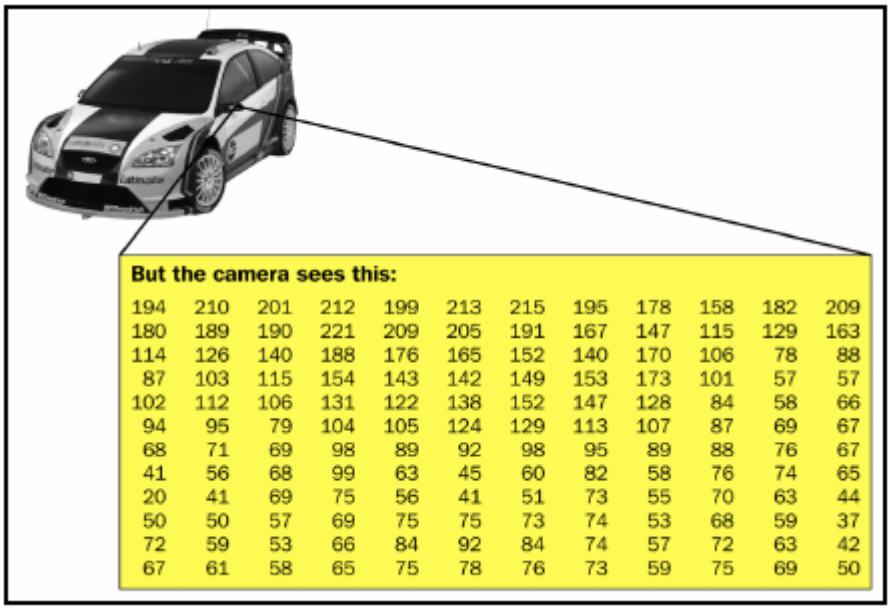
OpenCV

Was ist das?

Eine weit verbreitete Opensource Software für Programmbibliothek für Computervision und Maschinenlernen, die verschiedene Formen der Detektoren, Filter und Bildableitungen ermöglicht.

Kern Funktonalität:

Die Hauptfunktionalität von OpenCV besteht aus der Verarbeitung von Bildern. Um diese effizient zu verarbeiten, benötigen sie ein spezielles Speicherformat, dieses hier verwendete Format heißt Mat und ist eine Klasse, die die Werte der einzelnen Pixel repräsentiert. Dabei besitzt jedes Pixel dieser Klasse eine Größe von 8 Bit, bei Farbildern kann durch die Verwendung von drei Farbkanälen eine Farbtiefe von 24 Bit erreicht werden.



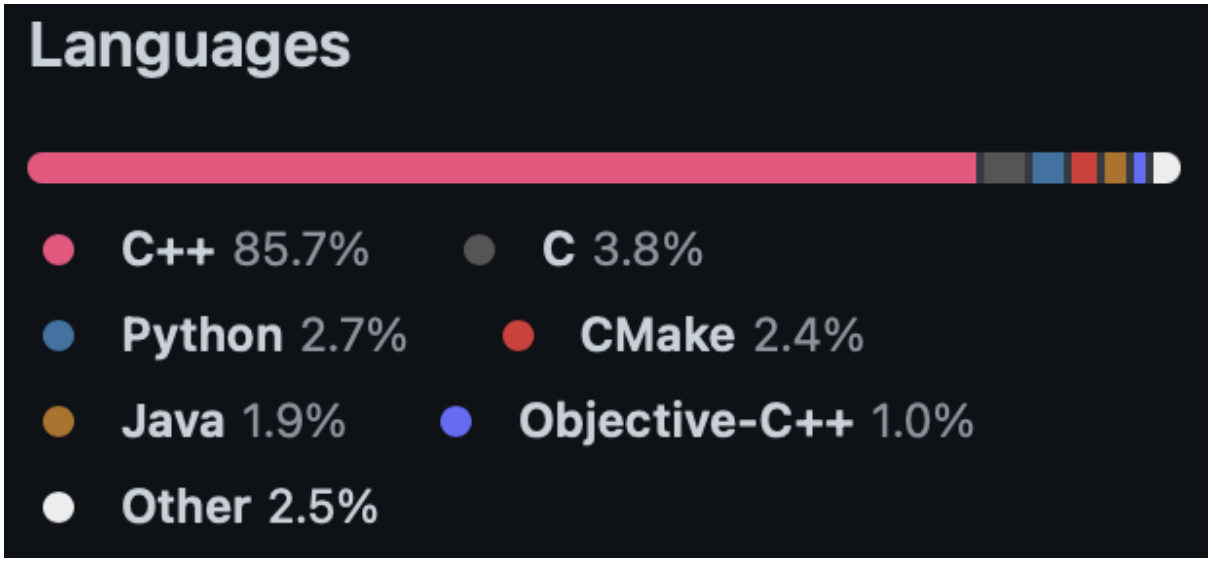
Darüber hinaus besitzt die Klasse einen Header der Informationen über die Bilddimensionen beinhaltet.

Namensgebung, Historie

- Entwickelt von Intel und der Willow Garage

Initiiert von Intel mit dem “Hintergedanken” die Computer Vision voranzutreiben und somit einen Bedarf an schnelleren Prozessoren zu erwecken, um mehr von diesen zu verkaufen, bis 2013 von der Willow Garage gepflegt, nun bei Itseez fortgeführt welches seit 2016 zu Intel gehört

- Erstmals erschienen Juni 2000.
- Plattformübergreifend: Windows, Linux, Mac, Android und Java.
- Programmiert in C++ (Laufzeit optimiert, advantage of multicore processors).



- Verwendbar in C, C++, Java und Python

▼ Apache-Lizenz

  - Man darf Software unter dieser Lizenz frei in jedem Umfeld verwenden, modifizieren und verteilen.
  - Eine Kopie der Lizenz (oder der Verweis auf die Lizenz) muss dem Paket beiliegen.
  - Änderungen am Quellcode der unter der Apache-Lizenz stehenden Software müssen nicht zum Lizenzgeber zurückgeschickt werden.
  - Eigene Software, die unter Apache-Lizenz stehende Software verwendet, muss nicht unter der Apache-Lizenz stehen.
  - Die eigene Software darf nur dann *Apache* heißen, wenn eine schriftliche Genehmigung der Apache Foundation vorliegt.

Funktionsumfang

- 2 und 3D Merkmalsextraktion
- Klassische Filter
- Gesichter und Gesten erkennen
- Mensch-Maschine Interaktion
- Segmentierung
- Klassifizierung
- Steriskopisches Sehen
- Kalman-Filter

Maschinelles Lernen:

- NNK Nächster Nachbar Klassifikation
- Deep Learning

Vor und Nachteile:

Vorteile

- Auf allen Plattformen verfügbar
- Schnelle Berechnungen
- Weit verbreitet
- Freie Lizenz
- Sehr gut dokumentiert
- Kompatibel mit anderen Bibliotheken

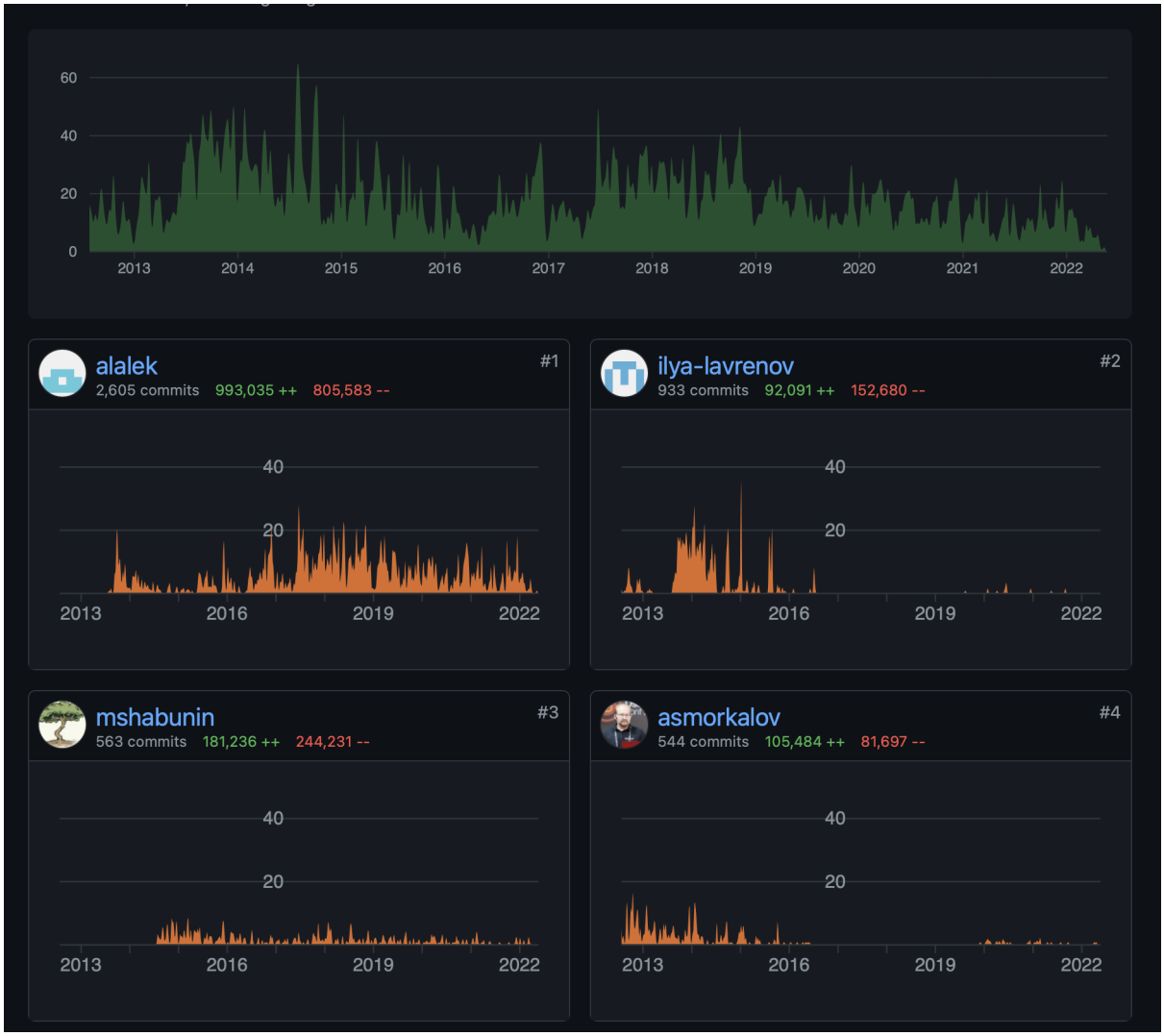
Nachteile

- Nicht ganz einfache Nutzung im Vergleich zu GUI Anwendungen
- Teilweise aufwendige Installation

Wo gehosted? (Wo sind die Quellen zu finden?)

Git Hub: https://github.com/opencv/opencv

Wer sind die Maintainer (aktiven Betreuer der Software)?



Hardware

Min 4GB Ram

GPU Mitnutzung bei Grafikkarten mit CUDA Codec

Installation

Verdammt schwierig mit wie auf der Seite angegebn (Build from source/git bash/cmake)

Mit PIP oder Anaconda viel einfacher, Inhalt kann aber vom Orginal abweichen, weil die Anbieter dies selbstständig organisieren.

```
pip3 install opencv-python %%Mac/Linux
pip install opencv-python  %%Windows
```

Darüber hinaus wird NumPy benötigt

Auserdem sind im Kontext von OpenCV die Bibliotheken **SciPy, OpenNI und SensorKinect** hilfreich

Mini-Übung betreuen

Bild Öffnen und Anzeigen:

```
import cv2    ##Laden der Bibiothek

filename = "example.png".  ##Pfad zum Bild zuweisen

image = cv2.imread(filename,0)  ##Bild Laden
if image is None:               ##Prüfen ob das öffnen geklapt hat
    print("Unable to open " + filename)
    exit(-1)

cv2.imshow("An example image", image)    ##Bild anzeigen
cv2.waitKey(0)
cv2.destroyAllWindows()
```

WebCam als Bild Quelle:

```
import cv2
   clicked = False
   def onMouse(event, x, y, flags, param):
       global clicked
       if event == cv2.EVENT_LBUTTONUP:
           clicked = True
   cameraCapture = cv2.VideoCapture(0)
   cv2.namedWindow('MyWindow')
   cv2.setMouseCallback('MyWindow', onMouse)
   print 'Showing camera feed. Click window or press any key to
       stop.'
   success, frame = cameraCapture.read()
   while success and cv2.waitKey(1) == -1 and not clicked:
       cv2.imshow('MyWindow', frame)
       success, frame = cameraCapture.read()
   cv2.destroyWindow('MyWindow')
   cameraCapture.release(
```

Kanten Erkenung:

```
import cv2
import numpy as np
```

```
image = cv2.imread('imge.png',0)

height, width = image.shape

canny = cv2.Canny(image, 50, 120)
cv2.imshow('Canny', canny)

cv2.destroyAllWindows()
```