Ćwiczenie 6: Rozpoznawanie obrazów w środowisku MATLAB

Adrian Jałoszewski

13 listopada 2017

1 SURF

SURF – speeded up robust features – zainspirowany SIFT. Cechy:

- szybki (kilkakrotnie szybszy od SIFT)
- odporniejszy od SIFT na transformacje obrazu
- opis cech na podstawie sumy falek Haara w okolicy punktów zainteresowania
- wykorzystuje obraz całgowany (integral image) dla optymalizacji
- przez zastosowanie piramidowania odporny na skalowanie
- aproksymuje rozmycie Gaussa – przyśpieszenie w stosunku do SIFT
- wykorzystuje macierz Hessego do znalezienia punktów zainteresowania
- nieodporny na rozmycia obrazu

2 Wyznaczenie cech charakterystycznych dla wzorców

2.1 Nazwy plików

```
file_names = {
    'magneB6_wzorzec.jpg'
    'NoSpa_wzorzec.jpg'
    'chlorchinaldin_wzorzec.jpg'
    'vitaminumB_wzorzec.jpg'
};
```

2.2 Wykrywanie cech SURF

```
modelData = struct('objImage', [], ...
                'objValidPoints', [], ...
                'objFeatures', []);

for i=1:length(file_names)
    file_name = file_names{i};
    image = rgb2gray(imread(file_name));
    surfFeatures = detectSURFFeatures(image);
    strongest = surfFeatures.selectStrongest(100);
    [features, validPoints] = ...
        extractFeatures(image, strongest);

    modelData(i).objImage = image;
    modelData(i).objValidPoints = validPoints;
    modelData(i).objFeatures = features;
```

```
end

save modelData
```

W ten sposób zostały wykryte następujące 100 najsilniejszych cech:

3 Klasyfikacja obiektów

3.1 Wczytywanie obiektu

```
load modelData
objImage = modelData(wzorzecNr).objImage;
objValidPoints = modelData(wzorzecNr).objValidPoints;
objFeatures = modelData(wzorzecNr).objFeatures;
```

3.2 Wczytywanie obrazu

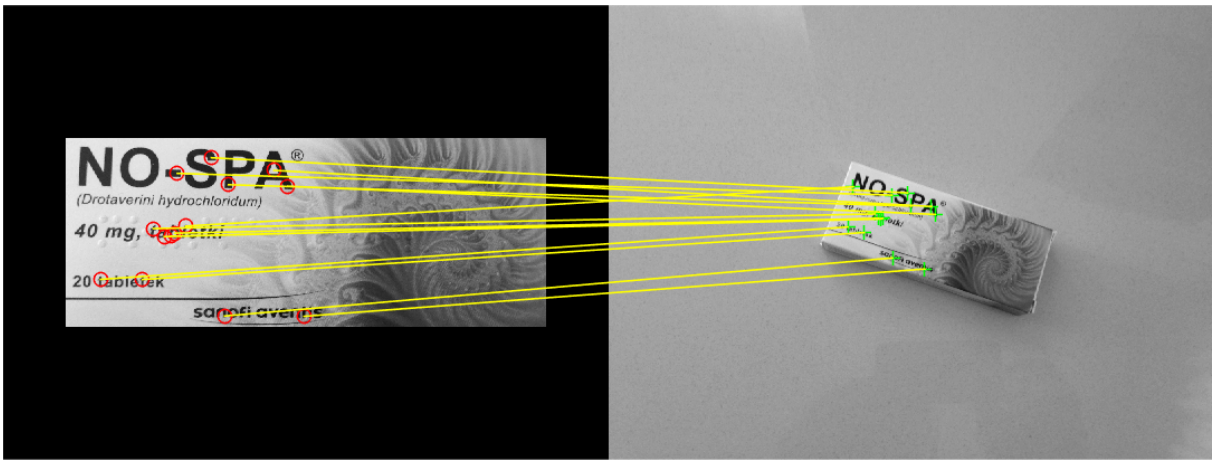
```
nazwa1 = [baseFileName, num2str(i), fileExtension];

disp(nazwa1);
RGB=imread(nazwa1);
sceneImage = rgb2gray(RGB);
```

3.3 Wykrywanie cech SURF i dobieranie parami ze wzorcem

```
surfFeatures = detectSURFFeatures(sceneImage);
strongest = selectStrongest(surfFeatures, 100);
[sceneFeatures, sceneValidPoints] = ...
    extractFeatures(sceneImage, strongest);

featurePairs = matchFeatures(objFeatures, sceneFeatures, ...
                             'unique',true);
matchedObjPoints = objValidPoints(featurePairs(:, 1), :);
matchedScenePoints = sceneValidPoints(featurePairs(:, 2), :);
```

3.4 Wizualizacja

```
if visON==1
    figure;
    showMatchedFeatures(objImage, sceneImage, ...
    matchedObjPoints, matchedScenePoints, 'montage');
    print(['second/', num2str(wzorzecNr), ...
        '_',num2str(i) , '_obraz'], '-dpng')
end
```



Algorytm zwraca szczególnie uwagę na krawędzie liter i probuje je dopasować pomiędzy obrazami. W przypadku jak te krawędzie są rozmazane, mimo tego, że ma doczynienia z tymi samymi przedmiotami nie jest w stanie ich rozróżnić. Obrazy, które zawierają te same opakowania zawierają wiele wspólnych wykrytych cech, obrazy, które są różne zawierają ich mało (albo wskazują na inne miejsca niż powinny).

3.5 Wyznaczenie stosunku dopasowanych punktów do wszystkich

```
metric1(i) = length(matchedObjPoints)/length(objValidPoints);
```

3.6 Prosty algorytm klasyfikacji obrazów testowych

3.6.1 Ręczne etykietowanie

Zmienna `groundTruthTab` zawiera w sobie ręcznie poetykietowane informacje, czy na obrazie znajduje się dany obiekt, czy nie.

```
groundTruthTab={
     [1  1  1  1  0  0  0  0  0  0  0  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  0  0  0  0  0  0  0  1  1  1  1  0  0  0  0]  ...
     [0  0  0  0  0  0  0  0  0  0  0  0  0  1  1  1  1]
};
```

3.6.2 Algorytm

Wyznaczana jest największa procentowo liczba cech wykrytych dla obrazu, ktory nie powinien być wykryty. Następnie jest wyznaczana najmniejsza procentowo liczba cech wykrytych dla obrazu, który powinien być wykryty taka, że jest większa od największej procentowo liczby cech, dla obrazu, który nie powinien być wykryty. Wartość graniczna jest wyznaczana jako średnia tych dwóch liczb.

```
groundTruth = groundTruthTab{wzorzecNr};
correct = find(groundTruth);
```

```
incorrect = find(groundTruth == 0);
max_incorrect = max(metric1(incorrect));
min_propper_correct = find(metric1 > max_incorrect);
threshold = (min(metric1(min_propper_correct)) ...
             + max_incorrect) / 2;

correct_rate = sum(min_propper_correct) / sum(correct);
```

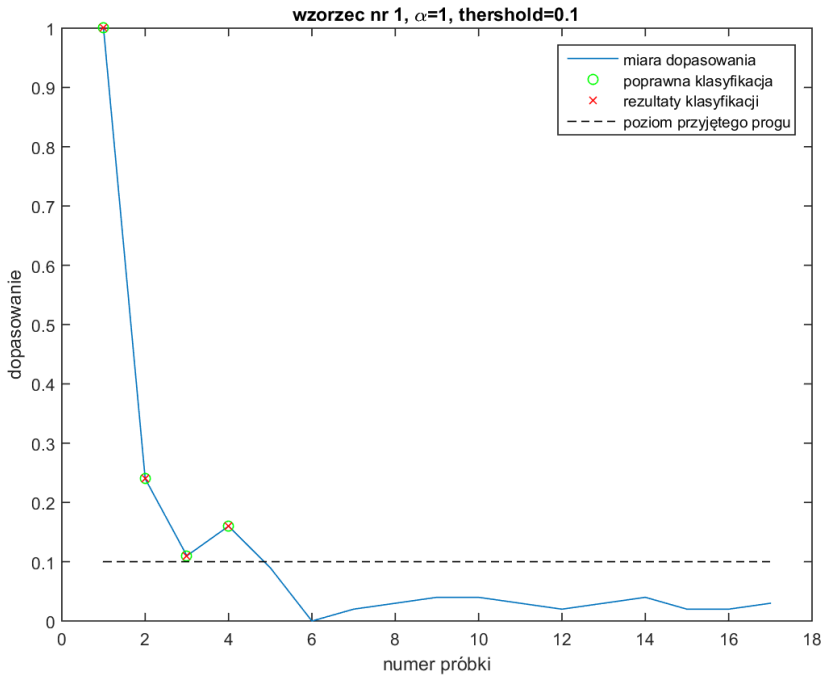
3.6.3 Prezentacja wyników

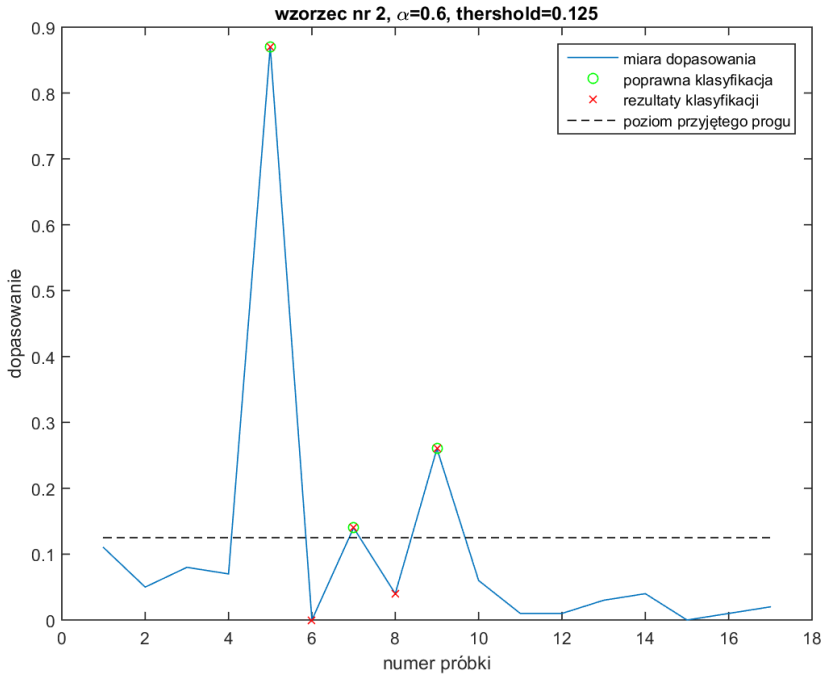
Ponieważ wynik znajduje się w punktach, które znajdują się albo po lewej stronie wykresu albo po prawej miejsce legendy jest uzależnione od numeru próbki.

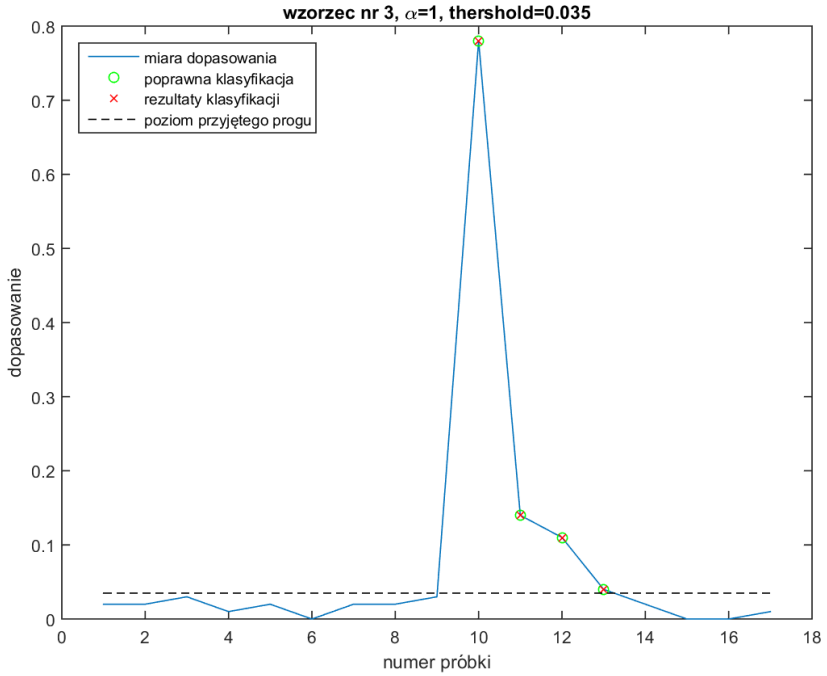
```
figure
plot(metric1)

hold on
plot(min_propper_correct, metric1(min_propper_correct), 'go')
plot(correct, metric1(correct), 'rx')
plot([1, 17], [threshold, threshold], 'k--');

if wzorzecNr > 2
    legend('miara dopasowania', ...
        'poprawna klasyfikacja', ...
        'rezultaty klasyfikacji', ...
        'poziom przyjętego progu', ...
        'Location', 'northwest');
else
    legend('miara dopasowania', ...
        'poprawna klasyfikacja', ...
        'rezultaty klasyfikacji', ...
        'poziom przyjętego progu');
end
title(['wzorzec nr ', num2str(wzorzecNr), ...
       ', \alpha=' num2str(correct_rate), ...
       ', threshold=' num2str(threshold)]);
print(['second_rozpoznane/', 'metric_', ...
        num2str(wzorzecNr)], '-dpng')
```

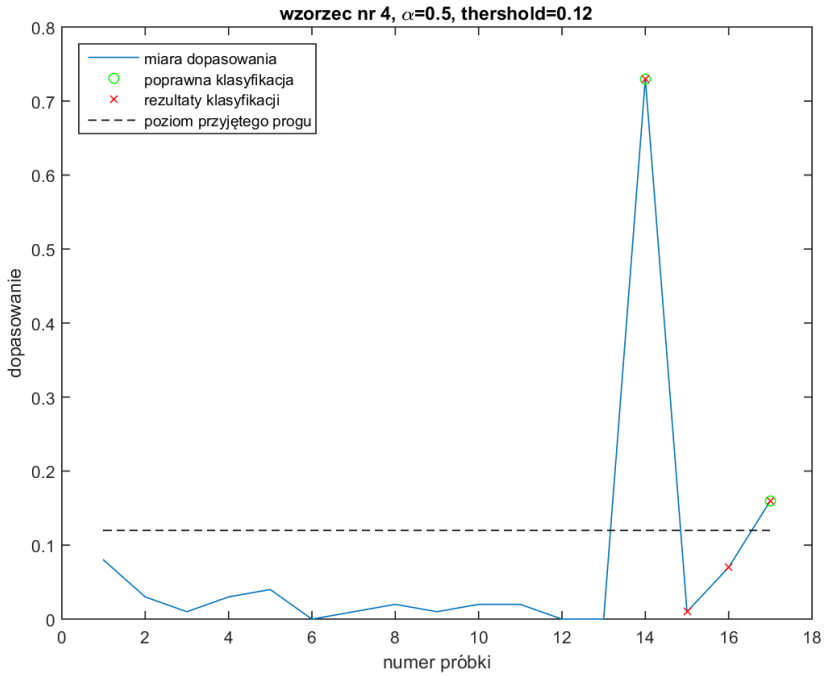
Współczynnik α jest współczynnikiem elementów wykrytych poprawnie do wszystkich, które powinny były być wykryte, a *threshold* jest progiem.

3.7 Wyniki algorytmu klasyfikacji









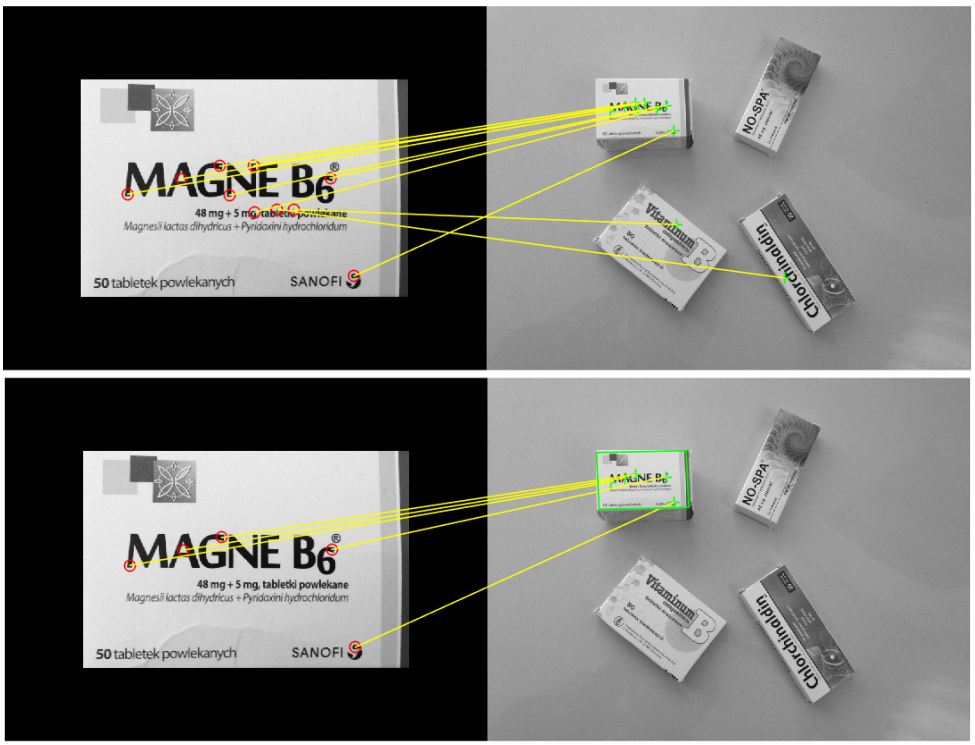
4 Rozpoznawanie na obrazie z wieloma elementami

Rozpoznawanie zostało tu dokonane dokładnie tak jak w przypadku poprzedniego ćwiczenia. Po wynikach można zauważyć, że punkty są porozrzucane po kilku obiektach, a często mało z nich zostało wykrytych. Po zastosowaniu algorytmu RANSAC w dwóch przypadkach udało się poprawnie zlokalizować obiekt, a w dwóch nie zadziałał (w jednym z powodu złego dobrania punktów, w drugim z powodu zbyt małej liczby wykrytych punktów).

```
[tform, inlierObjPoints, inlierScenePoints] = ...
    estimateGeometricTransform(matchedObjPoints,...
                               matchedScenePoints,...
                               'affine');
```

```
figure;
showMatchedFeatures(objImage, sceneImage, inlierObjPoints, ...
    inlierScenePoints, 'montage');

hold on;
[height, width] = size(objImage);
objPolygon = [
    1, 1;
    1, height;
    width, height;
    width, 1
];

[height, width] = size(sceneImage);
newObjPolygon = transformPointsForward(tform, objPolygon);
newObjPolygon = newObjPolygon';
newObjPolygon = [newObjPolygon, newObjPolygon(:, 1)];

plot(newObjPolygon(1,:) + width, newObjPolygon(2,:), 'g');
print(['third/second_' num2str(wzorzecNr)], '-dpng')
```

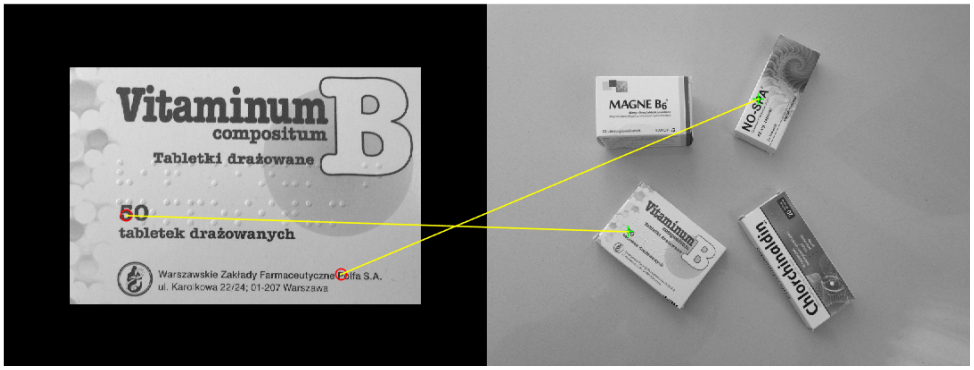
4.1 Wyniki



Rysunek 1: MAGNE B6 – poprawne wykrycie

Rysunek 2: NO-SPA – błędne wykrycie

Rysunek 3: Chlorochinaldin – poprawne wykrycie mimo obrotu



Rysunek 4: Vitaminum B – za mało punktów do przeprowadzenia RANSAC