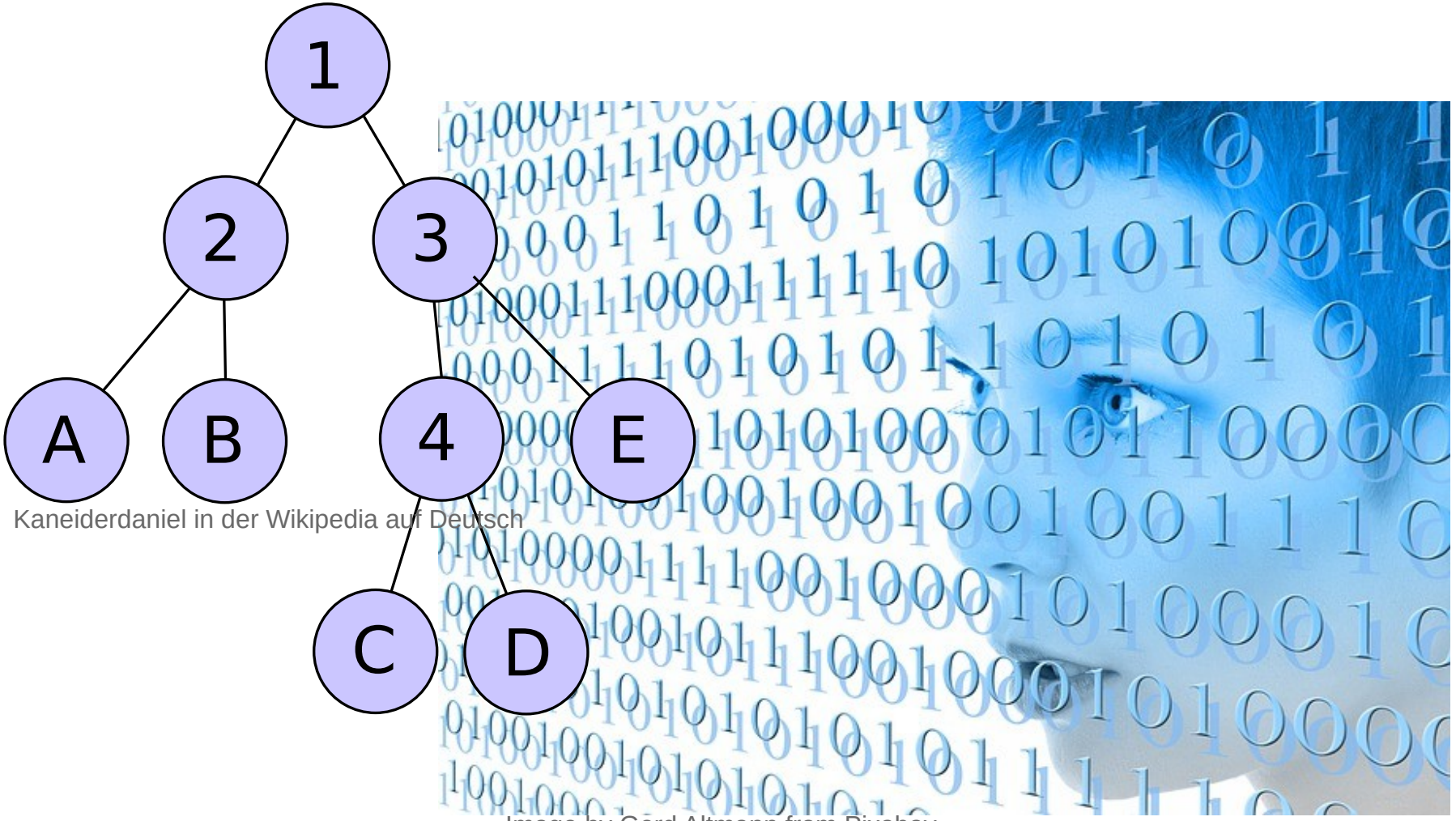
Komprimierung

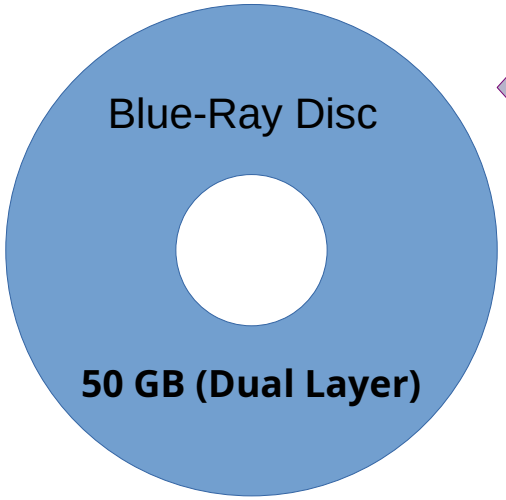


Kaneiderdaniel in der Wikipedia auf Deutsch

Image by Gerd Altmann from Pixabay

Zechnall (September 2021)
Schiebel (Oktober 2023)

Komprimierung – Braucht man das?



Wie viele Sekunden an unkomprimiertem 4K-Filmmaterial in 24bit-Farbtiefe und 60Hz Bildwiederholungsrate passen auf eine Blue-ray Disc?

~ 12,5 Minuten

~ 1 Stunde

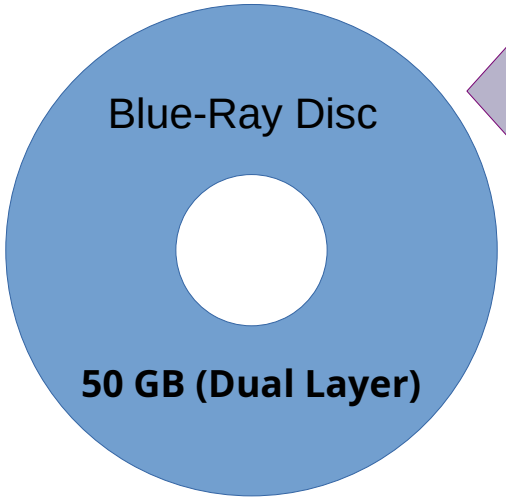
~ 30 Sekunden

~ 2,5 Stunden

4K-Auflösung: 3840 x 2160 Pixel

Zechnall (September 2021)
Schiebel (Oktober 2023)

Komprimierung – Braucht man das?



Wie viele Sekunden an unkomprimiertem 4K-Filmmaterial in 24bit-Farbtiefe und 60Hz Bildwieder-holungsrate passen auf eine Blue-ray Disc?

~ 12,5 Minuten

~ 1 Stunde

~ 30 Sekunden

~ 2,5 Stunden

4K-Auflösung: 3840 * 2160 Pixel = 8.294.400 Pixel

jeweils 24 Bit: 8.294.400 * 3 Byte = 24.883.200 Byte = 24300 KiB ≈ 23,73 MiB je Bild

60 Bilder/Sekunde: 23,73 MiB * 60/s = 1.423,83 MiB/s ≈ **1,39 GiB je Sekunde**

50 GB ≈ 46,56 GiB → 46,56 GiB / 1,39 GiB/s ≈ **33,5 s**

Verlustfreie Kompression – Drei Ansätze

Entropiecodierung	Wörterbuchcodierung	Lauflängencodierung
oft vorkommende Zeichen benötigen weniger Speicherplatz als selten vorkommende Zeichen	ein Wörterbuch enthält mehrfach vorkommende Zeichenfolgen, auf das referenziert wird	mehrfach nacheinander vorkommende Zeichen(folgen) werden abgekürzt
Huffman-Codierung	LZW-Codierung	RLE-Code

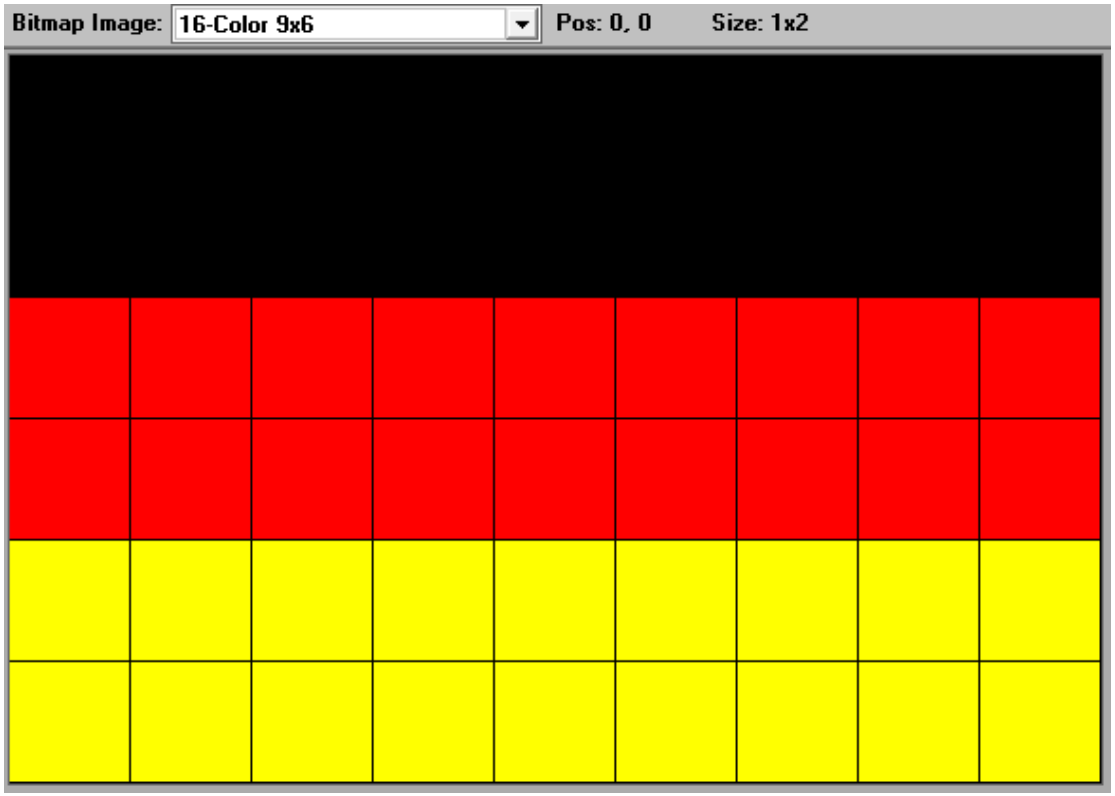
Anforderung:

- Die Codierungen müssen eindeutig entzifferbar sein
- Präfixfreiheit → kein Code eines Zeichens darf mit dem Anfang des Codes eines anderen Zeichens identisch sein. („Fano-Bedingung“)

Zechnall (September 2021)
Schiebel (Oktober 2023)

Komprimierung – verlustfrei?

Wie könnte man dieses Bild verlustfrei komprimieren?



Zechnall (September 2021)
Schiebel (Oktober 2023)

Run Length Encoding (RLE)

Bilder mit großen gleichfarbigen Flächen werden bei Speicherung im bisherigen Format sehr groß. Man kann solche Bilder mithilfe der **Lauflängencodierung (RLE-Verfahren)** komprimieren.

Die Idee ist, **gleiche, aufeinanderfolgende Bits zusammenzufassen** und durch ihre **Anzahl** und ihren Bitwert zu kodieren.

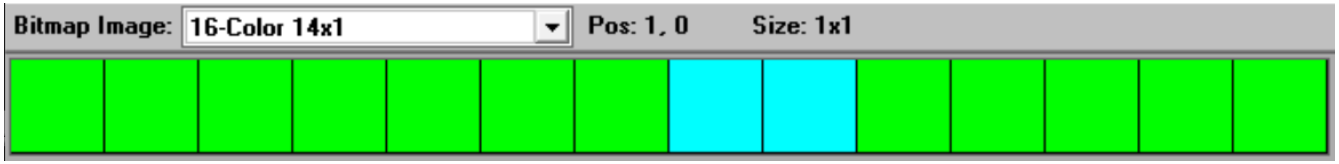
00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00
01 01 01 01 01 01 01 01 01
01 01 01 01 01 01 01 01 01
10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10

18-mal 00, 18-mal 01, 18-mal 10

18 00 18 01 18 10

10010 00 10010 01 10010 10

Run Length Encoding (RLE) – Problem?



111111100 ...

7-mal 1, 2-mal 0

7 1 2 0

111 1 10 0

- **Welches Problem tritt beim Decodieren auf, wenn man die Bitfolge ohne Leerzeichen übermittelt?**
- **Was kann man dagegen tun?**

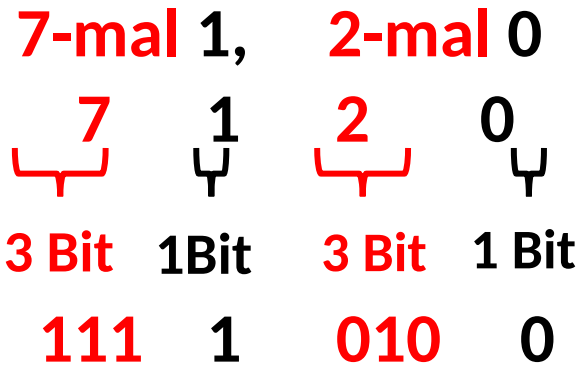
Zechnall (September 2021)
Schiebel (Oktober 2023)

Komprimierung – Lauflängenkomprimietrung (RLE)

Feste Bitlänge

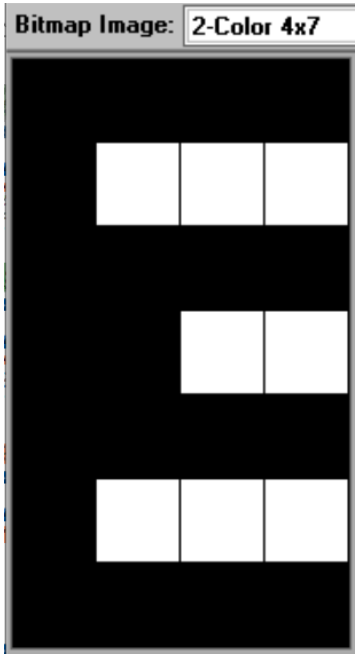
Damit das Verfahren eindeutig ist, legt man die Anzahl der Bit fest, sowohl für die Codierung der Anzahl als auch für die Codierung des Farbwerts.

111111100...



Meist wird in einem ersten Durchlauf die **maximale Anzahl zusammenfassbarer Bits (Run)** bestimmt und demenentsprechend die feste Bitlänge n gewählt.

Unkomprimiert – bekannte Codierung



Bekannte Codierung (Header: je 4 Bit für Breite und Höhe):

0100 0111 11111 000 111111 00 11111 000 1111

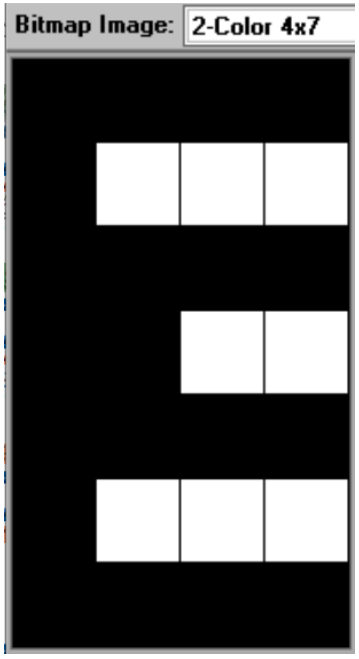
→ **36 Bit**

„6x1“

Längster Run 6 → 3 Bit für die Anzahl

Zechnall (September 2021)
Schiebel (Oktober 2023)

RLE – Zwei Ansätze der Komprimierung



„Zahl+Farbe“

Ansatz 1: (Anzahl gleicher Farbpixel & Codierung der Farbe)

4 7 5 1 3 0 6 1 2 0 5 1 3 0 4 1

Ansatz 1 in Bitschreibweise (**3 Bit** pro Anzahl):

0100 0111 101 1 011 0 110 1 010 0 101 1 011 0 100 1

→ **36 Bit** (keine Komprimierung!)

„Farbwechselstrecken“

Ansatz 2: (Nur Anzahl – begonnen wird mit Weiß)

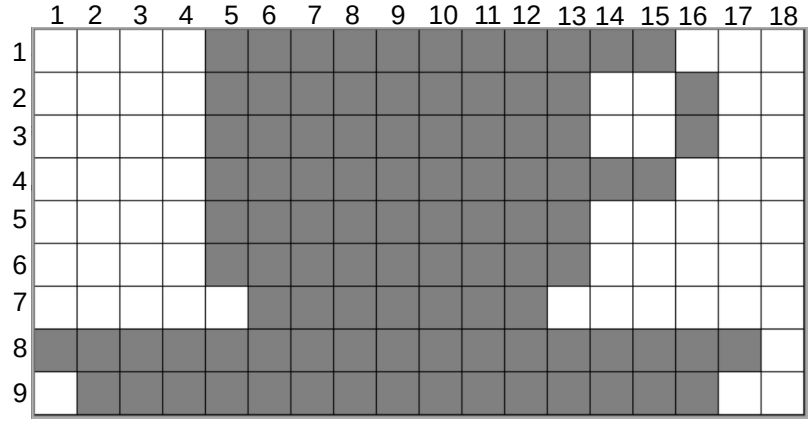
6 5 0 5 3 6 2 5 3 4

Ansatz 2 in Bitschreibweise (**3 Bit** pro Anzahl):

0100 0111 000 101 011 110 010 101 011 100

→ **32 Bit** (ca. 89% der Originalgröße!)

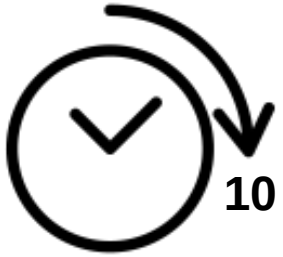
RLE – Weiteres Beispiel



Anzahl Zeilen: 9
Anzahl Spalten: 18
Längster Run: 17

Bild: https://classic.csunplugged.org/image-representation/

Bekannte Codierung:
16 Bit für den Header + 9*18 = 162 Bit für die Nutzlast
→ **178 Bit**

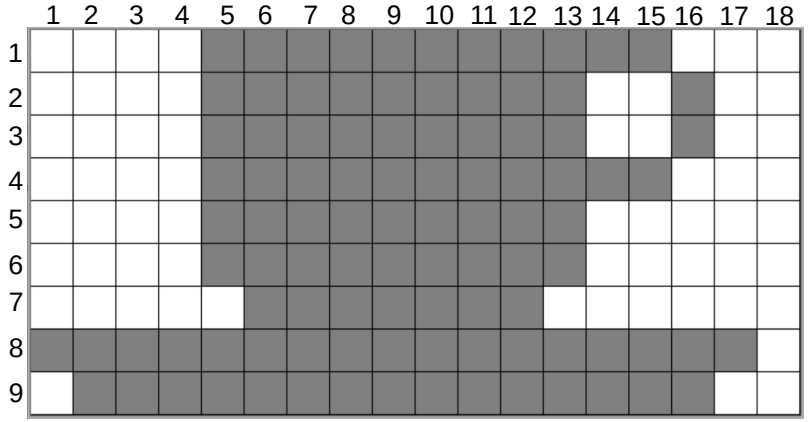


10 Minuten

Arbeitsphase

- **Komprimiere das Bitmap mittel RLE – wie viel Prozent der Originalbildgröße erreichst du?**

RLE – Weiteres Beispiel



Anzahl Zeilen: 9
Anzahl Spalten: 18
Längster Run: 17

Bild: https://classic.csunplugged.org/image-representation/

Bekannte Codierung:
16 Bit für den Header + 9*18 = 162 Bit für die Nutzlast
→ **178 Bit**

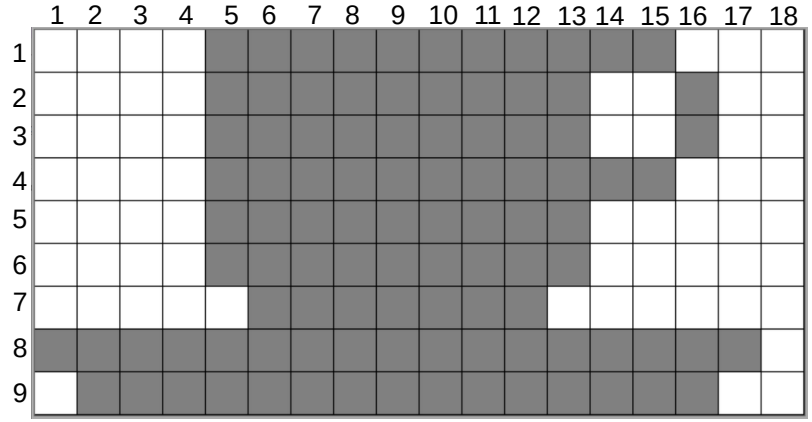
„Farbwechselstrecken“

9 18 4 11 7 9 2 1 6 9 2 1 6 11 7 9 9 9 10 7 6 17 2 15 2
Ansatz 2 in Bitschreibweise (**5 Bit** pro Anzahl, längster Run 17):
0000 1001 0001 0010 00100 01011 00111 01001 00010 00001 00110 01001…
→ **131 Bit** (das entspricht ca. 73,6% der Größe vom Originalbild!)

Zechnall (September 2021)
Schiebel (Oktober 2023)

Komprimierung – Lauflängenkomprimietrung (RLE)

RLE – Ein weiterer Ansatz



Bekannte Codierung: 178 Bit

Anzahl Zeilen: 9
Anzahl Spalten: 18
Längster Run: 17

Bild: https://classic.csunplugged.org/image-representation/

„Farbwechselstrecken“ **131 Bit** (das entspricht ca. 73,6 % der Größe vom Originalbild!)

„Farbwechsel“ **(mit nur 4 Bit pro Anzahl)**

9 18 4 11 7 9 2 1 6 9 2 1 6 11 7 9 9 9 10 7 6 15 0 2 2 15 2
In Bitschreibweise (**4 Bit** pro Anzahl, obwohl längster Run 5 Bit benötigt):
0000 1001 0001 0010 0100 1011 0111 1001 … 1111 0000 0010 … 1111 0010

→ **116 Bit** (das entspricht ca. 65,2 % der Größe vom Originalbild!)

Zechnall (September 2021)
Schiebel (Oktober 2023)

Komprimierung – Lauflängenkomprimietrung (RLE)

Nicht immer komprimiert ein Komprimierungsverfahren

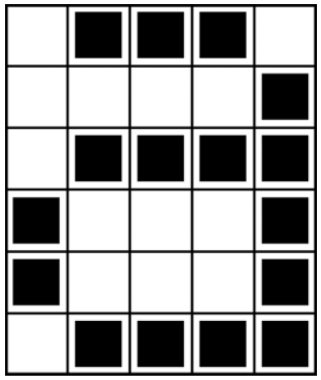


Bild:
https://classic.csunplugged.org/image-representation/

Bekannte Codierung:
0110 0101 0 111 00000 1 0 11111 000 11 000 1 0 1111
→ **38 Bit**

„Zahl+Farbe“

6 5 1 0 3 1 5 0 1 1 1 0 5 1 3 0 2 1 3 0 1 1 1 0 4 1
Ansatz 1 in Bitschreibweise (**3 Bit** pro Anzahl):
0110 0101 001 0 011 1 101 0 001 1 001 0 101 1 011 0 010 1 011 0 001 1 001 0 100 1
→ **56 Bit** (147,4% der Originalbildgröße!)

„Farbwechselstrecken“

6 5 1 3 5 1 1 5 3 2 3 1 1 4
Ansatz 2 in Bitschreibweise (**3 Bit** pro Anzahl):
0110 0101 001 011 101 001 001 101 011 010 011 001 001 100
→ **44 Bit** (115,8% der Originalbildgröße!)

Zechnall (September 2021)
Schiebel (Oktober 2023)