

WYDZIAŁ MATEMATYKI I NAUK INFORMACYJNYCH  
POLITECHNIKA WARSZAWSKA

---

## Aplikacja do przetwarzania obrazów

---

BIOMETRIA  
PROJEKT 1 - DOKUMENTACJA

Maciej Momot, Filip Szlingiert

27 marca 2025

## Spis treści

<b>1 Opis aplikacji.</b>	<b>3</b>
<b>2 Opis metod i filtrów.</b>	<b>4</b>
2.1 Zmiana jasności i kontrastu . . . . .	4
2.2 Konwersja do skali szarości - Uśrednianie . . . . .	4
2.3 Konwersja do skali szarości - Metoda dekompozycji (Lightness) . . . . .	4
2.4 Desaturacja . . . . .	5
2.5 Negatyw . . . . .	6
2.6 Binaryzacja . . . . .	6
2.7 Filtr rozmywający z regulowanym jądrem . . . . .	7
2.8 Filtr wyostrzający . . . . .	8
2.9 Filtr Gaussa . . . . .	8
2.10 Znajdowanie krawędzi - Laplasjan . . . . .	9
2.11 Metody gradientowe wykrywania krawędzi . . . . .	9
2.12 Własny filtr . . . . .	11
<b>3 Wnioski</b>	<b>12</b>

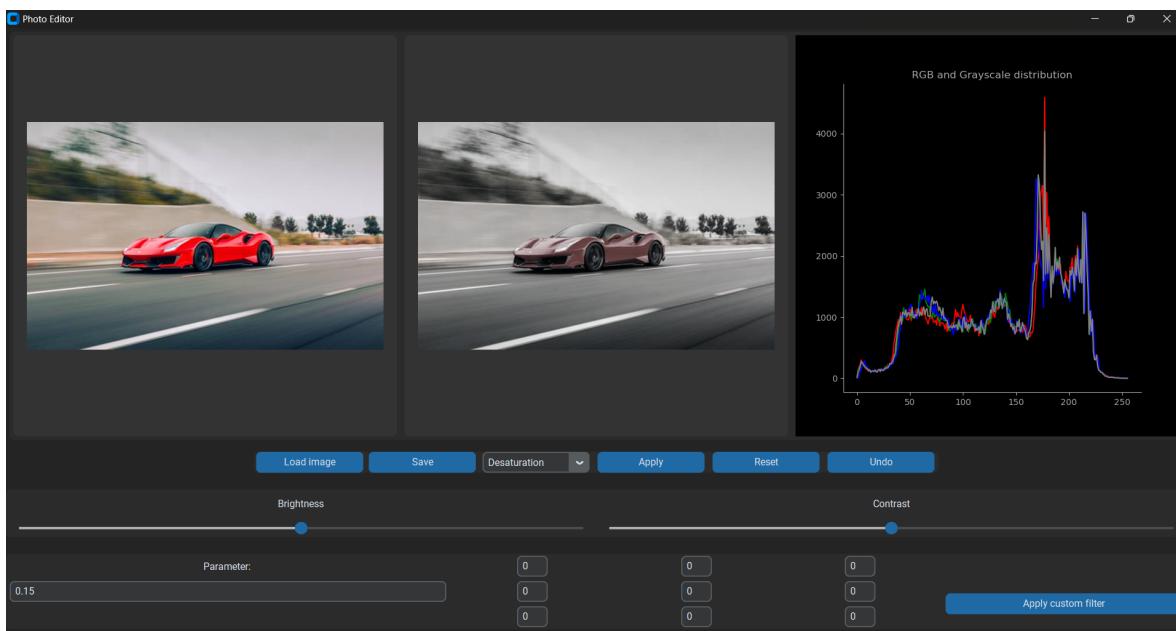
# 1 Opis aplikacji.

Aplikacja okienkowa "Edytor Obrazów" umożliwia podstawowe przetwarzanie obrazów, w tym filtrację oraz regulację parametrów jasności i kontrastu. Została zaimplementowana w Pythonie przy użyciu bibliotek customtkinter (interfejs graficzny), tkinter.filedialog (obsługa plików), numpy (operacje macierzowe), PIL (przetwarzanie obrazów) oraz matplotlib (wizualizacja histogramów). Obsługiwane są formaty plików PNG, JPG, JPEG i BMP.

Aplikacja pozwala użytkownikowi otwieranie obrazu poprzez wybranie zdjęcia z wybranego folderu na swoim komputerze, stosowanie filtrów na wybranym obrazie, cofanie wprowadzonych zmian do poprzedniej wersji (lub do wersji początkowej) oraz zapisywanie przetworzonego obrazu do wybranego pliku.

Filtры и методы przetwarzania obrazów wykorzystane w aplikacji, to zmiana jasności i kontrastu w czasie rzeczywistym, konwersja do skali szarości, negatyw, binaryzacja, zastosowanie filtra uśredniającego, zastosowanie filtra wyostrzającego, różne metody wykrywania krawędzi obrazu(np. krzyż Robertsa, operator Sobela) oraz wykorzystanie customowego filtra.

Oprócz samych filtrów aplikacja ma funkcjonalności takie jak wyświetlanie histogramów przedstawiających rozkład wartości pikseli poszczególnych kanałów R, G oraz B dla wybranego zdjęcia. Wyświetlana jest też projekcja pionowa i pozioma dla obrazu, na który nałożony został filtr binaryzacji. Poniżej przedstawiamy wygląd naszej aplikacji.



Rysunek 1: Interfejs aplikacji "Edytor Obrazów"

Aplikacja charakteryzuje się intuicyjnym interfejsem, estetycznym wyglądem, przyjazną obsługą i szybkim działaniem, co czyni ją idealnym narzędziem do podstawowej edycji obrazów.

## 2 Opis metod i filtrów.

### 2.1 Zmiana jasności i kontrastu.

Pierwszym udogodnieniem naszego programu jest manipulacja jasnością i kontrastem zdjęcia przy pomocy interaktywnych suwaków. Nasz program przechowuje kopię aktualnie przetwarzanego zdjęcia na którą nakłada zmiany wprowadzone przez użytkownika przy użyciu suwaków. W przypadku zmiany kontrastu najpierw obraz jest centrowany względem średniej wartości jasności. Następnie każdy piksel jest skalowany przez  $(1 + \alpha)$ , gdzie  $\alpha$  to wartość z suwaka z przedziału  $[-1,1]$ , co zwiększa lub zmniejsza różnice jasności między pikselami. Po tej operacji piksele są z powrotem przesuwane o średnią wartość jasności i obcinane do zakresu  $[0, 255]$ .

W przypadku zmiany jasności sytuacja jest dużo bardziej oczywista. Program zczytuje wartość  $\beta$  z suwaka o wartości z przedziału  $[-255, 255]$  i dodaje do każdego piksela tę wartość pamiętając o ograniczeniu się do zakresu  $[0, 255]$ , tak aby nie wyjść poza przedział odpowiedni dla pikseli.



Rysunek 2: Efekty zmian jasności i kontrastu: (lewo) zwiększona jasność, (środek) zwiększony kontrast, (prawo) zwiększona jasność i kontrast.

### 2.2 Konwersja do skali szarości - Uśrednianie

W przypadku konwersji obrazu do skali szarości, program dla każdego piksela oblicza średnią arytmetyczną wartości kanałów R, G i B. Dzięki temu wartości tych kanałów stają się identyczne, co prowadzi do uzyskania jednolitego odcienia szarości dla każdego piksela. Taki sposób konwersji jest efektywny i szybki, ponieważ wymaga jedynie prostego uśrednienia trzech składowych koloru.

Poniższy obraz przedstawia nałożony filtr szarości na oryginalne zdjęcie czerwonego ferrari.



Rysunek 3: Efekt konwersji obrazu do skali szarości: (lewo) oryginalne zdjęcie, (prawo) nałożony filtr szarości.

### 2.3 Konwersja do skali szarości - Metoda dekompozycji (Lightness)

Metoda Lightness przekształca obraz do skali szarości poprzez obliczenie średniej arytmetycznej z maksymalnej i minimalnej wartości spośród składowych R, G i B dla każdego piksela. Wzór można zapisać jako:

$$\text{grey} = \frac{\max(R, G, B) + \min(R, G, B)}{2}$$

Dla każdego piksela wyznaczona jest wartość maksymalna i minimalna spośród trzech kanałów (R, G, B), a następnie obliczana jest średnia tych dwóch wartości, dzięki czemu otrzymujemy charakterystycznie wyglądający obraz w skali szarości.



Rysunek 4: Porównanie metody Lightness: (lewo) oryginalny obraz, (prawo) wynik konwersji.

W przeciwieństwie do uśredniania, metoda nie uwzględnia wszystkich trzech kanałów równomiernie oraz daje mniej naturalne wyniki dla ludzkiej percepacji niż metoda dekompozycji (lightness). Zachowuje kontrast w obszarach o dominującym jednym kolorze, ale traci detale w obszarach o zrównoważonych składowych RGB. Metoda ta jest szybka obliczeniowo, ale rzadko używana w praktycznych zastosowaniach ze względu na nieliniowe zniekształcenie jasności.

## 2.4 Desaturacja

Desaturacja to proces zmniejszania intensywności kolorów w obrazie, prowadzący do częściowego lub całkowitego przejścia do skali szarości. W programie użytkownik może kontrolować stopień desaturacji za pomocą parametru z zakresu  $[0, 1]$ . Dla wartości 1 obraz pozostaje niezmieniony, a dla 0 całkowicie przekształca się do skali szarości. Wartości pośrednie mieszają oryginalne kolory z ich odcieniami szarości, co pozwala na płynne dostosowanie nasycenia obrazu, stopniowy zanik kolorów można zaobserwować na poniższych zdjęciach.



Rysunek 5: Efekt desaturacji obrazu: (lewo) oryginalne zdjęcie, (środek) desaturacja z parametrem 0.5, (prawo) desaturacja z parametrem 0.1.

Metoda desaturacji implementuje płynne przejście między obrazem kolorowym a skalą szarości realizowane przez interpolację liniową między wartościami pikseli oryginalnego obrazu a ich odpowiednikami w skali szarości:

$$\text{desaturated} = \alpha \cdot \text{RGB} + (1 - \alpha) \cdot \text{Luma}$$

gdzie:

- $\alpha$  - parametr desaturacji z zakresu  $[0, 1]$  ( $0$  = pełna skala szarości,  $1$  = oryginalny kolor)
- Luma - wartość luminancji obliczona jako ważona suma:  $0.299R + 0.587G + 0.114B$

Przekształcanie obraz w pełni do skali szarości polega na obliczeniu wartości jasności piksela jako średniej ważonej składowych R, G i B, z uwzględnieniem ludzkiej percepji kolorów. Zastosowano współczynniki 0.299 dla kanału czerwonego (R), 0.587 dla zielonego (G) i 0.114 dla niebieskiego (B), co odzwierciedla większą wrażliwość ludzkiego oka na kolor zielony. W efekcie, obraz w skali szarości zachowuje naturalną percepcję jasności, a wynik jest bardziej zbliżony do rzeczywistego postrzegania kolorów niż w przypadku prostego uśrednienia.

## 2.5 Negatyw

W celu uzyskania negatywu obrazu, program wykonuje operację, która polega na zamianie wartości kolorów każdego piksela na ich odwrotność. Dla każdego piksela, wartości kanałów R, G, B są obliczane jako różnica między maksymalną wartością (255) a wartością danego kanału. W efekcie jasne obszary obrazu stają się ciemniejsze, a ciemne obszary jaśniejsze, tworząc charakterystyczny efekt negatywu. Po zastosowaniu tej operacji obraz zyskuje odwrotność swoich pierwotnych kolorów, tworząc klasyczny efekt negatywu. Możemy zauważać to zjawisko na poniższym porównaniu oryginalnego zdjęcia ze zdjęciem po zastosowaniu negatywu.



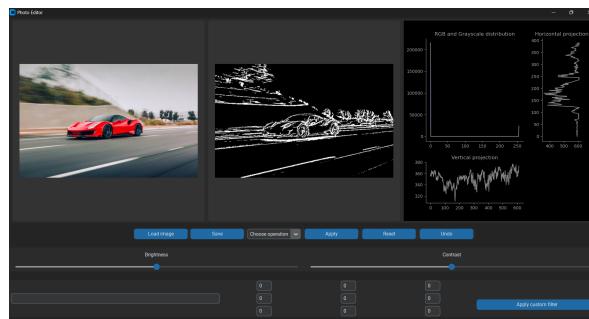
Rysunek 6: Efekt konwersji obrazu do negatywu: (lewo) oryginalne zdjęcie, (prawo) nałożony negatyw.

## 2.6 Binaryzacja

Binaryzacja obrazu to proces konwersji kolorowego obrazu lub obrazu w odcieniach szarości do postaci czarno-białej, w której piksele przyjmują jedną z dwóch wartości – 0 (czarny) lub 255 (biały). Program umożliwia ustawienie własnego progu (threshold) do przeprowadzenia tej operacji.

Proces działa w następujący sposób: dla każdego piksela obrazu program porównuje jego wartość jasności z wartością progu. Jeśli jasność piksela jest większa lub równa ustawionemu thresholdowi, piksel zostaje przypisany do wartości 255 (biały). W przeciwnym razie, jeśli jasność piksela jest mniejsza od progu, przypisywana jest wartość 0 (czarny). Dzięki tej operacji, obraz staje się wyraźnie rozdzielony na obszary jasne i ciemne, co pozwala na wydobycie kontrastujących elementów, takich jak kontury lub istotne detale obrazu.

Podczas gdy na zdjęcie nałożymy filtr binaryzacji, to nasza aplikacja automatycznie wykryje tę zmianę i wyświetli wykresy projekcji pionowej i poziomej w pobliżu histogramu. Dzięki tym wykresom możemy wyraźnie wskazać, gdzie znajdują się konkretne obiekty, na przykład przy detekcji tączówki oka.



Rysunek 7: Wykresy projekcji pionowej i poziomej binaryzowanego zdjęcia.

Użytkownik może dostosować wartość progu, co wpływa na ostateczny wygląd binaryzowanego obrazu. Poniżej przedstawiamy binaryzację obrazu z przykładowymi wartościami thresholdu.

Jak można zauważać im wyższa wartość thresholdu, tym więcej pikseli zostanie zaklasyfikowanych jako czarne (zwiększąc obszary ciemne), a im niższa wartość, tym więcej pikseli stanie się białych (powiększając obszary jasne). Dzięki tej funkcji program daje większą kontrolę nad procesem binaryzacji, umożliwiając dostosowanie efektu do różnych typów obrazów i potrzeb analizy.



Rysunek 8: Efekt zmiany thresholdu w binaryzacji: (lewo) threshold 64, (środek) threshold 128, (prawo) threshold 192.

## 2.7 Filtr rozmywający z regulowanym jądrem

Filtr rozmywający w implementuje adaptacyjne rozmycie poprzez odpowiednio skonstruowane jądro  $3 \times 3$  o następującej strukturze:

$$\text{jądro} = \frac{1}{8 + \alpha} \begin{bmatrix} 1 & 1 & 1 \\ 1 & \alpha & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

gdzie  $\alpha$  to parametr wprowadzany przez użytkownika (domyślnie 1), który kontroluje stopień centralnego ważenia.

- **Dynamiczna normalizacja:** Jądro jest automatycznie normalizowane poprzez dzielenie przez sumę wszystkich elementów (w tym parametru  $\alpha$ ), co zapobiega zmianom ogólnej jasności obrazu.
- **Regulacja efektu:** Wartość  $\alpha$  pozwala kontrolować charakter rozmycia:
  - Dla  $\alpha > 1$ : subtelniejsze rozmycie
  - Dla  $\alpha = 1$ : klasyczne rozmycie uśredniające
  - Dla  $0 < \alpha < 1$ : zwiększyły wpływ sąsiedztwa, intensywniejsze rozmycie

Filtr rozmywający stosuje operację splotu z wyżej opisanym jądrem. Celem jest wygładzenie obrazu poprzez zmianę stosunku wartości pikseli w sąsiedztwie piksela do środkowego piksela.

Podczas splotu, dla każdego piksela w obrazie, wybierany jest region o rozmiarze  $3 \times 3$  wokół niego. Każdy z tych pikseli jest mnożony przez odpowiadającą mu wagę w jądrze, a następnie wartości są sumowane. Otrzymana suma daje nową wartość piksela w wynikowym obrazie. Zaletą operacji jest wygładzanie obrazu, redukcja szumów i wygładzone krawędzie.

Proces ten powtarza się dla każdego piksela w obrazie, a wynikowa mapa pikseli tworzy nowy obraz, który jest następnie wyświetlany. Oto kilka wyników rozmycia:



Rysunek 9: Efekt rozmycia obrazu: (lewo) oryginalne zdjęcie, (środek) uśrednienie ( $\alpha = 1$ ), (prawo) rozmycie ( $\alpha = 0.1$ ).

Po zastosowaniu filtru możemy zauważać, że obraz staje się bardziej wygładzony, a szczegółowy, takie jak drobne krawędzie czy szумy, zostają zredukowane. Usuwa on wysokie częstotliwości w obrazie, co powoduje, że widoczność szczegółów jest ograniczona, a powierzchnie stają się bardziej jednolite. W efekcie obraz może wyglądać bardziej zamglony lub rozmyty, ponieważ w wyniku uśredniania piksele stają się bardziej podobne do siebie, co wygładza różnice między sąsiadującymi obszarami.

## 2.8 Filtr wyostrzający

W przypadku filtru wyostrzającego zasada działania jest identyczna jak w filtrze rozmywającym. Jedyną różnicę, aczkolwiek bardzo znaczącą jest inne jądro (kernel) postaci

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

Po zastosowaniu tego filtru wyostrzającego obraz staje się bardziej wyraźny, a krawędzie stają się ostrzejsze. Kernel użyty w tym przypadku, który zawiera elementy ujemne wokół centralnego elementu, ma na celu wzmacnianie różnic między sąsiednimi pikselami. W wyniku działania tego filtru, wartości pikseli są modyfikowane tak, by kontrast między nimi się zwiększył, co powoduje, że szczegóły obrazu, takie jak krawędzie, stają się bardziej widoczne. Filtr wyostrzajający działa poprzez zwiększenie intensywności różnic w sąsiednich pikselach, co skutkuje efektem wzmacnionych konturów i wyraźniejszych detali. To zjawisko przedstawia poniższe zestawienie oryginalnego zdjęcia oraz zdjęcia po zastosowaniu filtra wyostrzającego.



Rysunek 10: Efekt wyostrzania obrazu: (lewo) oryginalne zdjęcie, (prawo) nałożony filtr wyostrzajjące.

## 2.9 Filtr Gaussa

Filtr Gaussa służy do wygładzania obrazu poprzez usuwanie szumów i drobnych detali, przy jednoczesnym zachowaniu głównych krawędzi. W programie filtr ten jest implementowany jako splot z jądrem Gaussa, którego rozmiar i wartości zależą od parametru sigma. Im większa wartość sigma, tym silniejsze rozmycie. Kernel jest generowany dynamicznie na podstawie wzoru Gaussa, co zapewnia płynne ważenie pikseli w sąsiedztwie, z większym naciskiem na piksele bliżej centrum. Efektem działania filtru jest rozmycie obrazu, które redukuje szумy i przygotowuje obraz do dalszego przetwarzania, np. wykrywania krawędzi.



Rysunek 11: Efekt po zastosowaniu filtru Gaussa: (lewo) oryginalne zdjęcie, (środek) sigma = 1, (prawo) sigma = 10.

## 2.10 Znajdowanie krawędzi - Laplasjan

W przypadku tego filtra wykorzystujemy kernel postaci

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Działa on na zasadzie analizy różnic intensywności pomiędzy pikselami i ich sąsiadami w okolicy. Jeśli wokół piksela nie występuje wyraźna zmiana intensywności (czyli brak krawędzi), to różnice między sąsiednimi pikselami będą małe, a wynik splotu będzie bliski零. W przypadku wystąpienia krawędzi, różnice intensywności w okolicy piksela będą duże, a wynik splotu będzie większy, co wskazuje na obecność krawędzi w obrazie. Poniżej przedstawiamy wynik tej operacji.



Rysunek 12: Efekt znajdowania krawędzi: (lewo) oryginalne zdjęcie, (środek) nałożony filtr znajdowania krawędzi, (prawo) zalecany negatyw dla lepszej czytelności.

Zauważmy, że filtr znajdowania krawędzi zwraca obraz na czarnym tle. Naszą propozycją do łatwiejszej analizy krawędzi jest dodatkowe zadzialanie na to zdjęcie negatywem. Otrzymujemy wówczas kontur obiektów obrazu na białym tle.

## 2.11 Metody gradientowe wykrywania krawędzi

### Zasady działania

Metody gradientowe wykrywają krawędzie poprzez obliczenie pochodnych kierunkowych obrazu:

1. Obliczane są gradienty w kierunkach poziomym ( $G_x$ ) i pionowym ( $G_y$ ) poprzez splot z odpowiednimi jądrami
2. Wielkość gradientu ( $G$ ) wyznaczana jest jako średnia kwadratowa:  $G = \sqrt{G_x^2 + G_y^2}$
3. Wynik normalizowany jest do zakresu [0,255]

### Implementowane operatory

**Sobela:**  $K_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad K_y = K_x^T$

**Prewitta:**  $K_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad K_y = K_x^T$

**Robertsza:**  $K_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad K_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$

**Sobela-Feldmana:**

$$K_x = \begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix} \quad K_y = K_x^T$$

**Scharra:**

$$K_x = \begin{bmatrix} 47 & 0 & -47 \\ 162 & 0 & -162 \\ 47 & 0 & -47 \end{bmatrix} \quad K_y = K_x^T$$

Jądra przed dokonaniem splotu są normalizowane przez sumę wartości bezwzględnych wag:

$$\text{norm} = \sum |k_{ij}| \quad K_{normalized} = \frac{1}{\text{norm}} K$$

### Obliczanie gradientu

1. Dla każdego piksela obliczane są dwie pochodne:

$$G_x = I * K_x$$

$$G_y = I * K_y$$

gdzie  $*$  oznacza operację splotu

2. Wielkość gradientu:

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

3. Normalizacja wynikowa:

$$G_{final} = \frac{G}{G_{\max}} \cdot 255$$

### Charakterystyka operatorów

- **Operator Sobela** – wykorzystuje jądra wykrywające poziome i pionowe krawędzie, a wynik jest kombinacją obu kierunków.
- **Operator Prewitta** - Podobny do Sobela, prostsze jądro, bardziej wrażliwy na szumy, ale lepiej wykrywa słabe krawędzie
- **Krzyż Robertsa**: Działa na zasadzie różniczkowania po przekątnych, mały rozmiar jądra czyni go wrażliwym na szumy, dobry do wykrywania ostrych krawędzi
- **Sobel-Feldman**: Wzmocnione jądro dla lepszego wykrywania ukośnych krawędzi
- **Operator Scharra** – wzmocniona wersja operatora Sobela, lepiej wykrywająca krawędzie pod kątem.

Wartości gradientu reprezentują siłę krawędzi – im wyższa wartość, tym wyraźniejsza krawędź. Efektem jest obraz przedstawiający krawędzie na ciemnym tle, który można przekształcić negatywem dla lepszej widoczności. Oto wyniki przekształcenia obrazu przy pomocy wymienionych operatorów:



Rysunek 13: Efekt zastosowania wykrywania krawędzi: (lewo) oryginalne zdjęcie, (środek) operator Sobela, (prawo) operator Prewitta.



Rysunek 14: Efekt zastosowania wykrywania krawędzi: (lewo) krzyż Robertsa, (środek) operator Sobela-Feldmana, (prawo) operator Scharr'a.

## 2.12 Własny filtr

Nasza aplikacja daje użytkownikowi możliwość zadziałania na obraz własnym filtrem. Do dyspozycji jest kernel o wymiarach 3 na 3.

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ a_7 & a_8 & a_9 \end{bmatrix}$$

Nie ograniczamy użytkownika. Może on wprowadzić do komórek macierzy dowolne wartości.  $\forall_{i \in [9]} a_i \in (-\infty, +\infty)$

### 3 Wnioski

- Wydajność przetwarzania piksel po pikselu – Implementacja filtrów działających na pojedynczych pikselach w pętli for okazała się mało efektywna czasowo, szczególnie dla dużych obrazów.
- Jakość wyników filtracji – Zastosowane metody przetwarzania obrazów pozwoliły na skuteczne modyfikowanie ich właściwości wizualnych, takich jak jasność, kontrast czy ostrość. Jednak niektóre filtry, jak np. binaryzacja, mogą wymagać dostosowania parametrów progowych w zależności od analizowanego obrazu.
- Różnice w metodach konwersji do skali szarości – Wykorzystanie różnych metod (uśrednianie, dekompozycja, desaturacja) pokazało, że efekt końcowy zależy od sposobu przekształcania kanałów kolorystycznych. Wybór odpowiedniej metody może wpływać na dalsze etapy przetwarzania obrazu, np. wykrywanie krawędzi.
- Filtry wykrywające krawędzie – Operator Sobela i krzyż Robertsa dobrze uwidaczniają krawędzie, ale są wrażliwe na szum. Wprowadzenie filtra Gaussa przed wykrywaniem krawędzi mogłoby poprawić jakość wyników.
- Histogramy i analiza obrazu – Możliwość wizualizacji rozkładu wartości pikseli ułatwiła ocenę wpływu filtrów na obraz. Przykładowo, histogramy RGB pokazały zmiany w rozkładzie jasności po zastosowaniu różnych filtrów.