Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych

Rozpoznawanie Obrazów

Semestr VII, Informatyka

mgr inż. Marcin Skobel

2021

Laboratorium nr 10: Segmentacja obrazu - metoda wododziałowa.

I. Zagadnienia teoretyczne

Wstęp

Segmentacja obrazu stanowi metodę detekcji obiektów na obrazie pod kątem homogeniczności obszarów. Homogeniczność jest determinowana poprzez intensywność jasności, parametry kolorymetryczne, teksturę, kształt. Obszary jednolite pod kątem cech są ostatecznie grupowane w zbiory pikseli. Dzięki temu możemy wyodrębnić z obrazu interesujące element bądź obiekty. Segmentacja może stanowić także metodę oddzielania obszarów połączonych ze sobą na pojedyncze obiekty.

Rozplot (dekonwolucja) obazu

Rozplot obrazu stanowi metodę odfiltrowywania istotnych danych od od zakłóceń. W przypadku dekonwolucji obrazów histopatologicznych zadanie polegać będzie na uzyskaniu wzmocnienia intensywności obszarów zajmowanych przez jądra komórkowe. Z teoretycznego punktu widzenia rozplot stanowi proces odwrotny do splotu a równanie dekonwolucji może przyjąć następującą postać:

$$i * j = k \tag{1}$$

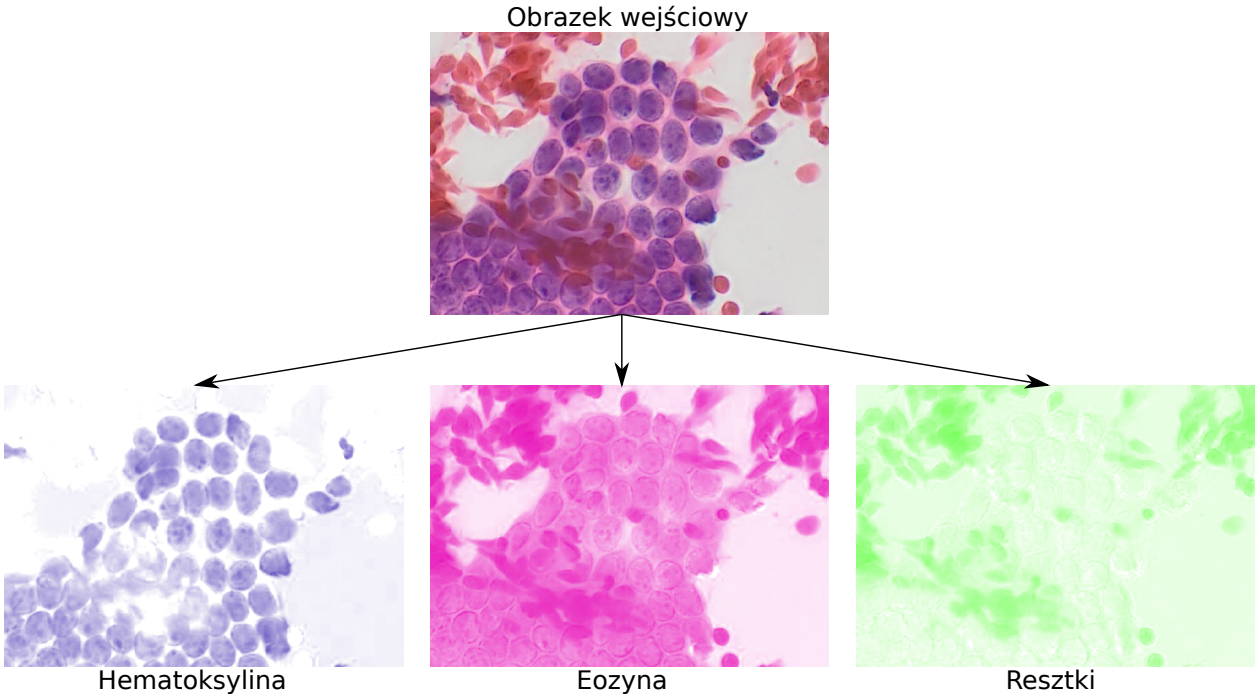
gdzie k stanowi zarejestrowany sygnał (obraz), natomiast i jest sygnałem (obrazem), który będzie podlegał procesowi odzyskania ze splotu z innym sygnałem określonym we wzorze jako g.

Obrazy medyczne, które służą do testów w trakcie naszych zajęć są barwione przy użyciu zasadochłonnej hematoksyliny, służącej do wybarwiania jąder komórkowych oraz przy użyciu eozyny służącej do wybarwiania kwasochłonnych obiektów tkanki. Z uwagi, że obrazy mikroskopowe są wybarwiane w identyczny sposób przyjęto pewną koncepcję dotyczącą dekonwolucji

obrazów wybarwionych metodą H&E. Okazuje się bowiem, że w oparciu o Prawo Lamberta-Beera opisującemu absorpcję światła w zależności od obiektu na który pada można wyznaczyć macierz dekonwolucji dzięki której można wzmocnić odpowiednie piksele na obrazie. W wyniku eksperymentów uzyskano macierz rozplotu H&E o następującej postaci:

$$\begin{matrix} H \\ E \\ R \end{matrix} \begin{bmatrix} 0.65 & 0.70 & 0.29 \\ 0.07 & 0.99 & 0.11 \\ 0.00 & 0.00 & 0.00 \end{bmatrix} \tag{2}$$

Ostatecznie po wykonaniu rozplotu można oczekiwać mniej więcej takich rezultatów:

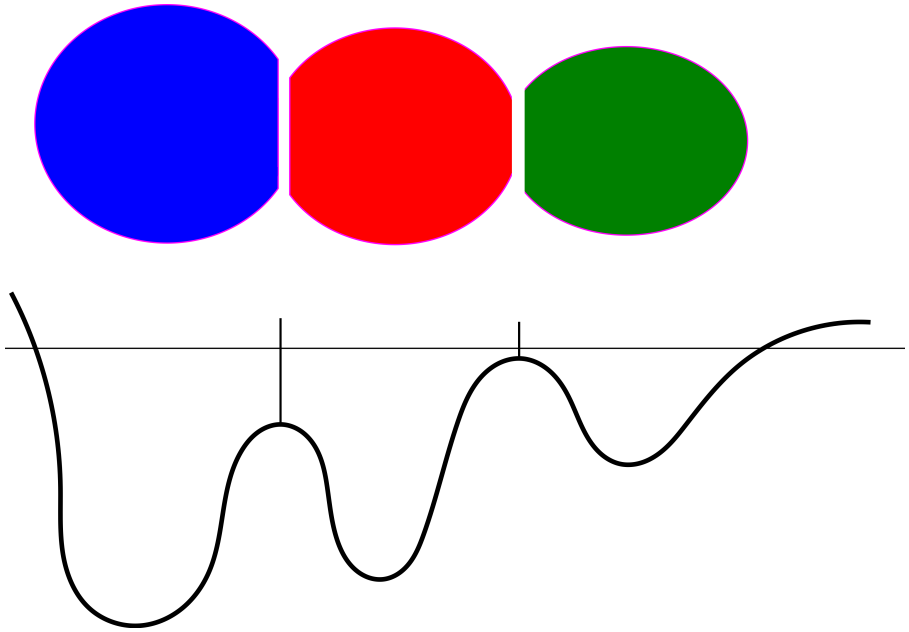


Rysunek 1: Przykładowy obrazek po wykonaniu dekonwolucji

Ostatecznie po rozplocie obrazu otrzymujemy 3 kanały odpowiednio nasycone barwami. Pierwszy kanał uzyskuje wzmocnienie pikseli odpowiedzialnych za wybarwianie hematoksyliną drugi za wzmocnienie fragmentów wybarwionych hematoksyliną a trzeci za pozostałe piksele czyli resztki. Uzyskane kanały są w odcieniach szarości widoczne zabarwienie zostało sztucznie dodane przez oprogramowanie ImageJ.

Segmentacja Wododziałowa

Segmentacja wododziałowa jest specjalnym algorytmem segmentacji obszarowej. Idea tego algorytmu odnosi się do geomorfologicznego zjawiska działu wodnego. W naukach geologicznych dział wodny stanowi zlewisko, które powstaje po ograniczeniu go względem sąsiednich zlewni. W praktyce wygląda to tak, że metoda segmentacji wododziałowej może wykonać zadanie segmentacji obiektów od tła, ale może też posłużyć do rozdzielenie obiektów należących do jednej klasy pomiędzy sobą, jak na załączonym przykładzie (Rys. 2).



Rysunek 2: Segmentacja wododziałowa - wyjaśnienie idei

Algorytm segmentacji wododziałowej składa się z kilku kroków standardowo wykonywanych w kompletnym procesie segmentacji:
1. Pierwszy krok polega na wykonaniu binaryzacji obiektów w celu uzyskania pierwszoplanowych elementów oraz tła.
2. W drugim kroku usuwane są szumy obrazu w postaci małych obiektów bądź artefaktów.
3. Krok trzeci polega na wyznaczeniu transformacji odległościowej, na bazie której zostaną zbudowane wododziały.
4. Ostatni krok polega na wyznaczeniu wododziałów. Jest wiele podejść do ich wyznaczenia na przykład: wododziały poprzez zalewanie obszarów począwszy od najniższych obiektów na mapie. Kolejne podejście polega na poszukiwaniu kierunku do najbliższego minimum lokalnego, każdy piksel otrzymuje przynależność do odpowiedniej zlewni. Podejść jest znacznie więcej, ale najważniejsze w tym momencie jest zrozumieć idee tego algorytmu.

II. Przykład praktyczny

Wstępna konfiguracja

Google Colaboratory podczas instalacji pakietu Histomicstk powoduje powstanie konfliktów pakietów. W tym celu instalujemy Histomicstk w poniższy sposób:

!pip install --upgrade –force-reinstall histomicstk--find-links https://girder.github.io/large_image_wheels

Po instalacji musimy nacisnąć klawisz RESTART RUNTIME w celu przygotowania środowiska do pracy. Niestety również pakiet CV2 wymaga interwencji ponieważ wersja domyślna dla Colaboratory jest zbyt nowa i należy ją zredukować do wersji 4.1:

!pip install --force-reinstall opencv-python-headless==4.1.2.30

```
import histomicstk as htk
import numpy as np
import skimage.io
import skimage.color
import matplotlib.pyplot as plt
import cv2
import scipy.ndimage
from google.colab import drive
drive.mount('/content/drive')
```

Wczytanie pliku

```
path = 'drive/My Drive/Colab Notebooks/Images/Test03.tif'
imInput = cv2.imread(path)
```

Rozplot (dekonwolucja) obrazu

```
# DEKONWOLUCJA
# Na poczatku nalezy zbudowac wektory do budowy macierzy dekonwolucji
# wektory sa wbudowane w biblioteke histomicstk
stain_color_map = htk.preprocessing.color_deconvolution.stain_color_map
print('stain_color_map:', stain_color_map, sep='\n')
# W kolejnym kroku tworzymy liste dzieki ktrorej wybierzemy odpowiednie wektory
# do zbudowania  macierzy dekonwolucji
stains = ['hematoxylin',  # nuclei stain
          'eosin',        # cytoplasm stain
          'null']         # set to null if input contains only two stains
# Przechodzimy do utworzenia macierzy dekonwolucji. Musimy wiedziec jakie
# barwniki zostaly uzyte przez lekarza
W = np.array([stain_color_map[st] for st in stains]).T
# Wykonanie dekonwolucji kolorow
imDec = htk.preprocessing.color_deconvolution.color_deconvolution(imInput, W)
# Wybor obrazu ze wzmocnieniem Hematoksylina
imHematox = imDec.Stains[:, :, 0]
plt.imshow(imHematox, cmap='gray')
plt.title('Hematoxylin', fontsize=16)
plt.show()
```

Segmentacja wododziałowa

```
# Progowanie metoda Otsu
ret2,th2 = cv2.threshold(imHematox,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
binary = 255-th2
plt.imshow(binary, cmap='gray')
plt.title('Otsu', fontsize=16)
plt.show()

#USUWANIE MALYCH OBIEKTOW
#odszukanie wszystkich pojedynczych elementow na obrazie
nb_comp, output, stats, ctr = cv2.connectedComponentsWithStats(binary, connectivity=8)
```

```
#connectedComponentswithStats zwraca kazdy oddzielny komponent z informacjami
#o kazdym z nich, takimi jak rozmiar. Nastepna czesc po prostu usuwa tlo, ktore
#rowniez jest uwazane za komponent, ale w wiekszosci przypadkow tego nie chcemy.
sizes = stats[1:, -1]; nb_comp = nb_comp - 1
#elementy mniejsze od tego progu beda usuwane (wielkosc w pikselach)
min_size = 1000
# deklaracja obrazu po usunieciu malych obiektow
DeleteSmall = np.zeros((output.shape), dtype='uint8')
#przeglad obiektow z pozostawieniem tylko najwiekszych
for i in range(0, nb_comp):
    if sizes[i] >= min_size:
        DeleteSmall[output == i + 1] = 255

plt.imshow(DeleteSmall, cmap='gray')
plt.title('After delete small objects', fontsize=16)
plt.show()

# WYPELNIENIE DZIUR
FillHolesStart = DeleteSmall/255
FillHoles = scipy.ndimage.binary_fill_holes(FillHolesStart).astype(int)
FillHoles = np.array(FillHoles*255, dtype='uint8')
plt.imshow(FillHoles, cmap='gray')
plt.title('After holes fill', fontsize=16)
plt.show()

#Segmentacja wododzialowa
kernel = np.ones((3,3),np.uint8)
# Wyznaczanie pewnych obszarow tla
sure_bg = cv2.dilate(FillHoles,kernel,iterations=3)
# Wyznaczanie pewnych obszarow wnetrz jader komorkowych
dist_transform = cv2.distanceTransform(DeleteSmall,cv2.DIST_L2,5)
ret, sure_fg = cv2.threshold(dist_transform,0.25*dist_transform.max(),255,0)
# Wyznaczanie nieznanych regionow
sure_fg = np.uint8(sure_fg)
unknown = cv2.subtract(sure_bg,sure_fg)
# Oznaczenie markerow
ret, markers = cv2.connectedComponents(sure_fg)
# Dodaj jeden do wszystkich etykiet, aby upewnic sie, ze tlo nie jest 0, ale 1
markers = markers+1
# Teraz oznacz nieznany region zerami
markers[unknown==255] = 0
# Uzycie gotowej funkcji do wykonania segmentacji
markers = cv2.watershed(imInput,markers)
imInput[markers == -1] = [0,255,0]

plt.figure(figsize=(35, 30))
plt.subplot(4,1,1),plt.imshow(FillHoles)
plt.title('Binary'), plt.xticks([]), plt.yticks([])
plt.subplot(4,1,2),plt.imshow(imHematox,cmap = 'gray')
plt.title('Hematoxylin'), plt.xticks([]), plt.yticks([])
```

```
plt.subplot(4,1,3),plt.imshow(markers,cmap = 'gist_ncar')
plt.title('Segmentation'), plt.xticks([]), plt.yticks([])
plt.subplot(4,1,4),plt.imshow(imInput)
plt.title('Input + Segmentation'), plt.xticks([]), plt.yticks([])
plt.show()
```

III. Uwagi

Do wykonania dzisiejszego przykładu warto pobrać plik testowy dostępny pod adresem:
`http://staff.uz.zgora.pl/mskobel/Test03.tif`
Plik z przykładami można pobrać ze strony: `http://staff.uz.zgora.pl/mskobel/lab10.py`

IV. Lista zadań

1. Dekonwolucja obrazu na podstawie macierzy o stałych wartościach dla każdego obrazu nie uwzględnia specyficznych cech każdego obrazu indywidualnie. Aby naprawić ten problem na bazie przykładu zamieszczonego na stronie internetowej: `https://digitalslidearchive.github.io/HistomicsTK/examples/color_deconvolution.html` spróbuj na obrazie medycznym wykonać dekonwolucję nienadzorowaną (Unsupervised color deconvolution).
2. Zweryfikuj jak zostanie posegmentowany obraz jeśli nie zostanie poddany procesowi dekonwolucji. W tym celu wykonaj konwersję obrazka do odcieni szarości a następnie na bazie uzyskanego obrazu wykonaj segmentację wododziałową.