Sprawozdanie z Aplikacji do Przetwarzania Obrazów

Urszula Szczęsna, Kornel Tłaczała

28 marca 2025

Spis treści

1 Opis aplikacji

Aplikacja jest interaktywnym narzędziem do przetwarzania obrazów oraz manipulacji, zaimplementowanym w języku Python. Główne biblioteki użyte w projekcie to:

- `numpy` – do obliczeń numerycznych, w tym operacji na macierzach reprezentujących obrazy.
- `PIL (Pillow)` – do wczytywania, przetwarzania i zapisywania obrazów w różnych formatach.
- `matplotlib` – do wizualizacji danych, w tym generowania histogramów oraz projekcji obrazu.
- `streamlit` - do budowy interaktywnego interfejsu użytkownika, umożliwiające łatwe przeprowadzenie operacji na obrazach oraz ich wizualizację w aplikacji webowej.

Aplikacja umożliwia użytkownikowi wykonywanie różnych operacji na obrazach, takich jak:

1. Operacje na pikselach
    - korekta jasności
    - korekta ekspozycji (exposure)
    - korekta kontrastu
    - przekształcenie gamma
    - konwersja do odcieni szarości
    - negatyw
    - binaryzacja
2. Nakładanie filtrów graficznych
    - Gaussa
    - uśredniający
    - wyostrzający
3. Wykrywanie krawędzi
    - Krzyż Robertsa
    - Operator Sobela
4. Tworzenie projekcji oraz histogramów obrazu

Dzięki interaktywnemu interfejsowi użytkownik może w łatwy sposób eksplorować różne metody przetwarzania obrazów i dostosowywać parametry filtrów do własnych potrzeb. Dodatkowo została zaimplementowana możliwość zapisania przetworzonego obrazu za pomocą przycisku.

1.1 Implementacja

Aplikacja przetwarza obrazy metodą kaskadową. Każdy rodzaj operacji który można wykonać jest reprezentowany przez procesor odpowiedniej klasy, który przy pomocy metody `process()` dokonuje przekształcenia obrazu. Wszystkie procesory zapisują ostatnio wyliczony obraz w swojego rodzaju pamięci cache i są opakowane w obiekt klasy `ProcessorFlow`. Takie rozwiązanie pozwala na powtórne wykorzystanie wcześniej zapamiętanych wyników i nie powoduje konieczności wyliczania wszystkiego od początku.

W celu szybszego działania aplikacji dla dużych obrazów, została zaimplementowana również możliwość zmniejszenia rozdzielczości obrazu do zadanej maksymalnej wartości.

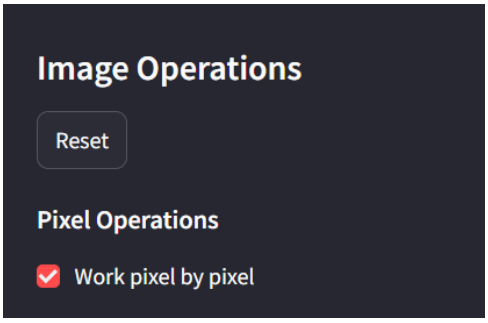
Rysunek 1: Zmiana rozdzielczości przetwarzanego obrazu

2 Opis metod

2.1 Operacje na pikselach

Dla każdej z poniższych operacji na pikselach zostały zaimplementowane dwie metody obliczeń:

1. Pikselowa iteracja – operacje wykonywane na poziomie jednego piksela w obrazie, z uwzględnieniem poszczególnych kanałów kolorystycznych.
2. Operacje macierzowe – przetwarzanie całego obrazu jednocześnie przy użyciu operacji na macierzach.



Rysunek 2: Możliwość wyboru pomiędzy metodami macierzową i pikselową.

2.1.1 Korekta ekspozycji

Procesor `ExposureProcessor` zmienia jasność obrazu poprzez przemnożenie wartości pikseli przez zadany przez użytkownika współczynnik. Wartości pikseli są następnie ograniczane do zakresu [0, 255] (przy użyciu funkcji clip z pakietu numpy).

$$I_{\text{exposure}}(x, y) = \text{clip}(I(x, y) \times \text{factor}, 0, 255)$$

2.1.2 Korekta jasności

Procesor `BrightnessProcessor` zmienia jasność obrazu poprzez dodanie do każdego piksela wartości zadanej przez użytkownika. Wartości pikseli są następnie ograniczane do zakresu [0, 255] (przy użyciu funkcji clip z pakietu numpy).

$$I_{\text{brightened}}(x, y) = \text{clip}(I(x, y) + \text{factor}, 0, 255)$$

2.1.3 Regulacja kontrastu

Procesor `ContrastProcessor` dostosowuje kontrast obrazu poprzez przesunięcie wartości pikseli wokół średniej wartości obrazu, a następnie ich skalowanie przez zadany przez użytkownika współczynnik. Następnie wartości pikseli są ograniczane do zakresu [0, 255].

$$I_{\text{contrast}}(x, y) = \text{clip}((I(x, y) - \mu) \times \text{factor} + \mu, 0, 255)$$

gdzie μ to średnia wartość pikseli w obrazie.

2.1.4 Przekształcenie gamma

Procesor `GammaProcessor` wykonuje przekształcenie jasności i kontrastu obrazu poprzez operację nieliniową. Przekształcenie to stosuje funkcję potęgową do wartości intensywności pikseli, co pozwala na ich modyfikacje w sposób lepiej odpowiadający percepcji ludzkiego oka.

$$I_{gamma}(x, y) = 255 \times \left(\frac{I(x, y)}{255} \right)^{\text{factor}}$$

gdzie:

- $I(x, y)$ – wartość piksela w obrazie wejściowym,
- $I_{gamma}(x, y)$ – skorygowana wartość piksela po przekształceniu gamma,
- factor – parametr gamma określający siłę korekcji.

Przekształcenie gamma pozwala na kontrolowanie rozkładu jasności w obrazie:

- Jeśli factor > 1, obraz staje się ciemniejszy – redukowane są wartości jasnych pikseli, co uwydatnia detale w jasnych obszarach.
- Jeśli factor < 1, obraz staje się jaśniejszy – wzmacniane są wartości ciemnych pikseli, co poprawia widoczność detali w cieniach.
- Jeśli factor $= 1$, obraz pozostaje bez zmian.

2.1.5 Konwersja do skali szarości

Procesor `GrayscaleProcessor` konwertuje obraz kolorowy na obraz w odcieniach szarości, wykorzystując standardowe współczynniki dla kanałów R, G i B:

$$I_{\text{gray}} = 0.2989 \cdot R + 0.5870 \cdot G + 0.1140 \cdot B$$

2.1.6 Negatyw obrazu

Procesor `NegativeProcessor` wykonuje inwersję kolorów obrazu, tworząc jego negatyw czyli odwrotność pierwotnego obrazu. Działa to poprzez odjęcie wartości każdego piksela od 255. Funkcja zwraca nowy obraz z negatywem.

$$I_{\text{negative}}(x, y) = 255 - I(x, y)$$

gdzie $I(x, y)$ to wartość piksela w obrazie wejściowym.

2.1.7 Binarizacja

Procesor `BinarizationProcessor` przekształca obraz do postaci binarnej, przypisując wartość 0 dla pikseli mniejszych niż próg i wartość 255 dla pikseli większych lub równych progowi.

$$I_{\text{binary}}(x, y) = \begin{cases} 0 & \text{jeśli } I(x, y) < \text{threshold} \\ 255 & \text{w przeciwnym razie} \end{cases}$$

2.2 Filtry

Filtry są stosowane do wygładzania lub wyostrzania obrazu. Użytkownik ma możliwość wyboru jednego z 3 filtrów dostępnych w aplikacji:

2.2.1 Filtr gaussowski

Procesor `GaussianFilterProcessor` wykonuje filtrację za pomocą jądra gaussowskiego. Jest to realizowane jako splot obrazu $I(x, y)$ z jądrem gaussowskim $G(x, y)$, czyli:

$$I_{gaussian}(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} G(i, j) \cdot I(x + i, y + j)$$

gdzie:

- $I(x, y)$ – oryginalny obraz
- $G(x, y)$ – jądro filtru Gaussowskiego
- $k = \frac{\text{size}-1}{2}$ – połowa rozmiaru okna filtra

Jądro jest generowane na podstawie rozmiaru filtra i standardowego odchylenia σ. Jądro jest zdefiniowane wzorem:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp(-\frac{x^2 + y^2}{2\sigma^2})$$

Dodatkowo wagi jądra są normalizowane do 1, aby zachować jasność obrazu.

W przypadku obrazów kolorowych dla każdego z kanałów RGB splot jest dokonywany oddzielnie.

2.2.2 Filtr uśredniający

Procesor `MeanFilterProcessor` wykonuje filtrację średnią, w której każdy piksel jest zastępowany średnią wartością pikseli w jego otoczeniu. Przy pomocy suwaka `size` użytkownik wybiera wielkość 'okna', wewnątrz którego piksele będą zastępowane średnia. Im większy rozmiar okna, tym bardziej wygładzony obraz.

$$I_{meanfilter}(x,y) = \frac{1}{N}\sum_{i=-k}^{k}\sum_{j=-k}^{k} I(x+i, y+j)$$

gdzie:

- $I(x,y)$ to oryginalny obraz
- $k = \frac{\text{size}-1}{2}$ to połowa rozmiaru okna
- $N = (2k+1)^2$ to liczba pikseli w oknie filtra
- (i,j) to przesunięcie względem środka okna
- (x,y) to współrzędne piksela

Dla obrazu kolorowego filtr uśredniający działa niezależnie na każdym kanale RGB.

2.2.3 Filtr wyostrzający

Procesor `SharpeningFilterProcessor` stosuje filtr wyostrzający, który wykorzystuje odpowiednią macierz splotu w celu wyostrzenia krawędzi obrazu. Użytkownik ma do wyboru możliwość skorzystania z jednej z 2 rodzajów macierzy. Po lewej stronie macierz wyostrzania regularnego, po prawej stronie macierz wyostrzania silnego. Macierze przedstawione zostały dla `size = 3` oraz σ `= 1`, natomiast mogą zostać wyliczone również dla innych wartości parametrów

$$K = \begin{bmatrix} 0 & -\sigma & 0 \\ -\sigma & 4\sigma+1 & -\sigma \\ 0 & -\sigma & 0 \end{bmatrix} \qquad K = \begin{bmatrix} -\sigma & -\sigma & -\sigma \\ -\sigma & 8\sigma+1 & -\sigma \\ -\sigma & -\sigma & -\sigma \end{bmatrix}$$

Operacja ta może być zapisna jako:

$$I_{sharpened}(x,y) = \sum_{i=-k}^{k}\sum_{j=-k}^{k} K(i,j)\cdot I(x+i, y+j)$$

gdzie:

- $I(x,y)$ – oryginalny obraz,
- $K(i,j)$ – jądro konwolucyjne (w tym przypadku filtr wyostrzający),
- $k = \frac{\text{size}-1}{2}$ – połowa rozmiaru okna filtra,
- (x,y) – współrzędne piksela obrazu,
- (i,j) – przesunięcie względem środka okna.

2.3 Wykrywanie krawędzi

Krawędzie wykrywane są poprzez analizę zmian jasności obrazu. W zależności od wybranej metody wykorzystuje się odpowiednie macierze, które po wykonaniu splotu z obrazem zwracają mapę gradientu horyzontalnego `Gx` oraz wertykalnego `Gy`. Natępnie te mapy są na siebie nakładane. W tym celu obliczone zostają wartości:

$$G = \sqrt{G_x^2 + G_y^2}$$

Po normalizacji, mogą one zostać zamienione na macierz obrazu krawędzi w skali szarości. Można też zastosować próg klasyfikacji. Wartości większe od niego są przedstawione na biało, a wartości mniejsze na czarno. Białe piksele pokrywają się z krawędziami w oryginalnym obrazie

2.3.1 Krzyż Robertsa

Wartości `Gx` oraz `Gy` obliczamy przy pomocy macierzy:

$$\texttt{Gx} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \texttt{Gx} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Krzyż Robertsa nakłada na siebie krawędzie wykryte idąc po 2 skosach.

2.3.2 Operator Sobela

Wartości `Gx` oraz `Gy` obliczamy przy pomocy macierzy:

$$\texttt{Gx} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad \texttt{Gx} = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Operator Sobela nakłada na siebie krawędzie horyzontalne oraz wertykalne

2.4 Tworzenie projekcji oraz histogramów obrazu

2.4.1 Histogram obrazu

Funkcja `compute_histogram(img)` tworzy histogram obrazu, przedstawiający rozkład wartości pikseli w obrazie. Histogram oblicza liczbę pikseli dla każdej wartości jasności (od 0 do 255 w przypadku obrazów w skali szarości). Dla obrazów kolorowych jest to wyliczane oddzielnie dla każdego kanału. Histogram obrazu H można zdefiniować za pomocą funkcji:

$$H(i) = \sum_{x=0}^{M-1} \sum_{y-0}^{N-1} I(f(x,y) - i), i = 0, 1, ..., L-1$$

gdzie:

- $f(x, y)$, to wartość piksela w punkcie (x, y)
- M i N to wymiary obrazu
- L to liczba poziomów jasności (zwykle 256 dla obrazów 8-bitowych)
- I to funkcja indykatorowa

$$I_A(x) = \begin{cases} 1, & \text{jeśli } x \in A \\ 0, & \text{jeśli } x \notin A \end{cases}$$

2.4.2 Projekcja

Projekcja obrazu polega na zrzutowaniu wartości pikseli na osie pozioma i pionową.

Projekcja pozioma Funkcja `horizontal_projection(img)` oblicza sumę wartości pikseli w każdym wierszu obrazu, generując wektor wartości projekcji poziomej:

$$P_h(i) = \sum_{j=0}^{w-1} I(i,j)$$

gdzie:

- $P_h(i)$ – wartość projekcji dla i-tego wiersza,
- $I(i, j)$ – wartość piksela w wierszu i i kolumnie j,
- w – szerokość obrazu w pikselach.

Algorytm działa według następujących kroków:

1. Konwersja obrazu na tablicę `numpy`.
2. Jeśli obraz jest kolorowy (posiada 3 kanały), zostaje przekonwertowany do skali szarości.
3. Iteracja przez wszystkie wiersze i sumowanie wartości pikseli w każdej kolumnie.
4. Zwrócenie wektora wartości projekcji poziomej.

Projekcja pionowa Funkcja `vertical_projection(img)` działa analogicznie, ale oblicza sumy pikseli w każdej kolumnie, tworząc wektor projekcji pionowej:

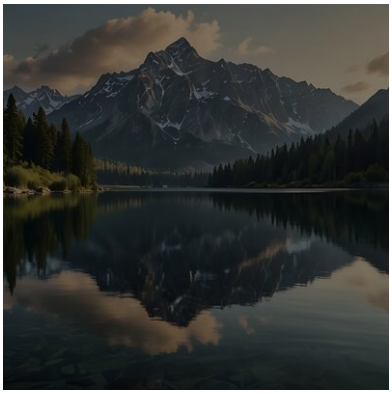
$$P_v(j) = \sum_{i=0}^{h-1} I(i,j)$$

gdzie:

- $P_v(j)$ – wartość projekcji dla j-tej kolumny,
- $I(i,j)$ – wartość piksela w wierszu i i kolumnie j,
- h – wysokość obrazu w pikselach.

3 Prezentacja wyników działania

3.1 Ekspozycja



Rysunek 3: Exposure 0.5

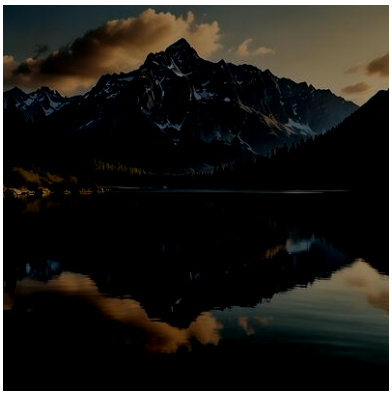
Rysunek 4: Orginalny obrazek

Rysunek 5: Exposure 1.5

Rysunek 6: Porównanie zmiana paramertru exposure

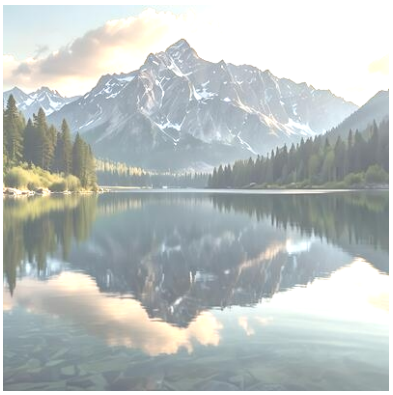
Manipulacja parametrem Exposure wpływa na jasność pikseli – niższe wartości powodują przyciemnienie obrazu, natomiast wyższe prowadzą do jego rozjaśnienia.

3.2 Jasność



Rysunek 7: Brightness -100

Rysunek 8: Orginalny obrazek

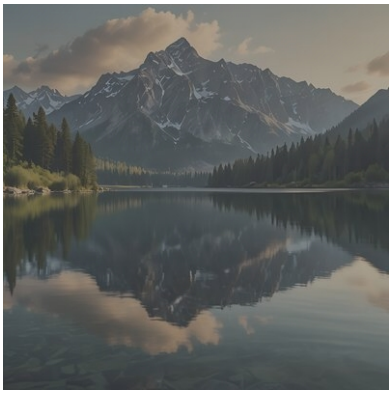


Rysunek 9: Brightness +100

Rysunek 10: Porównanie zmiana parametru brightness

Manipulacja parametrem Brightness dodaje określoną wartość do każdego piksela – niższe wartości powodują przyciemnienie obrazu, natomiast wyższe prowadzą do jego rozjaśnienia.

3.3 Kontrast



Rysunek 11: Contrast 0.5

Rysunek 12: Orginalny obrazek

Rysunek 13: Contrast 1.5

Rysunek 14: Porównanie zmiana parametru contrast

Manipulacja parametrem kontrastu wpływa na różnice jasności między elementami obrazu – zwiększa je, uwydatniając szczegóły, lub zmniejsza, gdy jego wartość jest mniejsza niż 1, prowadząc do bardziej jednolitego wyglądu.

3.4 Gamma

Rysunek 15: Gamma 0.7

Rysunek 16: Orginalny obrazek

Rysunek 17: Gamma 1.3

Rysunek 18: Porównanie zmiana parametru gamma

Gamma reguluje nieliniowe przetwarzanie jasności obrazu, dostosowując poziomy światła zgodnie z percepcją ludzkiego oka. Niższe wartości rozjaśniają ciemniejsze obszary, natomiast wyższe przyciemniają jasne fragmenty. Efekt korekcji gamma jest subtelniejszy i mniej intensywny niż manipulacja parametrami Exposure lub Brightness.

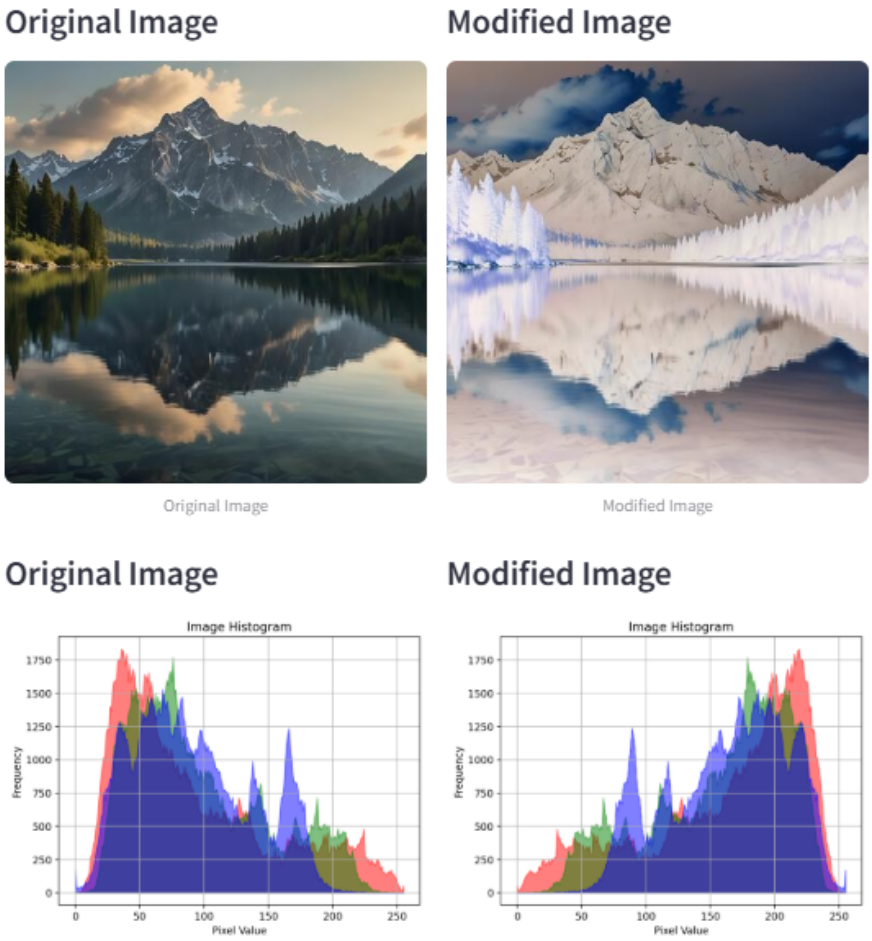
3.5 Skala szarości

Rysunek 19: Obraz oryginalny

Rysunek 20: Obraz w skali szarości

Konwersja obrazu do odcieni szarości zastępuje wartości kanałów RGB pojedyńczą wartością, wspólną dla 3 kolorów, zapisaną w dwuwymiarowej (zamiast trójwymiarowej macierzy)

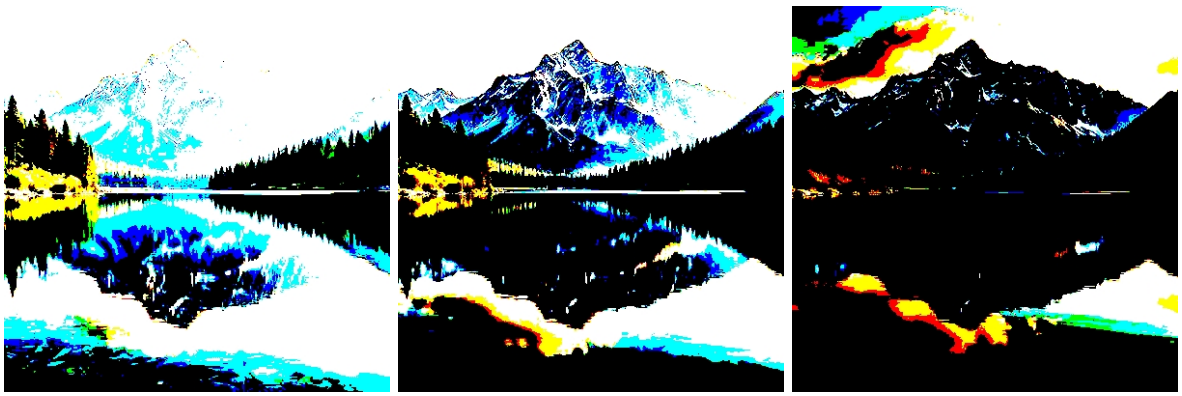
3.6 Negatyw



Rysunek 21: Negatyw oraz histogramy

Na podstawie histogramów (które zliczają ilość pikseli o danej jasności) możemy zauważyć, że negatywizacja została przeprowadzona poprawnie, ponieważ wykres uległ odwróceniu.

3.7 Binaryzacja



Próg 60 Próg 90 Próg 140

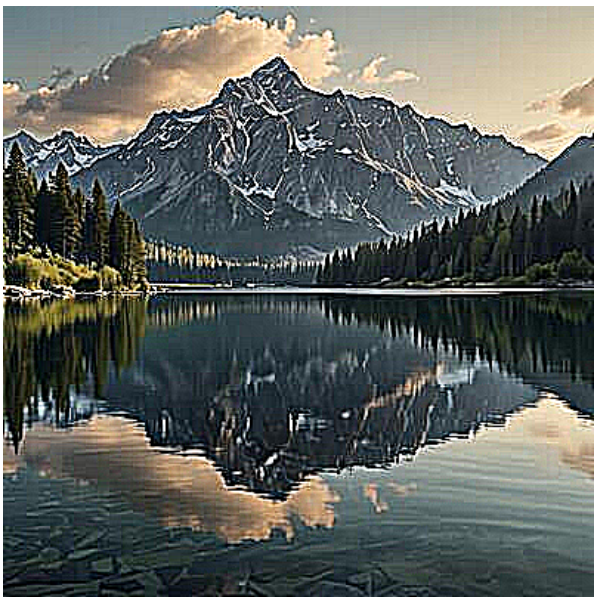
Rysunek 22: Porównanie zmiany progu binaryzacji

3.8 Filtry

Filtr Gaussa, size 5, $\sigma = 3$

Filtr uśredniający, size 11

Ostrzenie, size 3, strength 1

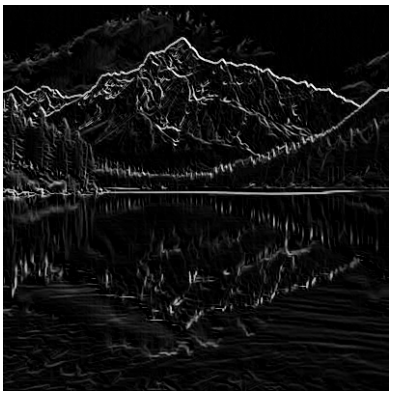


Ostrzenie, size 5, strength 1.5

Filtry Gaussa oraz uśredniający powodują rozmycie obrazu. Filtr wyostrzający zwiększa ziarnistość oraz lokalny kontrast z różną siłą, w zależności od parametrów.

3.9 Wykrywanie krawędzi

Wykrywanie krawędzi Krzyżem Robertsa



Wykrywanie krawędzi Operatorem Sobela

Operator Sobela na obrazku wyostrzonym

Rysunek 23

Obraz oryginalny

Krzyż Robertsa

Operator Sobela

Rysunek 24