

# **GRUNDLAGEN DER DIGITALEN BILDVERARBEITUNG**

Testvorlesung mit DynamicNodes

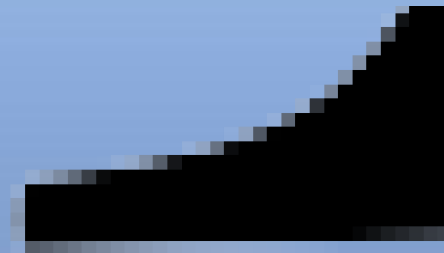
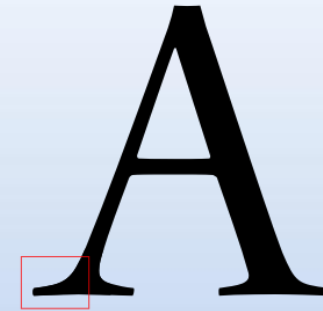
- Arten von Bildern
- Aufbau von Bitmaps
- Mehrkanal-Bitmaps
- Arten von Operationen
- Modifikation der Grauwerte
- Operationen im Ortsbereich

# **Arten von Bildern**

Worum dreht sich digitale Bildverarbeitung?

## Arten von Bildern

- Grafiken
  - Punkte und Vektoren
  - Füll- und Umrissfarben
  - Beliebig skalierbar
  - Datenmenge wächst mit Details
- Bitmaps
  - Raster mit Farbwerten
  - Skalierung macht Pixel sichtbar
  - Datenmenge wächst mit Auflösung bzw. Größe

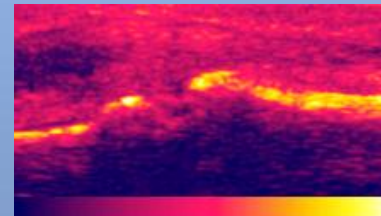
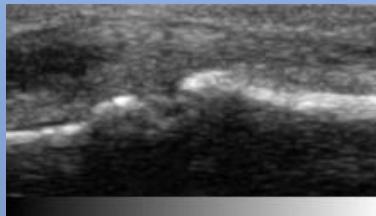
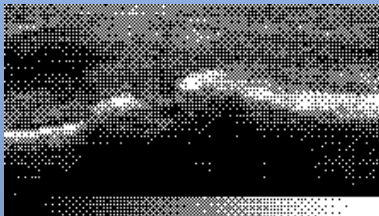


# **Aufbau von Bitmaps**

Was sind Pixel?

## Aufbau von Bitmaps

- Eine „Bitmap“ ist eine Matrix
- Jede Zelle entspricht einem Pixel
- Werte können unterschiedlich interpretiert werden
  - Schwarz/Weiß (1Bit = 2 Stufen)
  - Graustufen (8, 16Bit = 256, 65536 Stufen)
  - Paletten-Indizes (1, 2, 4, 6, 8Bit = 2, 4, 16, 64, 256 Farben)



# Aufbau von Bitmaps

DynamicNodes verarbeitet hauptsächlich 8-Bit-Bitmaps mit einem Wertebereich von 0 - 255

## Praxis

## Aufbau von Bitmaps

- Benötigte Knoten
  - Bildverarbeitung: Bild laden
  - Bildverarbeitung: ARGB-Splitter
  - Mathematik: Matrix-Display
- Aufgabe
  - *Das geladene Bild mit Hilfe des ARGB-Splitters in einzelne Farbkanäle aufspalten und den Rot-Kanal in das Matrix-Display leiten*



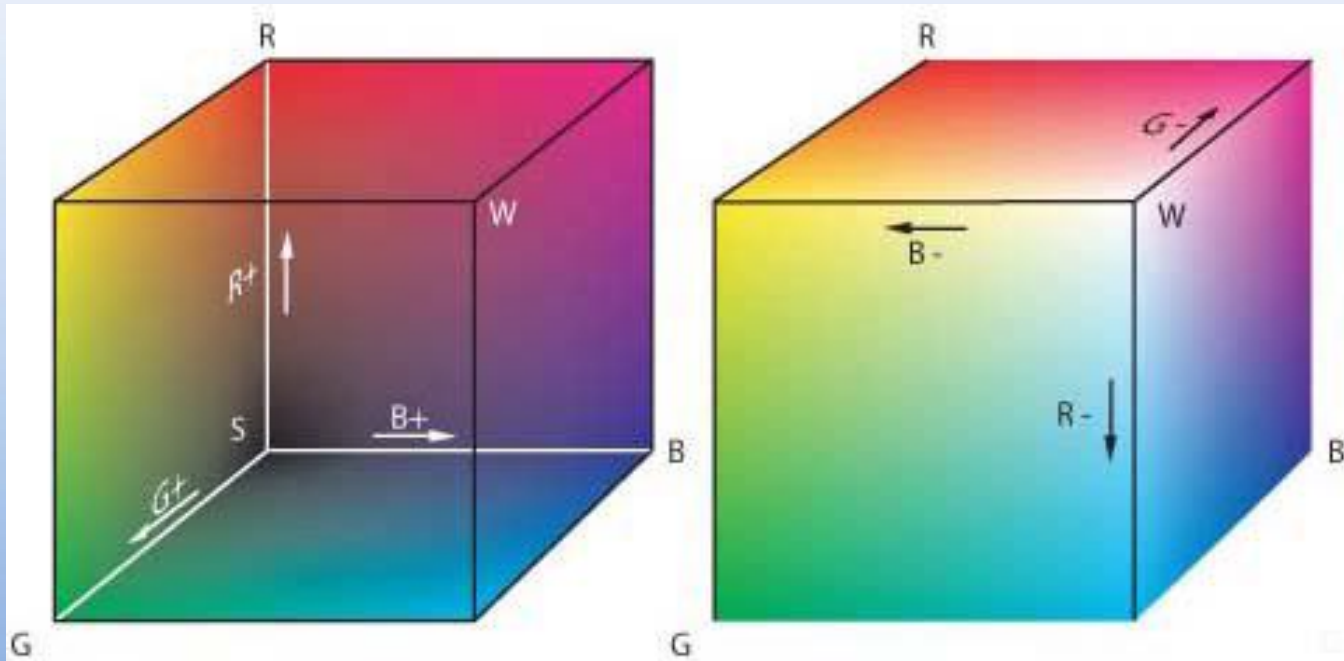
# Mehrkanal-Bitmaps

Was ist Farbe?

# Mehrkanal-Bitmaps

- Zur Beschreibung einer Farbe wird ein Farbraum benötigt
- Farbe ist Licht mit einem Frequenzgemisch
- Das Auge unterscheidet die Grundfarben Rot, Grün und Blau
  - RGB-Farbraum
- Das Gehirn unterscheidet Farbe, Sättigung und Helligkeit
  - HSB-Farbraum
- Je Kanal ein Graustufen-Bitmap  
(24, 48 Bit =  $3 \times 256$ ,  $3 \times 65536$  Stufen)

## RGB-Farbraum



R = Red = Rot

G = Green = Grün

B = Blue = Blau

## RGB-Farbraum

Drei Bitmaps bilden ein RGB-Farrbild

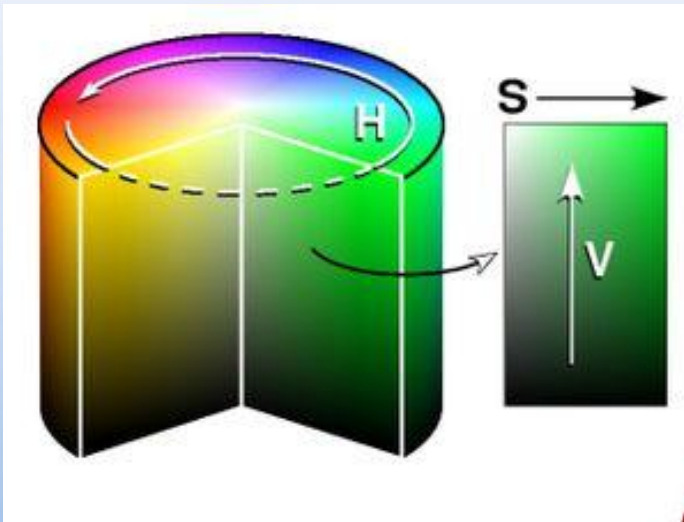
34	0	27	20	50	0	190	120
6	18	200	230	0	18	200	230
128	210	255	160	20	240	255	0

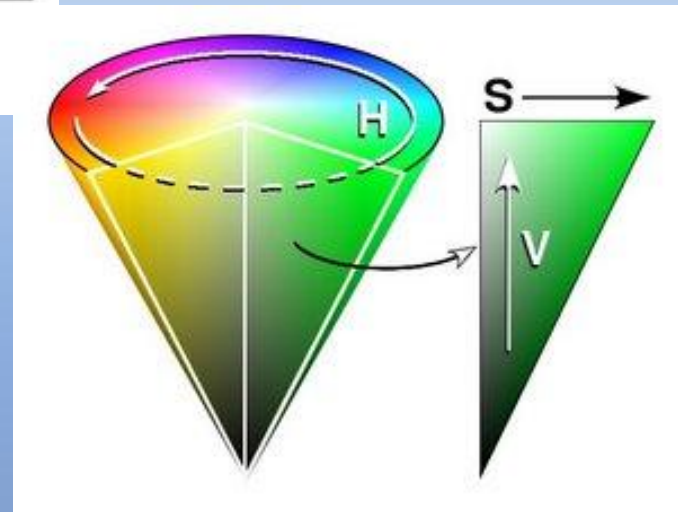
42	0	200	80
220	18	200	230
10	40	255	200



## HSB-Farbraum



H = Hue = Farbe  
S = Saturation = Sättigung  
B = Brightness = Helligkeit  
V = Value = Wert  
B = V



# Mehrkanal-Bitmaps

- Transparenz = Opazität = Alpha
- ist zusätzlicher Kanal (32Bit = (3+1)x256 Stufen)
  - Niedriger Wert = durchsichtig
  - Hoher Wert = undurchsichtig
- RGB mit Transparenz ist ARGB

# Mehrkanal-Bitmaps

DynamicNodes kann RGB und ARGB-Bilder verarbeiten und beherrscht die Transformationen RGB->HSB, HSB->RGB

## Praxis

## Mehrkanal-Bitmaps

- Benötigte Knoten
  - Bild laden
  - ARGB-Splitter
  - Bildbetrachter 2x
  - RGB->HSB
- Aufgabe
  - *Leiten sie das RGB-Bild in den ARGB-Splitter und vergleichen Sie die Kanäle mit Hilfe des Bildbetrachters*
  - *Leiten Sie das RGB-Bild in den RGB-HSB-Konverter und vergleichen Sie die HSB-Kanäle mit den RGB-Kanälen*



## Arten von Operationen

Wie kann ein Bild verarbeitet werden?

## Arten von Operationen

- Modifikation der Grauwerte
  - Addition = Verschiebung
  - Multiplikation = Skalierung
  - Logische Verknüpfung, Schwellwert, ...
- Operationen im Ortsbereich
  - Lineare Filter
    - Convolution = Faltung (z.B. Schärfe, Unschärfe)
  - Nichtlineare Filter
    - Maximum, Minimum, Median, Zufall

## Arten von Operationen

- Transformationen (werden nicht weiter behandelt)
  - Farbraumtransformation
  - Affine Transformationen
    - Skalierung
    - Rotation
    - Scherung

# Modifikation der Grauwerte

Wie funktioniert ein Helligkeits- oder Kontrastregler?

# Modifikation der Grauwerte

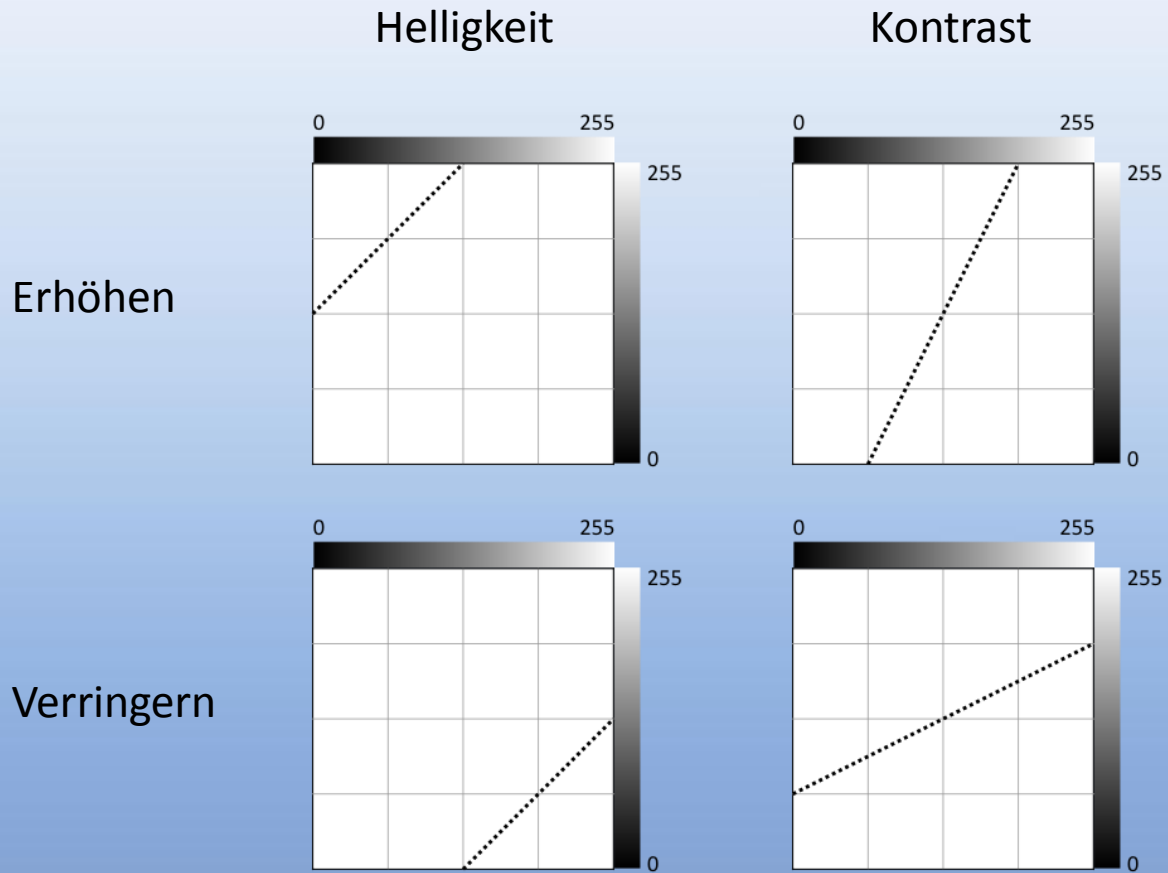
- Mathematische Funktion für jeden Pixel
  - $x' = f(x)$
  - Helligkeit:  $f(x) = x + c$
  - Kontrast:  $f(x) = x * c$
- Optimierte Berechnung mit Lookup-Table
  - Formel wird auf alle 256 möglichen Werte angewendet
  - Ergebnisse werden in Tabelle gespeichert
  - Bei Anwendung auf Bild keine Berechnung mehr

## Lookup-Table

- Lookup-Table lässt sich grafisch darstellen



## Lookup-Table



## Praxis

## Mehrkanal-Bitmaps

- Benötigte Knoten
  - Bild laden, Bildbetrachter 2x
  - Lookup
  - S-Funktion, Freie Funktion
- Aufgabe
  - *Leiten Sie ein Bild durch den Lookup-Knoten in einen Bildbetrachter*
  - *Probieren Sie die S-Funktion und die Freie Funktion als Tabelle für den Lookup-Knoten aus*

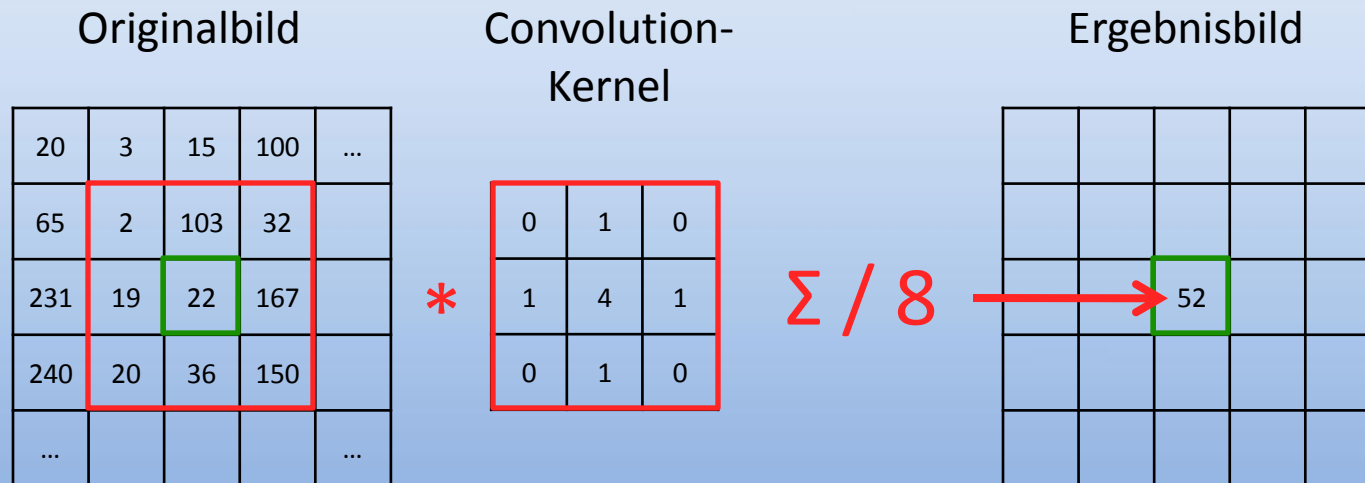


## Operationen im Ortsbereich

Wie funktioniert ein Unschärfe-Filter?

## Lineare Operationen

### Convolution als Bildverarbeitungsoperator



Für jeden Pixel wird mit einem Convolution-Kernel eine gewichtete Summe seiner Umgebung gebildet und durch die Summe des Kerns geteilt

## Lineare Operationen

Unschärfe

1	1	1
1	1	1
1	1	1

Unschärfe  
horizontal

0	0	0
1	1	1
0	0	0

Unschärfe  
vertikal

0	1	0
0	1	0
0	1	0

Kanten-  
detektion  
(Laplace)

0	-1	0
-1	4	-1
0	-1	0

Kanten-  
detektion  
horizontal

0	0	0
0	1	-1
0	0	0

Kanten-  
detektion  
vertikal

0	0	0
0	1	0
0	-1	0

Schärfen

-1	-2	-1
-2	30	-2
-1	-2	-1

Sobel  
horizontal

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

Sobel  
vertikal

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

## Praxis

## Lineare Operationen

- Benötigte Knoten
  - Bild laden, Bildbetrachter 2x, Dynamik
  - Convolution
  - Mathematik: 3x3-Matrix
- Aufgabe
  - *Leiten Sie ein Bild durch den Convolution-Knoten in einen Bildbetrachter*
  - *Probieren Sie die verschiedenen Kernel aus*
  - *Verstärken Sie das Ergebnis mit dem Dynamik-Knoten*  
(Auto-Min=False, Auto-Max=False, Input-Min=70, Input-Max=184)

## Nichtlineare Operationen

- Rangordnungs-Filter
  - Minimum, Maximum, Median, Zufall
- Sortieren die Werte der Umgebung eines Pixels
- Und wählen einen Wert aus

20	3	15	100	...
65	2	103	32	
231	19	22	167	
240	20	36	150	
...				...

(2, 19, 20, 22, 32, 36, 103, 150, 167)

Minimum = 2

Median = 32

Maximum = 167

## Praxis

## Nichtlineare Operationen

- Benötigte Knoten
  - Bild laden, ARGB-Splitter, Bildbetrachter 2x
  - Rangordnung (funktioniert nur für Graustufen)
- Aufgabe
  - *Leiten Sie ein Bild durch den Rangordnungs-Knoten in einen Bildbetrachter*
  - *Probieren Sie verschiedene Methoden aus*
  - *Verwenden Sie den Rangordnungs-Knoten 3-fach um ein RGB-Bild zu verarbeiten*

**Vielen Dank!**