Parametrische Audiocodierung in MPEG-4

Heiko Purnhagen

Laboratorium für Informationstechnologie
Universität Hannover

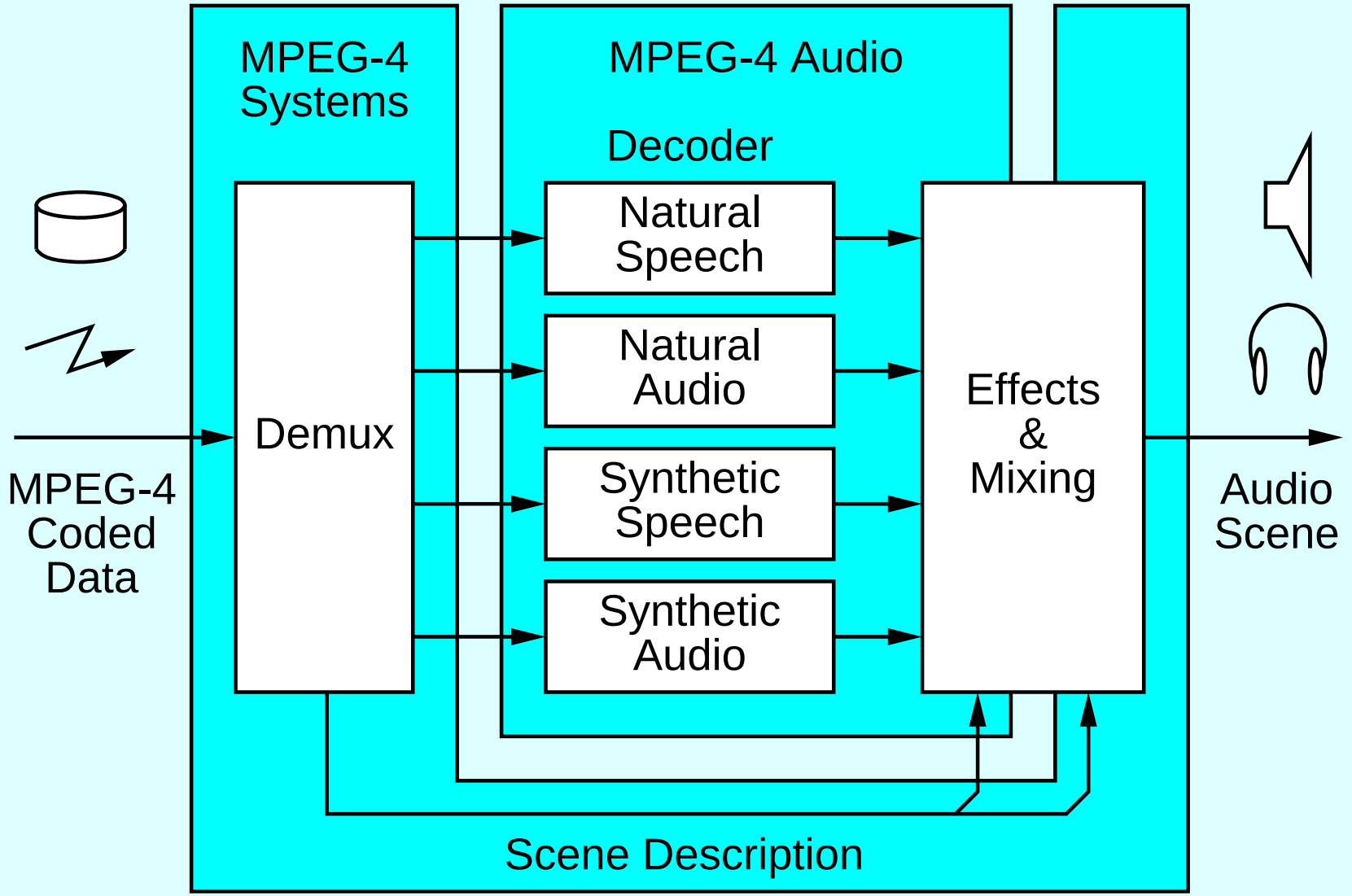
Media Event, TU Ilmenau, 12. Dez. 2000

Der MPEG-4 Audio Standard

Was ist MPEG-4 Audio?

- ISO/IEC Standard 14496-3
  $\Rightarrow$ Codierung von Multimedia-Objekten
- Tools zur Sprach- und Audiocodierung
  (2 .. 64+ kbit/s/ch)
- Zusätzliche Funktionalität
  - Datenraten-Skalierbarkeit (embedded coding)
  - Fehler-Robustheit
  - Kombination von Objekten in einer Szene
  - Natürliche und synthetische Objekte

Der MPEG-4 Audio Standard: Decoder



Der MPEG-4 Audio Standard: Tools

Audio Tools: Codierung von natürlichen Objekten

- Sprache:
  - HVXC (parametrisch, 2 .. 4 kbit/s)
  - CELP (NB + WB, 4 .. 24 kbit/s)
- Audio:
  - TwinVQ (6 .. 16 kbit/s/ch)
  - AAC-scalable (16 .. 64+ kbit/s/ch)
  - BSAC (feinstufig skalierbar)
  - LowDelay (20 ms Verzögerung)
  - HILN (parametrisch, 4 .. 16 kbit/s)

Der MPEG-4 Audio Standard: Tools

Audio Tools: Codierung von synthetischen Objekten

- Sprache: TTS-Interface
- Audio: “Structured Audio” (DSP) incl. MIDI

Systems Tools: Mischen von Audio-Objekten

- Mischen von Audio-Objekten
- Effekte: DSP-Blöcke aus “Structured Audio”
- 3D Audio: “Environmental Spatialisation”

Der MPEG-4 Audio Standard: Tools

3D Audio: “Environmental Spatialisation”

- physikalischer Ansatz:
  Beschreibung der akustischen Eigenschaften
  (Raum-Geometrie, Position der Schallquelle, ...)
  $\Rightarrow$ Audio- und Video-Szene korrespondieren
- wahrnehmungsorientierter Ansatz:
  abstrakte, wahrnehmungsorientierte Beschreibung
  (Raumhall, Präsenz der Schallquelle, ...)
  $\Rightarrow$ Audio- und Video-Szene unabhängig

Motivation

. . . und was ist parametrische Audiocodierung ?

- **Problem:**
  bei MPEG-4 Sprach- und Audiocoder erforderlich
- **Idee:**
  verallgemeinerter Ansatz zur Audiocodierung

Motivation

Darstellung eines Audiosignals x

- *physikalisch*: Wellenform $x(t)$
- *abstrakt*: Partitur (kompakt, mehrdeutig)
  $\Rightarrow$ vielversprechender Ansatz für Audiocodierung

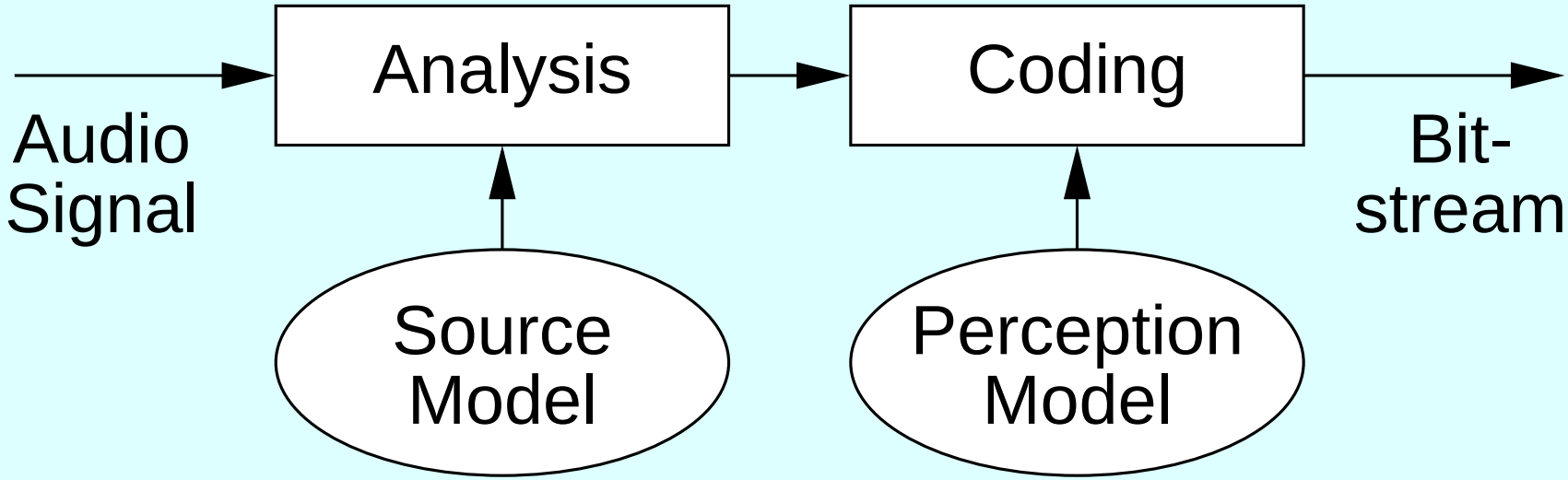
Audiocodierung mit abstrakter Signaldarstellung

- Encodierung: *physikalische* $\Rightarrow$ *abstrakte Darstellung*
  aber: automatische Transkription sehr schwierig !!!
- $\Rightarrow$ *kompakte Darstellung eines Audiosignals automatisch aus realem Signal erzeugt*

Motivation

Signaldarstellung: Quellenmodell + Modellparameter

$\Rightarrow$ **Parametrische Audiocodierung**



- Quellenmodell $\Rightarrow$ Redundanz-Reduktion
- Wahrnehmungsmodell $\Rightarrow$ Irrelevanz-Reduktion

Quellenmodelle für Audiosignale

Spektrale Zerlegung

- stationäres Signal innerhalb Zeitrahmen (Dauer T)
  $\Rightarrow$ Zeit/Frequenz-Transformation ("T/F", z.B. MDCT)
- signaladaptive Zeit/Frequenz-Auflösung

Physikalische Modellierung: Erregung + Resonanz

- Sprache: periodische/zufällige Erregung + LPC
- Musiksynthese: z.B. Saite/Rohr als "Wellenleiter"

Quellenmodelle für Audiosignale

Sinusoide Modellierung

$$\hat{x}(t) = \sum_{i=1}^{N} a_i(t) \cdot \sin(\varphi_i + 2\pi \int_0^t f_i(\tau)\, d\tau)$$

- Anwendungen:
  - Musikinstrument-Analyse/Synthese
  - Sprach- und Audiocodierung
- Nachbildung spektraler “Spitzen”
- Verfolgung von Trajektorien / Phasenkontinuität
- Phase φ_i oft nicht wahrnehmungsrelevant

Quellenmodelle für Audiosignale

Transienten-Modelle

- Sinustöne mit Amplituden-Hüllkurve (attack & decay)
- sinusoide Modellierung des DCT-Spektrums
- T/F-Codierung / Wavelet / “matching pursuit”

Rausch-Modelle

- Teilband-Rauschmodelle (Bark, ERB)
- MA-Modell: DCT des Rauschspektrums
- AR-Modell: weißes Rauschen + LPC-Filter
- “Bark-warped” LPC

Quellenmodelle für Audiosignale

Erweiterte sinusoide Modelle

- Sinustöne mit gemeinsamer Grundfrequenz
  $\Rightarrow$ harmonischer Ton
- “bandwidth enhanced sinusoids”:
  Sinuston $\Rightarrow$ Schmalbandrauschen (AM/FM-Mod.)

$\Rightarrow$ **Problem: Wahl des Quellenmodells ?**

Effizienz vs. Generalität

spezialisierte Quellenmodelle
nicht für beliegige Signale geeignet

Parametrische Audiocodierung

Konzept der Parametrische Audiocodierung

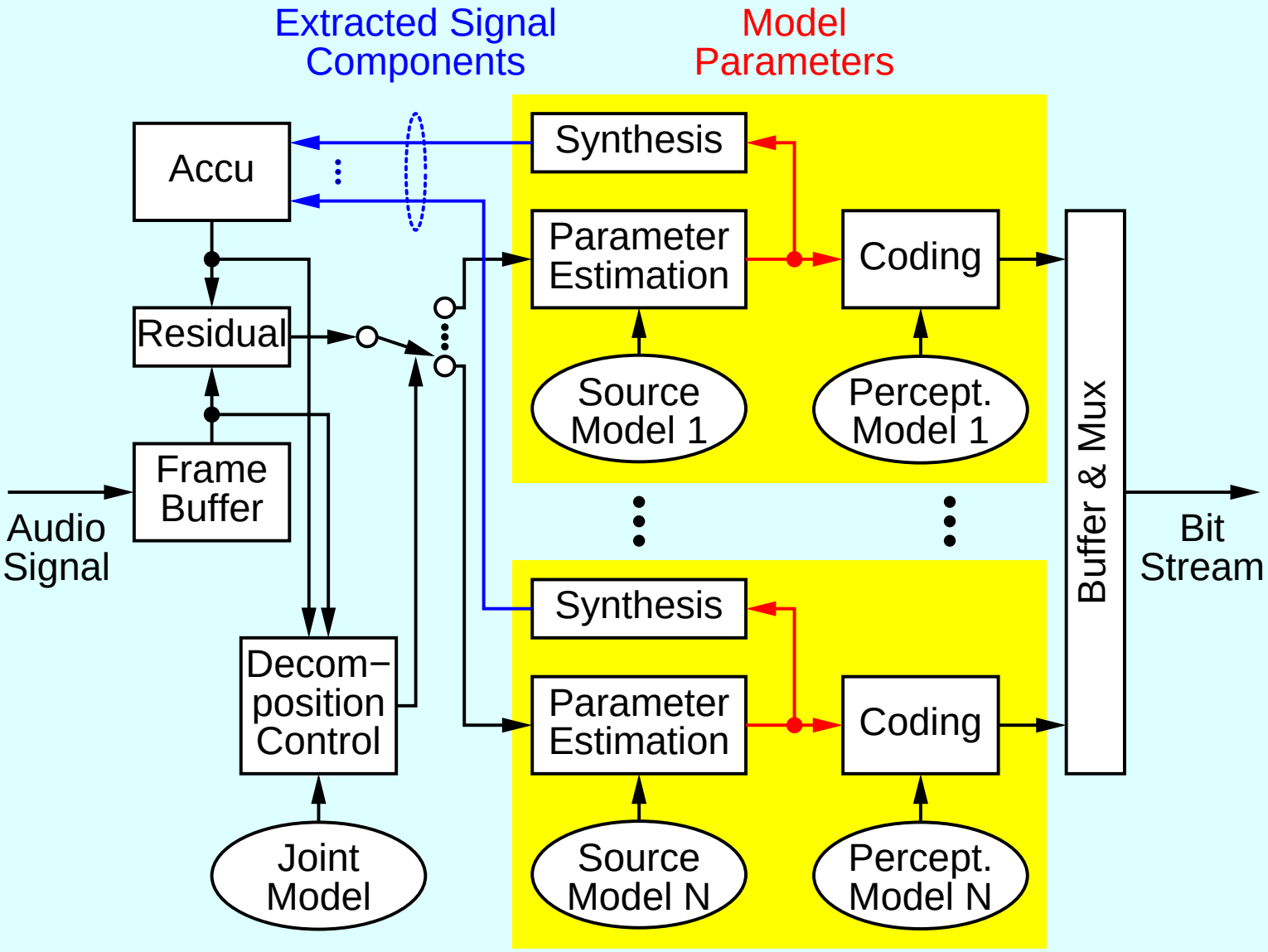
- Kombination von verschiedenen *Quellenmodellen*
  $\Rightarrow$ Zerlegung des Audiosignals in Komponenten
- Berücksichtigung von *Wahrnehmungsmodellen*
  $\Rightarrow$ “optimale” Zerlegung (wichtigste Komponenten)

$\Rightarrow$ **Analyse/Synthese-Ansatz**

Parameter-Quantisierung und Codierung

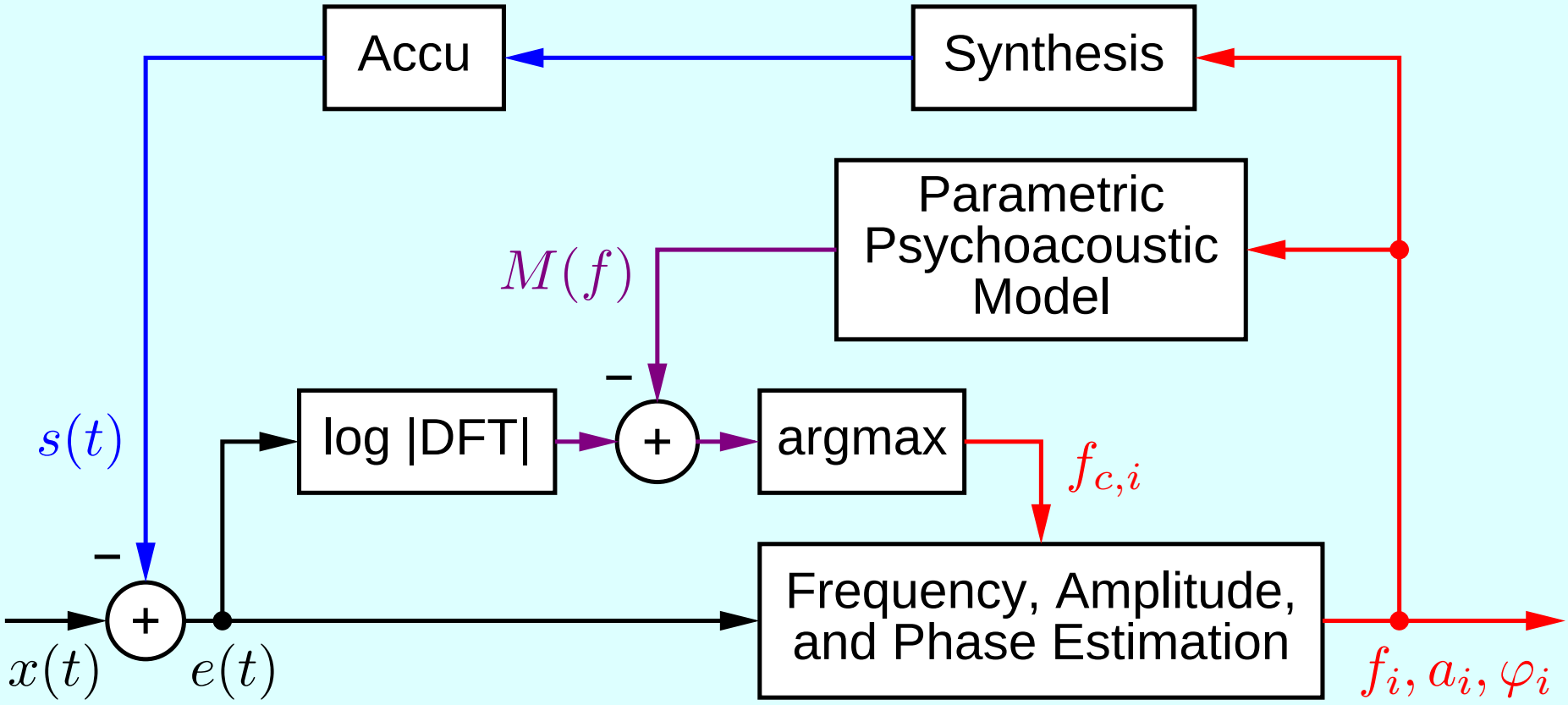
- Quant.-Stufengröße: “just noticeable differences”
- Parameter-Prädiktion & Entropiecodierung

Parametrischer Audiocodierung: Encoder



Parametrischer Audiocodierung: Encoder

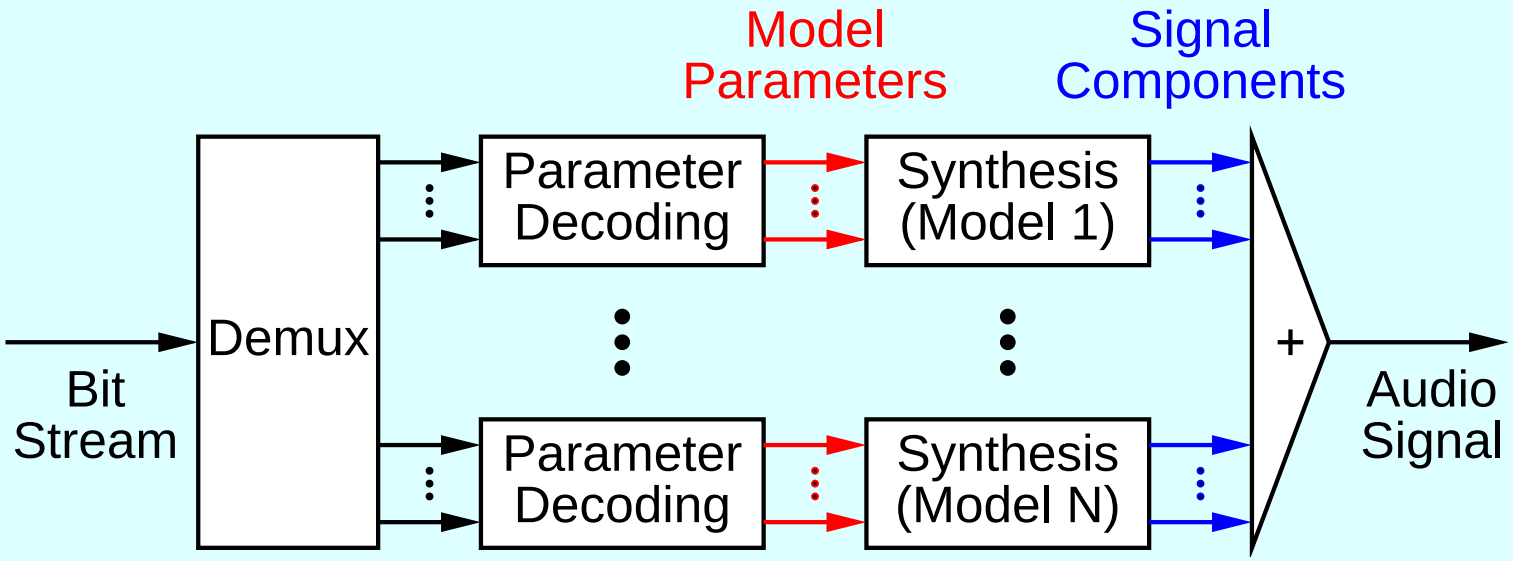
Beispiel: Zerlegung in sinusoide Komponenten



Analyse/Synthese-Schleife

Parametrischer Audiocodierung: Decoder

Parameterdecodierung & Signalsynthese



Zusätzliche Funktionalitäten

- Skalierbarkeit: Basis- + Enhancement-Bitstrom
- Signal-Modifikation: Geschwindigkeit & Tonhöhe

Vergleich: Sprach- und Audiocoder

Audiocoder (z.B. MPEG-1/2)

- Wahrnehmungsmodell zur Codersteuerung
  $\Rightarrow$ effizient für beliebige Signale ($\geq$ 32 kbit/s/ch)

Sprachcoder (z.B. CELP)

- spezialisiertes Quellenmodell (Vokaltrakt)
  $\Rightarrow$ effizient für Sprachsignale (4 .. 24 kbit/s)

Parametrische Coder

- Nachbildung des Klang-Eindrucks
  $\Rightarrow$ keine Nachbildung der Wellenform erforderlich

Parametrische Audiocodierung: Beispiel

Beispiel: MPEG-4 Parametric Audio Coder HILN

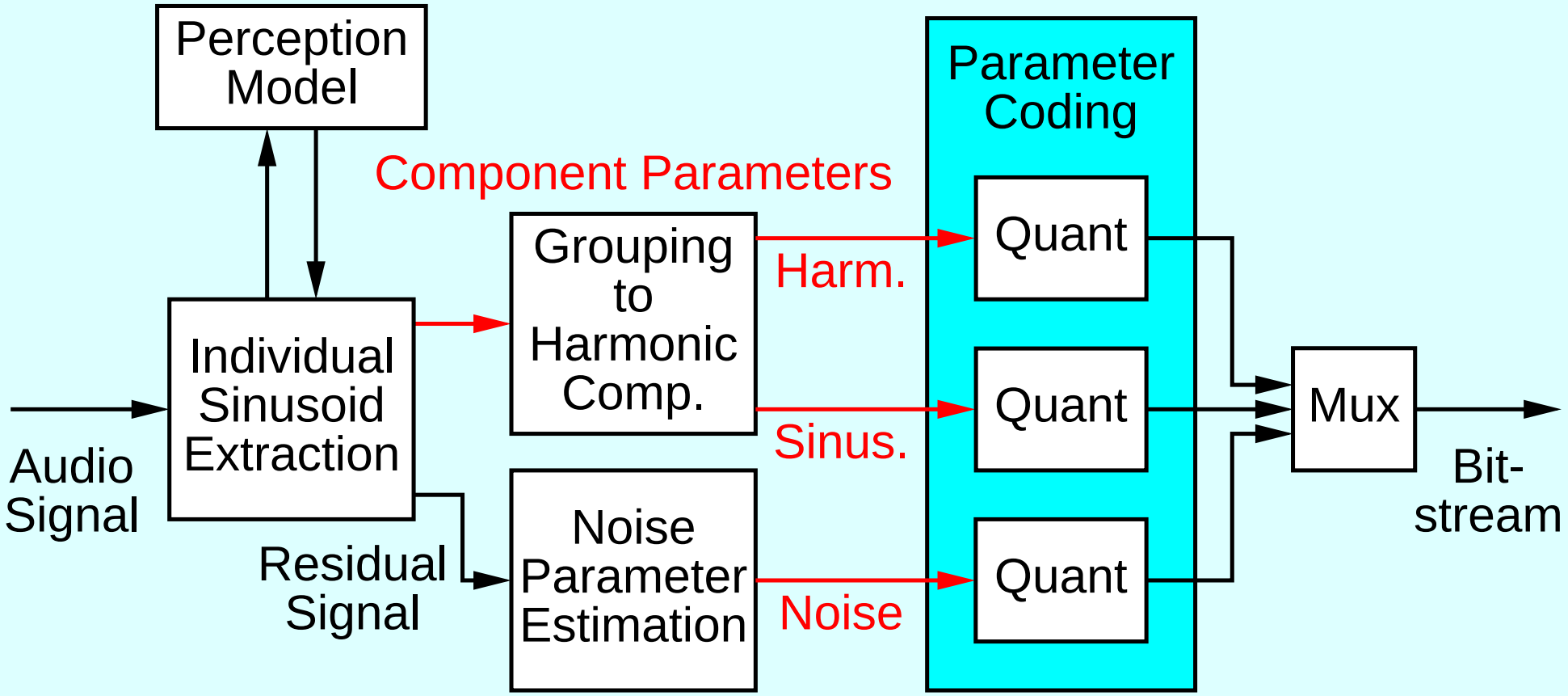
"Harmonic and Individual Lines plus Noise"

Modelle und Parameter in HILN:

harmon. Ton:	Grundfrequenz & LPC-Spektrum
Sinustöne:	Frequenz & Amplitude [opt.: Amplitudenhüllkurve, Startphase]
Rauschen:	LPC-Spektrum

- Zeitrahmen 32 ms (typ.)
  $\Rightarrow$ 4 .. 16 kbit/s @ 8 kHz Bandbreite (typ.)

HILN Encoder

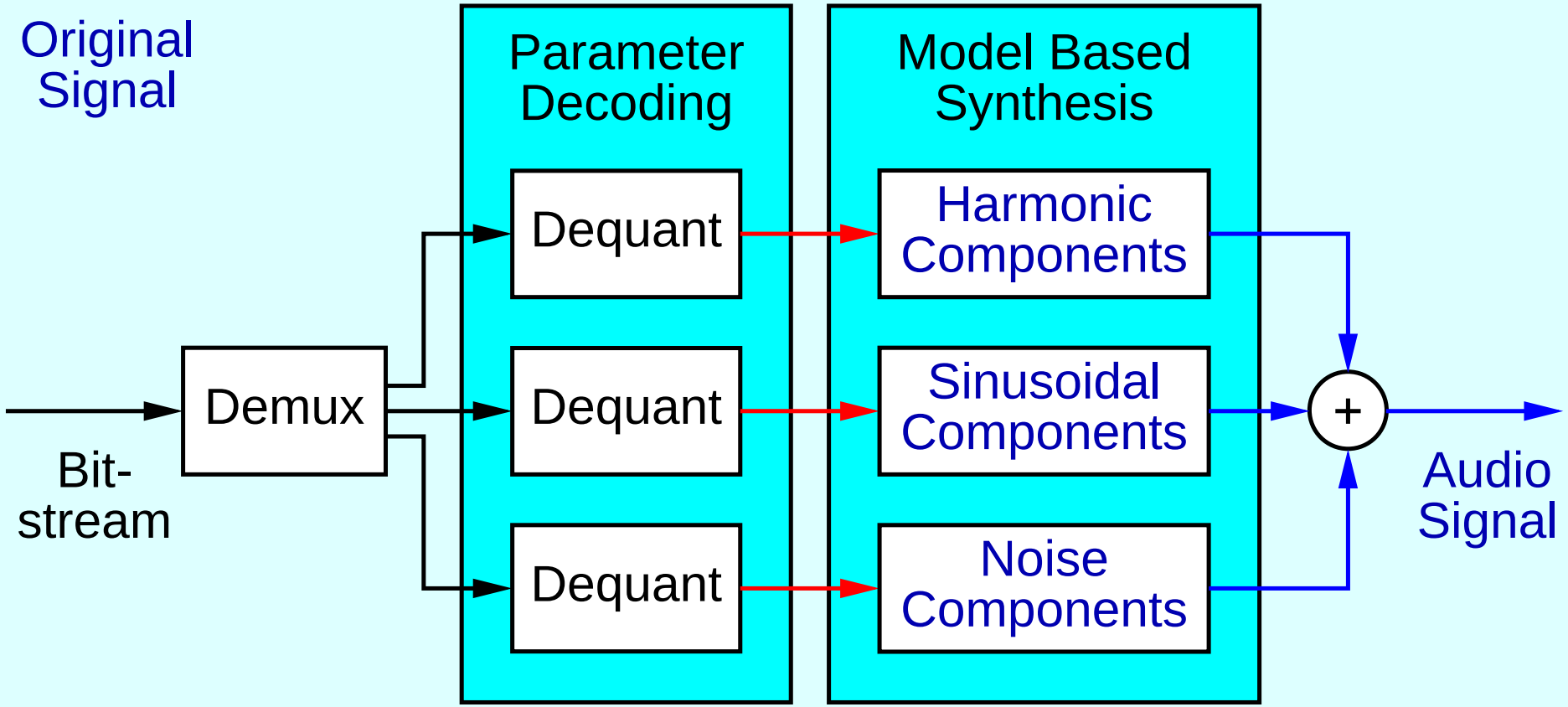


Blockdiagramm eines HILN-Encoders

(Wahrnehmungsmodell steuert Komponenten-Auswahl)

HILN Decoder: Audiobeispiele

Beispiel: MPEG-4 HILN @ 6 kbit/s (f_s = 16 kHz)



Blockdiagramm des HILN-Decoders

HILN Decoder: Audiobeispiele

Signal-Modifikation (HILN 16 kbit/s, f_s = 16 kHz)

- Echtzeit-Steuerung: Tonhöhe und Geschwindigkeit

Datenraten-Skalierbarkeit (HILN f_s = 16 kHz)

- Basis-Bitstrom: 6 kbit/s
- Basis- + Enhancement-Bitstrom: 6+10 kbit/s
- nicht-skalierbarer Bitstrom: 16 kbit/s

HILN Decoder: Audiobeispiele

Vergleich von Codierungsverfahren (6 kbit/s)

- Original (8 kHz Bandbreite)
- Sprachcoder (MPEG-4 CELP)
- Transformationscoder (MPEG-4 TwinVQ)
- Parametrischer Audiocoder (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 functionality 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)

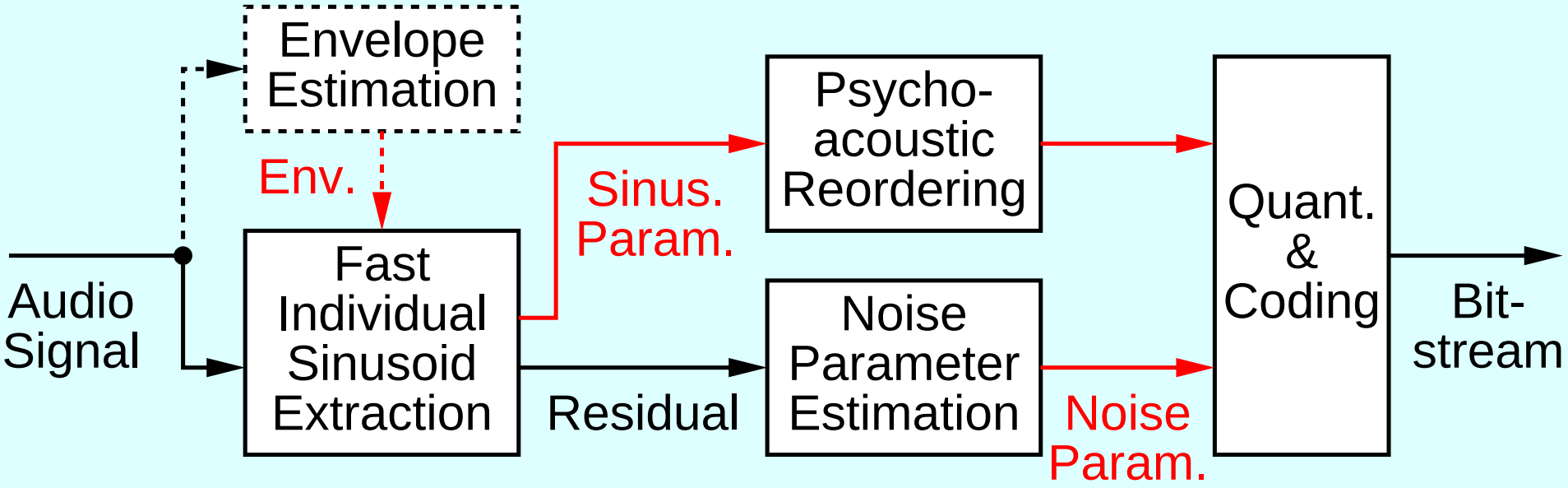
HILN Encoder Optimisation

Goal: real-time HILN encoding on a normal PC

Possible approaches to reduce complexity:

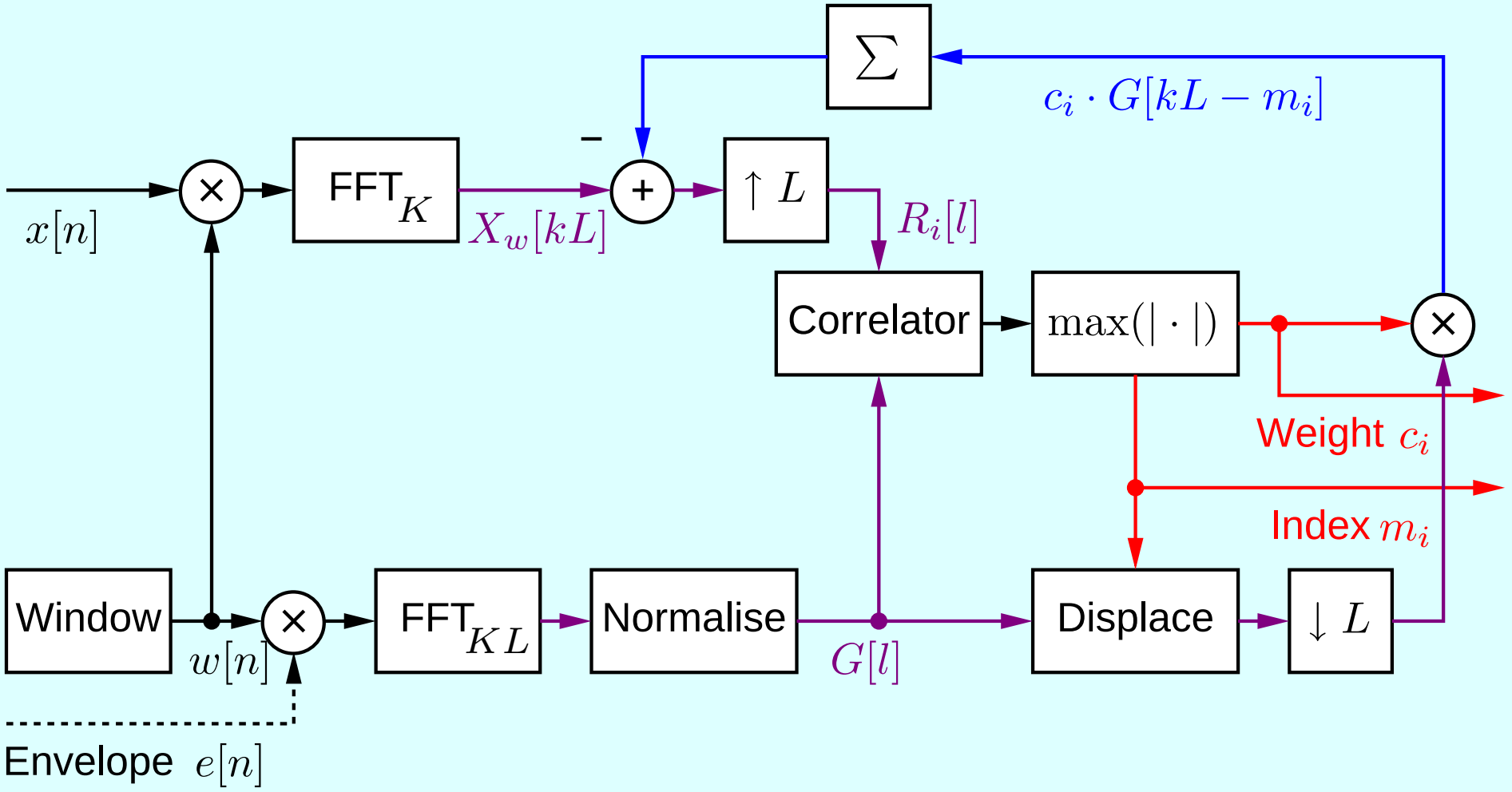
- Sinusoid extraction by matching pursuit [Goodwin 1997, Verma 1999]
- Fast sinusoid extraction in the frequency domain
  - correlation of spectrum with “prototype”
  - stepwise refined frequency search
  - energy metric $\Rightarrow$ needs psychoacoustic reordering
- Fast HILN encoder implemented in 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

Encoder	Bitrate [kbit/s]	CPU Load [MHz]	Rel. Speed
Reference Encoder	6	26000	1
Fast Encoder (Env.)	6	51	510
Fast Encoder	6	24	1080
Reference Encoder	16	31000	1
Fast Encoder (Env.)	16	100	310
Fast Encoder	16	46	680

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
    $\Rightarrow$ 10 .. 20% of items (slightly) worse than ref. enc.
  - fast encoder (no ampl. envelope)
    $\Rightarrow$ 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

Original	f_s = 16 kHz	f_s = 16 kHz
Reference Encoder	6 kbit/s	16 kbit/s
Fast Encoder (Env.)	6 kbit/s	16 kbit/s
Fast Encoder	6 kbit/s	16 kbit/s

Audio demonstration: real-time encoding+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
  $\Rightarrow$ 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)

Ausblick

Verbesserung der Quellenmodelle

- “Loch” zwischen Rauschen und tonalen Signalen
- bessere Transienten-Modelle
- Kombination mit Sprachcoder und T/F-Coder

Optimierung des Encoders (Signal-Zerlegung)

- Trennung von tonalen/rauschartigen Komponenten
- Zeit- und Frequenz-Gruppierung (Trajekt. / Harmon.)
- automatische Segmentierung von Sprache/Musik

. . . Audio-Objekte sind transparent ;-)

Literatur

- Parametrische Audiocodierung – 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/