Obrazy 16-sto bitowe

Kamil Sudoł
Jakub Strugała
Patryk Śledź

14 czerwca 2020

1. Tytuł i autorzy projektu.

Tematem projektu są obrazy 16-sto bitowe. Jest to projekt o numerze 42.
Autorami projektu są:

- Kamil Sudoł
- Jakub Strugała
- Patryk Śledź

2. Opis projektu.

Celem projektu było napisanie programu umożliwiającego obsługę podstawowych operacji na plikach w formacie TIFF. Format ten pozwala zapisać informację w 16 bitach na kanał koloru. Aplikacja powinna umożliwiać otwarcie plików TIFF oraz pozwalać na kilka rodzajów konwersji.

3. Założenia wstępne przyjęte w realizacji projektu.

- Program powinien pozwalać na wczytanie oraz zapisanie pliku w formacie (.tiff) lub (.tif).
- Obraz wczytany powinien zostać wyświetlony po skompresowaniu do 8 bitów.
- Program powinien umożliwiać wybór jednej z trzech konwersji:

    a) konwersji z pełnym zakresem dynamicznym obrazu skompresowanego liniowo z 16 bitów na 8 bitów,
    b) konwersji z pełnym zakresem zdjęcia skompresowanym z uwzględnieniem krzywej gamma z 16 na 8 bitów oraz z regulacją parametru gamma,
    c) konwersji z przeniesieniem ośmiobitowej części informacji (bez kompresji) z obrazu szesnastobitowego z umożliwieniem wyboru bitów do przeniesienia.

- Program powinien wyświetlać histogramy jasności dla obrazu wczytanego i wyświetlanego oraz pozwalać na zapisanie wyświetlanego pliku w 8-bitowym formacie.

4. Analiza projektu.

Specyfikacja danych wejściowych

Stworzony przez nas program przetwarza pliki graficzne w formacie TIFF. Obrazy mogą zostać wczytane z dowolnej przestrzeni dyskowej poprzez okno dialogowe.

Opis oczekiwanych danych wyjściowych

Obrazy wczytane są wyświetlane na ekranie monitora.
W prawej dolnej części okna programu są wyświetlane histogramy jasności. Wraz ze zmianą parametru gamma bądź wybraniu bitów do przeniesienia podczas konwersji, jeden z histogramów jest przekształcany w czasie rzeczywistym.
Pliki graficzne po przeprowadzeniu wybranych operacji mogą zostać zapisane w formacie TIFF. Obrazy wyjściowe mają taką samą rozdzielczość, jak obrazy wejściowe.

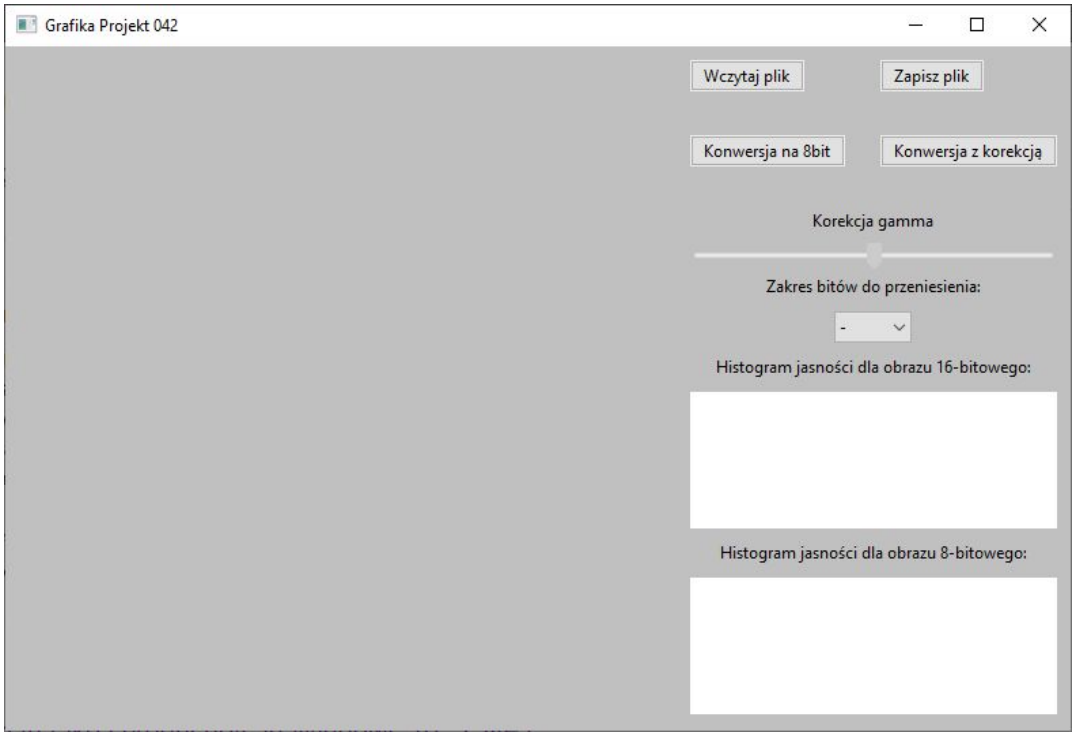
Zdefiniowanie struktur danych

Podczas działania programu, obrazy przechowywane są w obiektach:

- Klasy WxImage
- Klasy WxBitmap
- Tablicy unsigned char
- Klasy cv::Mat

W zależności od potrzeb, dokonywane są konwersje pomiędzy tymi typami.

Specyfikacja interfejsu użytkownika

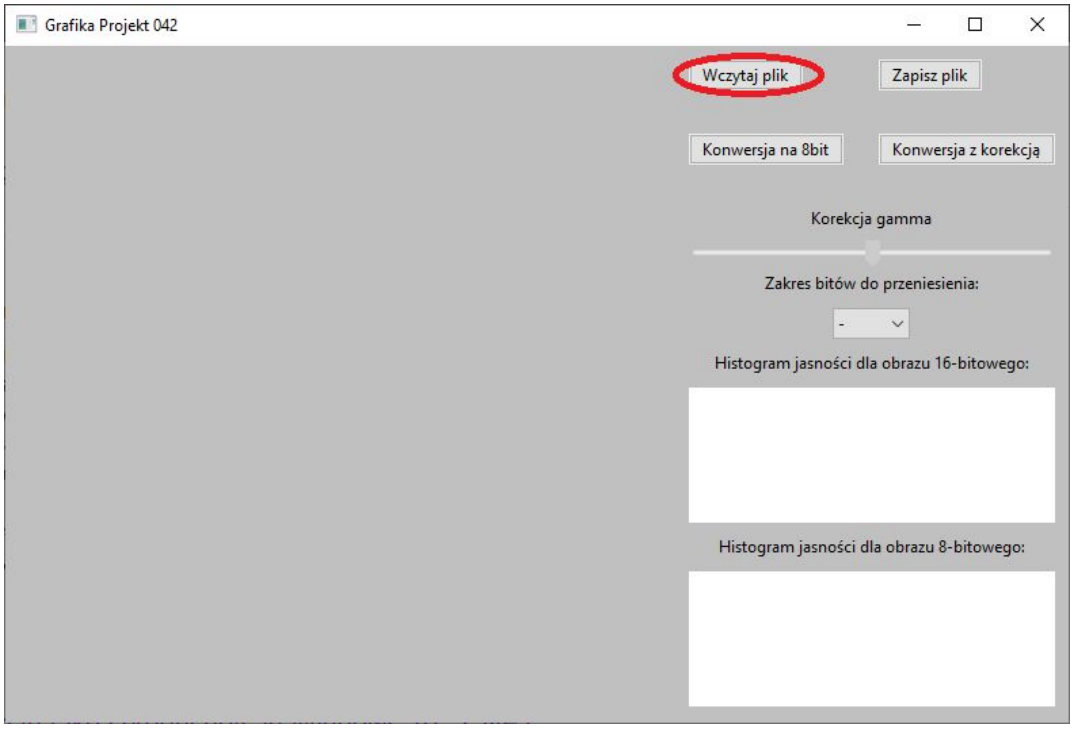


Interfejs składa się z następujących elementów:

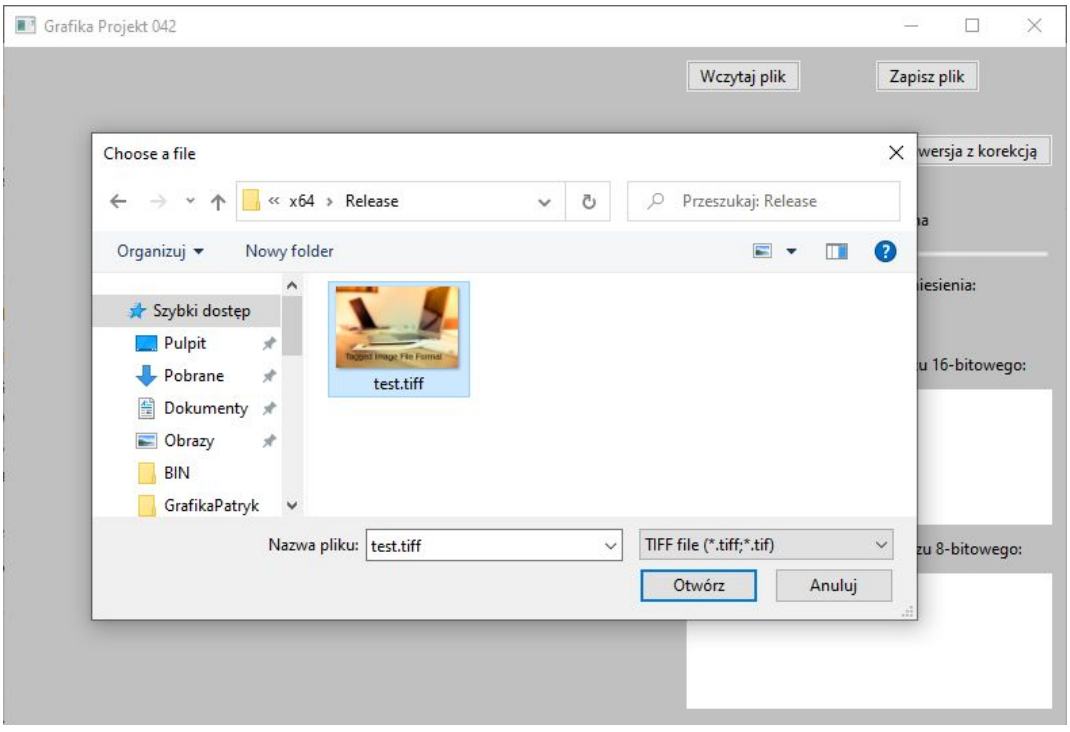
- głównego okna, w którym wyświetlany jest wczytany obraz,
- przycisków wykorzystywanych do wczytania/zapisania obrazu,
- przycisków wykorzystywanych do odpowiednich konwersji,
- paska przesuwania regulującego wartość parametru gamma,
- droplisty z wyborem danego zakresu bitu do przeniesienia,
- obszarów, na których wyświetlane zostają histogramy.

Odpowiednie przekształcenia możliwe do przeprowadzenia wykonywane są na wyświetlanym obrazie.
W prawym dolnym rogu wyświetlane są histogramy: pierwszy od góry dla obrazu wczytanego (jest on znormalizowany) oraz znajdujący się pod nim dla obrazu skonwertowanego na 8-bitowy. Histogramy aktualizują się wraz z wprowadzonymi zmianami. Użytkownik ma możliwość przeprowadzenia konwersji obrazu do 8bit skompresowanego liniowo, z uwzględnieniem krzywej gamma oraz bez kompresji ośmiobitowej części informacji z obrazu 16-bitowego (z możliwością wyboru zakresu bitów z droplisty).

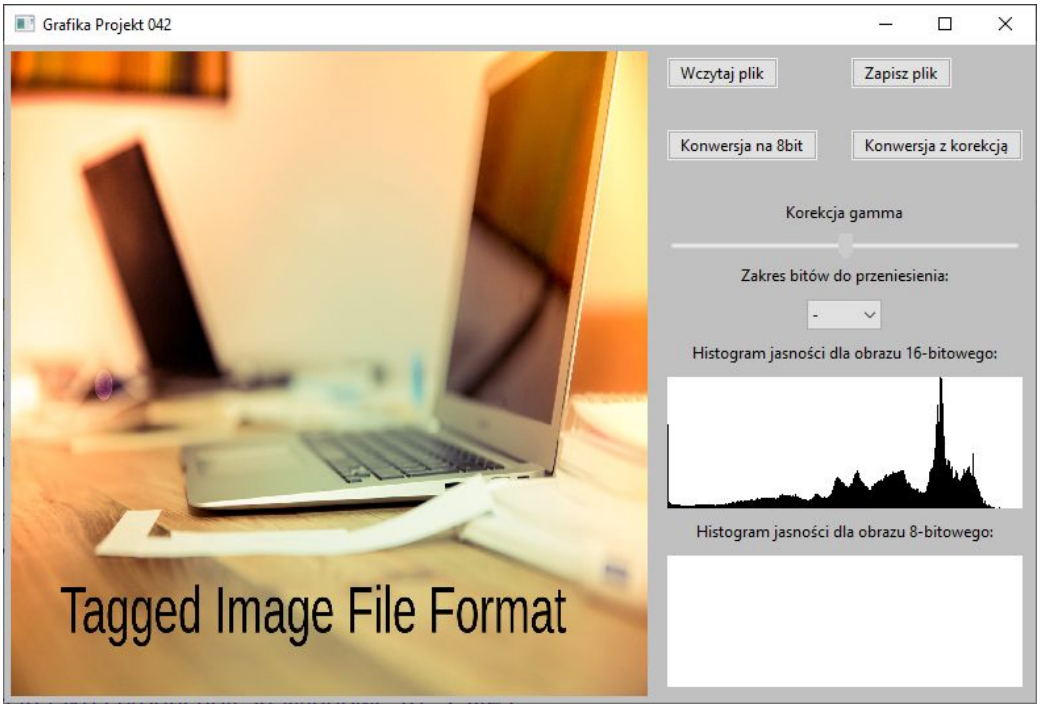
Krótka prezentacja możliwości aplikacji



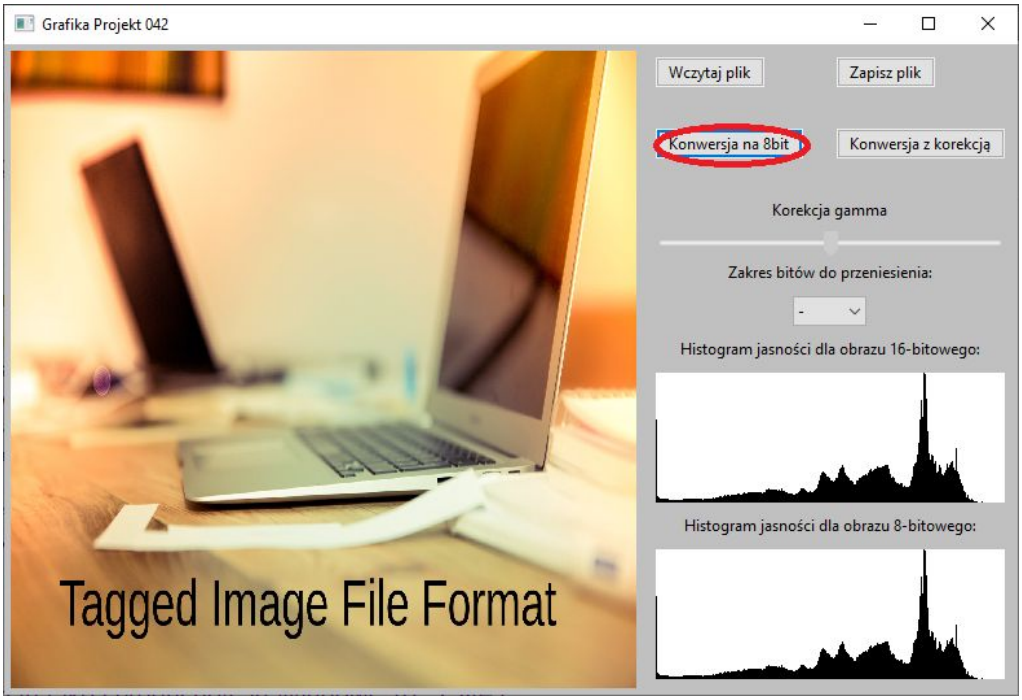
Na samym początku należy wczytać obraz w celu dalszych operacji. Po naciśnięciu przycisku ”Wczytaj plik” zostaje uruchomione okno dialogowe w celu wyboru pliku.



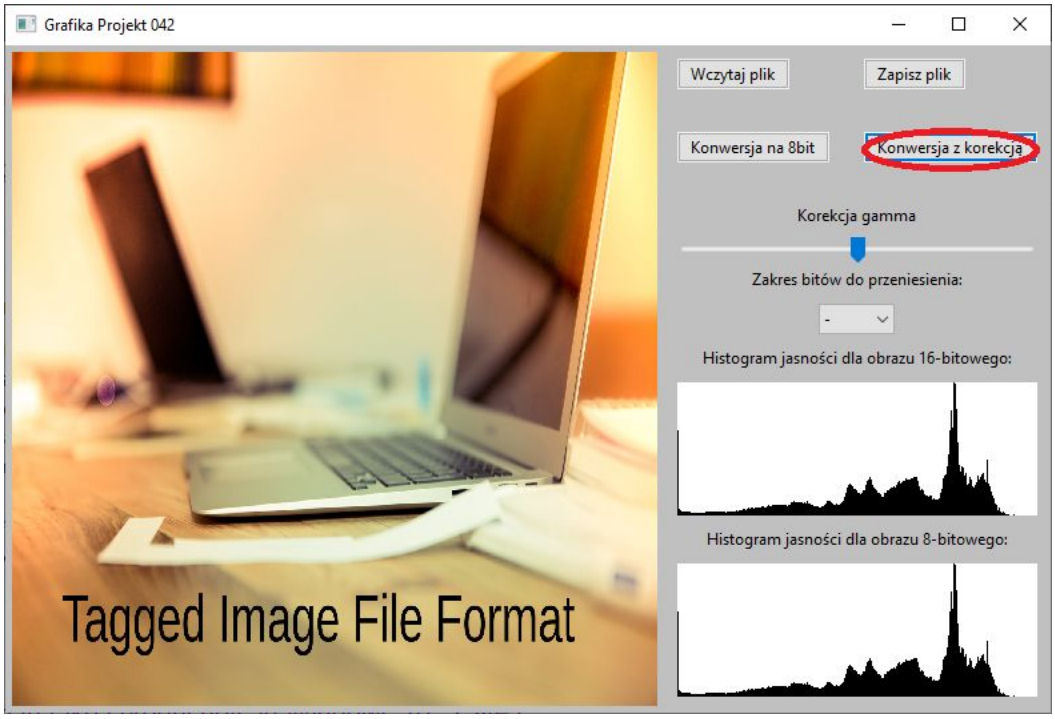
Po uprzednim wyborze nasz obraz wyświetlany jest w głównym oknie, a obok jest generowany histogram jasności dla obrazu 16-bitowego.

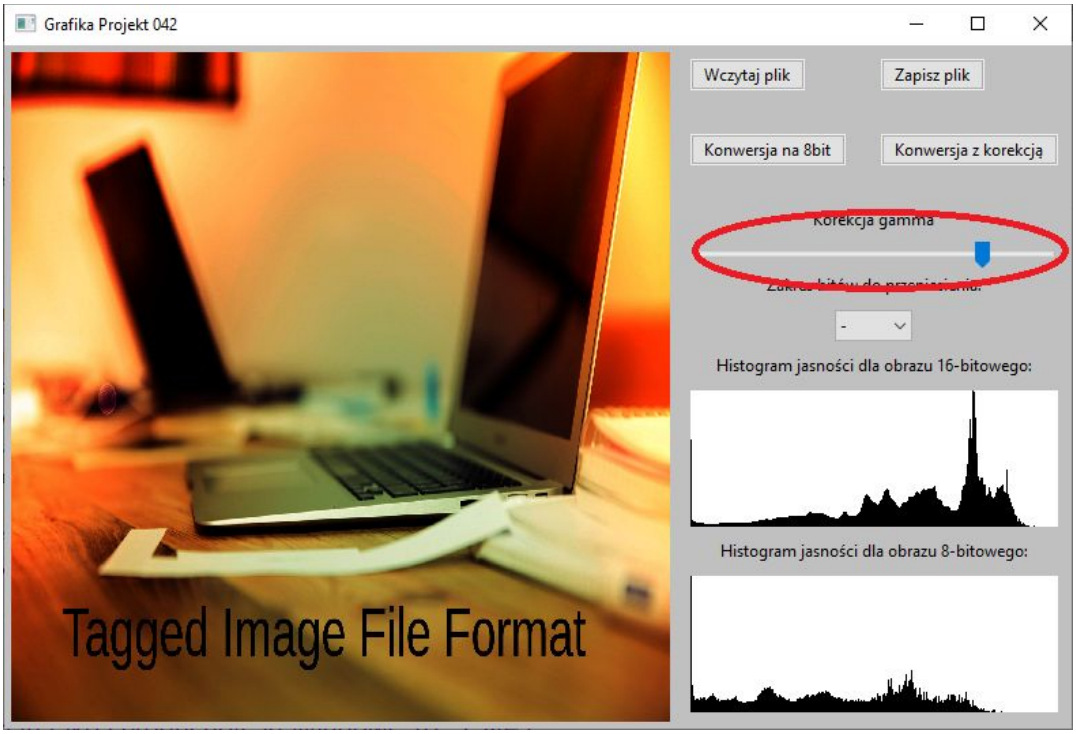


Klikając na przycisk “Konwersja na 8bit” aplikacja przekształca wybrany obraz do zadanego formatu oraz dodatkowo generuje histogram jasności dla obrazu 8-bitowego.

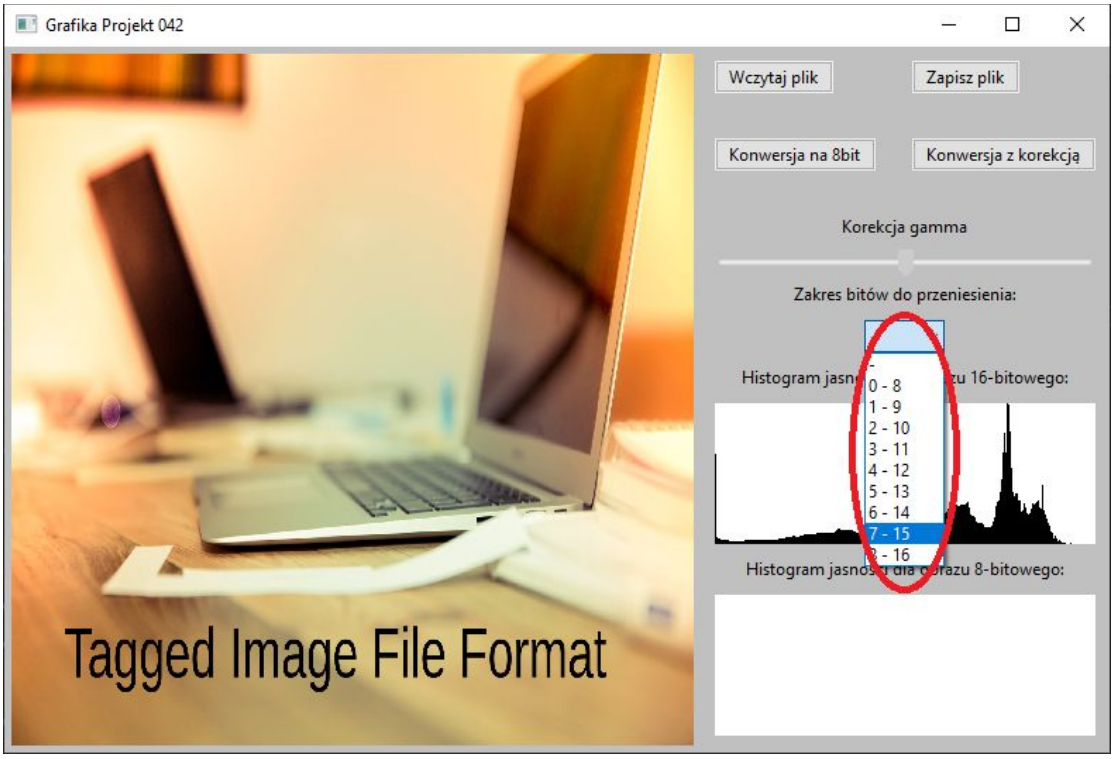


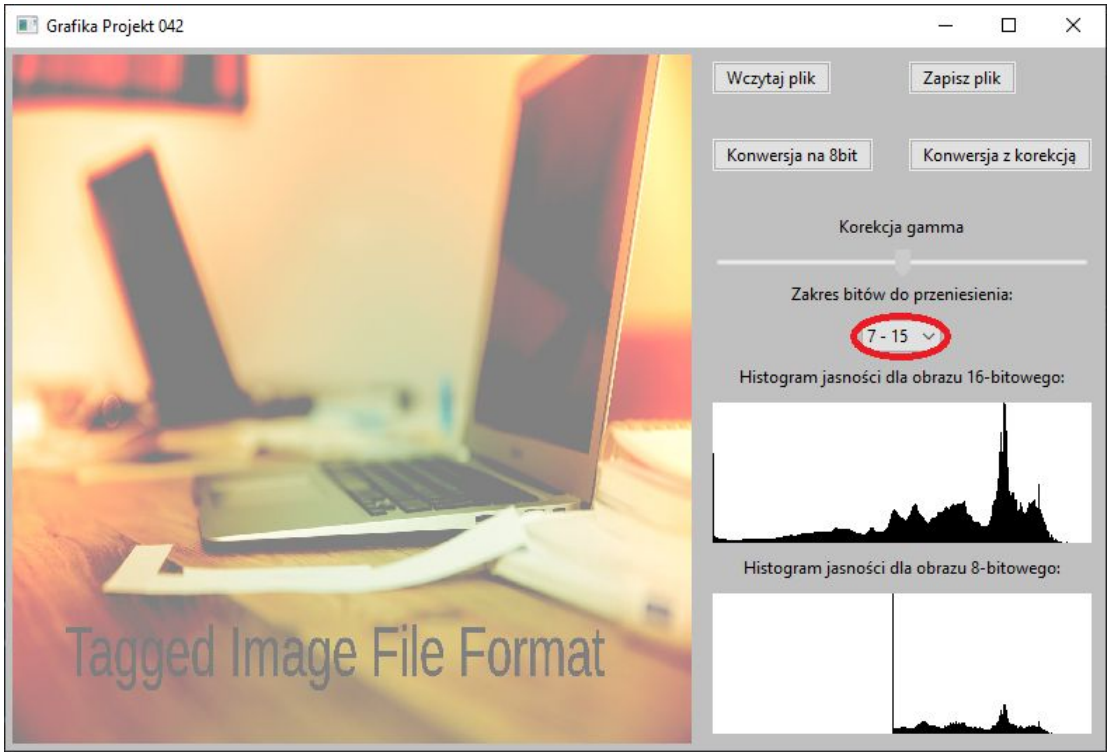
Klikając na przycisk “Konwersja z korekcją” aplikacja daje nam dostęp do suwaka, za pomocą którego możemy ustalić interesujący nas parametr gamma, generując przy tym odpowiedni histogram jasności dla obrazu 8-bitowego.



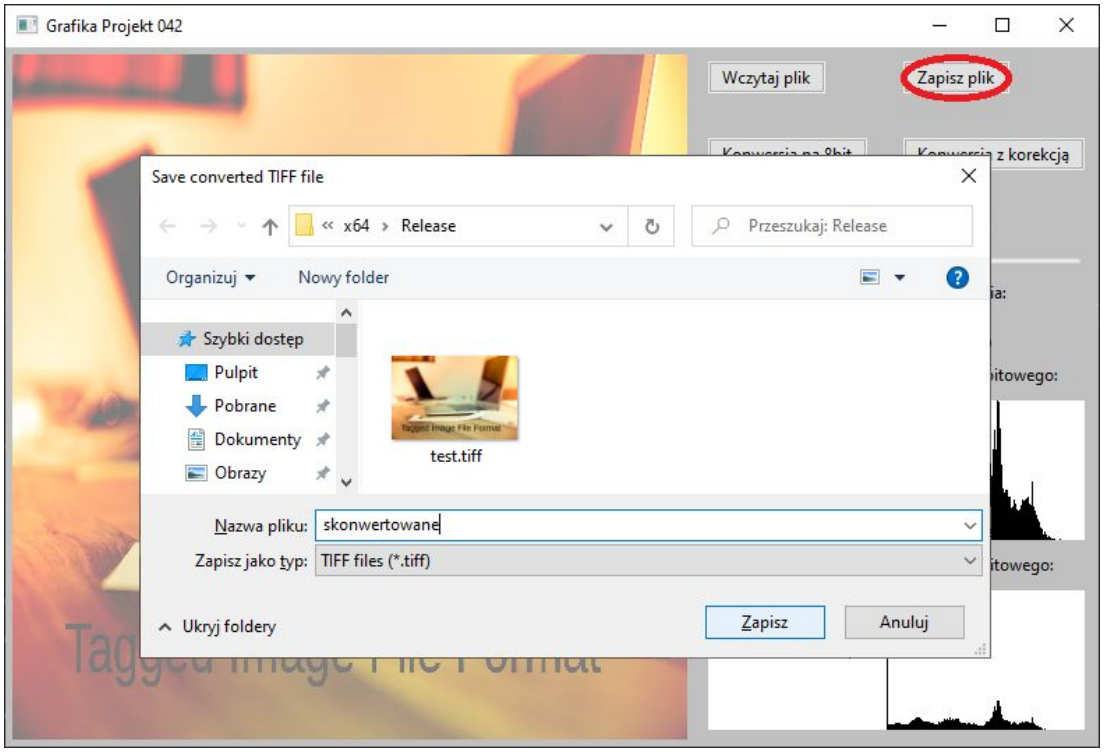


Klikając na droplistę, wyświetlą nam się przedziały bitów, które możemy przenieść z obrazu 16-bitowego, generując przy tym odpowiedni histogram jasności dla obrazu 8-bitowego.





Aby zapisać nasze modyfikacje, wystarczy tylko nacisnąć przycisk “Zapisz plik”, który uruchamia okno dialogowe umożliwiające zapis pod zadaną nazwą.



Wyodrębnienie i zdefiniowanie zadań

Określone w punkcie: 5. Podział pracy i analiza czasowa

Decyzja o wyborze narzędzie programistycznych

Do zrealizowania projektu wykorzystano bibliotekę wxWidgets w wersji 3.1.3 z uwagi na to, iż pozwala ona na tworzenie GUI w przystępny i prosty sposób. Ponadto użyto biblioteki OpenCV w wersji 3.2.0, która jest odpowiedzialna za przechowywanie obrazu oraz za wykonywanie na nim istotnych operacji związanych z konwersją. Do kompilacji użyto kompilatora g++ w wersji 19. Projekt został zrealizowany w wersji 64-bitowej w programie Visual Studio 2019.

W celu szybkiej komunikacji między członkami zespołu stworzono repozytorium w serwisie GitHub. Umożliwiło to dzielenie się wynikami działania programu oraz pozwoliło na modyfikację programu przez każdą osobę z zespołu.

Dodatkowo, aby omawiać napotkane błędy bądź problemy w implementacji kodu korzystano z prywatnych konwersacji grupowych w serwisie Facebook.

5. Podział pracy i analiza czasowa.

Podział realizowanych zadań:

- Kamil Sudoł - projekt okna aplikacji, implementacja przeniesienia wybranych bitów z obrazu 16 bitowego, testy interfejsu użytkownika, dokumentacja
- Jakub Strugała - projekt okna aplikacji, implementacja kompresji z uwzględnieniem krzywej gamma ,test konwersji obrazu, dokumentacja,
- Patryk Śledź - projekt okna aplikacji, implementacja kompresji liniowej z 16bit do 8bit, testy wczytywania i zapisywania, dokumentacja

Pierwszym etapem prac było stworzenie projektu w wxFormBuilder. Następnie, dodano wszystkie potrzebne przyciski, droplistę oraz suwak, które związano z odpowiednimi funkcjami z GUIMyFrame1.h. Kolejnym krokiem była implementacja funkcji wczytującej i wyświetlającej obraz z dysku oraz stworzenie histogramów na podstawie których widoczne są zmiany, przy użyciu poszczególnych konwersji. Kolejno, uzupełniano definicje funkcji o wymagane konwersje, poczynając od 16bit do 8bit obrazu skompresowanego liniowo, przez konwersję z wyborem zakresu bitów do przeniesienia, a kończąc na tej z uwzględnieniem parametru krzywej gamma. Na samym końcu dodano możliwość zapisu obrazu na dysk w formacie TIFF i wykonano testy. Przybliżony czas realizowania projektu: około 4 tygodni (3 tygodnie poświęcone na kodowanie projektu, tydzień na sporządzenie dokumentacji).

6. Opracowanie i opis niezbędnych algorytmów.

Wczytywanie pliku

Za wykonanie tej czynności odpowiedzialny jest przycisk “Wczytaj plik”, związany z metodą:

- ***void open_file_click(wxCommandEvent& event),***

która wykorzystuje okno dialogowe ***wxFileDialog*** umożliwiające wczytanie pliku typu “*.tiff” lub “*.tif” do zmiennej *cv::Mat image_org*. W przypadku niepowodzenia wyświetlany jest komunikat informujący użytkownika o zaistniałej sytuacji.

Zapisywanie pliku

Za wykonanie tej czynności odpowiedzialny jest przycisk “Zapisz plik”, związany z metodą:

- ***void save_file_click(wxCommandEvent& event),***

która wykorzystuje okno dialogowe ***wxFileDialog*** umożliwiające zapisanie przez nas pliku typu “*.tiff” lub “*.tif” o zadanej nazwie. Na koniec wyświetlany jest komunikat informujący użytkownika o powodzeniu wykonania operacji.

Odświeżanie okna

Do tego celu wykorzystywana jest metoda:

- ***void Repaint(),***

która dla wybranego okna tworzy bitmapę na podstawie zmiennej *wxImage Img_tmp*, a następnie wyświetla ją na ekran wykorzystując funkcję *wxClientDC DrawBitmap()*.

Konwersja typu cv::Mat do wxImage

Aby kolory były poprawnie wyświetlane, zaimplementowano algorytm transformacji obrazów cv::Mat do wxImage. W pierwszej kolejności zaalokowaliśmy pamięć dla obiektu unsigned char* (rozmiar jest iloczynem wymiarów obrazu oraz ilości kanałów RGB).
Dodatkowo potrzebne były: macierz cv::Mat w której przechowywaliśmy obraz BGR oraz tablica typu int {0,2, 1,1, 2,0} mapująca kanały pomiędzy typami RGB oraz BGR.
Jako, że obraz przechowywany w obiekcie z biblioteki OpenCV posiadał zamienione kanały R oraz B, musieliśmy dokonać przekształcenia, aby wyświetlano kolory w poprawnych barwach. Po sprawdzeniu czy operujemy na obrazie trój-kanałowym, rozpoczęto konwersję.

Skorzystano z metody dostępnej w bibliotece OpenCV:

- ***void mixChannels(cv::Mat& src, size_t nsrcs, cv::Mat& dst, size_t dsts, int*, size_t npairs)***

Powyższa metoda kopiuje wyspecyfikowane kanały z tablicy wejściowej do wyjściowej.

Konwersja liniowa 16bit do 8bit

Odpowiedzialny za nią przycisk to “Konwersja na 8bit”. W metodzie:

- ***void conversion_8bit_click(wxCommandEvent& event)***

Użyto funkcji **convertTo()** z kodem CV_8UC3 (oznaczającym typ wynikowej macierzy) z biblioteki OpenCV do przekształcenia obiektu cv::Mat do 8-bitowej macierzy o niezmienionej liczbie kanałów.

Konwersja z uwzględnieniem krzywej gamma

Za wykonanie tej konwersji odpowiedzialny jest przycisk “Konwersja z korekcją”, związany z metodą:

- ***void conversion_with_gamma_click(wxCommandEvent& event),***

który jest odpowiedzialny za odblokowanie dostępu do slider’a, ustalającego wartość parametru gamma. Omawiany slider jest połączony z metodą:

- ***void gamma_correction_scroll(wxScrollEvent& event),***

która ustala wartości parametru gamma w następujący sposób:

- jeżeli wskaźnik suwaka znajduje się w położeniu równym połowie długości slider’a, przyjmowana jest wartość 1.
- jeżeli położenie wskaźnika suwaka znajduje za połową długości slider’a, przyjmowana jest wartość różnicy maksymalnej wartości slider’a oraz obecnie wskazywanej wartości, podzielonej przez połowę długości slider’a w celu otrzymania wartości parametru gamma z przedziału [0,1].
- jeżeli położenie wskaźnika suwaka znajduje się przed połową długości slider’a, przyjmowana jest wartość różnicy maksymalnej wartości slider’a oraz obecnie wskazywanej wartości, podzielonej przez jedną dziesiątą długości slider’a powiększoną o 1 oraz dodatkowo podniesioną do kwadratu w celu otrzymania wartości parametru gamma z przedziału [1,36].

Następnie, wyznaczona przez nas wartość parametru gamma została przekazana wraz z oryginałem obrazu do metody:

- ***cv::Mat &correctGamma(cv::Mat& img, double gamma),***

która modyfikuje każdy piksel obrazu według następującego wzoru:

$$pixel' = (\frac{pixel}{255})^g * 255, \quad (1)$$

gdzie *g* jest zdefiniowane jako stosunek 1 do zadanej wartości parametru gamma.

Przeniesienie wybranych bitów z obrazu 16 bitowego

Za przeniesienie bez kompresji ośmiobitowej części informacji z obrazu szesnastobitowego odpowiedzialna jest droplista “Zakres bitów do przeniesienia”, związana z metodą:

- ***void m_b_choice_click(wxCommandEvent& event).***

Omawiana droplista przyjmuje następujące wartości dla zadanego wyboru:

- ***“-”*** - 0 - element neutralny, nie modyfikuje obrazu,
- ***“0 - 8”*** - 1,
- ***“1 - 9”*** - 2,
- ***“2 - 10”*** - 3,
- ***“3 - 11”*** - 4,
- ***“4 - 12”*** - 5,
- ***“5 - 13”*** - 6,
- ***“6 - 14”*** - 7,
- ***“7 - 15”*** - 8,
- ***“8 - 16”*** - 9,

Metoda ***m_b_choice_click*** pobiera wskaźnik z danymi z obrazu, a następnie w pętli na podstawie zadanego wyboru ustalane są elementy do modyfikacji/pozostawienia bez zmian. Modyfikacja odbywa się według następującego algorytmy:

- jeżeli $wartość\ obrazu > 2^{pozycja\ wybrana + 7} - 1,$ ustaw $wartość\ obrazu$ jako $2^{pozycja\ wybrana + 7} - 1,$
- jeżeli $wartość\ obrazu < 2^{pozycja\ wybrana - 1},$ ustaw $wartość\ obrazu$ jako $2^{pozycja\ wybrana - 1},$
- jeżeli $2^{pozycja\ wybrana - 1} < wartość\ obrazu < 2^{pozycja\ wybrana + 7} - 1,$ pozostaw bez zmian.

Jedynym wyjątkiem od tej reguły jest wybór ***“0 - 8”***, gdzie dla dolnego przedziału jest odejmowane 1 ze względów matematycznych, ponieważ niemożliwym jest otrzymanie zera z operacji potęgowania liczby o podstawie “2”.

Warto także wspomnieć, że przesunięcie o 1 w wykładniku wynika z występowania w dropliście elementu neutralnego.

Generowanie histogramów

Histogram jest funkcją przyporządkowującą możliwym poziomom jasności lub możliwym kolorom liczbę odpowiadających im pikseli w obrazie.
Jasność w modelu barw RGB możemy zdefiniować jako średnią arytmetyczną wartości kanałów R, G, B:

$$\mu = \frac{R+G+B}{3} \tag{2}$$

W naszym przypadku generowaniem histogramu zajmowała się jedna metoda:

- ***void histogram_fun(wxScrolledWindow*, wxImage&).***

Pobiera ona kontekst okna oraz obraz, dla którego histogram chcemy wygenerować. Następnie, odczytywane są wartości kanałów RGB i uzupełniana jest tablica jasności przy pomocy wzoru nr 1. Kolejno w celu wyświetlenia histogramu, wyszukiwana jest maksymalna wartość jasności oraz wartości są odpowiednio przekształcane do rozmiaru okna histogramu. Za pomocą dwóch pętli for iterujemy po: i-szerokości okna oraz j-znormalizowanej do rozmiarów okna wartości jasności dla danego bitu.
Od $j = 0$ do $j = max$ kanały R G B są wypełniane odpowiednio 0 0 0, tym samym wypełniając nasz wykres czarnymi słupkami. Ostatecznie z obiektu unsigned char tworzona jest bitmapa, która będzie wyświetlana w odpowiednim, dedykowanym obszarze okna.

7. Kodowanie.

W programie można wyszczególnić dwie klasy: MyFrame1 oraz GUIMyFrame1.

Pierwsza z nich jest klasą bazową okna aplikacji **MyFrame1**. Jej elementami są:

- **wxScrolledWindow*** m_Window - obszar poświęcony wyświetlaniu obrazu,
- **wxScrolledWindow*** histogram - obszar przeznaczony do wyświetlania histogramu dla oryginału obrazu,
- **wxScrolledWindow*** histogram2 -obszar przeznaczony do wyświetlania histogramu dla obrazu po modyfikacjach,
- **wxButton*** conversion_8bit - przycisk dla konwersji z 16bit do 8bit,

- **wxButton*** conversion_with_gamma - przycisk dla konwersji z 16bit do 8bit z uwzględnieniem krzywej gamma,
- **wxButton*** open_file - przycisk okna dialogowego umożliwiający wczytywanie pliku,
- **wxButton*** save_file - przycisk okna dialogowego umożliwiający zapisywanie pliku,
- **wxSlider*** gamma_correction_slider - suwak służący do ustawienia parametru gamma,
- **wxChoice*** m_choice - droplista wyboru zakresu bitów do przeniesienia,
- **wxStaticText*** m_staticText_font2, m_staticText1, m_staticText2, m_staticText3 - obiekty przechowujące tekst,
- **int** slid_size - rozmiar suwaka *gamma_correction_slider.*

Klasa **GUIMyFrame1** jest klasą dziedziczącą po **MyFrame1**. Zawiera ona metody i obiekty pozwalające dokonywać przekształceń. Jej elementami są:

Obiekty i zmienne:

- **wxBitmap** pBitmap - bitmapa pomocnicza, służąca do wyświetlania obrazu,
- **wxImage** Img_tmp - obiekt przechowujący kopię obrazu skonwertowanego ze zmiennej typu cv::Mat result,
- **wxImage** Org - obiekt przechowujący kopię obrazu skonwertowanego ze zmiennej cv::Mat image_org,
- **cv::Mat** image_org - obiekt biblioteki OpenCV przechowywujący wczytany obraz oryginalny,
- **cv::Mat** result - obiekt biblioteki OpenCV przechowywujący kopię obrazu służącą do przekształceń.

Metody:

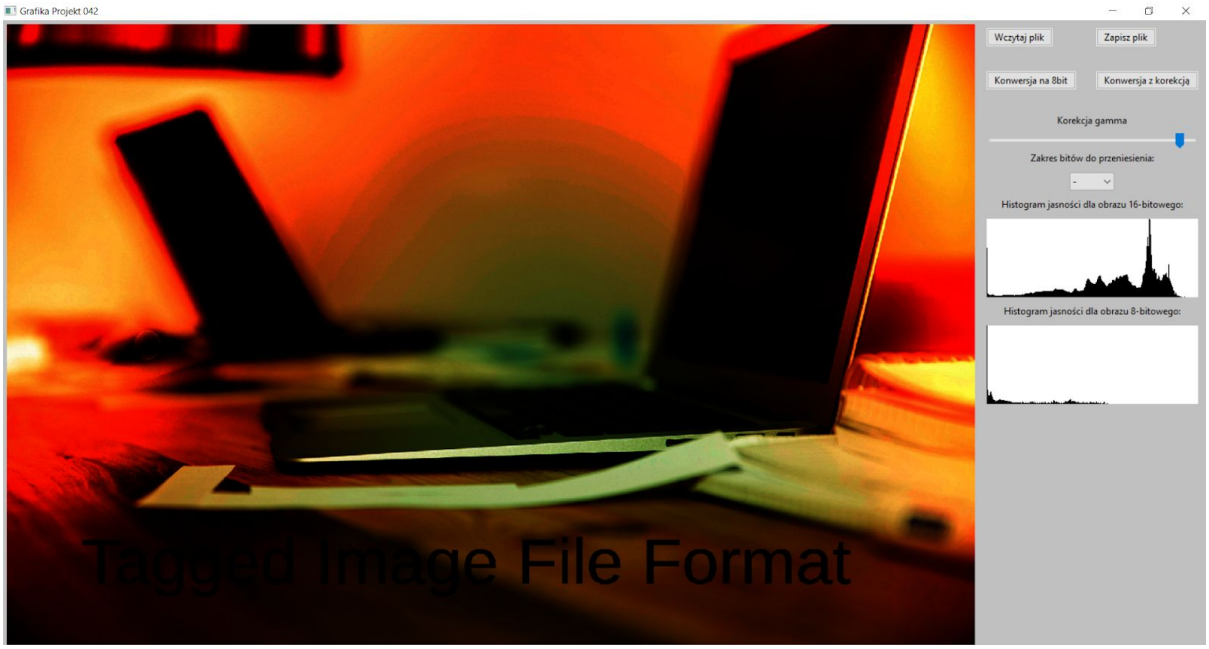
- **void** Repaint() - metoda wyświetlająca obraz na ekran,
- **wxImage** getImage(**cv::Mat &** img) - metoda pozwalająca na poprawne przekonwertowanie obrazu z openCV do wxWidgets,
- **void** histogram_fun(**wxScrolledWindow *** window, **wxImage &** img) - metoda odpowiadająca za stworzenie i wyświetlenie histogramu dla zadanego obrazu,
- **cv::Mat &**correctGamma(**cv::Mat&** img, **double** gamma) - metoda modyfikująca obraz o dany parametr gamma,
- **void** open_file_click(**wxCommandEvent&** event) - metoda odpowiadająca za wczytanie pliku z dysku,
- **void** save_file_click(**wxCommandEvent&** event) - metoda odpowiadająca za zapis zmodyfikowanego pliku na dysku,
- **void** conversion_8bit_click(**wxCommandEvent&** event) - metoda realizująca konwersje 16bit do 8bit, rysująca przy tym odpowiednie histogramy dla obrazu
- **void** conversion_with_gamma_click(**wxCommandEvent&** event) - metoda aktywująca suwak korekcji gamma,

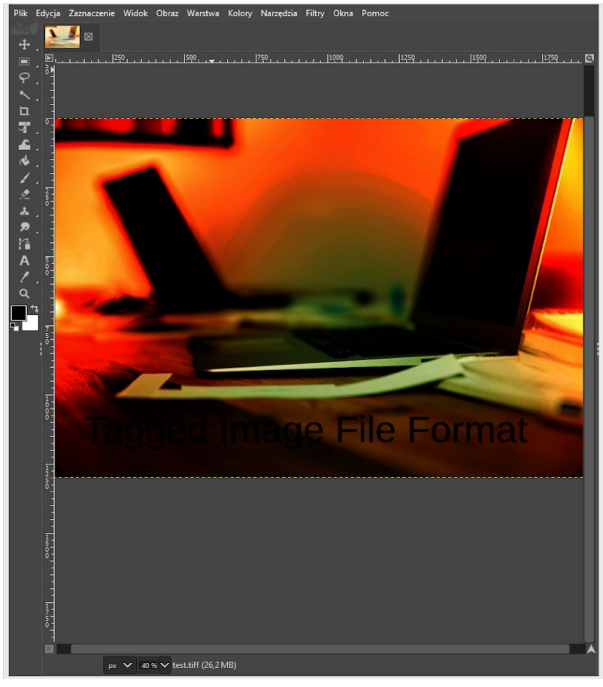
- **void** gamma_correction_scroll(**wxScrollEvent&** event) - metoda realizująca modyfikację obrazu o zadany parametr gamma, wyświetlająca przy tym aktualny podgląd obrazu oraz odpowiednie histogramy,
- **void** m_b_choice_click(**wxCommandEvent&** event) - metoda realizująca konwersję w zależności od wyboru zakresu bitów do przeniesienia, rysująca przy tym odpowiednie histogramy dla obrazu.

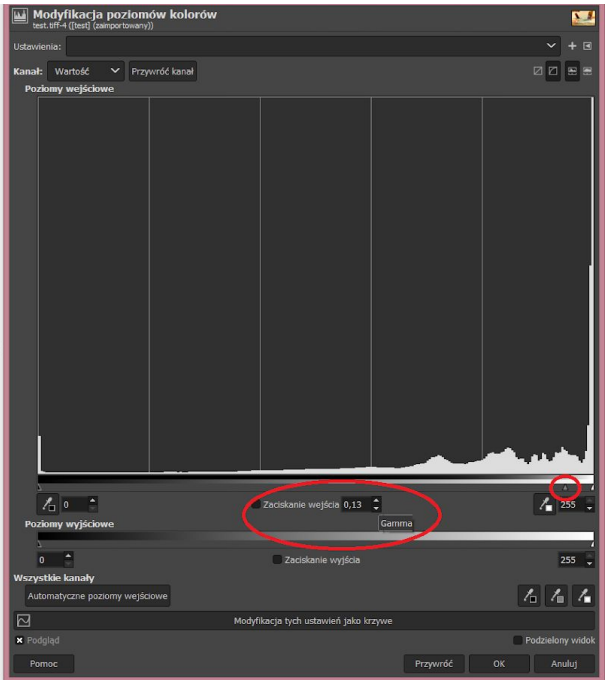
8. Testowanie.

Działanie programu przetestowano manualnie, porównując wyniki przekształceń na obrazie z efektami otrzymanymi w programie Gimp2 - zarówno sam efekt na obrazie jak i wygląd histogramu jasności dla wczytanego obrazu.
Na zamieszczonych poniżej screenach, widać iż efekty konwersji z uwzględnieniem parametru gamma odpowiadają zmianom tego parametru w programie Gimp, natomiast histogram jasności względnie pokrywa się z histogramem w naszym programie (należy nadmienić, iż u nas jest on znormalizowany).

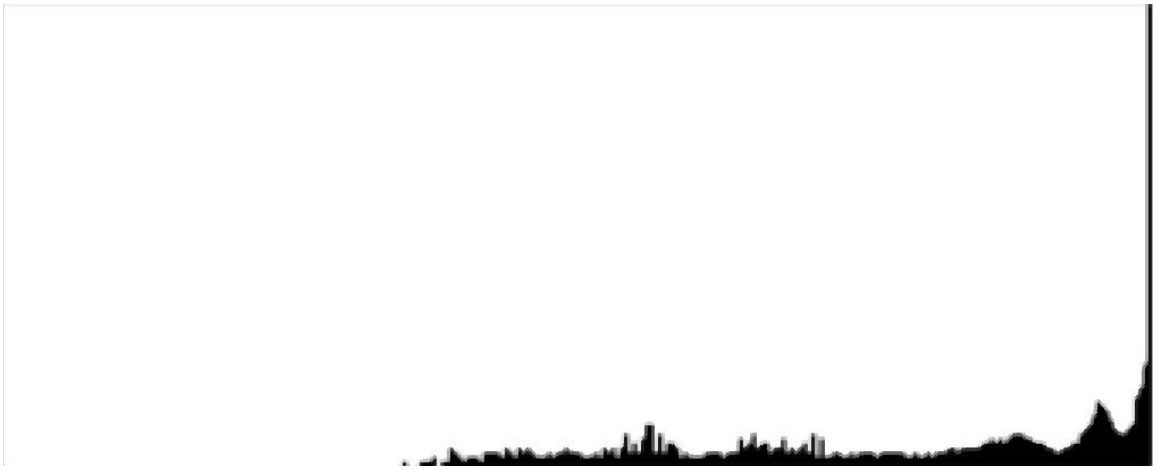
- Porównanie efektu konwersji gamma w naszym projekcie i w Gimp'ie:

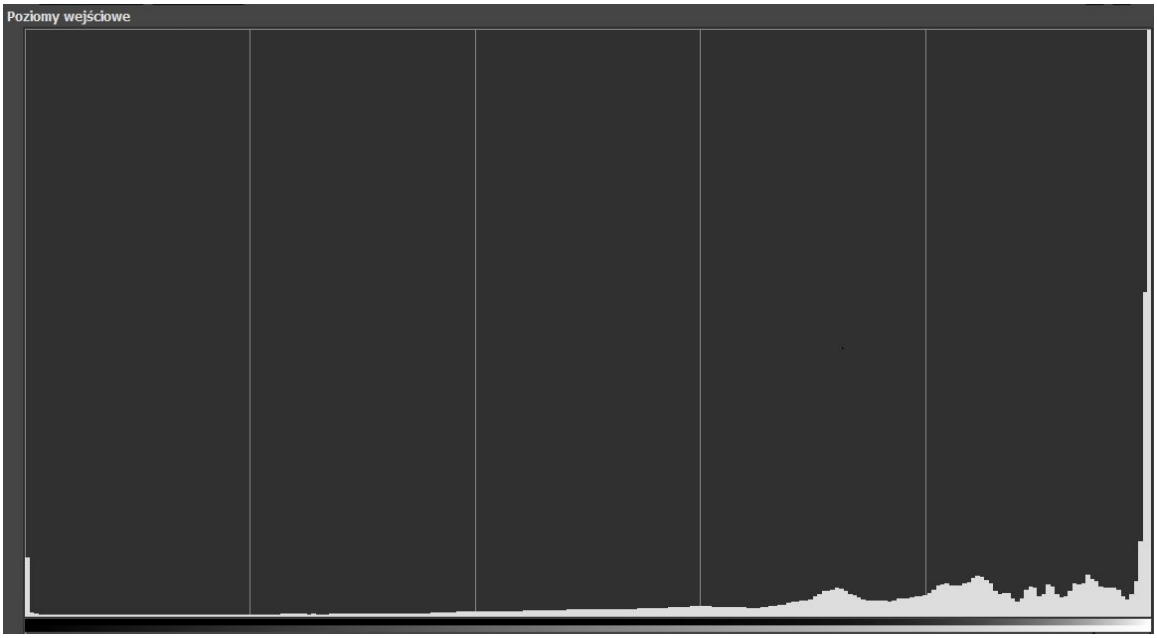






- Porównanie histogramów:





Następnie przeprowadzono testy interfejsu użytkownika, które wykazały, w których miejscach należy zablokować użytkownikowi niektórych funkcjonalności programu. Przykładem może tutaj być wyświetlanie komunikatu informującego o braku wczytanego obrazu w przypadku, kiedy użytkownik wcisnął jedną z opcji konwersji, uprzednio nie wczytując pliku. Ponadto dla celów estetycznych, gdy mamy wybraną opcję konwersji z parametrem gamma z już ustawioną wartością i jednak zdecydujemy się z niej zrezygnować na rzecz innej z dostępnych opcji, suwak zostaje wówczas zresetowany na pozycję startową oraz staje się on ponownie niedostępny.
Na końcu przetestowano działanie funkcji zapisywania i wczytywania obrazu, które okazały się działać bez zarzutów.

9. Wdrożenie, raport i wnioski.

W wyniku realizacji projektu, względnie udało się wykonać wszystkie wymagania podstawowe: możliwe jest wczytanie obrazu 16-bitowego w formacie TIFF, konwertowanie go przy użyciu wymienionych w punkcie 3. konwersji, zapisanie zmodyfikowanego obrazu do nowego pliku czy też wyświetlanie histogramów jasności dla wczytanego i zmodyfikowanego obrazu. Jedyną niedziałającą w pełni metodą zdaje się być konwersja z wyborem zakresu bitów do przeniesienia. Udało się ją zaimplementować w pewnym stopniu, jednakże nie produkuje ona całkowicie poprawnych wyników.
Mając na względzie inne niedoskonałe elementy projektu, należałoby nadmienić, iż wymieniona metoda nie jest zoptymalizowana, co powoduje wydłużenie jej czasu działania dla dużych plików.
W przyszłości, poza poprawą tejże metody, w celu usprawnienia działaniu programu, należałoby przeprowadzić ogólną optymalizację programu.