Politechnika Wrocławska

# Zastosowania Konwolucji

Piotr Frac

December 2025

## Spis treści

# 1 Zastosowanie splotu - wstęp

Do wykonania było zaimplementowanie następujących zastosowań splotu:

1.1 **Edge Detection**

1.2 **Image Blurring**

1.3 **Image Sharpening**

1.4 **Demosaicing**

Porównując kilka filtrów, dla każdego z nich

## 1.1 Edge Detection

Do wykrywania krawędzi wykorzystywane są gradienty janości obrazów - operatory:

1.1.1 **Laplace operator**

1.1.2 **Sobel operator**

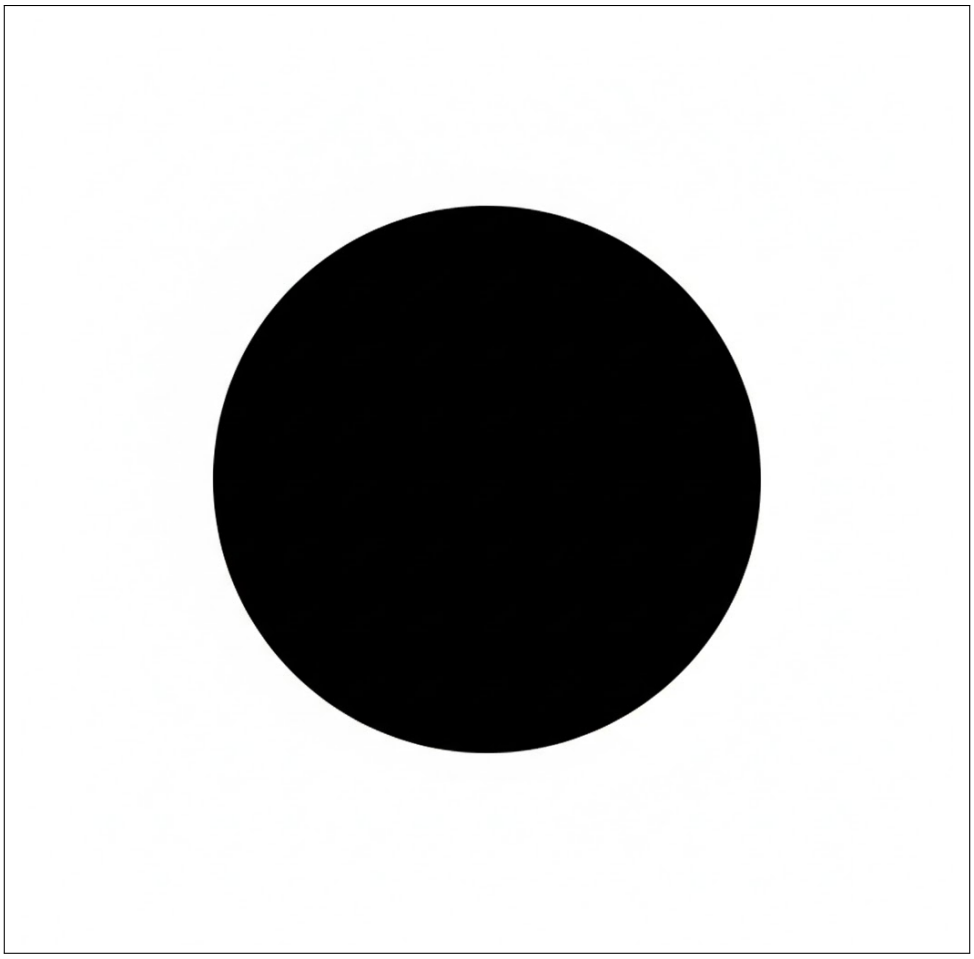
1.1.3 **Prewitt operator**

1.1.4 **Scharr operator**

1.1.5 **Roberts cross**

Do wykonania zadania wykorzystałem zdjecie okregu:

Rysunek 1: Obraz orginalny: 1024px x 1024px

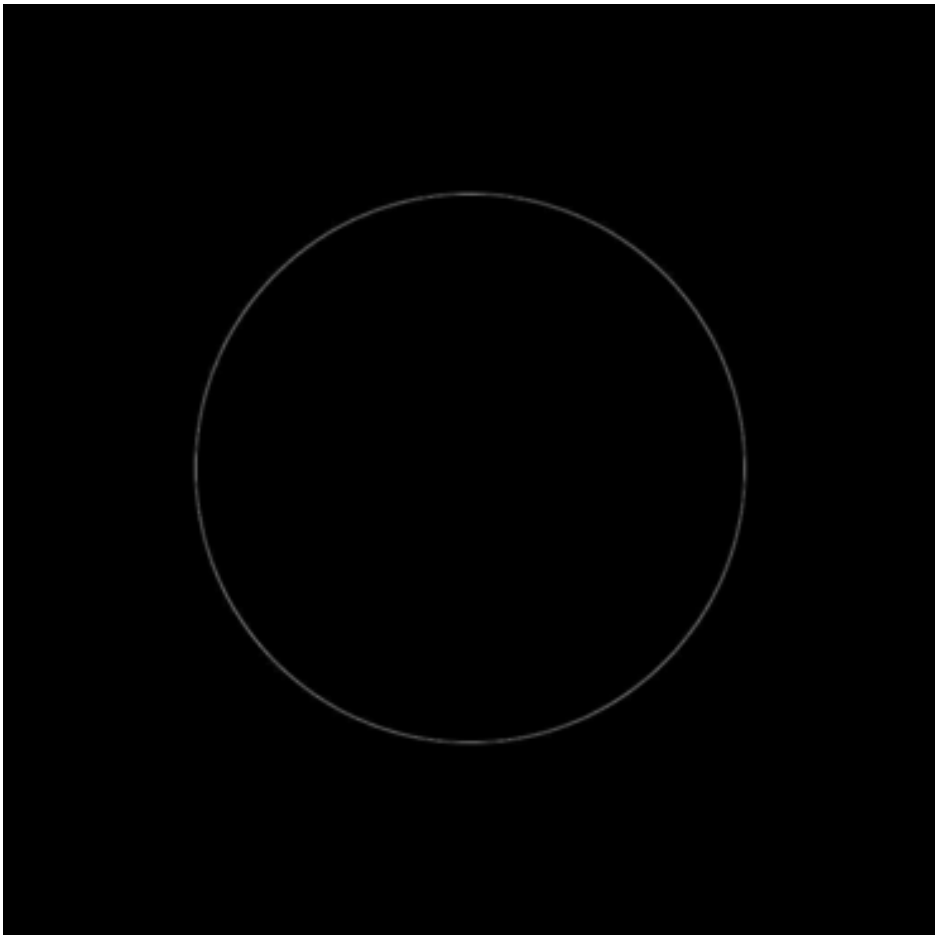


### 1.1.1 Laplace operator

Laplace operator zgodnie z treścią zadania wygląda tak:

$$L = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{1}$$

Po wykonaniu operacji splotu z operatorem Laplaca i przeniesieniem na skale szarosci obraz wyglada tak:



Rysunek 2: Obraz okręgu po spolcie z operatorem Laplaca

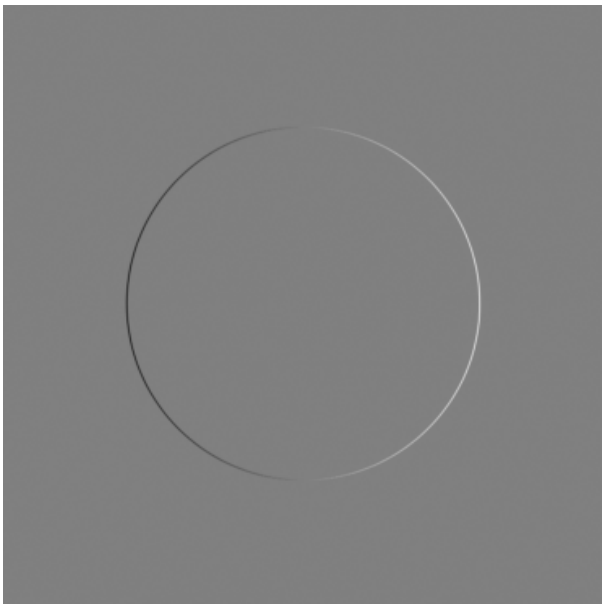
### 1.1.2 Sobel operator

Są to tak naprawde dwa operatory Sobela, których później wyliczamy magnitude:

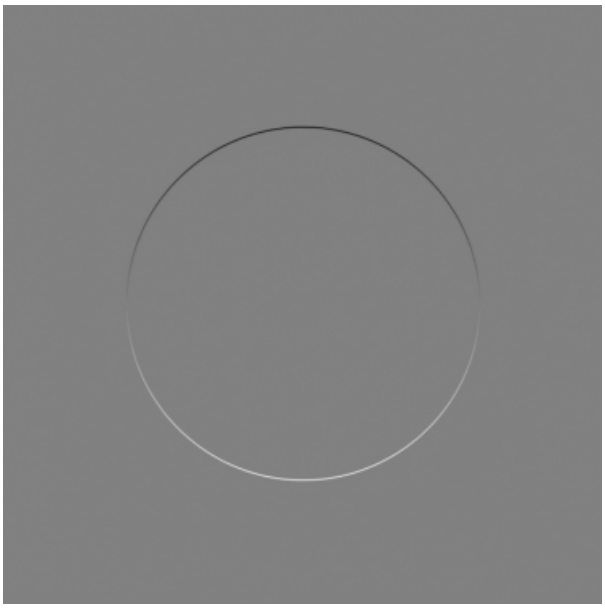
$$S_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad S = \sqrt{(S_x)^2 + (S_y)^2} \tag{2}$$

Wykonujemy konwolucje dla operatorów:

Rysunek 3: Operatory Sobel'a

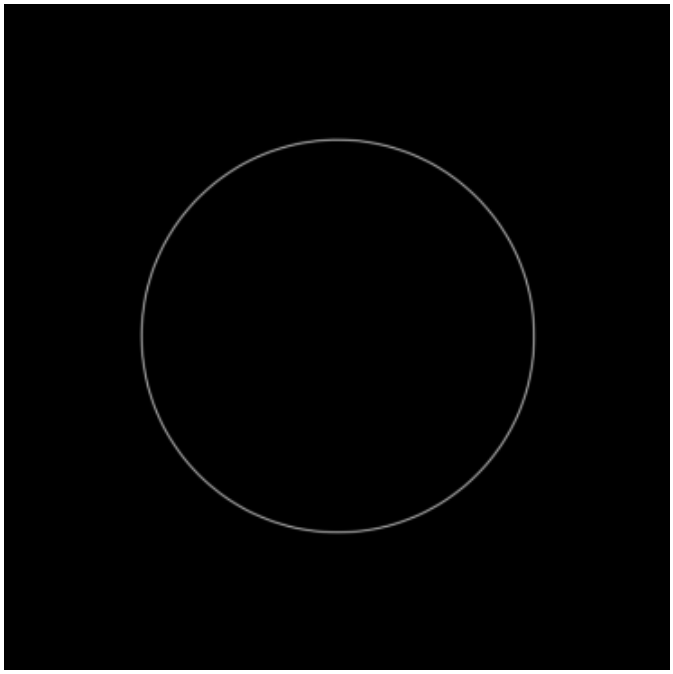


(a) Operator Sobel X



(b) Operator Sobel Y

Następnie obliczamy dla kazdego punktu magnitude:



Rysunek 4: Wynik końcowy operacji używając operatów Sobel'a

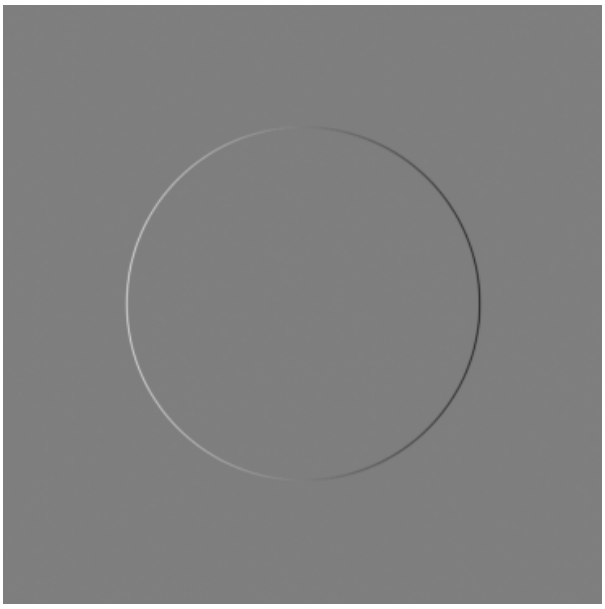
### 1.1.3 Prewitt operator

Prewitt operator jest bardzo podobny do operatora Sobel'a:

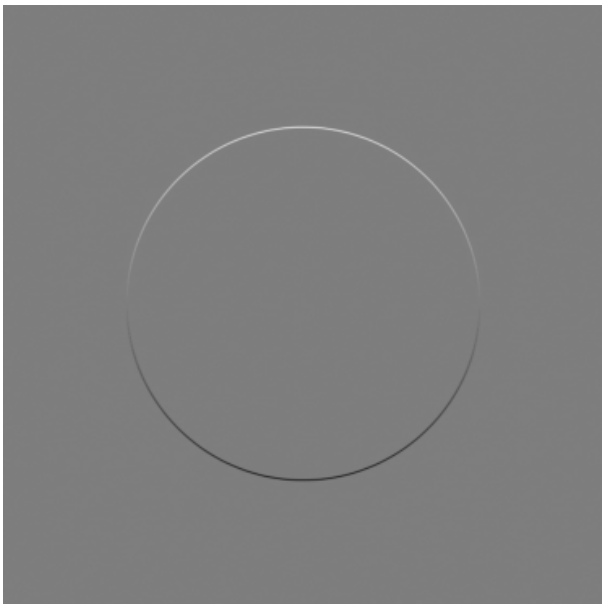
$$P_x = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix} \quad P_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad P = \sqrt{(P_x)^2 + (P_y)^2} \tag{3}$$

Wykonujemy konwolucje dla operatorów:

Rysunek 5: Operatory Prewitt'a

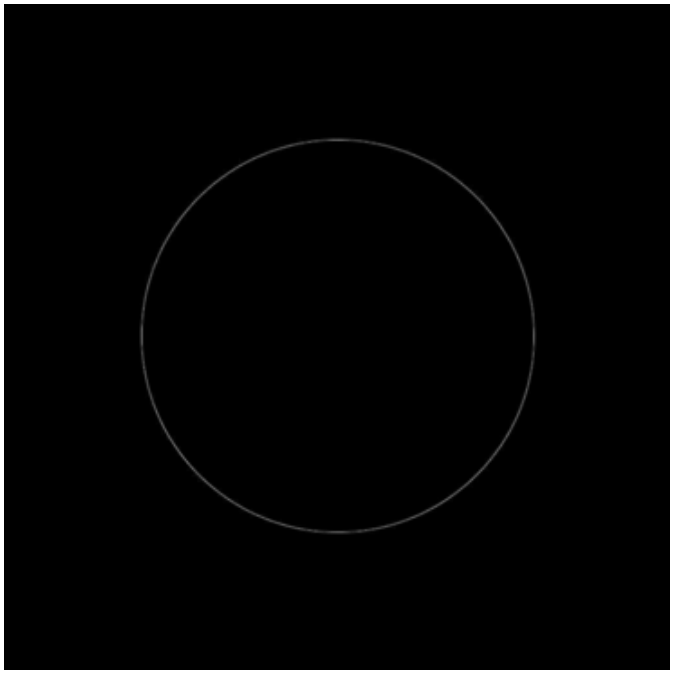


(a) Operator Prewitt X



(b) Operator Prewitt Y

Następnie obliczamy dla kazdego punktu magnitude:



Rysunek 6: Wynik końcowy operacji używając operatów Prewitt'a

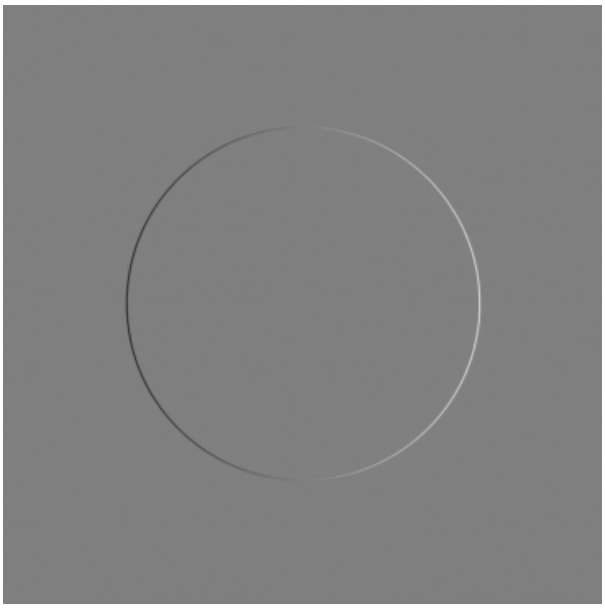
### 1.1.4 Scharr operator

Kolejny bardzo podobny operator do poprzednich:

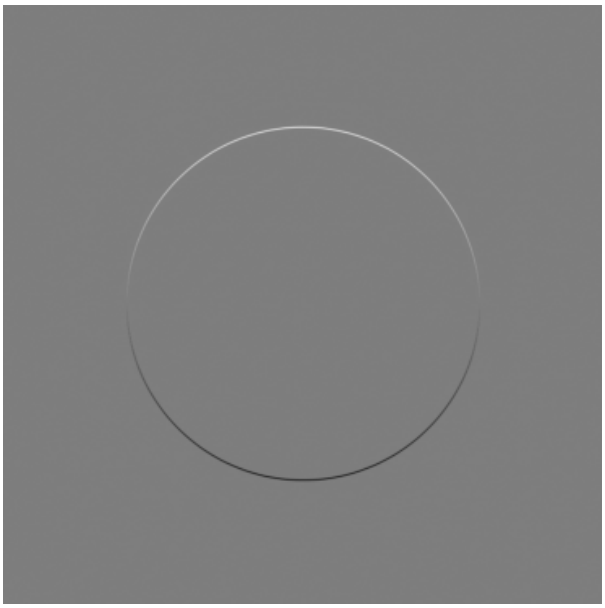
$$Sch_x = \begin{bmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{bmatrix} \quad Sch_y = \begin{bmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix} \quad Sch = \sqrt{(S_x)^2 + (S_y)^2} \tag{4}$$

Wykonujemy konwolucje dla operatorów:

Rysunek 7: Operatory Scharr'a

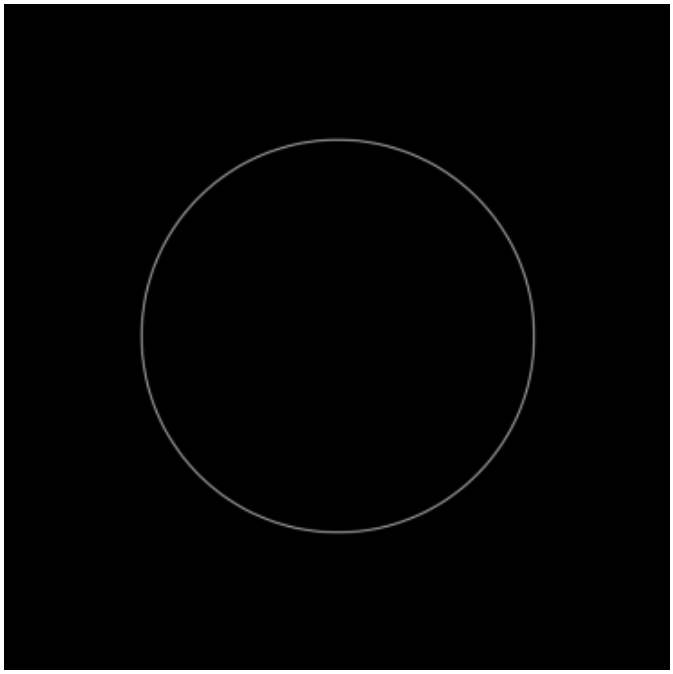


(a) Operator Scharr X



(b) Operator Scharr Y

Następnie obliczamy dla kazdego punktu magnitude:



Rysunek 8: Wynik końcowy operacji używając operatów Scharr'a

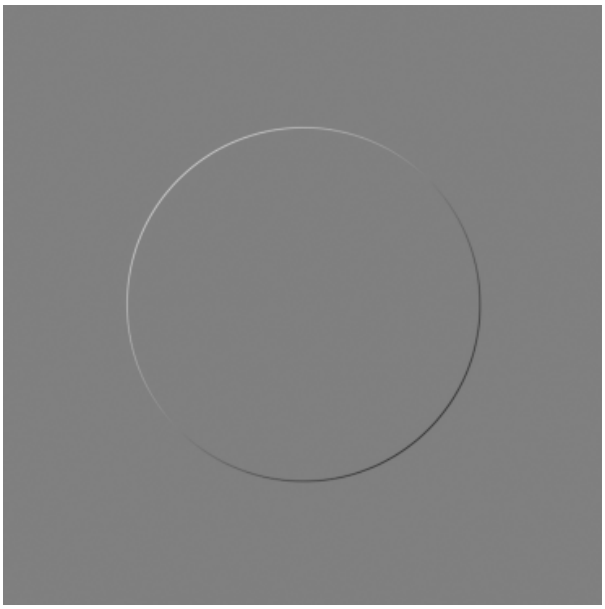
### 1.1.5 Roberts cross

W końcu troche inny operator:

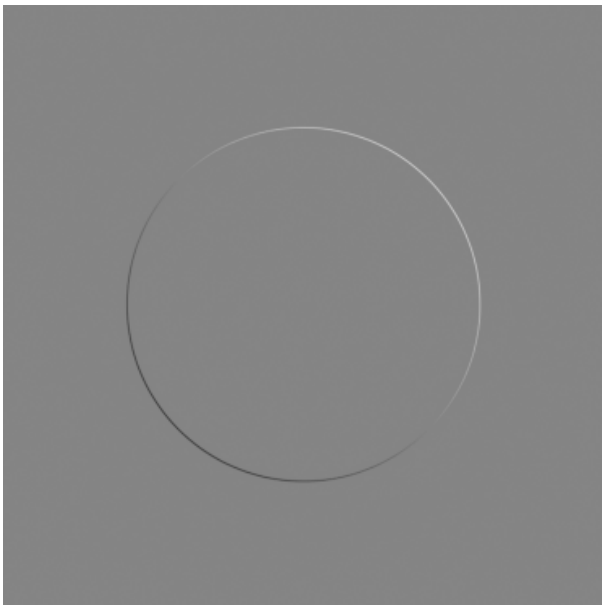
$$R_1 = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad R_2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \quad R = \sqrt{(R_1)^2 + (R_2)^2} \tag{5}$$

Wykonujemy konwolucje dla operatorów:

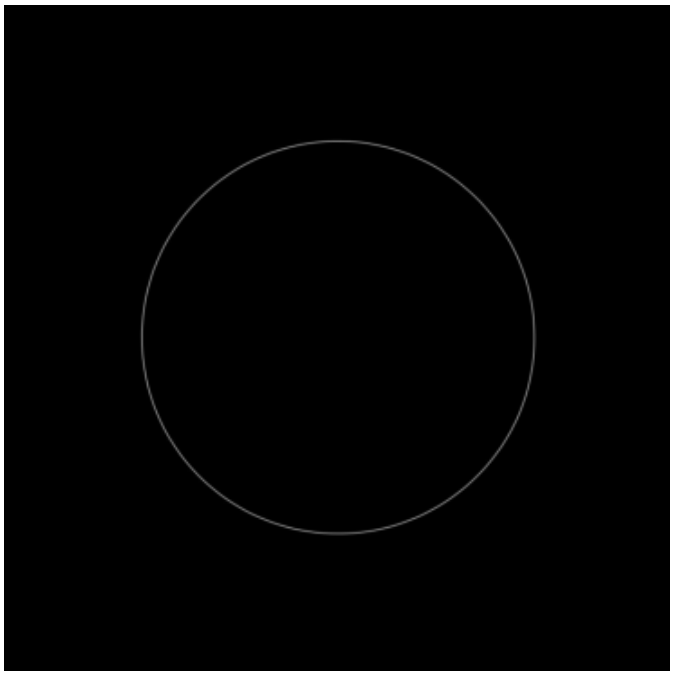
Rysunek 9: Operatory Robert'a



(a) Operator Robert 1



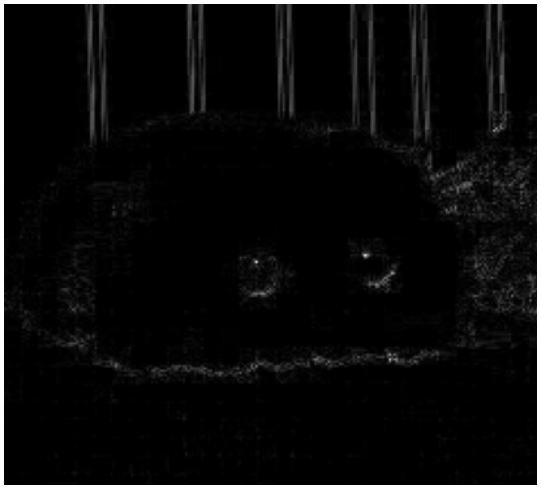
(b) Operator Robert 2



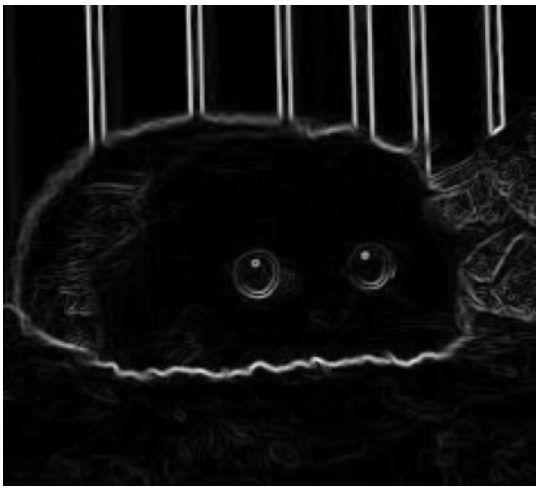
Rysunek 10: Wynik końcowy operacji używając operatów Roberta'a

### 1.1.6 Porównanie operatorów

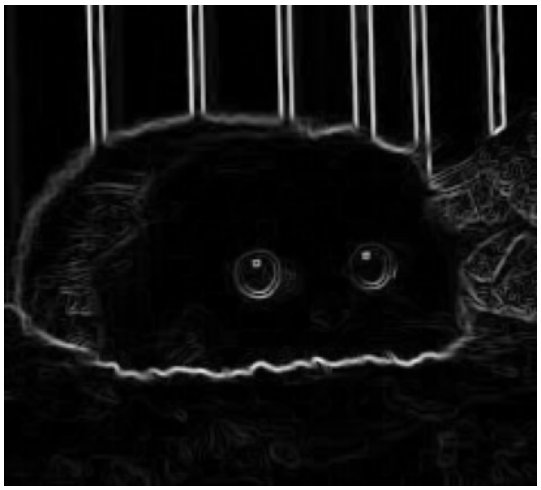
(a) Orginalny Obraz



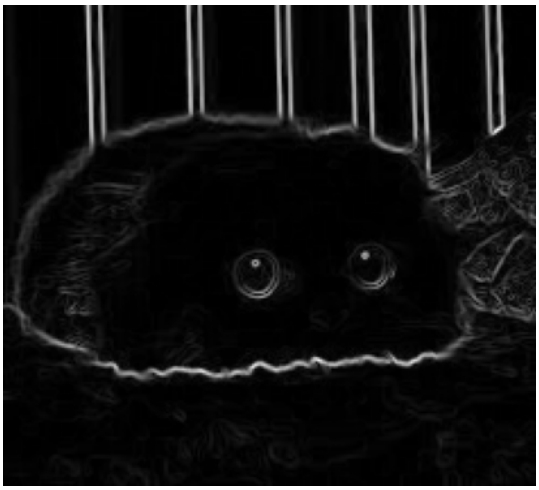
(b) Obraz przetworzony operatorem Laplaca



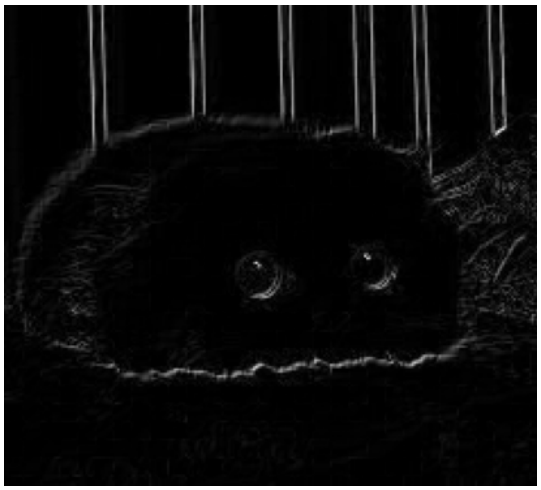
(a) Obraz przetworzony operatorem Sobla



(b) Obraz przetworzony operatorem Perwita



(a) Obraz przetworzony operatorem Scharr'a



(b) Obraz przetworzony operatorem Roberta

## 1.2 Image Blurring

Do rozmycia obrazu należało wykorzystać jądra uśredniające między innymi gaussowskie:

$$G = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{6}$$

inne kernele, których uzyłem wywodzą się z tego równania:

$$K(size) = \begin{bmatrix} size^{-2} & size^{-2} & \cdots & size^{-2} \\ size^{-2} & size^{-2} & \cdots & size^{-2} \\ \vdots & \vdots & \ddots & \vdots \\ size^{-2} & size^{-2} & \cdots & size^{-2} \end{bmatrix}_{size \times size} \tag{7}$$

Do tego zadania użyłem kerneli o wielkosci 5 i 11

$$K_{(5)} = \begin{bmatrix} 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \\ 0.04 & 0.04 & 0.04 & 0.04 & 0.04 \end{bmatrix} \qquad K_{(11)} = \begin{bmatrix} 0.00826 & 0.00826 & \cdots & 0.00826 \\ 0.00826 & 0.00826 & \cdots & 0.00826 \\ \vdots & \vdots & \ddots & \vdots \\ 0.00826 & 0.00826 & \cdots & 0.00826 \end{bmatrix}_{11 \times 11}$$

**Obrazy**

(a) Obraz orginalny

(b) Obraz rozmyty jądrem Gaussowskim

(a) Obraz rozmyty $K(5)$

(b) Obraz rozmyty $K(11)$

### 1.3 Image Sharpening

Do wyostrzenia obrazu należało wykorzystać podany w zadaniu kernel W, a reszta kerneli wyszla z takiego równania:

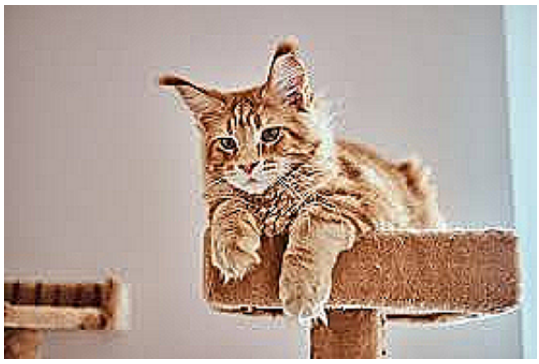
$$W(amount) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} * amount \tag{8}$$

**Kernele:**

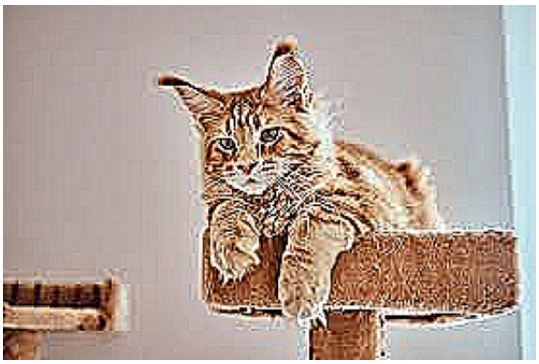
$$W(1) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \qquad W(2) = \begin{bmatrix} -0 & -2 & 0 \\ -2 & 9 & -2 \\ 0 & -2 & 0 \end{bmatrix} \qquad W(8) = \begin{bmatrix} -0 & -8 & 0 \\ -8 & 33 & -8 \\ 0 & -8 & 0 \end{bmatrix}$$

**Obrazy**

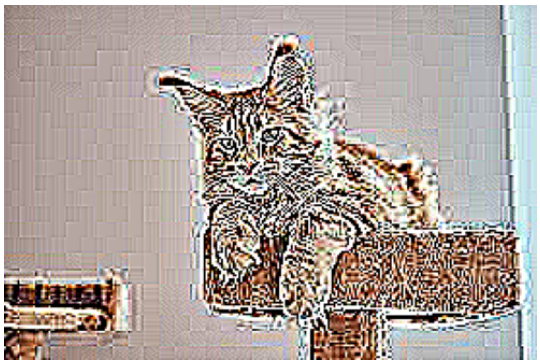
(a) Obraz orginalny



(b) Obraz zaostrzony W(1)



(a) Obraz zaostrzony W(2)



(b) Obraz zaostrzony W(8)

## 2 Źródła


Laplace operator - `https://en.wikipedia.org/wiki/Laplace_operator`
Sobel operator - `https://en.wikipedia.org/wiki/Sobel_operator`
Prewitt operator - `https://en.wikipedia.org/wiki/Prewitt_operator`
Sharr operator + All above - `https://www.geeksforgeeks.org/software-engineering/edge-detection-using-prewitt-scharr-and-sobel-operator/`
Roberts cross - `https://en.wikipedia.org/wiki/Roberts_cross`
Konwolucja - `https://taylorpetrick.com/blog/post/convolution-part3`