
	Politechnika Bydgoska im. J. J. Śniadeckich Wydział Telekomunikacji, Informatyki i Elektrotechniki <b>Zakład Systemów Teleinformatycznych</b>			
<b>Przedmiot</b>	Przetwarzanie obrazów			
<b>Prowadzący</b>	mgr inż. Grzegorz Czeczot			
<b>Temat</b>	Krawędzie			
<b>Student</b>	Paweł Jońca			
<b>Nr lab.</b>	6	<b>Data wykonania</b>	26.11.2024	
<b>Ocena</b>		<b>Data oddania spr.</b>	26.11.2024	

### Zad 1

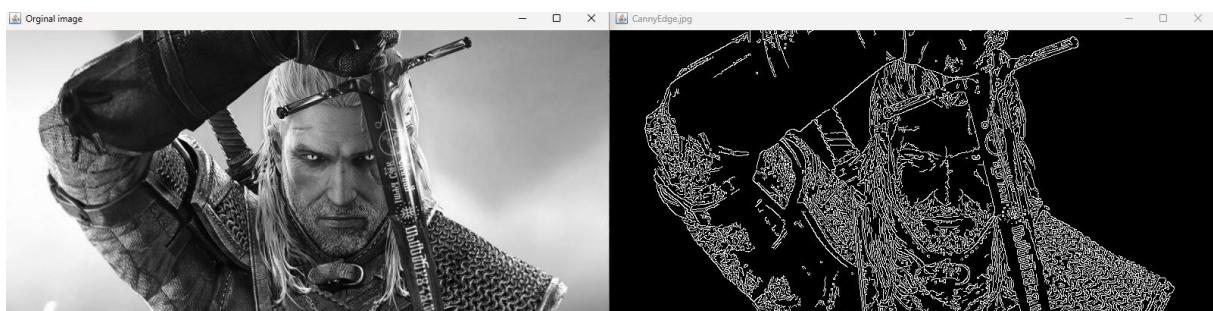
```

> import ...

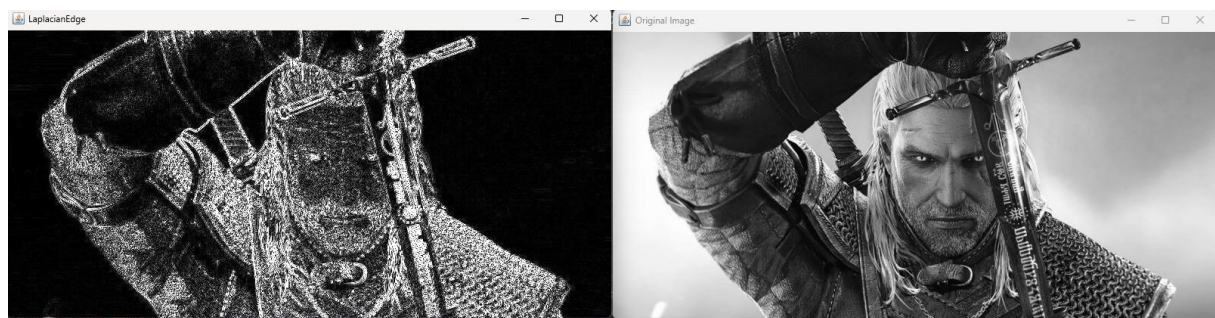
public class Zad1 {  ± Pawelgabrieljonca *
    public static void main(String[] args) {  ± Pawelgabrieljonca *
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        Mat src = Imgcodecs.imread( filename: "geralt.jpg", Imgcodecs.IMREAD_GRAYSCALE);
        if (src.empty()) {
            System.out.println("Failed to load image.");
            return;
        }
        Mat edges = new Mat();
        Imgproc.Canny(src, edges, threshold1: 100, threshold2: 200);

        HighGui.imshow( winname: "CannyEdge.jpg", edges);
        HighGui.imshow( winname: "Original image", src);
        HighGui.waitKey( delay: 0);
    }
}

```



## Zad 2

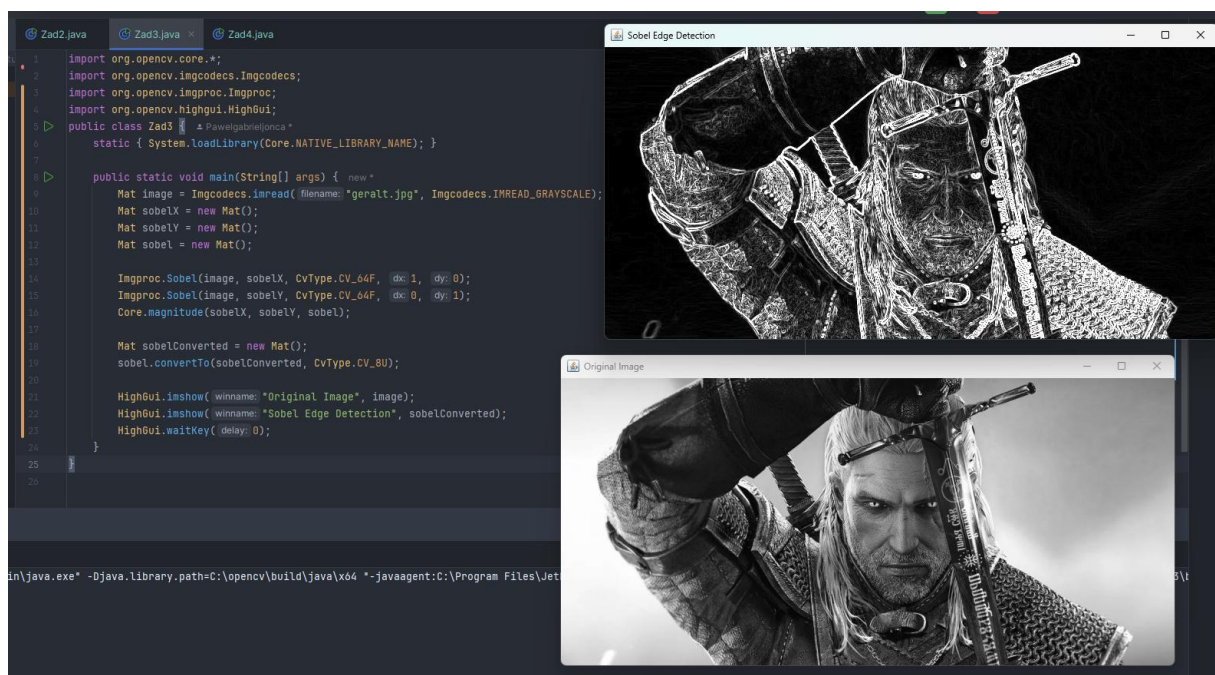


```
import org.opencv.core.*;
import org.opencv.highgui.HighGui;
import org.opencv.imgproc.Imgproc;
import org.opencv.imgcodecs.Imgcodecs;

public class Zad2 {
    // Pawelgabrieljonca
    public static void main(String[] args) {
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        Mat src = Imgcodecs.imread("geralt.jpg", Imgcodecs.IMREAD_GRAYSCALE);
        if (src.empty()) {
            System.out.println("Failed to load image.");
            return;
        }
        // Wykonaj detekcję krawędzi Laplacian
        Mat edges = new Mat();
        // Parametry: źródło, cel, głębokość, rozmiar operatora Sobela (kernela), skalowanie, delta, borderType
        Imgproc.Laplacian(src, edges, CvType.CV_16S, 3, 1, 0, BorderType.BORDER_REPLICATE);
        // Przekształć wynik na format 8-bitowy (widoczny)
        Mat absEdges = new Mat();
        Core.convertScaleAbs(edges, absEdges);

        HighGui.imshow("Original Image", src);
        HighGui.imshow("LaplacianEdge", absEdges);
        HighGui.waitKey(0);
    }
}
```

## Zad 3



## Zad 4

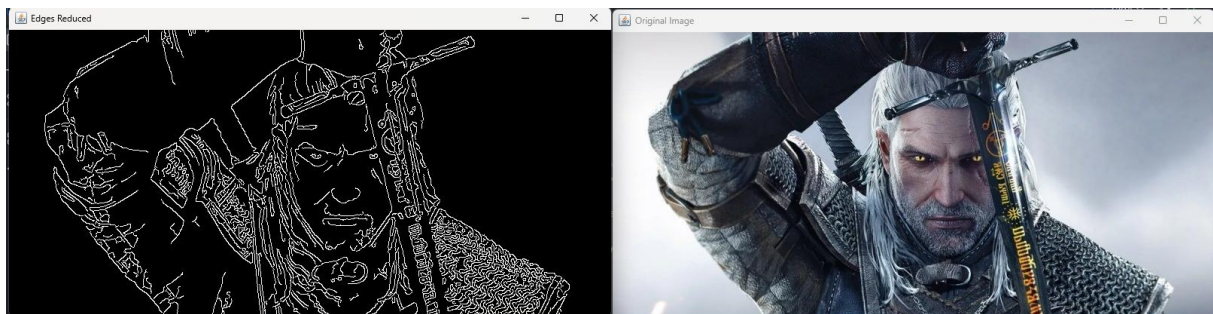
```
import org.opencv.core.*;
import org.opencv.imgcodecs.Imgcodecs;
import org.opencv.imgproc.Imgproc;
import org.opencv.highgui.HighGui;

public class Zad4 {
    public static void main(String[] args) {
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);

        Mat image = Imgcodecs.imread("geralt.jpg");
        if (image.empty()) {
            System.out.println("Nie udało się wczytać obrazu.");
            return;
        }
        Mat blurred = new Mat();
        Mat edges = new Mat();

        Imgproc.GaussianBlur(image, blurred, new Size(5, 5), 1.5);
        Imgproc.Canny(blurred, edges, 50, 150);

        HighGui.imshow("Original Image", image);
        HighGui.imshow("Edges Reduced", edges);
        HighGui.waitKey();
    }
}
```



## Zad 5

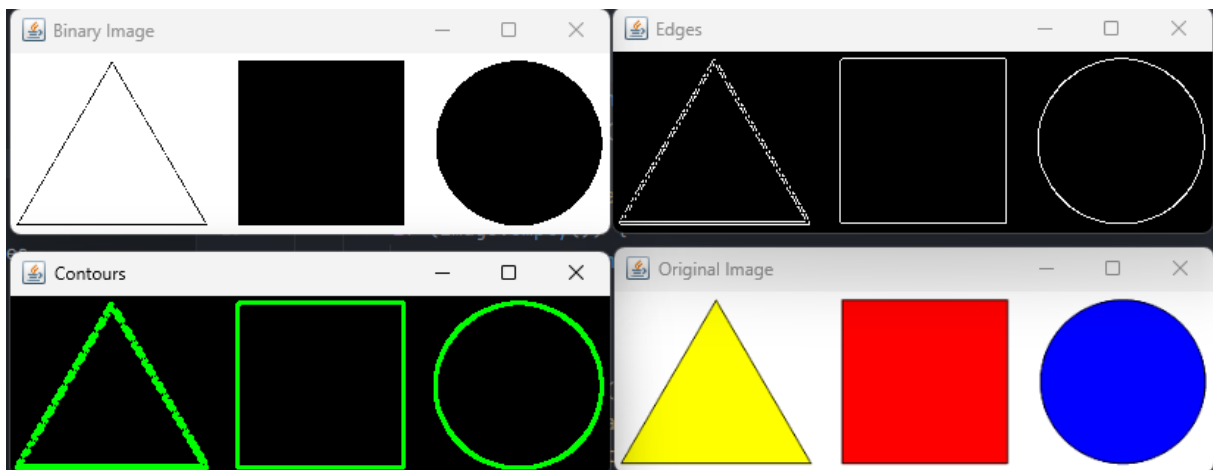
```
> import ...
> public class Zad5 { new *
>     public static void main(String[] args) { new *
        System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
        Mat image = Imgcodecs.imread(filename: "images.png");
        if (image.empty()) {
            System.out.println("Nie udało się wczytać obrazu.");
            return;
        }
        Mat gray = new Mat();
        Mat binary = new Mat();
        Mat edges = new Mat();
        Mat contourOutput = new Mat();

        Imgproc.cvtColor(image, gray, Imgproc.COLOR_BGR2GRAY);
        Imgproc.threshold(gray, binary, thresh: 128, maxval: 255, Imgproc.THRESH_BINARY);
        Imgproc.Canny(binary, edges, threshold1: 100, threshold2: 200);

        List<MatOfPoint> contours = new ArrayList<>();
        Mat hierarchy = new Mat();
        Imgproc.findContours(edges, contours, hierarchy, Imgproc.RETR_TREE, Imgproc.CHAIN_APPROX_SIMPLE);

        contourOutput = Mat.zeros(edges.size(), CvType.CV_8UC3);
        for (int i = 0; i < contours.size(); i++) {
            Imgproc.drawContours(contourOutput, contours, i, new Scalar(0, 255, 0), thickness: 2);
        }

        HighGui.imshow(winname: "Original Image", image);
        HighGui.imshow(winname: "Binary Image", binary);
        HighGui.imshow(winname: "Edges", edges);
        HighGui.imshow(winname: "Contours", contourOutput);
        HighGui.waitKey();
    }
}
```



#### Wnioski:

Metoda Canny skutecznie wykrywa krawędzie na obrazie, a zmiana progów pozwala kontrolować ich ilość i szczegółowość – niższe wartości dodają więcej detali, ale mogą wprowadzać szumy. Metody Laplacian i Sobel również dobrze wykrywają krawędzie, jednak wymagają wcześniejszego wygładzenia obrazu, aby wyniki były bardziej czytelne. Do wykrywania konturów konieczna była binaryzacja i detekcja krawędzi, co najlepiej sprawdziło się przy prostych obrazach, takich jak figury geometryczne.