

Computer Vision

Von Pixeln zur Wahrnehmung - Einführung, Anwendungsgebiete
& Digitale Bildrepräsentation

Alexandra Posekany

December 2025

Was ist Computer Vision?

Ein interdisziplinäres Feld, das Computern beibringt, visuelle Daten (Bilder/Videos) zu “sehen”, zu verarbeiten und zu verstehen.

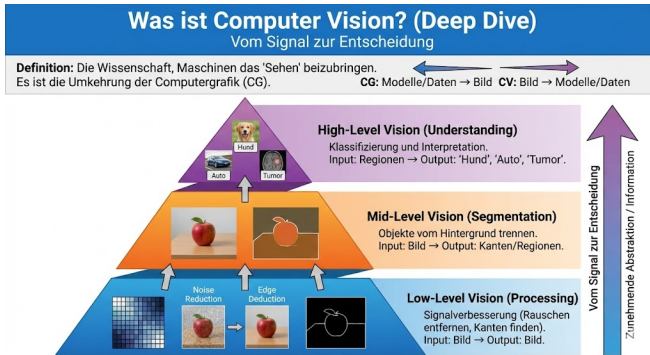
Kernfrage: Wie **extrahieren wir semantische Informationen** (Bedeutung) aus rohen Daten (Pixeln)?

Abgrenzung:

- ▶ **Image Processing**: Bild rein → Bild raus (z.B. Filter, Helligkeitskorrektur).
- ▶ **Computer Vision**: Bild rein → Information/Entscheidung raus (z.B. “Da ist ein Fußgänger”).

Das Ziel: Nachbildung der menschlichen **visuellen Wahrnehmung**.

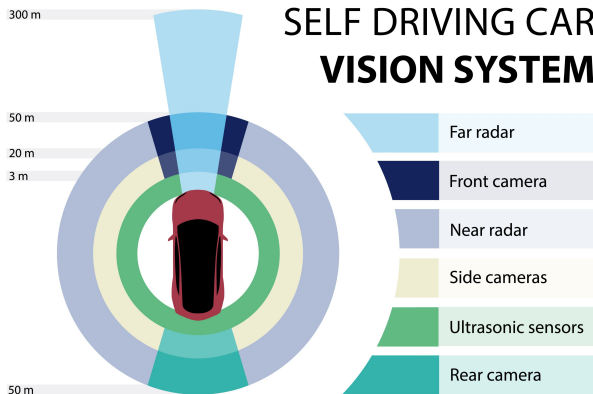
Low vs. High Level Computer Vision



Anwendungsgebiete

► Automotive (Autonomes Fahren):

Spurhalteassistenten, Verkehrszeichenerkennung, Fußgängererkennung (Tesla Autopilot, Mobileye).



Anwendungsgebiete

- Medizintechnik (Medical Imaging):

Automatisierte Tumorkennung in MRT/CT-Scans, Zellzählung, KI-gestützte Diagnostik.



Anwendungsgebiete

► Industrie 4.0 & Robotik:

Qualitätskontrolle (Defekterkennung), Bin Picking (Roboter greift unsortierte Teile), visuelle Navigation.



Anwendungsgebiete

- ▶ Augmented Reality (AR) & Entertainment:

Gesichtsfilter (Snapchat/TikTok), AR-Navigation (Google Maps), Überlagerung von Montageanleitungen (HoloLens).

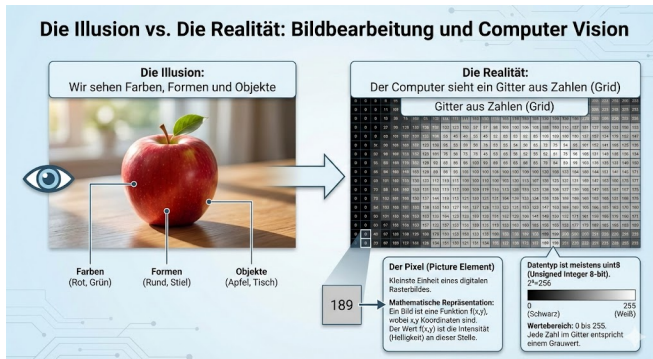


Wie "sieht" ein Computer? Das Bild als Datenmatrix

Die Illusion: Wir sehen Farben, Formen und Objekte.

Die Realität: Der Computer sieht ein Gitter aus Zahlen (Grid).

Der Pixel (Picture Element): Kleinste Einheit eines digitalen Rasterbildes.



Mathematische Repräsentation

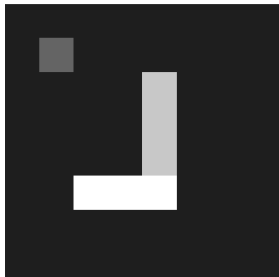
Datentyp ist meistens uint8 (Unsigned Integer 8-bit). $2^8 = 256$
Graustufen

Wertebereich: 0 (Schwarz) bis 255 (Weiß).

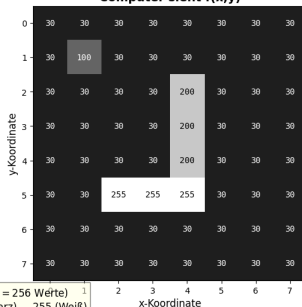
Mathematische Repräsentation:

- ▶ Ein Bild ist eine Funktion $f(x,y)$, wobei x,y Koordinaten sind.
- ▶ Der Wert $f(x,y)$ ist die Intensität (Helligkeit) an dieser Stelle.

Die Illusion:
"Wir sehen Formen"



Die Realität:
"Computer sieht $f(x,y)$ "



Datentyp: uint8 ($2^8 = 256$ Werte)
Wertebereich: 0 (Schwarz) ... 255 (Weiß)
Modell: $f(x, y) = \text{Intensität}$

Helligkeit und Kontrast

- ▶ **Helligkeit:** Beschreibt die allgemeine Intensität aller Pixel in einem Bild. Ein höheres Helligkeitsniveau bedeutet, dass das Bild insgesamt heller erscheint.
- ▶ **Kontrast:** Beschreibt den Unterschied zwischen den hellsten und dunkelsten Bereichen eines Bildes. Ein hoher Kontrast bedeutet, dass die Unterschiede zwischen den verschiedenen Grauwerten deutlich sichtbar sind.

Normalisierung/Standardisierung: Durch eine einheitliche Anpassung von Helligkeit und Kontrast können Bilder verschiedener Patienten oder Geräte besser verglichen werden.

Kanäle & Farbräume (RGB)

Graustufenbild: Matrix Dimension: (Höhe \times Breite \times 1).

Farbbild (RGB): Matrix Dimension: (Höhe \times Breite \times 3).

Tensor-Dimensionen:

- ▶ Ein Bild ist ein 3D-Array (Tensor): (H,W,C).

H: Height (Zeilen), W: Width (Spalten), C: Channels (Kanäle).

- ▶ Drei übereinanderliegende Matrizen (Channels):

Red, **Green**, **Blue** (Additive Farbmischung)

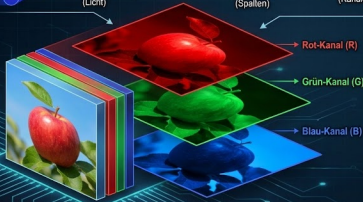
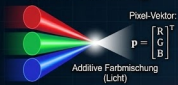
Beispiel Pixel:

- ▶ Rot: [255, 0, 0]
- ▶ Weiß: [255, 255, 255]
- ▶ Schwarz: [0, 0, 0]

Achtung bei OpenCV: OpenCV liest standardmäßig als BGR (Blue-Green-Red), nicht RGB!

Farbkanäle (RGB vs. BGR): Mehr als nur Grau – Tensor-Strukturen

Das RGB Modell (Additive Farbmischung)



Der OpenCV-Quirk (BGR vs. RGB)



OpenCV speichert Bilder historisch bedingt als BGR (Blue-Green-Red)



Matplotlib erwartet RGB



RGB (Korrekt für Matplotlib)



Wenn konvertiert (cv2.cvtColor)

Folge: Wenn nicht konvertiert, sehen rote Objekte blau aus.

BGR (Falsch in Matplotlib)



Die Lösung: Farbkonvertierung mit `cv2.cvtColor(image, cv2.COLOR_BGR2RGB)`

Graustufen (Grayscale)

Warum? Reduktion der Datenmenge um Faktor 3.

- ▶ Speichereffizienz
- ▶ Recheneffizienz

Viele Algorithmen (Kantenerkennung) basieren auf Helligkeitsunterschieden, Farbe ist oft irrelevante Information.

- ▶ Näherungsformel:

$$Y = 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$

Das menschliche Auge ist empfindlicher für Grün.

Alternative Farbräume (HSV)

Das Problem mit RGB:

- ▶ RGB korreliert Farbe und Helligkeit.
- ▶ Beispiel: Ein rotes Auto im Schatten hat völlig andere RGB-Werte als in der Sonne ($R=100$ vs. $R=200$). Wie soll man programmieren: `if pixel == red`?

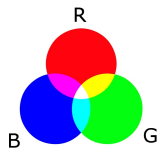
Die Lösung: **HSV** (Hue, Saturation, Value)

- ▶ Hue (Farbton): Winkel 0-179 (in OpenCV halbiert, damit es in 8-bit passt). Rot ist bei 0/180.
- ▶ Saturation (Sättigung): Wie "rein" ist die Farbe?
- ▶ Value (Helligkeit): Wie hell/dunkel ist die Farbe?

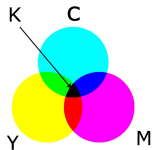
Vorteil: Trennt Farbinformation von der Beleuchtung. Besser für Farberkennung bei wechselndem Licht (z.B. "Erkenne den roten Ball im Schatten").

HSV vs. RGB

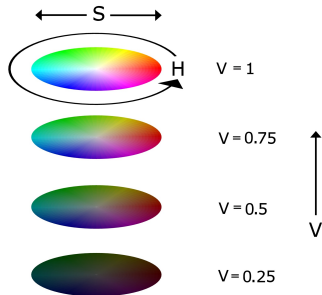
RGB



CMYK



HSV



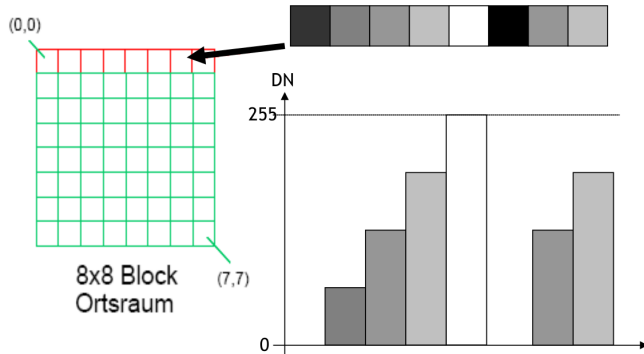
Helligkeitsvariationen in Bildern

Ein Bild kann als die Summe von Helligkeitsvariationen unterschiedlicher räumlicher Frequenz betrachtet werden

Die **räumliche Frequenz** bezieht sich auf die **Anzahl der (periodischen) Variationen** der Helligkeitswerte (in z.B. Linienpaar pro mm)

- ▶ Hohe räumliche Frequenz: **abrupte Variation** der Helligkeitswerte in einer Richtung (z.B. Grenze Schwarz/Weiß)
- ▶ Niedrige räumliche Frequenz: **allmähliche Variation** der Helligkeitswerte in einer Richtung (z.B. Abstufung von Grauwerten)

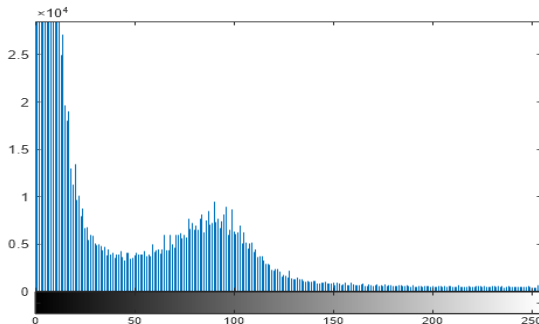
Räumliche Variation der Helligkeit im Bild



Histogramm

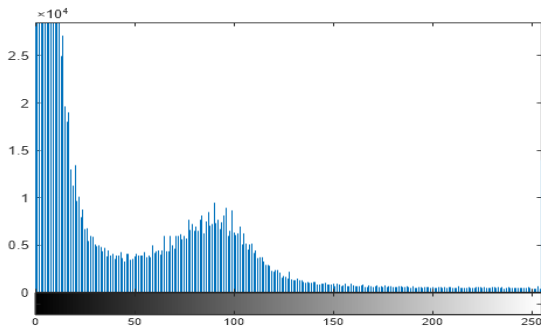
Häufigkeitsverteilung der Pixelwerte im Bild.

- ▶ X-Achse: Pixelwert (0-255).
- ▶ Y-Achse: Anzahl der Pixel mit diesem Wert.



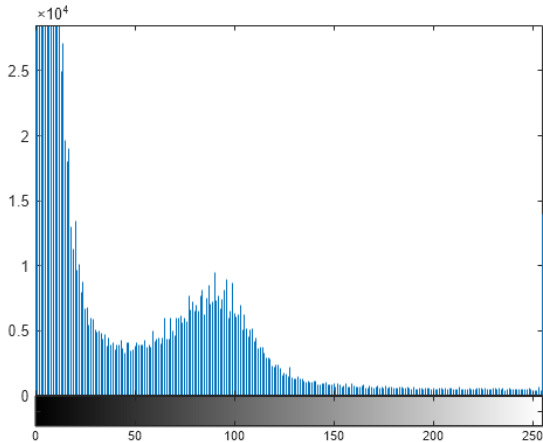
Anwendung: Automatische Belichtungskorrektur,
Schwellenwertbestimmung (Thresholding).

Histogramminterpretation

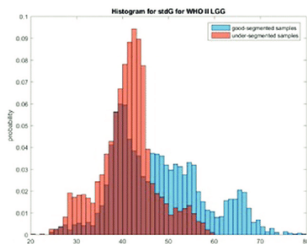
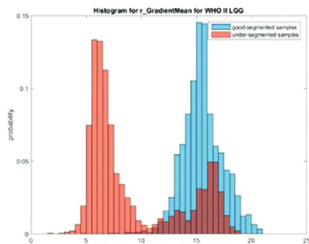


Histogramm lesen lernen als “Fingerabdruck” der Belichtung:

- ▶ Dynamikbereich: Wie groß ist der Bereich zwischen dem dunkelsten und hellsten Pixel?
- ▶ Kontrast: Wie stark sind die Unterschiede zwischen den Grauwerten?
- ▶ Helligkeit: Wo liegen die Modi (“Peaks”) der Verteilung?



- ▶ Schmäler Berg in der Mitte = Kontrastarm (grau in grau).
- ▶ Breiter Berg über alles = Guter Kontrast.

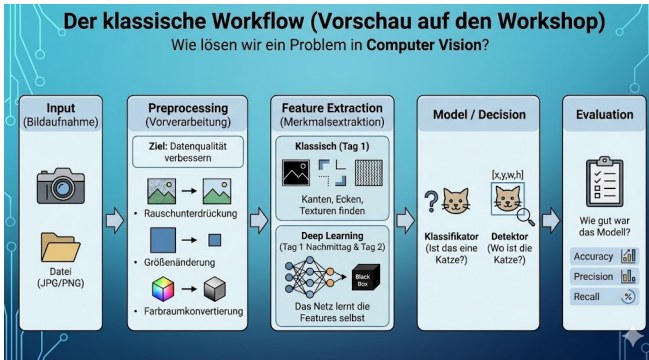


- Zwei Berge (bimodal) = Gute Trennbarkeit (z.B. dunkles Objekt auf hellem Grund → einfaches Thresholding möglich).

Workflow

Der klassische Workflow (Vorschau auf den Workshop)

Wie lösen wir ein Problem in Computer Vision?



Zu den Übungen

Github Repository zu Computer Vision

Beispiel 1 und Strawberry, Gentiana Bilder herunterladen

Ausführen mithilfe von

- ▶ Jupyter Labs
- ▶ Google Colab