

Ćwiczenie S1

Detekcja twarzy

Cele laboratorium:

- a) Zaznajomienie się z podstawami detekcji twarzy na obrazach i sekwencjach video.

Przygotowanie

W katalogu wskazanym przez prowadzącego utwórz podkatalog, w którym będą zapisywane wszystkie dane podczas ćwiczenia. Nazwa katalogu powinna zawierać: **<nr ćwiczenia>_<data>_<nazwisko1>_<nazwisko2>**

Zapisz i rozpakuj pliki ćwiczenia (pobrane z MOODLE) do utworzonego katalogu.

Po skończonych zajęciach pamiętaj o usunięciu danych z dysku.

Informacje teoretyczne

Ludzka twarz jest źródłem wielu informacji. W przeważającej liczbie przypadków wystarcza do zidentyfikowania obserwowanej osoby. Przenosi również informacje ułatwiające komunikację pomiędzy ludźmi (mimika). Dlatego detekcja twarzy na obrazach, będąca pierwszym etapem analizy, znalazła liczne zastosowania w systemach interakcji człowiek-komputer.

Przez detekcję twarzy rozumiemy znajdowanie jej w obrębie przetwarzanego obszaru obrazu, wraz ze ścisłym wyznaczeniem pikseli należących do niej. Jedną z metod jest algorytm detekcji w oparciu o model koloru skóry [1]. Opierając się jedynie na chrominancji (eliminując luminancję), można w sposób dosyć uniwersalny opisać kolor ludzkiej skóry. Wynika to z faktu że skóra poszczególnych osób różni się w głównej mierze jasnością, ma natomiast zbliżone wartości chrominancji. Można więc określić zakres kolorów opisujących ludzką skórę.

Najbardziej rozpowszechnioną przestrzenią kolorów jest RGB, niestety w przestrzeni tej informacje o jasności i chrominancji nie są od siebie odseparowane. Dlatego najczęściej wykorzystywane są następujące przestrzenie:

- HSV (ang. Hue Saturation Value) – składowa Hue odpowiedzialna jest za określenie koloru dominującego, Saturation określa nasycenie, a Value luminancję. Po wyeliminowaniu składowej Value otrzymujemy przestrzeń zdolną do opisu koloru skóry.
- YCrCb – składowe są przeliczane z przestrzeni RGB (dla obrazów JPEG) na podstawie zależności:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.419 & -0.081 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Uwaga!

Dla konwersji przeznaczonej dla standardu SDTV (ang. Standard Definition TeleVision) oraz HDTV (ang. High Definition TeleVision) macierze przekształcenia różnią się [2].

Bez straty ogólności pozostawiamy jedynie składowe Cr i Cb. Jedną z najważniejszych zalet tej przestrzeni, poza dobrą separacją chrominancji od luminancji, jest łatwość konwersji kolorów z przestrzeni RGB.

Zagadnienie lokalizacji twarzy na obrazie zostało przeglądowo przedstawione m. in. w [3, 4, 8].

Inną metodą detekcji twarzy jest algorytm „Haar Cascade Face detector”. Jest to metoda wykorzystująca techniki uczenia maszynowego. Bazując na przykładach pozytywnych oraz negatywnych, uczony jest zbiór klasyfikatorów, które następnie pozwalają na efektywną detekcję obiektów (np. twarzy). Więcej informacji na ten temat można znaleźć w [9][10].

Najczęściej oprócz lokalizacji twarzy na obrazie, przewodnim zagadnieniem jest rozpoznawanie osoby na podstawie wyszczególnionej twarzy [5, 6]. Jedną z metod postępowania jest analiza składowych głównych PCA (ang. Principal Component Analysis) [5, 7, 8].

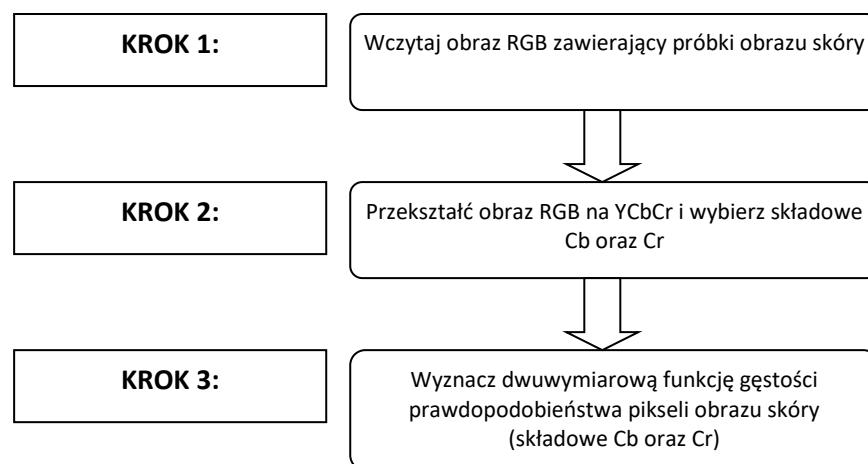
Gdzie szukać dodatkowych informacji:

- Internet – słowa kluczowe: face detection survey
- [1] Yang J., Waibel A., “A Real-Time Face Tracker”, In: Proceedings. 3rd IEEE Workshop on Applications of Computer Vision, WACV '96, Sarasota, Fla. 1996. Los Alamitos, Calif. 1996
- [2] <http://www.equasys.de/colorconversion.html>
- [3] Ming-Hsuan Yang, Kriegman, D.J., Ahuja, N., "Detecting faces in images: a survey", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol 24(1), pp. 34 - 58, Jan 2002
- [4] Hjelm E., “Face Detection: A Survey”, Computer Vision and Image Understanding, vol. 83, pp. 236–274, 2001
- [5] Ślot K., Wybrane zagadnienia biometrii, WKiŁ, 2008
- [6] Bolle R. M., Connell J. H., Pankanti S., Ratha N. K., Biometria, WNT, 2008
- [7] Draper B. A, Baek K., Barlett M. S., Beveridge J. R., “Recognizing faces with PCA and ICA”, Computer Vision and Image Understanding, vol. 91(1-2), pp. 115-137, July-August 2000
- [8] <http://www.face-rec.org/algorithms/>
- [9] Viola, P. ; Mitsubishi Electr. Res. Labs., Cambridge, MA, USA ; Jones, M. “Rapid object detection using a boosted cascade of simple features”: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=990517&tag=1
- [10] Dokumentacja OpenCV: http://docs.opencv.org/master/d7/d8b/tutorial_py_face_detection.html#gsc.tab=0

Część I – Detekcja twarzy z wykorzystaniem modelu koloru skóry

Tworzenie modelu koloru skóry

W pierwszej kolejności utworzony zostanie model koloru skóry w przestrzeni kolorów YCbCr (składowe Cb,Cr). Do tego wykorzystany zostanie specjalnie przygotowany obraz zawierający wybrane fragmenty obrazów skóry wielu osób (plik `model_skory.bmp`). Model koloru skóry tworzony jest w następujący sposób:

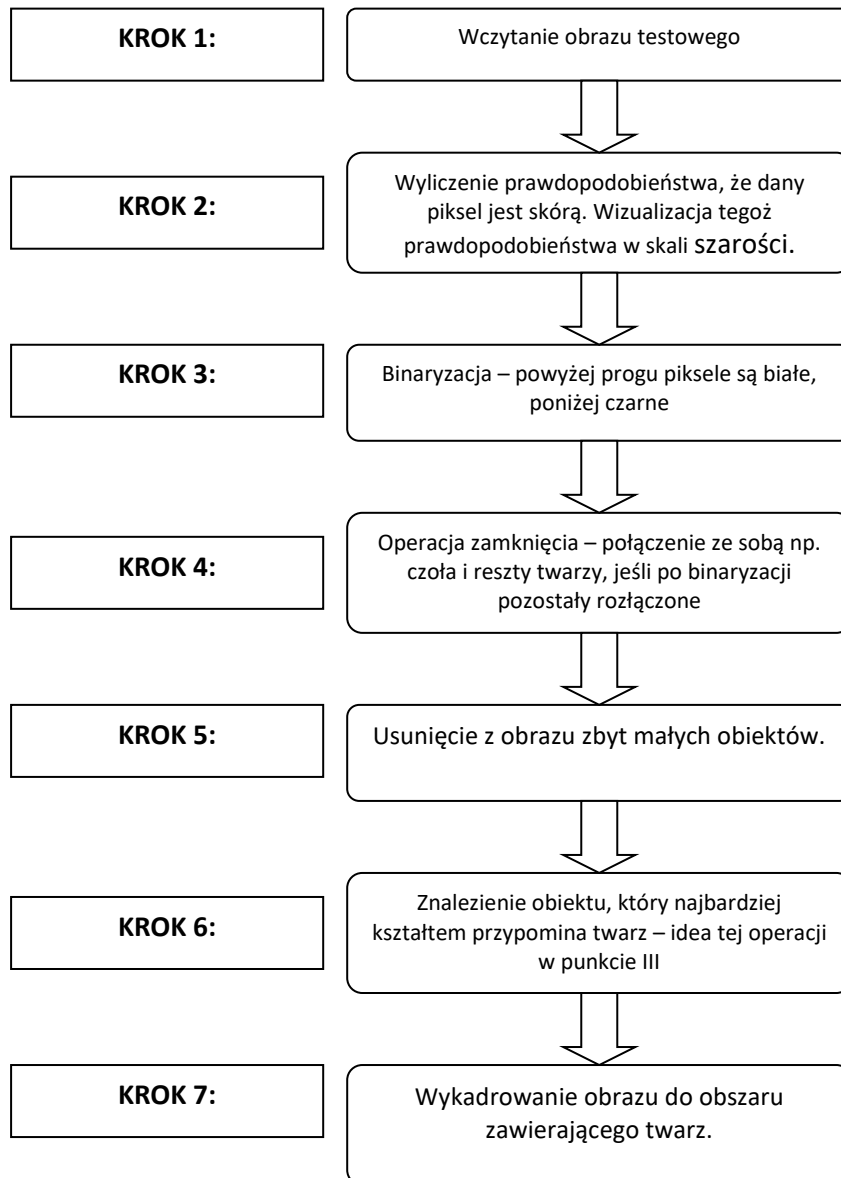


Znormalizowana funkcja gęstości prawdopodobieństwa może być następnie wykorzystana jako dwuwymiarowa tablica korekcji LUT (ang. Look-Up Table) przekształcająca obraz testowy na obraz prawdopodobieństwa.

- Otwórz w programie graficznym (np .Paint) plik `model_skory.bmp` i oceń go wizualnie.
- Wywołaj funkcję `createSkinModel` tworzącą model koloru skóry.
`>> model = createSkinModel('model_skory.bmp');`
- Wyświetl utworzony model i zaobserwuj w jakim obszarze wartości mieści się kolor skóry człowieka.
`>> mesh(model);`
- Otwórz w edytorze MATLABa funkcję `createSkinModel` i zapoznaj się z kolejnymi operacjami.
`>> edit createSkinModel;`
- Utwórz nowy model skóry używając do tego jednego z obrazów testowych (otwórz i wykonaj krok po kroku m-skrypt: `kolor_skory2.m`)
- Porównaj obydwa modele koloru skóry przy pomocy funkcji `imagesc`. Zwróć uwagę na zakresy składowych Cr,Cr w obydwu modelach.
`>> figure;imagesc(model);colormap(jet)`
`>> figure;imagesc(model2);colormap(jet)`

Detekcja twarzy

Detekcja twarzy będzie odbywała się wg następujących etapów:



a) Dodaj ścieżkę do katalogu z plikami obrazów:

```
>> addpath('BazaObrazow')
```

b) Wczytaj obraz testowy (testowy_0_0000.jpeg), na którym będzie szukana twarz. Wyświetl go na osobnym oknie wykresu przy użyciu subplot:

```
>> testowy = imread(testowy_0_0000.jpeg);
>> figure, subplot(2,3,1), imshow(testowy);
```

c) Użyj funkcji probabilityIM do uzyskania obrazu prawdopodobieństwa, w którym jasny odcień oznacza bardzo dużą zgodność barwy danego piksela z modelem koloru skóry, natomiast odcienie ciemne mniejszą

zgodność. Wykorzystana funkcja realizuje dwuwymiarową operację LUT dla składowych Cb oraz Cr obrazu testowego.

```
>> szary = probabilityIM(testowy,model);
>> subplot(2,3,2), imshow(szary,[])
>> colormap(gray);
```


Porównaj rezultaty dla innego modelu koloru skóry. Do dalszych obliczeń przyjmij model skóry, który daje lepsze rezultaty.

```
>> szary = probabilityIM(testowy,model2);
>> subplot(2,3,2), imshow(szary,[])
>> colormap(gray);
```

- d) Otwórz w edytorze MATLABa funkcję `probabilityIM` i zapoznaj się z kolejnymi operacjami.

```
>> edit probabilityIM;
```

- e) Wykonaj binaryzację obrazu prawdopodobieństwa, tak aby uzyskać jak najlepszą segmentację obszaru twarzy. Do binaryzacji użyj funkcji `im2bw`. Dobierz odpowiedni próg binaryzacji (obszar twarzy w kolorze skóry powinien być odpowiednio segmentowany). W tym celu sprawdź jakie wartości pikseli znajdują się w

obszarze twarzy (pomocne może być narzędzie Data Cursor , znajdujące się na pasku narzędzi okna figure; możesz również wykorzystać funkcję `roipoly` do utworzenia maski obszaru twarzy a następnie wyznaczyć średnią wartość obrazu prawdopodobieństwa w tym obszarze). Zadany próg ma wpływ na zakwalifikowanie szarych pikseli do potencjalnej twarzy (białe) lub do otoczenia (czarne).

```
>> level = ...
>> binarny = im2bw(szary, level);
>> subplot(2,3,3), imshow(binarny);
```

- f) Obraz po binaryzacji wymaga dalszego przetwarzania, ponieważ zawiera szereg zakłóceń – np. wiele małych obiektów, oddzielone obszary twarzy, itp. Dlatego następnym krokiem jest wykonanie operacji zamknięcia (funkcja `imclose`), która pozwoli na połączenie białych obiektów (jeśli są odpowiednio blisko siebie). Operacja zamknięcia usunie także zbyt małe czarne obiekty wewnątrz białych obszarów. Zdefiniuj odpowiedni element strukturalny (funkcja `strel` i jej parametry) i dokonaj operacji zamknięcia (funkcja `imclose`).

```
>> se = strel('disk', 6); % dobierz parametry element strukturalnego
>> zamkniety = imclose(binarny, se);
>> subplot(2,3,4), imshow(zamkniety);
```

- g) Na obrazie pozostały jeszcze zakłócenia – niewielkie obiekty. Aby je usunąć można użyć funkcji `bwareaopen`, która usuwa z obrazu wszystkie obiekty o ilości pikseli mniejszej od niż zadany próg. Dobierz odpowiedni próg, wykorzystując funkcję `bwlabel` oraz `regionprops`.

```
>> label1 = bwlabel(zamkniety);
>> res = regionprops(label1)

>> [res.Area]
>> wyczyszczony = bwareaopen(zamkniety,400); % dobierz parametry
>> subplot(2,3,5), imshow(wyczyszczony);
```

- h) W ostatnim etapie użyta zostanie funkcja `szukaj_twarz`. Funkcja ta rekurencyjnie, dla wszystkich obiektów na obrazie, kolejno sprawdza warunki owalności, istnienia obiektu wewnątrz i największej powierzchni. Pozwala na wyeliminowanie z obrazu obiektów które nie spełniają powyższych kryteriów.

```
>> [x1, x2, twarz] = szukaj_twarz(wyczyszczony);  
>> subplot(2,3,6), imshow(twarz);  
  
>> %dodatkowa wizualizacja  
>> subplot(2,3,1)  
>> pos=[x1(2) x1(1) x2(2)-x1(2) x2(1)-x1(1)];  
>> hold on;rectangle('Position',pos,'EdgeColor','red'); hold off
```

- i) Detekcję twarzy przeprowadź dla trzech wybranych obrazów testowych. Zanotuj w raporcie dobrane: progi binaryzacji (pkt. d), elementy strukturalne i ich parametry (pkt. e) oraz minimalną wielkość obiektu (pkt. f). Zanotuj również obserwacje i spostrzeżenia (np. problemy z detekcją,...)
- j) Opcjonalnie – pobierz obraz twarzy z kamery i przeprowadź detekcję twarzy na tym obrazie.

Część II – Detekcja twarzy z wykorzystaniem kaskad Haara

- a) Zapoznaj się z funkcjami MATLABa związanymi z detektorem twarzy:

```
>> doc vision.CascadeObjectDetector
```

- b) Wczytaj obraz testowy (`testowy1.bmp`), na którym będzie szukana twarz.

```
>> testowy = imread('testowy_0_0000.jpeg');
```

- c) Utwórz obiekt detektora:

```
>> faceDetector = vision.CascadeObjectDetector();
```

- d) Zrealizuj detekcję twarzy wywołując funkcję `step` utworzonego detektora – zwróć uwagę czy funkcja zwraca wartości współrzędnych wykrytych twarzy. Zwizualizuj rezultaty detekcji przy pomocy funkcji: `insertObjectAnnotation`

```
>> bbox = step(faceDetector, testowy)
```

```
>> Out = insertObjectAnnotation(testowy, 'rectangle', bbox, 'Twarz');
```

```
>> figure, imshow(Out), title('Wykryta twarz');
```

- e) Wykonaj detekcję twarzy dla wszystkich obrazów testowych. Poeksperymentuj z parametrami obiektu `vision.CascadeObjectDetector` (`np. MinSize, MaxSize, ClassificationModel`). Zanotuj rezultaty (liczbę poprawnych i niepoprawnych detekcji twarzy).

Raport z ćwiczenia nr S1

Data:
Imię i nazwisko:
Imię i nazwisko:

<u>Rezultaty</u> <ul style="list-style-type: none">▪ (cz.I): Zanotuj dobrane: progi binaryzacji, elementy strukturalne oraz minimalną wielkość obiektu dla poszczególnych obrazów testowych. Zanotuj również obserwacje i spostrzeżenia (np. problemy z detekcją,...).▪ (cz.II) Zanotuj rezultaty detekcji twarzy przy pomocy kaskad Haara.
<u>Analiza i wnioski</u> <ul style="list-style-type: none">▪ (cz.I) Podaj kolejność i rolę wykonywanych operacji w etapie wyznaczania modelu koloru skóry. Wyjaśnij rolę operacji zamknięcia w procesie detekcji twarzy. Z jakich operacji morfologicznych składa się operacja zamknięcia?▪ (cz.II) Wyznacz dokładność detekcji twarzy przy pomocy kaskad Haara.
<u>Pytania</u> <ul style="list-style-type: none">▪ W oparciu o przedstawioną literaturę wymień inne metody lokalizacji twarzy na obrazie.▪ Wyjaśnij na czym polega metoda detekcji twarzy przy pomocy kaskad Haara.