Hochschule Konstanz
Technik, Wirtschaft und Gestaltung

Signale, Systeme und Sensoren

KALIBRIERUNG VON DIGITALKAMERAS

Kiattipoom Pensuwan, Thanh Son Dang

Konstanz, 21. November 2018

Zusammenfassung

Thema:	KALIBRIERUNG VON DIGITALKAMERAS	
Autoren:	Kiattipoom Pensuwan	ki851pen@htwg-konstanz.de
	Thanh Son Dang	th851dan@htwg-konstanz.de
Betreuer:	Prof. Dr. Matthias O. Franz	mfranz@htwg-konstanz.de
	Jürgen Keppler	juergen.keppler@htwg-konstanz.de
	Simon Christofzik	si241chr@htwg-konstanz.de

Zur Überprüfung der Qualität der Digitalkamera, nimmt man einen stufenförmigen Grauwertverlauf auf, die innerhalb jeder Stufe gleichen Wert haben sollte. Mithilfe des Python-Pakets OpenCV kann man die Belichtungsparameter der Kamera ändern und das Bild im verlustfreien Format png aufnehmen und die Informationen(RGB-Werte) von Bildpunkte auslesen. Damit kann man mit Dunkelbild und Weißbild Bildfehler und Sensorrauschen suchen und beseitigen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1

Aufnahme und Analyse eines Grauwertkeiles

1.1 Fragestellung, Messprinzip, Aufbau, Messmittel

Ein Python Skript schreiben (s. Anhang A.1.1), welches das Objekt mithilfe der OpenCV aufnehmen. Die Position und die Distanz zwischen der Digitalkamera (in diesem Fall Webcam) und das Grauwertkeil so einzustellen, dass sich die möglichst kompletten Grauwertverlauf in das Bild befindet. Dabei sollte die Grauwertstufen parallel zu den Bildrändern verlaufen. Mit OpenCV die Belichtungsparameter einzustellen, so dass den weißen Bereich des Bildes keinen Überlauf hat (255 nicht überschreiten), da die Informationen beim Überlauf verloren gehen. Diese Einstellung (Distanz und Belichtungsparameter) benutzt man für alle durchzuführende Versuche. Das aufgenommene Bild(Farbbild) in ein Grauwertbild mit cv2.cvtColor() umwandeln. Der Grauwertverlauf in 5 Grauwertstufe aufteilen und als eigene Bilder speichern. Von jeder Grauwertstufe sind der Mittelwert und Standardabweichung zu ermitteln.(s. Anhang A.1.2)

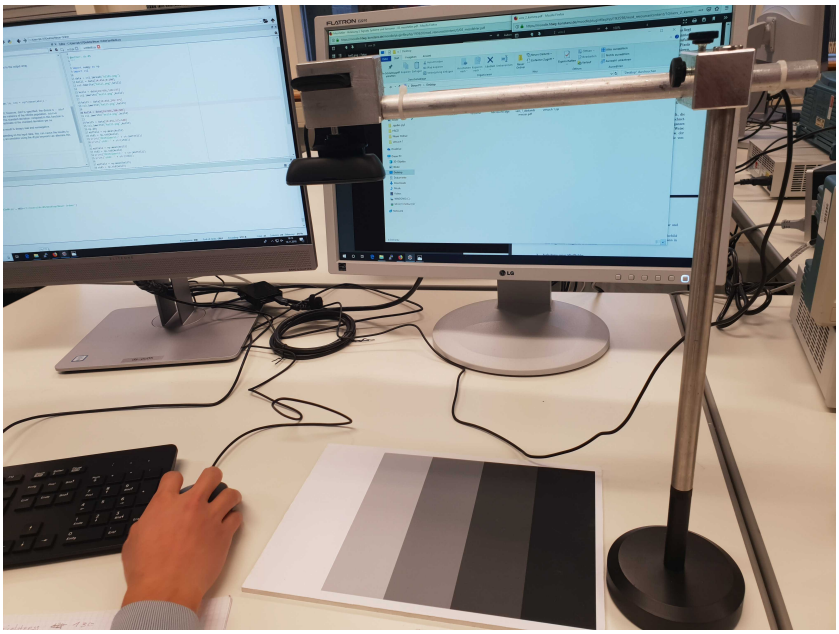

Abbildung 1.1: Versuchaufbau

Belichtungsparameter	
framewidth	640
frameheight	480
brightness	130
contrast	30
saturation	64
gain	0
exposure	-4
white balance	4980

Tabelle 1.1: Kameraeinstellung

1.2 Messwerte

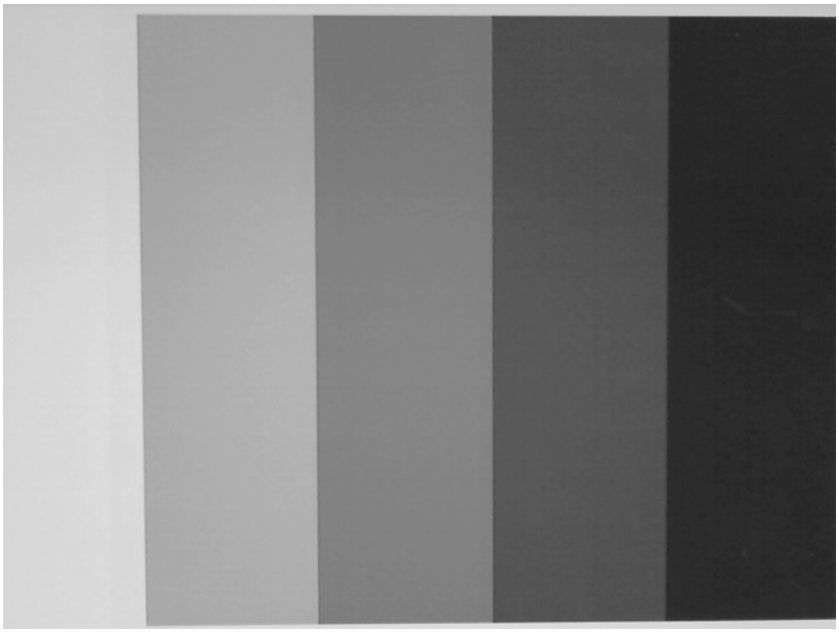

Abbildung 1.2: Das umgewandelte Grauwertkeilbild

1.3 Auswertung

Den Code für die Aufteilung und Berechnung bitte den Anhang A.1.2 sehen.

Mittelwert:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Standardabweichung:

$$s = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (\bar{x} - x_i)^2}$$

	Mittelwert	Standardabweichung
Grauwertstufe 1	212.504769	5.643870
Grauwertstufe 2	170.716981	6.199961
Grauwertstufe 3	128.680235	4.864795
Grauwertstufe 4	83.420426	4.632894
Grauwertstufe 5	41.393944	2.515164

Tabelle 1.2: Mittelwert und Standardabweichung von Grauwertstufen (Grauwertstufe 1 die hellste bis Grauwertstufe 5 die dunkelste)

1.4 Interpretation

Das OpenCV-Paket liest das Bild als Rot-Grün-Blau-Werte(RGB) von jedem Pixel unter eine Matrix ein, deswegen umso dunkler das Keil ist, umso geringer sind die Werte(d.h. (0,0,0) ist komplett schwarz bzw. (255,255,255) komplett weiß) und da das Farbbild in Graubild umgewandelt wurde sind die RGB-Werte jedes Pixels immer gleich. Es folgt darauf, dass die Mittelwerte absteigen, je nach der Helligkeit der Stufen.

Aus den berechneten Standardabweichungen kann man interpretieren, dass die mit größeren RGB-Werten Grauwertstufe mehr variieren als die mit kleineren. Bei Grauwertstufe 1 haben wir kleineres Unterbild(weniger Pixel) als anderen Stufe, deshalb zu kleinere Standardabweichung führt. Die dunklere Grauwertstufen haben auch kleineren Standardabweichung, weil der Unterschied von kleineren Wert kleiner sind als größeren Wert.

2

Aufnahme eines Dunkelbildes

2.1 Fragestellung, Messprinzip, Aufbau, Messmittel

Um das Schwarzbild aufzunehmen, sollte das Objektiv der Kamera abgedeckt werden, sodass das Bild komplett schwarz wird. Die Belichtungsparameter und Distanz sind gleich wie das erste Versuch einzustellen.

Um das thermische Ausleserauschen zu eliminieren, werden 10 Bilder aufgenommen und daraus berechnet ein pixelweisen Mittelwert(ein ganzes Bild von Mittelwert). Es bleibt nur noch der Offset. bzw der Dunkelstrom jedes Pixels übrig. Das Mittelwertbild ist das Dunkelbild.

Das Eingabebild wird durch Subtraktion von Dunkelbild korrigiert.

2.2 Messwerte

Abbildung 2.1: Ein aufgenommenes Beispielschwarzbild

Abbildung 2.2: Kontrastmaximierte Darstellung des Dunkelbildes

2.3 Auswertung

Um das kontramaximierte Bild darzustellen, verwendet man diese Formel für jeden Pixel:

$$\frac{pixel - min}{max - min} \cdot 255$$

min ist der minimalwert des Arrays/Bildes, max der maximale.

Das Dunkelbild mit einem Pythonskript einlesen, von dem Eingabebild subtrahiert und bekommt man das Bild:

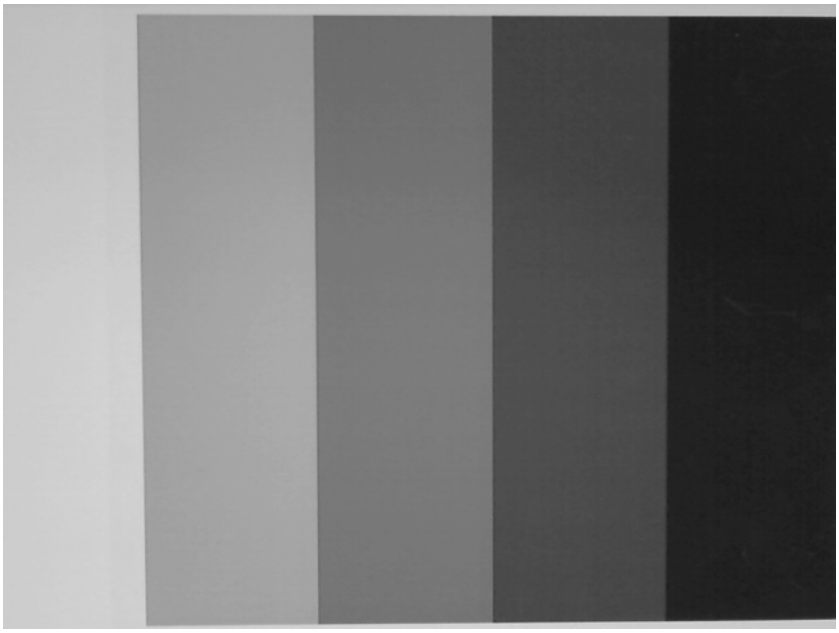

Abbildung 2.3: Korrigiertes Eingabebild nach dem Subtrahieren

Den Code davon bitte den Anhang A.1.3 sehen.

2.4 Interpretation

Das berechnete Dunkelbild hat die Werte ungefähr 10 d.h. dass der Offset der von Fertigungstoleranzen und Wärmezufuhr entsteht beträgt 10. Da das Eingabebild von Dunkelbild abgezogen wurde, sieht das korrigierte Bild dunkler aus.

3

Aufnahme eines Weißbildes

3.1 Fragestellung, Messprinzip, Aufbau, Messmittel

Ein leeres weißes Blatt Papier unter dieselbe Bedingungen wie vorherige Versuche einstellen und 10 Weißbilder aufnehmen zur Elimination des thermischen Rauschens, das Mittelwertbild wie vorher berechnen. Danach muss man noch das Dunkelbild von dem Mittelwertbild subtrahieren. Das resultierende Weißbild sollte normiert werden, sodass sein Mittelwert 1 ist, damit das Eingangbild durch Dividieren weiter korrigiert werden könnte.

3.2 Messwerte

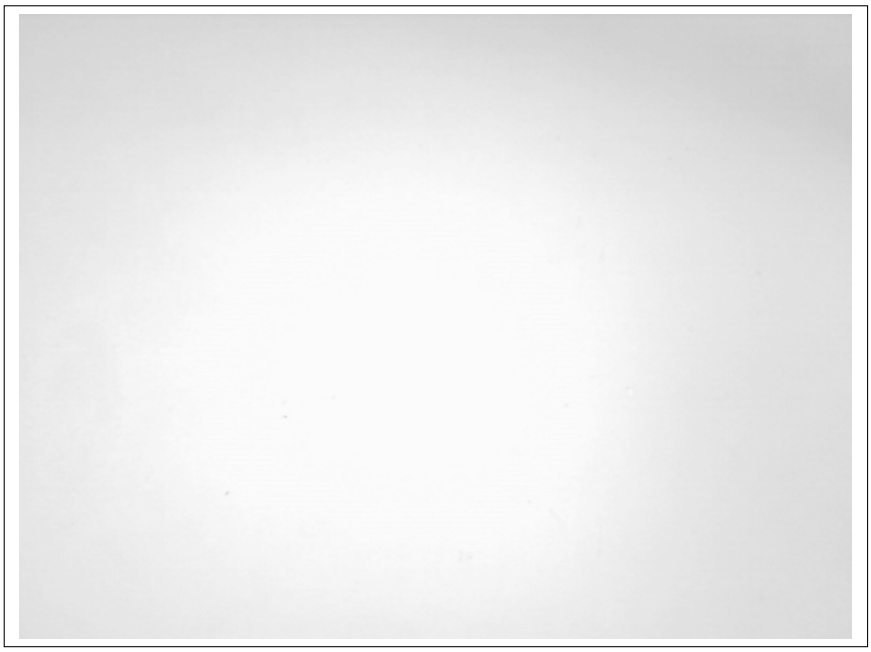

Abbildung 3.1: Ein aufgenommenes Beispielweißbild

Abbildung 3.2: Kontrastmaximierte Darstellung des Weißbildes

3.3 Auswertung

Um das kontramaximierte Bild darzustellen, verwendet man diese Formel für jeden Pixel:

$$\frac{pixel}{max - min} \cdot 255$$

min ist der minimalwert des Arrays/Bildes, max der maximale.

Da man die Korrektur mit dem Schwarzbild gemacht hat, ist die Formel anders als die von dem 2. Versuch (pixel - min wird pixel).

Um das Mittelwertbild zu normieren, verwendet man die Formel pixelweise:

$$normW = \frac{korrigiertes Weissbild}{Mittelwert\ von dem korrigierten Weissbild}$$

dadurch bekommt ein neues Bild mit Mittelwert 1.

Abbildung 3.3: Normiertes Weißbild

Den Code davon bitte den Anhang A.1.3 sehen.

3.4 Interpretation

Da der Mittelwert von dem normierten Bild 1 ist, sollte das resultierte Bild fast komplett schwarz sein.

Normalerweise hat das aufgenommene Bild immer eine Vignettierung(dunklere Ränder),die schwer zu erkennen(weil das Objekt eigene Farbwerte besitzt) und aufgrund der unterschiedliche Helligkeitsübertragung von der Optik der Kamera entstand. Mit Weißbild entdeckt man die Vignettierung leichter(weiße Farbe hat größere RGB-Werte -> Unterschied zwischen Ränder und andere Teile des Bildes besser erkennen).

4

Pixelfehler

4.1 Fragestellung, Messprinzip, Aufbau, Messmittel

Dunkelbild auf dem Bildschirm auf stuck und hot pixels und Weißbild auf dead pixels überprüfen. Das Bild des Grauwertkeils mit gleicher Methode im Versuch 3 korrigieren. Korrigiertes Bild in 5 Stufen wie im Versuch 1 aufteilen und überprüft ob es da Verbesserung gibt.

4.2 Messwerte

Abbildung 4.1: Schwarzbild ohne gefundene hot und stuck pixels

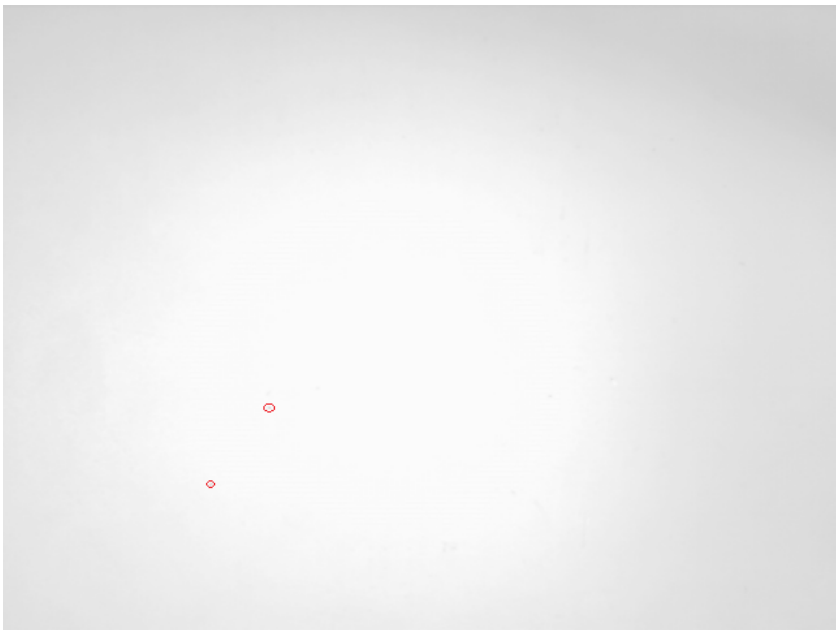

Abbildung 4.2: Weißbild mit dead pixels Markierung

4.3 Auswertung

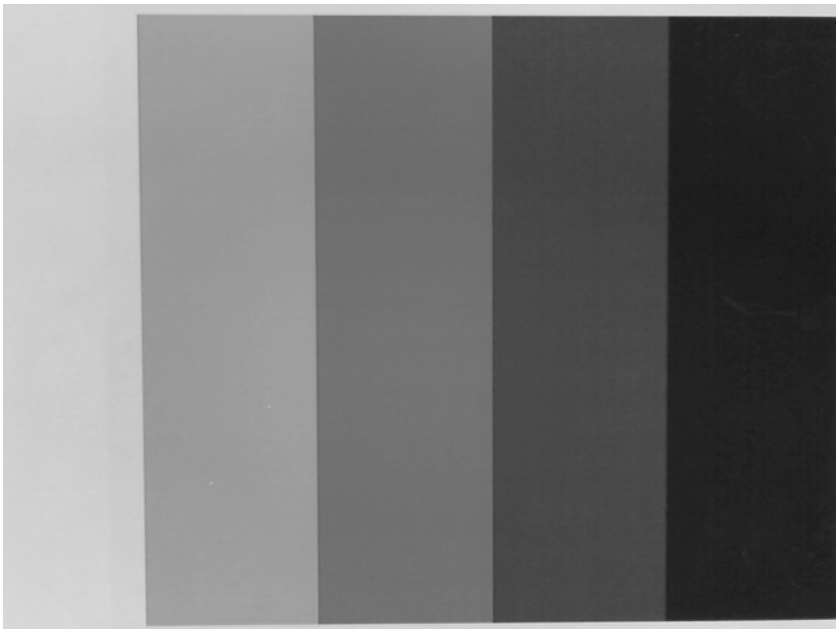

Abbildung 4.3: Korrigiertes Bild

	Mittelwert	Standardabweichung
Grauwertstufe 1	205.782106	3.502565
Grauwertstufe 2	155.267629	2.611407
Grauwertstufe 3	113.188105	2.333321
Grauwertstufe 4	72.236704	2.395081
Grauwertstufe 5	33.400926	2.649814

Tabelle 4.1: Mittelwert und Standardabweichung von Grauwertstufen (Grauwertstufe 1 die hellste bis Grauwertstufe 5 die dunkelste)

4.4 Interpretation

Bei der Markierung der hot und stuck pixels konnten gar keine Weißpunkte mit normalen Augen entdeckt werden. Das könnte bedeuten, dass die Qualität des Bildsensors noch in guten Zustand ist.

Bei der Markierung der dead pixels sieht man 2 kleine Dunkelbereiche, von der wir behaupten können, dass es nicht nur aus dem Sensor kommt sondern auch wegen der Oberfläche von dem Objekt(nicht ganz homogene Fläche).

Das korrigierte Bild ist dunkler geworden(da Mittelwert geringer ist). Die geringere Standardabweichung weist darauf hin, dass die Unterschied zwischen Pixels in jeder Grauwertstufe kleiner wird. D.h. wir haben eine bessere Bildqualität bekommen.

Anhang

A.1 Quellcode

A.1.1 Quellcode Bild aufnehmen

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)
cap.set(11,30)
cap.set(10,130)
cap.set(14,0)
cap.set(15,-4)
cap.set(17,4980)

print("framewidth:" + str(cap.get(3)))
print("frameheight:" + str(cap.get(4)))
print("----------")
print("brightness:" + str(cap.get(10)))
print("contrast:" + str(cap.get(11)))
print("saturation:" + str(cap.get(12)))
print("----------")
print("gain:" + str(cap.get(14)))
print("exposure:" + str(cap.get(15)))
print("----------")
print("white_balance:" + str(cap.get(17)))


while(True):
    ret, frame = cap.read()
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
    cv2.imshow('frame', gray)
    if cv2.waitKey(1) & 0xFF == ord('q'):
```

```
        cv2.imwrite('bildweiß10.png',gray)
        print(np.min(gray),np.max(gray))
        break;

cap.release()
cv2.destroyAllWindows()
```

A.1.2 Quellcode für Bildunterteilung und der Berechnung von Mittelwert und Standardabweichung

```
import numpy as np
import cv2

data = cv2.imread('bild1.png')
print(data)
keil1 = data[24:456,0:100]
cv2.imwrite("keil1.png",keil1)

keil2 = data[24:456,110:235]
cv2.imwrite("keil2.png",keil2)

keil3 = data[24:456,245:375]
cv2.imwrite("keil3.png",keil3)

keil4 = data[24:456,380:505]
cv2.imwrite("keil4.png",keil4)

keil5 = data[24:456,515:640]
cv2.imwrite("keil5.png",keil5)

data = cv2.imread('bild1_korrekt.png')
keil1_k = data[24:456,0:100]
cv2.imwrite("keil1_korrekt.png",keil1_k)

keil2_k = data[24:456,110:235]
cv2.imwrite("keil2_korrekt.png",keil2_k)

keil3_k = data[24:456,245:375]
cv2.imwrite("keil3_korrekt.png",keil3_k)

keil4_k = data[24:456,380:505]
```

```
cv2.imwrite("keil4_korrekt.png",keil4_k)

keil5_k = data[24:456,515:640]
cv2.imwrite("keil5_korrekt.png",keil5_k)

def MuS(keil, n):
    mittel = np.mean(keil)
    std = np.std(keil)
    print('Mittelwert'+ str(n) +':' + str(mittel))
    print('std'+ str(n) +':' + str(std))
MuS(keil1, 1)
MuS(keil2, 2)
MuS(keil3, 3)
MuS(keil4, 4)
MuS(keil5, 5)
print('korrigierte Werte:----------')
MuS(keil1_k, 1)
MuS(keil2_k, 2)
MuS(keil3_k, 3)
MuS(keil4_k, 4)
MuS(keil5_k, 5)
```

A.1.3 Quellcode zum Einlesen, Auslesen, Korrigieren des Eingangbild mithilfe des Weiß-und Schwarzbildes

```
import numpy as np
import cv2

bild1 = cv2.imread('bild1.png')

def mw(bild): #Bildern einlesen und Mittelwert für jede Pixel bilden
    data = []
    datad = []
    for i in range(1,11):
        data.append(cv2.imread(str(bild) + str(i) + '.png'))
    for i in data:
        datad.append(i.astype(float))
    mw = np.zeros(datad[0].shape)
    for i in range(0,480):
        for j in range(0,640):
            for k in range(0,10):
```

```
            mw[i,j] += datad[k][i,j]
    for i in range(0,480):
        for j in range(0,640):
            mw[i,j] /= 10
    return mw

def konmaxb(pixel):
    return ((pixel − np.min(pixel)) / (np.max(pixel) − np.min(pixel))) ∗ 255

def konmax(pixel):
    return (pixel / (np.max(pixel) − np.min(pixel))) ∗ 255

mwb = mw('bildschwarz') #Mittelwert von Dunkelbildern
cv2.imwrite("mBlack.png", mwb)
cv2.imwrite("kontrastmaxBlack.png", konmaxb(mwb)) #Ausgabe kontrastmaximiertes Dunkelbildes

korBv2 = bild1 − mwb
cv2.imwrite("korrigiertes Bild v2.png", korBv2)

mww = mw('bildweiss') #Mittelwert von Weißbildern
cv2.imwrite("mWhite.png", mww)
cv2.imwrite("kontrastmaxWhite.png", konmax(mww−mwb)) #Ausgabe kontrastmaximiertes Weißbildes
normW = (mww−mwb)/np.mean(mww−mwb) # nomiertes Weißbild
cv2.imwrite("normW.png", normW)
korW = korBv2 / normW
cv2.imwrite("Bild1_korrekt.png", korW)
```