www.czasopisma.pan.pl PAN POLSKA AKADEMIA NAUK www.journals.pan.pl

ARCHIVES OF ACOUSTICS
36, 2, 461–478 (2011)

DOI: 10.2478/v10168-011-0032-x

Parametric Audio Based Decoder and Music Synthesizer for Mobile Applications

Marek SZCZERBA[(1),(2),(3)], Werner OOMEN[(1)], Dieter THERSSEN[(3)]

[(1)] *Philips Research*
Eindhoven, The Netherlands
e-mail: marek.szczerba@philips.com

[(2)] *Philips Lighting*
Eindhoven, The Netherlands

[(3)] *Philips Lifestyle Incubators*
Eindhoven, The Netherlands

(*received February 22, 2011; accepted April 12, 2011*)

This paper reviews parametric audio coders and discusses novel technologies introduced in a low-complexity, low-power consumption audio decoder and music synthesizer platform developed by the authors. The decoder uses parametric coding scheme based on the MPEG-4 Parametric Audio standard. In order to keep the complexity low, most of the processing is performed in the parametric domain. This parametric processing includes pitch and tempo shifting, volume adjustment, selection of psychoacoustically relevant components for synthesis and stereo image creation. The decoder allows for good quality 44.1 kHz stereo audio streaming at 24 kbps. The synthesizer matches the audio quality of industry-standard sample-based synthesizers while using a twenty times smaller memory footprint soundbank. The presented decoder/synthesizer is designed for low-power mobile platforms and supports music streaming, ringtone synthesis, gaming and remixing applications.

Keywords: parametric audio, music synthesis, music creativity, decoding, low-complexity, mobile, psychoacoustics.

1. Introduction

Modern mobile devices, such as smartphones, put high demands on their audio engines. There are numerous requirements that must be fulfilled such as audio quality, CPU usage, memory footprint and available bandwidth. Mobile phones are more and more used as personal audio devices, setting up the necessity for high audio quality. Furthermore, audio is often used in the background (e.g. in

case of gaming) along with other computationally demanding processes, such as animations. Nonetheless, while used alone, audio playback should not drain the battery of a mobile device too quickly. These two requirements require the audio processing complexity to be as low as possible. Furthermore, since the importance of online audio services grows rapidly, the observed quality of such service is directly related to the quality of the Internet connection. Brief drop-outs in an Internet connection should not impact the perceived quality of an online audio service.

Along with the growth of online audio services, the number of creative applications also grows. These include, but are not limited to, mixing and mashing applications, allowing users to combine a number of audio tracks, modify and mix them together in order to create personalized audio content. In parallel, gaming applications that implement creative elements raise similar requirements. Therefore it is desired that audio content should be easily adjustable, specifically in terms of adjusting its volume, pitch and tempo independently.

Most of the mobile devices available on the market today, rely on the state-of-the-art psychoacoustic audio codecs for music playback, such as MPEG-2 Audio Layer 3 or AAC (Brandenburg, 1999). Multiple hardware and software based implementations of such decoders are available. Since these codecs allow for high-quality audio, their decoding algorithms are not very computationally demanding and their implementations strongly optimized, these technologies allow for low complexity audio playback on a mobile device. However in the case of Internet bandwidth limitations, which often occurs in dense urban areas or while travelling, audio quality is typically compromised. Furthermore, audio processing as required for creative applications requires high-complexity processing. Hence, independent pitch and tempo adjustment of an audio signal is hardly feasible in mobile applications.

For the music synthesis, as in the case of ringtone and gaming applications, sample-based synthesis is widely used. Most of the synthesizers use DLS (Downloadable Sounds) or equivalent soundbanks consisting of PCM samples as specified in (MMA Technical Standards Board [DLS], 2000). Transform codecs are hardly ever used for encoding soundbank audio data. In order to obtain a sound of a required pitch, decimation and interpolation algorithms are used, as in the case of sample rate conversion (Horner *et al.*, 1993). Furthermore, in order to keep the synthesis complexity low, the interpolation filter quality of a sample-based synthesizer deteriorates proportionally to the current polyphony (Horner *et al.*, 1993). This can result in noisy artefacts accompanying dense fragments of a synthesized music.

This paper discusses a novel audio engine for mobile applications based on parametric audio coding. The presented technology allows for combining playback, synthesis and creativity functions in a single software stack, while avoiding most of the drawbacks of the state-of-the-art transform coding based audio playback and sample-based synthesis.

The first part of the paper provides a short tutorial on the parametric audio coding principles. The following sections discuss technologies applied to the parametric audio decoder and the parametric audio based synthesizer. The technical solutions used for parametric audio decoding and synthesis, already presented in (Szczerba *et al.*, 2004) and in (Szczerba *et al.*, 2008) are discussed in more detail.

2. Principles of parametric audio coding

Parametric audio coding schemes go beyond traditional transform codecs with respect to sound analysis and more severely exploit the psychoacoustic constraints. The parametric audio engine discussed in this paper employs the parametric audio model as standardized by MPEG (Parametric Coding for High Quality Audio [MPEG], 2003) and further discussed in (Myburg, 2004; Brinker *et al.*, 2002; Schuijers *et al.*, 2003). This codec, also referred to as SSC (Sinusoidal Coding), decomposes the audio input into transient, sinusoids and noise components and generates a parametric representation for each of these components. The basic block diagram of the SSC encoder is shown in Fig. 1.

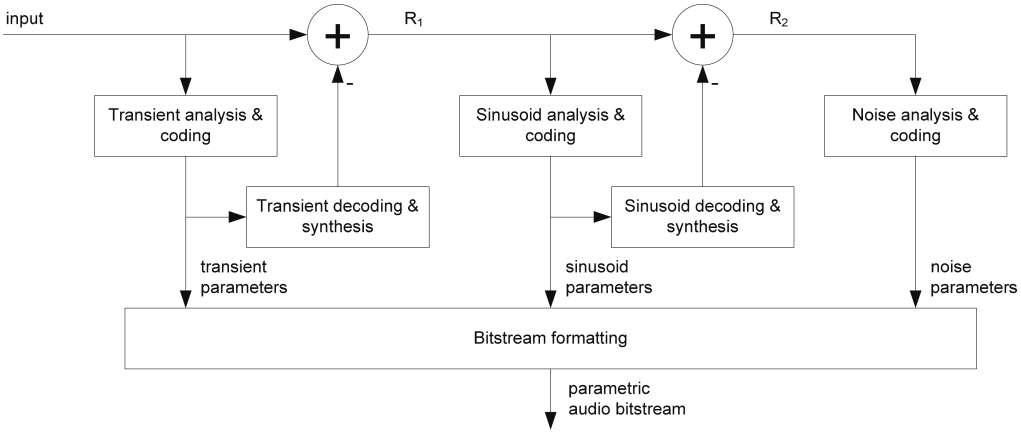

Fig. 1. SSC encoder block diagram.

The input signal is initially split into overlapping time frames. The encoding process for each frame is performed sequentially. First a transient detection is performed. If a transient is detected, the interval containing the transient is analyzed in order to acquire parametric representation of the transient. The transient is reconstructed in turn and subtracted from the original signal forming the first residual signal R_1. In the case no transient is detected, the first residual signal R_1 is equal to the input signal.

Subsequently, a sinusoidal analysis is performed based on the residual signal R_1. Periodic components are identified based on peak analysis of the power spec-

trum representation of the selected time frame. The power spectrum is obtained using the Fast Fourier Transform of a Hanning windowed time frame of the input signal. The most relevant spectrum peaks related to periodic signal components are selected and analyzed in order to acquire the parametric representation of periodic components. As in the case of transients, the sinusoidal portion of the signal is reconstructed and subtracted from the first residual signal R_1 to form the second residual signal R_2.

The second residual signal R_2 mainly contains the noise components of the input signal. The R_2 signal is subsequently analyzed in order to acquire its spectral and temporal envelopes.

All of the parametric data describing audio components are further quantized and coded using lossless data compression, specifically Huffman coding (Cormen *et al.*, 2001).

In order to reflect the perceived relevancy of parameter time-variance, the analysis is performed in blocks of different size for the various components. This in turn is reflected by the constitution of the parametric audio bitstream. The bitstream consists of blocks, as illustrated in Fig. 2.

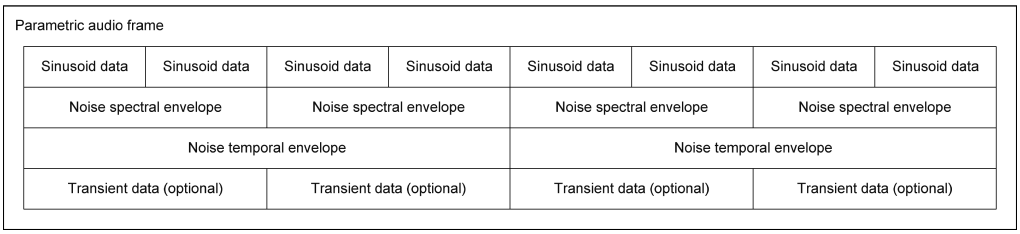

Fig. 2. Parametric audio data frame structure.

The audio quality of the SSC codec has been evaluated against the state-of-the-art MPEG-4 AAC Profile codec (Information technology – Generic coding of moving pictures and associated audio information – Part 7: Advanced Audio Coding (AAC) [AAC], 2004) by means of formal listening tests, in the course of the MPEG standardization (Report on the Verification Tests of MPEG-4 Parametric Coding for High Quality Audio [MPEG], 2004). The tests were performed using the MUSHRA (Multiple Stimuli with Hidden Reference and Anchor) methodology (Method for the subjective assessment of intermediate quality level of coding systems [MUSHRA], 2003). Along with the encoded audio, a hidden reference and two bandwidth limited anchor signals (3.5 kHz and 7 kHz) were provided. The coders as used in the test are listed in Table 1. The test results are illustrated in Fig. 3.

The audio quality has been indicated using the MOS scale where the numbers denote the following artifact perception: 5 – imperceptible, 4 – perceptible but not annoying, 3 – slightly annoying, 2 – annoying, 1 – very annoying (MPEG, 2004).

Table 1. Coders under test (MPEG, 2004).

Coding scheme	Label	Bitrate [kbps]	Sampling rate [kHz]	Typical audio bandwidth [kHz]
MPEG – 4 AAC Profile	MP4-AAC-24	24	24	7
	MP4-AAC-32	32	32	10.5
	MP4-AAC-48	48	44.1	11
SSC	SSC-16	16	44.1	18
	SSC-20	20	44.1	18
	SSC-24	24	44.1	18

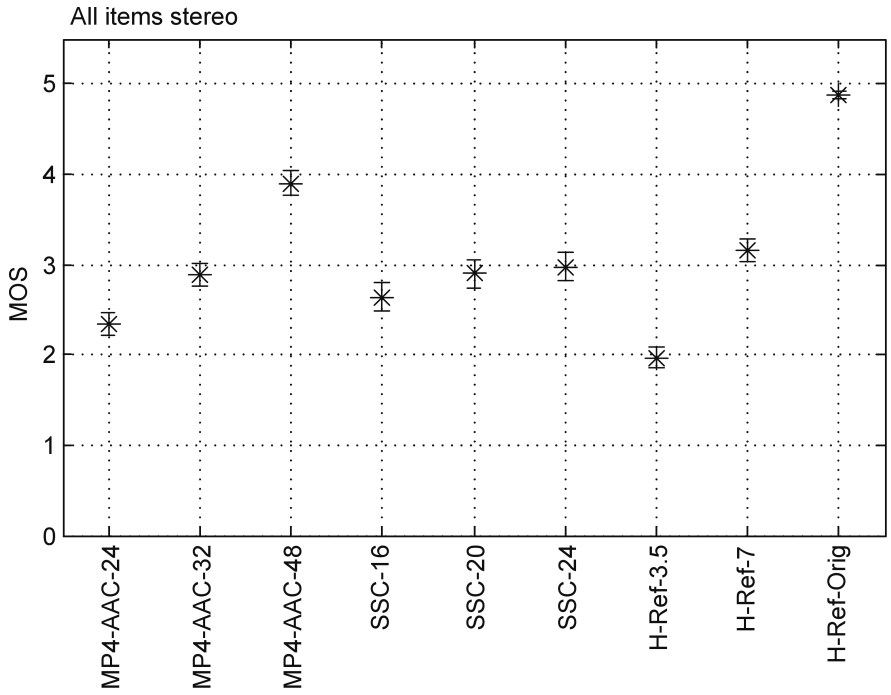

Fig. 3. Stereo listening test results (MPEG, 2003).

The results show, that the SSC codec performs statistically better than the AAC codec at very low bit-rates (24 kbps or below). The most distinctive quality discriminator of the SSC codec is its intrinsic property to maintain full bandwidth (44.1 kHz sampling frequency) at very low bit-rates. Traditional waveform codecs, as AAC in this case, operate at reduced sampling frequency at very low bit-rates in order to limit distortions. This results in band limitation at the decoded signal output. On the other hand the SSC codec quality saturates at bit-rates above 24 kbps, meaning that further increase in bandwidth hardly results in quality improvement. The listening tests prove, that the SSC parametric audio codec allows for good quality, full bandwidth stereo audio at a bitrate of 24 kbps.

Figure 4 illustrates SSC audio quality against state-of-the-art waveform codecs. The results presented are taken from different tests (EBU subjective listening tests on low-bitrate audio codecs [EBU], 2003; Report on the Verification Tests of MPEG-4 High Efficiency AAC [HE-AAC], 2003; MPEG, 2004; Listening test report on MPEG-4 High Efficiency AAC v2 [HE-AAC2], 2005). Therefore, this illustration should only be considered as an indication.

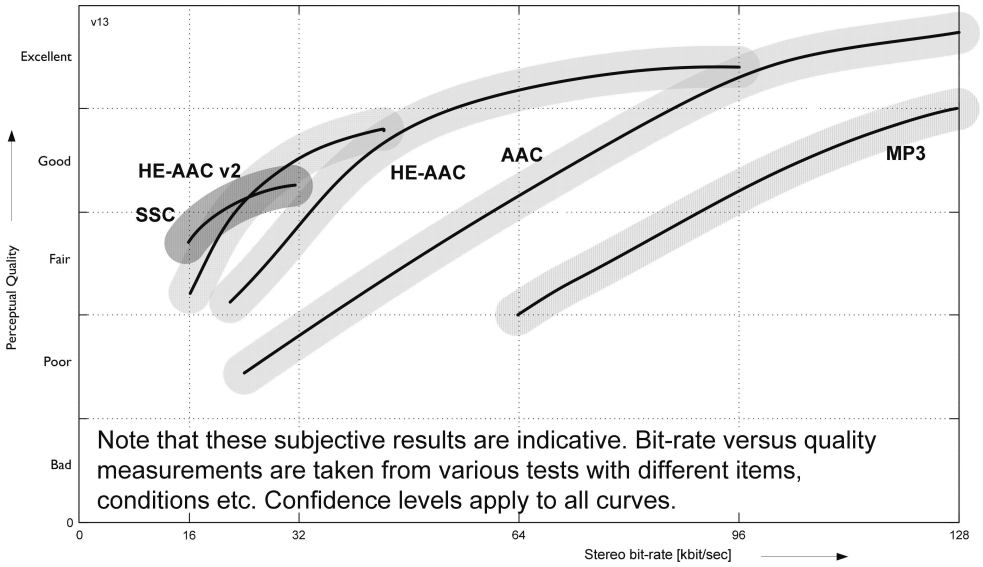

Fig. 4. Audio quality vs. bit-rate for the SSC codec and the state-of-the-art waveform codecs.

2.1. Transients

The transient components in the SSC codec represent those parts of the input signal that are characterized by impulsive amplitude changes. The SSC audio coding scheme supports two types of transients: the step transient and the Meixner transient, as discussed in (Schuijers *et al.*, 2003).

The step transients describe a sudden change in a signal power level, and it is represented by its position only. It determines the way the underlying sinusoids are synthesized.

A Meixner transient provides for a more detailed transient description including transient temporal envelope and sinusoidal components constituting the transient. The sinusoidal components underlying the Meixner transient are encoded identically as regular sinusoidal components.

2.2. Sinusoids

Sinusoidal components describe deterministic, tonal components of the encoded signal, which can be well represented by a number of sinusoidal waveforms.

The following model is used for describing sinusoidal components (SCHUIJERS *et al.*, 2003):

$$s(t) = \sum_{i=1}^{I(t)} A_i(t) \cos(\Phi_i(t)) + n(t), \tag{1}$$

with

$$\Phi_i(t) = \phi_{s,i} + \int_{t_{s,i}}^{t} \omega_i(\tau)\, \mathrm{d}\tau, \tag{2}$$

where i denotes the i-th sinusoid, $A_i(t)$ – the amplitude of the i-th sinusoid (slowly varying), $\omega_i(t)$ – the frequency of the i-th sinusoid (slowly varying) and $\Phi_i(t)$ represents the corresponding phase function with the start phase $\phi_{s,i}$. The sinusoidal analysis is performed in overlapping time frames. In order to reduce Fourier transform distortions, Hanning windowing is applied to the input signal frame.

The signal analyzer detects sinusoidal tracks by joining corresponding sinusoidal components between subsequent frames. While sinusoidal parameters of the first frame need to be provided as absolute values, subsequent sinusoidal parameters are provided differentially with respect to their previous values. This allows for benefiting from the slowly varying frequency and amplitude of periodic components in a typical audio content.

2.3. Noise

Noise components describe the remaining residual portion of an audio signal, assumed to be of a stochastic nature. The noise features are represented using temporal and spectral envelopes.

The temporal envelope is described using a gain value g and a set of linear prediction coefficients $\beta = [\beta_1, \beta_2, \ldots, \beta_{K_t}]$. The temporal envelope $e(n)$ is synthesized in the frequency domain as follows (HERRE, JOHNSTON, 1996):

$$e(n) = \frac{1}{\left|1 - \sum_{k=1}^{K_t} \beta_k e^{-j2\pi k/N}\right|}. \tag{3}$$

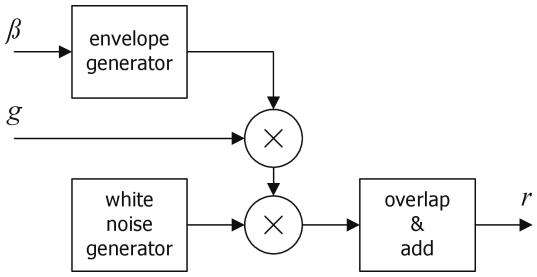

Fig. 5. Temporally-shaped noise signal generator.

The temporal envelope data is encoded every four subsequent audio subframes. Successive temporal envelope segments are combined using overlap and add algorithm. The noise signal frame with its temporal envelope shaped accordingly is further used as the input for the spectral-envelope shaping filter.

The noise synthesis filter is based on the Laguerre model (Schuijers *et al.*, 2003) and is characterized by the following transfer function:

$$\frac{1}{H(z)} = 1 - A_0(z) \sum_{k=1}^{K} \alpha_k \{A(z)\}^{k-1}, \tag{4}$$

where

$$A_0(z) = \sqrt{1-\lambda^2}\frac{z^{-1}}{1-z^{-1}\lambda}, \tag{5}$$

$$A(z) = \frac{-\lambda + z^{-1}}{1-z^{-1}\lambda} \tag{6}$$

and λ is a Laguerre filter parameter, which can be tuned according to perceptual relevant frequency scale.

The block diagram of the noise synthesis filter is shown in Fig. 6.

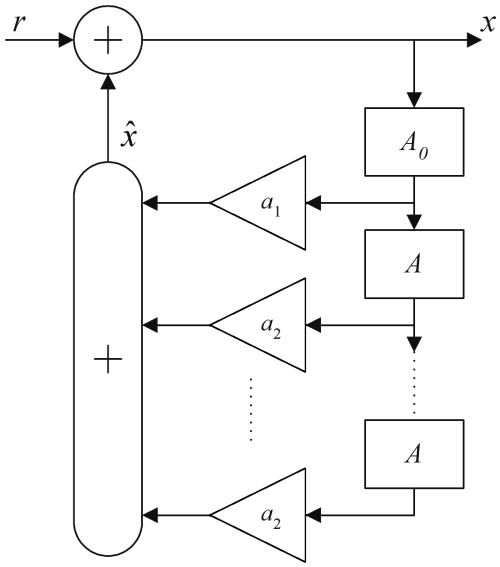

Fig. 6. Laguerre-based noise synthesis model.

The Laguerre filter allows for an efficient representation of a spectral envelope of a typical residual of an acoustic signal while preserving harmonic-related formant characteristics (Voitshchuk *et al.*, 2001).

2.4. Parametric stereo

The encoder employs a parametric stereo encoding, as discussed in (SCHUIJERS *et al.*, 2004). The block diagrams of a parametric stereo encoder and decoder are illustrated in Fig. 7 and Fig. 8 respectively.

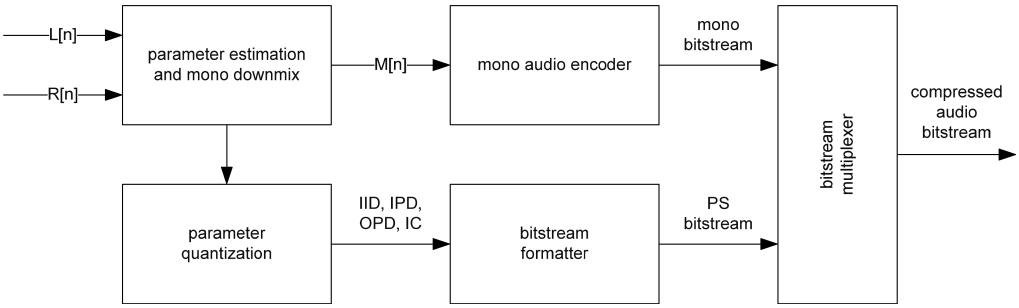

Fig. 7. Parametric stereo encoder block diagram.

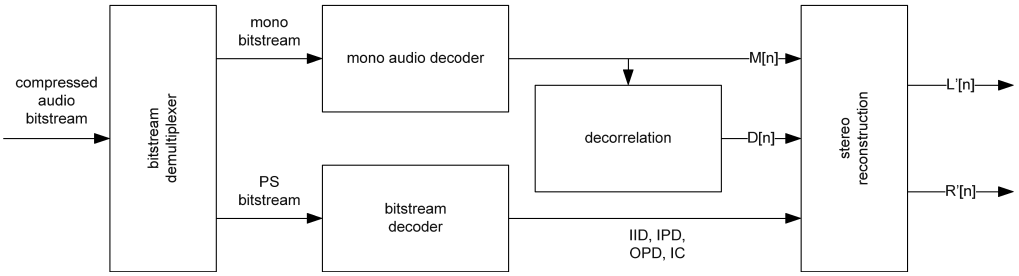

Fig. 8. Parametric stereo decoder block diagram.

The stereo parameters describing the audio signal include Inter-Channel Intensity Differences (IID), Inter-Channel Phase Differences (IPD), Inter-Channel Coherence (IC) and Overall Phase Difference (OPD).

In the applications related to the waveform codecs, parametric stereo decoding is generally applied to the signal portions obtained using hybrid filter banks, such as QMF banks, as discussed in (SCHUIJERS *et al.*, 2004). The decorrelation can be efficiently implemented by means of reverberator-like algorithms or even simple delays.

3. Parametric audio decoder

Parametric audio decoding requires a number of computationally demanding processing steps. The reference decoding scheme as provided by MPEG standard (MPEG, 2003) does not allow for an efficient implementation on mobile platforms. In order to reduce the computational complexity of the decoding process a num-

ber of optimizations have been proposed by the authors. Optimized decoding algorithms are further discussed in more detail regarding individual parametric audio components.

3.1. Parametric stereo

Application of the parametric stereo in the sub-band domain, as applied in the state-of-the-art waveform based decoders, would result in high computational complexity. Therefore, an alternative solution has been proposed, based on application of parametric stereo data directly to the parametric audio representation, as discussed in (Szczerba, 2008). This specific solution is illustrated in Fig. 9.

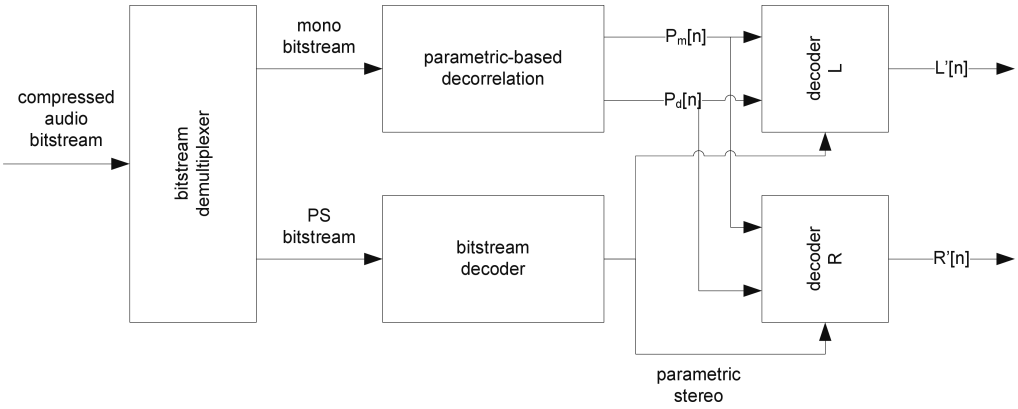

Fig. 9. Parametric stereo decoder operating in parametric audio domain.

Details regarding parametric stereo decoding of separate parametric audio components are discussed in the following sections.

3.2. Sinusoidal component synthesis

Sinusoidal components are encoded in eight blocks per complete parametric audio frame. Each block (called sub-frame) represents 384 PCM samples at a sampling rate of 44.1 kHz.

In order to reduce computational complexity, sinusoidal components are synthesized in the spectral domain. For each sinusoidal component its complex spectrum representation is generated and convolved with the appropriate portion of the oversampled complex spectrum of the Hanning window, as discussed in (Szczerba, 2004). The convolution is performed using a pre-defined spread value. In the current implementation the spread value is set to 7, such that 7 frequency bins are generated for each sinusoidal component. The spectra of individual sinusoidal components are subsequently accumulated. Once a complete complex spectrum for a sub-frame is generated, its time-domain PCM representation is computed using a single IFFT operation. The first half of the PCM

sub-frame is added to the content stored in the PCM buffer and provided to the output. The second part is stored in the same PCM buffer in order to be used while generating a subsequent sub-frame. No signal data needs to be buffered.

If parametric stereo data is present, the stereo parameters are applied directly to the sinusoidal components. The decorrelation is obtained by means of delaying sinusoidal parameters by a number of subframes.

3.3. Noise component synthesis

For noise synthesis, optimized algorithms as proposed in the reference decoder (MPEG, 2003) are used. Initially, a frame of white noise signal is generated. Subsequently, the temporal envelope of the white noise is shaped. Thereafter, a Laguerre filter is applied in order to shape its spectral characteristics.

If the parametric stereo encoding is applied, the decorrelated signal is generated using modified, delayed output of the white noise generator, which is further shaped using delayed temporal and spectral shaping parameters.

3.4. Transient component synthesis

Internal listening tests show that step transients are usually not relevant for a typical audio content in terms of psychoacoustics. Since step transients would require synthesis in the time domain, this would generally result in a significant increase in computational complexity. For these reasons, step transients are synthesized as regular sinusoidal components. For typical audio content this simplification only results in insignificant perceived audio quality deterioration.

Meixner transients are usually much more relevant than step transients. Transient synthesis is based on optimized algorithms as used in the reference decoder, as provided in (MPEG, 2003).

A presence of transients in combination with high phase difference values of parametric stereo may result in deteriorated audio quality. Therefore, parametric stereo decoders are often equipped with transient detectors, which allow for temporarily decreasing the amplitude of the decorrelated portion of a signal at the transient interval. Since parametric audio components are synthesized separately, this is not an issue in the case of the decoder under discussion. In the case of a transient component, only intensity difference parameters (IID) are taken into account, disregarding the phase related parameters (IPD).

3.5. Features and audio quality

An ability to efficiently and independently modify pitch and tempo of audio content provides for a unique feature of parametric audio decoders. This feature is very useful for a variety of music applications as well as for adjusting speech tempo. Pitch shifting is realized by adjusting the frequency parameters of the

transient and sinusoidal components. The noise parameters remain unchanged preventing from displacement of potentially relevant noise formants. Tempo can be changed by repeating or skipping sinusoidal data sub-frames and accordingly adjusting the length of the noise signal obtained by temporal envelope shaping.

The audio quality of the optimized parametric audio decoder has been evaluated by means of informal listening tests and compared to the one of the MPEG Parametric Audio reference decoder (MPEG, 2003). A set of MPEG listening test samples was used for this experiment, containing the following audio items: es01 (Suzanne Vega), es02 (male speech, German), es03 (female speech, English), sc01 (trumpet), sc02 (orchestra), sc03 (pop music), si01 (cembalo), si02 (castanets), si03 (pitch pipe), sm01 (bagpipe), sm02 (glockenspiel), sm03 (plucked strings). The tests were performed by three listeners. The quality was compared against reference decoder and a hidden reference signal without using anchor signals. Therefore these test results should be taken only as an indication and should by no means be compared to the results as presented in Sec. 2. The test results are illustrated in Fig. 10.

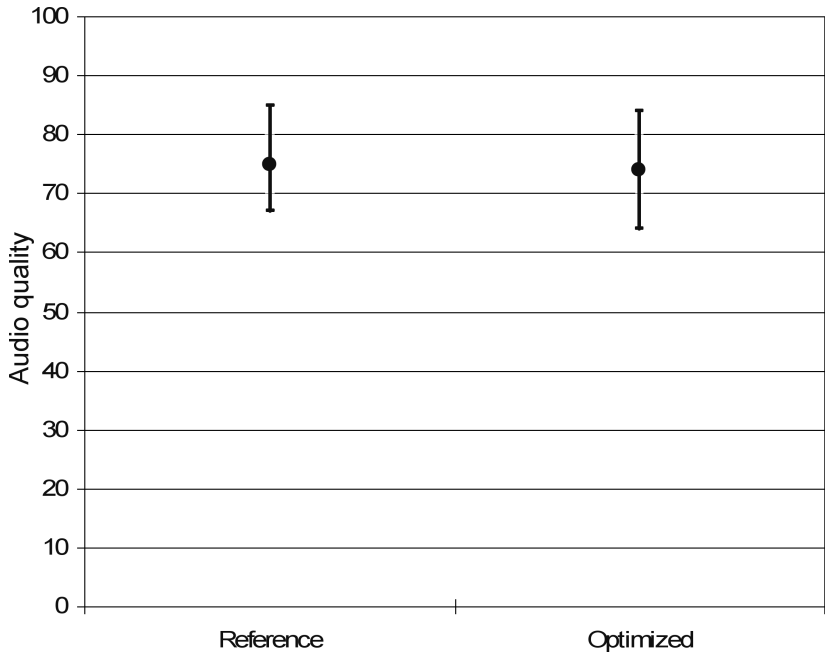

Fig. 10. The comparison of the audio quality of the MPEG reference decoder (MPEG, 2003) and the optimized parametric decoder.

The test results show only a marginal impact to audio compared to the MPEG Parametric Audio reference decoder. At the same time, the proposed decoding scheme allows for significant computational complexity reduction, making this solution suitable for demanding mobile audio applications.

4. Parametric audio synthesizer

The parametric audio synthesizer is based on the same core processing components as the parametric audio decoder. The synthesizer can be controlled employing a standard MIDI data stream. It employs a dedicated parametric audio based soundbank and efficient synthesizers for audio components.

The computational complexity of the synthesis engine depends on the sampling frequency and the number of output channels. Contrary to the traditional sample-based synthesis, where complexity is linearly proportional to the polyphony, in case of parametric audio based synthesizer, increasing the polyphony level does not result directly in increased complexity.

4.1. Soundbank

The parametric audio synthesizer employs soundbanks consisting of parametric audio samples. Audio samples are stored using a slightly modified parametric audio bitstream format, allowing for better suitability for the synthesis applications. A soundbank sample can be provided with an additional sub-frame of data (glue frame). A glue frame provides a difference between the last and the first sub-frame of the loop, as illustrated in Fig. 11. There is no need for providing a glue frame if a sample does not use looping or if the loop is exactly one sub-frame long.

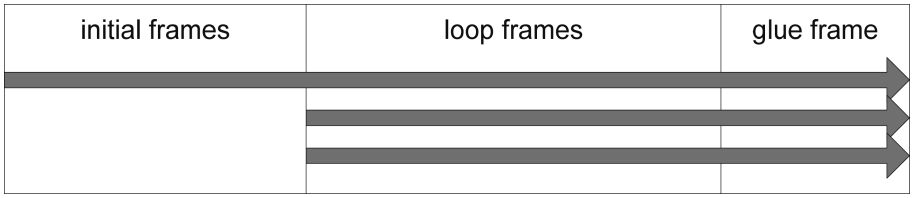

Fig. 11. Parametric audio synthesizer sample data blocks.

The soundbank definition data is stored along with the parametric audio samples using a proprietary binary format loosely associated with the DLS soundbank definition data (DLS, 2000). The looping definitions are frame-based instead of sample-based as in the case of the DLS format.

Most of the commercially available PCM sample-based synthesizers use resampling for pitch shifting. Hence, one octave transposition results in altering the sample tempo twice. This causes the sample tempo to be altered proportionally with a pitch shift rate, which might lead to artificial sound while transposing a sample significantly, which is especially annoying in the case of rich acoustic instrument sounds, such as piano, strings, etc. In order to avoid significant pitch alteration, a soundbank needs to contain multiple samples of such instruments, what in turn results in increased soundbank size.

In the case of the parametric audio based synthesizer, tempo and pitch changes can be achieved independently. However, if the tempo of co-sounding samples of rich acoustic instruments would remain the same, it would result in annoying, pulsing artefacts. Therefore a sample tempo is modified only slightly, as a function of pitch change. The difference in tempo changes reflecting pitch alterations between PCM based and the parametric audio based synthesizers is illustrated in Fig. 12.

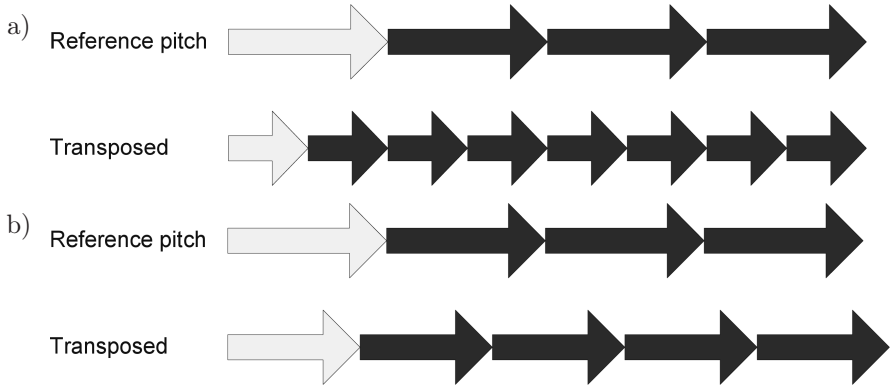

Fig. 12. Impact of pitch change on tempo change of a rich acoustic instrument sample: a) sample-based synthesis, b) parametric based synthesis.

Since the sample tempo does not scale linearly with the pitch-shifting ratio, as in the case of the sample-based synthesis, the samples of acoustic instruments can be used in a significantly larger pitch range. The number of samples used for a single instrument is therefore determined by its acoustic properties alone, not by looping artefacts caused by resampling. It allows for extensively reducing a number of sound samples stored in a soundbank. This, in combination with the superb compression ratio of the parametric audio encoder allowed for the development of a parametric soundbank of 100 kB footprint, which a performance in quality that is comparable to the state-of-the-art PCM soundbanks of 4 MB.

4.2. Sinusoidal component synthesis

Mobile music synthesizers are required to handle high polyphony levels, where often more than 32 notes need to be played simultaneously. Therefore, synthesizing a vast number of sinusoidal components originating from different instrument samples might result in an excessive computational complexity. In order to keep the computational complexity low, only the most relevant sinusoidal components are synthesized. The selection is performed per ERB (Equivalent Rectangular Bandwidth), (Smith, Abel, 1999); the components of the highest amplitude are selected for each band. The maximum number of selected components per ERB band is set to three. This number represents the best compromise between

audio quality and processing complexity. The quantized sinusoid frequency data format as provided in the bitstream allows for an efficient mapping into ERB scale. The pitch offset in the quantized frequency domain is further estimated based on MIDI parameters (key, pitch bend, LFO, etc.) and a MIDI reference key of a sample in the soundbank. The computed pitch offset is further applied to the quantized frequency data, resulting in an ERB band estimate. The energy of selected sinusoids is not adjusted by the energy of omitted ones since the correlation between sinusoids is assumed to be zero.

Similarly to the parametric audio decoder, sinusoidal components are synthesized in the spectral domain. Since the sound to be synthesized in most of the cases has a different pitch than the appropriate sound from the soundbank, pitch processing needs to be applied. The pitch-shifting is eventually applied after dequantization of the frequency data.

Pitch is altered in the parametric domain, affecting frequency and phase parameters of sinusoidal components. Furthermore, amplitude scaling is applied to sinusoidal components reflecting MIDI data as provided by the sequencer, e.g. velocity, MIDI channel volume, current ADSR amplitude, amplitude LFO, etc.

Initial volume is eventually adjusted for stereo output signals by applying appropriate MIDI panning values. As a result of the abovementioned processing, a matrix of sinusoidal parameters is available for the spectrum domain sinusoidal synthesizer, as discussed in Subsec. 3.1. However, in the case of music synthesis, no decorrelated (side) signal is used, as in the case of the parametric audio decoder using parametric stereo. Therefore no convolution of decorrelated signal portion is necessary, resulting in a further complexity reduction.

4.3. Noise component synthesis

As in the case of the sinusoidal component synthesis, synthesizing multiple noise components originating from different instruments is not feasible bearing in mind complexity constraints. Therefore, psychoacoustic-based selection of most relevant noise components takes place based on maximum noise portion energy estimate based on a product of MIDI volume and temporal envelope energy estimate. The volume of the selected noise component is adjusted in order to reflect the energy of noise components not used for synthesis.

Initially, a solution has been proposed providing a combination of three strongest noise components by means of computing common Laguerre filter and temporal shaping filter coefficients, as discussed in (Szczerba, 2004). However, in order to further optimize the complexity, it has been eventually chosen to use a single strongest noise component filter coefficients for shaping the temporal and spectral envelopes of the noise portion. The temporal envelope is further adjusted using a fraction of the energy estimate of the noise components not used for synthesis. Internal listening tests show only a small decrease in quality when applying this simplified solution over the algorithm incorporating combin-

ing filter coefficients of three most relevant noise components. In the same time, a significant computational complexity reduction is achieved.

Pitch shifting is not applied to the noise components at all. This generally results in a better transposition quality of acoustic instrument sounds. For example, while shifting a pitch of a flute sample, the sinusoidal components are transposed, while the noise components, associated mostly with a non-resonant air flow, are not. However, in certain cases the lack of noise component transposition might lead to shift in masking correlations between sinusoidal and noise components, which may in turn lead to unmasking of the underlying noise patterns. This effect is reduced by attenuating the noise proportionally to the absolute pitch change.

4.4. Transient component synthesis

The soundbank encoding tools do not use step transient encoding, although Meixner transients are widely used in the soundbank, especially in the case of percussive sounds. Sinusoidal components comprising Meixner transients are decoded in the time domain using optimized processing routines. Pitch-shift and amplitude change are applied initially to the de-quantized sinusoidal parameters. In order to keep the processing complexity low, per frame only one transient component can be synthesized. The selection is based on a peak estimate of temporal envelopes. The selected transient amplitude is eventually adjusted reflecting the energy of transient components not used for synthesis.

5. Conclusions

The parametric audio decoder and synthesizer presented in this paper provides for a number of features making it an attractive solution for a variety of mobile audio applications. It provides for a good to high quality stereo audio playback at 24 kbps stereo. An independent pitch and tempo alteration capability make it suitable for creative audio applications. Furthermore, the parametric audio representation allows for minimized storage and transmission requirements and for efficient post-processing. The synthesizer features an ultra light soundbank providing for high-quality music synthesis capabilities.

The parametric audio decoder and synthesizer have been implemented in a single software stack. Depending on the input provided, the library instantiates an appropriate configuration of synthesizer and/or decoder objects as necessary. This allows for optimum resource allocation on a target device. For instance, a MIDI parser is only instantiated while synthesizing MIDI content and decorrelation related parameter and PCM buffers are only instantiated while playing back parametric audio streams provided with parametric stereo data.

The integrated parametric audio based decoder/synthesizer has been efficiently implemented on ARM7 and ARM9 processors. Its features, such as low

CPU and RAM footprint combined with good quality audio output along with creativity capabilities, fit very well to the high demands of nowadays mobile applications, portable gaming consoles, etc. The CPU usage and memory footprint is given below.

	Audio decoder with 64 polyphony synthesizer	Audio decoder (w/o synthesizer)
Average MCPS (ARM9)	35 MHz	30 MHz
Average MCPS (ARM7+DSP)	12+18 MHz	5+18 MHz
RAM	46 KB	44 KB
Data ROM	153 KB	48 KB
Program ROM	77 KB	64 KB

The audio engine has been extensively tested using an experimental online streaming service. Low bandwidth along with efficient buffering techniques allowed for seamless audio playback even in densely populated urban areas and during high speed travel.

References

1. Brandenburg K. (1999), *MP3 and AAC Explained*, proceedings of the AES 17th International Conference on High Quality Audio Coding, pp. 139–146, Florence, Italy.
2. den Brinker A.C., Schuijers E.G., Oomen A.W.J. (2002), *Parametric Coding for High-Quality Audio*, Proceedings of the AES 112th Convention, Munich, Germany, preprint 5554.
3. Cormen T.H., Leiserson C.E., Rivest R.L., Stein C. (2001), *Introduction to Algorithms*, pp. 385–392, 2nd Edition, MIT Press and McGraw-Hill.
4. EBU-UER (2003), *EBU subjective listening tests on low-bitrate audio codecs*, EBU Listening Tests, EBU, Tech 3296.
5. Herre J., Johnston J.D. (1996), *Enhancing the Performance of Perceptual Audio Coders by Using Temporal Noise Shaping*, proceedings of the AES 101st Convention, Los Angeles, USA, preprint 4384.
6. Horner A., Beauchamp J., Haken L. (1993), *Methods for Multiple Wavetable Synthesis of Musical Instrument Tones*, Journal of Audio Engineering Society, **41**, 5, 336–356.
7. ISO/IEC (2004), *Information technology – Generic coding of moving pictures and associated audio information – Part 7: Advanced Audio Coding (AAC)*, ISO/IEC 13818-7:2004.
8. The ITU Radiocommunication Assembly (2003), *Method for the subjective assessment of intermediate quality level of coding systems*, Recommendation ITU-R BS.1534-1.
9. MMA Technical Standards Board (2000), Technical Note – *Downloadable Sounds Level 2.1 Specification (RP-025/Amd1)*, AMEI MIDI Committee.
10. MPEG (2003), *Parametric Coding for High Quality Audio*, ISO/IEC 14496-3:2001/FDAM JTC/SC29/WG11/N6130, December 2003, Hawaii.

11. MPEG (2003), *Report on the Verification Tests of MPEG-4 High Efficiency AAC*, ISO/IEC JTC 1/SC 29/WG 11N6009, October 2003, Brisbane, Australia.

12. MPEG (2004), *Report on the Verification Tests of MPEG-4 Parametric Coding for High Quality Audio*, ISO/IEC JTC 1/SC 29/WG 11N6675, July 2004, Redmond, US.

13. MPEG (2005), *Listening test report on MPEG-4 High Efficiency AAC v2*, ISO/IEC JTC 1/SC 29/WG 11N7137, April 2005, Busan, Korea.

14. Myburg F.P. (2004), *Design of a Scalable Parametric Audio Coder*, Ph.D. Thesis, Technische Universiteit Eindhoven.

15. Schuijers E., Breebaart J., Purnhagen H., Engdegård J. (2004), *Low complexity parametric stereo coding*, Proceedings of the AES 116th Convention, Berlin, Germany, preprint 6073.

16. Schuijers E.G.P., Oomen A.W.J., den Brinker A.C., Breebaart D.J. (2003), *Advances in Parametric Coding for High-Quality Audio*, Proceedings of the AES 114th Convention, Amsterdam, The Netherlands, preprint 5852.

17. Smith J.O., Abel J.S. (1999), *Bark and ERB bi-linear transform*, IEEE Trans. Speech Audio Processing, **7**, 697–708.

18. Szczerba M., Oomen W., Klein Middelink M. (2004), *Parametric Audio Coding Based Wavetable Synthesis*, Proceedings of the 116th AES Convention, Berlin, Germany, preprint 6063.

19. Szczerba M.Z., Schuijers E.G.P., Dillen P.H.A. (2008), *Low complexity parametric stereo decoder*, Patent Application WO/2008/096313.

20. Voitshchuk V., den Brinker A.C., van Eijndhoven S.J.L. (2001), *Alternatives for Warped Linear Predictors*, Proceedings of the 12th ProRISC Workshop, pp. 29–30, Veldhoven, The Netherlands.