Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych

Rozpoznawanie Obrazów

Semestr VII, Informatyka

mgr inż. Marcin Skobel

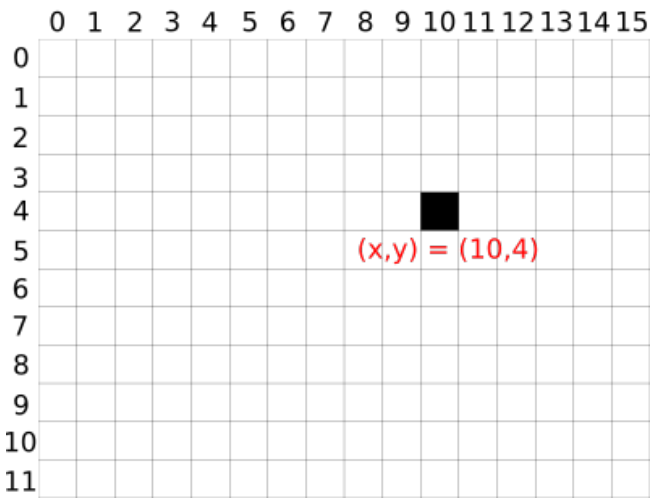
2020

Laboratorium nr 2: Kadrowanie i operacje afiniczne.

I. Zagadnienia teoretyczne

Wstęp

Obraz cyfrowy stanowi prostokątny zbiór pikseli. Położenie poszczególnych pikseli w przestrzeni obrazu może być definiowane przy zastosowaniu współrzędnych x i y. Innymi słowy obraz składa się z pikseli ułożonych w kolumny oraz wiersze, które tworzą układ przypominający macierz. Stosując układ współrzędnych możemy jednoznacznie określić położenie poszczególnych pikseli:



Początek układu współrzędnych obrazu cyfrowego najczęściej definiowany jest w lewym górnym rogu. Współrzędne początkowe obrazka przyjmują wartość (0,0) oznacza to, że kolumny i wiersze są numerowane od liczby 0. Biblioteka, która będzie przydatna w wykonaniu zadania nosi nazwę Pillow (`https://pillow.readthedocs.io/en/stable/`).

Kadrowanie obrazu

Pierwszym zagadnieniem poruszanym w bieżącym laboratorium jest kadrowanie fragmentu obrazu cyfrowego. Kadrowanie polega na wycięciu fragmentu obrazu pierwotnego. Do poprawnego wykonania zadania kadrowania niezbędne jest posiadania kilku podstawowych informacji. Informacja pierwsza dotyczy rozmiaru obrazka wejściowego czyli liczby kolumn i wierszy w całym obrazku. Druga informacja to współrzędne początku kadru, natomiast trzecia informacja to wysokość i szerokość kadru.

Przeskalowanie

Zmiana skali obrazu cyfrowego jest dość powszechnie używana. Redukcja rozdzielczości może znaleźć zastosowanie np.: w redukcji wielkości pliku obrazu do wymagań portali i stron internetowych. Z teoretycznego punktu widzenia przeskalowanie definiujemy przy użyciu wzoru:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} u & 0 & 0 \\ 0 & v & 0 \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}, \tag{1}$$

gdzie u odnosi się do skali rozciągnięcia w kierunku poziomym, natomiast v w kierunku pionowym. Można zatem zauważyć, że przeskalowanie w obu kierunkach nie musi być jednakowe. W zależności od potrzeby można obraz rozciągnąć (spłaszczyć) inaczej w pionie, a inaczej w poziomie.

Translacja

Kolejnym przekształceniem afinicznym jakie należy poznać jest translacja (przesunięcie) obrazu. Przekształcenie to polega na jednakowym przesunięciu wszystkich pikseli na obrazie w ustalonym kierunku. Do przesunięcia niezbędny będzie zatem wektor przesunięcia. Translacja obrazu może zostać przedstawiona przy użyciu wzoru:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 1 & 0 & m \\ 0 & 0 & n \end{bmatrix} \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix}, \tag{2}$$

gdzie m to wartość przesunięcia w kierunku poziomym, natomiast n to wartość przesunięcia w kierunku pionowym.

Obrót

Obrót (rotacja) obrazu stanowi kolejny rodzaj podstawowej geometrycznej transformacji obrazu. Obracanie obrazu wykonywane jest względem środka obrazu. Transformacja obrazu przy użyciu rotacji wymaga zdefiniowania macierzy obrotu:

$$R(\theta) = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \tag{3}$$

Po zdefiniowaniu macierzy obrotu możemy jej użyć do rotacji wektorów kolumnowych:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{4}$$

Z tego względu poszczególne współrzędne (x', y') można zapisać przy użyciu wzorów:

$$\begin{cases} x' = x\ cos\theta & - & y\ sin\theta, \\ y' = x\ sin\theta & + & y\ cos\theta \end{cases} \tag{5}$$

II. Przykład praktyczny

Wstępna konfiguracja

Przykład praktyczny tym razem rozpoczynamy od zaimportowania biblioteki Pillow i matplotlib.pyplot do Colaba oraz zamontowaniu Dysku Google:

```
import PIL
import matplotlib.pyplot as plt
from google.colab import drive
drive.mount('/content/drive')
```

W kolejnym oknie kodu można dokonać podstawowej konfiguracji obrazu testowego:

```
testImagePath = 'drive/My Drive/Colab Notebooks/Images/'
testImage = PIL.Image.open(testImagePath+'Test.tif')
width, height = testImage.size
print(width, height)
imgplot = plt.imshow(testImage)
```

Pierwsza zmienna (*testImagePath*) jest ścieżką do folderu zawierającego obrazy. Zmienna *testImage* przechowuje obraz zaimportowany przy użyciu biblioteki *PIL.Image*. Możemy bardzo łatwo uzyskać zmienne dotyczące szerokości i wysokości obrazu (*width, height*) wpisując polecenie *testImage.size*. Wymiary obrazka można w tym momencie wyświetlić funkcją *print* a następnie wyświetlić obraz przy użyciu biblioteki *matplotlib.pyplot*.

Kadrowanie

Obraz testowy został wczytany przy użyciu biblioteki *PIL.Image* zatem możemy od razu rozpocząć zadanie kadrowania. W tym celu należy zdefiniować współrzędne okna kadrowania (x1, y1, x2, y2). Współrzędne x1 oraz y1 oznaczają lewy górny róg okna kadrowania natomiast współrzędne x2 oraz y2 oznaczają prawy dolny róg tego okna. Oczywiście uzyskany obraz wynikowy można wyświetlić przy użyciu biblioteki *matplotlib.pyplot*:

```
cropCoordinates = (500, 150, 1012, 662)
croppedImage = testImage.crop(cropCoordinates)
imgplot = plt.imshow(croppedImage)
```

Przeskalowanie

W kolejnym oknie kodu warto przetestować działanie funkcji przeskalowania:

```
resizedImage = testImage.resize((1800,500))
imgplot = plt.imshow(resizedImage)
```

W tym przykładzie mamy do czynienia z dwiema wartościami. Pierwsza z nich (1800) oznacza wynikową szerokość obrazu po przeskalowaniu natomiast druga wartość (500) oznacza wysokość obrazu po przeskalowaniu.

Translacja

Nieco bardziej skomplikowanym zagadnieniem z punktu widzenia kodu Python jest translacja obrazu. Przed wykonaniem translacji należy wprowadzić 6 zmiennych pomocniczych (a, b, c, d, e, f)

```
a = 1
b = 0
c = -300
d = 0
e = 1
f = -450
translatedImage = testImage.transform(testImage.size, PIL.Image.AFFINE, (a, b, c, d, e, f))
imgplot = plt.imshow(translatedImage)
```

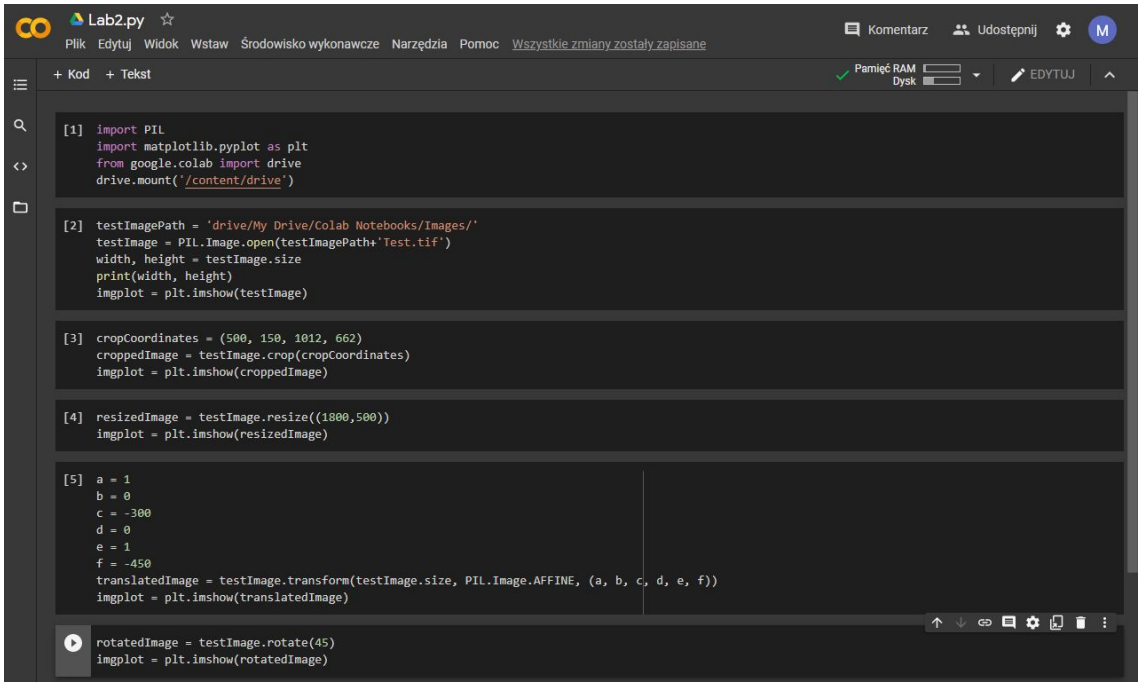
Do wykonania translacji obrazu przy użyciu biblioteki *PIL.Image.AFFINE* najważniejsza jest zmienna *c* oraz zmienna *f*. Zmienna *c* definiuje przesunięcie w kierunku lewo/prawo (-/+) , natomiast zmienna *f* oznacza przesunięcie góra/dół (+/-).

Obrót

Zaskakująco łatwo definiuje się również obrót obrazu przy użyciu biblioteki *PIL.Image*:

```
rotatedImage = testImage.rotate(45)
imgplot = plt.imshow(rotatedImage)
```

Obrót definiowany jest przy użyciu kąta obrotu określonego przy użyciu miary kąta w stopniach. Obrót obrazu, podczas gdy kąt jest zdefiniowany liczbą dodatnią, wykonywany jest odwrotnie do ruchu wskazówek zegara. Aby uzyskać obrót zgodny z ruchem wskazówek zegara należy używać ujemnych wartości kątów obrotu.



Rysunek 1: Efekt końcowy

III. Uwagi

Zadania zostały wykonane przy użyciu biblioteki Pillow.Image. Import całej biblioteki Pillow nie jest konieczny, wystarczy zaimportować sam moduł Image używając na przykład polecenia:

```
from PIL import Image
```

Do wykonania zadań można również użyć innych bibliotek języka Python jednak znajomość przedstawionych zagadnień z biblioteki Pillow będzie wymagana na zajęciach.

IV. Lista zadań

1. Wykonaj modyfikację kodu Kadrowania tak aby na wejściu podawać zmienne lewego górnego rogu okna kadrowania oraz jego szerokość i wysokość.
2. Wykonaj modyfikację kodu Przeskalowania tak aby można było modyfikować szerokość i wysokość obrazu przy użyciu zmiennych skali (jedna zmienna będzie skalować szerokość a druga długość obrazka wynikowego).
3. Wykonaj modyfikację kodu Obrotu tak aby na wejściu podawać wartość kąta w Radianach.
4. Napisz skrypt, który wycina z obrazu wejściowego krok po kroku fragmenty obrazu o rozmiarze 256x256 oraz zapisuje je do plików. Następnie skrypt po kolei wczytuje utworzone pliki obraca je o 180 stopni i ponownie zapisuje do nowego pliku z przyrostkiem 180.