Hochschule Darmstadt
B. Frömmer, B. Meyer

h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES
fbi
FACHBEREICH INFORMATIK

16. DEZEMBER 2023

Visual Computing WiSe 2023/2024
Aufgabe 5

Das Ziel dieses Aufgabenblattes ist es, sich mit der OpenCV-Bibliothek vertraut zu machen. OpenCV bietet die Grundlagen für die Arbeit mit Bildern sowie viele Funktionen, die in der Bildverarbeitung und -analyse häufig verwendet werden. Sie ist als Open-Source-Projekt verfügbar unter: `http://opencv.org/`

4.1 Installation von OpenCV

Eine detaillierte Installationsanweisung für C++ liegt im Moodle bereit.
Eine gute Dokumentation finden Sie in dem Buch Learning OpenCV von Gary Bradski und Adrian Kaehler oder hier: `https://docs.opencv.org/4.7.0/d1/dfb/intro.html`

(Empfohlene) Alternative: Die Installation von OpenCV unter C++ kann etwas mühsam sein. Deutlich einfacher lässt sich OpenCV in ein Python-Projekt integrieren:

a) Installieren Sie eine Python-IDE Ihrer Wahl (beispielsweise PyCharm Community von JetBrains, kostenlos als Studentenlizenz unter `https://www.jetbrains.com/pycharm/download/#section=windows`)

b) Installieren Sie eine aktuelle Python-Version (unter `https://www.python.org/downloads/`, achten Sie bei der Installation darauf, dass die PATH-Variable entsprechend gesetzt wird)

c) Legen Sie testweise ein Python-Projekt in der IDE an. In PyCharm können Sie im Terminal mittels *python* $--$*version* die installierte Python-Versionsnummer ausgeben. Funktioniert das nicht, wird Ihre Python-Umgebung nicht gefunden.

d) Installation von zusätzlichen Packages: Python kommt mit einem integrierten Package Wizard, PIP (Package Installer for Python). Möchten Sie verschiedene Projekte mit unterschiedlichen Python-Versionen verwalten, empfiehlt sich der Package Manager *Anaconda* (für dieses Praktikum jedoch nicht benötigt).

e) Mit `pip install opencv-python` installieren Sie automatisch die aktuelle Version von OpenCV. Mehr Infos gibts unter `https://pypi.org/project/opencv-python/`.

f) Optional: Das Editieren von eingelesenen Bilddaten kann mit Hilfe der Mathebibliothek Numpy noch etwas vereinfacht werden. `pip install numpy` installiert die Bibliothek entsprechend.

Kopieren Sie das Bild `yoshi.png` in Ihren Projektordner und testen Sie abschließend Ihre Installation durch folgende Zeilen :

```
# import numpy as np ## optional
import cv2
print("OpenCV-Version: " + cv2.__version__)
img = cv2.imread("yoshi.png")
cv2.imshow("image", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4.2 Erste Schritte

Zunächst wollen wir uns mit einfachen OpenCV-Funktionen zur Bildmanipulation und -darstellung beschäftigen. Verwenden Sie bekannte Funktionalitäten oder nutzen Sie Online-Ressourcen wie die Dokumentation der OpenCV-Bibliothek, um:

- das beigefügte "yoshi.png" zu laden.
- die Breite, Höhe und Anzahl der Farbkanäle auf der Konsole auszugeben.
- das Bilddatenformat auf Fließkomma zu ändern.
- das Bild anzuzeigen, bis eine Taste gedrückt wird (sowohl als uint8 als auch als Float).
- ein rotes Quadrat mit 10x10 Pixeln in die Mitte des Bildes einzuzeichnen.
- jede 5. Zeile durch schwarze Pixel zu ersetzen.
- das Bild auf der Festplatte abzuspeichern.

4.3 Farbräume

In dieser Aufgabe wollen wir uns mit Farbraumkonvertierungen beschäftigen.

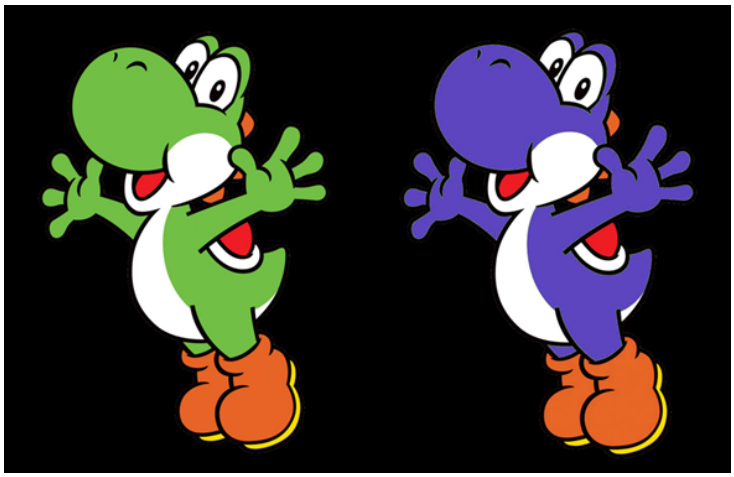
(1) Laden Sie zusätzlich zu Yoshi die zugehörige Maske "mask.png" in Ihr Programm.

(2) Übertragen Sie das Yoshi-Bild in den HSV-Farbraum.

(3) Ändern Sie für alle weißen Pixel im Maskenbild den H-Wert im Yoshi-Bild.

(4) Zeigen Sie das neue Bild auf dem Bildschirm an (beachten Sie aber, dass imshow() nur BGR-Bilder korrekt verarbeiten kann). Das Ergebnis sollte wie abgebildet aussehen.

(5) Optional: Bauen Sie einen Schieberegler, um den H-Wert manuell auf beliebige Werte einzustellen und das angezeigte Bild zu aktualisieren.



4.4 Reinhards Color Transfer

In dieser Übung werden Sie eine Variante des Reinhardschen Farbtransfers implementieren. (Quelle: Color Transfer between images von Reinhard, Ashikmin, Gooch und Shirley, verfügbar im Moodle). Die Idee des Color Transfers ist es, die grundsätzliche Farbgebung eines Bildes (hier: `Input`) auf ein bestimmtes Ziel-Farbschema (hier: `Target`) anzupassen. Ein Beispiel ist in Abb. 1 zu sehen.
Gehen Sie wie folgt vor:

(1) Laden Sie zwei Bilder als `"Input"` und `"Target"` und wandeln Sie sie in Fließkommazahlen um (im Bereich von 0 bis 1).

(2) Konvertieren Sie beide Bilder in den Lab-Farbraum.

(3) Für jeden Farbkanal (separat):

i) Subtrahieren Sie den Mittelwert des `Input` von dem `Input` (der neue Mittelwert des `Input` wird 0) .

ii) Teilen Sie den `Input` durch dessen Standardabweichung (die neue Standardabweichung wird 1).

iii) Multiplizieren Sie den `Input` mit der Standardabweichung des `Target`.

iv) Addieren Sie den Mittelwert des `Target` zum `Input`.

(4) Konvertieren Sie den `Input` zurück in BGR.

Testen Sie den Algorithmus auch mit den Ihnen bekannten Farbräumen RGB und HSV und anderen Bildern. Welche Farbräume sind für den Algorithmus geeignet, welche nicht? Mit welchen Bildern funktioniert die Farbtransformation gut? Können Sie erklären, warum?

Abbildung 1: **Links: Input-Bild mit hellblauem Himmel. Mitte: Target-Bild mit gewünschter Farbgebung. Rechts: angepasstes Input-Bild mit neuer Farbgebung.**