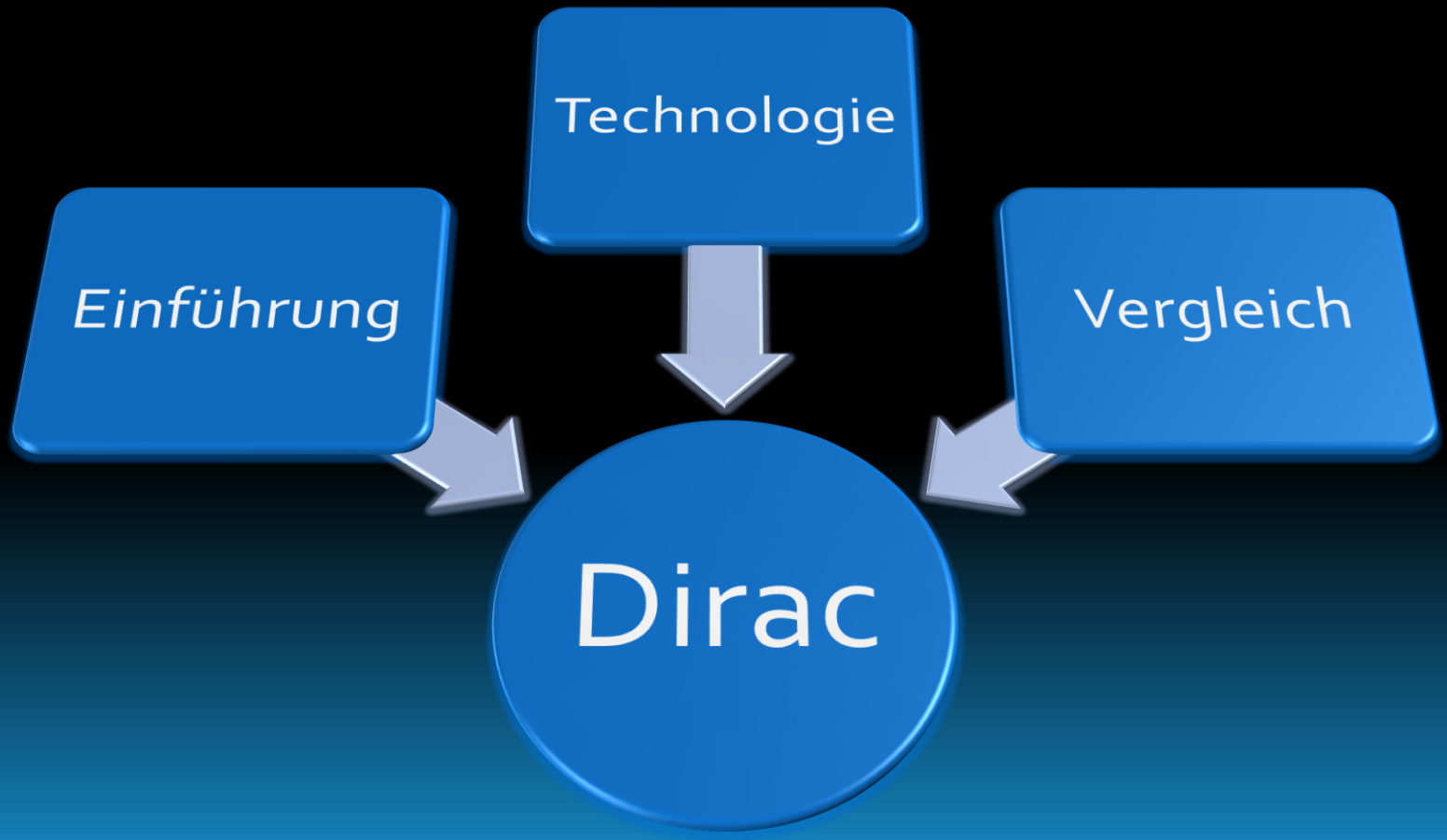
Seminar - Multimedia Codecs 2009

MARC SEEGER
MS241

CLEMENS KERN
NK035

BENJAMIN GUTBROD
BG019

Überblick



Einführung

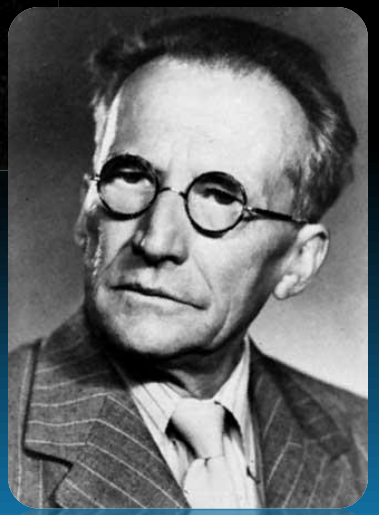

AUCKLAND
WELLINGTON 243 k
VANCOUVER 10,391 km
NEW YORK 12,749 km
WESTPORT 8 km
ROME 15,869 km
ZURICH 16,093 km
SYDNEY 1,719 km
PARIS 16,376 km

Was ist Dirac?

- Paul Dirac
  - Britischer Physiker
  - Mitbegründer der Quantentheorie
  - 1933 Nobelpreis für Beitrag zu Quantentheorie

Paul Dirac

Schrödinger



Was ist Dirac?

- Laut Entwickler-Homepage
  - “Dirac is a general-purpose video compression family suitable for everything from internet streaming to HDTV”
- Laut offiziellem Pressebericht
  - “(…) a cost effective, multi-platform open source compression codec for hardware and software, from web formats to ultra high definition post production”

Allgemeines

- **Video**-Kompressions-Familie
  - Breite Palette von Tools (ähnlich wie Mpeg4)
  - Bestimmte Applikationen benötigen i.d.R. nur Untermenge dieser Tools -> Profile
  - Standardkonforme Geräte implementieren nur bestimmte Profile

Allgemeines

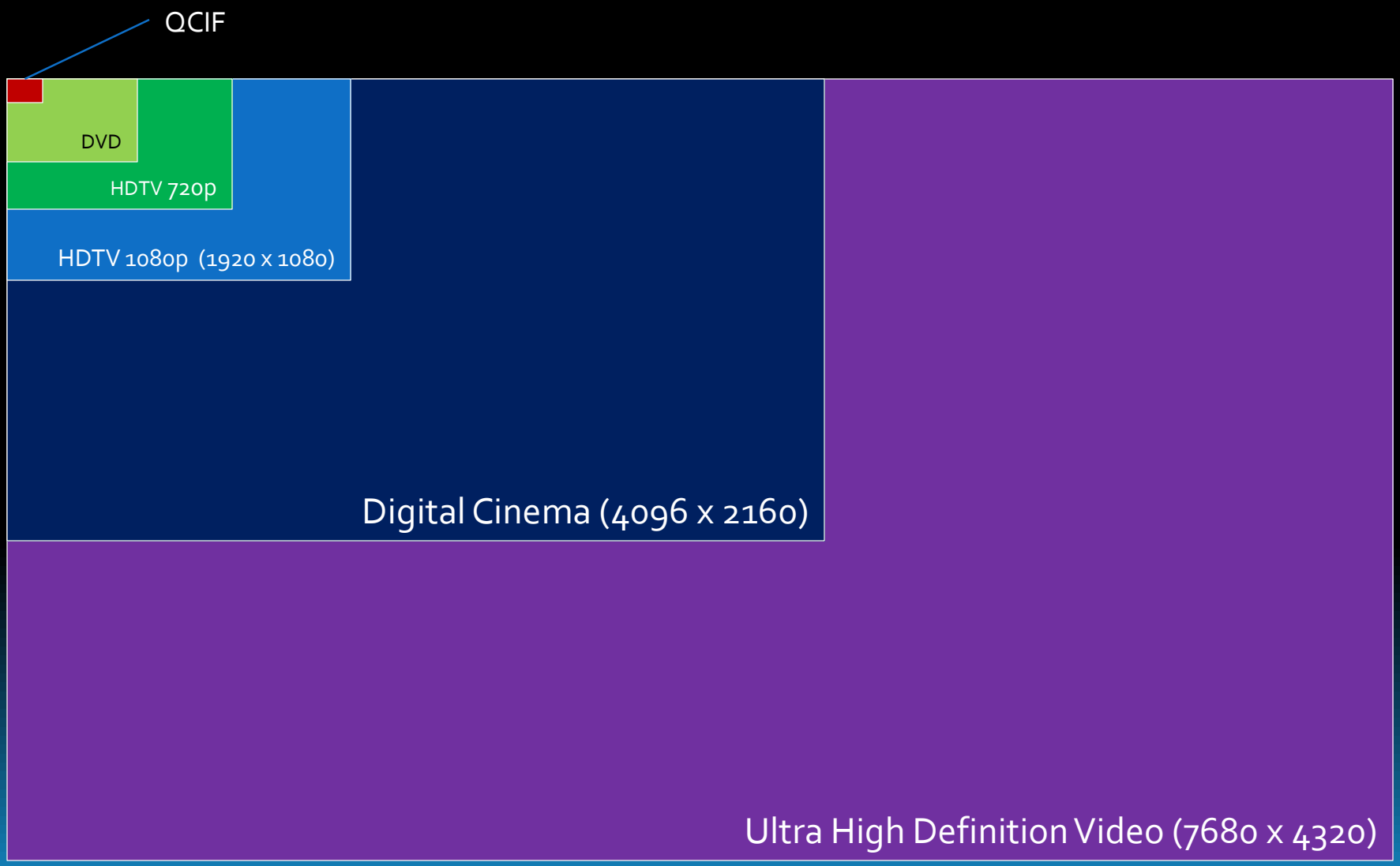
- Auflösungen, u.a.
  - **QCIF** (180 x 144)
  - **HDTV** (1.920 x 1.080)
  - **Digital Cinema 4k** (4096 × 2160)
  - **Ultra HDTV** (7680 x 4320)

mobile streaming

(full) HDTV

Digital Cinema

Allgemeines



Allgemeines

- Kompression:
  - Verlustfrei / optisch verlustfrei
  - Verlustbehaftet
- Leistung vergleichbar mit aktuellen Codecs
  - H264/MPEG-4 AVC und VC-1
- Bitraten von unter 100kbit/s – über 1Gbit/s ++

Allgemeines

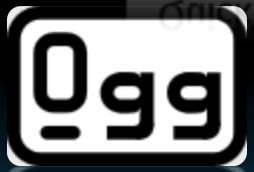
- Technologie
  - Wavelet Transformation
    - Statt DCT wie bei MPEG1,2,4
  - Motion Compensation
    - Überlappende Blöcke reduzieren Block-Artefaktbildung
  - Arithmetische Codierung
    - Verlustfreie Datenkompression

Allgemeines

- Profile
  - Dirac für bestimmte Aufgaben optimieren, z.B.
    - Verteilung und Übertragung → Hohe Kompression
    - Post Production → Hohe Qualität
    - Live Broadcasting → Geringe Latenz
    - usw…
  - Vordefinierte Einstellungen bestimmter Parameter

Allgemeines

- Unterstützte Container
  - AVI
  - MKV
  - ISOM (MP4, Quicktime)
  - OGG
  - MXF



Wer hat es Entwickelt?

- British Broadcasting Corporation (BBC)
  - In Kingswood Warren, UK
  - Projektleiter:
    - Tim Borer
  - Algorithmus Guru:
    - Thomas Davies

Wer hat es Entwickelt?

- British Broadcasting Corporation (BBC)
  - 50 Jahre Erfahrung mit Kompressions-Technologie und digitalem TV
  - Wünscht sich Zusammenarbeit mit
    - OpenSource Community
    - Akademikern
    - Sonstigen Helfern

    → Open Technology

Wer hat es Entwickelt?

- Wie alles begann…
  - 2001 - Thomas Davies experimentiert mit Kompressionstechniken
  - Sein Ziel:
    HD Services auf Basis von MPEG-2 Broadcasts
  - → Dirac war geboren…

Wer hat es Entwickelt?

- Mittlerweile Einstieg von Hardwarefirmen
- Numedia Technology
  - Encoder + Decoder
    - Dirac Pro 1.5
    - Dirac Pro 270

Warum wurde es entwickelt?

- Grund für die Entwicklung
  - Lizenzgebühren proprietärer Codecs
    - Damals bereits 50 000 Streams → hohe Kosten
    - Geplante Erweiterung auf 50 000 000+ Streams
  - Abhängigkeit von den Entwicklern
    - Unnötige Features
    - Lange Release-Zyklen

Warum wurde es entwickelt?

- Philosophie
  - Keep it simple!
  - Einfaches und modulares Design
- Warum?
  - Zusammenarbeit verschiedener Gruppen soll ermöglicht bzw. erleichtert werden

Warum wurde es entwickelt?

- Ziele der Entwicklung
  - Lizenzfreier hocheffizienter Codec
    - von Streaming bis zu High-End Qualität
  - Möglichst hohe Kompressionsrate ohne Artefaktbildung
  - Mit anderen komplexen "State of the Art" Codecs mithalten (trotz einfachem Design)

Besonderheiten

- Zukunftssicher
  - Unabhängig von einzelnen Firmen
- Komplett lizenzfrei
  - Im Sinn von Lizenzgebühren
- Individuell anpassbar
  - Kann auf jedes Einsatzgebiet zugeschnitten werden

Features

- Auflösungen von QCIF bis UHDTV
- Inter- / Intraframe Codierung
- Frame- / Field Codierung
- Konstante / variable Bitraten

Features

- Verschiedene Wavelet-Filter
  - Performance ⇔ Komplexität
- Variable Farb-/Kanaltiefe
  - 8, 10, 12 und 16 bit
- Verschiedene Farbmodelle
  - RGB & YUV 4:4:4 / 4:2:2 / 4:2:0

Chroma Formats

4:4:4

RGB

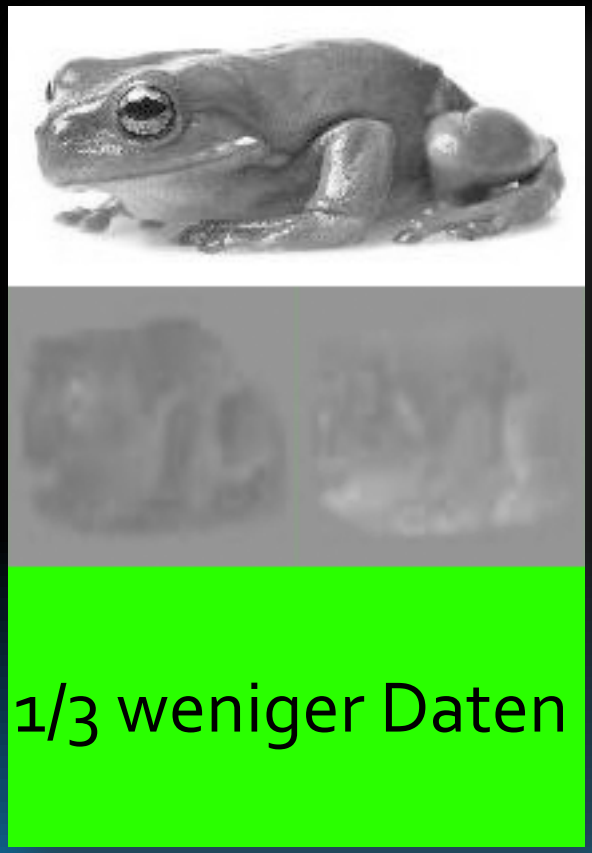
Y

Cb

Cr

Chroma Formats

4:2:2



1/3 weniger Daten

4:2:0

1/2 weniger Daten

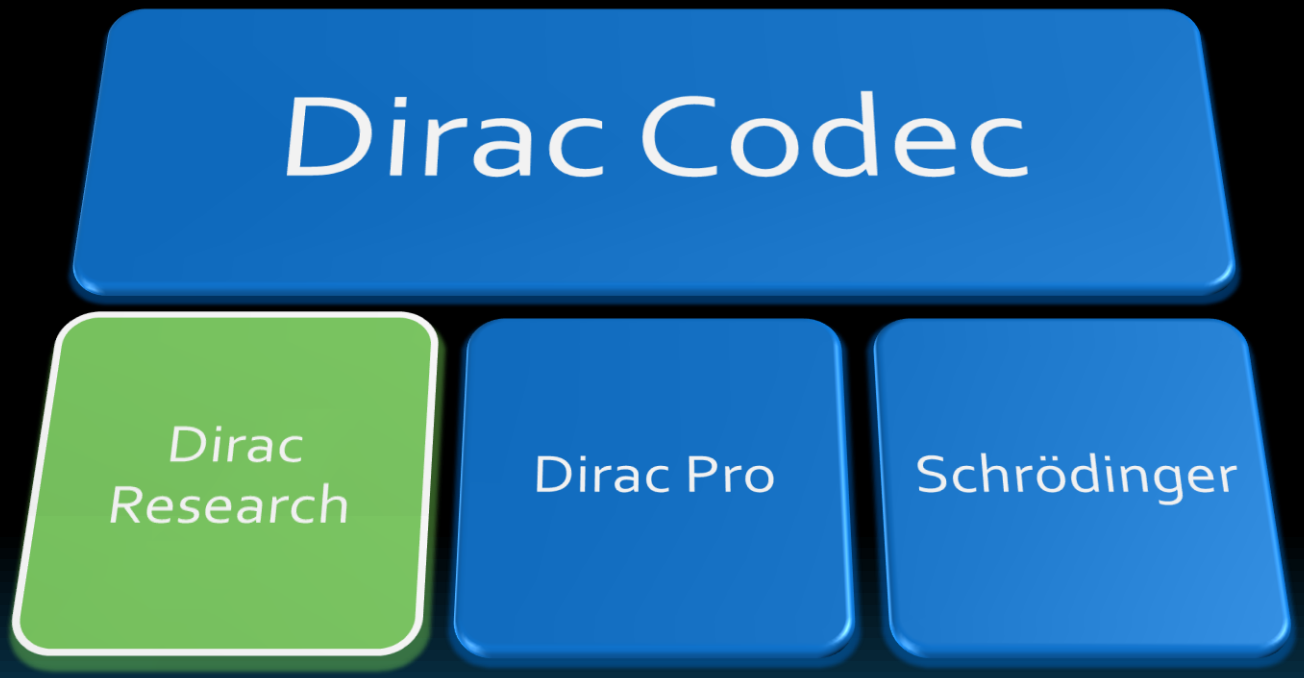
Features

- Anpassbares Bildseitenverhältnis
- Variable Bildwiederholrate
  - von 23,97 bis 60 fps
- Einfache Streamnavigation
  - Bildnummern (32 Bit)
  - Doppelt verkettete Listen

Features

- Multiresolution Transformation
- Dual Syntax
- Konvertierung RGB in YUV

Die Dirac Familie



Dirac Research

- Referenzimplementierung in C++
  - V1.0 – September 2008
  - V1.2 – Februar 2009
- Laut BBC „State of the Art“ Qualität und Performance bei hoher Kompressionsrate

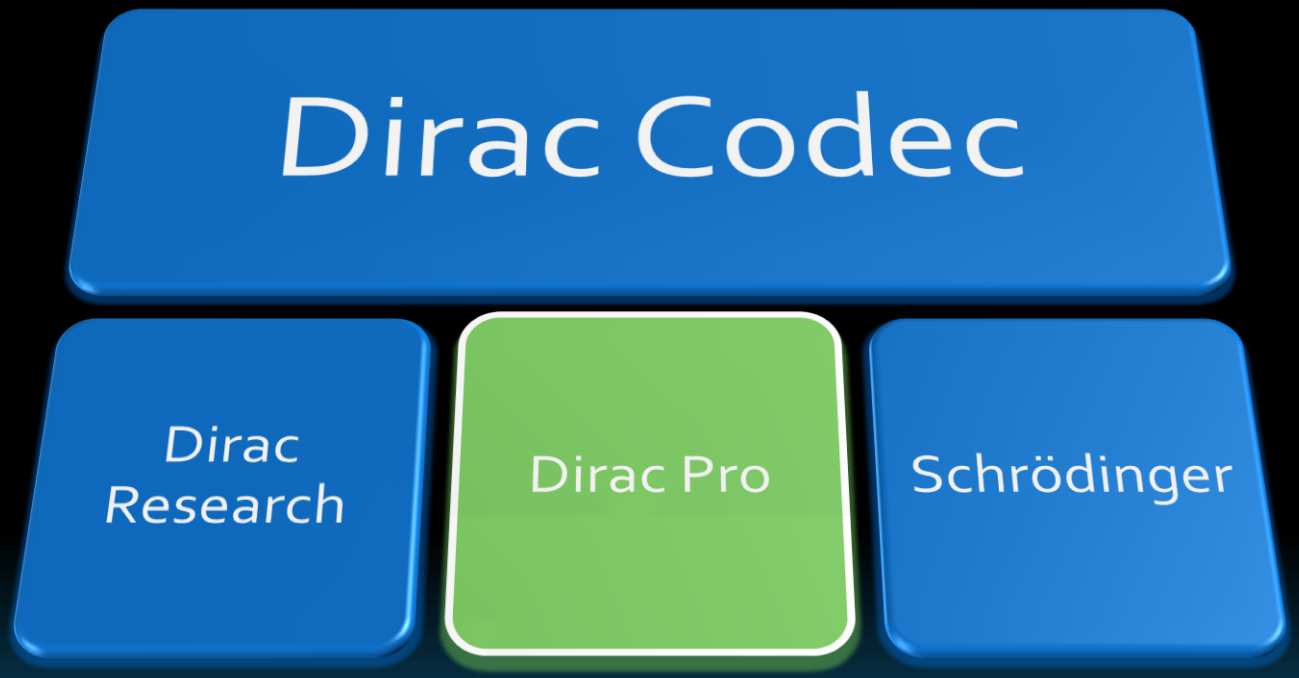
Dirac Research

- Technologie
  - Inter-Frame Wavelet-Transformation
  - Long-GOP (optional)
  - Arithmetische Codierung
- Ziel
  - Maximale Kompressions-Effizienz

Dirac Research

- Anwendungsgebiete
  - Internet/Mobile Streaming
  - Embedded Video auf Webseiten
  - Video on Demand (Streaming)

Die Dirac Familie



Dirac Pro

- Implementiert in C
- Spezielle Version von Dirac, optimiert für
  - Professionelle Produktion
  - Archivierungsanwendungen
  - Einsatz in Film-Studios / Broadcast-Zentren
- Standardisiert durch SMPTE als VC-2

Dirac Pro

- Anforderungen
  - Hohe Bild-Qualität
  - Verlustfreie Kompression
  - Hohe Bitraten
  - Sehr geringe Latenz

→ Erfordert höhere Bandbreiten (Mbit/s – Gbit/s)

Dirac Pro

- Unterschied zu Dirac Research
  - Intra-Frame Wavelet-Transformation
    - Ermöglicht framegenaue Editierung
  - Exponential-Golomb Codierung
    - Ermöglicht sehr hohe Bitraten
  - Keine Arithmetische Codierung
    - Niedrige Latenz
    - Bessere Performance

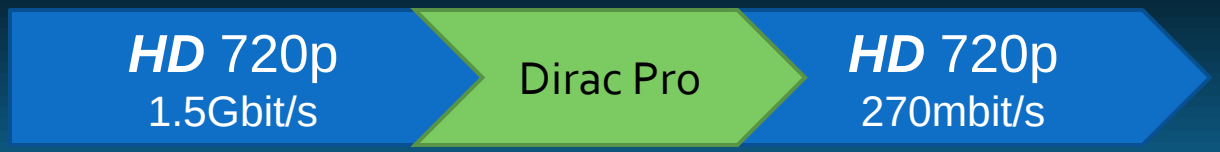
Dirac Pro

- Anwendungsgebiete
  - Post Production
  - Digital Cinema Mastering
  - Archivierung

Dirac Pro

- Ermöglicht Übergang zu HD-Produktion
  - Übertragung von
    - Full HD-Signale über standard HD-Infrastruktur

    - HD-Signale über bestehende SD-Infrastruktur



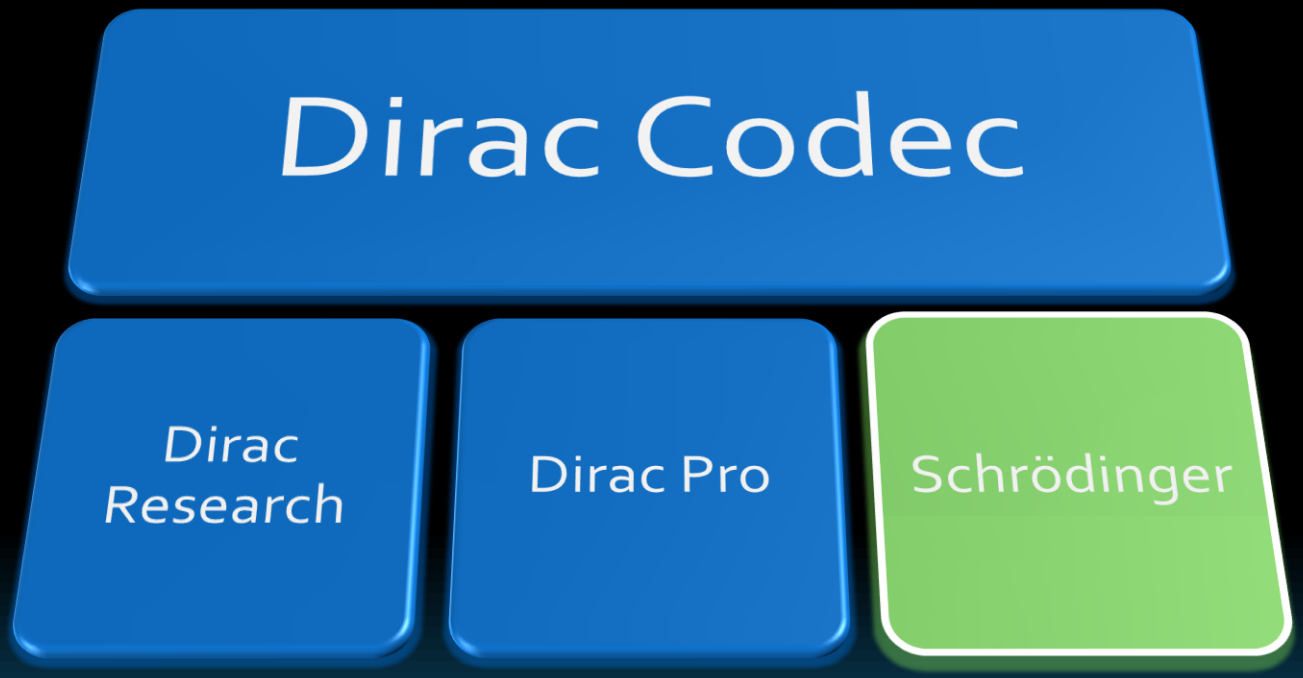
Dirac Pro 1.5

- Full HDTV (1080P) über konv. HD-Infrastruktur
  - (1,5Gbit/s HD-SDI)
- Komprimierungsverhältnis von 2:1
  - Quasi verlustfrei
- Sehr geringe Latenz: << 1ms (end to end)
- Geplanter Einsatz bei Olymp. Spielen 2012

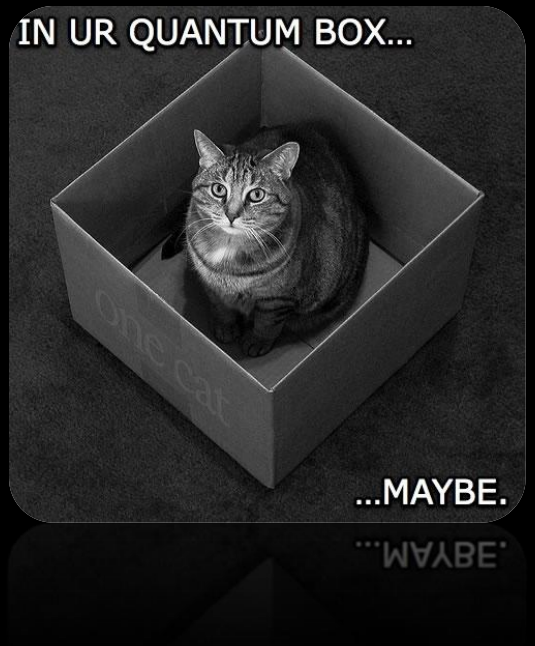
Dirac Pro 270

- HDTV (720p) über SD-Infrastruktur
  - (270Mbit/s SD-SDI)
- Stärkere Komprimierung als bei Dirac Pro 1.5
  - Nur optisch verlustfrei
- Geringe Latenz: < 3ms (end to end)
- Erfolgreich eingesetzt bei Olymp. Spielen 2008

Die Dirac Familie



Schrödinger Projekt



- Implementiert in ANSI C
  - V1.00 – Februar 2008
  - V1.07 – April 2009
- Optimierte Implementierung der Dirac-Spezifikation
- Finanziert durch BBC

Schrödinger Projekt

- Ziele
  - Unterstützung von Dirac in Multimedia-Anwendungen beschleunigen
  - Codier-Performance erhöhen
    - Auf Kosten der Bild-Qualität

Schrödinger Projekt

- David Schleef, Leitender Entwickler:

"...you either get slow and good (dirac-research) or fast and crappy (Schrödinger)"

Schrödinger Projekt

Dirac playback compatibility matrix

	Raw Bitstream	Ogg	MPEG Transport Stream	AVI	Matroska	MOV / MP4
GStreamer / Totem	Yes	Yes	Yes	Yes	Yes	Yes
FFMpeg / FFplay	Yes	No	Yes	Yes	Yes	Yes
MPLAYER	Yes	No	Yes	Yes	Yes	Yes
VLC 1.0	Yes	Yes	Yes	Yes (?)	Yes (?)	Yes (?)
DirectShow/ Windows Media Player	Yes (unreleased)	Yes (unreleased)	No	WIP	No	No
QuickTime Player / OSX	No	No	No	No	No	Yes (unreleased)

Quelle: http://www.diracvideo.org/wiki/index.php/Dirac_Compatibility_Matrix

Lizenzierung

- Freier Videocodec
  - MPL 1.1
  - Keine Patente auf Seiten der BBC an Dirac
  - Keine Verwendung von patentierten Technologien Dritter

→Dirac auch mit GPL und LGPL Lizenz

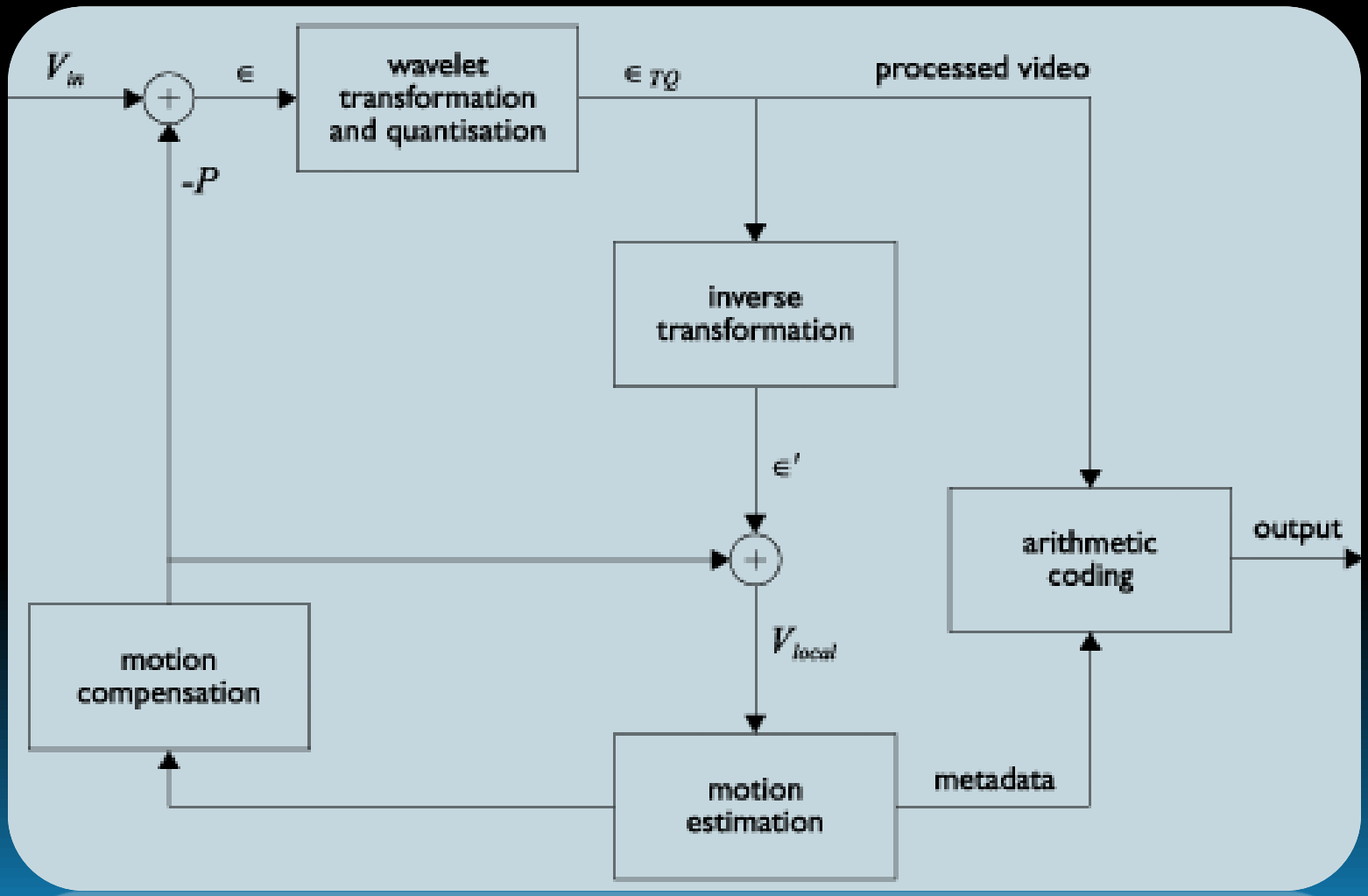
→Schrödinger zusätzlich mit MIT Lizenz

Lizenzierung

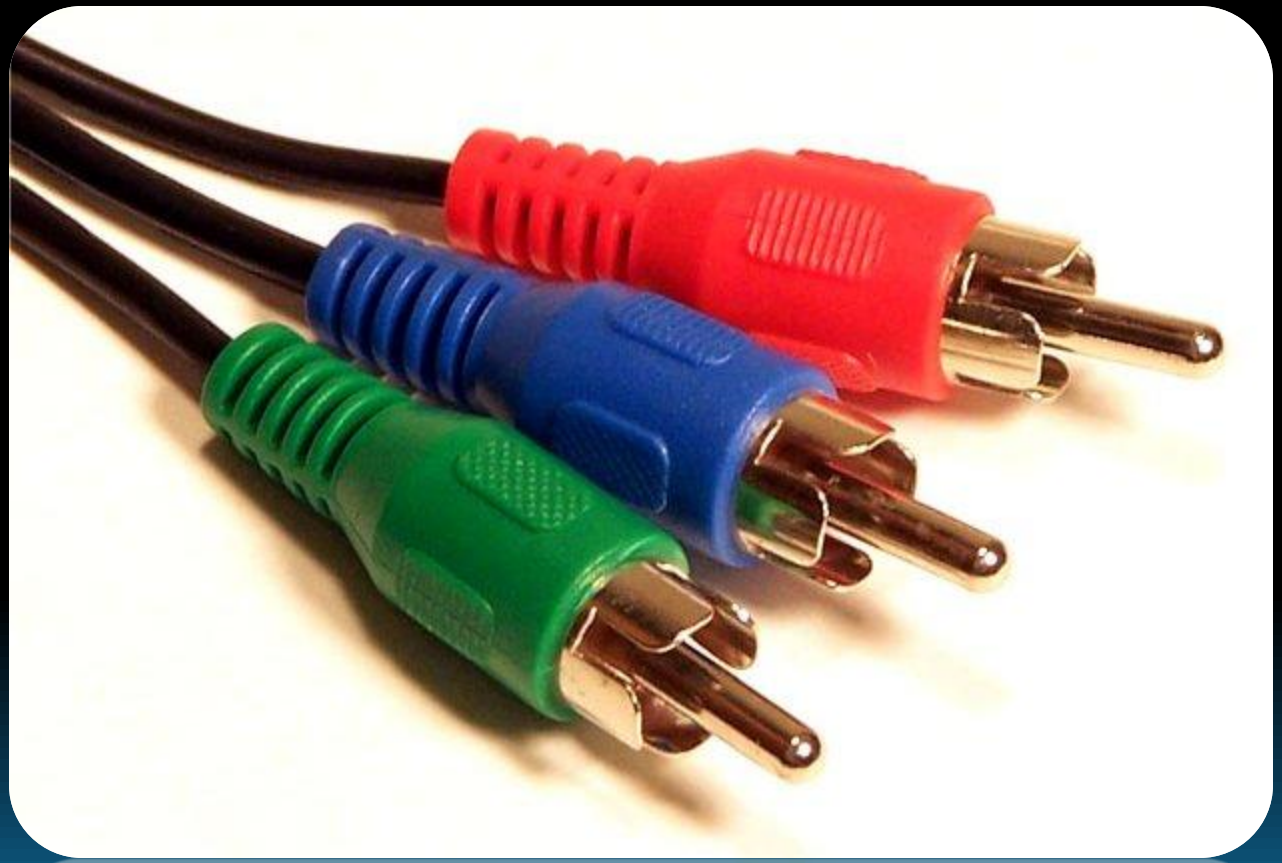
- **GPL** (GNU General Public License)
  - Erlaubt keine Einbindung in proprietäre Software
- **LGPL** (GNU Lesser General Public License)
  - Erlaubt indirekte Einbindung in proprietäre Software
- **MPL** (Mozilla Public License)
  - Erlaubt Verbindung mit proprietärem Code, der sich in separaten Dateien befindet, aber gemeinsam kompiliert wird
- **MIT** (Massachusetts Institute of Technology)
  - Erlaubt direkte Einbindung in proprietärerSoftware

Technologie

Technologie Übersicht



Input



Y: luma
U: chroma
V: chroma

≈YCbCr

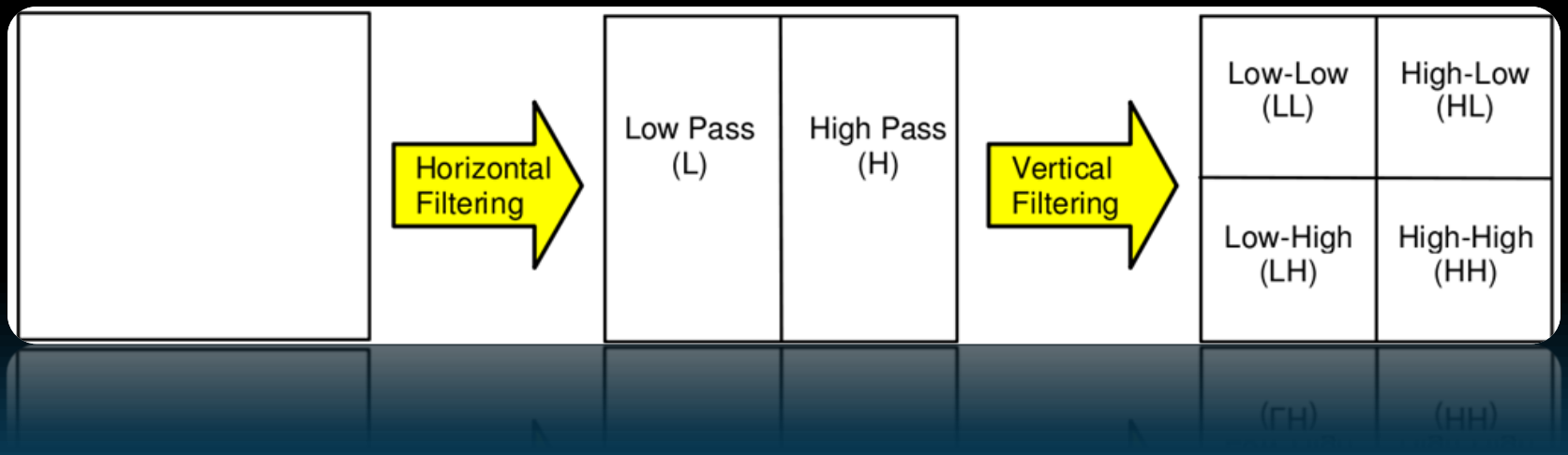
RGB Input?

RGB

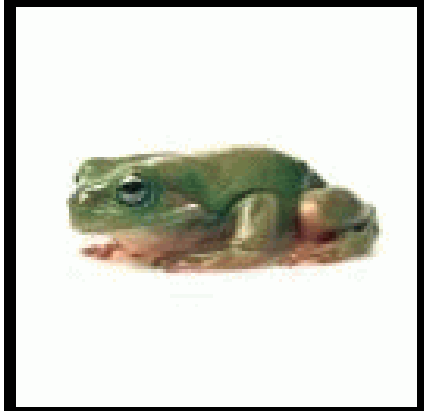
ITU-T H.264
residual color transform

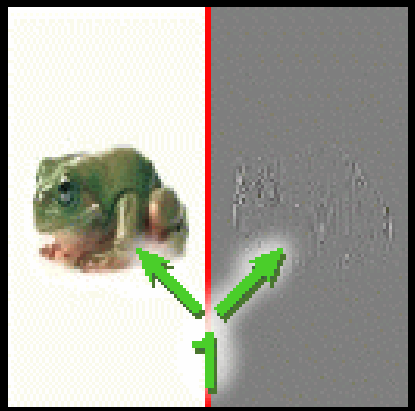
YC_oC_g

Wavelet Transformation

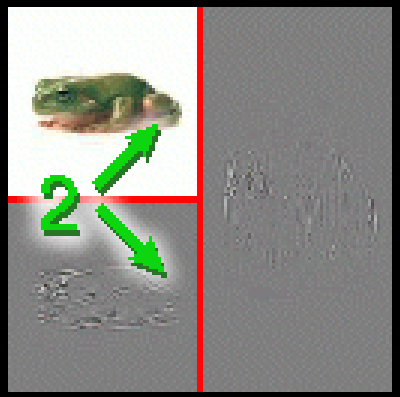


Wavelet Transformation



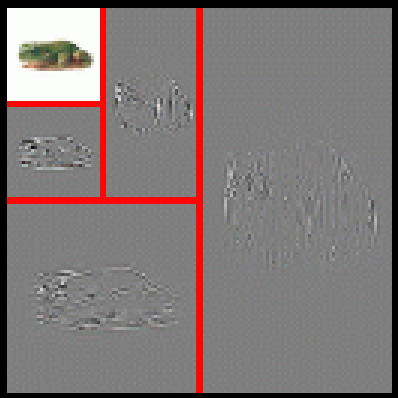


Zeilen
Links:Tiefpass,
Rechts: Hochpass



Spalten des Tiefpass-Anteils gefiltert
(links oben Level-2-Tiefpass,
links unten Level-2-Hochpass)

redo()
links oben:
Level-4-Tiefpass



Quelle:
http://user.cs.tu-berlin.de/~rammelt/wavelets/index.html

Wavelet Transformation

Diesmal mit Quoten Lenna:

Wavelets vs DCT

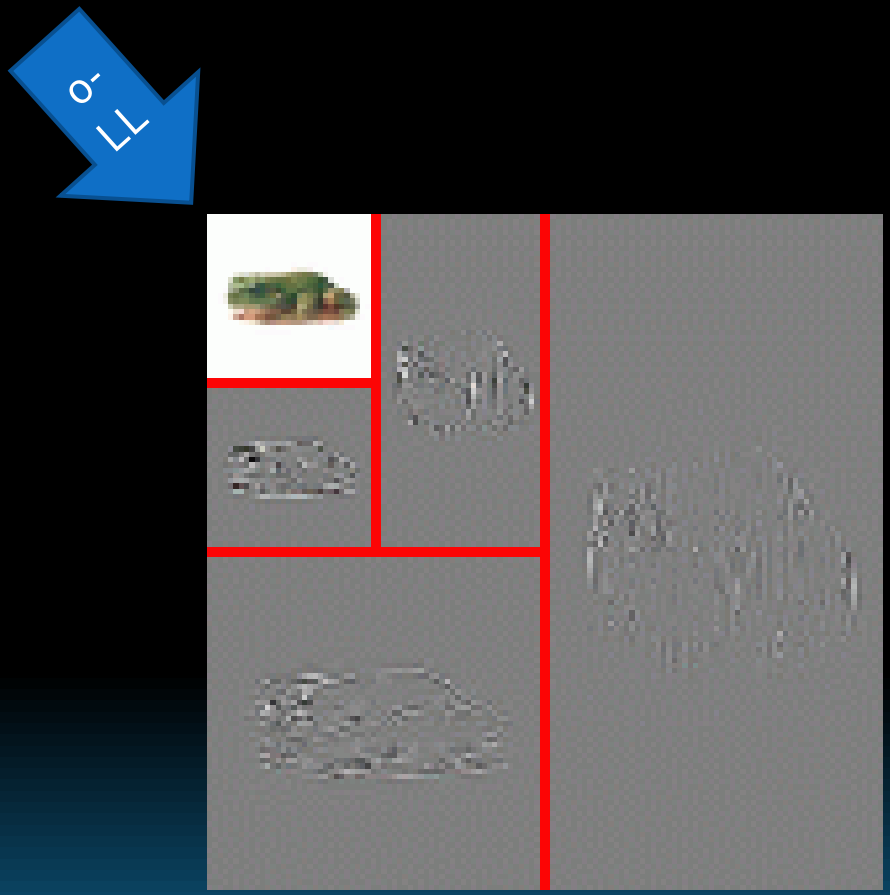
DCT

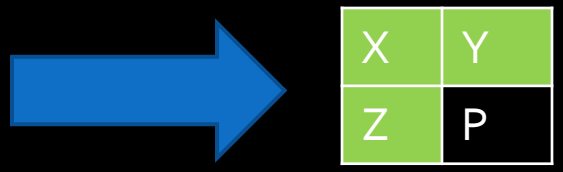
DWT

A wavelet is a wavelet is a ...

Wavelet	Usage
Deslauriers-Dubuc (9,3)	default intra-frame filter
LeGall (5,3)	default inter-frame filter
Deslauriers-Dubuc (13,7)	
Haar with no/single/double shift per level	
Fidelity filter	improved downconversion and anti-aliasing
Daubechies (9,7)	compatibility with JPEG 2000

DC Subband Prediction

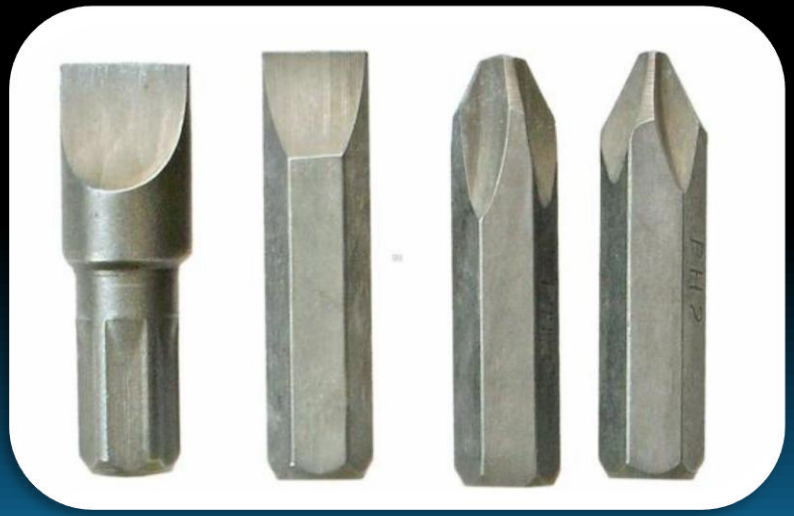




Quantisation

DWT = lossless!

But we haven‘t got bits to spare…

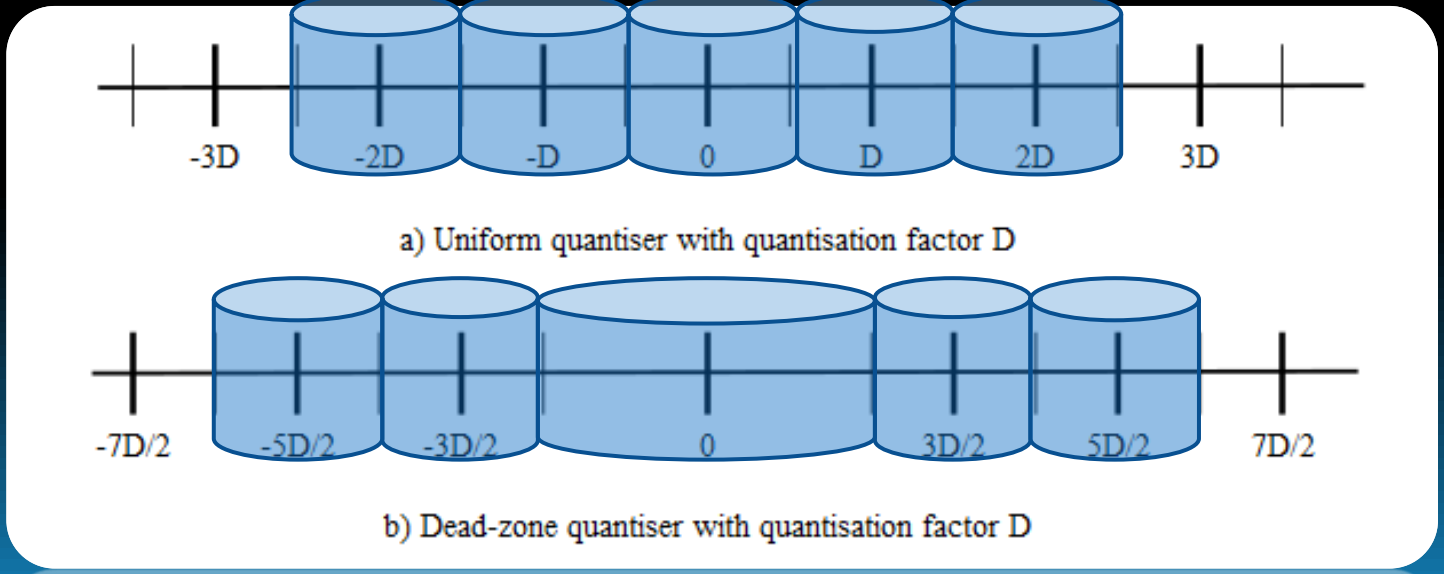


dead-zone quantiser

Vorteil:

- mehr quantisation der kleinen Werte → de-noising
- implementation: teile mit Qunatisierungsfaktor und runde ab.

(In Dirac: näherungsweise via multiplication +bitshift.)



(N-0.5) * D → (N+0.5) * D

>0: N*D → (N+1)*D
<0: (N-1)*D → N*D

Quantisation

The current Dirac encoder uses an RDO technique to pick a quantiser by minimising a Lagrangian combination of rate and distortion.

Rate is estimated via a an adaptively-corrected measure of zeroth-order entropy measure Ent(q) of the quantised symbols resulting from applying the quantisation factor q, calculated as a value of bits/pixel.

Distortion is measured in terms of the perceptually-weighted error fourth-power error E(q,4), resulting from the difference between the original and the quantised coefficients

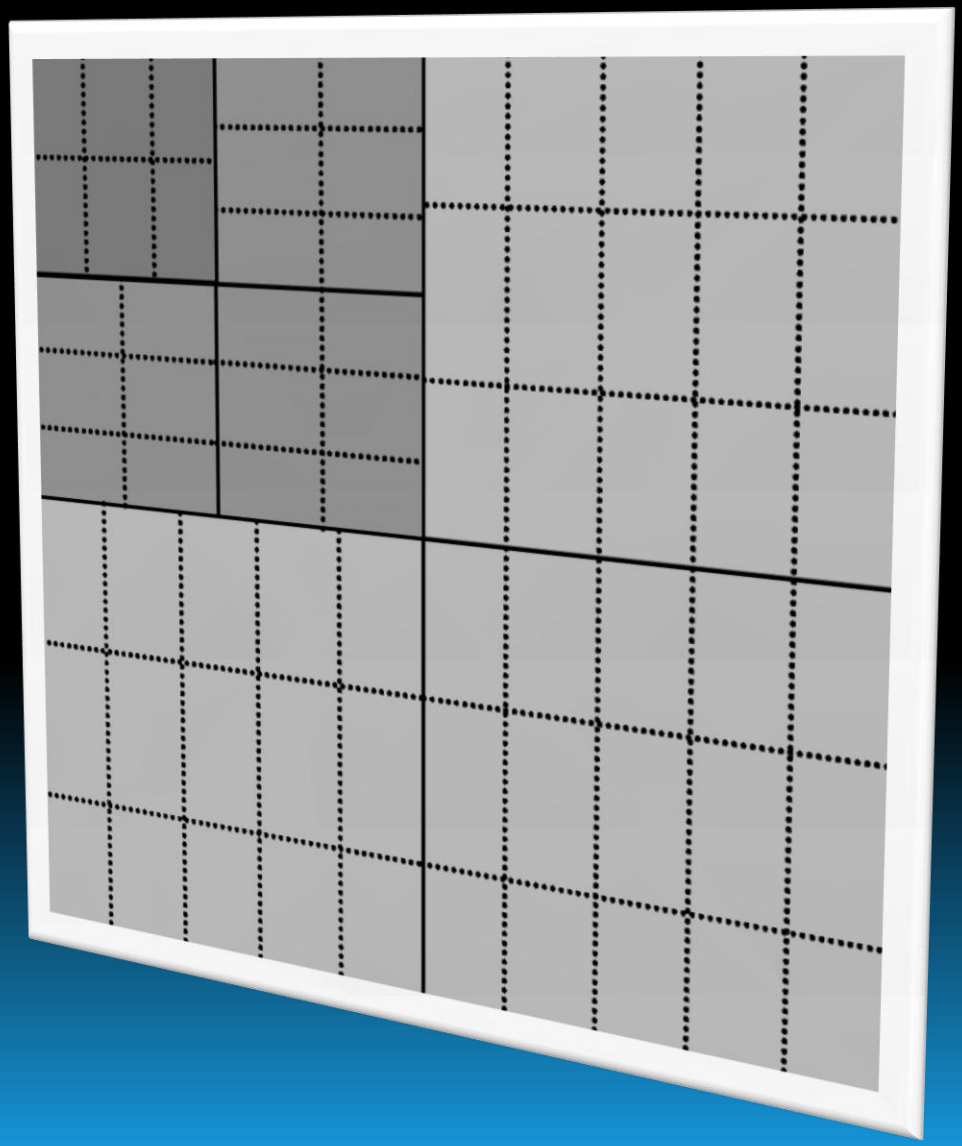
Quantisation

Essentially, lots of quantisers are tried and the best* one is picked

*rate-distortion framework

Coefficient scanning

Eigene Quantisierungs-faktoren für Codeblocks statt ein einzelner Faktor pro Subband

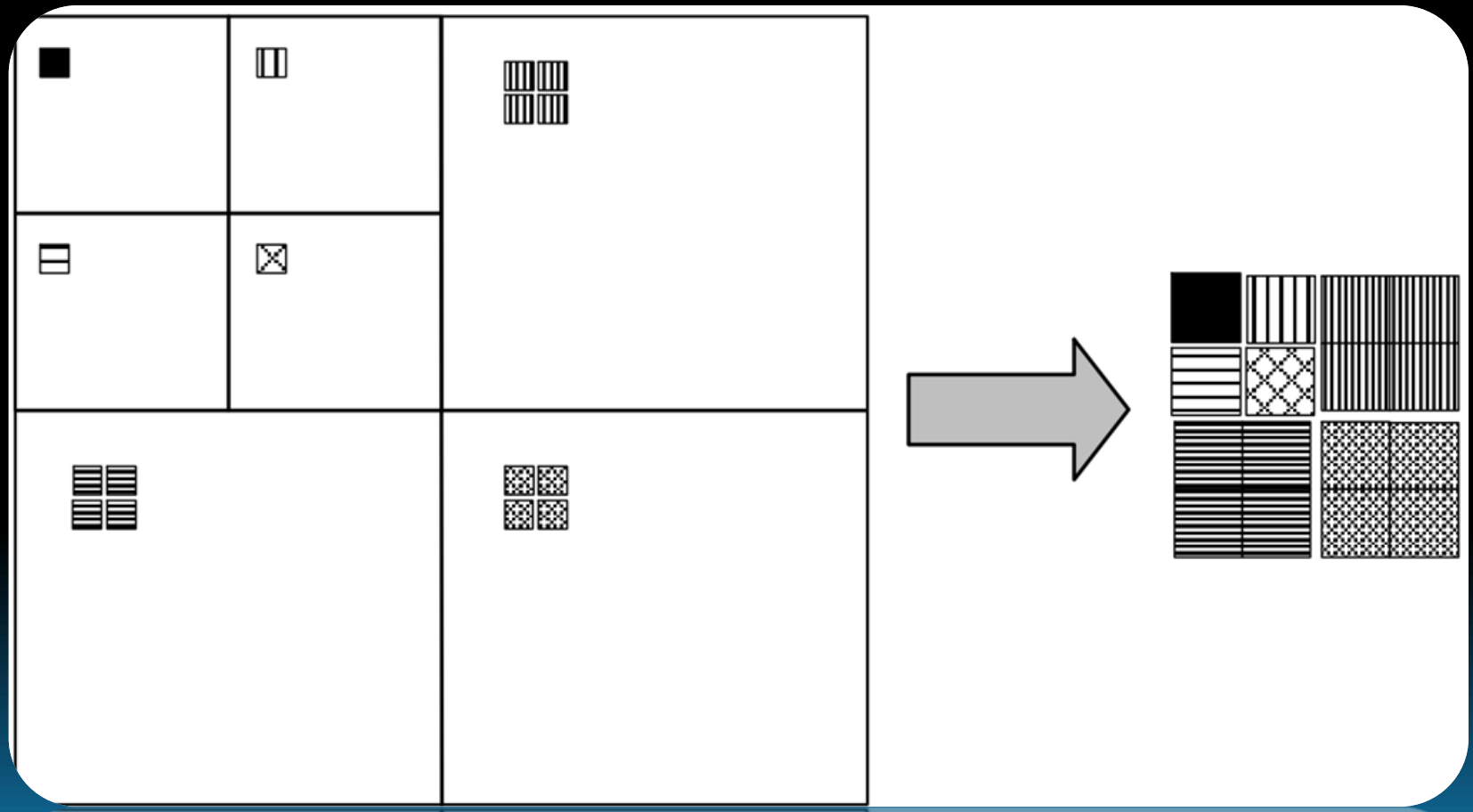


Low Delay Quantisierung

Für die low delay syntax werden die Koeffizienten jedes Subbandes in ein Array von “lokalen” Wavelets angeordnet dass sich “Slice” nennt.

Jeder Slice korrespondiert mit einer Region des Originalbildes (in Frequenzen zerlegt).

Low delay coefficient scanning



Quantisierung des DC Subbandes

Koeffizienten basieren im DC Subband auf Vorhersagen!

➔ mögliche Artefakte

Motion Compensation

Motion Compensation

Motion Compensation

Motion Estimation

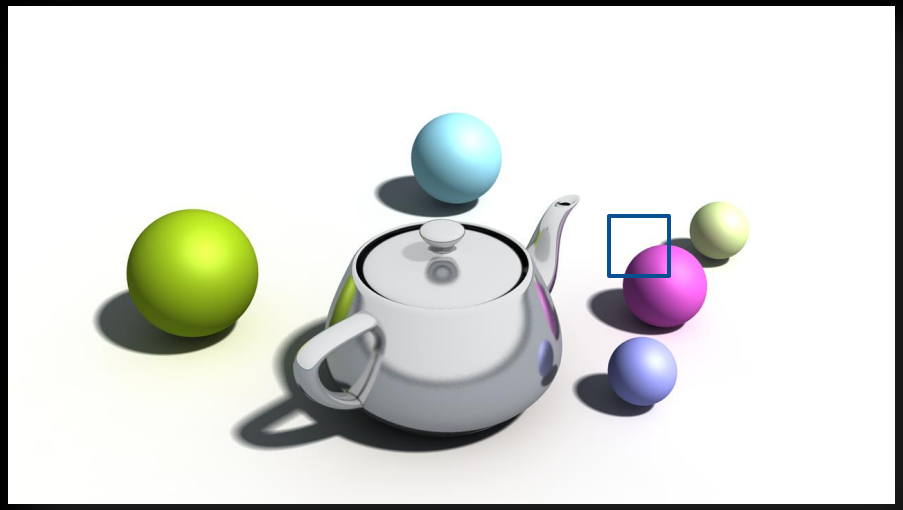


Bild zur Zeit t

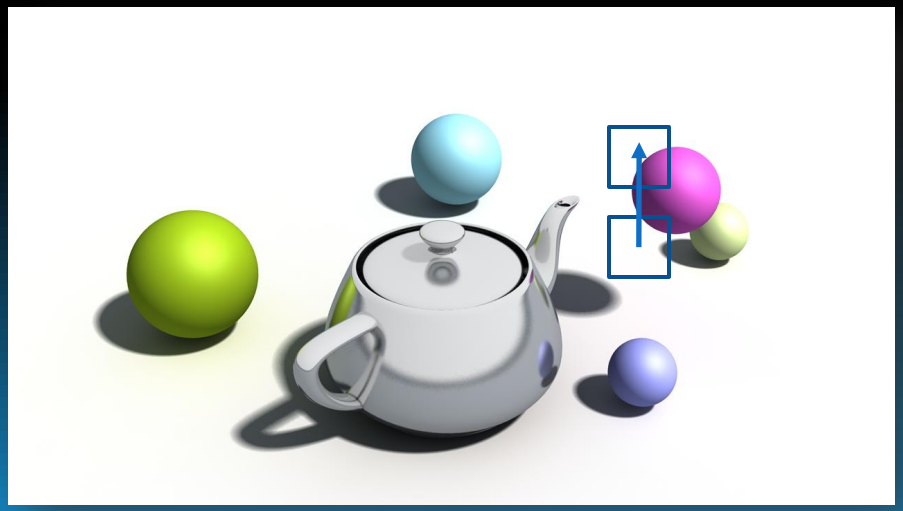
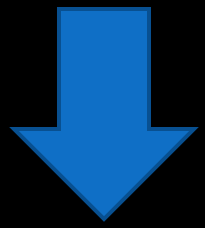


Bild zur Zeit t‘

Motion Estimation

$$< p > (x, y, t) = \sum_i w_i \, p(x - u, y - v, t'), \sum_i w_i = 1$$



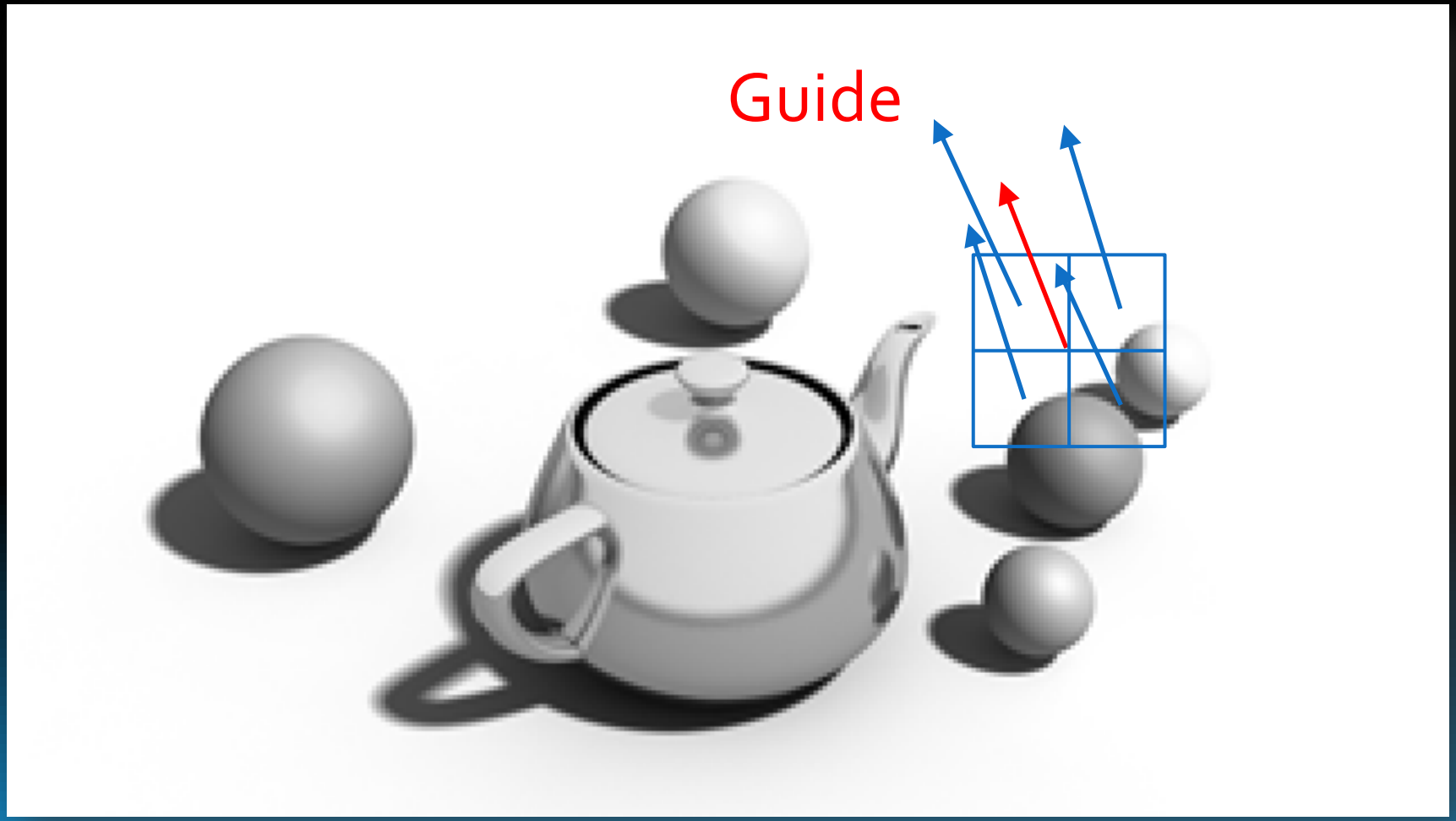
Suchen und minimieren von SAD (SSD, MSE etc.)

Motion Estimation

1. Stufe - Search

Hierarchische Suche

1. Stufe - Search



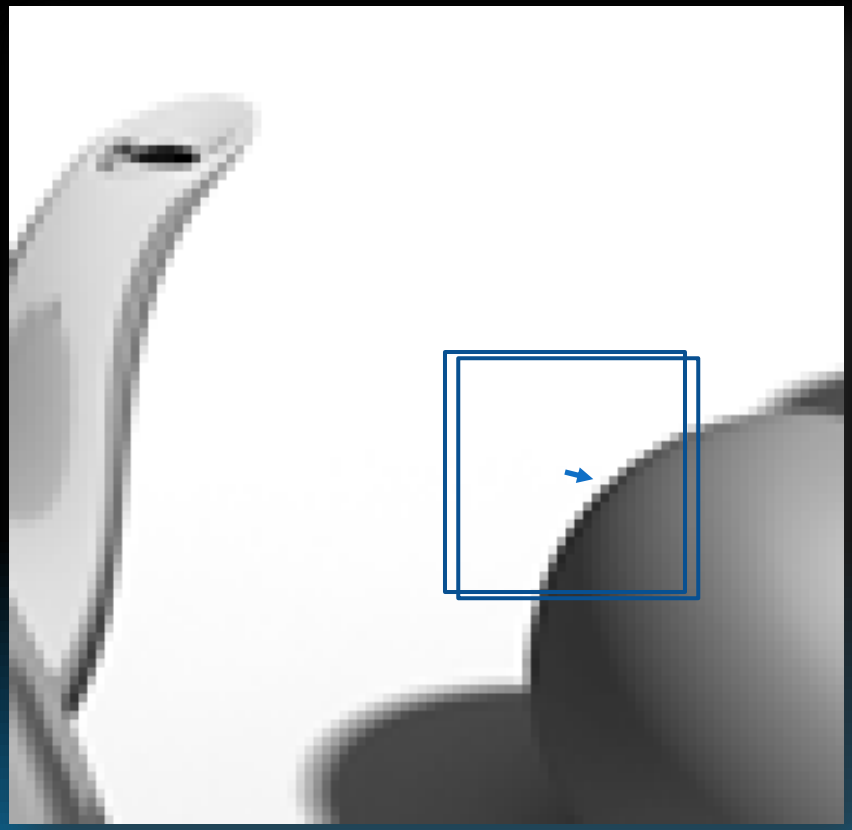
1. Stufe - Search

- Probleme
  - Sehr kleine Bewegungen
  - Viele unterschiedliche Bewegungen

1. Stufe - Search

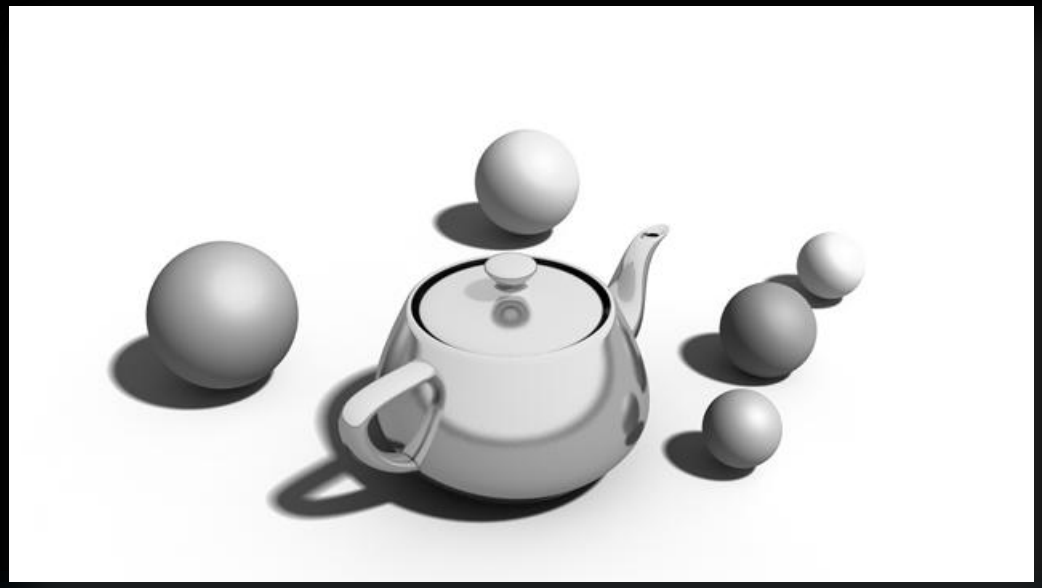
- Lösung:
  - Null-Vektor als zusätzlichen Guide
- Optimierung:
  - Schon gefundene Nachbarvektoren als Guides

2. Stufe - Refinement



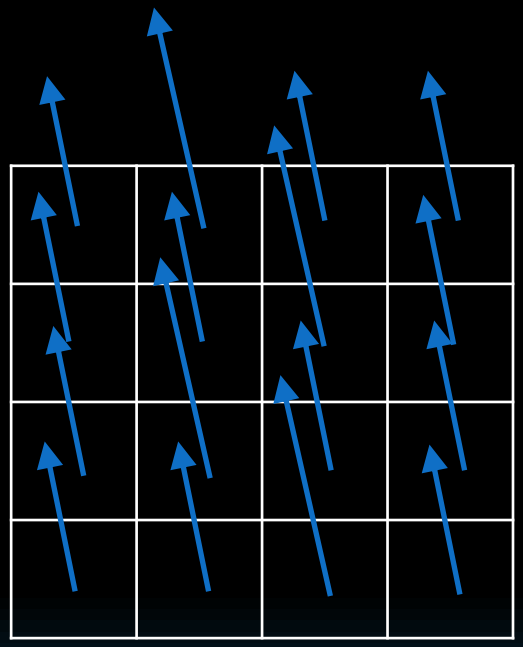
Subpixel-Genauigkeit

2. Stufe - Refinement



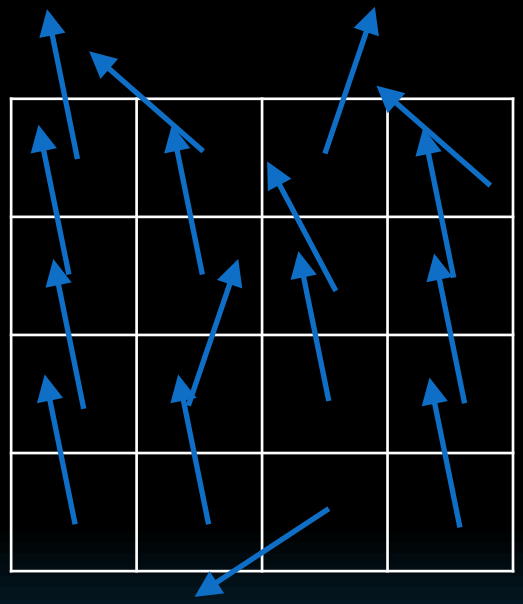
4 facher Upscale => 16 fache Datenmenge

3. Stufe – Mode Decision



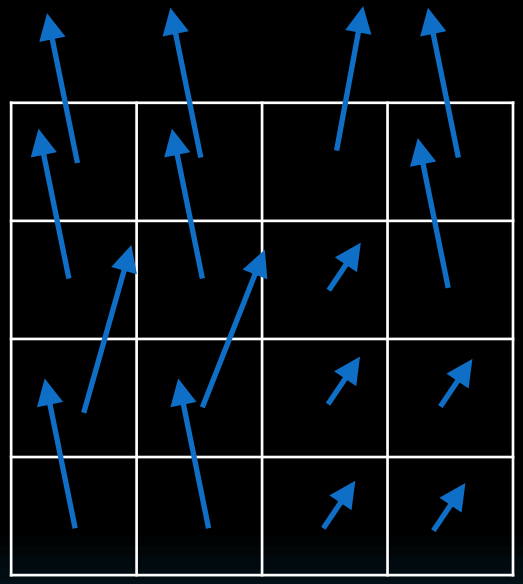
Längen-Differenz zwischen Vektoren übertragen

3. Stufe – Mode Decision



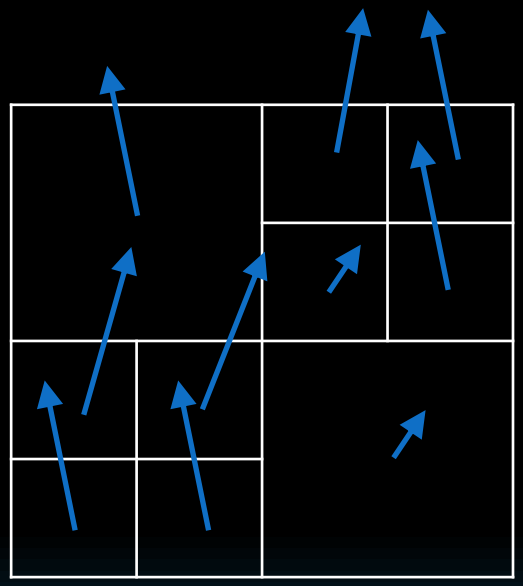
Rotation zwischen Vektoren übertragen

3. Stufe – Mode Decision



Aggregieren

3. Stufe – Mode Decision

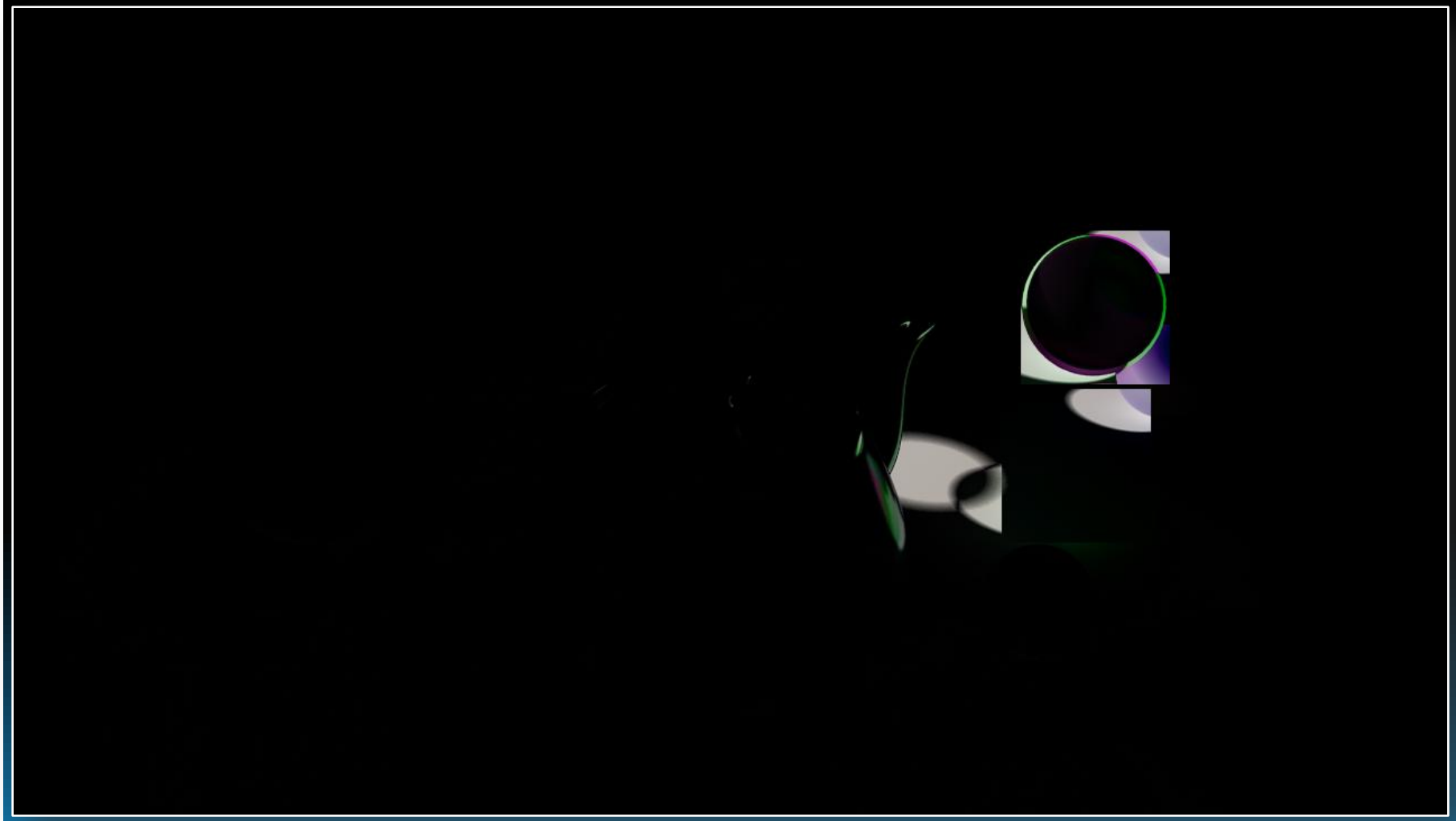


Aggregieren

Compensation

Compensation

Compensation



Codierung

- Exponential Golomb Binarisation
- Arithmetic Coding

Exponential Golomb

exp()

Exp. Golomb - Encode

Schreibe Zahl binär	1111
Lasse K-Bits weg (K=0)	1111
Addiere 1 (arithm.)	10000
Zähle Binärzeichen N	N = 5
Hänge N-1 Nullen vorne an	000010000
Schreibe K-Bits hinten an	

15 -> 000010000

Exp. Golomb - Encode

- Schreibe Zahl binär 1111
- Lasse K-Bits weg (K=2) 11
- Addiere 1 (arithm.) 100

- Zähle Binärzeichen N N = 3
- Hänge N-1 Nullen vorne an 00100
- Schreibe K-Bits hinten an 0010011

15 -> 0010011

Exp. Golomb - Decode

- Zähle Nullen bis zur ersten 1, lese N+1+K Bits weiter
- Lasse K-Bits weg (K=2)
- Ziehe 1 ab (arithm.)
- Schreibe K-Bits hinten an

N=2

10011

100

011

1111

0010011 -> 15

Exp. Golomb - Interleaved

- Schreibe Zahl (13) binär 1101
- Lasse K-Bits weg (K=0) 1101
- Addiere 1 (arithm.) 1110

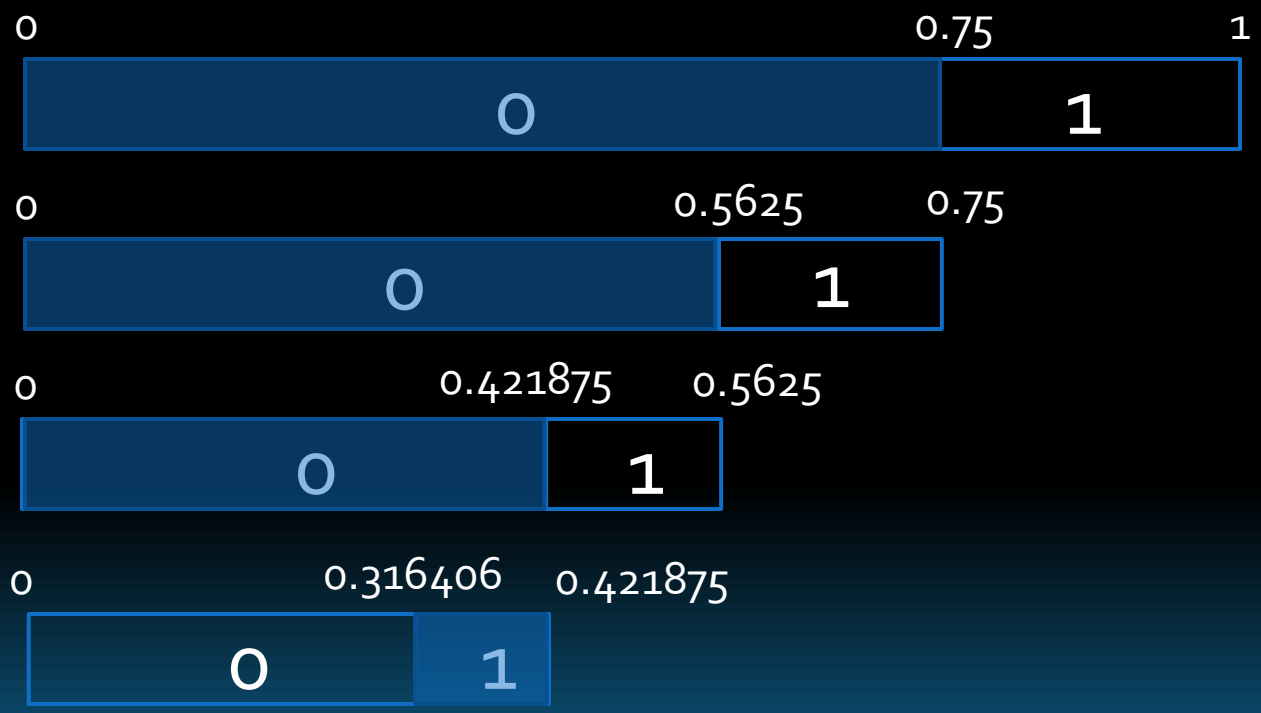
- Interleaving

13 -> 0101010 01

0 => es kommen noch weitere Bits
1 => Ende

Arithmetische Codierung

Zu codierendes Wort: 0001



Wähle Zahl aus letzem Intervall => 0.375

Arithmetische Codierung

Warum gerade 0.375?

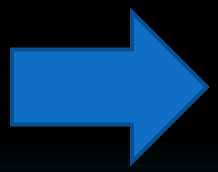
2^{-1}	0.5	0	
2^{-2}	0.25	1	0.25
2^{-3}	0.125	1	0.25 + 0.125 = 0.375

Übertrage 011

Arithmetische Codierung

Warum so gut?

I(0) = 0.415 Bits
I(1) = 2 Bits



Zeichenkette 0001 hat Informationsgehalt 3.245 Bits

Arithmetische Codierung

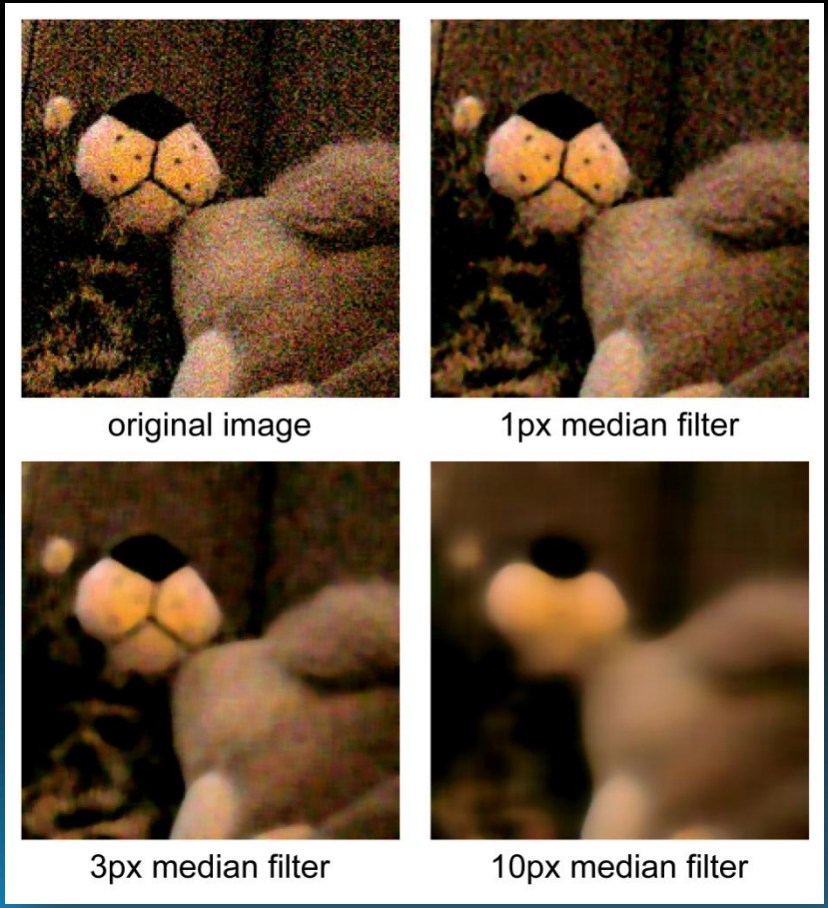
Probleme?

0.6364662458926446873484348845555445224785599876633221241579 61

Arithmetische Codierung

Denoiser

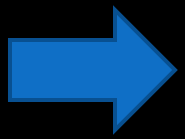
Centre-weighted 3x3 median



Denoiser

Centre-weighted 3x3 median

3	5	10
23	60	3
6	7	9



[3, 3, 5, 6, 7, 9, 10, 23, 60, 60, 60, 60, 60]

3	5	10
23	10	3
6	7	9

Demonstration

Setup

Linux 64 Bit

liboil-0.3.16.tar.gz 19-Mar-2009 14:12 836K

schroedinger-1.0.7.tar.gz 22-Apr-2009 20:56 857K

dirac-1.0.2.tar.gz 11-Feb-2009 19:42 897K

ffmpeg: SVN vom 15. Juni 09

x264: SVN vom 15. Juni 09

FFMPEG:

./configure --enable-libdirac --enable-libschroedinger --enable-libx264 --enable-gpl

-->

FFmpeg version SVN-r19200, Copyright (c) 2000-2009 Fabrice Bellard, et al.
configuration: --enable-libdirac --enable-libschroedinger --enable-libx264 --enable-gpl
libavutil 50. 3. 0 / 50. 3. 0
libavcodec 52.31. 2 / 52.31. 2
libavformat 52.34. 0 / 52.34. 0
libavdevice 52. 2. 0 / 52. 2. 0
libswscale 0. 7. 1 / 0. 7. 1
built on Jun 15 2009 18:19:46, gcc: 4.3.3

Australia

Australia

Original Input:

australia.mkv

VIDEO: [avc1] 1920x816 24bpp 23.976 fps, Avg Bitrate 6252 kbit/s

Verlustfrei Transcodiert:

$ mencoder australia.mkv -ovc lavc -lavcopts vcodec=ffvhuff:vstrict=-1:pred=2:context=1:format=422p -nosound -o australia_huff.mkv
--> Video stream: 165493.422 kbit/s (20686677 B/s) size: 656597040 bytes (627M) 31.740 secs 761 frames

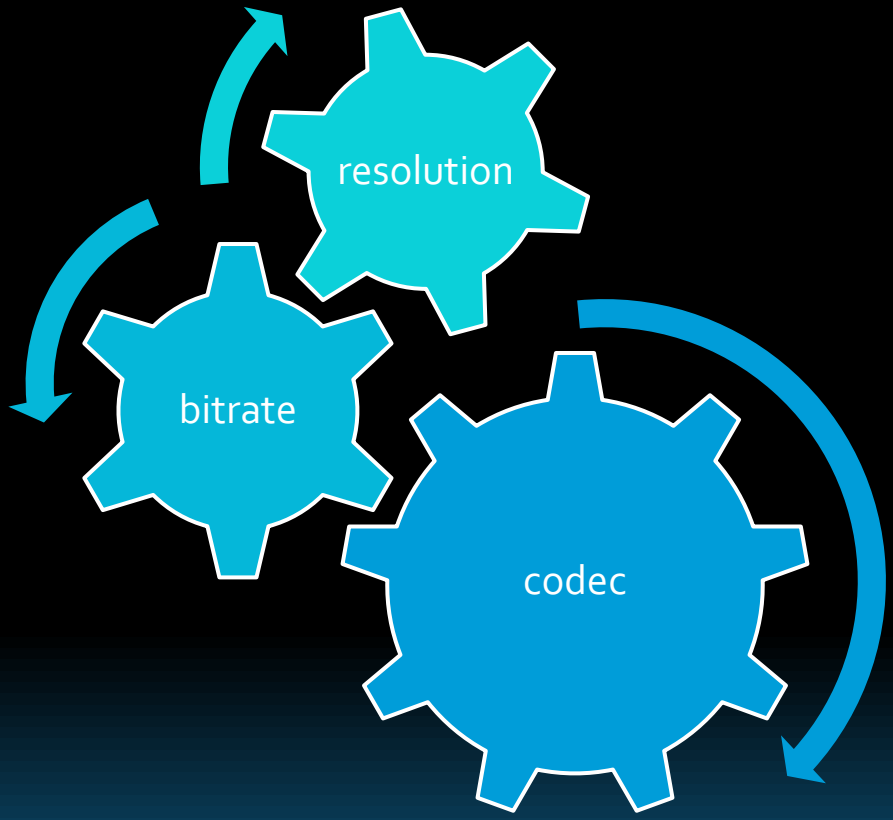
Family Guy Intro

x264

720x544

avg: 1180 kbit/s

Encoding Parameter



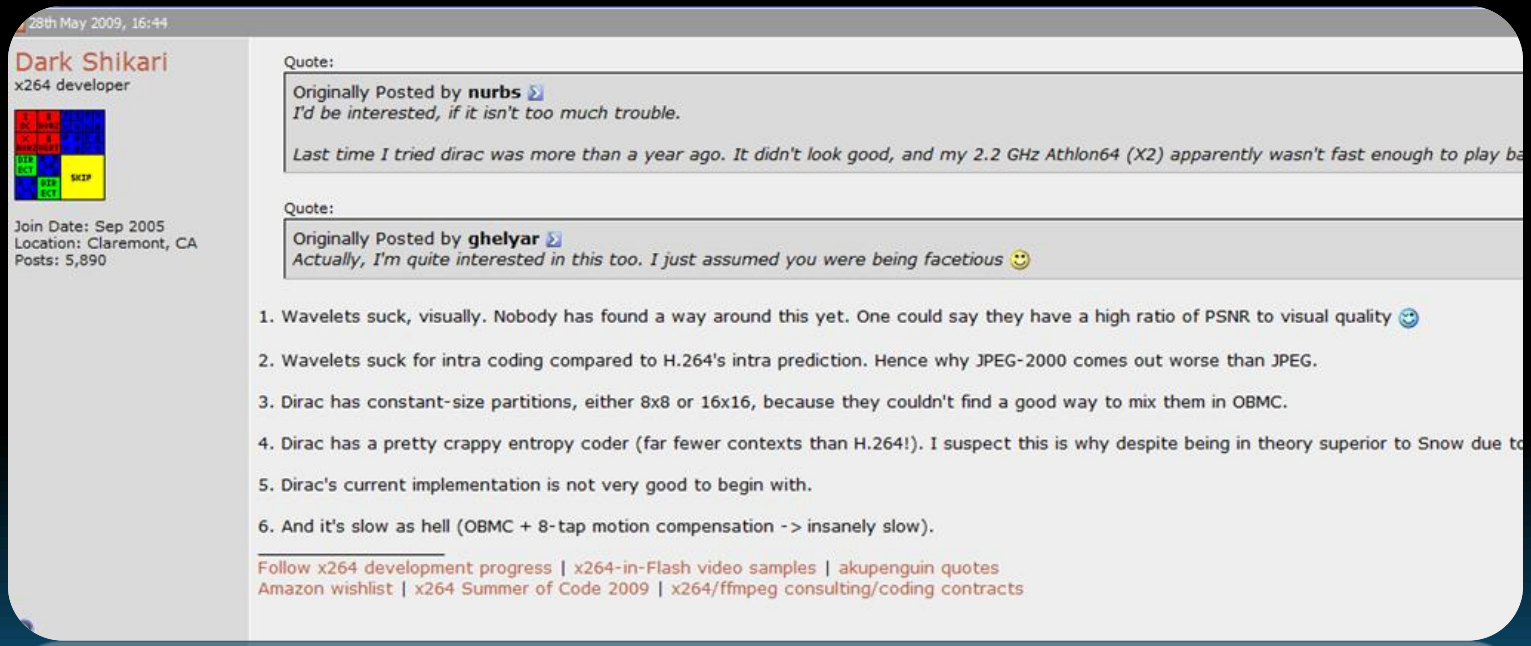
ffmpeg -i input.mkv -vcodec libschroedinger -s 720x480 -b 2000k output.mkv

ffmpeg -i input.mkv -vcodec libx264-s 720x480 -b 2000k output.mkv

DEMO

Kritik

Doom9.org



28th May 2009, 16:44

Dark Shikari
x264 developer

Join Date: Sep 2005
Location: Claremont, CA
Posts: 5,890

Quote:

Originally Posted by **nurbs**
I'd be interested, if it isn't too much trouble.

Last time I tried dirac was more than a year ago. It didn't look good, and my 2.2 GHz Athlon64 (X2) apparently wasn't fast enough to play ba

Quote:

Originally Posted by **ghelyar**
Actually, I'm quite interested in this too. I just assumed you were being facetious

1. Wavelets suck, visually. Nobody has found a way around this yet. One could say they have a high ratio of PSNR to visual quality

2. Wavelets suck for intra coding compared to H.264's intra prediction. Hence why JPEG-2000 comes out worse than JPEG.

3. Dirac has constant-size partitions, either 8x8 or 16x16, because they couldn't find a good way to mix them in OBMC.

4. Dirac has a pretty crappy entropy coder (far fewer contexts than H.264!). I suspect this is why despite being in theory superior to Snow due to

5. Dirac's current implementation is not very good to begin with.

6. And it's slow as hell (OBMC + 8-tap motion compensation -> insanely slow).

Follow x264 development progress | x264-in-Flash video samples | akupenguin quotes
Amazon wishlist | x264 Summer of Code 2009 | x264/ffmpeg consulting/coding contracts

Dark Shikari (x264 developer)

(28th May 2009, 16:44)

"1. Wavelets suck, visually. Nobody has found a way around this yet. One could say they have a high ratio of PSNR to visual quality
2. Wavelets suck for intra coding compared to H.264's intra prediction. Hence why JPEG-2000 comes out worse than JPEG.
3. Dirac has constant-size partitions, either 8x8 or 16x16, because they couldn't find a good way to mix them in OBMC. (Overlapped block-based motion compensation)
4. Dirac has a pretty crappy entropy coder (far fewer contexts than H.264!). I suspect this is why despite being in theory superior to Snow due to having B-frames, it often comes out worse.
5. Dirac's current implementation is not very good to begin with.
6. And it's slow as hell (OBMC + 8-tap motion compensation -> insanely slow)."

Ben Waggoner (Silverlight Video Strategist, Microsoft)

"The nut no one has cracked has been to get decent motion estimation with a wavelet codec, while DCT is very good at that: small blocks + block based motion estimation are a great match. A wavelet transform touches a **much** bigger image area, so motion doesn't map to the transform at all.

So, near as I can tell is that wavelet codecs bet that wavelets > dct for intra by a big enough margin that dct > wavelets for inter won't matter that much.

However, if you think about the average video encode, what's the ratio of bits spent on intra blocks to predicted blocks? And lets say you made the intra blocks 2x as efficient while reducing the efficiency of predicted blocks by 20%? Probably still a lousy deal.

And wavelet motion estimation is more than 20% less efficient than the best dct motion estimation, while wavelet intra coding isn't anywhere near 2x as efficient as the best dct intra coding."

Quellen

- http://diracvideo.org
  - http://diracvideo.org/download/press/diracoverview-ibc2008.pdf
  - http://diracvideo.org/download/specification/dirac-spec-latest.pdf
  - http://diracvideo.org/download/specification/diracpro-spec-latest.pdf
- http://www.bbc.co.uk/rd/projects/dirac/index.shtml
- http://dirac.sourceforge.net/documentation/algorithm/algorithm/toc.htm
- http://www.numediatechnology.com/
- http://hdmasters2007.com/pdf/Presentations/HDM2007_Wilson-HDDC.pdf