Sprawozdanie z przedmiotu widzenie komputerowe i rozumienie obrazów

# Progowanie, segmentacja semantyczna

Autor: Marek Sigmund

# Wstęp

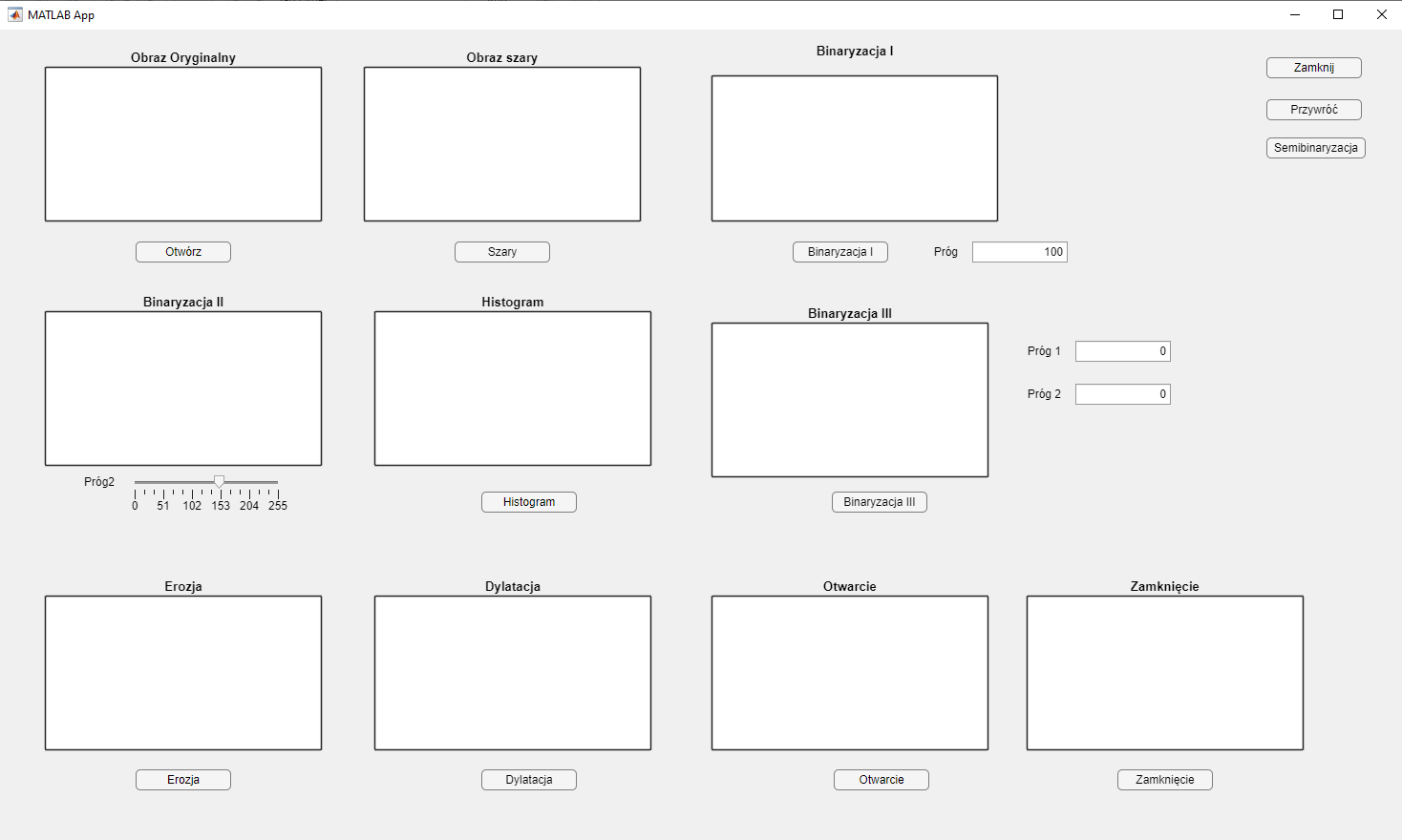
Obróbka i segmentacja obrazów to podstawowe techniki stosowane w cyfrowym przetwarzaniu obrazów, umożliwiające ekstrakcję istotnych informacji z danych wizualnych. Proces ten pozwala na podział obrazu na obszary odpowiadające określonym cechom, co jest niezbędne w zadaniach takich jak rozpoznawanie obiektów, analiza tekstur czy klasyfikacja obszarów.

W ramach niniejszego laboratorium skoncentrowano się na implementacji operacji binarnej segmentacji obrazów oraz klasycznych morfologicznych przekształceń, takich jak erozja, dylatacja, otwarcie czy zamknięcie. Zastosowano również techniki analizy histogramów oraz binaryzacji adaptacyjnej w celu podkreślenia różnic między metodami. Przygotowana aplikacja umożliwia użytkownikowi interaktywną analizę przetwarzania obrazu, co pozwala na praktyczne zrozumienie omawianych technik.

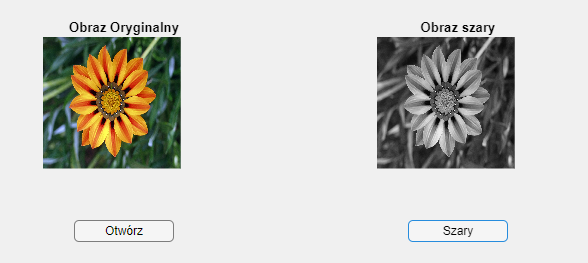
# Przebieg zadań

Pierwszym krokiem było przygotowanie interfejsu graficznego w MATLAB App Designer, obejmującego przyciski funkcyjne oraz panele Axes do wyświetlania obrazów. Zaimplementowano podstawowe funkcje, takie jak zamykanie aplikacji, wczytywanie obrazu kolorowego oraz jego konwersję do skali szarości, wyświetlaną w dedykowanych panelach Axes.

W kolejnym etapie aplikacja została wzbogacona o mechanizm modyfikacji aktywnego obrazu, umożliwiający wprowadzanie kolejnych zmian. Dodano również przycisk „Przywróć”, pozwalający na powrót do pierwotnych wersji obrazu kolorowego i szarego, które są wyświetlane w odpowiednich panelach interfejsu.



*Początkowy wygląd aplikacji po uruchomieniu.*



*Wczytany obraz oryginalny oraz obraz w skali szarości wygenerowany po naciśnięciu odpowiednich przycisków.*

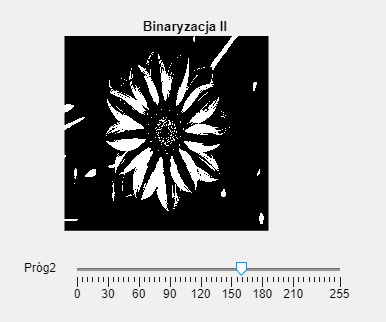
W kolejnym etapie zrealizowano funkcjonalność binaryzacji obrazu w skali szarości. Dodano przycisk „Binaryzacja I” oraz pole „Prog” umożliwiające ustawienie wartości progu w zakresie od 0 do 255, z domyślną wartością 100. Po naciśnięciu przycisku obraz w skali szarości zostaje przekształcony na obraz binarny, gdzie piksele o intensywności wyższej niż zadany próg przyjmują wartość 1 (biały), a pozostałe wartość 0 (czarny).



*Wynik binaryzacji obrazu w skali szarości przy domyślnym progu wynoszącym 100*

Kolejnym krokiem w realizacji aplikacji było dodanie funkcji binaryzacji obrazu w skali szarości z możliwością dynamicznej zmiany progu. W tym celu w interfejsie użytkownika umieszczono obiekt typu „Slider” o nazwie „Próg II”. Domyślną wartość suwaka ustawiono na 150, z zakresem od 0 do 255, umożliwiając użytkownikowi wygodne dostosowywanie progu binaryzacji.

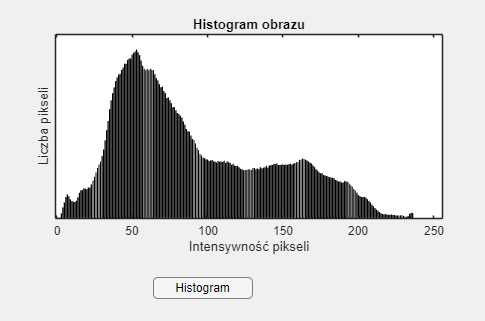
Zastosowano funkcję obsługującą dynamiczne zmiany wartości suwaka za pomocą opcji „ValueChangingFcn”, co pozwala na natychmiastowe aktualizowanie wyniku binaryzacji w trakcie przesuwania suwaka. Wynik binaryzacji obrazu na podstawie ustawionego progu jest wyświetlany w panelu Axes, umożliwiając użytkownikowi obserwację wpływu zmian progu w czasie rzeczywistym.



*Dynamiczny wynik binaryzacji obrazu w skali szarości na podstawie wartości progu ustawianego za pomocą suwaka*

W kolejnym etapie dodano funkcję generowania histogramu obrazu w skali szarości. W interfejsie użytkownika umieszczono przycisk „Histogram”, którego zadaniem jest obliczenie oraz wyświetlenie histogramu.

Do obliczenia wartości histogramu wykorzystano funkcję imhist, która analizuje obraz w skali szarości, dostarczając liczbę pikseli dla poszczególnych poziomów intensywności. Wynik został przedstawiony w formie wykresu, gdzie oś X reprezentuje intensywność pikseli, a oś Y liczbę pikseli dla danego poziomu intensywności.



*Wykres histogramu obrazu w skali szarości*

Kolejnym krokiem w realizacji było wprowadzenie funkcjonalności umożliwiającej wykonanie binaryzacji z dwoma progami. Dodano dwa pola tekstowe umożliwiające wprowadzenie wartości progów („Prog 1” i „Prog 2”) oraz przycisk „Binaryzacja III”.

Proces binaryzacji przebiega w kilku etapach. Najpierw wykonano binaryzację dla wartości „Prog 1”, a wynik wyświetlono w trzecim panelu Axes. Następnie przeprowadzono binaryzację dla „Prog 2”, której wynik ukazano w czwartym panelu Axes. Ostatecznie połączono oba wyniki za pomocą operacji logicznej AND, stosując funkcję „bitand”. Uzyskany wynik został przedstawiony w panelu Axes.



*Wynik binaryzacji z dwoma progami*

Binaryzacja obrazu to proces przekształcenia obrazu w skalę szarości do formy binarnej, gdzie piksele przyjmują jedną z dwóch wartości: czarną lub białą. W ramach realizacji zaimplementowano różne podejścia do binaryzacji, uwzględniając różne poziomy progów. Wykorzystanie jednego progu umożliwia prostą segmentację obrazu, podczas gdy zastosowanie dwóch progów pozwala na bardziej precyzyjne wydzielenie interesujących obszarów. Binaryzacja z progami oparta jest na analizie wartości pikseli w odniesieniu do wybranych progów, co pozwala na podział obrazu zgodnie z określonymi kryteriami. Te metody są szczególnie przydatne w zadaniach związanych z przetwarzaniem obrazu, takich jak segmentacja czy analiza kształtów.

W ramach kolejnych etapów realizacji, aplikacja została rozszerzona o funkcje umożliwiające podstawowe operacje morfologiczne na obrazie binarnym, uzyskanym we wcześniejszych krokach.

**Erozja:**

Dodano przycisk „Erozja”, którego zadaniem jest wykonanie operacji erozji. Operacja ta usuwa piksele z brzegów obiektów na obrazie binarnym, co pozwala na eliminację drobnych zakłóceń i wygładzenie krawędzi obiektów. Do realizacji tej operacji zastosowano funkcję imerode. Wynik operacji jest wyświetlany w panelu Axes.

**Dylatacja:**

Dodano przycisk „Dylatacja”, którego funkcją jest rozszerzanie obiektów na obrazie binarnym poprzez dodawanie pikseli do ich krawędzi. Operacja ta jest często stosowana do wypełniania luk w obiektach lub połączenia elementów. Do implementacji wykorzystano funkcję imdilate. Wynik jest prezentowany w kolejnym panelu Axes.



*Erozja i Dylatacja obrazu binarnego*

**Otwarcie:**

Przy pomocy przycisku „Otwarcie” zaimplementowano operację otwarcia, która łączy erozję i dylatację w celu usunięcia drobnych zakłóceń bez wpływu na kształt większych obiektów. Operację wykonano za pomocą funkcji imopen, a wynik jest wyświetlany w dedykowanym panelu Axes.

**Zamknięcie:**

Funkcja „Zamknięcie” umożliwia wykonanie operacji domykania, która składa się z dylatacji, a następnie erozji. Operacja ta pozwala na wypełnienie drobnych dziur w obiektach oraz wygładzenie krawędzi. W implementacji wykorzystano funkcję imclose, a obraz wynikowy prezentowany jest w panelu Axes.



*Otwarcie i zamknięcie obrazu binarnego*

Ostatnim krokiem w realizacji było zaimplementowanie funkcji semibinaryzacji. Dodano przycisk „Semibinaryzacja”, który umożliwia przetwarzanie oryginalnego obrazu kolorowego oraz jego wersji w skali szarości przy użyciu binarnej maski uzyskanej w poprzednich etapach.

* W przypadku obrazu kolorowego funkcja wykonuje operację mnożenia maski binarnej przez każdą składową koloru obrazu (R, G, B), co pozwala na zachowanie jedynie tych fragmentów obrazu, które odpowiadają aktywnym pikselom maski.
* Dla obrazu w skali szarości stosuje się podobną operację, ograniczając wartości intensywności pikseli do obszarów zdefiniowanych przez maskę binarną.

Operacja ta pozwala na wizualizację fragmentów obrazu określonych wcześniej binarną maską, co znajduje zastosowanie w zadaniach takich jak wyodrębnianie regionów zainteresowania czy segmentacja obrazu.



*Semibinaryzacja obrazu kolorowego i w skali szarości*

# Podsumowanie

Aplikacja zrealizowana w środowisku MATLAB pozwala na kompleksowe przetwarzanie obrazów z wykorzystaniem zaawansowanych narzędzi analizy i modyfikacji. Wprowadzone funkcje, takie jak konwersja do skali szarości, różne rodzaje binaryzacji, operacje morfologiczne oraz semibinaryzacja, umożliwiły efektywne przetwarzanie obrazów w celu wydobycia istotnych informacji i segmentacji kluczowych obszarów.

Aplikacja wykazała, że połączenie metod binaryzacji z operacjami morfologicznymi pozwala na znaczne zwiększenie precyzji w przetwarzaniu obrazów binarnych, co ma kluczowe znaczenie w takich dziedzinach jak analiza kształtów, wykrywanie obiektów czy filtrowanie szumów. W szczególności, histogram obrazu okazał się przydatnym narzędziem wspomagającym dobór progów binaryzacji, co pozytywnie wpłynęło na jakość uzyskanych wyników.

Wnioski z realizacji projektu wskazują, że integracja różnych metod przetwarzania obrazu w jednej aplikacji umożliwia szerokie zastosowanie w analizie wizualnej, od segmentacji obiektów po oczyszczanie i modyfikację obrazów. Opracowana aplikacja może znaleźć zastosowanie w systemach wizyjnych, medycynie, robotyce czy w analizie obrazów satelitarnych. Projekt ten wykazał również potencjał dalszego rozwijania aplikacji poprzez implementację bardziej zaawansowanych algorytmów lub automatyzację doboru parametrów, co czyni ją elastycznym i praktycznym narzędziem w analizie obrazów.

# Kod źródłowy

classdef Lab05 < matlab.apps.AppBase

% Properties that correspond to app components

properties (Access = public)

UIFigure matlab.ui.Figure

Prg2EditField matlab.ui.control.NumericEditField

Prg2EditFieldLabel matlab.ui.control.Label

Prg1EditField matlab.ui.control.NumericEditField

Prg1EditFieldLabel matlab.ui.control.Label

Prg2Slider matlab.ui.control.Slider

Prg2SliderLabel matlab.ui.control.Label

PrgEditField matlab.ui.control.NumericEditField

PrgEditFieldLabel matlab.ui.control.Label

ZamknicieButton matlab.ui.control.Button

DylatacjaButton matlab.ui.control.Button

ErozjaButton matlab.ui.control.Button

OtwarcieButton matlab.ui.control.Button

SemibinaryzacjaButton matlab.ui.control.Button

BinaryzacjaIIIButton matlab.ui.control.Button

HistogramButton matlab.ui.control.Button

BinaryzacjaIButton matlab.ui.control.Button

PrzywrButton matlab.ui.control.Button

SzaryButton matlab.ui.control.Button

OtwrzButton matlab.ui.control.Button

ZamknijButton matlab.ui.control.Button

UIAxes10 matlab.ui.control.UIAxes

UIAxes9 matlab.ui.control.UIAxes

UIAxes8 matlab.ui.control.UIAxes

UIAxes7 matlab.ui.control.UIAxes

UIAxes6 matlab.ui.control.UIAxes

UIAxes5 matlab.ui.control.UIAxes

UIAxes4 matlab.ui.control.UIAxes

UIAxes3 matlab.ui.control.UIAxes

UIAxes2 matlab.ui.control.UIAxes

UIAxes matlab.ui.control.UIAxes

end

properties (Access = private)

imgColorOriginal % Oryginalny obraz kolorowy

imgGrayOriginal % Oryginalny obraz w skali szarości

imgActive % Aktywny obraz (zmieniany przez kolejne funkcje)

imgBinary % Wynik binaryzacji

end

% Callbacks that handle component events

methods (Access = private)

% Button pushed function: ZamknijButton

function ZamknijButtonPushed(app, event)

delete(app.UIFigure);

end

% Button pushed function: OtwrzButton

function OtwrzButtonPushed(app, event)

[file, path] = uigetfile({'\*.jpg;\*.png'}, 'Wybierz obraz');

if isequal(file, 0)

return;

end

imgPath = fullfile(path, file);

app.imgColorOriginal = imread(imgPath);

app.imgGrayOriginal = rgb2gray(app.imgColorOriginal);

app.imgActive = app.imgGrayOriginal;

imshow(app.imgColorOriginal, 'Parent', app.UIAxes);

end

% Button pushed function: SzaryButton

function SzaryButtonPushed(app, event)

if isempty(app.imgColorOriginal)

uialert(app.UIFigure, 'Najpierw wczytaj obraz!', 'Błąd');

return;

end

app.imgActive = rgb2gray(app.imgColorOriginal);

imshow(app.imgActive, 'Parent', app.UIAxes2);

end

% Button pushed function: PrzywrButton

function PrzywrButtonPushed(app, event)

if isempty(app.imgColorOriginal)

uialert(app.UIFigure, 'Najpierw wczytaj obraz!', 'Błąd');

return;

end

app.imgActive = app.imgGrayOriginal;

imshow(app.imgColorOriginal, 'Parent', app.UIAxes);

imshow(app.imgGrayOriginal, 'Parent', app.UIAxes2);

end

% Button pushed function: BinaryzacjaIButton

function BinaryzacjaIButtonPushed(app, event)

if isempty(app.imgActive)

uialert(app.UIFigure, 'Najpierw wczytaj obraz i przekonwertuj go do skali szarości!', 'Błąd');

return;

end

threshold = app.PrgEditField.Value;

app.imgBinary = app.imgActive > threshold;

imshow(app.imgBinary, 'Parent', app.UIAxes3);

end

% Value changed function: Prg2Slider

function Prg2SliderValueChanged(app, event)

% Pobranie aktualnej wartości suwaka

threshold = event.Value;

if isempty(app.imgActive)

uialert(app.UIFigure, 'Najpierw wczytaj obraz i przekonwertuj go do skali szarości!', 'Błąd');

return;

end

app.imgBinary = app.imgActive > threshold;

imshow(app.imgBinary, 'Parent', app.UIAxes4);

end

% Button pushed function: HistogramButton

function HistogramButtonPushed(app, event)

if isempty(app.imgActive)

uialert(app.UIFigure, 'Najpierw wczytaj obraz i przekonwertuj go do skali szarości!', 'Błąd');

return;

end

[counts, binLocations] = imhist(app.imgActive);

cla(app.UIAxes5);

% Rysowanie histogramu w UIAxes5

bar(app.UIAxes5, binLocations, counts, 'FaceColor', 'black');

title(app.UIAxes5, 'Histogram obrazu');

xlabel(app.UIAxes5, 'Intensywność pikseli');

ylabel(app.UIAxes5, 'Liczba pikseli');

end

% Button pushed function: BinaryzacjaIIIButton

function BinaryzacjaIIIButtonPushed(app, event)

if isempty(app.imgActive)

uialert(app.UIFigure, 'Najpierw wczytaj obraz i przekonwertuj go do skali szarości!', 'Błąd');

return;

end

% Pobranie wartości progów z pól edycji

prog1 = app.Prg1EditField.Value;

prog2 = app.Prg2EditField.Value;

% Sprawdzenie poprawności progów

if prog1 < 0 || prog1 > 255 || prog2 < 0 || prog2 > 255

uialert(app.UIFigure, 'Progi muszą być w zakresie od 0 do 255!', 'Błąd');

return;

end

% Binaryzacja dla Progu 1

binary1 = app.imgActive > prog1;

%imshow(binary1, 'Parent', app.UIAxes3); % Wyświetlenie wyniku w UIAxes3

% Binaryzacja dla Progu 2

binary2 = app.imgActive > prog2;

%imshow(binary2, 'Parent', app.UIAxes4); % Wyświetlenie wyniku w UIAxes4

% Połączenie wyników binaryzacji przy użyciu operacji AND

combinedBinary = bitand(binary1, binary2);

imshow(combinedBinary, 'Parent', app.UIAxes6); % Wyświetlenie wyniku w UIAxes6

end

% Button pushed function: ErozjaButton

function ErozjaButtonPushed(app, event)

% Sprawdzenie, czy obraz binarny istnieje

if isempty(app.imgBinary)

uialert(app.UIFigure, 'Najpierw wykonaj binaryzację obrazu!', 'Błąd');

return;

end

se = strel('disk', 2);

% Wykonanie operacji erozji

erodedImage = imerode(app.imgBinary, se);

cla(app.UIAxes7);

imshow(erodedImage, 'Parent', app.UIAxes7);

title(app.UIAxes7, 'Erozja obrazu');

end

% Button pushed function: DylatacjaButton

function DylatacjaButtonPushed(app, event)

if isempty(app.imgBinary)

uialert(app.UIFigure, 'Najpierw wykonaj binaryzację obrazu!', 'Błąd');

return;

end

se = strel('disk', 1);

dilatedImage = imdilate(app.imgBinary, se);

% Wyświetlenie wyniku w UIAxes8

cla(app.UIAxes8);

imshow(dilatedImage, 'Parent', app.UIAxes8);

title(app.UIAxes8, 'Dylatacja obrazu')

end

% Button pushed function: OtwarcieButton

function OtwarcieButtonPushed(app, event)

if isempty(app.imgBinary)

uialert(app.UIFigure, 'Najpierw wykonaj binaryzację obrazu!', 'Błąd');

return;

end

se = strel('disk', 1);

openedImage = imopen(app.imgBinary, se);

cla(app.UIAxes9);

imshow(openedImage, 'Parent', app.UIAxes9);

title(app.UIAxes9, 'Otwarcie obrazu')

end

% Button pushed function: ZamknicieButton

function ZamknicieButtonPushed(app, event)

if isempty(app.imgBinary)

uialert(app.UIFigure, 'Najpierw wykonaj binaryzację obrazu!', 'Błąd');

return;

end

se = strel('disk', 1);

closedImage = imclose(app.imgBinary, se);

cla(app.UIAxes10);

imshow(closedImage, 'Parent', app.UIAxes10);

title(app.UIAxes10, 'Zamknięcie obrazu');

end

% Button pushed function: SemibinaryzacjaButton

function SemibinaryzacjaButtonPushed(app, event)

% Sprawdzenie, czy obraz binarny istnieje

if isempty(app.imgBinary)

uialert(app.UIFigure, 'Najpierw wykonaj binaryzację obrazu!', 'Błąd');

return;

end

% Sprawdzenie, czy oryginalny obraz kolorowy istnieje

if isempty(app.imgColorOriginal)

uialert(app.UIFigure, 'Najpierw wczytaj obraz kolorowy!', 'Błąd');

return;

end

% Sprawdzenie, czy obraz w skali szarości istnieje

if isempty(app.imgGrayOriginal)

uialert(app.UIFigure, 'Najpierw wczytaj obraz i przekonwertuj go do skali szarości!', 'Błąd');

return;

end

% Mnożenie oryginalnego obrazu kolorowego przez binarny obraz

semiBinaryColor = app.imgColorOriginal .\* uint8(repmat(app.imgBinary, [1, 1, 3]));

% Mnożenie obrazu w skali szarości przez binarny obraz

semiBinaryGray = app.imgGrayOriginal .\* uint8(app.imgBinary);

% Wyświetlenie semibinaryzacji obrazu kolorowego w pierwszym Axes

cla(app.UIAxes); % Czyszczenie UIAxes

imshow(semiBinaryColor, 'Parent', app.UIAxes);

title(app.UIAxes, 'Semibinaryzacja - Kolorowy');

% Wyświetlenie semibinaryzacji obrazu w skali szarości w drugim Axes

cla(app.UIAxes2); % Czyszczenie UIAxes2

imshow(semiBinaryGray, 'Parent', app.UIAxes2);

title(app.UIAxes2, 'Semibinaryzacja - Skala szarości');

end

end

% Component initialization

methods (Access = private)

% Create UIFigure and components

function createComponents(app)

% Create UIFigure and hide until all components are created

app.UIFigure = uifigure('Visible', 'off');

app.UIFigure.Position = [100 100 1470 849];

app.UIFigure.Name = 'MATLAB App';

% Create UIAxes

app.UIAxes = uiaxes(app.UIFigure);

title(app.UIAxes, 'Obraz Oryginalny')

app.UIAxes.XTick = [];

app.UIAxes.YTick = [];

app.UIAxes.Box = 'on';

app.UIAxes.Position = [43 644 300 185];

% Create UIAxes2

app.UIAxes2 = uiaxes(app.UIFigure);

title(app.UIAxes2, 'Obraz szary')

app.UIAxes2.XTick = [];

app.UIAxes2.XTickLabel = '';

app.UIAxes2.YTick = [];

app.UIAxes2.YTickLabel = '';

app.UIAxes2.Box = 'on';

app.UIAxes2.Position = [377 644 300 185];

% Create UIAxes3

app.UIAxes3 = uiaxes(app.UIFigure);

title(app.UIAxes3, {'Binaryzacja I'; ''})

app.UIAxes3.XTick = [];

app.UIAxes3.YTick = [];

app.UIAxes3.Box = 'on';

app.UIAxes3.Position = [741 644 310 191];

% Create UIAxes4

app.UIAxes4 = uiaxes(app.UIFigure);

title(app.UIAxes4, 'Binaryzacja II')

app.UIAxes4.XTick = [];

app.UIAxes4.XTickLabel = '';

app.UIAxes4.YTick = [];

app.UIAxes4.YTickLabel = '';

app.UIAxes4.Box = 'on';

app.UIAxes4.Position = [43 388 300 185];

% Create UIAxes5

app.UIAxes5 = uiaxes(app.UIFigure);

title(app.UIAxes5, 'Histogram')

app.UIAxes5.XTick = [];

app.UIAxes5.YTick = [];

app.UIAxes5.Box = 'on';

app.UIAxes5.Position = [388 388 300 185];

% Create UIAxes6

app.UIAxes6 = uiaxes(app.UIFigure);

title(app.UIAxes6, 'Binaryzacja III')

app.UIAxes6.XTick = [];

app.UIAxes6.XTickLabel = '';

app.UIAxes6.YTick = [];

app.UIAxes6.YTickLabel = '';

app.UIAxes6.Box = 'on';

app.UIAxes6.Position = [741 376 300 185];

% Create UIAxes7

app.UIAxes7 = uiaxes(app.UIFigure);

title(app.UIAxes7, 'Erozja')

app.UIAxes7.XTick = [];

app.UIAxes7.YTick = [];

app.UIAxes7.Box = 'on';

app.UIAxes7.Position = [43 90 300 185];

% Create UIAxes8

app.UIAxes8 = uiaxes(app.UIFigure);

title(app.UIAxes8, 'Dylatacja')

app.UIAxes8.XTick = [];

app.UIAxes8.XTickLabel = '';

app.UIAxes8.YTick = [];

app.UIAxes8.YTickLabel = '';

app.UIAxes8.Box = 'on';

app.UIAxes8.Position = [388 90 300 185];

% Create UIAxes9

app.UIAxes9 = uiaxes(app.UIFigure);

title(app.UIAxes9, 'Otwarcie')

app.UIAxes9.XTick = [];

app.UIAxes9.YTick = [];

app.UIAxes9.Box = 'on';

app.UIAxes9.Position = [741 90 300 185];

% Create UIAxes10

app.UIAxes10 = uiaxes(app.UIFigure);

title(app.UIAxes10, 'Zamknięcie')

app.UIAxes10.XTick = [];

app.UIAxes10.XTickLabel = '';

app.UIAxes10.YTick = [];

app.UIAxes10.YTickLabel = '';

app.UIAxes10.Box = 'on';

app.UIAxes10.Position = [1071 90 300 185];

% Create ZamknijButton

app.ZamknijButton = uibutton(app.UIFigure, 'push');

app.ZamknijButton.ButtonPushedFcn = createCallbackFcn(app, @ZamknijButtonPushed, true);

app.ZamknijButton.Position = [1327 799 100 22];

app.ZamknijButton.Text = 'Zamknij';

% Create OtwrzButton

app.OtwrzButton = uibutton(app.UIFigure, 'push');

app.OtwrzButton.ButtonPushedFcn = createCallbackFcn(app, @OtwrzButtonPushed, true);

app.OtwrzButton.Position = [143 606 100 22];

app.OtwrzButton.Text = 'Otwórz';

% Create SzaryButton

app.SzaryButton = uibutton(app.UIFigure, 'push');

app.SzaryButton.ButtonPushedFcn = createCallbackFcn(app, @SzaryButtonPushed, true);

app.SzaryButton.Position = [477 606 100 22];

app.SzaryButton.Text = 'Szary';

% Create PrzywrButton

app.PrzywrButton = uibutton(app.UIFigure, 'push');

app.PrzywrButton.ButtonPushedFcn = createCallbackFcn(app, @PrzywrButtonPushed, true);

app.PrzywrButton.Position = [1327 755 100 22];

app.PrzywrButton.Text = 'Przywróć';

% Create BinaryzacjaIButton

app.BinaryzacjaIButton = uibutton(app.UIFigure, 'push');

app.BinaryzacjaIButton.ButtonPushedFcn = createCallbackFcn(app, @BinaryzacjaIButtonPushed, true);

app.BinaryzacjaIButton.Position = [831 606 100 22];

app.BinaryzacjaIButton.Text = 'Binaryzacja I';

% Create HistogramButton

app.HistogramButton = uibutton(app.UIFigure, 'push');

app.HistogramButton.ButtonPushedFcn = createCallbackFcn(app, @HistogramButtonPushed, true);

app.HistogramButton.Position = [505 344 100 22];

app.HistogramButton.Text = 'Histogram';

% Create BinaryzacjaIIIButton

app.BinaryzacjaIIIButton = uibutton(app.UIFigure, 'push');

app.BinaryzacjaIIIButton.ButtonPushedFcn = createCallbackFcn(app, @BinaryzacjaIIIButtonPushed, true);

app.BinaryzacjaIIIButton.Position = [872 344 100 22];

app.BinaryzacjaIIIButton.Text = 'Binaryzacja III';

% Create SemibinaryzacjaButton

app.SemibinaryzacjaButton = uibutton(app.UIFigure, 'push');

app.SemibinaryzacjaButton.ButtonPushedFcn = createCallbackFcn(app, @SemibinaryzacjaButtonPushed, true);

app.SemibinaryzacjaButton.Position = [1327 715 104 22];

app.SemibinaryzacjaButton.Text = 'Semibinaryzacja';

% Create OtwarcieButton

app.OtwarcieButton = uibutton(app.UIFigure, 'push');

app.OtwarcieButton.ButtonPushedFcn = createCallbackFcn(app, @OtwarcieButtonPushed, true);

app.OtwarcieButton.Position = [874 53 100 22];

app.OtwarcieButton.Text = 'Otwarcie';

% Create ErozjaButton

app.ErozjaButton = uibutton(app.UIFigure, 'push');

app.ErozjaButton.ButtonPushedFcn = createCallbackFcn(app, @ErozjaButtonPushed, true);

app.ErozjaButton.Position = [143 53 100 22];

app.ErozjaButton.Text = 'Erozja';

% Create DylatacjaButton

app.DylatacjaButton = uibutton(app.UIFigure, 'push');

app.DylatacjaButton.ButtonPushedFcn = createCallbackFcn(app, @DylatacjaButtonPushed, true);

app.DylatacjaButton.Position = [505 53 100 22];

app.DylatacjaButton.Text = 'Dylatacja';

% Create ZamknicieButton

app.ZamknicieButton = uibutton(app.UIFigure, 'push');

app.ZamknicieButton.ButtonPushedFcn = createCallbackFcn(app, @ZamknicieButtonPushed, true);

app.ZamknicieButton.Position = [1171 53 100 22];

app.ZamknicieButton.Text = 'Zamknięcie';

% Create PrgEditFieldLabel

app.PrgEditFieldLabel = uilabel(app.UIFigure);

app.PrgEditFieldLabel.HorizontalAlignment = 'right';

app.PrgEditFieldLabel.Position = [973 606 31 22];

app.PrgEditFieldLabel.Text = 'Próg';

% Create PrgEditField

app.PrgEditField = uieditfield(app.UIFigure, 'numeric');

app.PrgEditField.Limits = [0 255];

app.PrgEditField.Position = [1019 606 100 22];

app.PrgEditField.Value = 100;

% Create Prg2SliderLabel

app.Prg2SliderLabel = uilabel(app.UIFigure);

app.Prg2SliderLabel.HorizontalAlignment = 'right';

app.Prg2SliderLabel.Position = [83 365 38 22];

app.Prg2SliderLabel.Text = 'Próg2';

% Create Prg2Slider

app.Prg2Slider = uislider(app.UIFigure);

app.Prg2Slider.Limits = [0 255];

app.Prg2Slider.ValueChangedFcn = createCallbackFcn(app, @Prg2SliderValueChanged, true);

app.Prg2Slider.Position = [142 374 150 3];

app.Prg2Slider.Value = 150;

% Create Prg1EditFieldLabel

app.Prg1EditFieldLabel = uilabel(app.UIFigure);

app.Prg1EditFieldLabel.HorizontalAlignment = 'right';

app.Prg1EditFieldLabel.Position = [1071 502 41 22];

app.Prg1EditFieldLabel.Text = 'Próg 1';

% Create Prg1EditField

app.Prg1EditField = uieditfield(app.UIFigure, 'numeric');

app.Prg1EditField.Limits = [0 255];

app.Prg1EditField.Position = [1127 502 100 22];

% Create Prg2EditFieldLabel

app.Prg2EditFieldLabel = uilabel(app.UIFigure);

app.Prg2EditFieldLabel.HorizontalAlignment = 'right';

app.Prg2EditFieldLabel.Position = [1071 457 41 22];

app.Prg2EditFieldLabel.Text = 'Próg 2';

% Create Prg2EditField

app.Prg2EditField = uieditfield(app.UIFigure, 'numeric');

app.Prg2EditField.Limits = [0 255];

app.Prg2EditField.Position = [1127 457 100 22];

% Show the figure after all components are created

app.UIFigure.Visible = 'on';

end

end

% App creation and deletion

methods (Access = public)

% Construct app

function app = Lab05

% Create UIFigure and components

createComponents(app)

% Register the app with App Designer

registerApp(app, app.UIFigure)

if nargout == 0

clear app

end

end

% Code that executes before app deletion

function delete(app)

% Delete UIFigure when app is deleted

delete(app.UIFigure)

end

end

end