

# Bild und Ton Kompression

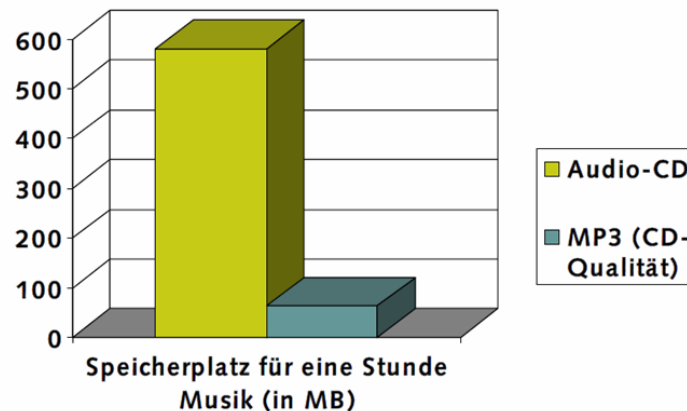
© J. Rau 2025

# Kompression

**... bedeutet die Verringerung des Datenumfangs einer Mediendatei und damit des benötigten Speicherplatzes.**

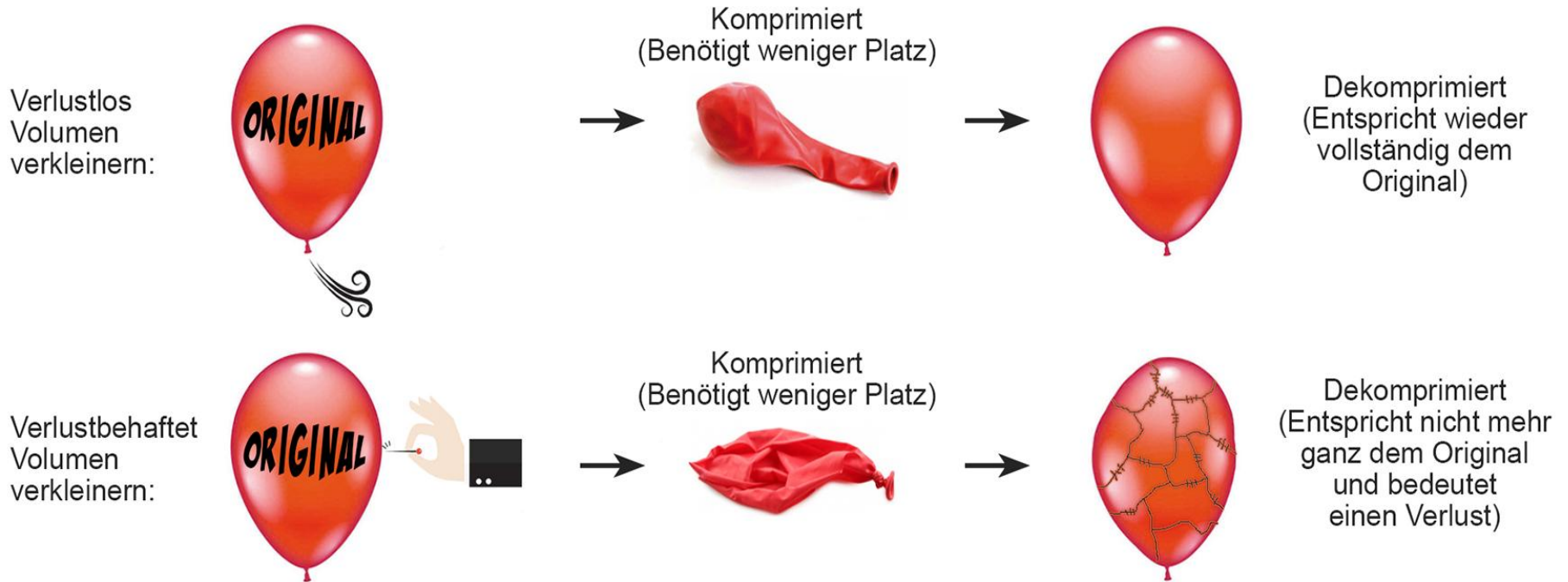
Man verwendet Verfahren, welche die Informationen bei geringerem Speicherbedarf so codieren, dass alle komplett Informationen rekonstruiert werden können oder mit geringen Verlusten, die im Normalfall durch die menschlichen Sinne nicht .

Beispiel: Komprimierung von Tönen als MP3 Format



# Kompression

Es gibt verlustfreie und verlustbehaftete Kompression.



**Kompressionsfaktor = 1 - Komprimierte Größe / Originalgröße**

# Kompression und Grafiktypen

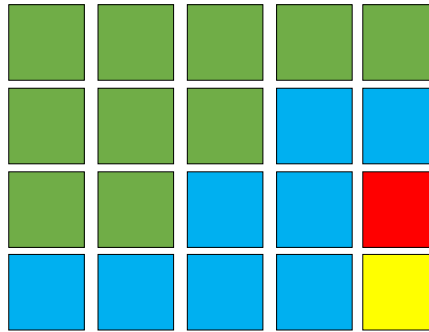
Format	Anzahl Farben	Komprimierung	Anwendung
BMP	2, 6, 256 oder 16 Mio.	keine	Windows Bilder
TIFF	Maximal 16 Mio	gering, verlustfrei	gescannte Bilder
GIF	Maximal 256	gering, verlustfrei	Text als Grafik, Strichzeichnungen, WWW
PNG	Maximal 16 Mio	hoch, verlustfrei	alle Bilder
JPEG	Maximal 16 Mio	hoch, verlustbehaftet	Fotos und Bilder mit weichen Farbverläufen

# Verlustfreie Komprimierung- RLE

Eine Möglichkeit zur verlustfreien Bildkomprimierung ist die **Laufängen Codierung** (**RLE**-Codierung [*run length encoding*])

- Jede Sequenz gleicher Daten wird durch ein Symbol und dessen Anzahl ersetzt.
- Der Speicherplatz nach der Komprimierung hängt davon ab, wie oft sich die Daten verändern.
- Besonders effizient ist diese Komprimierung bei vielen langen Wiederholungssequenzen.

# Verlustfreie Bildkomprimierung - RLE



Speicherplatz Originalbild:  $4_{\text{[Pixel]}} \cdot 5_{\text{[Pixel]}} \cdot 3\text{B}_{\text{[für Farben]}} = 60\text{B}$



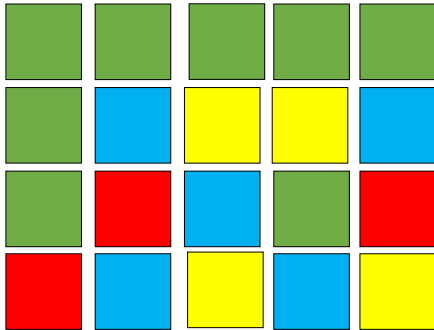
Speicherplatz nach RLE:  $3 \cdot 7\text{ B}_{\text{[für die Farbe]}} + 7\text{ B}_{\text{[für den Zahlenwert]}} = 28\text{ B}$

Kompressionsfaktor  $\rightarrow 0,533 \rightarrow 53,3\%$

RLE ist sinnvoll für Bilder mit großen gleichfarbigen Flächen.

# Verlustfreie Bildkomprimierung – bitweise Codierung

Es geht besser als mit RLE , nämlich mit der bitweisen Lauflängen-Kompression. Nützlich ist dies z.B. bei eingeschränkter Farbenzahl.



z.B. 4 Farben → es reichen 2 **bit**.

Grün: 00, Gelb: 01, Rot: 10, Blau:11

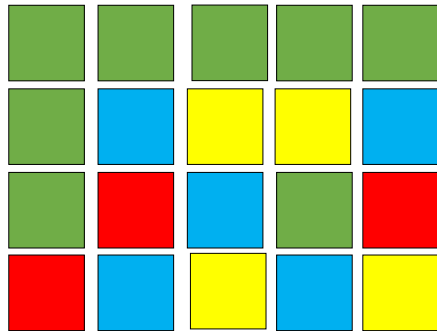
Originalbild: 3 B [für die Farbe] x 5 x 4 = 60B

kompr. Bild: 20 x 2 bit = **40 bit = 5 B**

Kompressionsfaktor → 0,92 → 92%

## Verlustfreie Bildkomprimierung – bitweise Codierung mit Häufigkeit

Es geht noch besser ...



z.B. 4 Farben → es reichen 2 **bit**.

Wir betrachten die Häufigkeit der Farben:

Grün 8x, Blau 5x, Gelb 4x und Rot 3x

Wir verkürzen den Code für die häufigeren

Farben: Grün: **0**, Gelb: 01, Rot: 10, Blau: 1

Originalbild: 3 B [für die Farbe] x 5 x 4 = 60B

kompr. Bild: 13 x 1 bit + 7 x 2 bit = **27 bit = 4B**

Kompressionsfaktor → 0,934 → 93,4%



# Verlustfreie Komprimierung- Huffman

**Die Huffman Codierung** ist eine **VLE-Codierung** [variable length encoding].

Den Symbolen werden Codes unterschiedlicher Länge so zugeordnet, dass

- (1) kein Codewort das wie das "Anfangsstück" eines anderen Codewortes aussieht. (--> eindeutige Zuordnung)
- (2) häufiger verwendete Zeichen erhalten einen kürzeren Code als seltenere. (--> weniger Speicherplatz)

# Verlustfreie Komprimierung- Huffman

Das bedeutet, dass die Information zuerst nach einem bestimmten Prinzip analysiert werden muss.

- Die Buchstabenhäufigkeit ermitteln und geordnet nach Häufigkeit aufschreiben
- einen kompletten binären Baum aufzeichnen
- den Code ablesen und Codetabelle erstellen
- die Information binär-huffman-codiert aufschreiben

[Erklärung im Video](#)

# Verlustfreie Komprimierung- Huffman

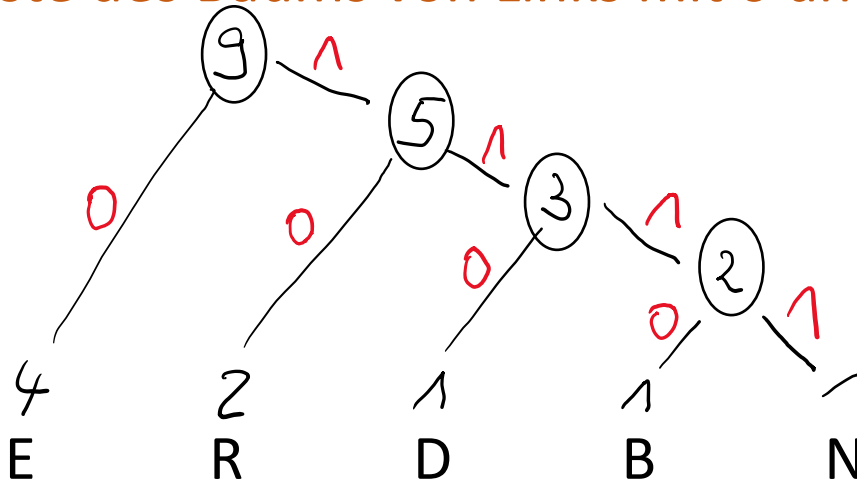
Beispiel: ERDBEEREN

Speicherplatz in ASCII  $\rightarrow 9 \cdot 8 \text{ bit} = 72 \text{ bit} = 9 \text{ B}$

- Buchstabenhäufigkeit ermitteln, geordnet nach Häufigkeit aufschreiben

E	R	D	B	N
4	2	1	1	1

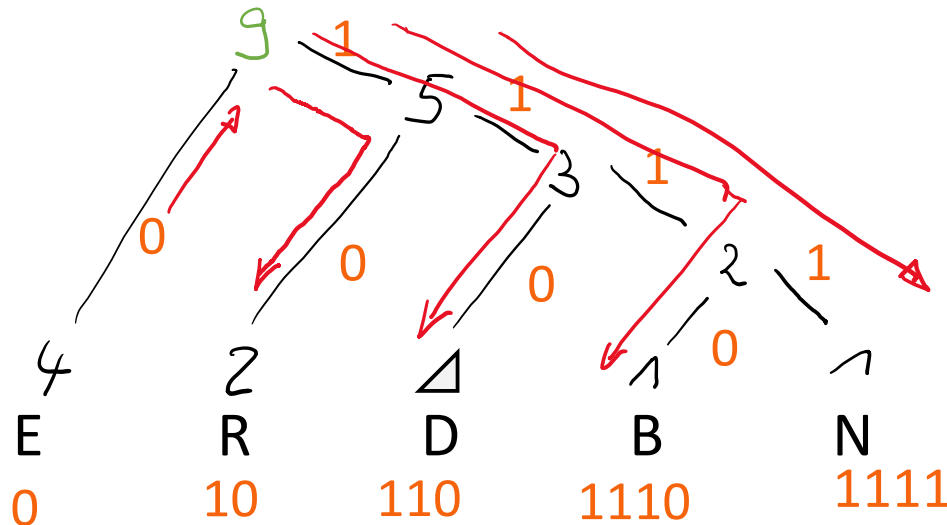
- Kompletten **binären Baum** aufzeichnen  $\rightarrow$  rechts beginnen
- Äste des Baums von Links mit 0 und Rechts mit 1 beschriften



# Verlustfreie Komprimierung- Huffman

Beispiel: ERDBEEREN

- Code ablesen und Codetabelle erstellen



$\bar{E} = 0$   
 $R = 10$   
 $D = 110$   
 $B = 1110$   
 $N = 1111$

- Information binär-Huffman-codiert aufschreiben

0101101110001001111

Der Code ist rückwärts wieder eindeutig zuzuordnen.

Speicherplatz nach Komprimierung  $\rightarrow$  19 bit

Kompressionsfaktor  $\rightarrow 1 - 19/72 = 0,74 \rightarrow 74\%$

# Aufgabe - Huffman

Erstelle den Huffman-Code für: KOBOLDGOLD.

Vergleiche die Codeeffizienz gegenüber der ASCII-Codierung.  
Gib den Kompressionsfaktor an.

Ist die Huffman-Codierung für eine Zeichenkette immer eindeutig?

**Welche Arten der Datenkompression lassen sich unterscheiden?  
Was sind die grundsätzlichen Eigenschaften der beiden Arten von  
Datenkompression?**

**Nennen Sie mindestens drei Anwendungen oder Techniken,  
die mit verlustbehafteten Kompressionsverfahren möglich  
wurden.**

**Worauf beruht die Verwendung der Huffman-Codierung zur Datenkompression?**

**Angenommen, die Zeichenfolge 31115522222 wird als 13312552 dargestellt. Um welche Form der Codierung handelt es sich und wie funktioniert sie?**

**Worauf beruhen verlustbehaftete Kompressionsverfahren?**