2008

Performance and Complexity Co-Evaluations of MPEG4-ALS Compression Standard for Low-Latency Music Compression

Isaac Kevin Matthew
Wright State University

PERFORMANCE AND COMPLEXITY CO-EVALUATIONS OF MPEG4-ALS COMPRESSION STANDARD FOR LOW-LATENCY MUSIC COMPRESSION

A thesis submitted in partial fulfillment of
the requirements for the degree of
Master of Science(Computer Science)

By

ISAAC KEVIN MATTHEW
M. S. Physics

2008
Wright State University

WRIGHT STATE UNIVERSITY

SCHOOL OF GRADUATE STUDIES

21 August, 2008

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Isaac Kevin Matthew ENTITLED Performance and Complexity Co-evaluation of MPEG4-ALS Compression Standard for Low-latency Music Compression BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science, Computer Science.

Dr. Yong Pei (Advisor)
Assistant Professor, CS&E

Thomas Sudkamp
Interim Chair, CS&E

Committee on
Final Examination

Dr. Yong Pei
Assistant Professor, CS&E

Dr. Bin Wang
Associate Professor, CS&E

Dr. Thomas Hartrum
Assistant Research Professor, CS&E

Joseph F. Thomas, Jr., Ph.D.
Dean, School of Graduate Studies

ABSTRACT

Matthew, Isaac Kevin. M.S., Department of Computer Sciences & Engineering, Wright State University, 2008. Performance and Complexity Co-Evaluations of the MPEG4 ALS Compression Standard for Low-Latency Music Compression.

In this thesis compression ratio and latency of different classical audio music tracks are analyzed with various encoder options of MPEG4–ALS. Different tracks of audio music tracks are tested with MPEG4-ALS coder with different options to find the optimum values for various parameters to obtain maximum compression ratio with minimum CPU time (encoder and decoder time). Optimum frame length for which the compression ratio saturates for music audio is found out by analyzing the results when different classical music tracks are experimented with various frame lengths. Also music tracks with varying sampling rate are tested and the compression ratio and latency relationship with sampling rate are analyzed and plotted. It is found that the compression gain rate was higher when the codec complexity is less, and joint channel correlation and long term correlations are not significant and latency trade off make the more complex codec options unsuitable for applications where latency is critical. When the two entropy coding options, Rice code and BGMC (Block Gilbert-Moore Codes) are applied on various classical music tracks, it was obvious that the Rice code is more suitable for low-latency applications compared to the more complex BGMC coding, as BGMC improved compression performance with the expense of latency, making it unsuitable in real-time applications.

TABLE OF CONTENTS

LIST OF FIGURES

PAGE

PAGE

LIST OF TABLES

PAGE

ACKNOWLEDGEMENTS

I am forever obliged to my Lord and Savior Jesus Christ for coming into my life and being with me all the time, filling me with hope, purpose and peace. Without His blessings, I never would have done this thesis.

I would like to record my gratitude to Dr. Yong Pei, for his supervision, advice and guidance. He has assisted me in numerous ways, including, editing the writings, summarizing the results and in particular giving insightful ideas. My sincere thanks also go to Dr. Bin Wang and Dr. Thomas Hartrum for being members of my thesis committee. I am thankful that in the midst of their busy schedule, they accepted to be members of my thesis committee.

I am grateful to all the staff and faculty of the Dept. of Computer Science and Engineering at Wright State University for giving me the opportunity and assistance to conduct research and study. I am thankful for DAGSI(Dayton Area Graduate Studies Institute) for providing me with full tuition assistance throughout my graduate program.

I am indebted to my parents for their unfailing love and support throughout my life. No words can express how grateful I am for the love and encouragement of my wife Leeba. I would also like to thank my brother, Andrew for his unflinching assistance throughout my MS program.

I am thankful for all the members of Dayton Bible Chapel and LexisNexis Bible Study Group for their persistent prayers and thoughtfulness.

I am thankful for my manager Mr. Andrew Lloyd and supervisor Mr. Paul Gossard for giving me the opportunity to work in LexisNexis with a flexible schedule.

Last, but not the least, I would like to thank all my friends, co-workers, relatives, and well wishers for their support and inspiration.

To my wife, Leeba

Chapter 1
Introduction

Internet is now playing a very significant role in our daily life. Effective streaming of different types of media like speech, audio, video, text and images are critical for interactive applications through the Internet. Due to the limitation of bandwidth, at any given time, the ability of the Internet to transfer data is fixed. For multimedia applications involving high data transfer, one should consider compressing the data before streaming. By effectively compressing the data significant improvements of data throughput can be achieved.

1.1 Objective

Telepresence is the most effective communication tool for remote collaborations. Telepresence is a very time-sensitive application in which the transmission must operate in real time. In order to create the perception of real-time communication between end users, the network delay should be very small. For telepresence in general and music telepresence in particular, the delay should be less than 100 ms for acceptable performance. For good performance the delay should be less than 50 ms. In this thesis a careful study of the effects of applying data compression techniques in minimizing overall delay of music telepresence is analyzed.

There is obviously a trade of between compression delay (coding delay) and network transmission delay. Compression can save bandwidth and reduce the transmission delay but with the expense of encoding and decoding time.

Here we have used classical music tracks to co-evaluate the performance and complexity of MPEG 4-ALS codec by applying various encoding options. In short, the objective of this study is to find the possibility of using data compression techniques to advance the state of network-based telepresence by minimizing the overall delay within reasonable limits.

1.2 Data Compression

Data compression seeks to reduce the number of bits used to store or transmit information by the identification and extraction of source redundancy, which is connected with statistical inference. Compression helps reduce the consumption of expensive resources, such as disk space or transmission bandwidth but at the expense of extra processing that may be detrimental to some applications. The task of compression consists of two components, an encoding algorithm that takes a message and generates a "compressed" representation and a decoding algorithm that reconstructs the original message or some approximation of it from the compressed representation.

These two components are typically intricately tied together since they both have to understand the shared compressed representation.

As is the case with any form of communication, compressed data communication only works when both the sender and receiver of the information understand the encoding scheme. The theoretical background of compression is provided by information theory and rate-distortion theory.

The two basic terms referred in interactive data compression are compression ratio and latency, which are calculated as follows;

Compression ratio = original size / compressed size

Latency = encoding time + decoding time

The amount of compression that can be achieved depends mainly on the efficiency of algorithm and the amount of redundancy in the source whereas the latency depends on the efficiency of algorithm and hardware efficiency (such as CPU speed). Data compression can be divided into two main types, the lossless and lossy compression.

Lossless compression can be used when exact reconstruct of the original is essential. Lossless compression schemes are reversible, i.e., it can recover the exact original data after compression, but it may fail to compress data containing no discernible patterns. The lossless data compression methods typically also offer a tradeoff between latency and compression ratio.

Lossy compression will result in a certain loss of accuracy in exchange for a substantial increase in compression. Lossy compression is more effective when used to compress

graphic images and digitized voice where losses outside visual or aural perception can be tolerated.

Most lossy compression techniques can be adjusted to different quality levels, gaining higher accuracy in exchange for less effective compression. The lossy data compression methods typically offer a three-way tradeoff between latency, compression ratio and quality loss.

1.3 Speech coding/Lossy Audio coding

In speech and lossy audio coding the quality is based on the properties of human auditory perception. Speech compression uses a model of the human vocal tract to express particular signals in a compressed format. As speech production model is available, speech can be coded very efficiently. But due to the complexity of audio signals such as music, such a model would be too complex to implement and hence the lossy encoding of music is usually not as efficient as speech coding.

The lossy audio compression will try to eliminate information that is inaudible to the ear. The audio compression algorithms rely on the field of psychoacoustics (the study of human sound perception).

The signals become inaudible to ear when they obscure or mask each other. These occur under three conditions namely, threshold cut-off, frequency masking and temporal masking.

Threshold cut-off: For humans, hearing is limited to frequencies between about 20 Hz and 20,000 Hz (20 kHz). Human ear detects sounds as air pressure variations measured as

Sound Pressure Level (SPL). Therefore, human ear cannot detect sound if the variations in the SPL are below a certain threshold in amplitude.

Frequency Masking: Some of the signal components that exceed the hearing threshold may be masked by louder components that are near it in frequency. These shadowed or masked components will not be heard.

Temporal Masking: A sudden increase in sound can temporarily mask neighboring signals. Sounds that occur before and after the volume increase can be masked.

Lossy coding can exploit these phenomenons to eliminate those signals and can achieve significant compression performance.

1.4 Lossless Audio coding

Lossless compression compresses a signal without loss of information. After decoding, the resulted signal is identical to the original signal. Compared to lossy compression, lossless compression achieves a very limited compression ratio.

It is difficult to maintain all the data in an audio stream and achieve substantial compression expecially when the audio is music due to its high complexity.

As one of the key methods of compression is to find patterns and repetition, more chaotic data such as audio doesn't compress well.

In most cases, the values of audio samples change very quickly, generic data compression algorithms don't work well for audio, and strings of consecutive bytes don't generally appear very often.

Since lossless audio codecs have no quality issues, the efficiency can be estimated by

- Speed of compression and decompression (latency)
- Compression ratio
- Software and hardware support
- Robustness and error correction

1.4.1 The Basic Principle

Lossless audio compression is split into two main parts - filtering and entropy coding as shown in Fig. 1.1 and Fig. 1.2.

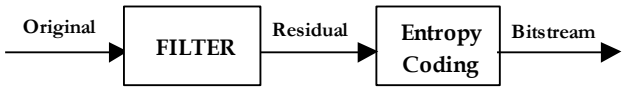

Fig.1.1 Principle of Lossless Encoding

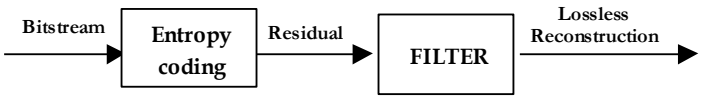

Fig.1.2 Principle of Lossless decoding

1.4.2 Filter

A filter essentially takes a set of numbers and returns a new set. For the purposes of lossless audio compression, the transformation must be done in such a way that it is

reversible and hopefully the transformation will reduce the range of numbers so that they compress better.

Filtering or transforming signals (e.g. Fast Fourier Transform (FFT)) slightly decorrelate (make flat) the spectrum, thereby allowing traditional lossless compression at the encoder to do its job; integration at the decoder restores the original signal. Many lossless codecs (e.g. FLAC, Shorten, TTA etc.) use linear prediction to estimate the spectrum of the signal.

At the encoder, the estimator's inverse is used to whiten the signal by removing spectral peaks while the estimator is used to reconstruct the original signal at the decoder.

1.4.2.1 Prediction

In lossless codec, most of the filters used are constructed out of predictors. A predictor here is a function, which is passed the previous sample and returns a prediction of the next. The predictor may of course internally store some state or history.

A filter can thus be created out of any predictor, such that the output value (residual) is the difference between the actual sample and the prediction, i.e.

Residual = *Sample* – *Prediction*

and then to recover the original sample when decoding, uses:

Sample = *Residual* + *Prediction*

A number of different predictors are used in lossless codecs. Delta filter is a simple filter, which uses *last-sample* as *prediction*. In more complicated filters, they adjust the weight between the last-sample and the preceding prediction.

Prediction = *last-sample* * *weight*, and we adaptively adjust the value *weight*.

A simple method to adapt *weight* would be to increase it when the last prediction was too low, and decrease it when the last prediction was too high.

Compression performance can further be improved by successively applying multiple filters to the data. (i.e., the second filter takes the output of the first filter as input). Also predictors, which use 'n' preceding samples in prediction, can increase performance.

Another approach to improving upon this predictor is to create a single predictor, which takes into account the past *n* samples. It then needs to store a corresponding array of *n* weights, and will require loops to adapt the weights and to calculate the prediction. Most filters are based on these ideas. Filter selection depends on the performance as well as the encoding and decoding speed.

1.4.2.2 Stereo Decorrelation

Most lossless audio compressors, try to take into account the similarity between channels in stereo audio to improve compression performance. The standard way to do this is to convert the left channel (L) + right channel (R) signals to X+Y, where $X = L - R$ and $Y = R + (X / 2)$.

However, audio signals with low correlation between channels may decrease performance. A simple example is a file where one channel is silent - after the X+Y transformation both channels would contain the signal, thus potentially doubling the resultant file size.

In efficient lossless audio codecs, the predictors take into account samples from both channels. Thus, more complex correlation between the channels is better taken into

account of, and it adapts better to the actual level of correlation existing in the signal rather than simply assuming that the channels are correlated as with the X+Y transformation.

1.4.3 Entropy coding

The term entropy denotes amount of information in a signal. When entropy is lower more predictable is the signal. From a compression perspective, lower the entropy, greater will be the compression ratio. Claude Shannon formulated the theory of entropy of a system encoded into binary format, using bits/samples as a measurement method.

$$H = -\sum_{x} P_x \log_2 P_x \qquad (2.1)$$

In (2.1) H is the entropy of the signal and P the probability of a symbol occurring in a signal. H is the theoretical minimum code required to code the given data stream to binary.

The whole purpose behind the filters reducing the range of the samples is the assumption that smaller numbers can be stored more efficiently. Shannon's entropy measures the information contained in a message as opposed to the portion of the message that is determined (or predictable). After the data has been quantized into a finite set of values, it can be encoded using an entropy coder to give additional compression.

By entropy, we mean the amount of information present in the data, and an entropy coder encodes the given set of symbols with the minimum number of bits required to represent them. Two popular entropy-coding schemes are Huffman coding and Arithmetic coding.

These coding methods require prior knowledge of the signal statistics to decode the signals efficiently. Rice coding and cascade coding are used for signals with Laplacian distribution and stepwise distribution respectively.

1.5 Comparison of Lossless codecs

Features	FLAC	WavPack	Monkey's	TTA	LPAC	MPEG-4 ALS	Shorten	Real Lossless
Streaming	Yes	Yes	No	No	No	Yes	No	Yes
Open source	Yes	Yes	Yes	Yes	No	Yes	Yes	No
Multi-channel	Yes	Yes	No	Yes	No	Yes	No	No
OS support	All	All	All	All	Win/ Linux /Sol	All	All	Win/ Mac/ Linux

Table 1.3 Comparison of Lossless codecs

1.6 Summary

The current chapter describes the role of data compression in audio communication and an overview of data compression techniques. Basic terminology and theory behind data compression are discussed. Detailed description of various techniques used in lossy/speech coding and the theoretical difference between lossy and lossless coding are given. The basic constituent of audio lossless compression such as filters and entropy coding techniques are discussed in details. Detailed explanations of how the predictors efficiently exploit the correlation between adjacent samples are also given in this chapter.

Stereo decorrelation techniques used to exploit correlation between adjacent channels are discussed. Finally a general comparison of different lossless codecs is given in tabular form.

1.7 Organization of Thesis

The rest of the thesis is organized as follows: In Chapter 2, the motivation, scope, challenges and progress of network-enabled remote telepresence project (music-telepresence) is reviewed. Later on in this chapter, compression and latency Challenges in networking applications are discussed. In Chapter 3, an overview of MPEG-4 audio lossless coding standard (ALS) is given along with a detailed description of different techniques used to optimize the MPEG-4ALS codec for Internet related musical applications. In Chapter 4, results, performance evaluations and analysis of all proposed techniques to optimize MPEG4-ALS codec are given. Finally, we provide our conclusions and future research directions in Chapter 5.

Chapter 2
Interactive Multimedia Network Applications

Interactive multimedia networking is used in almost all area of human life including medical, corporate and entertainment fields. Interactive multimedia is widely used in advertising, information system, online multimedia training, patient monitoring networks and multimedia conferencing etc. In order to understand the challenges associated with a particular interactive multimedia network application, it is necessary to understand the level of interactivity and multimedia data transfer associated with the application. Network performance requirements depend on nature of these applications and the level of interactivity involved in these applications. For interactive applications involving continuous bi-directional multimedia data transfer, it requires network with sufficient bandwidth to satisfy the specific need.

2.1. Telepresence

Multimedia conferencing is the most effective modern tool of communication. Advanced multimedia conferencing makes it possible to allow persons to feel as if they were present at a common location other than their true locations.

A set of technologies that allow combining human factors of communication with the latest videoconferencing technologies is referred as telepresence. Telepresence make it possible to interact each other effectively by talking, hearing, seeing and communicating by other means. Telepresence not only provides effective virtual business meeting opportunities, but also provide support to other areas like the emergency and security services, entertainment and education industries.

Development of a network-enabled remote telepresence platform has the potential to broadly impact society through the benefits of improved interpersonal interactions, the economic advantages afforded by the elimination of physical barriers to collaboration and the need to travel, and the ability to provide new and improved services to economically and culturally deprived, geographically remote, or physically handicapped populations is the motivation behind the music telepresence project. Its application can vary from interactive performances and collaborations by performing artists to remote medical diagnosis, collaboration and treatment. However, the existing efforts to achieve network-based telepresence have yet to reach the ideal level of providing a widely accessible, medium-transparent, acceptably immersive interactive audio/video environment between remote locations while requiring only commodity network services and terminal platforms.

Here we emphasis on the challenges in telepresence applications involving transmission of audio (music) channels. In order to understand the challenges in telepresence and to measure various aspects of network and protocol performance, it is essential to know the basic network terminology.

2.2. Network Terminology

The values for the following metrics determine the performance of network applications.

Network latency refers to any of several kinds of delays typically incurred in processing of network data. A so-called low latency network connection is one that generally experiences small delay times, while a high latency connection generally suffers from long delays. Bandwidth can vary over time and is coupled with high latencies. The continuous flow of multimedia data across the nodes is affected by excessive network latency. The impact of latency on network bandwidth can be temporary or persistent depending on the cause of the delays.

Round Trip Time (RTT) expressed in milliseconds, is the elapsed time for a request to go from node 'A' to node 'B,' and for the reply from 'B' to return to 'A.' RTT is the total time for the trip. The forward and reverse path times can vary depending up on the network conditions. RTT depends on the distance between nodes, network conditions, and packet size. Packet size, congestion, compressibility and data compression have a significant impact on RTT.

Bandwidth in computer networking refers to the data rate supported by a network connection or interface. Network bandwidth is one of the major factors that affect the latency (network delay), which is one of the key elements of network performance. Essentially, bandwidth represents the capacity of the connection, and it is obvious that the greater the capacity, the more likely that greater performance will follow. Bandwidth rating of the modem or the Internet service is given in Mbps or Kbps.

T**hroughput** in communication networks is the average rate of successful message delivery over a communication channel. This data may be delivered over a physical or logical link, over a wireless channel, or between two specific computers. The throughput is usually measured in bits per second (bit/s or bps). The system throughput or aggregate throughput is the sum of the data rates that are delivered to all terminals in a network.

Jitter is an unwanted variation of one or more characteristics of a periodic signal in electronics and telecommunications. Jitter may be seen in characteristics such as the interval between successive pulses or cycles. Jitter is a significant factor in the design of almost all communications links. Jitter period is the interval between two times of maximum effect (or between two times of minimum effect) of a jitter characteristic, for a jitter that varies regularly with time. Inverse of jitter is refered as *jitter frequency*.

2.3. Network delay (Latency) Factors

Network delay in an IP network is the one-way delay for an IP packet within an IP network. In addition to delay in transmitting the packet serially through a link (*propagation delay*), IP network delay comprises of the sum of the *transmisson delays* and *queuing delays* experianced by the packet travelling through the collection of routers, switches and other hardware that comprise the network.

2.3.1. Propagation Delay

The time required to propagate from the beginning of the link to router is the propagation delay. The bit propagates at the propagation speed of the link, which depends on the

physical medium of the link. Propagation speed ranges from $2*10^8$ meters/sec to $3*10^8$ meters/sec (the speed of light). The propagation delay is the distance between two routers divided by the propagation speed.

In wide-area networks (WAN), propagation delays are on the order of milliseconds. In calculating distances, we should consider the fact that the actual distance between the places is not always the same as the network path distance. To minimize propagation delay we can use efficient network topology to link sites using the shortest, most direct route.

Music Telepresence duet music session test results indicated a network propagation delay of 15ms between WSU (Wright State University) and UR (University of Rochester). Propagation delay is less than 1 millisecond per 100 miles even if the speed of propagation is around 60% of speed of light.

2.3.2. Packetization Delay

Packetization delay occurs when data is being broken down into packets to be transmitted. This delay exists at the origin from where transmission started. Larger the packet the greater the packetization delay will be. Packetization delay can also be called Accumulation delay, as the data accumulate in a buffer before they are released. As a general rule packetization delay of no more than 30 ms is considered acceptable, but it varies from application to application.

2.3.3. Processing Delay

The time required in examining the packet's header and determining where to direct the packet is part of the processing delay. The processing delay can also include other factors,

such as the time needed to check for bit-level errors in the packet. Processing delays in high-speed routers are typically on the order of microseconds or less.

2.3.4. Queuing Delay

The packet experiences a queuing delay as it waits to be transmitted onto the link. The queuing delay of a specific packet will depend on the number of other, earlier-arriving packets that are queued and waiting for transmission across the link. The delay of a given packet can vary significantly from packet to packet. If the queue is empty and no other packet is currently being transmitted, then our packet's queuing delay is zero. On the other hand, if the traffic is heavy and many other packets are also waiting to be transmitted, the queuing delay will be long. Queuing delays can be on the order of milliseconds to microseconds. A fast packaging machine would pump more packets on the wire and thus can quickly put all the material into packets.

2.3.5. Transmission Delay

Assuming that packets are transmitted in first-come-first-serve manner, as is common in the Internet, the packet can be transmitted once all the packets that have arrived before it have been transmitted.

Transmission delay = L/R where,

L = Length of the packet in bits

R = Transmission rate of the link between routers

This is the amount of time required to transmit all of the packet's bits into the link. Transmission delays are typically on the order of microseconds or less in practice.

2.3.6. Coder Delay

Coder delay is the time taken to compress the raw data. This is also called encoder delay. This delay basically varies with complexity of the codec algorithm.

In most cases, decompression time is very small compared to compression time. Detailed discussion of the trade offs between complexity and latency of audio compression are discussed in the following chapters.

2.3.7. De-Jitter Delay

The de-jitter buffer transforms the variable delay into a fixed delay. It holds the first sample received for a period of time before it plays it out. This holding period is known as the initial play out delay. Constant bit-rate service like audio, video streaming, the jitter from all the variable delays must be removed before the signal leaves the network.

It is essential to handle properly the de-jitter buffer. If samples are held for too short a time, variations in delay can potentially cause the buffer to under-run and cause gaps in the audio. If the sample is held for too long a time, the buffer can overrun, and the dropped packets again cause gaps in the audio. Delay should be adjusted such that overall delay on the connection is within the acceptable limits.

The optimum initial play out delay for the de-jitter buffer is equal to the total variable delay along the connection. The de-jitter buffers can be adaptive, but the maximum delay is fixed. When adaptive buffers are configured, the delay becomes a variable figure. However, the maximum delay can be used as a worst case for design purposes.

2.4. Latency Requirement for Real-time Networking

Requirement for interactive multimedia network will vary widely from application to application. Here are some of the critical requirements for Telepresence and their recommended solutions.

For multi-user multimedia communication network linking several participants there is requirement for several simultaneous transmission channels. Each node should be capable of outputting audio and video channel and receiving and decoding several other audio video channels. Therefore intermixing of various channels and decoding them are critical requirements for interactive applications like telepresence. As the channels increases, the amounts of data transfer also increases. Bandwidth requirement is critical when large amount of multimedia data transfer is required. Since the bandwidth is fixed, reducing network delay is the major challenge in those applications involving large quantity of multimedia data transfer.

All interactive multi-user multimedia conferencing systems including telepresence are very time-sensitive applications. Transmissions must operate in real time and therefore network delay should be small in order to create the perception of real-time communication between end users. In order to meet the demands substantial data throughput of network is required. Faster processors as well as data compression are vital to faster multimedia data transfer.

The amount of data that a transmission link is able to transmit per second depends on the bandwidth of a transmission link and can be measured by different ways. There is a trade

off between quality and bandwidth when the bandwidth requirement is not met. Latency and jitter are the most significant issues in interactive multimedia transmission and the combined latency should not exceed the tolerable limit so that the continuity and quality of multimedia transmission is achieved.

One of the major challenges in interactive multimedia network applications involves the reduction of network delay to accepted limit considering the bandwidth and data quantity. As the bandwidth is limited, data quantity plays an important role in the network performance in interactive multimedia applications. Data Compression can reduce data thereby improving data transfer by reducing network delay by sending less data.

All the modern modems have compression algorithms built-in. Modem has slow processor. Compared with computers modems do compression very less efficiently due to its limitations. Moreover modems cannot take the advantage of data specifics, as it doesn't know the kind of data it receives. Computer can use data specific compression algorithms to improve the compression. Compression can trade off use of CPU power in exchange of lower bandwidth requirements and thus can make up the poor latency.

There is a trade off between compression performance and latency as compression itself takes encoding and decoding time (codec latency). So by optimizing the codec complexity, reasonable compression can be achieved that will reduce the effective combined latency (codec latency + network latency). For interactive applications involving audio signals, considerable reduction in data quantity can be achieved without compromising the quality by applying lossless compression techniques on audio signals exploiting the correlation between adjacent samples.

Efficient compression algorithm can use less time and CPU power to exploit correlation between audio samples and compressed data before sending through the networks thereby saving bandwidth.

2.5. Musical Telepresence

The objective of this project is to advance the state of network-based telepresence by focusing on a number of successively demanding applications to be delivered over Internet2 and based on established multimedia protocol architectures and near-commodity terminal platforms. The particular musical applications to be addressed in this research range from high-quality multicast streaming of musical performances to robust, low-latency, interactive full-duplex unicast transport to enable real-time distributed interactive performances, collaboration and rehearsal, musical training and education.

Musical performance and collaboration is highly demanding of audio and video quality, with latency a critical issue in such highly interactive situations. Thus, the proposed musical scenarios provide an appropriately stressing application for pushing the envelope of the ability of Internet2 in providing interactive, immersive multimedia environments.

The technical challenges of this project encompass issues ranging from the development of efficient and reliable low-latency audio and video compression and transport protocols to acoustical and visual perceptual studies, and will include exploration of novel musical experiences enabled by the technology developed. The significant demanding musical telepresence applications include interactive and distributed performance.

2.5.1. Project Description

We have developed a music telepresence software platform running on PC's, which is capable of supporting multiple musicians participating in a music session remotely. The platform effectively supports the low-latency, high-quality audio/video needs demanded in musical applications and provides tolerance to music/video packet loss and late arrivals and associated delay jitter over Internet. Software is based on the existing Open H.323 software, but with major modifications in order to support the low-latency, high quality needs from demanding music applications.

We have carried out a series of cross-campus tests among the three sites at UR (University of Rochester), UM (University of Miami) and WSU(Wright State University) e.g., our latest session brought together 4 musicians, one guitar at UR, one piano at UM and 2 guitars at WSU, to successfully rehearse together the music - "Passion" by Olga Harris. These tests show promising music/video quality, low latency and stable operation that demonstrate the feasibility of a relatively simple PC-based music telepresence system over Internet 2 to support distributed musical collaboration.

Specifically, the system achieves very low end-to-end latency, e.g., an average end-to-end latency of about 35 ms for a WSU-UR duet music session, including a network propagation delay of 15 ms. For comparison, for the same connection between WSU and UR, when using such online collaboration tools as MSN or Yahoo! Messenger, the experienced end-to-end delay is over 250 ms. Keep in mind, for musicians to be able to play together remotely, the latency should be less than 100 ms so the current system is well within useable bounds.

The developed music telepresence system supports high sampling rate (44,100 Hz) stereo audio, (standard CD quality stereo music), which is not supported in most existing VoIP, Video Conference and online collaboration tools. A better than SDTV quality video is supported over the Internet2 sessions, while a wide range of video quality is available for use in bandwidth-constraint connections.

Tests were also carried out beyond the Internet2 links by linking homes connected through AT&T Yahoo! DSL and/or Time Warner Cable Modems which provides an asymmetric connection (downlink speed of up to 1.5 Mbps, and uplink speed of up to 500Kbps). Our system survives the very large delay jitter (sometime larger than 40 ms), and still performs robustly and maintains good quality. However, the network capacity limits the use of higher audio sampling rate, especially over the up-link; and the need for longer de-jitter buffering resulted in longer delay; we have an ongoing effort to improve the QoS over such home links through lossless or near-lossless music compression and adaptive protection against packet losses.

2.5.2. Features Supported by the Project

Duet through point-to-point link directly using client software and Trio, master class, and multiple-musician distributed rehearsal through connecting to Music Telepresence Server are supported. A wide range of audio/music sampling rates, e.g., 44K, 22K, 11K, 8K and very high music quality are supported. Better than SDTV quality video is supported over the Internet2 sessions, while a wide range of video quality is available for use in bandwidth-constraint connections: QCIF(176x144), CIF (352x288) and 4CIF (704x576). Test messaging is also supported by the project.

2.5.3. Performance

Very low end-to-end latency, e.g., an average end-to-end latency of about 35 ms for a WSU-UR duet music session, including a network propagation delay of 15 ms. For comparison, for the same connection between WSU and UR, when using such online collaboration tools as MSN or Yahoo! Messenger, the experienced end-to-end delay is over 250 ms. Keep in mind, for musicians to be able to play together remotely, the latency should be less than 100 ms so the current system is well within useable bounds.

It support high sampling rate, high quality stereo audio, e.g., CD quality stereo music, which is not supported in such tools as Yahoo! Messenger. Interference between real-time music application and other applications running on a same PC are successfully handled. Prioritized real-time thread scheduling in Linux is provided to synchronize and reduce the interference between the audio and video threads belong to the music session. It becomes clear that audio quality improves significantly when there is no interference. Moreover, the minimum packet size can go down to 2 ms per packet, which is critical to reduce the audio data holding delay.

Music tests beyond Internet2 were also was conducted by connecting between campus and home connected through SBC Yahoo! DSL which provides an asymmetric connection (downlink of upto 1.5 mbps, and uplink of upto 500Kbps), and the System is able to survive the very large delay jitter (sometime larger than 40 ms), and still performs robustly and maintains good quality, although the network capacity limits the use of higher sampling rate, especially over the up-link, and de-jitter buffering resulted in bigger delay.

A new music telepresence server was designed and implemented. The original MCU in OpenH323 designed for video conferencing can't meet our needs in many aspects. In particular, it is not able to support the small audio package used in our project in order to reduce the overall end-to-end delay. For example, the VoIP and Video Conference type of application normally packs 20-50 ms of audio into a package, while in our project we use packets as small as 2-10 ms of audio per package. The reduction in package size puts extreme challenges on the real-time demand in the software design. Each process cycle has to be synchronized to much higher accuracy in order to avoid software-caused artificial jitter which leads to unnecessary packet drop and degrades the audio quality significantly.

We have observed severe problems when we tested using the original H.323 software. The difficulty in high performance real-time design is due to the lack of hard real-time support in most general OS's used in PCs. So we came up with a solution that uses the embedded sound card to serve as a pacer that controls each process cycle. The result turns out as expected since this pacer works separately from the main CPU and provides constant hard real-time guarantee. Our cross-campus test show outstanding performance and it overcomes previous problems.

Another major challenge is to support the audio/video processing for multiple musicians under a tight time budget. To gain a quick insight, let's assume the audio packet carries 5 ms data. This means audio/video from all attending musicians must be decoded, mixed, re-encoded and resent back to each musician within this 5 ms; otherwise, it can't keep pace. Thus we are investigating the use of a multiple processor high-end PC to serve as a MCU. However, it will not be good enough without a new design of the MCU to eliminate many of the unfit implementation problems. Among them, firstly we replace the

original random CPU access scheme used between threads for each client connections by a new semaphore-enabled coordinated CPU access scheme, which orders all the process in a sequence. Benefits of the new approach include elimination of the potential jitters caused by random access and also this may help the future echo canceling. It also helps meet the hard real-time demand. Secondly, we change the data structure and mixing procedure to reduce the computational complexity from O(N^2) to O(N) (N is the number of musicians) in order to reduce the individual connection overhead and make the system more scalable.

2.6. Summary

In this chapter, we have reviewed the motivation, scope, challenges and progress of Network-enabled remote telepresence project (Music-telepresence). Audio and video compressions are so significant for better performance, as latency and quality are the greatest challenge. Due to the network contention and routing delays, to achieve the latency tolerance limit, the need of an efficient low latency lossless or near lossless compression codec for audio and video signals is essential. In summary, the need of compression is reiterated for the betterment of the network performance of music Telepresence. In the chapters ahead a detailed discussion of the need, selection and fine-tuning of audio compression codec to exploit the correlation of music signals is done.

Chapter 3
MPEG-4-ALS

The Moving Picture Experts Group (MPEG) is a working group of ISO/IEC in charge of the development of international standards for compression, decompression, processing, and coded representation of moving pictures, audio and their combination. MPEG-4 is an ISO/IEC standard developed by MPEG as a result of international effort involving various researchers and engineers from all over the world. MPEG-4, with formal as its ISO/IEC designation 'ISO/IEC 14496', was finalized in October 1998 and became an international standard in the first months of 1999.

3.1 MPEG4-ALS - Overview

MPEG-4 provides the standardized technological elements enabling the integration of the production and distribution of:

- Digital television
- Interactive graphics applications (*synthetic content*) and
- Interactive multimedia (*World Wide Web, distribution of and access to content*).

The interactive multimedia application includes music network (music telepresence), which is covered under background (chapter 2).

MPEG-4 potentially covers all digital sound and television applications by providing technology to represent stereo and multichannel sound with transparency achieved at 128 kbit/s (stereo) and 320 kbit/s (5.1 multichannel).

MPEG-4 Audio facilitates a wide variety of applications which could range from intelligible speech to high quality multichannel audio, and from natural sounds to synthesized sounds. It support for coding general audio ranging from very low bitrates up to high quality provided by transform coding techniques. With this functionality, a wide range of bitrates and bandwidths is covered. It starts at a bitrate of 6 kbit/s and a bandwidth below 4 kHz and extends to broadcast quality audio from mono up to multichannel. High quality can be achieved with low delays making it possible to be used in communications applications.

3.1.1 General Features

MPEG-4 ALS defines efficient and fast lossless audio compression techniques for both professional and consumer applications. It offers many features not included in other lossless compression schemes.

- General support for virtually any uncompressed digital audio format (*including wav, aiff, au, bwf, raw*).
- Support for PCM resolutions of up to 32-bit at arbitrary sampling rate (*including, e.g., 16/44.1, 16/48, 24/48, 24/96, 24/192*).

- Multi-channel/multi-track support for up to 65536 channels (*including 5.1 surround*).
- Support for 32-bit IEEE floating point audio data.
- Fast random access to any part of the encoded data.
- Optional storage in MP4 file format (*allows multiplex with video*).
- High flexibility of codec parameters for various applications.
- Global MPEG standard for lossless audio coding will facilitate interoperability between different hardware and software platforms, promoting long-lasting multivendor support.

3.1.2 Codec Structure

MPEG-4 Audio Lossless Coding (ALS) enables the compression of digital audio data without any loss in quality due to a perfect reconstruction of the original signal. Its encoder is based on linear prediction, which enables high compression even with moderate complexity, while the corresponding decoder is straightforward.

Since the encoding process has to be perfectly reversible without loss of information, several parts of both encoder and decoder have to be implemented in a deterministic way.

The input audio data is partitioned into blocks. For each block, a prediction residual is calculated using *short-term prediction* and then *long-term prediction*. After that, the prediction residual is entropy-coded.

3.1.2.1 Encoder Structure

MPEG4 Audio Lossless Coding encoder (Fig. 3.1) typically consists of these main building blocks, which describe the basics of MPEG4-ALS:

Buffer: Stores one audio frame. A frame is divided into blocks of samples, typically one for each channel.

Coefficients Estimation and Quantization: Estimates (and quantizes) the optimum predictor coefficients for each block.

Predictor: Calculates the prediction residual using the quantized predictor coefficients.

Entropy Coding: Encodes the residual using different entropy codes.

Multiplexing: Combines coded residual, code indices and predictor coefficients to form the compressed bitstream.

For each channel, a prediction residual is calculated using linear prediction with adaptive predictor coefficients and (preferably) adaptive prediction order in each block. The coefficients are quantized prior to filtering and transmitted as side information. The prediction residual is entropy coded using one of several different entropy codes (e.g. Rice code). The indices of the chosen codes have to be transmitted. Finally, a multiplexing unit combines coded residual, code indices, predictor coefficients and other additional information to form the compressed bitstream.

The codec provides techniques to verify the decoded data and to ensure that the compressed data is losslessly decodable. Additional encoder options comprise block length switching, random access, joint stereo coding, multi-channel correlation etc. Different levels of complexity are offered by the codec through several encoding options, which in turn will give different compression levels with differing encoding decoding times.

For application where latency is critical, (e.g. music telepresence) it is appropriate to abstain from the highest compression in order to reduce the computational effort and thereby latency.

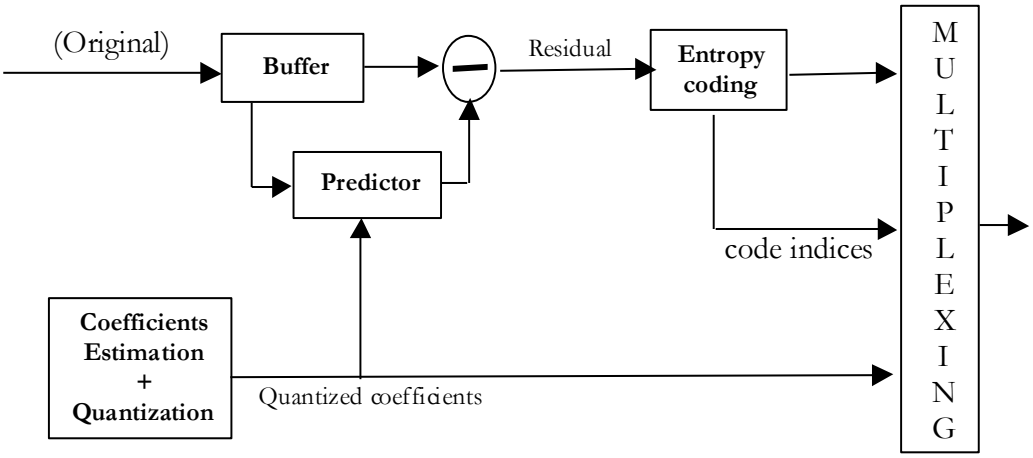

Fig. 3.1 MPEG4-ALS Encoder

3.1.2.2 Decoder Structure

The MPEG-4 ALS decoder (Fig. 3.2) is significantly less complex than the encoder. It decodes the entropy-coded residual and, using the predictor coefficients, calculates the lossless reconstruction signal. The computational effort of the decoder mainly depends on the order of the predictor chosen by the encoder.

Since the maximum order usually depends on the encoder's compression level, higher compressed files might take slightly longer to decode. Apart from the predictor order, the decoder complexity is nearly independent from the encoder options.

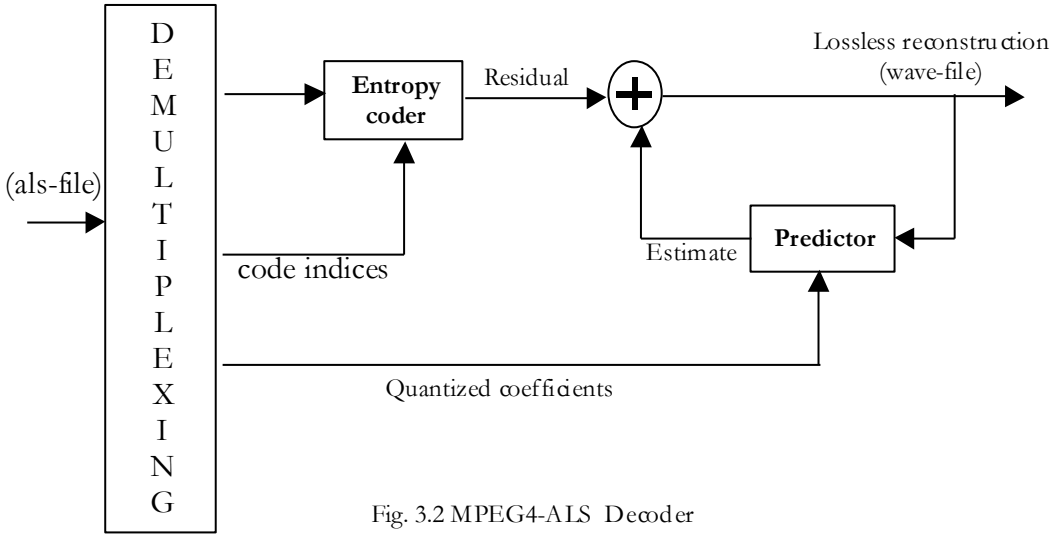

Fig. 3.2 MPEG4-ALS Decoder

3.1.3 Linear Predictive Coding

It is well known that most audio signals have significant amount of correlation between samples. They have harmonic or periodic components originating from the fundamental frequency or pitch of musical instrument. In order to reduce the entropy of the signal, decorrelation of the signals is done prior to entropy coding. In audio lossless coding decorrelation is done by predictive coding. Predictive coding predicts the value of the current signal from past samples using non-linear or linear prediction techniques. As the name implies, non-linear prediction predicts the current sample using a non-linear combination of past samples. Due to the difficulty in approximating the prediction function, non-linear prediction is not extensively used in audio compression. Most of the lossless audio coders including MPEG4-ALS implements Linear Predictive Coding (LPC). The current sample of a time-discrete signal can be approximately predicted from the previous samples.

If the predicted samples are close to the original samples, the residual has a smaller variance than the original sample itself; hence it can be encoded more efficiently.

Normal linear prediction, especially the short-term linear prediction utilizes the correlation between neighboring samples to reduce the amplitude. Speech and audio signals sometimes have long-term correlation due to the pitch. Long-term prediction can further reduce the prediction residual after short-term prediction by capturing the periodic components of audio signals. Multi-tap LTP is sequentially applied to the short-term prediction residual signal. To reduce the amplitude, the best delay parameter is found and a set of predictive coefficients is calculated. These are also compressed by Rice code and transmitted as side information.

The integer value of the prediction residual signal and the quantized partial autocorrelation coefficients obtained from the prediction parameters are transmitted to the decoder. The decoder has a recursive filter that can reconstruct the original waveform losslessly from the transmitted bitstream.

The analysis method for linear prediction is not a concern in the standard bitstream. The Levinson-Durbin (LD) method is implemented in the reference software (http://www.ics.uci.edu/~euzun/pub/267.pdf) though other methods, such as the Burg method, the covariance-lattice method, and Laguerre-based pure linear prediction (L-PLP), are acceptable for ALS.

The MPEG4-ALS codec uses forward-adaptive *Linear Predictive Coding (LPC)* to reduce bit rates compared to PCM, leaving the optimization entirely to the encoder to implement. Thus, various encoder implementations are possible, offering a certain range in terms of efficiency and complexity.

In forward linear prediction, the optimal predictor coefficients (in terms of a minimized variance of the residual) are usually estimated for each block by the autocorrelation method or the covariance method. The autocorrelation method, using the Levinson-Durbin algorithm, has additionally the advantage of providing a simple means to iteratively adapt the order of the predictor.

Increasing the predictor order decreases the variance of the prediction error, leading to a smaller bit rate for the residual. On the other hand, the bit rate for the predictor coefficients will rise with the number of coefficients to be transmitted. Thus, the task is to find the optimal order that minimizes the total bit rate.

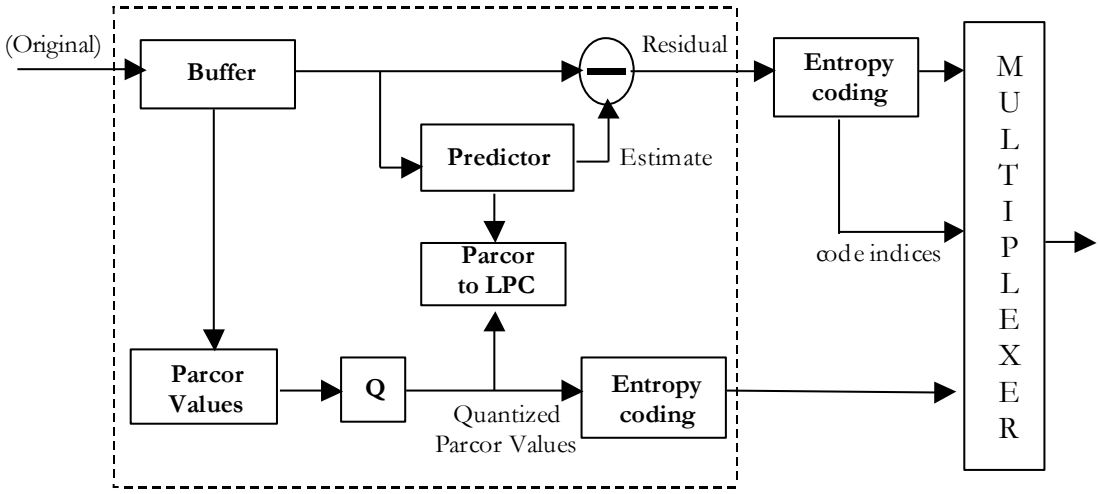

Fig. 3.3 Encoder of forward-adaptive prediction scheme

The Levinson-Durbin algorithm determines recursively all predictors with increasing order. For each order, a complete set of predictor coefficients is calculated. Moreover, the variance of the corresponding residual can be calculated, resulting in an estimate of the expected bit rate for the residual.

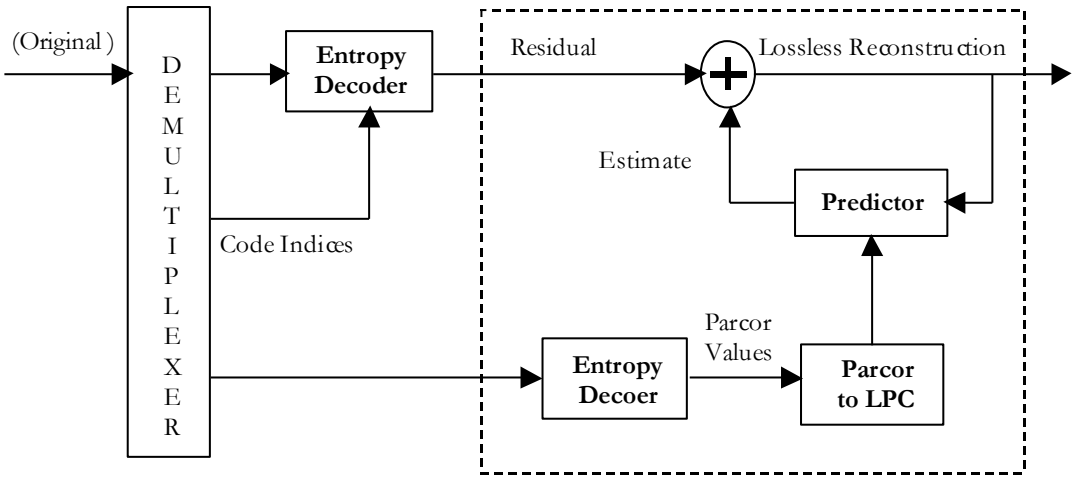

Fig. 3.4 Decoder of forward-adaptive prediction scheme

Together with the bit rate for the coefficients, the total bit rate can be determined in each iteration, i.e. for each predictor order. The optimal order is set at the point where the total bit rate no longer decreases.

3.1.4 Entropy Coding of Residual

The residual values are entropy coded using Rice Codes by default. Alternatively, the encoder can use a more complex and efficient coding Scheme called BGMC (Block Gilbert-Moore Codes).

For each block, either all values can be encoded using the same Rice code, or the block can be further divided into four parts, each encoded with a different Rice code. The indices of the applied codes have to be transmitted. Since there are different ways to determine the optimal Rice code for a given set of data, it is up to the encoder to select suitable codes depending on the statistics of the residual.

BGMC is a more complex and efficient coding scheme. In BGMC mode, the encoding of residuals is accomplished by splitting them in two categories: Residuals that belong to a

central region of the distribution and ones that belong to its tails. The residuals in tails are simply re-centered and encoded using Rice codes as described earlier. However, to encode residuals in the center of the distribution, the BGMC encoder splits them into LSB and MSB components first, then it encodes MSBs using block Gilbert-Moore (arithmetic) codes, and finally it transmits LSBs using direct Fixed Length Codes. Increased complexity of BGMC leads to more encoder/decoder time and therefore Rice code is preferred over BGMC in applications where latency is critical.

3.1.5 Encoder Options

The ALS encoder is designed to offer different compression levels. While the maximum level achieves the highest compression at the expense of slowest encoding speed.

3.1.5.1 Block Length Switching

The basic version of the encoder uses one sample block per channel in each frame, where the frame length can initially be adjusted to the sampling rate of the input signal. While the frame length is constant for one input file, optional *block length switching* enables a subdivision into four shorter sub-blocks to adapt to transient segments of the audio signal.

3.1.5.2 Random Access

Random access enables fast access to any part of the encoded audio signal without costly decoding of previous parts. The encoder optionally generates bitstream information allowing random access at intervals of several frames by inserting frames that can be decoded without decoding previous frames where no samples from previous frames are used for prediction. For enabling fast search, each random access frame starts with an info

field that specifies the distance in bytes to the next random access frame. Selecting Random Access option will increase the codec complexity and thereby will increase compression time.

3.1.5.3 Independent coding

Different channels are coded independently. When the correlation between the channel are not so significant, then independent coding may be more efficient than joint-channel coding due to the additional overhead associated with the joint channel coding.

3.1.5.4 Joint Stereo Coding

Joint stereo coding can be used to exploit dependencies between the two channels of a stereo signal. One of the ways to exploit correlation between channels is to the difference signal. The intra-channel correlations and inter-channel correlations among samples are exploited by the RLS-LMS predictor through *joint-stereo prediction*, where past samples from both left and right audio channels are used in estimating the current sample of each channel. Join stereo coding is used to exploit dependencies between the two channels and is done by Difference Coding i.e. by encoding the difference signal. $d(n) = x2(n) – x1(n)$
To improve compression performance for multi-channel signals, adaptive subtraction from reference channels with weighting factors is applied based on inter-channel dependencies. At least one channel has to be encoded independently in order to decode all channels losslessly.

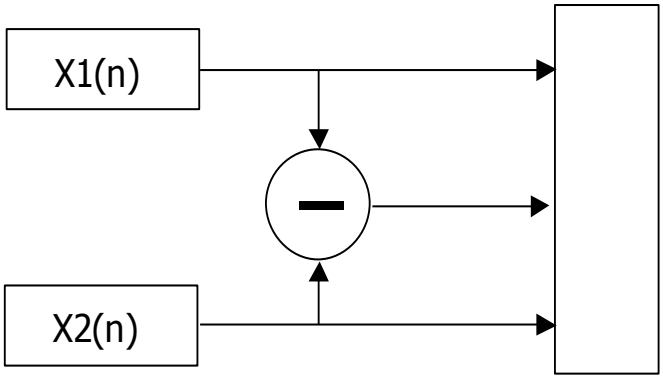

Fig. 3.5: Differential coding

3.1.5.5 Multichannel Correlation

There is inter-channel correlation between multiple channels. Inter-channel prediction is applied to the prediction error to reduce the amplitude of the prediction residual after short-term linear prediction or long-term prediction. For multichannel coding, a search is performed to find the channel-pair combination that provides the maximum inter-channel correlation.

For the selected channel-pair, multi-tap inter-channel prediction is applied. In addition, the relative delay parameter between the channel-pair is found and the associated weighting coefficients are determined. All these coefficients are then quantized and compressed by Rice code. Lossless audio coding technology will be widely used for compressing various multichannel signals, such as wave-field-synthesis, bio-medical, and seismic signals as well as surround audio signals.

To improve the compression performance for these multichannel signals, adaptive subtraction from reference channels with weighting factors is applied. This process is based on the inter-channel dependence of the time domain prediction residual signal. At least one channel must be encoded in the independent coding mode for lossless decoding of all the channels.

3.2 Tuning MPEG4-ALS for Music Compression

Lossless audio codec MPEG4-ALS is used for the compression of classical music tracks. Since it is lossless, the usability and robustness depends not on quality issues, but on the following,

Latency (Speed of compression and decompression)

Compression ratio

Software and hardware support

Error correction

Based on the characteristics of classical music, major techniques used to optimize and analyze the MPEG4-ALS codec includes optimizing codec complexity by applying encoding options such as independent coding, joint-stereo coding, multi-channel coding and entropy coding of Residual viz. Rice coding and BGMC (Block Gilbert-Moore Codes).

By applying linear prediction over varying frame lengths and downsampling the audio tracks, the dependencies of frame length and sample rates with latency and compression ratio are analyzed.

3.2.1 Characteristics of Classical Music

Classical music encompasses many styles of music spanning over 700 years. Classical music composures usually impart unity and logic to music. A classical composition has a wealth of rhythmic patterns. The classical style also includes unexpected pauses, syncopations, and frequent changes from long notes to shorter ones. And the change from one pattern of note lengths to another may be either sudden or gradual. Classical music is basically homophonic. However, texture is treated as flexibly as rhythm. Pieces shift smoothly or suddenly from one texture to another. A work may begin homophonically (characterized by a single melodic line with accompaniment) but then changes to a more complex polyphonic texture that features two simultaneous melodies or melodic fragments imitated among the various instruments.

The crucial differences with the previous wave can be seen in the downward shift in melodies, increasing durations of movements, the acceptance of Mozart and Haydn as paradigmatic, the greater use of keyboard resources, the shift from "vocal" writing to "pianistic" writing, the growing pull of the minor and of modal ambiguity, and the increasing importance of varying accompanying figures to bring "texture" forward as an element in music. In short, the late Classical was seeking a music that was internally more complex.

The growth of concert societies and amateur orchestras, marking the importance of music as part of our everyday life, contributed to a booming market for pianos, piano music, and virtuosi to serve as examplars. Therefore, considering the special charectaristics of the classical music and by effectively tuning the parameters of audio codec, audio compression

could be optimized. Here I used the classical music tracks to simulate and experiment the effect of adjusting the parameters to optimize compression within acceptable latency.

3.2.2 Codec Complexity

The encoder and decoder complexity of MPEG 4-ALS codec depends on the combination of techniques and options that are applied to the codec. Increasing the levels of encoder and decoder complexity may increase compression ratio but at the expense of encoding and decoding time. The effect of codec complexity on latency and compression ratio is analyzed. By determining the latency limit (e.g., 20 ms) the codec can be optimized to a low delay coder with optimum compression ratio to enable the codec to be used in internet musical applications.

Some of the techniques and options that increase algorithmic complexity of the codec include Independent coding, joint-stereo coding, multi-channel, adaptive linear Prediction, multi-channel prediction and entropy coding of the residual. Some of the options of MPEG-4 ALS codec that determines the operating points (levels) in terms of compression and complexity are:

Options	Description
-a	Adaptive prediction order
-b	Use BGMC codes for prediction residual (default: use Rice codes)
-i	Independent stereo coding (turn off joint stereo coding)
-n#	Frame length: 0 = auto (default), max = 65536
-o#	Prediction order (default = 10), max = 1023
-p	Use long-term prediction
-s#	Multi-channel correlation (#=1-65536, jointly code every # channels) *(# must be a divisor of number of channels, otherwise -s is ignored)*
-t#	Two methods mode (Joint Stereo and Multi-channel correlation) *(# must be a divisor of number of channels)*

Table 3.1 MPEG 4-ALS Encoding Options

Codec complexity increases as we choose join stereo coding or multi-channel correlation instead of independent coding due to the additional computation required to exploit correlation between the different channels.

Codec complexity increases as we choose forward adaptive linear prediction to improve the compression ratio. Complexity increases further when the prediction is extended to multiple prediction of any number of channels where, for a particular channel, the estimate (prediction) can be calculated using previous samples from all channels. Computation of all the orders of automatic prediction selection by using Levinson-Durbin algorithm and choosing the optimal order becomes much more complicated as the number of channels increases. Adaptive prediction yields better compression ratio with the expense of latency when more channels are correlated with each other.

In addition to the test conducted with different classical music stereo tracks, two (2) identical channels are combined together to make stereo channels with 100% correlation before applying joint stereo, multi-channel correlation, adaptive prediction and long term prediction to test the trade offs between compression ratio and codec latency (encoder + decoder time).

By default, the residual values are entropy coded using Rice codes. Alternatively, the encoder can use a more complex and efficient coding Scheme called BGMC (Block Gilbert-Moore codes). The entropy-coding options Golomb-Rice codes and Block Gilbert-Moore codes (BGMC) are applied to compress various classical music stereo tracks to analyze the trade offs between latency and compression ratio. The results are tabulated in the following chapter.

3.3 Frame Length

Compression ratio and latency results from applying linear prediction over the varying frame lengths are analyzed to quantify the dependency between frame length and compression ratio/latency to determine the optimal frame length that can yield maximum compression ratio within latency limits. Increase in frame length increases the computational complexity and thus increasing the latency but improves compression performance.

3.4 Downsampling

Sampling rate of the audio tracks can be increased or decreased using the process of upsampling and downsampling the audio tracks. Audio downsampling (also called subsampling) is the process of reducing the sampling rate of an audio track which in turn will reduce the data rate and thus the size of the data. The downsampling factor (*M*) is usually an integer or a rational fraction greater than unity which divides the sampling rate.

Here, compact disc audio tracks with sampling rate 44,100 Hz are downsampled to 22,050 Hz (M=2), 11,025 Hz(M=4) and so on. Compression ratio and latency varies with respect to sampling rate. By downsampling same audio tracks the dependency between sample rate with compression ratio and latency is analyzed.

Below is a list of sampling rate and its common use:

Sample Rate	Common use
8,000 Hz	Telephone
11,025 Hz	Lower-quality PCM, MPEG audio and for audio analysis of subwoofer bandpasses
22,050 Hz	Lower-quality PCM and MPEG audio and for audio analysis of low frequency energy. Suitable for digitizing early 20th century audio formats.
32,000 Hz	MiniDV digital video camcorder, video tapes with extra channels of audio (e.g. DVCAM with 4 Channels of Audio), DAT (LP mode), high-quality digital wireless microphones.
44,100 Hz	Audio CD, also most commonly used with MPEG-1 audio (VCD, SVCD, MP3), adopted from the PCM adaptor using PAL videotapes.

Table 3.2 Audio Sample Rate and Common Use

Compression ratio and latency varies with respect to sampling rate. By downsampling same audio tracks the dependency between sample rate with compression ratio and latency is analysed.

3.5 Summary

This chapter gives an overview of the MPEG4-ALS technology. General features and applications of MPEG4-ALS are described in details. Schematic descriptions of encoder and decoder structures are explained in this chapter. Decorrelation techniques such as short-term prediction and long-term prediction are utilized in ALS. Short-term prediction exploit the correlations among neighboring audio samples, and long-term prediction captures the periodic components in audio signals, in order to further reduce the prediction residual. Linear predictive coding is commonly used in ALS as it has less

complexity compared to other predictors. The residual signal is entropy coded using either the Rice code, or the more complex but more efficient Gilbert-Moore arithmetic code. Various encoder options such as Block Length Switching, Random Access, Joint stereo coding and multichannel correlation are also briefly discussed in this chapter.

Optimization of MPEG4-ALS codec for Internet related musical applications, where latency is critical is discussed. Various options of the codec are used to find the relationship between frame length and codec complexity with compression ratio and latency. Also audio tracks of different sample rates are used to derive the effect of sample rate on latency and compression ratio. Detailed analyses of experimental results are done in the following chapter.

Chapter 4
Test Results and Analysis

Here various classical music tracks are compressed using MPEG 4 Audio Lossless System using different options. Complexity variations in algorithm, which is proportional to latency, and compression ratio, are used to assess the effectiveness of using data compression in interactive multimedia network applications where latency is critical.

4.1 Experimental Platform

Described here are some of the results obtained using MPEG4-ALS on Audio (classical music) tracks. Latency (encoding + decoding time) and compression ratio variation are recorded and reported graphically.

Test was done on 12 stereo classical music tracks with the following Audio Properties:

Sample type: integer

Resolution: 16 bit

Sample Rate: 44100 Hz

Channels: 2

4.2 Comparing Codec Complexity Levels

By applying various available options to MPEG4-ALS codec, the compression performance can be increased by increasing the complexity levels of the codec algorithm. There is a trade off between the compression performance and latency.

By optimizing the level of complexity, reasonable compression can be achieved by keeping the latency within the allowed limit. Joint-channel coding, multichannel Coding (MCC), long-term prediction, etc. increases the codec complexity and there by latency, but offer better compression performance.

Test was done on stereo classical music tracks with Codec (MPEG4-ALS) encoding options independent, MCC, joint channel, adaptive prediction, long-term prediction, etc. used. Rice code is used as entropy coding. Default options are selected for other encoder options.

Track#	PCM Size (MB)	Independent		Joint Channel		MCC	
		Comp. Ratio	CPU Time (sec)	Comp. Ratio	CPU Time (sec)	Comp. Ratio	CPU Time (sec)
T1	91.74	2.17	16.75	2.17	24.75	2.17	44.20
T2	77.24	2.37	14.18	2.37	20.71	2.37	37.68
T3	42.88	2.16	7.67	2.16	11.66	2.16	20.67
T4	42.83	2.09	8.05	2.09	11.53	2.09	20.41
T5	85.19	2.11	15.85	2.11	23.05	2.12	41.43
T6	81.60	2.39	15.70	2.39	22.16	2.40	39.51
T7	48.16	2.23	9.01	2.23	13.20	2.24	23.99
T8	51.79	2.07	9.67	2.08	13.89	2.09	24.85
T9	77.04	2.04	14.29	2.04	20.97	2.04	37.69
T10	77.29	2.29	14.28	2.28	20.54	2.29	37.56
T11	54.44	2.09	10.18	2.09	14.96	2.10	26.65
T12	64.62	1.91	12.25	1.91	17.41	1.91	31.21

Table 4.1 Average Comparisons of different codec complexity levels, Independent, joint channel, and MCC

Track#	PCM Size (MB)	MCC & joint-Channel		Adaptive Prediction		Long-Term Prediction	
		Comp. Ratio	CPU Time (sec)	Comp. Ratio	CPU Time (sec)	Comp. Ratio	CPU Time (sec)
T1	91.74	2.17	67.90	2.17	19.72	2.19	96.24
T2	77.24	2.37	57.43	2.37	16.55	2.40	80.29
T3	42.88	2.16	31.76	2.16	9.31	2.19	44.42
T4	42.83	2.09	31.36	2.09	9.30	2.11	44.56
T5	85.19	2.11	63.45	2.11	17.90	2.13	90.75
T6	81.60	2.40	61.04	2.40	17.20	2.42	87.44
T7	48.16	2.24	35.90	2.23	10.52	2.25	51.10
T8	51.79	2.08	37.93	2.08	11.23	2.10	53.56
T9	77.04	2.04	57.31	2.04	16.81	2.06	81.48
T10	77.29	2.29	57.56	2.29	16.21	2.31	80.76
T11	54.44	2.10	40.87	2.09	11.92	2.12	57.58
T12	64.62	1.91	48.25	1.91	13.72	1.93	67.84

Table 4.2 Comparisons of different codec complexity levels, Long-Term Prediction, Adaptive Prediction, and MCC & Joint channel

L#	Description	T1		T2		T3		T4		T5	
		Time	C.R	Time	C.R	Time	C.R	Time	C.R	Time	C.R
1	Independent	16.75	2.17	14.18	2.37	7.67	2.16	8.05	2.09	15.85	2.11
2	Adaptive Prediction	19.72	2.17	16.55	2.37	9.31	2.16	9.3	2.09	17.9	2.11
3	Joint channel	24.75	2.17	20.71	2.37	11.66	2.16	2.09	11.53	23.05	2.11
4	MCC	44.2	2.17	37.68	2.37	20.67	2.16	20.41	2.09	41.43	2.12
5	MCC & joint channel	67.9	2.17	57.43	2.37	31.76	2.16	31.36	2.09	63.45	2.11
6	Long Term Prediction	96.24	2.19	80.29	2.4	44.42	2.19	44.56	2.11	90.75	2.13

Table 4.3 Comparisons of different codec complexity levels in sorted order, Independent Coding, Adaptive Prediction, Joint channel coding, MCC, MCC & Joint channel and Long-Term Prediction

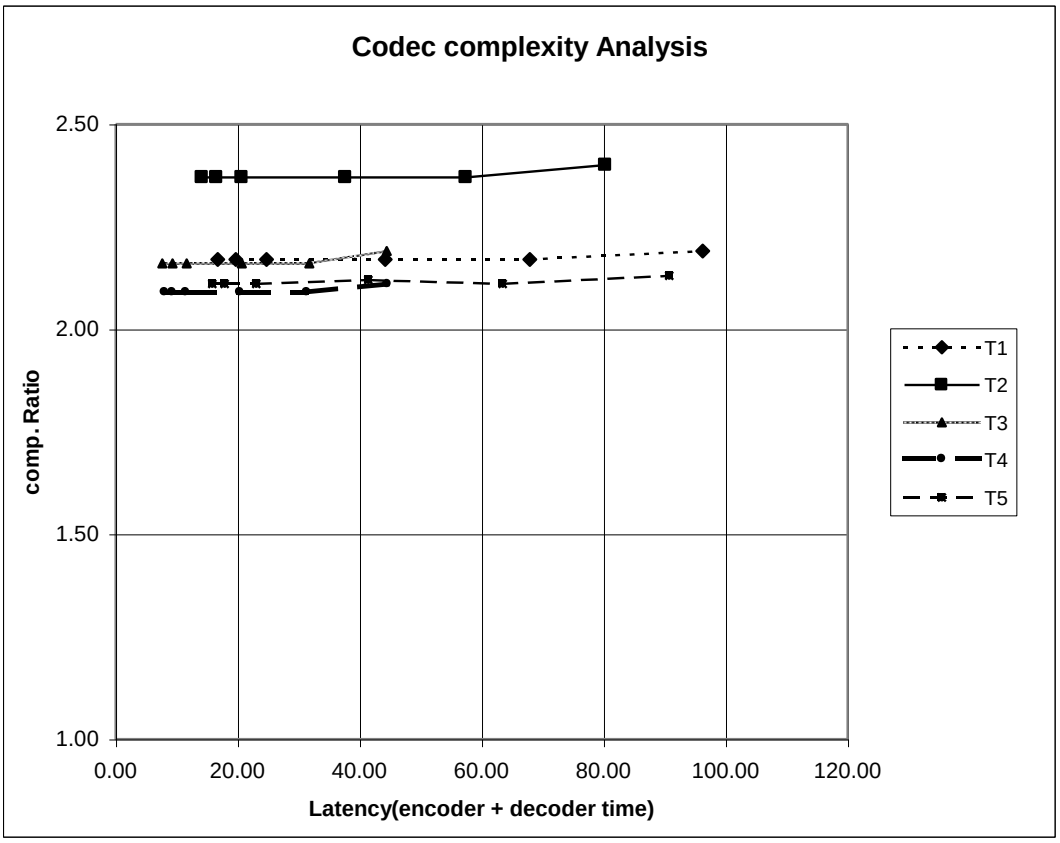

Fig 4.1 Graph showing the variations of Compression Ratio with respect to codec complexity
Codec Latency plotted against Compression Ratio

When we analyze the experimental results of applying different codec complexity levels on stereo classical music tracks as shown in table 4.1, 4.2, 4.3 and fig. 4.1, it becomes obvious that the compression ratio improvements are not significant even though latency is much higher for higher complexity levels.

Therefore, for applications where latency is critical, it is preferable to use simple algorithms for compressing classical music audio tracks. From the experimental results we see that the compression ratio is between 2.0 to 2.5. Joint stereo coding option did not yield expected performance over independent coding. These results imply that correlation between the stereo channels in classical music tracks seems to be lower than expected. This may be due to the nature of classical music tracks with high sampling rate.

In order to assess how the overall latency (codec latency + network latency) is affected by applying ALS codec for compressing the audio before streaming it to the network, we should calculate the bits saved (reduced) by the application of compression per unit compression time. Kilobytes saved per second by the applications of compression are tabulated for five classical music tracks are as follows.

Bytes saved by compression per milliseconds ($S_{KB/ms}$) can be calculated by the formula,

$$S_{KB/ms} = \frac{F_{KB} * (C-1)}{C * T_{ms}}$$

where F_{KB} = size of the file (KB) before compressing,

C = compression Ratio and

T_{ms} = CPU Time (ms) taken for compression.

(KB/ms = File Size in Kilobytes / Compression time incurred in milliseconds)

#	**Tracks**	**T1**	**T2**	**T3**	**T4**	**T5**
	File Size (Bytes)	**96199196.00**	**80988812.00**	**44967932.00**	**44909143.00**	**89326652.00**
1	Independent	3.02	3.22	3.07	2.84	2.90
2	Adaptive	2.57	2.76	2.53	2.46	2.56
3	Joint channel	2.05	2.21	2.02	1.98	1.99
4	MCC	1.15	1.21	1.14	1.12	1.11
5	MCC & joint channel	0.75	0.80	0.74	0.73	0.72
6	Long Term Prediction	0.53	0.57	0.54	0.52	0.51
		(KB/ms)	(KB/ms)	(KB/ms)	(KB/ms)	(KB/ms)

Table 4.4 Comparisons of different codec complexity levels- KB/ms saved

In real-time network applications where latency is critical, rather than the amount of compression achieved, we should take into account the amount of compression achieved with respect to the time taken for compression. The following is a plot showing the effect of codec complexity on the rate of kilobytes saved (original file size minus compressed file size) by compression.

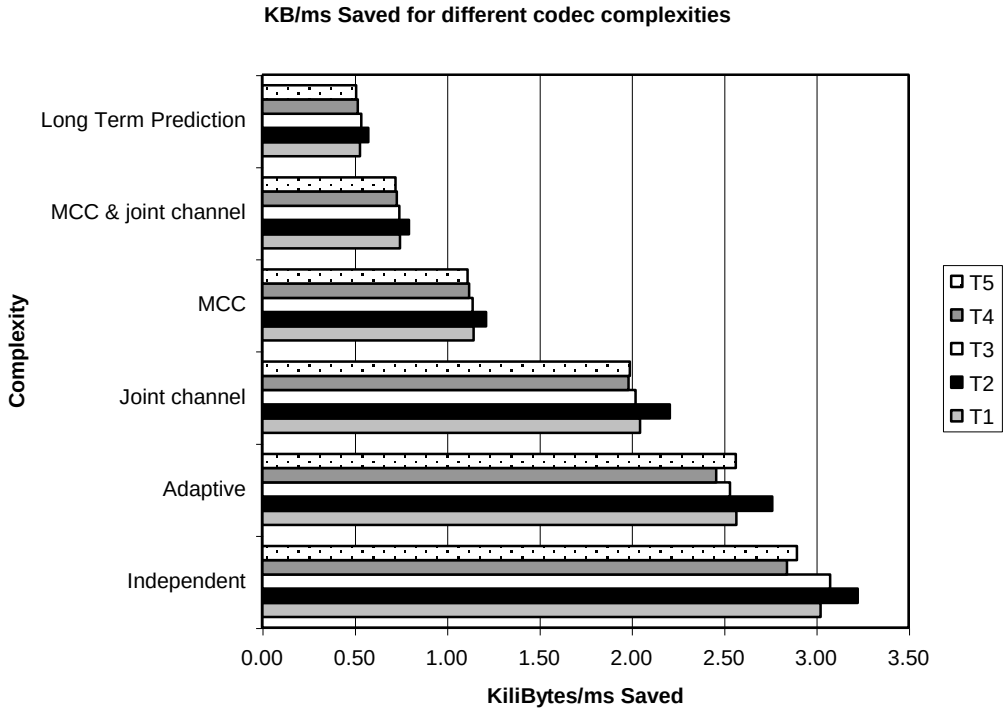

Fig 4.2 Graph showing the variations of Kilo Bytes saved per Milliseconds (KB/ms) with respect to codec complexity

From the above results, we clearly see the tradeoffs between compression performance and latency while using different encoder options for compression. Independent coding or the encoding option with lesser complexity yielded better result and therefore simple encoder option is obviously the better choice to achieve reasonable compression performance within acceptable limits of latency.

4.3 Multi-Channel Correlation

Experimental results when two (2) classical music tracks tested with different codec options.

Audio characteristics:

Resolution: 16-bit
Sample Rate: 44100 Hz
Channels: 2
Format: Wave

Codec options:

Independent, joint channel, MCC, MCC & joint channel, and adaptive prediction

Other options: default

Tracks Properties:

T1 – 2 channels Left & Right
T2 – 2 channels Left & Right
T1L – 2 channels Left & Left
T1R – 2 channels Right & Right
T2L – 2 channels Left & Left
T2R – 2 channels Right & Right

Tracks	Independent Coding		Joint channel Coding		Adaptive Prediction		**MCC**		MCC & joint Channel	
	Comp Ratio	CPU Time (Sec)	Comp Ratio	CPU Time (Sec)	Comp Ratio	CPU Time (Sec)	Comp Ratio	CPU Time (Sec)	Comp Ratio	CPU Time (Sec)
T1	1.367	3.10	1.367	4.55	1.369	3.49	1.369	8.00	1.368	12.32
T1L	1.356	3.11	2.257	4.44	2.265	3.36	1.960	7.92	2.256	12.15
T1R	1.375	3.11	2.284	4.44	2.291	3.25	1.972	7.92	2.283	12.17
T2	1.512	2.60	1.512	3.84	1.514	3.04	1.514	6.73	1.513	10.41
T2L	1.507	2.62	2.463	3.77	2.471	2.77	2.153	6.69	2.462	10.26
T2R	1.513	2.62	2.471	3.75	2.479	2.76	2.144	6.68	2.469	10.23

Table 4.5 Comparisons of different codec options on tracks having different degree of correlations

The artificially created stereo tracks T1L, T2L, T1R and T2R have 100% correlations between its left and right channels as the two channels are identical to each other and results shows that they compressed almost twice as much compared to T1 or T2 joint channel encoding options are used. The experimental observations confirm that the codec is able to exploit the correlation between channels and generally, the classical music tracks have very less correlation between its left and right channels. Therefore compressing those tracks with joint channel encoding option turned off will be more preferable in applications like telepresence where latency is critical.

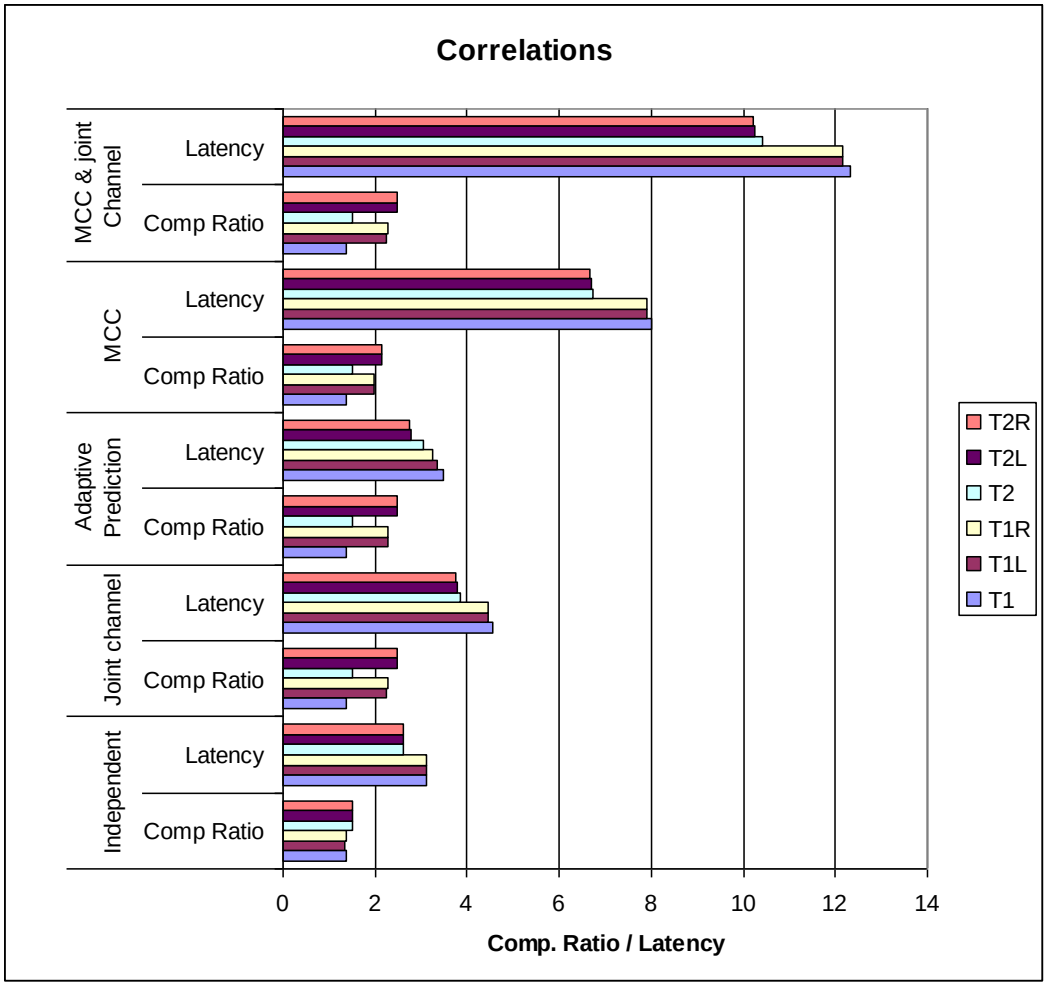

Fig 4.3 Comparisons of different codec options on tracks having different degree of correlations

4.4 Variations in Compression with Frame Lengths

T1 – Track # 1, Stereo, Sample Rates (8K, 11K, 22K & 44K)

T2 – Track # 2, Stereo, Sample Rates (8K, 11K, 22K & 44K)

T3 – Track # 3, Stereo, Sample Rates (8K, 11K, 22K & 44K)

T1R – Track # 1, Right Channel (Mono), Sample Rates (8K, 11K, 22K & 44K)

T2R – Track # 2, Right Channel (Mono), Sample Rates (8K, 11K, 22K & 44K)

T3R – Track # 3, Right Channel (Mono), Sample Rates (8K, 11K, 22K & 44K)

T1L – Track # 1, Left Channel (Mono), Sample Rates (8K, 11K, 22K & 44K)

T2L – Track # 2, Left Channel (Mono), Sample Rates (8K, 11K, 22K & 44K)

T3L – Track # 3, Left Channel (Mono), Sample Rates (8K, 11K, 22K & 44K)

[Sampling Rate = 8K]

Tracks	T1 (Stereo – 8K)			T2 (Stereo – 8K)			T3 (Stereo – 8K)		
Size (*bytes*)	**17451150**			**14691898**			**8157498**		
N	**C. R.**	**Latency**	**KB/ms**	**C. R.**	**Latency**	**KB/ms**	**C. R.**	**Latency**	**KB/ms**
128	1.335	2.41	1.77	1.468	2.09	2.19	1.331	1.15	1.72
256	1.354	2.41	1.85	1.493	2.05	2.31	1.350	1.17	1.77
512	1.364	2.50	1.82	1.508	2.11	2.29	1.361	1.16	1.82
1024	1.369	2.43	1.89	1.514	2.06	2.36	1.365	1.15	1.85
2048	1.370	2.48	1.86	1.515	2.11	2.31	1.365	1.20	1.78

Table 4.6 Variation in Latency and Comp. Ratio with frame length for tracksT1, T2 & T3
Tracks with 8K sampling rate

Tracks	**T1R (Mono-8K)**			**T2R (Mono-8K)**			**T3R (Mono-8K)**		
Size (*bytes*)	**8725616**			**7345990**			**4078790**		
N	**C. R.**	**Latency**	**KB/ms**	**C. R**	**Latency**	**KB/ms**	**C. R.**	**Latency**	**KB/ms**
128	1.384	1.27	1.86	1.529	1.11	2.24	1.380	0.61	1.80
256	1.404	1.26	1.95	1.557	1.07	2.40	1.400	0.61	1.87
512	1.415	1.27	1.97	1.572	1.10	2.37	1.412	0.61	1.91
1024	1.420	1.26	2.00	1.579	1.06	2.48	1.416	0.58	2.02
2048	1.421	1.24	2.04	1.580	1.05	2.51	1.416	0.58	2.02

Table 4.7 Variation in Latency and Comp. Ratio with frame length for tracksT1R, T2R & T3R
Tracks with 8K sampling rate

Tracks	T1L (Mono-8K)			T2L (Mono-8K)			T3L (Mono-8K)		
Size (*bytes*)	8725616			7345990			4078790		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	1.384	1.27	1.86	1.529	1.10	2.26	1.380	0.62	1.77
256	1.404	1.27	1.93	1.557	1.08	2.38	1.400	0.62	1.84
512	1.415	1.29	1.94	1.572	1.10	2.37	1.412	0.60	1.94
1024	1.420	1.25	2.02	1.579	1.05	2.51	1.416	0.58	2.02
2048	1.421	1.25	2.02	1.580	1.05	2.51	1.416	0.57	2.05

Table 4.8 Variation in Latency and Comp. Ratio with frame length for tracksT1L, T2L & T3L
Tracks with 8K sampling rate

Sampling Rate = 11K

Tracks	T1 (Stereo – 11K)			T2 (Stereo – 11K)			T3 (Stereo – 11K)		
Size (*bytes*)	24049846			20247250			11242030		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	1.405	3.35	2.02	1.555	2.85	2.48	1.403	1.55	2.03
256	1.430	3.33	2.12	1.581	2.81	2.59	1.421	1.52	2.14
512	1.441	3.37	2.13	1.595	2.90	2.54	1.432	1.58	2.10
1024	1.446	3.35	2.16	1.601	2.87	2.59	1.436	1.53	2.18
2048	1.445	3.35	2.16	1.601	2.80	2.65	1.432	1.46	2.27

Table 4.9 Variation in Latency and Comp. Ratio with frame length for tracks T1, T2 & T3
Tracks with 11K sampling rate

Tracks	T1R (Mono-11K)			T2R (Mono-11K)			T3R (Mono-11K)		
Size (*bytes*)	12024746			10123482			5620954		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	1.466	1.75	2.13	1.626	1.48	2.57	1.457	0.82	2.10
256	1.487	1.68	2.29	1.653	1.47	2.66	1.478	0.82	2.16
512	1.499	1.76	2.22	1.669	1.47	2.70	1.489	0.82	2.20
1024	1.504	1.73	2.27	1.676	1.44	2.77	1.493	0.80	2.27
2048	1.503	1.60	2.46	1.675	1.38	2.89	1.490	0.72	2.51

Table 4.10 Variation in Latency and Comp. Ratio with frame length for tracksT1R, T2R & T3R
Tracks with 11K sampling rate

Tracks	T1L (Mono-11K)			T2L (Mono-11K)			T3L (Mono-11K)		
Size (*bytes*)	12024746			10123482			5620954		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	1.466	1.75	2.13	1.626	1.50	2.54	1.457	0.81	2.13
256	1.487	1.70	2.26	1.653	1.45	2.69	1.478	0.81	2.19
512	1.499	1.74	2.25	1.669	1.48	2.68	1.489	0.83	2.17
1024	1.504	1.72	2.29	1.676	1.44	2.77	1.493	0.80	2.27
2048	1.503	1.67	2.35	1.675	1.35	2.95	1.490	0.72	2.51

Table 4.11 Variation in latency and comp. ratio with frame length for tracks T1L, T2L & T3L tracks with 11K sampling rate

Sampling Rate = 22K

Tracks	T1 (Stereo – 22K)			T2 (Stereo – 22K)			T3 (Stereo – 22K)		
Size (*bytes*)	48099634			40494442			22484002		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	1.670	6.15	3.06	1.843	5.10	3.55	1.658	2.95	2.95
256	1.694	5.85	3.29	1.872	5.07	3.63	1.682	2.85	3.12
512	1.708	6.25	3.12	1.889	5.28	3.52	1.695	2.98	3.02
1024	1.714	6.30	3.11	1.897	5.30	3.53	1.702	2.99	3.03
2048	1.715	6.15	3.18	1.889	5.25	3.54	1.703	3.02	3.00

Table 4.12 Variation in latency and comp. ratio with frame length for tracks T1, T2 & T3 tracks with 22K sampling rate

Tracks	T1R (Mono-22K)			T2R (Mono-22K)			T3R (Mono-22K)		
Size (*bytes*)	24049858			20247262			11242042		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	1.749	3.28	3.07	1.943	2.71	3.54	1.735	1.55	3.00
256	1.776	3.15	3.26	1.976	2.63	3.71	1.761	1.48	3.21
512	1.790	3.24	3.20	1.994	2.70	3.65	1.776	1.55	3.09
1024	1.798	3.23	3.23	2.003	2.73	3.63	1.783	1.55	3.11
2048	1.800	3.33	3.13	2.006	2.82	3.52	1.784	1.56	3.09

Table 4.13 Variation in latency and comp. ratio with frame length for tracks T1R, T2R & T3R tracks with 22K sampling rate

Tracks	T1L (Mono-22K)			T2L (Mono-22K)			T3L (Mono-22K)		
Size (*bytes*)	24049858			20247262			11242042		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	1.749	3.20	3.14	1.943	2.70	3.55	1.735	1.55	3.00
256	1.776	3.12	3.29	1.976	2.60	3.76	1.761	1.49	3.18
512	1.790	3.20	3.24	1.994	2.75	3.58	1.776	1.54	3.11
1024	1.798	3.24	3.22	2.003	2.75	3.60	1.783	1.53	3.15
2048	1.800	3.33	3.13	2.006	2.91	3.41	1.784	1.55	3.11

Table 4.14 Variation in latency and comp. ratio with frame length for tracks T1L, T2L & T3L tracks with 22K sampling rate

Sampling Rate = 44K

Tracks	T1 (Stereo – 44K)			T2 (Stereo – 44K)			T3 (Stereo – 44K)		
Size (*bytes*)	96199196			80988812			44967932		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	2.063	11.51	4.21	2.261	9.75	4.52	2.061	5.54	4.08
256	2.119	11.49	4.32	2.316	9.77	4.60	2.114	5.56	4.16
512	2.147	12.51	4.01	2.346	10.63	4.27	2.140	6.01	3.89
1024	2.161	12.92	3.91	2.363	11.01	4.14	2.154	6.19	3.80
2048	2.168	13.01	3.89	2.372	11.00	4.16	2.160	6.25	3.77

Table 4.15 Variation in latency and comp. ratio with frame length for tracks T1, T2 & T3 tracks with 44K sampling rate

Tracks	T1R (Mono-44K)			T2R (Mono-44K)			T3R (Mono-44K)		
Size (*bytes*)	48099646			40494454			22484014		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	2.188	6.16	4.14	2.417	5.18	4.48	2.183	2.91	4.09
256	2.252	6.10	4.28	2.479	5.17	4.56	2.243	2.89	4.21
512	2.283	6.60	4.00	2.514	5.51	4.32	2.272	3.13	3.93
1024	2.299	6.85	3.87	2.533	5.70	4.20	2.288	3.17	3.90
2048	2.307	6.93	3.84	2.543	5.79	4.14	2.295	3.30	3.75

Table 4.16 Variation in latency and comp. ratio with frame length for tracks T1R, T2R & T3R tracks with 44K sampling rate

Tracks	T1L (Mono-44K)			T2L (Mono-44K)			T3L (Mono-44K)		
Size (*bytes*)	48099646			40494454			22484014		
N	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms	C. R.	Latency	KB/ms
128	2.188	6.16	4.14	2.417	5.21	4.45	2.183	2.99	3.98
256	2.252	6.10	4.28	2.479	5.17	4.56	2.243	2.91	4.18
512	2.283	6.60	4.00	2.514	5.55	4.29	2.272	3.08	3.99
1024	2.299	6.71	3.96	2.533	5.70	4.20	2.288	3.17	3.90
2048	2.307	6.82	3.90	2.543	5.90	4.07	2.295	3.25	3.81

Table 4.17 Variation in latency and comp. ratio with frame length for tracks T1L, T2L & T3L tracks with 44K sampling rate

4.4.1 Compression Ratio Vs Frame Length

Music tracks with sample rate 8K, 11K, 22K and 44K are used for compression. The variations in compression ratio as frame length takes values 110, 220, 440, 880, 1760 and 4400 are plotted.

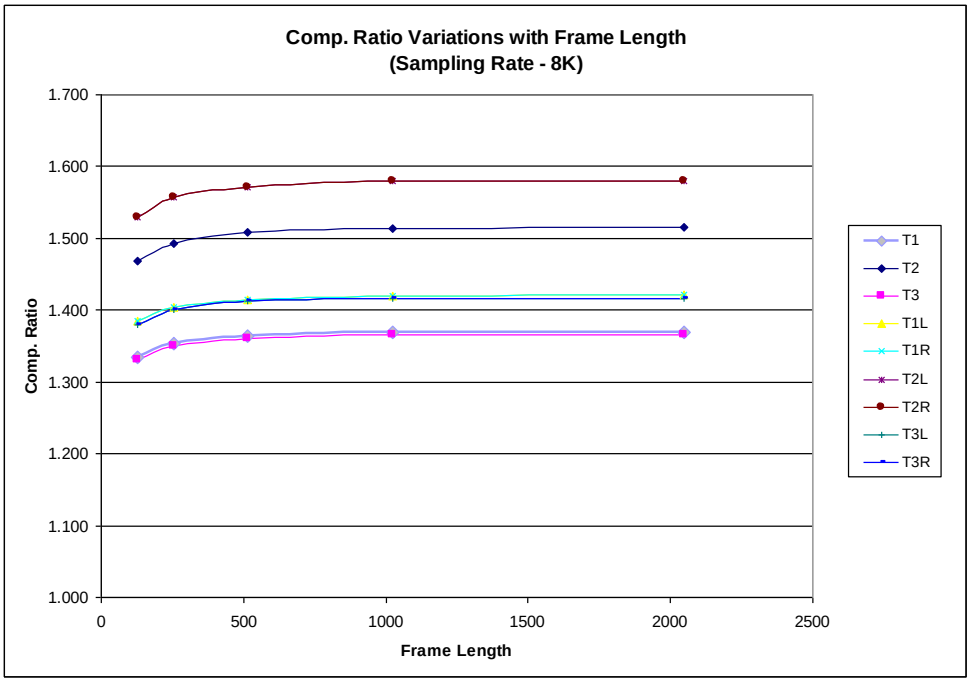

Fig. 4.4 Variation in Comp. Ratio with frame length for Tracks with 8K sampling rate

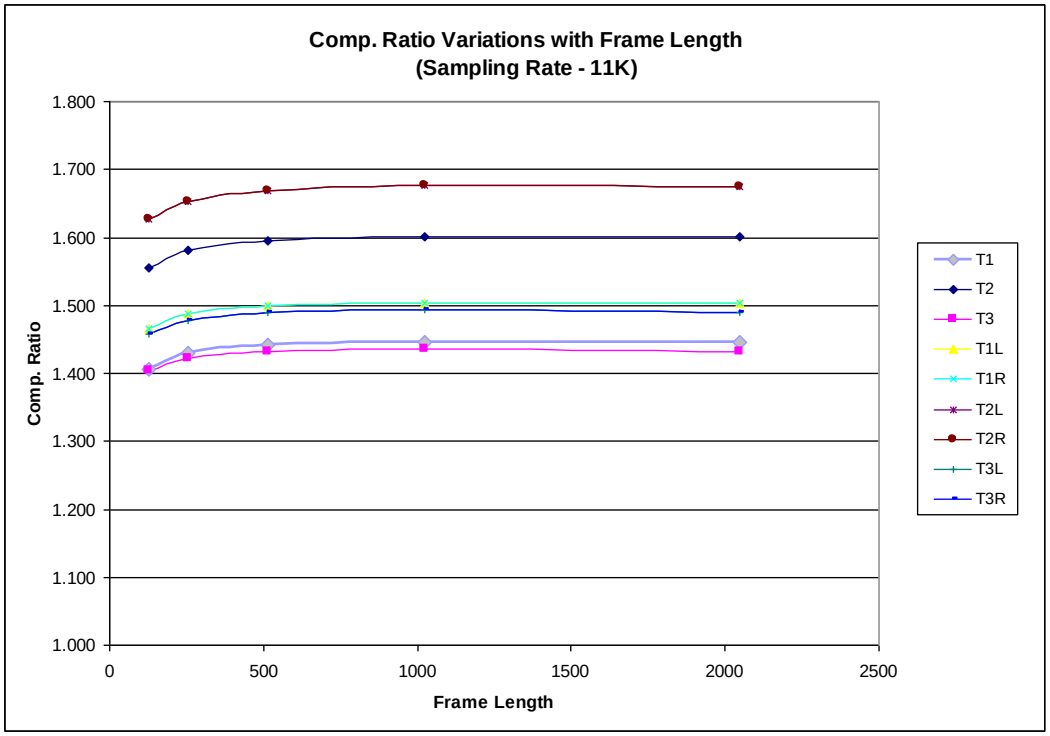

Fig. 4.5 Variation in Comp. Ratio with frame length for Tracks with 11K sampling rate

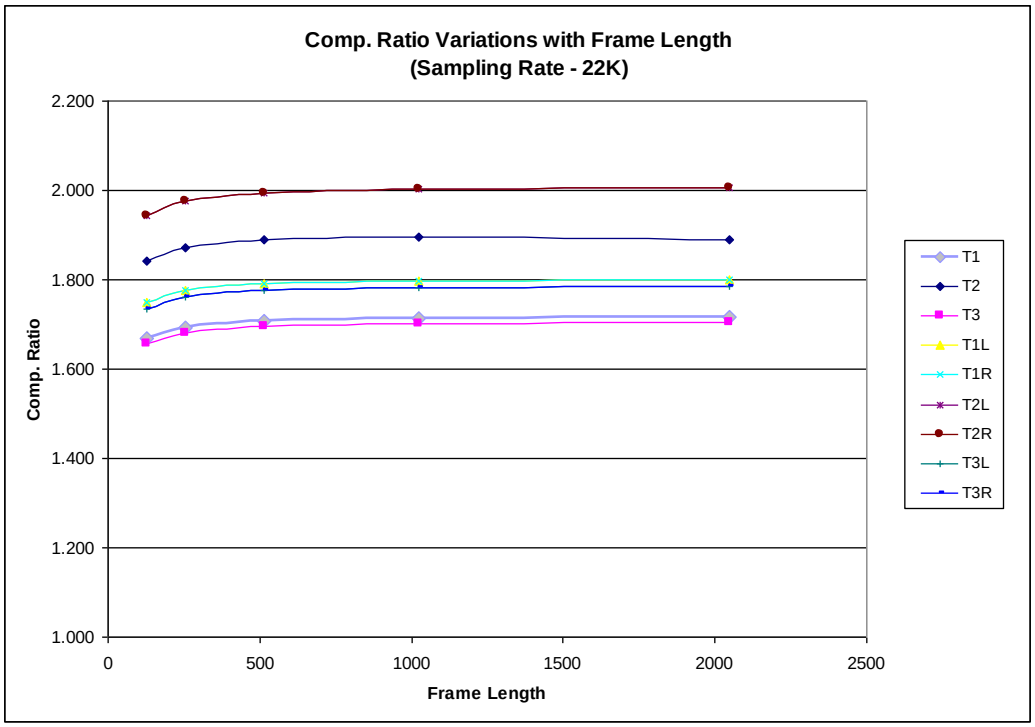

Fig. 4.6 Variation in Comp. Ratio with frame length for Tracks with 22K sampling rate

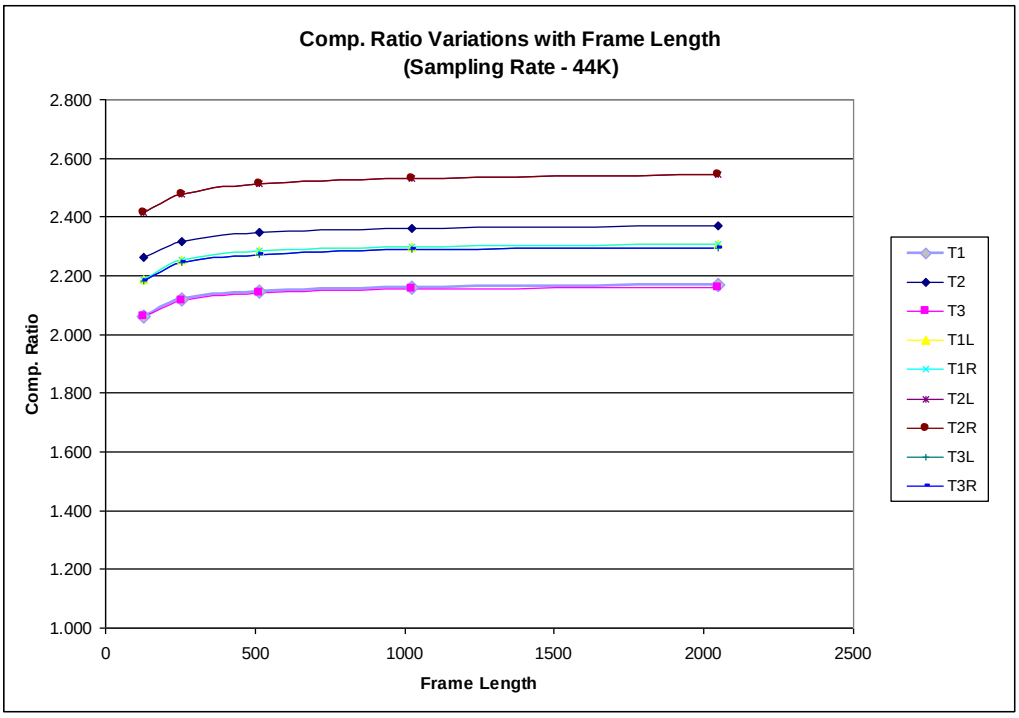

Fig. 4.7 Variation in Comp. Ratio with frame length for Tracks with 44K sampling rate

Experimental results in Fig. 4.4 - 4.7 show that the variation of compression ratio with frame length. Compression performance always slightly increases with frame length. However, compression ratio saturates when frame length reaches around 1000. The variation in compression ratio is seen always less than 0.2.

Results also indicate that the long-term prediction do not yield satisfactory results in the case of classical music signals. Codec yields satisfactory compression performance even when the frame length is very small. Compression therefore can be applied to audio signals of < 5ms data. MPEG4-ALS can be successfully used for audio streaming even if the packet size is chosen to be very small, which is the use for music telepresence.

4.4.2 Latency Vs Frame Length

Classical music tracks with sample rate 8K, 11K, 22K and 44K are used for compression. The variations in latency as frame length takes values 110, 220, 440, 880, 1760 and 4400 are plotted.

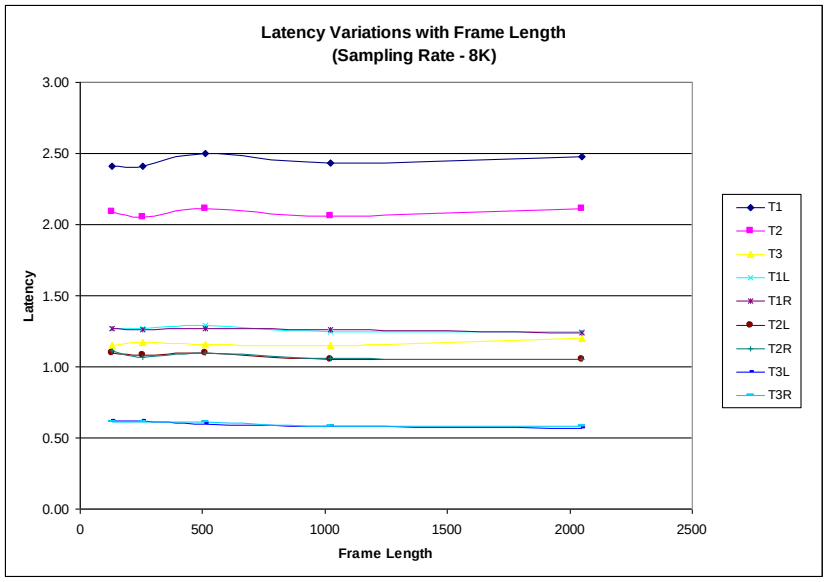

Fig. 4.8 Variation in latency with frame length for tracks with 8K sampling rate

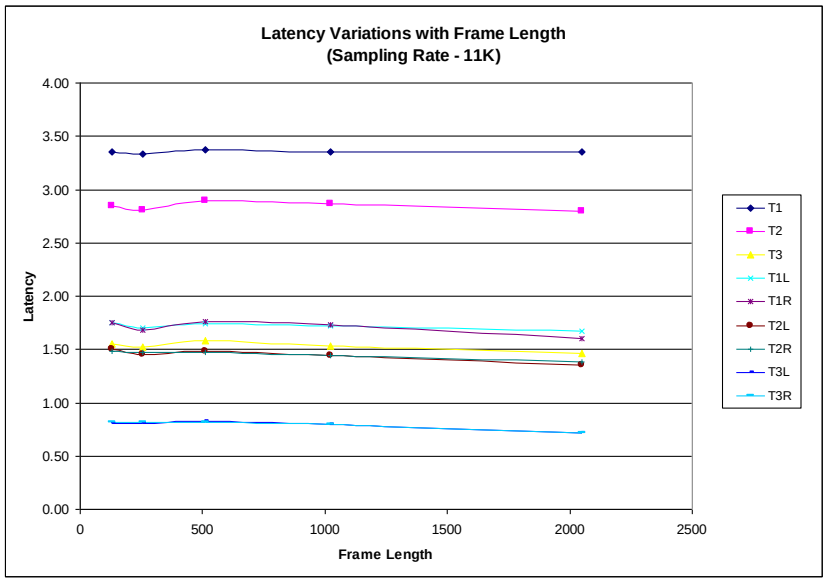

Fig. 4.9 Variation in latency with frame length for tracks with 11K sampling rate

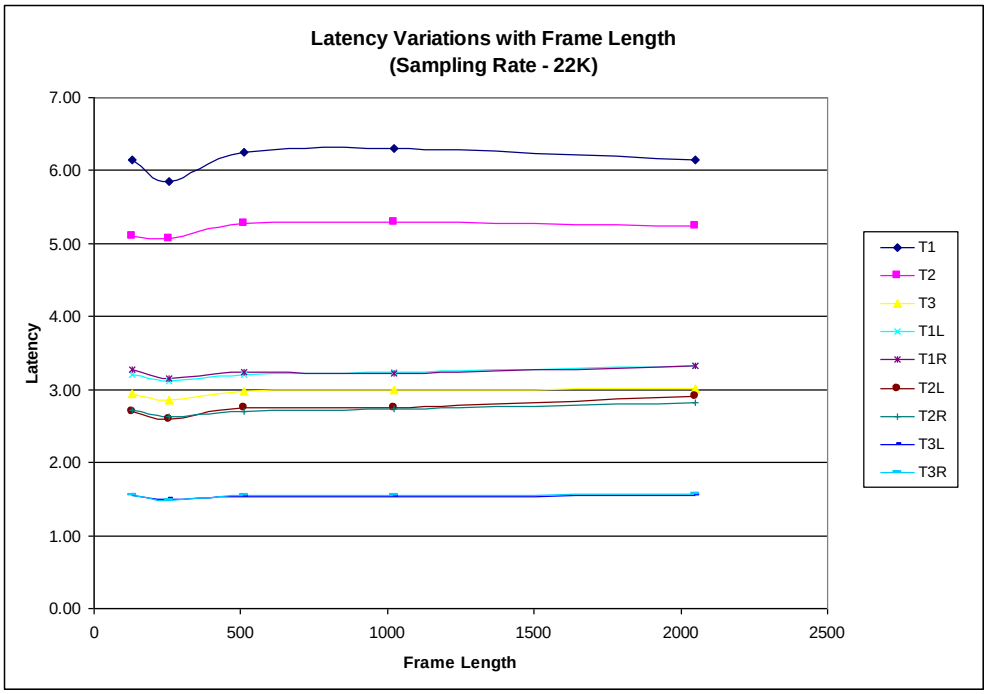

Fig. 4.10 Variation in latency with frame length for tracks with 22K sampling rate

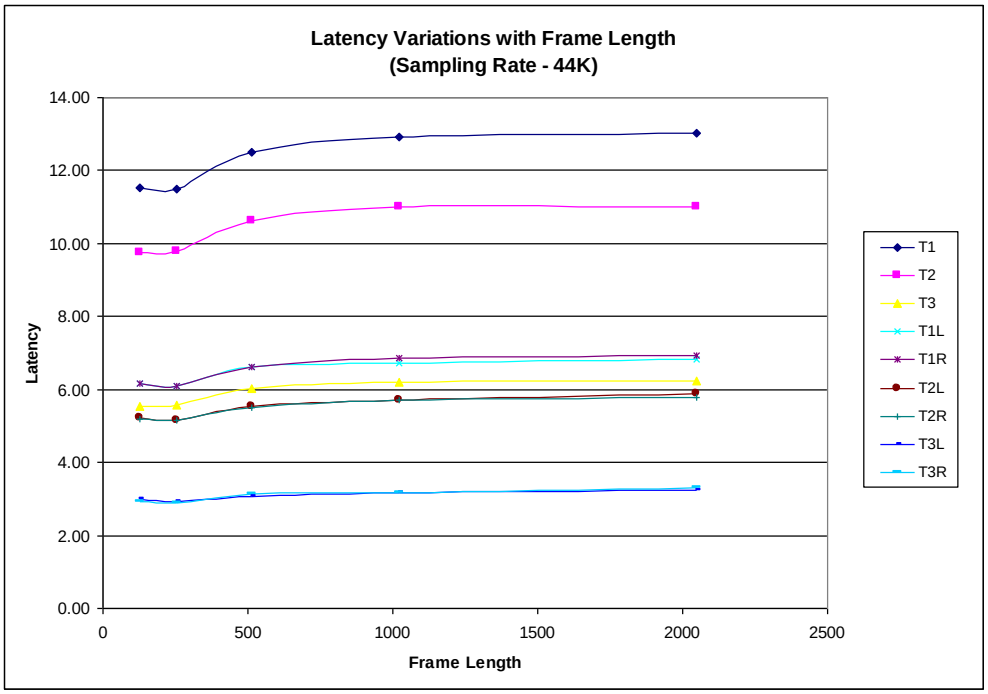

Fig. 4.11 Variation in latency with frame length for tracks with 44K sampling rate

Like compression ratio, experimental results show that the variation of latency with frame length also is almost constant. Latency increases with frame length.

4.4.3 KB/ms Saved Vs Frame Length

Music tracks with sample rate 8K, 11K, 22K and 44K are used for compression. The variations in KB/ms saved by applying compression as frame length takes values 110, 220, 440, 880, 1760 and 4400 are plotted.

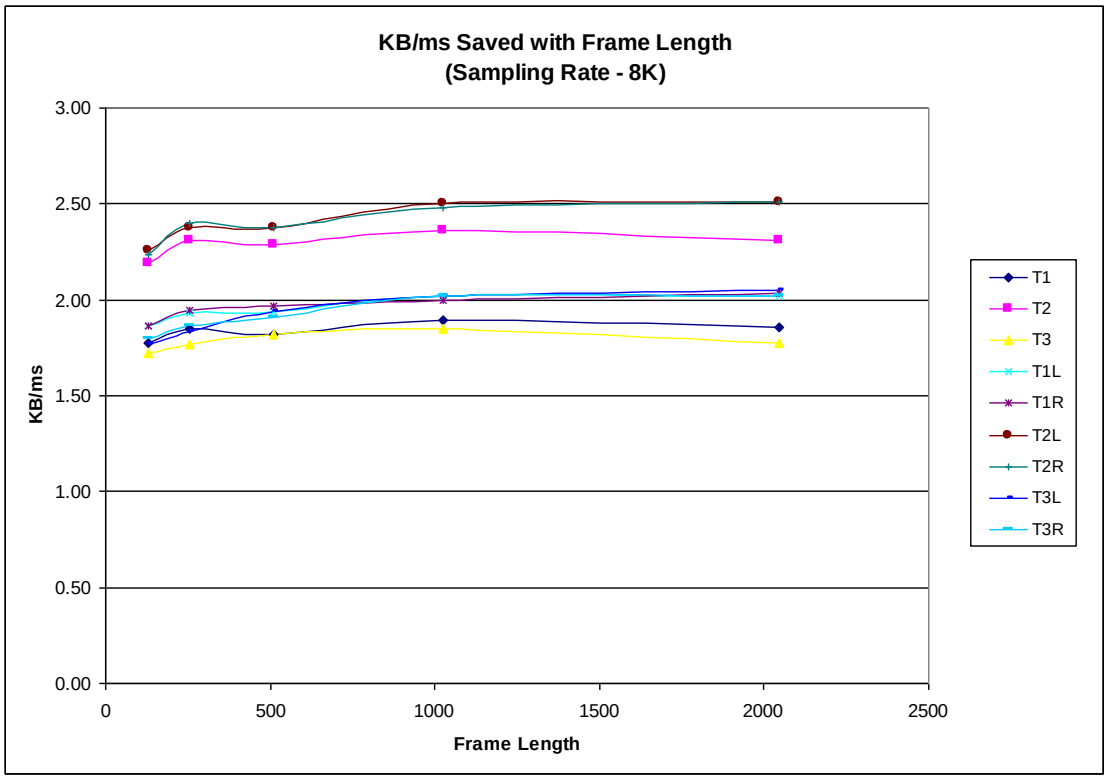

Fig. 4.12 Variation in File size reduced/ms with frame length for Tracks with 8K sampling rate

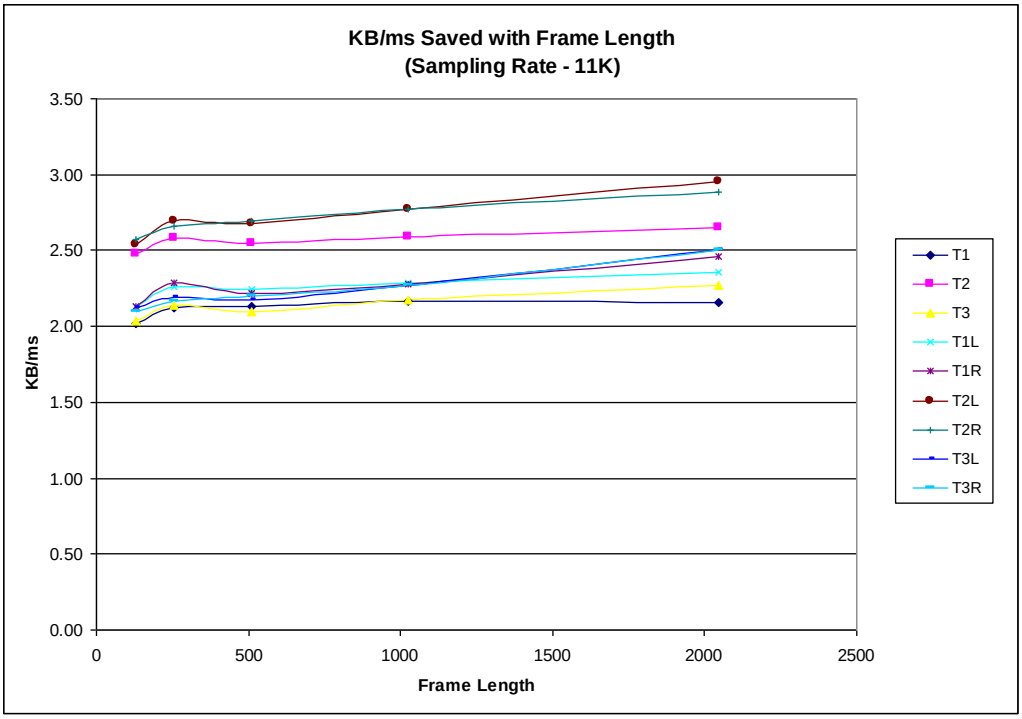

Fig. 4.13 Variation in File size reduced/ms with frame length for Tracks with 11K sampling rate

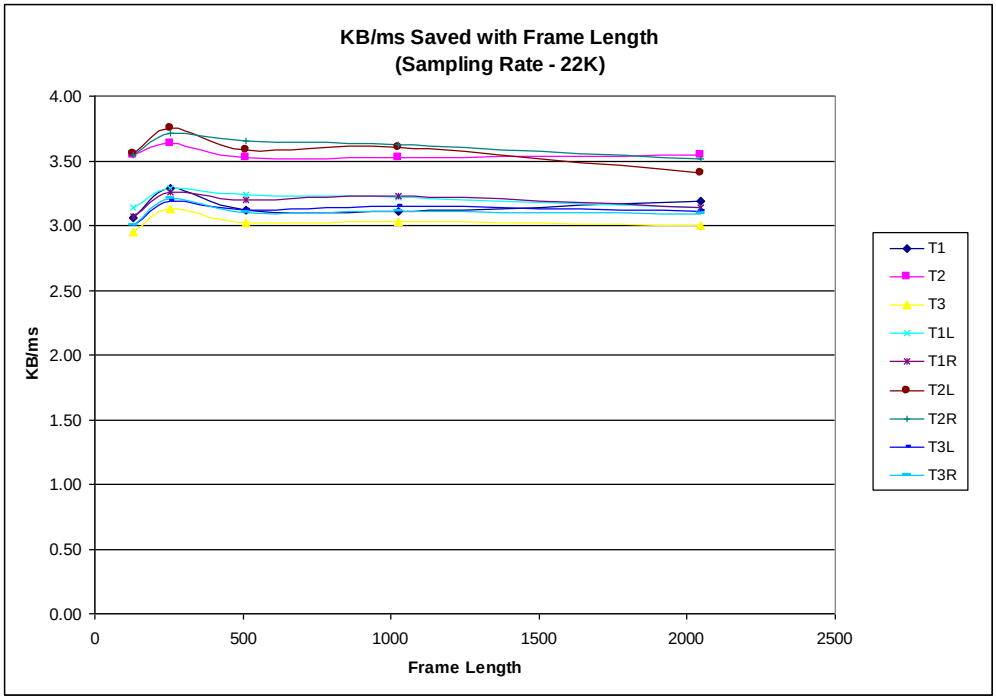

Fig. 4.14 Variation in File size reduced/ms with frame length for Tracks with 22K sampling rate

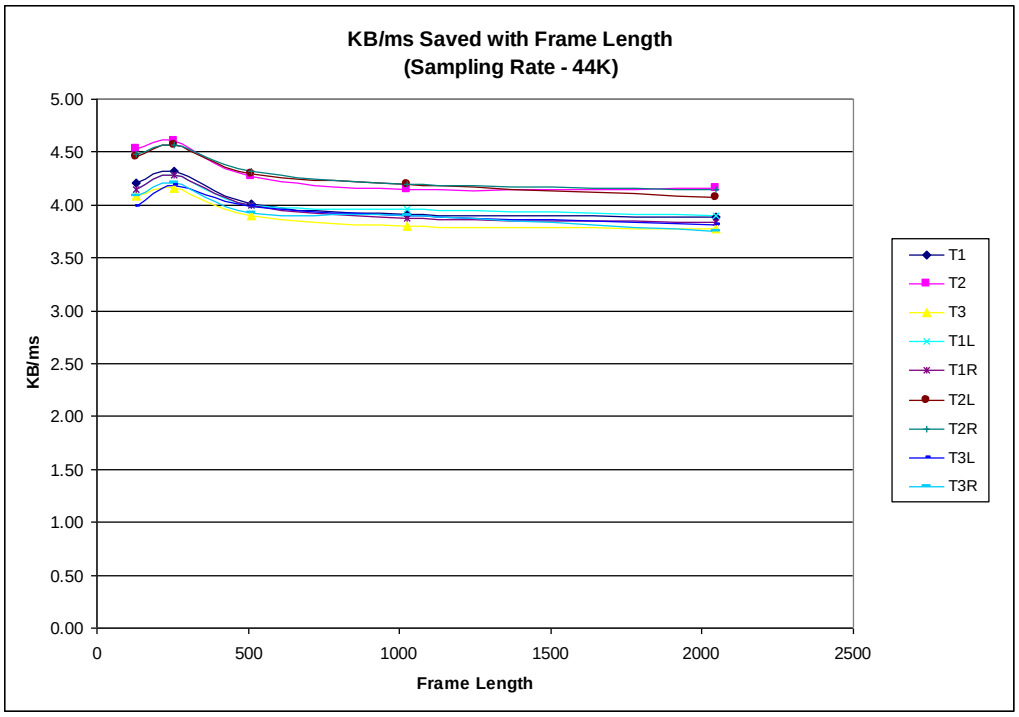

Fig. 4.15 Variation in File size reduced/ms with frame length for Tracks with 44K sampling rate

Like compression ratio and latency, compression file size reduced/ms, experimental results show that the variation of compression file size reduced/ms with frame length also is almost a constant. File size reduced/ms with frame length increases initially (when frame length =256) then slightly decreases with frame length.

4.4.4 Sampling Rate Vs Compression Ratio

Classical music tracks with sample rate 8K, 11K, 22K and 44K are plotted against the latency for frame lengths 110, 220, 440, 880, 1760 and 4400.

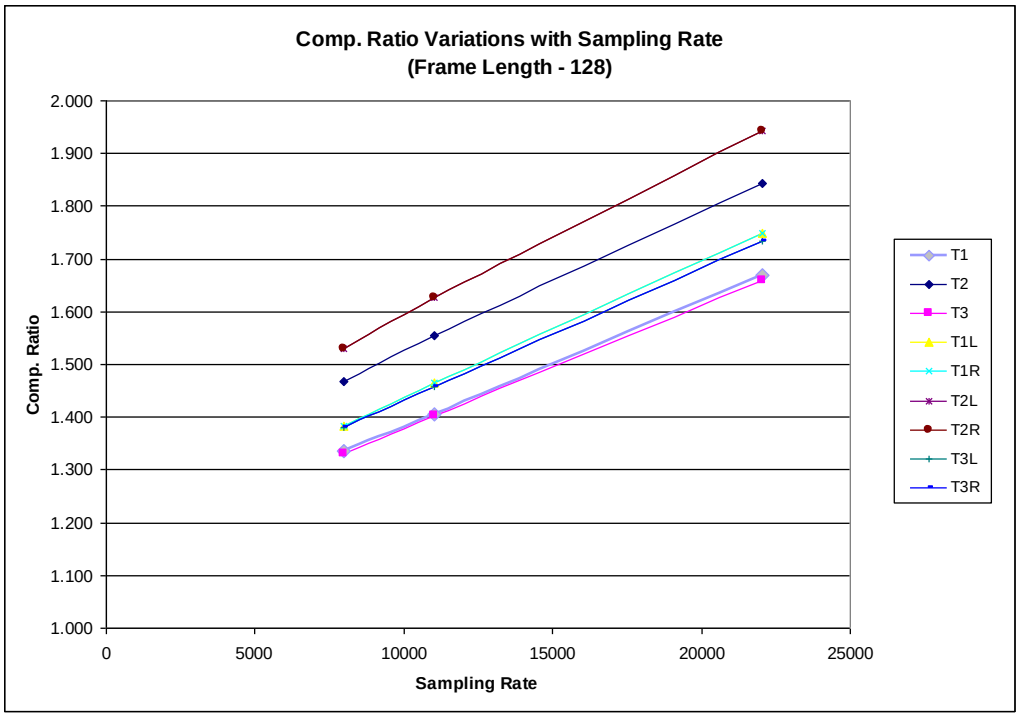

Fig. 4.16 Variation in compression ratio with sampling rate (frame length =128)

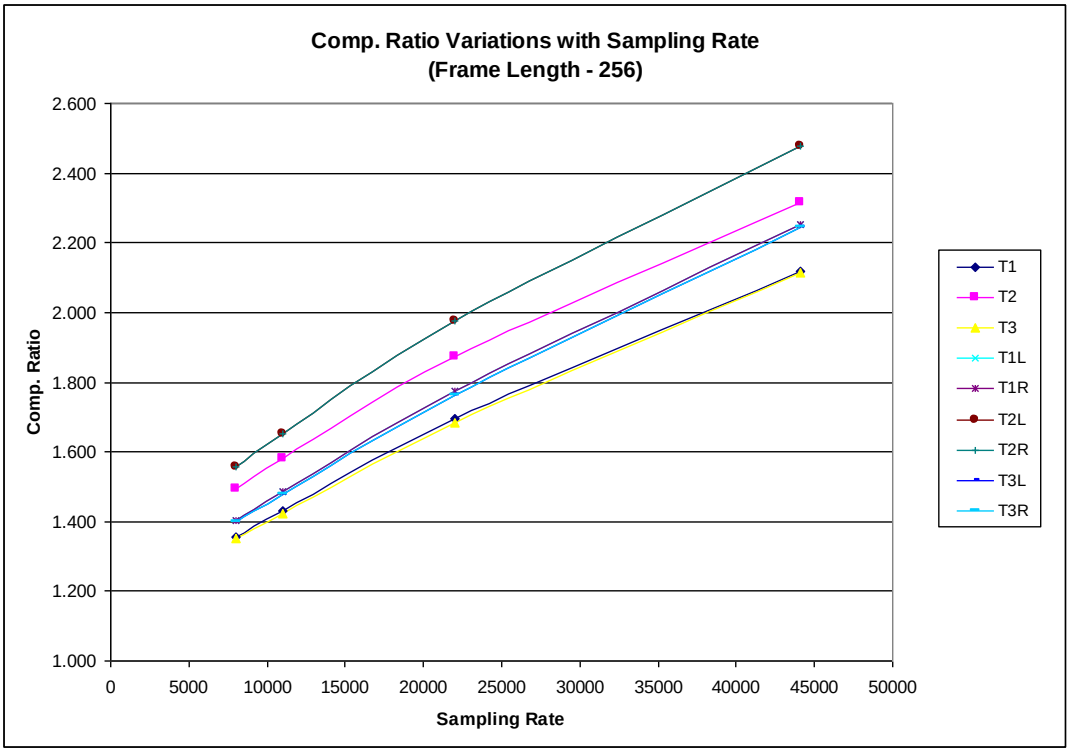

Fig. 4.17 Variation in compression ratio with sampling rate (frame length =256)

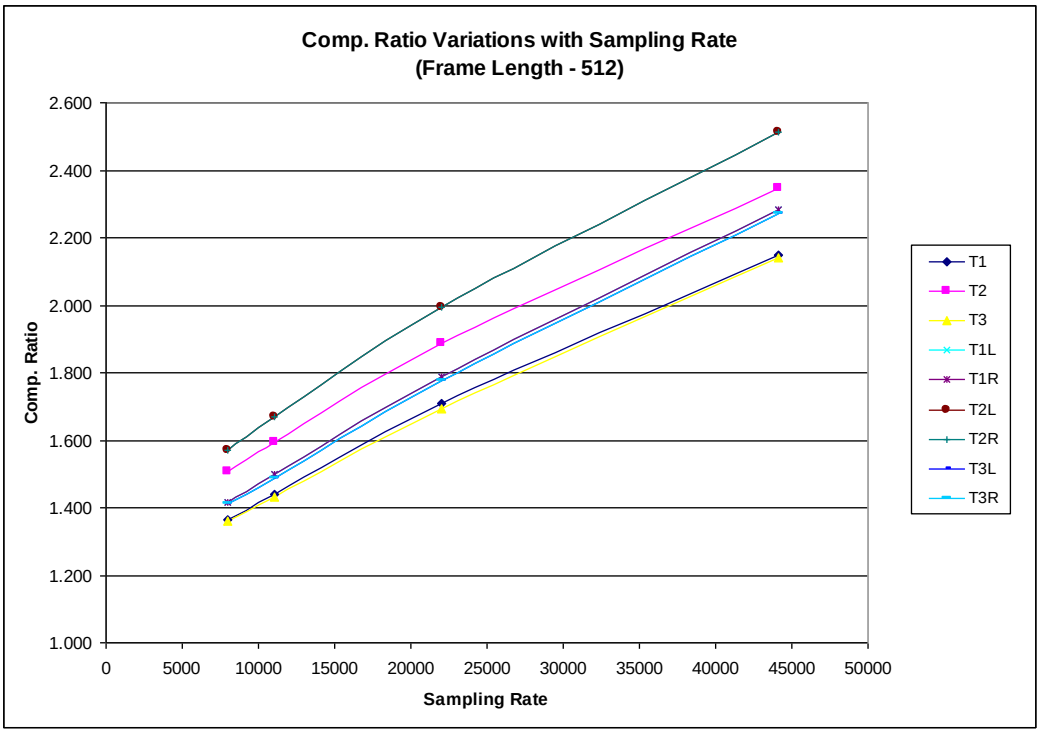

Fig. 4.18 Variation in compression ratio with sampling rate (frame length =512)

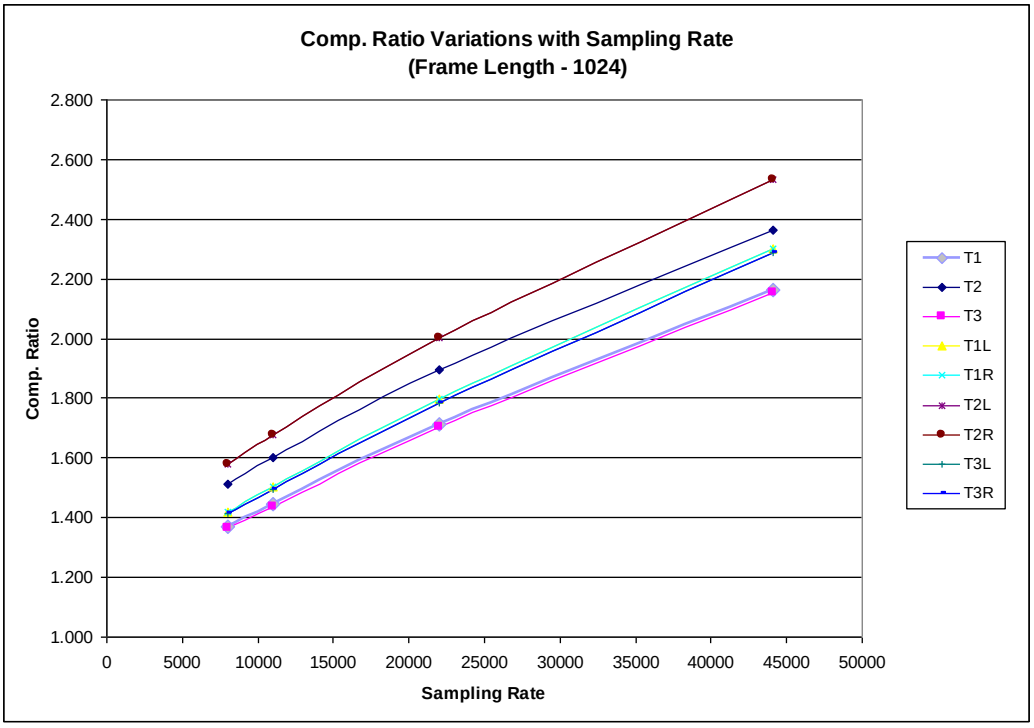

Fig. 4.19 Variation in compression ratio with sampling rate (frame length =1024)

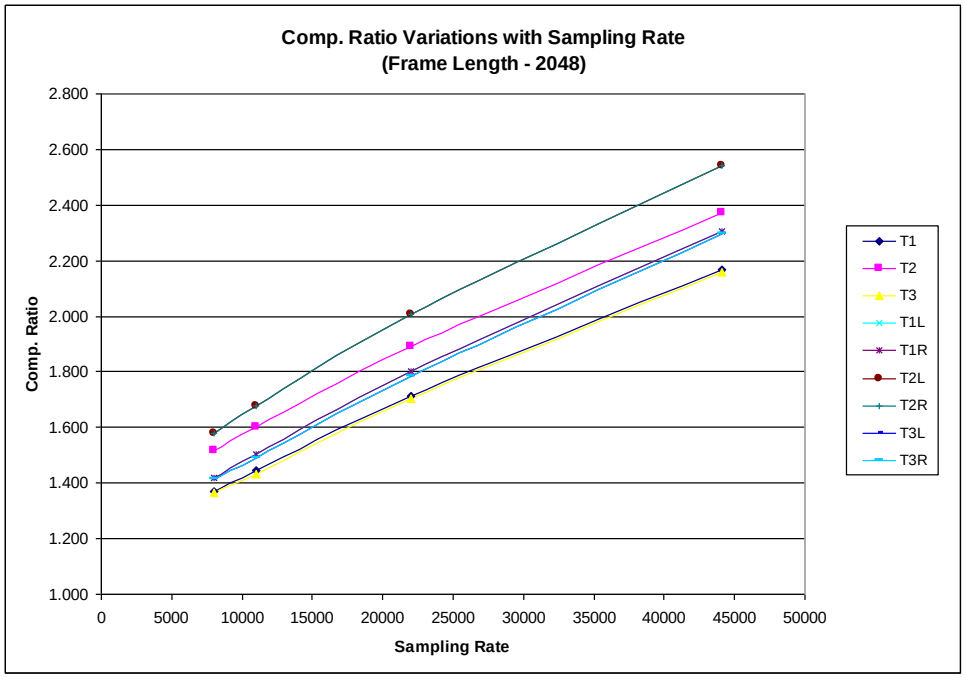

Fig. 4.20 Variation in compression ratio with sampling rate (frame length =2048)

Experimental results show that the compression ratio is proportional to the sampling rate. Downsampling of the audio tracks will yield lesser compression. In otherverse more compression is possible when the audio quality increases. Network enables music telepresence will therefore definitely benefit by compressing the audio tracks prior to transferring the data across network.

4.4.5 Sampling Rate Vs Latency

Music tracks with sample rate 8K, 11K, 22K and 44K are plotted against the compression ratio for frame lengths 110, 220, 440, 880, 1760 and 4400.

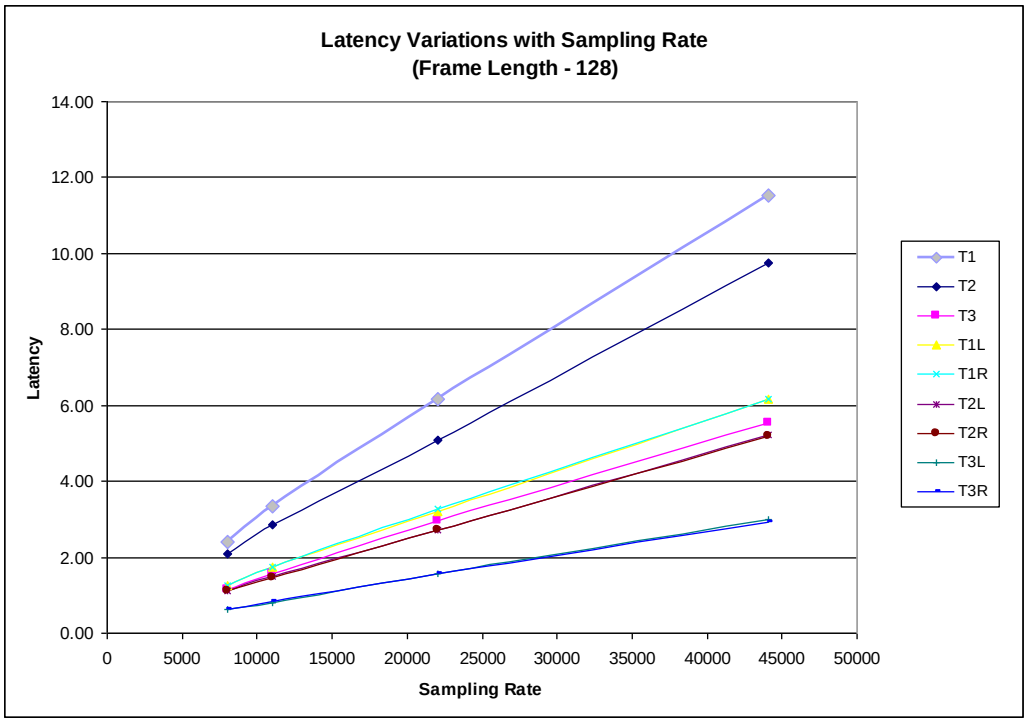

Fig. 4.21 Variation in latency ratio with sampling rate (frame length =128)

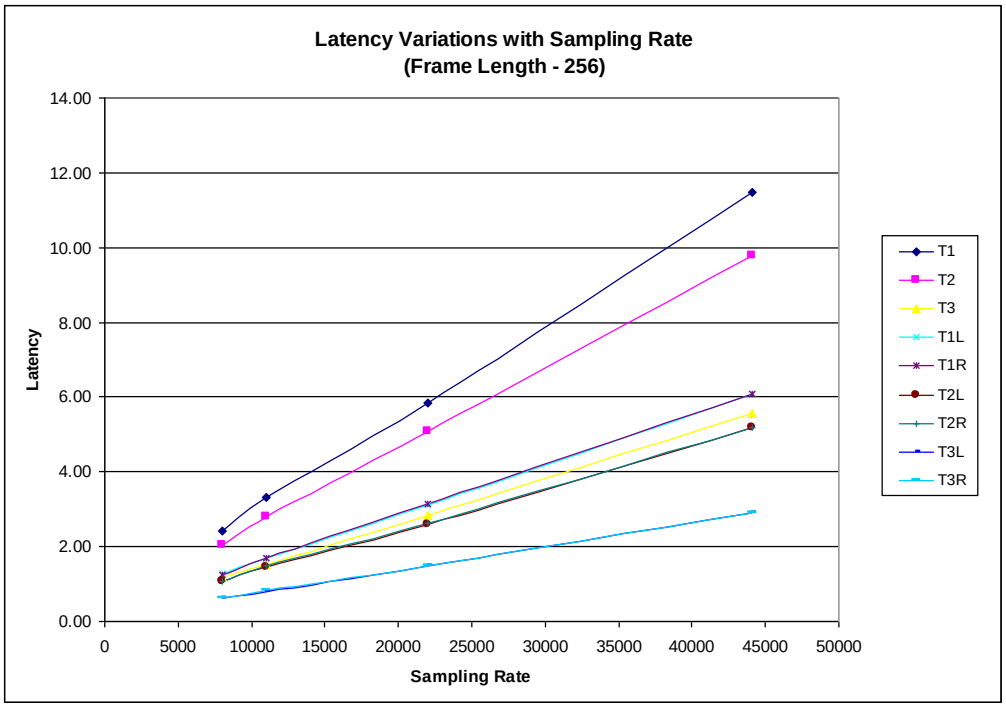

Fig. 4.22 Variation in latency ratio with sampling rate (frame length =256)

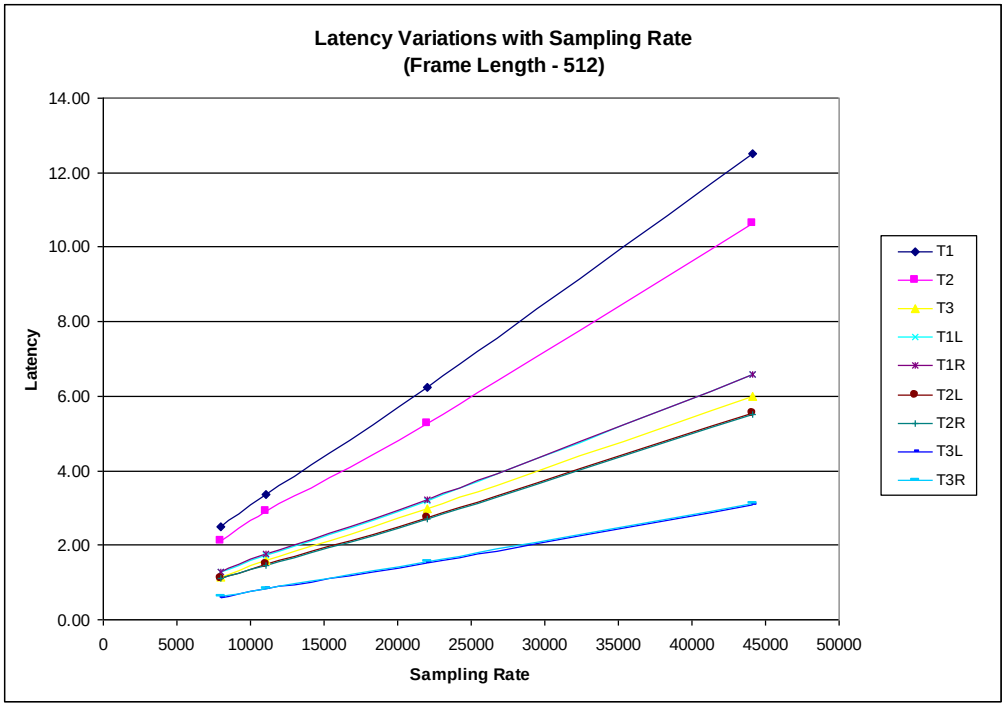

Fig. 4.23 Variation in latency ratio with sampling rate (frame length =512)

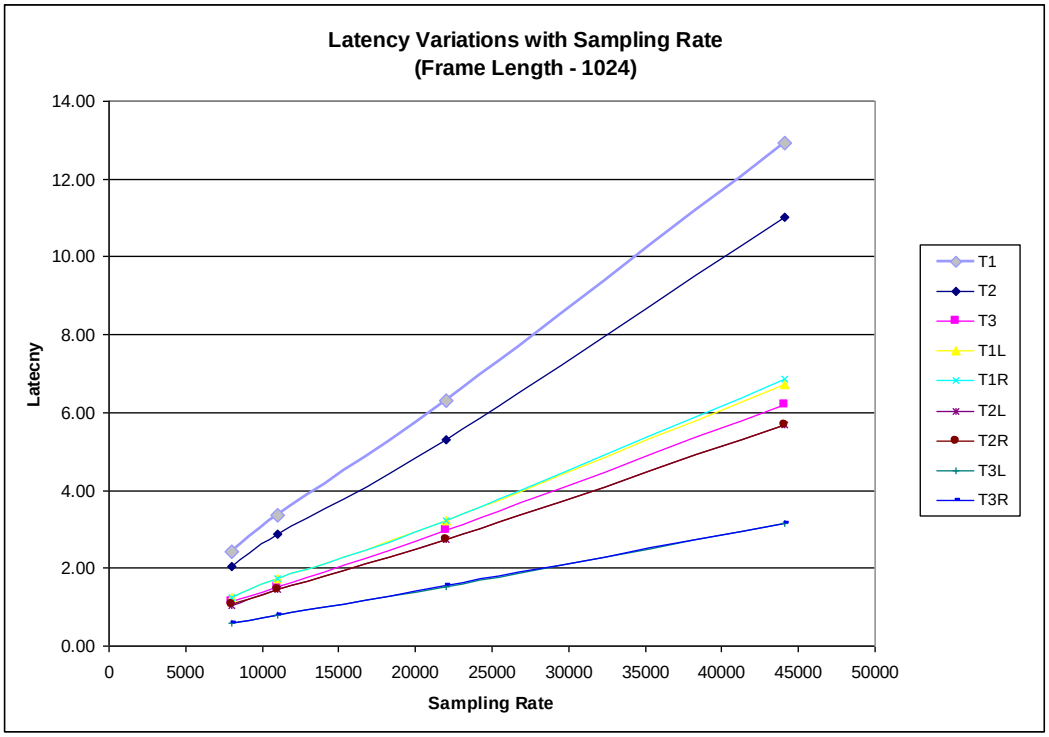

Fig. 4.24 Variation in latency ratio with sampling rate (frame length =1024)

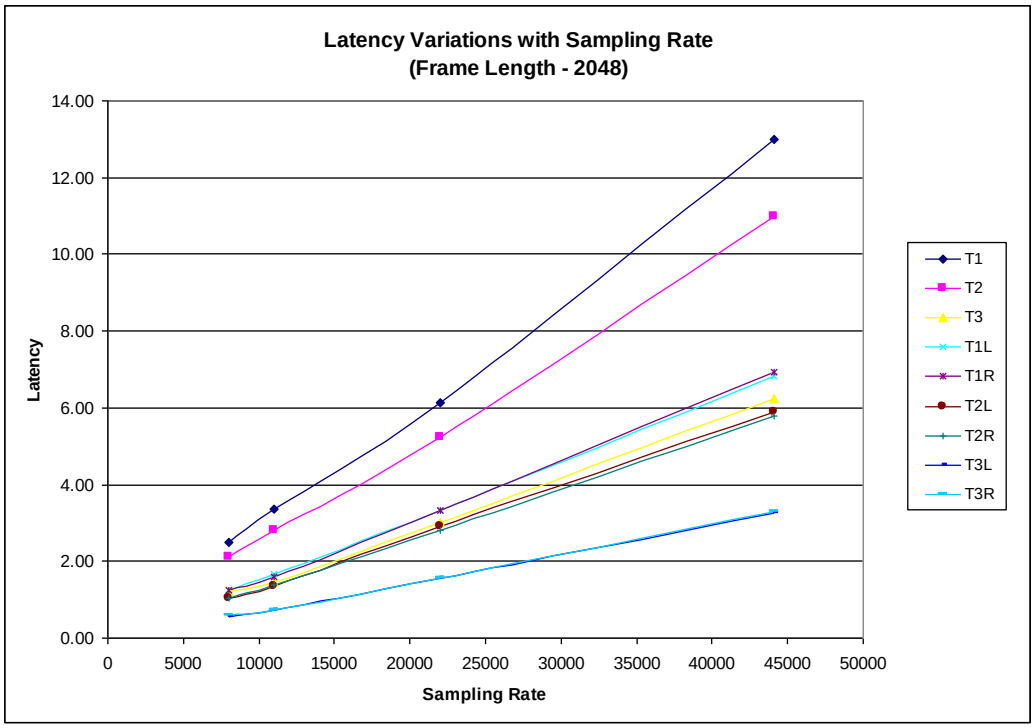

Fig. 4.25 Variation in latency ratio with sampling rate (frame length =2048)

Latency increases with sampling rate. This is due to the increase in file size associated with the sampling rate.

4.4.6 Sampling Rate Vs KB/ms saved by compression

Music tracks with sample rate 8K, 11K, 22K and 44K are plotted against the KB/ms saved by applying compression for frame lengths 110, 220, 440, 880, 1760 and 4400 are plotted.

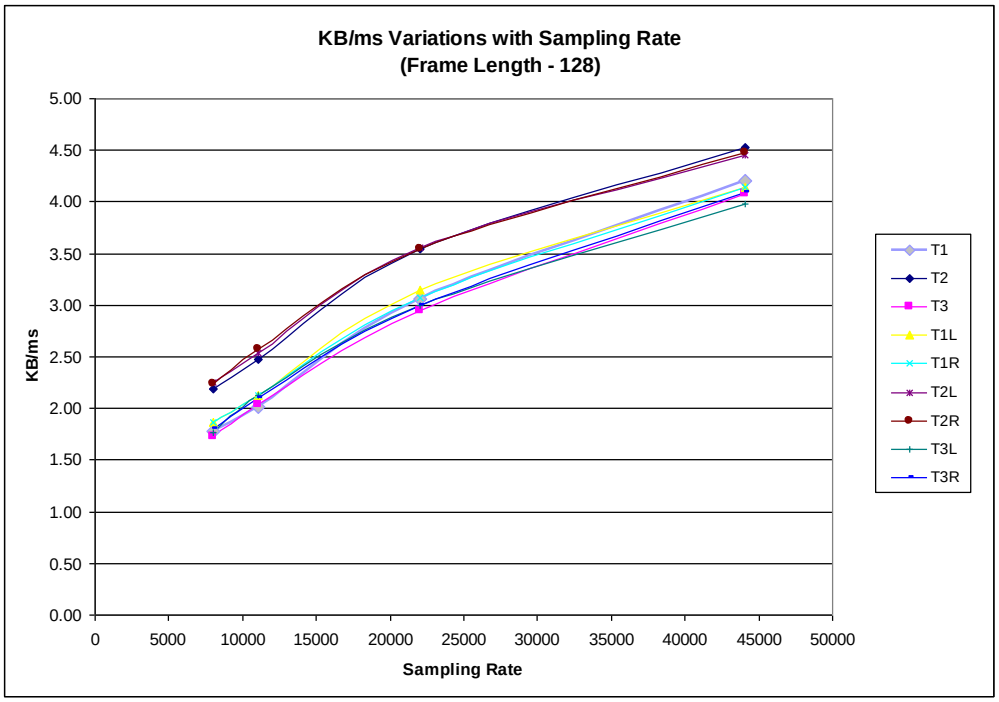

Fig. 4.26 Variation in compression size reduced with sampling rate (frame length =128)

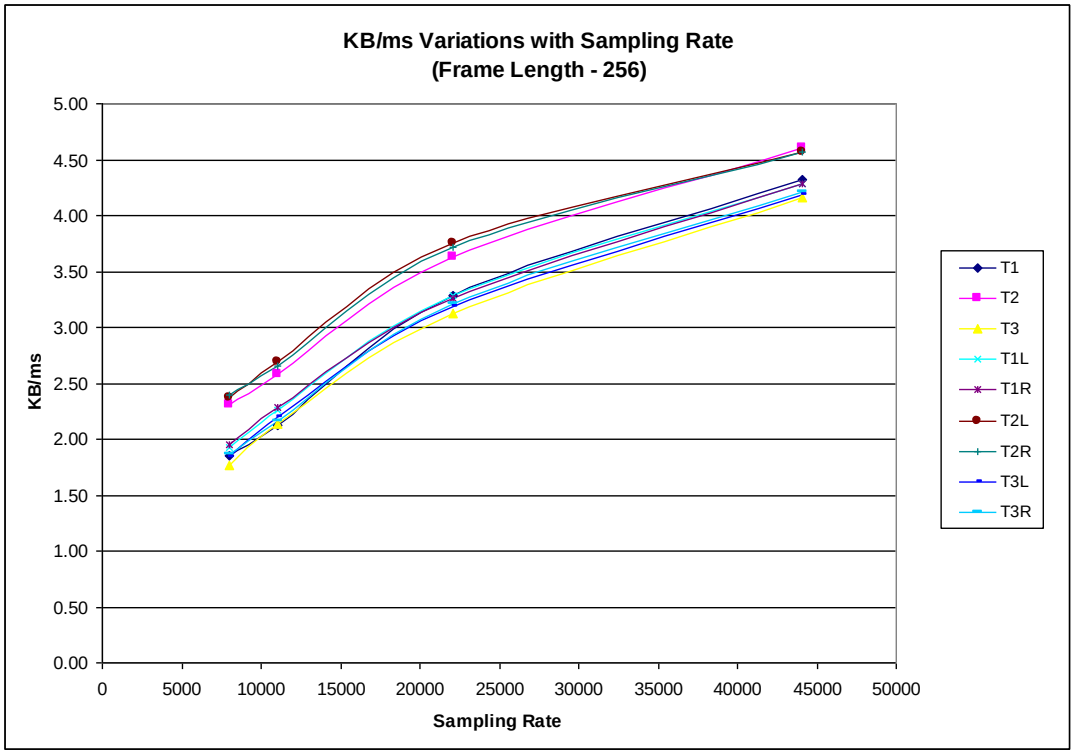

Fig. 4.27 Variation in compression size reduced with sampling rate (frame length =256)

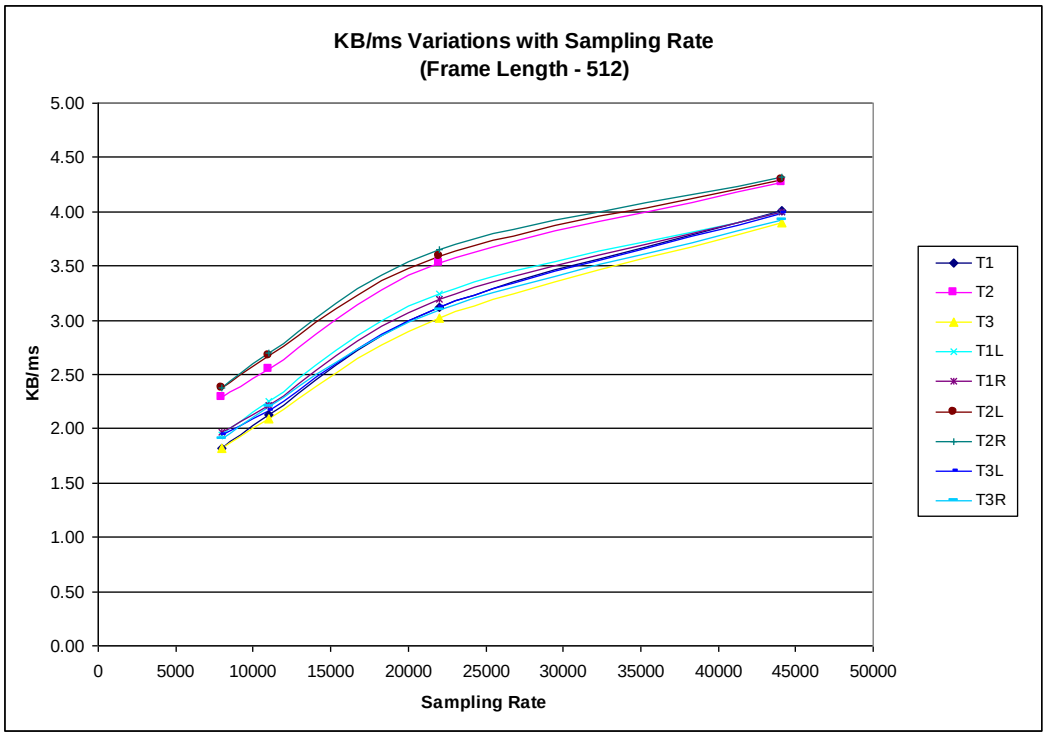

Fig. 4.28 Variation in compression size reduced with sampling rate (frame length =512)

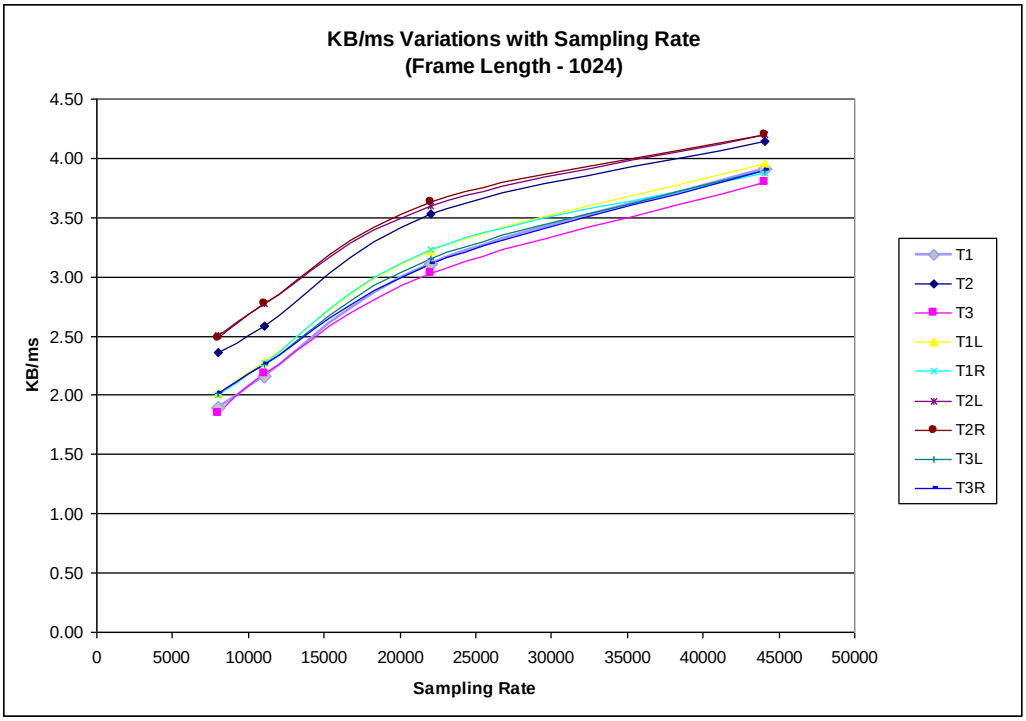

Fig. 4.29 Variation in Compression size reduced with sampling rate (frame length =1024)

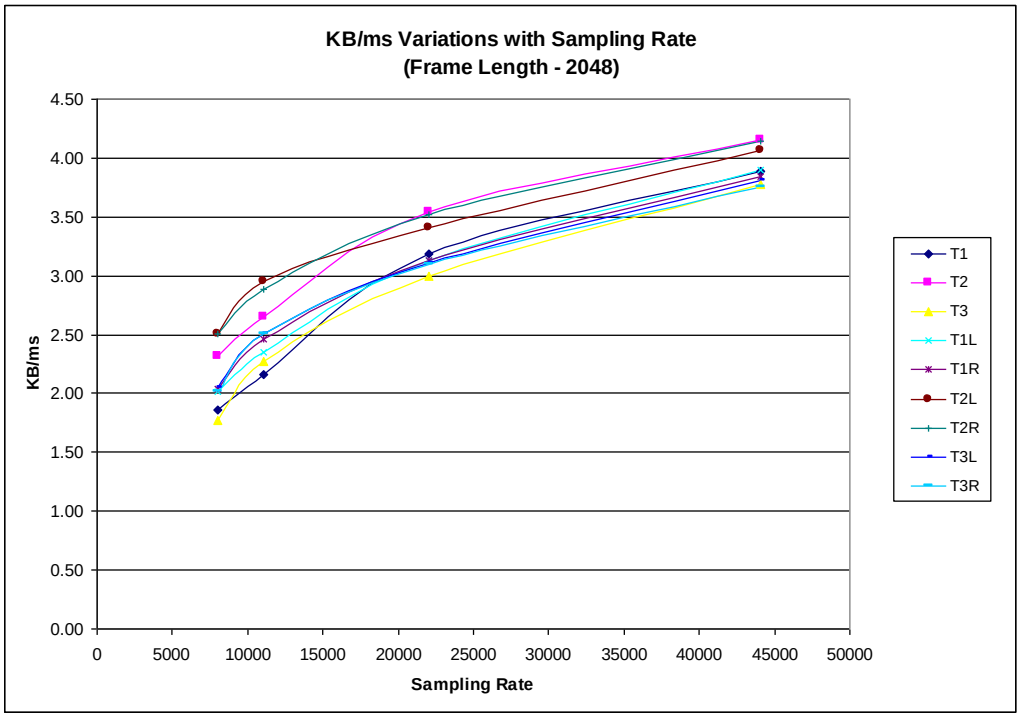

Fig. 4.30 Variation in compression size reduced with sampling rate (frame length =2048)

Even though latency increases with sampling rate, compression ratio also increases. Proportional increase in compression size reduced /ms with sampling rate indicate the advantage of using compression for high-resolution audio signals before transmission through networks.

4.5 Entropy Coding of the Residual

When using MPEG-4-ALS codec to compress audio signals, the residual values are entropy coded using Rice codes (by default). Alternatively, the encoder can use a more complex and efficient coding scheme called BGMC (Block Gilbert-Moore Codes).

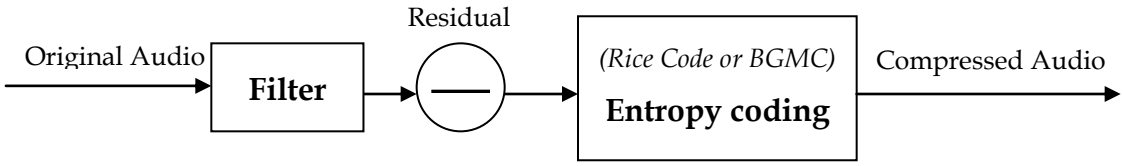

Fig 4.31 Audio Encoding Block diagram

Test was done on 12 stereo classical music tracks with the following Audio Properties:

Sample type: int
Resolution: 16 bit
Sample Rate: 44100 Hz
Channels: 2

Codec Properties (MPEG4-ALS options used)

Entropy coding: Rice Code or BGMC
Other Settings: default

Comparison of the results shows that compression done using Rice code takes less CPU time compared to using BGMC, but the use of BGMC entropy coding method yielded slightly better compression than using the Rice code.

Track#	PCM Size (MB)	Rice Code		BGMC	
		Comp. Ratio	CPU Time (sec)	Comp. Ratio	CPU Time (sec)
T1	91.74	2.17	24.75	2.20	35.72
T2	77.24	2.37	20.71	2.41	29.44
T3	42.88	2.16	11.66	2.19	16.78
T4	42.83	2.09	11.53	2.12	16.63
T5	85.19	2.11	23.05	2.14	33.14
T6	81.60	2.39	22.16	2.43	31.65
T7	48.16	2.23	13.20	2.26	18.91
T8	51.79	2.08	13.89	2.10	19.84
T9	77.04	2.04	20.97	2.06	30.59
T10	77.29	2.28	20.54	2.32	30.36
T11	54.44	2.09	14.96	2.12	21.32
T12	64.62	1.91	17.41	1.93	25.62

Table 4.18 Comparisons of BGMC and Rice Code

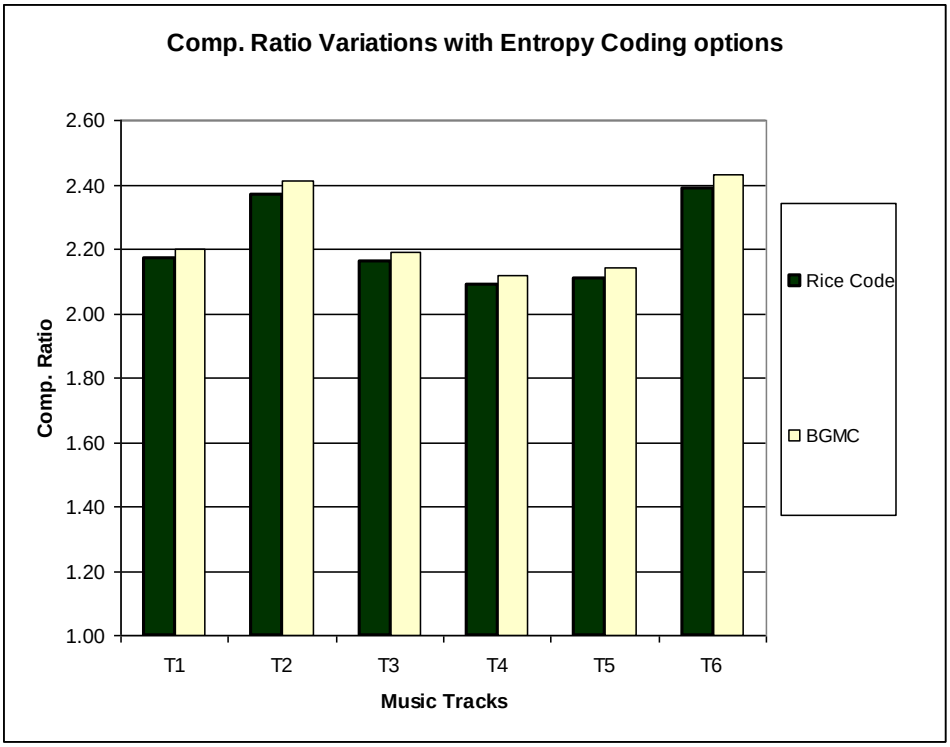

Fig 4.32 Graph showing the variations of compression ratio with respect to entropy coding options viz. Rice Coding and BGMC coding

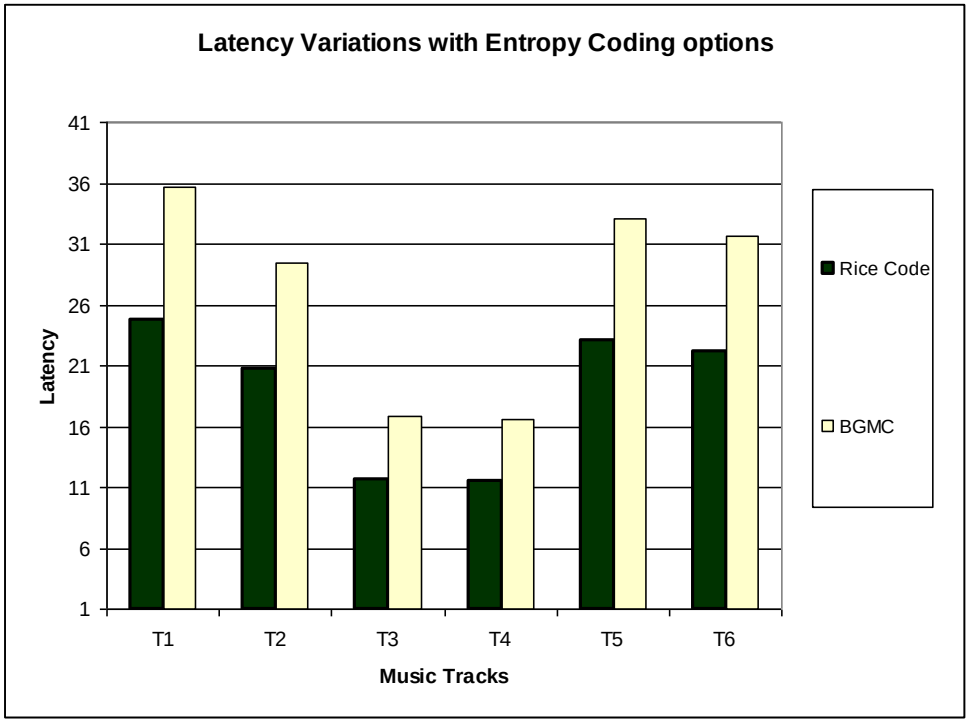

Fig 4.33 Graph showing the variations of latency with respect to entropy coding options viz. Rice coding and BGMC coding

OPTIONS	T1 KB/ms	T2 KB/ms	T3 KB/ms	T4 KB/ms	T5 KB/ms	T6 KB/ms
Rice Code	2.05	2.21	2.02	1.98	1.99	2.19
BGMC	1.43	1.57	1.42	1.39	1.40	1.55

Table 4.19 Table showing the variations of KB/ms saved with respect to entropy coding options viz. Rice coding and BGMC coding

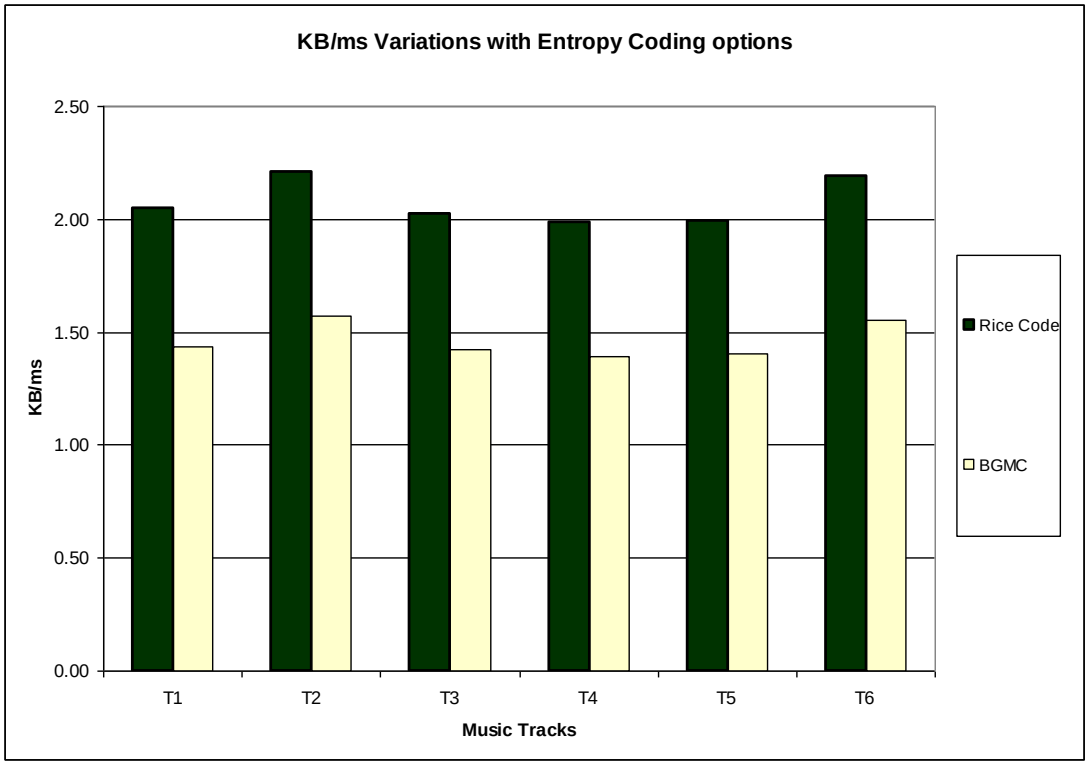

Fig 4.34 Graph showing the variations of KB/ms saved with respect to entropy coding options viz. Rice coding and BGMC coding

Codec complexity increases significantly when BGMC entropy coding option is selected over the default Rice code. For real-time interactive application like telepresence where latency is critical Rice code is preferable, as it takes less encoding, decoding time (codec latency) compared to using BGMC.

4.6 Summary and Analysis

In summary, the experimental results clearly show the advantages in terms of latency savings by using compression techniques in multimedia network applications involving audio musical signs. There is always a trade-off between codec complexity and latency. A higher coding option will be used only if it is within the required latency limit.

Compression ratio slightly increases with frame length initially. But later on compression ratio remains constant after the saturation point. Holding latency also increases with frame length but due to the compression gain, there is a significant advantage in compressing the audio signals before transmitting through the network, but with an appropriate choice of frame length.

Compression performance significantly increases with sampling rate and thus the high-resolution audio signals compress better. This implies that double the sampling rate doesn't double the resulting bit rate when compression is used.

Rice code is preferable than BGMC as entropy coding option in real-time network application. Experimental results showed significant latency increase when BGMC option is used to encode the residual.

Codec latency also depends also on the processor speed. So by increasing the processor speed we can further reduce the codec latency.

Chapter 5
Conclusion and Future Works

5.1 Conclusions

In this thesis, we have proposed techniques to finely tune MPEG-4 ALS encoder parameters to use the codec for network applications involving interactivity where latency is very critical. Audio samples can be streamed in slices depending upon the latency and packet size limitations so that maximum compression can be achieved within the latency limits. Downsampling of the audio stream is done if required, taking into consideration the audio quality requirements, along with latency, bandwidth and packet size limitations.

The complexity of the codec can be so adjusted to satisfy the latency limit. In application involving interactivity where the audio streams are repetitive, the initial audio streams can be used as a mock-up sample (as a template) to assess the optimum parameters for the encoder options so that the following audio samples can be used for interactivity. Music rehearsal is a typical scenario in which these techniques can be used to optimize the encoder parameters. The first step is to identify the requirements of the applications and to find the limits of latency (maximum allowed latency) considering the bandwidth, network delay and other related factors. Then the encoder options can be so adjusted to yield maximum compression within the stipulated latency limit.

5.2 Contributions

MPEG 4-ALS is a general purpose audio compression codec with outstanding compression performance. However, in order to effectively support music telepresence applications where latency is critical, encoding options and parameters have to be carefully and thoroughly tested and selected. This thesis work provided a co-evaluation of latency and compression performance on various classical music tracks which indicated that the use of following options and parameters can significantly improve the overall music compression performance for delay sensitive applications like telepresence:

- Independent channel coding, with significantly reduced compression complexity and coding latency, can achieve comparable compression performance than joint channel coding or multichannel coding options due to the limited cross channel correlation in classic music;

- Long term prediction option is not preferable for interactive applications as long-term correlations are not significant for classical music tracts and it involves longer coding time consumption.

- Overall performance is better when the frame length selected is small (< 10 ms) as latency is seen to be proportional to frame length.

- Out of the two entropy coding options viz. Rice code (default) and BGMC, Rice code option is preferable over BGMC for interactive applications like telepresence.

- Higher compression is observed when the sampling rate is higher as expected.

Overall analysis shows that it is preferable to choose the encoding options and parameters in such a way that the codec complexity is least since the loss in compression gain is only marginal.

5.3 Future Works

Future goals in the implementation of audio compression are quit broad. Some of the future goals include codec optimization, model-based compression, creating efficient algorithms, detailed analysis of techniques used in audio compression.

The ALS reference software is not optimized; particularly not in terms of encoder speed therefore there is room for optimizing MPEG4-ALS codec to increase the encoding speed. More research is needed to create, modify and analyze the efficiency of the algorithm and to compare other entropy codes such as arithmetic code over Rice code.

Predictive model can be created for music signal and for classical music or audio signals. Creating models for musical instruments such as piano, guitar etc. will help the compression of individual channels of musical instruments in interactive application, musical rehearsals etc.

To integrating the compression codec into music telepresence and testing compressed and uncompressed stream through the network to assess the advantages of using compression techniques in multimedia interactive applications.

REFERENCES

[1] **Introduction to Data Compression**, Khalid Sayood Morgan Kaufman Publishers, Second Edition 2003.

[2] **A Mathematical Theory of Communication**, *C. E. Shannon*

[3] **Lossless Audio Coding**, http://www.lossless-audio.com/theory.htm

[4] **FLAC - Free Lossless Audio Codec**, http://flac.sourceforge.net

[5] **WavPack Audio Compression**, http://www.wavpack.com

[6] **Monkey's Audio Compression**, http://www.monkeysaudio.com

[7] **The True Audio (TTA) codec**, http://tta.sourceforge.net, http://www.true-audio.com

[8] **Shorten** http://etree.org/shncom.html

[9] **LPAC,** http://www.nue.tu-berlin.de/wer/liebchen/lpac.html

[10] **Real Audio Lossless** http://www.realnetworks.com/products/codecs/realaudio.html

[11] **Multimedia networking**, *Bohdan O. Szuprowicz*

[12] **Computer Networks,** *Larry L. Peterson and Bruce S. Davie*

[13] **An Introduction to MPEG-4 Audio Lossless Coding,** *Tilman Liebchen, Technical University of Berlin*

[14] **MPEG-4 Audio Lossless Coding**, *Tilman Liebchen, Yuriy Reznik, Takehiro Moriya, and Dai Tracy Yang*, Convention Paper (116th Convention '04) Berlin, Germany.

[15] **MPEG-4 ALS: an emerging standard for lossless audio coding, in Data Compression**, Liebchen, T. Reznik, Y.A., Conference, 2004.

[16] **Enhancement of MPEG-4 ALS Lossless Audio Coding**, Yutaka Kamamoto [*NTT Communication Science Laboratories*], Takehiro Moriya, Noboru Harada, and Csaba Kos https://www.ntt-review.jp/archive/ntttechnical.php?contents=ntr200712sp2.html

[17] **MPEG-4ALS**, Ersin Uzun (http://www.ics.uci.edu/~euzun/pub/267.pdf)