Grafika rastrowa i fotografia
Instrukcja do laboratorium 1

inż. Andrii Shekhovtsov

Październik 2021

1 Zasady oceniania

Oceny będą wystawiane na podstawie sprawozdań wykonywanych z każdej instrukcji. W treści instrukcji są podane **Zadania** wraz przypisaną **[liczbą punktów]**. W tej instrukcji jest **6** zadań, łącznie za **10** punktów. Ocena zależy od liczby punków za wykonane zadania.

Liczba punków	**Ocena**
4	3,0
5	3,5
$6-7$	4,0
$8-9$	4,5
10	5,0

Termin oddania sprawozdania jest ustawiony w systemie moodle. W przypadku nie oddania zadania w terminie, uzyskana ocena będzie zmniejszana o 0,5 za każdy zaczęty tydzień opóźnienia.

W przypadku pracy w środowisku Jupyter Notebook proszę o przesłanie tylko wygenerowanego pliku pdf.

W przypadku pracy w innych środowiskach proszę o przesłanie sprawozdania w formacie pdf zawierającego wykonane zadania (w zależności od zadania to mogą być odpowiedzi na pytania, kod funkcji, obrazek (screenshot) i wyjaśnienie), **a także całości kodu wykonanego w ramach przygotowania tego sprawozdania**.

2 Materiał pomocniczy

2.1 Narzędzia

2.1.1 Bibleoteki Python

Do pracy na laboratoriach używamy języka programowania Python 3, potrzebne będą dwie dodatkowe biblioteki, które można zainstalować wykonując następujące polecenie w konsoli:

```
pip install numpy matplotlib
```

2.1.2 Jupyter Notebook

Do pracy zalecane jest środowisko Jupyter Notebook, które umożliwia generowanie pdf dokumentów zawierających kod wraz z wynikiem działania tego kodu.

Zainstalować środowisko można za pomocą `pip`:

```
pip install notebook
```

a uruchomić za pomocą `jupyter notebook` (musi otworzyć się w przeglądarce).

2.1.3 Generowanie pdf z notebooka

Sposób 1. Ten sposób nie wymaga instalowania żadnych dodatkowych rzeczy: Klikamy File -> Download as -> HTML (.html). Otwieramy HTML za pomocą przeglądarki, i skrótem Ctrl+P zapisujemy plik HTML do pdf. **Proszę nie przesyłać plików html, akceptowane są tylko pliki pdf!**.

Sposób 2. Wymaga zainstalowania dodatkowych narzędzi: `nbconvert` (`pip install nbconvert`), oraz środowiska do kompilacji LaTeX(Windows: https://miktex.org/). Po zainstalowaniu ma działać opcja PDF via LaTeX (.pdf) w menu Download as.

2.2 Szybkie wprowadzenie w Python

```
# Komentarz

a = 4 # Zmienna
print(a) # Wypisać wartość zmiennej na ekran

b = 5.0 # Zmienna typu float
print(a + b) # Pokazać wynik dodawania

# Jak widać nie definiujemy typów zmiennych
# Możemy przypisać do wcześniej utworzonej zmiennej co innego
a = "To jest string"
# Można też używać pojedynczy cudzysłów

# Warunek
if b == 5:
    print('b to 5')
else:
    print('b to nie 5')
# Część z else nie jest obowiązkowa

# Można też podawać kilka warunków po kolei:
if b == 5:
    print('b to 5')
elif b == 4:
    print('b to 4')
else:
    print('b to nie 4 i nie 5')

# Pętla for
for i in range(5):
    print(i)

# Tworzenie i wywołanie funkcji
def power2(n):
    return n ** 2

# Przypisanie wyniku funkcji i jego wyświetlenie
c = power2(b)
print(c)
```

Dobra strona dla przypomnienia języka https://learnxinyminutes.com/docs/python/.

2.3 Przykład użycia `plt.subplot`

Funkcje `plt.subplot` pozwala na ułożenie wyświetlancyh wykresów (w naszym przypadku - obrazków) w postaci siatki. Funkcje przyjmuję następujące argumenty: `plt.subplot(h, w, n)`, gdzie `h` i `w` to wysokość i szerokość siatki (w liczbie wykresów), a `n` mówi do której kratki siatki ma być wpasowany kolejny wykres/obrazek.

```
plt.subplot(1, 2, 1)
plt.title('Obrazek')
plt.imshow(img, cmap='gray')

plt.subplot(1, 2, 2)
plt.title(f'Też obrazek')
plt.imshow(img, cmap='gray')

plt.show()
```

3 Instrukcja

1. Zaimportować biblioteki.

```
import numpy as np
import matplotlib.pyplot as plt
```

2. Znaleźć, pobrać i wczytać dowolny obrazek z tęczą (albo inny różnokolorowy obrazek, najlepiej żeby rozmiar tego obrazku nie był zbyt duży).

```
img = plt.imread('rainbow.png')
```

3. Wypisać typ i rozmiar wczytanego obrazku:

```
print(img.shape, img.dtype)
```

[0.5 pkt.] Zadanie 1: Co oznaczają liczby zawarte w `shape`?
[0.5 pkt.] Zadanie 2: Jaki mamy typ zmiennych? Jaki zakres wartości posiada w przypadku obrazku?

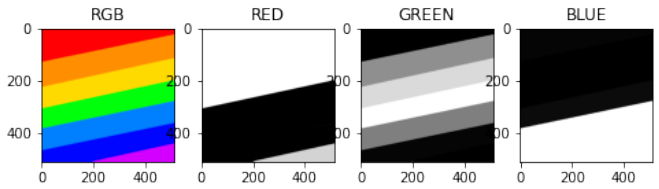
4. Wyświetlić obrazek.

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

5. Wyświetlić tylko wartości koloru czerwonego obrazku (w odcieniach szarości).

```
plt.imshow(img[:,:,0], cmap='gray')
plt.show()
```

[2 pkt.] Zadanie 3: Używając poleceń `plt.subplot` i `plt.title` wyświetlić składowe RGB tak jak pokazano na Rysunku 1 (w przykładzie pokazany też oryginalny obrazek, lecz nie jest jego dodanie obowiązkowe).



Rysunek 1: Przykład do Pytania.

6. Zaimplementować wzór (1), który zamienia obrazek na obrazek w odcieniach szarości.

$$Y = 0.299R + 0.587G + 0.114B, \tag{1}$$

gdzie R - składowa czerwona piksela, G - składowa zielona, B - składowa niebieska, a Y jest wartością wynikową (wynikowy obraz będzie miał tylko jedną warstwę). Wzór musiałby być stosowany dla poszczególnych pikseli, jednak, korzystając z możliwości biblioteki `numpy` (a obrazek właśnie jest przechowywany w tablice, pochodzącej z tej biblioteki) można zaimplementować ten wzór nie używając pętli.

[1 pkt.] Zadanie 4: Zaimplementować wzór (1) w postaci funkcji i bez użycia pętli.

7. Znaleźć, pobrać i wczytać dowolny inny obrazek, najlepiej jakieś zdjęcie.

8. Zamienić wczytany obrazek na odcienie szarości za pomocą poprzednio utworzonej funkcji.

9. Zaimplementować wzory (2) i (3), służące odpowiednio dla obliczenia jasności i kontrastu obrazka w odcieniach szarości.

$$J = \frac{1}{M \cdot N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} I(i,j) \tag{2}$$

$$K = \sqrt{\frac{1}{M \cdot N} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (I(i,j) - J)^2} \tag{3}$$

W powyższych wzorach zakładamy że mamy do czynienia z obrazkiem I o wymiarach M na N, a $I(i,j)$ oznacza wartość piksela o współrzędnych i, j. Te wzory mogą zostać zaimplementowane używając pętli, lub korzystając z możliwości biblioteki `numpy`, a mianowicie używając funkcji `np.mean`, `np.sum`, `np.sqrt`.

[2 pkt.] Zadanie 5: Zaimplementować wzory (2) i (3) w postaci funkcji, bez użycia pętli.

10. Wyliczyć jasność i kontrast dla załadowanego obrazka, wypisać na ekran.

[4 pkt.] Zadanie 6: Zbadać w jaki sposób na obrazek wpływają operacje globalne, takie jak:

- dodawanie stałej do obrazu;
- mnożenie obrazu przez stałą;
- potęgowanie obrazu;
- pierwiastkowanie obrazu.

Za pomocą poleceń `plt.subplot` oraz `plt.title` wyświetlić dla każdej badanej operacji oryginalny obrazek i obok obrazek po zastosowaniu operacji, tak jak pokazano na Rysunku 2. W sprawozdaniu proszę też opisać w jaki sposób badane operacje zmieniają jasność i kontrast obrazku.

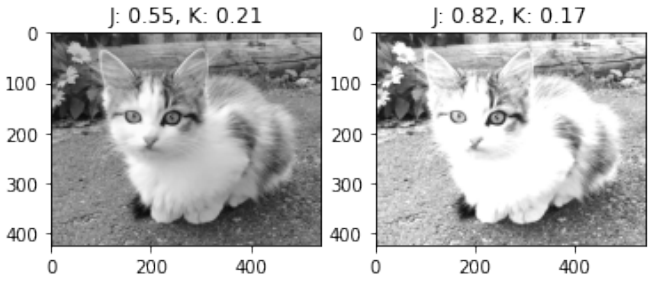
Uwaga! W przypadku niektórych operacji możliwe jest przekroczenie zakresu wartości obrazka. Żeby tego uniknąć należy zastosować następujący kod:

```
img[img > 1] = 1 # Dla wartości typu float
img[img > 255] = 255 # Dla wartości typu int
```

Proszę też zwrócić uwagę na to że operacja przypisania (e.g. `img2 = img`) tworzy odniesienie do obrazu `img` i nie wykonuje kopiowania, a zatem wszystkie zmiany zastosowane do zmiennej `img2` będą też zastosowane do zmiennej `img`. Żeby uniknąć tego, należy wykonywać przypisanie w taki sposób: `img2 = img.copy()`. Spowoduje to utworzenie kopii obrazku `img`.

Podpowiedź: Do wyświetlenia nagłówka można użyć poniższe polecenie. Zakładamy że funkcje `brightness()` i `contrast()` wyliczają odpowiednio jasność i kontrast podanego obrazka.

```
plt.title(f'J: {brightness(img):0.2f}, K: {contrast(img):0.2f}')
```



Rysunek 2: Przykład pokazujący efekt dodawania stałej do obrazku.