Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych

Rozpoznawanie Obrazów

Semestr II, Informatyka

mgr inż. Marcin Skobel

2022

Laboratorium nr 3: Operacje sąsiedztwa, filtry liniowe i operacje morfologiczne.

I. Zagadnienia teoretyczne

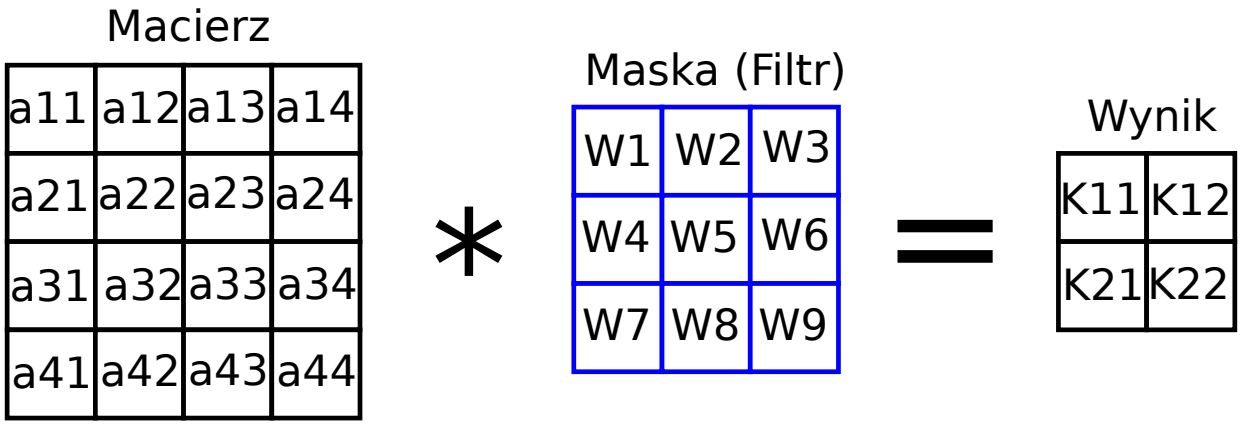
Wstęp

Oprócz poznanych na poprzednich zajęciach operacji punktowych istnieją również operacje sąsiedztwa, w których ostateczna wartość transformowanego piksela wynika także z lokalnych wartości sąsiednich pikseli. Pojęcie operatorów sąsiedztwa wiąże się z pojęciem splotu oraz filtrów obrazu. Operacje morfologiczne z kolei są zazwyczaj stosowane do przekształcania obrazów binarnych, jednak można również wykonywać je na obrazach w odcieniach szarości. Na tym laboratorium skupimy się na przetwarzaniu obrazów binarnych, na których wartość zerowa oznacza tło obrazu przetwarzanego, natomiast elementy pierwszoplanowe oznaczane są wartościami 1. W celu ułatwienia sobie procesu wyświetlania obrazu w języku Python można wartość 1 zastąpić wartością 255. W bieżącym dokumencie wszystkie przykładowe rysunki mają odwróconą kolorystykę dla większej czytelności.

Operacja splotu

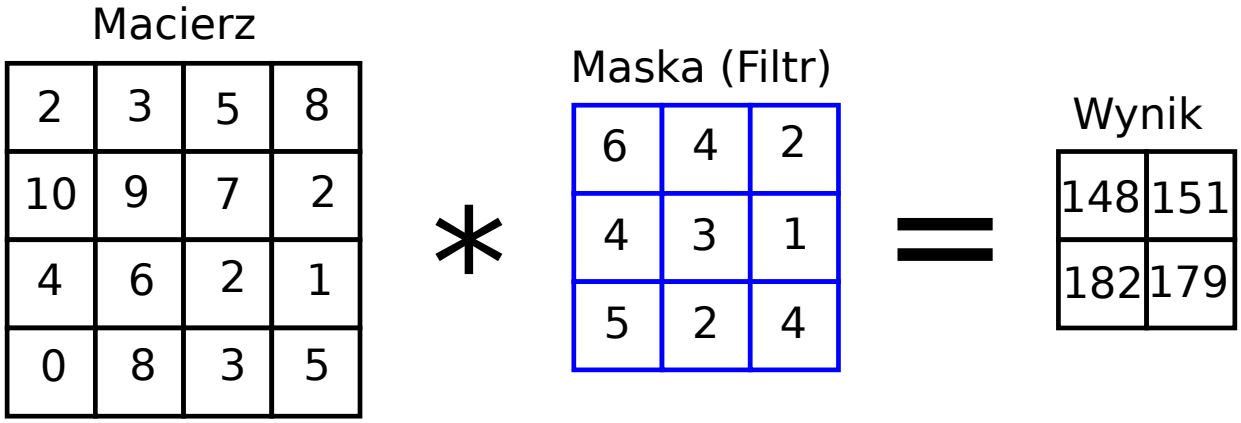
Zanim przejdziemy do filtracji obrazu warto najpierw przyjrzeć się operacji splotu. Dokładne zrozumienie operacji splotu jest niezbędne do kontynuacji tematu filtracji obrazów, a także może być przydatne w zrozumieniu działania splotowych sieci neuronowych. Do wykonania operacji splotu niezbędna będzie macierz wejściowa czyli na przykład obraz, następnie filtr wraz z wagami. Splot w literaturze często określa się symbolem *. Splot polega na mnożeniu poszczególnych pikseli obrazu przez odpowiadające im piksele na masce a następnie zsumowaniu tych iloczynów. Chcąc obliczyć przykładowo wartość K11 z obrazka 1 należy dokonać następującego obliczenia:

$$a11 \times W1 + a12 \times W2 + a13 \times W3 + a21 \times W4 + a22 \times W5 + a23 \times W6 + a31 \times W7 + a32 \times W8 + a33 \times W9 \quad (1)$$



Rysunek 1: Przebieg operacji splotu

Ostatecznie otrzymamy macierz splotu o wymiarach 2×2: Tutaj może się pojawić pytanie co



Rysunek 2: Wynik splotu

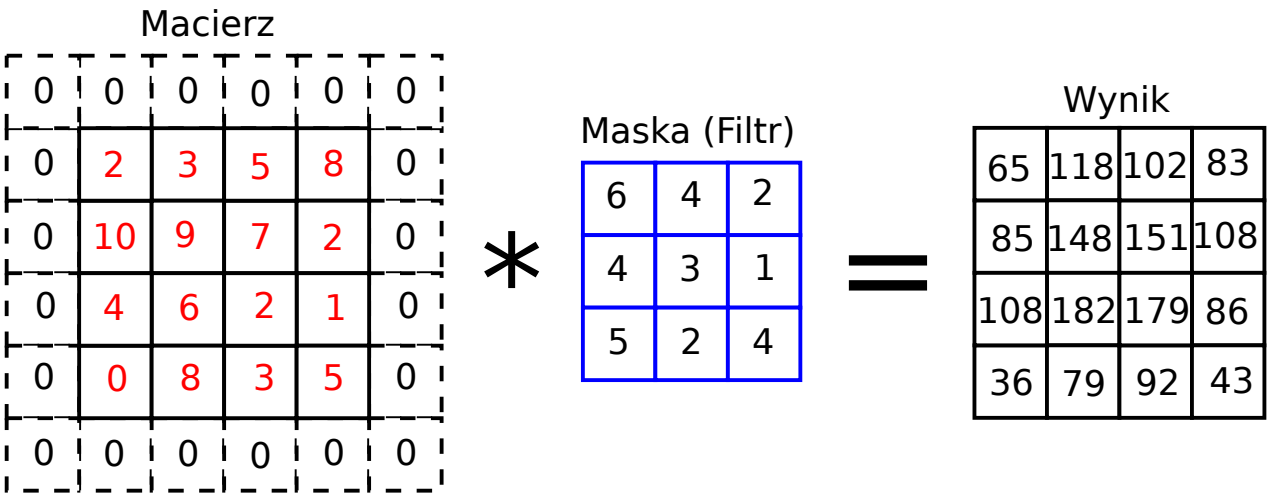
zrobić w przypadku gdy chcemy uzyskać macierz splotu o identycznych wymiarach jak macierz wejściowa. Istnieją dwa rozwiązania. Pierwsze rozwiązanie polega na powiększeniu obrazu wejściowego o odpowiednią liczbę zer wokół obrazu. Jeżeli maska splotu jest wymiaru 3×3 to dopisujemy jedną kolumnę z zerami po prawej i po lewej stronie macierzy a następnie po jednym wierszu na dole i jednym na górze. Drugie rozwiązanie polega na dopisaniu obwiedni z zer do macierzy wynikowej. Drugie rozwiązanie jest stosowanie w splotowych sieciach neuronowych.

Filtr jednorodny i trójkątny - wygładzanie obrazu

Filtr jednorodny występuje w postaci macierzy z jednakowymi wartościami wag w przypadku macierzy 3×3:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \tag{2}$$

Biorąc pod uwagę, że po zastosowaniu takiego filtra należy potem dokonać normalizacji obrazu, można też zastosować wersję filtra podzielonego przez sumę elementów macierzy filtrującej



Rysunek 3: Rozwiązanie problemu brzegowego

ostatecznie otrzymując:

$$\begin{bmatrix} \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \\ \frac{1}{9} & \frac{1}{9} & \frac{1}{9} \end{bmatrix} \tag{3}$$

Filtr ten służy do wygładzania obrazów w tym usuwania szumu oraz zakłóceń na obrazie. Wielkość filtra może być większa niż 3×3, ale wraz z powiększaniem rozmiaru filtra tracimy jeszcze bardziej ostrość obrazu. Filtry jednorodne to nie jedyna grupa filtrów przeznaczonych do wygładzania obrazu. Drugi pokrewny rodzaj filtrów są to filtry trójkątne. Przykładowy filtr trójkątny może przybrać następującą formę:

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 3 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{4}$$

A także po podzieleniu przez sumę elementów:

$$\begin{bmatrix} \frac{1}{15} & \frac{2}{15} & \frac{1}{15} \\ \frac{2}{15} & \frac{1}{5} & \frac{2}{15} \\ \frac{1}{15} & \frac{2}{15} & \frac{1}{15} \end{bmatrix} \tag{5}$$

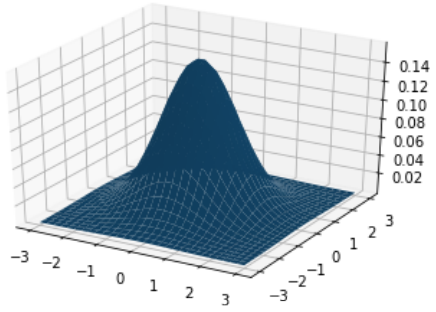
Zastosowanie tego typu filtrów generuje efekty uboczne w postaci zmniejszenia ostrości rozmazania krawędzi i innych szczegółów obrazu.

Filtr Gaussa

Do rozmycia obrazu często stosowany jest filtr Gaussa. Macierz filtracyjna otrzymuje wartości na podstawie funkcji Gaussa dla rozkładu normalnego:

$$g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \tag{6}$$

Oznacza to, że kiedy tworzymy filtr zgodnie z rozkładem Gaussa, centralny piksel ma największą wagę a sąsiednie piksele uzyskują mniejszą wagę podczas wykonywania splotu. Piksel, który ma



Rysunek 4: Rozkład Gaussa

zostać zmodyfikowany, będzie miał największą wagę w filtrze, a waga zmniejszy się dla pikseli, które są daleko. Przykładowo dla filtra o rozmiarze 3×3 i wartości $\sigma = 1$ filtr przyjmie wartości:

$$\begin{bmatrix} 0.060 & 0.098 & 0.060 \\ 0.098 & 0.162 & 0.098 \\ 0.060 & 0.098 & 0.060 \end{bmatrix} \tag{7}$$

Co oznacza że suma wartości poszczególnych elementów zgodnie z rozkładem Gaussa powinna się sumować do wartości 0.78. Często stosowany jest również zapis filtru Gaussa w postaci potęgowej:

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{8}$$

Wersja filtra o wartości 1 σ dla rozmiaru 5×5 przyjmuje następującą postać:

$$\begin{bmatrix} 0.002 & 0.013 & 0.022 & 0.013 & 0.002 \\ 0.013 & 0.060 & 0.098 & 0.060 & 0.013 \\ 0.022 & 0.098 & 0.162 & 0.098 & 0.022 \\ 0.013 & 0.060 & 0.098 & 0.060 & 0.013 \\ 0.002 & 0.013 & 0.022 & 0.013 & 0.002 \end{bmatrix} \tag{9}$$

gdzie suma wszystkich elementów macierzy wynosi 0.98.

Filtr kierunkowy

W zależności od potrzeb istnieją również metody rozmywania obrazu w określonym kierunku. Przykładowo rozmycie w kierunku horyzontalnym można wykonać poprzez użycie poniższego filtra:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

Chcąc rozmyć obraz w kierunku wertykalnym użyjemy takiej macierzy:

$$\begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix} \tag{11}$$

Można też przetestować rozmycie po diagonalnej bądź po drugim ukosie:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad lub \quad \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{12}$$

Wyostrzenie obrazów

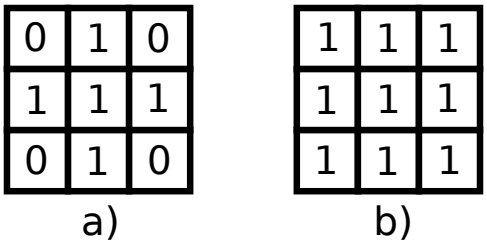
Proste filtry mogą być stosowane nie tylko do rozmazywania obrazów, ale także mogą pomóc wyostrzyć obraz. Przykładowo filtry o rozmiarze 3×3:

$$F1 = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad lub \quad F2 = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix} \tag{13}$$

Wyostrzając obraz nie trzeba poprzestawać na jednym splocie. Kolejny splot jeszcze bardziej wyostrza obraz. Niestety efektem ubocznym wyostrzania jest powstawanie artefaktów na obrazie.

Elementy strukturalne

Na początku zaczniemy od zdefiniowania podstawowych elementów strukturalnych służących do operacji morfologicznych. Elementy strukturalne mogą powstać na bazie sąsiedztwa. W literaturze wyróżnia się dwa podstawowe typy sąsiedztwa: czterospójne (von Neumanna) oraz ośmiospójne (Moore'a). Dzięki zastosowaniu sąsiedztwa powstają dwa podstawowe elementy strukturalne również czterospójne oraz ośmiospójne. Elementy strukturalne mogą jednak przyj-



Rysunek 5: Elementy strukturalne: a) czterospójne, b) ośmiospójne

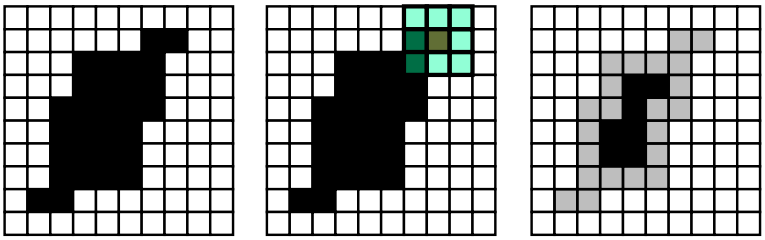
mować inne formy, na przykład posiadać jedynie wybrane sąsiedztwa lub mogą przyjmować kształty większych macierzy np.: 5x5, 7x7.

Erozja

Erozja obrazów może się kojarzyć z pojęciami z zakresu geomorfologii związanymi z procesami powodującymi ubytek powierzchni terenu pod wpływem czynników zewnętrznych. Te skojarzenia są słuszne ponieważ proces erozji w przypadku obrazów cyfrowych polega na redukcji liczby skrajnych pikseli. Operacja erozji polega na przyłożeniu elementu strukturalnego do każdego z pikseli i jeżeli któremukolwiek elementowi o wartości 1 na obiekcie strukturalnym odpowiada wartość 0 na obrazie wówczas piksel obrazu znajdujący się w tej samej pozycji co środek elementu strukturalnego otrzymuje wartość 0. Erozja obrazu I przy użyciu elementu strukturalnego E wyraża się wzorem:

$$I \ominus E = x | E'_x \subset I \tag{14}$$

gdzie E'_x oznacza element przyłożenie elementu strukturalnego E' do piksela x.

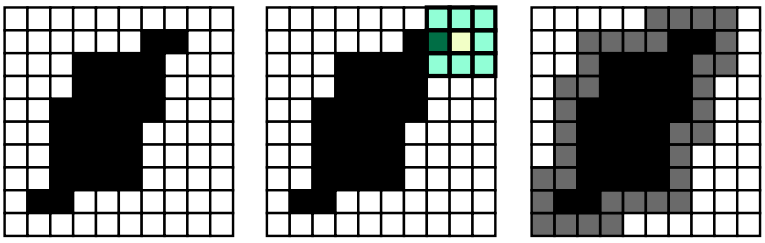


Rysunek 6: Przebieg operacji erozji

Dylatacja

Operacja dylatacji stanowi przeciwieństwo erozji. Innymi słowy dylatacja powoduje nadbudowanie obiektu w zależności od przyłożonego elementu strukturalnego. Operacja dylatacji polega zatem na przyłożeniu elementu strukturalnego do każdego z pikseli i jeżeli któremukolwiek elementowi o wartości 1 na obiekcie strukturalnym odpowiada wartość 1 na obrazie wówczas piksel obrazu znajdujący się w tej samej pozycji co środek elementu strukturalnego otrzymuje wartość 1. Dylatacja obrazu I przy użyciu elementu strukturalnego E wyraża się wzorem:

$$I \oplus E = x | E'_x \subset I \tag{15}$$



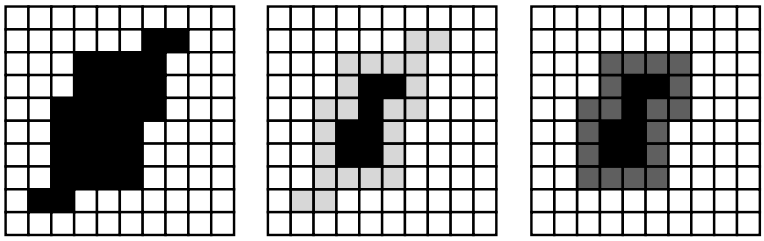
Rysunek 7: Przebieg operacji dylatacji

Otwarcie

Oprócz podstawowych operacji morfologicznych istnieją również operacje złożone. Jedną z takich operacji jest otwarcie. Operacja ta jest definiowana jako nałożenie operacji dylatacji na obraz po erozji. Otwarcie możemy zadem zdefiniować przy użyciu wzoru:

$$I \circ E = (I \ominus E) \oplus E \tag{16}$$

Operacja otwarcie zachowuje zbliżony do pierwotnego kształt obiektu jednocześnie usuwając „odstające" elementy obiektu.



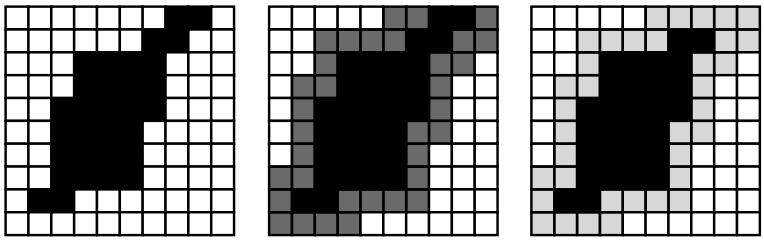
Rysunek 8: Przebieg operacji otwarcia

Domknięcie

Druga ze złożonych operacji nazywana jest dopełnieniem. Operacja ta jest realizowana przez nałożenie operacji erozji na obraz po dylatacji. Otwarcie możemy zadem zdefiniować przy użyciu wzoru:

$$I \bullet E = (I \oplus E) \ominus E \tag{17}$$

Operacja domknięcia zachowuje zbliżony do pierwotnego kształt obiektu jednocześnie doklejając „odstając" elementy obiektu i reedukując element przy krawędzi obrazu.



Rysunek 9: Przebieg operacji domknięcia

Znajdowanie konturu

Wyznaczanie konturu może zostać wykonane przy użyciu operacji morfologicznych. Pierwsze metoda polega na wyznaczeniu konturu zewnętrznego z użyciem operacji dylatacji:

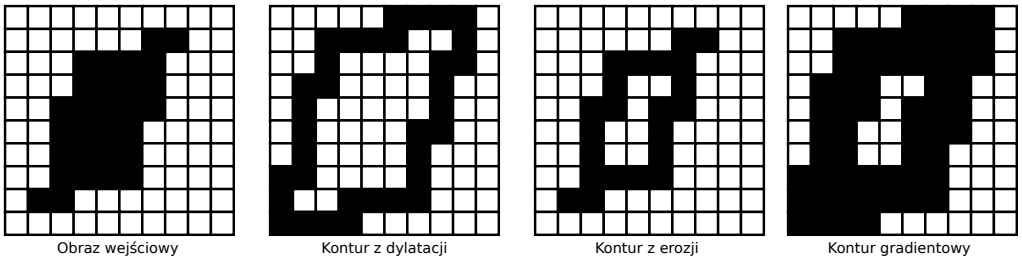
$$K(I) = (I \oplus E) - I \tag{18}$$

Wyznaczenie konturu przy użyciu operacji erozji można przeprowadzić zgodnie ze wzorem:

$$K(I) = I - (I \ominus E) \tag{19}$$

Można również użyć obu podstawowych operacji morfologicznych w celu uzyskania krawędzi metodą gradientu morfologicznego. Metodę można zrealizować przy użyciu wzoru:

$$G(I) = (I \oplus E) - (I \ominus E) \tag{20}$$



Rysunek 10: Kontury

Szkieletyzacja

Zadanie szkieletyzacji wiąże się z fundamentalną transformacją morfologiczną o nazwie Hit-or-Miss. Transformacja Hit-or-Miss jest stosowana do wykrywania krawędzi oraz narożnych pikseli a także do operacji pogrubiania i pocieniania. Podstawowym elementem strukturalnym transformacji Hit-or-Miss jest macierz rozmiaru 3×3 o poniższej konfiguracji zer i jedynek:

$$\begin{bmatrix} & 1 & \\ 0 & 1 & 1 \\ 0 & 0 & \end{bmatrix} \tag{21}$$

W macierzy występują puste miejsca. W zależności od potrzeb miejsca te możemy wypełniać zerami lub jedynkami. Ponadto taki pojedynczy element strukturalny nie wykryje wszystkich punktów narożnych obiektu. Aby poradzić sobie z problemem wykrywania narożników wykonuje się kilkukrotną transformację Hit-or-Miss z elementem strukturalnym w pierwotnej wersji, a następnie obróconym o 90°, 180° oraz 270°:

$$\begin{bmatrix} & 1 & \\ 0 & 1 & 1 \\ 0 & 0 & \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & \\ 0 & 1 & 1 \\ & 1 & \end{bmatrix} \quad \begin{bmatrix} & 0 & 0 \\ 1 & 1 & 0 \\ & 1 & \end{bmatrix} \quad \begin{bmatrix} & 1 & \\ 1 & 1 & 0 \\ & 0 & 0 \end{bmatrix} \tag{22}$$

Transformacja Hit-or-Miss może również służyć do pogrubiania lub pocieniania. Pogrubienie polega na powiększeniu pikseli obiektu na obrazie i przypomina nieco dylatowanie. Operacja pogrubiania może być zrealizowana na podstawie wzoru:

$$T_{(I,E)} = I \cup T_{Hit-or-Miss\ (I,E)} \tag{23}$$

Operacja polega na przykładaniu elementu strukturalnego do kolejnych grup pikseli, gdzie analizowany piksel znajduje się w miejscu punktu centralnego obiektu strukturalnego, a następnie jeśli element strukturalny pasuje do otoczenia badanego punktu wartość piksela nie ulega zmianie a jeśli element strukturalny nie pasuje do otoczenia wówczas punkt uzyskuje wartość jeden. Bardzo podobnie wygląda operacja pocieniania:

$$T_{(I,E)} = I - T_{Hit-or-Miss\ (I,E)} \tag{24}$$

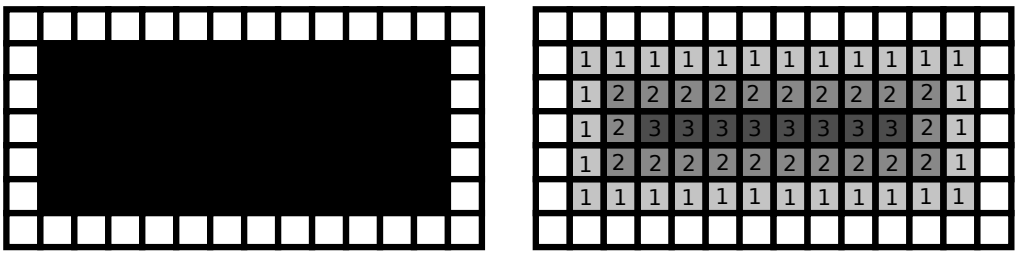
Operacja tym razem polega na przykładaniu elementu strukturalnego do kolejnych grup pikseli, gdzie analizowany piksel znajduje się w miejscu punktu centralnego obiektu strukturalnego, a następnie jeśli element strukturalny pasuje do otoczenia badanego punktu wartość piksela nie ulega zmianie a jeśli element strukturalny nie pasuje do otoczenia wówczas punkt uzyskuje wartość zero. Możemy teraz przejść do sedna sprawy czyli szkieletyzacji obiektów, która polega na wielokrotnym wykonaniu operacji pocieniania przy użyciu 8 elementów strukturalnych. Te 8 elementów uzyskujemy poprzez obrócenie dwóch elementów strukturalnych o 0°, 90°, 180° i 270°. Dwa podstawowe elementy mają następującą postać:

$$\begin{bmatrix} & 1 & \\ 0 & 1 & 1 \\ 0 & 0 & \end{bmatrix} \quad \begin{bmatrix} 0 & 0 & 0 \\ & 1 & \\ 1 & 1 & 1 \end{bmatrix} \tag{25}$$

Druga metoda pozyskiwania szkieletu obiektu polega za zastosowaniu transformacji odległościowej. Przykładem metryki do służącej do obliczania transformacji odległościowej może być metryka euklidesowa, metryka Manhattan, metryka Czebyszewa. Najprostszym przykładem może być metryka euklidesowa określana w przestrzeni dwuwymiarowej wzorem:

$$d_{|AB|} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{26}$$

W wyniku transformacji odległościowej otrzymujemy piksele z wartościami określonymi jako odległość od obszaru tła a zatem przykładowo:



Rysunek 11: Transformacja odległościowa

Najodleglejsze od krawędzi piksele stanowią szkielet obiektu. Transformacje odległościowe mogą być przydatne również w innych obliczeniach na przykład w algorytmie segmentacji wododziałowej.

II. Przykład praktyczny

Wstępna konfiguracja

Tym razem bez większych zmian:

```
from PIL import Image
import numpy as np
import matplotlib.pyplot as plt
import cv2
from google.colab import drive
drive.mount('/content/drive')
```

Wczytanie pliku

Następnie należy wczytać plik testowy i skonwertować do go odcieni szarości. Znów użyjemy gotowej funkcji do konwersji obrazu do odcieni szarości i skonwertujemy do tablicy numpy.

```
path = testImagePath = 'drive/My Drive/Colab Notebooks/Images/Test.tif'
img = Image.open(path)
imgGray = img.convert('LA')
npImageArray = np.asarray(imgGray,dtype="uint8")
```

Filtr jednorodny

```
IA = npImageArray[:,:,0]
Kernel = np.array([[1/9,1/9,1/9],[1/9,1/9,1/9],[1/9,1/9,1/9]])
Size = Kernel.shape + tuple(np.subtract(IA.shape, Kernel.shape) + 1)
subM = np.lib.stride_tricks.as_strided(IA, shape = Size, strides = IA.strides * 2)
m = np.einsum('ij,ijkl->kl', Kernel, subM)
imgAveraging = Image.fromarray(np.uint8(m))
```

W pierwszej linijce kodu pobieramy wartości tablicy do której został zapisany obraz wejściowy. W drugiej linijce podajemy interesujący nas filtr, czyli musi mieć odpowiedni rozmiar i wartości. W trzeciej linijce przygotowujemy wektor zawierający 4 wartości. Pierwsze dwie to rozmiar filtra a kolejne dwie to rozmiar obrazu wyjściowego po splocie czyli pomniejszony o dwie kolumny i dwa wiersze. Czwarta linijka służy do utworzenia widoku na tablicę z podanym dokładnym krokiem i kształtem tablicy. Kolejna linijka kodu używa funkcji einsum do wykonania operacji splotu. Funkcja einsum może w efektywny sposób zastępować różne kombinacje sum i mnożenia elementów tablic. Dopisek w postaci 'ij,ijkl-¿kl' pozwala na wykonanie operacji splotu. Ostatnia linijka kodu to po prostu zamiana otrzymanej tablicy na obraz cyfrowy.

Filtr Gaussa

```
IA = npImageArray[:,:,0]
Kernel = np.array([[0.06,0.098,0.06],[0.098,0.162,0.098],[0.06,0.098,0.06]])
Size = Kernel.shape + tuple(np.subtract(IA.shape, Kernel.shape) + 1)
subM = np.lib.stride_tricks.as_strided(IA, shape = Size, strides = IA.strides * 2)
m = np.einsum('ij,ijkl->kl', Kernel, subM)
imgGauss = Image.fromarray(np.uint8(m))
```

Filtracja Gaussa została wykonana analogicznie do poprzedniego przykładu. Zmieniona została tablica filtracji.

Wyostrzanie obrazu

```
IA = npImageArray[:,:,0]
Kernel = np.array([[-1,-1,-1],[-1,9,-1],[-1,-1,-1]])
Size = Kernel.shape + tuple(np.subtract(IA.shape, Kernel.shape) + 1)
subM = np.lib.stride_tricks.as_strided(IA, shape = Size, strides = IA.strides * 2)
m = np.einsum('ij,ijkl->kl', Kernel, subM)
imgSharpening = Image.fromarray(np.uint8(m))
```

Wyostrzanie obrazu również w podobny sposób zostało wykonane.

Wyświetlenie obrazów

```
fig = plt.figure()
fig.set_size_inches(20, 10)
ax1 = fig.add_subplot(2,2,1)
ax1.imshow(imgGray)
ax1.set_title("Input")
ax1 = fig.add_subplot(2,2,2)
ax1.imshow(imgAveraging, cmap="gray")
ax1.set_title("Averaging operator")
ax1 = fig.add_subplot(2,2,3)
ax1.imshow(imgGauss, cmap="gray")
ax1.set_title("Gaussian")
ax1 = fig.add_subplot(2,2,4)
ax1.imshow(imgSharpening, cmap="gray")
ax1.set_title("Sharpening")
```

W ostatnim okienku wykonano wyświetlanie wszystkich przetworzonych w ćwiczeniu obrazów.

Wczytanie pliku

```
path = testImagePath = 'drive/My Drive/Colab Notebooks/Images/Test.tif'
#flaga 0 oznacza, ze obrazek na etapie wczytywania jest konwertowany do skali szarosci
img = cv2.imread(path,0)
#do binaryzacji uzyjemy metody Otsu
ret, imgT = cv2.threshold(img,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
#jesli obiekty wynikowe sa czarne nalezy odwrocic kolory
imgT = 255-imgT
#w efekcie dostajemy taki oto obraz
plt.imshow(imgT, cmap="gray")
```

Element strukturalny

```
#deklaracja elementu strukturalnego (nie ma znaczenia czy wpiszemy 1 czy 255)
kernel = np.array([[0,1,0],[1,1,1],[0,1,0]],np.uint8)
```

Erozja

```
erosion = cv2.erode(imgT,kernel,iterations = 1)
plt.imshow(erosion, cmap="gray")
```

Dylatacja

```
dilation = cv2.dilate(imgT,kernel,iterations = 1)
plt.imshow(dilation, cmap="gray")
```

Otwarcie

```
opening = cv2.morphologyEx(imgT, cv2.MORPH_OPEN, kernel)
plt.imshow(opening, cmap="gray")
```

Domknięcie

```
closing = cv2.morphologyEx(imgT, cv2.MORPH_CLOSE, kernel)
plt.imshow(closing, cmap="gray")
```

Znajdowanie konturu - Gradient morfologiczny

```
gradient = cv2.morphologyEx(imgT, cv2.MORPH_GRADIENT, kernel)
plt.imshow(gradient, cmap="gray")
```

Szkieletyzacja

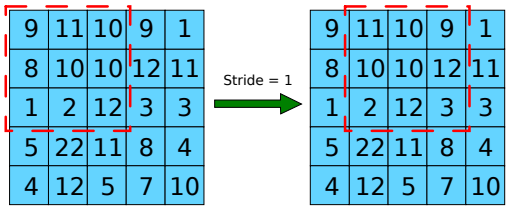
```
thinned = cv2.ximgproc.thinning(imgT)
plt.imshow(thinned, cmap="gray")
```

Transformacja odległościowa

```
distTrans = cv2.distanceTransform(imgT,1,0)
plt.imshow(distTrans, cmap="gray")
```

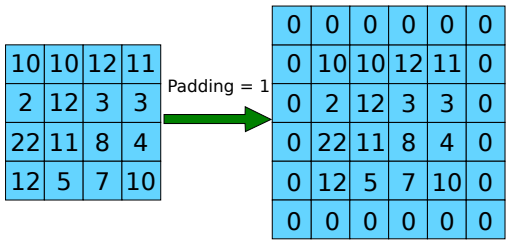
III. Uwagi

Istnieją gotowe funkcje pozwalające na przeprowadzenie filtracji. Funkcje te można odnaleźć m. in. w bibliotekach Pillow, OpenCV. Filtrowanie obrazów oraz operacja splotu nieodzownie wiąże się z dwoma ważnymi pojęciami. Pierwsze pojęcie do przesunięcie (*ang. stride*), które oznacza wielkość kroku przemieszczenia macierzy filtrującej po obrazie. Pojedynczy skok najczęściej przyjmuje wartość 1. Drugie pojęcie określa się jako dopełnienie (*ang. padding*) i oznacza



Rysunek 12: Przesunięcie

ono obszar o jaki poszerzono obraz wyjściowy po filtracji w celu uzyskania wymiarów identycznych jak obraz wejściowy.



Rysunek 13: Dopelnienie

Lista laboratoryjna może być przydatna i kompatybilna z zadaniami projektowymi. Tym razem używamy gotowych funkcji w bibliotece OpenCV. Funkcja cv2.ximgproc.thinning jest stosunkowo nowym elementem biblioteki openCV i nazwa może być trochę myląca bo oznacza pocienianie, które jest używane do szkieletyzacji zaś sama funkcja zgodnie z literaturą powinna się raczej nazywać skeletonization. Plik z przykładami można pobrać ze strony: `http://staff.uz.zgora.pl/mskobel/lab3_niest.py`

IV. Lista zadań

1. Na podstawie przykładowych zadań wykonaj filtrację w kierunku horyzontalnym wertykalnym i jednym skośnym następnie wyświetl obrazy wspólnie z obrazem wejściowym i porównaj wizualnie wyniki.
2. Użyj wzoru 6 odnoszącego się do rozkładu normalnego w celu wygenerowania macierzy filtru Gaussa na podstawie zmiennych w postaci parametru σ oraz rozmiaru filtra. Następnie użyj zaimplementowany generator do wykonania filtracji Gaussa. Można wszystko obudować w funkcję.
3. Na podstawie wzoru: $IM = I * F * F$ oraz przykładów praktycznych wykonaj operację podwójnego wyostrzenia obrazu. Symbol $*$ oznacza operację splotu.
4. Obraz po operacji splotu ma mniejsze wymiary niż obraz wejściowy. Skorzystaj z wycinania obrazów w celu pobrania z obrazu wejściowego brakującej obwiedni obrazu wynikowego. Następnie dodaj obwiednię do obrazu wynikowego. Sprawdź czy po przekształceniu obrazy mają identyczne wymiary.
5. Użyj narzędzia erozji w celu utworzenia własnego algorytmu wykonującego erozję ze zmiennym elementem strukturalnym, który ma przybierać formę $np.array([[0,1,0],[1,1,1],[0,1,0]], np.uint8)$ na zmianę z $np.array([[1,1,1],[1,1,1],[1,1,1]], np.uint8)$
6. Użyj transformacji odległościowej na dowolnym obrazie binarnym. Następnie odszukaj wartość maksymalną na obrazie po transformacji. Zamień wszystkie wartości większe lub równe (maks-20) na 255 a pozostałe wartości na 0. Wyświetl wynik.
7. Wykonaj na dowolnym obrazie wejściowym gradient morfologiczny. Następnie wykonaj kontur przy użyciu erozji oraz drugi kontur przy użyciu dylatacji. Połącz kontury z erozji i dylatacji w jeden kontur i porównaj go z konturem uzyskanym z gradientu morfologicznego. Jak wygląda obraz wynikowy? Czy zgadza się to z teorią?