

Aufgabenblatt 1

Dies ist das erste Aufgabenblatt zum Praktikum des Kurses "Medizinische Bildverarbeitung und bildgebende Verfahren in der Medizin". Das Praktikum wird eine Reihe von Aufgabenblättern sein, die von dir gelöst und jede Woche vorgestellt werden müssen.

Inhaltlich baue ich auf der Vorlesung auf und werde dir über das Semester Programmierungsprobleme mit steigender Komplexität geben, die du mit den gelernten Methoden und Algorithmen lösen sollst. Zur Lösung werden wir die Programmiersprache **Python** nutzen. Wenn du noch nie mit Python gearbeitet hast, ist dies ein guter Zeitpunkt, um damit anzufangen.

Python hat viele Vorteile, gerade im Prototyping Bereich kommt man nicht um diese Sprache herum. Sie bietet einen einfachen Einstieg an und lebt von einem großen Eco-System mit tausenden von Bibliotheken. Solltest du bereits mit Python gearbeitet haben, kannst du direkt zur ersten Aufgabe springen. Ansonsten findest du eine Erklärung, wie du Python installierst am Ende dieses Dokumentes.

1. Anzeigen eines Bildes in Python

Lade dir ein beliebiges Bild (wenn möglich im JPG Format) aus dem Internet herunter und öffne es in Python. Nutze das Paket **Numpy**, um aus dem Bild ein Numpy Array zu machen und plotte dieses Array mit dem Paket **Matplotlib**.

1.1 Hilfestellung

Bilder einlesen mit Python

Python selbst kann nichts mit Bildern anfangen. Deshalb benutzen wir die Library **Python Image Library** oder kurz PIL. Sollte diese nicht bereits installiert sein, kannst du sie mit

```
pip install Pillow
```

installieren.

PIL bietet viele unterschiedliche Teilpakete, wir brauchen hier das Paket **Image** um ein Bild einzulesen. Ich empfehle auch mal in die Dokumentation von Pillow hineinzuschauen:

- [Pillow Dokumentation](#)

Numpy Arrays als Bildersatz

Numpy ist eigentlich eine mathematische Library. Jedoch können wir Bilder als 2-Dimensionale Arrays ansehen und diese so in Numpy konvertieren. Da Pillow weitbekannt ist, hat auch Numpy eine Methode um Pillow-Bilder in Numpy-Arrays umzuwandeln. Matplotlib verbindet sogar Pillow und Numpy zu einem einzigen Methodenaufruf.

- [Numpy Dokumentation](#)


- [Matplotlib Dokumentation](#)

Matplotlib Bilder darstellen

Kapitel [0.5.1 Matplotlib Plots anzeigen](#) gibt hier schon die Lösung.

2. Tiling

Erweitere nun den Code, in dem du eine Funktion hinzufügst, die das Bild mehrmals vertikal und horizontal kopiert. Hier ein Beispiel:

Wir haben das Bild  gegeben. Die Funktion bekommt als Input die Anzahl der Vertikalen und Horizontalen Kopien, in diesem Fall

Vertikal: 3

Horizontal: 4

Das Ergebnis sollte wie folgt aussehen:

XXXX

XXXX

XXXX

Plote das Endergebnis mit Matplotlib als ein Bild.

2.1 Hilfestellung

Auch hier kommt Numpy zur Rettung. Bei der Verarbeitung von Bildern werden wir in fast allen Übungen - mit Pillow/Matplotlib das Bild einlesen - mit Numpy das Bild verändern - mit Matplotlib das Bild ausgeben.

Deshalb brauchen wir nur eine Methode, Numpy Arrays zu konkatinieren.

3. Cropping

Erweitere dein Programm um eine Funktion, die das Bild nun ausschneidet. Hierbei sollte man sowohl Start- als auch Endposition in vertikaler und horizontaler Position angeben können. Plote das Ergebnis (den ausgeschnittenen Teil) mit Matplotlib.

In Zukunft, ist es nicht anders gefordert, solltest du immer das Ergebnis mit Matplotlib darstellen.

3.1 Hilfestellung

Ausschneiden können wir mit verschiedensten Methoden erreichen. Ich würde aber hier auf die Numpy Array Indexing Funktionalität mal [hinweisen](#).

0. Python Installation

0.1 Python 3.11 installieren

Lade dir die neuste Python Version 3.11 von <https://www.python.org/downloads/> herunter und installiere es. Damit einhergehend sollte direkt auch "pip" installiert sein, der Python Package Manager. Du kannst die erfolgreiche Installation überprüfen, indem du ein Terminal öffnest und

```
python --version
python -m pip --version
```

eingibst.

0.2 Python Packages

Der normale Weg, um Packages zu installieren, läuft nun über pip. Der erste Schritt sind die Pakete Numpy, Matplotlib und OpenCV.

```
pip install matplotlib
pip install numpy
pip install opencv-python
```

0.3 IDE of Choice

Du brauchst ein Integrated Development Environment (IDE), also einen Editor, mit dem du deinen Python Code schreibst. PyCharm und Spyder sind Optionen für Python, ich nutze und empfehle aber Visual Studio Code. Herunterladen kannst du das hier: <https://code.visualstudio.com/>.

0.4 Python programmieren mit VS Code

Der erste Schritt, um in Python mit VS Code zu programmieren, ist einen Ordner (den Projektordner) mit VS Code zu öffnen. Dies kannst du im Tab "Explorer" im Code auswählen. Danach erstellst du Python-Dateien und Unterordner in VS Code (Rechtsklick im Code Explorer).

Hast du eine Python-Datei erstellt, kannst du sie mit Linksklick auf die Datei öffnen. Dein erster Schritt ist das Auswählen eines Interpreters für Python. Das bedeutet, du stellst ein, welche Python-Version VS Code nutzen wird.

Dafür drückst du die Taste **F1**. Dies öffnet eine "Command Search Bar", mit der du schneller und einfacher in den verschiedenen Einstellungen von Code navigieren und zusätzlich Funktionen von Extensions ausführen kannst.

Gib dort nun ein "Select Interpreter" und es sollte "Python: Select Interpreter" erscheinen. Mit ENTER bestätigst du dies und wählst dann die von dir installierte Python Version erneut mit ENTER aus.

0.5 Python Programme Ausführen

Nachdem du ein Python Programm geschrieben hast, kannst du dies direkt in VS Code ausführen. Oben Rechts ist ein "Play" Button, der die aktuelle Datei ausführt.

0.5.1 Matplotlib Plots anzeigen

VS Code benutzt im Hintergrund blankes Python und führt für dich Python Befehle im Terminal aus. Damit wir auch Plots anzeigen können, schreiben wir am Ende eines Programms folgendes:

```
plt.imshow(img)
plt.show()
```

Ohne die Methode "show" wird kein Plot angezeigt. In anderen IDEs kann dies auch nicht notwendig sein.

0.Z Zusatz

Im Laufe des Praktikums wirst du weitere Pakete brauchen. PIP Packages werden jedoch nicht über eine GUI installiert, sondern über ein Terminal. In VS Code kannst du eine integriertes Terminal mit STRG+Ö öffnen und schließen. Dieses Terminal kann die Windows Console, Git Bash, Powershell oder normales Bash selbst sein. Dort kannst du dann Befehle wie

```
pip install numpy
```

eingeben.