

Sprawozdanie 1.

Laboratoria 2., 3. i 4.

Mirosław Kołodziej

05.11.2021

1. Laboratoria 2.

Celem naszych zajęć laboratoryjnych było zapoznanie się z podstawowymi pojęciami wykorzystywanymi w analizie obrazów. Wykorzystywaliśmy do podstawowych przekształceń poniższy obrazek:



Rysunek 1. Oryginalna wersja obrazka wykorzystywanego w trakcie zajęć

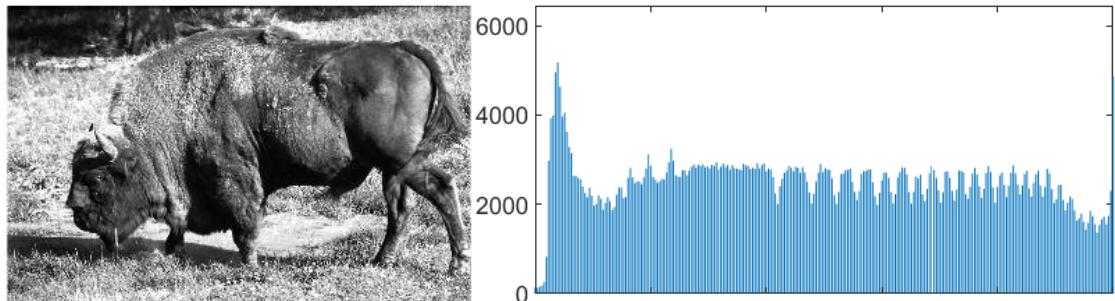
Po wczytaniu obrazek był przechowywany w trójwymiarowej macierzy (wysokość x szerokość x 3 kanały zawierające informacje o natężeniu danego koloru w RGB) typu uint8, woleliśmy jednak pracować na typie double. Musieliśmy jednak przy tym unormować wartości natężenia kolorów.

Następnie wyświetliśmy każdy kanał (R, G oraz B) z osobna, zerując wartości znajdujące się w pozostałych kanałach.

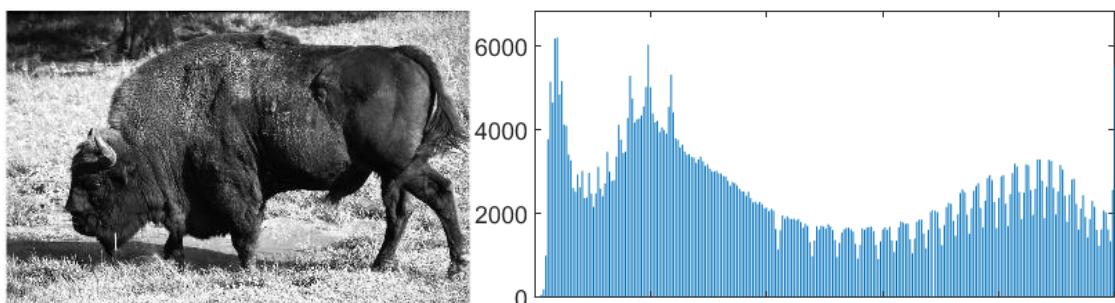


Rysunki 2., 3., 4. Obrazki z jednym kanałem, kolejno R, G oraz B

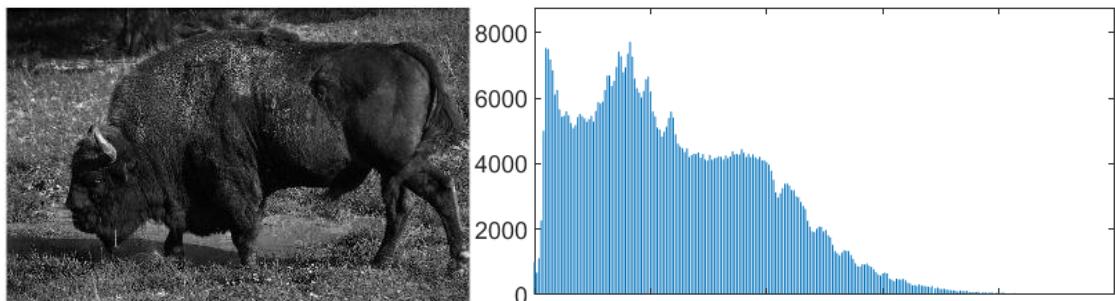
Jak możemy zauważyc, najsłabiej widoczny jest obrazek zawierający tylko niebieski kanał. Aby sprawdzić dlaczego tak się dzieje, sporządziliśmy histogramy pokazujące ilość pikseli w danym natężeniu każdego z kolorów R, G oraz B. Przy okazji przekonwertowaliśmy powyższe obrazki do skali szarości.



Rysunki 5., 6. Obrazek pokazujący czerwony kanał w skali szarości wraz z histogramem



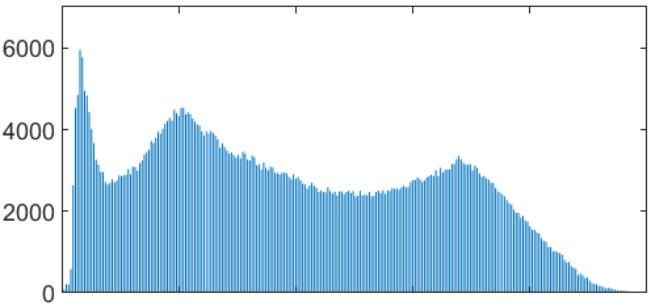
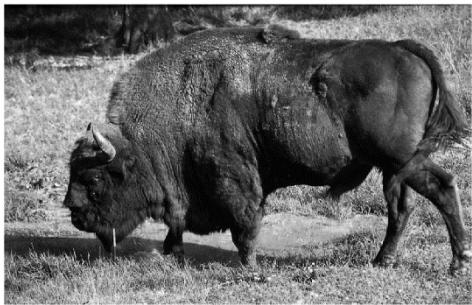
Rysunki 7., 8. Obrazek pokazujący zielony kanał w skali szarości wraz z histogramem



Rysunki 9., 10. Obrazek pokazujący niebieski kanał w skali szarości wraz z histogramem

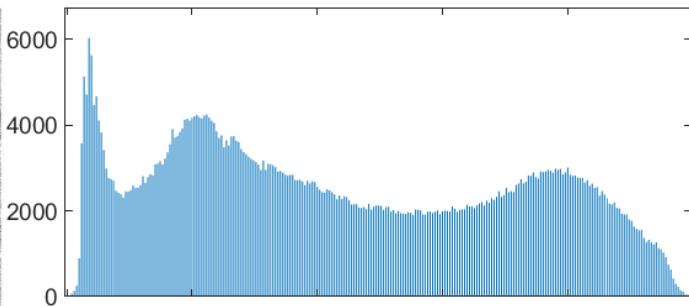
Możemy zauważyc, że histogram dla koloru czerwonego jest jednorodny, w przypadku koloru zielonego otrzymujemy histogram dwumodalny. Histogram dla koloru niebieskiego wyjaśnia nam słabą widoczność – ten kanał ma bardzo małą ilość jasnych odcieni, zaś dominują ciemniejsze.

Konwertując jedynie pojedynczy kanał tracimy jednak wiele danych, postanowiliśmy więc uzyskać skalę szarości w inny sposób – policzyliśmy dla każdego piksela średnią wartość z trzech kanałów R, G i B.



Rysunki 11., 12. Obrazek pokazujący średnią wartość dla pikseli w skali szarości wraz z histogramem

Pomimo braku utraty danych chcieliśmy uzyskać jeszcze lepszy efekt. Skorzystaliśmy z modelu barw YUV, który był używany w telewizji w trakcie przekształcania obrazu na czarno-biały. Przemnożyliśmy każdy kanał przez następujące wartości: [0.299, 0.587, 0.114].



Rysunki 13., 14. Obrazek pokazujący konwersję do skali szarości za pomocą modelu barw YUV wraz z histogramem

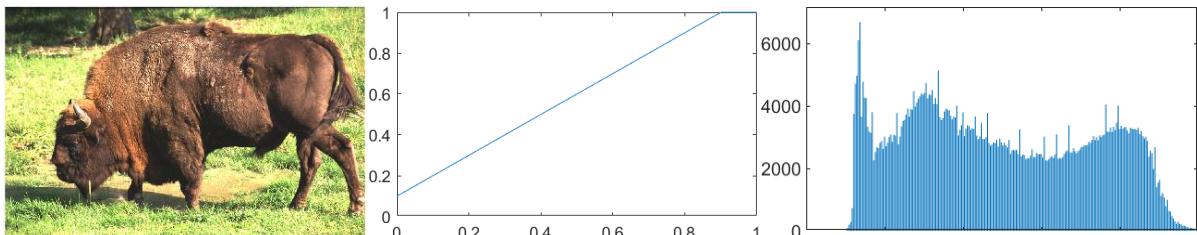
Ostatnimi zagadnieniami omówionymi przez nas w trakcie zajęć były podstawowe przekształcenia obrazu stosowanymi między innymi w grach komputerowych, czyli jasność, kontrast oraz korekcja gamma.

Zaczęliśmy od zmiany jasności. Jest to po prostu zwiększenie wartości natężenia danego koloru w każdym pikselu – histogram przesuwa się w prawo lub w lewo. Wartość dodawana powinna się mieścić w przedziale $(-1,1)$, żeby obrazek nie zmienił się w jednokolorowy (biały lub czarny). Zastosowaliśmy również warunki, które zabezpieczały nas przed wyjściem poza zakres:

$$imb(imb > 1) = 1;$$

$$imb(imb < 0) = 0;$$

Otrzymaliśmy następujące wyniki:



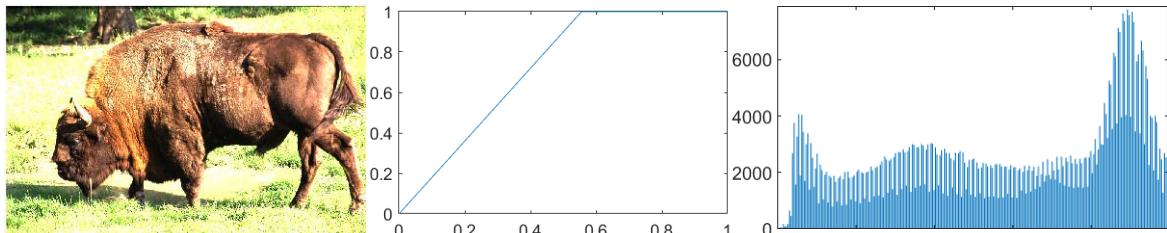
Rysunki 15., 16., 17. Obrazek mający zwiększoną jasność o 0.1 wraz z wykresem obrazującym przekształcenie oraz histogramem

Dla uzyskania lepszego zobrazowania stosowanych przekształceń pracowałem na oryginalnym obrazku, zaś przedstawiony histogram to średnia wartości z trzech warstw kolorów R, G i B.

Kolejnym omówionym przez nas przekształceniem była zmiana kontrastu, czyli przemnożenie każdego piksela o zadaną wartość. W tym przypadku również musielibyśmy dodać warunek zabezpieczający:

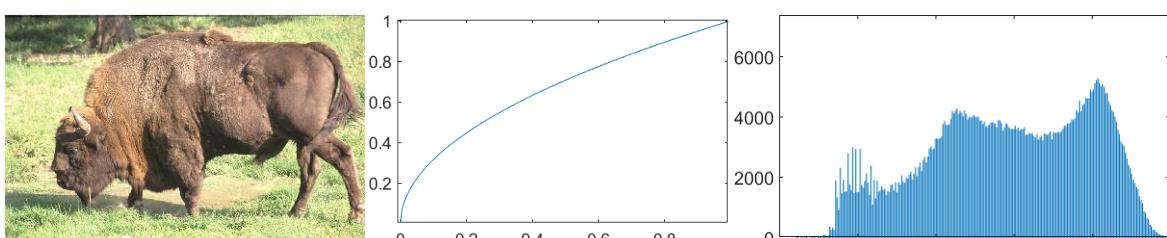
$$imc(imc > 1) = 1;$$

Otrzymaliśmy następujące wyniki:



Rysunki 18., 19., 20. Obrazek mający kontrast ustawiony na 1.8 wraz z wykresem obrazującym przekształcenie oraz histogramem

Ostatnim omawianym przez nas przekształceniem była korekcja gamma. Wartość każdego piksela podnosimy do określonej potęgi. Otrzymaliśmy następujące wyniki:



Rysunki 21., 22., 23. Obrazek z korekcją gamma o 0.5 wraz z wykresem obrazującym przekształcenie oraz histogramem

2. Laboratoria 3.

Na kolejnych zajęciach zajęliśmy się tematyką filtrów. Podobnie jak podczas drugich zajęć, korzystaliśmy z obrazka żubra przekonwertowanego do skali szarości. Tym razem nie przekształcaliśmy wszystkich pikseli w ten sam sposób, lecz staraliśmy się obliczyć ich wartości na podstawie pikseli sąsiadujących z nimi.

Zajęcia rozpoczęliśmy od zapoznania się z pojęciem sąsiedztwa w kontekście pikseli i jego dwoma rodzajami:

- sąsiedztwo Moore'a, czyli sąsiedztwo centralnego piksela z ośmioma innymi, które mają z nim wspólną krawędź lub wierzchołek
- sąsiedztwo von Neumanna, czyli sąsiedztwo centralnego piksela z czterema innymi, które mają z nim wspólną krawędź.

Następnie przeszliśmy do tematyki filtrów. Pierwszym omawianym przez nas filtrem był filtr dolnoprzepustowy (nazywanym także jako wygładzający). Ustala on nową wartość piksela na podstawie uśrednienia wartości jego sąsiadów. Skorzystaliśmy przy tym z macierzy kwadratowej (w przypadku poniższego obrazka jest to macierz 6x6) wypełnionej jedynkami, którą następnie podzieliliśmy przez kwadrat rozmiaru macierzy.

$$\begin{bmatrix} 1/36 & 1/36 & 1/36 & 1/36 & 1/36 & 1/36 \\ 1/36 & 1/36 & 1/36 & 1/36 & 1/36 & 1/36 \\ 1/36 & 1/36 & 1/36 & 1/36 & 1/36 & 1/36 \\ 1/36 & 1/36 & 1/36 & 1/36 & 1/36 & 1/36 \\ 1/36 & 1/36 & 1/36 & 1/36 & 1/36 & 1/36 \\ 1/36 & 1/36 & 1/36 & 1/36 & 1/36 & 1/36 \end{bmatrix}$$

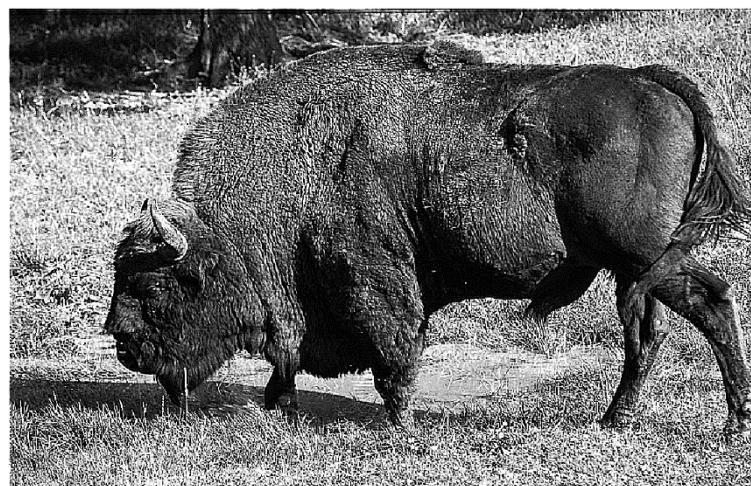


Rysunek 24. Obrazek z nałożonym filtrem dolnoprzepustowym

Nałożenie tego filtra powoduje rozmycie obrazka. Wokół niego pojawia się również cienka ramka przez to, że przy wykonywaniu operacji na pikselach znajdujących się na granicy zdjęcia macierz wychodzi poza obrazek.

Kolejnym omawianym filtrem był filtr górnoprzepustowy. Skorzystaliśmy z niego w identyczny sposób jak z dolnoprzepustowego, jednak wprowadziliśmy dwie zmiany. Użyliśmy ujemnych wag, a na środek macierzy wstawiliśmy kwadrat jej wymiarów (w poniższym przypadku była to wartość 9). Również nie normalizowaliśmy macierzy.

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$



Rysunek 25. Obrazek z nałożonym filtrem górnoprzepustowym

Jak możemy zauważyc, nałożenie filtra spowodowało zwiększenie ostrości oraz kontrastu.

Następnie omówiliśmy filtr, który ma ujemne wartości (-1) oraz dodatnią w środku (kwadrat wymiarów macierzy minus 1). Wagi te sumują się do zera.

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



Rysunek 26. Obrazek z nałożonym filtrem wykrywającym krawędzie

Możemy zauważyc, że filtr dobrze ukazuje krawędzie przy zmianach kolorów (np. dobrze jest do widoczne na żubrzu).

Następnie poznaliśmy pojęcie inwersji. Polega ona na odwróceniu wartości każdego piksela. Jasne piksele stają się ciemne i odwrotnie dla pikseli ciemnych. Żeby wykonać tą operację odejmowaliśmy wartość 1 od każdego piksela.



Rysunek 27. Obrazek po inwersji

Kolejnym omawianym przez nas filtrem był filtr medianowy. Początkowo musieliszy określić wielkość obszaru, na którym będziemy filtrować nasz obrazek (w poniższym przypadku 10×10). Następnie poddajemy go filtracji za pomocą polecenia medfilt2.



Rysunek 28. Obrazek z nałożonym filtrem medianowym

Możemy zauważyc, że filtr medianowy zmniejszył szумy oraz rozmył obrazek. Działa podobnie do filtra dolnoprzepustowego, jednak nie generuje on nowych wartości pikseli, tylko korzysta z wartości występujących w określonym otoczeniu. Zaletą tego filtra jest jego odporność na zakłócenia jednostkowe.

Następnie omówiliśmy pojęcie binaryzacji. Polega ona na przejściu ze skali szarości do obrazu czarno-białego, czyli zamienienia pikseli poniżej pewnej określonej wartości na białe, zaś powyżej tej wartości na czarne.



Rysunki 29., 30. Obrazek po binaryzacji oraz po inwersji i binaryzacji dla progu 0.4

Stwierdziliśmy jednak, że zamiast podawać określony próg, możemy go wyznaczyć w bardziej dokładny sposób. Pomocna okazała się w tym przypadku metoda progowania Otsu, która wyznaczyła nam próg.



Rysunki 31., 32. Obrazek po binaryzacji oraz po inwersji i binaryzacji dla progu wyznaczonego metodą Otsu równego 0.4706

Następnie wykonaliśmy ponownie binaryzację. Tym razem użyliśmy jednak do tego metody 'adaptive', która pozwala na lokalne (adaptacyjne) wyznaczenie progu binaryzacji.



Rysunki 31., 32. Obrazek po binaryzacji oraz po inwersji i binaryzacji przy użyciu metody adaptive

Później, na przykładzie z dokumentacji Matlaba, dowiedzieliśmy się, że binaryzację stosuje się między innymi do odzyskania tekstu ze zbyt ciemnego zdjęcia.

What Is Image Filtering in the Spatial Domain?

Filtering is a technique for modifying or enhancing an image. For example, you can filter an image to emphasize certain features or remove other features. Image processing operations implemented with filtering include smoothing, sharpening, and edge enhancement.

Filtering is a *neighborhood operation*, in which the value of any given pixel in the output image is determined by applying some algorithm to the values of the pixels in the neighborhood of the corresponding input pixel. A pixel's neighborhood is some set of pixels, defined by their locations relative to that pixel. (See [Neighborhood or Block Processing: An Overview](#) for a general discussion of neighborhood operations.) *Linear filtering* is filtering in which the value of an output pixel is a linear combination of the values of the pixels in the input pixel's neighborhood.

Convolution

Linear filtering of an image is accomplished through an operation called *convolution*. Convolution is a neighborhood operation in which each output pixel is the weighted sum of neighboring input pixels. The matrix of weights is called the *convolution kernel*, also known as the *filter*. A convolution kernel is a correlation kernel that has been rotated 180 degrees.

For example, suppose the image is

```
A = [17 24 1 8 15
      23 5 7 14 16
      4 6 13 20 22
      10 12 19 21 3]
```

Rysunek 33. Zdjęcie tekstu z dokumentacji Matlaba

What Is Image Filtering in the Spatial Domain?

Filtering is a technique for modifying or enhancing an image. For example, you can filter an image to emphasize certain features or remove other features. Image processing operations implemented with filtering include smoothing, sharpening, and edge enhancement.

Filtering is a *neighborhood operation*, in which the value of any given pixel in the output image is determined by applying some algorithm to the values of the pixels in the neighborhood of the corresponding input pixel. A pixel's neighborhood is some set of pixels, defined by their locations relative to that pixel. (See Neighborhood or Block Processing: An Overview for a general discussion of neighborhood operations.) *Linear filtering* is filtering in which the value of an output pixel is a linear combination of the values of the pixels in the input pixel's neighborhood.

Convolution

Linear filtering of an image is accomplished through an operation called *convolution*. Convolution is a neighborhood operation in which each output pixel is the weighted sum of neighboring input pixels. The matrix of weights is called the *convolution kernel*, also known as the *filter*. A convolution kernel is a correlation kernel that has been rotated 180 degrees.

For example, suppose the image is

```
A = [17 24 1 8 15  
23 5 7 14 16  
4 6 13 28 22  
10 12 19 21 3  
11 16 25 20 14]
```

Rysunek 34. Zdjęcie tekstu z dokumentacji Matlaba po binaryzacji

Kolejnym poznanym przez nas pojęciem była erozja. Polega ona na tym, że jeśli z danym pikselem sąsiaduje przynajmniej jeden czarny piksel, to zamienia go na czarny. W przeciwnym wypadku pozostaje on biały.



Rysunki 35., 36. Obrazek po erozji oraz po inwersji i erozji



Rysunki 37., 38. Porównanie efektów binaryzacji metodą progowania Otsu oraz erozji

Następnie omówiliśmy dylatację. Ta operacja zamienia czarny piksel na biały, gdy sąsiaduje on z przynajmniej jednym pikselem białym. W przeciwnym wypadku nie zmienia się on.



Rysunki 39., 40. Obrazek po dylatacji oraz po inwersji i dylatacji

Na sam koniec zajęć omówiliśmy połączenie dwóch poprzednio opisanych operacji. Najpierw zamknięcie obrazu, czyli wykonanie dylatacji, a potem erozji. Ta operacja usuwa „dziury” w obrazie.



Rysunek 41. Obrazek po inwersji i operacji zamknięcia

Kolejnym przekształceniem była operacja otwarcia, czyli wykonanie erozji, a po niej dylatacji. Wygładza ona krawędzie obrazka przy jednoczesnym zachowaniu rozmiarów wygładzanego obiektu.



Rysunek 42. Obrazek po inwersji i operacji otwarcia

3. Laboratoria 4.

Na kolejnych zajęciach zajęliśmy się tematyką transformaty Fouriera. Wykorzystywaliśmy ją do przeprowadzania różnych operacji, np. filtracji. Tym razem korzystaliśmy z obrazka przedstawiającego Sydney Opera House:



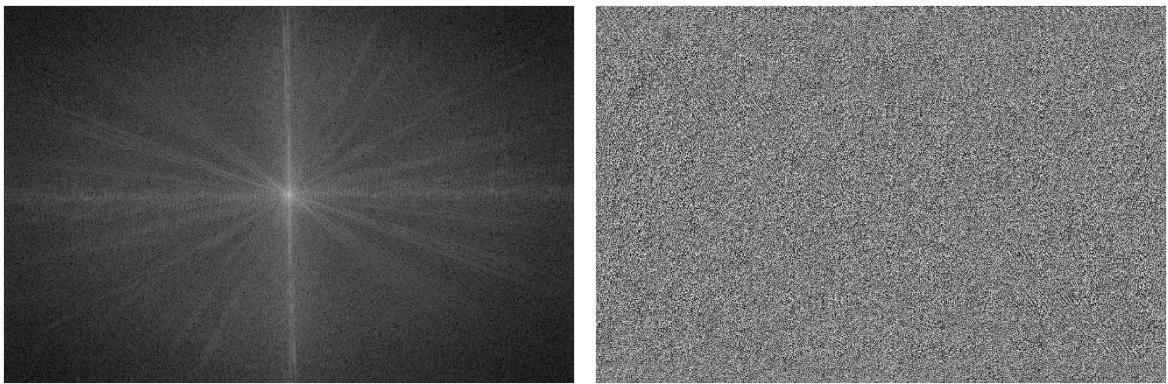
Rysunek 43. Oryginalna wersja obrazka wykorzystywanego w trakcie czwartych zajęć

Przekształciliśmy go do skali szarości i wykonywaliśmy na nim kolejne przekształcenia. Rozpoczęliśmy od szybkiej transformaty Fouriera (fft2).



Rysunek 44. Obrazek w skali szarości

Po przekształceniu chcieliśmy uzyskać parametry takie jak amplituda i faza. Za pomocą odpowiednich poleceń wyświetliśmy obrazy przedstawiające je:



Rysunki 45., 46. Widmo amplitudowe i widmo fazowe

Widmo amplitudowe daje nam informacje na temat cech geometrycznych obrazu, zaś widmo fazowe pozwala nam stwierdzić, czy obraz był modyfikowany. Jeżeli na widmie fazowym można zauważać regularne kształty, może to oznaczać wcześniejszą edycję obrazu.

Następnie próbowaliśmy przywrócić obrazek do jego wcześniejszej formy. Pomogła nam w tym odwrotna transformata Fouriera, którą wykonaliśmy za pomocą funkcji `ifft2`.



Rysunek 47. Przywrócony obrazek za pomocą odwrotnej transformaty Fouriera

Kolejną wykonaną przez nas operacją była zmiana amplitudy, a dokładniej jednego z jej pikseli. Przestawiliśmy jego wartość na 10^5 , i za pomocą transformaty odwrotnej ponownie odzyskaliśmy obrazek.



Rysunek 48. Obrazek po zmianie amplitudy

Jak można zauważyć, ingerencja w tylko jeden piksel spowodowała pojawięcie się ukośnych linii na całym obrazku.

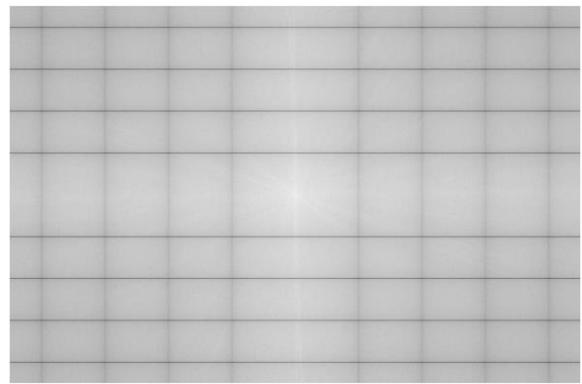
Następnie zmieniliśmy fazę obrazka. Podobnie jak w poprzednim przypadku, zmieniliśmy wartość jednego piksela – dodaliśmy do niego wartość π .



Rysunek 49. Obrazek po zmianie fazy

W tym przypadku zmiana wartości jednego piksela spowodowała inwersję barw całego obrazka.

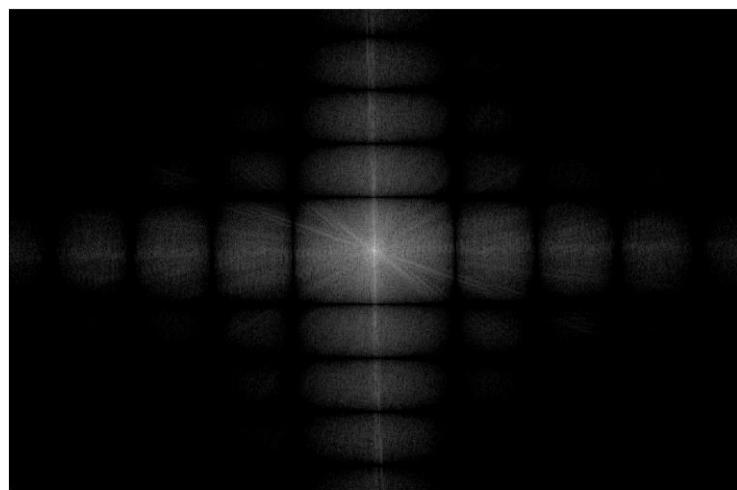
Później przeszliśmy do filtracji. Używaliśmy filtra uśredniającego. Ponownie pomocna okazała się funkcja `fspecial`.



Rysunki 50., 51. Obrazek po filtracji oraz jego widmo amplitudowe

Filtracja spowodowała rozmycie się obrazka.

Na sam koniec zajęć wyświetliliśmy widmo amplitudowe po transformacie cosinusowej.



Rysunek 52. Widmo amplitudowe po transformacie cosinusowej