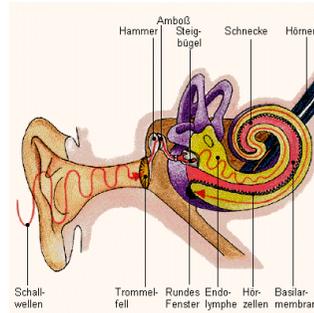


Audio: Agenda

- Schall und Schallausbreitung
 - Aufbau des Gehörs
 - Hörwahrnehmung
 - Sprache, Vokalartikulation
-
- Signalverarbeitung
 - Abtastung, Abtasttheorem
 - A/D- und D/A-Wandlung
 - Filter, elementare Algorithmen
-
- Kodierung von Audiodaten
 - psychoakustische Verfahren, MP3



Medientechnik | WS 2001 | 18.204

Audio: Literatur

K.v.d.Heide, Skripte zu "Signalverarbeitung" und "Nachrichtentechnik", FB Informatik, Uni HH, tech-www.informatik.uni-hamburg.de/lehre/ (Matlab)

Mathworks, Inc., Matlab 5.3 User and Toolbox Manuals, www.matlab.com

U. Zölzer, Digitale Audiosignalverarbeitung, Teubner 1996

P.Gerdsen, P.Kröger, Digitale Signalverarbeitung in der Nachrichtenübertragung, Springer 1997

R.W.Hamming, Digital Filters, Prentice Hall, 1983

W.H.Press, B.P.Flannery, S.A.Teukolsky, W.T. Vetterling, Numerical Recipes, Cambridge Univ. Press

U. Tietze, Ch. Schenk, Halbleiter-Schaltungstechnik, Springer 1993 (analoge Schaltungen)

IEEE Journal Signal Processing

diverse Konferenzen zum Thema, Audio u.a. Audio Engineering Convention

DSP-Datenbücher (www.motorola.com, www.analog.com)

Medientechnik | WS 2001 | 18.204

Motivation

wozu die Gehörwahrnehmung untersuchen?

- Forschung / Erkenntnisgewinn
möglichst optimale Verfahren zur:
- Aufzeichnung / Speicherung / Wiedergabe von Audio
- so gut wie nötig - aber nicht besser (!)
- psychoakustische Verfahren, etwa MP3-Kompression
- bessere Algorithmen für Musik und Musikproduktion
- Raumsimulation für Spiele oder virtuelle Realität
- möglichst effiziente (low-level) Sprachverarbeitung
- medizinische Anwendungen, vor allem Hörgeräte

Medientechnik | WS 2001 | 18.204

Schall

physikalischer Schallbegriff:

- alle elastischen Schwingungen eines Materials
- im allg. Fall longitudinal und transversal
- aber in Gasen (Luft) nur longitudinal (=Druckschwingungen)

• Schallgeschwindigkeit	ca. 340 m/s in Luft	
• Schallintensität	lokale Druckänderung	
Luftdruck auf Meereshöhe	1 bar	= 10.000 Pa
Schmerzschwelle	1 mbar	= 10 Pa
wahrnehmbare Druckänderung	10E-9 bar	= 0.00001 Pa

- Ohr hat optimale Empfindlichkeit
- weitere Steigerung: Wahrnehmen der Molekularbewegung

Medientechnik | WS 2001 | 18.204

Schall: Leistung, dB

- Kenngrößen des Schallfelds: Druckamplitude P bzw. Intensität I
- Gesamtleistung: Integral der Intensität über eine Oberfläche
- Lautstärke (Intensität): dB

$$L / \text{dB} := 10 \log_{10}(I / I_{\text{ref}})$$

$$:= 20 \log_{10}(P / P_{\text{ref}})$$

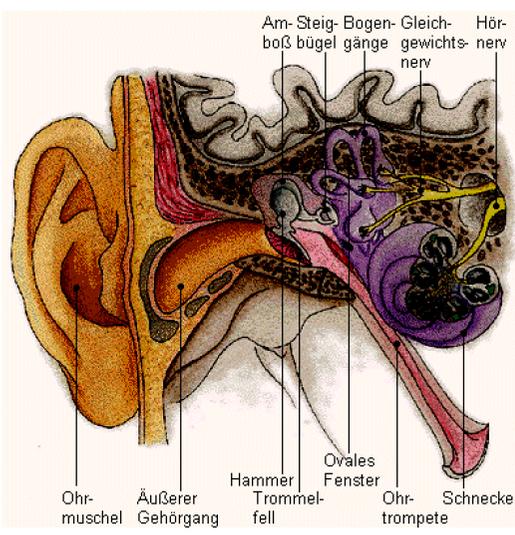
$$P_{\text{ref}} := 0.00002 \text{ Pa}$$

- alternativ: dB als Maß für das Verhältnis (p_1/p_2)

- 20 dB zehnfacher Schalldruck
- 6 dB doppelter Schalldruck
- 6 dB halber Schalldruck

Quelle	Leistung (typ.)
Sprache	10 μ W
Geige	1 mW
Lautsprecher	100 W

Ohr: Schema



drei Untereinheiten:

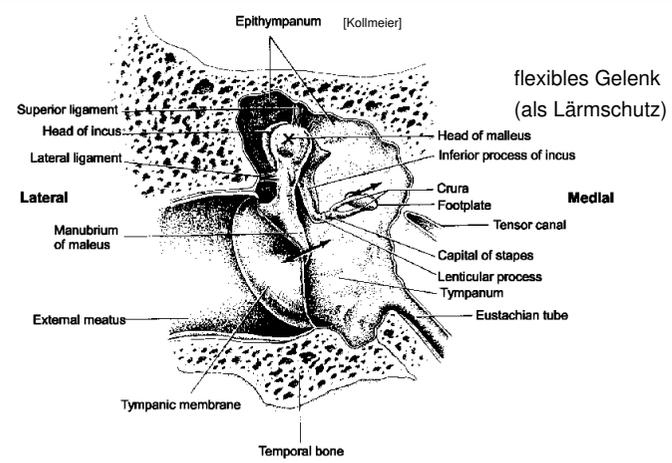
- Außenohr
- Mittelohr
- Innenohr

bzw.:

- Verstärkung, Delay
- Impedanzwandlung
- Frequenzanalyse

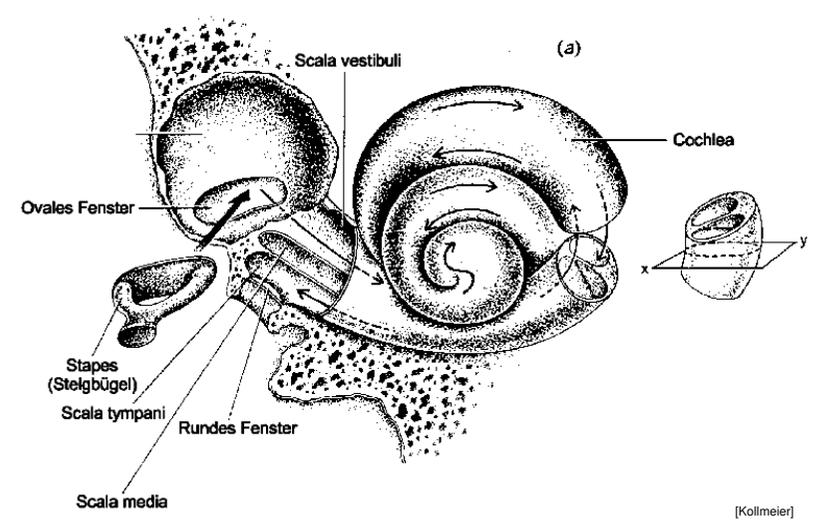
[glasklar Mensch 3D]

Mittelohr



- Verstärkung (Impedanzwandlung), Übergang Luft nach Wasser
- Trommelfeld ca. 20fache Fläche des ovalen Fensters

Innenohr: Aufbau und Cochlea



[Kollmeier]

Innenohr: cortisches Organ

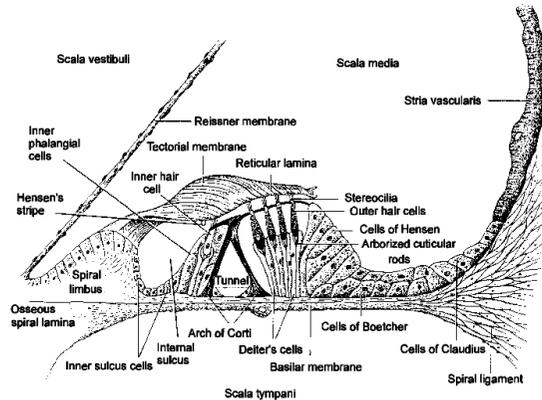
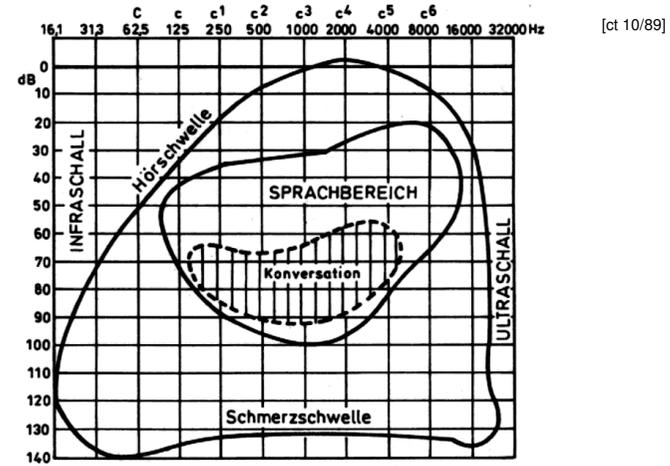


Abbildung 3.7: Schematische Darstellung des Cortischen Organs (aus Gulick, W. L., Gescheider, G. A., Frisina, R. D.: Hearing. Oxford University Press, Oxford 1989)

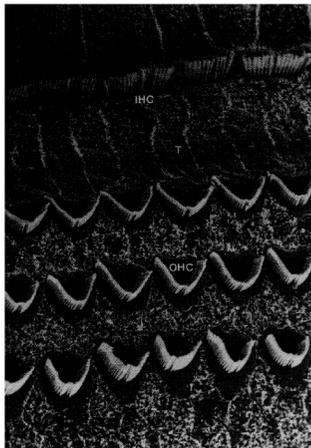
[Kollmeier]

Hörbereich und Sprachbereich



gute Sprachverständlichkeit im "Hauptsprachbereich" (ca. 300-3500 Hz)

Innenohr: Haarzellen



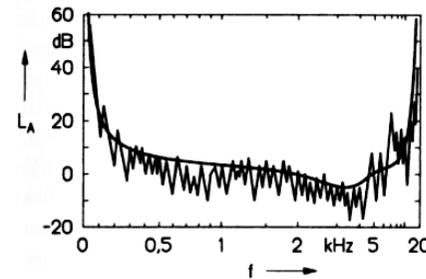
innere Haarzellen
"Sensoren"

äußere Haarzellen
"Aktoren"

Abbildung 3.8: Detailsicht der Innenohr-Anatomie, IHC = Innere Haarzellen, OHC = Äußere Haarzellen (aus Dallos, P., Popper, A. N., Fay, R. R. (Hrsg.): The Cochlea. Springer Verlag, New York 1996)

[Kollmeier]

absolute Hörschwelle

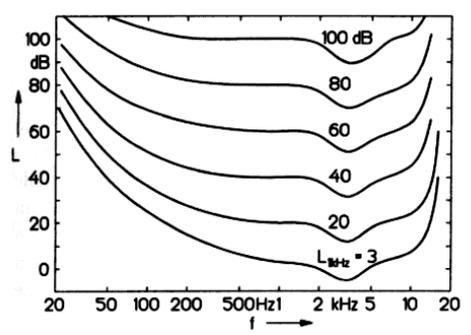


[Terhardt]

Abb. 9.11. Monaurale Absoluthörschwelle für Sinustöne. Glatte Kurve: Mittelwert des Schwellenpegels L_A von normalhörenden Personen. Gezackte Kurve: Mit der Methode des pendelnden Regels (Békésy tracking) gemessen (25-jähriger Mann, rechtes Ohr). Abszisse: Tonfrequenz, SPINC-skaliert; Ordinate: Schallpegel

- geringe Empfindlichkeit bei tiefen und hohen Tönen
 - maximale Empfindlichkeit zwischen 2..5 KHz
- => Loudness-Kurven

Lautheit

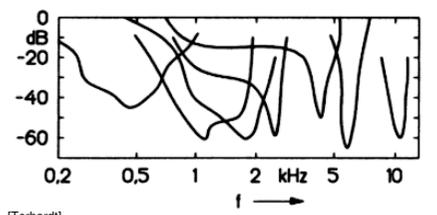


[Terhardt]

Abb. 10.8. Kurven gleicher Lautheit von Sinustönen im ebenen Schallfeld: Schallpegel L eines Sinustones (Ordinate), welcher die gleiche Lautheit hervorruft wie ein 1 kHz-Ton mit dem angegebenen Pegel, in Abhängigkeit von der Tonfrequenz f (Abszisse)

- bei leisen Tönen schwache Wahrnehmung von Höhen/Bässen
- Empfindlichkeitsmaximum bei ca. 3 KHz
- "loudness"-Regler am Hifi-Verstärker zur Kompensation

Tuningkurven: Katzen



[Terhardt]

Abb. 9.10. Tuningkurven, gemessen an anästhesierten Katzen. Jede Kurve gehört zu einer anderen Faser des akustischen Nerven. Die Ordinate gibt denjenigen Schallpegel eines Sinustones an, welcher zur Erregung einer festen Anzahl von Aktionspotentialen pro Zeiteinheit erforderlich ist. Abszisse: Frequenz des Sinustones. Nach Kiang [501]

- hohe Güte der Frequenzselektion
- "aktive" Frequenzbereiche der Fasern überlappen
- menschliches Gehör entsprechend

Maskierung

Mithörschwelle eines Sinustones bei Verdeckung durch Schmalbandrauschen. In Abb.11.4 sind drei Mithörschwellen von Sinustönen verdeckt durch Schmalbandrauschen der angegebenen Mittenfrequenzen dargestellt [1125]. Die Fourier-Spektren der maskierenden Schmalbandgeräusche sind durch Verwendung von Bandpaßfiltern hoher Flankensteilheit auf Bandbreiten von ungefähr 100, 160 und 700 Hz beschränkt [CD 2].

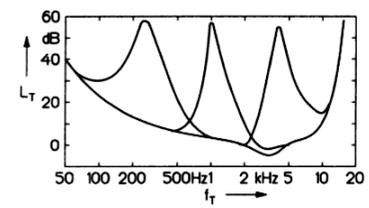
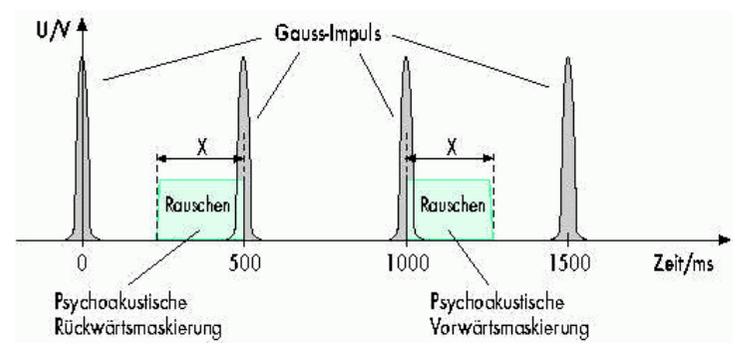


Abb. 11.4. Mithörschwellen von Sinustönen bei Verdeckung durch frequenzgruppenbreite Schmalbandrauschen mit den Mittenfrequenzen 250, 1000, 4000 Hz und dem Maskierpegel $L_M = 60$ dB. Nach [1125]

[Terhardt]

- Frequenzanalyse in der Cochlea nicht perfekt
- laute Töne verdecken benachbarte leisere
- Ausnutzung in psychoakustischen Modellen (MiniDisc, MP3, usw)

Maskierung: temporal



[ct]

- laute Töne verdecken spätere leisere Töne
- und frühere leisere Töne (!!)
- Ausnutzung in psychoakustischen Modellen (MiniDisc, MP3, usw)

Frequenzgruppen

Tabelle 9.1: Frequenzgruppen nach Zwicker 1982

z/Bark	f_u /Hz	f_s /Hz	Δf_G /Hz	f_m /Hz
0	0	100	100	50
1	100	200	100	150
2	200	300	100	250
3	300	400	100	350
4	400	510	110	450
5	510	630	120	570
6	630	770	140	700
7	770	920	150	840
8	920	1080	160	1000
9	1080	1270	190	1170
10	1270	1480	210	1370
11	1480	1720	240	1600
12	1720	2000	280	1850
13	2000	2320	320	2150
14	2320	2700	380	2500
15	2700	3150	450	2900
16	3150	3700	550	3400
17	3700	4400	700	4000
18	4400	5300	900	4800
19	5300	6400	1100	5800
20	6400	7700	1300	7000
21	7700	9500	1800	8500
22	9500	12000	2500	10500
23	12000	15500	3500	13500
24	15500			

Tonhöhenwahrnehmung?

- Messungen
- vgl. Tuningkurven
- tiefe Frequenzen: ca. 100 Hz Bandbreite
- hohe Frequenzen: log. Breite der Gruppen
- => "Bark" Frequenzskala
- Ausgangspunkt für psychoakustische Modelle

[Zwölzer]

Phasen: Hilbertfilter

original



Sprachsample "Oberseminar"

phasengedreht



[vdHeide]

- Ohr unempfindlich gegen Phaseninformation
- verändertes Signal klingt einzeln (vollkommen) gleich
- erst bei Interferenzen mehrerer Signale hörbar
- Phasengang vieler Systeme (Telefon!) extrem schlecht

Klangfarben

akustisches Signal mit vielen Teiltönen:

- keine Wahrnehmung aller Einzeltöne
- sondern Zuordnung zu Quellen/Instrumenten
- tiefste Frequenz bestimmt die wahrgenommene Tonhöhe
- Obertonspektrum identifiziert die Quelle
- mit Eigenschaft "Klangfarbe"
- Klangfarbe auch stark vom "attack" des Klangs abhängig (=> Synthesizer mit kurzen Samples plus Schleifen)
- entsprechend automatische "Sprecheridentifikation"
- komplexe Zusammenhänge für Akkordwahrnehmung

Klangfarben: Spektrogramm

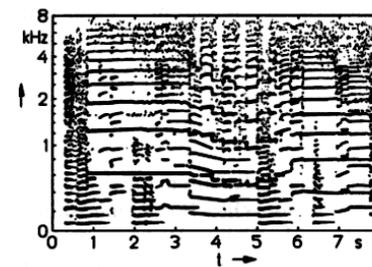


Abb. 1.9. Beispiel eines Teiltonzeitmusters. Ausschnitt einer mehrstimmigen Blasmusikdarbietung. Die hauptsächlich beteiligten Instrumente sind Trompete, Posaune, Klarinette und Tuba. Die instantane Teilton-Amplitude wird durch die Strichstärke angedeutet. Ordinate Bark-skaliert. Nach [969]

[Terhardt]



Abb. 1.10. Transkription der Trompeten- und der Tuba-Stimme des Musikbeispiels, dessen Teiltonzeitmuster in Abb. 1.9 dargestellt ist

Klangfarben: Beispiele

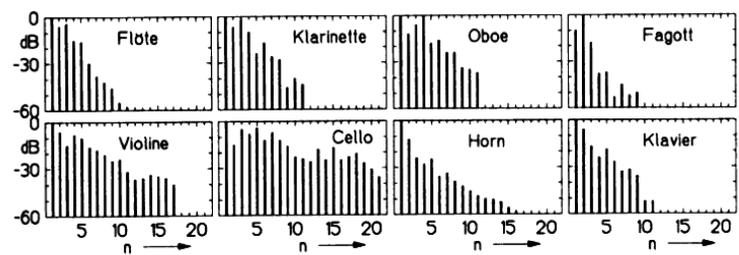
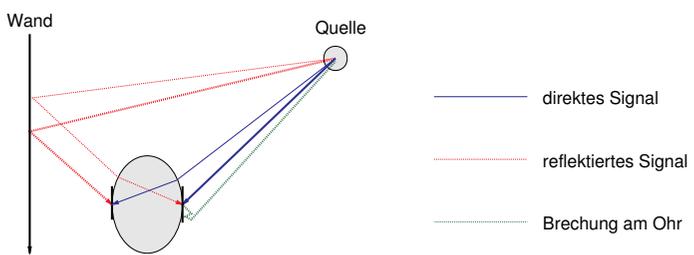


Abb. 8.12. Teiltonspektren verschiedener Musikinstrumente schematisch [Terhardt]

- Streicher mit reichem Obertonspektrum
- Flöten mit sehr reinen Tönen (insb. Blockflöte)
- Klavier stark anschlag- und tonhöhenabhängig
- aber auch "Attack-Phase" maßgeblich für Klangwahrnehmung

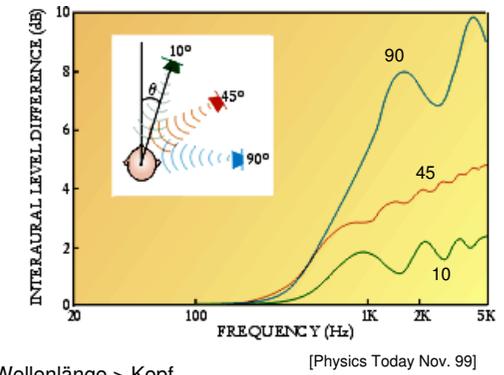
Akustische Raumwahrnehmung



- direktes Signal
- reflektierte Signale
- jeweils mit Beugung um den Kopf herum
- jeweils mit Brechung an Ohr, Schultern, ...
- zusätzlich "obstruction" und "occlusion" für verdeckte Quellen ...

ILD: Lautstärkedifferenz

interaural level difference:



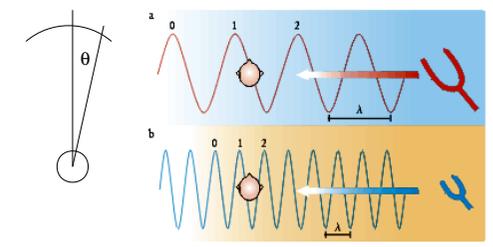
- Welle kaum gestört, wenn Wellenlänge > Kopf
- daher kaum Effekt bei niedrigen Frequenzen
- Bässe schlecht ortbar

ITD: Zeit- und Phasendifferenz

interaural time difference:

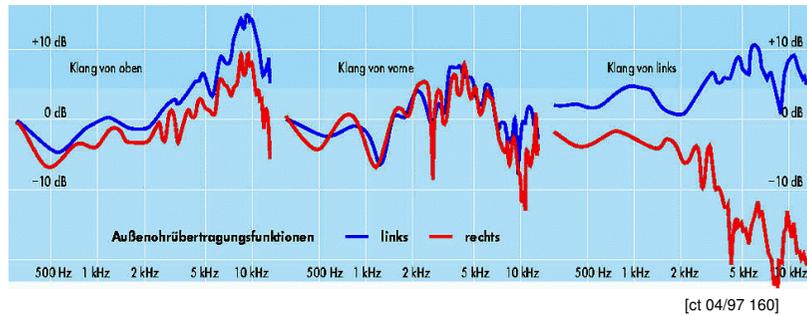
$$dt = \frac{3a}{c} \sin \theta$$

a = Kopradius ~ 8.75 cm
c = Schallgeschw ~ 340 m/s



- 500 Hz Sinuston von vorne ortbar mit ca. 1 Grad Auflösung
=> ca. 13 ms Zeitauflösung
- mehrdeutig, wenn n* Wellenlänge ~ = Kopfdurchmesser
- Ohr unempfindlich für Phaseninformation oberhalb ca. 1 KHz
- vgl. Hilbertfilter-Experiment

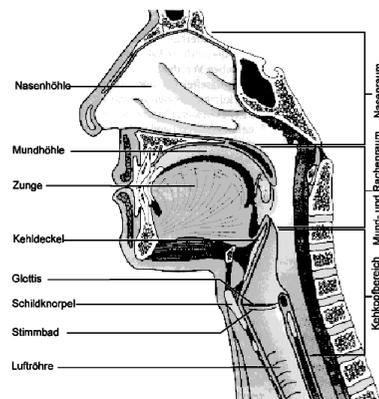
HRTF



"head related transfer function": Außenohr/Kopf-Übertragungsfunktion

- Einfluß von Kopf, Schultern, Außenohr auf Schallsignale
- sehr komplexe Winkel- und Frequenzabhängigkeit
- Verbesserung der Ortung durch (kleine) Kopfbewegungen

Sprache: Vokaltrakt



- Glottis, Stimmband
- Klang zusätzlich durch Zungen-/Lippenstellung
- typische Spektren: Mann / Frau / Kind normal / erregt

Abbildung 6.1: Vokaltrakt im Querschnitt (schematisch) (aus Kießling, J., Kollmeier, B., Diller, G.: Versorgung und Rehabilitation mit Hörgeräten. Thieme Verlag, Stuttgart 1997)

[Kollmeier]

Sprache: Spektrogramm

Formanten, d. h. die Frequenzbereiche, bei denen der Vokaltrakt eine besonders hohe Verstärkung des akustischen Signals bewirkt. Die Formanten stellen sich damit als Spitzen im Spektrum dar und sind charakteristisch für den jeweils artikulierten Vokal (s. unten).

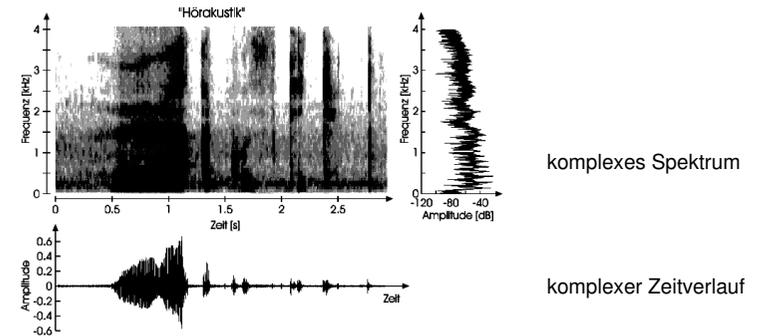


Abbildung 6.2: Zeit-Signal (unten), Spektrum (rechts) und Spektrogramm (links oben) des Wortes „Hörakustik“ [Kollmeier]

Sprache: Artikulation

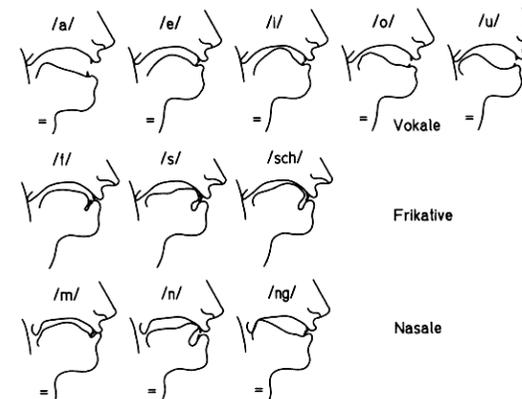


Abb. 7.1. Schematische Darstellung der Artikulation einiger Sprachlaute

- Zungen- und Lippenstellung maßgeblich

[Terhardt]

Sprache: Vokaltrakt

- Stimmband + Resonanzrohr
- Lage der Formanten:

Wellenlänge: $L/4 = 17\text{cm}$

$F_1 = c/L = 340\text{m/s} / 68\text{cm} = 500\text{ Hz}$

$F_2 = 1500\text{ Hz}$

$F_3 = 2500\text{ Hz}$

$F_4 = 3500\text{ Hz}$

Gehör: Zusammenfassung

Gehör außerordentlich leistungsfähig:

- überdeckt 13 Größenordnungen Schalldruck
 - fast optimale Empfindlichkeit
 - Hörbereich etwa 20Hz .. 20 KHz
 - Frequenzanalyse "mechanisch" in der Cochlea
 - komplexe Verarbeitung/Erkennung im Gehirn
 - Wahrnehmung Klang vs. Geräusch
- insbesondere zur Raumwahrnehmung:
- Zeit- und Intensitätsdifferenzen der Signale
 - Kopfübertragungsfunktion
 - Auswertung von Echos
- => entsprechende Modelle für effiziente Signalverarbeitung

Sprache: Vokaldreieck

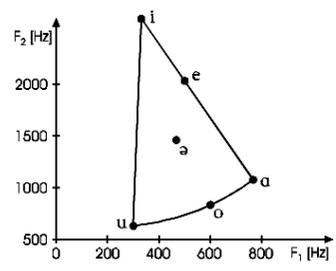


Abbildung 6.1.4: Vokaldreieck. Aufgetragen ist die Frequenz des ersten Formanten F_1 (Abzisse) und des zweiten Formanten F_2 (Ordinate) für unterschiedliche Vokale (schematisch) [Kollmeier]

- typische Frequenzen / Formanten der Vokale
- verhältnismäßig einfache Erkennung von stimmhaften Lauten

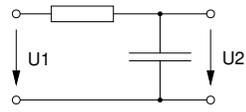
Digitale Signalverarbeitung

- Motivation
- analoge vs. digitale Signalverarbeitung
- Zahlenfolgen, LTI-Systeme
- Abtasttheorem
- Quantisierung, AD/DA Konverter, Dithering
- Beispiele für Audio-Algorithmen
- digitale Filter

DSP: Motivation

analoge Signalverarbeitung:

- Modellierung mit Differentialgleichungen
- direkte Umsetzung mit diskreten Bauteilen
- hauptsächlich mit "analoger" Elektronik
- Widerstände, Kondensatoren, Spulen, Transistoren



aber:

- entweder geringe Genauigkeit (z.B. 10%) oder sehr teuer
- Alterung, Exemplarstreuungen, aufwendiger Abgleich
- wirtschaftlich nur für geringe Anzahl der Bauelemente
- Datenspeicherung (analog) problematisch

=> komplexe Algorithmen nicht umsetzbar
=> digitale Verarbeitung verspricht Abhilfe

DSP: Definition

"digital signal processing" (digitale Signalverarbeitung):

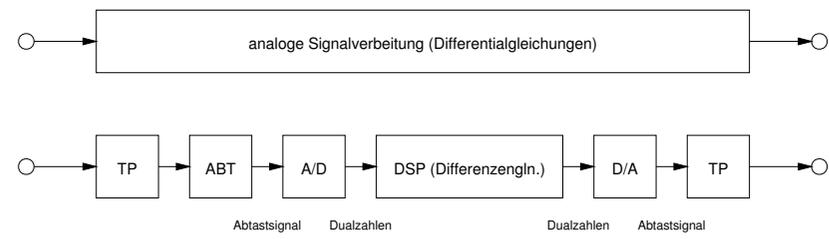
- | | |
|---|--|
| <p>Verbeitung von</p> <ul style="list-style-type: none"> • zeitdiskreten • wertdiskreten • Zahlenfolgen <p>auf Digitalrechnern</p> | <p>für:</p> <p>Nachrichtenübertragung</p> <p>Bildverarbeitung</p> <p>Audioverarbeitung</p> <p>usw.</p> |
| <ul style="list-style-type: none"> • als Ersatz analoger Verfahren • Differenzen- statt Differentialgleichungen • oft mit Echtzeitanforderungen • im Prinzip beliebige Genauigkeit • oft mit Spezialhardware, etwa Signalprozessoren / ASICs | |

DSP: Eigenschaften

Verarbeitung zeit- und wertdiskreter Zahlenfolgen:

- als Ersatz für zeit/wert-kontinuierliche Verarbeitung
- wenn Abtasttheorem erfüllt (genügend hohe Abtastrate)
- beliebige Genauigkeit (Wortbreite anpassen)
- optimale Stabilität (z.B. keine Temperaturabhängigkeit)
- kein Abgleich notwendig (z.B. Nullpunktseinstellung)
- keine Alterung, keine Toleranzen
- auch für sehr komplexe Algorithmen
- die analog nicht (wirtschaftlich) realisierbar sind
- insbesondere: Datenspeicherung
- VLSI-Technologie erlaubt sehr hohe Performance

DSP: vs. analoge Verarbeitung



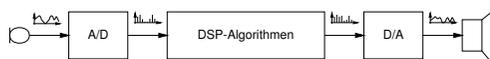
- | | |
|---|--|
| <p>"analoge" Signalverarbeitung</p> <ul style="list-style-type: none"> • zeitkontinuierlich • wertkontinuierlich • Differentialgleichungen • Drift, Rauschen, ... | <p>"digitale"</p> <ul style="list-style-type: none"> • zeitdiskret (Abtastung) • wertdiskret (Quantisierung) • Differenzgleichungen • Überlauf, Rundungsfehler |
|---|--|

DSP: Audio-Anwendungen

Signalverarbeitung im Audio-Bereich:

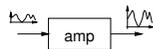
- Datenspeicherung (CD, DVD, ...)
 - Datenkompression (MP3, AC3, MiniDisc, ...)
 - Datenübertragung (DSR, GSM, ...)
- jeweils für Musik und Sprache
- Musikproduktion (digitales Tonstudio)
 - Musiksynthese (z.B. virtuell analog)
 - Audioeffekte (z.B. Hall)
 - Surround-Verfahren (z.B. DirectX mit HRTF)
- Samplefrequenz typ. 44.1 / 48 KHz
 - hohe Genauigkeit erforderlich: 16 .. 24 bit

DSP: Verstärker, Mixer



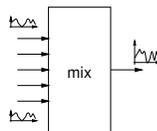
- digitaler Verstärker

$output[t] = input[t] * gain;$



- digitaler Mixer:

```
output[t] = 0;
for( int i=0; i < n_inputs; i++) {
    output[t] += input[i][t] * gain[i];
}
```



- viele MAC-Operationen (multiply-accumulate)
- Problem Aussteuerung vs. Überlauf => saturation arithmetic

DSP: Echo, Hall, und mehr



- Verzögerung: auf alte Abtastwerte zurückgreifen

$delay[t] = input[t]*gain + input[t-t2]*gain2;$

- Rekursion möglich: Echo, Hall, Raumsimulation

$hall[t] = input[t]*gain + hall[t-t2]*gain2;$

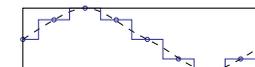
- Algorithmen für viele Anwendungen:

Verzögerung, Echo, Hall, Overdrive, Verzerrung, ...
Filter, Formatfilter, Tonhöhenänderung, Tempoänderung, ...

DSP: Sampling und Synthese

- Sampling: Abtastwerte abspeichern

$sample[t] = input[t];$

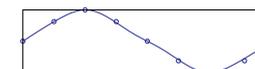


- Samples direkt abspielbar (CD, Spiele)



- "Wavetable"-Synthesizer:

```
output[t] = sample[t*pitch]
            interpolate( sample[] )
            filter( interpolate( sample[] ) )
            effects( filter( interpolate( sample[] ) ) )
```



keine/lineare/Spline-Interpolation

- Interpolation: nearest / linear / splines / ...

Audiosysteme: Anforderungen

Grundanforderungen:

- CD/DAT Qualität 16 bit linear, 44.1 / 48 KHz
- höhere Genauigkeit 24 bit linear PCM
- oder Gleitkomma 32 bit IEEE

Beispiele:

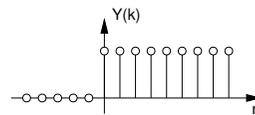
- | | # MAC/s | (48 KHz) |
|------------------------------------|---------------|----------|
| • 10 Kanal Mixer | 480.000 | |
| • 100-tap FIR-Filter | 4.800.000 | |
| • 1000-tap FIR-Filter | 48.000.000 | |
| • 10 Kanal Mixer, je 3 100-tap FIR | 144.480.000 | |
| • ... | | |
| • Pentium-200 (10 Takte/Mult.) | 20.000.000 | |
| • Pentium-III, 1 GHz, ISSE (max:) | 4.000.000.000 | |

=> Spezialhardware oder modern(st)e Prozessoren

Zahlenfolgen, Elementarfolgen

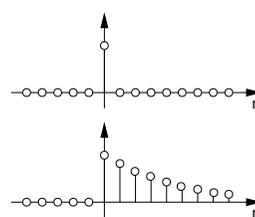
Wertemenge $\{Y(k)\}$

Zuordnung: Index $k \rightarrow Y(k)$



wichtige "Elementarfolgen":

- Impulsfolge
- Sprungfolge
- Exponentialfolge
- Sinusfolge



lineare Systeme:

- Signale aus Elementarfolgen zusammensetzen
- Systemverhalten entsprechend berechenbar

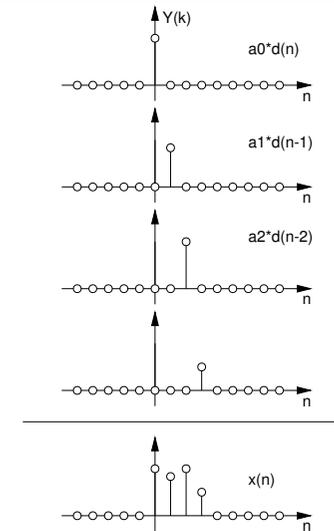
Impulsfolge:

Konstruktion beliebiger Zahlenfolgen:

- Summe
- skaliertes
- zeitverschobener Impulsfolgen

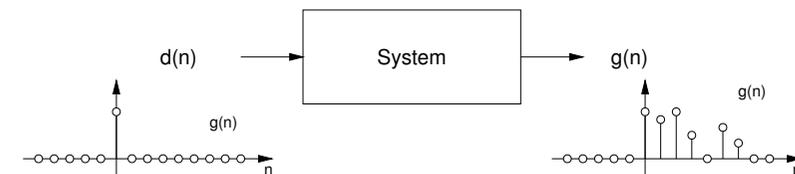
$$x(n) = a_0 * d(n) + a_1 * d(n-1) + \dots$$

- gleichwertig: Konstruktion mit
- skalierten, zeitverschobenen
- Sprungfolgen



Impulsantwort

"Impulsantwort" := Ausgangsfolge als Reaktion auf die Impulsfolge



=> liefert vollständige Beschreibung von LTI-Systemen
"linear, time-invariant"

z.B. Verstärker, Filter, usw.

=> auch als lineare Approximation nichtlinearer Systeme

Differenzgleichungen

Berechnung des Ausgangswerts $Y(n)$

- aus gegenwärtigem Eingangswert $X(n)$
- aus früheren Eingangswerten $X(n-k)$
- aus früheren Ausgangswerten $Y(n-k)$ (Rekursion)
- besonders wichtig: linear, konstante Koeffizienten, 2. Ordnung

$$y(n) = A_0 x(n) + A_1 x(n-1) + A_2 x(n-2) - B_1 y(n-1) - B_2 y(n-2)$$

Ordnung N: $B_1 \dots B_N$ kommen vor, $A_1 \dots A_M$, $M < N$

- Übertragungsfunktion $H(z)$ durch Z-Transformation:

$$Y(z) = A_0 X(z) + A_1 X(z)/z + A_2 X(z)/z^2 - B_1 Y(z)/z + B_2 Y(z) / z^2$$

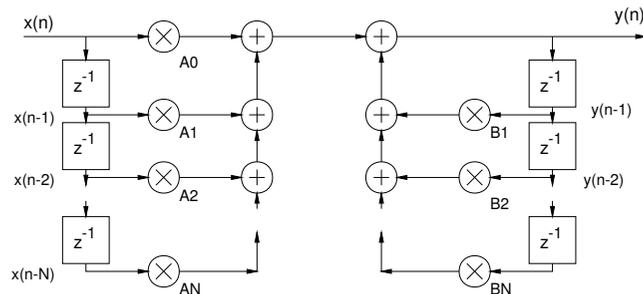
DGL: Direktform

Differenzgleichungen, linear, konstante Koeffizienten

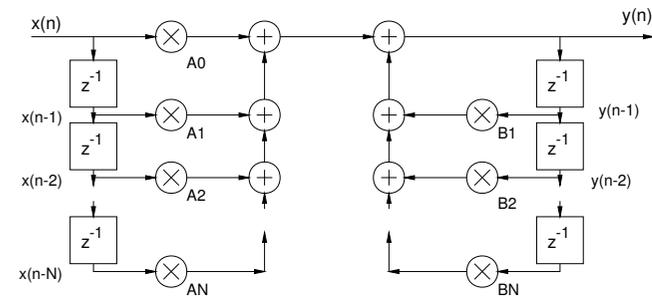
$$y(n) = A_0 x(n) + A_1 x(n-1) + A_2 x(n-2) - B_1 y(n-1) - B_2 y(n-2)$$

Umsetzung in "Direktform":

- Addition, Multiplikation, Verzögerung (=Register)



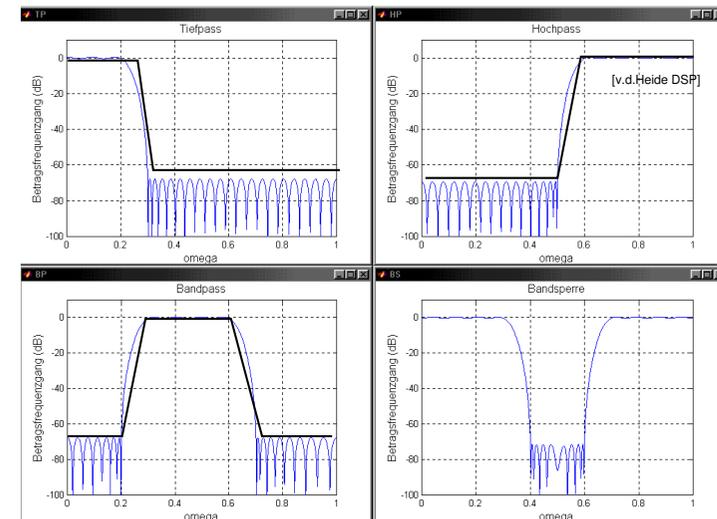
DGLs: Blockschaltbilder



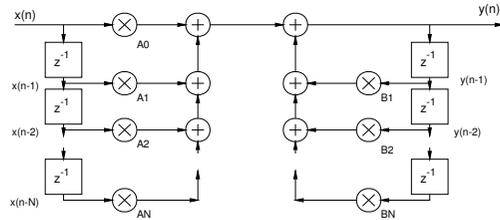
- Direktform erlaubt sofortige Realisierung von DSP-Algorithmen
- Varianten möglich (Vertauschen von Registern/Operatoren)
- bei Bedarf zusätzlich weitere (z.B. nicht-lineare) Operatoren

=> Darstellung mit Blockschaltbildern

Filter: Tiefpaß, Hochpaß, etc.



Filter: Frequenzgang



- normierte Frequenz $F = f/FA$
- direkte Berechnung des Frequenzgangs via $H(z)$:

$$|\underline{A}(j\omega)| = \sqrt{\frac{\left[\sum_{k=0}^n \alpha_k \cos 2\pi k F \right]^2 + \left[\sum_{k=0}^n \alpha_k \sin 2\pi k F \right]^2}{\left[\sum_{k=0}^n \beta_k \cos 2\pi k F \right]^2 + \left[\sum_{k=0}^n \beta_k \sin 2\pi k F \right]^2}}$$

Abtasttheorem

macht die Verarbeitung von Zahlenfolgen überhaupt Sinn ?

Abtasttheorem:

ein auf die halbe Abtastfrequenz (=Nyquistfrequenz)
bandbegrenzte Signal kann nach der Abtastung
perfekt rekonstruiert werden

- Grundlage aller digitalen Signalverarbeitung
- Beweis mit Fouriertransformation
- siehe [Shannon 48]
- siehe Matlab-Skript Signalverarbeitung

Abtasttheorem

7.4 Abtasttheorem für Zeitfunktionen

Ein reelles bandbegrenzte Signal $u(t)$ hat eine Signalgrenzfrequenz f_{gs} . Für die Fourier-Transformierte $\underline{U}(f)$ dieses Signals gilt:

$$\underline{U}(f) = 0 \text{ für } |f| > f_{gs} \quad (7.5)$$

Durch periodische Wiederholung von $\underline{U}(f)$ mit der Periode $2 f_{gs}$ entsteht eine periodische Frequenzfunktion, die in eine Fourier-Reihe

$$\underline{U}_{per}(f) = \sum_{n=-\infty}^{\infty} c_n \exp(-j 2\pi n \frac{f}{2 f_{gs}}) \quad (7.6)$$

entwickelt werden kann. Die Zeitfunktion

$$u(t) = \int_{-\infty}^{\infty} \underline{U}(f) \exp(j 2\pi f t) df \quad (7.7)$$

ergibt sich als inverse Fourier-Transformierte der Frequenzfunktion $\underline{U}(f)$. Da diese

[Gerdsen u. Kröger]

Abtasttheorem

nur im Bereich

$$-f_{gs} < f < +f_{gs} \quad (7.8)$$

von null verschieden ist und innerhalb dieses Bereiches mit $\underline{U}_{per}(f)$ übereinstimmt, kann für die Zeitfunktion auch

$$u(t) = \int_{-f_{gs}}^{f_{gs}} \left\{ \sum_{n=-\infty}^{\infty} c_n \exp(-j 2\pi n \frac{f}{2 f_{gs}}) \right\} \exp(j 2\pi f t) df \quad (7.9)$$

geschrieben werden. Man fasst die e-Funktionen zusammen und bildet die Summe der Integrale. So erhält man:

$$u(t) = \sum_{n=-\infty}^{\infty} c_n \int_{-f_{gs}}^{f_{gs}} \exp(-j 2\pi f t + j \frac{n}{2 f_{gs}}) df \quad (7.10)$$

Die Lösung des Integrals in dieser Summe ergibt:

$$\int_{-f_{gs}}^{f_{gs}} \exp(-j 2\pi f t + j \frac{n}{2 f_{gs}}) df = 2 f_{gs} \frac{\sin(2\pi f_{gs} t + \frac{n}{2 f_{gs}})}{2\pi f_{gs} t + \frac{n}{2 f_{gs}}} \quad (7.11)$$

Zur Bestimmung der Fourier-Koeffizienten c_n wird

$$t = k \frac{1}{2 f_{gs}} \quad k \text{ ganze Zahl} \quad (7.12)$$

gesetzt. Dann sind in der Summe (7.10) alle Terme mit $n \neq k$ null. Damit gilt für

Abtasttheorem

die Koeffizienten:

$$s_n = \frac{1}{f_{gs}} u\left(n \frac{1}{2f_{gs}}\right) \quad (7.13)$$

So entsteht aus (7.10) die Interpolationsformel

$$u(t) = \sum_{n=-\infty}^{\infty} u\left(n \frac{1}{2f_{gs}}\right) \frac{\sin\left(2\pi f_{gs}\left(t - \frac{n}{2f_{gs}}\right)\right)}{2\pi f_{gs}\left[t - \frac{n}{2f_{gs}}\right]} \quad (7.14)$$

die aussagt, daß ein bandbegrenzte Signal $u(t)$ mit der Signalgrenzfrequenz f_{gs} vollständig beschrieben ist durch seine Abtastwerte zu den Zeiten

$$t_k = k \frac{1}{2f_{gs}} \quad k \text{ ganze Zahl} \quad (7.15)$$

Dieses Abtasttheorem für Zeitfunktionen ist von grundlegender Bedeutung für die digitale Signalverarbeitung.

Abtasttheorem: Schema

Eingangsspektrum

Eingangstiefpaß

bandbegrenzte Signal

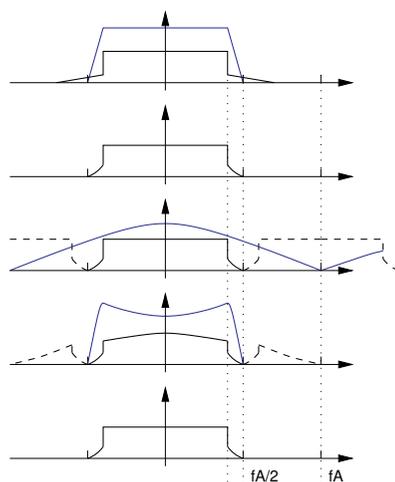
Abtastung

Abtastung $\sin(x)/x$

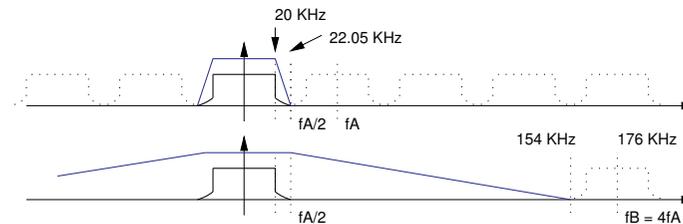
Rekonstruktion

Ausgangstiefpaß

Ausgangsspektrum



Abtasttheorem: Überabtastung



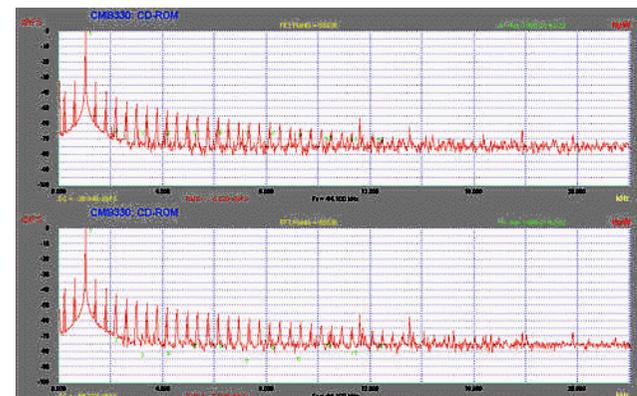
Abtasttheorem:

- vollständige Rekonstruktion des Signals, aber:
- erfordert analogen Tiefpaß sehr hoher Qualität
- Beispiel CD: Dämpfung 96 dB von 20 KHz bis 22.05 KHz
- typische analoge Filter: 6 .. 24 dB / Oktave

Oversampling:

- digitale Neu-Abtastung mit höherer Rate (z.B. 4x, 8x)
- digitale Tiefpaßfilterung beseitigt die Spiegelfrequenzen
- einfacher analoger Tiefpaß genügt

Abtasttheorem: Aliasing-Fehler . . .



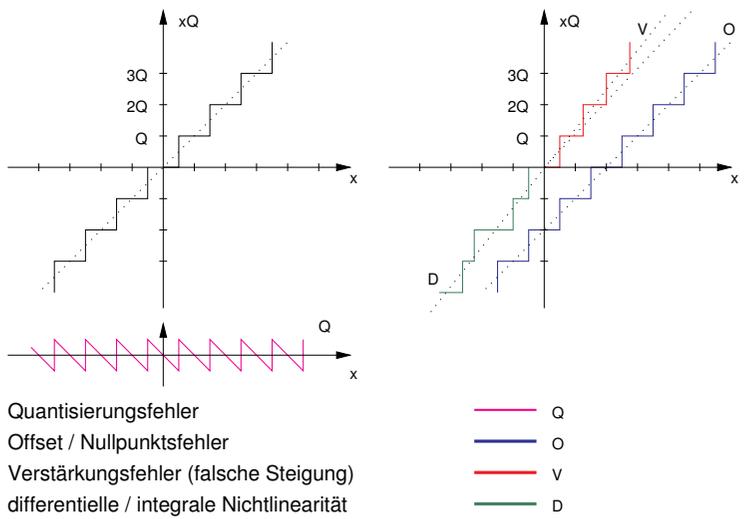
- Meßwerte der CMI 8330 Soundkarte [ct 10/98]
- Sinussignal und massive Anti-Aliasing Fehler (Spiegelfrequenzen)
- massiver Qualitätsverlust durch Stümperei

A/D-Wandlung

Digitalisierung eines Eingangssignals:

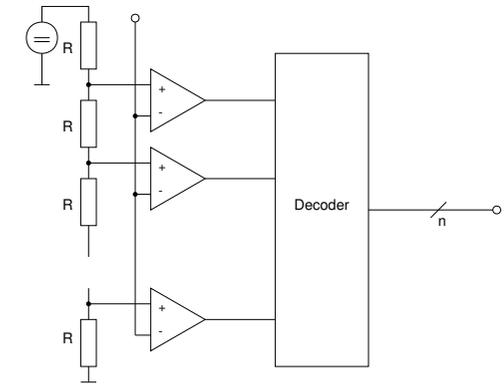
- Zeitquantisierung (Abtaster, Sample&Hold)
 - Amplitudenquantisierung (A/D-Konverter)
 - Codierung (meistens 2-Komplement)
-
- nur für bandbegrenzte Eingangssignale
 - analoger Tiefpaß vor dem Eingang erforderlich
-
- diverse Verfahren
 - Trade-off zwischen Performance / Genauigkeit / Kosten
 - für Audio: 24-bit Wandler bereits erhältlich

A/D: Parameter



- Quantisierungsfehler
- Offset / Nullpunktsfehler
- Verstärkungsfehler (falsche Steigung)
- differentielle / integrale Nichtlinearität

A/D: Parallel-Umsetzer

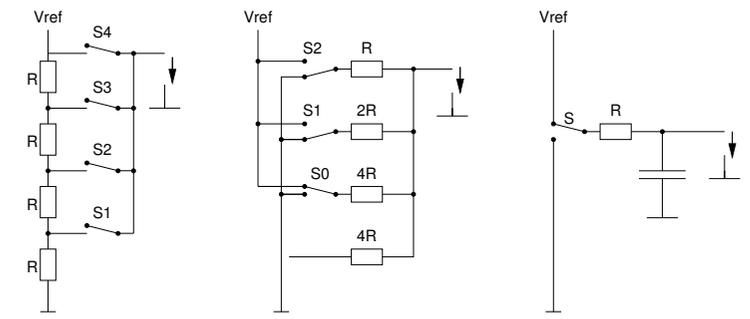


- erfordert $2 \cdot n$ Komparatoren
- sehr schnell
- aber nur für geringe Auflösungen (z.B. 8-bit)

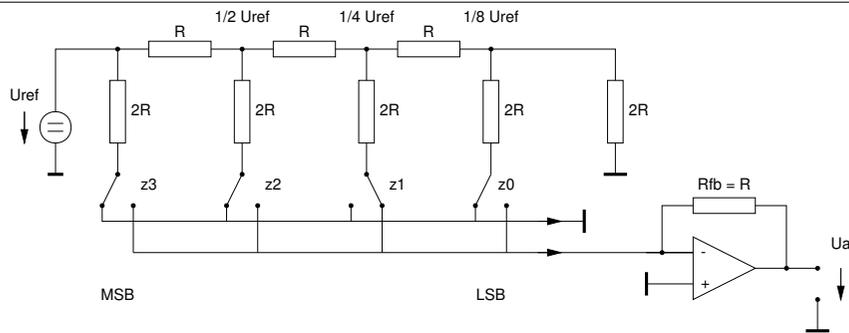
D/A-Wandlung

D/A: Umwandlung numerischer Werte in analoge Werte (Spannungen)

- Parallelverfahren
- Wägeverfahren
- Zählverfahren



D/A: R-2R Netzwerk



- VLSI: Herstellung unterschiedlicher Widerstände schwierig
- aber gleiche Widerstände mit hoher Genauigkeit
- => fortgesetzte Spannungsteilung mit R-2R Netzwerk
- $$U_a = -U_{ref} \cdot (R_{fb}/16R) \cdot (8z_3 + 4z_2 + 2z_1 + z_0)$$
- in CMOS-Technologie oft verwendet

Quantisierungsprobleme

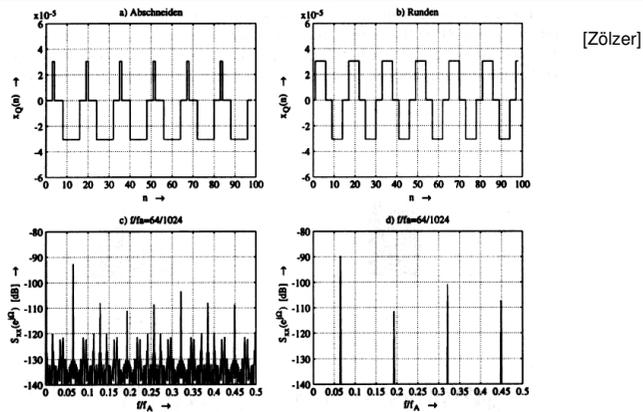
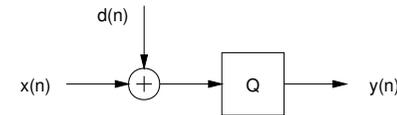


Bild 2.19: 1 Bit Aussteuerung des Quantisierers bei Abschneiden a/c und Runden b/d

- Quantisierung erzeugt störende "Nebensignale"
- Verbesserung durch Hinzufügen von Rauschen: "Dithering"

Dither: Prinzip



		bits
x(n)	Eingangssignal	w+r
d(n)	Zufallssignal (dither)	s < r
y(n)	Ausgangssignal	w

- Q: Re-Quantisierung des Signals, oder D/A-Wandler
- Amplitude des Dithers kleiner als LSB des Eingangssignals
- Verteilung (Statistik) des Dithers frei wählbar: noise shaping
- Mathematik siehe Zölzer S.38 ff

Dither: bei 0.25 bit Aussteuerung

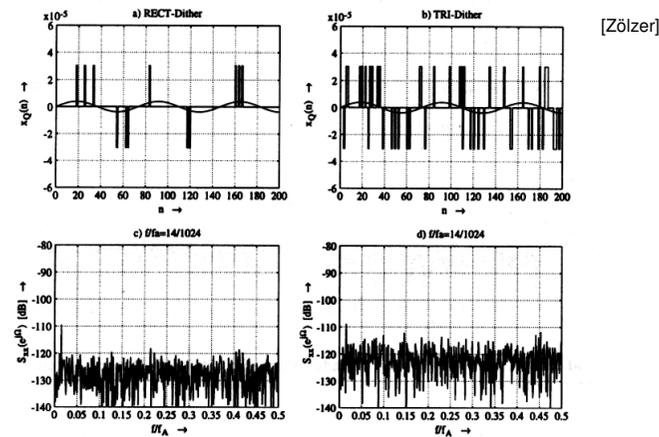


Bild 2.21: 0.25 Bit Aussteuerung - Rundungskennlinie mit RECT-Dither a/c und TRI-Dither b/d

Datenformate

effiziente Speicherung und Übertragung von Audiodaten?

- unkomprimierte Darstellung, PCM
- WAV-Format
- ADPCM
- Sprachcodecs, Kompandierung

spätere Themen:

- psychoakustische Verfahren
- Streaming
- "Meta"-Kodierung: MIDI, MPEG4 Structured Audio

Datenformate: Zahldarstellung

geeignete Zahldarstellung für Audiodaten ?!

- 8/16/24-bit Integer, Offset oder Zweierkomplement
- 32-bit Gleitkomma
- Bitströme

1000 0000

 8-bit mit Offset 128

0000 0000	0000 0000
-----------	-----------

 Zweierkomplement

0000 0000	0000 0000	0000 0000
-----------	-----------	-----------

 Zweierkomplement

0000 0000	0000 0000	0000 0000	0000 0000
-----------	-----------	-----------	-----------

 Zweierkomplement

V	EXP	MANTISSE
---	-----	----------

 IEEE 754 32-bit FP
Konvention: $-1 \leq X \leq +1$

31 24 23 16 15 8 7 0

Datenformate: Kodierung

"direct coding"		PCM
"entropy coding"	repetitive sequence suppression	zero suppression run-length encoding
	statistical encoding	pattern substitution Huffman encoding
"source coding" (e.g. speech)	transform encoding	FFT DCT ...
	differential encoding	DPCM delta modulation ADPCM
	vector quantization	general / fractal / ...

Datenformate: Entropie/Quellenkodierung

Entropiekodierung:

- Eigenschaften der Datenquelle werden ignoriert
- Signalwiederholungen entfernen
- statistische Verfahren, z.B. Huffman-Kodierung
- verlustfrei und reversibel
- für Audiodaten: ca. Kompressionsfaktor 2 erreichbar

Quellenkodierung (source encoding):

- Eigenheiten der Datenquelle / senke berücksichtigen
- z.B. Frequenzgang / Maskierung / Rauschschwellen des Ohrs
- verlustfrei
- verlustbehaftet für bessere Kompression, z.B. MP3 bis ca 10:1

Datenformate: RAW

Audio-CD "raw" Format:

- direkte PCM-Kodierung
- 16-bit Samplewerte, Zweierkomplement
- 44.1 KHz Abtastrate
- Stereo
- Datenrate $44.100 \cdot 2 \cdot 16 \text{ bps} = 1411200 \text{ bps} = 176 \text{ KB/s}$
- bzw. 10.5 MB/min
- großer Speicherbedarf
- aber leicht zu bearbeiten (schneiden, skalieren, ...)
- Details später (CD/DVD)

Medientechnik | WS 2001 | 18.204

Datenformate: SND

```
typedef struct {
    int magic;           /* 0x2e736e64 = ".snd" */
    int dataLocation;   /* offset to the data */
    int dataSize;       /* number of bytes of data */
    int dataFormat;     /* 1=μ-law, 2= linear8, ... */
    int samplingRate;   /* samples per second */
    int channelCount;   /* 1=mono, 2=stereo, ... */
    char info[4];       /* optional text info */
} SNDSoundStruct;
```

- erstes Audioformat auf NeXT und Sun
- einfache Dateistruktur mit Kopf (SNDSoundStruct) und Daten
- übliche Datenrate: 8-bit mono, 8 KHz Samplerate
- diverse Datenformate von 8-bit linear bis G.723
- Zugriff über entsprechende API (NeXT Sound Kit)

Medientechnik | WS 2001 | 18.204

Datenformate: WAV

- Standard-Dateiformat für Audiodaten unter Windows
- Abkömmling des EA IFF85 Formats

"Chunk"-Format:

- Datei besteht aus einzelnen "Häppchen"
- jeder Chunk enthält eigene Headerinformation
- und optional Daten
- Format kann nachträglich um neue Chunks erweitert werden

Zugriff auf hintere Chunks:

- durch Verketteten der Länge der vorherigen Chunks
- erfordert Kenntnis aller vorangegangenen Chunks
- ungeeignet für Streaming / verlustbehaftete Kanäle

Medientechnik | WS 2001 | 18.204

Datenformate: WAV

Hierarchie mit Unzahl von Chunk-Typen:

RIFF Chunk	(Wave File Header)
Format Chunk	(Struktur des Data Chunks)
Data Chunk	(Daten, z.B. PCM Samples)
Fact Chunk	(Info über komprimierte Daten)
Cue Chunk	(Offset zu wichtigen Zeitpunkten)
Playlist Chunk	(Anspielfolge von Cuepunkten)
Associated Data Chunk	(z.B. Songtitel)
Label Chunk	(eigentlicher Titel)
...	

- oft nur drei Chunks: Header/Format/Data
- alle Chunks "word-aligned", evtl. ein Füllbyte 0 ergänzen

Medientechnik | WS 2001 | 18.204

Datenformate: WAV header

```
typedef struct {
    ID ckID; /* 0x52494646 = "RIFF" */
    long ckSize; /* file size -8 */
    ID formType; /* 0x57415645 = "WAVE" */
    char pad[]; /* padding, if ckSize odd */
} WaveChunk;
```

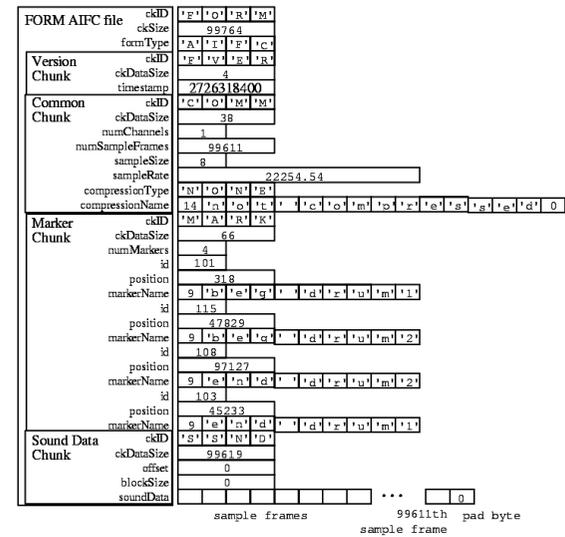
```
typedef struct {
    ID ckID; /* 0x666D7420 = ".fmt" */
    long ckSize; /* 16 + extra format bytes */
    short wFormatTag; /* e.g. WAVE_FORMAT_PCM */
    ushort nChannels;
    ushort nSamplesPerSec;
    ushort nAvgBytesPerSec;
    ushort nBlockAlign;
    ushort nBitsPerSample;
} FormatChunk;
```

Datenformate: WAV data

```
typedef struct {
    ID ckID; /* 0x64617461 = "data" */
    long ckSize; /* in bytes */
    char[] data;
}
```

- WAVE_FORMAT_PCM: 16-bit Zweierkomplementdaten
- aber auch G.711 / G. 721 / GSM / MPEG implementiert
- Stereo/Mehrkanaldaten als Frames
links / rechts, bzw. Kanal 1, 2, 3, ...
- erlaubt das Abspielen, ohne die Datei komplett laden zu müssen

AIFF: Beispiel



DPCM, ADPCM:

DPCM, "differential PCM":

- Differenz zum vorhergehenden Samplewert abspeichern
- lohnt sich, wenn aufeinanderfolgende Werte ähnlich
- anschließend noch RLE- oder Huffman-Kodierung möglich
- für Audiodaten wenig effektiv

ADPCM, "adaptive differential PCM":

- internes Modell zur Vorhersage von Samplewerten
- Abspeichern der Differenz von Vorhersage und Samplewert
- Kompressionsraten bis ca. Faktor 2..4 erreichbar
- Rauschen / rauschartige Klänge verhindern höhere Werte
- Mathematik und Beispiel: siehe Zölzer, Kap. 9.1

ADPCM: Beispiel

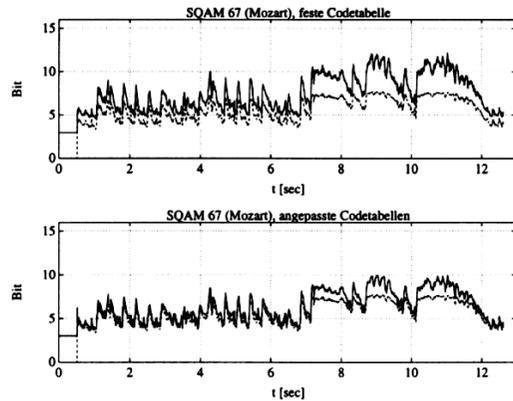


Bild 9.2: Verlustlose Datenkompression (Mozart): Wortbreite [Bit] über der Zeit (Entropie - - , lineare Prädiktion mit Huffman-Codierung —)

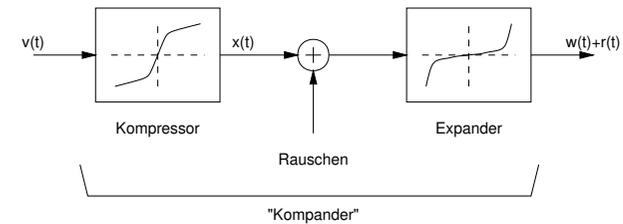
[Zölzer]

Sprach-Codecs:

Sprachkodierung vs. Audio/Musik allgemein:

- möglichst gute Verständlichkeit
- Klangverluste sind durchaus akzeptabel
- Bandbreite von < 4KHz reicht aus
- enorme Bedeutung für die Telefonie
- diverse internationale Standards etabliert
- G.711 8-bit, 8 KHz, μ -Law / a-Law (ISDN)
- G.721 32 kbps ADPCM (4bit/sample)
- G.723 24/40 Kbps ADPCM (3bit/5bit/sample)
- GSM Varianten: full/enhanced full/half-rate
- viele weitere

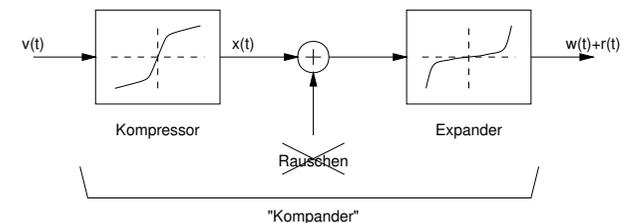
G.711: Kompander



"Überdeckungseffekt" des Gehörs:

- Rauschen stört bei leisen Signalen stärker als bei lauten
 - in leisen Signalen möglichst wenig Rauschen
- => leise Signale vor der Übertragung anheben, später absenken
- Herleitung der Kennlinien: siehe Kammeyer

G.711: Kompander

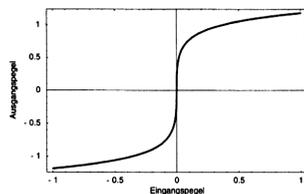


- digitale Übertragung (ISDN) ist rauschfrei

trotzdem Einsatz der Kompanderung:

- Erhöhung des Dynamikbereichs für die Sprachübertragung
- bzw. Reduzierung des Quantisierungsrauschens (leise Signale) durch Anwendung der Kompressor-Kennlinie

μ -Law, a -Law

Bild 5.3: μ -Law-Kompressionskennlinie für normierte Signalpegel

μ -Gesetz (μ -Law)

Diese Art der Dynamikkompression ist im angelsächsischen Raum verbreitet, sie wird manchmal irreführend als mu-Law bezeichnet. Der zu digitalisierende Wert ergibt sich aus dem Eingangspegel S (auf 1 normierter Signalpegel) als

$$S' = \text{sign}(S) \cdot \frac{1 + \ln(1 + \mu \cdot \text{abs}(S))}{\ln(1 + \mu)}, \quad \mu = 255$$

- Kompression von 12..16 bit linear auf 8 bit
- Idee: logarithmische statt linearer Kodierung
- Berechnung: stückweise lineare Approximation / Tabellen

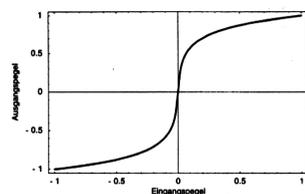


Bild 5.2: A-Law-Kompressionskennlinie für normierte Signalpegel

A-Gesetz (A-Law)

Der zu digitalisierende Wert ergibt sich aus dem Eingangspegel S (auf 1 normierter Signalpegel) als logarithmische Funktion, die für praktische Zwecke meist in linearen Stücken approximiert wird (siehe Tabelle 5.2). Mit einem konstanten Wert $A = 87,6$ lautet die Transferfunktion

$$S' = \frac{\text{sign}(S)}{1 + \ln A} \begin{cases} A \cdot \text{abs}(S) & \text{wenn } \text{abs}(S) \leq 1/A \\ 1 + \ln(A \cdot \text{abs}(S)) & \text{sonst} \end{cases}$$

a -Law: Umrechnung

```
unsigned char _a2u[128] = { /* A- to u-law conversions */
    1, 3, 5, 7, 9, 11, 13, 15,
    16, 17, 18, 19, 20, 21, 22, 23,
    24, 25, 26, 27, 28, 29, 30, 31,
    32, 32, 33, 33, 34, 34, 35, 35,
    36, 37, 38, 39, 40, 41, 42, 43,
    44, 45, 46, 47, 48, 48, 49, 49,
    50, 51, 52, 53, 54, 55, 56, 57,
    58, 59, 60, 61, 62, 63, 64, 64,
    65, 66, 67, 68, 69, 70, 71, 72,
    73, 74, 75, 76, 77, 78, 79, 79,
    80, 81, 82, 83, 84, 85, 86, 87,
    88, 89, 90, 91, 92, 93, 94, 95,
    96, 97, 98, 99, 100, 101, 102, 103,
    104, 105, 106, 107, 108, 109, 110, 111,
    112, 113, 114, 115, 116, 117, 118, 119,
    120, 121, 122, 123, 124, 125, 126, 127};
```

- Konvertierung zwischen a-Law und μ -Law mit Tabellen
- beträchtliche Rundungs/Quantisierungsfehler

a -Law: lineare Approximation

```
/*
 * linear2alaw() - Convert a 16-bit linear PCM value to 8-bit A-law
 *
 * Linear Input Code      Compressed Code
 * -----
 * 0000000wxyz          000wxyz
 * 0000001wxyz          001wxyz
 * 000001wxyzab         010wxyz
 * 00001wxyzabc         011wxyz
 * 0001wxyzabcd         100wxyz
 * 001wxyzabcde         101wxyz
 * 01wxyzabcdef         110wxyz
 * 1wxyzabcdefg         111wxyz
 *
 * For further information see John C. Bellamy's Digital Telephony, 1982,
 * John Wiley & Sons, pps 98-111 and 472-476.
 */
```

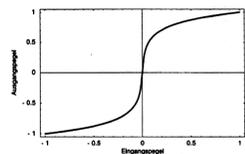


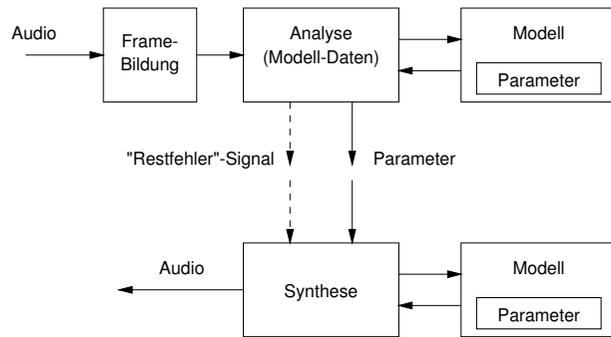
Bild 5.2: A-Law-Kompressionskennlinie für normierte Signalpegel

CELP

"Code(book) Excited Linear Prediction"

- spezieller Codec für Sprachkodierung bei 4.8 kbps
- verwendet Modell für Sprache (Vokaltrakt)
- Modellparameter sind zeitabhängig
- Anpassung der Modellparameter an die Eingabedaten
- LPC: Übertragung der Modellparameter
- Empfänger liest Parameter, Modell erzeugt die Ausgabedaten
- Sprachverständlichkeit ist ok, aber "synthetischer" Klang
- CELP: Berechnung eines Fehlersignals (Modell - Samples)
- Übertragung der Modellparameter und des Fehlersignals

CELP: Blockschaltbild



LPC: einfaches, lineares Modell
 CELP: Suche nach besten Parametern in (fester) Codetabelle

GSM

"Global Standard for Mobile Communication"

- spezieller Codec für Sprachkodierung
- gute Sprachverständlichkeit
- ungeeignet für Musiksignale
- möglichst einfacher Dekoder (Mobilgeräte!)
- trotzdem sehr rechenaufwendig
- mehrere Datenraten: full / enhanced-full / half-rate
- z.B. full-rate mit 13.3 kb/s
13-bit Samples, 8 KHz, 160 Samples -> 260 bits
- Dokumentation und Demo-Code:
<http://kbs.cs.tu-berlin.de/~jutta/toast.html>

Sprach-Codex: Sprachqualität?

Gauging the speech quality is an important but also very difficult task. The signal-to-noise ratio (SNR) is one of the most common objective measures for evaluating the performance of a compression algorithm. This is given by:

$$SNR = 10 \log_{10} \left\{ \frac{\sum_{n=0}^M s^2(n)}{\sum_{n=0}^M (s(n) - \hat{s}(n))^2} \right\} \quad (1)$$

where $s(n)$ is the original speech data while $\hat{s}(n)$ is the coded speech data. The SNR is a long term measure for the accuracy of speech reconstruction and as such it tends to "hide" temporal reconstruction noise particularly for low level signals. Temporal variations of the performance can be better detected and evaluated using a short-time signal-to-noise ratio, i.e., by computing the SNR for each N-point segment of speech. A performance measure that exposes weak signal performance, is the segmental SNR (SEGSNR) which is given by

$$SEGSNR = \frac{10}{L} \sum_{i=0}^{L-1} \log_{10} \left\{ \frac{\sum_{n=0}^{N-1} s^2(iN+n)}{\sum_{n=0}^{N-1} (s(iN+n) - \hat{s}(iN+n))^2} \right\} \quad (2)$$

Sprach-Codex: Bitrate vs. MIPS

Algorithm	Bit Rate (bits/sec)	MOS/DRT/DAM	MIPS*	References
PCM (G.711)	64k	4.3/95/73	0.01	[150][152]
ADPCM (G.721)	32k	4.1/94/68	~2	[22][32][150]
LD-CELP (G.728)	16k	4.0*/-/-	~19	[35][38]
RPE-LTP (GSM)	13k	3.47*/-/-	6	[19][307]
Skyphone-MPLP	9.6k	3.4/-/-	11	[25]
VSELP (IS-54)	8k	3.45*/-/-	13.5	[70][100]
CELP (FS1016)	4.8k	3.2/93.7/62.2	16	[30][78]
STC-1	4.8k	3.52/92.7/63.	13	[210][212][213]
IMBE	4.15k	3.4/-/-	3	[26][121][141]
STC-2	2.4k	2.9/90.1/56	13	[210][212][213]
LPC-10e (FS 1015)	2.4k	2.3/89.9/52.3	~7	[77][301]
LPC-LSP	800	-/91.2/-	~20	[166]
~ estimated, * low score reported, * processor dependent				
The above complexity and performance figures were obtained from different sources and correspond to different implementation platforms and test environments. Therefore the performance and complexity figures do not always constitute an absolute measure for comparison.				

Perceptual Coding: Agenda

- Motivation für psychoakustische Kodierung
- Übersicht der wichtigsten Verfahren
- Literatur

MPEG1 - Audio:

- Konzept
 - Encoder / Decoder
 - Layer 1 / 2 / 3
 - Einige Details der Algorithmen
 - Beispiele zur Klangqualität
- Ausblick auf MPEG2 und MPEG4
 - Ausblick auf MS-Audio (Windows Media) und RealAudio



Perceptual Coding: Motivation

perceptual coding = psychoakustische Kodierung :=

- verlustbehaftete Audiokodierung
- minimale Dateigröße bei möglichst hoher Klangqualität
- Kompression bis ca. 10:1 bei CD-Qualität: 2 bits / sample (!)
- durch Ausnutzen der Gehörwahrnehmung:
 - Ruhegehörschwelle
 - Maskierungseffekte
- erfordert psychoakustische Modelle des Gehörs
- erfordert (Frequenz-) Analyse der Eingangsdaten:
 - "subband coders"
 - "transform coders"

Perceptual Coding: Verfahren

Verfahren	Methode	kHz	Kanäle	kb/s	Qualität
MPEG-1 layer 1	hybrid	32, 44.1, 48	1, 2	32 .. 448	AM / FM / (CD)
MPEG-1 layer 2	hybrid	32, 44.1, 48	1, 2	32 .. 384	AM / FM / (CD)
MPEG-1 layer 3	hybrid	32, 44.1, 48	1, 2	32 .. 320	AM / FM / CD
MPEG-2 AAC	hybrid	16 .. 48	1 .. 5.1	16 ..	AM / FM / CD
ATRAC	subband	44.1	2	256/ch	CD
Dolby AC-2	trans.	44.1	2	256/ch	CD
Dolby AC-3	trans.	44.1	1 .. 5.1	32 .. 384	CD
AT&T PAC	subband	44.1	1 .. 5.1	128/stereo	FM / CD
MS Audio ASF	?	44.1	2	10 ..	AM / FM / CD

- viele weitere Verfahren vorgeschlagen / erforscht
- detaillierte Übersicht siehe Spanias Review

Perceptual Coding: Literatur

- T. Painter, A. Spanias: Perceptual Coding of Digital Audio, Proc. IEEE 88-4, p.451ff (2000)
- T. Painter, A. Spanias: Review of Algorithms for Perceptual Coding of Audio Signals
www.eas.asu.edu/~spanias/papers.html
- D. Pan, An Overview of the MPEG/Audio Compression Algorithm, Proc. SPIE 260 (1997)
- S. Shlien, Guide to MPEG-1 Audio Standard, IEEE Trans. Broadcasting 40-4, p.206 (1994)
- Microsoft / RealAudio: msdn.microsoft.com/windowsmedia/, www.realaudio.com
- Fraunhofer IIS Dokumentation, www.iis.fhg.de
- LAME MP3 Encoder, www.sulaco.org/mp3/
- ISO Reference Software, ftp.tnt.uni-hannover.de/mpeg2/software/
- ISO/IEC Standards:
- MPEG-1, part 3: audio, ISO/IEC 11172-3 (1993)
 - MPEG-2, part 3: audio, ISO/IEC 13818-3 (1994) (Mehrkanal)
 - MPEG-4, part 3: audio
- c't, MP3-Feature (Algorithmus, Tools, Klangtest), c't 03/2000 p.122-151
- c't, MP3 vs MSAudio vs. RealAudio (mit Klangtest), c't 23/2000 p.152-161

MPEG-1 Audio

- erster internationaler Standard für (Hifi-) Audio-Kompression
- entwickelt im Rahmen von MPEG-1
- bei Bitraten bis 1.5 Mb/s für Video mit Audio (Video-CD)
- verabschiedet Ende 1992 als ISO/IEC 11172-3 (1993)
- definiert das Format des Audio-Bitstroms
- und die Funktion des Decoders
- aber nicht den Encoder: Raum für spätere Verbesserungen
- Format geeignet für Sprache und Musik
- keine Quellenkodierung / keine Annahmen über Herkunft des Signals
- statt dessen psychoakustisches Modell
- Ausnutzen von Maskierungseffekten
- drei Varianten (layers), jeweils mono / stereo / joint stereo
- Sampleraten 32, 44.1, 48 KHz, Bitraten 32 .. 224 kb/s/channel

Medientechnik | WS 2001 | 18.204

MPEG-1 Audio: Varianten

- drei Varianten, Bandbreite gesamt bis 1.5 Mb/s für Video+Audio
- zwei psychoakustische Modelle
- Encoder dürfen eigene psychoakustische Modelle verwenden

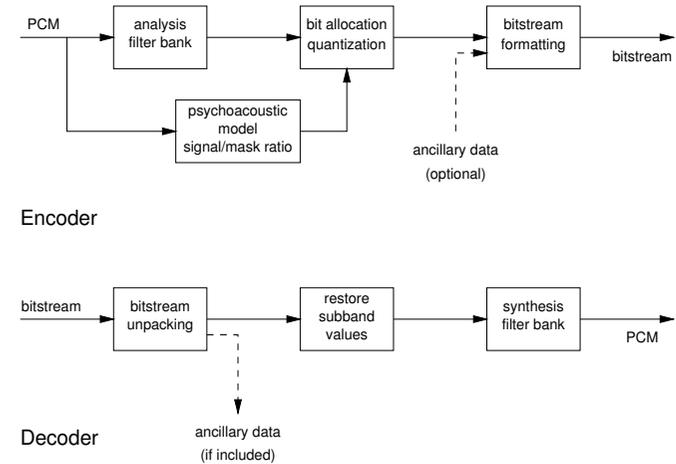
Layer 1: einfachstes Verfahren
für Bitraten höher als 128 kb/s pro Kanal
eingesetzt für Philips DCC: 384 kb/s stereo

Layer 2: mittlere Komplexität
für Bitraten von 128 kb/s pro Kanal
eingesetzt für CD-I und Video-CD

Layer 3: bessere Qualität, aber deutlich komplexer
ab 64 kb/s pro Kanal, gute Qualität ab 128 kb/s
"MP3", ISDN-Übertragungen

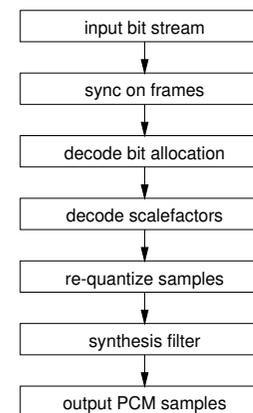
Medientechnik | WS 2001 | 18.204

MPEG-1: Blockschaltbild (grob)



Medientechnik | WS 2001 | 18.204

MPEG-1: Decoder für Layer 1 und 2



- Layer 3 Decoder deutlich komplexer (siehe ISO 11172-3 S.40)

Medientechnik | WS 2001 | 18.204

MPEG-1: Bitstrom-Header

```

audio sequence()
{
  while (true)
  {
    frame()
  }
}

header()
{
  syncword      12 bits  bslbf
  ID            1 bit   bslbf
  layer        2 bits  bslbf
  protection_bit 1 bit  bslbf
  bitrate_index 4 bits  bslbf
  sampling_frequency 2 bits bslbf
  padding_bit   1 bit  bslbf
  private_bit   1 bit  bslbf
  mode         2 bits  bslbf
  mode_extension 2 bits bslbf
  copyright    1 bit  bslbf
  original/home 1 bit  bslbf
  emphasis     2 bits  bslbf
}

frame()
{
  header()
  error_check()
  audio_data()
  ancillary_data()
}

```

- bslbf: bit string, left bit first
- Synchronisation über Bitmuster '1111 1111 1111'

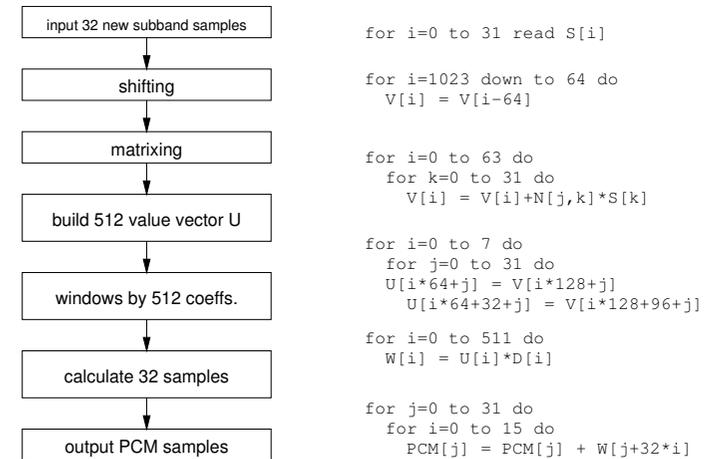
MPEG-1: Bitstrom-Daten, Layer 1

```

audio_data()
{
  if (mode==single_channel)
  {
    ...
  }
  if (mode==stereo) || (mode==dual_channel)
  {
    for (sb=0; sb<32; sb++)
    for (ch=0; ch<2; ch++)
      allocation[ch][sb] 4 bits  bsmsbf
    for (sb=0; sb<32; sb++)
    for (ch=0; ch<2; ch++)
      if (allocation[ch][sb]!=0)
        scalefactor[ch][sb] 6 bits  uimsbf
    for (s=0; s<12; s++)
    for (sb=0; sb<32; sb++)
    for (ch=0; ch<2; ch++)
      if (allocation[ch][sb]!=0)
        sample[ch][sb][s] 2..15 bits  uimsbf
  }
  if (mode==intensity_stereo)
  {
    ...
  }
}

```

MPEG-1: Synthesis Subband Filter



MPEG-1: Zahldarstellung

- möglichst kompakte Zahldarstellung gesucht
- also mit datenabhängiger Genauigkeit
- aber mit hohem Dynamikbereich

=> MPEG-1 Bitstrom verwendet spezielles Gleitkommaformat:

- gemeinsamer Skalenfaktor (Exponent) für eine Anzahl von Samples
- variable Anzahl nb von Nachkommastellen
- Auswahl der jeweiligen Stellenanzahl über Codetabellen
Mantisse hat nb Bits, Scalefactor aus Tabelle
- Layer 3 zusätzlich Huffman-Codierung

$$\text{wert} = \text{scalefactor} * (2^{**}\text{nb} / (2^{**}\text{nb}-1)) * (\text{mantisse} + 2^{**}(-\text{nb}+1))$$

MPEG-1: Subband-Kodierung, Filterbank

"subband-Kodierung":

- Filterbank zerlegt das Eingangssignal
- in mehrere (gleich breite) Frequenzbänder

Idee:

- Gesamtenergie des Signals verteilt sich auf die Bänder
- und zwar meistens ungleichmäßig
- wenige Bänder mit viel Energie: präzise kodieren
- viele Bänder mit wenig Energie: mit wenig Bits kodieren
- Maskierung und Hörschwelle: kleine Koeffizienten auf Null setzen
- inverse Filterbank setzt das Signal (fast) perfekt wieder zusammen

MPEG-1: Filterbank, Layer 1 und 2

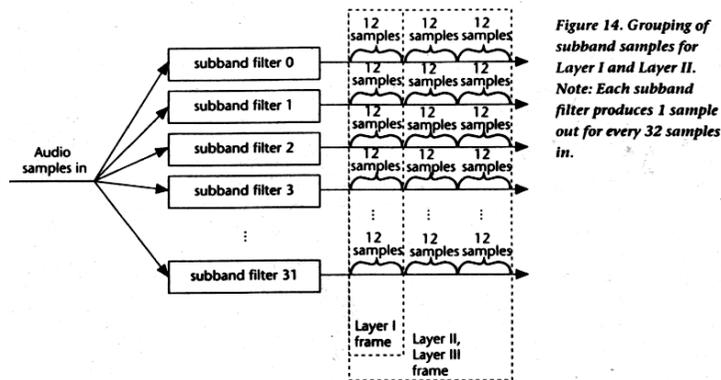


Figure 14. Grouping of subband samples for Layer I and Layer II. Note: Each subband filter produces 1 sample out for every 32 samples in.

- 32 Filterbänder, gleiche Frequenzbreite (je 20/32 KHz)
- Layer 1: 384 samples/frame input, 32x12 samples output
- Layer 2: 1152 samples/frame, 32x 3groups x 12 samples

MPEG-1: Filterbank, Layer 1 und 2

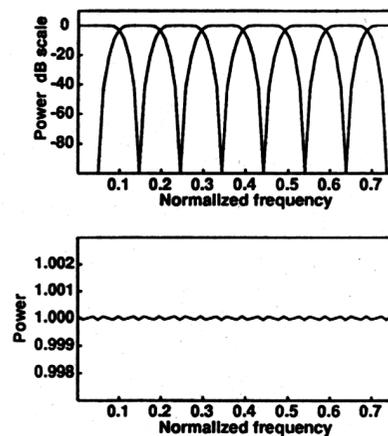
$$s_t[i] = \sum_{k=0}^{63} \sum_{j=0}^7 M[i][k] * (C[k+64j] * x[k+64j]) \quad (1)$$

where:

i is the subband index and ranges from 0 to 31,
 $s_t[i]$ is the filter output sample for subband *i* at time *t*, where *t* is an integer multiple of 32 audio sample intervals,
 $C[n]$ is one of 512 coefficients of the analysis window defined in the standard,
 $x[n]$ is an audio input sample read from a 512 sample buffer, and
 $M[i][k] = \cos\left(\frac{(2^i+1) * (k-16) * \pi}{64}\right)$ are the analysis matrix coefficients.

- Berechnung der Subband-Samples $St[i]$
- $M[i][k]$ ist um *k* bzw. *j* verschoben: "polyphase" Filterbank
- effizientere Berechnung via FFT möglich
- Layer 3 verwendet aufwendigere Filterbank

MPEG-1: Filterbank



Filter überlappen stark

- aber Überlagerung führt zu fast-idealem Frequenzgang

Fig. 1. Top: frequency response of the first 8 subband filters. Bottom: frequency response of the analysis filter bank. The frequency has been normalized to the Nyquist frequency 2π .

MPEG-1: Filterbank Aliasing

- einzelne Filterbänder überlappen
- Signale nahe der Übergänge werden in zwei Bänder kodiert
- Alias-Effekte möglich
- Layer-3 verwendet nachgeschaltete MDCT-Transformation
- zur Verbesserung der Frequenzauflösung
- mit Versuch der Alias-Reduzierung

Frequenzgruppen, Bark-Skala

Tabelle 9.1: Frequenzgruppen nach Zwicker 1982

z/Bark	f_u /Hz	f_o /Hz	Δf_G /Hz	f_m /Hz
0	0	100	100	50
1	100	200	100	150
2	200	300	100	250
3	300	400	100	350
4	400	510	110	450
5	510	630	120	570
6	630	770	140	700
7	770	920	150	840
8	920	1080	160	1000
9	1080	1270	190	1170
10	1270	1480	210	1370
11	1480	1720	240	1600
12	1720	2000	280	1850
13	2000	2320	320	2150
14	2320	2700	380	2500
15	2700	3150	450	2900
16	3150	3700	550	3400
17	3700	4400	700	4000
18	4400	5300	900	4800
19	5300	6400	1100	5800
20	6400	7700	1300	7000
21	7700	9500	1800	8500
22	9500	12000	2500	10500
23	12000	15500	3500	13500
24	15500			

Tonhöhenwahrnehmung?

- Messungen
- vgl. Tuningkurven
- tiefe Frequenzen: ca. 100 Hz Bandbreite
- hohe Frequenzen: log. Breite der Gruppen
- => "Bark" Frequenzskala
- Ausgangspunkt für psychoakustische Modelle

[Zoelzer]

MPEG-1: Maskierung

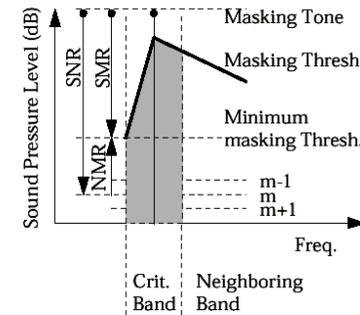
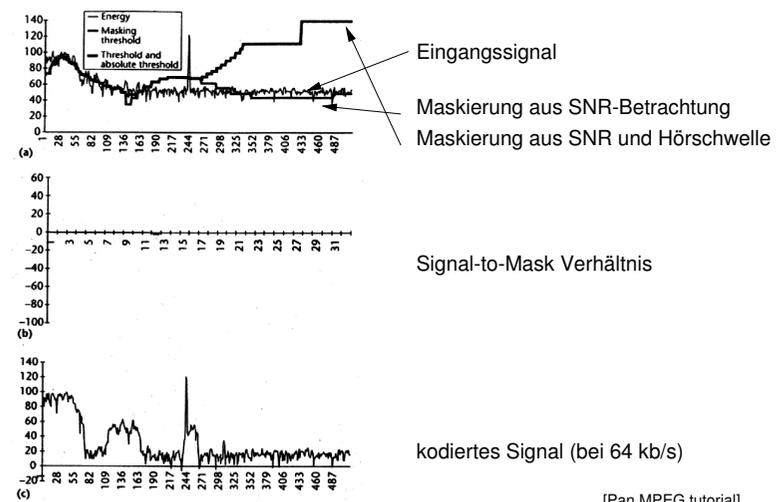


Fig. 6. Schematic Representation of Simultaneous Masking (after [30])

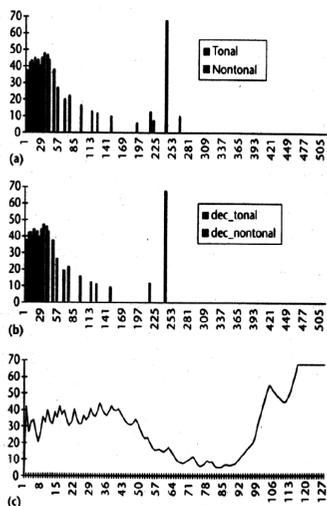
- lautester Ton im Frequenzband maskiert benachbarte
- Maskierung linear darstellbar, wenn Frequenz in Bark gemessen wird
- Töne leiser als der Masking-Threshold nicht wahrnehmbar
- SNR / SMR / NMR Verhältnisse

MPEG-1: psychoakustisches Modell 2



[Pan MPEG tutorial]

MPEG-1: psychoakustisches Modell 1



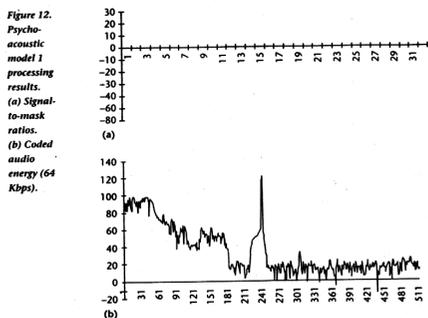
tonale und nicht-tonale Komponenten

nach der Dezimierung

masking thresholds

[Pan MPEG tutorial]

MPEG-1: psychoakustisches Modell 1



- Ergebnis der Kodierung nach Modell 1
- teilweise deutlich schlechter als Modell 2

[Pan MPEG tutorial]

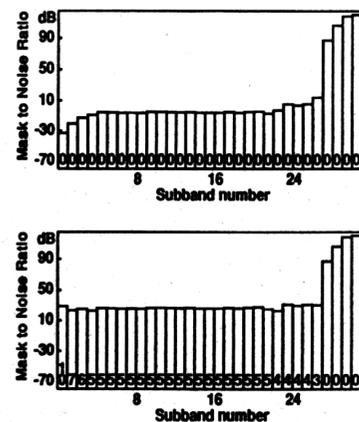
MPEG-1: bit allocation

- psychoakustisches Modell liefert "signal-to-mask" Verhältnis
- für jedes einzelne Subband

Bit-Allokation Algorithmus:

- Berechnung von MNR und SNR (mask-noise, signal-noise-ratio)
- wieviele Bits stehen zur Verfügung? Bitrate minus Sync. etc.
- Aufgabe: Bitzuweisung, so daß das minimale MNR maximal
 - für die Skalierungsfaktoren
 - und die Koeffizienten aller einzelnen Subbänder
 - Details für die Layer 1,2,3 leicht unterschiedlich
 - iterativ: berechne MNR und SNR für jedes Subband
 - weise dem Subband mit der schlechtesten MNR Bits zu
 - solange noch Bits zur Verfügung stehen

MPEG-1: bit allocation



Mask-to-noise ratio

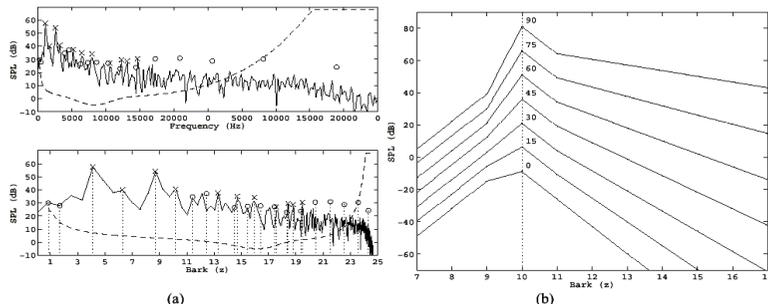
- beim Start der bit-allokation

- nach der bit-allokation
- Anzahl bits / Band

Fig. 8. Mask to noise ratio at the beginning (top) and end (bottom) of the bit allocation algorithm. The number of bits assigned to each subband sample appears at the base of each bar.

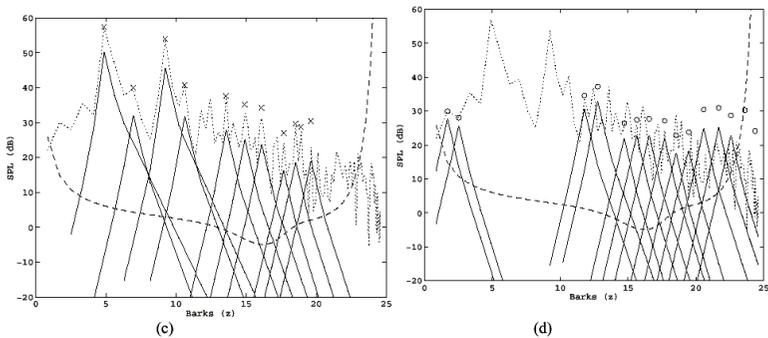
[Shlien MPEG guide]

MPEG-1: model 1, tonal / noise maskers



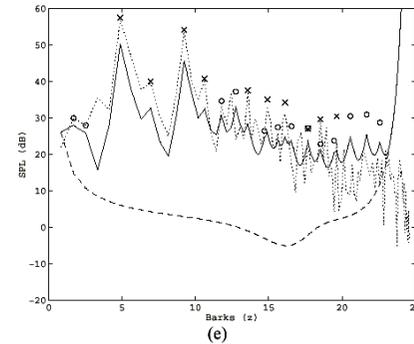
(a) Step 1: Obtain PSD, express in dB SPL. Top panel gives linear frequency scale, bottom panel gives Bark frequency scale. Absolute threshold superimposed. Step 2: Tonal maskers identified and denoted by 'X' symbol; Noise maskers identified and denoted by 'O' symbol. (b) Collection of prototype spreading functions (Eq. 28) shown with level as the parameter. These illustrate the incorporation of excitation pattern level-dependence into the model. Note that the prototype functions are defined to be piecewise linear on the Bark scale. These will be associated with maskers in steps 3, 4.

MPEG-1: model 1, masking and spreading



(c) Steps 3,4: Spreading functions are associated with each of the individual tonal maskers satisfying the rules outlined in the text. Note that the Signal-to-Mask Ratio (SMR) at the peak is close to the widely accepted tonal value of 14.5 dB. (d) Spreading functions are associated with each of the individual noise maskers that were extracted after the tonal maskers had been eliminated from consideration, as described in the text. Note that the peak SMR is close to the widely accepted noise-masker value of 5 dB.

MPEG-1: model 1, masking and spreading



(e) Step 5: A global masking threshold is obtained by combining the individual thresholds as described in the text. The maximum of the global threshold and the absolute threshold are taken at each point in frequency to be the final global threshold. The figure clearly shows that some portions of the input spectrum require SNRs of better than 20 dB to prevent audible distortion, while other spectral regions require less than 3 dB SNR. In fact, some high-frequency portions of the signal spectrum are masked and therefore perceptually irrelevant, ultimately requiring no bits for quantization without the introduction of artifacts.

MPEG-1: Bitraten für Layer 1 .. 3

MPEG-1 Bitraten (mono/stereo/joint stereo):

index	Layer I		Layer II		Layer III	
'0000'	free	format	free	format	free	format
'0001'	32	kbit/s	32	kbit/s	32	kbit/s
'0010'	64	kbit/s	48	kbit/s	40	kbit/s
'0011'	96	kbit/s	56	kbit/s	48	kbit/s
'0100'	128	kbit/s	64	kbit/s	56	kbit/s
'0101'	160	kbit/s	80	kbit/s	64	kbit/s
'0110'	192	kbit/s	96	kbit/s	80	kbit/s
'0111'	224	kbit/s	112	kbit/s	96	kbit/s
'1000'	256	kbit/s	128	kbit/s	112	kbit/s
'1001'	288	kbit/s	160	kbit/s	128	kbit/s
'1010'	320	kbit/s	192	kbit/s	160	kbit/s
'1011'	352	kbit/s	224	kbit/s	192	kbit/s
'1100'	384	kbit/s	256	kbit/s	224	kbit/s
'1101'	416	kbit/s	320	kbit/s	256	kbit/s
'1110'	448	kbit/s	384	kbit/s	320	kbit/s

- 32 .. 448 kb/s

MPEG-1: bitstream format

Header (32)	CRC (0,16)	Bit allocation (128-256)	Scale factors (0-384)	Samples	Ancillary data
-------------	------------	--------------------------	-----------------------	---------	----------------

(a)

Header (32)	CRC (0,16)	Bit allocation (26-188)	SCFSI (0-60)	Scale factors (0-1080)	Samples	Ancillary data
-------------	------------	-------------------------	--------------	------------------------	---------	----------------

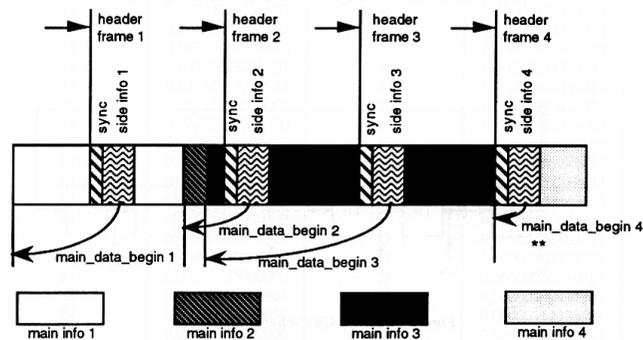
(b)

Header (32)	CRC (0,16)	Side information (136, 256)	Main data; not necessarily linked to this frame. See Figure 18.
-------------	------------	-----------------------------	---

(c)

- Header mit speziellem Sync-Muster, dann CRC
- Layer 3 erlaubt "borrowing" zwischen Frames

MPEG-1: bitstream format

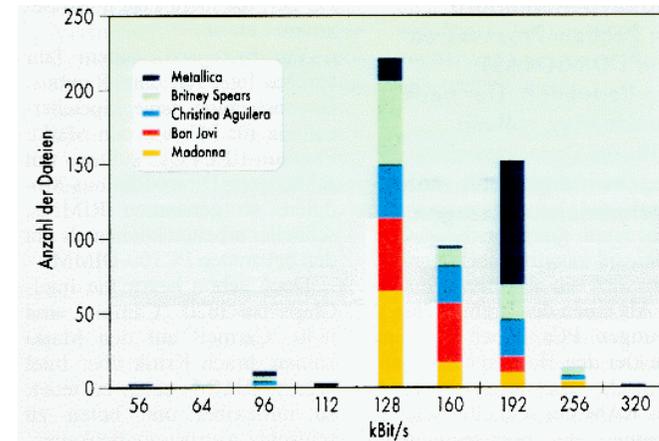


***) main_data_begin 4 == 0: This signifies that main data starts directly after the side information for frame 4. This is the lower limit for main_data_begin; main_data cannot start later than this point. Note that data bytes used by "sync" and "side info" are not counted by the main_data_begin pointer.

Note: 'info' means information

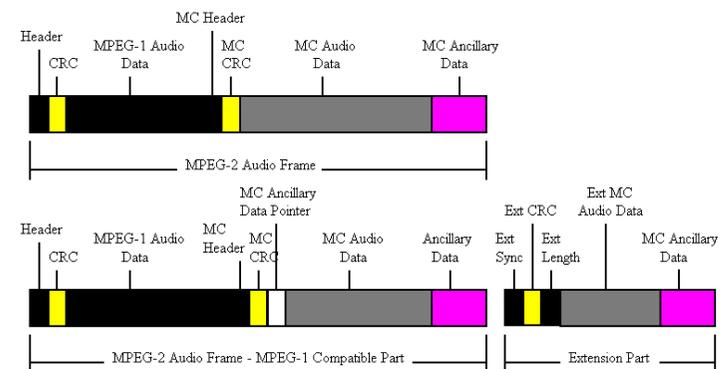
Figure A.7.b -- Layer III bitstream organization with peak demand at main info 3 and small demand at main info 2.

MPEG-1: gängige Bitraten



- Histogramm der Bitraten (MP3)
- Suche per Napster, aus [c't 2x/2000]

MPEG-2: multichannel bitstream



- MPEG-1 kompatibles Stereosignal, plus Mehrkanal-Erweiterung

Figure A.2 Structure of the ISO 13818-3 Layer II multichannel extension, using the ISO/IEC 11172-3 compatible bit stream as well as the extension bit stream

MPEG-1: pre-echos

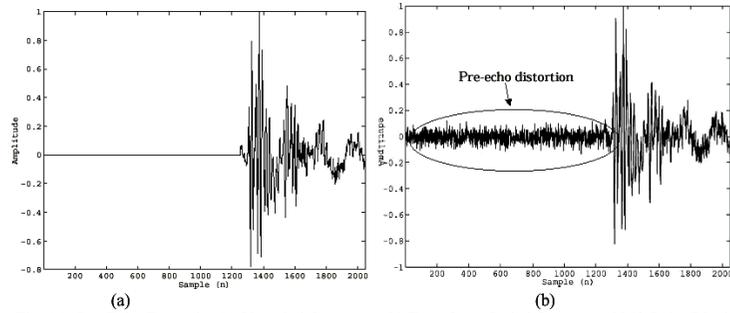
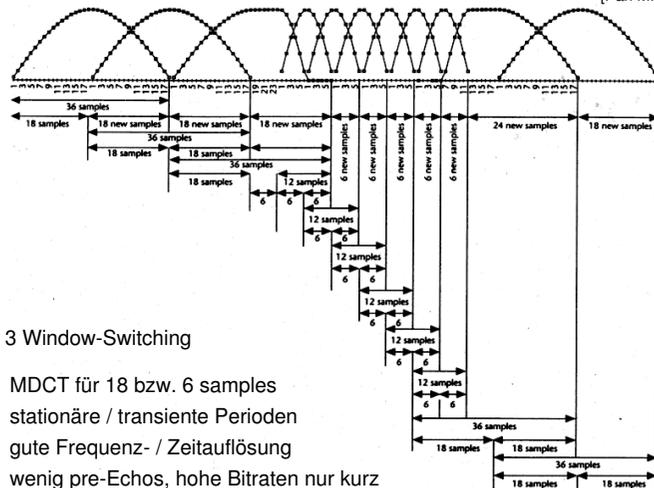


Fig. 14. Pre-Echo Example: (a) Uncoded Castanets. (b) Transform Coded Castanets, 2048-Point Block Size [Painter Spanias]

- Problem mit plötzlichen lauten Signalen nach einer Blockgrenze
 - vorheriger Block hat sehr wenig Bits kodiert
 - Rauschen des lauten Signals wird nicht mehr maskiert
- => Situation erkennen und behandeln (kürzere MDCT Blöcke)

MPEG-1: window-switching

[Pan MPEG tutorial]



Layer 3 Window-Switching

- MDCT für 18 bzw. 6 samples
- stationäre / transiente Perioden
- gute Frequenz- / Zeitauflösung
- wenig pre-Echos, hohe Bitraten nur kurz

MPEG-1: temporal noise-shaping

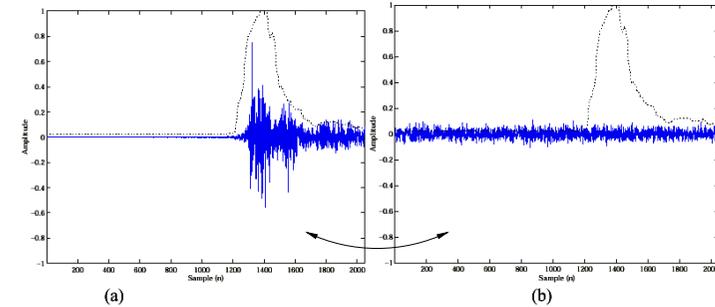


Fig. 17. Temporal Noise Shaping Example Showing Quantization Noise and the Input Signal Energy Envelope for Castanets: (a) Without TNS, and (b) With TNS [Painter Spanias, a) b) vertauscht]

weitere Verbesserung durch TNS:

- Verlagerung des Rauschens in den Bereich mit höchster Maskierung
- wird für MPEG-2 AAC eingesetzt

MPEG-2: BC und NBC (AAC)

"backwards compatible":

- ähnliches Format wie MPEG-1 layer 3
- aber mit 3/2-Mehrkanal-Erweiterung
- abwärtskompatibel (für die 2/0 Stereosignale)
- ungeeignet für 3/2 mit Bitraten unter 640 kb/s

"non backwards compatible" / "advanced audio coding"

- gleiches Konzept wie MPEG-1 layer 3
- aber viele Verbesserungen, mit Ziel geringerer Bitrate
- viele neue Details, bessere Filterbank, bessere Zeitauflösung
- temporal noise shaping, . . .
- zusätzliche Bitraten bis herunter zu 16 kb/s/channel

MPEG-2: AAC Eigenschaften

MPEG-2 AAC (ISO 13818-7, 1997):

- Mehrkanal-Audio (3/2) bei Bitraten bis 384 kb/s
- transparente Kodierung (Hörtests) ab 320 kb/s
- mehrere "Profile": main / low / scalable sample rate profile
- bessere Filterbank als MP3:
- Frequenzauflösung 23 Hz (75Hz), Zeitauflösung 2.6 ms
- keine Hybrid-Filterbank, sondern nur MDCT
- MDCT window shape switching (stationary / transient signals)
- time prediction of transform coefficients
- more efficient Huffman code tables
- temporal noise shaping
- scalability and complexity management tools
- eingesetzt u.a. für DVD und Liquid Audio

Medientechnik | WS 2001 | 18.204

MPEG-2: AAC-Decoder Performance

"revised report on complexity of MPEG-2 AAC tools2"

"ISO/IEC JTC1/SC229/WD11/N2957 MPEG99"

welche Hardware-Anforderungen für AAC-Dekodierung?!

Decoder	Complexity
2-channel MP software decoder	40% Pentium/133
2-channel low complexity sw decoder	25% Pentium/133
5-channel MP hardware decoder	90 mm ² 0.5 µm CMOS

Huffman decode	13.3 %
inv. quantization, scaling	1.7 %
temporal noise shaping	6.6 %
prediction	43.2 %
IMDCT	19.4 %
coupling channel	15.5 %

Medientechnik | WS 2001 | 18.204

MPEG-2: und Nachfolger

- entwickelt für geringe Bitraten
- Hörtests bescheinigen (sehr) gute Qualität
- auch für kritisches Audio-Material
- aber nicht unbedingt besser als MP3
- siehe MPEG-2 quality report Hörtests / Vergleich mehrerer Algorithmen
- MPEG-4 (ISO/IEC 14496, 1998) enthält MPEG-2
- zusätzlich "objekt-orientierte" Algorithmen
- für Bitraten von 64 kb/s bis hinunter zu 200 b/s (!)
- für Sprachkodierung / -synthese
- für Musikkodierung / -synthese (a la MIDI)

Medientechnik | WS 2001 | 18.204

MPEG-1: LAME

"Lame ain't an MPEG encoder"

- MP3-Encoder unter GPL Lizenz, im Quellcode verfügbar
- ursprünglich als Patch gegen den ISO Referenzcode
- derzeit als völlig neuentwickeltes Programm
- Decoder basiert auf mpeglib (www.mpg123.de)
- mit eigenem psychoakustischem Modell
- Qualität vergleichbar mit dem FhG-Encoder
- ideal als Grundlage für Eigenentwicklungen
- unterstützt MPEG-1 layer 1 bis 3
- recht schnelle Kodierung
- www.sulaco.com/mp3 (aktuell ist Lame 3.87)

Medientechnik | WS 2001 | 18.204

LAME: psychoakustisches Modell

```

/* psymodel.c
 * Copyright (c) 1999 Mark Taylor
 *
 * compute FFTs
 * FFT data for mid and side channel is derived from L & R
 * compute energies
 * compute unpredictability of first six spectral lines
 * compute unpredictability of next 200 spectral lines
 *
 * Calculate the energy and the unpredictability in the threshold
 * calculation partitions
 *
 * convolve the partitioned energy and unpredictability
 * with the spreading function, s3_l[b][k]
 *
 * calculate the tonality of each threshold calculation partition
 * calculate the SNR in each threshold calculation partition
 * tonality = -0.299 - .43*log(ctb/ecb);
 * tonality = 0:      use NMT (lots of masking)
 * tonality = 1:      use TMN (little masking)
 *
 * at this point, tbb represents the amount the spreading function
 * will be reduced. The smaller the value, the less masking.
 *
 * long block pre-echo control.
 * compute masking thresholds for both short and long blocks
 * threshold calculation for short blocks
 * compute M/S thresholds from Johnston & Ferreira 1992 ICASSP paper
 * determine final block type
 *
 * compute side_energy / (side+mid)_energy
 * 0 = no energy in side channel
 * .5 = half of total energy in side channel
 *
 * Read long block data for converting threshold calculation
 * Read short block data for converting threshold calculation
 *
 * compute which partition bands are in which scalefactor bands
 * compute bark value and ATH of each critical band
 *
 * Now compute the spreading function, s[j][i], the value of the spread-
 * ing function, centered at band j, for band i, store for later use
 * setup stereo demasking thresholds
 */

```

Medientechnik | WS 2001 | 18.204

LAME: Status, Performance

- Echtzeit-Kodierung ist möglich
- auch bei hohen Bitraten / guter Qualität

			vs. Echtzeit
lame --preset studio	256 kb/s	3X	
lame --preset cd	192 kb/s	3X	
lame --preset fm	112 kb/s	3X	
			auf Celeron/466

- psychoakustisches Modell liegt im Quelltext vor
- weitere Verbesserungen vorgeschlagen und geplant
- aber massive rechtliche Probleme:
- fast alle MPEG-1/2 Verfahren patentiert und lizenzpflichtig
- siehe MPEG-2 AAC Lizenzvertrag (PDF)

Medientechnik | WS 2001 | 18.204

LAME: Presets

Presets are some shortcuts for common settings.
They can be combined with -v if you want VBR MP3s.

```

=====
                phone phon+  lw   sw   fm  voice radio tape  hifi  cd  studio
-----
--resample      8   11   11   11   32   24
--highpass     0.125 0.1
--lowpass       3.4   4   4   4   15   12   15   18   18
--lowpass-width 0   0   0   0   0   0   0   0.9  0.9
--nosshort      yes  yes  no  no  no  yes  no  no  no  no  no
                -mm  -mm  -mm  -mm  -mj  -mm  -mj  -mj  -mj  -ms  -ms
                -h   -h
-b              16   24   24   24  112  56  128  128  160  192  256
-- PLUS WITH -v -----
-V              6   4   3   3   3   4   3   3   2   1   0
-b              8  16  16  16  80  40  96  96  112  128  160
-B              24  32  56  56 256 112 256 256 320 320 320
=====
EXAMPLES:
a) --preset fm
   equal to: -mj -b112 --resample 32 --lowpass 15 --lowpass-width 0
b) -v --preset studio
   equals to: -h -ms -V0 -b160 -B320

```

- feste Bitraten von 16 kb/s (mono) bis 256 kb/s (stereo)
- -B setzt maximal zulässigen Wert bei variabler Bitrate
- -V 0 für beste Qualität, -V 9 für minimale Dateigröße

Medientechnik | WS 2001 | 18.204

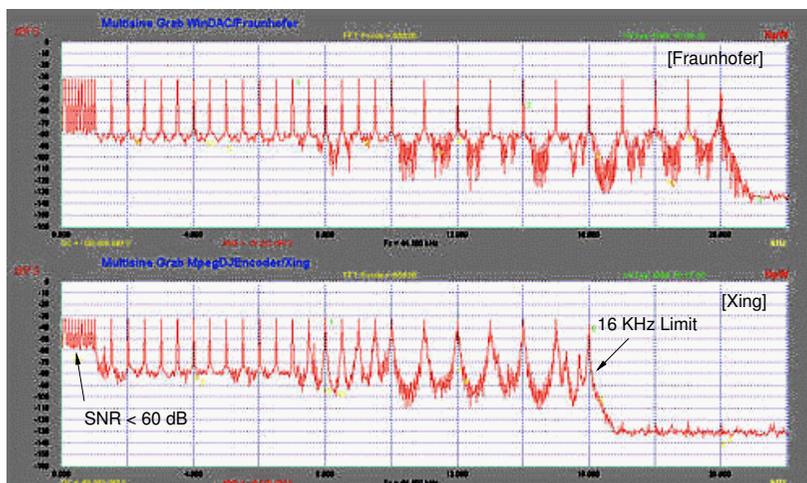
Klangvergleich

	MP3	WMA	RA8	ePAC	AAC	MP+
Modem	o	++	+	++	-	x
Rock	+ 160	+ 160	+ 185	+ 160	o 256	++ 184
Pop	+ 160	++ 128	+ 185	++ 128	+ 192	++ 171
Klassik	+ 160	+ 160	++ 136	+ 160	+ 190	++ 158
Stresstest	+ 320	-- 192	o 360	-- 160	++ 320	++ 296

- Vergleichstest in c't 23/2000 p.152
- Klangqualität bei 128/138 kb/s / Bitrate für CD-nahe Qualität
- MP+ ist Eigenentwicklung der Autoren, "audiophil"

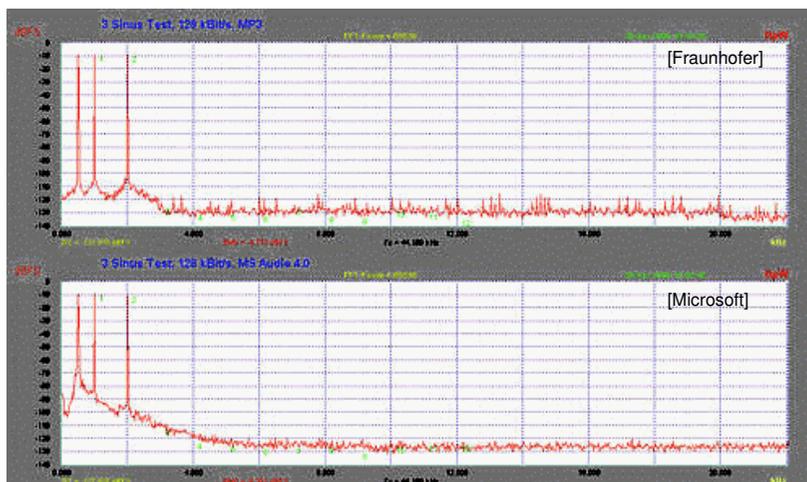
Medientechnik | WS 2001 | 18.204

MP3: zwei Encoder . . .



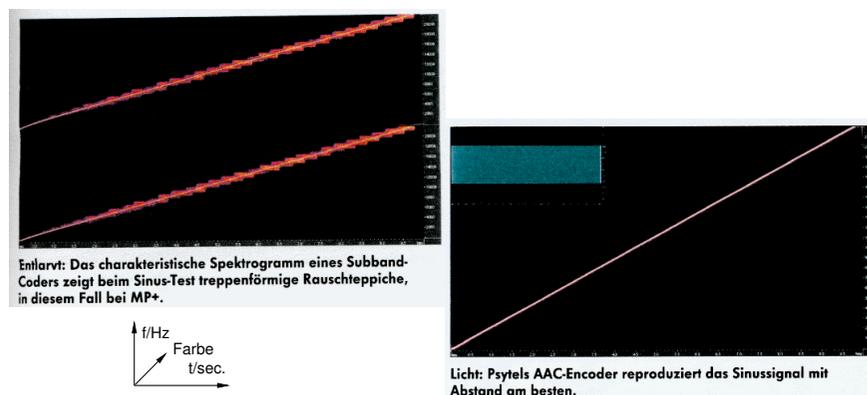
- Eingangssignal mit 44 Sinustönen . . . [ct 21/98 242]

MP3: vs. MS Audio 4.0



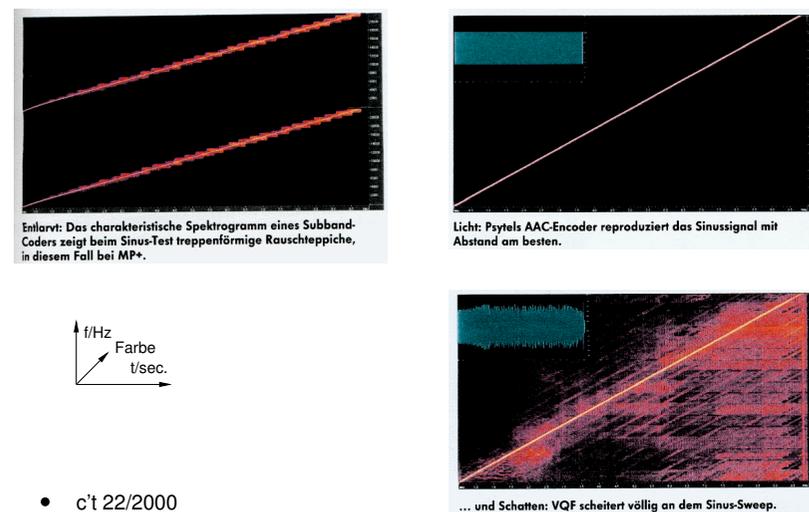
- neuere Versionen von WMA sind besser [ct 21/98 242]

MPEG-1: Sinus-Sweep



- Subband-Kodierung zerlegt das Signal in Frequenzbereiche
- Frequenz-Sweep besonders schwer zu kodieren
- Test für MP3, AAC, MP+, VQF in c't 22/2000,152

Sinus-Sweep: MP+, AAC, VCQ



- c't 22/2000