Studenci: Michał Leszczyński, Marek Kiełtyka

Numery albumów: 297883, 297870

Rok studiów: trzeci

Kierunek: Informatyka Stosowana

BINARYZACJA

Dokumentacja projektu zrealizowanego w ramach ćwiczeń projektowych przedmiotu „Analiza Obrazów”

1. Opis działania i przeznaczenia projektu

Projekt został zrealizowany w formie aplikacji programu Matlab z graficznym interfejsem użytkownika. Jej głównym zadaniem jest wczytywanie i progowanie (*thresholding*) obrazów. Procesowanie danych wejściowych odbywa się na odpowiednich, sąsiadujących ze sobą pulpitach. Po załadowaniu obrazu użytkownik może za pomocą przycisków dokonać binaryzacji z progiem:

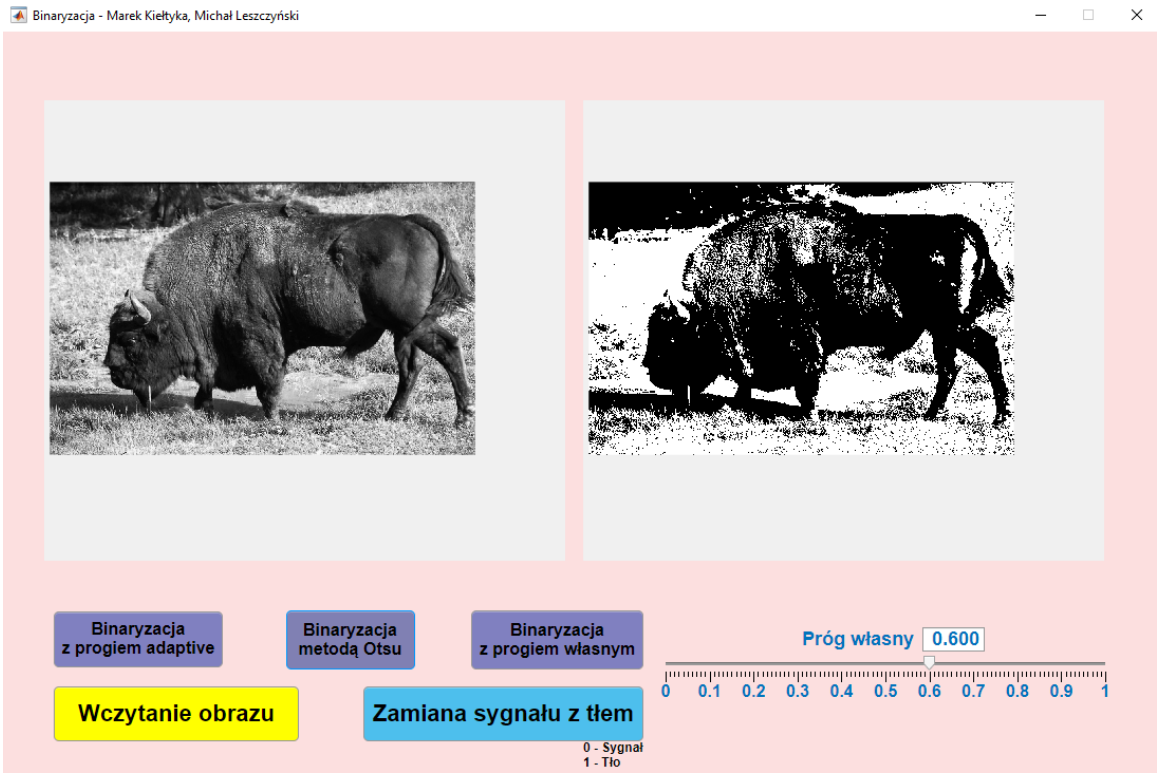
- ustalonym przez niego (należy odpowiednio ustawić suwak lub bezpośrednio wprowadzić wartość w okienku nad suwakiem)
- wyznaczanym metodą Otsu
- wyznaczanym metodą Adaptive

Dodatkową funkcjonalnością jest możliwość zamiany kolorów tła i sygnału na obrazie, niezależnie od użytej metody. Obraz musi najpierw zostać wczytany oraz zbinaryzowany. Aplikacja została zabezpieczona przed niepoprawną kolejnością czynności wykonywanych przez użytkownika. W momencie zaistnienia takiego przypadku program wyświetla stosowne komunikaty.

Aby uruchomić program należy wczytać plik *projekt.mlapp* do App Designera załączony do projektu (działanie zostało sprawdzone na wersji R2018b Matlaba).

Program wczytuje obrazy z rozszerzeniami *.jpg, .png, .jpeg*.

Rysunek 1. Wygląd aplikacji po starcie



Rysunek 2. Przykładowe działanie programu

2. Zastosowane algorytmy

- **Metoda Otsu**

Metoda opiera się na minimalizacji sumy ważonej wariancji wewnątrzklasowej (co jest równoważne z maksymalizacją wariancji międzyklasowej) dwóch klas (sygnału i tła). Metoda jest szczególnie efektywna, gdy liczby pikseli tła i sygnału są do siebie zbliżone. Algorytm wylicza wartości wariancji dla różnych progów binaryzacji i wybiera ten, dla którego jest ona najmniejsza.

```
function result = binarizeWithOtsu(~, image)
    tmpImage = image;
    curr_k = -1;
    curr_W_w = 1000;
    N = numel(tmpImage);

    for k = 0.01:0.02:1

        below = tmpImage(tmpImage<k);
        above = tmpImage(tmpImage>k);


        N_0 = numel(below);
        N_1 = numel(above);

        S_0 = sum(below);
        S_1 = sum(above);

        X_0 = S_0/N_0; % średnia jasność piksela w klasie
        X_1 = S_1/N_1;

        W_0 = (sum((below - X_0).^2)) / N_0;
        W_1 = (sum((above - X_1).^2)) / N_1;
        W_w = (N_0/N)*W_0 + (N_1/N)*W_1; % suma ważona wariancji wewnątrzklasowych

        if curr_k < 0 || W_w < curr_W_w
            curr_W_w = W_w;
            curr_k = k;
        end
    end

    tmpImage(tmpImage > curr_k) = 1;
    tmpImage(tmpImage <= curr_k) = 0;

    result = tmpImage;
end
```

Rysunek 3. Kod metody Otsu

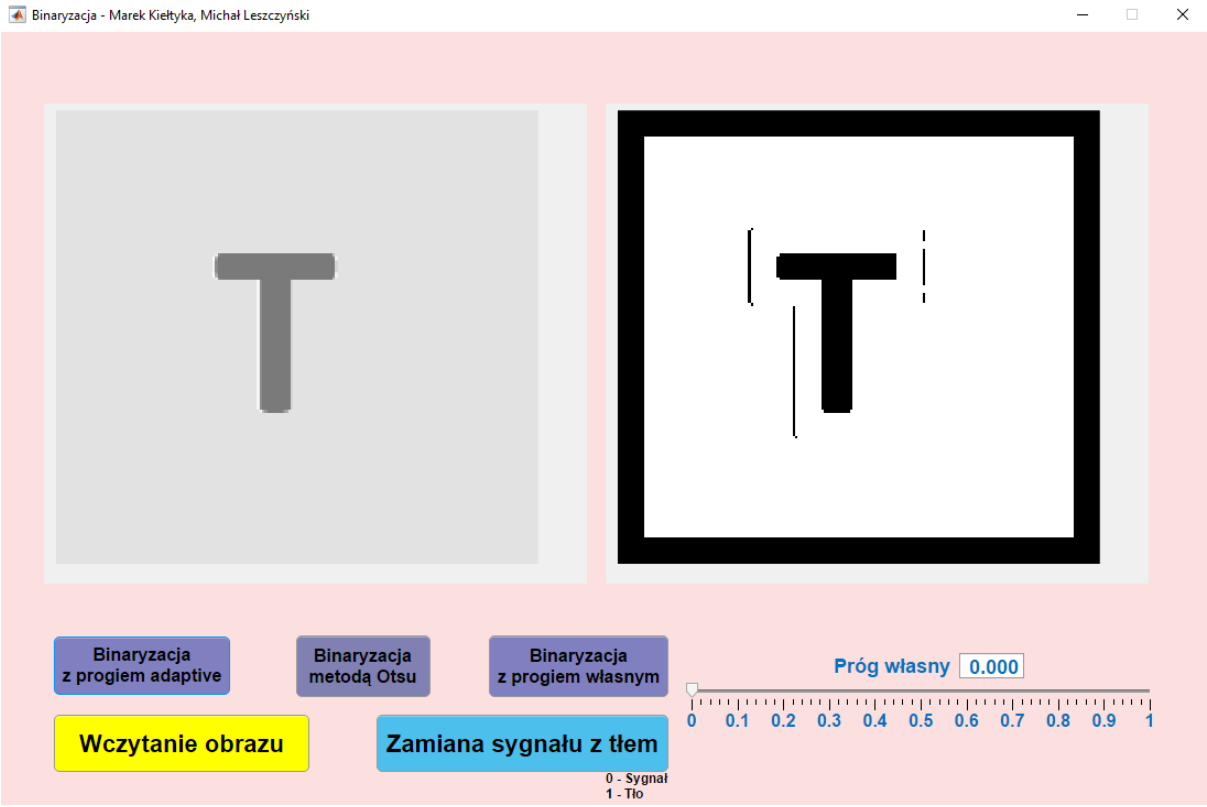
- **Adaptive**

Idea algorytmu opiera się na lokalnej analizie obrazu (fragment w kształcie kwadratu, dla którego w każdej iteracji obliczana jest średnia progu sąsiadujących pikseli). Okno fragmentu jest przesuwane kolumnami, zaś w momencie dotarcia do prawej krawędzi rysunku, obniża się o jeden wiersz pikseli w dół i ponownie iteruje od lewej do prawej. Uzyskuje się zadowalające rezultaty zarówno dla nieregularnych kształtów obiektów, jak i zbliżonych do foremnych figur geometrycznych.

```
function result = binarizeWithAdaptive(~, image)

    N = 2*floor(size(image)/16)+1; % wartosc domyslna dla matlaba
    [h, w, ~] = size(image);
    result = zeros(h, w);
    N2 = floor(N/2);

    for i=1+N2:h-N2
        for j=1+N2 : w-N2
            image2 = image(i-N2:i+N2 , j-N2:j+N2);
            threshold = mean(mean(image2));

            if image(i, j) > threshold
                result(i,j) = 1;
            else
                result(i,j) = 0;
            end
        end
    end
end
```

Rysunek 4. Kod metody ‘adaptive’

Rysunek 5. Metoda progowania ‘adaptive’ dla obiektów o nieregularnych kształtach



Rysunek 6. Metoda progowania ‘adaptive’ dla obiektów o regularnych kształtach

3. Wady projektu

Przy algorytmie ‘adaptive’ dla większych obrazów (krawędź o długości tysiąca pikseli i więcej) czas oczekiwania na wynik jest odczuwalnie dłuższy.

4. Podział obowiązków

- Projekt oraz wykonanie GUI, binaryzacja metodą adaptive i z progiem użytkownika, dokumentacja – Marek Kiełtyka
- Binaryzacja metodą Otsu oraz metodą adaptive, dokumentacja – Michał Leszczyński

5. Ewentualne usprawnienia

W przyszłości można by dokonać optymalizacji obliczania średniego progu dla sąsiadujących wartości pikseli, co usprawniłoby procesowanie obrazów o dużych rozmiarach. Ponadto ciekawym dla użytkownika usprawnieniem byłoby dynamiczne progowanie obrazu w zależności od chwilowego położenia suwaka (wyeliminowanie konieczności klikania przycisku).