Multi-Stages Hybrid ARQ with Conditional Frame Skipping and Reference Frame Selecting Scheme for Real-Time Video Transport Over Wireless LAN

Min Chen Gang Wei

chenminnew@21cn.com *ecgwei@scut.edu.cn*

Dept. of Electronic and Communication Engineering, South China University of Technology, Guang Zhou, P.R. China

Abstract - *This paper proposes a multi-stages hybrid ARQ/FEC scheme with frame skipping, reference frame selecting and intra-frame refreshing techniques for H.26L real-time video streaming over WLAN. In this paper, adaptive ARQ/FEC is adopted to save unnecessary FEC overhead; conditional frame skipping is adopted to avoid ETE delay inflation due to MAC ARQ/FEC error recovery; reference frame selecting is adopted to save bandwidth; intra-frame refreshing is adopted to prevent residual error propagation among consecutive frames. Through analysis and simulation, we show that multi-stages hybrid ARQ scheme has the following key properties: (1) high error recovery rate for real-time video frames, (2) excellent scalability, (3) high transmission efficiency. In case of severe unreliable channel which causes losing a lot of IP packets, we obtain a peak signal-to-noise ratio improvement up to about 10dB compared to the hybrid ARQ proposed in [1].*

Index Terms — ARQ, IEEE 802.11b, H.26L video, multimedia communication, wireless local area network.

I. INTRODUCTION

Unlike wired networks where packet losses are mainly caused by congestion, there are two major challenges for video streaming over WLAN: 1) fluctuations in channel quality and 2) high bit-error rates caused by fading or multi-path effect. With wireless networks gaining prominence and acceptance, especially the WLAN (wireless local area network) based on the IEEE 802.11 standards, it is foreseeable that streaming of audio/video will be a critical part of the wireless digital infrastructure. Specially, we investigate H.26L real-time video transmission over wireless IEEE 802.11b LANs. H.26L is a new video coding standard being established recently by ITU-T [3,4]. It is an effective video coding standard due to the combination of transform and quantization. Different from H.263 [5], the residual coding of H.26L is based on 4x4 blocks, and an integer transform is used. The high compression ratio makes H.26L real-time video application very suit transmitting over a low bit-rate channel, but also causes the signal susceptible to transmission errors. Even a single-bit error may cause the error to propagate to many frames because of motion compensated prediction and the variable-length coding used, which makes H.26L video communication with quality of service (QoS) a high challenging task. To have QoS provision or guarantee for real-time video application in the current best-effort WLAN, beyond using two effective approaches of congestion control and error control, we also modified IEEE 802.11b radio MAC algorithm that ensures that packets sent by a mobile host are differentiated [6-8]. A simple prioritization mechanism specifies that real-time video stream has higher service priority than data traffic.

Since that the compressed video bit stream is extremely sensitive to transmission error due to the frame dependency, error control techniques such as FEC and ARQ are necessary to obtain high transmission reliability required by video services [10,11,17,18,19,20,21]. There are two basic error correction mechanisms, namely Automatic Repeat request (ARQ) and Forward Error Correction (FEC). ARQ requires the receiver to explicitly (by means of negative acknowledgement) or implicitly (using positive acknowledgements and timeouts) request the retransmission of the lost/corrupted packets. While in FEC scheme, some redundant data, called parties, are transmitted together with original data to allow reconstruction of lost/corrupted packets at the receiver. Of these two error control mechanisms, FEC has been commonly suggested for real-time applications due to the strict delay requirements and semi-reliable nature of media streams [12]. Typical FEC schemes are stationary and must be implemented to guarantee a certain QoS requirement for the worst case, where unnecessary bandwidth overhead is wasted when the channel is in good state.

In order to overcome their individual drawbacks, the combination of these two basic classes of error control schemes, calls hybrid ARQ scheme, has been developed [1,2,13]. Paper [1] proposed a novel hybrid ARQ algorithm which specifically addresses the problem of video streaming over 802.11 networks. In the scheme, transmitter only sends data packets to the receiver initially, then parity packets until one of the following two events occurs: either an acknowledgment from the receiver arrives, or the deadline of transmission reaches. The scheme achieves high channel utilization because it efficiently reduces the amount of ACKs which share the same channel with data packets. A novel selective repeat ARQ protocol proposed in paper [14] also can achieve this goal. Instead of sending only one packet each time, selective repeat ARQ sends all the MPDU (MAC protocol data unit[9]) belonging to an MSDU (MAC service

Contributed Paper
Original manuscript received May 5, 2003
Revised manuscript received December 1, 2003

data unit [9]) before waiting for ACK or NAK. The receiver responds with ACK if all the MPDU have successfully received, or NAKs bearing sequence number of those MPDU received in error. Though our proposed scheme is similar to the algorithms mentioned above at the point of efficiently reducing the amount of ACKs, we further consider the estimated channel condition and real-time video encoder's ability to antagonize fluctuations in channel quality.

The acknowledgment of video packet considered in hybrid ARQ algorithm of paper [1] is initiated by application layer. Since ARQ protocols in MAC layer are able to active more rapidly than application layer, our scheme will directly handle the acknowledgment in MAC layer by introducing the packetization scheme described in section III.

Note that most ARQ schemes attempt to retransmit each MPDU up to three times. Similar to this idea, we divide a video frame's transmission into several transmission stages and limit the number of transmission stages. By numerical experiments, the selection of "three stages" is a compromise between delay boundaries and delivery reliability (we do not exclude the possibility that a "higher stages" algorithm show better performance results than the former one). As can be seen in [15], it is more feasible to allow MAC protocol giving up after a number of attempts passing the reliability responsibility to transport or application layer, especially for real-time traffic whose minimization of delay is more important than reliable transmission. If the decoding deadline of a video frame arrives, but the video frame has not been received completely, we will give up MAC's reliable service and seek help from H.26L encoder where frame skipping is implemented. It is necessary that the value of *Frame_Skipping_Threshold* be set correctly in order to keep the system at an optimum level. In our scenarios, we set the parameter to optimize the performance of system, in other word, a video frame is delivered as intact as possible by having to compromise little delay boundaries of a real-time application.

All the efforts mentioned above try to guarantee video frame be received perfectly as well as ETE (end-to-end) delay be reduced as much as possible. No matter what methods are implemented, video frame corruption is inevitable because of burst packet loss. In this case, we focus on adjusting encoder's parameters based on feedback information to reduce the ill effect that packet loss has on the restored video frame. To achieve this goal, H.26L coding scheme adopts multi-reference frames for motion estimation, and the number of reference frames can be specified before encoding each macro-block. Encoder takes each reference frame as reference for motion estimation, and computes corresponding PSNR (peak signal-to-noise ratio), then finds the best one with highest PSNR. It is an effective method to improve coding efficiency. Our contribution here is only taking intact frames received by decoder as reference. To achieve this goal, encoder will mark corrupted frames based on feedback information. If only one reference frame is intact, inter-coding can be implemented. In this case, bandwidth can be saved because inter-coding can get higher compression ratio than intra-coding. If all the reference frames are corrupted, intra coding is implemented by force. In this case, we will skip the next frame to achieve dynamic adjustment of bandwidth requirement since intra-frame is large but last for short periods of time. The scheme will be presented in section VI.

Simulation results show that our proposed schemes can effectively reduce the number of corrupted video frames so as to improve good throughput. In the case of severe unreliable channel causing heavy packet loss, we obtain a PSNR improvement up to about 10dB compared with the hybrid ARQ algorithm proposed in [1].

The rest of this paper is organized as follows. Section II depicts IEEE 802.11 WLAN architecture and prioritization enhancement mechanism. Section III presents our packetization and FEC strategy. Section IV introduces proposed hybrid ARQ based on three transmission stages with a pseudo-code illustration of the novel algorithm, then explains how to overcome the problems existing in hybrid ARQ scheme proposed in paper [1] when real-time video is transmitted. Section V depicts our conditional frame skiping algorithm based on feedback information of three stages. Section VI presents our reference frame selecting algorithm based on feedback information. Simulation results and analysis to justify the significance of our schemes will be explained in Section VII. Finally, Section VIII will conclude the paper.

II. IEEE 802.11 WLAN Architecture and Prioritization Enhancement Mechanism

A. IEEE 802.11 WLAN Architecture

In this section, we will briefly describe the IEEE 802.11 WLAN architecture. There are two major WLAN architectures, independent basic service set (IBSS, it is a simple, one-hop ad-hoc network) and infrastructure networks. Our research focuses on the former.

Fig.1 shows WLAN modeled by OPNET simulation tool. It consists of a pair of H.26L video server/client and several interference stations which generate background (best-effort) traffic. The real-time video streams are produced by the H26L server, and are encapsulated into RTP/UDP/IP packets. All the packets generated by H26L server and interference stations are sent to H26L client.

Though real-time video transmission over IEEE 802.11 WLAN has been studied by several researches, they either directly read the pre-encoded video from hard disk [1] or simply generated fake video packet which contains no real data. In other word, they were intent to fine tune network performance by controlling MAC layer, but didn't consider application layer. After translating the tremendous Visual C++ code of H.26L program (we chose recent H.26L video coding program: version JM 5.0) into external C file of OPNET, we integrated H.26L encoding & decoding functions into application layer so as to adjust video encoding parameters conveniently according to feedback information. We finally realized real-time video transmission in deed.

Introducing IEEE 802.11 protocol is beyond the scope of this paper. Interested readers are referred to [9] and references therein.

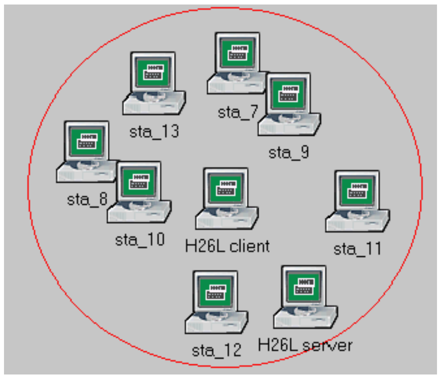

Figure 1. IEEE 802.11 WLAN Architecture

B. *Prioritization mechanisms for IEEE 802.11*

In papers [6-8], service differentiation priority per station is introduced by modifying the following functions or parameters: 1) back-off increase or decrease rule, 2) maximum frame length, and 3) AIFS (Arbitration Inter-frame Space). We specify that the AIFS of real-time video stream be smaller than that of background traffic. Thus real-time video traffic has higher priority.

The MAC protocol specified in IEEE 802.11 is distributed coordination function (DCF) known as carrier sense multiple access with collision avoidance (CSMA/CA). Under the load specified in Table II, ETE delay of video stream is too large to achieve the requirements of real-time service, as can be seen from Fig.2, where the horizontal distance between two "sent" and "received" curves represents ETE delay.

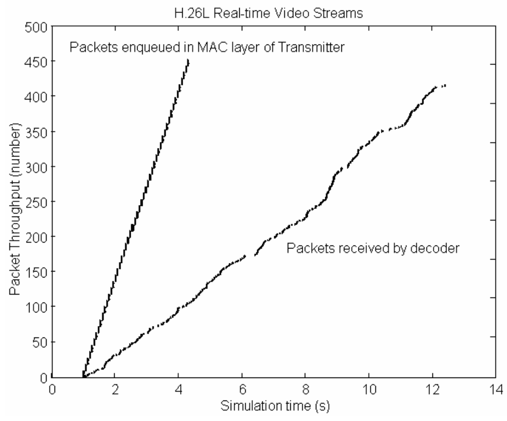

Figure 2. Num of packet sent and received without prioritization mechanism

To alleviate this problem, we introduce a simple prioritization mechanism by adjusting "AIFS_slot_num" in Table I. The value of AIFS further can be calculated according to (1). Under the same load, final ETE delay decrease 5.5 seconds, compared Fig.3 with Fig.2. But this effort still can't achieve the requirement of real-time service that ETE delay should be less than 150ms. In following sections, we will present our schemes to achieve this goal.

$$AIFS\_time = SIFS\_time + AIFS\_slot\_num \times slot\_time \quad (1)$$

TABLE I. A simple prioritization mechanism

Prioritization Mechanism	Traffic Type	
	BE traffic	*H.26L video*
SIFS_time(μs)	16	
Slot_time (μs)	9	
AIFS_slot_num	5	2
AIFS (μs)	61	34

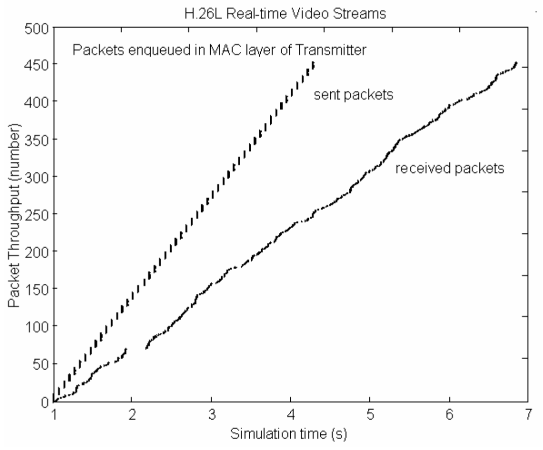

Figure 3. Num of packet sent and received with prioritization mechanism

III. Packetization and FEC Scheme

A. *Packetization Scheme*

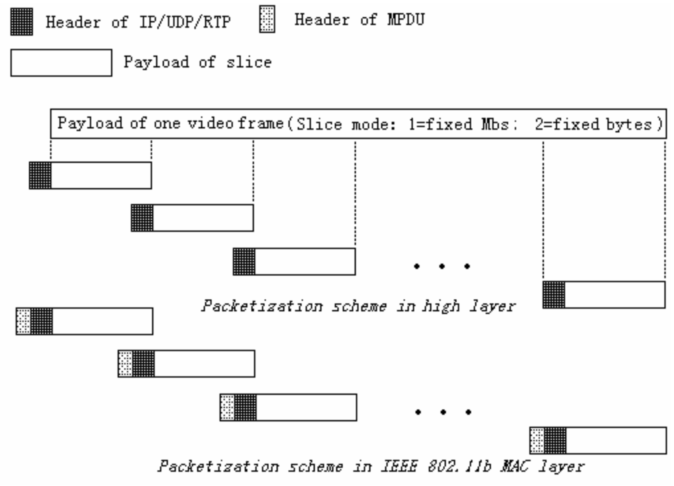
Header of IP/UDP/RTP
Header of MPDU
Payload of slice
Payload of one video frame(Slice mode: 1=fixed Mbs; 2=fixed bytes)
Packetization scheme in high layer
Packetization scheme in IEEE 802.11b MAC layer

Figure 4. Packetization Scheme

Each raw video frame is divided into several slices in H.26L packetization scheme. There are many slice modes for choice, among which two slice modes are in common use, one is "Fixed_Mbs", where the number of macro-blocks contained in each slice is fixed, the other is "Fixed_Bytes", where the maximum quanta of byte contained in each slice is fixed (the macro-block must be intact, so the size of each slice maybe different slightly).

When “Fixed_ Bytes” mode is chosen, intra-frame will be packetized into many slices, which is propitious to perform error concealment. However, for most P frame and B frame, relatively few slices will be generated. In this case, error concealment is hard to be effective, but packetization overhead is saved. On the contrary, “Fixed_ Mbs” mode is propitious to conceal the lost part of P frame and B frame, but exhibits lower channel utilization. We will choose “Fixed_Mbs” mode because it is simple and flexible.

As mentioned in section II, this paper focus on one hop Ad-Hoc WLAN architecture, so route discovery can be directly realized using MAC address contained in the header of MPDU, then IP/UDP header can be ignored. In our packetization scheme, each MSDU is packetized into only one MPDU, then the responsibility of transport reliability in application layer is shifted to MAC layer, which can save ETE delay since MAC layer actives more rapidly than application layer.

B. FEC (Forward Error Correction) Scheme

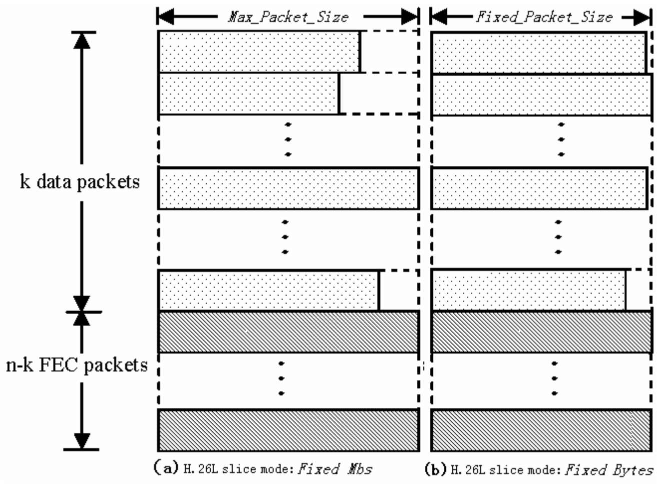
Max_Packet_Size
Fixed_Packet_Size
k data packets
n-k FEC packets
(a) H.26L slice mode: Fixed Mbs
(b) H.26L slice mode: Fixed Bytes

Figure 5. FEC Scheme

We implement the $RS(n,k)$ coding scheme proposed in [16], where $n-k$ redundant packets are generated to protect k data packets. The “k data packets” form a Transmission Group (TG).

In Fig.5 (a), the size of data packet is not fixed in “Fixed_Mbs” mode, however, the $n-k$ FEC packets must be same size (equals to Max_Packet_Size among all the k data packets). So this mode decreases the utilization efficiency of FEC code. As we known, maximizing the efficiency of FEC code can reduce channel-coding rate so as to increase the available source-coding rate and eventually improve the video quality.

By comparison, “Fixed_Bytes” mode achieves higher efficiency of FEC code. Seen from Fig.5 (b), the efficiency is improved approximately by:

$$\frac{Max\_Packet\_Size - Fixed\_Packet\_Size}{Max\_Packet\_Size} \times 100\% \qquad (2)$$

Note that the first $k-1$ data packets approximately have the same size except the last one in “Fixed_Bytes” mode. The payload of one encoded video frame can be calculated by:

$$\begin{aligned} &OneVideofr\,ame\ \_\ Payload\ = \\ &\quad Fixed\ \_\ Packet\ \_\ Size \times (k-1) + PacketSize\ (k) \end{aligned} \qquad (3)$$

IV. Hybrid ARQ Based on Multi-Transmission Stages

Though FEC scheme can improve the performance for reliability, this improvement is achieved at the expense of additional bandwidth. Typical FEC schemes are stationary and must be implemented to guarantee a certain QoS requirement for the worst case, where unnecessary bandwidth overhead is wasted when the channel is in good state. In order to overcome this drawback, hybrid ARQ schemes have been developed.

By numerical experiments, we find some ARQ frames will be corrupted or lost in a bad channel state. In this case, conventional hybrid ARQ scheme still indicate transmitter to send useless FEC packets to receiver. Seen from Fig.6, all the data packets are received successfully, but transmitter ignores this fact because of failing to receive the ARQ frame. To solve the problem, we introduce a RACK (RequestACK) control frame by modifying IEEE 802.11b MAC protocol. Transmitter uses RACK to request receiver to feedback ACK immediately after having transmitted all data packets. If ACK is lost or corrupted, only a RACK frame is retransmitted. In this case, the introducing of RACK can decrease ETE delay and save bandwidth because: (1) the size of control frame (RACK or ACK) is much smaller than that of data frame (data packet or FEC packet), and (2) SIFS (Short Inter-frame Space, used by control frame) is shorter than DIFS (Distributed Inter-frame Space, used by data or FEC frame) in IEEE802.11 MAC protocol.

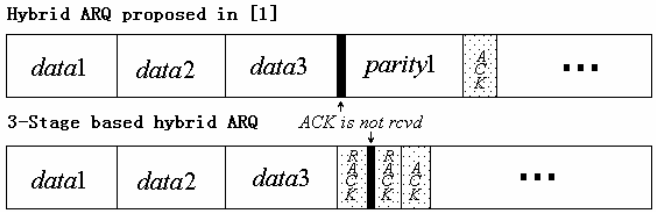

Figure 6. Comparison of two schemes handling ACK loss

Though the hybrid ARQ algorithm in paper [1] can efficiently reduces the amount of ACKs, but receiver must be informed how many data packets in a Transmission Group (TG) in advance. So the scheme requires the number of packets in TG be fixed. In section III, we packetize H.26L video frame into k data packets forming a TG. If *Fixed_Bytes* mode is chosen, different TG will have different packet quantities, for example, intra-frame has more data than P frame and B frame. In this case, algorithm in [1] is not suitable.

We also find another problem. Considering burst arrives in a bad channel state, a TG can’t be received completely before its deadline arrives. Neither can the next TG possibly. It is well known that many corrupted video frames are not as good as few intact frames on the assumption that their payloads be the same. So it would be better to terminate the transmission of current TG and save the bandwidth for another TG, if its destiny of failure can be foreseen.

To solve the two problems mentioned above, transmitter uses RACK to request receiver to feedback information. By piggybacking in RACK, receiver also can be informed how many packets should be received in current TG. In multi-stages based hybrid ARQ algorithm, RACK will collect feedback information for transmitter to make the following decisions: (1) whether or not the next transmission stage is needed, (2) whether or not current transmission should be terminated and declared failure, and (3) whether or not encoder should perform frame skipping.

By numerical experiments, the selection of "three stages" is a compromise between delay boundaries and delivery reliability, but we do not exclude the possibility that a "higher stages" algorithm show better performance results. Since the end of stage three implies the end of transmitting a TG, the feedback information is no use in stage three, so RACK is only applied in the end of stage one or stage two.

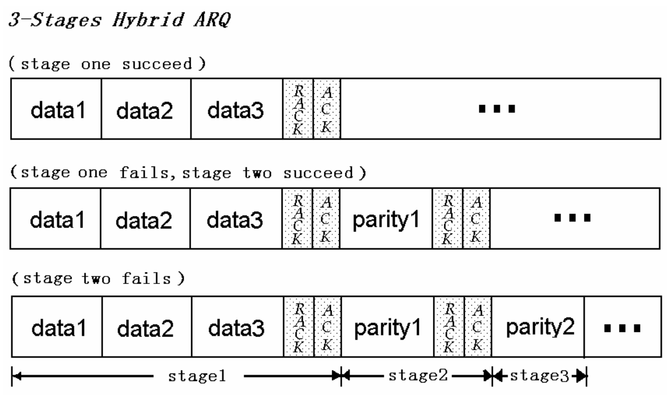

Figure 7. Schematic representation 3-stage Hybrid ARQ

First, a video frame is divided into k data packets forming a TG, and n-k FEC packets are generated based on the TG. Then all data packets are transmitted before the transmitter stops to transmit RACK and waits for receiver's acknowledgement. Receiver will piggyback the number of lost packets with ACK to invoke "Stage Two". This process is called "Stage One". The other two transmission stages will supply FEC packets needed. All the three stages are activated in succession to transmit each video frame until one of the following four events occurs: 1) an ACK from the receiver arrives and informs transmitter that all the packets needed in current stage have been received, which implies success in transmission, 2) an ACK from the receiver arrives and indicates the number of lost packets exceeds FEC_Threshold, then the transmission of TG will be terminated, 3) transmitter accomplishes "Stage Three", which implies success in transmission, 4) the deadline of current frame reaches, then the transmission of TG will be terminated. Our 3-stages hybrid ARQ scheme is schematically shown in Fig.7. According to time flow, the transmission processing is divided into five steps, namely "1.Transmitter high layer handle H.26L video streams", "2.Transmitter MAC layer handle H.26L video streams", "3.Receiver handle H.26L packets", "4.Transmitter MAC layer handle H.26L ACK" and "5.Receiver decode H.26L video". More specifically, this transmission policy works as follows:

Step1:

Transmitter High Layer Handle H.26L video streams

```
While (not End_of_video  &&
   Next_frame_encode_time arrives) {
        If (Frame_Skipping is needed)
                 frame_number ++;
        Select Silce_Mode;
                 /* 1:Fixed_Mbs,  2:Fixed_Bytes*/
        Packetize current video frame into
                   k packets forming a TG;
        FEC_encoding(TG, n, k);
        Enqueue TG to MAC queue;
}
```

Step2:

Transmitter MAC Layer Handle H.26L video streams

```
While (Reserve_channel_successfully) {
     Dequeue(Head (Q));
     If ( Packet_Format = H26L video ){
        Expected_Frame = NONE;
        Switch (current transmission status ).
            Case "During The Stage-1":
                 Continue Transmit k data packets of TG;
            Case "During The Stage-2":
                 Continue to Transmit parity packets
                            according to ACK of stage-1;
            Case "During Stage-3" :
                 Continue to Transmit parity packets
                            according to ACK of stage-2;
            Case "End of Transmission Stage-1 or Stage-2":
                 Next Frame immediately = H26L_RACK;
                 Expected frame = H26L_ACK;
            Case "End of Transmission Stage-3" :
                 Continue to transmit the next TG;
     }else { Expected_Frame = ACK; /*data packet*/}
}
```

Step3:

Receiver Handle H.26L Packets

```
If ( RcvPacket_Format = H26L video ){
        RcvPacket_Num ++;
        Buffer();
}
If (RcvPacket_Format =  H26L_RACK) {
        Respond frame = ACK;
        Get Packet_Num_of_TG from RACK;
        RequiredPacketNum =
           Packet_Num_of_TG  - RcvPacket_Num;
        Write ACK frame;
}
```

Step 4:

```
Transmitter MAC Layer Handle H.26L ACK
Get RequiredPacketNum from ACK;
If (ACK is initiated by Stage one)
    If (RequiredPacketNum = 0)
        Continue to transmit the next TG;
    Else if (RequiredPacketNum <
            Stage-2_FEC_Threshold)
        Activate Stage two;
    Else {
        Abort Current TG;
        Continue to transmit the next TG; }
If (ACK is initiated by Stage two)
    If (RequiredPacketNum = 0)
        Continue to transmit the next TG;
    Else if (RequiredPacketNum <
            Stage-3_FEC_Threshold)
        Activate Stage two;
    Else {
        Abort Current TG;
        Continue to transmit the next TG; }
```

Step 5:

```
Receiver Decode H.26L video
If (All the packets of TG are rcvd) {
        FEC_dencoding( );
        Decode intact frame and Display;
}else if (Playback  Deadline arrives
        || Abort Current TG){
        Decode corrupted frame with
                        Error_Concealment;
    }
```

V. Conditional Frames Skip Algorithm Based on Feedback Information of Three Stages

The transmitter keeps a deadline timer for each video frame. Deadline timer is activated when the first "intra-frame" is received successfully. Let B be the number of backlogged frames, IFS be the time quanta of inter-frame space, and FD (frame delay) be the interval that a frame is transmitted and expected to be received. Then FD can be calculated according to (4):

$$FD = IFS \times \mathrm{B} \qquad (4)$$

We can further calculate the deadline of a video frame according to (5):

$$frame\_deadline = frame\_encoding\_time + FD \qquad (5)$$

Before encoding and transmitting the n^{th} frame, the transmitter is expected to receive the feedback information of $(n-B)^{th}$ frame. The feedback information include: (1) the number of packets unsuccessfully transmitted by video server, and (2) the number of packets unacknowledged. The packets mentioned here belong to the same video frame. When its $frame\_deadline$ arrives, "*Frame_Skipping_Factor*" should be calculated according to (6):

$$\begin{aligned} Frame\_Skipping\_Factor = {} & Unsccessful\_Packet\_Num \\ & - Unacknowledged\_Packet\_Num \times p \end{aligned} \qquad (6)$$

If the value of "*Frame_Skipping_Factor*" exceeds *Frame_Skipping_Threshold*, the video encoder will skip next frame, and lengthen the time quanta of an IFS for the deadline of next transmission. If frame skipping policy is adopted, current frame's transmission should not be terminated to keep video frame intact. The conditional frame skipping policy works as follows:

Algorithm for conditional frames skip algorithm based on feedback information of three stages:

```
Conditional Frame Skipping Algorithm Based on
Feedback Information of 3-Stages
If (the first "I frame" is transmitted successfully)
    Activate Deadline Timer;
If (Deadline_of_current_video_frame arrives)
  If (current_video_frame is transmitted successfully)
      Next_Frame_Deadline =
          Current_Simulation_Time + Frame_Interval;
  else {
    Switch (current transmission status )
      Case "During The Stage-3":
      Unsuccessful_Packet_Num =
          Unsuccessful FEC packet num during Stage-2;
      Unacknowledged _Packet_Num =
          FEC packet num transmitted during Stage-3;
      Case "During The Stage-2":
      Unsuccessful_Packet_Num =
          Unsuccessful FEC packet num during Stage-1;
      Unacknowledged _Packet_Num =
          FEC packet num transmitted during Stage-2;
      Case "During Stage-1" :
      Unsuccessful_Packet_Num =
          Data packet num of current TG;
      Unacknowledged_Packet_Num =
          Data packet num transmitted during Stage-1;
    End of Switch
    Estimate packet transmission success ratio $p$;
    If (Unsuccessful_Packet_Num −
        Unacknowledged _Packet_Num × $p$)
                > Frame_Skipping_Threshold {
        Inform encoder to skip next video frame;
        Next_Frame_Deadline =
          Current Simulation Time + Frame_Interval × 2;}
    else Next_Frame_Deadline =
          Current Simulation Time + Frame_Interval;
  }
```

VI. The Reference Frame Selecting And Intra-Frame Refreshing Algorithms Based on Feedback Information of Three Stages

After a frame is encoded, its reference frame information is stored. As mentioned in section V, before encoding the n^{th} frame, the encoder is expected to receive the feedback information of the $(n-B)^{th}$ frame. If encoder is informed that the $(n-B)^{th}$ frame is corrupted in receiver, it will mark the $(n-B)^{th}$ frame can't be referred. In succession, the encoder will check the reference frame information of the $(n-B+1)^{th}$ frame. If the $(n-B+1)^{th}$ frame refers to the $(n-B)^{th}$ frame, the error has possibly propagated to the $(n-B+1)^{th}$ frame. As a result, the $(n-B+1)^{th}$ frame is also marked as non-reference frame. Let the number of reference frames specified to inter-frame be $N_{multpred}$, the above checking process will be performed for $N_{multpred}$ times when encoding a frame.

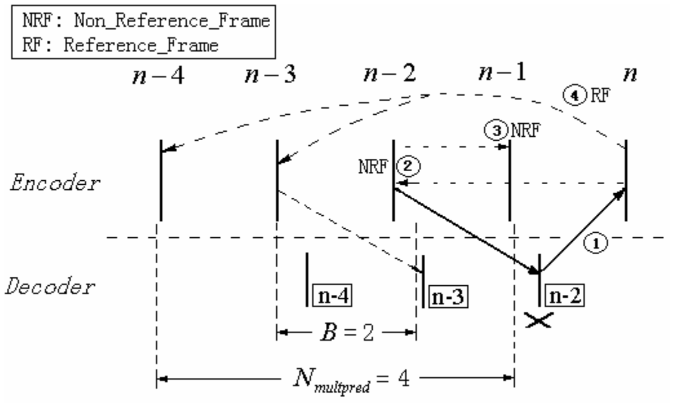

Figure 8. Selection for the reference frames

If only one reference frame is intact, inter-coding can be implemented. The reference frame selecting algorithm is shown in the Fig.8, where B is 2 and $N_{multpred}$ is 4. Before encoding the n^{th} frame, the feedback information of the $(n-2)^{th}$ frame indicates it was damaged (step 1), and the $(n-2)^{th}$ frame is marked as non-reference frame (step 2). So is the $(n-1)^{th}$ frame since it refers to the $(n-2)^{th}$ frame (step 3). The n^{th} frame only can take the $(n-3)^{th}$ frame or the $(n-4)^{th}$ frame as reference (step 4). Seen from Fig.8, the corruption of the $(n-2)^{th}$ frame only influences the $(n-1)^{th}$ frame, so the error is prevented from propagating efficiently.

If all the specified reference frames are corrupted, the frame must be encoded in intra-mode. Because intra-frame coding generates more bit quanta than inter-frame coding, its transmission time (equals to FD in expectation) tends to be longer. So its deadline should be lengthened. To achieve this goal, the next frame must be skipped, as can be seen from Fig.9. The transmission of intra-frame will be lengthened time quanta of "IFS". This effort can guarantee that intra-frame be received as perfectly as possible, so the following frames can refer to this intact intra-frame.

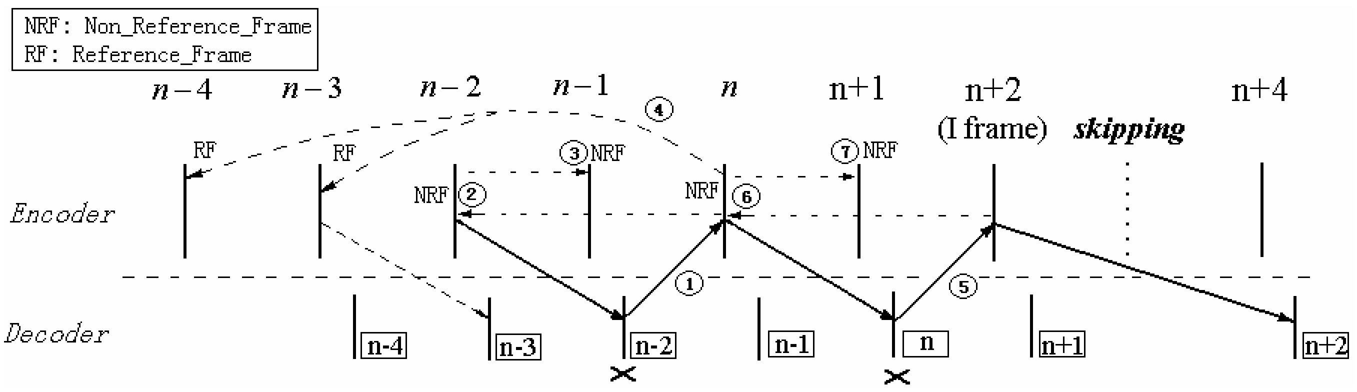

Figure 9. Frame skipping because all the reference frames are corrupted

VII. Simulation And Analysis

In this paper, ON-OFF model is implemented in application layer to generate background traffic. State ON represents the interference station is in "active" state, while state OFF represents a "close" state. Total background load can be calculated based on the traffic parameters showed in Table II, where traffic level is classified into three types: (a) Heavy; (b) Medium; (c) Light. In the case of severe unreliable channel causing heavy packet loss, simulation results can intensively demonstrate the effectiveness of our proposed scheme.

TABLE II. Traffic Parameters Configuration

Number of interference stations	7	4	3
Traffic Level	Heavy	Medium	Light
AIFS (us)	85	85	85
ON (s)	0.01	0.01	0.01
OFF (s)	0.01	0.01	0.01
Average Packet Inter-arrival time (s)	0.006	0.006	0.006
Average Packet length (bytes)	1024	1024	1024
Data Rate (Mbps)	1	1	1
Total background load (Mbps)	4.7788	2.731	2.048

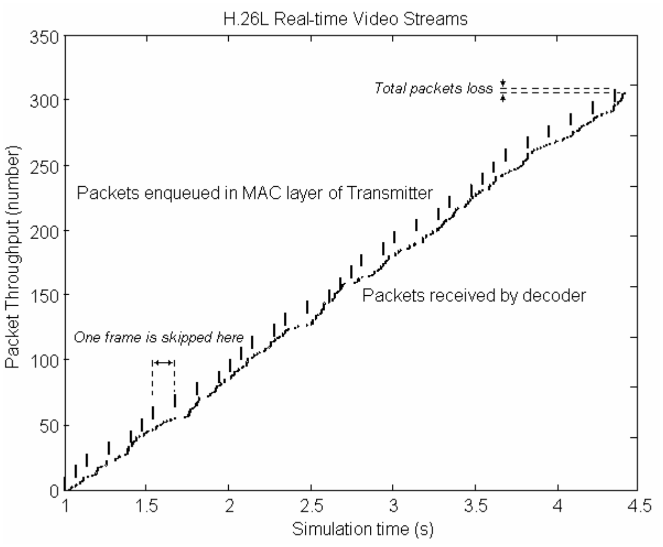

Figure 10. Num of packet sent and received using our scheme

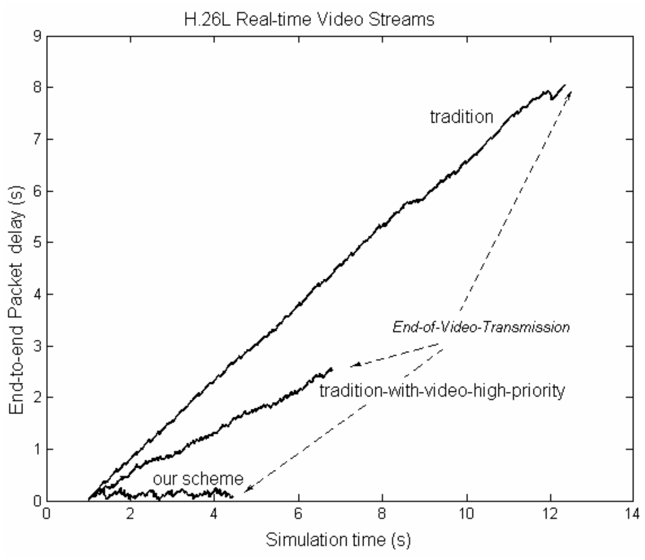

Figure 11. Comparison of ETE packet delay

Test video sequence is *Forman* that was coded in QCIF format (176×144 pixels/frame) at a temporal resolution of 15 fps (frames/s). The first frame is intra-coded and the remaining frames are inter-coded.

Fig.10 shows the number of packet sent and received using our scheme. Horizontal distances of Fig.2, Fig.3 and Fig.10 between two curves represent ETE delay, which respectively correspond to three ETE packet delay curves showed in Fig.11. In addition, the vertical distance between "sent" and "received" curves denotes the amount of bits backlogged in MAC queue of transmitter.

Clearly, our scheme can achieve both guaranteeing real-time service and reducing the buffer occupancy.

Table III shows the comparison of average throughput, ETE delay and PSNR for tradition 802.11b MAC scheme under different priority, traffic level, and using our proposed scheme. Although the PSNR is similar, the ETE delay decrease is up to about 3130 ms with priority mechanism, and up to 1270 ms more for our proposed scheme. Table IV shows the comparison of average throughput, ETE delay and PSNR for hybrid ARQ scheme proposed in paper [1] under different traffic level, and using our proposed scheme.

TABLE III. COMPARISON OF AVERAGE THROUGHPUT, ETE DELAY AND PSNR FOR TRADITION 802.11B MAC SCHEME UNDER DIFFERENT PRIORITY, TRAFFIC LEVEL, AND USING OUR PROPOSED SCHEME

Scheme	Tradition 802.11b		Tradition 802.11b		Tradition 802.11b		Our scheme	
QoS Differentiation:	*BE*	*video*	*BE*	*video*	*BE*	*video*	*BE*	*video*
AIFS_slot_num	—	—	2	2	5	2	5	2
Traffic Level	Clean Channel		Light		Light		Light	
Average Throughput (kpbs)	121.2		32.47		68.45		87.78	
PSNR(dB)	35.91		28.58		35.91		32.14	
ETE Delay (ms)	83		4530		1400		130	

TABLE IV. COMPARISON OF AVERAGE TRHOUGHPUT, ETE DELAY AND PSNR FOR HYBRID ARQ SHCEME PROPOSED IN PAPER [1] UNDER DIFFERENT TRAFFIC LEVEL, AND USING OUR PROPOSED SCHEME

Abbreviation	Scheme A: Hybrid ARQ algorithm proposed in paper [1]					
	Scheme B: 3-Stages based Hybrid ARQ with Conditional Frames Skipping scheme					
Scheme	A	B	A	B	A	B
Traffic Level	Heavy	Heavy	Medium	Medium	Light	Light
PSNR(dB)	20.13	31.24	19.63	31.49	20.66	32.14
ETE Delay (ms)	186	325	167	224	104	130
Average Throughput (kpbs)	64	57.92	77.34	73.33	82.01	87.78
BE Traffic Throughput (kpbs)	726	706.2	758.6	757.8	766	767.5
Total Throughput (kpbs)	790	764.12	853.94	831.13	848.01	845.28

Although ETE delay of our proposed scheme is higher about 30 ms than that of hybrid ARQ scheme proposed [1], PSNR performance of our proposed scheme is much better, as we can seen from Fig.12. It shows compared result of PSNR for test sequence Foreman. Our scheme achieves an improvement of more than 10dB compared with hybrid ARQ scheme proposed in [1] under same condition. The improvement is due to three points: 1) save bandwidth by reducing useless amount of FEC packets using RACK; 2) conditional frame skipping strategy adapts video coding rate to fluctuations of channel bandwidth; and 3) keep transmitted video frame as intact as possible by having to compromise little delay boundaries.

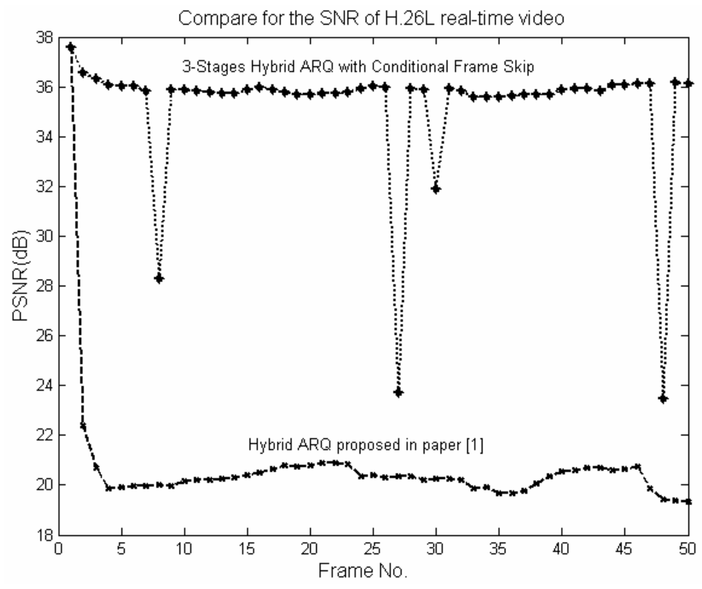

Figure 12. Comparison of PSNR

Because the dramatic improvement of PSNR in multi-stages hybrid ARQ scheme with conditional frame skipping is at the cost of ETE delay's increase. In the case that delay performance is more critical than restored picture quality for real-time video service, we can displace conditional frame skipping algorithm with reference frame selecting and intra-frame refreshing algorithms.

TABLE V. COMPARISON OF PSNR AND ETE DELAY FOR HYBRID ARQ SHCEME PROPOSED IN PAPER [1] , AND USING OUR PROPOSED SCHEME

Abbreviation				
Scheme A: Hybrid ARQ algorithm proposed in paper [1]				
Scheme B: 3-Stages Hybrid ARQ with Conditional Frame Skipping scheme				
Scheme C: 3-Stages Hybrid ARQ with reference frame selecting and Intra frame refreshment algorithms				
Scheme D: 3-Stages Hybrid ARQ with Intra frame refreshment algorithm				
Scheme	A	B	C	D
Traffic Level	Light	Light	Light	Light
PSNR(dB)	21.78	33.27	29.96	26.77
ETE Delay (ms)	105	132	105	105

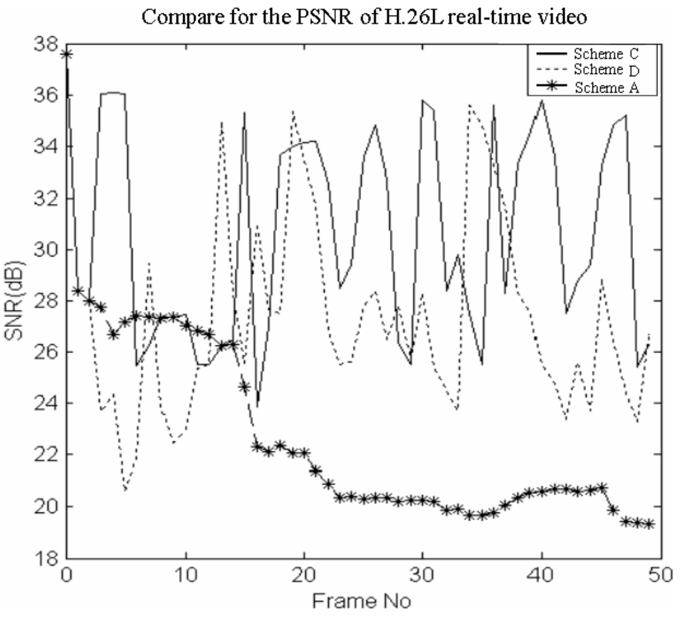

Figure 13 . Comparison of PSNR for Scheme A, C, and D

Table V lists the four schemes involved in this paper and their abbreviations. It also shows the comparison of PSNR and ETE delay for these schemes. It can be seen, scheme B has highest PSNR, but its delay performance is worst, which has already analyzed above. Scheme C has better delay performance, and the average PSNR is about 8dB higher than scheme A. Since background load is heavy, video packets are lost seriously during the experiment. The Scheme A doesn't take any measure to prevent error propagation, so the PSNR of reconstructed image decreases fast, and its subjective quality is poor.

Scheme D handles all corrupted frames by intra-frame refreshment algorithm, which causes large number of intra-frames. So average PSNR of scheme D is 3.2dB lower than that of scheme C. PSNR curves of scheme A, C, and D are depicted in Fig.13.

VIII. CONCLUSION AND FUTURE WORK

In this paper, we address the problem of real-time video streaming over WLAN. Investigating H.26L real-time video transmission over wireless IEEE 802.11b LANs, we present a novel hybrid Automatic Repeat reQuest (ARQ) with conditional frame skipping and reference frame selecting algorithms that efficiently combines forward error control (FEC) coding with the ARQ protocol based on multiple transmission stages. The scheme includes: (1) multi-stages algorithm with corresponding ARQ feedback information used to determine how many transmission stages should be activated; (2) conditional frame skipping algorithm based on accumulated feedback information; and (3) reference frame selecting and intra frame refreshing algorithms based on feedback information. These techniques explain that the proposed scheme can achieve higher throughput, lower delay, and better PSNR, as observed in simulation presented in this paper.

Unequal loss protection for scalable video transmission is proposed in many papers [19, 20]. Similar to these ideas, future work can specify different priority for different transmission stages, for example, stage-2 and stage-3 are more urgent than stage-1. We will find a way to valuate the cost of different priorities to guarantee fairness so as to compare the schemes with and without this prioritization mechanism.

ACKNOWLEDGMENTS

This work was supported by the NSF of China under Grant 60072048, and the Doctoral Program Fund under grant 20010561007.

REFERENCES

[1] 1 A. Majumdar, D.G. Sachs, I. Kozintsev, K. Ramchandran, and M. Yeung, “Multicast and unicast real-time video streaming over WLAN,” *IEEE Trans. CSVT.*, vol. 12, pp. 524-534, June 2002.

[2] 2 Aramvith, C.-W. Lin, S. Roy, and M.-T. Sun, “Wireless Video Transport Using Conditional Retransmission and Low-Delay Interleaving,” *IEEE Trans. CSVT.*, vol. 12, pp. 558-565, June 2002.

[3] ITU-T SG16. *Draft Call for Proposals for H.26L Video Coding[s]*. February 1998.

[4] BJONTEGAARD G. *H.26L Test Model Long Term Number 9(TML-9) Draft0[s]*. July 2002.

[5] ITU-T Recommendation H263. Video Coding for Low Bit Rate Communication[S]. February 1998

[6] Imad Aad and Claude Castelluccia, “Differentiation mechanisms for IEEE 802.11,” in *Proceedings of IEEE Infocom 2001*, Anchorage - Alaska, April 2001.

[7] Michael Barry, Andrew T. Campbell, and Andras Veres, “Distributed control algorithms for service differentiation in wireless packet networks,” in *Proceedings of IEEE Infocom 2001*, Anchorage - Alaska, April 2001.

[8] Jing-Yuan Yeh, C. Chen, “Support of multimedia services with the IEEE 802.11 MAC protocol,” in *ICC 2002. IEEE International Conference*, vol: 1, 2002, Page(s): 600 –604

[9] Brain P. Crow, Indra Widjaja, Jeong Geun Kim, and Prescott T.Sakai, “IEEE 802.11 wireless local area network,” *IEEE Communication magazine*, September 1997.

[10] Y.Wang, Stephan Wenger, Jiangtao Wen, and Aggelos K. Katsaggelos, “Error Resilient Video Coding Techniques,” *IEEE Signal Processing magzine* , vol. 86, pp. 61 – 82, July 2000.

[11] Hang Liu a, Hairuo Ma a, Magda El Zarki a and Sanjay Gupta, "Error control schemes for networks: An overview", *Mobile Networks and Applications 2* (1997) 167–182

[12] H. Ma and M. El Zarki. Broadcast/Multicast MPEG-2 Video over Broadband Fixed Wireless Access Networks. *IEEE Network Magazine*, Vol.13 (6), Nov./Dec. 1998, pp. 80-93

[13] Q. Zhang and S. A. Kassam, “Hybrid ARQ with Selective Combining for Fading Channels,” *IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS*, VOL. 17, pp. 867-880, MAY 1999

[14] Chen Jin, Wang Jinlong, “A Novel Selective Repeat Stop-Wait ARQ for Half-Duplex Channels,” *Proceedings of IEEE TENCON’02*

[15] R. Marasli, P. D. Amer, and P. T. Conrad, “Retransmission-based partially reliable transport service: An analytic model,” in *Proc. IEEE Infocom, 1996*, pp. 621–628.

[16] L. Rizzo, “Effective erasure codes for reliable computer communication protocols,” *ACM Computer Communication Review*, vol. 27, pp. 24-36, April 1997

[17] C. Papadopoulos and G. M. Parulkar, “Retransmission-based error control for continuous media applications,” in *Proc. NOSSDAV*, 1996.

[18] D. Xu, B. Li, and K. Nahrstedt ”QoS-Directed Error Control of Video Multicast in Wireless Networks,”*Proc. IEEE ICCCN99.*

[19] Q. Zhang, W. Zhu, and Y.-Q. Zhang, “Network-adaptive rate control and unequal loss protection with TCP-friendly protocol for scalable video over Internet,” *special issue selected from IEEE ICME’00* on Multimedia Communications Journal of VLSI Signal Processing - System for Signal, Image and Video Technology, 2001.

[20] Wenwu Zhu, Qian Zhang, and Ya-Qin Zhang, “Network-adaptive rate control with unequal loss protection for scalable video over Internet”, *ISCAS 2001*. vol. 5 , pp. 109 -112

[21] K. Stuhlmuller, N. Farber, M. Link, and B. Girod, “Analysis of video transmission over lossy channels,” *IEEE Journal on Selected Areas in Communications*, vol. 18, pp. 1012 – 1032, June 2000.

Min Chen was born in LinChuan, China, on Dec. 1980. He received the B.S. and the M.S. degree in electronic & communication engineering from South China University of Technology, in 1999 and 2001, respectively. He is currently pursuing the Ph.D. degree in communication and information system in South China University of Technology.

Gang Wei was born in January 1963. He received the B.Sc., M.Sc., and Ph.D. degrees in 1984, 1987, and 1990, respectively, from Tsinghua University and South China University of Technology. He was a Visiting Scholar to University of Southern California from June 1997 to June 1998. He is currently a Professor with the Department of Electronic Engineering, South China University of Technology. He is a Committee Member of the National Natural Science Foundation of China. His research interests are signal processing and personal communications.