Teilnahme auf ALMA registrieren

BESPRECHUNG ÜBUNG 9

Audiokompression

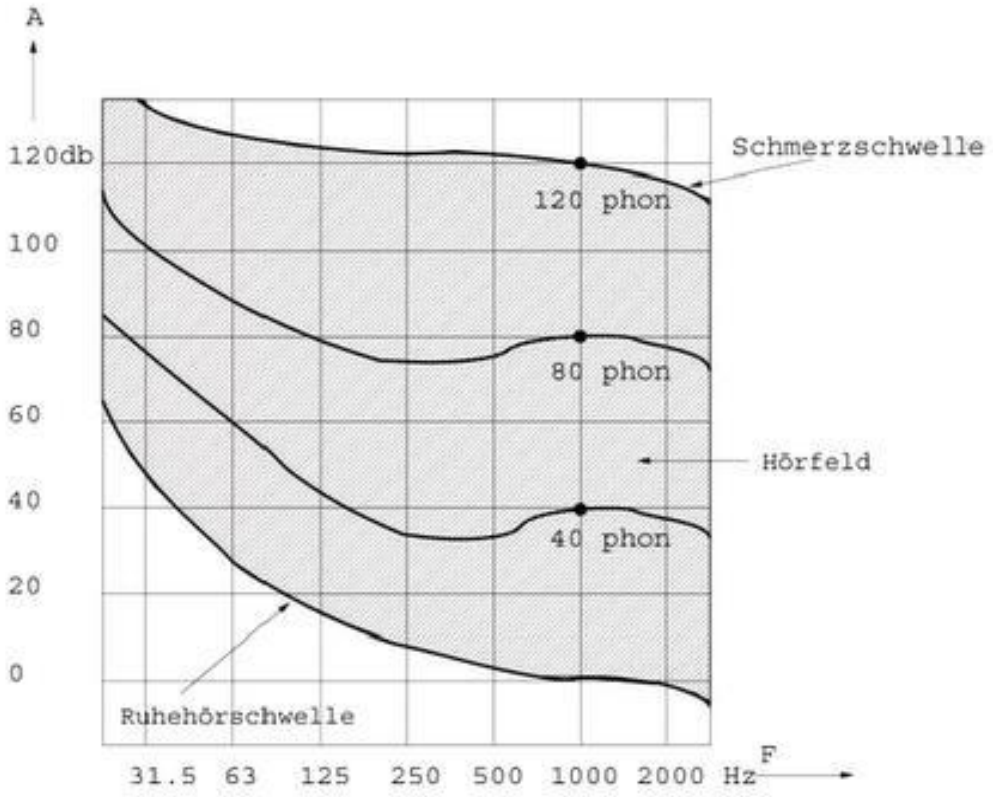
Hürden bei der Audiokompression

- **Kaum Korrelation** in einer Audioaufnahme (in Bildern haben benachbarte Pixel oft ähnliche Farben)
- **Keine erkennbaren Muster** → aus einem Gespräch können wir nicht häufige Wörter nur einmal abspeichern und damit die ursprüngliche Datei rekonstruieren
- **Datenwerte** (Frequenzen) sind annähernd **gleichverteilt** → Entropiekodierungen wie Huffman haben eine geringe Effizienz

⇒Wir müssen uns **Eigenschaften des menschlichen Hörens** (psychoakustisches Modell) zunutze machen, um Daten einsparen zu können

Psychoakustisches Modell

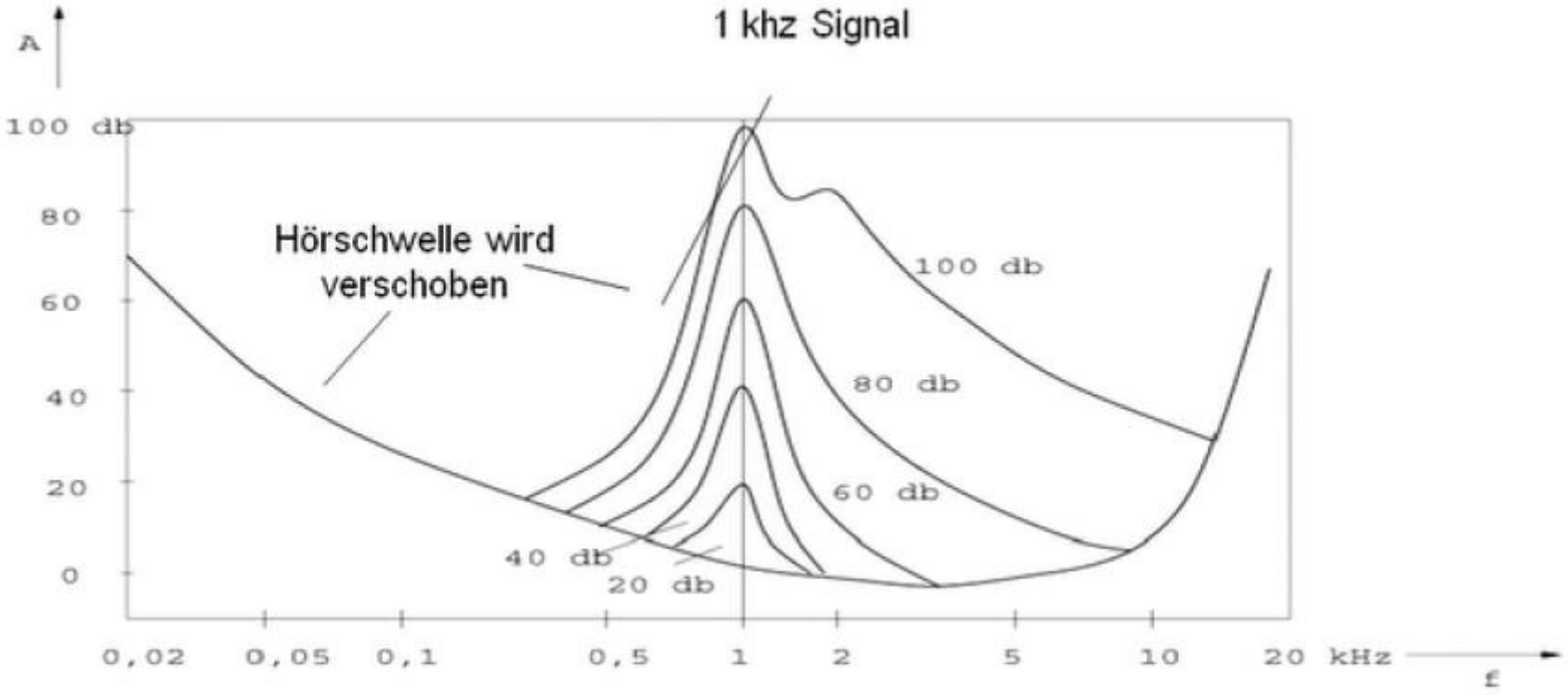
Menschliches Hörfeld: ca. 20-20.000 Hz bei 0 dB – 120 dB



Psychoakustisches Modell

- **Simultane Verdeckung**:

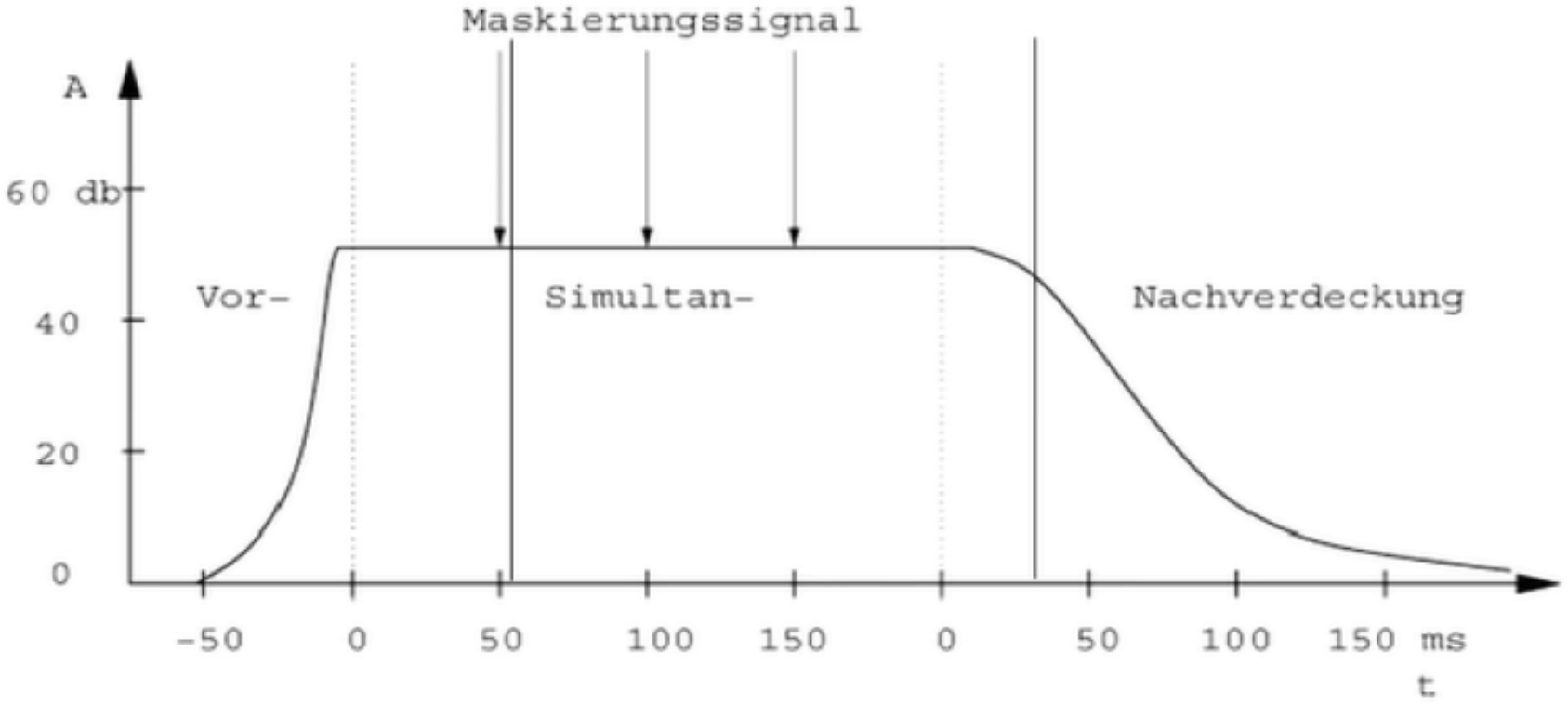
  → Laute Töne maskieren zeitgleiche leisere Töne auch in anderen Frequenzen



Psychoakustisches Modell

- **Temporäre Verdeckung**:

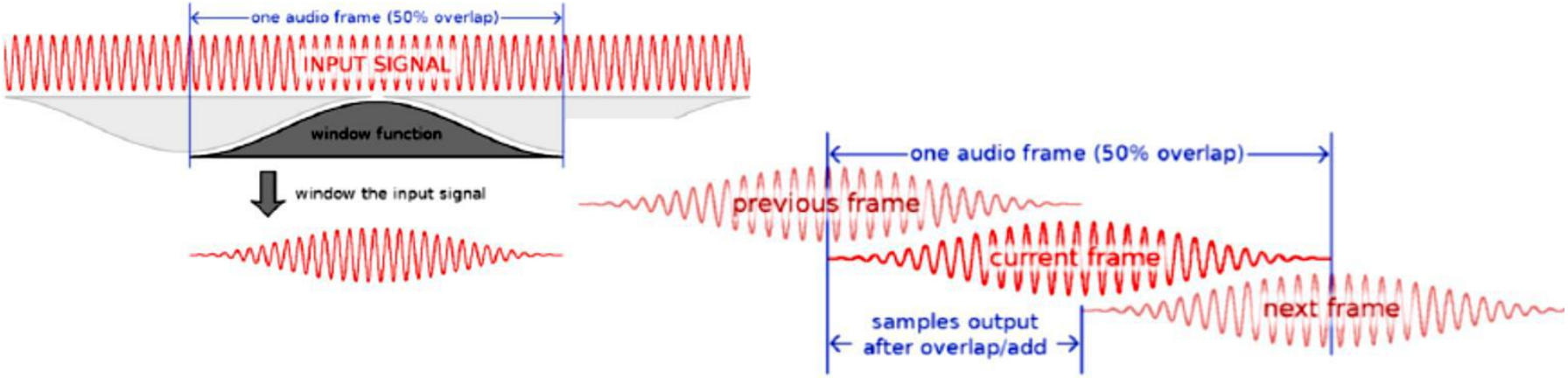
  → Laute Töne maskieren leisere Töne kurz vor und nach dem lauten Ton



Modifizierte Diskrete Cosinus Transformation - MDCT

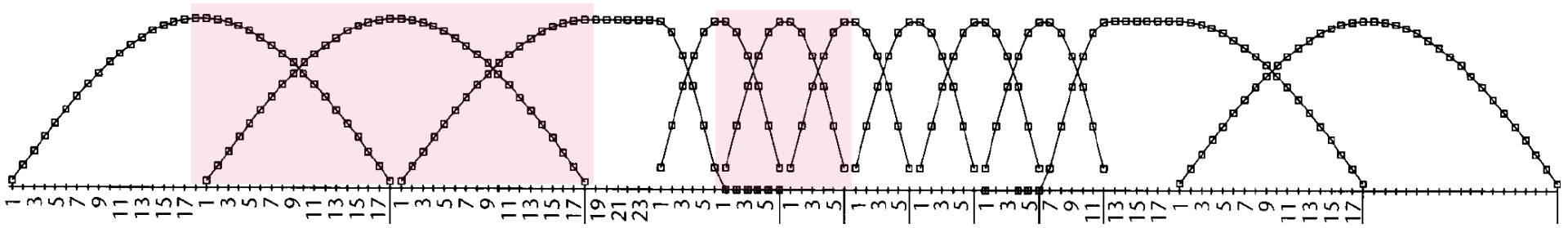
Motivation: Wir erhalten Audiofragmente an Blockgrenzen

- **MDCT**: Sample-Blöcke überschneiden sich zu 50% und werden gewichtet mit einer Fenster-Funktion
  - Vermeidet Artefakte an Blockgrenzen
  - Die doppelte Verwendung jedes Signal-Samples führt zu Time-Domain Aliasing Cancellation (TDCA)



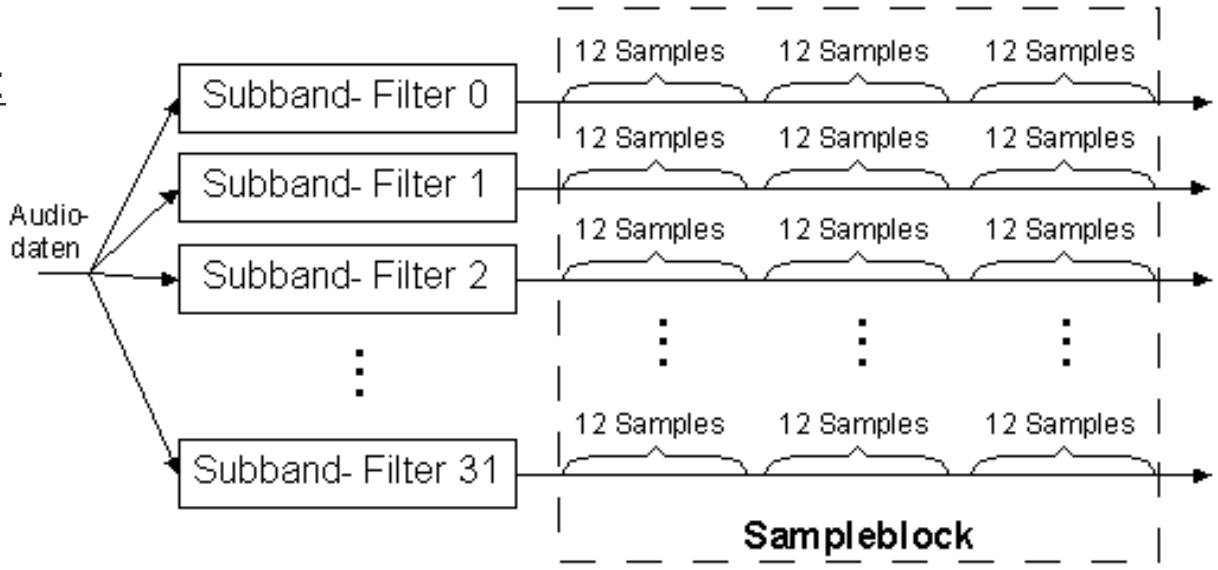
MP3 Kodierung

1. Aufteilung in **32 Frequenzbänder** durch Polyphasenfilterbank (verlustbehaftet)
   → hohe Auflösung (schmale Frequenzbänder) für gut hörbare Frequenzen (300-1200Hz)
2. Aufteilung der Frequenzbänder in **Sampel-Blöcke** für die MDCT mit folgenden Größen:
   i. 12 Sample pro Block mit 6 Grundfrequenzen
   → **gut für schnelle Änderungen**
   ii. 36 Samples pro Block mit 18 Grundfrequenzen
   → **gute Frequenzauflösung**



MP3 Kodierung

Zwischenergebnis:

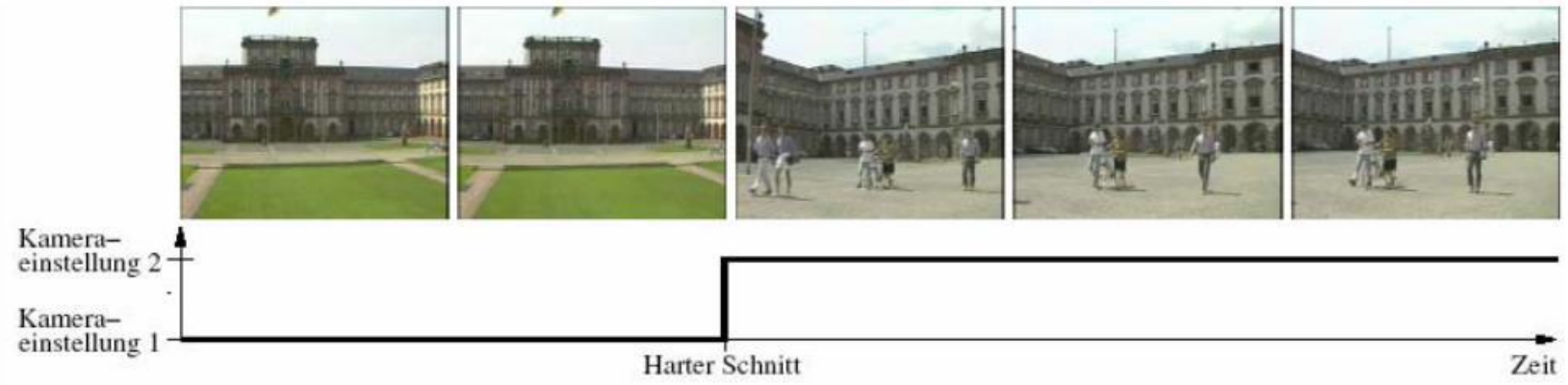


3. **Quantisierung** der Frequenzbänder: maskierte Frequenzen können mit geringerer Auflösung oder gar nicht gespeichert werden
4. Huffman-Kodierung der quantisierten Frequenzen

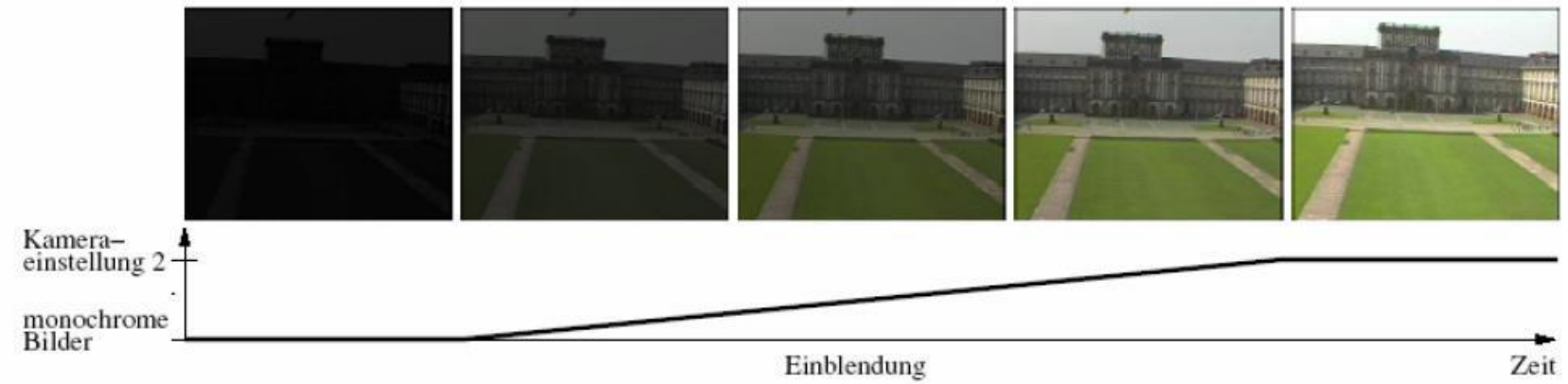
Videoanalyse

Szenen-Übergänge

- **Schnitt (Cut)**

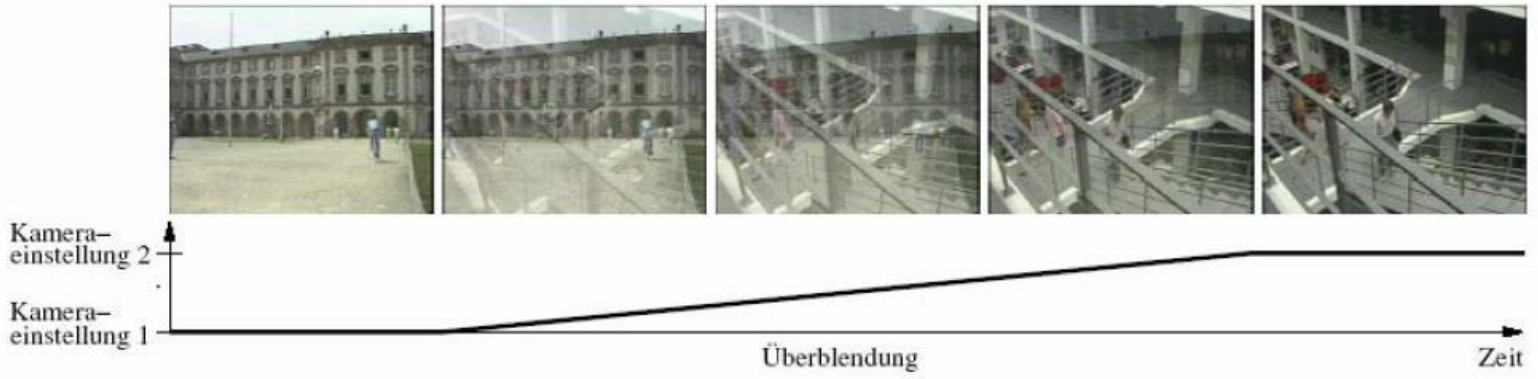


- **Einblendung (Fade in)**



Szenen-Übergänge

- **Überblendung (Dissolve)**



Algorithmen zur Erkennung:

- Pixelbasierte Verfahren
- Analyse von Farbhistogrammen
- Kantenextraktion – Edge Change Ratio (ECR)
- Kantenorientierte Kontrast

Pixelbasierte Schnitterkennung

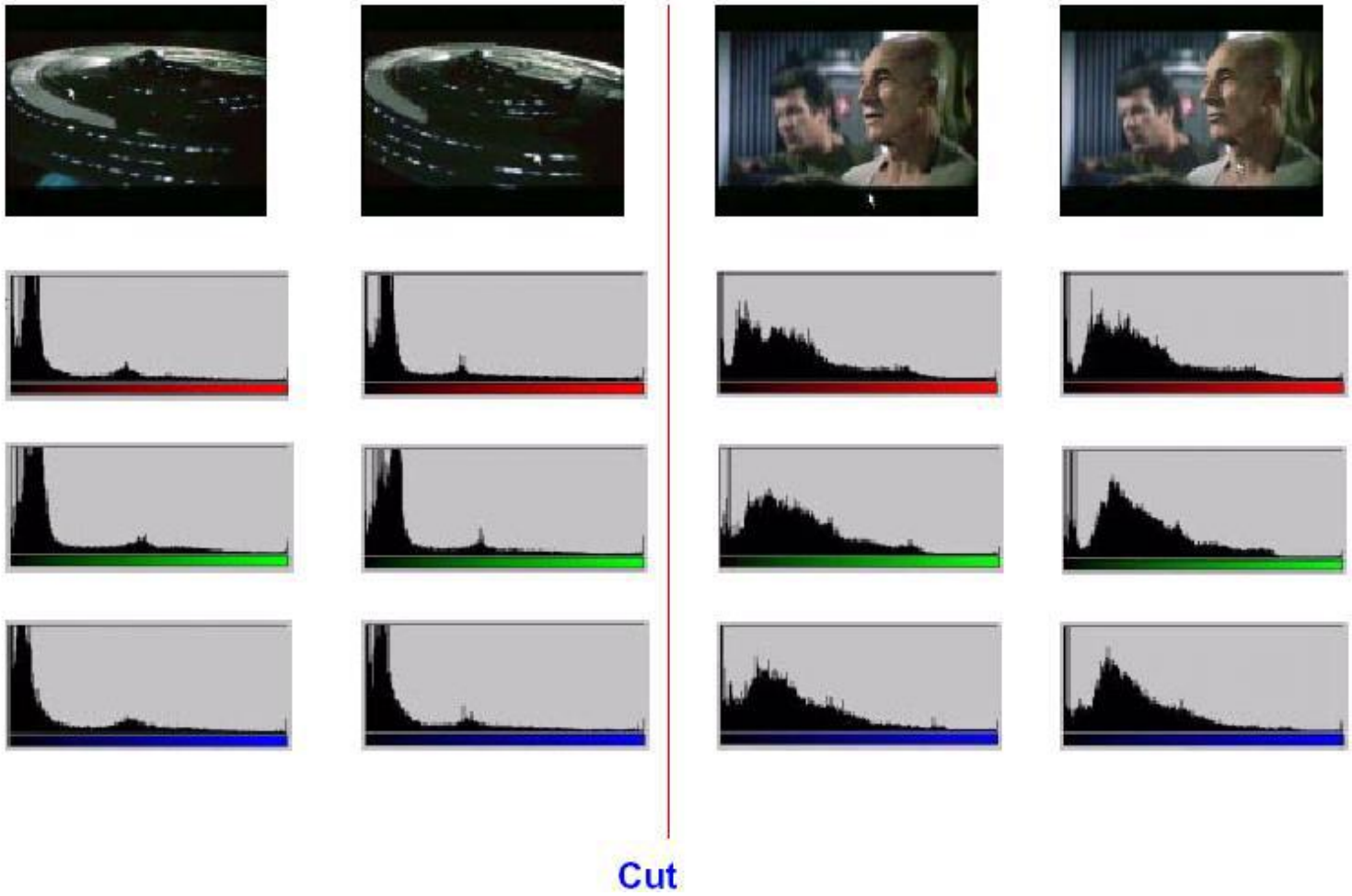
Idee: Wir berechnen das Differenzbild von jeweils zwei Frames und berechnen daraus die absolute Pixeldifferenz.

$$D_{SAD} = \frac{1}{N_x \cdot N_y} \cdot \sum_{x=1}^{N_x} \sum_{y=1}^{N_y} |I_i(x,y) - I_{i-1}(x,y)|$$

Falls $D_{SAD} > Grenzwert \Rightarrow$ Harter Schnitt

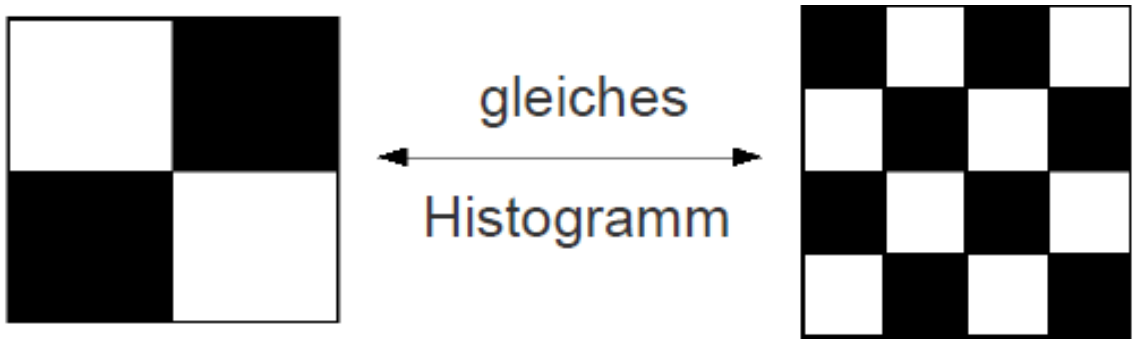
Vorteil: geringe Komplexität, robuste Schnitterkennung
Nachteil: hohe Fehlerrate bei starker Bewegung und Szenen-Überblendungen werden gar nicht erkannt

Histogrammbasierte Schnitterkennung

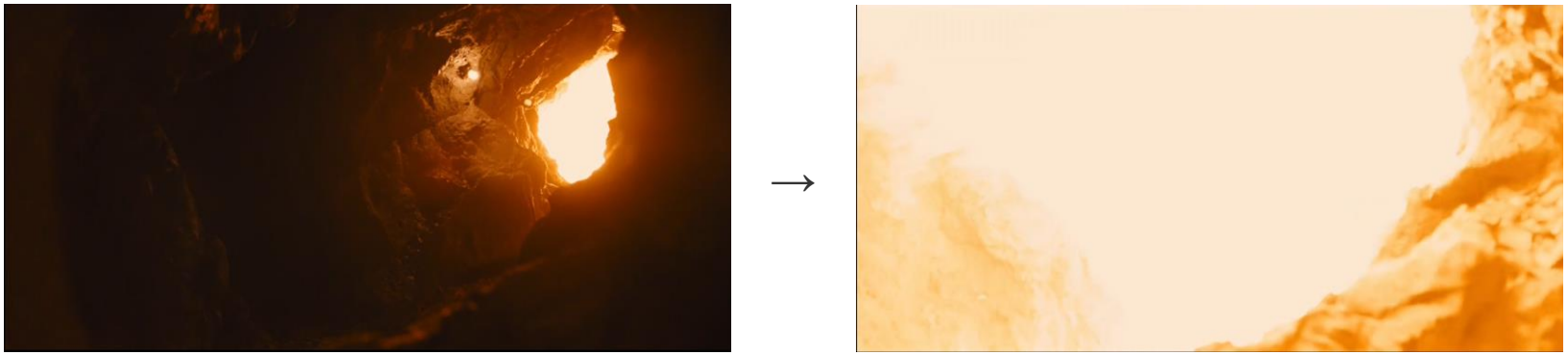


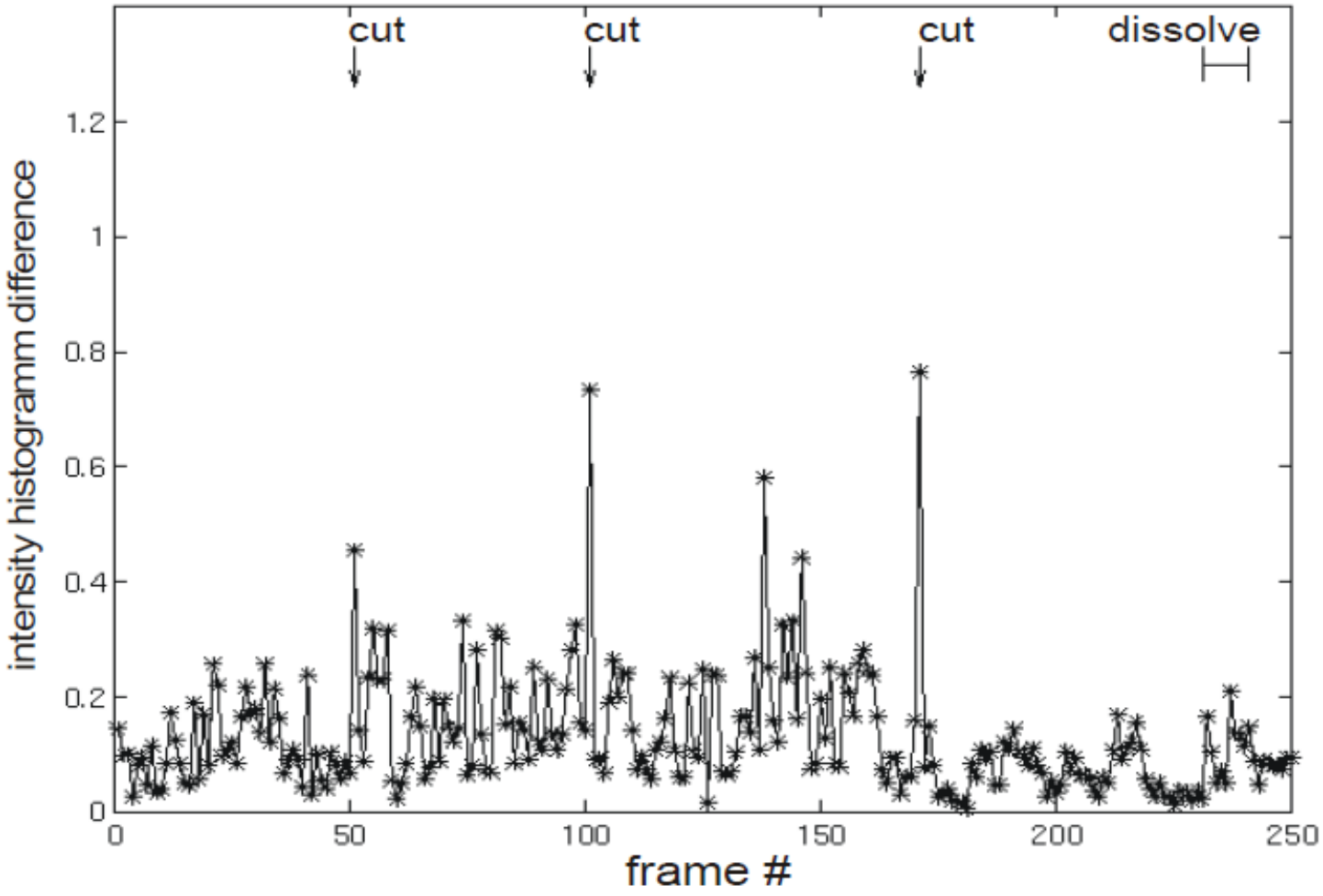
Histogrammbasierte Schnitterkennung

Probleme:



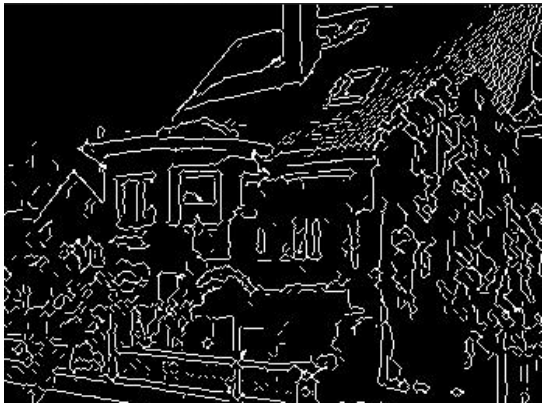
Große Änderung in Action-Szenen:





Kantenbasierte Schnitterkennung

Idee: Wenn sich nur ein Objekt oder die Kamera bewegen, besitzen aufeinander folgende Frame ähnlich viele Kantenpixel.



Vorgehen:

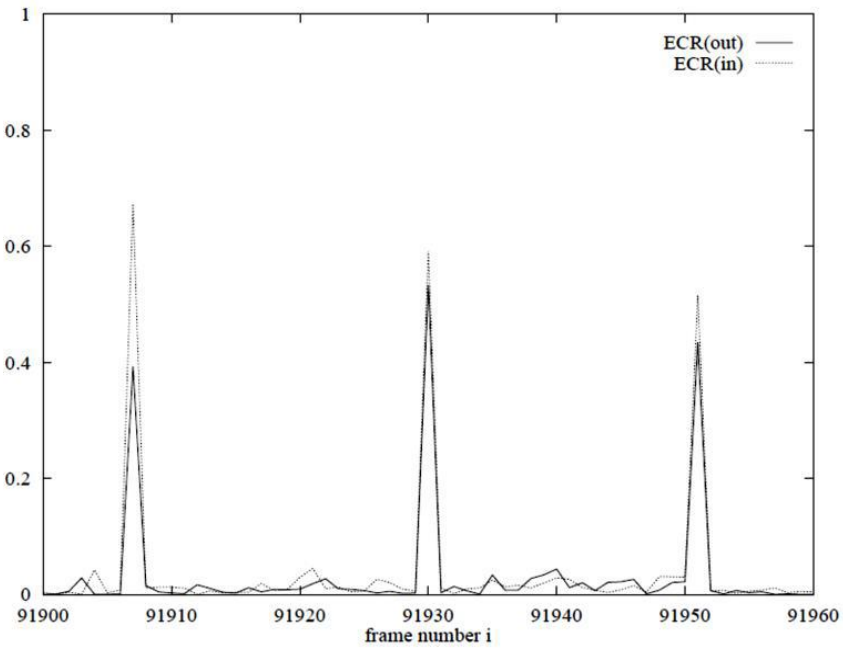
1. Berechnung von Kantenbildern durch z.B. Canny-Algorithmus

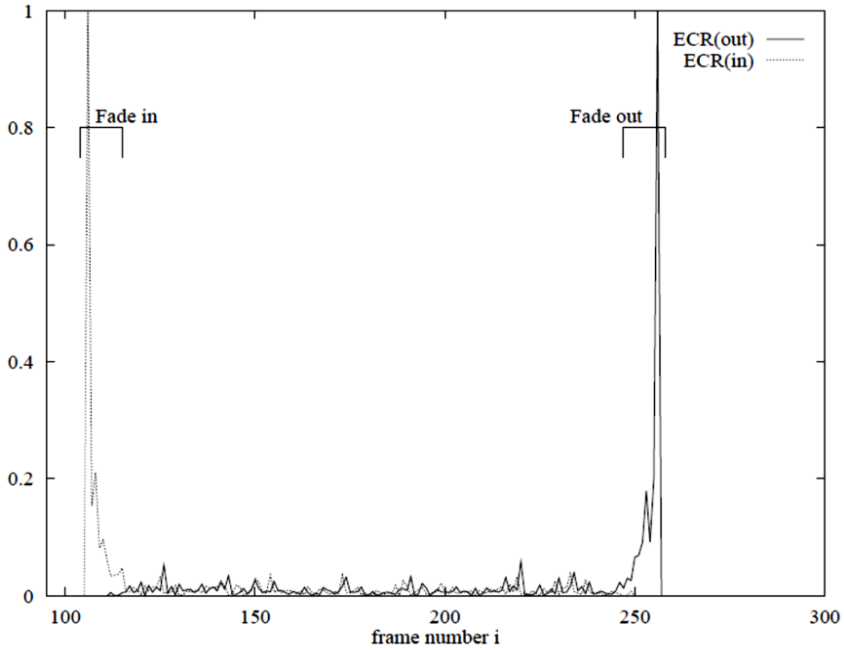
2. Berechnung der Edge-Change-Ratio (ECR):

$$ECR_{i-1} = \max\left(\frac{E_{in}}{s_{i-1}}, \frac{E_{out}}{s_i}\right)$$

ECR-Schnitterkennung

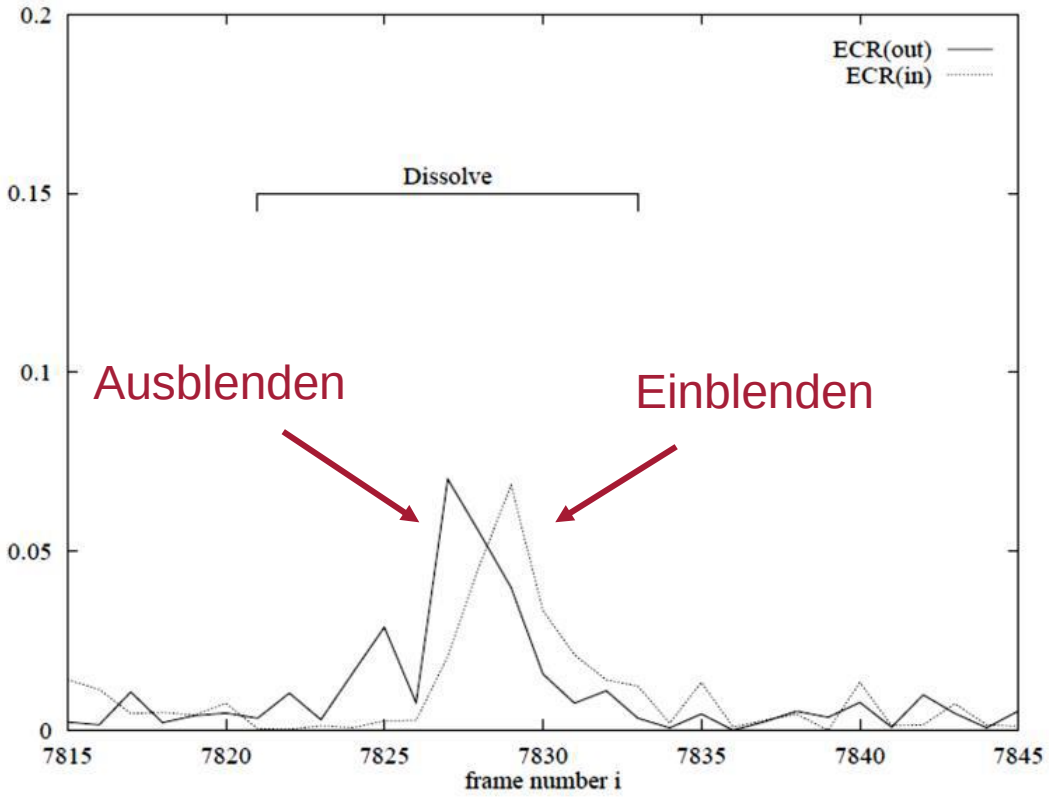
- Gut bei harten Schnitten
- Relativ gut bei Ein- und Ausblendungen





ECR-Schnitterkennung

- Schlecht bei Überblendungen



Kantenorientierter Kontrast

Idee: Kantenstärke ist niedrig bei Überblendungen

Vorgehen:

1. Wir bestimmen für jeden Pixel in **einem** Frames, ob er ein starker oder schwacher Kantenpixel ist
2. Und berechnen dann das Verhältnis zwischen diesen:

Anz. starke Kantenpixel

$$EC(i) = 1 + \frac{s(i) - w(i) - 1}{s(i) + w(i) + 1}$$

Anz. schwache Kantenpixel