

Audio CODECs using the interpolation "LSM" and comparison with PCM

Abstract: In this paper we take advantage from the known interpolation least squares method "LSM" in order to deduce audio CODEC and algorithms for audio compression. The core concept is to replace or improve the CODECs depend on the known quantization process or direct operations on samples values. The resultant audio CODEC algorithms can replace efficiently the quantization part used in a lot of known audio CODECs. Twenty audio files have been experimented to show the progress in this work. We made a comparison with PCM used in telephony system to show clearly the efficiency of using our algorithms instead of the quantization part.

1. Introduction

Audio processing and coding is a very wide area of interest. A lot of researchers are trying continuously to improve existing audio CODECs, where others are trying to create new ones. Improving the quality besides increasing the compression efficiency always is a target. A lot of attempts have been made for achieving this target using many different algorithms. Through decades, the mathematical concepts like the interpolation or the approximation through polynomials have played a substantial role in audio, image and even video coding/encoding. Lately, some authors used these methodologies in their contributions. Yung-Gi Wu, Chia-Hao Wu in [16] tried Lagrange interpolation method in image coding and processing, Shugang Wei in [12] presented signal level compressor using Newton's interpolation formula, McPherson, T, Jr. in [11] presented compression in PCM using interpolation. As well as, Bhaskar, U in [1] who worked with interpolation in his processes. The authors, G. CARBALLO, R ALVAREZ-NODARSE and J. S. DEHESA [5] have efficiently used Chebychev Polynomials in a Speech Recognition Model. We have cited number of papers, which studied and surveyed recent other CODECs to be sure about our originality.

In this paper, we will propose three approaches (algorithms) for audio encoding/decoding (CODEC). We used the known interpolation method "Least Squares Method" introduced in [12] and a lot of numerical texts as a basic for these approaches.. Through our paper, we will investigate the ability of using the interpolation "LSM" for achieving audio CODEC by various ways, and we have accomplished progress in our results which opens promising perspectives for using this idea in the processing world, not only on audio but also on images and even videos, and now we are working on several researches depend on this idea. The algorithms we approached could be applied with/without some of the other known audio CODECs, as we tried them efficiently on MP3 audios. Efficient compression ratio besides high quality audio after recovery are tested and experimented through 20 audio files that present human voices with/without quiet music and songs with different types of music. We made a specific comparison with PCM used in telephony system because our work supposed to replace the quantization part in any CODEC fully depends on quantization or uses it as part of its processing, We have shown progress and advances. We concluded three CODECs which can be applied in storing/transmitting audio data.

2. Fundamental basics and assumptions

We have to show clearly the mathematical basic idea we used, besides assumptions we have considered to seek successful attempts.

2.1. Modeling the values of audio samples

The samples' values, which define and form the behavior and the pattern of the audio, basically are data, which is defined by specific value per sample. We will propose models for these data values using the interpolation "Least Square Method" known as "LSM". After some experiments with

polynomials, exponential and power models we selected the polynomial models to represent the audio data because LSM with polynomial models have shown best fitting for samples' values.

$$y_i \cong a_0 + a_1 x_i + a_2 x_i^2 \dots a_n x_i^n \quad (1)$$

Where y is a vector (row of data) that contains audio's sample values, x is a vector that contains the values which represents it, and will be considered as the number of samples ordered 1, 2, 3, ..., where n is the polynomial degree and i is the index ranged from 1- N_s . We usually know the values of x and y , then, by considering polynomials models, we will have to find $a_0, a_1, a_2 \dots a_n$ by solving system of linear equations, which can be represented in matrix form

$$Y = XA \quad (2)$$

$$A = \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix}_{(n+1) \times 1}, Y = \begin{bmatrix} \sum_{i=1}^{N_s} y_i \\ \sum_{i=1}^{N_s} x_i y_i \\ \sum_{i=1}^{N_s} x_i^2 y_i \\ \vdots \\ \sum_{i=1}^{N_s} x_i^n y_i \end{bmatrix}_{(n+1) \times 1}$$

$$\& X = \begin{bmatrix} N_s & \sum_{i=1}^{N_s} x_i & \sum_{i=1}^{N_s} x_i^2 & \dots & \sum_{i=1}^{N_s} x_i^n \\ \sum_{i=1}^{N_s} x_i & \sum_{i=1}^{N_s} x_i^2 & \sum_{i=1}^{N_s} x_i^3 & \dots & \sum_{i=1}^{N_s} x_i^{n+1} \\ \sum_{i=1}^{N_s} x_i^2 & \sum_{i=1}^{N_s} x_i^3 & \sum_{i=1}^{N_s} x_i^4 & \dots & \sum_{i=1}^{N_s} x_i^{n+2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \sum_{i=1}^{N_s} x_i^n & \sum_{i=1}^{N_s} x_i^{n+1} & \sum_{i=1}^{N_s} x_i^{n+2} & \dots & \sum_{i=1}^{N_s} x_i^{2n} \end{bmatrix}_{(n+1) \times (n+1)} \quad (3)$$

Then

$$A = X^{-1}Y \quad (4)$$

Therefore, we deduce the coefficients of the polynomials using Eq.(4). In order to judge the modeling, we have to determine the value r (coefficient of determination) which clarify the fitting of the proposed polynomial corresponding to the real data values (audio samples values).

$$r = \sqrt{\frac{\sum_{i=1}^{N_s} (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{N_s} (y_i - \bar{y})^2}} \quad (5)$$

Where $\hat{y}_i$ is the theoretical y -value corresponding to x_i (calculated through the polynomial) and $\bar{y}$ is the mean value of all experimental y -values. Whenever r % tend to 100, then we have perfect fitting with no error at all. This value will help us to judge the proposed polynomial whether it presents the audio data efficiently or not.

2.2. Basic assumptions

Our method is concentrated on reducing the length of the vector y , which contains the audio's samples values. In order to do that, we will apply LSM with several aspects as a basic for different algorithms. As we store/transmit the values of the polynomial coefficients instead of the vector of the audio's sample values. All of these algorithms are common in specific assumptions in which we built our work on them.

1. During the encoding, the algorithms are applied on set of N_s samples repeatedly. The number of samples N_s each time can be the same (algorithms 1 and 3) or changing every time (algorithm 2). So, the algorithms are applied until all the samples are processed.
2. The compression ratio "CMR" will be calculated

$$CMR = \text{length}(\text{vector of coefficients}) / \text{length}(\text{vector of audio's samples})$$

3. The values of the coefficients must be in the numeric range of the audio's sample values. For example, if we scaled the samples' values [-1, 1]. Then, the coefficients must be in this range. The differences between them could be represented by known number of levels like those in the audio's values themselves. We assume this constraint to ensure the ability of using our algorithms for digital transmission.
4. The method could be applied with/without some of the other CODEC techniques because our method depends on the samples' values. So, it could be applied directly after sampling process of voice or after coding the audio values with other techniques, to reduce the audio storage vector.
5. Durations of silence are one of the known voice characteristics. Considering this, would make some of our algorithms (algorithm 2) more efficient by taking unfixed N_s each time according to amount of error tolerance, will be called "diff".
6. To ensure compression ratio, we have to restrict our choice for N_s to follow the inequality $N_s \geq n + 1$ in each operation, where n is the polynomial degree.
7. Mostly, no difference between using different n with the same CMR in the algorithms; we will work with $n = 4$ along the research. However, other assumption for n will not damage the results or even effect negatively the quality. We will recommend that choosing of n will follow the inequality $2 \leq n \leq 10$ to ensure better adaptive model and to be far from oscillating degrees of n .

3. The proposed algorithms and experiments results

We experimented a lot of files and reach for three efficient algorithms, each one of them can be considered as CODEC. Twenty audio files were tested and experimented, about 11 files for human voices and 9 music files are listed in Table.1. Also r% Eq.(5) and SNR for the tested files were measured and introduced in figures 1 and some mp3 files in figure 2.

File Index	Description	Duration (sec)	n-bits/sample	F_s (Hz)
M.1	Loud Music	5	16	44100
M.2	Quiet Music	5	16	44100
M.3	Quiet Music + Arabic male voice	6	16	44100
M.4	Old Music + Arabic male voice	5	16	44100
M.5	Old Music + English male voice (Dean Martin)	5	16	44100
M.6	Music + English female voice (Celine Dion)	5	16	44100
M.7	Loud Music + English female voice (Shakira)	5	16	44100
M.8	Old Music + English male voice (Louis Armstrong)	5	16	44100
M.9	Audience cheering + indistinct music	5	16	44100
H.1	Male voice (Morgan Freeman)	30	16	44100
H.2	Female voice (Anne Hathaway) + Male voice (Patrick Warburton)	8	16	44100
H.3	Male voice (Robert Di Nero) + Loud noise	5	16	44100
H.4	Female voice (Susan Sarandon)	14	16	44100
H.5	Female voice (Susan Sarandon) + Quiet music	41	16	44100
H.6	Female voice (Suzanne Pleshette)	6	16	44100
H.7	Male voice (Heath Ledger)	11	16	44100
H.8	Male voice (Heath Ledger)	31	16	44100
H.9	Female voice (Jennifer Garner) + Male voice (Ricky Gervais)	11	16	44100
H.10	Female voice(Kate Ashfield) + male voice(Simon Pegg)	11	16	44100
H.11	Male voice(Michael Caine) + Male voice (Christian Bale)	21	16	44100

Table.1 files to be experimented

a. The first algorithm (simplest one by using directly LSM with fixed N_s)

Encoding part;	Decoding part;
Input: y, n, N_s Step1: Create empty array for compressed values (store the coefficients).	Input: <i>compressed</i> , n, N_s Step1: Create empty array for y (recovered values). Step2: For each $n + 1$ elements in <i>compressed</i> (the

Step2: For each N_s samples in : - Perform LSM with n degree on picked samples. - Concatenate the resulting coefficients with compressed. Step3: Return compressed.	coefficients): - Deduce N_s elements present samples values by evaluating n polynomial. - Concatenate the resulting samples values with y. Step3: Return y.
--	--

b. The second algorithm (N_s changed every operation according to amount of error tolerance between the original and the recovered audio samples called *diff*)

Encoding part;	Decoding part;
Input: $y, n, diff$. Step1: Create two empty arrays one for compressed and one for samples. Step2: Search for maximum number of samples in y where after performing LSM and recover $y_{max}(differences) < diff$. Step3: Store the Coefficients in <i>compressed</i> and number of samples used in <i>samples</i>. Step4: Repeat step3 and step4 for the whole file. Step5: Return <i>compressed</i> & <i>samples</i>.	Input: <i>compressed, samples, n</i> Step1: Create empty array for y (recovered values). Step2: For each element value (N) in . - Evaluate N samples values using $n + 1$ elements from <i>compressed</i> - Concatenate the results with y. Step3: Store the Coefficients in <i>compressed</i> and number of samples used in <i>samples</i>. Step4: Return y.

c. The third algorithm (reconstructing the audio samples vector to make convergence and apply fixed N_s)

Encoding part;	Decoding part;
Input: n, N_s . Step1: Create two empty arrays one for compressed values (store the coefficients). Step2: For each N_s samples in : - For each sample in the N_s that picked up. $sample(i) = samples(i) + samples(i - 1)$. - Perform LSM with n degree on the modified samples. - Concatenate the resulting coefficients with compressed. Step3: Return <i>compressed</i> .	Input: <i>compressed, n, N_s</i> Step1: Create empty array for y (recovered values). Step2: For each $n + 1$ elements in <i>compressed</i> (the coefficients): - Deduce N_s elements present samples values by evaluating n polynomial. - For each sample in N_s (in reverse order) $sample(i) = samples(i) - samples(i - 1)$. - Concatenate the resulting samples values with y Step3: Return y.

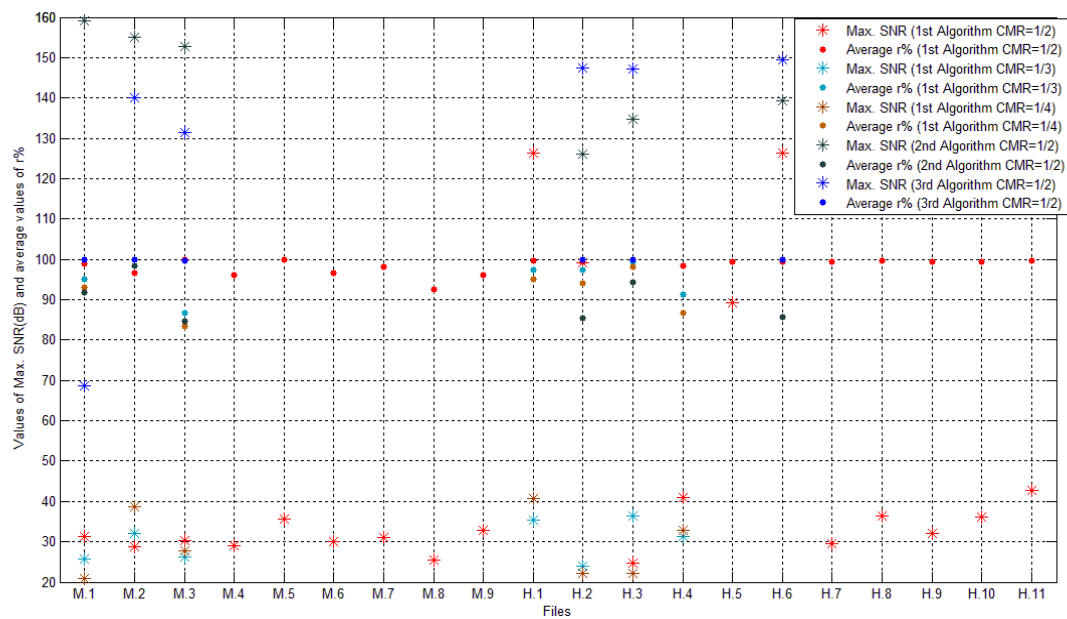


Figure 1: collect Max. SNR and average r% for the results from the three algorithms

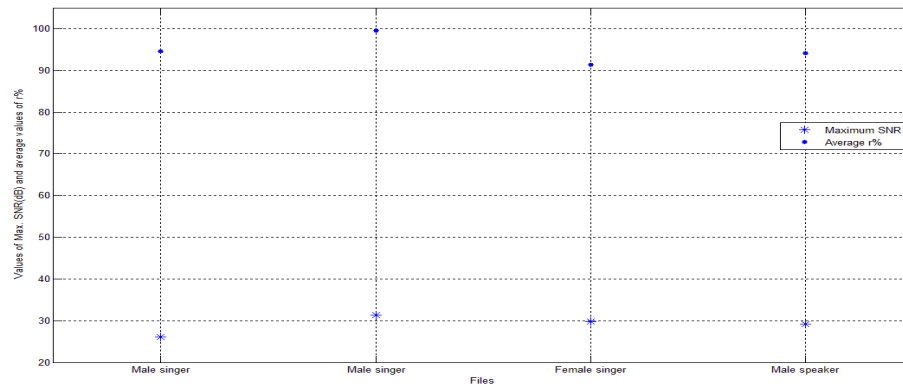


Figure.2 average r% and maximum SNR for some tested mp3 files

4. Results and comparison with PCM

In this section, we will present some results obtained by using the algorithms deduced in this research, also we will make comparison between our algorithm and PCM used in telephony system.

4.1. Results

We present in Figure 3 the results of applying the first algorithm by using $n = 4$ and $N_s = 10$ ($CMR = 0.5$) on three audio files. The recovered audio resembles the original one in the quality and our ears could not notice a quality difference, as it is clear that SNRs are high.

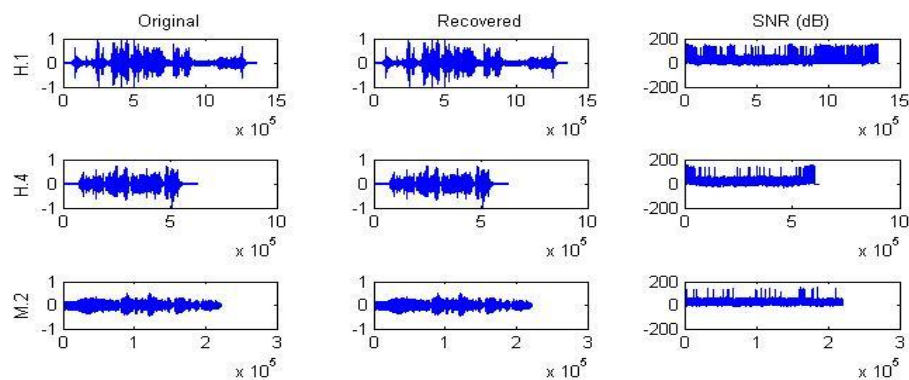


Figure. 3 full experiments' results using our 1st algorithm on 3 audio files

We present in figure 4 the results of applying the first algorithm by using $n = 4$ and $N_s = 15,20$ and $CMR = \frac{1}{3}, \frac{1}{4}$ on H.1 and we sense some differences in the quality. However, the voice was recognized well and SNR is relatively high.

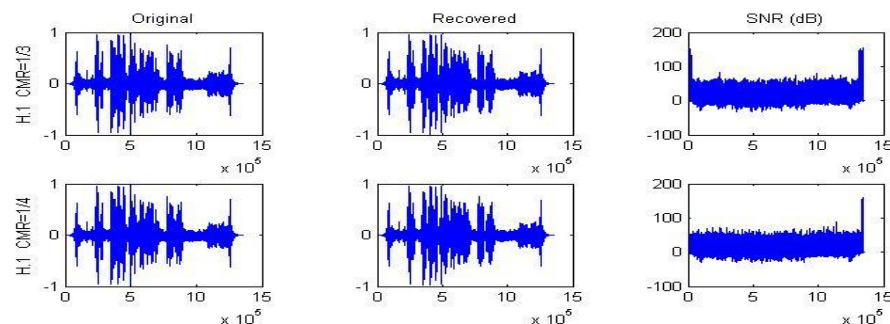


Figure. 4 full experiments' result using our 1st algorithm on file with different CMR

We present in figure.5 the recovered audio using the first algorithm with $CMR = \frac{1}{3}$ and the same $CMR = \frac{1}{3}$ but with the second algorithm. It's clear that the SNR in the second algorithm show more consistency because we designed this algorithm for obtaining high and consistent quality all over the samples.

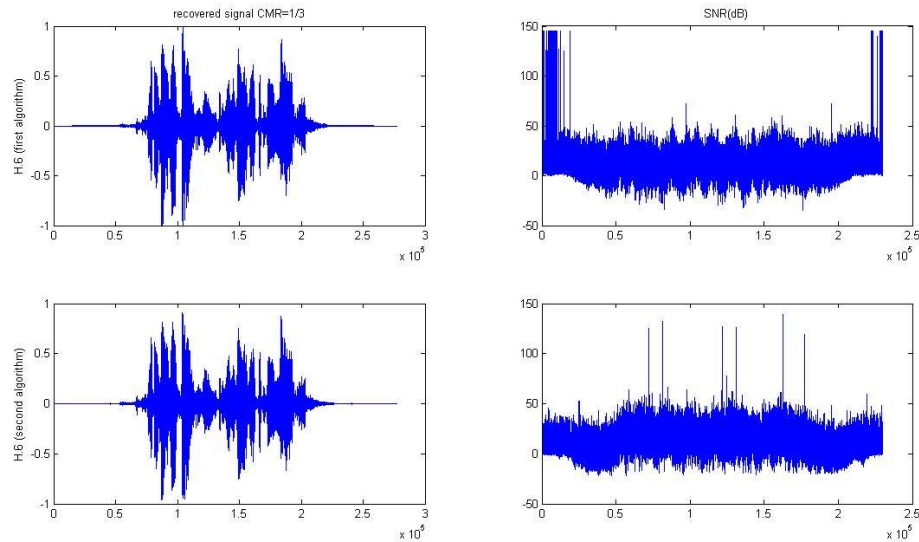


Figure. 5 comparison on the same file between our 1st and 2nd algorithms

4.2. Comparison with PCM

The telephony system uses PCM concept as a modulation technique through transmission. We preferred to compare our simplest CODEC which is the first algorithm with the results obtained using PCM. We believe that the ability of adding our algorithm directly to the samples values after sampling process will be better than applying quantization used generally in many CODECs and specifically in PCM. Consequently, we applied PCM on several files and in our files' case that will produce $CMR = 0.5$ because bit rate is 16 and in telephony system it's usually 8. We applied our first algorithm to obtain $CMR = 0.5$ on the same files. We achieved higher average SNR in most files and significantly higher maximum SNR in all files. We also clarified the difference between our algorithm and PCM in specific number of samples to show our superior results. Figure 6 shows the comparisons of average and maximum SNR made on some files.

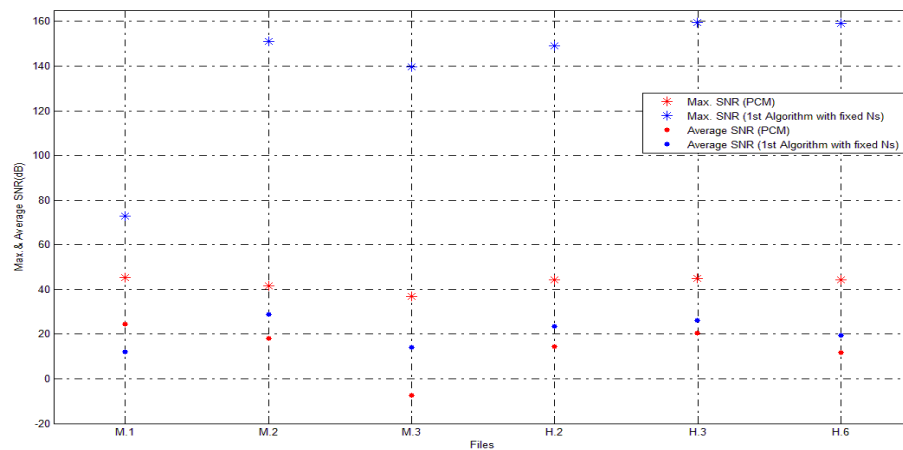


Figure.6 comparisons between our algorithm with PCM on the same files

We preferred to present 100 samples from one file two times, and compare the recovered samples values from PCM with our first algorithm, in figure 7.

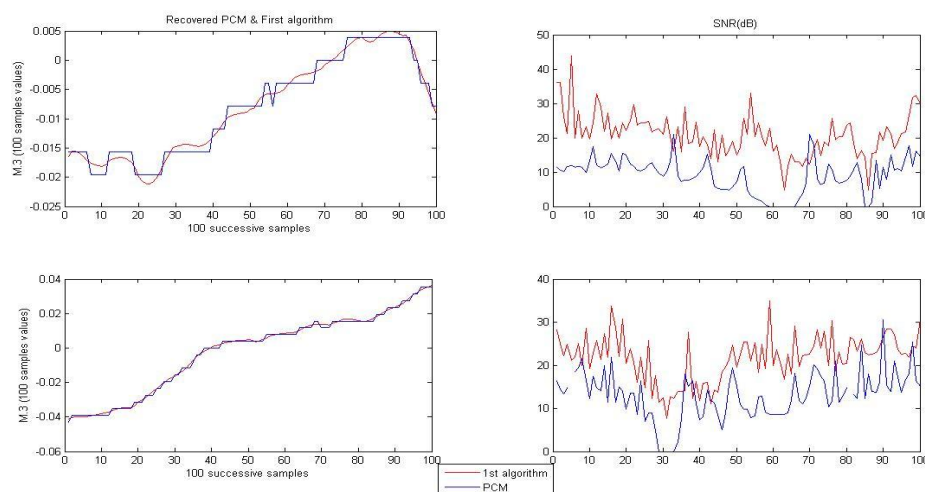


Figure.7 comparisons over 100 samples from audio file (our algorithm with PCM)

5. Conclusions and expectation

We introduced three algorithms for encoding/decoding (CODEC) audio data. We used the interpolation method known as "LSM" in order to deduce these algorithms. The first algorithm based on applied LSM on fixed N_s until all the samples are processed. The second algorithm improved the first one by initializing a amount of error tolerance "diff" between the original values and the recovered values, to ensure the quality all over the data, taking advantages from some of the voice characteristics like silence durations. The third algorithm works like the first one but in a different distribution of sample values. The new distribution results from adding each sample's value to the previous one to generate new consistent vector to increase convergence between the values of the new vector. Pseudo-codes proposed in this research for every algorithm. We presented average value of r Eq.(5) and SNR for a lot of files (including MP3 files) experimented by the three algorithms..

Clear comparison between our simplest algorithm (the first) and PCM used in telephony system have shown progress for our work. The values of the coefficients stored resemble the values of the audio's samples in the scale and the sensitivity, so the values of the coefficients can be used not only in compression applications but also in transmission applications, because they can be sent as serial data with the same digital sequences known in quantization levels, but with higher quality (same compression). We believed that our algorithms can replace the quantization step in any CODEC depending on quantization as a part or as a whole like PCM.

The methodology we used in this research is a basic idea for a lot of researches we are working now on, to make image and video compression beside voice and image recognition. We are expecting that this path will open a lot of applications and ideas that depend on LSM. We now are using the results of this paper as a basic for many contributions.

References:

- [1]. Bhaskar, U. , Low bit-rate voice compression based on frequency domain interpolative techniques, Audio, speech, and language processing, IEEE Transactions, March 2006, issue 2, pages 558-576.
- [2]. Din-Yuen Chan, Cheng-Yuan Ku, A low-complexity, high-quality, 64-Kbps audio codec with efficient bit allocation, Digital Signal Processing 13 (2003) 23–41.
- [3]. Dongge Li, Ishwar K.Sethi, Nevenka Dimitrova, Tom McGee, Classification of general audio data for content-based retrieval, Pattern recognition Letters 22 (1001) 533-544.
- [4]. Eric D. Scheirer, Youngjik Lee, Jae-Woo Yang, Synthetic and SNHC audio in MPEG-4, Signal Processing: *Image Communication* 15 (2000) 445}461.
- [5]. G. CARBALLO, R ALVAREZ-N• DARSE and J. S. DEHESA, Chebychev Polynomials in a Speech Recognition Model, Applied Mathematics Letters 14 (2001) 581-585 .
- [6]. Ge Gao, P.C. Ching, Tan Lee, A new approach to generating pitch cycle waveform (PCW) for waveform interpolation codec, Microprocessors and Microsystems 25 (2002) 421-426.
- [7]. JEROEN BREEBAART, GERARD HOTHO, JEROEN KOPPENS, ERIK SCHUIJERS, WERNER OOMEN, STEVEN VAN DE PAR, Background, Concept, and Architecture for the Recent MPEG Surround Standard on Multichannel Audio Compression, J. Audio Eng. Soc., Vol. 55, No. 5, 2007 May.
- [8]. Karlheinz Brandenburg, Oliver Kunz, Akihiko Sugiyama, MPEG-4 natural audio coding, Signal Processing: *Image Communication* 15 (2000) 423-444.
- [9]. Kris Hermus, Werner Verhelst, Philippe Lemmerling, Patrick Wambacq, Sabine Van Huffel, Perceptual audio modeling with exponentially damped sinusoids, Signal Processing 85 (2005) 163–176.
- [10]. L. Contina, B. Edler, D. Meares, P. Schreiner, Tests on MPEG-4 audio codec proposals, Signal Processing: *Image Communication* 9 (1997) 327-342.
- [11]. McPherson, T., Jr. , PCM speech compression via ADPCM/TASI, Acoustics, speech, and signal processing, IEEE international conference on ICASSP 77, pages 184-187.
- [12]. Richard L.Burden, J.Douglas Faires, Numerical Analysis Ninth Edition, ISBN-13: 978-0-538-73351-9; ISBN-10: 0-538-73351-9 .
- [13]. Shugang Wei , Audio dynamic range compression characteristics based on an interpolating polynomial, Circuits and systems, 2004, NEWCAS 2004, The 2nd Annual IEEE Northeast Workshop 133 – 136.
- [14]. S. Molla, B. Torr sani, A hybrid scheme for encoding audio signal using hidden Markov models of waveforms, Appl. Comput. Harmon. Anal. 18 (2005) 137–166.
- [15]. Vladimir Britanak, A survey of efficient MDCT implementations in MP3 audio coding standard: Retrospective and state-of-the-art, Signal Processing 91 (2011) 624–672.
- [16]. Yung-Gi Wu, Chia-Hao Wu, Image vector quantization codec indices recovery using Lagrange interpolation, Image and Vision Computing 26 (2008) 1171–1177.