A Framework for Robust and Scalable Audio Streaming

Ye Wang, Wendong Huang, Jari Korhonen
School of Computing, National University of Singapore
{wangye, huangwd, jari}@comp.nus.edu.sg

ABSTRACT

We propose a framework to achieve bandwidth efficient, error robust and bitrate scalable audio streaming. Our approach is compatible with most audio compression format. The main contributions of this paper include: 1) the proposal of a Multi-Stage Interleaving (MSI) strategy which translates packet loss into loss of separate frequency components that are less perceptually significant; and 2) the design of a Layered Unequal-Sized Packetization (LUSP) scheme which enables bitrate scalability and prioritized packet transmission. The combination of the proposed MSI and LUSP allows the use of a set of simple yet effective methods of error concealment in the compressed domain. Our approach offers significant advantages over existing methods in terms of memory consumption (a savings of over 40 times in the sample MP3 implementation), and computational complexity, which are critical issues for battery-powered small devices.

Categories and Subject Descriptors

C. 3. [**Special-Purpose and Application-based Systems**] Signal Processing Systems

H.5.5. [**Sound and Music Computing**]: Signal Analysis, Synthesis and Processing, Systems

General Terms

Algorithms, Performance, Reliability, Experimentation, Human Factors

Keywords

Multi-stage Interleaving (MSI), Layered Unequal-Sized Packetization (LUSP), Compressed Domain Processing, Streaming, Robustness, Scalability

1. INTRODUCTION

Streaming audio over the Internet is a popular application, and currently two proprietary systems, Real™ and Windows Media Audio (WMA™), dominate the market. Standardized technology such as MPEG-1 layer 3 (MP3), MPEG-2 or MPEG-4 Advanced Audio Coding (AAC) [1] is in widespread use for music downloading for storage and playback and is gaining in popularity for audio streaming [9]. As the technology becomes mature, there are increasingly more standardized and proprietary audio compression formats entering the market. Most of the new formats are designed to achieve not only compression efficiency, but also bitrate scalability and error robustness, which are important requirements for the new application scenario - streaming audio content over heterogeneous IP networks consisting of both wired and wireless components.

To meet the requirement of bitrate scalability, various scalable codecs have been proposed [2]][3][4]. Their solutions all require the adoption of a new audio format, which is neither an easy task nor a desirable feature for both content providers and end users. To meet the requirement of error robustness, various methods have been proposed [5][6]. There are a few works which address both bitrate scalability and error robustness [4][7]. These existing solutions generally involve a new scalable audio codec. As yet, few have addressed the question whether it is feasible to achieve error robust and bitrate scalable streaming using existing single-layer audio formats such as MP3 and AAC. It is an appealing proposal to have a single content format for storage, downloading and streaming.

How to use standard single-layer audio format for *bandwidth efficient, error robust and bitrate scalable* streaming services is the question we seek to address in this paper.

Based on our initial survey of content providers, device manufacturers and end users, the market desires a few (ideally one) dominant and open audio formats. Too many different formats can only result in market fragmentation and user confusion. Before proposing yet another audio format for streaming services, we have to consider the following questions: 1) is it convenient for content providers to manage content of different formats in addition to expensive format conversion? 2) Is it feasible and cost effective for device manufacturers to implement many different codecs in their devices, especially small devices? 3) Do end users really want to use many different audio formats to enjoy their music or listen to broadcast? Our scheme is designed with due consideration of these important questions.

Although the proposed scheme can be implemented with most existing audio codec, we have implemented our scheme with the MP3 format for the sake of proof of concept, due to its popularity.

For storage and downloading, compression efficiency is the most important concern, and error robustness and bitrate scalability are irrelevant issues. This explains why the MP3 format, which is both error-sensitive and non-scalable, has maintained its popularity – it perfectly satisfies the user's need to enjoy music with sufficient quality, and to store and exchange music files over the Internet efficiently. In terms of compression efficiency, the research community has made tremendous effort in the past decade after the standardization of MP3 in 1992, especially within the MPEG framework, and has achieved noticeable progress. However, this progress is largely an optimization of the perceptual coding paradigm, in comparison to MP3 which represents the

MM'04, October 10–16, 2004, New York, New York, USA.

paradigm shift - the introduction of perceptual audio coding technology.

According to our observation, we are approaching the theoretical limit of the paradigm of perceptual audio coding in terms of coding efficiency. It has become an increasingly difficult task to achieve any major breakthrough, and improvements within the current paradigm will have to be incremental. Therefore, instead of developing more audio formats, we feel that a more exciting research frontier now is to build systems which enable appealing applications and services, leveraging mature technologies.

The performance of a scalable codec is upper-bounded by a single-layer codec optimized for a specific bitrate. Therefore, our scheme merely tries to maintain the single-layer codec's compression efficiency. An additional rationale for us to choose a single-layer audio format in our current implementation is that a scalable codec usually has much higher computational complexity compared with a single-layer codec [2]. For battery-powered small devices, the computational complexity is an important design consideration.

Interleaving is a key technique in our framework due to its advantage that it does not increase the bandwidth requirement of a stream [6]. In the case of packet loss, it can simplify error concealment significantly and is therefore a good option for streaming audio to small mobile devices.

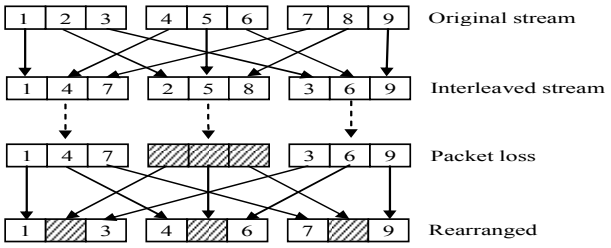

Figure 1. Principle of interleaving

The general principle of interleaving is to separate a large chunk of error into several small sections of errors as shown in Figure 1. Errors of shorter duration are computationally less expensive to conceal with satisfactory perceptual quality. To increase transmission throughput, an audio packet usually contains several data units, which are codec dependent frames. In this scenario, a lost packet can be translated into a loss of several separate audio frames. However, traditional single-stage interleaving performs well only if the audio frame is very short, e.g., 5 ms. According to our experience, a loss of an MP3 frame (~26 ms) is still too large to be concealed satisfactorily, especially if computational complexity is constrained. Moreover, traditional interleaving is not robust against burst packet loss.

The second technique relevant to our scheme is error concealment. Traditional error concealment methods are usually performed in the spectral domain, mostly the modified discrete cosine transform (MDCT) domain [8][9]. The major problems of those approaches are: 1) they require modification of the decoder, which is not desirable because error concealment is not a mandatory requirement for a standard MPEG audio decoder; 2) they perform error concealment in the MDCT domain, which is a *transform domain representation*, but not a *compressed domain representation.* Error concealment in the MDCT domain is unnecessarily expensive in terms of memory consumption and computational complexity. As a consequence, this type of method is also power hungry, which is a serious drawback for battery-powered small devices.

To give a numerical example, a frame of MP3 stereo audio data in the PCM domain is 4608 (= 576*2*2*2) bytes and every PCM sample is represented in 2 bytes. However, after going through the filterbank, the frequency domain coefficients require a floating point representation (float or double in C language), which is typically represented in 4 bytes (float) or 8 bytes (double). That means we need 18432 (= 576*2*2*8) bytes to store a single frame of data in the MDCT domain, which is a factor of 4 in comparison with PCM data. Prediction in the MDCT domain as suggested in [9] is also computationally expensive. In contrast, if we perform the error concealment operation directly in the *Huffman coded domain*, which we define as the *compressed domain*, we need only a small fraction of memory (~418 bytes in our example) to store an MP3 frame in comparison with the much larger amount (18432 bytes) for a single frame in the MDCT domain approach. The efficiency in memory utilization, together with many other desirable features, makes our solution a very attractive alternative for streaming services to battery-powered small devices. That is, the clear difference between our scheme and the existing schemes is that most existing schemes work in the MDCT domain, while ours works in the quantized MDCT (QMDCT) domain or Huffman coded domain. This distinction is illustrated in Figure 2, with the help of the MP3 codec architecture. We define the compressed domain as the coded representations after the quantization block in the encoder and before the de-quantization block in the decoder as shown in Figure 2. We hope that our definition will clarify the conflicting and confusing definitions regarding *compressed domain processing* in the research community.

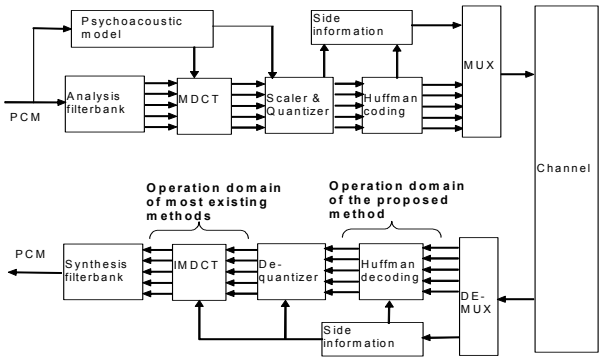

Figure 2. MP3 codec architecture with different error concealment domains

In this paper, we propose an integrated solution to convert an MP3 bitstream into a temporary transportation format which maintains compression efficiency while adding error robustness and bitrate scalability. Due to the advantages – in terms of memory consumption, computational complexity, power consumption, and convenience – that our solution can provide to content providers, equipment manufacturers and end users, it represents a more attractive alternative in comparison with the existing solutions, especially when streaming to small mobile

devices. In essence, we try to build a "plug-in" to solve the mismatch between perceptually coded formats, such as MP3, and error prone channels. Our solution enables content providers to extend streaming services to wireless domains, similar to WMA and RealPlayer on the Internet, using their preferred audio format.

The paper is organized as follows. After this introduction, our conceptual framework and methodology are outlined in Section 2, followed by our current implementation of the system in Section 3. Perceptual evaluation results are presented in Section 4. Discussions are given in Section 5. Finally, Section 6 concludes the paper.

2. CONCEPTUAL FRAMEWORK

Our framework is a combination of a novel Multi-Stage Interleaving (MSI), Layered Unequal-Sized Packetization (LUSP), and receiver-based compressed domain error concealment.

We first remove the dependency between MP3 audio frames to prevent inter-frame error propagation, and then exploit the inherent data divisions in the compressed bitstream according to their perceptual significance to achieve bitrate scalability.

Our MSI strategy is designed to combat packet loss, especially burst packet loss effectively. The proposed MSI translates a single packet loss and two consecutive packet losses into a loss of a few separate Huffman codes in an MP3 frame, which can be simply concealed by muting with satisfactory perceptual results. According to our experience, muting can only be effective, if the losses are separate individual Huffman codes or QMDCT coefficients - this is especially true for the low frequency components. In the case that a single QMDCT coefficient or Huffman code is lost, more advanced methods such as repetition or interpolation do not yield noticeable improvement over muting. This is verified independently in [16][9].

The MSI pushes the performance of interleaving to its upper limit – with three interleaving stages our scheme can translate a loss of two consecutive packets into a loss of several separate individual QMDCT coefficients in an MP3 frame. This strategy allows us to use the simplest error concealment directly in the compressed domain in the case of packet loss. A further advantage of this strategy is that it allows the easy re-assembling of the MP3 bitstream from the de-packetizing buffer for storage purpose, which enables the use of standard MP3 players for streaming applications. This is because our scheme works in the Huffman coded domain and requires only one single step, namely the MP3 bitstream re-assembler, to accomplish the conversion as opposed to the existing methods which have to go through almost the entire encoding process to bring the recovered MDCT coefficients (by existing error concealment methods) back to the MP3 bitstream. Combining all the advantages it can offer, our scheme is a much more attractive alternative for deployment for large scale audio streaming services, especially to battery-powered small devices such as mobile phones and solid state MP3 players.

The proposed MSI strategy has a solid psychoacoustic foundation underlying its superior performance. The audibility of frequency response irregularities has been studied in [10], where the overall findings are that peaks are more audible than dips, in the case of frequency domain distortions. This is the rationale behind our simple error concealment – muting. Furthermore, small dips across the frequency band are less detectable in comparison with a single but deep dip [11]. This is the foundation for designing our MSI.

Essentially, our scheme achieves error robustness and bitrate scalability at the expense of increased delay. Our scheme, therefore, is based on the assumption that delay constraints can be relaxed in the range of a few hundred milliseconds to a few seconds in streaming applications.

3. SYSTEM IMPLEMENTATION

We start with a brief analysis of the MP3 format. An MP3 frame structure is shown in Figure 3, and a constant bitrate MP3 stream employing the bit reservoir technique is shown in Figure 4 [12].

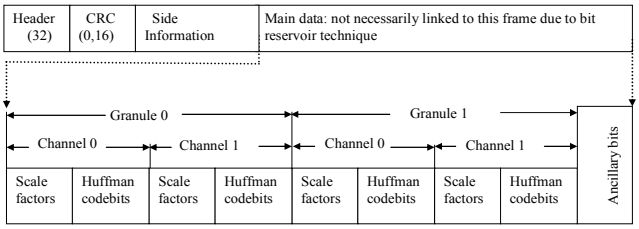

Figure 3. MP3 frame structure

The bit reservoir technique is a smoothing technique which smoothes out bitrate fluctuations. The main data of a frame is typically distributed in two adjacent frames, or in three adjacent frames with peak demand, which results in an inter-frame dependency. For example, if frame 2 in Figure 4 is lost, the data in both frame 2 and frame 3 cannot be decoded. The main data of frame 2 stored in frame 1 cannot be decoded as well. This kind of error propagation is not desirable in streaming applications.

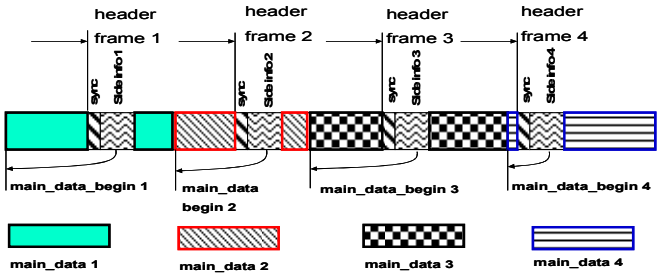

Figure 4. Constant bitrate MP3 stream with bit reservoir technique

Therefore, our first step is to eliminate the inter-frame dependency of the MP3 bitstream, thus making the frame length variable. Then, we split every MP3 frame into three parts, namely the critical data (CRI), the scale factors (SCF) and the QMDCT coefficients, according to their perceptual importance. CRI consists of the frame header and side information, SCF consists of all scale factor data in the main data and QMDCT follows the SCF. QMDCT coefficients may go through an optional pre-processing unit for interleaving (This is discussed in more detail later). Then the three parts are packetized according to different strategies detailed in the next section. Finally, the packets are sent to a scheduler for transmission. The receiver side performs the reverse processing. A block diagram of each process is shown in Figure 5. It should be noted that the receiver architecture can be further optimized if a standard MP3 decoder is not required. In

this case, we can combine the post-processing unit and the MP3 decoder, thus saving the MP3 bitstream re-assembler.

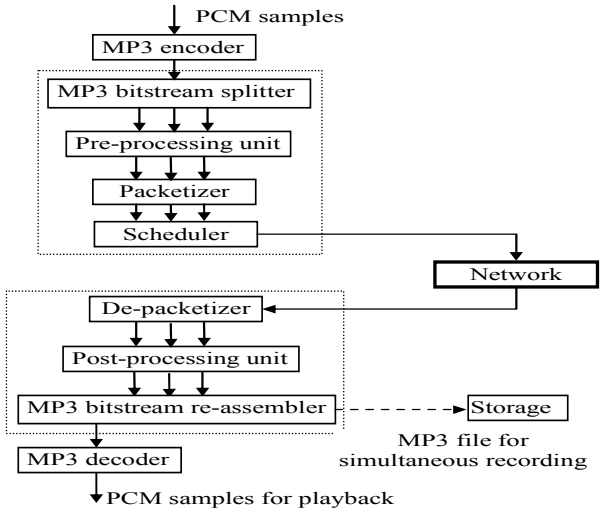

Figure 5. A possible system configuration for live streaming

Figure 5 shows a possible system configuration which employs a standard MP3 decoder and is suitable for live multicasting applications. Alternatively, a configuration which employs a modified MP3 decoder to reduce receiver complexity can be deployed for streaming pre-recorded content for example. The modified MP3 decoder can be directly embedded in small devices such as mobile phones.

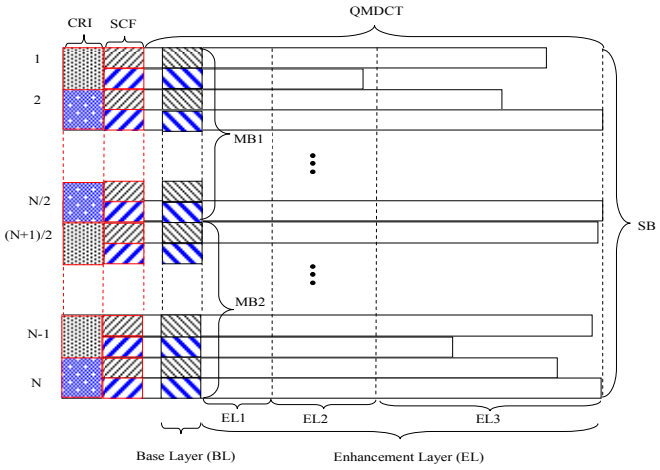

Figure 6a. Structure of the base layer and the enhancement layer of a mono signal, and structure of Macro-Block (MB) and Super-Block (SB). N is the number of MP3 frames in a SB

The advantage of the system in Figure 5 is that the end user just needs a plug-in module described in this paper and a standard MP3 decoder to enjoy the streaming service. In the case of packet loss, our simple error concealment scheme is performed in the post-processing unit before the bitstream is fed to the MP3 decoder. With the help of the bitstream re-assembler, the streaming content can be easily recorded simultaneously for storage purpose.

3.1 At the Sender Side

3.1.1 Bitstream splitter

As shown in Figure 6a, we decompose every MP3 frame into three parts: critical data (CRI), scale factors (SCF) and QMDCT coefficients represented in Huffman codes. This step is quite straightforward.

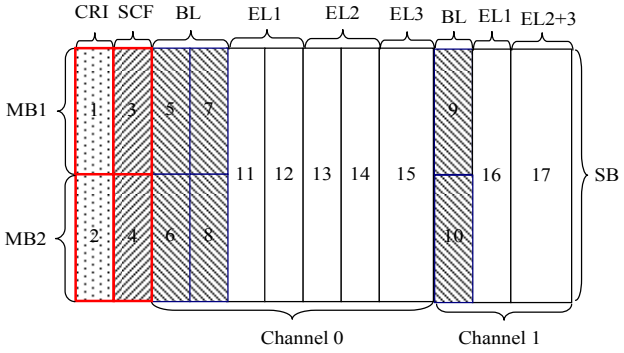

Figure 6b. Structure of MB and SB and the packet sequencing strategy for a joint stereo signal

3.1.2 Pre-processing unit

Pre-processing unit forms elementary packets according to MSI and LUSP principles. It separates the QMDCT coefficients into base layer (BL) and enhancement layers (ELs). Then it rearranges the QMDCT coefficients within each layer according to two larger time frames, namely the macro-block (MB) and super-block (SB).

3.1.2.1 Base layers and enhancement layers

Bitrate scalability is achieved by dividing the QMDCT coefficients into one BL and three ELs. BL, EL1 and EL2 are further split into two sub-layers for increased scalability. The detailed layer structure is shown in Table 1.

Layer structure	Index of QMDCT	Frequency interval (Hz)	Number of QMDCT coefficients
BL1	8-39	269–1493	32
BL2	40-87	1494–3330	48
EL1_1	88-135	3331–5168	48
EL1_2	136-199	5169–7618	64
EL2_1	200-263	7619–10068	64
EL2_2	0-7 & 264-319	0–268 & 10069–12211	64
EL3	320-575	12212–22050	256

Table 1. Layer structure and its corresponding frequency bands

This new layer structure is based on our evaluations of the perceptual significance of different layers and other considerations such as ease of implementation, type of services etc.

The perceptual significance decreases as we go from BL1 up to EL3 (see Figure 6). Furthermore, the bandwidth of the layers also increases from BL to EL as shown in Table 1. That is, the loss of a packet in different layers has very different perceptual impact on audio quality. For example, if we discard the entire EL3, the impairment of audio quality is minimal. In contrast, a loss of a packet in BL1 would cause most serious distortion in comparison with any higher layer.

To provide basic service, it is sufficient to use the base layer (BL1+BL2) only. This bandwidth roughly corresponds to the narrow band (300 – 3400 Hz) speech that we experience with the fixed-line public telephony service today. EL1 corresponds to wideband speech [13], and EL2 corresponds to broadband music signal and sound effects. EL3 can be considered minor enrichment to sound quality, but is irrelevant and can be discarded first in the case of bandwidth constraints.

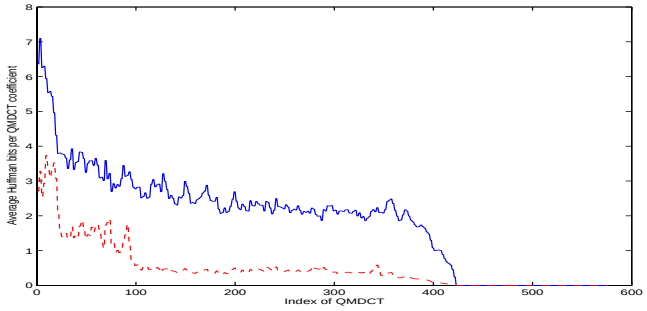

Figure 7. Huffman bits per QMDCT coefficient in an MP3 granule averaged over an entire test song. Solid line represents Channel 0, and dashed line represents Channel 1

It should be noted that the DC component for audio in the QMDCT domain has a different perceptual significance in comparison with its counterpart in video coding. For example, QMDCT coefficients 0-7 represent very low frequency components such as drums. It consumes quite a significant amount of bits as shown in Figure 7, but is not essential for providing basic services. Therefore, we put the lowest eight QMDCT coefficients in the sub-layer EL2_2.

3.1.2.2 Super-block and Macro-blocks

We define two time domain data structures, namely macro-block (MB) and super-block (SB) for multi-stage interleaving and packetization. An SB contains *N* frames or *2N* granules of MP3 data, which are equally divided into two MBs as shown in Figure 6. In our current implementation, *N=64*. MB is for packetizing CRI, SCF and BL. SB is for packetizing ELs. This packetization scheme is similar to Group of Picture (GOP) in hierarchical video encoding.

The different time domain granularity, together with our layer structure in the frequency domain, ensures that we can form small packets for important data and large packets for less important data. This is designed based on the assumption that smaller packets have lower loss rate than larger packets in wireless networks, where considerable losses are due to corruption. Therefore, we use small packets to deliver important data with increased reliability but decreased bandwidth efficiency, and we use large packets to deliver less important data with increased bandwidth efficiency but decreased reliability. With this approach, which can be considered unequal error protection (UEP), we can achieve a better tradeoff between loss and distortion.

For basic service, we can discard all enhancement layers (ELs) data shown as blank rectangles in Figure 6. In doing so, we not only reduce the bandwidth requirement significantly, but also reduce system delay by half, because the two MBs are packetized independently as shown in Figure 6. The SB structure is only relevant for EL. That is, if we want high quality audio, the price to pay is increased delay and bandwidth requirement. Our scheme enables such tradeoff, while the baseline system, which packetizes an MP3 frame into a packet, cannot.

3.1.2.3 Elementary packets

Our data partition and packetization scheme of the joint stereo mode is illustrated in Figure 6b. The numerical index in Figure 6b indicates the sequence of packetization in an SB. That is, the CRI is packetized first, followed by SCF and BL. Finally, the ELs are packetized. In the case of bandwidth constraints, packets can be selectively dropped from the highest enhancement layer downwards.

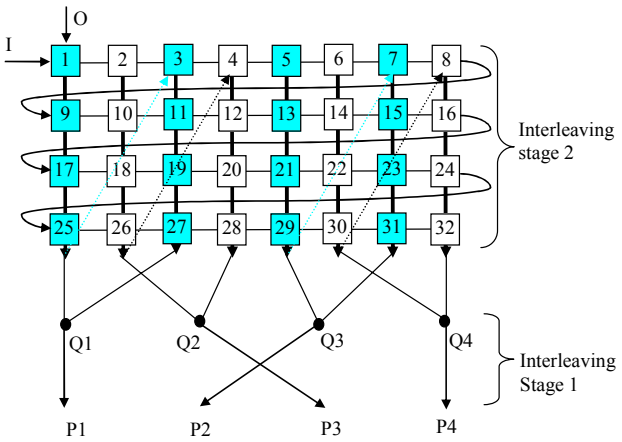

Figure 8. Interleaving critical data from 32 MP3 frames of an MB to form four packets for transmission

In our current implementation, we keep CRI and SCF unchanged and packetize them independently. CRI and SCF are packetized in the same manner as shown in Figure 8. Four packets are formed for MB1 and MB2 respectively. Therefore, eight CRI packets and eight SCF packets are formed. It is clear that a loss of CRI or SCF packet results in a loss of eight separate frames rather than a loss of eight consecutive frames. The later case is significantly more difficult to conceal. Nevertheless, our scheme facilitates various methods to prevent loss of CRI packets and SCF packets. The error recovery is discussed in the receiver side.

We use two-stage interleaving to generate the CRI and SCF packets. With MSI we can translate a loss of two consecutive CRI packets into a loss of separate individual frames, which are significantly simpler to conceal in comparison with a loss of two consecutive frames. This is the rationale for the introduction of MSI instead of employing traditional single-stage interleaving.

We now introduce our MSI and LUSP for the QMDCT coefficients. Figure 9 gives an intuitive illustration of our scheme

for BL1. Every MB is divided into four packets. Every packet contains 64*4 Huffman codes of interleaved MDCT coefficients, which are packed in a zigzag manner as shown in Figure 9. Interleaving stage 1 prevents adjacent Huffman codes from getting lost even if two consecutive packets of base layer data are lost. Interleaving stage 2 translates a single packet loss into a loss of four separate Huffman codes in the set of frames of an MB.

Interleaving stage 3 is optional. This stage affects the way Huffman codes are associated with QMDCT coefficients. The original Huffman code represents two adjacent QMDCT coefficients. And the new Huffman code represents two separate QMDCT coefficients. Stage 3 produces noticeable improvement in sound quality in the case of packet loss over the original Huffman coding scheme in MP3. The reason for such improvement can be found in [11].

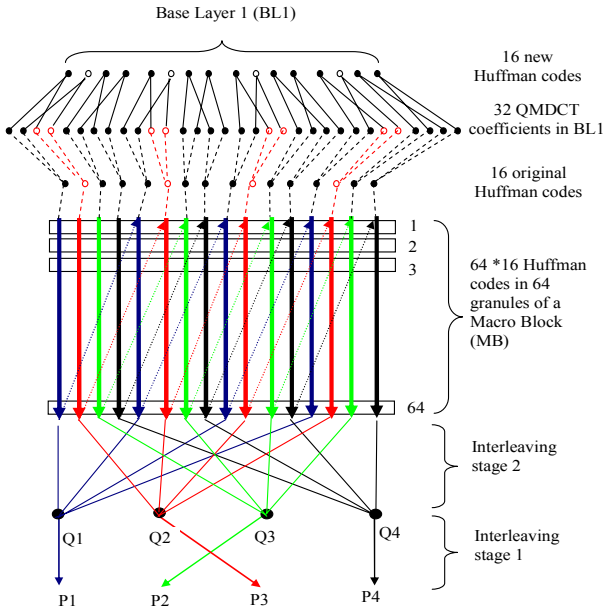

Figure 9. Multi-stage interleaving and packetization scheme for BL1

As an example, a loss of two consecutive packets would result in a loss of eight separate Huffman codes, which represent eight pairs of QMDCT coefficients, if the original MP3 Huffman coding is used. If stage 3 interleaving scheme is used, this would result in a loss of 16 separate QMDCT coefficients out of 1152 QMDCT coefficients in a MP3 frame.

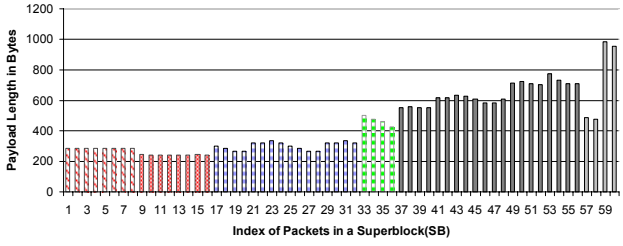

Figure 10. Average payload length in an SB for packet transmission

The price to pay for this improvement by including stage 3 interleaving is a slightly increased computational complexity and slightly decreased coding efficiency. The reason for the decreased coding efficiency is that the MP3 Huffman tables are optimized for QMDCT pairs, not for two separate QMDCT coefficients. Nevertheless, it is not a big hurdle to employ stage 3. According to our experiments, the bitrate increases less than 0.5% for most of our test signals when stage 3 is employed. Therefore, it is a good option in the R-D sense. Both BL2 and ELs are packetized similar to BL1. Each sub-layer consists of four packets and some packets in the higher ELs are combined to form larger packets. This is done to improve average throughput.

We choose a joint stereo test sample to illustrate our packetization scheme, due to the fact that a vast majority of MP3 files in our test database, which are samples from the Internet, are coded using joint stereo mode.

The number of bytes in a coded frame can be calculated as:

$$frame\_length = \frac{N \cdot b_r}{F_s \cdot 8} \qquad (1)$$

where N is the number of samples per frame, which is 1152 for MP3; b_r is the bitrate of the stream, which is 128kbps for our samples; F_s is the sampling frequency, which is 44.1 kHz. For this parameter set, the number of bytes per frame is 418.

Figure 10 shows the result of our packetization scheme using an MP3 file, which is coded with joint stereo mode. Our scheme generated 60 packets from 64 MP3 frames in one SB. The first eight packets are CRI packets, with the first four being MB1 and the second four MB2. The next eight packets are SCF packets, with the first four being MB1 and the second four MB2. The third set of eight packets is QMDCT coefficients of BL1 in channel 0. The fourth eight packets are QMDCT coefficients of BL2 in channel 0. This is followed by four packets which contain QMDCT coefficients of BL in channel 1. The above mentioned 36 packets out of the total 60 packets are sufficient for providing basic services. The next 20 long packets are generated from the ELs in channel 0 and the last four packets are generated from the ELs in channel 1. These 24 long packets are only necessary if we want full band audio. Therefore, we can drop packets from the least significant (e.g., ELs in channel 1 as shown in Figure 10) in the case of bandwidth constraints. On the other hand, we can employ techniques such as replication (adding redundancy), selective retransmission, and smart scheduling to protect the most important packets (e.g., the first eight packets in Figure 10).

3.1.3 Packetizers and schedulers

Packetizers and schedulers generate transportation packets from elementary packets and transport them according to various transport schemes and network status. These issues are out of the scope of this paper.

3.2 At the Receiver Side

The receiver performs a reversed processing of the sender as shown in Figure 5. According to the thin client principle, the error concealment operation in our scheme is extremely simple: 1) if a CRI packet is lost, we reconstruct the lost frames using their previous good frames; 2) if a SCF packet is lost, we copy the lost

SCFs from their previous good frames; 3) if a Huffman code packet is lost, we simply set the affected QMDCT coefficients to zero.

4. PERCEPTUAL EVALUATIONS

For our evaluation, we set the following conditions for both the baseline which packetizes one MP3 frame into one packet and our new scheme: 1) we generate the same amount of packets (64 packets in our current implementation) in an SB for both schemes; 2) we keep the bitrate roughly the same for both schemes; 3) we keep the error concealment simple so that the schemes are comparable in computational complexity. It should be noted that an important difference between the two is that packet size remains constant in the baseline scheme while packet size varies in our new scheme as shown in Figure 10. The impact of this difference is that a short packet has a lower loss rate than long packets in wireless networks. However, this impact can only be observed in a simulation or real network environment, but not in our current evaluation setup.

We perform two sets of subjective evaluations – one for evaluating error robustness and the other for evaluating bitrate scalability.

For the first evaluation, we form 64 packets in an SB in the manner shown in Figure 11. The numbers in the rectangles represent the number of packets in the section. We repeat the critical data twice and the scale factors and BL1 once, to reduce the probability of loss. In order to keep the bitrate roughly the same as the baseline, we drop the last 12 packets in the enhancement layer. We use ITU recommended packet loss patterns and specified error concealment methods to generate audio files for the evaluation. We use three systems for test: the original with no error (original), a baseline technique that merely repeats the previous good frame (baseline), and our MSI based method (new). Error concealment in our method includes the following three options: 1) In case a critical data packet is lost, we simply use the baseline method – muting. However, the probability of critical data loss is very low due to the increased redundancy. 2) In case a scale factor packet is lost, we just repeat the good SCF of the previous frame. 3) In case a QMDCT packet is lost, we simply mute the affected QMDCT coefficients.

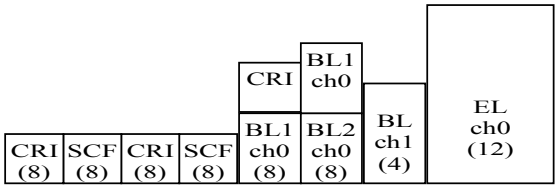

Figure 11. The 64 packets in an SB using LUSP for perceptual evaluation

We carry out our test on a group of 16 subjects (young male and female undergraduate and graduate students). One entire MP3 file of 3.5 minutes is used to generate the test signals. The bitrate of our MP3 file is 128 kbps coded with joint-stereo mode, and the sampling frequency is 44.1 kHz. All subjects are asked to evaluate the audio quality using the mean opinion score (MOS), which is a 5-point scale (5 – excellent, 4 – good, 3 – fair, 2 – poor, and 1 – bad). The obtained results are illustrated in Table 2 and Figure 12. Our evaluation method is similar to that in [15].

loss rate		0%	3%	5%	10%	20%
MOS (baseline)	mean	4.78	2.85	2.25	1.63	1.13
	stddev	0.27	0.73	0.51	0.56	0.29
MOS (new)	mean	3.88	3.79	3.36	3.10	2.48
	stddev	0.44	0.64	0.74	0.78	0.54

Table 2. Results of the first perceptual evaluation

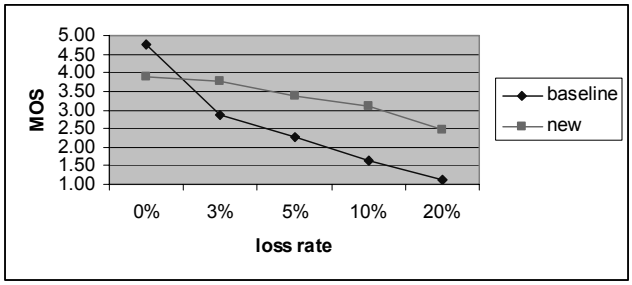

Figure 12. Performance comparison between the baseline and our new scheme

To summarize the results of our evaluation, our scheme can achieve good perceptual quality when the packet loss rate is relatively low (3-5%). When packet loss is heavy, our scheme achieves consistently better results than the baseline. If there is no packet loss at all, there is no need for our scheme.

For the second evaluation, we consider the case in which we know apriori that the receiver has a limited amount of bandwidth, and simply drop packets from the highest enhancement layer as shown in Figure 10. We start with 10% packet drop and end with 40% packet drop which corresponds to the dropping of all enhancement layer packets. The subjective results of audio with 10-40% packets dropped are shown in Table 3. This shows clearly that our scheme allows graceful quality degradation when packets are selectively dropped.

drop rate	0%	10%	20%	30%	40%
mean	4.78	4.68	3.88	3.49	3.05
stddev	0.27	0.42	0.44	0.44	0.55

Table 3. Results of the second perceptual evaluation

Our initial results encourage us to perform a full scale evaluation. An ideal method to evaluate the performance of the proposed scheme would be to implement it into a wireless packet network for actual live streaming. Therefore, we are currently building a WLAN-based test-bed for this purpose.

5. DISCUSSIONS

By taking users' need into our design loop, we believe that our solution provides a more attractive alternative in comparison with the existing solutions. Our framework is an application layer solution and is independent to any specific codec or operating system.

It is well-known that all perceptually coded media formats can be decomposed to elements which have different perceptual

significance. This is the foundation why our framework can be extended to other formats and even other media types such as video. We believe that leveraging a storage format to become a streaming format adds value to everyone: from content providers to device producers, and to end users. We have shown that it is possible to achieve bandwidth efficiency, error robustness and bitrate scalability using a single layer audio bitstream such as MP3, if delay constraints can be relaxed to the range of a few hundred milliseconds to a few seconds. A further advantage is that the computational load of our scheme is almost constant while other schemes suffer from computational load fluctuation.

The proposed framework enables various unequal error protection (UEP) schemes and application layer selective packet dropping with graceful quality degradation in the case of packet loss or bandwidth constraints. We can also employ scheduling for prioritized packet transmission – the critical packets are transmitted first to allow sufficient time for retransmission.

Due to the fact that current objective evaluation tools are only capable of predicting audio quality relatively accurately for audio samples with small impairment, they are not suitable for evaluating audio samples with large impairment such as audio samples with packet loss. This is the rationale for us to perform subjective listening tests.

Since our system is an initial prototype, there are many interesting problems open for future research, which include: 1) To investigate the feasibility to achieve Fine Granule Scalability (FGS) using the proposed scheme. 2) To investigate the feasibility of incorporating finer granularity in CRI and SCF (that is, applying three-stage interleaving not only on QMDCT coefficients, but also on CRI and SCF). 3) To study interleaving depth adaptation. 4) To test the scheme using other codecs such as AAC and OggVorbis. Our preliminary results show that it is fairly straightforward to implement the proposed scheme using the AAC bitstream. 5) To perform more rigorous evaluations with different loss models, NS-2 simulation environment and real wireless packet networks. 6) To investigate the applicability of the proposed scheme for error robust video streaming.

6. CONCLUSION

We have proposed a novel framework to convert perceptually coded audio formats, such as MP3, to an error robust and bitrate scalable streaming audio format. We hope to leverage the popularity of existing audio compression formats for many attractive streaming services, such as broadcasting, multicasting, and peer-to-peer streaming over the Internet and mobile networks. Our emphasis is not on the performance of individual aspects, such as coding efficiency, error robustness and bitrate scalability, but to strive for balanced system-level performance considering all major factors, including implementation simplicity, re-configurability and deployability. We have taken the needs of content providers and end users into our system design loop, which should increase its significance to practitioners. With minor reconfiguration, the proposed scheme can convert most existing and future audio format for error robust and bitrate scalable streaming services. Our scheme has been designed with due consideration to the new network standard – 802.11e which supports QoS – so that it can be deployed immediately after the new standard.

7. REFERENCES

[1] Brandenburg, K., "MP3 and AAC Explained," Audio Engineering Society (AES) 17th International Conference on High Quality Audio Coding, Florence, Italy, September 1999

[2] Grill, B., "A Bit Rate Scalable Perceptual Coder for MPEG-4 Audio," 103rd AES Convention, 1997, Preprint 4620

[3] Li, J., "Embedded Audio Coding (EAC) With Implicit Auditory Masking," ACM Multimedia 2002, Juan-Les-Pins, France

[4] Leslie, B., Sandler, M., "Packet Loss Resilient, Scalable Audio Compression and Streaming for IP Networks," 2nd International Conference on 3G Mobile Communication Technologies, pp 119-123, March 2001

[5] Wah, B.W., Su, X. and Lin, D., "A Survey of Error Concealment Schemes for Real-time Audio and Video Transmissions over the Internet," IEEE International Symposium on Multimedia Software Engineering, Taipei, Taiwan, pp.17-24, December 2000

[6] Perkins, C., Hodson, O., Hardman, V., "A Survey of Packet Loss Recovery Techniques for Streaming Audio," IEEE Network, pp.40-48, Sept/Oct, 1998

[7] Zhang, Q., Wang, G., Xiong, Z., Zhou, J., Zhu, W., "Error Robust Scalable Audio Streaming over Wireless IP Networks", to appear in IEEE Trans. On Multimedia, 2004.

[8] Herre, J., Eberlein, E., "Error Concealment in the spectral domain", 93rd AES Convention, Oct 1992, Preprint 3364

[9] Quackenbush, S., Driessen, P., "Error Mitigation in MPEG-4 Audio Packet Communication Systems," 115th AES Convention, Oct 2003, New York, USA, Preprint 5981

[10] Bucklein, R., "The Audibility of Frequency Response Irregularities," J. of AES, Vol.29, No.3, pp.126-131, 1981

[11] Moore, B.C.J, "Masking in the Human Auditory System," In Collected Papers on Digital Audio Bit-Rate Reduction (Gilchrist and Crewin, eds.), pp. 9-19, AES, 1996

[12] Brandenburg, K., Stoll, G., "ISO-MPEG-1 Audio: A Generic Standard for Coding of High Quality Digital Audio," In Collected Papers on Digital Audio Bit-Rate Reduction (Gilchrist and Crewin, eds.), pp. 31-42, AES, 1996

[13] http://www.3gpp.org/

[14] Wenger, S., "Common Conditions for Wire-line, Low Delay IP/UDP/RTP Packet Loss Resilient Testing," ITU VCEG 14th Meeting, Santa Barbara, CA, USA, September 2001

[15] Niedzwiecki, M., Cisowski, K., "Smart Copying – A New Approach to Reconstruction of Audio Signals," IEEE Trans. Signal Processing, pp.58-63, Vol. 49, No. 10, October 2001

[16] Korhonen, J., Wang, Y., "Schemes for Error Resilient Streaming of Perceptually Coded Audio," IEEE ICASSP2003, Hong Kong, pp. 740-743, April 2003