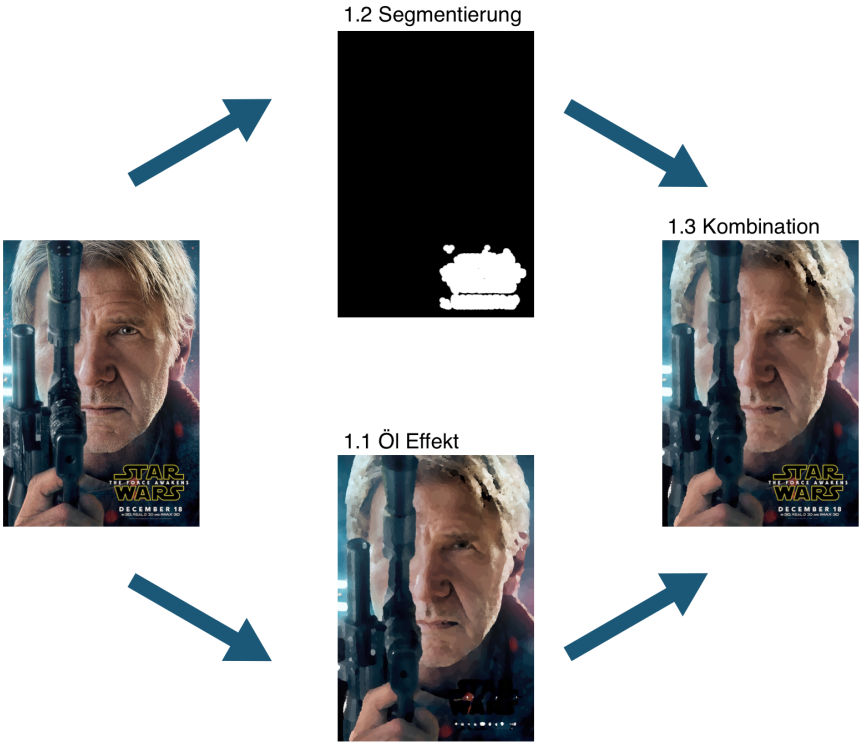


Abbildung 1: Anwendung des Öl-Filter Effekts auf den Bildhintergrund in mehreren Schritten. Zuerst wird der Öl-Effekt auf das gesamte Eingabebild angewendet und eine Segmentierung des Bildes in Vordergrund und Hintergrund erstellt. Mithilfe dieser Segmentierung wird nun der Hintergrund mit dem Öl-Effekt und der Vordergrund vom Eingabebild kombiniert, sodass die Schrift im Vordergrund lesbar bleibt, aber der Hintergrund den Öl-Effekt enthält.

CV/task1a — Selektiver Effekt Filter

Im Zuge der ersten Aufgabenstellung werden einige grundlegende Methoden und Vorgehensweisen der OpenCV-Library vorgestellt. Ziel der Übung ist es, einen *Öl-Filter Effekt* auf ein Eingabebild anzuwenden, wobei der Effekt nur auf den Bildhintergrund und nicht auf Schriftzüge, die sich im Vordergrund befinden anzuwenden ist (siehe Abbildung 1). Um dies zu erreichen, müssen der Vordergrund vom Hintergrund mittels Schwellwert segmentiert und die resultierenden Bilder entsprechend kombiniert werden.

1 Aufgaben

Die Übung ist in drei unterschiedliche Aufgabenstellungen unterteilt, die jeweils in einer eigenen Funktion im Framework zu implementieren sind. Alle **nicht** mit `TODO` gekennzeichneten Funktionen im Framework sollten **nicht** verändert werden. Die drei zu implementierenden Funktionen werden in folgende Aufgabenbereiche unterteilt:

- Anwendung des Öl-Filter Effekts auf das gesamte Eingabebild
- Erstellen einer Maske für den Vordergrund durch Schwellwerten in einem vorgegebenen Bereich
- Kombinieren des gefilterten Bildes mit dem Eingabebild unter Anwendung der zuvor erstellten Maske

Parameter, die für OpenCV-Funktionen gebraucht werden, werden in der Aufgabenstellung explizit angegeben. Darüber hinaus werden Parameter, die den Algorithmus beeinflussen (für jedes Testbild unterschiedlich z.B. `levels`, `threshold`) eingelesen.

1.1 Anwendung des Öl-Filter Effekts auf das gesamte Eingabebild (3 Punkte)

Dieser Teil der Aufgabenstellung ist in der Funktion `getFilteredImage(...)` zu implementieren. Der Öl-Filter Effekt wird mithilfe des im Folgenden erläuterten Algorithmus implementiert. Dazu wird für jedes Pixel dessen Nachbarschaft (siehe Abbildung 2) in einem quadratischen Fenster betrachtet. Die Größe des Fensters wird durch den Parameter `radius` definiert. Pixel mit den Koordinaten $origin \pm radius$ fallen in dieses Fenster. Koordinaten, welche außerhalb des Bildbereiches liegen, können vernachlässigt werden.

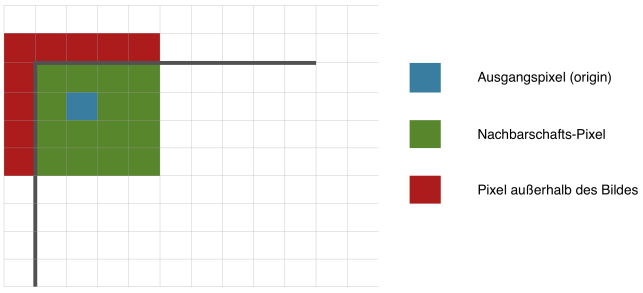


Abbildung 2: Pixel Nachbarschaft für einen gegebenen Ausgangspixel für $radius = 2$. Das Ausgangspixel ist blau hinterlegt, die Pixel die zur Nachbarschaft gehören sind grün. Achten Sie darauf, dass Sie nicht auf Pixel außerhalb des Bildes (rot) zugreifen.

Aus den innerhalb des Fensters liegenden Pixeln (inklusive des in der Mitte liegenden Ausgangspixels), wird nun ein Histogramm der Intensitäten H_I erstellt. Die Anzahl der Intervalle für dieses Histogramm wird durch den Parameter `levels` festgelegt. Für einen Pixel errechnet sich dessen Intensität aus $\frac{I_R+I_G+I_B}{3}$, wobei I_R die Intensität des roten, I_G des grünen und I_B des blauen Kanals darstellen. Der Intervall l, in welchen die Intensität eines Pixels fällt, wird durch folgende Formel ermittelt:

$$l = \left\lfloor \frac{I_R + I_G + I_B}{3.0} \cdot \frac{\texttt{levels}}{255.0} \right\rfloor \tag{1}$$

Dabei erfolgt die Division als Gleitkommazahl, welche mit der Funktion `floor(...)` abzurunden ist. Parallel zum Intensitäts-Histogramm soll der durchschnittliche RGB-Wert für jedes Intensitäts-Level (ein Level entspricht jeweils einem Intervall des Histogramms) berechnet werden. Um dies zu erreichen, werden die drei Arrays S_G, S_R, S_B der Länge $\texttt{levels} + 1$ angelegt. Fällt ein Pixel mit seinem Intensitäts-Wert in das Level l, so werden dessen Rot-, Grün- und Blau-Werte in den Arrays ($S_G[l]$, $S_R[l]$, $S_B[l]$) akkumuliert. Sobald alle Histogramme über die Pixel-Nachbarschaft erstellt wurden, ist für das Intensitäts-Histogramm der Intervall L, in den die meisten Pixel fallen und die Anzahl der Pixel N in diesem Intervall zu ermitteln. Die Werte für Rot (R), Grün (G) und Blau (B) Kanal des Ausgangspixels ergeben sich nun aus folgendem Zusammenhang:

$$N = \max(H_I)$$
$$L = \underset{l}{\operatorname{argmax}}(H_I[l])$$
$$R = \frac{S_R[L]}{N}, \quad G = \frac{S_G[L]}{N}, \quad B = \frac{S_B[L]}{N}$$

Die aus den Divisionen resultierenden Werte sind jeweils das Ergebnis einer Integer-Division. Dieser Vorgang ist nun für jedes Pixel im Bild zu wiederholen. In Abbildung 3 ist ein Beispiel für die Ausgabe dieser Funktion dargestellt.

Hilfreiche OpenCV-Befehle:

- `cv::Mat::at<cv::Vec3b>(...)`

1.2 Erstellen einer Maske für den Bild-Vordergrund (2 Punkte)

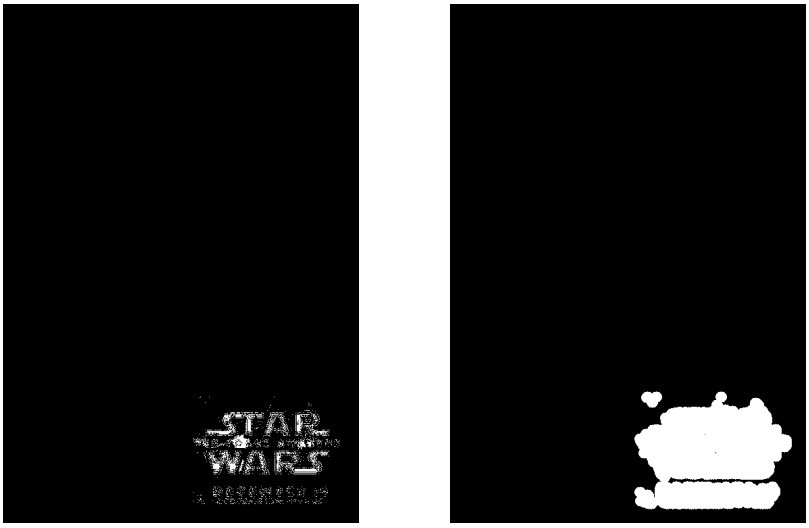
Dieser Teil der Aufgabenstellung ist in der Funktion `getForegroundMask(...)` zu implementieren. Die Segmentierung des Vordergrundes wird durch ein simples Schwellwerten

Abbildung 3: Anwendung des Öl-Filter Effekts auf das gesamte Eingabebild.

des H-Kanals im HSV-Farbraum implementiert. Im HSV-Farbraum setzen sich Farben aus Farbwert (Hue), Sättigung (Saturation) und Hellwert (Value) zusammen. Das Schwellwerten des Farbwerts H erlaubt es Bildregionen, welche einen bestimmten Farbbereich abdecken zu segmentieren. Um die Maske zu erstellen sind zwei Schritte durchzuführen, die in Abbildung 4 veranschaulicht werden. Im ersten Schritt muss das Originalbild zunächst mithilfe der Funktion `cv::cvtColor(...)` in den HSV-Farbraum konvertiert werden. Anschließend wird in dem durch den Parameter `location` vorgegebenen Bereich (bounding box) für jedes Pixel im konvertierten Originalbild O dessen H-Wert mit einem Schwellwert τ (Parameter `threshold`) verglichen und in der Binärmaske B nach folgenden Schema gesetzt:

$$B(x,y) = \begin{cases} 1, & \text{wenn } O(x,y) < \tau \\ 0, & \text{sonst} \end{cases} \tag{2}$$

Dabei bezeichnen x und y die Zeile bzw. Spalte des Bildes. Pixel außerhalb des vorgegebenen Bereiches sind auf den Wert 0 zu setzen. Abbildung 4a veranschaulicht beispielhaft eine Maske nach der Durchführung oben genannter Schritte.



(a) Binäre Maske vor Dilatation.

(b) Binäre Maske nach Dilatation.

Abbildung 4: Erstellung der Maske für den Vordergrund in zwei Schritten.

Um einen möglichst nahtlosen Übergang an den Rändern der Maske zu gewährleisten, wird nun mithilfe einer Dilatation die Maske erweitert. Verwenden Sie für den Typ des Struktur-Elements `CV_MORPH_ELLIPSE`. Die Höhe und Breite des Elements soll auf den Parameter `dilation_size` gesetzt werden. Alle anderen Parameter des Struktur-Elements sollen auf den Standardwerten belassen werden. Das Ergebnis nach durchgeführter Dilatation wird in Abbildung 4b dargestellt.

Hilfreiche OpenCV-Befehle:

- `cv::dilate(...)`
- `cv::getStructuringElement(...)`
- `cv::cvtColor(...)`
- `cv::Mat::copyTo(...)`

1.3 Kombinieren des gefilterten Bildes mit dem Eingabebild (1 Punkt)

Dieser Teil der Aufgabenstellung ist in der Funktion `combineImages(...)` zu implementieren. In dieser Funktion sollen das Originalbild und das gefilterte Bild mithilfe der

Vordergrund-Maske kombiniert werden. Dabei soll für das kombinierte Bild an Positionen mit Vordergrund-Maske logisch 1 das Originalbild, überall sonst das gefilterte Bild herangezogen werden. Ein Pixel des kombinierten Bildes K ergibt sich also aus dem Originalbild O, dem gefilterten Bild F und der Maske nach der Dilatation M durch folgenden Zusammenhang:

$$K(x,y) = \begin{cases} O(x,y), & \text{wenn } M(x,y) = 1 \\ F(x,y), & \text{sonst} \end{cases} \tag{3}$$

Das kombinierte Bild ist in Abbildung 5 veranschaulicht.

Abbildung 5: Ausgabebild nach Anwendung aller Schritte.

2 Ein- und Ausgabeparameter

Die folgenden Parameter in der Konfigurationsdatei bestimmen das Verhalten des Programms (Anwendungsbeispiele finden Sie in den Dateien «`solo.json`», «`avatar.json`»):

- Pfad des Eingabebildes: `input_path`
- Pfad der Ausgabedateien: `output_path`
- Fenstergröße für die Pixelnachbarschaft: `radius`
- Level für den Öl-Effekt Algorithmus: `levels`
- Schwellwert zur Segmentierung des Vordergrunds: `threshold`
- Höhe und Breite des Struktur-Elements für die Dilatation: `dilation_size`
- Position des Bereichs für die Vordergrund-Segmentierung: `label_x`, `label_y`, `label_width`, `label_height`

3 Programmgerüst

Die folgende Funktionalität ist in dem vom ICG zur Verfügung gestellten Programmgerüst bereits implementiert und muss von Ihnen nicht selbst programmiert werden:

- Die JSON-Konfigurationsdatei wird vom Programmgerüst gelesen.
- Lesen der Eingabebilder.
- Lesen des `radius`.
- Lesen der `levels`.
- Lesen des `threshold`.
- Lesen der `dilation_size`.
- Lesen von `label_x`, `label_y`, `label_width`, `label_height`.
- Schreiben der Ausgabebilder.
- Ihre Implementierung ist in `main.cpp` durchzuführen.

Diese Aufgabe ist mit Hilfe von OpenCV[1] zu implementieren. Nutzen Sie die Funktionen, die Ihnen OpenCV zur Verfügung stellt und achten Sie auf die un-

[1] `http://opencv.org`

terschiedlichen Parameter und Bildtypen! Beachten Sie auch, dass die Anordung der Farbkanäle in OpenCV $[B,G,R]$ und nicht $[R,G,B]$ ist. Das C++ Cheatsheet[2] könnte bei dieser Aufgabe sehr hilfreich sein.

[2] http://docs.opencv.org/2.4/opencv_cheatsheet.pdf

4 Abgabe

Die Aufgaben bestehen jeweils aus mehreren Schritten, die zum Teil aufeinander aufbauen, jedoch unabhängig voneinander beurteilt werden. Dadurch ist einerseits eine objektive Beurteilung sichergestellt und andererseits gewährleistet, dass auch bei unvollständiger Lösung der Aufgaben Punkte erzielt werden können.

Wir weisen ausdrücklich darauf hin, dass die Übungsaufgaben von jedem Teilnehmer *eigenständig* gelöst werden müssen. Wenn Quellcode anderen Teilnehmern zugänglich gemacht wird (bewusst oder durch Vernachlässigung eines gewissen Mindestmaßes an Datensicherheit), wird das betreffende Beispiel bei allen Beteiligten mit 0 Punkten bewertet, unabhängig davon, wer den Code ursprünglich erstellt hat. Ebenso ist es nicht zulässig, Code aus dem Internet, aus Büchern oder aus anderen Quellen zu verwenden. Es erfolgt sowohl eine automatische als auch eine manuelle Überprüfung auf Plagiate.

Die Abgabe der Übungsbeispiele und die Termineinteilung für die Abgabegespräche erfolgt über ein Webportal. Die Abgabe erfolgt ausschließlich über das Abgabesystem. Eine Abgabe auf andere Art und Weise (z.B. per Email) wird nicht akzeptiert. Der genaue Abgabeprozess ist im TeachCenter beschrieben.

Die Tests werden automatisch ausgeführt. Das Testsystem ist zusätzlich mit einem Timeout von 7 Minuten versehen. Sollte Ihr Programm innerhalb dieser Zeit nicht beendet werden, wird es vom Testsystem abgebrochen. Überprüfen Sie deshalb bei Ihrer Abgabe unbedingt die Laufzeit Ihres Programms.

Da die abgegebenen Programme halbautomatisch getestet werden, muss die Übergabe der Parameter mit Hilfe von entsprechenden Konfigurationsdateien genauso erfolgen wie bei den einzelnen Beispielen spezifiziert. Insbesondere ist eine interaktive Eingabe von Parametern nicht zulässig. Sollte aufgrund von Änderungen am Konfigurationssystem die Ausführung der abgegebenen Dateien mit den Testdaten fehlschlagen, wird das Beispiel mit 0 Punkten bewertet. Die Konfigurationsdateien liegen im JSON-Format vor, zu deren Auswertung steht Ihnen `rapidjson` zur Verfügung. Die Verwendung ist aus dem Programmgerüst ersichtlich.

Jede Konfigurationsdatei enthält zumindest einen Testfall und dessen Konfiguration. Es ist auch möglich, dass eine Konfigurationsdatei mehrere Testfälle enthält, um gemeinsame Parameter nicht mehrfach in verschiedenen Dateien spezifizieren zu müssen. In manchen Konfigurationsdateien finden sich auch einstellbare Parameter, die in Form eines `select` Feldes vorliegen. Diese sollen die Handhabung der Konfigurationsdateien erleichtern und ein einfaches Umschalten der Modi gewährleisten.

Es steht Ihnen frei, z.B. zu Testzwecken eigene Erweiterungen zu implementieren. Stellen Sie jedoch sicher, dass solche Erweiterungen in Ihrem abgegebenen Code deaktiviert sind,

damit ein Vergleich der abgegebenen Arbeiten mit unserer Referenzimplementierung möglich ist.

Die Programmgerüste, die zur Verfügung gestellt werden, sind unmittelbar aus unserer Referenzimplementierung abgeleitet, indem nur jene Teile entfernt wurden, die dem Inhalt der Übung entsprechen. Die Verwendung dieser Gerüste ist nicht zwingend, aber Sie ersparen sich sehr viel Arbeit, wenn Sie davon Gebrauch machen.