instrukcja obsługi programu

instrukcja obsługi

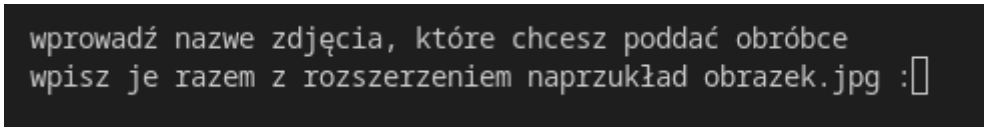
funkcje programu

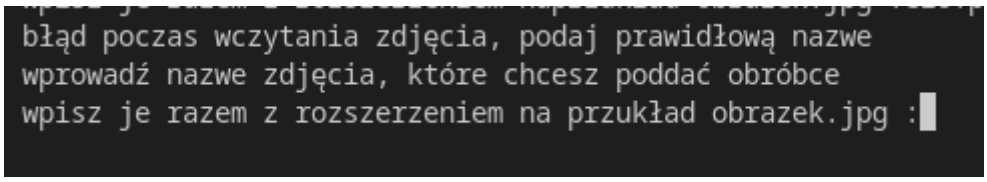
- Za pomocą programu mozna:
  - skalować dowolnie obrazek
  - zastosować dla niego operacje progowania
  - zastosować filtrykolorów: czerwonego, zielonego oraz niebieskiego
  - wygenerować histogram

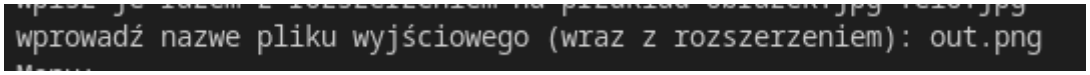
ogarniczenia:

- ograniczony dla plików
  - jpeg
  - png
  - bmp
  - psd
  - tga
  - gif
  - hdr
  - pnm,ppm,pgm
  - ale testy przeprowadzone tylko na png i jpg

wstęp

- orginalny plik zdjęcia nie zostanie zmodyfikowany
- należy wprowadzaić za pomocą klawiatury nazwę pliku który chcemy poddać obróbce i wcisnąć enter
  - 

  - jeśli nastpi błąd zostaniemy o tym powiadomieni oraz poproszeni o ponowne wprowadzenie nazwy pliku wejściowego
    - 

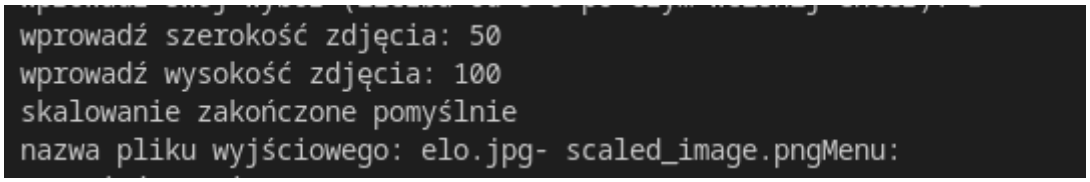
  - prawidłowe będą pliki w formacie jpg, png
- następnie zostaniemy poproszeni o podanie nazwy pliku wyjściowego (tak będzie się nazywał nasz plik po progowaniu) należy wprowadzić jego nazwe z klawiatury i klinąć enter
  - 

- jeśli użytkownik wprowadzi plik z błędnym rozszerzeniem należy zmienić go za pomocą opcji 8, program nie wyświetli błędu pomimo błędnego formatu pliku wyjściowego

wykorzystywanie programu

- możemy wybrać jedną z 9 dostepnych opcji:
  - zmień rozmiar
  - histogram
  - progowanie
  - filtr czerwonego
  - filtr zielonego
  - filtr niebieskiego
  - zmień nazwe pliku wejściowego
  - zmień nazwe pliku wyjściowego
  - wyjdź z programu
- aby wybrać którąś z opcji nalerzy wybrać numer na klawiaturze po czym wcisnąć enter

zmień rozmiar

- po wciśnięciu 1 na klawiaturze, a następnie kliknieciu enter zostaniemy poproszeni o podanie szerokości do jakiej chcemy zdjęcie przeskalować puźniej o jego wysokość
  - 

  - uzyskanie takiego komunikatu oznacza że operacja przebiegła pomyślnie
  - w przypdaku błędu wyświetlony zostanie komunikat: Błąd zapisu przeskalowanego obrazu

histogram

- po wybraniu 2 zostaną stworzone 3 pliki o nazwach:
  - histogram_red.png
  - histogram_green.png
  - histogram_blue.png
  - dla odpowiednich kanałów
- histogram jest to częstotliwosć z jaką wystepuje piksel danego koloru (przyjmujacego daną wartosc rgb)

progowanie

- po wybraniu 3 zostanie wykonane progowanie, o pomyślnej operacji poinformuje nas komunikat
  - 

- plik wyjściowy będzie miał nazwę taką jak podaliśmy na początku programu

filtry

- wybiermy
  - 4 dla filtra czerwonego
  - 5 dla filtru zielonego
  - 6 dla filtru niebieskiego
- zostaniemy powiadomieni o zapisie zdjęcia po dodaniu filtra
  - 

- jeśli nie jesteśmy zadowoleni z tego jak bardzo wyrany filtr został zastosowany wystarczy wybrać opcje 7 i zmienic nazwe pliku wejściowego na tą jaką program nam przed chwilą podł i ponownie zastosować operacje dodania filtra

zmiana pliku wejściowego

- po wciśnięciu 7 zostaniemy poproszeni o podanie nazwy nowego pliku wejściowego, proces przebiega identycznie jak na poczatku programu

zmiana pliku wyjściowego

- po wybraniu 8 zostaniemu poproszeni o podanie nazwy nowego pliku wyjściowego, proces przebiega identycznie jak na poczatku programu

wyjście

- wystarczy nacisnąć na klawiaturze 0 i klinąć enter a program zostanie zakończony

Dokumentacja

Opis programu

Program jest prostą aplikacją do obróbki obrazów, umożliwiającą zmiane rozmiaru, generowanie histogramu, progowanie oraz zastosowanie filtrów kolorów. Oprogramowanie korzysta z biblioteki `stb Image` do manipulacji obrazami.

biblioteka stb_image

- GitHub - nothings/stb: stb single-file public domain libraries for C/C++
- z tej biblioteki pochodzą pliki
  - stb_image_resize.h
    - ważna informacja: używana jest wersja pierwsza nie stb_image_resize2.h w wersji drógiej zaszła zmiana składni i program nie jest kompatybilny z nowszą wersją tej biblioteki
  - stb_image_wrtite.h
  - stb_image.h

Struktura programu

Zmienne globalne

`inputFilePath` (typ: string)

Przechowuje nazwę pliku wejściowego (obrazu), który ma zostać poddany obróbce.

`outputFilePath` (typ: string)

Przechowuje nazwę pliku wyjściowego, do którego zostaną zapisane przetworzone obrazy.

`width` (typ: int)

Przechowuje szerokość obrazu (ilość pikseli w jednym wierszu).

`height` (typ: int)

Przechowuje wysokość obrazu (ilość pikseli w jednej kolumnie).

`channels` (typ: int)

Przechowuje liczbę kanałów w obrazie (np. 3 dla obrazów RGB).

`originalImageData` (typ: unsigned char*)

Wskaźnik do danych obrazu wczytanego z pliku wejściowego. Przechowuje oryginalne dane obrazu przed wszelkimi operacjami.

`imageData` (typ: unsigned char*)

Wskaźnik do danych obrazu, na których wykonywane są operacje. Przechowuje kopię oryginalnych danych, która może być modyfikowana w trakcie działania programu.

rozdział na `orginalImageData` oraz `imageData` jest potżebny żeby filtry były zastosowywane tylko do pliku źródłowego oraz żeby nie mogły sie na siebie nakładać, jesli użytkownik chce nałożyć kolejny filtr powinien zmienić nazwe pliku wejściowego na plik wyjściowy filtrowania

`choice` (typ: int)

Przechowuje wybór użytkownika z menu głównego programu.

Funkcje

`thresholdImage`

```
void thresholdImage(unsigned char* imageData, int threshold, int width, int height, int channels);
```

Funkcja przetwarza dane obrazu, stosując progowanie dla każdego kanału obrazu. Ustawia wartości pikseli na 255, jeśli są większe niż określony próg, a w przeciwnym razie na 0, dla każdego kanału.

- `imageData` : Wskaźnik do tablicy danych obrazu (kanały ułożone naprzemiennie).
- `threshold` : Wartość progu do zastosowania dla każdego kanału.

- `width` : Szerokość obrazu.
- `height` : Wysokość obrazu.
- `channels` : Liczba kanałów w obrazie.

`generateHistogramImage`

```
void generateHistogramImage(const vector<int>& histogram, const string& outputPath);
```

Funkcja generuje obraz histogramu na podstawie danych histogramu i zapisuje go do pliku. Obraz zawiera słupki reprezentujące częstość występowania poszczególnych wartości pikseli.

- `histogram` : Wektor zawierający dane histogramu.
- `outputPath` : Ścieżka do pliku, do którego zostanie zapisany obraz histogramu.

`calculateHistogram`

```
void calculateHistogram(const unsigned char* image, int width, int height, int channels, vector<int>& histogram, const string& channelName);
```

Funkcja oblicza histogram dla danego obrazu i zapisuje go do pliku PNG. Histogram reprezentuje liczbę pikseli dla każdego poziomu intensywności.

- `image` : Wskaźnik do danych obrazu.
- `width` : Szerokość obrazu.
- `height` : Wysokość obrazu.
- `channels` : Liczba kanałów obrazu.
- `histogram` : Wektor przechowujący wartości histogramu.
- `channelName` : Nazwa kanału, dla którego jest tworzony histogram.

`scaleImage`

```
void scaleImage(const unsigned char* inputData, int inputWidth, int inputHeight, int channels, int outputWidth, int outputHeight, const string& outputFilePath);
```

Funkcja skaluje wejściowy obraz na podstawie podanych wymiarów wyjściowych i liczby kanałów, a następnie zapisuje przeskalowany obraz do pliku o podanej ścieżce.

- `inputData` : Wskaźnik do danych wejściowego obrazu.
- `inputWidth` : Szerokość wejściowego obrazu.
- `inputHeight` : Wysokość wejściowego obrazu.
- `channels` : Liczba kanałów wejściowego obrazu.
- `outputWidth` : Szerokość docelowego obrazu (po skalowaniu).
- `outputHeight` : Wysokość docelowego obrazu (po skalowaniu).
- `outputFilePath` : Ścieżka do pliku, do którego zostanie zapisany przeskalowany obraz.

`applyRedFilter`

```
void applyRedFilter(unsigned char* imageData, int width, int height, int channels, const string& outputFileName);
```

Funkcja stosuje filtr czerwony, zwiększając wartość czerwonej składowej pikseli o 50. Zapisuje przetworzony obraz pod nową nazwą.

- `imageData` : Wskaźnik do danych obrazu.
- `width` : Szerokość obrazu.
- `height` : Wysokość obrazu.
- `channels` : Liczba kanałów obrazu.
- `outputFileName` : Nazwa pliku wyjściowego dla przetworzonego obrazu.
- pozostałe filtry działają tak samo różnica jest tylko taka, że indexy sie zmieniają i dla kanału green +1 dla blue +2
  - `imageData[i] = min(255, imageData[i+1] + 50);` dla green
  - `imageData[i] = min(255, imageData[i+2] + 50);` dla blue