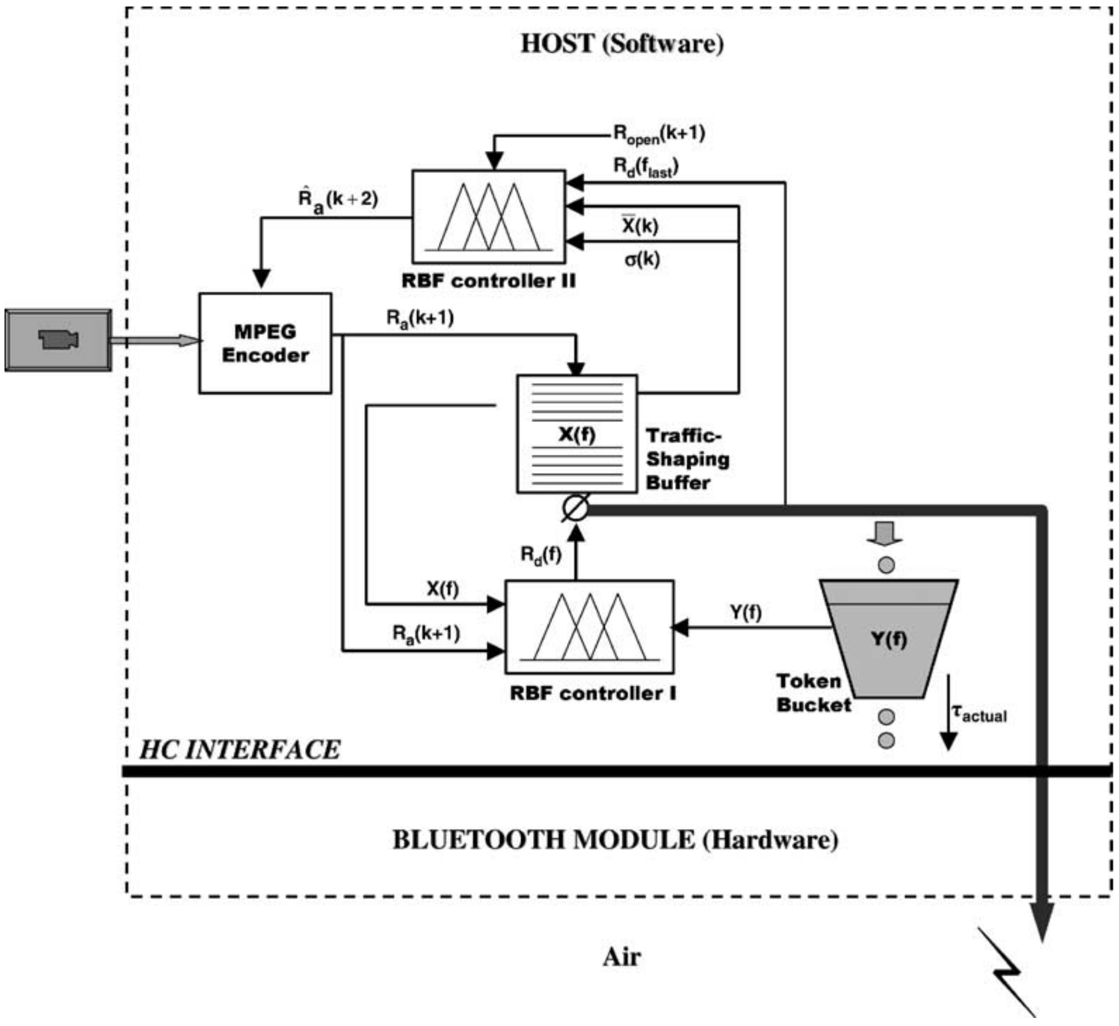


Fig. 4.1 Architecture of the Project Model

4.1 MPEG Encoder:

Basic Function:

- AVI/MPEG video will be given to MPEG Encoder to encode the information before it is sent to Traffic-Shaping Buffer.
- The level of image quality can be determined by the level of compression specified by the quantization parameter (QP) of the encoder.

4.1.1 General description:

The MPEG-4 encoder is a hardware module optimized for FPGA technologies, making use of a limited number of logic resources and being able to encode a 4CIF (704x576) sequence in real time. It is fully compliant with the Video part of ISO/IEC 14496-2. All visual tools of the Simple Profile are implemented, including full support of I-VOP (intra-coded frames, without motion estimation) and P-VOP (predictive-coded frame, with motion estimation on previously encoded frame). The core is a good compromise between coding efficiency (resulting in lower bit rate for same quality), logic complexity and coding throughput, thanks to the use of an efficient motion estimation algorithm (directional search) leading to fast and precise matching, at half pixel resolution, of blocks between the current frame and its reference. Supported image resolutions include pre-defined levels Level1 to Level5 (QCIF/CIF/VGA/SDTV) and custom definitions up to 4CIF (704x576). The core can be customized to provide support for even larger resolutions, such as HD format. The core features a Variable Bit Rate mode (VBR mode: fixed, user-specified quality). It can optionally also provide Constant Bit Rates (CBR mode: with regulation of the output bit rate) by using an external small microprocessor running a rate allocation algorithm (such as Nios or Microblaze). When used in CBR mode, the core delivers high-quality regulation thanks to its patented rate allocation algorithm making use of statistical information available at the motion estimation engine.

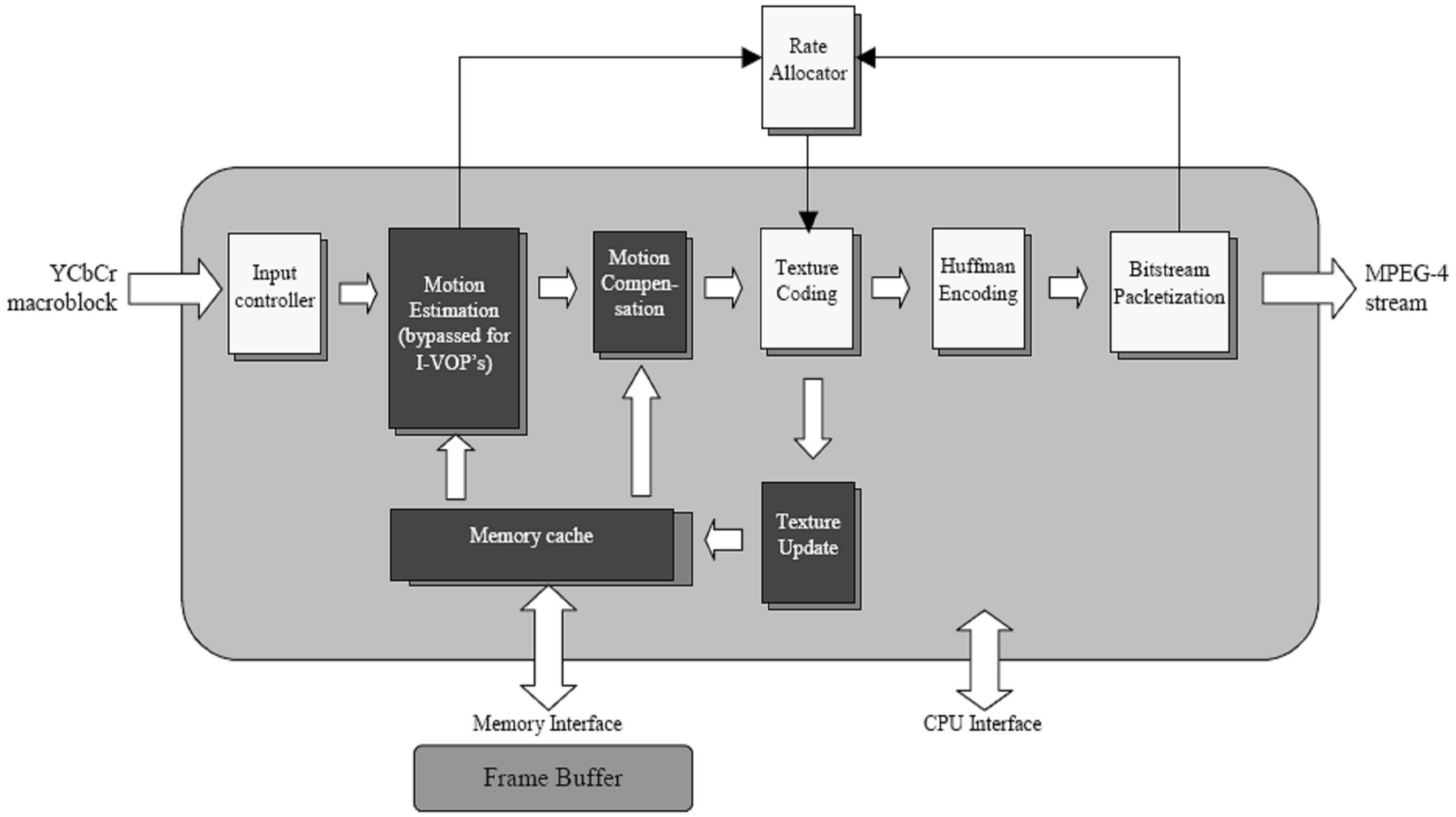


Fig 4.1.1 Block Diagram of the BA131MPEG4E IP

4.1.2 Applications:

• Video broadcast

• Security and Surveillance

• Multimedia streaming over TCP/IP

• Mobile communications

4.1.3 Technical description:

Figure 4.1.1 illustrates a simplified block diagram of the BA131MPEG4E IP showing the internal modules and its interfaces. The video data is organized in macro blocks under YUV format (4:2:0 resolutions). One macro block is made of 4 luminance blocks (8x8), 1 Cb block (8x8) and 1 Cr block (8x8). The video data is sent to the core through its video interface in macro block raster scan order. It generates the compressed stream at its Compressed Data Interface. The stream contains fully compliant headers and is regulated to a given bit rate if the CBR option is enabled (together with external microprocessor running the rate allocation algorithm).

The encoder has a generic interface to a memory controller, allowing the connection to any custom memory controller. Thanks to the burst nature of data transfers at this interface, the core can be used with simple SRAM but also SDRAM or DDR SDRAM. The core can be delivered with a standard SRAM controller; a suitable SDRAM controller is separately available. The core has been optimized in order to minimize the amount and bandwidth of off-chip memory. A single frame needs to be stored and accesses are reduced to 1 read and 1 write per input sample. The encoder has a simple generic interface to an external CPU in order to configure the various parameters of the core and to monitor the status of the encoding process. The following sections describe the modules constituting the BA131MPEG4E core as depicted under Figure 4.1.1.

4.1.4 Motion estimation:

The first module of the core is the motion estimation engine. This module is bypassed for Intra-coded pictures (I-VOP), which are not coded with reference to any other picture. Technical specifications are subject to change without prior notice The motion estimation engine uses an advanced directional search algorithm able to precisely and rapidly match the current macro block (16x16 pixels) with its equivalent in the reference frame. The core uses the frame stored in external memory as a reference. The processing generates one motion vector per macro block, giving the direction and amplitude of the detected motion. The matching precision is half a pixel. The motion estimation engine features advanced capabilities in order to shorten the search time as much as possible. This module also delivers statistics used by the rate allocation algorithm. The module also detects when a macro block cannot be registered correctly to the reference frame and should better be encoded as an Intra macro block.

4.1.5 Motion compensation:

This module computes the estimation error induced by the use of the vector generated by the motion estimation engine. This module is bypassed for Intra-coded pictures (I-VOP). It makes the difference between the current macro block (in luminance and chrominance planes) and the predicted macro block from the reference frame, using the estimated motion vector. The result is known as the prediction error and must be encoded by the texture coding.

4.1.6 Texture coding:

This module encodes error frames when using P-VOPs (resulting from motion compensation) or completes frames when using I-VOPs. This module has advanced low-power features where part of the processing is switched off when it is detected to be useless. An approximation of its result is then used instead. The texture coding is made of Discrete Cosine Transform (DCT), AC/DC prediction, quantization and zigzag encoding and works on block level (8x8):

• The DCT de-correlates the frequency contents of the 8x8 blocks and delivers a matrix of 64 frequency coefficients, representing the frequency contents of the original block of data.

• This is then quantized using a scalar quantizer. The quantization factor is programmable by the user, allowing him to set the quality level.

• The AC/DC predictor is used for I-VOPs and performs a prediction of the first line or the first column of the quantized matrix, based on the transformed blocks situated on the left and on top of the current block. The prediction source (top or left) is determined by a gradient analysis of DC coefficients of the transformed blocks situated on top, top-left and left. This prediction results in a higher compression efficiency.

• The quantized matrix is then processed by the zigzag encoder, which reads this 8x8 matrix in a predefined scan order; this results in a chain of coefficients where most of these are zeros. This is then further encoded thanks to a run encoder in order to reduce the size of the representation.

4.1.7 Entropy encoder:

The entropy encoder finalizes the data compression by applying a Huffman encoding to both the motion vectors and the compressed pixels. This module uses pre-defined look-up tables.

4.1.8 Bitstream packetization:

This module generates compliant MPEG-4 VOL and VOP headers (short headers and data partitioning are not supported but the core can be customized to add these features). The encoder includes an output buffer allowing the user to generate a stream at constant bit rate (CBR mode) by coupling the core to a small microprocessor running a rate allocation algorithm (Nios or Microblaze for instance). The rate allocation algorithm can be purchased optionally.

4.1.9 Texture update:

This feedback loop is performing the inverse operations of the texture coding: unzigzag, inverse quantization and Inverse Discrete Cosine Transform (IDCT). This allows the encoder to take into account quantization errors occurring at the decoder side when the picture is decoded. The encoder then uses the result of this texture update module to update the contents of the frame store (when needed). This new contents is then ready to be used as a reference frame for encoding the next frame. Technical specifications are subject to change without prior notice.

4.1.10 Rate allocator:

This optional module is dedicated to regulate the output of the encoding IP core to the bit rate specified by the user. This module makes use of a patented rate allocation algorithm, exploiting statistical information available at the motion estimation to improve its efficiency and provide a more stable stream bit rate and quality. This module is implemented as software code able to run on a simple processor (Nios or Microblaze for instance). The rate allocator can also be customized to be mapped as a 100% hardware block.

4.1.11 Video Compression

The aim of video compression is to remove redundant information from a digitized video sequence. It is critical to choose an appropriate compression method for use in video streaming over Bluetooth, as it provides time-varying wireless link with limited bandwidth up to 732Kbps. This section briefly describes video compression techniques including MPEG-4 and H.263 that are used by current researches of this area.

4.1.12 MPEG-4

A large portion of works [1, 3, 4, and 8] reviewed in previous sections employ MPEG-4 as video codec for streaming over Bluetooth. MPEG-4 is one of the newest video compression techniques and allows much lower compression ratios than the previous MPEG-2. MPEG-4 is ideally suited to low bandwidth applications, exactly matching the requirements for video over a wireless Bluetooth network. MPEG-4 uses motion vectors between frames to encode temporal redundancy and the discrete cosine transform (DCT) to encode spatial redundancy. MPEG-4 provides three modes for encoding an input , these are namely:

1. Intra-frame (I-frame) is encoded independently of any other frame and can be constructed without reference to any other frames.
2. Predicted-frame (P-frame) is predicted (using motion compensation) based on another previously decoded I-frame.
3. Bidirectional Interpolated-frame (B-frame) is predicted based on past as well as future frames.

For frames other than I frames, the amount of information to be coded reduces to differences between frames. This differential coding means that I frames are more important since all future frames till the next I frame are coded based on it. Therefore, extensive research on exploiting the information on the type of video frames has been proposed. Among these researches, *upper layer retransmission* mentioned in **3.1.3** is a kind of selective retransmission based on semantic importance of MPEG -4 frames in the context of streaming over Bluetooth links.

4.1.13 H.263

Several works reviewed in previous section employ H.263 as video codec for streaming over Bluetooth.

H.263 is a video compression algorithm and protocol which is standardized by ITU. It was designed for low bit-rate communication. The video source coding algorithm of H.263 is based on Recommendation

H.261 and is a hybrid of inter-picture prediction to utilize temporal redundancy and transform coding of the remaining signal to reduce spatial redundancy, however with some changes to improve performance and error recovery. H.263 lets users scale bandwidth usage and can achieve full motion video (30 frames per second) at speeds as low as 128Kbps. H.263 was also developed to low-quality stream video at bandwidths as low as 20 to 64Kbps.

Compared to MPEG-4, H.263 does not support some of the features such as compression efficiency and channel error robustness. However, it is widely accepted that it performs well for the target application at bi-rate between 20 and 64 Kbps. Therefore it is widely used in wireless networks with limited bandwidth.

It is investigated that a fuzzy logic-based video rate control technique which aims to regulate compressed video to a constant transmission rate, without incurring objectionable quality degradation. Conventional fuzzy rule-based control (FRC) does not adequately control the output video quality. Video information is therefore added into the FRC design by incorporating feed-forward scaling factors, derived from scene change features. The performance of this coder has been compared with other approaches measuring buffer occupancy, the number of coded bits per frame and peak signal-to-noise ratio.

4.2 Traffic Shaping Buffer:

- This is a temporary storage to smooth the video output traffic and partially eliminate the burstiness of the video stream entering Bluetooth wireless.
- Traffic shaping allows you to control outgoing traffic on an interface to match the traffic speed of the remote target interface and to ensure that the traffic conforms to specific policies. Traffic that adheres to a particular profile can be shaped to meet downstream requirements, thereby eliminating bottlenecks in topologies caused by data-rate mismatches.
- Traffic management is an important function in Bluetooth networks. The leaky bucket algorithm is a general algorithm that can be effectively used to police real time traffic. Frame Relay and network use a form of the leaky bucket algorithm for traffic management. Frame Relay networks use a continuous state version of the leaky bucket algorithm called the Generic Cell Rate Algorithm (GCRA) to police traffic at the entrance to the Frame Relay network. The network traffic consists of fixed size cells/packets.

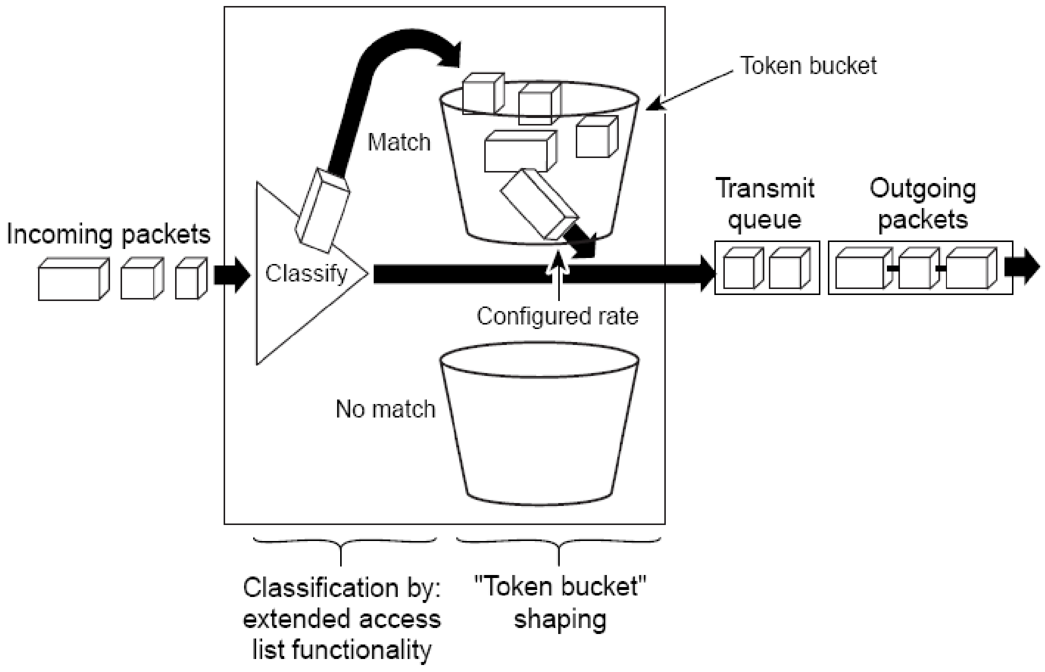


Fig. 4.2.1 A General example of Traffic Shaping Buffer

4.2.1 Benefits of Shaping Traffic on a Network

- The benefits of shaping traffic on the network include the following:
- It allows you to control the traffic going out an interface, matching the traffic flow to the speed of the interface.
- It ensures that traffic conforms to the policies contracted for it.
- Traffic shaping helps to ensure that a packet adheres to a stipulated contract and determines the appropriate quality of service to apply to the packet.
- It avoids bottlenecks and data-rate mismatches. For instance, central-to-remote site data speed mismatches.
- Traffic shaping prevents packet loss.

4.2.2 Using traffic shaping buffer in various scenarios:

- Control access to bandwidth when, for example, policy dictates that the rate of a given interface should not on the average exceed a certain rate even though the access rate exceeds the speed.
- Configure traffic shaping on an interface if you have a network with differing access rates. Suppose that one end of the link in a Frame Relay network runs at 256 kbps and the other end of the link runs at 128 kbps. Sending packets at 256 kbps could cause failure of the applications using the link.
- A similar, more complicated case would be a link-layer network giving indications of congestion that has differing access rates on different attached data terminal equipment (DTE); the network may be able to deliver more transit speed to a given DTE device at one time than another. (This scenario warrants that the token bucket be derived, and then its rate maintained.)
- If sub rate service is offered, traffic shaping makes use of the router to partition the T1 or T3 links into smaller channels.
- Traffic shaping is especially important in Frame Relay networks because the switch cannot determine which packets take precedence, and therefore which packets should be dropped when congestion occurs. Moreover, it is of critical importance for real-time traffic such as Voice over Frame Relay (VoFR) that latency be bounded, thereby bounding the amount of traffic and traffic loss in the data link network at any given time by keeping the data in the router that is making the guarantees. Retaining the data in the router allows the router to prioritize traffic according to the guarantees it is making. (Packet loss can result in detrimental consequences for real-time and interactive applications.)

4.3 Neuro-Fuzzy implementation

- This is an integrated controller which monitors the output rate or departure rate of the traffic-shaping buffer frame by frame to co-ordinate the video traffic entering Bluetooth.
- The inputs to the NF controller are the queue length in the traffic-shaping buffer $X(f)$ and the available memory space in the token-bucket $Y(f)$.
- The output from the NF controller is departure rate $R_d(f)$, will be measured in kilobits per second.

4.4 Rule-based-fuzzy implementation

- This controller regulates the average arrival rate to the traffic-shaper to prevent either overflow or starvation of the buffer on a Group Of Picture (GOP) by GOP basis.
- The inputs to the RBF controller are the mean X(f) and standard deviation Σ(X) of queue length in the traffic-shaping buffer.
- The output from the RBF controller is the desired arrival rate.

Automatic design of fuzzy rule-based classification systems based on labeled data is considered. It is recognized that both classification performance and interpretability are of major importance and effort is made to keep the resulting rule bases small and comprehensible. An iterative approach for developing fuzzy classifiers is proposed. The initial model is derived from the data and subsequently, feature selection and rule base simplification are applied to reduce the model, and a GA is used for model tuning. An application to the Wine data classification problem is shown.

Rule-based expert systems are often applied to classification problems in fault detection, biology, medicine etc. Fuzzy logic improves classification and decision support systems by allowing the use of overlapping class definitions and improves the interpretability of the results by providing more insight into the classifier structure and decision making process. The automatic determination of fuzzy classification rules from data has been approached by several different techniques: neuro-fuzzy methods, genetic-algorithm based rule selection and fuzzy clustering in combination with GA-optimization.

Traditionally, algorithms to obtain classifiers have focused either on accuracy or interpretability.

4.4.1 The Model Structure

Fuzzy classification rules are applied that each describe one of the N_c classes in the data set. The rule antecedent is a fuzzy description in the n-dimensional feature space and the rule consequent is a crisp (non-fuzzy) class label from the set f={1; 2; : : : ;N_c}:

$$R_i : \ \textbf{If } x_1 \text{ is } A_{i1} \textbf{ and } \ldots x_n \text{ is } A_{in} \textbf{ then } g_i = p_i, \ i = 1, \ldots, M$$

Eq….. (4.1.1.1)

Here n denotes the number of features, x = [x_1; x_2; : : : ; x_n].
T is the input vector, gi is the output of the ith rule and A_{i1}; : : : ;A_{in} are the antecedent fuzzy sets. The and connective is modeled by the product operator, allowing for interaction between the propositions in the antecedent. The degree of activation of the ith rule is calculated as:

$$\beta_i(x) = \prod_{j=1}^{n} A_{ij}(x_j), \ \ i = 1, 2, \ldots, M$$

Eq…. (4.1.1.2)

The output of the classifier is determined by the rule that has the highest degree of activation:

$$y = g_{i^*} \ , \ i^* = \underset{1 \le i \le M}{\arg\max} \ \beta_i$$

Eq.... (4.1.1.3)

In the following we assume that the number of rules corresponds to the number of classes, i.e., M = N_c. The certainty degree of the decision is given by the normalized degree of _ring of the rule:

$$CF = \beta_{i^*} / \sum_{i}^{M} \beta_i$$

Eq.. (4.1.1.4)

4.4.2 RBF Network Architecture:

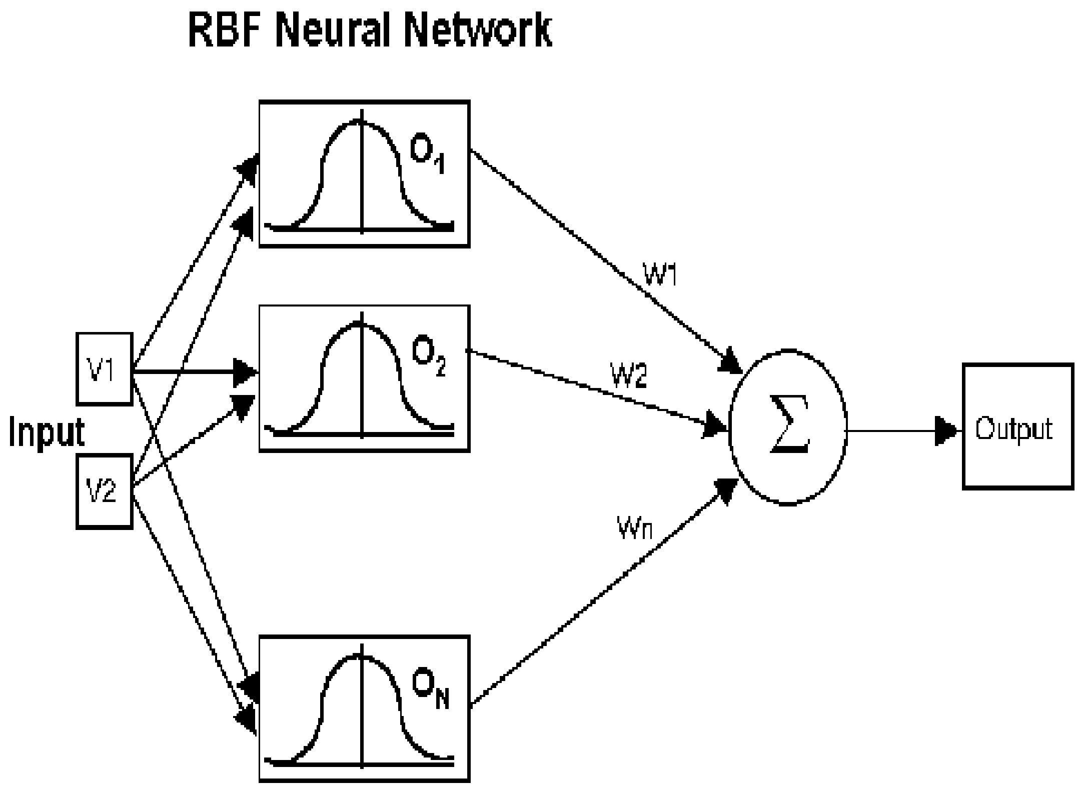


Fig 4.2.2.1 RBF networks Layers

4.4.3 RBF networks have three layers:

1. Input layer – There is one neuron in the input layer for each predictor variable. In the case of categorical variables, N-1 neurons are used where N is the number of categories. The input neurons (or processing before the input layer) standardize the range of the values by subtracting the median and dividing by the interquartile range. The input neurons then feed the values to each of the neurons in the hidden layer.

2. Hidden layer – This layer has a variable number of neurons (the optimal number is determined by the training process). Each neuron consists of a radial basis function centered on a point with as many dimensions as there are predictor variables. The spread (radius) of the RBF function may be different for each dimension. The centers and spreads are determined by the training process. When presented with the x vector of input values from the input layer, a hidden neuron computes the Euclidean distance of the test case from the neuron's center point and then applies the RBF kernel function to this distance using the spread values. The resulting value is passed to the the summation layer.
3. Summation layer – The value coming out of a neuron in the hidden layer is multiplied by a weight associated with the neuron (W1, W2, ...,Wn in this figure) and passed to the summation which adds up the weighted values and presents this sum as the output of the network. Not shown in this figure is a bias value of 1.0 that is multiplied by a weight W0 and fed into the summation layer. For classification problems, there is one output (and a separate set of weights and summation unit) for each target category. The value output for a category is the probability that the case being evaluated has that category.

4.5 Leaky Bucket (GCRA):

- Traffic-rate policing will be carried out by means of the Generic Cell Rate Algorithm (GCRA) – a rule by which video streams can be judged to be complying with the terms of the traffic contract, GCRA is commonly known as ‘leaky bucket’ or ‘token-bucket’.

- The token-bucket will be located prior to the HCI and measures the departure rate against the contracted mean rate.

- The actual token-rate r_{actual} will be varied according to the level of interferences in the Bluetooth channel and its environment.

- The contracted token-rate is the maximum bandwidth for a Bluetooth ACL link, which will be set around 650 kb/s with 32 kb/s for the sound and 618 kb/s for the video images.

- The leaky bucket is a “traffic shaper”: It changes the characteristics of a packet stream.

- Traffic shaping makes the network more manageable and predictable

- Usually the network tells the leaky bucket the rate at which it may send packets when a connection is established.

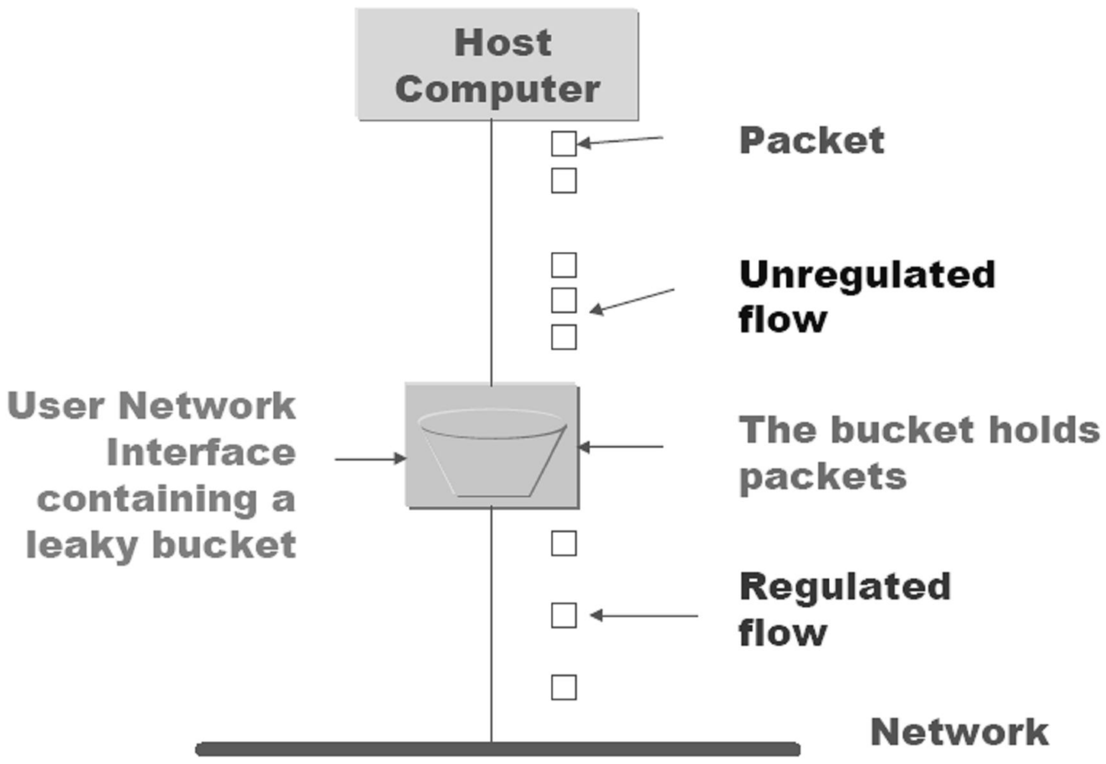


FIG. 4.5.1 AN EXAMPLE OF GCRA

4.6. DATA FLOW DIAGRAM

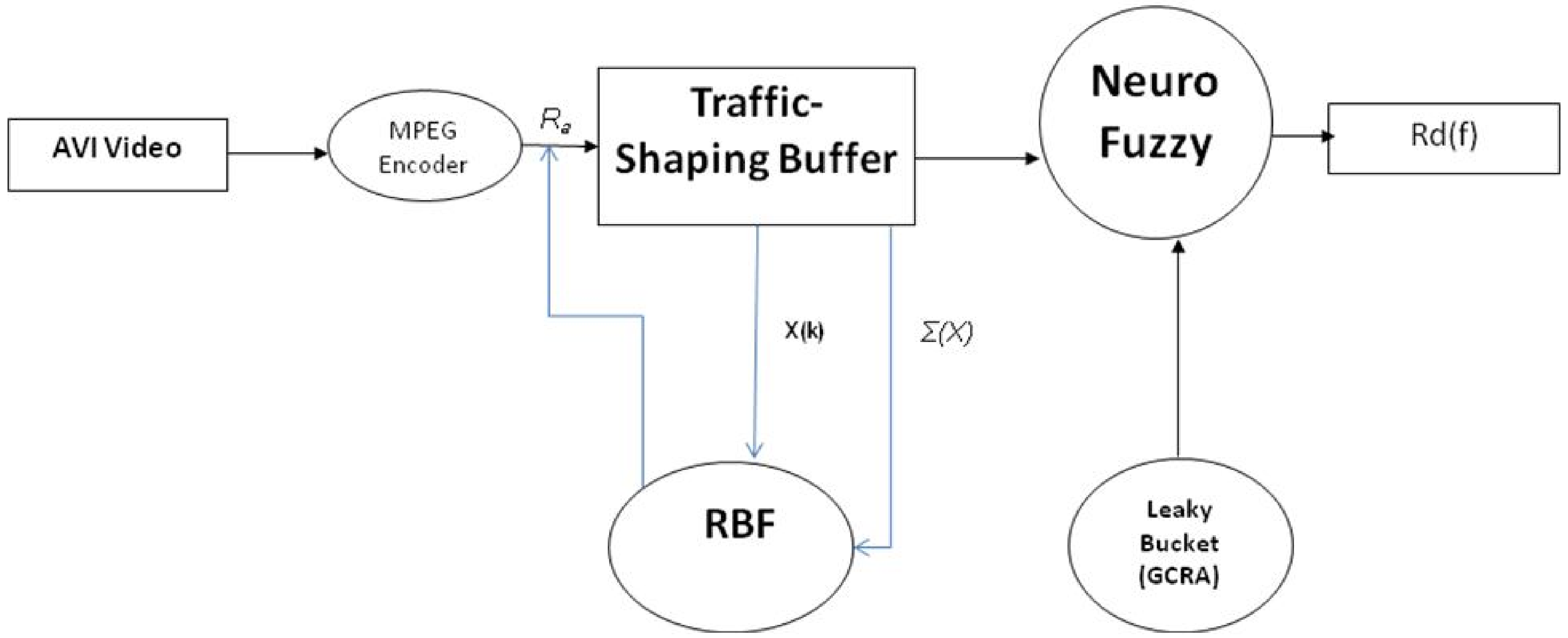


Fig 4.6.1 Data Flow Diagram

CHAPTER 5

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS:

- Intel MotherBoard.
- Pentium IV Processor 1.8 GHz.
- 40 GB HDD.
- Keyboard.
- Mouse.
- CRT/TFT/LCD Monitor.

SOFTWARE REQUIREMENTS:

- MatLab 7.0
- Windows XP/2003

Chapter 6

Implementation

X(f).m:

```
clear all;
Ractual = 0.1;
Ra = [2.1 2.2 2.3 2.09 1.98 1.97 1.8 1.9 2.1 2.2];
Td = [1 1 1 1 1 1 1 1 1 1];
Tf = [.001 .021 .021 .211 .021 .001 .0021 .0041 .061 .011];
f = 10;
Rd = [0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 ];

for k=1:9
    Ra(k) = rand+2;
    TDmax = max ( 1 / Ra(k+1), 1/Ractual);
    TDmin = min ( 1 / Ra(k+1), 1/Ractual);
    Rd(k) = 1 / ( Td(k) * ( TDmax - TDmin) + TDmin);
end

for k=1:9
    for ki = 0:9
        temp=0;
        for fi = 1: (f-1)
             temp = temp + (Ra(fi) - Rd(fi))*Tf(fi);
        end
        X(10-k) = 1/k * temp;
    end
end
plot(X);
ylabel('Degree of Membership')
xlabel ('X(f)')
grid
```

Y(f).m:

```
clear;
Ractual = 1.3;
Ra = [2.1 2.2 2.3 2.09 1.98 1.97 1.8 1.9 2.1 2.2];
Td = 1;
b = 231 ; % Size of Token Bucket
f = 10;
Rd = [0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 ];
for k=1:10

    TDmax = max ( 1 / Ra(mod(k+1,10)+1), 1/Ractual);
    TDmin = min ( 1 / Ra(mod(k+1,10)+1), 1/Ractual);
    Rd(k) = 1 / ( Td * ( TDmax - TDmin) + TDmin);
    temp = 0;
    for fi = 1: (f-1)
        Tf = rand * 34;
         temp = temp + (Ractual- Rd(mod(fi,10)+1)) * Tf ;
    end
        Y(k) = 1/b * temp;
end

plot(Y);
ylabel('Degree of Membership')
xlabel ('Y(f)')
grid
```

id(f):

```
clear all;
Ractual = 0.1;
Ra = [2.1 2.2 2.3 2.09 1.98 1.97 1.8 1.9 2.1 2.2];
Td = [1 1 1 1 1 1 1 1 1 1];
Tf = [.001 .021 .021 .211 .021 .001 .0021 .0041 .061 .011];
f = 10;
Rd = [0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 ];


for k=1:9
    Ra(k) = rand+2;
    TDmax = max ( 1 / Ra(k+1), 1/Ractual);
    TDmin = min ( 1 / Ra(k+1), 1/Ractual);
    Rd(k) = 1 / ( Td(k) * ( TDmax - TDmin) + TDmin);
end
Ra
for k=1:9
    for ki = 0:9
        temp=0;
        for fi = 1: (f-1)
             temp = temp + (Ra(fi) - Rd(fi))*Tf(fi);
        end
        X(10-k) = 1/k * temp;
    end
end

% End of Xf


Ractual = 1.3;
Ra = [2.1 2.2 2.3 2.09 1.98 1.97 1.8 1.9 2.1 2.2];
Td = 1;
%Tf = [10 21 21 21 21 15 21 41 61 11 12];
b = 231 ; % Size of Token Bucket
f = 10;
Rd = [0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 0.1 ];
for k=1:10
```

```
    TDmax = max ( 1 / Ra(mod(k+1,10)+1), 1/Ractual);
    TDmin = min ( 1 / Ra(mod(k+1,10)+1), 1/Ractual);
    Rd(k) = 1 / ( Td * ( TDmax - TDmin) + TDmin);
         temp = 0;
        for fi = 1: (f-1)
            Tf = rand * 34;
          temp = temp + (Ractual- Rd(mod(fi,10)+1)) * Tf ;
        end
        Y(k) = 1/b * temp;
end


%end of y(k)

for k=1:9

if (Y(k) >=0.0 && Y(k)<=0.2 )&& (X(k) >=0.0 && X(k)<=0.2)
id(k) = 0.1;

    elseif (Y(k) >=0.0 && Y(k)<=0.2 ) && (X(k) >0.2 &&
X(k)<=0.5)
        id(k) = 0.1;

    elseif (Y(k) >=0.0 && Y(k)<=0.2 )&& (X(k) >0.5 && X(k)
<=0.8)   id(k) = 0.09;

    elseif (Y(k)>= 0.0 && Y(k)<=0.2 )&& (X(k) >0.8 && X(k)
<=1.0)   id(k) = 0.7;

    elseif (Y(k) > 0.2 && Y(k)<= 0.5) && (X(k) >0.8 && X(k)
<=1.0)   id(k) = 0.95;

    elseif (Y(k) >0.2 && Y(k)<=0.5 ) && (X(k) >0.2 &&
X(k)<=0.5 )   id(k) = 0.45;

    elseif (Y(k) >0.2 && Y(k) <=0.5 )&& (X(k) >0.5 && X(k)
<=0.8)   id(k) = 0.3;
```

```
    elseif ((Y(k) >0.2 && Y(k)<=0.5 )&& (X(k) >0.8 && X(k)
<=1.0))   id(k) = 0.09;

    elseif (Y(k) > 0.5 && Y(k) <= 1.0 )&& (X(k) >=0.0 &&
X(k) <=0.2)   id(k) = 0.95;

    elseif (Y(k) > 0.5 && Y(k) <= 1.0)&& (X(k) >0.2 &&
X(k)<=0.5)   id(k) = 0.3;

    elseif (Y(k) > 0.5 && Y(k) <= 1.0 )&& (X(k) >0.5 &&
X(k) <=0.8)   id(k) = 0.1;

    elseif (Y(k) > 0.5 && Y(k) <= 1.0 )&& (X(k) >0.8 &&
X(k) <=1.0)   id(k) = 0.09;
end

end

plot( id);
id
text(2.1,0.85,'Very-Large', 'FontSize', 12 );
text(2.1,0.65,'Large','FontSize', 12);
text(2.1,0.55,'Intermediate','FontSize', 12);
text(2.1,0.25,'Small','FontSize', 12);
text(2.1,0.15,'Very-Small','FontSize', 12);

grid;
```

Id(f): Sugeno Model Impelementation:

1. If (token–Y(f) is empty) and (queue–X(f) is empty) then (output–id is very-large)
2. If (token–Y(f) is empty) and (queue–X(f) is medium) then (output–id is very-large)
3. If (token–Y(f) is empty) and (queue–X(f) is full) then (output–id is very-large)
4. If (token–Y(f) is empty) and (queue–X(f) is very-full) then (output–id is large)
5. If (token–Y(f) is medium) and (queue–X(f) is empty) then (output–id is very-large)
6. If (token–Y(f) is medium) and (queue–X(f) is medium) then (output–id is intermediate)
7. If (token–Y(f) is medium) and (queue–X(f) is full) then (output–id is small)
8. If (token–Y(f) is medium) and (queue–X(f) is very-full) then (output–id is very-small)
9. If (token–Y(f) is full) and (queue–X(f) is empty) then (output–id is very-large)
10. If (token–Y(f) is full) and (queue–X(f) is medium) then (output–id is small)
11. If (token–Y(f) is full) and (queue–X(f) is full) then (output–id is very-small)
12. If (token–Y(f) is full) and (queue–X(f) is very-full) then (output–id is very-small)

RTO: Retransmission Limit (Mamdani Model Impelementation)

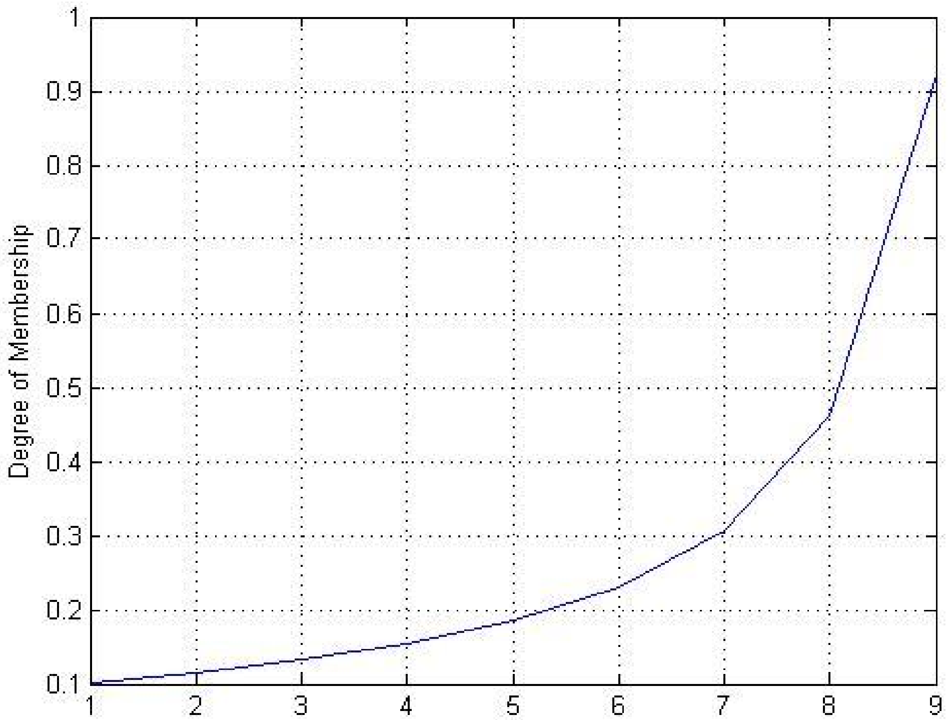
X: Buffer Fullness, Y: Deadline/Delay, Z: Retransmission Limit

1. If (X is High) and (Y is TooLow) then (Z is Normal)
2. If (X is High) and (Y is Low) then (Z is Normal)
3. If (X is High) and (Y is Normal) then (Z is Low)
4. If (X is High) and (Y is High) then (Z is TooLow)
5. If (X is High) and (Y is TooHigh) then (Z is TooLow)
6. If (X is TooHigh) and (Y is TooLow) then (Z is TooHigh)
7. If (X is TooHigh) and (Y is Low) then (Z is High)
8. If (X is TooHigh) and (Y is Normal) then (Z is Normal)
9. If (X is TooHigh) and (Y is High) then (Z is Low)
10. If (X is TooHigh) and (Y is TooHigh) then (Z is TooLow)
11. If (X is Low) and (Y is TooLow) then (Z is TooHigh)
12. If (X is Low) and (Y is Low) then (Z is High)
13. If (X is Low) and (Y is Normal) then (Z is Normal)
14. If (X is Low) and (Y is High) then (Z is Low)
15. If (X is Low) and (Y is TooHigh) then (Z is TooLow)

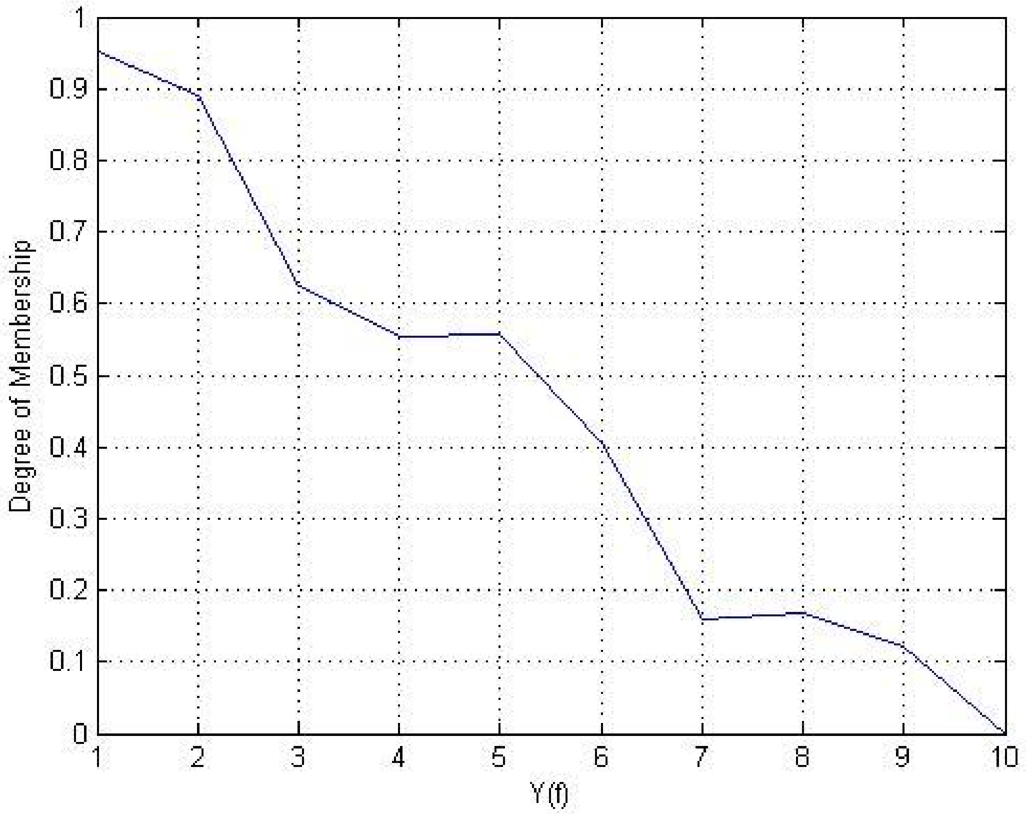
CHAPTER 7

RESULTS

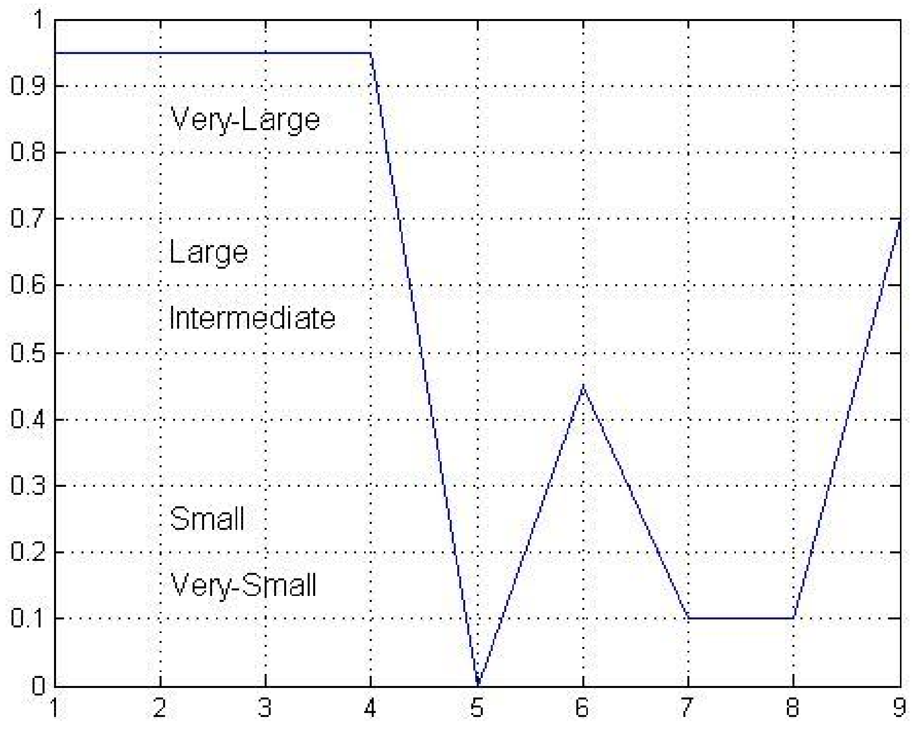
X(f): Output from the Traffic Shaping Buffer:



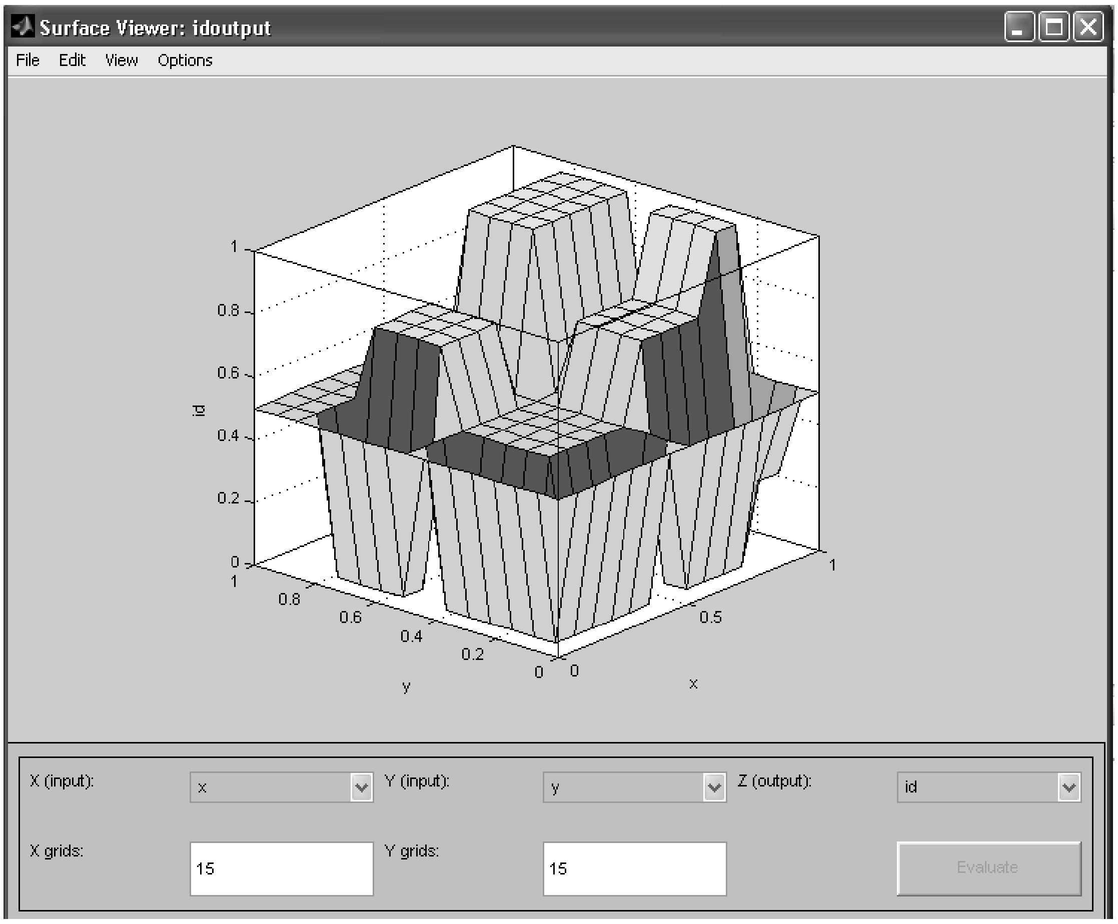
<u>Y(f): Token Space Available in Token Bucket:</u>



id(f): Final Departure of Packets to Bluetooth Link:

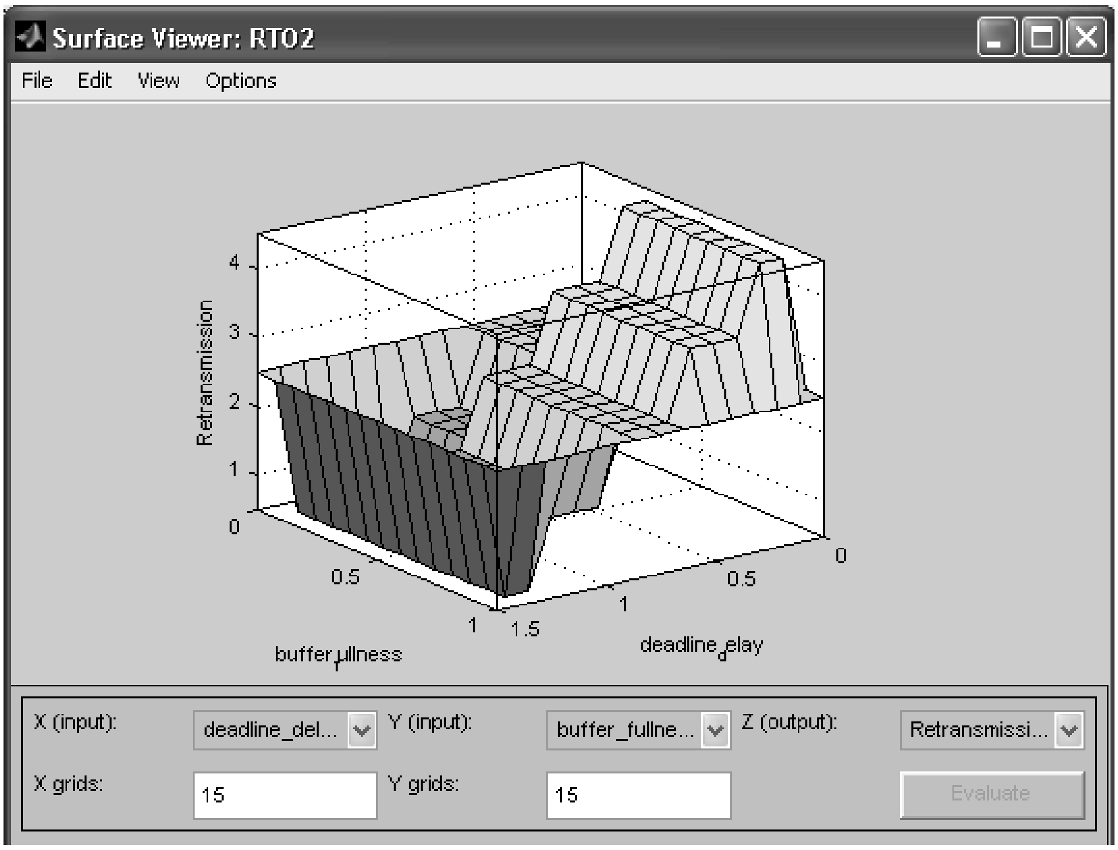


id(f): Sugeno Model:



X (input): x
Y (input): y
Z (output): id
X grids: 15
Y grids: 15

<u>*RTO (Retransmission Limit Mamdani Model):*</u>



X (input): deadline_del...
Y (input): buffer_fullne...
Z (output): Retransmissi...
X grids: 15
Y grids: 15

CHAPTER 8

CONCLUSIONS

This document presents the proposal on a fuzzy approach to video transmission over Bluetooth wireless. The VoIP technology will be used to allow MPEG video to be transmitted using the higher bandwidth supported by the Bluetooth ACL links. To simulate the interference in the Bluetooth channel, a Gaussian noise will be added to the contracted token rate. Furthermore, in order to address the network congestion problem caused by the burstiness of video stream bit rate, a fuzzy control scheme will be developed. The control scheme introduces a traffic-shaping buffer and two rule-based fuzzy controllers. The two rule-based fuzzy controllers will adjust the input bit rate to and the output bit rate from the traffic-shaper.

In comparison with the open loop VBR system, the proposed control system will significantly reduce both the variance of output bit rate to the network and the number of dropped data, which ultimately will result to a guarantee in data source stability while maintaining image quality at the receiving end. In comparison with a conventional CBR system, the proposed system takes the current network conditions into consideration; it will reduce the amount of dropped data in presence of very high-level (combined noise) interference in the Bluetooth channel.

This work is concerned with reduction of oversubscribed MPEG VBR video transmission over a Bluetooth ACL link. A novel integrated Neuro-Fuzzy (NF) control scheme is utilized to reduce the burstiness of the traffic-shaping buffer output rate to enable the VBR encoded video to enter the Bluetooth network reducing time delay, data loss and image quality degradation.

The proposed NF scheme considerably reduces the standard deviation of the output bit rate to the Bluetooth channel and the number of dropped-data, which ultimately result to data transmission stability, while maintaining image quality at the receiving end. The NF control scheme also improves and maintains the quality of service with noise interferences over the Bluetooth channel. In conclusion, as the integrated NF and RBF schemes are both based on simple algorithms, the overall control system only requires an acceptable processing power, which could be used for real-time implementations in delay-sensitive MPEG VBR video services in mobile devices.

Rule-based expert systems have been applied in a vast number of application areas. An important advantage of the fuzzy expert system is that the knowledge is expressed as easy-to-understand linguistic rules. If we have data, the fuzzy expert system can be taught using neural network learning, EC, or other adaptation techniques. It is to be expected that the number of applications will grow considerably in the future now that success is clearly proven for these methods.

CHAPTER 9

REFERENCES

1. Ajith Abraham, *Oklahoma State University, Stillwater, OK, USA*, “Rule-based Expert System”.
2. Anders Dahlberg, “Traffic Engineering in a Bluetooth Piconet”.
3. Bluetooth SIG, Specification of the Bluetooth system—wireless connections made easy, Bluetooth Core Specification vl.2, 2003. Available from: <https://www.bluetooth.org/spec>.
4. CISCO SYSTEM, “Regulating Packet Flow Using Traffic Shaping”
5. CISCO SYSTEM. “Shaping Traffic”
6. C.J. Chang, B.W. Vhen, T.Y. Liu, F.C. Ren, Fuzzy/neural congestion control for integrated voice and data DS-CDMA/FRMA cellular networks, IEEE Journal on Selected Areas in Communications 2 (2000) 283–293.
7. Computing, Communications Technology and Mathematics Dept, 100 Minories, London EC3N 1JY, UK.
8. D.L. Gall, MPEG: “A video compression standard for multimedia application, Communications of the ACM” 34 (4) (1991) 46–58.
9. Dr. Yeali S. Sun, National Taiwan University, “Policing”
10. H.B. Kazemian, Li Meng, “A fuzzy control scheme for video transmission in Bluetooth wireless”.
11. Hans Roubos, Magne Setnes, and Janos Abonyi, “Learning Fuzzy Classification Rules from Data”

12. H. B. Kazemian Senior Member, IEEE, and S. Chantaraskul, London Metropolitan University, “An Integrated Neuro-Fuzzy Approach to MPEG Video Transmission in Bluetooth”
13. JELENA MISIC VOJISLAV B. MISIC, “POLLING, SCHEDULING, AND TRAFFIC CONTROL”.
14. J. Bandara, X.M. Shen, Z. Nurmohamed, A fuzzy resource controller for non-real-time traffic in wireless networks, in: Proceedings of ICC 2000-IEEE International Conference on Communications, vol. 1, 200, pp. 75–79.
15. J. Bray, C.F. Sturman, Bluetooth 1.1: Connect with Cables, second ed., Prentice Hall, PTR, 2001.
16. Katalin Hangos, Department of Computer Science, University of Veszprem, “Fuzzy rule-based systems;Fuzzy sets and calculus”.
17. M. Albrecht, M. Frank, P. Martini, A. Wenzel, M. Schetelig, A. Vilavaara, “IP services over Bluetooth: leading the way to a new mobility, in Proceedings of the 24th Conference on Local Computer Networks”, Lowell, Massachusetts, USA, 1999, pp. 2–11.
18. Pal Halvorsen, Øystein Yri Sunde, Andreas Petlund and Carsten Griwodz, “Programmable Subsystems: A Traffic Shaping & Packet Prioritizing Case Study”, University of Oslo, Norway 2Simula Research Lab., Norway, e-mail: {paalh, oysteisu, andreape, griff}@ifi.uio.no.
19. PERFORMANCE MODELING, AND ANALYSIS, OF BLUETOOTH, NETWORKS
20. P.M. Grant, Yoo-Sok Saw and J.M. Hannah, “Nonlinear fuzzy rule-based approach for estimating video traffic rate”.
21. Queue management, Rutgers University.
22. R.Q. Hu, D. Peter, “A predictive self-tuning fuzzy logic feedback rate controller, IEEE/ACM Transactions on Networking” 8 (2000) 697–709.

23. Stephen Yurkovich, The Ohio State University, Department of Electrical Engineering, The Ohio State University. “Fuzzy Control, Kevin M. Passino, Department of Electrical Engineering”.
24. Toshihisa OZAWA, “Performance Characteristics of a Packet based Leaky-Bucket Algorithm for ATM Networks”
25. Todd Lizambri, Fernando Duran and Shukri Wakid, Present: Hongming Wu, “Priority Scheduling and Buffer Management for ATM Traffic Shaping”
26. Wang Xiaohang, “Video Streaming over Bluetooth: A Survey”, Institute for Infocomm Research (I2R) / School of Computing, NUS, xwang@i2r.a-star.edu.sg.