Parametric Audio Coding in MPEG-4

Heiko Purnhagen

Laboratorium für Informationstechnologie
University of Hannover

NTNU, Trondheim – April 25, 2001
The MPEG-4 Audio Standard

What is MPEG-4 Audio?

- ISO/IEC Standard 14496-3
  - efficient coding of speech and audio objects
- Tools: speech and audio coding (2 .. 64+ kbit/s/ch)
- Additional functionalities
  - bitrate scalability (embedded coding)
  - error robustness
  - natural and synthetic content
  - combination of multiple objects in a scene
The MPEG-4 Audio Standard: Decoder

MPEG-4 Systems

Demux

MPEG-4 Audio

Decoder

Natural Speech

Natural Audio

Synthetic Speech

Synthetic Audio

Effects & Mixing

Scene Description

MPEG-4 Coded Data

Audio Scene
The MPEG-4 Audio Standard: Tools

Audio Tools: Coding of natural objects

- **Speech:**
  - Original (4 kHz BW) 2 kbit/s
  - HVXC (parametric, 2 .. 4 kbit/s)
  - CELP (NB + WB, 4 .. 24 kbit/s)

- **Audio:**
  - Original (Stereo) 48 kbit/s
  - TwinVQ (6 .. 16 kbit/s/ch)
  - AAC-scalable (16 .. 64+ kbit/s/ch)
  - BSAC (fine-step scalability)
  - LowDelay (20 ms delay)
  - HILN (parametric, 4 .. 16 kbit/s)
The MPEG-4 Audio Standard: Tools

Audio Tools: Coding of synthetic objects
- Speech: TTS Interface
- Audio: “Structured Audio” (DSP) incl. MIDI

Systems Tools: Composition of audio objects
- Mixing of audio objects
- Effects: DSP blocks from “Structured Audio”
- 3D Audio: “Environmental Spatialisation”
3D Audio: “Environmental Spatialisation”

- Physical approach: description of acoustical properties of environment (room geometry, sound source position, ...) ⇒ corresponding audio and visual scene

- Perceptual approach: high-level perceptual description of “audio scene” (room reverberance, source presence, ...) ⇒ audio and visual scene independent
Fundamentals: Speech Coding (CELP)

Pitch Period

Voiced/Unvoiced

Gain

Vocal Tract Parameters

Impulse Train Generator

Random Noise Generator

Time-varying Digital Filter

Speech Signal

Simplified model of speech generation / synthesis
Fundamentals: Speech Coding (CELP)

Basic structure of a CELP decoder
Fundamentals: Speech Coding (CELP)

Basic structure of a CELP encoder
Fundamentals: Audio Coding (MPEG-1/2)

Basic structure of a perceptual audio encoder / decoder (T/F coder: time/frequency decomposition)
Perception: Simultaneous Masking

threshold in quiet and masked threshold

- Excitation
- Spectral component
- Masked threshold
- Threshold in quiet
- Maximum allowable distortion

Sound pressure level [dB]

Frequency [kHz]

0 20 40 60

- 0
- 10
- 20
- 40
- 60
- 80
Perception: Simultaneous Masking

- Sound pressure level [dB]
- Frequency [kHz]
- Distortion
- Excitation
- Masked threshold
- Spectral component

Original signal
Perception: Simultaneous Masking

sound pressure level [dB]

1 2 3 4

0 20 40 60

excitation

· · · spectral component

masked threshold

threshold in quiet

original signal + quantisation noise
Perception: Simultaneous Masking

Quantisation noise only

Sound pressure level [dB]

Threshold in quiet

Frequency [kHz]

Distortion

Spectral component
What is “Parameteric Audio Coding”?

… so what is “Parameteric Audio Coding” about?

- **Problem:**
  MPEG-4 range of applications
  \( \implies \)
  requires speech and audio coders

- **Idea:**
  generalised approach to audio coding
What is "Parameteric Audio Coding"?

Representation of audio signal $x$

- **physical**: waveform $x(t)$
- **abstract**: musical score (compact, ambiguous)  
  $\Rightarrow$ promising approach for efficient audio coding

Audio coding with abstract signal representation

- Encoding: **physical** $\Rightarrow$ **abstract representation**
  but: automatic transcription to score very difficult !!!
  $\Rightarrow$ **Compact representation of audio**
  automatically derived from real-world signal
What is “Parameteric Audio Coding”?

**Signal representation:** source and perception model

⇒ **Parametric Audio Coding**

- source model ⇒ redundancy reduction
- perception model ⇒ irrelevancy reduction
Source Models for Audio Signals

Spectral Decomposition

- stationary signal within a frame (duration $T$)  
  $\Rightarrow$ time-to-frequency (T/F) transform (e.g. MDCT)
- signal-adaptive time/frequency resolution

Physical Modelling: Excitation + Resonances

- speech: periodic/random excitation + LPC filter
- music synthesis: e.g. waveguide
Source Models for Audio Signals

Sinusoidal Modelling

\[ \hat{x}(t) = \sum_{i=1}^{N} a_i(t) \cdot \sin(\varphi_i + 2\pi \int_{0}^{t} f_i(\tau) \, d\tau) \]

- Applications:
  - music instrument analysis/synthesis
  - speech & audio coding
- modelling of “spectral peaks”
- tracking of trajectories / phase continuity
- phase \( \varphi_i \) often perceptually irrelevant
Source Models for Audio Signals

Transient Modelling

- sinusoids with amplitude envelope (attack & decay)
- sinusoidal modelling of DCT spectrum
- T/F coding / wavelets / “matching pursuit”

Noise Modelling

- subband noise models (Bark, ERB)
- MA model: DCT of noise spectrum
- AR model: white noise + LPC filter
- “Bark-warped” LPC
Source Models for Audio Signals

Extended Sinusoidal Models

• set of sinusoids with common fundamental frequency ⇒ harmonic tone

• “bandwidth enhanced sinusoids”: sinusoid ⇒ narrow-band noise (using AM/FM)

⇒ Problem: Choice of source model?

Efficiency vs. Generality

specialised source model not suitable for arbitrary signals
Parametric Audio Coding

Concept of Parametric Audio Coding

- combination of different source models
  ⇒ decompose audio signal into components

- utilise perception models
  ⇒ “optimal” decomposition (relevant components)

⇒ Analysis/Synthesis Approach

Parameter quantisation and coding

- quant. step size: “just noticeable differences”

- parameter prediction & entropy coding
Parametric Audio Coding: Encoder

Extracted Signal Components

Model Parameters

Synthesis

Parameter Estimation

Coding

Source Model 1

Percept. Model 1

Source Model N

Percept. Model N

Accu

Residual

Frame Buffer

Audio Buffer

Decomposition Control

Joint Model

Buffer & Mux

Bit Stream
**Parametric Audio Coding: Encoder**

**Example:** Decomposition into sinusoidal components.

\[
x(t) \rightarrow s(t) \rightarrow \log |\text{DFT}| \rightarrow \text{argmax} \rightarrow f_{c,i} \rightarrow f_i, a_i, \varphi_i
\]

\[
M(f) \rightarrow \text{parametric psychoacoustic model} \rightarrow \text{frequency, amplitude, and phase estimation}
\]

**Analysis/Synthesis Loop**
**Parametric Audio Coding: Decoder**

**Parameter Decoding & Signal Synthesis**

![Diagram showing the process of parameter decoding and signal synthesis.]

**Additional functionalities**

- **scalability**: base + enhancement bitstream
- **signal modification**: time-scaling & pitch-shifting
Comparison: Speech and Audio Coders

Audio Coder (e.g. MPEG-1/2)
- perception model for encoder control
  ⇒ efficient for arbitrary signals (≥ 32 kbit/s/ch)

Speech Coder (e.g. CELP)
- specialised source model (vocal tract)
  ⇒ efficient for speech signals (4 .. 24 kbit/s)

Parametric Coder (e.g. HVXC, HILN)
- recreate perceived sound
  ⇒ no waveform approximation required
Parametric Audio Coding: HILN

Example: MPEG-4 Parametric Audio Coder HILN
“Harmonic and Individual Lines plus Noise”

Models and parameters in HILN:

- **harmonic tone**: fundamental freq. & LPC spectrum
- **sinusoids**: frequency & amplitude
  [opt.: ampl. envelope, start phase]
- **noise**: LPC spectrum

- frame size 32 ms (typ.)
  ⇒ 4 .. 16 kbit/s @ 8 kHz bandwidth (typ.)
HILN Encoder

HILN Parametric Audio Encoder
(selection of relevant components by perception model)
HILN Decoder: Audio Demonstration

Example: MPEG-4 HILN @ 6 kbit/s, \( f_s = 16 \text{ kHz} \)

HILN Parametric Audio Decoder
HILN Decoder: Audio Demonstration

Signal modification (HILN, $f_s = 16$ kHz)

- Interactive pitch and speed control (16 kbit/s)
- Pitch-shifting: +20% (6 kbit/s)
- Time-scaling: -17% (6 kbit/s)

Bitrate scalability (HILN, $f_s = 16$ kHz)

- base layer: 6 kbit/s
- base + enhancement layer: 6+10 kbit/s
- non-scalable bitstream: 16 kbit/s
HILN Decoder: Audio Demonstration

Comparison of coding techniques: (6 kbit/s)

- original (8 kHz bandwidth)
- speech coding (MPEG-4 CELP)
- T/F coding (MPEG-4 TwinVQ)
- parametric audio coding (MPEG-4 HILN)
HILN Parametric Audio Coding

MPEG-4 Verification Test

- MPEG-4 TwinVQ and HILN comparable at 6 kbit/s
- MPEG-4 AAC and HILN comparable at 16 kbit/s
- Additional functionalities of HILN: bitrate scalability, speed & pitch change

Why to “speed up HILN encoding?”

- Reference encoder:
  - only optimised for audio quality
  - very high computational complexity (not real-time)
Goal: Real-time HILN encoding on a normal PC

Possible approaches to reduce complexity:

- Sinusoid extraction by matching pursuit [Goodwin 1997, Verma 1999]

- Fast sinusoid extraction in the frequency domain
  - correlation of spectrum with “prototype”
  - stepwise refined frequency search
  - energy metric $\Rightarrow$ needs psychoacoustic reordering

- Fast HILN encoder implemented in frequency domain (optional amplitude envelope supported)
HILN Encoder Optimisation

Block diagram of fast HILN encoder
HILN Encoder Optimisation

Fast sinusoid extraction in the frequency domain
Results: Computational Complexity

<table>
<thead>
<tr>
<th>Encoder</th>
<th>Bitrate [kbit/s]</th>
<th>CPU Load [MHz]</th>
<th>Rel. Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference Encoder</td>
<td>6</td>
<td>26000</td>
<td>1</td>
</tr>
<tr>
<td>Fast Encoder (Env.)</td>
<td>6</td>
<td>51</td>
<td>510</td>
</tr>
<tr>
<td>Fast Encoder</td>
<td>6</td>
<td>24</td>
<td>1080</td>
</tr>
<tr>
<td>Reference Encoder</td>
<td>16</td>
<td>31000</td>
<td>1</td>
</tr>
<tr>
<td>Fast Encoder (Env.)</td>
<td>16</td>
<td>100</td>
<td>310</td>
</tr>
<tr>
<td>Fast Encoder</td>
<td>16</td>
<td>46</td>
<td>680</td>
</tr>
</tbody>
</table>

Encoding speed on Intel Pentium III (500 MHz)
ANSI-C implementation, $f_s = 16$ kHz
Results: Subjective Quality

Subjective quality of fast encoder

- Informal comparison (6 & 16 kbit/s, 39 items)
  - fast encoder with ampl. envelope
    ⇒ 10 .. 20% of items (slightly) worse than ref. enc.
  - fast encoder (no ampl. envelope)
    ⇒ percussive items (clearly) worse than fast enc.

- Current limitations of fast encoder
  - simplified or no envelope estimation
  - no sweep estimation
  - no harmonic component grouping
Results: Audio Demonstration

Audio demonstration: Comparison of encoders

<table>
<thead>
<tr>
<th>Original</th>
<th>$f_s = 16$ kHz</th>
<th>$f_s = 16$ kHz</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference Encoder</td>
<td>6 kbit/s</td>
<td>16 kbit/s</td>
</tr>
<tr>
<td>Fast Encoder (Env.)</td>
<td>6 kbit/s</td>
<td>16 kbit/s</td>
</tr>
<tr>
<td>Fast Encoder</td>
<td>6 kbit/s</td>
<td>16 kbit/s</td>
</tr>
</tbody>
</table>

Audio demonstration: Real-time enco-decoding

- real-time encoding (16 kbit/s, $f_s = 16$ kHz)
- real-time decoding with interactive pitch change
Potential Parametric Coding Artifacts

Potential artifacts related to source models:

- limitations of source models
- bad decomposition (hard decisions are problematic)
- bad parameter estimation

Potential artifacts related to perception models:

- quantisation (consider “just noticeable differences”)
- selection of most relevant components
- is phase information irrelevant? (transients, clipping in sinusoidal synthesiser)
Examples of Artifacts

- Parametric coding: no waveform approximation
  ⇒ difference signal meaningless
  - original: pop music
  - coded by parametric audio coder
  - difference signal (original-coded)

- Limitations of source models:
  model noise with sinusoids (e.g. applause)
  - original: white noise
  - coded using 0 to 120 sinusoids
Examples of Artifacts

- Limitations of source models:
  no model for transient (percussive) components
  - original: castanets
  - coded using sinusoids + noise
  - same, but with amplitude envelopes enabled
Examples of Artifacts

- Limitations of source models: specialised speech model not suitable for music
  - original: speech
  - coded by parametric speech coder
  - original: pop music
  - coded by parametric speech coder
Examples of Artifacts

- Bad signal decomposition:
  many sinusoids forced on harmonic grid
  - original: orchestral music
  - coded (harmonic component too strong)

- Bad signal decomposition:
  many tonal components modelled as noise
  - original: pop music
  - coded (noise component too strong)
Outlook

Improved source models
- “Gap” between tonal signals and noise
- Better transient models
- Combination with speech coder and T/F coder

Encoder optimisation (signal decomposition)
- Improved segmentation (e.g. tonal vs. noise)
- Grouping in time and frequency (trajectory, harmonic)
- Automatic segmentation of speech/music

\[\text{... audio objects are transparent ;} - \)\]
further reading ...

- Parametrische Audio Coding – Bibliographie
  http://www.tnt.uni-hannover.de/~purnhage/

- MPEG Audio Web Page
  (tutorials, test reports, etc.)
  http://www.tnt.uni-hannover.de/project/mpeg/audio/

- Official MPEG Home Page
  http://www.cselt.it/mpeg/