Graphische Datenverarbeitung

Bildkompression & Dateiformate

Prof. Dr. Elke Hergenröther

Gründe für eine Kompression

Unkomprimierte Rasterbilder benötigen:

- viel Speicherplatz
- hohe Bandbreite zur Übertragung

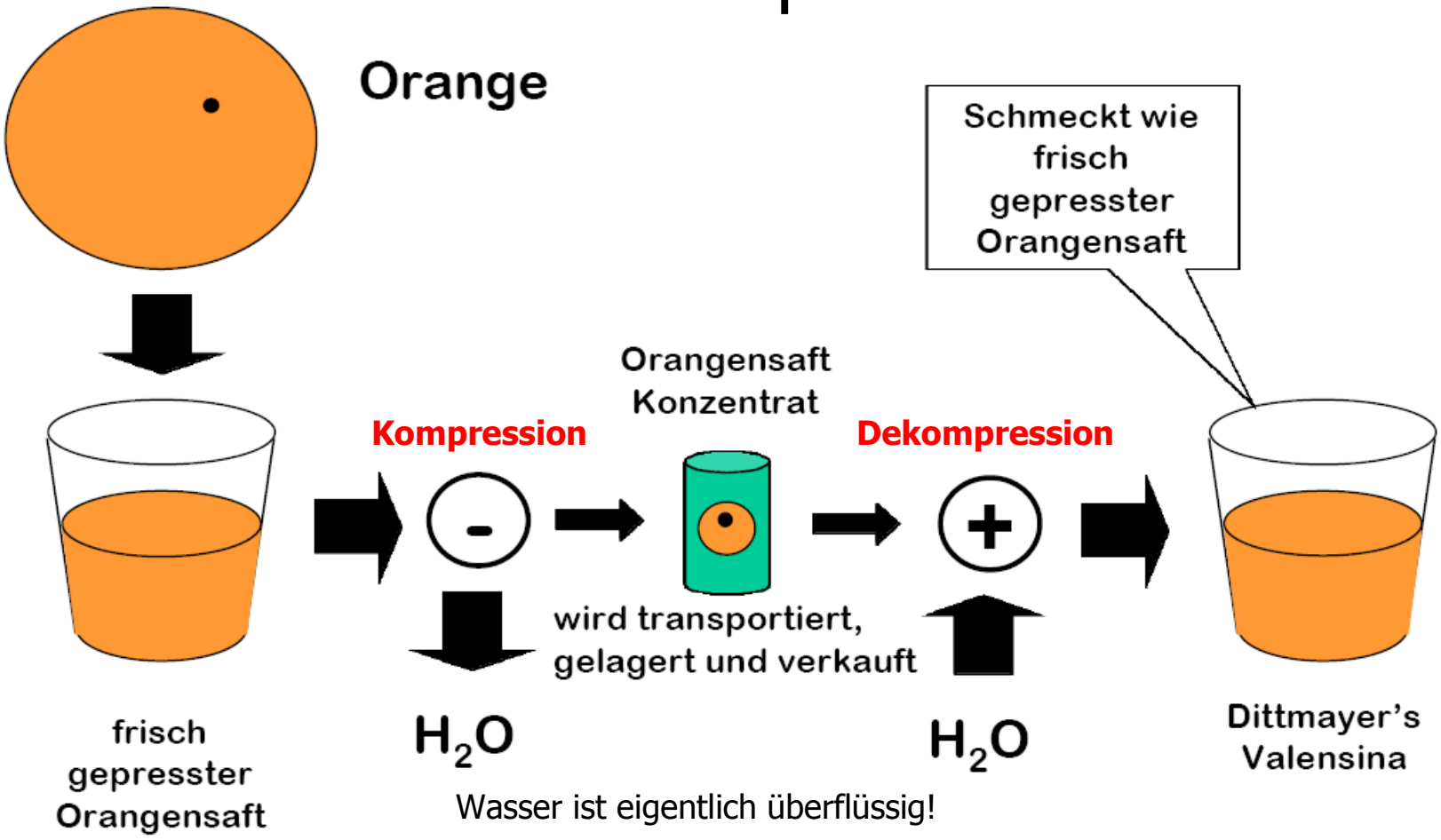
Beispiel:

Größe eines Bildes (oder einer Bildserie):
Breite x Höhe x Farbtiefe (x Bilder pro Sekunde)

Film mit 320 x 200 Pixeln, 24 Bit Farbtiefe, Bildrate 25 Bilder pro Sek.:
Datenvolumen von 4,6 MB/Sek. = 16,1 GB/h

⇨Datenkompression erforderlich.

Was ist Kompression?



Wasser ist eigentlich überflüssig!

Aus: Digitales Video, Klaus Diepold / Tobias Oelbaum

Kompressionsverfahren

Nicht verlustbehaftete Verfahren:

- Run Length Encoding (RLE)
- Lempel-Ziv-Welch (LZW)
- Huffman Kodierung

Verlustbehaftete Verfahren:

- JPEG
- Fraktale Kompression
- JPEG 2000 („Wavelet-Kompression“)

1. Unterscheidungskriterium für Kompressionsverfahren

Kriterium einer Kompression: Redundanz oder Irrelevanz reduzierend

Aus dem Duden:

- **Redundanz:**
  Überreichlichkeit, Überfluss, Üppigkeit ... *mehrfache Kennzeichnung der selben Information*.
  (nennt man auch Entropie-Kodierung)

- **Irrelevanz:**
  Unwichtigkeit, Bedeutungslosigkeit

Übersicht: Bildkompression und Dateiformate

- Kompression allgemein
- Nicht Verlustbehaftete Verfahren (*Redundanzreduzierende Verf.*)
  - RLE
  - LZW
  - Huffman Codierung
- Verfahren, die zur verlustbehafteten Kompression genutzt werden können (*Irrelevanzreduzierende Verfahren*):
  - Fraktale Kompression
  - Diskrete Cosinus Transformation
  - Wavelet Transformation

Achtung: Prinzipiell sind beide Verfahren verlustfrei!

- Bilddatenformate

RLE (Run Length Encoding)

RLE = Lauflängencodierung

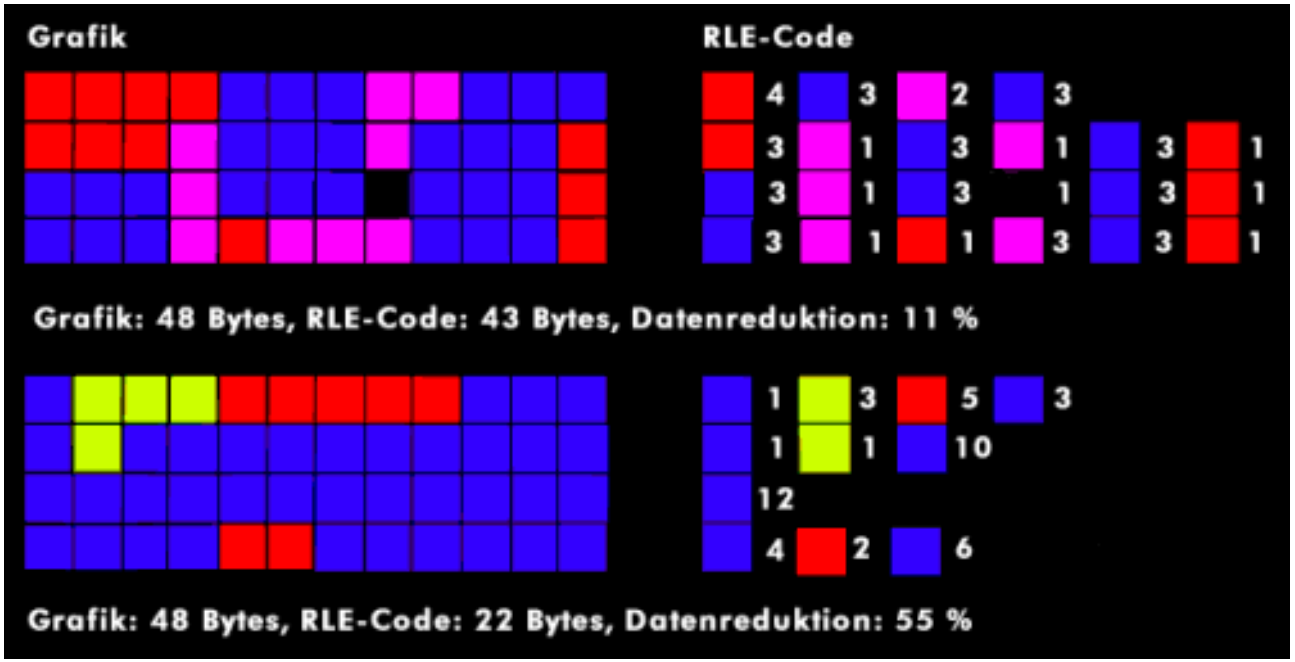
- Nutzt die Eigenschaft aus, das viele Daten aus einer Folge identischer Bytes bestehen
- Folgen gleicher Bytes werden durch Datenpaare codiert
- Datenpaar besteht aus: Zeichen und Anzahl

- Beispiel: CCCCXXXXXXYYYZZZ

4C6X3Y3Z

RLE (Run Length Encoding)

RLE Zeilenweise berechnet



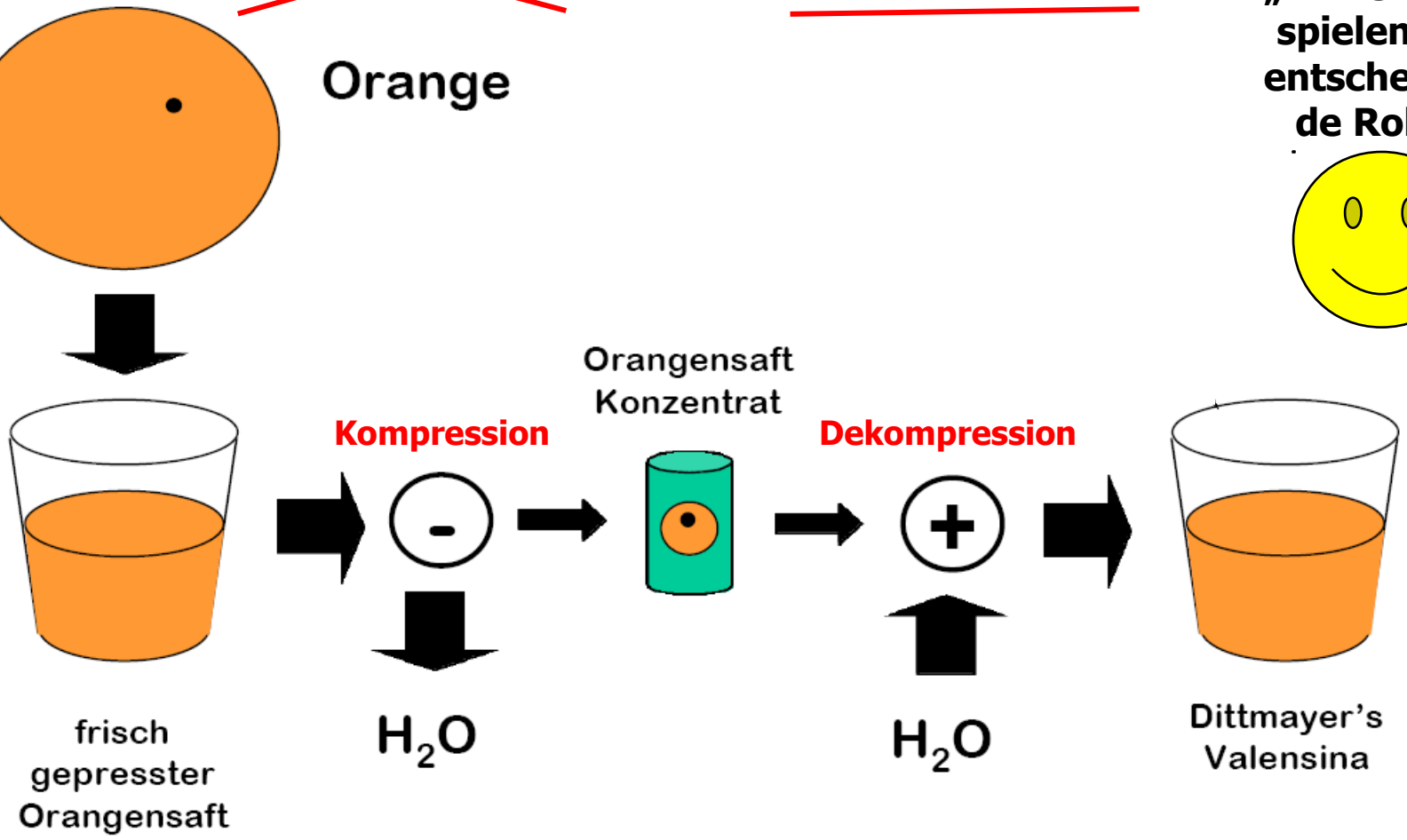
RLE (Run Length Encoding)

Lauflängen Codierung des Binärstroms:

- 0 und 1 wechseln sich ab
- daher genügt die Angabe, wie viel 1 und 0 folgen
- Codierer muss dem Decodierer aber mitteilen, ob er mit 0 oder 1 anfängt

~~Redundanz~~ oder Irrelevanz?

„Anwender" spielen die entscheiden de Rolle!



Was gibt es noch an Unterscheidungsmerkmalen?

Bei den Codierungsverfahren:

- Eindimensionale (zeilenorientierte) Verfahren
- Zweidimensionale (blockorientierte) Verfahren
- Globale Verfahren

Bei den Decodierungsverfahren:

- sequentiell
- progressiv

RLE (Run Length Encoding)

- Verlustfrei oder Verlustbehaftet?
- Redundanz- oder irrelevanzreduzierend?
- Symmetrisch oder asymmetrisch?
- Ein- oder zweidimensional oder global?

Übersicht: Bildkompression und Dateiformate

- Kompression allgemein
- Nicht Verlustbehaftete Verfahren (*Redundanzreduzierende Verf.*)
  - RLE
  - LZW
  - Huffman Codierung
- Verfahren, die zur verlustbehafteten Kompression genutzt werden können (*Irrelevanzreduzierende Verfahren*):
  - Fraktale Kompression
  - Diskrete Cosinus Transformation
  - Wavelet Transformation

Achtung: Prinzipiell sind beide Verfahren verlustfrei!

- Bilddatenformate

LZW (Lempel-Ziv-Welch)

Verfahren basiert auf der Suche nach sich wiederholenden Zeichenketten!

Idee:

Gleiche Strings werden nur beim ersten Auftreten direkt codiert, danach erfolgt ein Rückverweis auf bereits vorhandene identische Zeichenketten.

LZW (Lempel-Ziv-Welch)

Prinzip des Verfahrens basiert auf einer Art Wörterbuch

bestehend aus:

- **Statischem Teil**; wird mit Übertragen
- **Dynamischem Teil**; wird sowohl beim Codieren als auch beim Decodieren dynamisch aufgebaut

Beispiel nächste Folie, Quelle: https://de.wikipedia.org/wiki/Lempel-Ziv-Welch-Algorithmus

Pseudocode des LZW Algorithmus

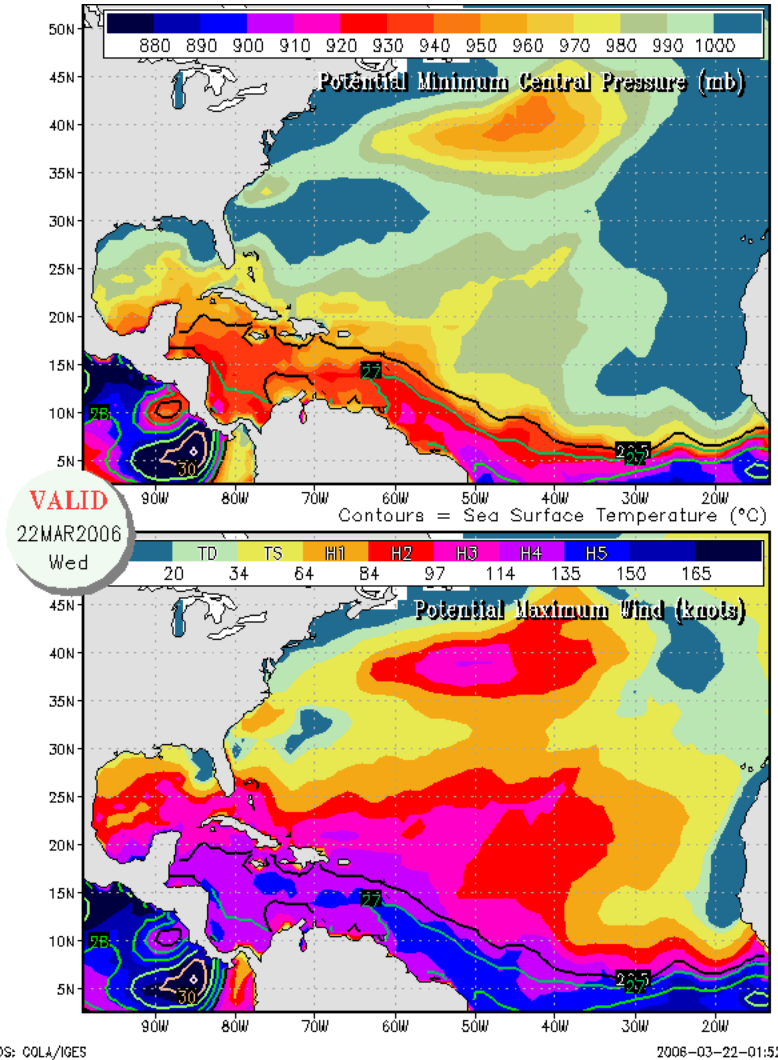
```
initialisiere Mustertabelle mit (<leeres Muster>+zeichen) für alle Zeichen
    muster := <leeres Muster>
    solange noch Zeichen verfügbar
        zeichen := lies nächstes Zeichen
        wenn (muster+zeichen) in Mustertabelle dann
            muster := (muster+zeichen)
        sonst
            füge (muster+zeichen) zur Mustertabelle hinzu
            Ausgabe muster
            muster := zeichen
    wenn muster nicht <leeres Muster> dann
        Ausgabe muster
```

Zeichenkette	gefundener Eintrag	Ausgabe	neuer Eintrag
LZWLZ78LZ77LZCLZMWLZAP	L	L	LZ (wird zu <256>)
ZWLZ78LZ77LZCLZMWLZAP	Z	Z	ZW (wird zu <257>)
WLZ78LZ77LZCLZMWLZAP	W	W	WL (wird zu <258>)
LZ78LZ77LZCLZMWLZAP	LZ (= <256>)	<256>	LZ7 (wird zu <259>)
78LZ77LZCLZMWLZAP	7	7	78 (wird zu <260>)
8LZ77LZCLZMWLZAP	8	8	8L (wird zu <261>)
LZ77LZCLZMWLZAP	LZ7 (= <259>)	<259>	LZ77 (wird zu <262>)
7LZCLZMWLZAP	7	7	7L (wird zu <263>)
LZCLZMWLZAP	LZ (= <256>)	<256>	LZC (wird zu <264>)
CLZMWLZAP	C	C	CL (wird zu <265>)
LZMWLZAP	LZ (= <256>)	<256>	LZM (wird zu <266>)
MWLZAP	M	M	MW (wird zu <267>)
WLZAP	WL (= <258>)	<258>	WLZ (wird zu <268>)
ZAP	Z	Z	ZA (wird zu <269>)
AP	A	A	AP (wird zu <270>)
P	P	P	-

LZW (Lempel-Ziv-Welch)

- Wird von vielen Graphikformaten u.a. GIF genutzt
- Variante davon enthalten in GZIP
- PNG LZW ähnliches Verfahren

Welche Bilder sind für die LZW Komprimierung geeignet?



LZW

- Verlustfrei oder Verlustbehaftet?
- Redundanz- oder irrelevanzreduzierend?
- Symmetrisch oder asymmetrisch?
- Ein- oder zweidimensional oder global?

Übersicht: Bildkompression und Dateiformate

- Kompression allgemein
- Nicht Verlustbehaftete Verfahren (*Redundanzreduzierende Verf.*)
  - RLE
  - LZW
  - Huffman Codierung
- Verfahren, die zur verlustbehafteten Kompression genutzt werden können (*Irrelevanzreduzierende Verfahren*):
  - Fraktale Kompression
  - Diskrete Cosinus Transformation
  - Wavelet Transformation
- Bilddatenformate

Achtung: Prinzipiell sind beide Verfahren verlustfrei!

Huffman Codierung

Idee:

Häufig vorkommenden Symbolen werden kürzere Codes zugeordnet als seltener vorkommenden.

Beispiel aus dem Morsealphabet:

e = .

y = -.-.

Huffman Codierung

1. Schritt:

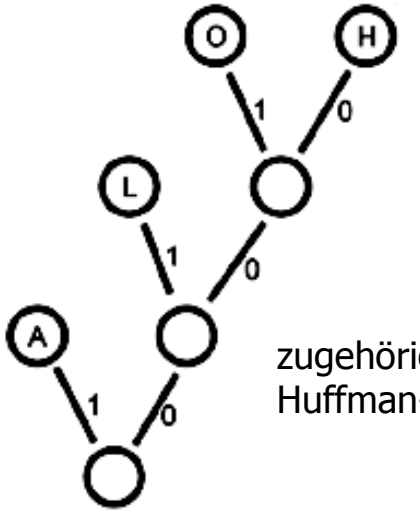
Eine Tabelle in der die relativen Wahrscheinlichkeiten für das Auftreten eines Symbols eingetragen werden, wird angelegt.

Zu codierendes Wort:

HAAAALLO (8 Buchstaben)

Wahrscheinlichkeitstabelle:

H	1/8
A	4/8
L	2/8
O	1/8



zugehöriger Huffman-Baum

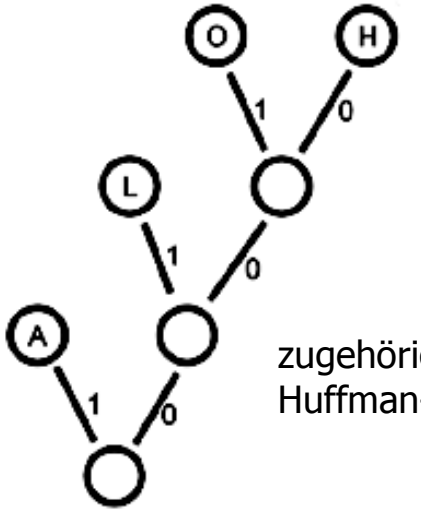
Huffman Codierung

2. Schritt:

Codierung in Abhängigkeit von der Wahrscheinlichkeit mit der die Symbole auftreten:

Beispiel:

H	2/8	0		H	4/8	00		H	8/8	000
O		1		O		01		O		001
A	4/8			L		1		L		01
L	2/8			A	4/8			A		1



zugehöriger Huffman-Baum

Kompressionsverfahren

Nicht verlustbehaftete Verfahren:

- Run Length Encoding (RLE)
- Lempel-Ziv-Welch (LZW)
- Huffman Kodierung

Verlustbehaftete Verfahren:

- Fraktale Kompression
- JPEG
- *JPEG 2000 („Wavelet-Kompression")*

JPEG

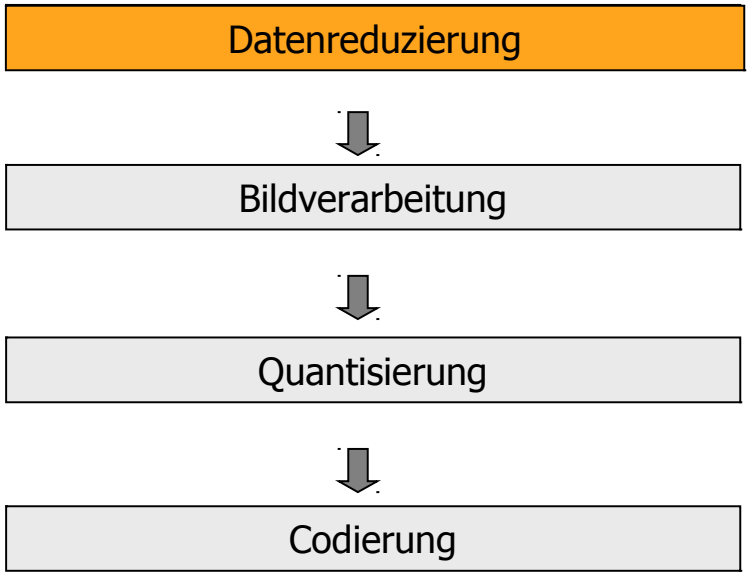
JPEG ist gleichzeitig der Name

- einer Standardisierungsorganisation,
- eines Grafikformates und
- eines Bildkompressionsverfahren

JPEG = Joint Photographic Experts Group

JPEG

Die Komprimierung der Blöcke erfolgt in vier Schritten:



JPEG

1. Schritt: Datenreduzierung

- Umwandlung: RGB Farbmodell in YUV Farbmodell
- entspricht einer Datenreduzierung von 50%

JPEG: YUV- oder YIQ-Modell

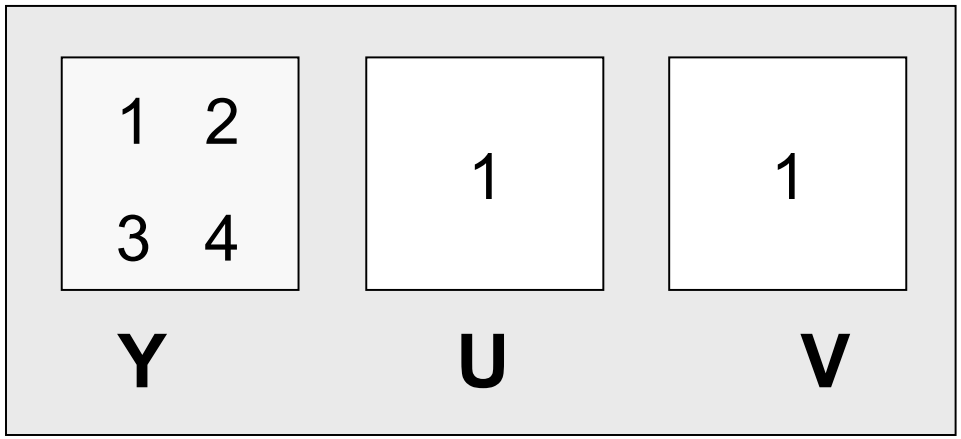
- Das menschliche Auge besitzt mehr Rezeptoren für Helligkeitswerte als für Farbänderung.
- Daher können Farbwerte in geringerer Auflösung als Helligkeitswerte vorliegen.
- YUV- oder YIQ-Modelle unterstützen diese Eigenschaft:

  Y: Helligkeit (Grauwerte)

  I,Q bzw. U, V: Farbcodierung (Chrominanzen)

JPEG: YUV- oder YIQ-Modell

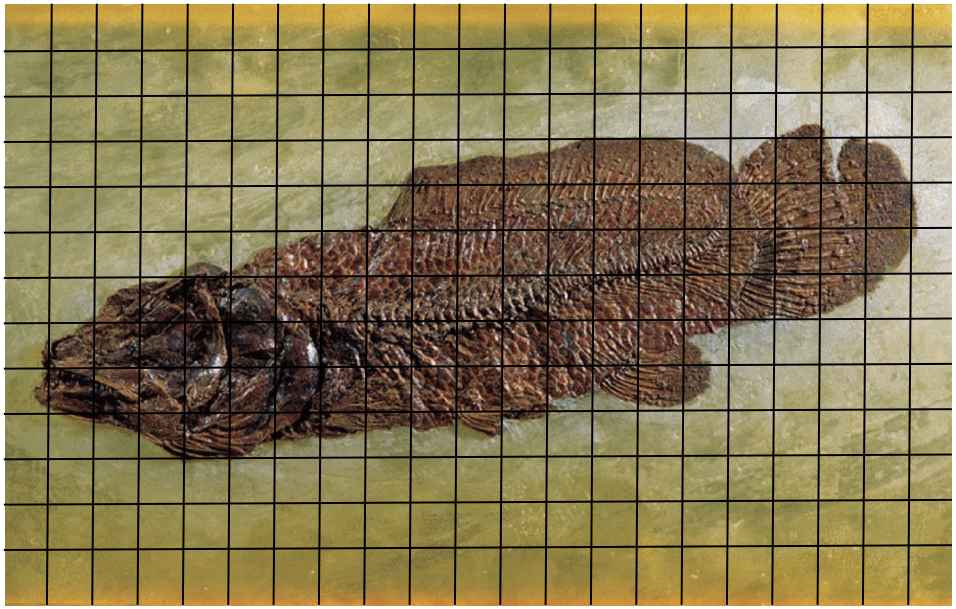
Codierung: 4:1:1-Farbraum im YUV-Modell



Komprimierungsrate: 50% (6 statt 12 Werte für 4 Pixel)

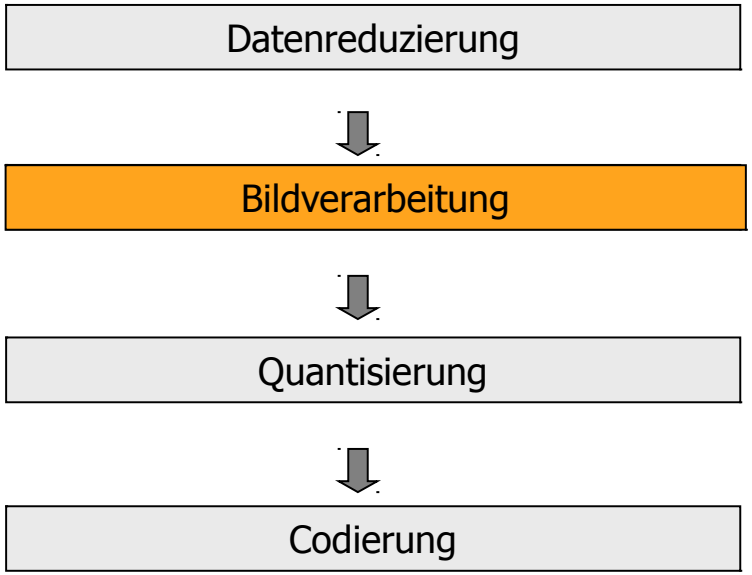
JPEG: Blockorientiertes Verfahren

Zerlegen des Originalbildes in 8 x 8 große Blöcke



JPEG

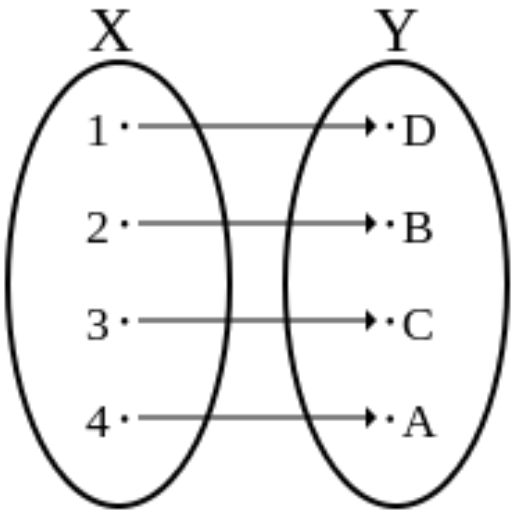
Die Komprimierung der Blöcke erfolgt in vier Schritten:



JPEG

2. Schritt: Diskrete Kosinus-Transformation

- Abbildung in den Frequenzraum (Ortsfrequenzbereich)
- Abbildung ist bijektiv (verlustfrei).



Zerlegung eines Bildes in seine unterschiedlichen Frequenzen

niedrige Frequenzen

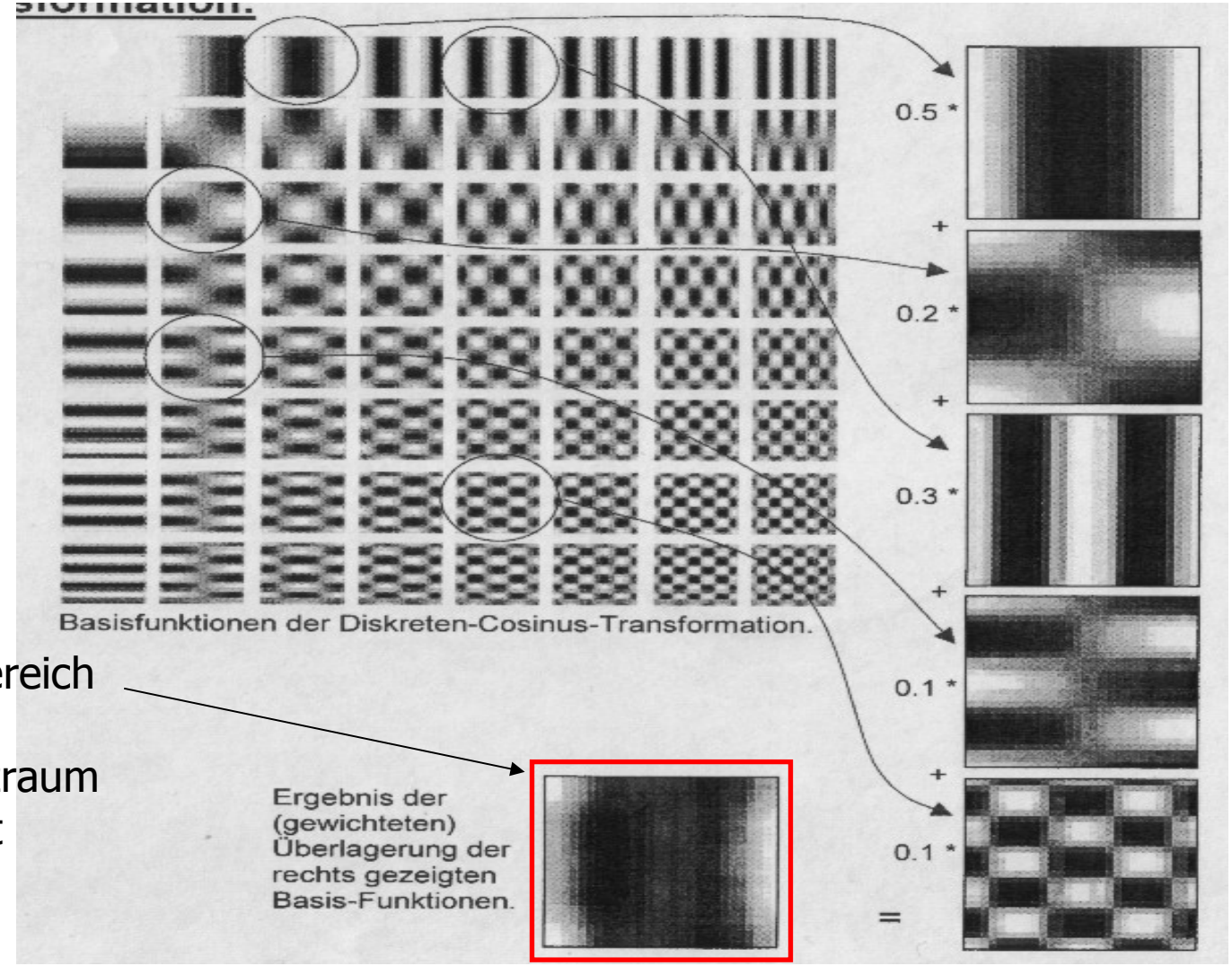
hohe Frequenzen

Überlagerungen

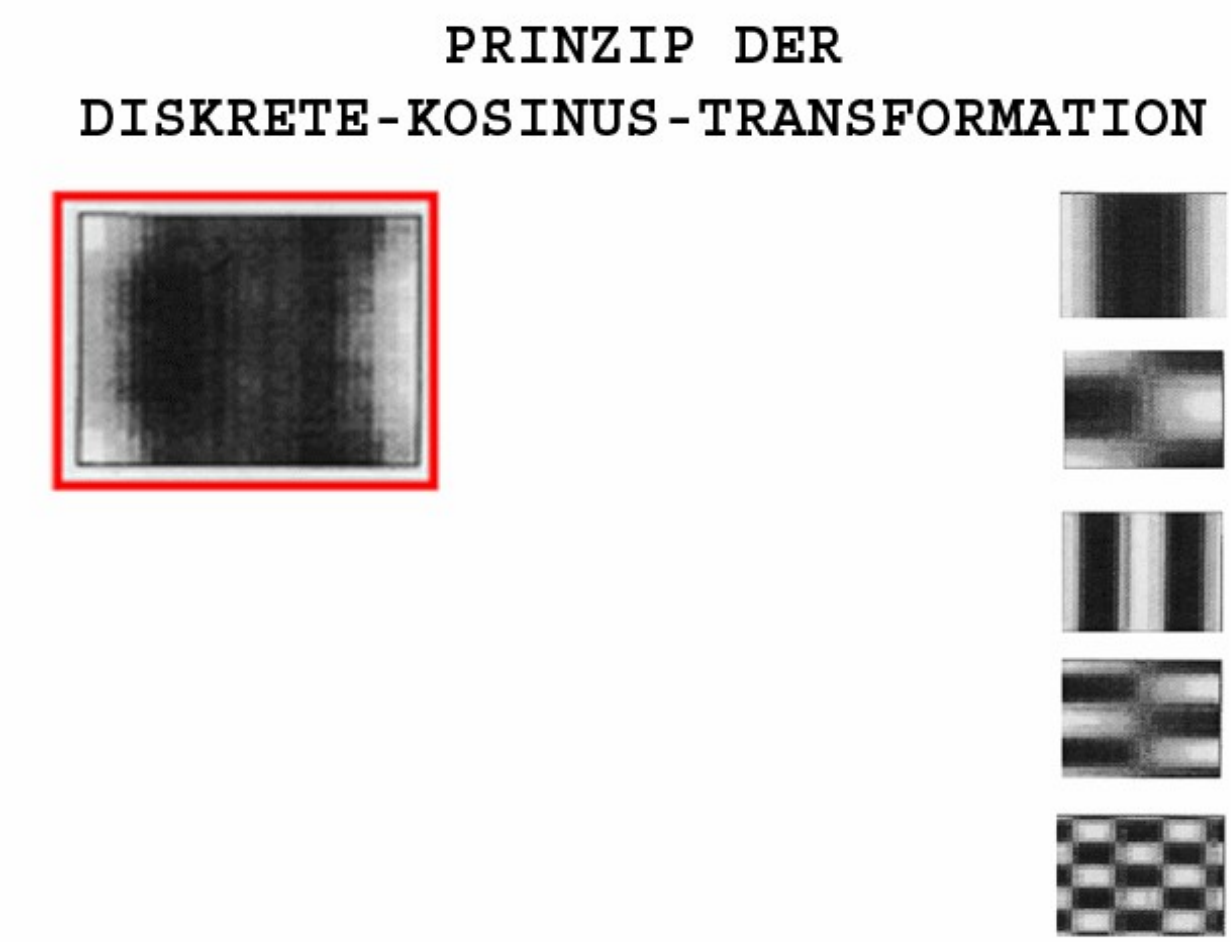
Bilder: Koll, MPEG-Video-Kompression

Relativ uninteressanter Bildinhalt!

Prinzip der Diskreten-Kosinus-Transformation



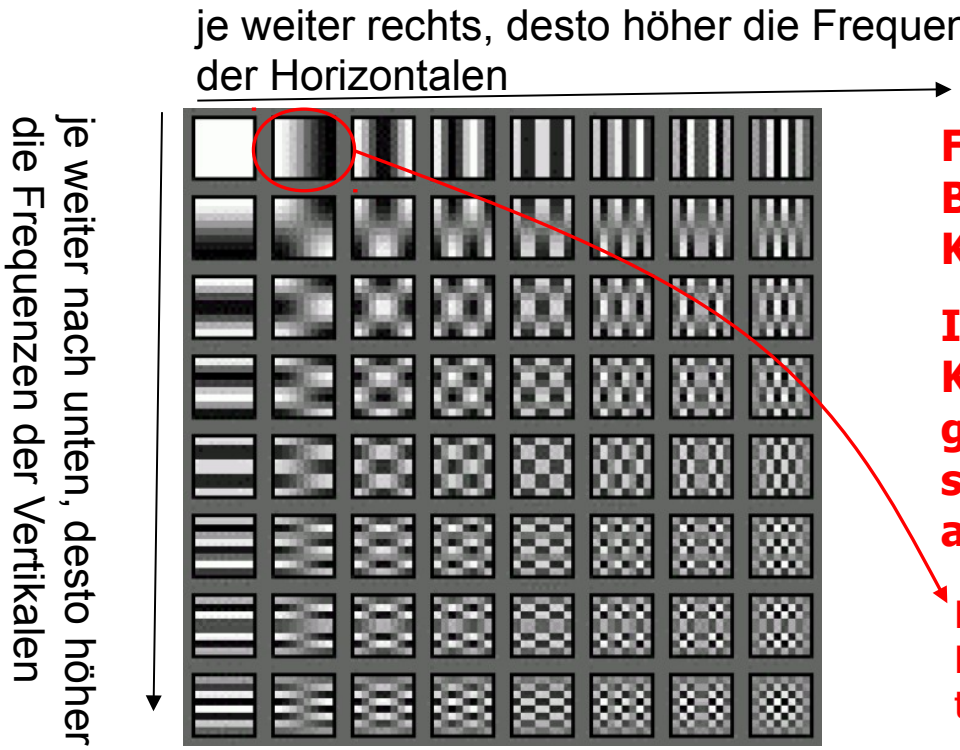
Dieser Bildbereich soll in den Ortsfrequenzraum transformiert werden!



von Marco Münch ... Danke :-)

Diskrete Kosinus-Transformation: Zusammenfassung

1. Bild wird in 8x8 Pixel große Blöcke aufgeteilt
2. Jeder Block wird in 64 vordefinierte „Basisfunktionen" zerlegt:



Basisfunktionen der Diskreten Cosinus Transformation

Für jeden 8x8 Pixel großen Block wird eine 8x8 große Koeffizientenmatrix erstellt!

In den einzelnen Feldern der Koeffizientenmatrix steht wie groß der Anteil der entsprechenden Basisfunktion am Block ist.

Bsp. Folien vorher:
Hier würde in der Koeffizientenmatrix 0,5 stehen

Diskrete Kosinus-Transformation

139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	161	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	161	163	158	158	158

Grauwertverteilung in einem 8x8 großen Block

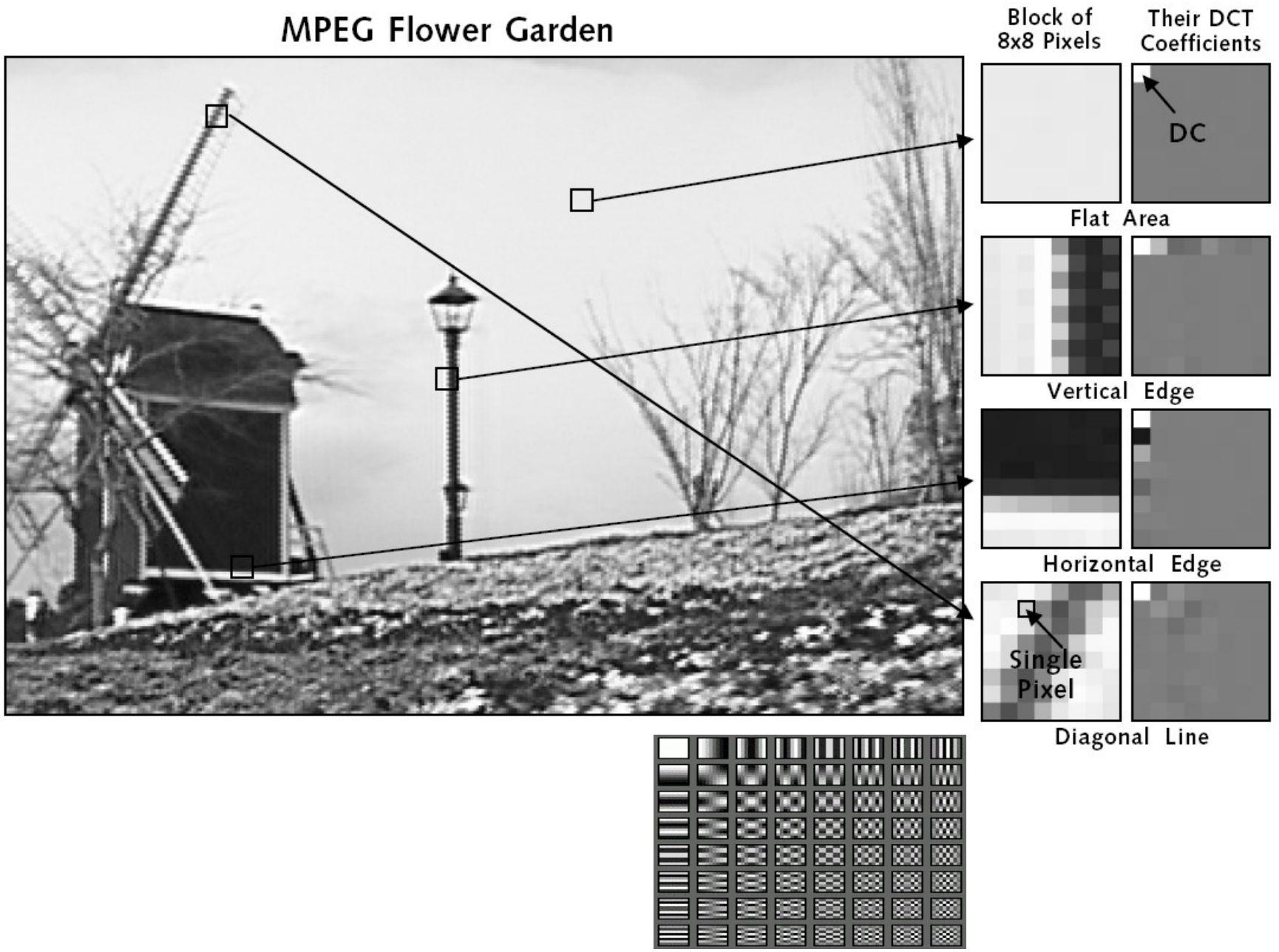
Mittelwert der Blockhelligkeit

(immer positiver Wert)

DCT

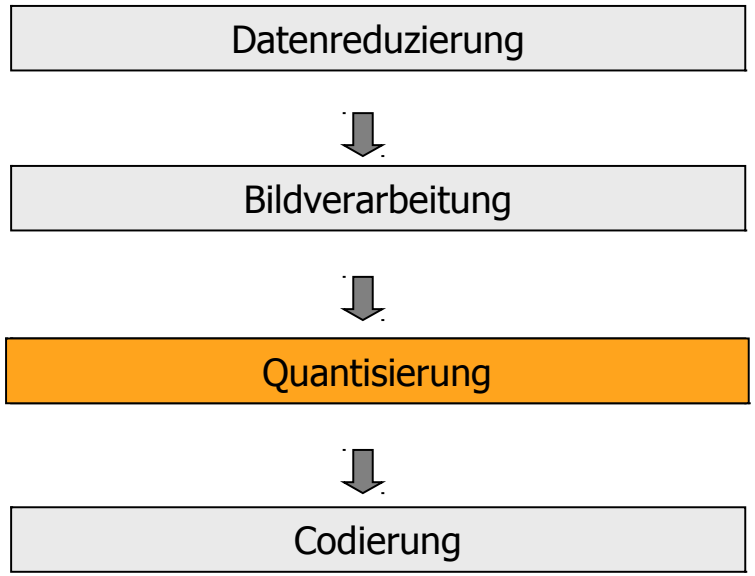
1260	-1	-12	-5	2	-2	-3	1
-23	-17	-6	-3	-3	0	0	-1
-11	-9	-2	2	0	-1	-1	0
-7	-2	0	1	1	0	0	0
-1	-1	1	2	0	-1	1	1
2	0	2	0	-1	1	1	-1
-1	0	0	-1	0	2	1	-1
-3	2	-4	-2	2	1	-1	0

Koeffizientenmatrix


MPEG Flower Garden
Block of 8x8 Pixels
Their DCT Coefficients
DC
Flat Area
Vertical Edge
Horizontal Edge
Single Pixel
Diagonal Line

JPEG

Die Komprimierung der Grafikdaten erfolgt in vier Schritten:



JPEG

3. Schritt: Quantisierung:

- Ganzzahlige Division der Koeffizienten durch vorgegebene Zahl (entweder für alle gleich oder aus Tabelle, die mitübertragen wird)
- Dadurch werden hochfrequente Anteile kontrolliert beseitigt.
- **Löschung von Information!**

JPEG

1260	-1	-12	-5	2	-2	-3	1
-23	-17	-6	-3	-3	0	0	-1
-11	-9	-2	2	0	-1	-1	0
-7	-2	0	1	1	0	0	0
-1	-1	1	2	0	-1	1	1
2	0	2	0	-1	1	1	-1
-1	0	0	-1	0	2	1	-1
-3	2	-4	-2	2	1	-1	0

Beispiel: 1260 / 16 = 79

(Quantisieren => abschneiden, bzw. runden des Ergebnisses)

79	0	-1	0	0	0	0	0
-2	-1	0	0	0	0	0	0
-1	-1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantisierung (Ganzzahldivision) erfolgt durch folgende Tabelle:

16	11	10	16	24	40	51	61
14	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

JPEG

3. Schritt: Beispiel für Quantisierungsmatrizen

Luminanz (Y)

16	11	10	16	24	40	51	61
14	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Chrominanz (UV)

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

Quantisierung

Durch die Quantisierung in JPEG werden die hochfrequenten Bildanteile kontrolliert beseitigt.

JPEG Dateiformat

Beispiel einer Bilddatei:

```
ff d8 ff e0 00 10 4a 46 49 46 00 01 01 01 00 64
00 64 00 00 ff fe 00 2f 48 61 6e 64 6d 61 64 65
20 53 6f 66 74 77 61 72 65 2c 20 49 6e 63 2e 20
49 6d 61 67 65 20 41 6c 63 68 65 6d 79 20 76 31
2e 36 2e 31 0a ff db 00 43 00 05 04 04 04 04 03
05 04 04 04 06 05 05 06 08 0d 08 08 07 07 08 10
0b 0c 09 0d 13 10 14 13 12 10 12 12 14 17 1d 19
14 16 1c 16 12 12 1a 23 1a 1c 1e 1f 21 21 21 14
19 24 27 24 20 26 1d 20 21 20 ff c0 00 0b 08 01
e0 02 80 01 01 11 00 ff c4 00 6e 00 00 00 07 01
01 01 00 00 00 00 00 00 00 00 00 00 00 01 02 03
04 05 06 07 08 09 10 00 02 01 03 03 02 03 05 05
05 03 07 09 06 05 05 01 02 03 00 04 11 05 12 21
06 31 13 41 51 07 22 61 71 81 14 32 91 a1 b1 15
23 42 c1 d1 52 62 f0 08 16 24 33 72 82 e1 17 25
...
```

Tag Beispiele:

`ff d8:` Beginn der Datei

`ff db:` Quantisierungstabellen (DQT-Tag)

`...`

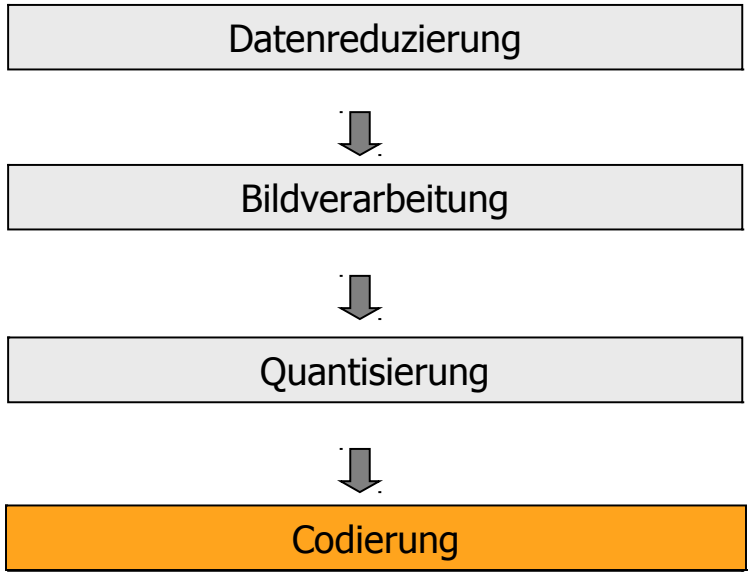
`ff d9:` Bildende

Mehr Information:

`http://jpeg.know-library.net/`

JPEG

Die Komprimierung der Grafikdaten erfolgt in vier Schritten:

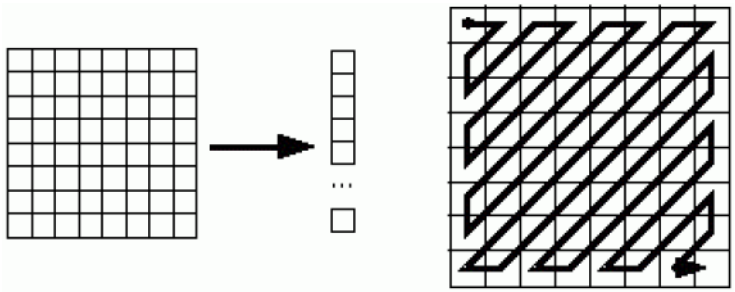


JPEG

4. Schritt: Entropiecodierung

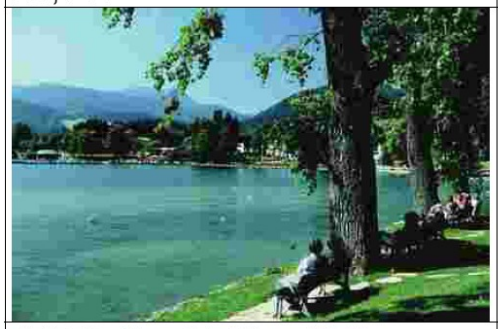
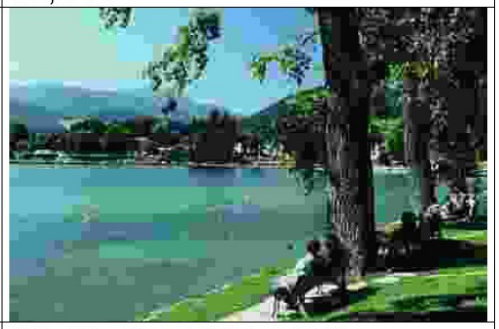
Codierung mit verlustfreier Kompression

- Werte werden in Zickzack Reihenfolge ausgelesen:

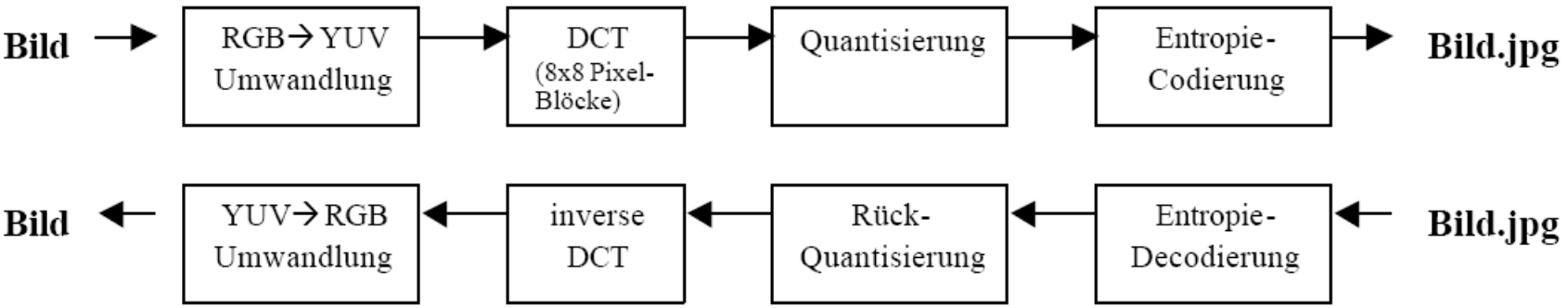


Beispiel einer ausgelesenen Koeffizientenmatrix (F(u,v)-Werte):

- durch **Lauflängen-Codierung (RLE)** (je mehr Koeffizienten == 0 sind, desto mehr Speicherplatz spart man!) und anschließender Huffman- bzw. Arithmetischer-Codierung komprimiert

	
JPEG, verlustfrei komprimiert: 123 K	JPEG, Qualitätsfaktor 75 %: 30,6 K
	
JPEG, Qual.faktor 50 %: 20,7 K	JPEG, Qual.faktor 25 %: 13,3 K
	
JPEG, Qual.faktor 10 %: 7,4 K	JPEG, Qual.faktor 5 %: 4,9 K

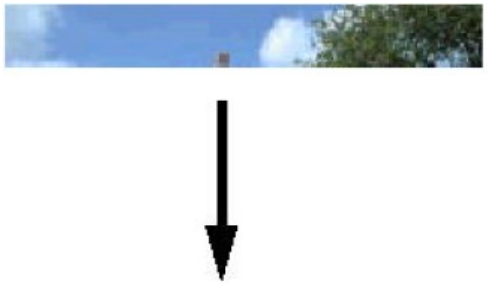
JPEG-Prozess



JPEG

Sequentieller Modus

sequentieller Bildaufbau (von oben nach unten)

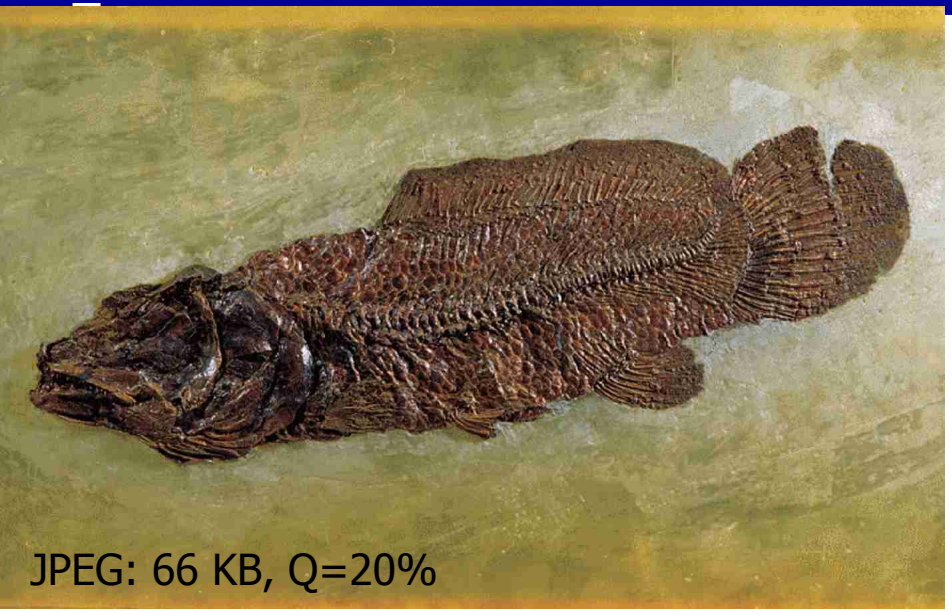


JPEG

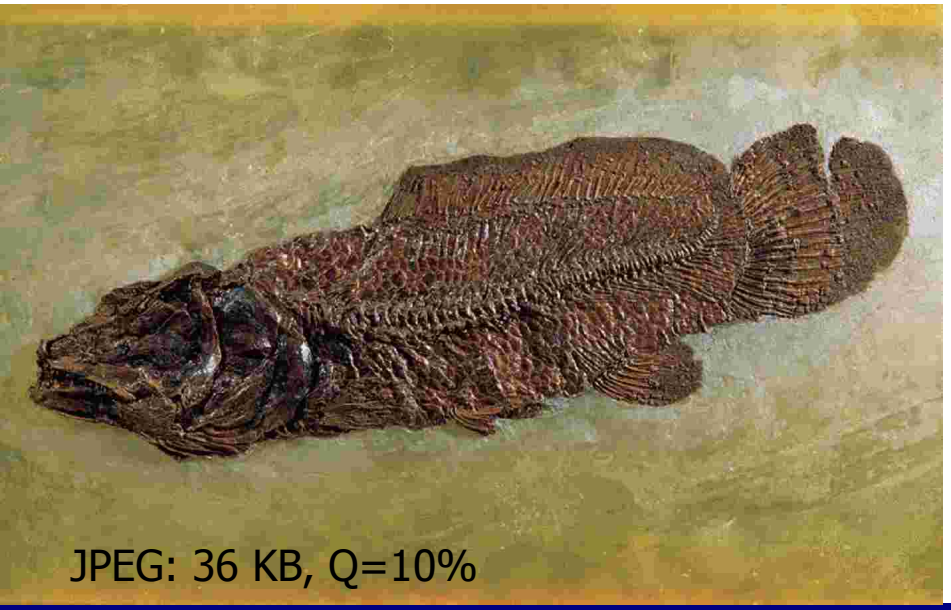
Progressiver Modus

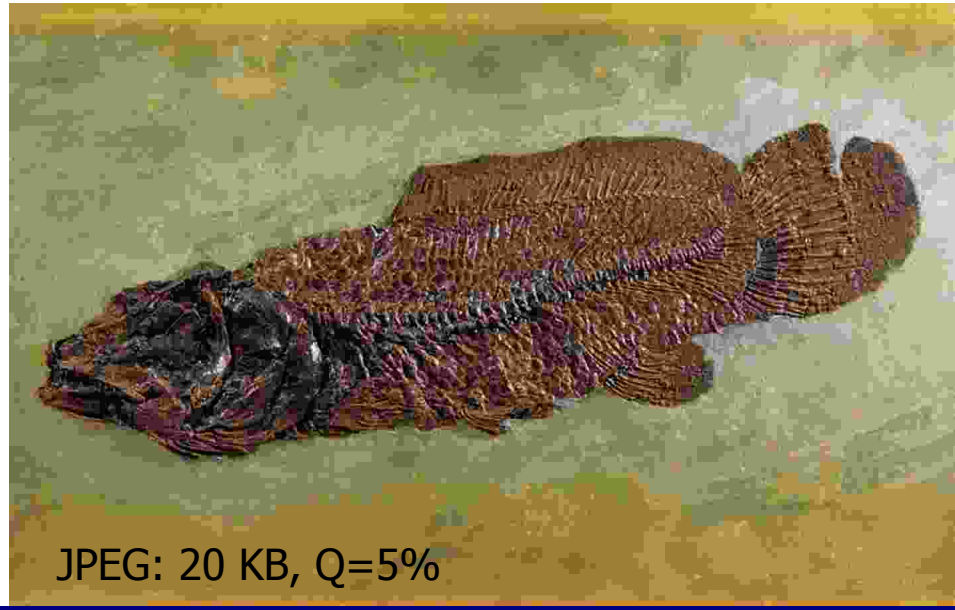
JPEG

- Verlustbehaftete JPEG-Kompression gut geeignet für Fotos (realistische Farbdarstellung)
- Schwächen bei Bildern mit:
  - scharfen Kontrasten
  - Kanten
  - bei künstlichen Bildern
  - Texten
  - detail-sensitiven Daten (z.B. medizinische Bilddaten)


JPEG: 66 KB, Q=20%


JPEG: 52 KB, Q=15%


JPEG: 36 KB, Q=10%


JPEG: 20 KB, Q=5%

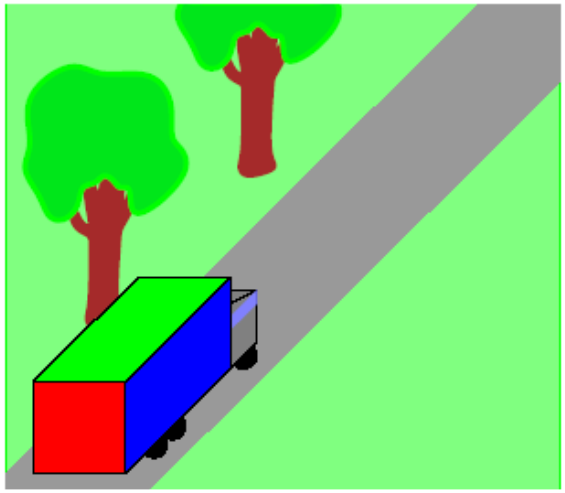
MPEG

Motion **P**icture **E**xperts **G**roup

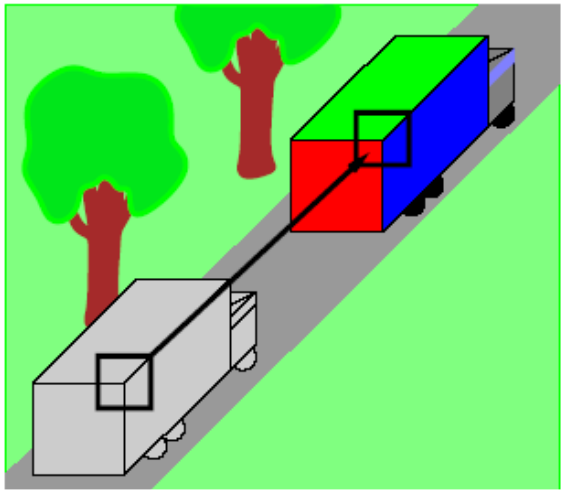
1988 mit dem Ziel gegründet:

- Standards für die codierte Repräsentation von Bewegtbildern, Audiodaten sowie deren Komposition zu entwickeln.
- Vorgestelltes Grundprinzip wird u. a. verwendet in: MPEG 1, 2 und 4

MPEG: Grundidee



Filmbild n



Filmbild n+1

1. Bild wird in 16x16 Pixel große Makroblöcke aufgeteilt
2. Verschiebung der Makroblöcke wird ermittelt
3. Bild wird aus den „verschobenen" Makroblöcken zusammen gesetzt
4. Differenz zwischen dem Bild n und dem neu entstanden Bild n+1 wird ermittelt (=Differenzbilder).
5. Das Differenzbild korrigiert das Bild n+1.

MPEG: Grundidee

Bild n

Intra-Frame (I-Frame)

intra innerhalb (lat.)

Bild n+1

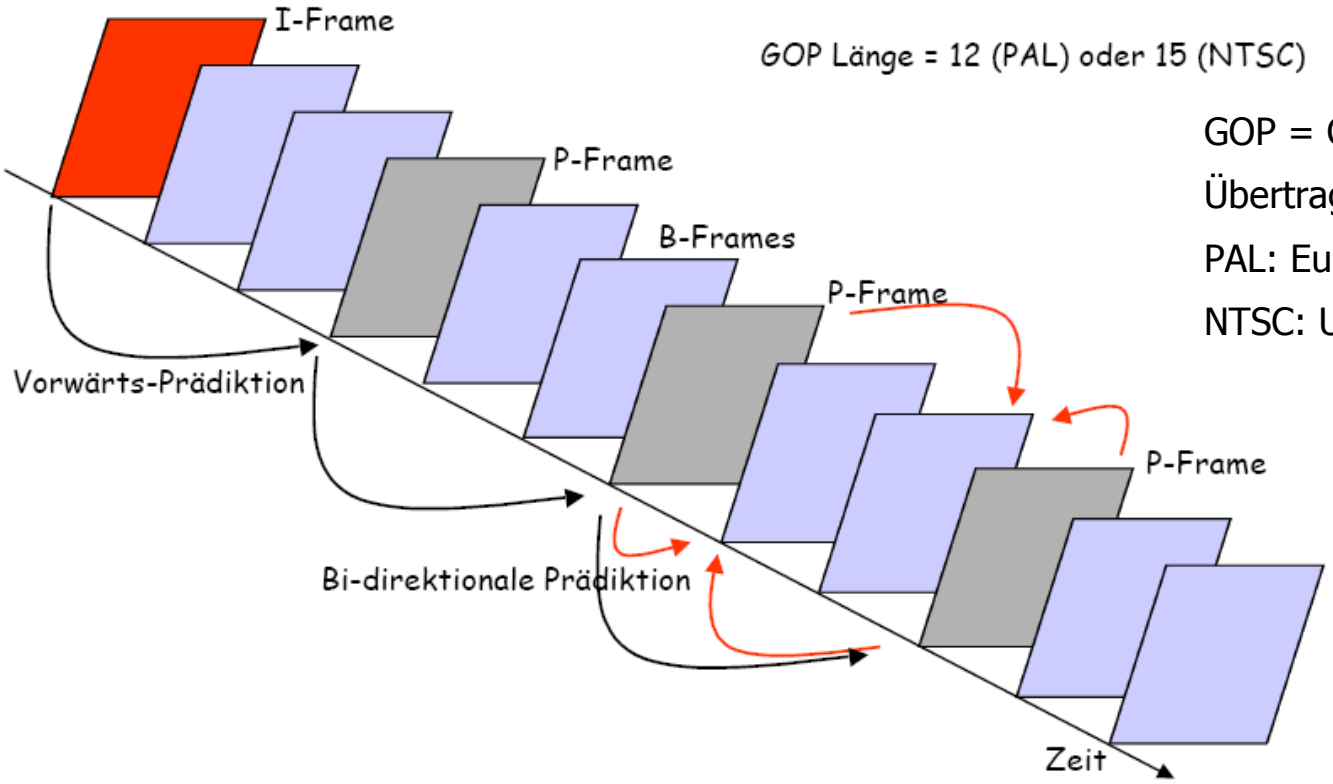
Verschiebungs-vektoren der Makroblöcke sind eingetragen

Differenzbild

Beide zusammen: Differenzbild und die Verschiebungsvektoren, ergeben zusammen-gesetzt das P-Frame

(prediction Prognose, Vorhersage (engl.))

MPEG: Grundidee



GOP Länge = 12 (PAL) oder 15 (NTSC)

GOP = Group of Pictures

Übertragungsstandards:

PAL: Europa & Südamerika

NTSC: USA & Japan

Für WS19/20:

Alle Kompressionsfolien ab Folie 63 sind nicht klausurrelevant!

Bilder in einer MPEG-Kompression

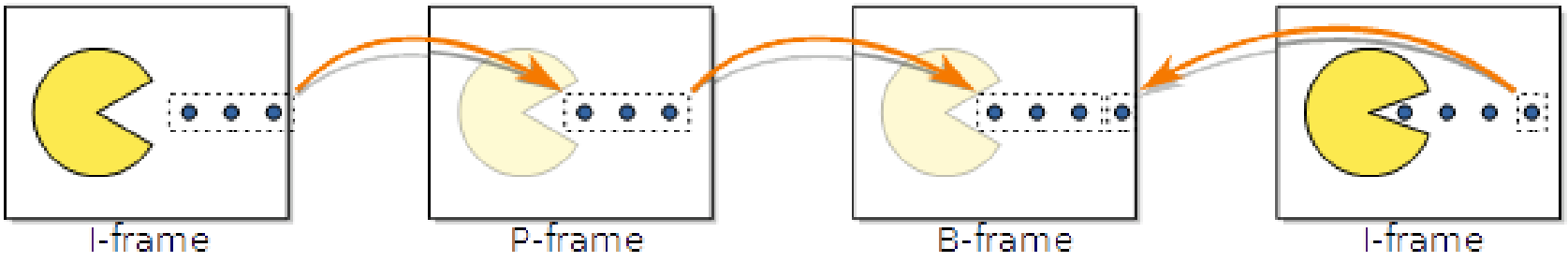
Intra-Frames (I-Frames):

komprimierte Originalbilder
(wird immer bei Szenenwechsel oder einem Schwenk genutzt)

Inter-Frames (zwischen (lat.)), P- und B-Frames:

- Predicted Pictures **(P-Frames)**:
  komp. Differenzbilder plus Verschiebungsvektoren nur in Bezug zum **vorangegangenen** I- bzw. P-Frame, also unidirektional
- Bidiretional Pictures **(B-Frames):**
  In einem B-Bild ist die Differenz zwischen dem vorhergehenden **oder** dem nachfolgenden Bild **oder** dem arithmetischen Mittel aus beiden gespeichert. Je nachdem welche Methode die besten Ergebnisse liefert.

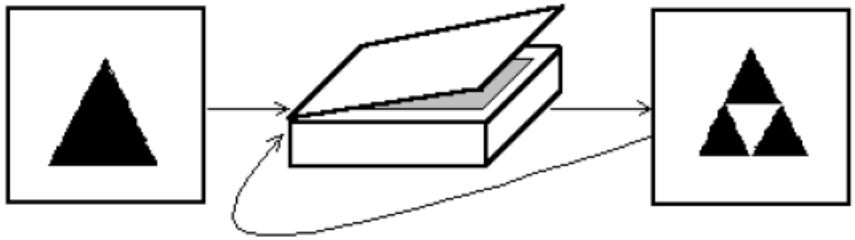
I-Frame, P-Frame und B-Frame



Eigenschaft der Fraktale: „Selbstähnlichkeit“

http://www.matheprisma.uni-wuppertal.de/Module/Fraktal/

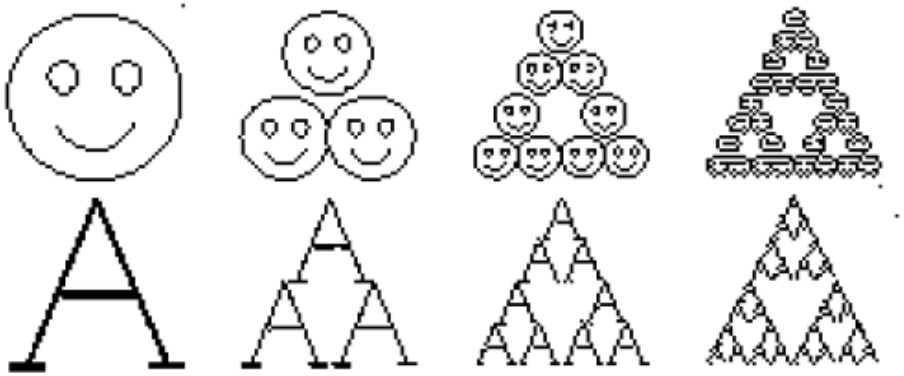
Weitere Eigenschaft: „Entstehung durch Iteration“



Kopiermaschine:
- Skalieren
- Rotieren
- Translieren

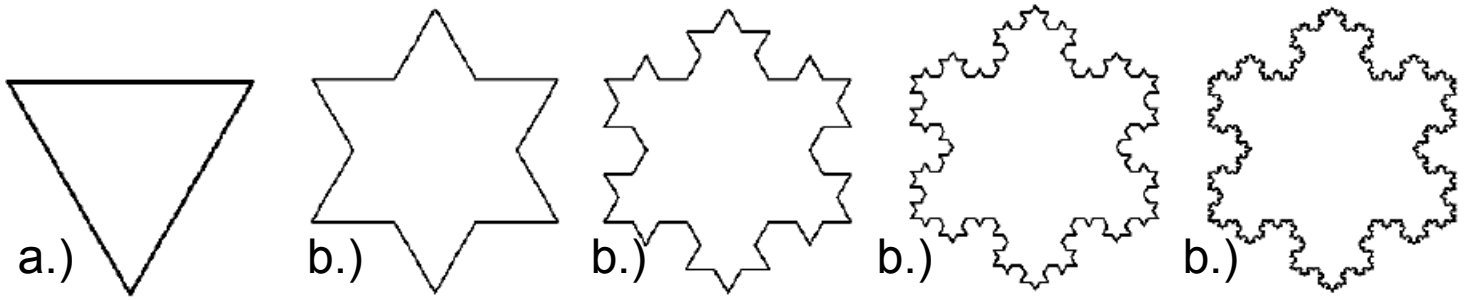
Affine Transformationen

Sierpinski Dreieicke



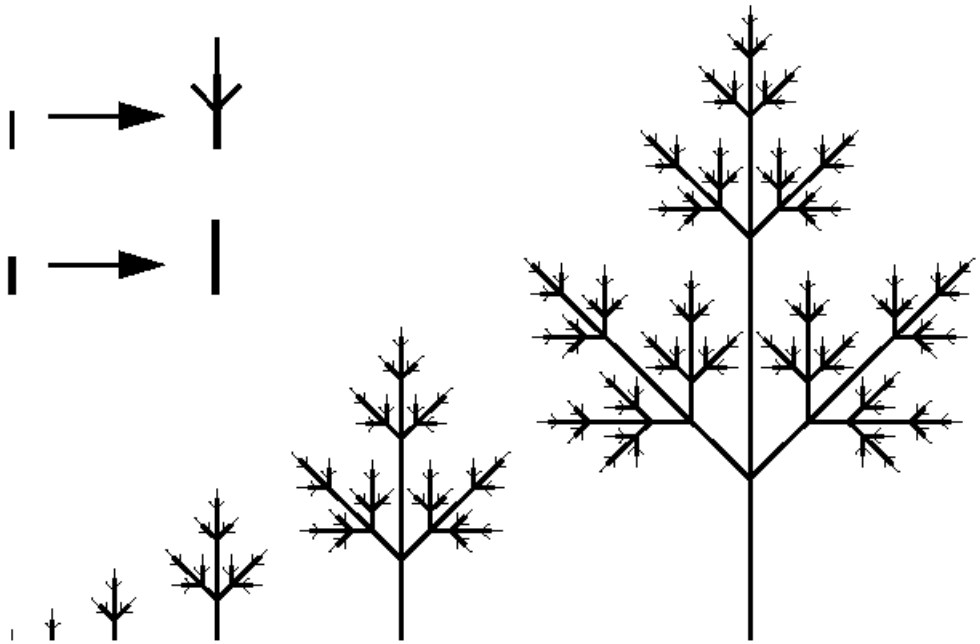
Fraktale

Beispiel: Koch‘s Schneeflocke



a.) S → F(1) -(120) F(1) -(120) F(1)
b.) F(s) → F(s/3) +(60) F(s/3) –(120)F(s/3) +(60) F(s/3)
„-“ entspricht: drehen mit „+“ bedeutet drehen gegen den Uhrzeigersinn

Fraktale: Beispiel L-System Grammatiken



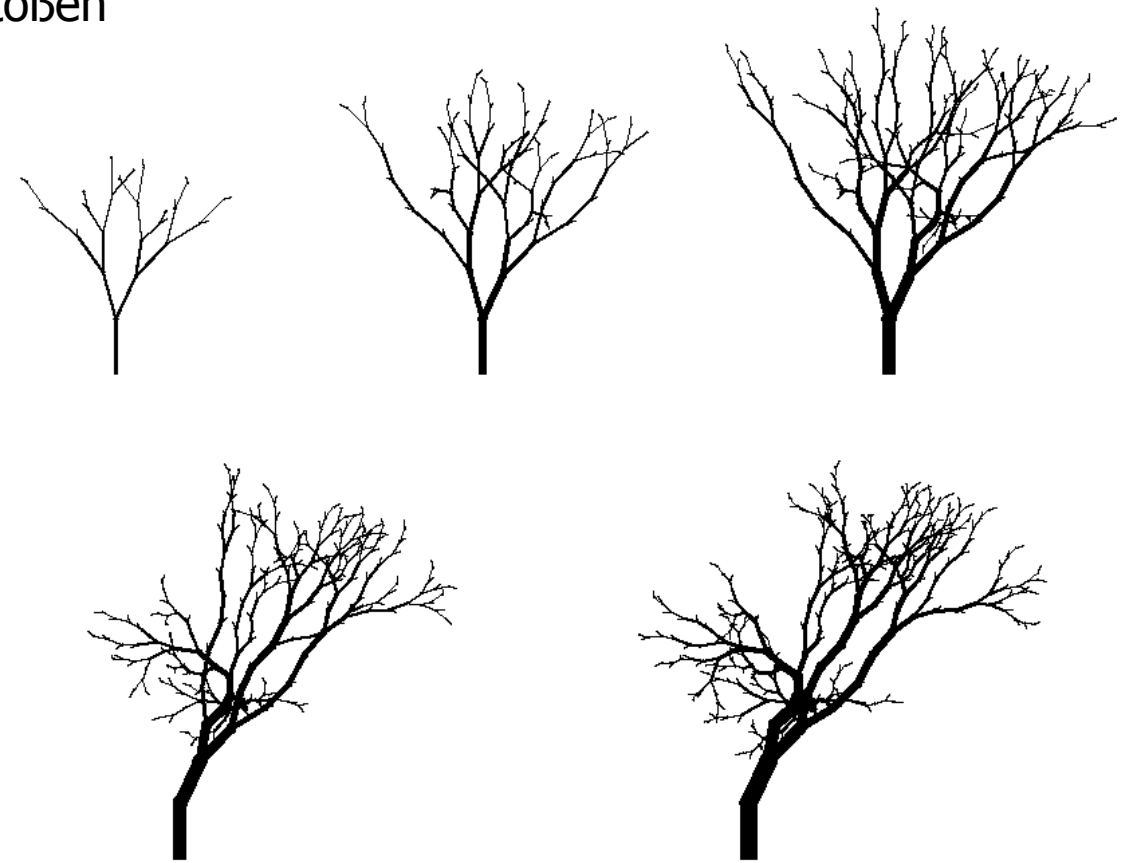
Pflanzen

von P. Prusinkiewicz

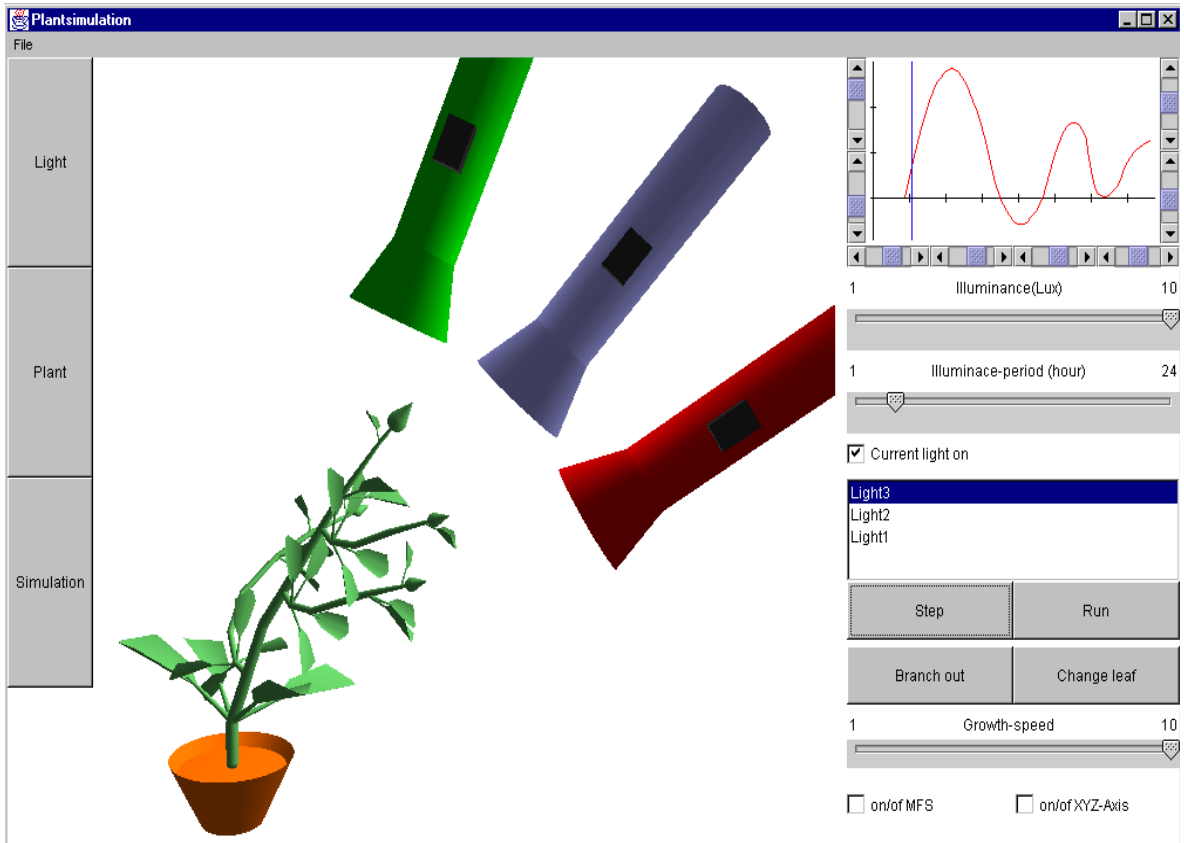
Fraktale: L-Systeme Grammatiken

Umwelteinflüsse:

Zweige, die nur wenig Photosynthese produzieren, werden abgestoßen



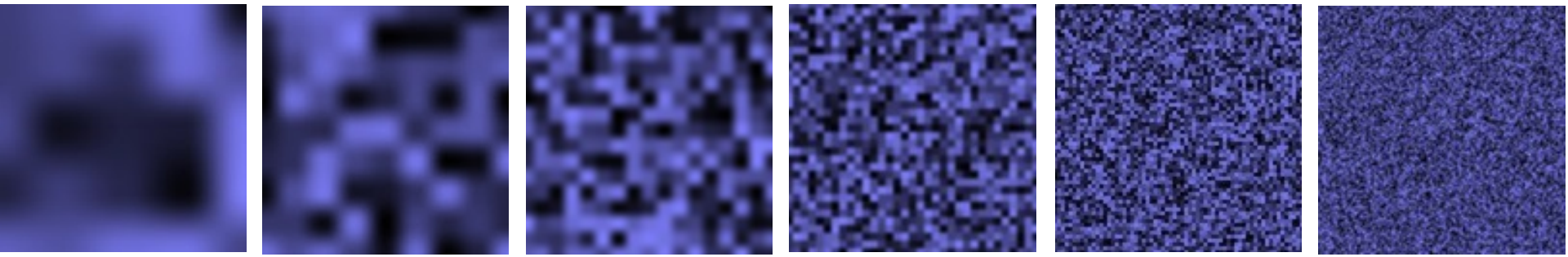
Interaktives Pflanzenwachstum ;-)



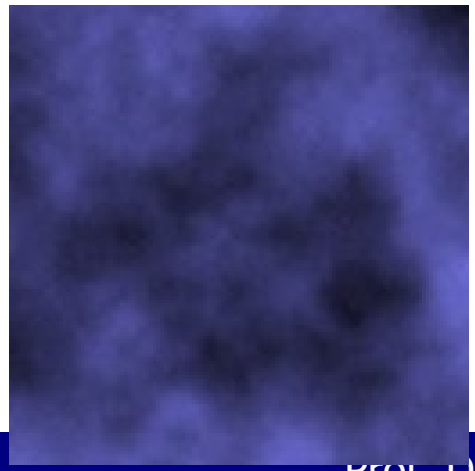
Diplomarbeit von Manuel Lazé (2000)

Fraktale: Beispiel Perlin Noise

Einige „Perlin Noise“ Funktionen in 2D:



Die verschiedenen „Octaven“ addiert:



Fraktale von F. Musgrave

Zusammenfassung der typischen Eigenschaften eines Fraktals

- **Selbstähnlichkeit**
- **Entstehung durch Iteration**
- **Gebrochene (fraktale) Dimension**
- **Komplexität**

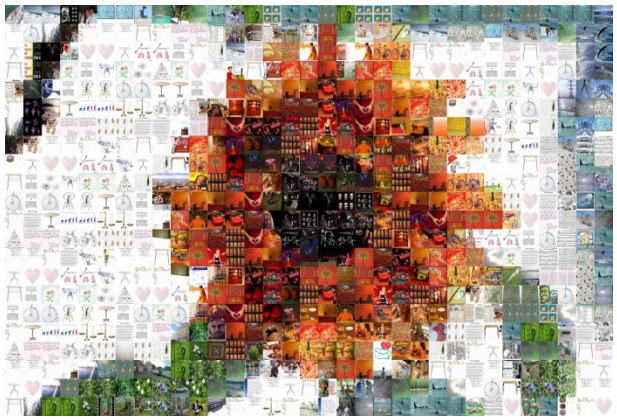
  Fraktale sind hochkomplex (wenn die sie beschreibenden Algorithmen auch sehr simpel sein können); dies äußert sich darin, dass sie mehr Details preisgeben, wenn man sie vergrößert. Die Komplexität geht dabei bis ins Unendliche.
- **Abhängigkeit von den Anfangsbedingungen**

"Mandelbrot Zoom":
https://www.youtube.com/watch?v=0jGaio87u3A

Basis der Fraktalen Kompression: Collage-Theorem

Collage-Theorem (Michael Barnsley):

- Jede beliebige Figur eines Bildes kann man aus verkleinerten Figuren ihrer selbst zusammensetzen.
- Man muss sich nur die entsprechenden **Transformationen** merken.



Sonnenblumen Kollagen entnommen aus:
„Bildmosaike mit Metapixel Steinchen für Steinchen"
von Oliver Frommel
Erschienen in: Erschienen in LinuxUser 06/2005

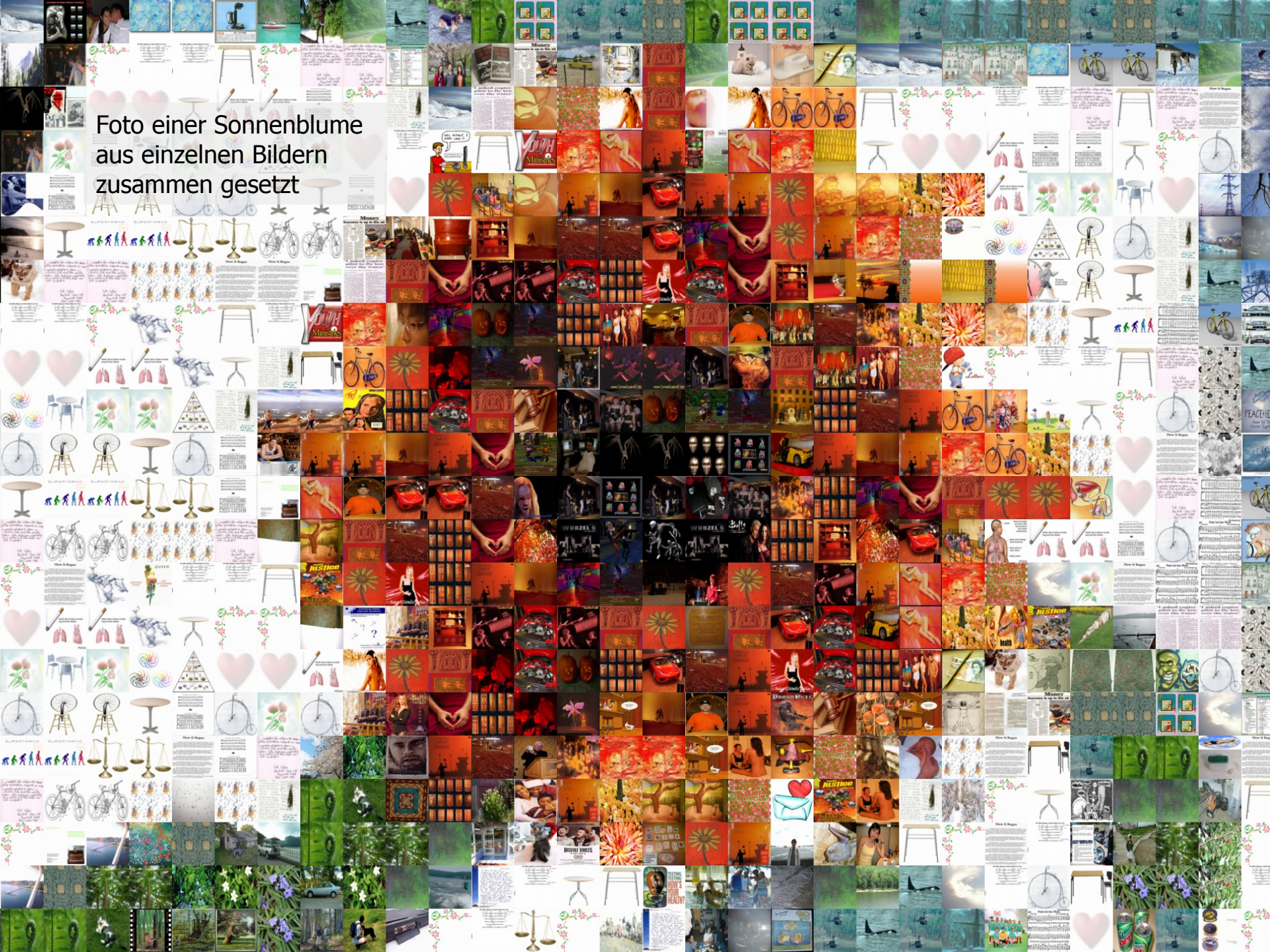
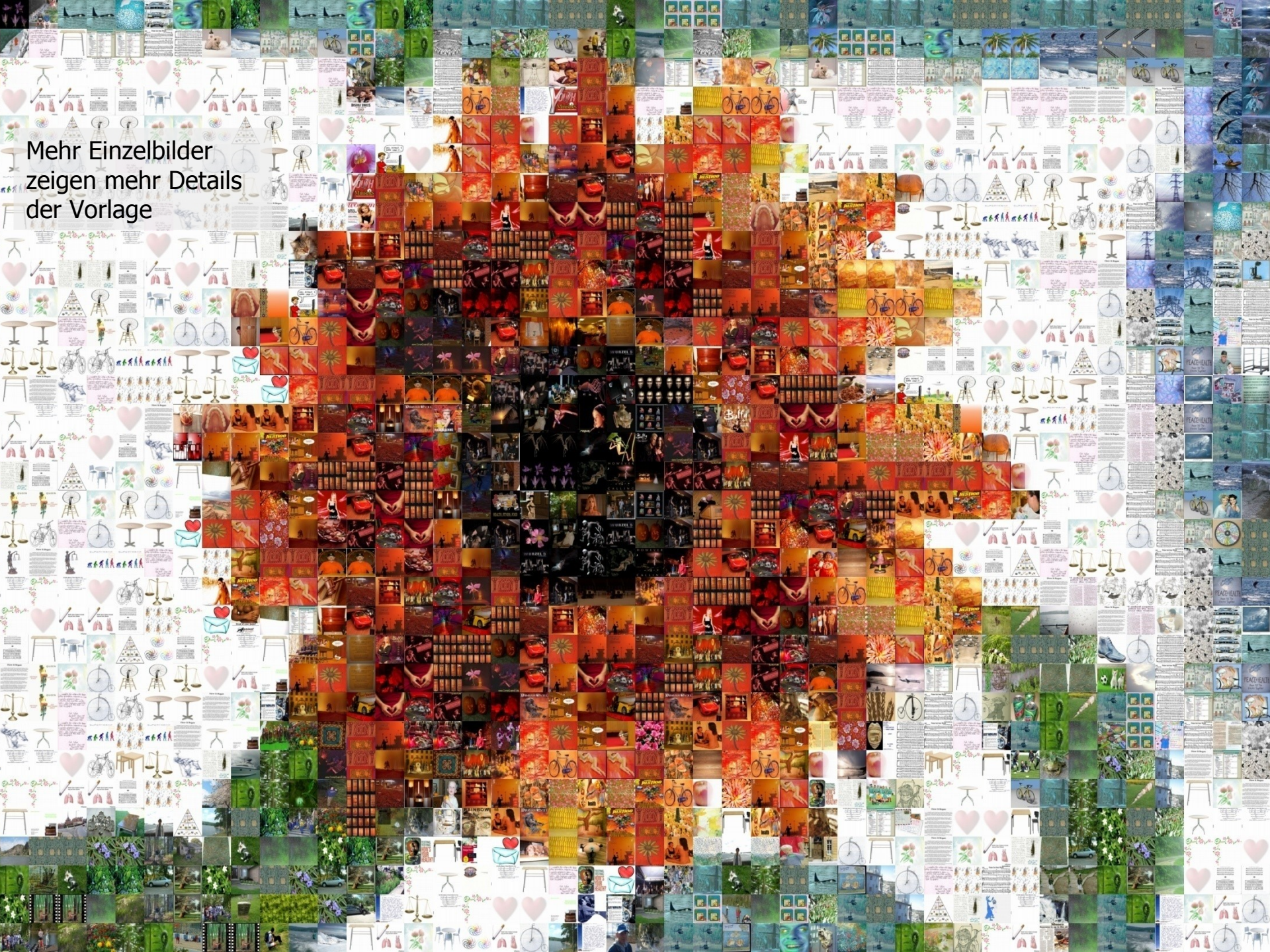


Foto einer Sonnenblume
aus einzelnen Bildern
zusammen gesetzt


Mehr Einzelbilder
zeigen mehr Details
der Vorlage

Fraktale Komprimierung: Codierung

Suche von Selbstähnlichkeiten in Bildern:

- Dazu muss man zunächst per Suchverfahren die korrespondierenden Flächen ermitteln.
- **Sehr Zeitaufwendig**

Affine Transformationen speichern:

- Abbildungsvorschriften entwickeln, wie flächenmäßig **größere Bildbereiche auf** flächenmäßig **kleinere Bildbereiche transformiert** werden: Helligkeit, Rotation, Skalierung, Translation (Verschiebung)

Fraktale Komprimierung: Decodierung

Asynchrones Verfahren:

Decodierung geht sehr schnell