

# **Instrukcja obsługi aplikacji**

## **„Wykrywanie Krawędzi w Obrazach Kolorowych”**

Akademia Tarnowska

**Autorzy projektu:**

**Bartłomiej Gwóźdź**

**Jakub Jarosz**

**Mikołaj Czernik**

# 1. Uruchomienie programu

Po uruchomieniu aplikacji zobaczysz główne okno, w którym znajdują się:

- górny pasek sterowania,
- sekcja wyboru języka,
- informacja o stanie obrazu,
- obszar przewijany z modułami analizy.

Aplikacja automatycznie tworzy jedną ramkę porównawczą, gotową do pracy.

# 2. Wczytywanie obrazu

1. Kliknij „**Wybierz obraz**”.
2. Wskaż plik graficzny (obsługiwane: PNG, JPG, JPEG).
3. Po wczytaniu:
  - obraz zostanie wyświetlony w każdej istniejącej ramce,
  - status zmieni się na „Wczytano: ...”.

# 3. Dodawanie i usuwanie ramek porównawczych

## Dodawanie

- Kliknij przycisk „**+**”, aby dodać kolejną ramkę porównawczą.
- Nowa ramka automatycznie przejmuje aktualnie wczytany obraz.

## Usuwanie

- Aby usunąć wybraną ramkę → użyj przycisku „**-**” znajdującego się w jej prawym górnym rogu.

## 4. Konfiguracja parametrów analizy

W każdej ramce możesz ustawić parametry przetwarzania:

### 4.1. Przestrzeń barw\*

Wybierz jedną z opcji:

- **RGB**
- **HSV**
- **LAB**
- **CMYK**

### 4.2. Metoda detekcji krawędzi\*

Najechnie na listę wyboru, po wybraniu konkretnej metody, prezentuje wykorzystane do niej maski.

Dostępne algorytmy:

- **Sobel** - Metoda ta wykorzystuje operację splotu (konwolucji) obrazu z dwiema macierzami (jądrami)  $3 \times 3$ , aby wykryć przybliżone pochodne intensywności w kierunku poziomym ( ) i pionowym ( ).

W implementacji zastosowano następujące maski:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Wynikowa krawędź jest obliczana jako magnituda gradientu przy użyciu funkcji przeciwprostokątnej:  $G = \sqrt{G_x^2 + G_y^2}$ . Wynik jest następnie normalizowany do zakresu 0–255.

- **Sobel CV2** - Jest to alternatywna wersja detekcji krawędzi wykorzystująca zoptymalizowaną funkcję biblioteczną cv2. Sobel. W przeciwieństwie do implementacji opartej na splotach, metoda ta korzysta z wewnętrznych mechanizmów OpenCV (często wspieranych instrukcjami procesora SIMD), co może zapewniać wyższą wydajność przy przetwarzaniu dużych obrazów.

- **Laplacian (4- i 8-sąsiedztwa)** - Filtr Laplasjanu jest operatorem drugiego rzędu, który wykrywa obszary gwałtownych zmian jasności (przejścia przez zero drugiej pochodnej). Aplikacja oferuje dwa warianty jąder konwolucyjnych:

**4-sąsiedztwo (Krzyż):** Uwzględnia tylko bezpośrednich sąsiadów piksela (góra, dół, lewo, prawo).

$$K_4 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

**8-sąsiedztwo (Pełne):** Uwzględnia również piksele po przekątnych, co czyni detekcję bardziej czułą na szumy, ale dokładniejszą w wykrywaniu skosów.

$$K_8 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- **Laplacian LoG** - Jest to rozszerzona wersja filtru Laplace'a, która przed wykrywaniem krawędzi stosuje wygładzanie Gaussa w celu redukcji szumów. W aplikacji zaimplementowano przybliżenie tego filtru za pomocą stałego jądra o wymiarach 5×5:

$$K_{\log} = \begin{bmatrix} 0 & 0 & -1 & 0 & 0 \\ 0 & -1 & -2 & -1 & 0 \\ -1 & -2 & 16 & -2 & -1 \\ 0 & -1 & -2 & -1 & 0 \\ 0 & 0 & -1 & 0 & 0 \end{bmatrix}$$

- **Scharr** - Algorytm Scharra działa analogicznie do operatora Sobela, ale wykorzystuje inne wagi w jądrach splotu. Został zaprojektowany tak, aby zapewnić lepszą symetrię obrotową i wykrywać krawędzie z większą dokładnością, szczególnie te, które nie biegną wzdłuż osi X lub Y.

$$G_x = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix}, \quad G_y = \begin{bmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{bmatrix}$$

- **Prewitt** - Jest to prostszy wariant operatora gradientowego. W przeciwieństwie do Sobela, nie kładzie on większego nacisku na piksele znajdujące się bliżej centrum maski. Jest mniej odporny na szum, ale szybszy obliczeniowo (choć przy współczesnych procesorach różnica jest pomijalna).

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

- **Canny** - W aplikacji zaimplementowano autorską wersję algorytmu Canny ("Soft Canny"), która realizuje następujące kroki:

1. Rozmycie Gaussa: Obraz jest wygładzany filtrem 5×5 ( $\sigma=1.4$ ) w celu usunięcia szumu.
2. Obliczenie gradientów: Wykorzystywane są maski Sobela do uzyskania magnitudy i kąta nachylenia krawędzi.
3. Tłumienie niemaksimum (Non-Maximum Suppression): Algorytm sprawdza piksele wzdłuż kierunku gradientu (zobróconego do jednego z czterech sektorów: 0°, 45°, 90°, 135°). Jeśli piksel nie jest lokalnym maksimum, jego wartość jest zerowana. Pozwala to na pocienienie krawędzi do szerokości jednego piksela.

- **Canny CV2** - Opcja ta wykorzystuje biblioteczną funkcję cv2.Canny z pakietu OpenCV. W odróżnieniu od implementacji własnej, wersja ta stosuje pełne progowanie z histerezą. Oznacza to użycie dwóch progów (dolnego i górnego):
  - Piksele o gradiencie powyżej górnego progu są pewnymi krawędziami.
  - Piksele poniżej dolnego progu są odrzucane.
  - Piksele pomiędzy progami są uznawane za krawędzie tylko wtedy, gdy są połączone z "pewną" krawędzią.

- **Roberts** - Najprostszy i najstarszy operator gradientowy, wykorzystujący małe jądra 2×2. Oblicza on różnice dyskretne na przekątnych. Ze względu na mały rozmiar maski, algorytm jest bardzo wrażliwy na szum, ale precyzyjnie lokalizuje krawędzie w obrazach o wysokim kontraście.

$$G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

### 4.3. Progi

Ustaw progi:

- **Low Threshold**
- **High Threshold**

### 4.4. Binaryzacja

Zaznaczenie pola „**Binarization**” powoduje zamianę wyniku na obraz 0–1 (czarno-biały).

**\* W celu ułatwienia dokonywania porównań i znajdowaniu różnic, możliwe jest ustawienie GLOBALNIE konkretnych ustawień przestrzeni barw i/lub metody detekcji krawędzi dla wszystkich aktualnie otwartych ramek**

## 5. Uruchamianie detekcji krawędzi

Aby przeprowadzić analizę:

1. Upewnij się, że obraz jest wczytany.
2. Ustaw parametry w wybranych ramkach.
3. Kliknij „**Uruchom funkcję**”.

Po chwili zobaczysz:

- obraz źródłowy,
- osobne obrazy krawędzi dla każdego kanału,
- obraz sumaryczny.

Każdy wynik pojawi się w osobnym oknie wewnątrz ramki.

## 6. Podgląd powiększony

Możesz obejrzeć każdy obraz w większej skali:

1. Kliknij lewym przyciskiem myszy miniaturę.
2. Na środku ekranu pojawi się powiększony podgląd.
3. Podgląd zamknie się po puszczeniu przycisku myszy.

Alternatywnie, aby obraz otworzył się w powiększeniu bez potrzeby ciągłego trzymania przycisku myszy, należy kliknąć na dany obraz prawym przyciskiem myszy.

Obraz zostanie wtedy otwarty w dodatkowym, nowym oknie.

W ten sposób można wyświetlić wiele obrazów jednocześnie.

## 7. Zapisywanie wyników

Aby zapisać dowolny obraz:

1. Kliknij przycisk „**Zapisz**” pod wybraną miniaturą.
2. Wybierz lokalizację zapisu.
3. Plik zostanie zapisany jako PNG.

Domyślnie proponowana nazwa zawiera:

- nazwę pliku wejściowego,
- nazwę kanału/metody.

Wszystkie obrazy możesz zapisywać do folderu **Zapisane obrazy wynikowe**.

## 8. Zmiana języka interfejsu

W prawym górnym rogu znajduje się lista wyboru języka:

Możesz przełączyć:

- **Polski (PL)**
- **Angielski (EN)**

Zmieniają się wszystkie napisy w aplikacji, również te w ramkach.

## 9. Przewijanie okna

- Jeżeli masz więcej niż jedną ramkę, możesz przewijać listę kółkiem myszy.
- Przy jednej ramce przewijanie w górę jest blokowane (dla stabilności UI).

## 10. Najczęstsze problemy

**Brak obrazu** → komunikat o konieczności wczytania.

**Błędne parametry / nieobsługiwany przypadek** → wyświetlane jako błąd.

**Nie można zapisać obrazu** → pojawi się okno z informacją o przyczynie.