

Bildverarbeitung mit NumPy

Christian Wilms

Computer Vision Group
Universität Hamburg

Sommersemester 2020

24. April 2020

Wie sieht der Computer das Bild?



Was wir sehen.

39	24	9	15	19	27	36	37	32	27	26	36	36	35	34	36	41	50	57	50	47	
51	40	19	12	16	26	36	38	34	31	31	39	34	33	41	41	53	60	57	51	61	56
48	48	39	13	17	25	33	35	32	31	34	36	33	34	45	59	64	56	46	64	61	
35	45	50	18	20	25	31	31	28	29	33	31	34	39	46	51	52	50	48	59	59	
19	36	61	35	17	19	35	37	32	31	33	25	41	50	46	46	52	51	43	61	53	
24	28	38	79	51	23	26	49	59	42	21	41	53	58	48	40	46	57	64	68	75	
42	37	34	71	60	32	27	60	77	56	33	43	52	62	68	70	77	90	101	107	114	
34	29	20	22	45	42	35	52	54	42	45	86	80	81	93	103	106	107	110	99	117	
52	52	46	39	65	73	76	88	84	86	112	99	87	81	91	103	111	120	130	163	162	
86	85	80	88	81	68	69	76	67	65	84	96	100	112	128	140	151	169	185	212	210	
88	82	77	81	70	71	91	110	117	126	139	163	175	190	200	203	205	210	216	232	224	
106	109	110	132	137	159	179	182	183	192	200	214	215	215	214	216	218	217	214	222	223	
169	190	190	201	203	210	215	214	209	207	209	210	217	215	215	213	211	218	225	225		
210	217	209	208	210	215	219	216	210	207	209	210	217	215	213	216	213	212	218	224	225	
208	214	208	208	210	214	218	215	208	206	208	211	217	216	213	216	214	212	219	224	225	
198	207	204	206	208	213	216	214	209	208	210	211	218	217	214	217	214	213	219	223	225	
201	207	203	210	211	216	219	217	212	211	214	212	219	217	215	217	215	213	220	222	224	
197	207	207	212	212	216	218	215	209	208	211	213	219	218	215	216	216	214	221	222	224	
200	212	215	208	209	212	215	212	207	206	210	213	220	218	216	219	216	215	221	222	224	
204	208	205	207	208	212	216	215	211	212	216	213	220	219	216	219	216	215	221	221	224	
212	210	208	205	210	214	215	212	210	211	213	220	220	220	218	215	216	220	224	225		

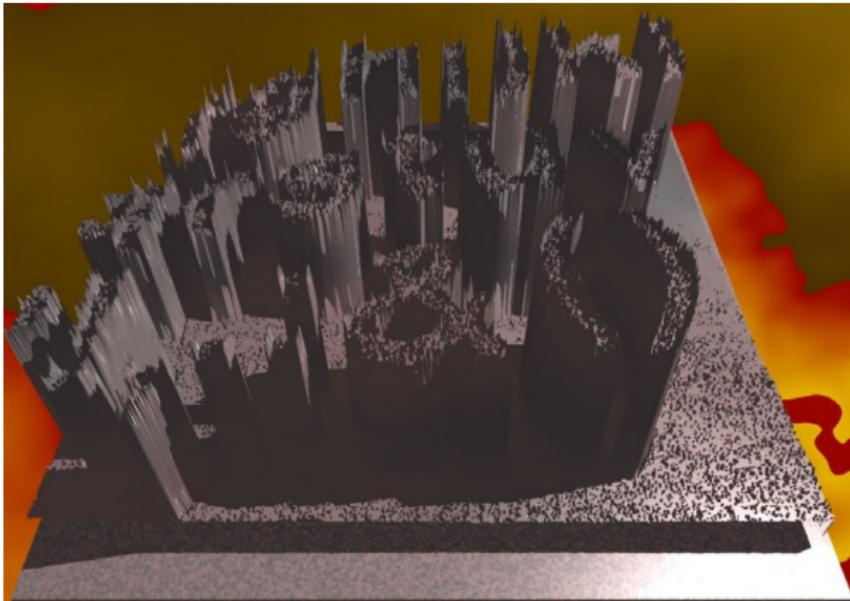
Was der Computer sieht.

Bilder im Computer

- üblicherweise rechteckiges Gitter (Array)
- in jeder Arrayzelle steht ein ganzzahliger Farbwert (oder ein Vektor)
- Farbwert bei Graustufenbildern bedeutet Helligkeit
- Weiß hat den Wert 255, Schwarz den Wert 0

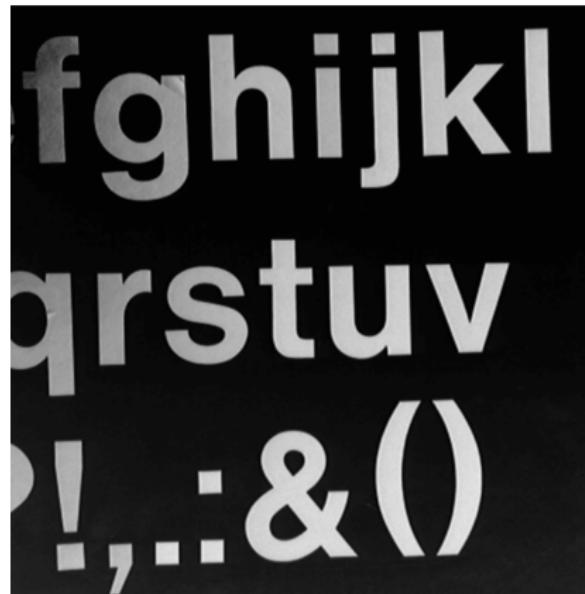
Alle weiteren Analysen des Bildes starten von diesen Helligkeitswerten des Arrays!

Ein anderer Blick auf Bilder - Grauwertgebirge



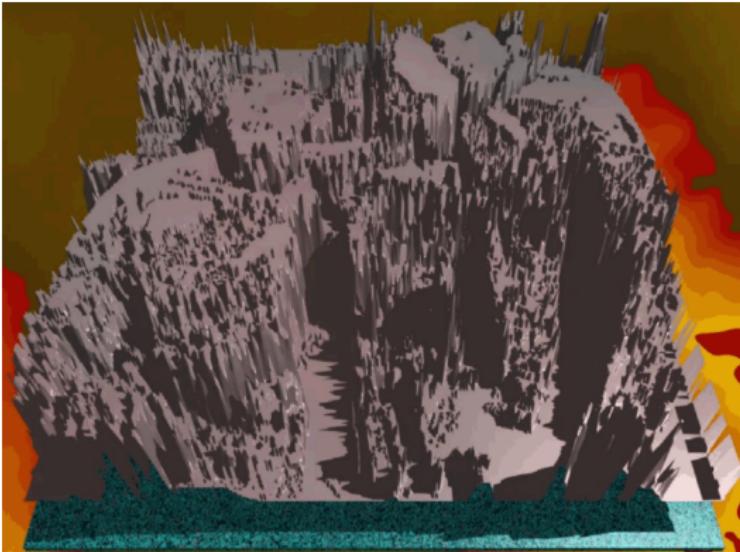
Was sehen wir hier?

Ein anderer Blick auf Bilder - Grauwertgebirge



Buchstaben!

Ein anderer Blick auf Bilder - Grauwertgebirge



Was sehen wir hier?

Ein anderer Blick auf Bilder - Grauwertgebirge



Star Trek!

Was ist...

NumPy

- eine Python-Bibliothek für das 'wissenschaftliche Rechnen'
- bietet effiziente nd-Arrays an
- viele mathematische Funktionen für die Arrays
- einfache broadcast-Operationen (Vektorisierung)
- enthält Paket für Lineare Algebra
- sehr flexibler Baukasten

scikit-image (skimage)

matplotlib

Was ist...

NumPy

scikit-image (skimage)

- Bibliothek für Bildverarbeitung
- speziell konzipiert, um mit NumPy zusammen zu arbeiten
- erstmal nur zum Laden und Speichern der Bilder

matplotlib

Was ist...

NumPy

scikit-image (skimage)

matplotlib

- Bibliothek zum Plotten von 2D Grafiken/Bildern
- hauptsächlich für mathematische Darstellungen und Bilder
- interaktive Betrachtung möglich
- manchmal etwas kompliziert in der Bedienung...

Bilder in NumPy - I

- Bilder werden als Arrays repräsentiert
- Funktionen für Laden und Speichern eines Bildes von skimage
- Verarbeitung der Arrays über die Funktionen von NumPy
- Arrays haben einen Datentyp



```
>>> import numpy as np
>>> from skimage.io import imread, imsave
>>> img = imread("./icon.png")
>>> # your magic here
>>> imsave("./iconModified.png", img)
```

Bilder in NumPy - II

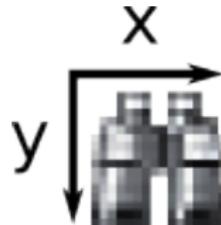


```
>>> img
array([[255, 255, 255, 90, 160, 132, 88, 255, 255, 93, 150, 110, 68, 255, 255, 255],
       [255, 255, 255, 100, 252, 235, 156, 255, 255, 165, 252, 232, 95, 255, 255, 255],
       [255, 255, 255, 75, 151, 89, 42, 255, 255, 81, 136, 56, 52, 255, 255, 255],
       [255, 124, 146, 82, 164, 114, 57, 255, 255, 152, 239, 78, 44, 63, 52, 255],
       [255, 122, 225, 192, 120, 80, 85, 60, 47, 174, 121, 157, 135, 144, 79, 255],
       [108, 93, 207, 232, 222, 119, 97, 100, 35, 83, 68, 249, 233, 220, 144, 64],
       [124, 109, 212, 213, 191, 108, 90, 89, 37, 101, 78, 227, 202, 180, 157, 63],
       [116, 132, 230, 239, 216, 118, 96, 94, 38, 104, 82, 234, 245, 208, 223, 79],
       [ 98, 53, 52, 43, 33, 57, 105, 86, 39, 105, 71, 20, 9, 10, 11, 38],
       [107, 130, 245, 173, 107, 107, 79, 99, 111, 82, 160, 240, 124, 109, 76, 59],
       [109, 163, 247, 153, 110, 107, 78, 255, 255, 90, 166, 242, 122, 108, 93, 66],
       [101, 179, 229, 128, 102, 99, 70, 255, 255, 96, 164, 221, 109, 97, 93, 60],
       [106, 200, 235, 134, 121, 104, 73, 255, 255, 106, 168, 241, 126, 109, 91, 59],
       [ 74, 200, 182, 80, 75, 70, 68, 255, 255, 100, 186, 159, 76, 71, 69, 45],
       [ 35, 179, 111, 16, 16, 25, 55, 255, 255, 68, 228, 47, 15, 21, 44, 49],
       [ 57, 132, 96, 41, 40, 44, 53, 255, 255, 53, 130, 49, 30, 32, 41, 44]], dtype=
uint8)
```

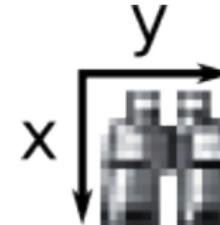
Bilder in NumPy - III

```
>>> img.ndim  
2  
>>> img.shape  
(16, 16)  
>>> img.dtype  
dtype('uint8')
```

Vorsicht bei den Koordinaten!



häufige Darstellung



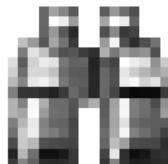
NumPy

Pixel und Bildausschnitte in NumPy

Der Zugriff auf das Bild funktioniert wie bei Listen, nur jetzt mit zwei Dimensionen!

```
img[startX:stopX:stepX, startY:stopY:stepY]
```

```
>>> img[0,0] #Pixel oben links  
255  
>>> img[3,:] #vierte Bildzeile  
  
>>> img[:,4] #fuenfte Bildspalte  
  
>>> img[:,8] #linke Bildhaelfte
```



Bilder und Schleifen

Man kann über ein Bild mit Schleifen iterieren...

```
for x in range(img.shape[0]):  
    for y in range(img.shape[1]):  
        img[x,y]
```

... allerdings nur, wenn nicht anders möglich.

Besser Broadcasting oder NumPy-Funktionen nutzen:

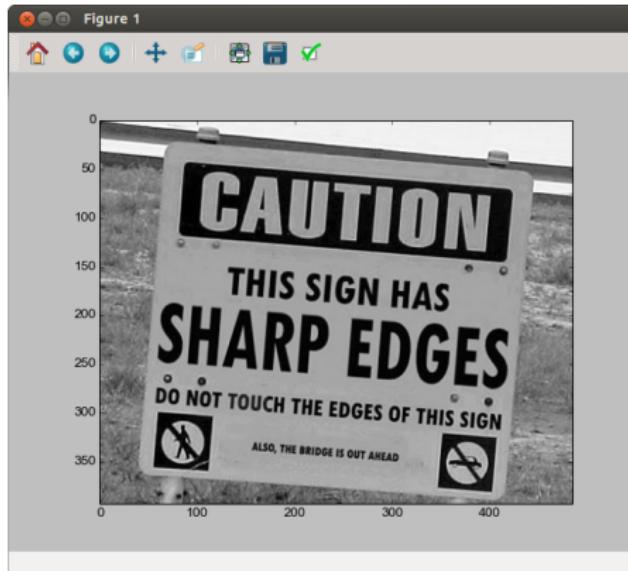
```
>>> imgSum = img1 + img2  
>>> imgSum = img1 + 1  
>>> imgSum = img1 == img2
```

Funktionen von NumPy (Auswahl)

```
>>> import numpy as np
>>> a = np.array([1,2,3]) #Erstellen eines Arrays
>>> np.max(a) #Maximum von a
3
>>> np.argmax(a) #Index des Maximums
2
>>> np.sum(a) #Summe aller Elemente
6
>>> np.where(a==2) #alle Indizes, wo Bedingung gilt
(array([1]),)
>>> b = np.zeros_like(a) #Array aus Nullen wie a
>>> np.maximum(a,b) #Maximum pro Element
np.array([1,2,3])
>>> c = np.copy(a) #Deepcopy eines Arrays
```

Bilder anzeigen mit matplotlib

```
>>> import matplotlib.pyplot as plt  
>>> plt.imshow(sign, cmap="gray")
```



Spyder: Bilder in eigenem Fenster anzeigen (interaktives Fenster)

Falls in Spyder das Bild nicht in einem eigenen Fenster geöffnet wird:

- Tools → Preferences → IPython console → Graphics
- unter Graphics backend die Option Automatic auswählen
- IPython Console neu starten

Zum automatischen Schließen des Fensters beim nächsten Start des Skripts nutzt plt.close('all').

```
>>> import matplotlib.pyplot as plt
>>> plt.close('all')
>>> plt.imshow(sign, cmap="gray")
```

NumPy

Alle diese Referenzen behandeln in etwa einen ähnlichen Kern von NumPy Funktionen und Beispielen. Die Ausnahme davon bildet das detailliertere Tutorial zum Thema Broadcasting in NumPy.

- Offizielles Tutorial zu NumPy NumPy Quickstart Tutorial ↗
- Detaillierteres Tutorial zum Thema Broadcasting in NumPy (Anwendung von arithmetischen Operation auf Arrays verschiedener Formen/Skalare) NumPy Basics - Broadcasting ↗
- Video: Numpy Array Slicing - Tutorial on Array Slicing in Numpy - Python Programmer @ YouTube ↗
- NumPy Tutorial in Kapitel 1.3.1.-1.3.3. Scipy Lecture Notes ↗
- J. Vanderplas, Python Data Science Handbook: Essential Tools for Working with Data, O'Reilly, 2016 (NumPy-Einführung in Kapitel 2) Bib-Katalog/E-Book ↗

matplotlib

- Offizielles Tutorial zu matplotlib: Beispiele und Möglichkeiten beim Plotten von Bildern ↗
- Offizielles Tutorial zu matplotlib: Viele weitere Beispiele zu den Möglichkeiten von matplotlib ↗
- Kurzes Tutorial zum Anzeigen von Bildern mit matplotlib (auch mehrere Bilder und verschiedene Colormaps) ↗
- Video: Displaying Multiple Images with Matplotlib - Ashwin Pajankar @ YouTube ↗ Hinweis: Sämtliche OpenCV-Befehle (cv2) können ignoriert werden, sie dienen lediglich dem Einlesen und bereitstellen der Bilder. Es werden hier Farbbilder verwendet.