

Grupa: -----
Numer albumu: -----

ImagePy

Projekt zaliczający przedmiot Algorytmy Przetwarzania Obrazów

Tytuł projektu:

„Implementacja funkcji linii profilu wzdłuż wskazanej prostej lub krzywej w obrazach monochromatycznych i kolorowych oraz implementacja operacji filtracji logicznych na obrazach binarnych.”

Projekt nr 3

Autor projektu: Kacper Madej
Język programowania: Python 3.10
Prowadzący: dr. hab. Anna Korzyńska
2022/2023 Warszawa

Spis treści

1.	Wymagania techniczne	4
1.1.	Plik wykonywalny	4
1.2.	Rozwój oprogramowania	4
2.	Podstawowe funkcjonalności	4
2.1.	Wymagania i uruchomienie aplikacji.....	4
2.2.	Otwieranie obrazu, wielu obrazów oraz wybieranie aktywnego obrazu.....	4
2.3.	Duplikacja obrazu	6
2.4.	Zapisywanie obrazu na dysk.....	6
2.5.	Wyjście z aplikacji	7
3.	Analiza obrazów	7
3.1.	Histogram dla obrazów kolorowych oraz monochromatycznych.....	7
3.2.	Eksport składowych wektora cech obiektu binarnego do pliku obsługiwanej przez program Excel.....	9
4.	Podstawowe operacje na obrazach monochromatycznych.....	10
4.1.	Progowanie obrazu.....	10
4.1.1.	Progowanie z zachowaniem poziomów jasności	11
4.1.2.	Progowanie binarne	12
4.1.3.	Progowanie metodą Otsu.....	12
4.1.4.	Progowanie adaptacyjne	13
4.2.	Negacja	13
5.	Przetwarzanie obrazów monochromatycznych	15
5.1.	Wyrównanie histogramu.....	15
5.2.	Liniowe rozciąganie histogramu.....	16
5.3.	Nieliniowe rozciąganie współczynnikiem gamma.....	17
6.	Operacje matematyczne na obrazach.....	18
6.1.	Dodawanie i odejmowanie obrazów	19
7.	Operacje logiczne na obrazach.....	19
7.1.	And	20
7.2.	Or	20
7.3.	Xor	20
8.	Operacje morfologii matematycznej na obrazach binarnych	21
8.1.	Erozja	22
8.2.	Dylacja	22
8.3.	Otwarcie	23
8.4.	Zamknięcie.....	23

9.	Wykrywanie krawędzi w obrazie monochromatycznym.....	24
9.1.	Wykrywanie krawędzi operatorami:	24
9.1.1.	kierunkowymi Sobela	24
9.1.2.	Sobela i Prewitta.....	26
9.1.3.	Canny'ego	28
10.	Wygładzanie obrazu monochromatycznego	29
10.1.	Wygładzanie liniowe oparte na masce:.....	29
10.2.	Filtr medianowy.....	30
11.	Wyostrzanie obrazu monochromatycznego	32
12.	Projekt	33
12.1.	Wprowadzenie teoretyczne	33
12.1.1.	Linia profilu.....	33
12.1.2.	Filtracja logiczna	33
12.2.	Opis implementacji.....	34
12.3.	Opis funkcjonalności.....	35
12.3.1.	Linia profilu.....	35
12.3.2.	Filtracja logiczna	38

1. Wymagania techniczne

1.1. Plik wykonywalny

Do uruchomienia aplikacji potrzebny jest jedynie plik wykonywalny *project.exe*. Aplikacja została stworzona z przeznaczeniem na system Windows 10 na sprzęt zgodny z minimalnymi wymaganiami zgodnymi z tym systemem operacyjnym. Dodatkowo do prawidłowej obsługi aplikacji wymagana jest myszka, klawiatura oraz ekran (program nie posiada wersji konsolowej). Do obsługi aplikacji nie wymagane jest dodatkowe oprogramowanie.

1.2. Rozwój oprogramowania

Środowisko programistyczne wymaga uprzednio przygotowanego interpretera języka Python 3.10 wraz z managerem paczek *pip*. Instrukcja konfiguracji środowiska programistycznego (z poziomu katalogu w którym umieszczony jest projekt):

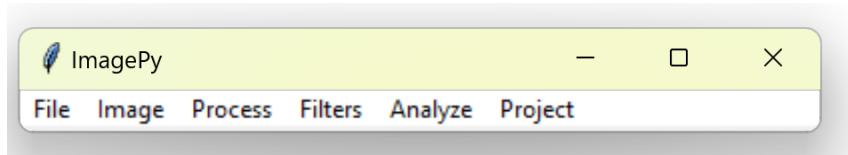
1. Instalacja zależności zewnętrznych: *pip install -r requirements.txt*
2. Wejście do katalogu *src*: *cd src/*
3. Uruchomienie aplikacji: *python app.py*

2. Podstawowe funkcjonalności

2.1. Wymagania i uruchomienie aplikacji

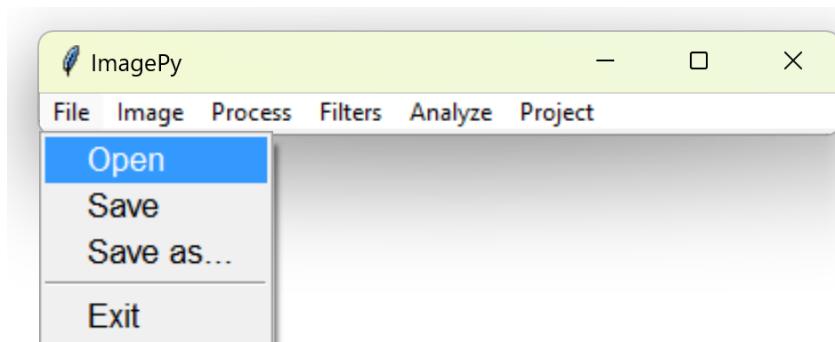
Aplikacja została napisana w języku Python 3.10 oraz przekształcona do pojedynczego pliku wykonywalnego *.exe* przeznaczonego do uruchamiania na systemie Windows 10. Aby uruchomić aplikację należy uruchomić plik *ImagePy.exe*

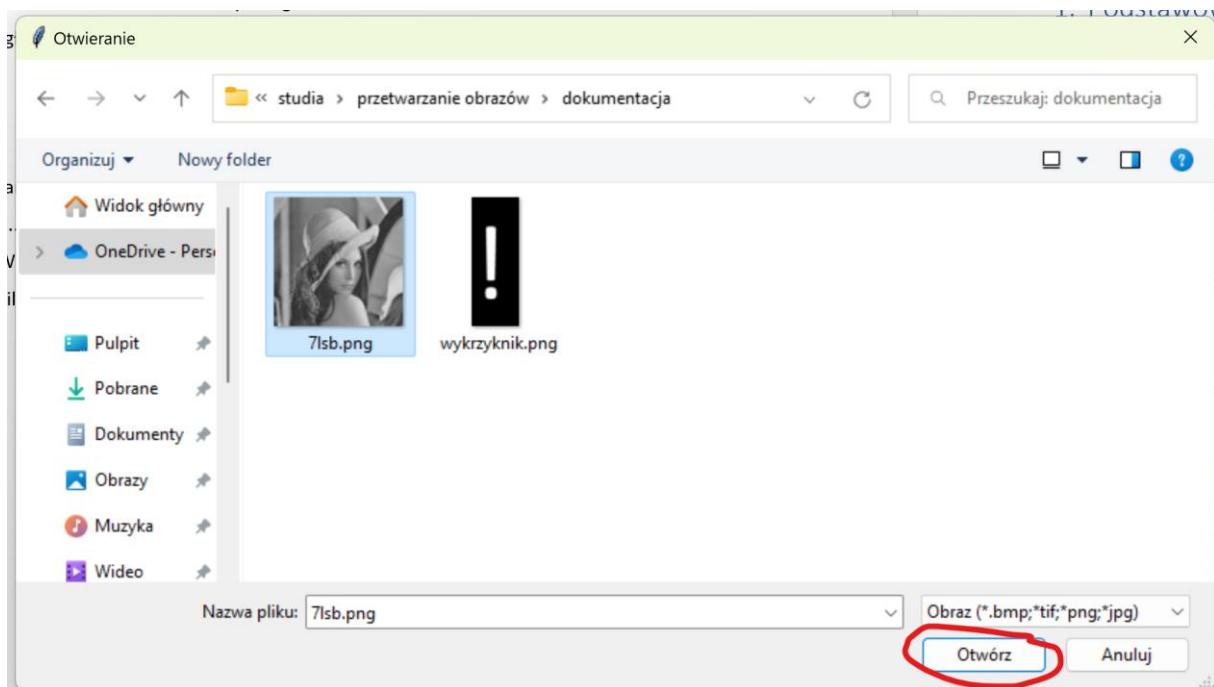
Po uruchomieniu pokaże się główne okno aplikacji:



2.2. Otwieranie obrazu, wielu obrazów oraz wybieranie aktywnego obrazu

Aby otworzyć pojedynczy obraz należy wybrać **File -> Open -> wybrać obraz, który chcemy otworzyć -> Otwórz**

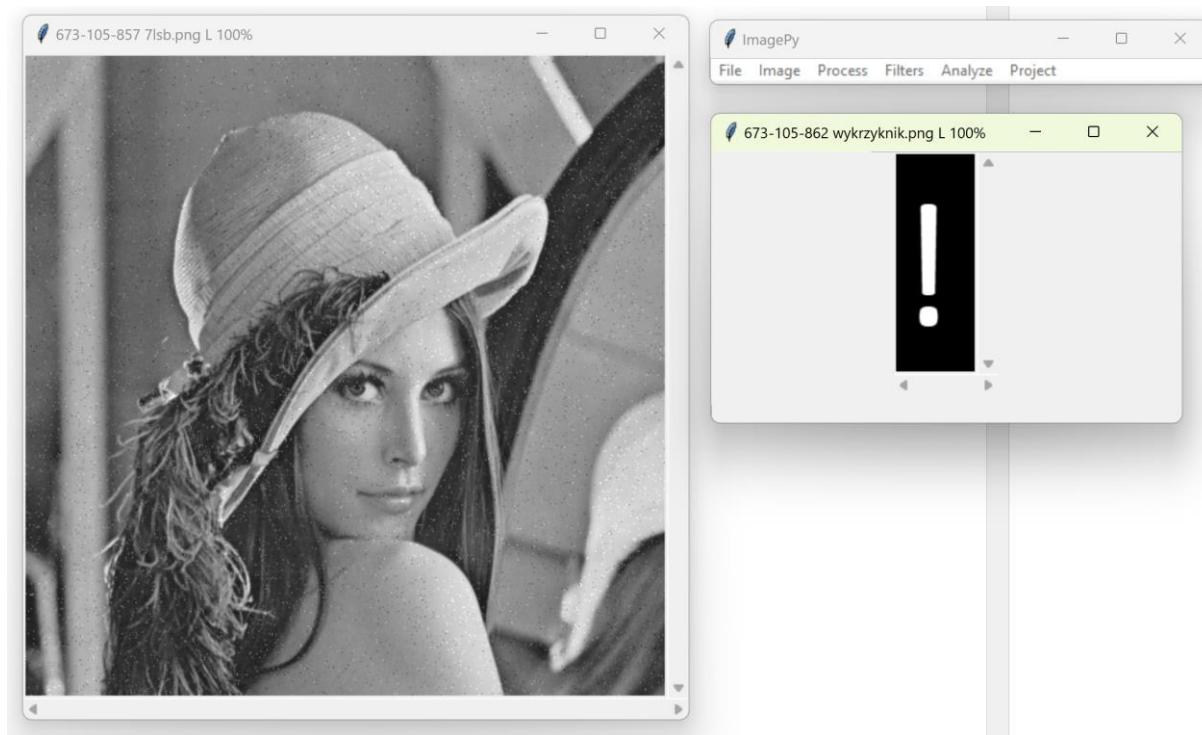




Nazwa okna otwartego obrazu składa się z:

- Unikalnego identyfikatora obrazu w formacie xxx-xxx-xxx
- Nazwy pliku np. lena.png
- Typu obrazu (przykładowe typy: „L” dla obrazu monochromatycznego, „RGB” dla obrazu kolorowego)
- Skala w jakiej obraz jest wyświetlany, domyślnie 100%

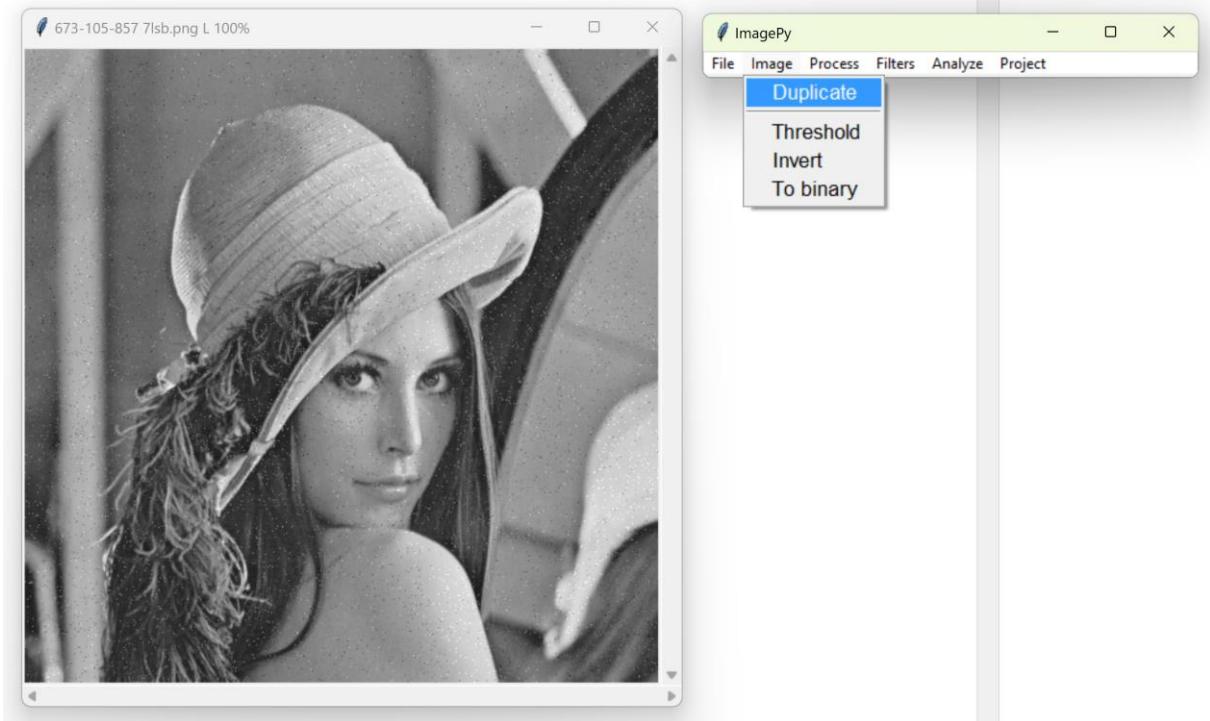
Aplikacja może mieć wiele obrazów otwartych w danym momencie. Pozostałe obrazy otwiera się analogicznie jak otwieranie pojedynczego obrazu.



Większość zaimplementowanych funkcjonalności wymaga wybrania aktywnego obrazu poprzez kliknięcie lewym przyciskiem myszki na obraz, który ma zostać wybrany, przed uruchomieniem danej funkcjonalności.

2.3. Duplikacja obrazu

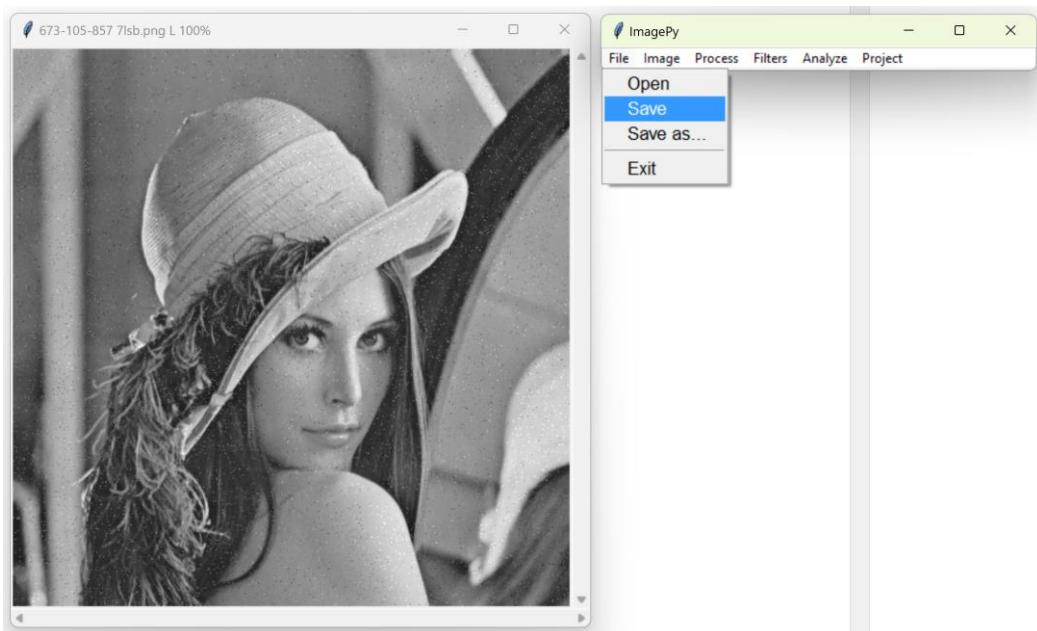
Duplikacja obrazu: **wybrać aktywny obraz -> Image -> Duplicate**



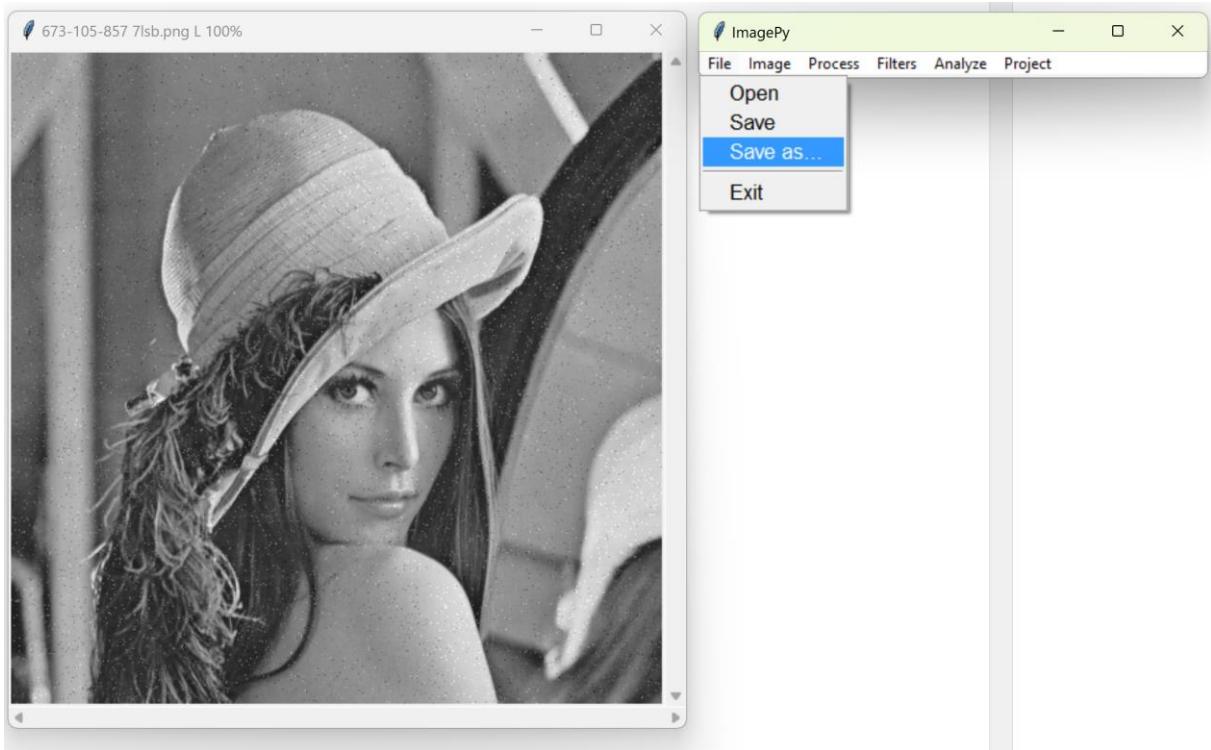
2.4. Zapisywanie obrazu na dysk

Zapisywanie modyfikacji na pliku bez zmiany nazwy: **wybrać aktywny obraz -> File -> Save**

Zmodyfikowany plik nadpisze plik źródłowy zachowując tą samą nazwę.

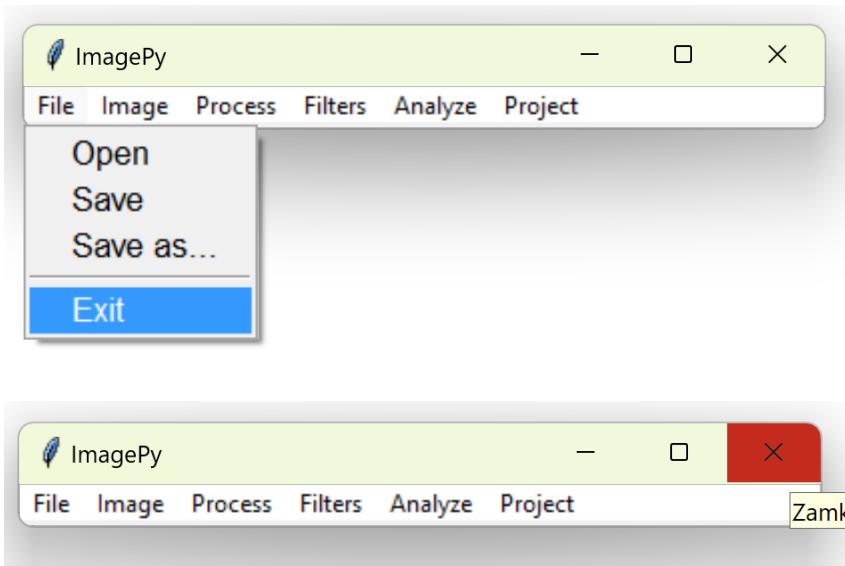


Zapisywanie pliku ze zmianą nazwy (lub nowego pliku utworzonego w programie): **wybierz aktywny obraz -> File -> Save as...**



2.5. Wyjście z aplikacji

Aby wyjść z aplikacji można wybrać: **File -> Exit** lub wybrać ikonę czerwonego krzyża w prawym górnym rogu okna programu.

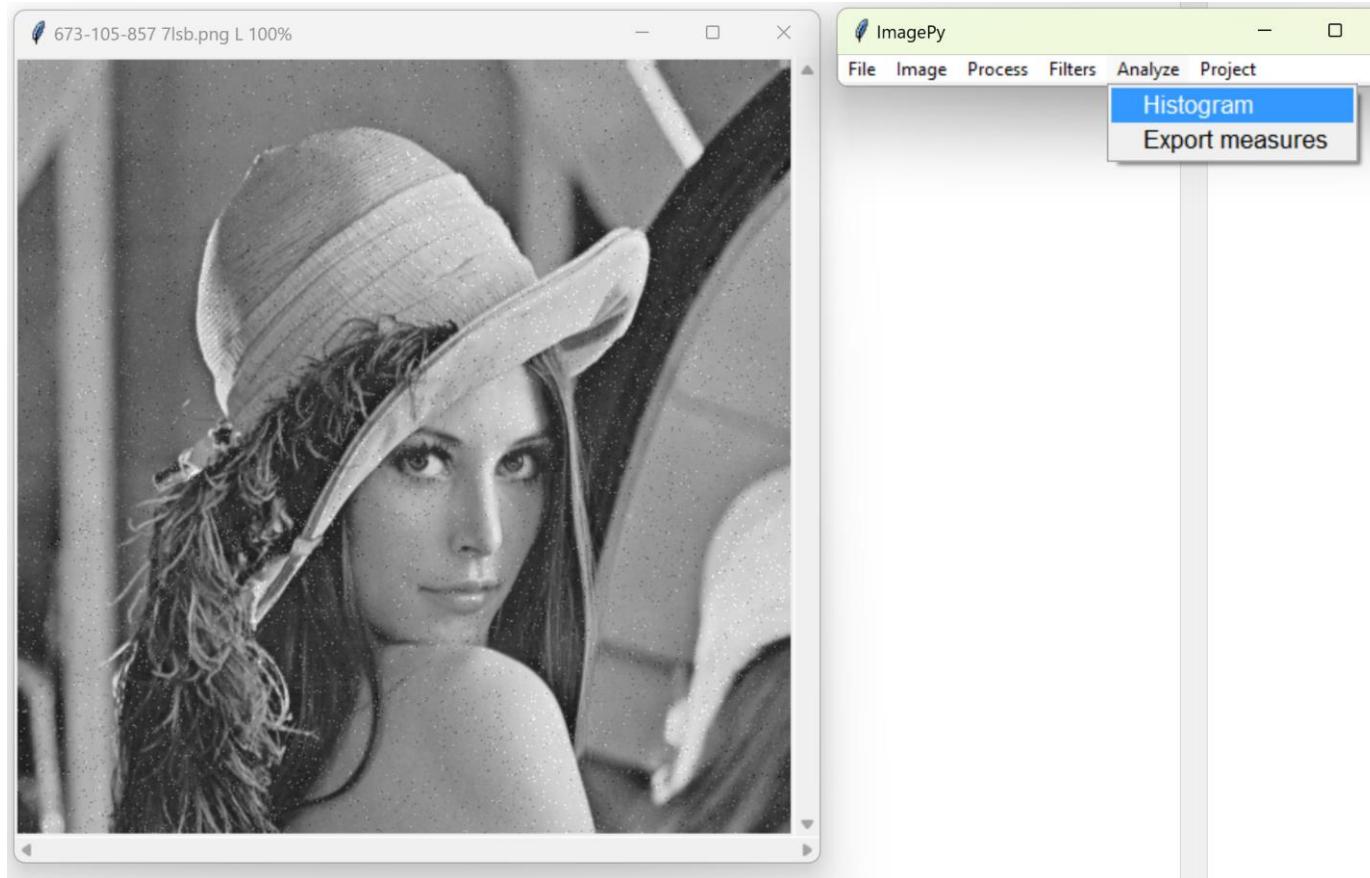


3. Analiza obrazów

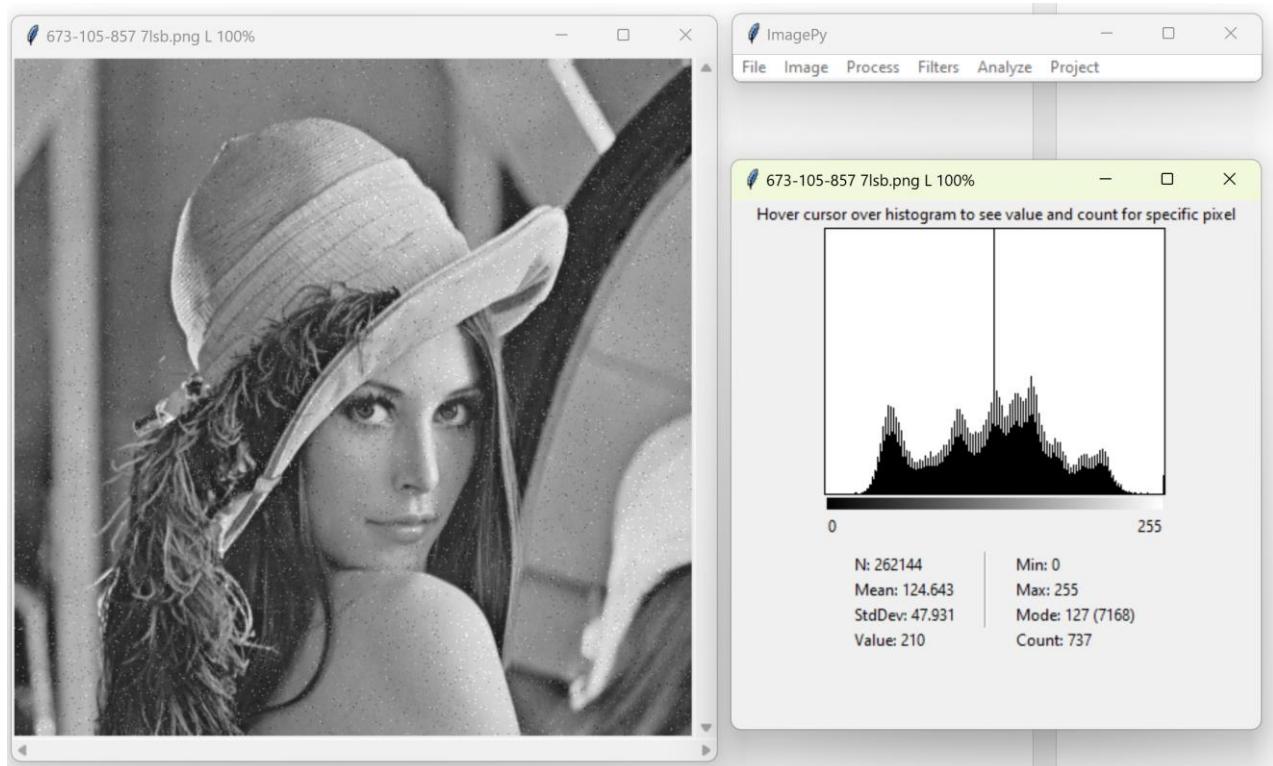
3.1. Histogram dla obrazów kolorowych oraz monochromatycznych

Aby wyświetlić histogram dla obrazu kolorowego lub monochromatycznego należy: **wybierz aktywny obraz -> Analyze -> Histogram**

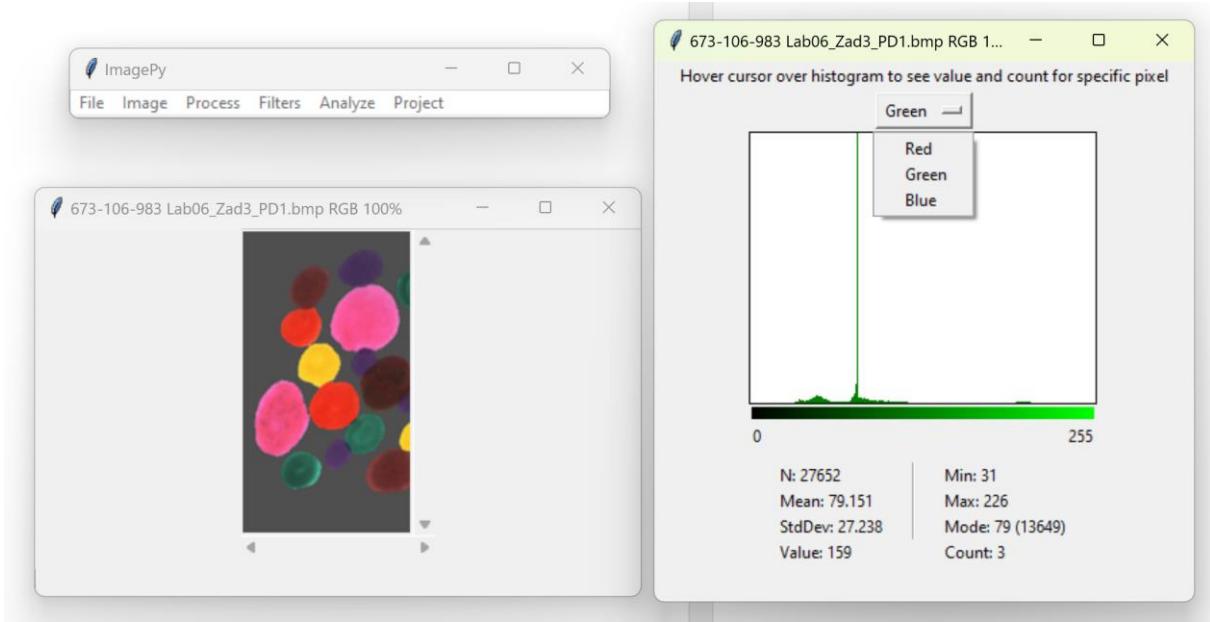
Dla obrazu kolorowego po wyświetleniu histogramu można wskazać dla jakiego kanału (RGB) chcemy wyświetlić histogram.



Przykład histogramu dla obrazu monochromatycznego:



Przykład histogramu dla obrazu kolorowego RGB:

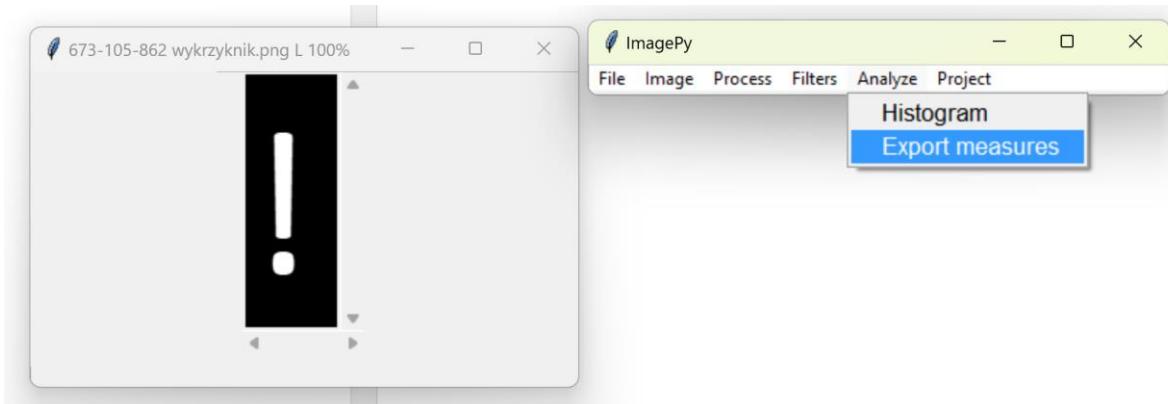


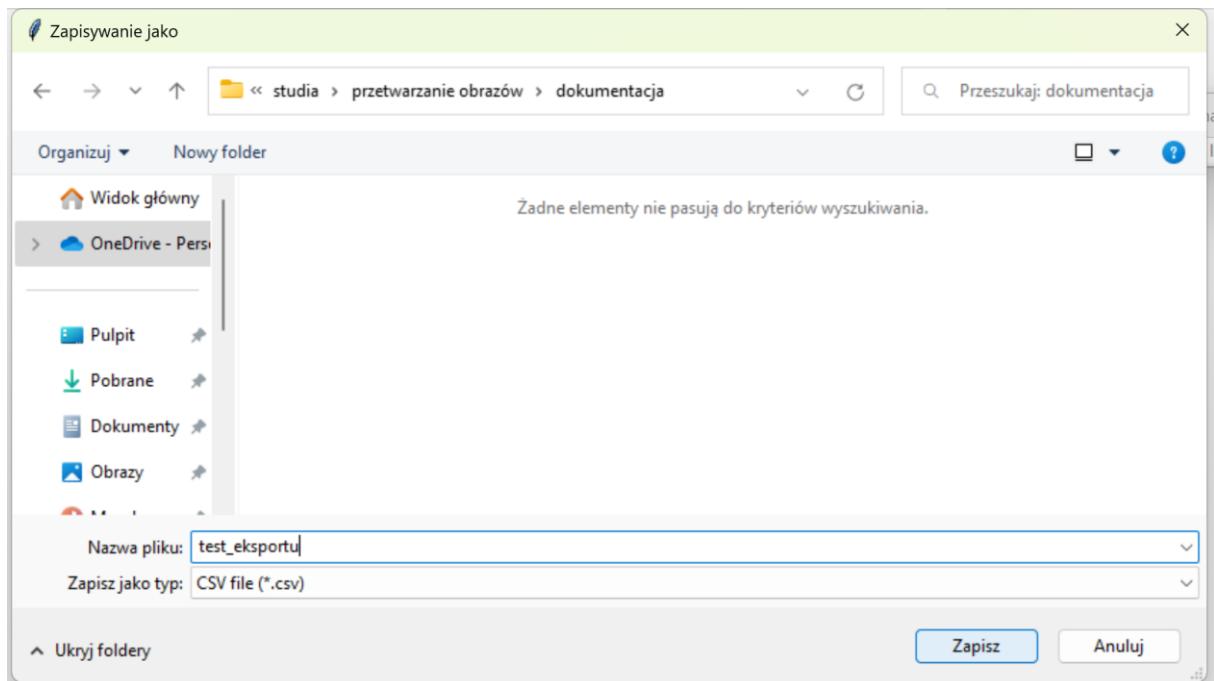
Dla obrazu kolorowego RGB należy wybrać wyświetlany kanał z listy wybory położonej nad histogramem.

3.2. Eksport składowych wektora cech obiektu binarnego do pliku obsługiwaneego przez program Excel

Aby wyeksportować wektor cech do pliku należy: **wybierz aktywny obraz binarny -> Analyze -> Export measures -> wybrać lokalizację docelową oraz nazwę pliku .csv -> Zapisz**

Wektor cech jest zapisywany w formacie .csv, ale jest on przystosowany tylko i wyłącznie do otwarcia w programie Excel ze względu na specyficzne dla tego oprogramowania metadane, które mogą zwiększać dane wyjściowe w innych programach.





Fragment danych wyeksportowanych do pliku: (każdy obiekt znajduje się w osobnym wierszu)

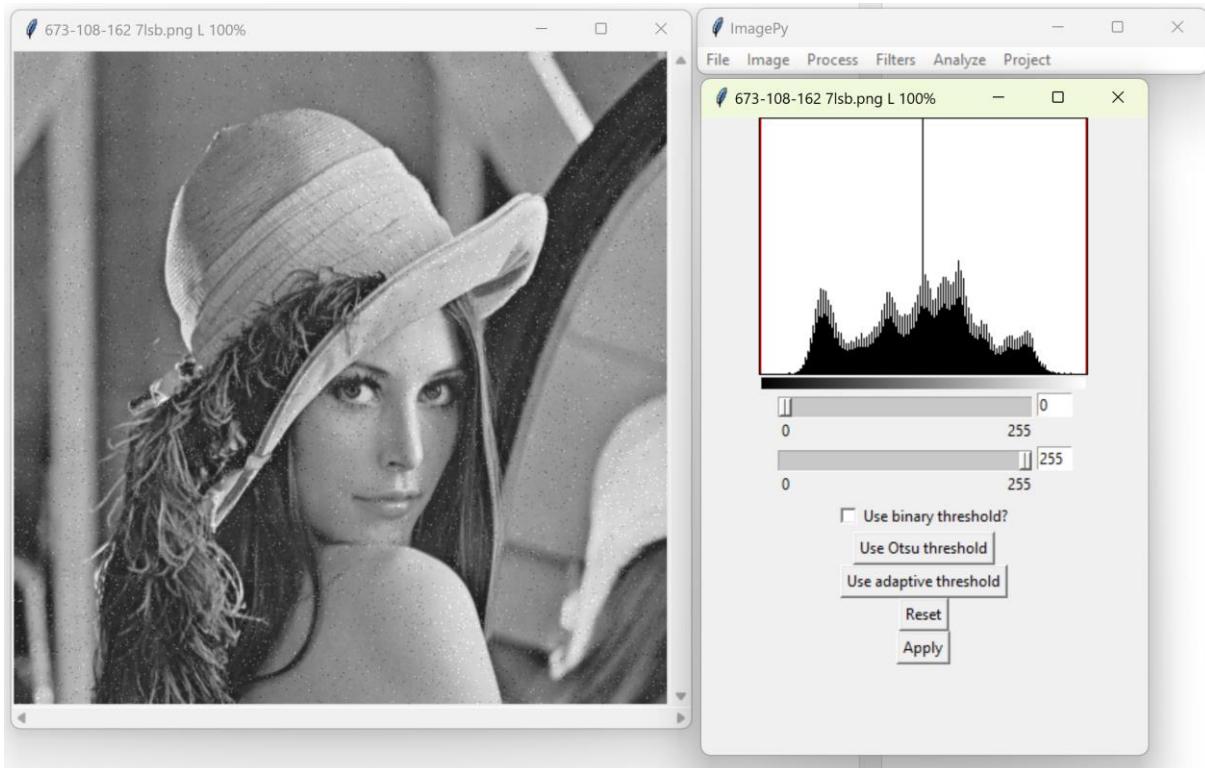
area	perimeter	w1	w2	w3	w9	w10
208.0000	54.1421	16.2737	17.2340	0.0590	0.9443	0.9905
869.0000	166.4853	33.2633	52.9939	0.5932	0.6277	0.9656

4. Podstawowe operacje na obrazach monochromatycznych

4.1. Progowanie obrazu

Aby uruchomić funkcję progowania należy: **wybrać aktywny obraz monochromatyczny -> Image -> Threshold -> po osiągnięciu efektów progowania wybrać Apply, aby zatwierdzić zmiany lub Reset, aby przywrócić obraz do stanu wejściowego.**

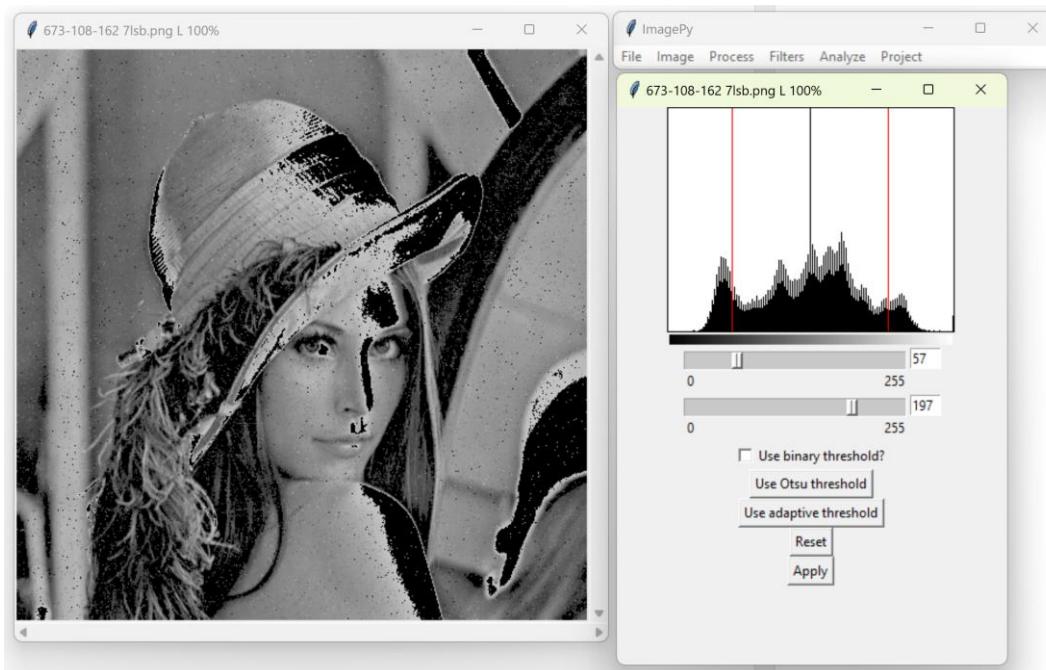
Obraz oryginalny używany do progowania:



Dopóki nie zostanie zamknięte okno z funkcją progowania możliwy jest powrót do obrazu oryginalnego poprzez opcję **Reset**. Aby zatwierdzić wynik progowania należy wybrać opcję **Apply**, która zamyka okno progowania i umożliwia dalszą pracę nad progowanym obrazem.

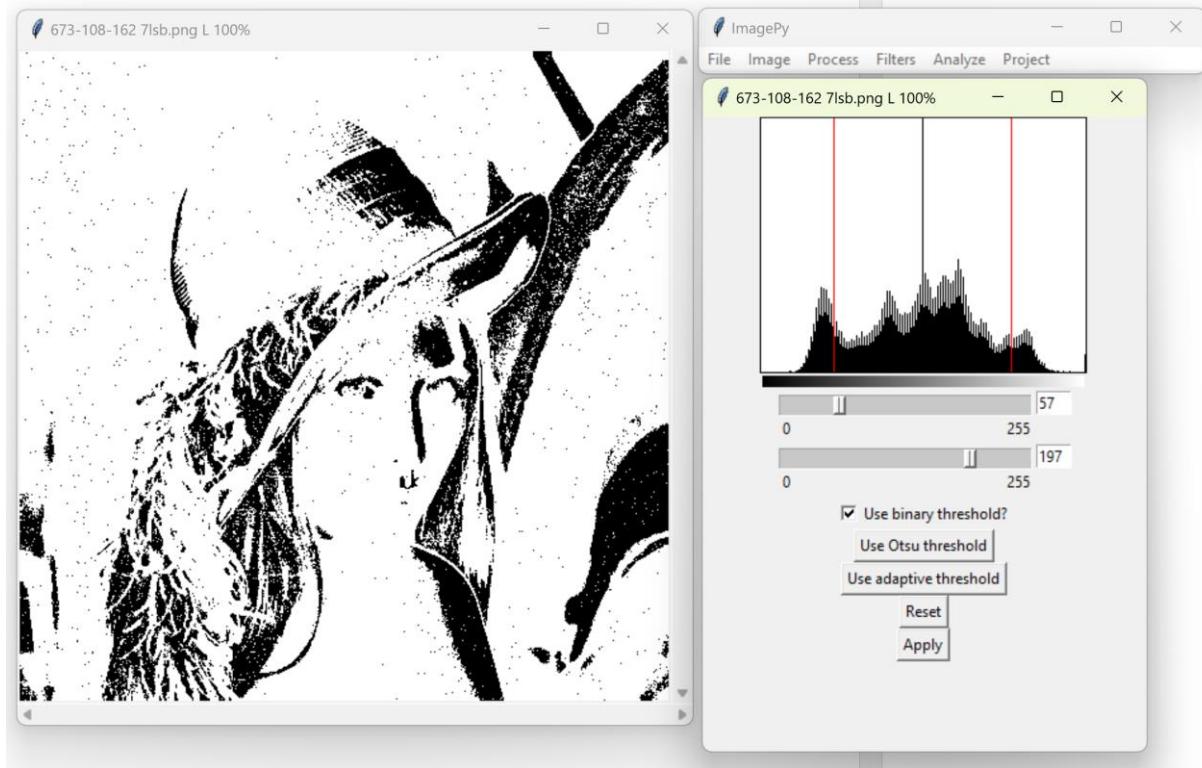
4.1.1. Progowanie z zachowaniem poziomów jasności

Progowanie z zachowaniem poziomów jasności jest możliwe za pomocą dwóch suwaków oznaczających minimalną oraz maksymalną wartość progowaną. Należy upewnić się, że flaga *Use binary threshold?* jest odznaczona.



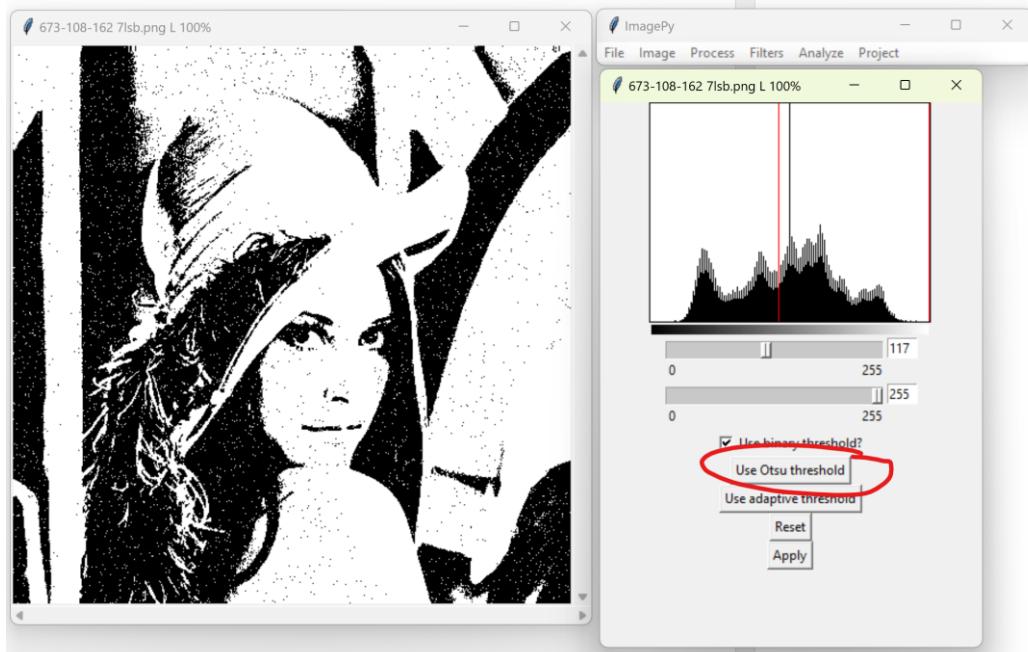
4.1.2. Progowanie binarne

Progowanie binarne jest możliwe za pomocą dwóch suwaków oznaczających minimalną oraz maksymalną wartość progowaną. Należy upewnić się, że flaga *Use binary threshold?* jest zaznaczona.



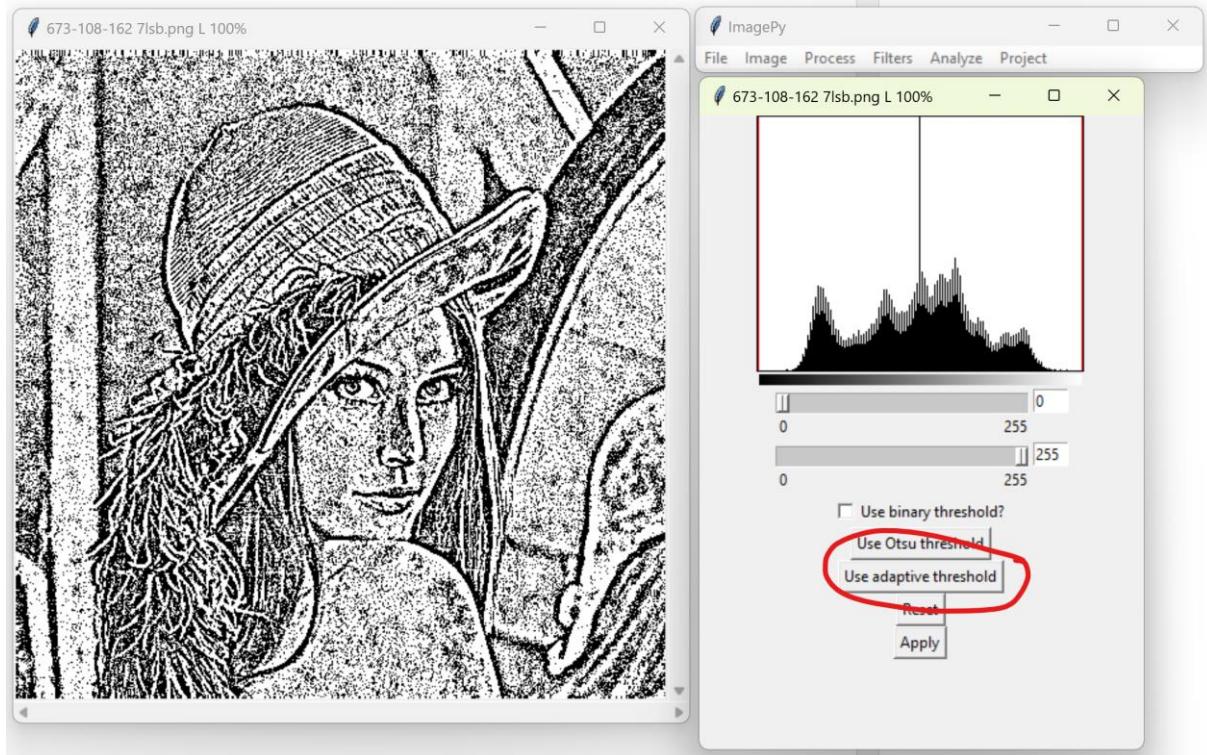
4.1.3. Progowanie metodą Otsu

Aby użyć progowania metodą Otsu należy wybrać opcję ***Use Otsu threshold***. Opcja ta wyznacza próg minimalny metodą Otsu (wynik tej metody jest wskazany przez pierwszy od góry suwak, na przykładzie jest to wartość 117) oraz ustawia flagę *Use binary threshold?*.



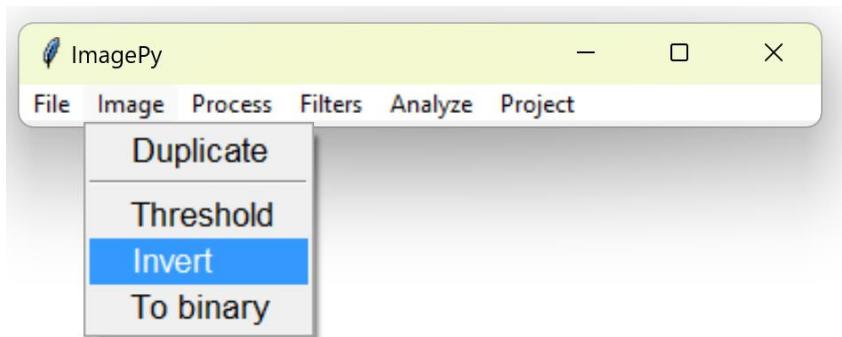
4.1.4. Progowanie adaptacyjne

Aby użyć progowania adaptacyjnego należy wybrać opcję **Use adaptive threshold**. Obecnie nie jest możliwe jednoczesne progowanie adaptacyjne wraz z progowaniem za pomocą suwaków (w tym Otsu).



4.2. Negacja

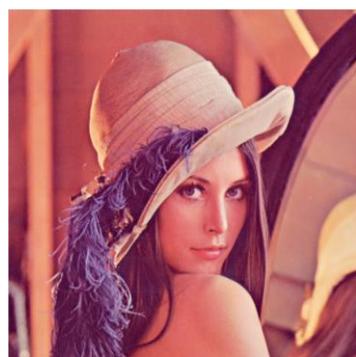
Negacja została zaimplementowana dla obrazów kolorowych oraz monochromatycznych: **wybierz aktywny obraz -> Image -> Invert**



Przykład dla obrazu kolorowego:



Obraz 1 Obraz kolorowy po negacji



Obraz 2 Obraz kolorowy oryginalny

Przykład dla obrazu monochromatycznego:



Obraz 3 Obraz monochromatyczny po negacji



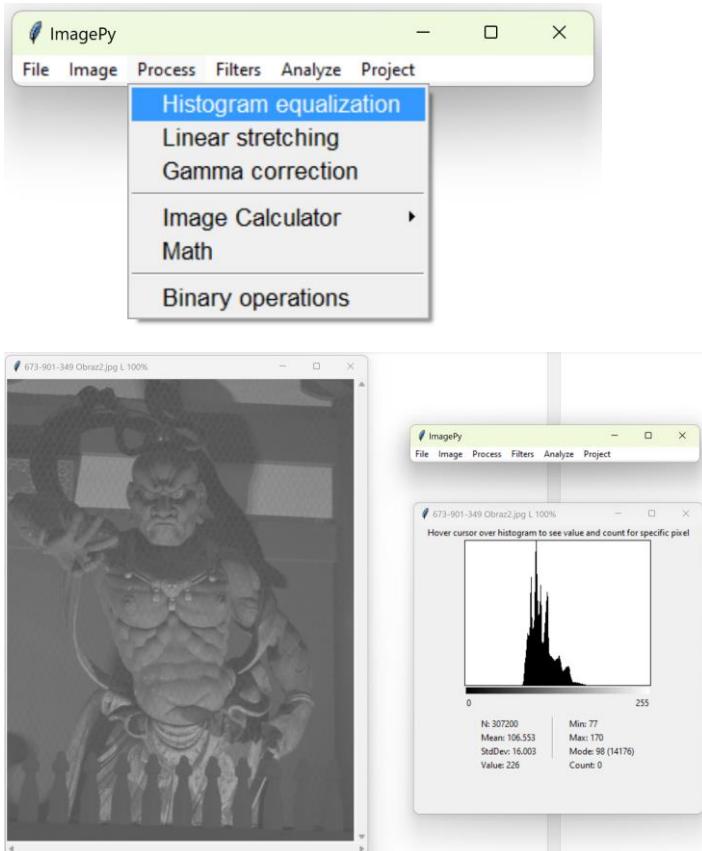
Obraz 4 Obraz monochromatyczny oryginalny

5. Przetwarzanie obrazów monochromatycznych

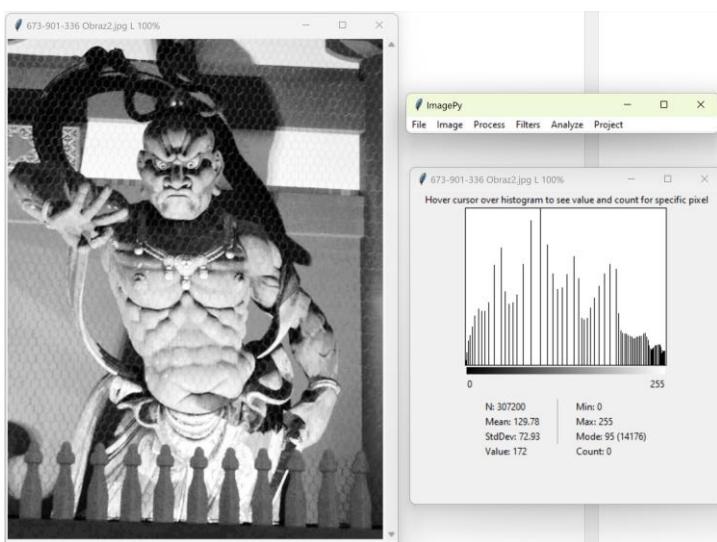
5.1. Wyrównanie histogramu

Wyrównywanie histogramu dąży do ujednolicenia rozkładu pikseli we fragmentach osi poziomów jasności. Wykorzystuje do tego całkę z histogramu do ustalania jasności docelowych.

Krok po kroku: **wybierz aktywny obraz -> Process -> Histogram equalization**



Obraz 5 Obraz oryginalny

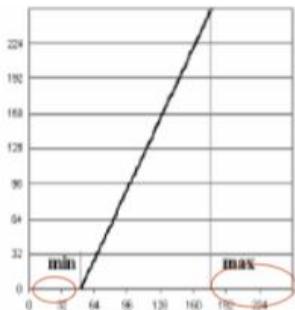


Obraz 6 Obraz po wyrównaniu histogramu

5.2. Liniowe rozciąganie histogramu

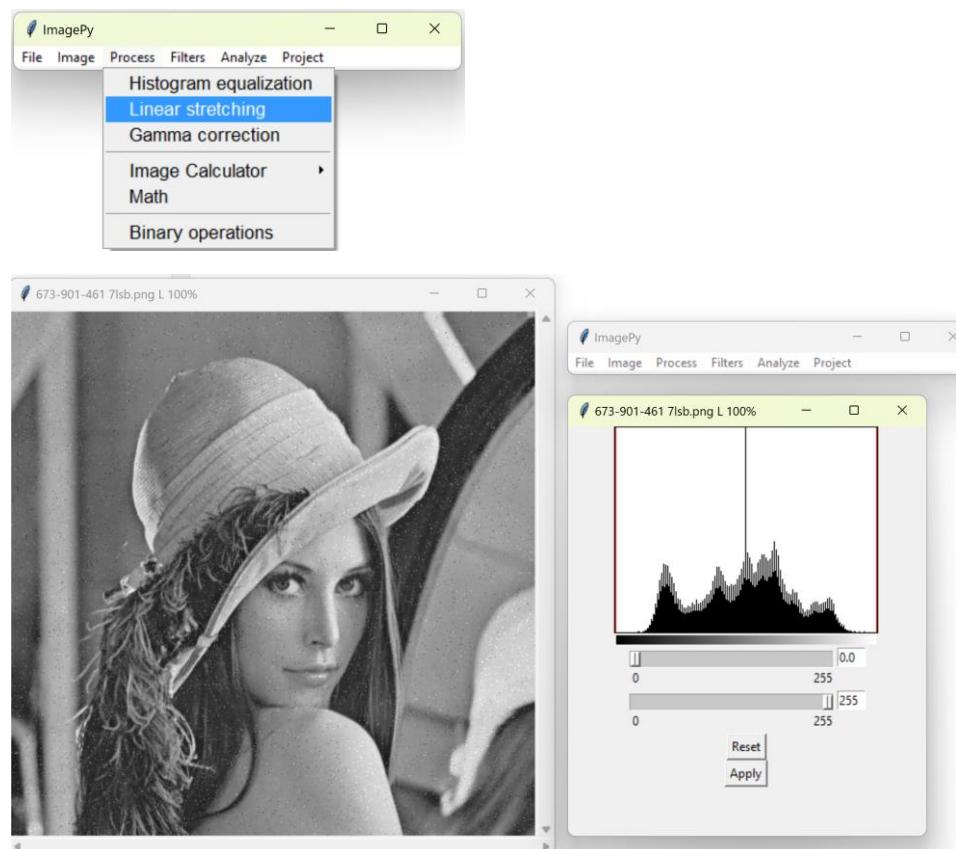
Liniowe rozciąganie histogramu umożliwia zagospodarowanie całego zakresu dostępnych poziomów szarości.

Rozciąganie przeprowadzane jest według wzoru:

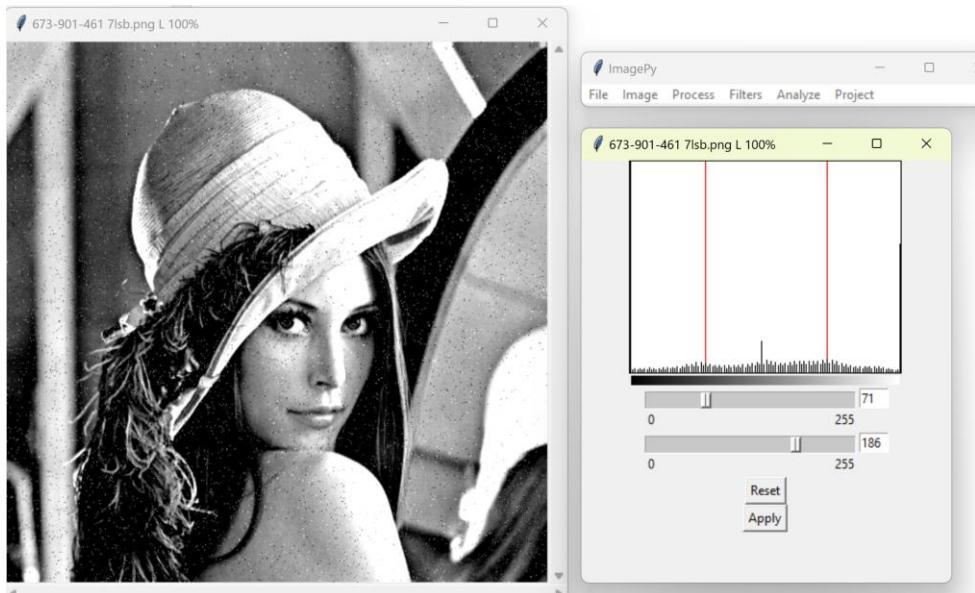


$$q(i,j) = \begin{cases} L_{\min} & \text{dla } p(i,j) < \min \\ \frac{(p(i,j) - \min) * L_{\max}}{\max - \min} & \text{dla } \min \leq p(i,j) \leq \max \\ L_{\max} & \text{dla } p(i,j) > \max \end{cases}$$

Krok po kroku: **wybrać aktywny obraz -> Process -> Linear stretching. Następnie wybrać progi rozciągania suwakami i wygenerować zmodyfikowany obraz poprzez przycisk Apply.**



Obraz 7 Obraz oryginalny



Obraz 8 Obraz po rozciągnięciu histogramu

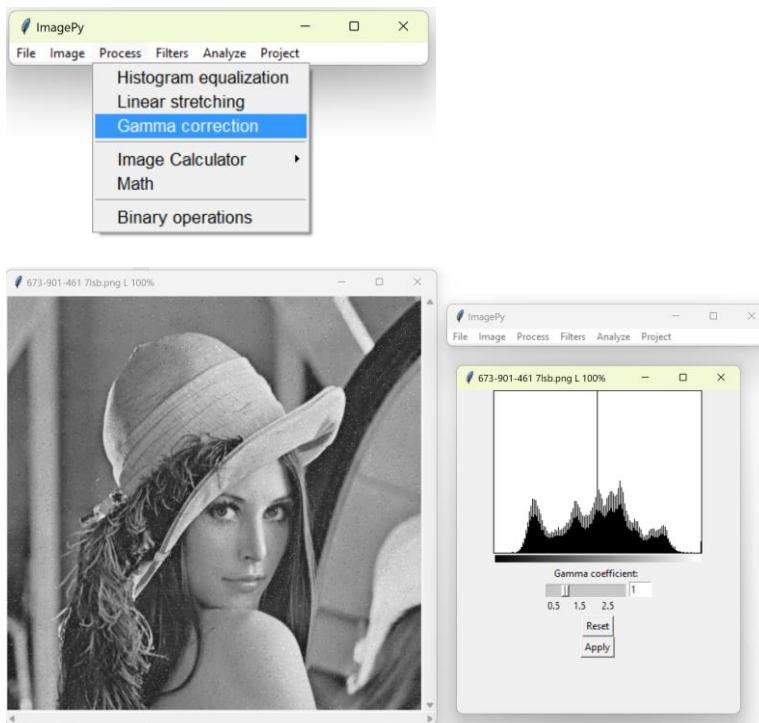
5.3. Nieliniowe rozciąganie współczynnikiem gamma

W przypadku obrazów, gdzie nieefektywnie wykorzystywana jest pełna dynamika odcieni i barw dostępna w danym zakresie, może zostać użyte nieliniowe rozciąganie według funkcji gamma.

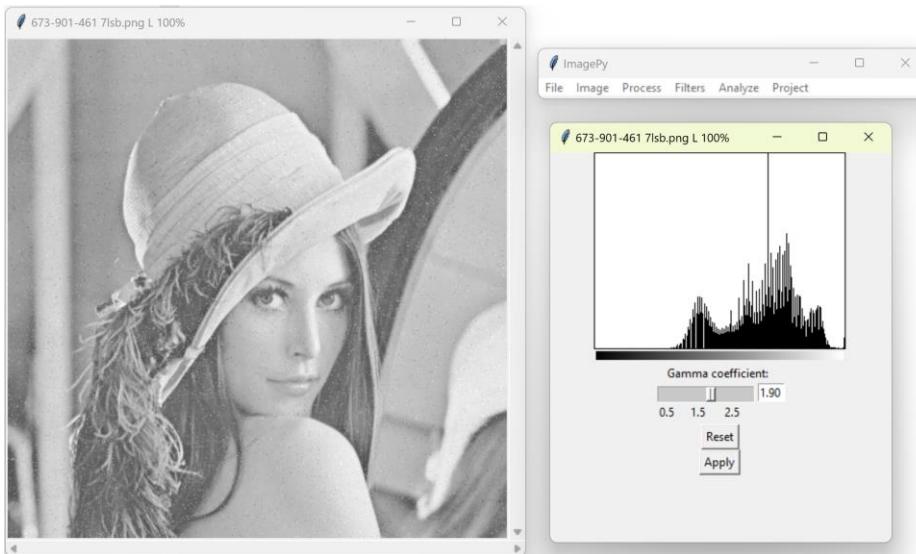
Realizowana jest poprzez wzór:

$$LUT(i) = i^{1/\gamma}$$

Krok po kroku: **wybrać aktywny obraz -> Process -> Gamma stretching -> wybrać wartość współczynnika gamma poprzez suwak -> Apply**



Obraz 9 Obraz oryginalny



Obraz 10 Efekt nieliniowego rozciągania histogramu

6. Operacje matematyczne na obrazach

Krok po kroku: **Process -> Image Calculator -> Add/Subtract ->** wybieramy, które obrazy chcemy użyć.

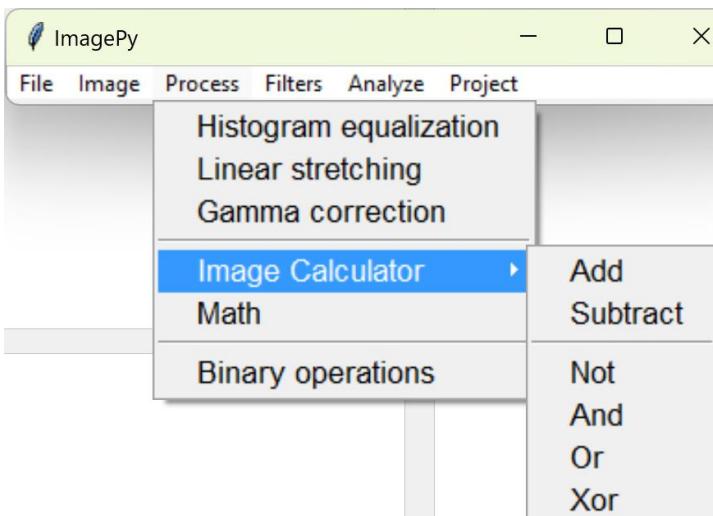
Dodawanie obrazów jest zaimplementowane następująco:

- 1) Przeskalowanie obu obrazów z 256 poziomów jasności na 128
- 2) Dodanie poszczególnych pikseli

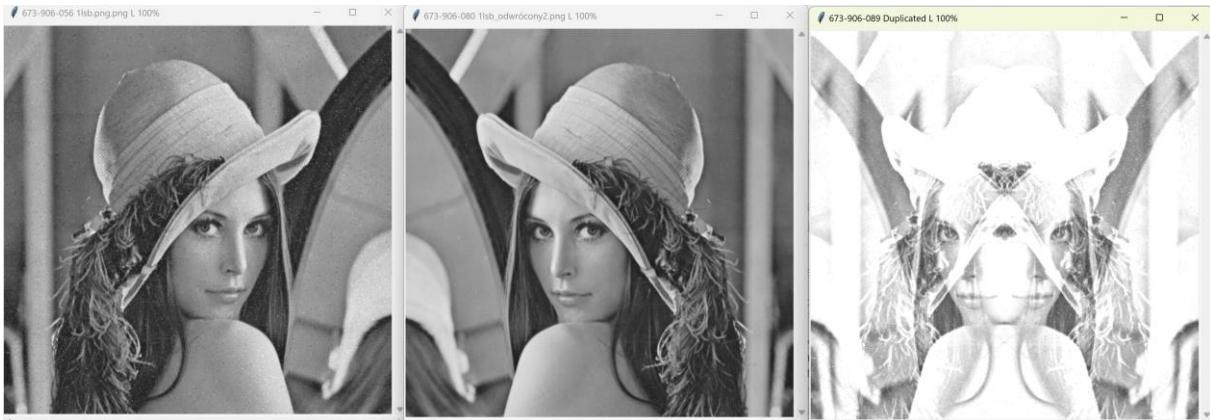
Dzięki temu liczba poziomów jasności obrazu wynikowego nie zmienia się i wynosi 256.

Odejmowanie zostało zaimplementowane poprzez obliczanie wartości bezwzględnej z wyniku odejmowania poszczególnych pikseli.

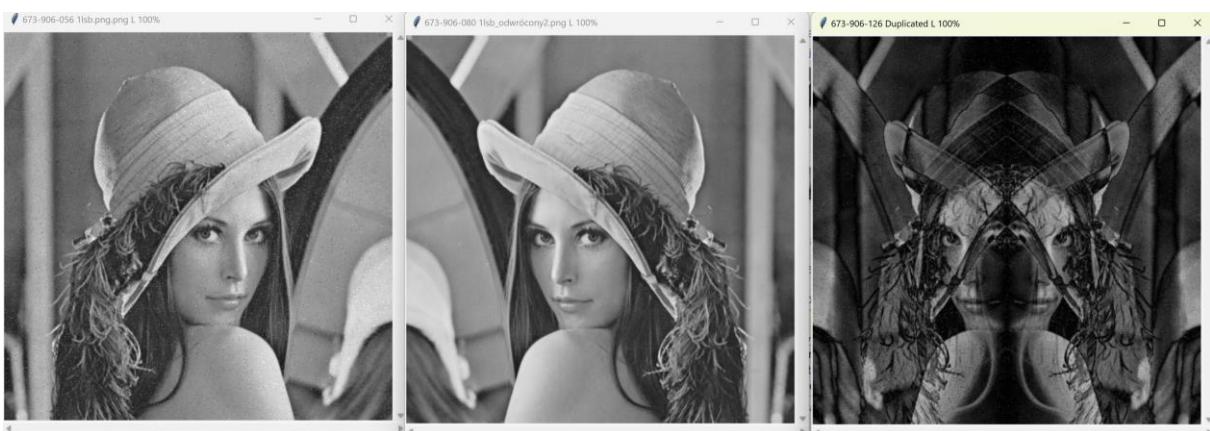
Oba obrazy wejściowe muszą być monochromatyczne i posiadać dokładnie takie same wymiary.



6.1. Dodawanie i odejmowanie obrazów



Obraz 11 Prawy obraz jest wynikiem dodawania dwóch pozostałych obrazów

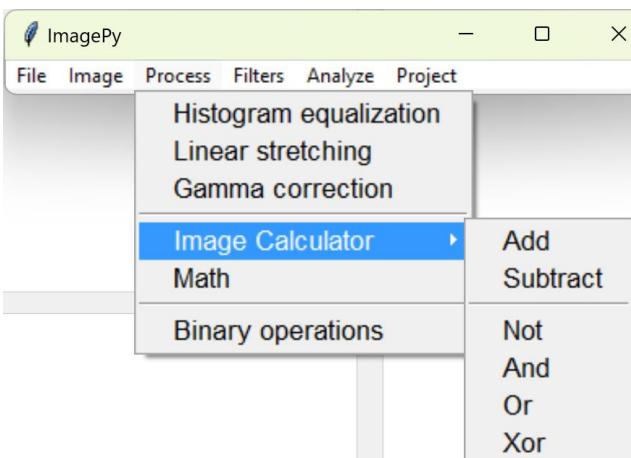


Obraz 12 Prawy obraz jest wynikiem odejmowania dwóch pozostałych obrazów

7. Operacje logiczne na obrazach

Operacje logiczne na obrazach wykonywane są na poszczególnych pikselach, a konkretnie na odpowiadających bitach wartości jasności tych pikseli. Można je wykonać na obrazach monochromatycznych oraz o takich samych wymiarach.

Krok po kroku: **Process -> Image Calculator -> Not/And/Or/Xor -> wybrać dwa obrazy wejściowe -> Apply**



7.1. And

Przykład: 1111 and 1010 -> 1010



Obraz 13 Środkowy obraz jest wynikiem operacji and

7.2. Or

Przykład: 1111 or 1010 -> 1111



Obraz 14 Środkowy obraz jest wynikiem operacji or

7.3. Xor

Przykład: 1111 xor 1010 -> 0101

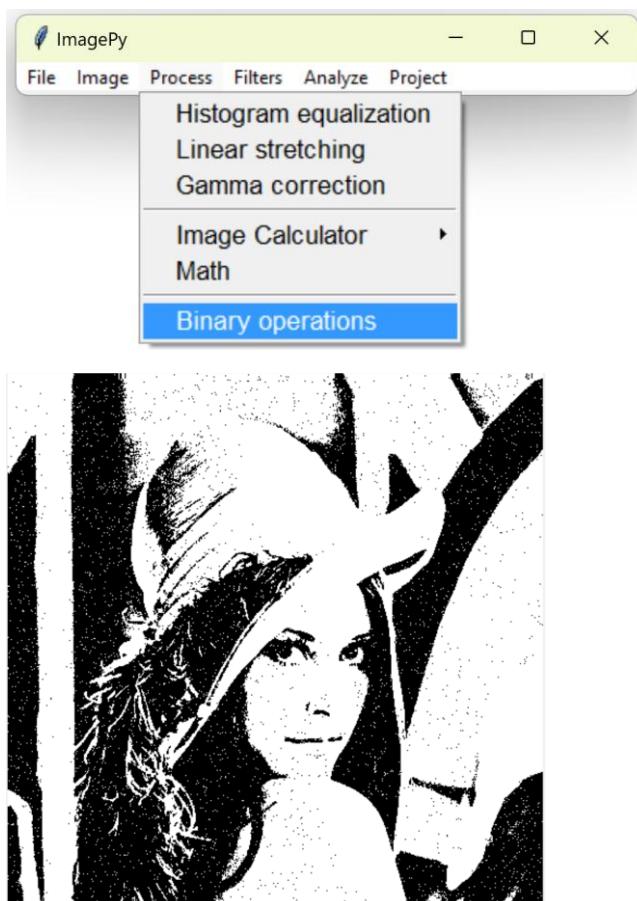


Obraz 15 Środkowy obraz jest wynikiem operacji xor

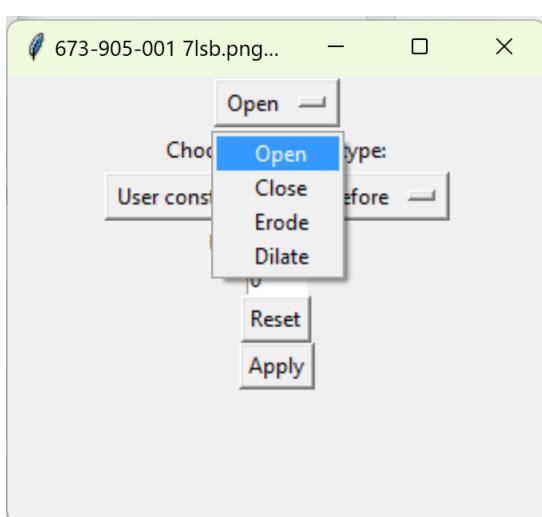
8. Operacje morfologii matematycznej na obrazach binarnych

Operacje morfologiczne pozwalają na budowanie złożonych operacji, pozwalających na analizę kształtu i wzajemnego położenia obiektów.

Krok po kroku: **wybierz aktywny obraz -> Process -> Binary operations -> wybrać rodzaj operacji -> Apply**



Obraz 16 Obraz oryginalny



Obraz 17 Menu operacji morfologicznych

8.1. Erozja



Obraz 18 Wynik erozji

8.2. Dylacja



Obraz 19 Wynik dylacjii

8.3. Otwarcie

Otwarcie polega na wykonaniu najpierw erozji, a następnie dylacji.



Obraz 20 Wynik otwarcia

8.4. Zamknięcie

Otwarcie polega na wykonaniu najpierw dylacji, a następnie erozji.



Obraz 21 Wynik zamknięcia

9. Wykrywanie krawędzi w obrazie monochromatycznym

9.1. Wykrywanie krawędzi operatorami:



Obraz 22 Obraz oryginalny

9.1.1. kierunkowymi Sobela

Filtr kierunkowe Sobela

Wschód	Południowy wschód	Południe	Południowy zachód
-1 0 1	-2 -1 0	-1 -2 -1	0 -2 -1
-2 0 2	-1 0 1	0 0 0	1 0 -1
-1 0 1	0 1 2	1 2 1	1 2 0

E

SE

S

SW

W	NW	N	NE
1 0 -1	2 1 0	1 2 1	0 1 2
2 0 -2	1 0 -1	0 0 0	-1 0 1
1 0 -1	0 -1 -2	-1 -2 -1	-2 -1 0

Zachód

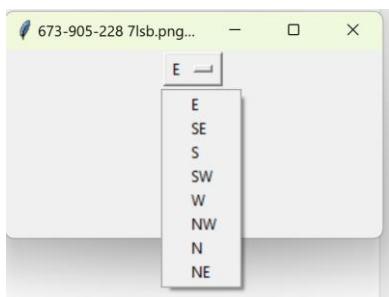
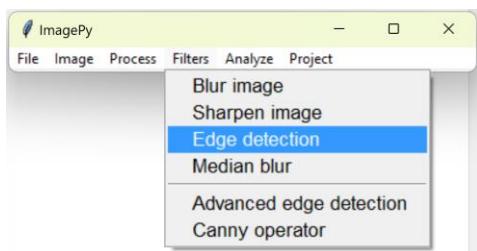
Północny zachód

Północ (N)

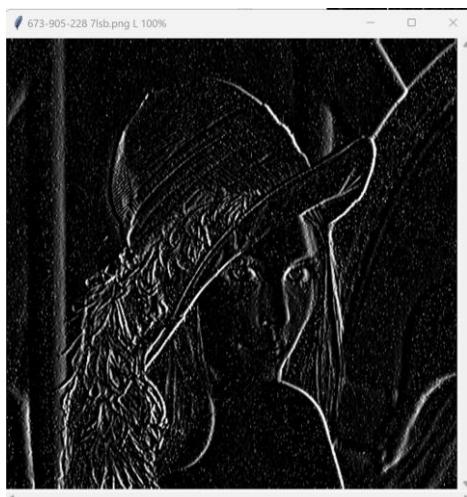
Północny wschód

Obraz 23 Zaimplementowane filtry kierunkowe

Krok po kroku: wybrać obraz -> Filters -> Edge detection -> wybrać filtr kierunkowy -> Apply



Obraz 24 Efekt filtra E

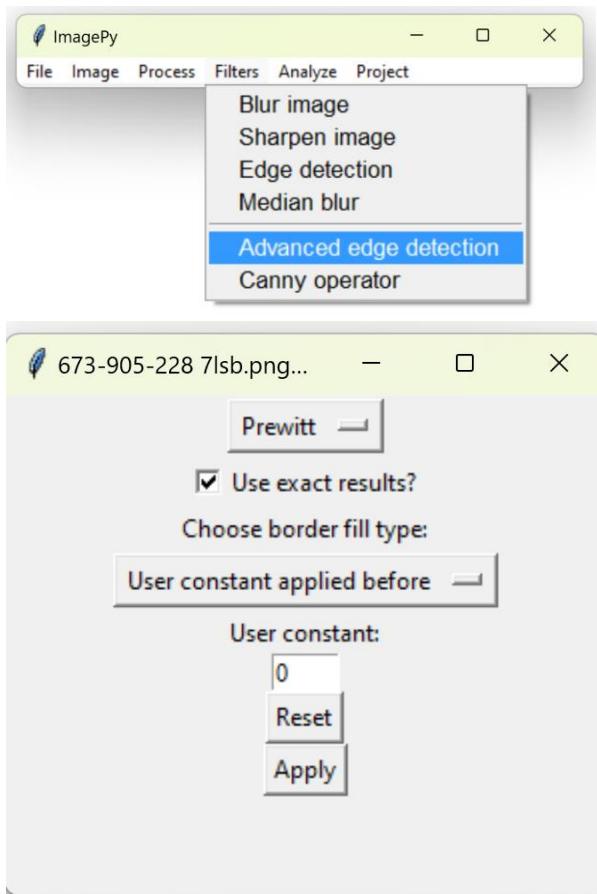


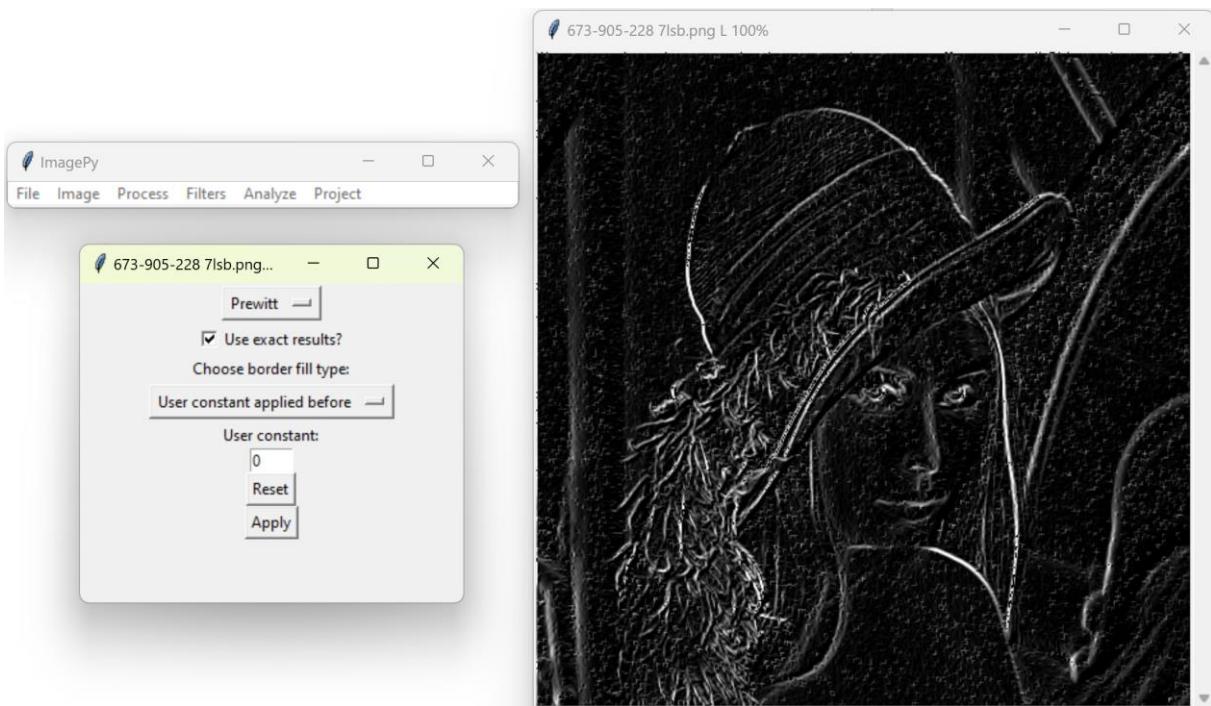
Obraz 25 Efekt filtra W

9.1.2. Sobela i Prewitta

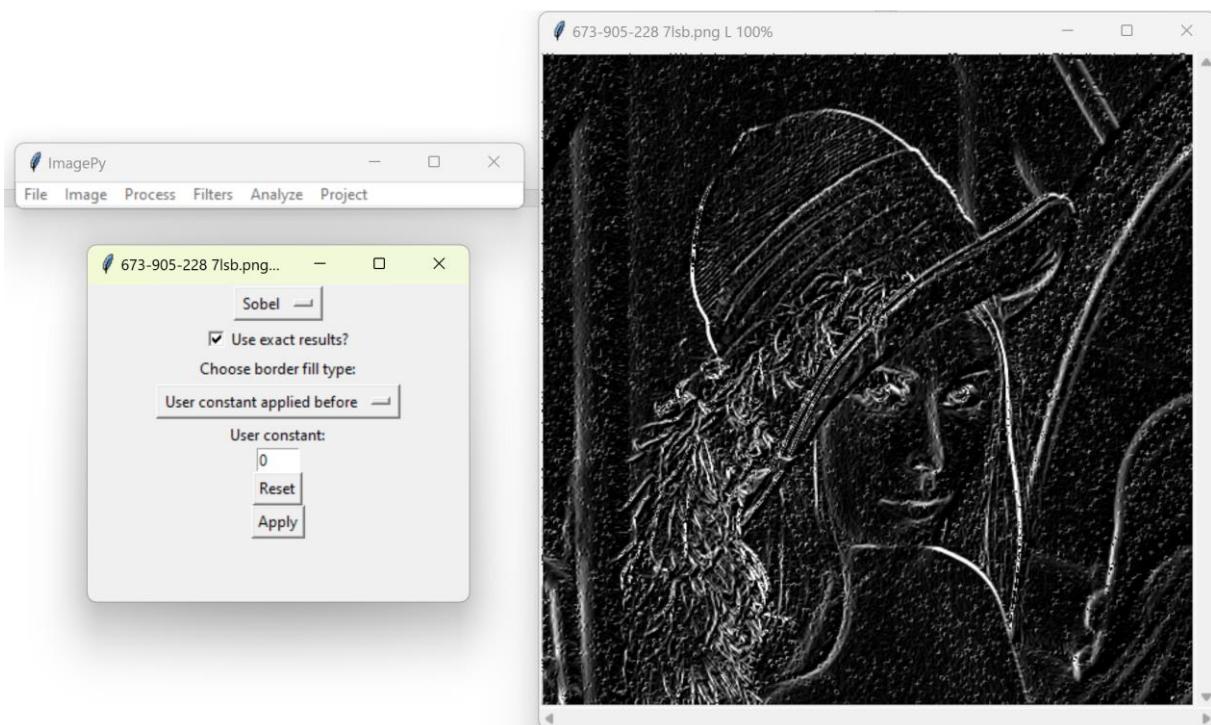
Krok po kroku: **wybierz obraz -> Advanced Edge detection -> wybierz rodzaj operatora Sobel/Prewitt -> Apply**

Flaga *Use exact results?* wyznacza, czy ma być użyty wzór przybliżony (ale szybszy), czy wzór dokładny.





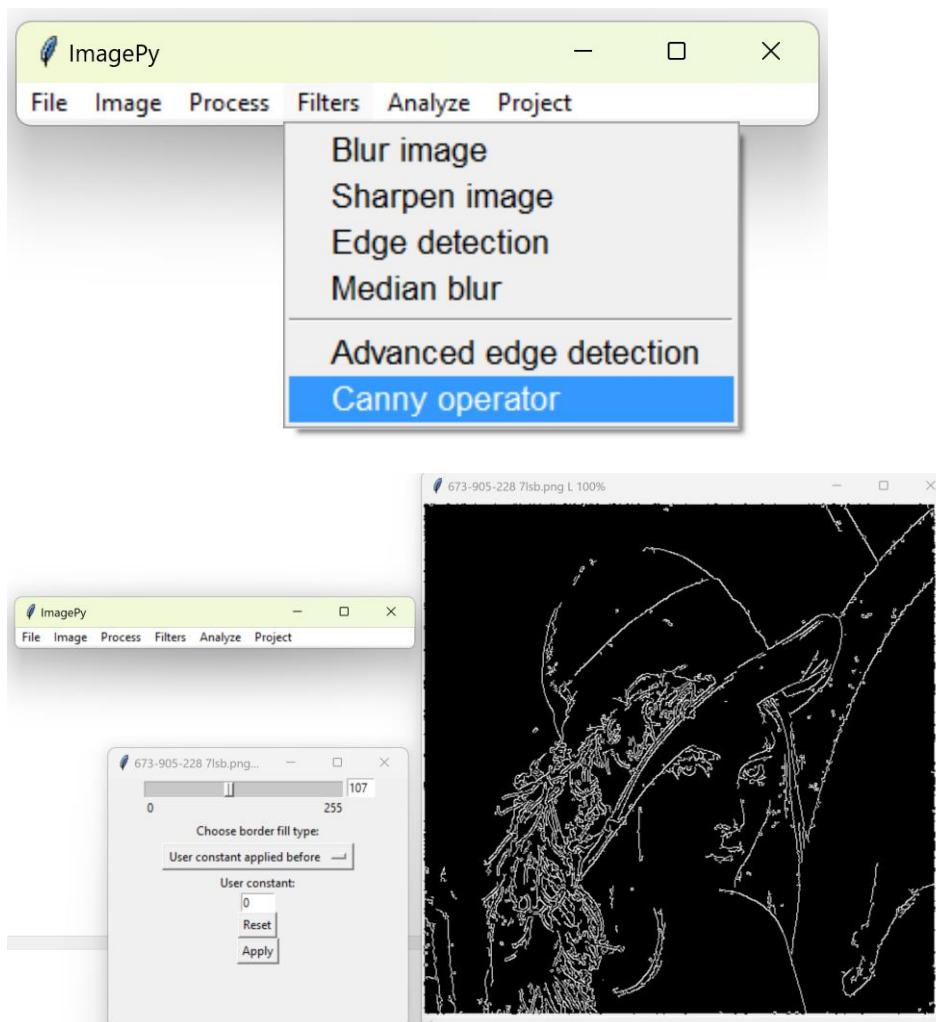
Obraz 26 Efekt użycia operatora Prewitta



Obraz 27 Efekt użycia operatora Sobela

9.1.3. Canny'ego

Krok po kroku: **wybierz aktywny obraz -> Filters -> Canny operator -> dostosować współczynnik progowania -> Apply**



Obraz 28 Efekt użycia operatora Canny'ego

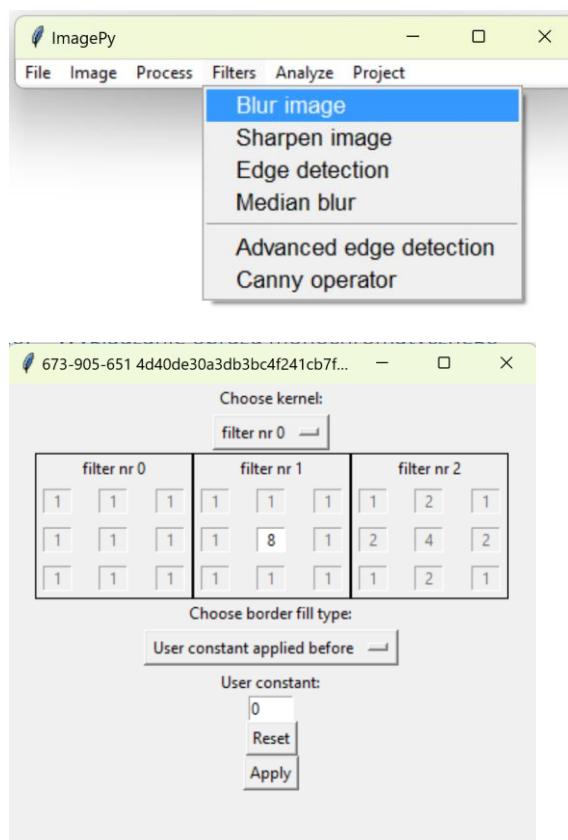
10. Wygładzanie obrazu monochromatycznego

10.1. Wygładzanie liniowe oparte na masce:

Krok po kroku: **wybierz obraz -> Filters -> Blur image -> wybierz filtr 1 z 3 -> Apply**

Dostępne filtry (od lewej do prawej):

- 0) Filtr uśredniający
- 1) Filtr k-pudełkowy, gdzie wartość k jest wskazywana przez użytkownika
- 2) Filtr gaussowski

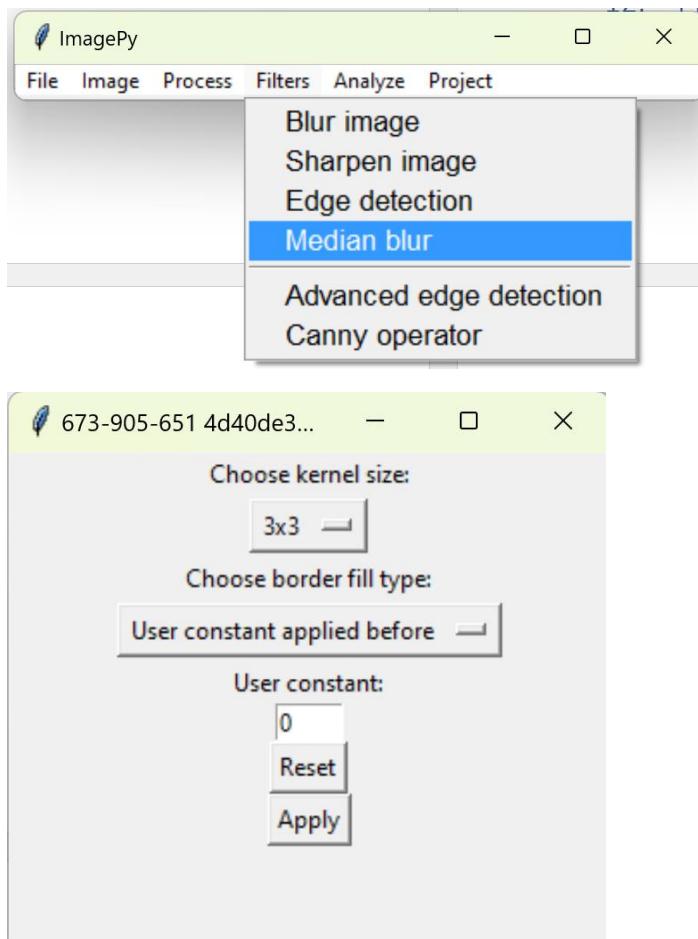


Obraz 29 Wynik wygładzania jednym z filtrów

10.2. Filtr medianowy

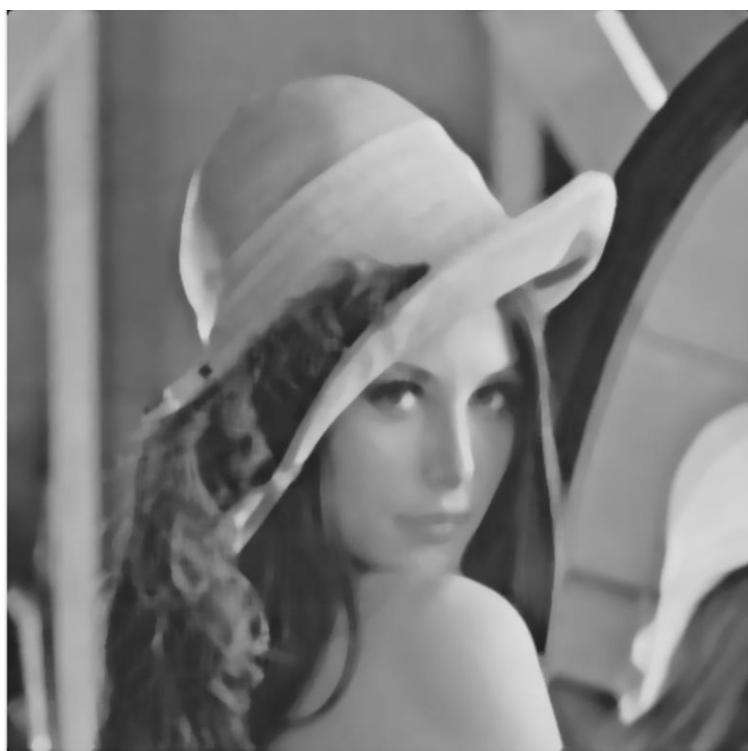
Krok po kroku: **wybierz obraz -> Filters -> Median blur -> wybierz rozmiar filtra (dostępne filtry: 3x3, 6x6 i 9x9) -> Apply**

Filtr medianowy jest filtrem konwolucyjnym, który wyznacza wartość danego piksela na podstawie mediany otoczenia wskazanego rozmiarem jądra.





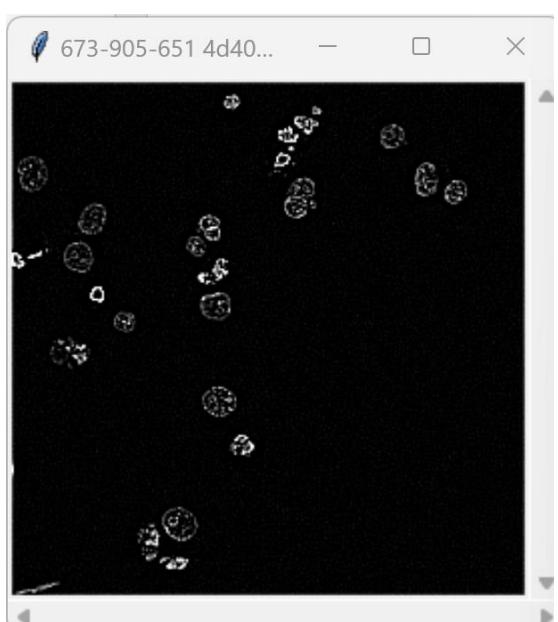
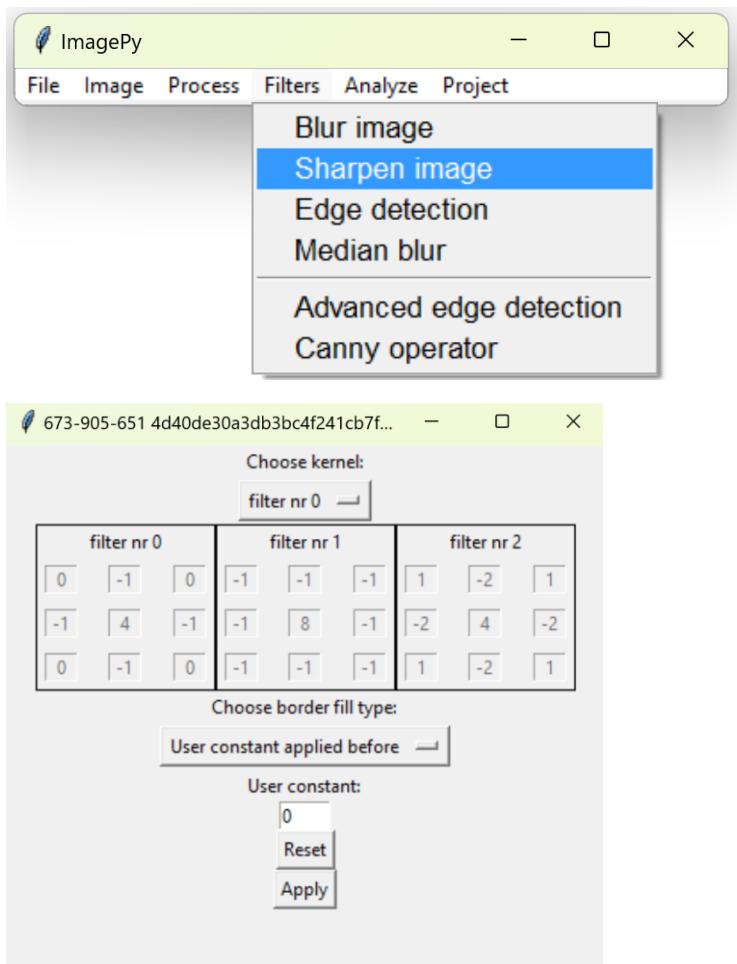
Obraz 30 Wynik filtrowania filtrem 3x3



Obraz 31 Wynik filtrowania filtrem 9x9

11. Wyostrzanie obrazu monochromatycznego

Krok po kroku: wybrać obraz -> Filters -> Sharpen image -> wybrać rodzaj filtru (1 z 3) -> Apply



Obraz 32 Wynik wyostrzania jednym z filtrów

12. Projekt

12.1. Wprowadzenie teoretyczne

12.1.1. Linia profilu

Pobieranie i wyświetlanie linii profilu w ramach projektu została zaimplementowana dla obrazów monochromatycznych i kolorowych. Linia profilu jest to wykres liniowy przedstawiający jasności pikseli zaznaczonych przez użytkownika. Dla obrazów monochromatycznych jest to wykres zawierający tylko jedną krzywą, a dla obrazów kolorowych są to trzy krzywe dla każdego z kanałów RGB. Dzięki linii profilu można zaobserwować dwie inherentne cechy akwizycji obrazu: szum oraz zakłóceń/zniekształceń. Linia profilu może służyć również do sprawdzania jasności pikseli tam, gdzie ciężko gołym okiem zauważać zmiany jasności np. przy niewielkim gradiencie.

W języku Python używając biblioteki standardowej *tkinter* nie jest możliwe pobranie wartości pikseli zaznaczonych odcinkiem, dlatego wymagana była implementacja narzędzia wyliczającego, jakie piksele wchodzą w skład danego odcinka obrazu. Do osiągnięcia tego celu został wybrany algorytm Bresenhama¹, który dla danej odcinka o zdefiniowanym punkcie początkowym i końcowym zwraca listę pikseli wchodzącej jego skład. Algorytm ten wyróżnia się bardzo małą złożonością pamięciową (wymagane jest tylko 5 zmiennych całkowitoliczbowych) oraz szybkością działania, ze względu na wykorzystywaniu tylko 1-2 operacji dodawania i porównywania dla każdego piksela.

12.1.2. Filtracja logiczna

Filtracja logiczna w ramach projektu jest wykonywana tylko i wyłącznie na obrazach binarnych. Polega ona na ustalaniu wartości poszczególnych pikseli na podstawie pikseli bezpośrednio sąsiadującymi z nimi (czyli w 4 głównych kierunkach, bez ukosów). Dzięki tego rodzaju filtracji możliwe jest przefiltrowanie obrazu z niepożądanych elementów takich jak: linie poziome o grubości 1 px, linie pionowe o grubości 1px i pojedynczych, odizolowanych pikseli. Może być używana np. do usuwania szumu z obrazu.

W ramach projektu zostały zaimplementowane trzy filtry logiczne:

	a	
b	X	c
d		

Tabela 1 Oznaczenia pikseli sąsiadujących z obliczanym pikselem X

X' – nowa wartość piksela X

1. Filtracja linii poziomych i odizolowanych pikseli

$$X' = \begin{cases} a & \text{if } a = d \\ \text{else } X \end{cases}$$

2. Filtracja linii pionowych i odizolowanych pikseli

$$X' = \begin{cases} b & \text{if } b = c \\ \text{else } X \end{cases}$$

3. Filtracja odizolowanych pikseli

$$X' = \begin{cases} a & \text{if } a = b = c = d \\ \text{else } X \end{cases}$$

¹ J.E. Bresenham. Algorithm for computer control of a digital plotter. „IBM Systems Journal”. 4 (1), 1965.

12.2. Opis implementacji

Całość projektu została napisana w języku Python 3.10. Większość funkcjonalności została zaimplementowana z wykorzystaniem biblioteki standardowej wbudowanej w język z naciskiem na wykorzystanie nowych funkcji wersji 3.10. Aplikacja została napisana z założeniem jak najmniejszego korzystania z zależności zewnętrznych na korzyść własnych implementacji z wykorzystaniem biblioteki standardowej.

Wymagane biblioteki zewnętrzne wraz z uzasadnieniem:

- 1) Pillow \approx 9.2.0 - powszechnie używana biblioteka do obsługi obrazów w języku Python, wymagana głównie ze względu na zgodność z wbudowaną biblioteką *tkinter* do wyświetlania obrazu w aplikacji
- 2) Numpy \approx 1.23.3 - używana do operacji na macierzach. Z założenia implementacja aplikacji nie używa operacji na macierzach zawartych w tej bibliotece, jedynie wykorzystuje własne implementacje w języku Python. Biblioteka *numpy* jest jedną z wymaganych zależności biblioteki *Pillow*, do przedstawiania wartości jasności pikseli w obrazie w formie macierzy liczb naturalnych (macierzy 2D dla obrazów monochromatycznych i 3D dla obrazów kolorowych RGB)
- 3) Opencv-python \approx 4.6.0.66 - wymagana przez założenia przedmiotu APO. Wykorzystywane są konkretne funkcjonalności wskazane przez zadanie laboratoryjne. W projekcie używana do wypełnienia ramki obrazu przy filtracji logicznej.

Użyte biblioteki standardowe:

- 1) Logging - używana do obsługi logów zdarzeń. Logi są przekierowywane na konsolę i domyślnie mają ustawiony poziom DEBUG
- 2) Math - używana głównie do pobierania wartości stałych matematycznych i obliczania logarytmu
- 3) Tkinter - używana do tworzenia GUI. Główna biblioteka używana w całości oprogramowania.
- 4) Dataclasses - moduł zawierający implementacje *@dataclass*. Wykorzystywana głównie do przechowywania stałych.
- 5) Os - moduł do obsługi ścieżek systemowych, używane do zapisu i odczytu z dysku
- 6) Time - moduł wykorzystywany do pobierania aktualnego czasu systemowego. Na podstawie czasu systemowego każde okno z obrazem zawiera unikalny numer identyfikacyjny w formacie xxx-xxx-xxx.

Kod implementujący projekt znajduje się w katalogu *src* w:

- 1) App.py – skrypt uruchamiający aplikację i ustawiający poziom logowania
- 2) Katalog *utils* – główna część aplikacji wspólna z implementacją laboratoryjną. Zawiera tworzenie okien aplikacji, obsługę menu, obsługę okien, klasy generycznych widgetów i stałe
- 3) Katalog *lab_project*
 - a) Logic_filter.py – implementacja filtracji logicznej na obrazach binarnych
 - b) Plot_profile.py – implementacja tworzenia wykresu linii profilu

12.3. Opis funkcjonalności

12.3.1. Linia profilu

W ramach projektu zostały zaimplementowane dwie metody zaznaczania pikseli obrazu do generowania linii profilu:

1. Rysowanie krzywej myszką

Rysowanie myszką polega na zaznaczenie pikseli według jednej, pojedynczej, zdefiniowanej przez ruch myszki krzywej. Piksele znajdujące się na linii ruchu myszki zostaną uwzględnione do generowania linii profilu.

2. Rysowanie odcinków, łamanej

Rysowanie odcinków polega na zaznaczenie pikseli według odcinka lub łamanej stworzonej z wielu odcinków.

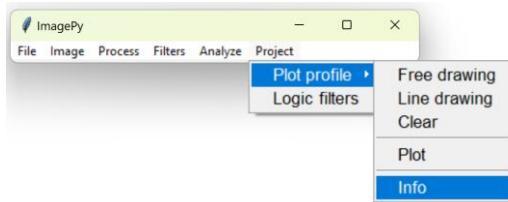
W wbudowanym w język Python module *tkinter* jest możliwe rysowanie odcinków podając jedynie punkt początkowy i końcowy odcinka, ale nie jest możliwe pobranie jakie punkty wchodzą w skład tego odcinka. Dlatego został wykorzystany algorytm Bresenhama do obliczenia tych punktów.

Algorytm ten jest używany w obu rodzajach rysowania, ponieważ moduł *tkinter* pobiera pozycję myszki z dość dużym interwałem, co skutkuje tym, że zaznaczane piksele nie tworzą ciągłej krzywej. Aby pobierać wszystkie piksele wyznaczane ruchem myszki zostało zaimplementowane rysowanie odcinków pomiędzy kolejnymi współrzędnymi myszy, a punkty leżące na tym odcinku wyznaczane na podstawie algorytmu Bresenhama.

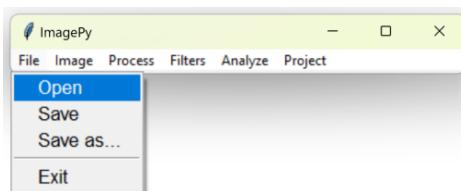
Rysowanie linii profilu zostało zaimplementowane dla obrazów monochromatycznych oraz kolorowych RGB. Poszczególne krzywe na wykresie oznaczone są kolorem, który kanał reprezentuje (czerwony, zielony, niebieski lub czarny). Piksele oznaczone są dystansem od piksela początkowego. Osie wykresów zostały oznaczone przedziałami, co 10 i 50 wartość. Aby odczytać konkretną wartość jasności danego piksela należy umieścić kurSOR myszy na odpowiedniej wartości osi poziomej.

12.3.1.1. Krok po kroku

0. Instrukcja do narzędzia znajduje się w aplikacji: Project -> Plot profile -> Info

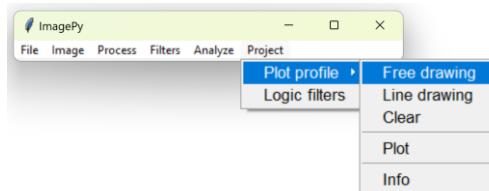


1. Otworzyć obraz monochromatyczny lub kolorowy: File -> Open -> wybieramy obraz -> Otwórz

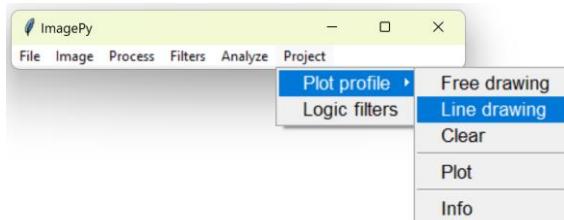


2. Uruchomić narzędzie rysowania: Project -> Plot profile -> ...

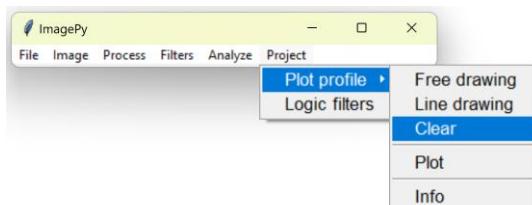
- a. Free drawing – opcja ta pozwala na rysowanie za pomocą myszki dowolnego kształtu krzywej. Aby zacząć rysowanie należy przytrzymać LPM i puścić aby zakończyć rysowanie. Rysowanie kolejnej krzywej czyści poprzednio narysowaną.



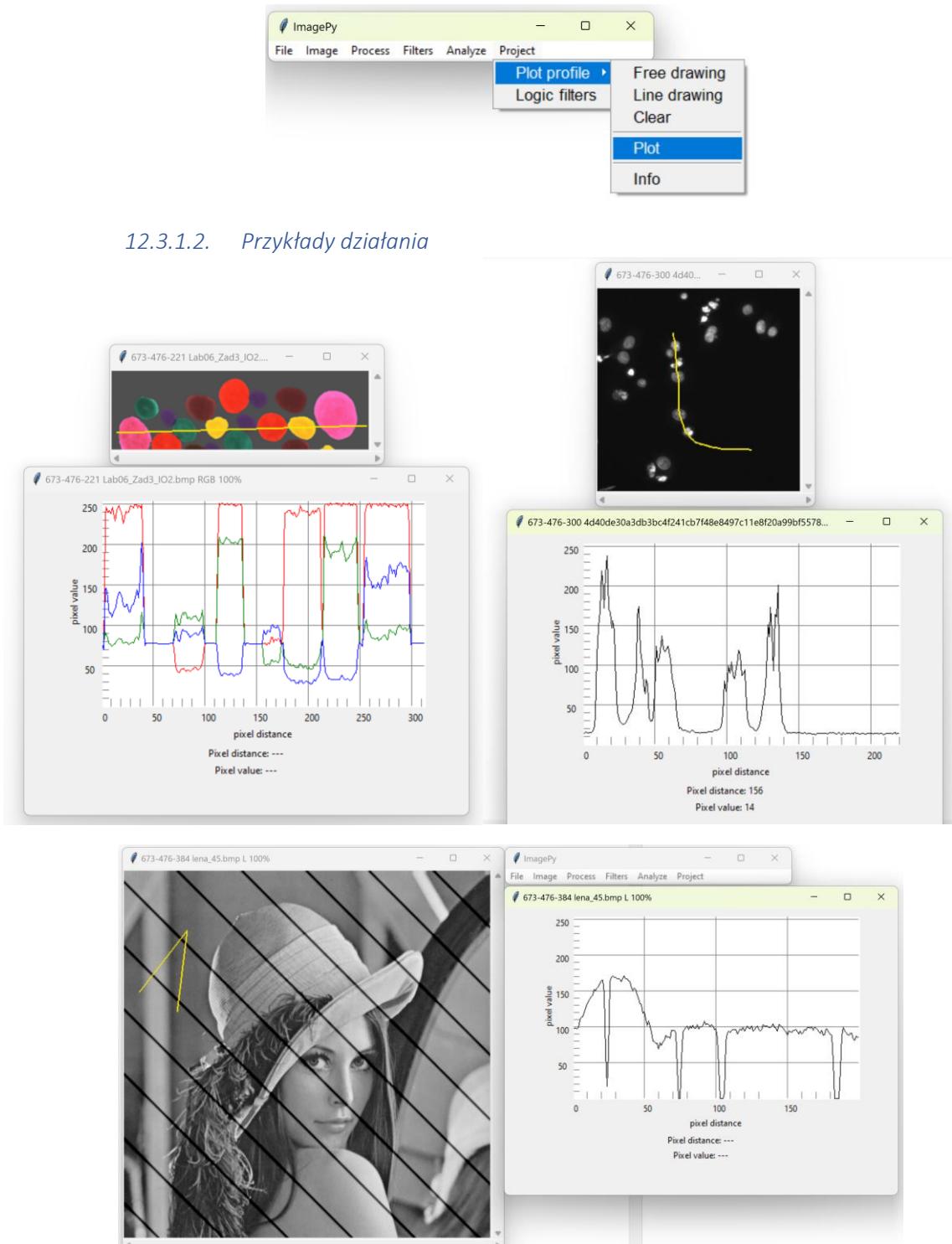
- b. Line drawing – opcja rysowania odcinka lub łamanej. Aby zacząć rysowanie należy użyć LPM w miejscu początkowym odcinka, a następnie wybrać inny punkt na obrazie i po raz kolejny wcisnąć LPM, zostanie wtedy narysowany odcinek pomiędzy punktami. Do narysowania łamanej należy wcisnąć LPM w kolejnym punkcie, co wyznaczy koniec kolejnego odcinka łamanej. Aby wyczyścić aktualną łamana należy wcisnąć PPM (w przeciwnieństwie do opcji Clear nie przerywa to funkcji rysowania)



- c. Clear – opcja usuwa wszelkie zaznaczenia na obrazie i przerywa rysowanie



3. Wygenerować wykres linii profilu dla zaznaczonych pikseli: Project -> Plot profile -> Plot

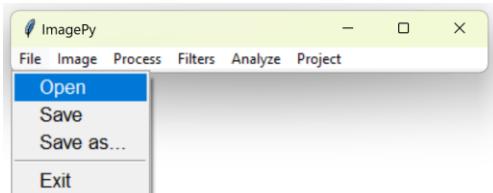


12.3.2. Filtracja logiczna

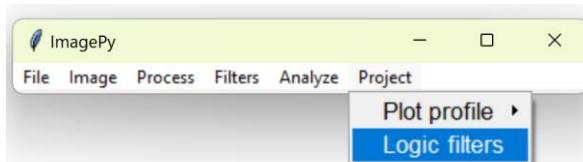
Aby użyć filtracji logicznej należy wcześniej przygotować obraz binarny. Algorytm polega na iterowaniu się przez wszystkie piksele obrazu i obliczaniu wartości wynikowej na podstawie odpowiednich wzorów.

12.3.2.1. Krok po kroku

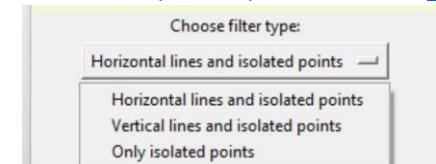
- Otworzyć obraz binarny: File -> Open -> wybieramy obraz binarny -> Otwórz



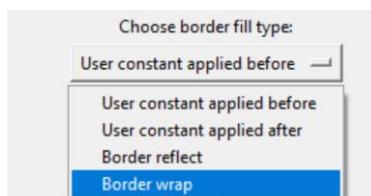
- Uruchomić narzędzie filtracji logicznej: Project -> Logic filters



- Wybrać metodę filtrowania (metody zostały omówione w [opisie teoretycznym](#))



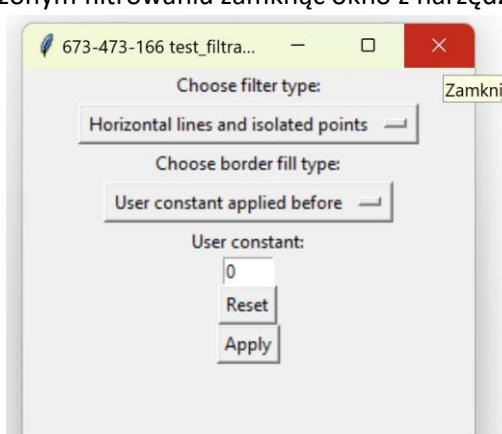
- Wybrać metodę wypełnienia ramki obrazu (metody wypełnienia są analogiczne jak w oprogramowaniu laboratoryjnym)



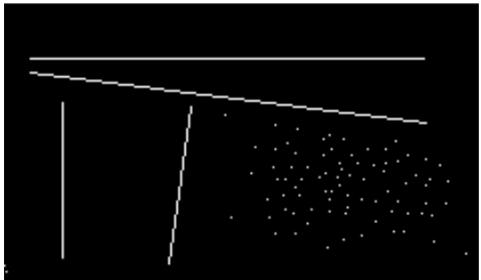
- Wygenerować przefiltrowany obraz: przycisk Apply



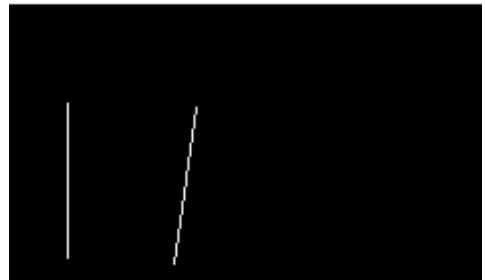
- Aby wrócić do stanu początkowego należy użyć przycisku Reset
- Po zakończonym filtrowaniu zamknąć okno z narzędziem filtracji



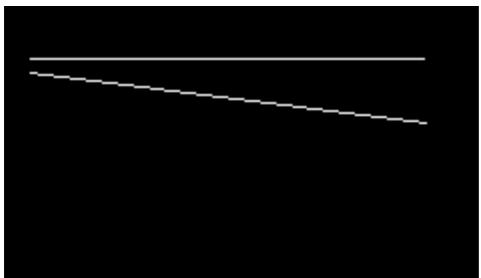
12.3.2.2. Przykłady działania



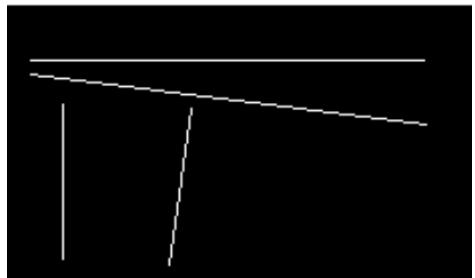
Rysunek 1 Oryginalny obraz binarny



Rysunek 2 Wynik filtrowania linii poziomych i izolowanych punktów



Rysunek 3 Wynik filtrowania linii pionowych i izolowanych punktów



Rysunek 4 Wynik filtrowania izolowanych punktów