INSTITUT FÜR
INFORMATIONSVERARBEITUNG
LEIBNIZ UNIVERSITÄT HANNOVER

Appelstraße 9A
30167 Hannover

Labor zur Vorlesung

Digitale Bildverarbeitung

Datum:	XX.XX.XXXX
Uhrzeit:	XX:XX
Anzahl der Blätter:	17 (einschließlich Deckblatt)

Name: **Matrikelnummer:**

Die Bearbeitung der Aufgaben erfolgt selbstständig in Kleingruppen. Alle Gruppenmitglieder sollen Arbeitsaufwand in gleicher Größenordnung einbringen. Betrugsversuche werden geahndet.

Inhaltsverzeichnis

1 Einleitung

Digitale Bildverarbeitung wird unter anderem in industriellen Prozessen, medizinischen Verfahren oder in Multimedia-Produkten angewandt. In diesem Labor soll eine beispielhafte Multimedia-Anwendung entwickelt werden, die über das Labor hinaus als erheiterndes Demonstrationsobjekt für Bildverarbeitung in Videokonferenzen genutzt werden kann. Als Inspiration dient ein Effekt aus der Filmreihe „Harry Potter“.

Abbildung 1: Harry Potter ohne magischen Tarnumhang

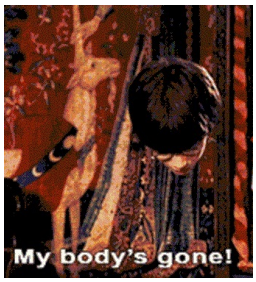

Abbildung 2: Harry Potter mit magischem Tarnumhang

Die Abbildungen 1 und 2 zeigen Harry Potter ohne [1] und mit[2] verzauberten Tarnumhang, welchen den Träger des Umhangs verschwinden lassen kann. In diesem Labor entwickeln Sie eine Bildverarbeitungs-Pipeline, mit welcher dieser Effekt mit einer einfachen Webcam simuliert werden kann. Sie festigen den Umgang mit

- Rauschunterdrückung
- Histogrammen
- verschiedenen Farbräumen
- Erosion und Dilatation
- Schwellwertverfahren

und üben gleichzeitig den Umgang mit der Programmiersprache Python.

[1] `https://assets.cdn.moviepilot.de/files/8a2cc0d2eb31c41136fb2be242540606dcd50821980e830481404b2760fill/1280/614/Harry%20Potter%20Tarnumhang.jpg`

[2] `https://www.tres-click.com/wp-content/uploads/2019/06/harry-potter-tarnumhang.gif`

1.1 Vorbereitung

Für das Labor wird ein Rechner mit Betriebssystem Windows, Linux-Ubuntu oder Mac benötigt. Zusätzlich muss eine Webcam vorhanden sein (im Laptop integriert oder extern). Melden Sie sich **vorab** bei dem Betreuer des Labors, sollte Ihnen eines der benötigten Mittel nicht zur Verfügung stehen.

Um an dem Labor erfolgreich teilnehmen zu können, muss die zu nutzende Programmierumgebung vorbereitet werden. Sollten Sie an der praktischen Übung zur Vorlesung *Digitale Bildverarbeitung* teilgenommen haben, ist die erforderliche Programmierumgebung sowie der Programmcode wahrscheinlich bereits installiert und eingerichtet. Ansonsten befolgen Sie die Installationsanweisungen auf `https://github.com/TimoK93/Digitale-Bildverarbeitung` im Bereich *0_Einführung*. Die Installation des Streaming Programms zur Erstellung einer virtuellen Kamera ist dabei optional und nicht verpflichtend.

Nachdem Sie die Installation vollendet haben, öffnen Sie die Programmierumgebung, sodass Ihr Bildschirm Ähnliches zu Abbildung 3 anzeigt.

Abbildung 3: Programmierumgebung in PyCharm

Machen Sie sich als nächstes mithilfe der Beschreibung in Kapitel 1.2 mit der Programmierumgebung vertraut.

1.2 Programmierumgebung

In der Programmierumgebung ist das Hauptprogramm bereits soweit vorbereitet, sodass Sie einen Videostream aus Ihrer Kamera auslesen, darauf einen vorbereiteten Beispielalgorithmus anwenden und diesen anzeigen können. In diesem Kapitel wird kurz erläutert, wie Sie diese Bildverarbeitungs-Pipeline starten und eigene Algorithmen integrieren können.

Starten der Bildverarbeitungs-Pipeline. Öffnen Sie die Datei *./CV-App/main.py* in PyCharm und klicken Sie mit der rechten Maustaste in den Programmcode. Öffnen Sie dann das Fenster *Modify Run Configuration...* wie in Abbildung 4 dargestellt. In dem Feld *Parameter* können Sie zusätzliche Argumente in die Umgebung eingeben. Argumente werden dabei im Schema *–Argument1 Value1 –Argument2 Value2* eingegeben. Die einzustellenden Parameter können Sie aus der Tabelle 1 entnehmen. Wenn Sie keine Argumente wählen, werden die Standard Einstellungen gewählt.

Tabelle 1: Argumente für die Programmausführung

Argument	Werte	Default-Wert
camera	-1, 0, 1, . . . (-1 öffnet ein Video)	0
mode	*virtual_cam* (virtuelle Kamera), *screen* (nur Fenster)	*virtual_cam*
video	Pfad zu Video, wenn *camera* den Wert -1 hat.	-

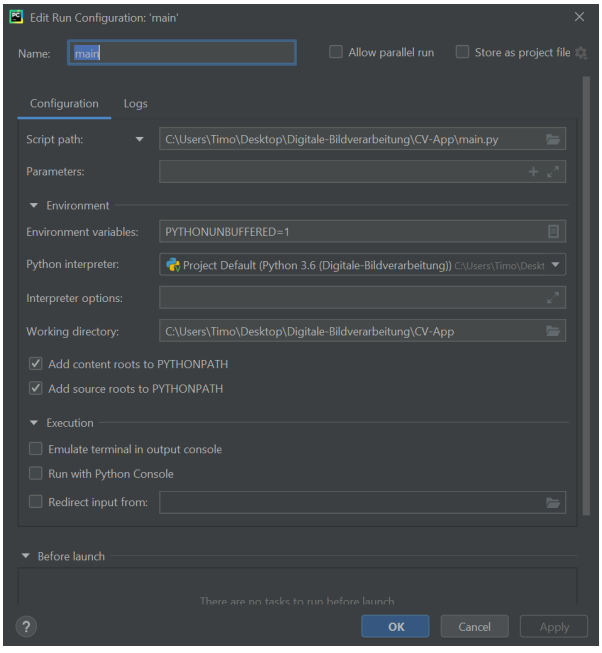

Abbildung 4: Run Configuration in PyCharm

Sie können das Programm nun mit *Run* starten. Kurz nach dem Start sollten Sie ein Fenster mit dem Videostream aus Ihrer Kamera sehen. Durch das Betätigen verschiedener Tasten auf der Tastatur (z.B. Taste 1 oder 2) können Sie verschiedene Algorithmen aktivieren. Sie sehen das Ergebnis direkt in dem ausgegebenem Videostream. Wichtig: Das Programm reagiert nur auf Tastendrücke, wenn das Videostream-Fenster im Vordergrund ihres Bildschirms ist.

Eigene Algorithmen einbinden. Für Sie sind nur die Dateien im Dateipfad *./CV-App/algorithms* relevant. Beispielhaft erstellen Sie im Folgenden den Algorithmus *YourAlgorithm*.

Erstellen Sie eine Datei *your_algorithm.py* und kopieren Sie den Inhalt aus dem Code 1. In der Funktion *__init__(self)* können Sie dauerhafte Variablen definieren. Die Funktion *process(self, img)* verarbeitet das Eingangsbild *img* und gibt es am Ende modifiziert wieder aus (Hinweis: Ein und Ausgangsbild müssen gleich groß sein!). Die Funktion *mouse_callback(self, ...)* reagiert auf Aktionen der Maus, wie zum Beispiel einem Mausklick an der Position x und y. So können Sie mit ihrem Algorithmus interagieren. Sie können sich einige Beispielalgorithmen in dem Ordner *./CV-App/algorithms* zur Veranschaulichung ansehen. Der Algorithmus in *white_balancing.py* veranschaulicht alle Funktionen mit Algorithmen der Klasse *Algorithm*.

Code 1: Eigener Algorithmus in *your_algorithm.py*

```
1  import cv2
2  import numpy as np
3  from . import Algorithm
4
5  class YourAlgorithm(Algorithm):
6      def __init__(self):
7        self.some_persistent_value = "i_am_alwys_existing"
8
9      def process(self, img):
10         print(self.some_persistent_value)
11         return img
12
13     def mouse_callback(self, event, x, y, flags, param):
14         if event == cv2.EVENT_LBUTTONUP:
15             print("A␣Mouse␣click␣happend!␣at␣position", x, y)
```

Um Ihren Algorithmus nun mit einer Aktivierungstaste-Taste zu verlinken, öffnen Sie die Datei *__init__.py*. Am Ende der Datei ist eine Verlinkung der Algorithmen zu bestimmten Tasten zu sehen (siehe Code 8).. In dem Beispiel in Code 8 ist Ihr neuer Algorithmus *YourAlgorithm* importiert und an die Taste 3 verlinkt.

Code 2: Verlinkung der Algorithmen in *__init__.py*

```
1 from .image_to_gray import ImageToGray
2 from .image_to_hue import ImageToHue
3 from .your_algorithm import YourAlgorithm
4
5 algorithms = dict()
6 algorithms["0"] = Algorithm
7 algorithms["1"] = ImageToGray
8 algorithms["2"] = ImageToHue
9 algorithms["3"] = YourAlgorithm
```

Nach Neustart des Programms durch erneute Betätigung der *Run* Funktion, ist Ihr Algorithmus durch betätigen der Taste 3 zugänglich.