Algorytmy Przetwarzania Obrazów

Segmentacja i analizy obrazów (inpaiting, graphcut)

WYKŁAD 3

Dla studiów niestacjonarnych 2025/2026

Dr hab. Anna Korzyńska, prof. IBIB PAN

Mini-projekty egzaminacyjne

- Tylko 19 na 29 osób w grupie wybrało tematy mini-projektu.
- Na kolejnym wykładzie około 30 minut jest przeznaczonych na komentarze do tematów.
- Plik z informacjami o punktach zdobytych przez studentów znajdzie się w materiałach wykładowych wraz z wykładami na dzień przed laboratorium.

Segmentacja obrazów to cześć procesu przetwarzania, analizy i rozpoznawania obrazów

Laboratorium 3 (max 5,5p)

Zadanie 2. (1,5 p)

Opracować algorytm i uruchomić funkcjonalność realizującą segmentację obrazów następującymi metodami:

- Implementacja progowanie z dwoma progami wyznaczonymi przez użytkownika.
- Implementacja progowanie z progiem wyznaczonym metodą Otsu,
- Implementacja progowanie adaptacyjnego (adaptive threshold).

Laboratorium 4 (max 5p)

Zadanie 1. (3 p)

Opracować algorytm i uruchomić funkcjonalność realizującą wyznaczanie następujących składowych wektora cech obiektu binarnego:
a) Momenty

b) Pole powierzchni i obwód

c) Współczynniki kształtu: *aspectRatio, extent, solidity, equivalentDiameter*
Przygotować zapis wyników w postaci pliku tekstowego do wczytanie do oprogramowania Excel. Program przetestować na podstawowych figurach znakach graficznych (gwiazdka, wykrzyknik, dwukropek, przecinek, średnik, itp.).

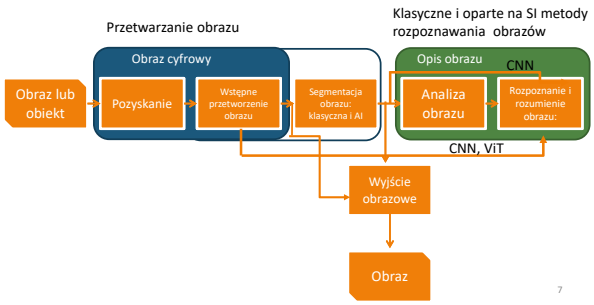
Zadanie 2. (2 p)

Opracować algorytm i uruchomić funkcjonalność jedna z trzech funkcjonalności:

- Inpainting
- Segmentacja metodą GraphCut
- Detekcji krawędzi z użyciem Transformaty Hought

Segmentacja i analiza obiektów

Schemat procesu



7

Etapy procesu rozpoznawania obrazu

Jest to proces składający się z następujących operacji:

1. **Pozyskanie** (*akwizycja*) obrazu i przetworzenie do postaci cyfrowej;
2. **Wstępne przetworzenie obrazu**, jego filtracja i wyostrzanie, wytłumienia nieistotnych informacji;
3. **Segmentacja obrazu** i wydzielenie poszczególnych obiektów oraz ich fragmentów (np. krawędzi i innych linii) lub obiektów jako całość przez konwolucyjne sieci neuronowe (CNN);
4. **Analiza obrazu** i wyznaczenie cech obiektów oraz informacji o ich lokalizacji;
5. **Rozpoznanie** (identyfikacja klasy) metodami klasycznymi i opartymi na Sztucznej Inteligencji (SI, AI).
6. W przyszłości: rozumienie obrazu (Image Understending) w kontekście wiedzy o świecie.

8

19

Segmentacja obrazu

Segmentacja

Wyodrębnienie spośród wybranych fragmentów tych, które stanowią obiekt zainteresowania (ang. ROI) ze względu na cel analizy obrazu.

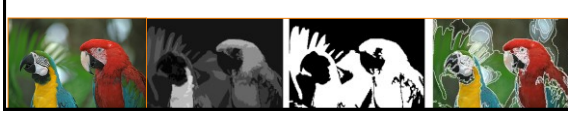
Maska/Mapa

Najbardziej skomplikowane algorytmy

10

Segmentacja

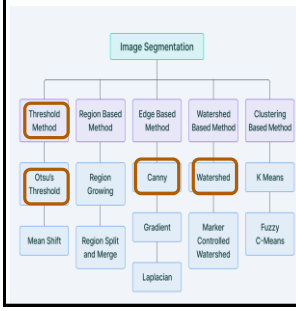
- Segmentacja to podział obrazu na rozłączne (nienakładające się) fragmenty.
- Segmentacja jest powiązana z semantyką (znaczeniem i rozumieniem) obrazu. bywa rozumiana dwojako:
  - Jako podział na jednorodne rejony, które składają się na znaną hierarchię lub strukturę
  - Jako podział na to, co nas interesuje z punktu widzenia celu przetwarzania, pozostałe nieinteresujące obiekty i tło



11

Metody segmentacji

Klasyczne



Oparte na sztucznej inteligencji

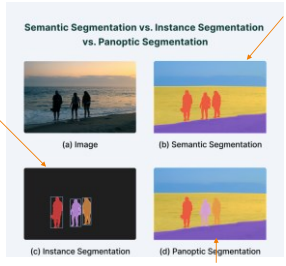
- Segmentacja semantyczna (ang. *Sematic segmentation*) wydzielenie klas
- Segmentacja pojedynczych obiektów jednej klasy - (ang. *Instance segmentaion*) wydzielenie poszczególnych obiektów w klasie
- Segmentacja panoptyczna połączenie poprzednich dwóch metod

Segmentacja semantyczna, instancyjna i panoptyczna

Segmentacja według pojedynczych obiektów (ang. Instance segmentaion)

Segmentacja według pojedynczych obiektów daje mapę obiektów danej kategorii i każdego obiektu danej kategorii oddzielnie. Jest operacją trudną wymagającą oddzielenia obiektów przyległych i nakładających się. Odzwierciedla reprezentację obiektów na poziomie pikseli.

Segmentacja według znaczenia (ang. Sematic segmentation)



Semantyczne segmentacja daje mapę klas obiektów na obrazie bez rozróżnienia pojedynczych obiektów z danej klasy. Piksele należące do obiektów danej klasy nie kumulują żadnej informacji o położeniu przyleganiu lub nakładaniu obiektów.

Segmentacja panoptyczna (ang. Panoptic segmentation)

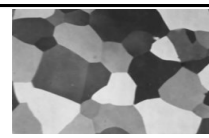
Klasyfikacja metod segmentacji

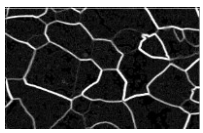
Segmentacja może być zarówno operacją kontekstową (sąsiedztwa) jak i niekontekstową (punktową), ale najczęściej jest kombinacją metod kontekstowych i punktowych.

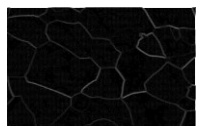
Jeśli metoda:

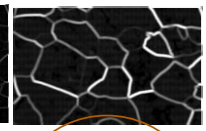
- Ignoruje zależności między pikslami i klasyfikuje je na podstawie globalnej cechy, np. wartości poziomu szarości – progowania, to jest metodą punktową
- Wykorzystuje zależności między pikslami, np.: podobieństwo wartości poziomu szarości – „dziel i łącz", to jest metodą kontekstową

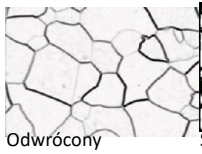
Segmentacja to proces wieloetapowy

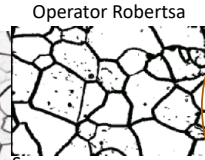



Operator Sobela


Operator Robertsa


Wariancja 5x5


Odwrócony operator Sobela


Sprogowany odwrócony operator Sobela


Pocieniony i oczyszczony sprogowany odwrócony operator Sobela

Ogólna teoria progowania

$$T=T(x, y, p(x, y), n(x, y))$$

x i y - współrzędne w obrazie;
p(x,y) – poziom szarości lub kolor w punkcie x, y
n(x,y) – jakieś lokalne własności punku x,y względem jego sąsiedztwa.

Globalne progowanie: T=T(p(x, y))

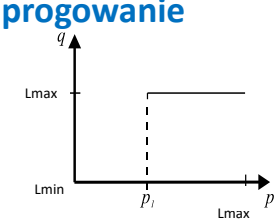
Lokalne progowanie: T=T(p(x, y), n(x, y))
progowanie zależne od tego czy w sąsiedztwie punktu występuje krawędź (skok jasności) czy też obszar jednorodny albo zależy od lokanego kontrastu.

Adaptacyjne (dynamiczne) progowanie:
T=T(x, y, p(x, y))
progowanie, w którym próg jest inny w różnych częściach obrazu – zależnie od występowania cieni, nierównomierności oświetlenia, itp..

Podstawowe progowanie

Operacja redukuje wartości poziomów szarości do dwóch wartości: Lmax i Lmin.

Ma jeden arbitralnie wybierany przez użytkownika parametr zwany progiem (*ang*. **threshold values** (p1); *pol.* próg) – czyli graniczna wartość poziomu jasności powyżej której przypisywana jest Lmax (*ang.* white; *pol.* biel), a dla niej i wszystkich wartości poniże przypisywana jest wartość Lmin (ang. black; *pol.* czerń).

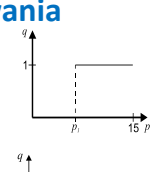


$$q=\begin{cases} L_{\min} & \text{dla } p \le p_1 \\ L_{\max} & \text{dla } p > p_1 \end{cases}$$

Inne warianty operacji progowania

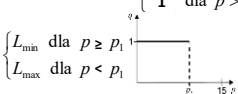
- Progowanie z binaryzacją: Lmin=0 and Lmax=1

$$q=\begin{cases} 0 & \text{dla } p \le p_1 \\ 1 & \text{dla } p > p_1 \end{cases}$$



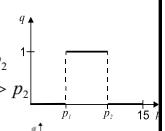
- Progowanie odwrotne

$$q=\begin{cases} L_{\min} & \text{dla } p \ge p_1 \\ L_{\max} & \text{dla } p < p_1 \end{cases}$$



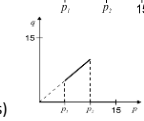
- Progowanie z dwoma progami (dwa parametr: **p1** and **p2**)

$$q=\begin{cases} L_{\max} & \text{dla } p_1 \le p \le p_2 \\ L_{\min} & \text{dla } p < p_1 \text{ lub } p > p_2 \end{cases}$$



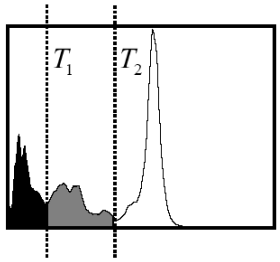
- Progowanie z dwoma progami z pozostawieniem wartości w przedziale

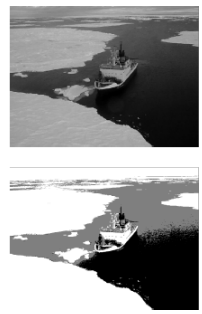
$$q=\begin{cases} p & \text{dla } p_1 \le p \le p_2 \\ 0 & \text{dla } p < p_1,\ p > p_2 \end{cases}$$



- Adaptive thresholding (p1=f(neighborhood(i,j)) i,j-coordinats)
- Recursive thresholding
- Hierarchical thresholding (pyramidal, scalable)

Progowanie z dwoma progami



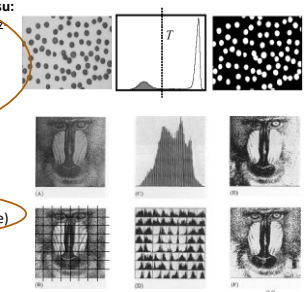


19

Wybór progu do operacji progowania

- **Globalny ustalony próg:**
  - Arbitralnie wyznaczony przez użytkownika
  - Automatycznie wyszukiwana w histogramie najniższa wartość w dolince
  - Próg globalny wyznaczany metodą **Otsu:**

Metoda sprawdza wartość błędu wewnątrz i/lub międzygrupowego we wszystkich możliwych położeniach progu 0 <t <225 i wybiera ten, który minimalizuje błąd wewnątrzgrupowy lub maksymalizuje błąd międzygrupowy.

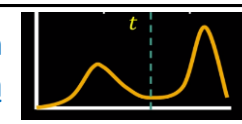


- **Próg o zmiennej wartości zależnej od położenie** na obrazie:
  - **adaptacyjna**
  (zależny od wartości jasności w sąsiedztwie)
  - **dynamiczny**
  (zależny od pozycji w obrazie)

20

Jak automatycznie wyznaczyć wartość progu

Progowanie z progiem wyznaczonym metodą Otsu



Metoda wyznaczania optymalnego progu zaproponowana przez Nobuyuki Otsu w 1979 roku – algorytm oparty na histogramie.

Algorytm sprawdza każdy z możliwych progów 0<t<225 poszukując takiego, który daje minimalną wariancję wewnątrz grupową lub maksymalizuje wariancją międzygrupową

Minimalizacja wewnątrzklasowa – odpowiada minimalizacji wariancji wewnątrzklasowej

$$\sigma^2 = \underbrace{\sigma_w^2(t)}_{\text{Within-class}} + \underbrace{q_1(t)[1-q_1(t)][\mu_1(t)-\mu_2(t)]^2}_{\text{Between-class}}$$

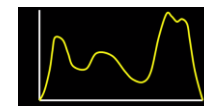
$$\sigma_w^2(t) = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$$

$$q_1(t) = \sum_{i=1}^{t} P(i) \qquad q_2(t) = \sum_{i=t+1}^{I} P(i)$$

$$\mu_1(t) = \sum_{i=1}^{t} \frac{iP(i)}{q_1(t)} \qquad \mu_2(t) = \sum_{i=t+1}^{I} \frac{iP(i)}{q_2(t)}$$

$$\sigma_1^2(t) = \sum_{i=1}^{t} [i-\mu_1(t)]^2 \frac{P(i)}{q_1(t)} \qquad \sigma_2^2(t) = \sum_{i=t+1}^{I} [i-\mu_2(t)]^2 \frac{P(i)}{q_2(t)}$$

Progowanie z dwoma progami wyznaczonymi metodą Otsu



Multiple Thresholds

- Otsu's method can be generalized to find multiple thresholds
- In the case of K classes, we maximize the between-class variance

$$\sigma_B^2 = \sum_{k=1}^{K} P_k (m_k - m_G)^2$$

where

$$P_k = \sum_{i \in C_k} p_i \qquad m_k = \frac{1}{P_k} \sum_{i \in C_k} i\, p_i$$

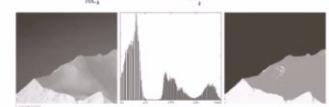


FIGURE 10.45 (a) Image of iceberg. (b) Histogram. (c) Image segmented into three regions using dual Otsu thresholds. (Original image courtesy of NOAA.)

9

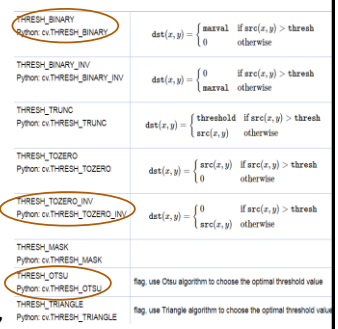
OpenCV

- threshold()

Double cv::threshold(
InputArray src,
OutputArray dst,
double thresh,
double maxval,
int type)

Python:

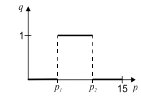
retval, dst=cv.threshold(src, thresh, maxval, type[, dst])



CV.THRESH_BINARY | CV.THRESH_OTSU

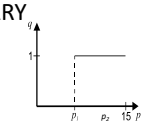
Progowanie z dwoma progami

Złożenie operacji progowania:



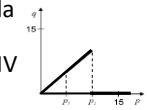
- Progowanie z mniejszym progiem p1

CV.THRESH_BINARY



- Progowanie odwrócone z pozostawieniem poziomów szarości dla większego progu p2

CV.THRESH_TOZERO_INV



Progowanie adaptacyjne

a b c

FIGURE 10.49 (a) Text image corrupted by spot shading. (b) Result of global thresholding using Otsu's method. (c) Result of local thresholding using moving averages.

To progowanie za zmiennym progiem

OpenCV

adaptiveThreshold (src, dst, maxValue, adaptiveMethod, thresholdType, blockSize, C)

src – An object of the class **Mat** representing the source (input) image.
dst – An object of the class **Mat** representing the destination (output) image.
maxValue – A variable of double type representing the value that is to be given if pixel value is more than the threshold value.
adaptiveMethod – A variable of integer the type representing the adaptive method to be used. This will be either of the following two values
- **ADAPTIVE_THRESH_MEAN_C** – threshold value is the mean of neighborhood area.
- **ADAPTIVE_THRESH_GAUSSIAN_C** – threshold value is the weighted sum of neighborhood values where weights are a Gaussian window.

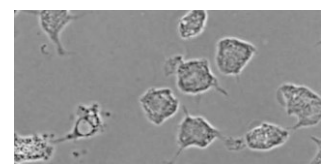
thresholdType – A variable of integer type representing the type of threshold to be used.
blockSize – A variable of the integer type representing size of the pixelneighborhood used to calculate the threshold value.
C – A variable of double type representing the constant used in the both methods (subtracted from the mean or weighted mean).

29

Podobieństwo tekstury

Tekstura reprezentuje, pewną relatywną jednorodność/jednolitość, odczuwaną wzrokowo przez odbiorcę lub udowadnianą jako matematyczna regularność dzięki analizie sygnału.

30

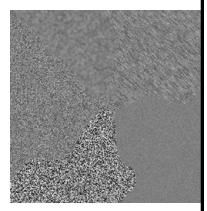


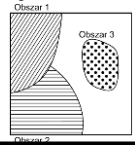
Jednorodność tekstury może być oparta na powtarzalności konstrukcyjnego elementu, wewnątrz którego istnieje pewna nierównomierność poziomów szarości (relacja między podelementami elementu konstrukcyjnego, czyli połączonymi grupami sąsiadujących ze sobą pikseli, jest stała) lub organizację lub uporządkowanie elementów w przestrzeni.

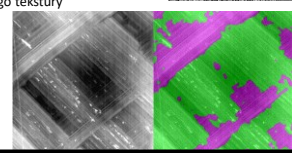
31

Analiza tekstury

- oparta na regularności ocenianej metodami statystycznymi, na podstawie macierzy opisującej częstość występowania dwóch pikseli odległych od siebie o dystans *d* w kierunku *θ* (po angielsku ***cooccurence matrix*), zdefiniowaną przez Haralicka**
  Cechy: kierunkowość, ziarnistość, ...
- oparta na różnych zaawansowanych modelach matematycznej regularności (np. model powtarzalności/zależności poziomów szarości w różnych kierunkach obrazu, oparty na stochastycznych polach Markowa ang. ***random Markov field*** lub model fraktalny samopodobieństwa ang. ***fractal model***)
- oparta na statystycznej powtarzalności (np.: stały zakres różnicy poziomów szarości, stałe odchylenie standardowe poziomów jasności).
- oparta na morfologii matematycznej, która używa różnych transformacji do porównywania struktur w obrazie do znanego elementu konstrukcyjnego tekstury







32

KOLOR

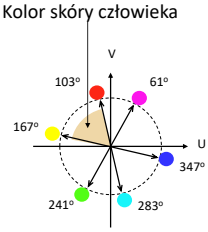
Obraz

Luminance Saturation Hue

33

Poszukiwanie ludzi na kolorowych obrazach

Kolor skóry człowieka

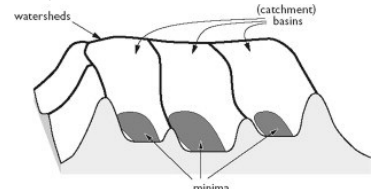


Chrominancje w modelu YUV

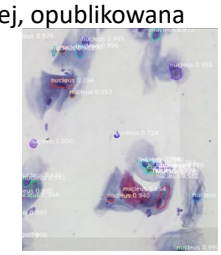
34

Wododział

- Wododział to metoda segmentacji wywodząca się z morfologii matematycznej, opublikowana przez Serra w 1982.



traktuje wartości pikseli jako lokalną topografię (wysokość)

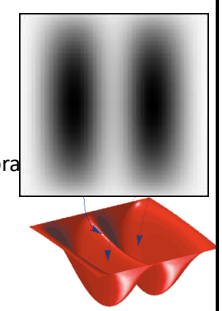


Najczęściej nie na obrazach ale ich wersji po krawędziowaniu

Wododział kontrolowany markerami

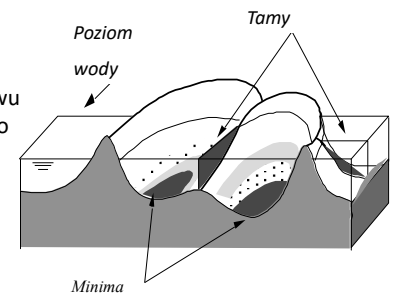
Wymaga znalezienia:

1. markerów (znaczników poszukiwanych obiektów)
2. kryterium segmentacji (funkcji, która zostanie wykorzystana do podziału regionów - najczęściej jest to kontrast lub gradient, ale niekoniecznie).
3. Analiza to konieczności postawienia/budowania tam



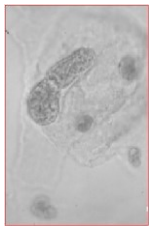
36

Wykonania zlewu kontrolowanego markerem np.: minimami

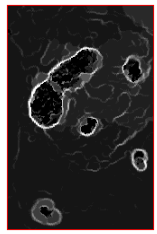


37

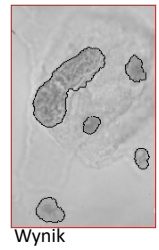
Segmentacja przez wododział na podstawie markerów minimów lokalnych i operator Sobela



Obraz



Jego gradient operator Sobela



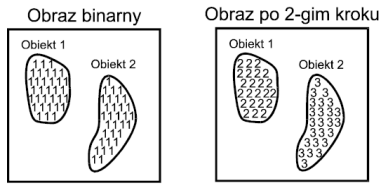
Wynik segmentacji przez wododział

38

Etykietowanie obiekt (indeksacja bezpośrednia)

39

Rozróżnienie obiektów znalezionych w obrazie



40

Indeksacja bezpośrednia obiektów po segmentacji (metoda stosu)

1 krok: *kasowanie* obiektu, zapamiętanie na *stosie.* Analiza kolejnych linii obrazu binarnego *b(x,y).* Po napotkaniu pierwszego punktu obiektu (*b=1*) następuje przeszukiwanie najbliższego otoczenia wykrytego punktu i **kasowanie** kolejnych punktów należących do tego samego obiektu. Jednoczesne **zapamiętywanie** skasowanych punktów na stosie zlokalizowanym w pamięci komputera.

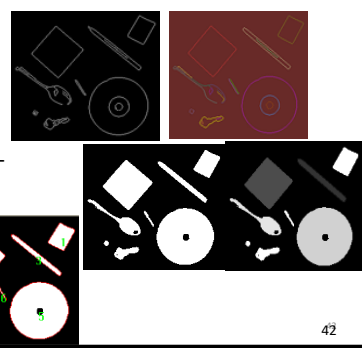
2 krok: odtwarzanie obiektu polegające na nadaniu pikselom wartości będących kolejnymi identyfikatorami odnalezionych obiektów - automatyczne indeksowanie (do zakodowania indeksu wystarcza w praktyce 1 bajt).

41

Etykietowanie obiektów na obrazie

Za pomocą:

- Koloru (pseudokoloru)
- Szarego odcienia – do automatycznej analizy
- Numeru



42

Analiza obrazu

Cechy obiektów

43

Najczęściej wykorzystywane cech

Pomiary lokalne:

1. Tekstura (oparte na statystyce lub wzorze elementarnym)
2. Kontrast globalny i miejscowy (bezwzględny lub wyrażony odchyleniem od jasności średniej)
3. Kolor/barwa, jasność i ich rozkład
4. Cechy topologiczne
5. Cechy geometryczne:
   a. - współczynniki kształtu
   b. - momenty geometryczne
6. Orientacja i wzajemna lokalizacja
7. Cechy dynamiki zmian w czasie dla sekwencji obrazów

Pomiary globalne

1. Liczba obiektów na jednostkę powierzchni obrazu
2. Udział powierzchniowy wybranych elementów obrazu
3. Długość linii na jednostkę pola powierzchni obrazu

Cech koloru/barwy, jasności i ich rozkładu

Opis cech obiektów na podstawie barwy lub jasności pikseli (np.: liść - monochromatyczny w rożnych tonach zieleni; chmury – jasne tony na niebieskim/szarym tle. Itd.).

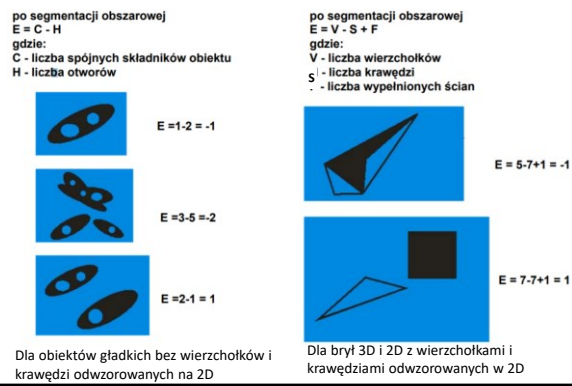
Na podstawie modelu RGB lub któregoś z modeli percepcyjnych można utworzyć trójwymiarowy model-wykres przedstawiając kolory występujące w obrazie i z niego zidentyfikować położenie obiektów o znanym kolorze.

Cechy topologiczne

Cechy topologiczne takie jak: **spójność, wklęsłość, wypukłość, liczba Eulera** są niezależne od położenia obiektu względem układu współrzędnych (zaleta), ale opisują zbyt ogólnych własności obiektów (wada).

Cechy topologiczne są użyteczne w analizie obrazów biomedycznych, ponieważ obiekty obserwowane w naturze wykazują znaczne zróżnicowanie kształtów i/lub spójności i pozwalają za pomocą pojęć topologicznych opisać właściwości wspólne dla całej klasy obiektów. Np.: komórki i ich jądra maja kształty cyrkularne, liczba jąder komórkowych zależy od typu komórek: hepatocyty -komórki wątrobowe są wielojądrzaste, a erytrocyty komórki krwi są bez jądrzaste co można liczbą Eulera.

Liczba Eulera



Dla obiektów gładkich bez wierzchołków i krawędzi odwzorowanych na 2D

Dla brył 3D i 2D z wierzchołkami i krawędziami odwzorowanych w 2D

Cechy geometryczne

Cechy geometryczne to

- współczynniki kształtu,
- współczynniki momentowe.

Mogą być wyznaczane zarówno w płaszczyźnie samego obrazu, jak i jego reprezentacji widmowej (np. w widmie amplitudowym obrazu) i wówczas mówimy o cechach zwanych deskryptorami fourierowskimi.

Współczynniki kształtu

Współczynniki kształtu

- Wyróżnia się następujące parametry służące do opisu geometrii obiektów:
  - a) pole powierzchni, którego pomiar sprowadza się do zliczenia pikseli należących do interesującego nas obszaru wyznaczonego przez zamknięty obrys. Cecha ta jest czuła na błędy wynikłe z niewłaściwej binaryzacji, jednak z drugiej strony jest nieczuła na przesunięcie i obrót obiektu w polu widzenia.
  - b) obwód, czyli długość brzegu obiektu. Pomiar tej cechy jest dość trudny z uwagi na konieczność przybliżania ciągłej linii dyskretną kombinacją punktów obrazu.

contourArea() – liczy pole figury (wyrażane liczbą pikseli) zawartej wewnątrz obwodu opisanego wektorem punktów w przestrzeni 2D stanowiących. (zapisany jako std::vector or Mat.

```
double
cv::contourArea ( InputArray contour,
                  bool oriented = false
```

contour Wektor punktów w przestrzeni 2D stanowiących obrys obiektu. (zapisany jako std::vector or Mat.

oriented Wartość logiczna wskazująca orientację zgodnie lub przeciwnie do wskazówek zegara. Jeśli true oddaje wartość pola z uwzględnieniem znaku jeśli false oddaje wartość bezwzględna

arcLength() – liczy obwód zamkniętego konturu lub długość otwartej krzywej

```
double
cv::arcLength ( InputArray curve,
                bool closed
              )
```

curve Wektor punktów w przestrzeni 2D stanowiących obrys obiektu. (zapisany jako std::vector or Mat.

closed Logiczna wartość wskazująca czy kontur jest zamknięty czy nie.

50

Wydzielenie konturów z obrazu binarnego

FindContours() – znajduje kontury obiektów stanowiących białe plamy na czarnym tle

FindContours(image-source, mode=CV_RETR_LIST, method=CV_CHAIN_APPROX_NONE, offset=(0, 0)) → contours

Mode:
CV_RETR_LIST
CV_RETR_EXTERNAL
CV_RETR_CCOMP
CV_RETR_TREE

```
enum cv::ContourApproximationModes {
  cv::CHAIN_APPROX_NONE = 1,
  cv::CHAIN_APPROX_SIMPLE = 2,
  cv::CHAIN_APPROX_TC89_L1 = 3,
  cv::CHAIN_APPROX_TC89_KCOS = 4
}
```

CV_RETR_LIST	Zbiera wszystkie punkty konturu figury bez ustalenia ich hierarchii
CHAIN_APPROX_NONE	Zbiera wszystkie punkty konturu figury według reguły:max(abs(x1-x2),abs(y2-y1))==1.

UWAGA: jednym z odnalezionych konturów jest ramka obrazu!!

51

W obrazie jest wiele obiektów

```
# Preporcessing
# pogowanie obrazu pierwotnego aby uzyskać obraz binarny
ret,thresh = cv2.threshold(img,127,255,0)

#Właściwe znajdowanie konturów
# funkcja znajdywania konturów w obrazie binarnym
contours,hierarchy = cv2.findContours(thresh, cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
#badanie ilości znalezionych konturów
len(contours) # liczb znalezionych konturów

# Rysowanie konturów na obrazie
# konwersja z szaroodcieniowego do RGB (właściwie to kolejność BGR) aby można było rysować kolorowe kontury
img2 = cv2.cvtColor(img,cv2.COLOR_GRAY2RGB)

# rysujemy pierwszy kontur kolorem niebieskim
cnt = contours[0]
cv2.drawContours(img2, [cnt], 0, (255,0,0), 3)

# rysujemy drugi kontur kolorem zielonym
cnt = contours[1]
cv2.drawContours(img2, [cnt], 0, (0,255,0), 3)

I tak dalej ..
```

52

Współczynniki cyrkularności – do implementacji

$$W1 = 2\sqrt{\frac{S}{\pi}}$$

W1 określa średnicę koła o równej **powierzchni badanego obiektu** *S* - powierzchnia obiektu

$$W2 = \frac{L}{\pi}$$

W2 określa średnicę koła o długości obwodu równej **długości obwodu badanego obiektu** *L* - obwód obiektu,

$$W3 = \frac{L}{2\sqrt{S \cdot \pi}} - 1$$

WM lubW3 - Współczynnik Malinowskiej

Współczynniki *W1,2,3* - szybkie obliczanie

53

Współczynniki cyrkularności – do implementacji

$$W9 = \frac{2\sqrt{\pi \cdot S}}{L}$$

Współczynnik Mz
Uproszczony współczynnik Ma=Malinowskiej

Masywność czyli stosunek pola powierzchni figury do jej wersji wypukłej

Średnica równoważona czyli średnica koła o takiej samej powierzchni jak obiekt

54

Współczynniki cyrkularności – nie będą implementowane

$$W4 = \frac{S}{\sqrt{2\pi \iint_S \left(r^2\right) ds}}$$

Współczynnik Blaira-Blissa
(większa wrażliwość na zmiany kształtu);
r – odległość elementu pola *ds* od środka ciężkości obiektu

$$W5 = \frac{S^3}{\left(\iint_S l ds\right)^2}$$

Współczynnik Danielssona
l - minimalna odległość elementu *ds* od konturu obiektu

$$W6 = \sqrt{\frac{(\sum d)^2}{n \sum d^2 - 1}}$$

Współczynnik Haralicka
d - odległość pikseli konturu od jego środka ciężkości
n - liczba punktów konturu.

Współczynniki *W4,5,6* - wolniejsze obliczanie niż *W1,2,3*

55

Współczynniki cyrkularności – nie będą implementowane

$$W7 = \frac{r_{\min}}{R_{\max}}$$

Współczynnik Lp1;
r_{min} - minimalna odległość konturu od środka ciężkości
R_{max} - maksymalna odległość konturu od środka ciężkości

$$W8 = \frac{L_{\max}}{L}$$

Współczynnik Lp2
Lmax - maksymalny gabaryt obiektu
L rzeczywisty obrys

56

```
#solidity
area = cv2.contourArea(cnt)
hull = cv2.convexHull(cnt)
hull_area = cv2.contourArea(hull)
solidity = float(area)/hull_area
Solidity
```

Masywność czyli stosunek pola powierzchni figury do jej wersji wypukłej

```
# equivalentDiameter
area = cv2.contourArea(cnt)
equi_diameter = np.sqrt(4*area/np.pi)
equi_diameter
```

Średnica równoważona czyli średnica koła o takiej samej powierzchni jak obiekt

57

Momenty geometryczne

Dwuwymiarowy moment geometryczny rzędu (p+q) dla funkcji f(x,y) jest zdefiniowany jako :

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p \cdot y^q \cdot f(x,y)dxdy$$

Moment centralny rzędu (p+q) dla funkcji f(x,y) jest Zdefiniowany jako :

$$M_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (x-\tilde{x})^p \cdot (y-\tilde{y})^q \cdot f(x,y)dxdy$$

gdzie :

$$\tilde{x} = \frac{m_{10}}{m_{00}} \qquad \tilde{y} = \frac{m_{01}}{m_{00}}$$

Dla cyfrowego obrazu podwójne całki mogą być aproksymowane sumami. W ten sposób dla prostokątnej matrycy o wymiarach [m x n] składającej się z punktów x_{ij} otrzymujemy :

$$m_{pq} = \sum_{i=1}^{m}\sum_{j=1}^{n} i^p \cdot j^q \cdot x_{ij}$$ moment zwykły

$$\tilde{i} = \frac{m_{10}}{m_{00}} \qquad \tilde{j} = \frac{m_{01}}{m_{00}}$$ współrzędne środka ciężkości

$$M_{pq} = \sum_{i=1}^{m}\sum_{j=1}^{n} (i-\tilde{i})^p \cdot (j-\tilde{j})^q \cdot x_{ij}$$ moment centralny

Moment zwykły rzędu zerowego jest równy :

$$m_{00} = \sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij}$$

Jest to po prostu suma wartości poszczególnych pikseli, czyli „ciężar" obiektu.

Moment zwykły rzędu pierwszego (1,0) jest równy :

$$m_{10} = \sum_{i=1}^{m}\sum_{j=1}^{n} i^1 \cdot j^0 \cdot x_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} i \cdot x_{ij}$$

Moment zwykły rzędu pierwszego (0,1) jest równy :

$$m_{01} = \sum_{i=1}^{m}\sum_{j=1}^{n} i^0 \cdot j^1 \cdot x_{ij} = \sum_{i=1}^{m}\sum_{j=1}^{n} j \cdot x_{ij}$$

Momenty te służą do wyznaczania środka ciężkości obiektu. Współrzędne tego punktu otrzymujemy obliczając :

$$\tilde{i} = \frac{m_{10}}{m_{00}} \qquad \tilde{j} = \frac{m_{01}}{m_{00}}$$

Z punktu widzenia rozpoznawania obrazu najbardziej interesują nas tzw. niezmienniki momentowe. Są to takie wartości, obliczone na podstawie momentów niskich rzędów, które są niezmiennicze ze względu na obrót, zmianę skali i przesunięcie.

Aby dogodniej zapisać niezmienniki momentowe, wprowadza się momenty znormalizowane:

$$N_{pq} = \frac{M_{pq}}{m^{\zeta}{}_{00}}$$

gdzie:

$$\zeta = \frac{p+q}{2} + 1 \qquad p+q=2,3,4,...$$

Wartości niektórych niezmienników momentowych:

$$M1 = N_{20} + N_{02}$$

$$M2 = (N_{20} - N_{02})^2 + 4 \cdot N_{11}^2$$

$$M3 = (N_{30} - 3 \cdot N_{12})^2 + (3 \cdot N_{21} - N_{03})^2$$

$$M4 = (N_{30} - N_{12})^2 + (N_{21} - N_{03})^2$$

$$M5 = (N_{30} - 3 \cdot N_{12}) \cdot (N_{30} + N_{12}) \cdot \cdot [(N_{30} + N_{12})^2 - 3 \cdot (N_{21} + N_{03})^2] + + (3 \cdot N_{21} - N_{03}) \cdot (N_{21} + N_{03}) \cdot \cdot [3 \cdot (N_{30} + N_{12})^2 - (N_{21} + N_{03})^2]$$

$$M6 = (N_{20} - N_{02}) \cdot [(N_{30} + N_{12})^2 - (N_{21} + N_{03})^2] + + 4 \cdot N_{11} \cdot (N_{30} + N_{12}) \cdot (N_{21} + N_{03})$$

$$M7 = N_{20} \cdot N_{02} - N_{11}^2$$

$$M8 = N_{30} \cdot N_{12} + N_{21} \cdot N_{03} - N_{12}^2 - N_{21}^2$$

$$M9 = N_{20} \cdot (N_{21} \cdot N_{03} - N_{12}^2) + N_{02} \cdot (N_{30} \cdot N_{12} - N_{21}^2) - - N_{11} \cdot (N_{30} \cdot N_{03} - N_{21} \cdot N_{12})$$

$$M10 = (N_{30} \cdot N_{03} - N_{12} \cdot N_{21})^2 - - 4 \cdot (N_{30} \cdot N_{12} - N_{21}^2) \cdot (N_{03} \cdot N_{21} - N_{12})$$

Momenty

Moments cv::moments (**InputArray** array,
bool binaryImage = false
)

array	Obraz lub jego fragment (binarny lub monochromatyczny osmiobitowy) lub wektor punktów przestrzeni 2D (1×N or N×1)
binaryImage	Jeśli prawda to 0 - tło a inne wartości 1 – sprowadzenie szaroodcieniowego do binarnego

moments()
#include <opencv2/imgproc.hpp>
Liczy momenty do momentów 3-rzędu
Returns
Moments w strukturz wyjścowej **cv::Moments**

Porównanie współczynników kształtu i momentów

Współczynniki kształtu

- wykazują większą czułość na zniekształcenia niż momenty;
- wpływ dyskretyzacji na współczynniki daje błąd rzędu kilku %;
- niektóre współczynniki (W1,W2) są silnie zależne od wielkości obiektu (zgodnie z ich definicją) i ich użyteczność jest zależna od stopnia normalizacji;
- zakres przyjmowanych wartości (z wyłączeniem W1 i W2) 0,01–100,0;
- wszystkie współczynniki mają zbliżoną wrażliwość na deformacje kształtów;

67

Porównanie współczynników kształtu i momentów

Momenty:

- wyrażenia momentowe nie się zbyt wrażliwe na zmiany kształtów obiektów;
- wpływ dyskretyzacji na momenty daje błąd rzędu kilku %;
- błąd rośnie w miarę wzrostu rzędu momentów;
- zakres przyjmowanych wartości momentów: 10^{-22}–10^{0};
- w zależności od kształtu obiektów (dla określonej klasy) niektóre momenty przyjmują wartości zbyt małe dla istotności analizy
- (poniżej 10^{-9}), wtedy przy wyborze **wektora cech** można je pominąć;
- największą **inwariantność** wykazują momenty M1 i M7;
- istnieją szybkie algorytmy obliczania momentów

68

Uwagi do samodzielnej implementacji algorytmów zaawansowanych:

- Inpaintingu
- Segmentacji metodą GraphCut

Segmentacja GrabCut

Segmentacja z użyciem algorytmu GrabCut

Jest to zaawansowany algorytm segmentacji obrazów, który integruje informacje o kolorze, teksturze i lokalizacji pikseli, aby dokonać segmentacji obrazów. Jest modyfikacją metody GraphCut i tak jak ona wykorzystuje **gradient energii**, który jest zdefiniowana na podstawie różnic w intensywności.

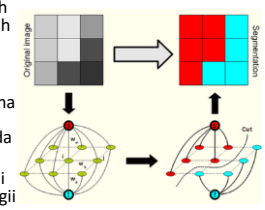
Algorytm przypisuje piksele do segmentów na podstawie optymalizacji minimalizującej energię w obrazie.

Może działać w wyznaczonym interaktywnie lub algorytmicznie fragmencie obrazu.

Algorytm GrabCut został opracowany przez Vivek Kwatra, Arno Schödl, Irfan Essa i Greg Turk z Georgia Institute of Technology jako interaktywna metoda segmentacji binarnej (obiekt-tło) na obrazach kolorowych

GraphCut (do GrabCut)

- Jest to segmentacja obrazów w odcieniach szarości poprzez znajdowanie optymalnych granic między segmentami korzystając z optymalizacji polegającej na minimalizacji energii obrazu.
- Działa przez funkcję kosztu zdefiniowaną na podstawie informacji lokalnych: obecność krawędzi oraz własności otoczenia. Zakłada binarny podział na pikseli obiektu i tła
- GraphCut koncentruje się na optymalizacji funkcji kosztu poprzez minimalizację energii obrazu (algorytm NO Max Flow), podczas gdy GrabCut umożliwia interaktywne segmentowanie obrazów kolorowych, uwzględniając informacje o kolorze, teksturze i lokalizacji pikseli.



Algorytm GraphCut został opracowany przez Y. Boykov i M-P. Jolly z Microsoft Research w 2001 r.

Algorytm GrabCut

1. Inicjalizacja:
   - Na początku przypisywane są segmenty do każdego piksela obrazu.
   - Obliczane są początkowe **wartości energii** dla każdego piksela na podstawie różnic w intensywności pikseli.
2. Iteracje:
   - Algorytm iteracyjnie minimalizuje energię obrazu poprzez optymalne przypisywanie pikseli do segmentów.
   - Dla każdego piksela obliczany jest gradient energii, który określa zmianę energii spowodowaną przypisaniem piksela do innego segmentu.
   - Piksele są przypisywane do segmentów na podstawie gradientu energii, aby osiągnąć zmniejszenie całkowitej energii obrazu.
3. Warunki stopu:
   - Algorytm kontynuuje iteracje aż do osiągnięcia warunku stopu, np. gdy zbiegnie do minimalnej energii lub osiągnie określoną liczbę iteracji.
   - Proces przypisywania pikseli i aktualizacji segmentów jest powtarzany iteracyjnie aż do osiągnięcia zbieżności, czyli momentu, gdy energia obrazu nie ulega znaczącej zmianie.
4. Segmentacja obrazu:
   - Po zakończeniu iteracji uzyskiwana jest ostateczna segmentacja obrazu, gdzie piksele są przypisane do segmentów na podstawie optymalizacji energii.

OpenCv **GrabCut**

```
void cv::grabCut   (   InputArray        img,
                       InputOutputArray  mask,
                       Rect              rect,
                       InputOutputArray  bgdModel,
                       InputOutputArray  fgdModel,
                       int               iterCount,
                       int               mode = GC_EVAL
                   )
```

•*img* - obraz wejściowy
•*mask* - obraz binarny specyfikujący gdzie jest/prawdopodobnie jest obiekt, a gdzie tło: cv.GC_BGD, cv.GC_FGD, cv.GC_PR_BGD, cv.GC_PR_FGD, albo 0,1,2,3.
rect - prostokąt opisujący obiekt do segmentacji (x,y,w,h) – współrzędne wierzchołków prostokąta
•*bdgModel*, *fgdModel* - tablice do użytku wewnętrznego algorytmu deklarowane zewnętrznie w (1,65).
•*iterCount* - ilość iteracji algorytmu.
•*mode* - cv.GC_INIT_WITH_RECT lub cv.GC_INIT_WITH_MASK lub ich kombinacja w zależności jak podajemy algorytmowi prostokąt

- Link do obszernego tutorialu:

https://pyimagesearch.com/2020/07/27/opencv-grabcut-foreground-segmentation-and-extraction/

Inpainting

non-blind lub blind inpainting

Polega na wypełnieniu części obrazu za pomocą konkretnego algorytmu.

Może być również skutecznie wykorzystywana do usuwania tekstu, bazgrołów, a nawet dużych obiektów z wybranego obrazu lub fotografii (non – blind) lub wpisanie czego nowego (blind) .

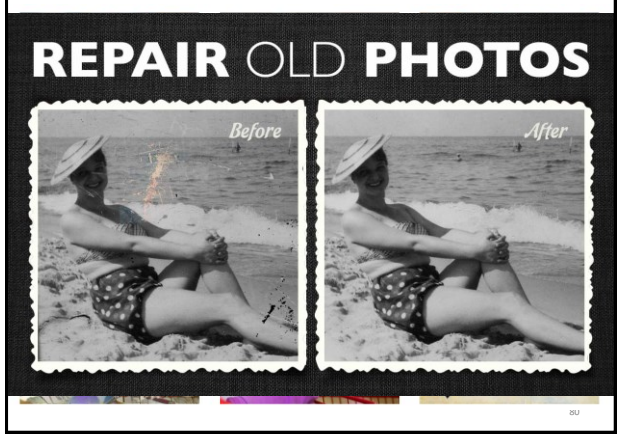
Korygowanie i uzupełnianie obrazu

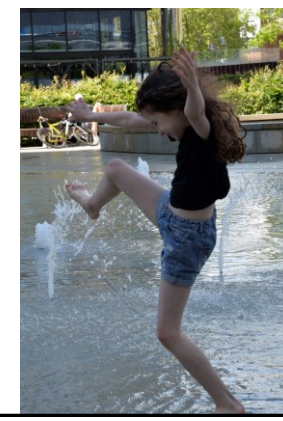
- *Inpaintingiem od lat 30 XX wieku* nazywano proces konserwacji obrazów i fotografii, w których uszkodzone, zniszczone lub brakujące fragmenty są wypełniane w celu stworzenia pełnego obrazu (ang. *image inpainting; inpaint lub inprinting*) – era przedcyfrowa
- Bezpośredni przodek inpaitingu w obrazach cyfrowych narzędzie stempla opracowanego po raz pierwszy przez Adobe do aplikacji PhotoShope. Stempel zastępował uszkodzone piksele pikselami podobnymi do sąsiednich, dzięki czemu dobrze wtapiają się w tło.
- Dwa typy:

1. non-blind inpainting - plamy, bazgroły na obrazie, uszkodzenia – technika FFM (szybkiego marszu)
2. blind inpainting – kompletnie nowe obiekty w obrazie – technika Navier-Stokes, Fluid Dynamics.

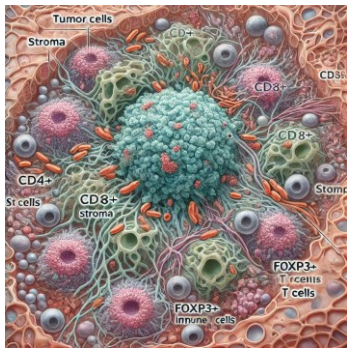
Skąd SI „wie” jak uzupełnić obraz?

- Z analizy otoczenia fragmentu uzupełnianego
- Z analizy stylu całego obrazu





Generatywna SI



OpenCV inpaint

```
void cv::inpaint (InputArray  src,
                  InputArray  inpaintMask,
                  OutputArray dst,
                  double      inpaintRadius,
                  int         flags
                 )
```

inpaintRedius – promień koła o środku w punkcie licznym przez algorytm wewnątrz którego wszystkie intensywności są wykorzystywane przez algorytm do wyliczenia nowej wartości

Flags – metoda liczenia: cv::INPAINT_NS or cv::INPAINT_TELEA

Tutoriale

- https://www.geeksforgeeks.org/image-inpainting-using-opencv/
- https://pyimagesearch.com/2020/05/18/image-inpainting-with-opencv-and-python/

Materiał:

- M.Doros, Przetwarzanie obrazów, skrypt WSISIZ
- Materiały wykładowe POBZ z zeszłego roku na UBIKu
- T.Pavlidis, Grafika i Przetwarzanie Obrazów, WNT Warszawa 1987.
- **I.Pitas,** Digital image processing, algorithms and applications, John Wiley &Sons, Inc. 2000,

85

Gdzie w materiale zgromadzonym na UBIKu znajdziemy algorytmy segmentacji

•Region growing algorithm (pp. 282-285)
•Merging algorithm (pp. 285-289)
•Region splitting algorithm (pp. 289-291)
•Split and merge algorithm (pp. 291-297)
•Relaxation labeling algorithm (pp.297-300)
•Connected component labeling (pp.300-303)
•Texture description (pp.303-317)
•Subroutines for the calculation of the central moments of a histogram (pp.303-306)
•Histograms of gray-level differences (pp.306-308)
•Algorithm for the calculation of horizontal gray-level run legths (pp.308-311)
•Calculation of co-occurence matrix (pp.311-313)
•The spectral characterization of the image texture based on the autocorrelation function) of a two-dimensional image or on its power spectrum (pp.313-318)

86

Literatura dodatkowa:

•W.Zieliński, M.Strzelecki: Komputerowa analiza obrazu biomedycznego, PWN Warszawa-Łódź 2002, str. 178-214; segmentacja z wykorzystaniem analizy tekstur, mozaika Voronoi (Voronoi tesselation), segmentacja metodą określenia działów wodnych (watershed transform)
•T.Pavlidis: Grafika i Przetwarzanie Obrazów, WNT Warszawa 1987
•R.Tadeusiewicz, P.Korohoda, Komputerowa analiza i przetwarzanie obrazów, Wydawnictwo Fundacji Postępu Telekomunikacji, Kraków 1997. http://winntbg.bg.agh.edu.pl/skrypty2/0098/
•Zasoby sieciowe:
•Segmentacja (w szczególności wododziałowa *(watershed)*)
•https://www.mathworks.com/company/newsletters/articles/the-watershed-transform-strategies-for-image-segmentation.html
•Definicja tekstury:
•http://ai.stanford.edu/~ruzon/tex_seg/node1.html

87