Error control techniques within multidimensional-adaptive audio coding algorithms

a technical white paper

by Neil Smyth, APTX

www.aptx.com

ABSTRACT

Multidimensional-adaptive audio coding algorithms can adapt multiple performance measures to the demands of different audio applications in real-time. Depending on the transmission or storage environment, audio processing applications require forms of error control to maintain acceptable audio quality. By definition, multidimensional-adaptive audio coding utilizes numerous error detection, correction and concealment techniques. However, such techniques also have implications for other relevant performance measurements, such as coded bit-rate and computational complexity. This paper discusses the signal-processing tools used by a multidimensional-adaptive audio coding algorithm to achieve varying levels of error control while the fundamental structure of the algorithm is also varying. The effects and trade-offs on other coding performance measures will also be discussed.

1. Introduction

The cognitive ability of a multidimensional-adaptive audio coding algorithm provides the ability to adapt to the presence of bit and packet errors. Whilst other conventional audio coding algorithms can utilize error control tools, these schemes typically have coarse-grained control and predetermined error control characteristics that cannot be easily altered or shaped. A multidimensional-adaptive audio coding algorithm can modify the error control tools in a dynamic manner, according to external measures of channel noise and other system parameters. However, due to the multidimensional nature of the adaptation, such a codec also needs to be aware of how the choice of error control strategy affects other performance goals, such as coded bit-rate, algorithmic latency, perceptual audio quality and computational complexity.

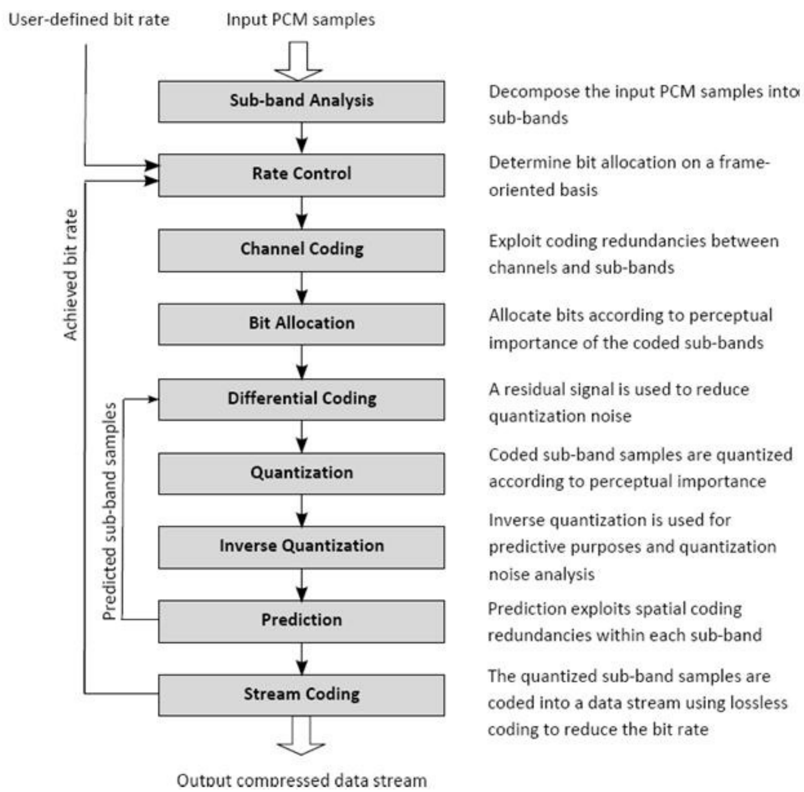
This paper describes a novel coding scheme that can adapt its audio coding functions and algorithm characteristics to achieve the optimal level of error control for a particular environment. This is achieved by providing the encoder with parameters describing the error characteristics of the transmission channel. In addition to transmission error characteristics, the multidimensional audio coding algorithm is capable of cognitively adapting to achieve performance goals such as computational complexity, algorithmic latency and bit rate.

The concept of a multidimensional audio coding algorithm is discussed in Section 2. Section 3 describes the underlying robustness of this algorithm. Section 4 provides a description of the intelligent-agent based adaptive control scheme and the performance achieved by the resulting multidimensional-adaptive audio coding algorithm. A conclusion is provided in Section 5 while further work is discussed in Section 6.

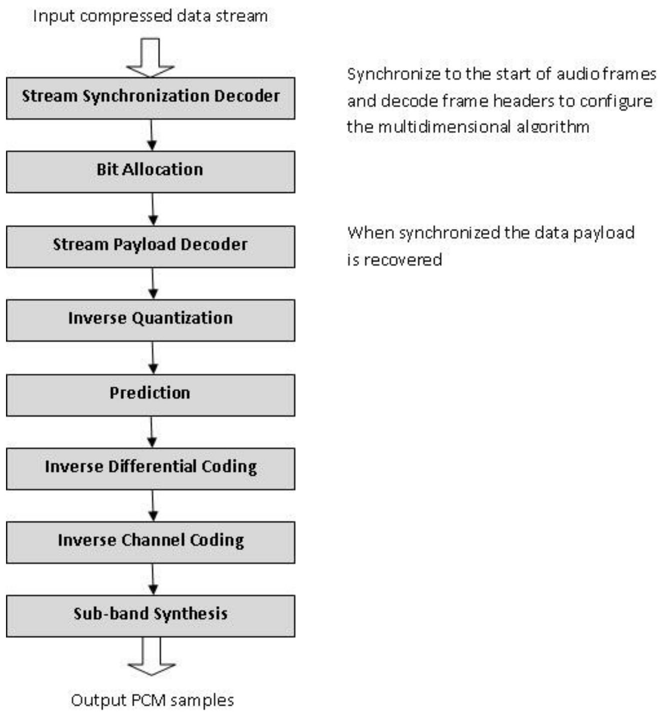
2. What is a Multidimensional-Adaptive Audio Coding Algorithm?

A multidimensional audio coding algorithm [1] is capable of functioning over a wide range of operating points, e.g. battery life, latency, bit rate and quality. This algorithm possesses a range of tools with different latency, complexity and quality attributes. A number of tools can be selected to perform the tasks of predictive coding, quantization, sub-banding, channel coding, error correction coding and entropy coding (see Figure 1). It is possible to dynamically modify the choice of coding tools at any given time, but the selected coding tools must be communicated with the decoder.

Applications will require the configuration of this algorithm to be modified over time to achieve varying performance goals. This configuration can be complex given the high number of possible tool combinations and their varying impact on the system. The algorithm will also behave differently depending upon the system and hardware platform on which it operates and the task it is performing at any given moment. This results in an algorithm that is difficult to characterize and control. An adaptive control mechanism is required to optimally select the appropriate set of coding tools at any given instant using system performance measures - thereby leading to the concept of a multidimensional-adaptive audio coding algorithm.



(a) encoder



(b) Decoder

Figure 1: A multidimensional audio coding algorithm

Table 1: Error resilient properties of basic multidimensional algorithm structures

Prediction	Quantization	Description
None	Scalar Uniform	All audio frames can be decoded in isolation, does not allow any propagation of errors. Suitable for very noisy environments
	Vector Non-Uniform	All audio frames can be decoded in isolation, adaptive vector quantization limits the propagation of errors. Suitable for moderately noisy environments with low burst error rates.
SS-LMS	Vector Non-Uniform	Predictive coding utilizing sign-sign LMS limits the propagation of errors. Suitable for low error rates with low burst error rates.
Adaptive SS-LMS	Vector Non-Uniform	Adaptive prediction is sensitive to errors as it has less opportunity to compensate for errors. Suitable for low noise transmission only.

3. Codec Robustness

A number of steps have been undertaken to improve the error robustness of the multidimensional

3.1 Error Correction

A novel feature of the proposed multidimensional audio coding algorithm is the ability to implement several error correction techniques on a frame basis. The syntax over which error detection/correction operates and the error correction properties are both variable. This provides us with a fine-grain scalable error correction scheme that can be applied in a time-varying manner to the transmission channel, such that the error correction used for any given frame and channel noise characteristics can be chosen to optimally reduce errors.

The error correction schemes provided by the multidimensional audio coding algorithm include:

- **Extended Binary Golay Code [4]** - Encodes 12 bits of data into a 24-bit code allowing three bits of error to be corrected and four bits of error to be detected. This is a fast and low complexity error correction scheme.

- **Interleaved Extended Binary Golay Code** - Interleaving of multiple Golay codes to form a larger code word. This disperses burst errors across multiple codes such that the effective error correcting capabilities are increased.
- **Reed-Solomon [3]** - A byte-oriented Reed-Solomon code with a variable redundancy. The block size can be reduced from a maximum of 255 bytes using block shortening. This is a relatively complex scheme and requires buffering of large blocks of data.

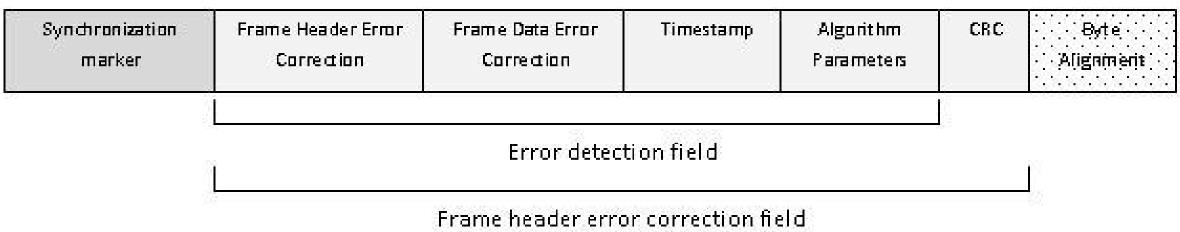
3.2 Data Payload Robustness

The use of error detection over the data field of audio frames is detrimental to audio quality, i.e. if an entire frame is discarded the resulting perceived loss in quality will be more significant than the corruption of a small number of audio samples. A number of basic structures can be employed by the multidimensional audio coder to facilitate error resilience to different degrees (see Table 2).

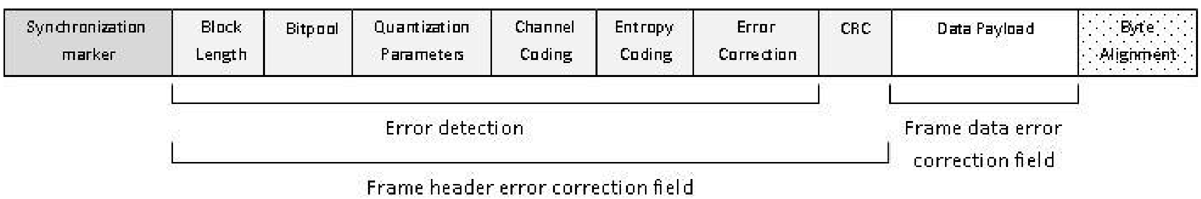
Golomb-Rice coding can be used to reduce the bit rate of the multidimensional audio coding algorithm. However, variable length codes (VLCs) are prone to error propagation, typically leading to large quality distortion and subsequent packet loss.

A derivative of EREC [2] has been developed to allow the VLCs to be packed into a fixed length structure. This limits the degree to which errors can propagate thereby reducing distortion and packet loss when using Golomb-Rice coding, but this will increase the computational effort of the algorithm. The EREC scheme requires the width and number of EREC blocks to be transmitted with each audio frame in the header.

3.3 Stream Syntax and Decoder Stability



(a) Parameter Frame



(b) Audio Frame

Figure 2: Example of frame syntax structure

Ensuring that the stream syntax is relatively insensitive to corruption is a fundamental first step to ensuring that the decoder can synchronize with the incoming data stream and decode sufficient information for audio playback. Once synchronized with the incoming data stream it is imperative that the decoder can safely process corrupted data such that the algorithm's stability is not compromised.

The multidimensional audio coding algorithm supports four basic packet types, two of which are detailed in figure 3.3. In order of perceptual importance these are: *Parameter Frames*, *Audio Frames*, *Padding Frames* and *User Data Frames*. To aid in the stream parsing process all packets begin at a byte-aligned position.

The *Parameter Frame* (PF) is used to describe the algorithm structure at a high-level. It contains definitions for all of the selected coding tools that must be applied to all subsequent audio frames. All audio frame data subsequent to a corrupted PF must be discarded as the correctness of the multidimensional decoder configuration cannot be relied upon. Therefore it is imperative that this packet is adequately protected.

Padding Frames are used to increase the bit rate under certain conditions of CBR rate control. The packet consists of an 8-bit synchronization marker, 32-bit length code, a variable number of padding bytes and a terminating byte. The padding bytes are are all equivalent to the value 255 whilst the terminating byte is the value 1. The length code and the padding data values provide a mechanism to verify the integrity of the padding frame. This ensures that a corrupted length code will not result in otherwise valid data being discarded and/or a loss of decoder stability.

User Data Frames are very similar to *Padding Frames*. However, the arbitrary nature of the user data payload does not provide a redundancy mechanism for verifying the frame length as in the case of *Padding Frames*. In order to improve stability and provide a means to determine the end of a frame (a) a maximum frame length of 256 bytes is enforced and (b) a unique 4-byte frame termination string is used.

The *Audio Frame* contains a condensed header providing an 8-bit synchronization marker, 8-bit bitpool and a 12-bit block size. All compressed audio data is transported using this packet type, therefore loss of these packets will require an audio concealment mechanism. The data payload is variable in terms of its content and length.

A number of measures are used to improve error resilience of the packetized stream structure.

- The decoder uses the first one to four bytes of all packets to determine the packet type.

These packet type markers form a byte-aligned static synchronization marker. It is important a decoder does not immediately discard a corrupted synchronization marker as this can lead to significant packet loss.

- The validity of each packet header must be verified through the use of a cyclic redundancy check (CRC). This allows errors to be detected and the packet to be discarded.
- A state machine is used to indicate the corruption status of the decoder. This allows the decoder to intelligently explore the validity of a packet header, depending upon the validity of past frame headers. For example, if the past few frame headers were successfully decoded the decoder will have a high confidence in the position of an upcoming synchronization marker and a small number of invalid bits in the synchronization marker can be ignored.
- PFs are periodically re-transmitted. Increasing the rate of transmission improves the probability of successfully decoding the PF and considerably reduces packet loss of audio frames.
- Each PF contains a timestamp indicating the PCM sample index of the left/mono channel. The decoder maintains an independent timestamp for every successfully decoded audio frame. In the case of audio frame loss the decoder can use the timestamps to re-synchronize the stream. For this purpose silence is used as an error concealment strategy.

4. Adaptive Control Scheme

A user wishing to utilize a multidimensional audio coding algorithm must determine the optimal configuration of that algorithm given a wide range of coding tools and operating environments. This can be a significant challenge, particularly in a system where complex external factors affect the performance of the audio compression system. An example of external environmental changes includes:

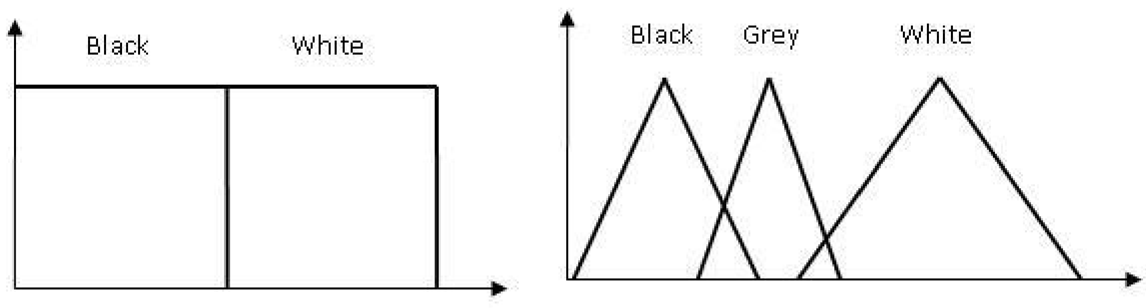
- A microprocessor in an embedded device running other tasks can experience processor, cache and memory performance variations over time that effect the efficiency of coding tools.
- The multidimensional audio coding algorithm can operate on different processor architectures, resulting in varying performance of coding tools based on hardware capabilities.
- A transmission channel can periodically be subjected to noise due to an adverse environment.
- The system enters a low power state to prolong the battery life.

We consider the system in which the multidimensional audio coding algorithm operates to be a black box. This circumvents the need to obtain accurate models of such complex systems through mathematical modeling. This requires us to utilize a learning algorithm that can adapt to an unknown environment. Such an adaptive and cognitive audio compressional algorithm can be implemented within any system or processor architecture and will not require tuning to achieve optimal performance. This leads to additional benefits in reducing engineering time when implementing the multidimensional-adaptive audio coding algorithm.

The management and control of the multidimensional audio coding algorithm must accept a range of widely varying performance goals within a system that is unknown. This adaptation is achieved using the concept of intelligent agents. These entities recognize the performance goals that a user requires and understand that they can perform a number of actions to achieve those goals. Each agent observes the environment that it operates within and the effect of actions that it exerts on that environment. The intelligent agent acts as an autonomous entity that continually adapts to the varying environment and goals.

4.1 Fuzzy Logic

Fuzzy logic [5] is a multi-valued logic utilized in soft computing to represent variables that contain a range of logic states. This differs from binary logic that represents only *true* or *false* states, as shown in Figure 3, where fuzzy logic allows us to represent concepts as *partially true*. This abstraction is similar to human logic and provides designers with a simple and effective means of mechanizing a task that a human operator can perform.



(a) Binary Logic (b) Fuzzy Logic

Figure 3: Fuzzy and binary logic comparison

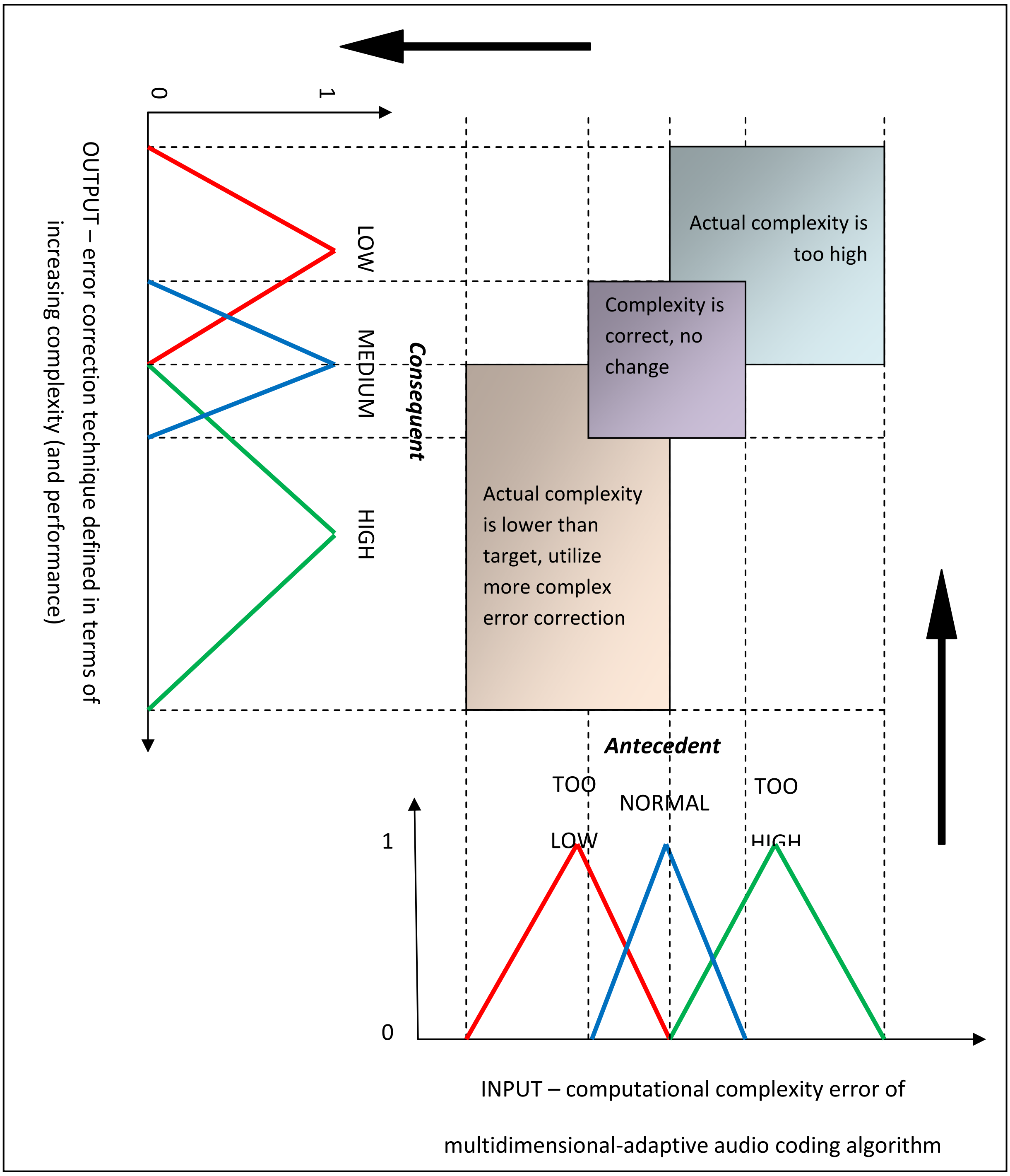
Rather than attempting to model a system mathematically, fuzzy logic implements a rule-based approach of the form *IF X AND Y THEN Z*. Such rules rely upon experience rather than technical understanding of a system to determine actions that must be taken. These rules are also imprecise but highly descriptive, providing a good approximation of human control logic.

The fuzzy logic outcome is controlled through manipulation of the shape of each fuzzy logic rule (drawn as simplified triangles in Figure 3). The parameters that can be manipulated include the height, width, centre position and gradient of each membership rule.

An example of a fuzzy logic controller is shown in Figure 4 where the input measurement of computational complexity error from the system is used to drive three fuzzy rules, represented by the three antecedent triangular membership functions. These three rules are used to describe the computational complexity of the audio coding algorithm as being *TOO LOW*, *NORMAL* or *TOO HIGH*. The fuzzy antecedent outputs for each possible output state are determined from the scaled sum of the membership functions for any given input.

The fuzzy consequent membership functions are used to combine the fuzzy antecedent state conclusions into a single conclusion. This process can be performed by the *fuzzy centroid algorithm* which can determine the centroid position of the combined area of fuzzy membership functions. Once a single conclusion has been reached the output value must undergo defuzzification to obtain a *crisp* variable. This variable forms the output of the fuzzy logic controller that is used to control the system. In our example the crisp output defines the use of one of three possible error correction coding schemes.

Figure 4: Example of a three rule fuzzy logic controller used to select the appropriate error correction tool based upon the complexity of the multidimensional-adaptive audio coding algorithm



4.2 Q-Learning

Q-Learning [6, 7] is a reinforcement learning technique [8]. The Q-learning agent operates by taking a given action in a given state. The states are learned as the algorithm operates through determination of the optimal solution to an action-value function. An advantage of Q-learning is its ability to take actions without knowledge of the system it is controlling.

A Q-learning system divides the range of states that the controller can take into a finite set. The size of this state-space is determined by the number of input variables provided by the system and the number of quantized levels for each variable. Each state can perform a number of actions, where the action taken at a particular instant allows the system state to be modified. A state-action Q value for each state and action is used to maintain a reward value. The goal of the Q-learning algorithm is to maximize its reward by learning which action is optimal for each state.

The optimal solution to the action-value function is found using the State-Action-Reward-State-Action (SARSA) algorithm. SARSA will update the state-action Q value using an error signal that is modified according to a learning rate.

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha[r_{t+1} + \beta Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{1}$$

The reward of the action that has been taken is represented by $r_(t + 1)$. It is this reward that modifies the Q state-actions to effect a learning process, whereby the action taken is determined by the state-action with the highest value. The learning rate is determined by α. The discount factor $0 < \beta < 1$ determines the impact of future state-actions that will be taken, as the discount factor tends toward 1 the learning algorithm will become more opportunistic. In many applications the discount factor decays over time to ensure steady-state operation.

4.3 Error Resilience Fuzzy Agent

When combined, a fuzzy logic controller and a Q-learning algorithm can be used to provide an intelligent agent. As shown in Figure 5 this fuzzy agent can be used to monitor an unknown system and to determine which actions should be taken to achieve the required goals.

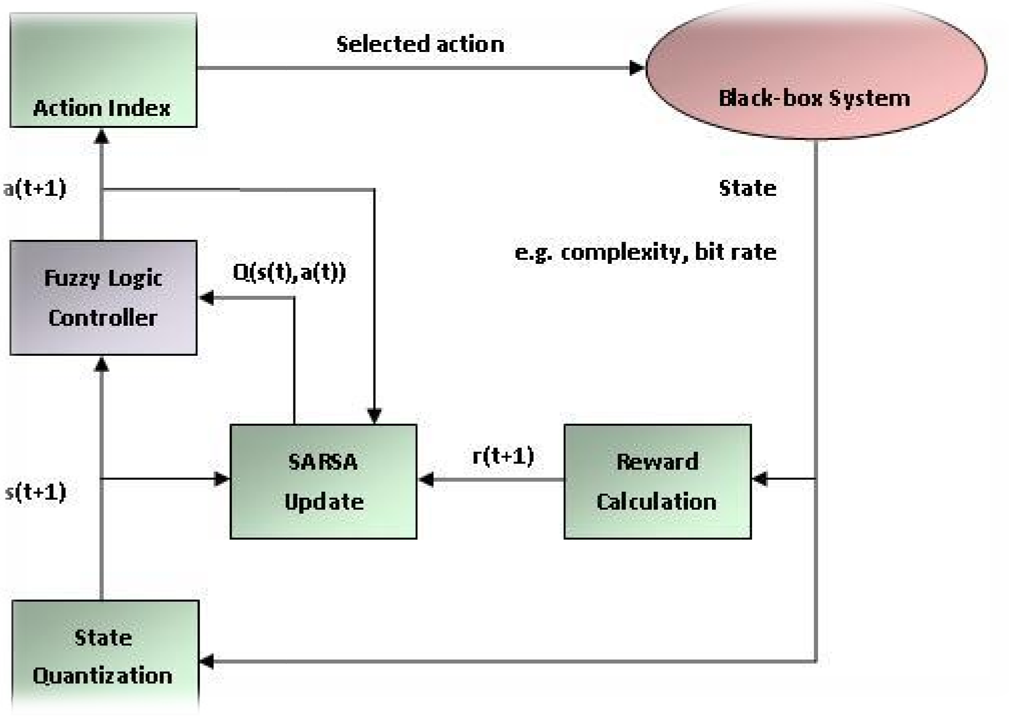


Figure 5: Fuzzy agent block diagram

Reward calculation utilizes a knowledge of the current state of the system to describe the reaction of the fuzzy agent. This reaction is based upon the goals that have been set and an understanding of what are deemed to be system failure conditions. The reward variable used to describe the fuzzy agent's reaction to the system state is referred to as $r(t+1)$.

The continuous state parameters of the system (computational complexity, computational latency, BER and bit burst error rate) are uniformly quantized to form an index into the finite state-space of the system. This index is used to form the next state of the fuzzy agent, $s(t+1)$. The state-space is maintained by the fuzzy agent, as are the list of Q state-actions.

The last state of the fuzzy agent $s(t)$ and the corresponding state-action values $Q(s(t), a(t))$ are used to determine the appropriate action to take. These state-action Q values are directly used to construct the consequent fuzzy membership functions. This allows us to reward a beneficial outcome such that the associated action is more likely to occur in the future. If the system behaves differently in future then the fuzzy consequent logic will adapt and a more appropriate action will be determined after an initial learning period.

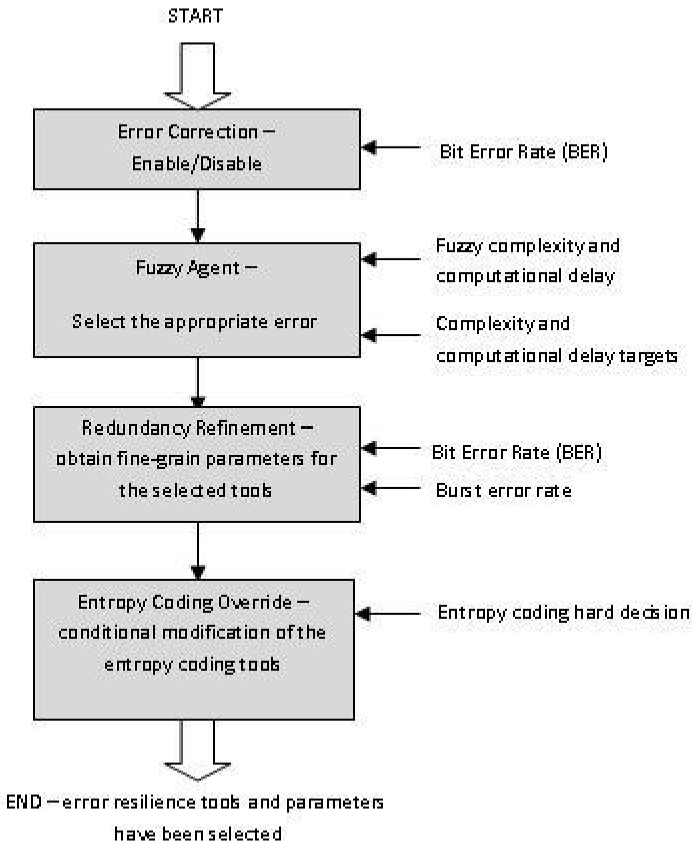


Figure 6: The control process used to achieve error resilience in a multidimensional-adaptive audio coding algorithm

We use fuzzy agents to control the complexity, computational latency, algorithmic latency and error resilience of our multidimensional-adaptive audio coding algorithm. The fuzzy agents are utilized in a sequential fashion, those agents that make critical decisions are applied last. The final fuzzy agent in our system is that which controls the error resilience, the process of which is described in Figure 6.

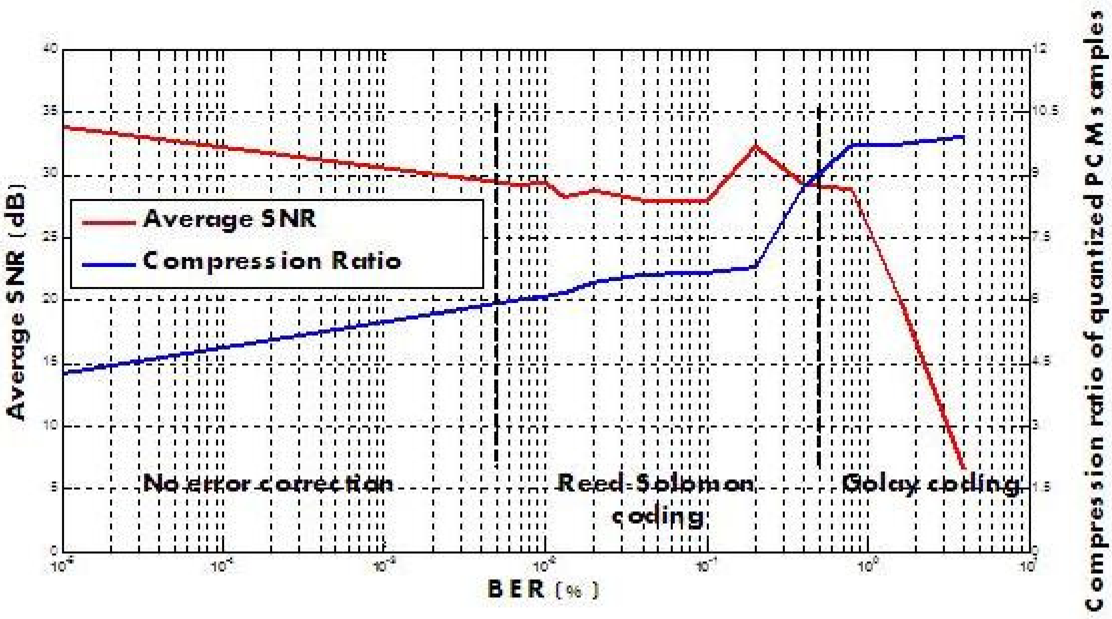


Figure 7: Average SNR and average compression ratio achieved with varying BER

The error resilience fuzzy agent is provided with input measures of the complexity error, computational latency error, bit error rate (BER) and maximum length of bit burst errors. This agent also has access to decisions taken by preceding fuzzy agents in regards to actions that will impact on the performance of error resilience. For example, decisions to utilize Golomb-Rice VLC codes can have a detrimental effect on error resilience and audio quality if the transmission channel suffers from noise.

The results achieved using this adaptive control scheme are shown in Figure 7. In this graph we have plotted the average signal-to-noise ratio (SNR) achieved by the multidimensional-adaptive audio coding algorithm when subjected to an increasing BER. The SNR is contrasted with the compression ratio of the quantized PCM samples. This graph indicates a graduated decrease in SNR and increase in redundancy whilst the BER is increased and a constant bit rate is maintained. The graph also indicates the positions where the control algorithm tends to utilize the Reed-Solomon and Golay coding schemes. The SNR rapidly deteriorates as the BER reaches approximately 4 %, a point corresponding to the error correction limits of Golay coding and a subsequent rapid increase in packet loss.

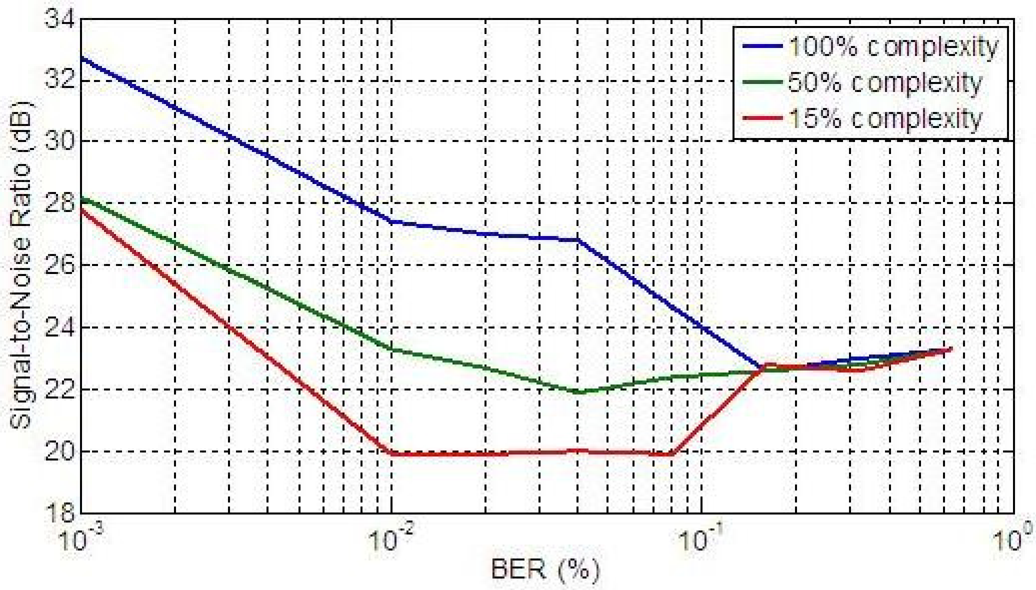


Figure 8: Performance of adaptation and learning when error resilience and computational complexity are controlled

The ability of the algorithm to adapt to varying computational complexity and error rates is shown in Figure 8. This graph describes the performance achieved when the algorithm must adapt and learn when the complexity must be controlled in conjunction with the error resilience capabilities. The error resilience of the lower complexity algorithms suffer to a greater degree at low error rates as more efficient quantization, entropy coding and APCM coding must be adopted to achieve the complexity performance goals whilst accommodating the additional processing load of error correction. When very high error rates are present the controller utilizes the simplest error correction and coding schemes in order to vastly reduce packet loss. This helps improve the average SNR quality, but significant quantization noise is present.

5.0 Conclusion

We have shown that a multidimensional-adaptive audio coding algorithm provides the ability to cognitively adapt to the presence of bit and packet errors. Whilst other conventional audio coding algorithms can utilize error control tools, these schemes typically have coarse-grained control and predetermined error control characteristics that cannot be easily altered or shaped. The proposed multidimensional-adaptive audio coding algorithm can modify the error control tools in a dynamic manner, according to external measures of channel noise and other system parameters.

We have also described how a fuzzy logic controller can be modified to use reinforcement learning to create an intelligent control system. This intelligent agent forms the controller within a multidimensional-adaptive audio coding algorithm. With no knowledge of the system into which it is placed this audio compression algorithm is capable of adapting its structure to achieve a high level of error resilience, whilst maintaining other performance goals such as computational complexity.

6.0 Further Work

The multidimensional audio coding algorithm will be further extended and enhanced as part of a continuing industrial research program. Areas of interest for coding tools include error-resilient quantization and coding techniques. As additional coding tools are researched the adaptive learning mechanism will also continue to be improved. This adaptive learning technique requires us to find a solution to a global optimization problem, i.e. finding a good approximation to the global minimum of a given function in a large search space. Other techniques that could be used here include evolutionary algorithms such as genetic algorithms and tabu search.

References

[1] APTX, http://www.aptx.com/, Accessed March 2010.

[2] D. Redmill and N. Kingsbury, "The EREC: An Error-Resilient Technique for Coding Variable-Length Blocks of Data", IEEE Transactions on Image Processing, Vol. 5, No. 4, pp. 565--574, April 1996.

[3] I. S. Reed and G. Solomon, "Polynomial Codes Over Certain Finite Fields", SIAM Journal of Applied Mathematics, Vol. 8, pp. 300--304, 1960.

[4] M. J. E. Golay, "Notes on Digital Coding", Proc. IRE, 37, p. 657, 1949.

[5] L. Zadeh, "Fuzzy Sets", Information and Control, Vol. 8, No. 3, pp. 338--353, 1965.

[6] C. Watkins, "Learning from Delayed Rewards", PhD Thesis, Cambridge, 1989. http://www.cs.rhul.ac.uk/ chrisw/thesis.html, Accessed March 2010.

[7] J. A. M. Hernández, Reinforcement Learning Matlab examples, http://www.dia.fi.upm.es/jamartin/download, Accessed March 2010.

[8] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement Learning: A Survey", Vol. 4, pp. 237--285, 1996.

