PODSTAWY PRZETWARZANIA OBRAZÓW CYFROWYCH

Klasyczne metody przetwarzania obrazów

adam.piorkowski@agh.edu.pl

Przetwarzanie i analiza obrazów - podejście klasyczne

Przetwarzanie wstępne (preprocessing)

konwersja, skalowanie, normalizacja, odszumianie, …

Przetwarzanie (processing)

detekcja, binaryzacja, segmentacja, ekstrakcja krawędzi, …

Przetwarzanie cd (postprocessing)

operacje morfologiczne, selekcja obiektów, wypełnianie, indeksacja, …

Analiza

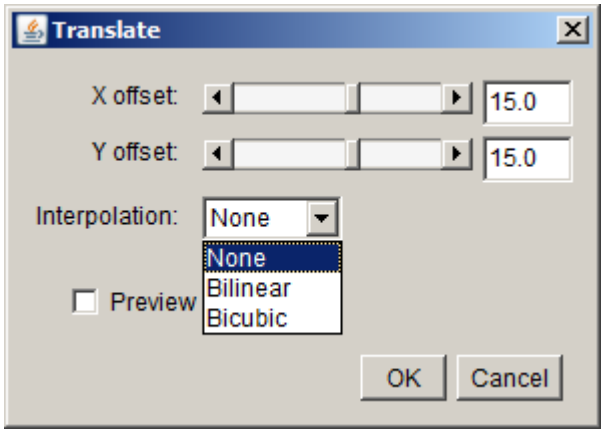
pomiary cech metrycznych, morfometria, a. ilościowo-jakościowa, …

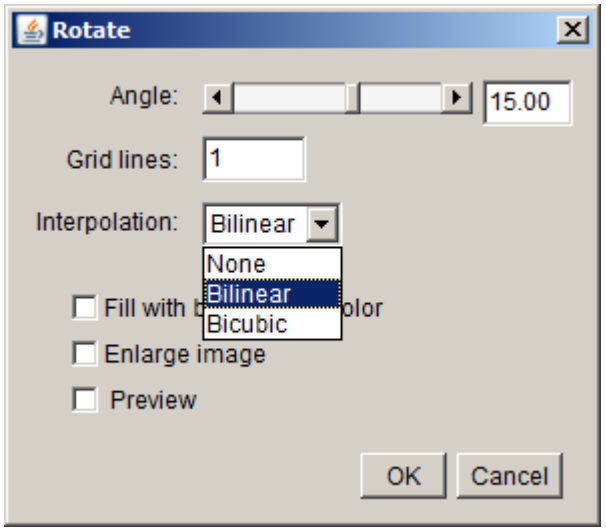
Klasyczne metody przetwarzania obrazów

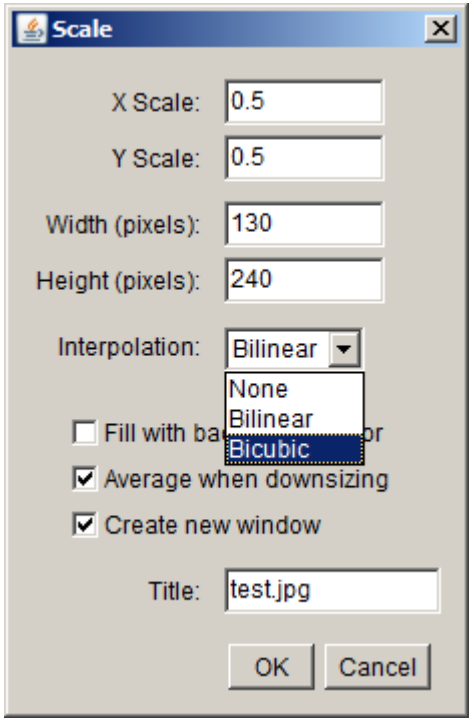
- przekształcenia geometryczne
- operacje punktowe
- operacje arytmetyczne na obrazach
- operacje kontekstowe
- operacje morfologiczne
- operacje widmowe
- operacje geodezyjne
- zaawansowane algorytmy
  - detekcji, segmentacji, wykrywania krawędzi

Przekształcenia geometryczne

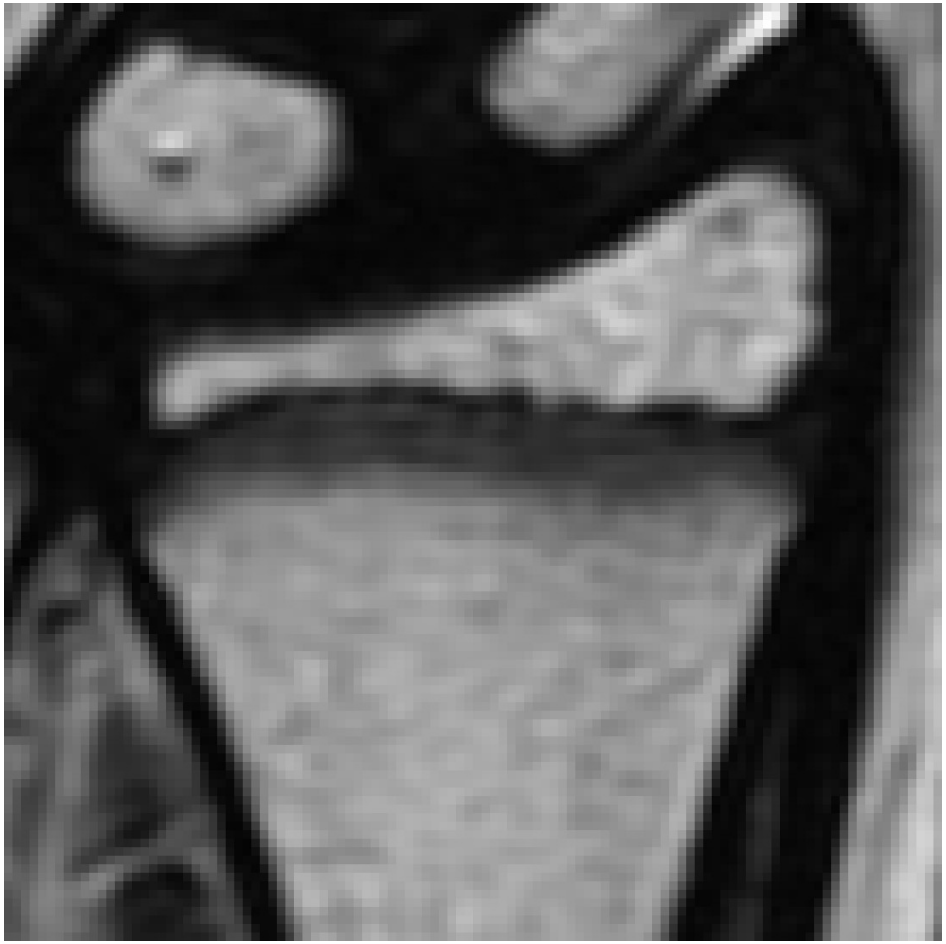
- translacja
- obrót
- skalowanie (powiększenie/pomniejszenie)
- inne transformacje







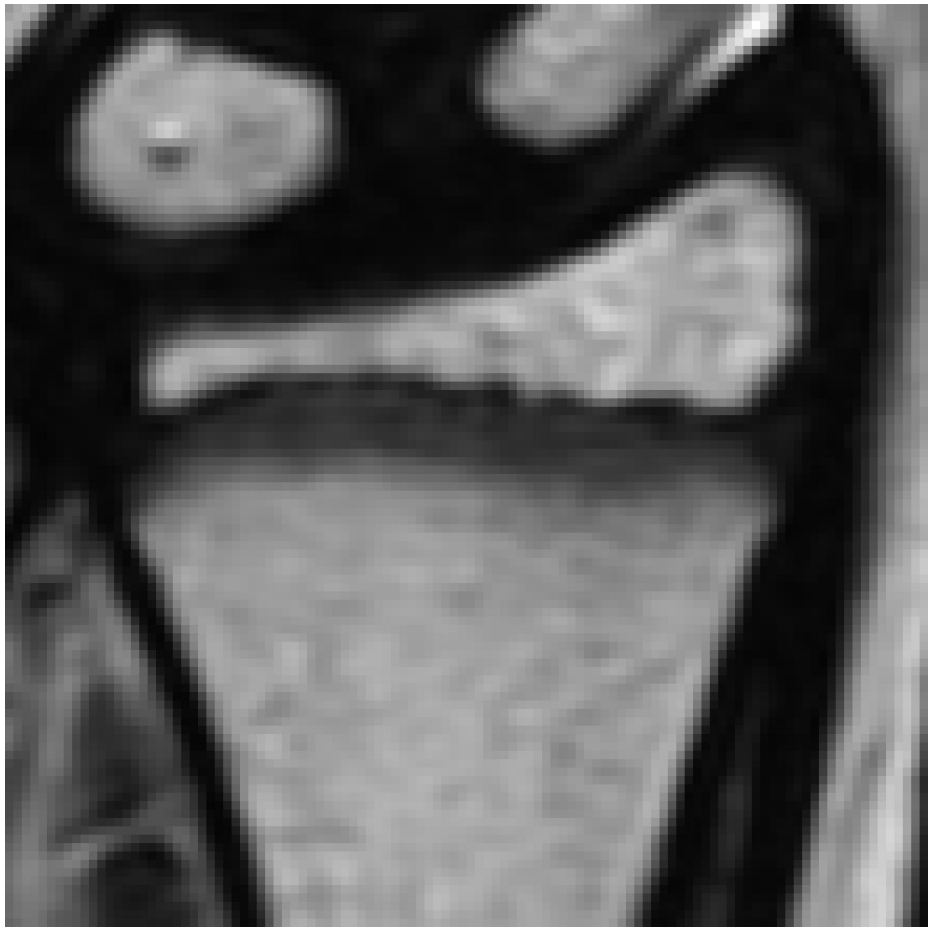
Przykład: obrazy sekwencji T1 i T2

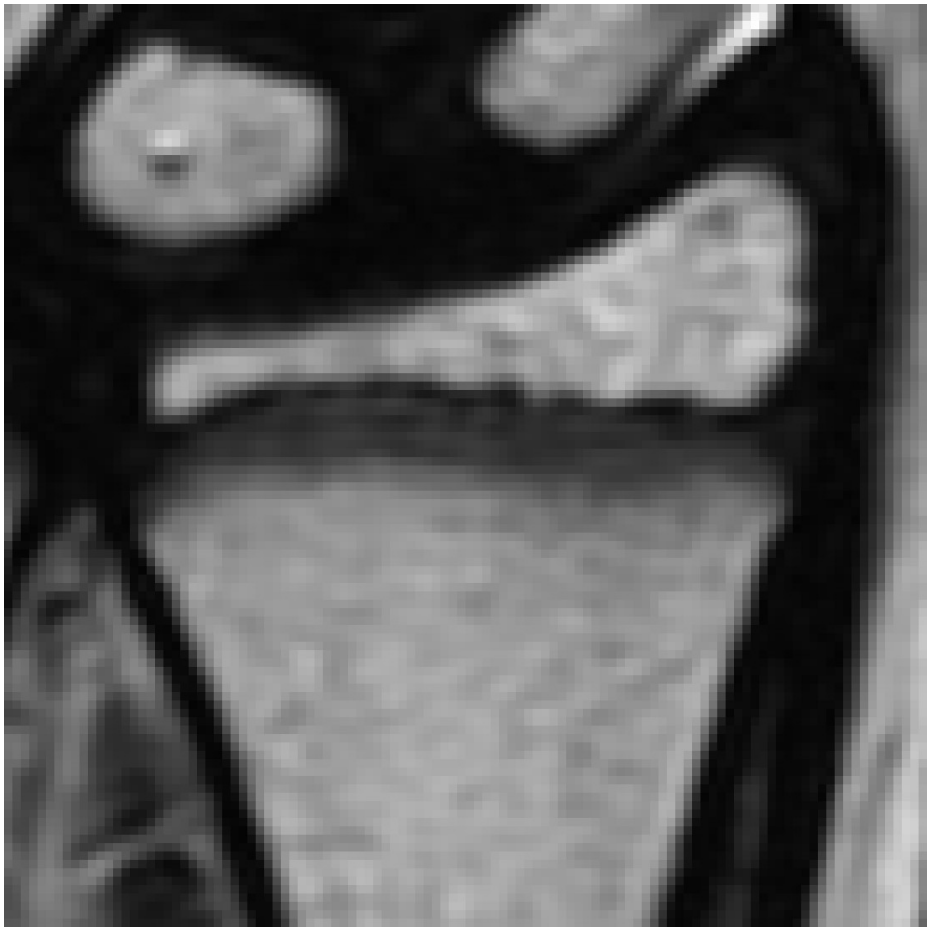


Przykład: obrazy sekwencji T1 i T2

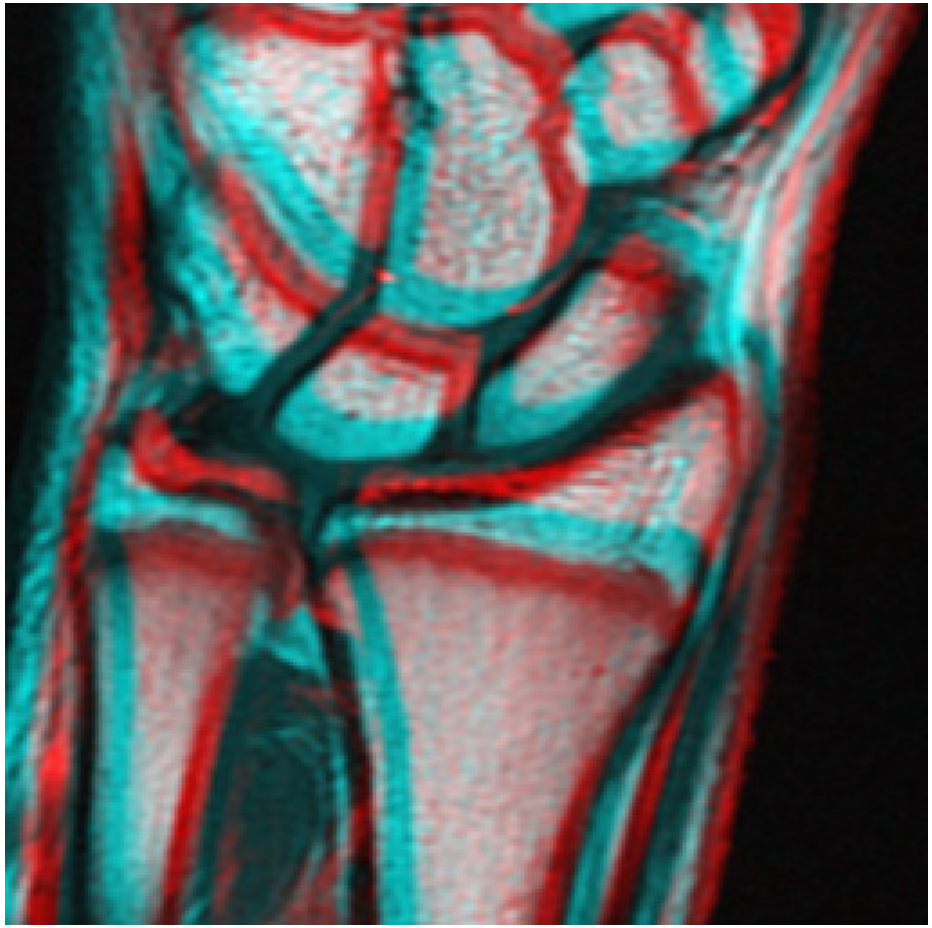
przed dopasowaniem **po dopasowaniu**

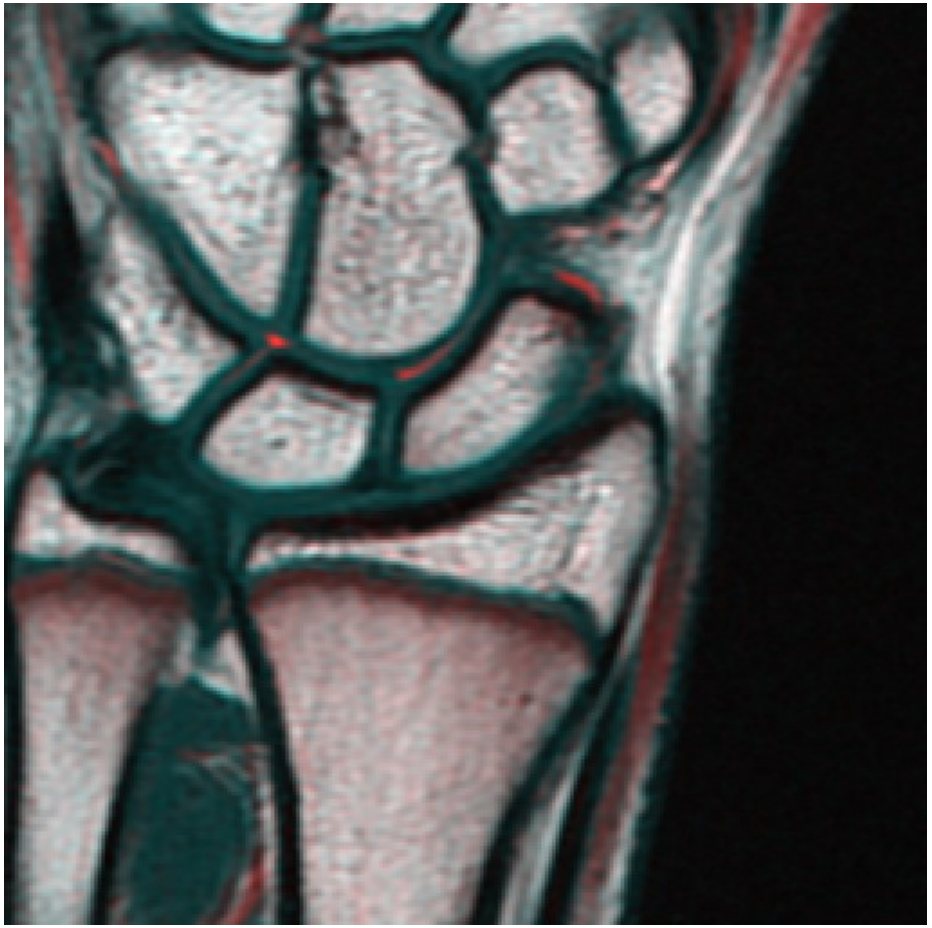
ang. co-registration



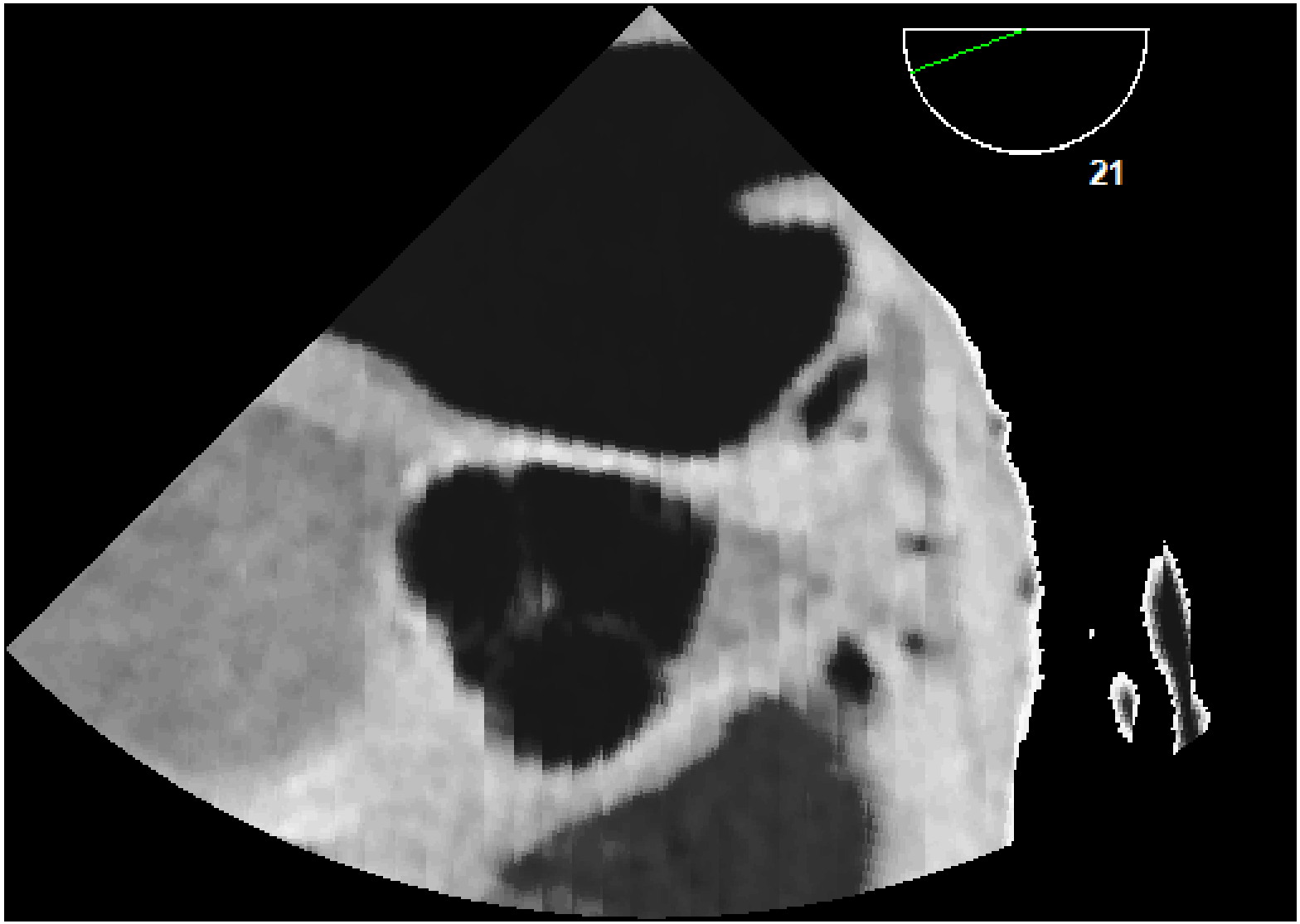


Przykład: obrazy sekwencji T1 i T2



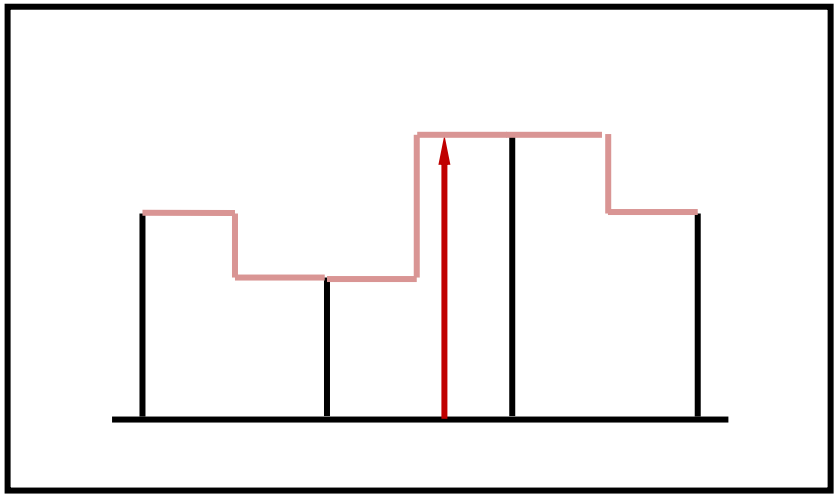


Interpolacja

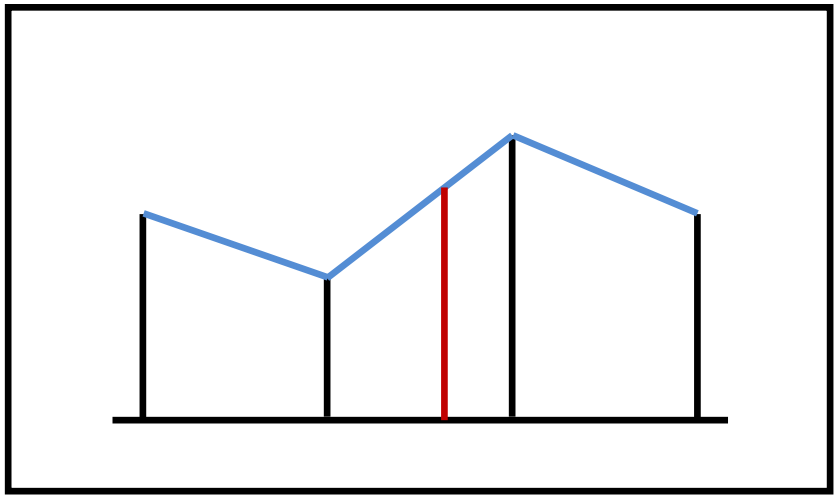


Interpolacja

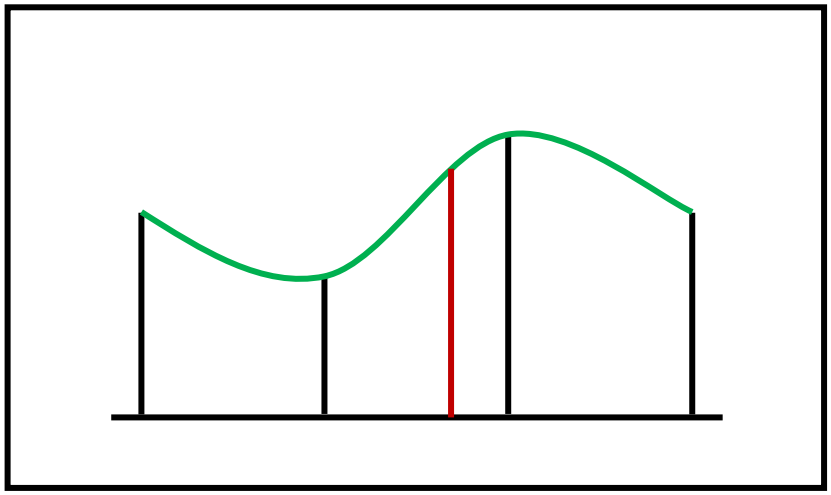
nearest
(none)



1D



linear

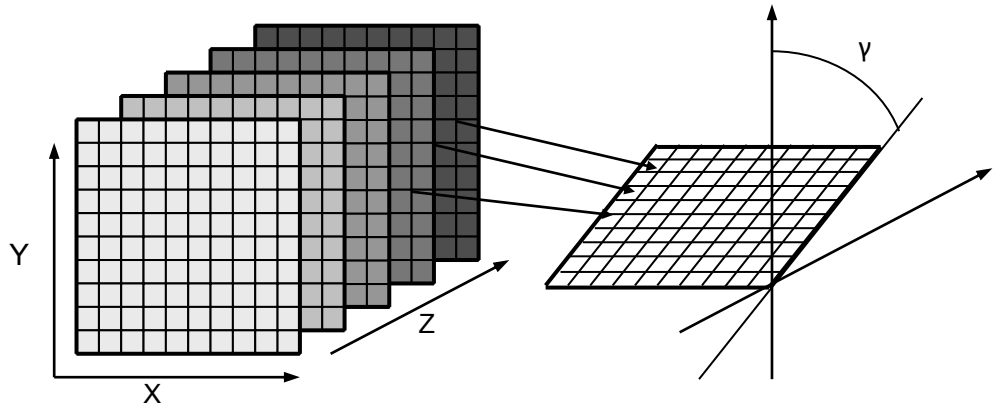


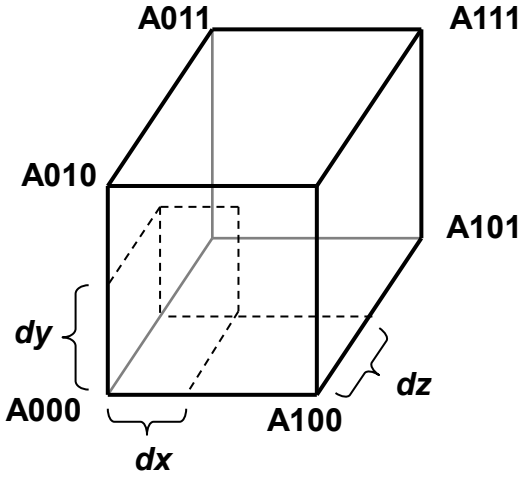
cubic

2D

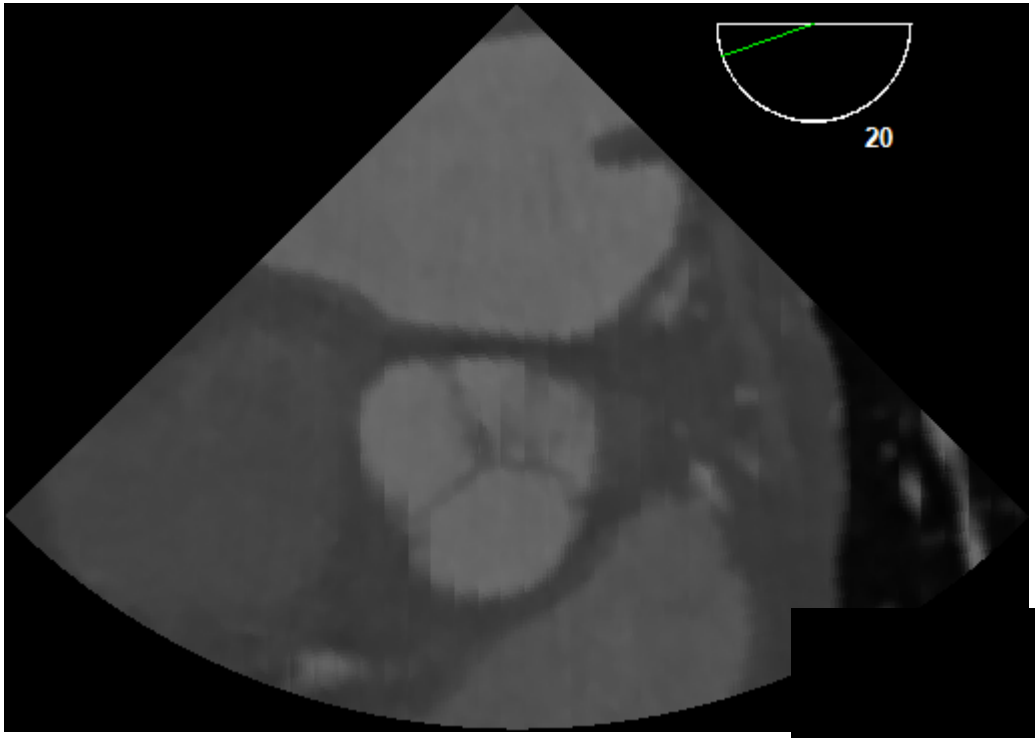
- interpolacja dwuliniowa (bilinear) – uśredniona wartość 4 najbliższych pikseli
- interpolacja dwusześcienna (bicubic) – wykorzystuje 8 sąsiadujących pikseli

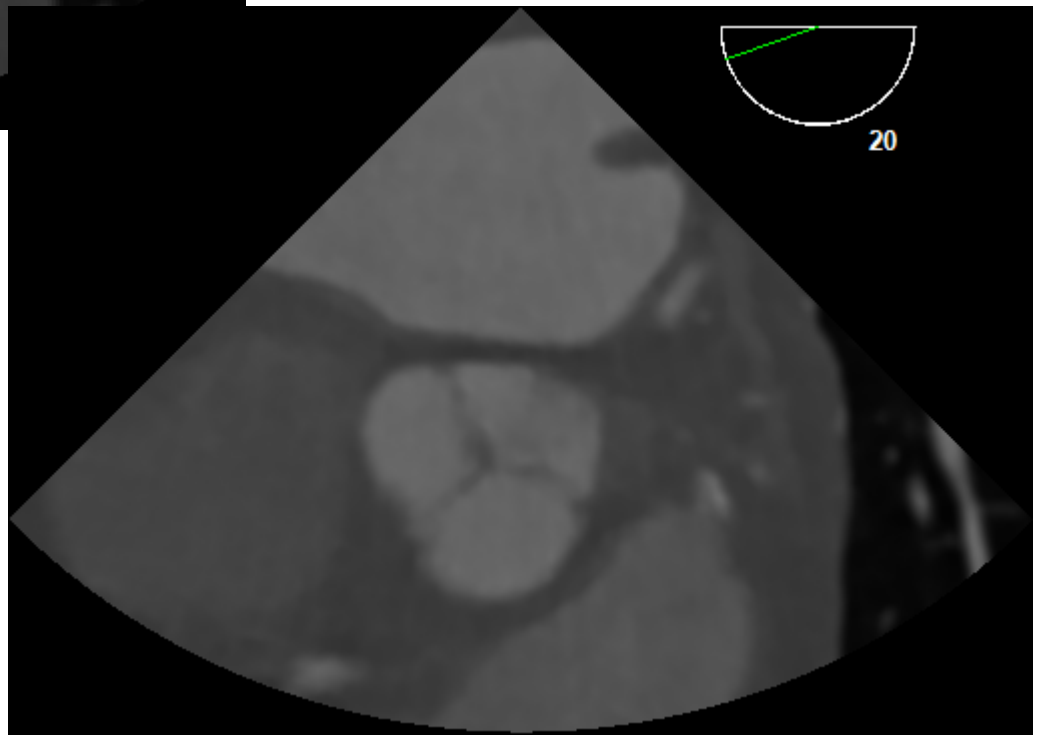
Interpolacja



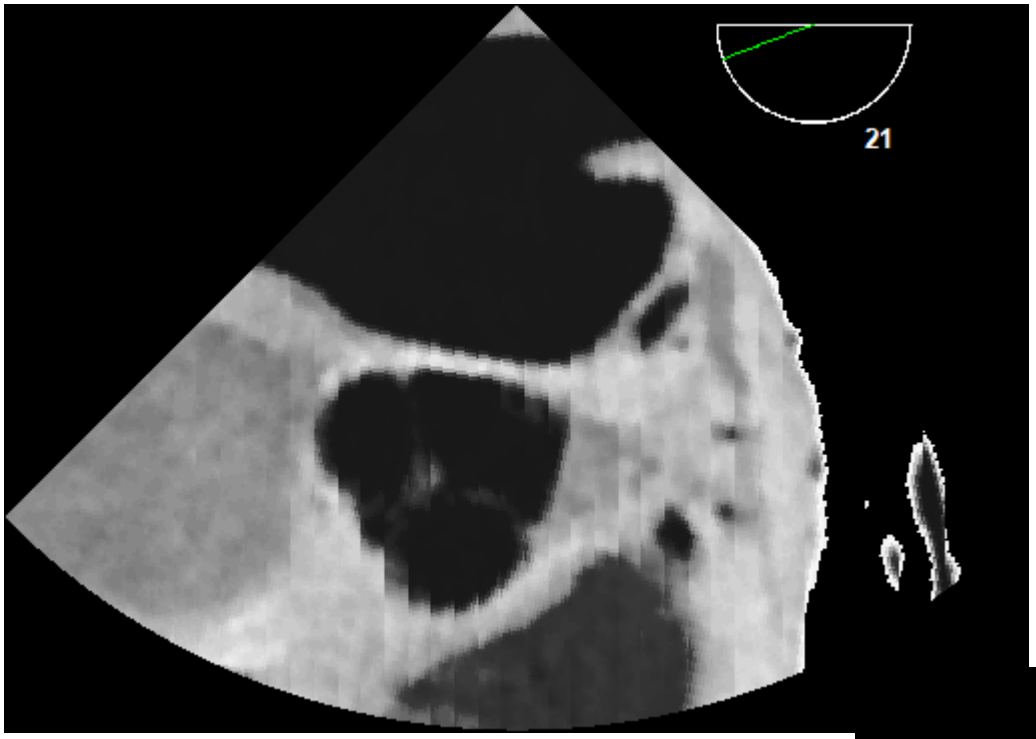


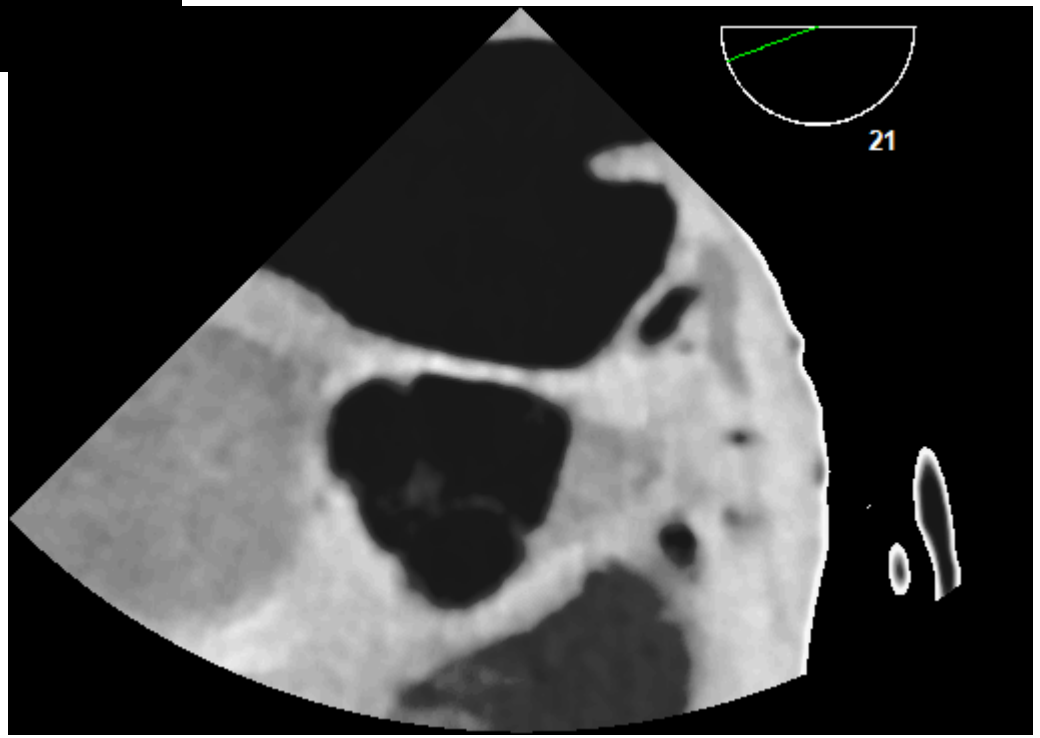
Interpolacja

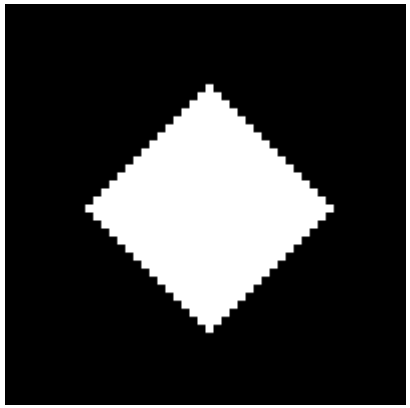




Interpolacja







none

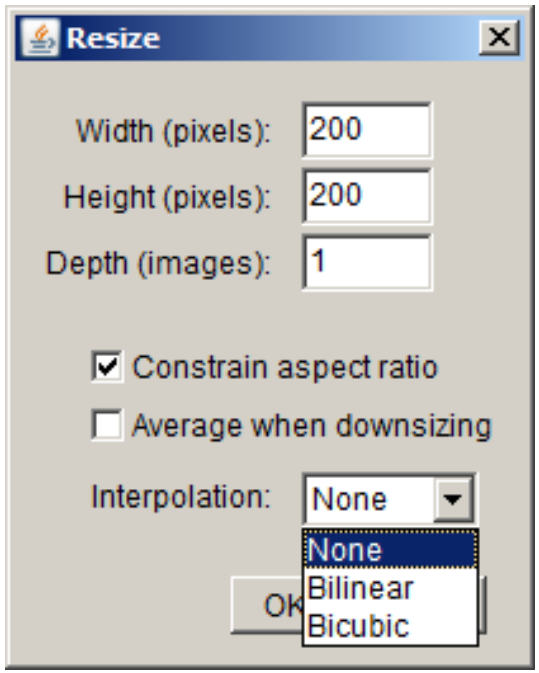
l.unikalnych k. 2

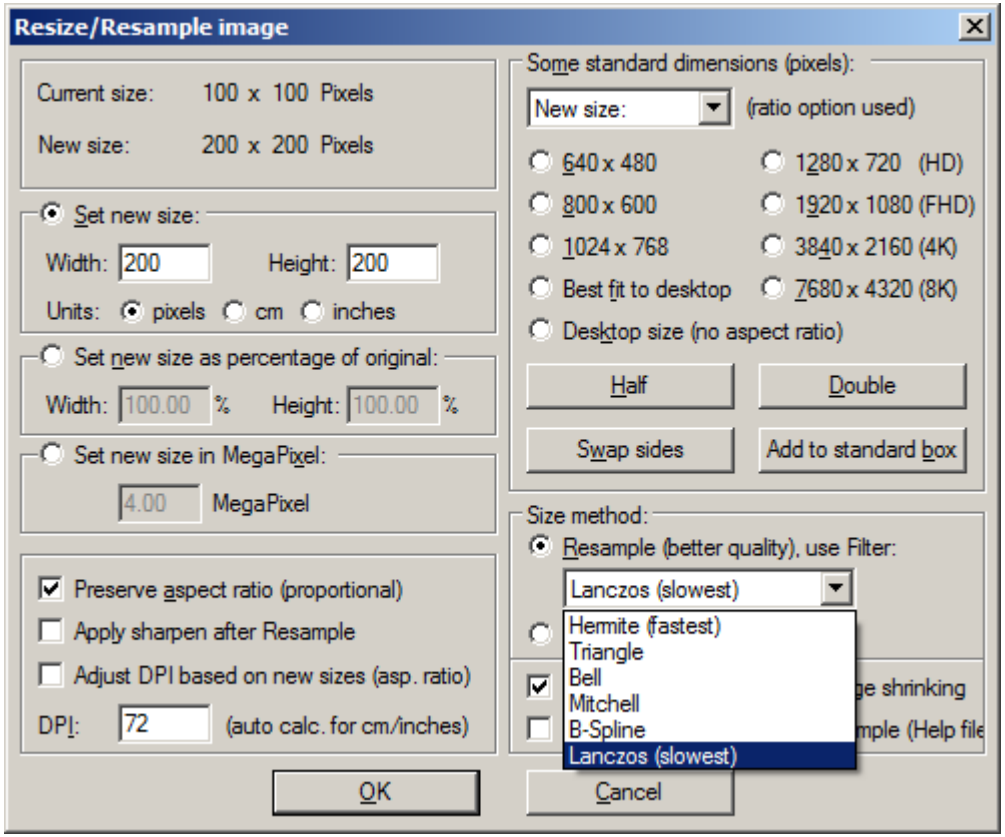
bilinear

l.unikalnych k. 15

bicubic

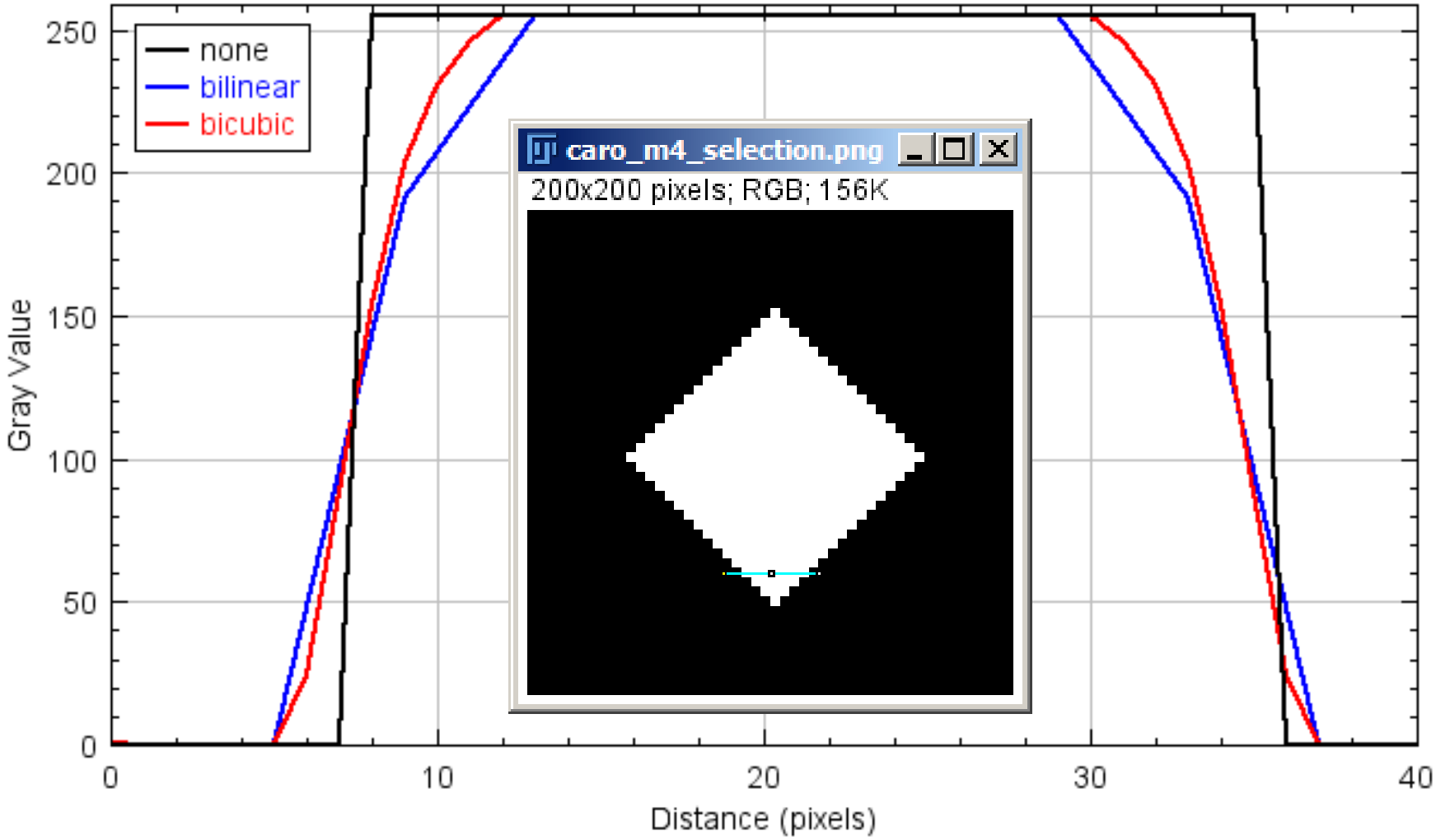
l.unikalnych k. 36



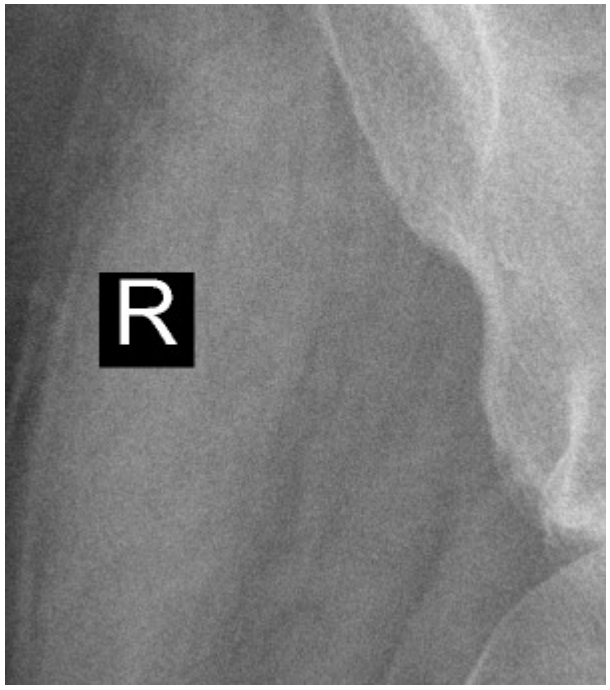


Interpolacja

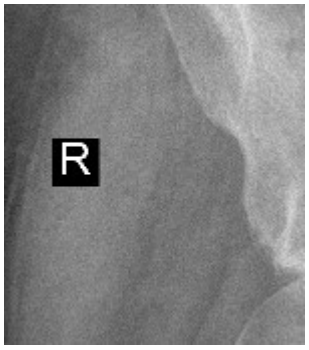
Interpolacja



Interpolacja - pomniejszanie

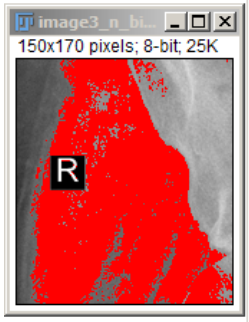


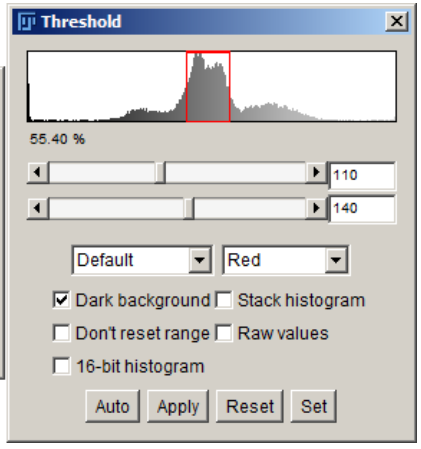
none

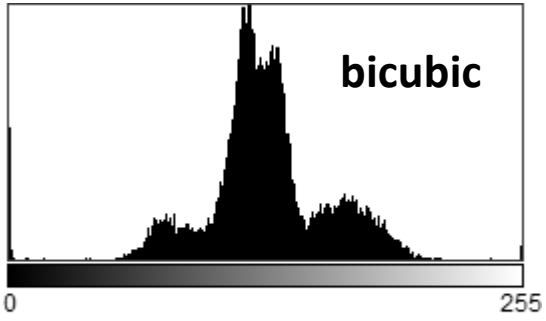


bilinear

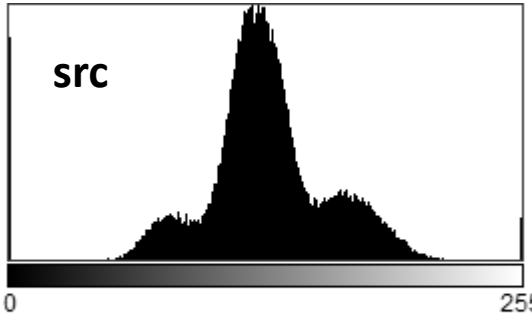
bicubic



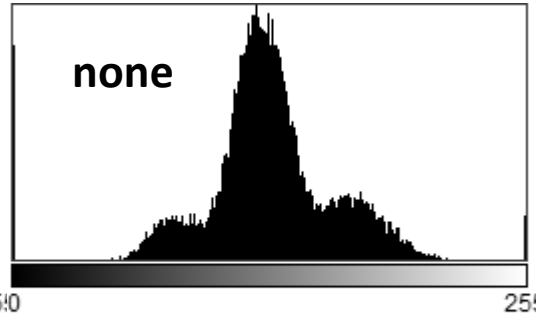




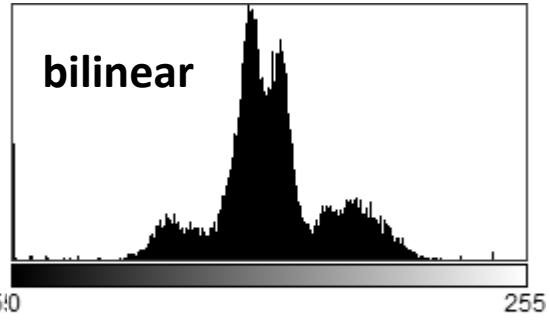
N: 25500
Mean: 127.183
StdDev: 32.809
Min: 0
Max: 255
Mode: 119 (595)



N: 102000
Mean: 127.166
StdDev: 33.798
Min: 0
Max: 255
Mode: 121 (2019)



N: 25500
Mean: 127.043
StdDev: 34.009
Min: 0
Max: 255
Mode: 121 (532)

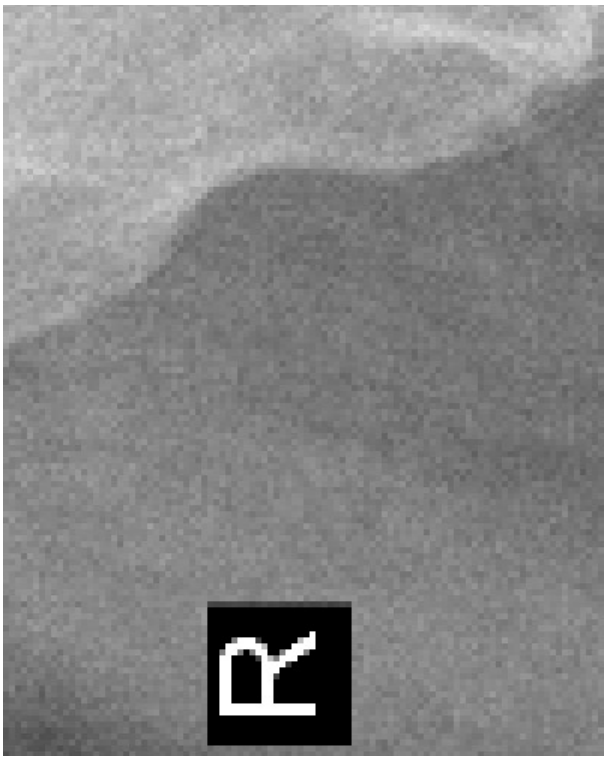


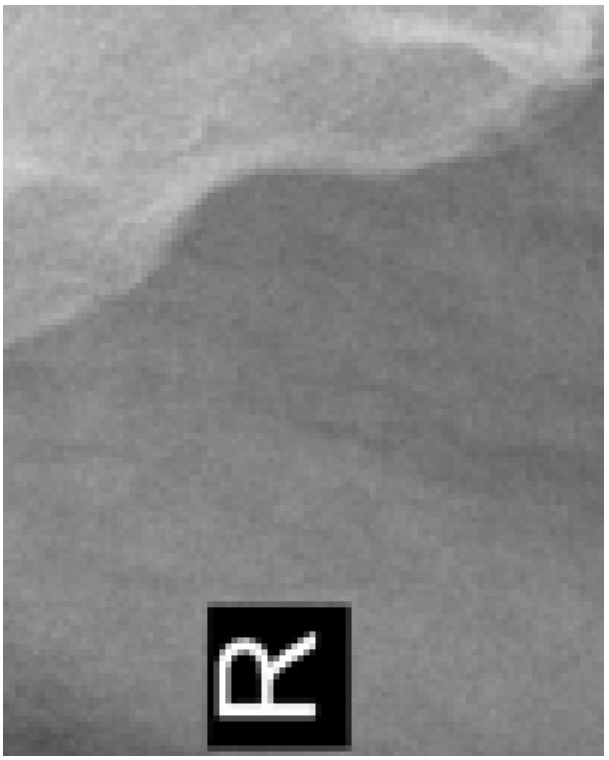
N: 25500
Mean: 127.169
StdDev: 32.365
Min: 0
Max: 255
Mode: 117 (631)

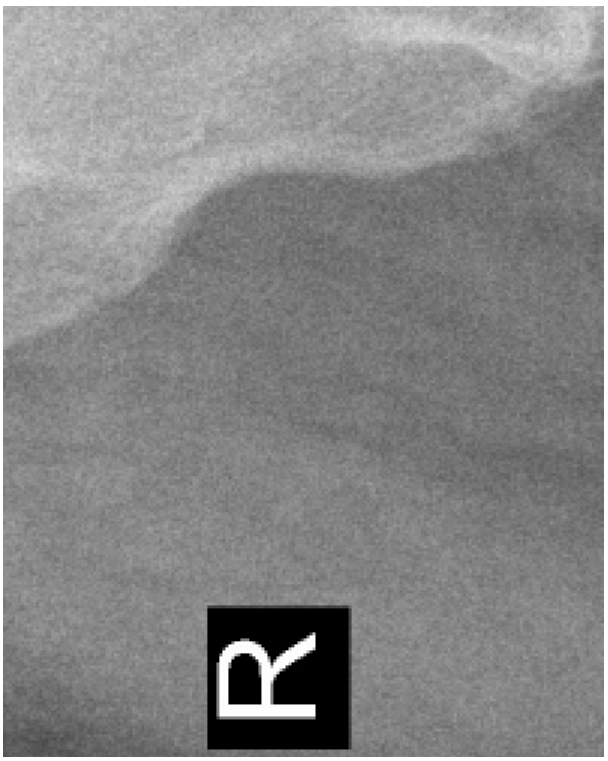
Interpolacja - pomniejszanie

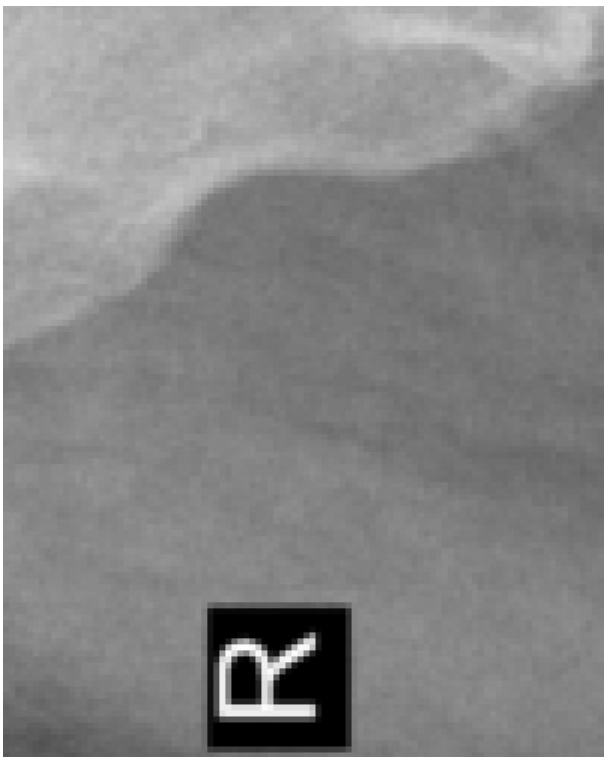
none, 2x

bicubic, 2x









SRC

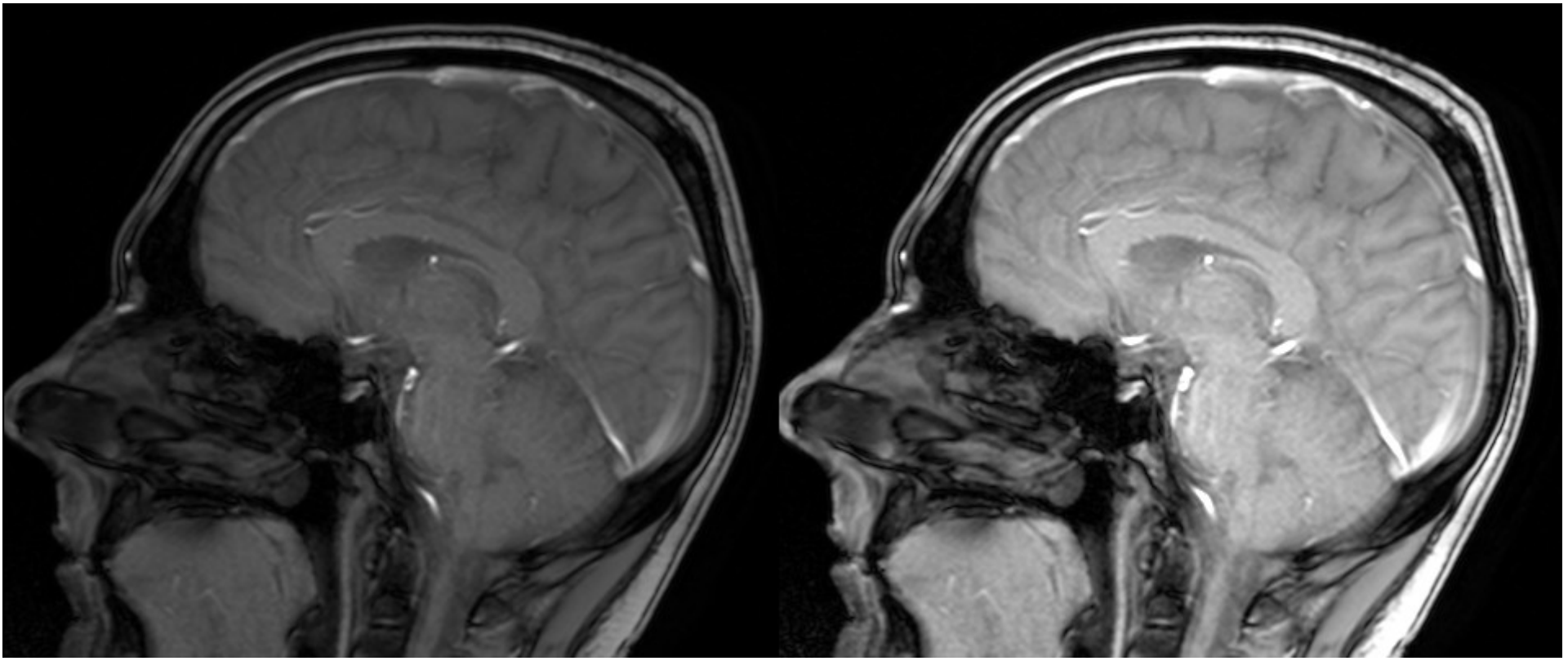
bilinear, 2x

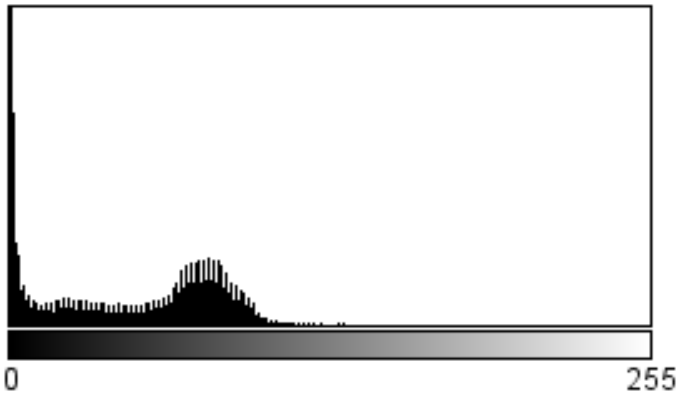
Proste przekształcenia punktowe

- arytmetyczne: dodanie, odjęcie, mnożenie, dzielenie (skalar)
- kombinacja dwóch różnych obrazów
- podnoszenie do potęgi/pierwiastkowanie
- logarytmowanie
- korekcja gamma
- LUT
- transformacje odcinkowo liniowe
- jasność i kontrast
- transformacje nieliniowe
- histogram i jego wyrównywanie

Przekroczenie zakresu po operacji:

- nasycenie wartości powyżej 255 i poniżej 0 (dla 8-bit)
- skalowanie do zakresu 0-255 (dla 8-bit)

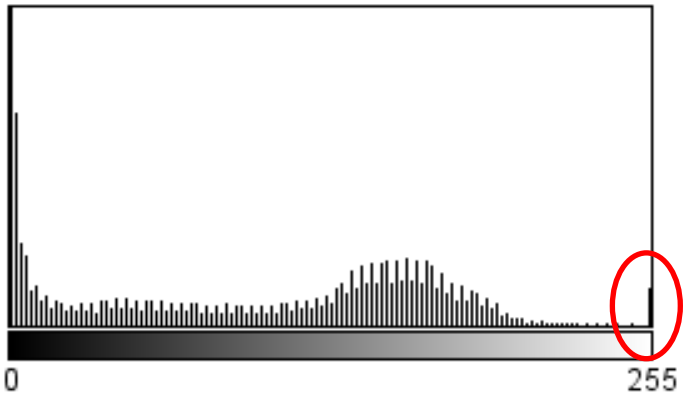




N: 192000
Mean: 45.004
StdDev: 36.960
Value: 142

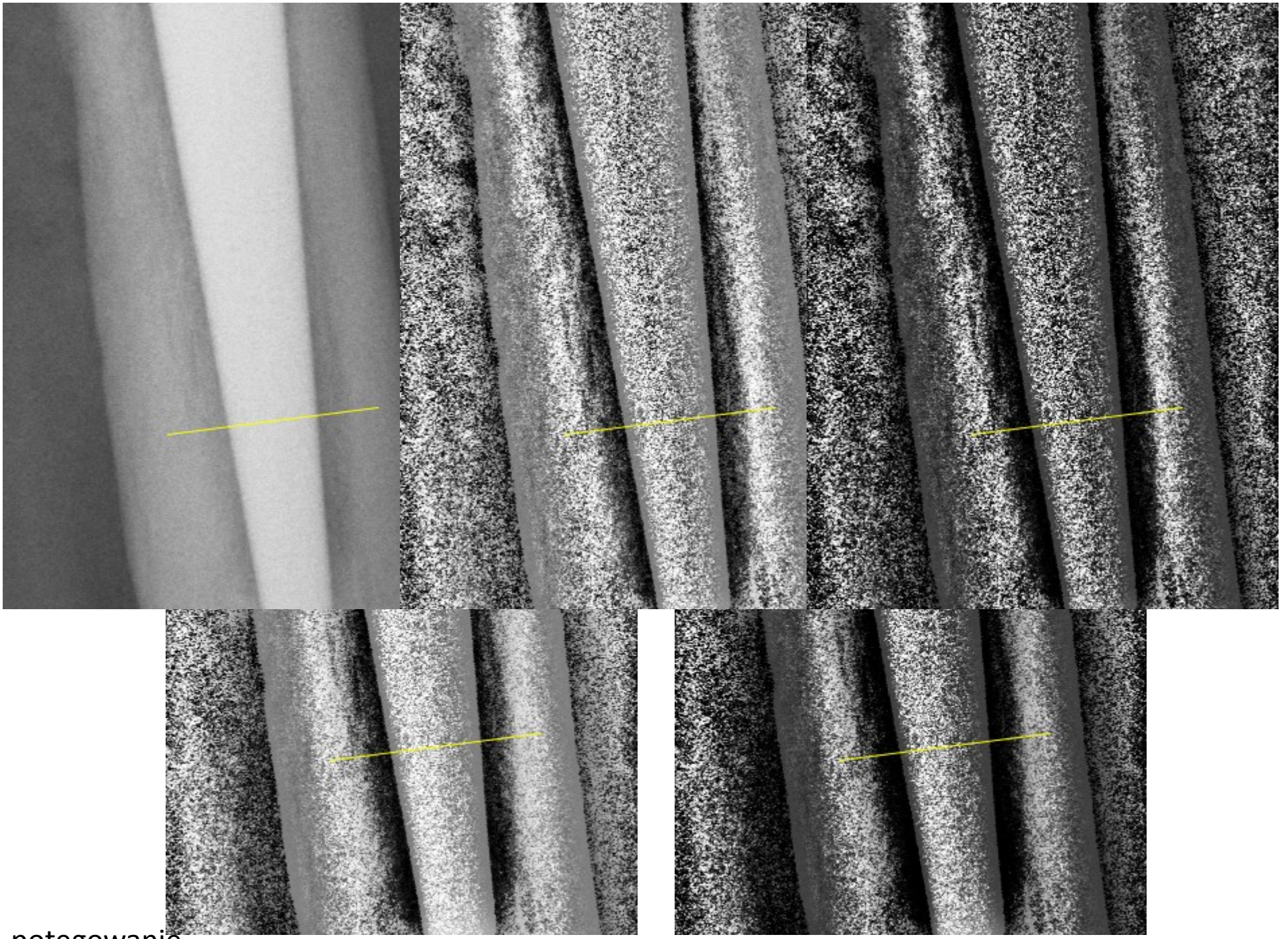
Min: 0
Max: 255
Mode: 0 (32954)
Count: 50

mnożenie x2

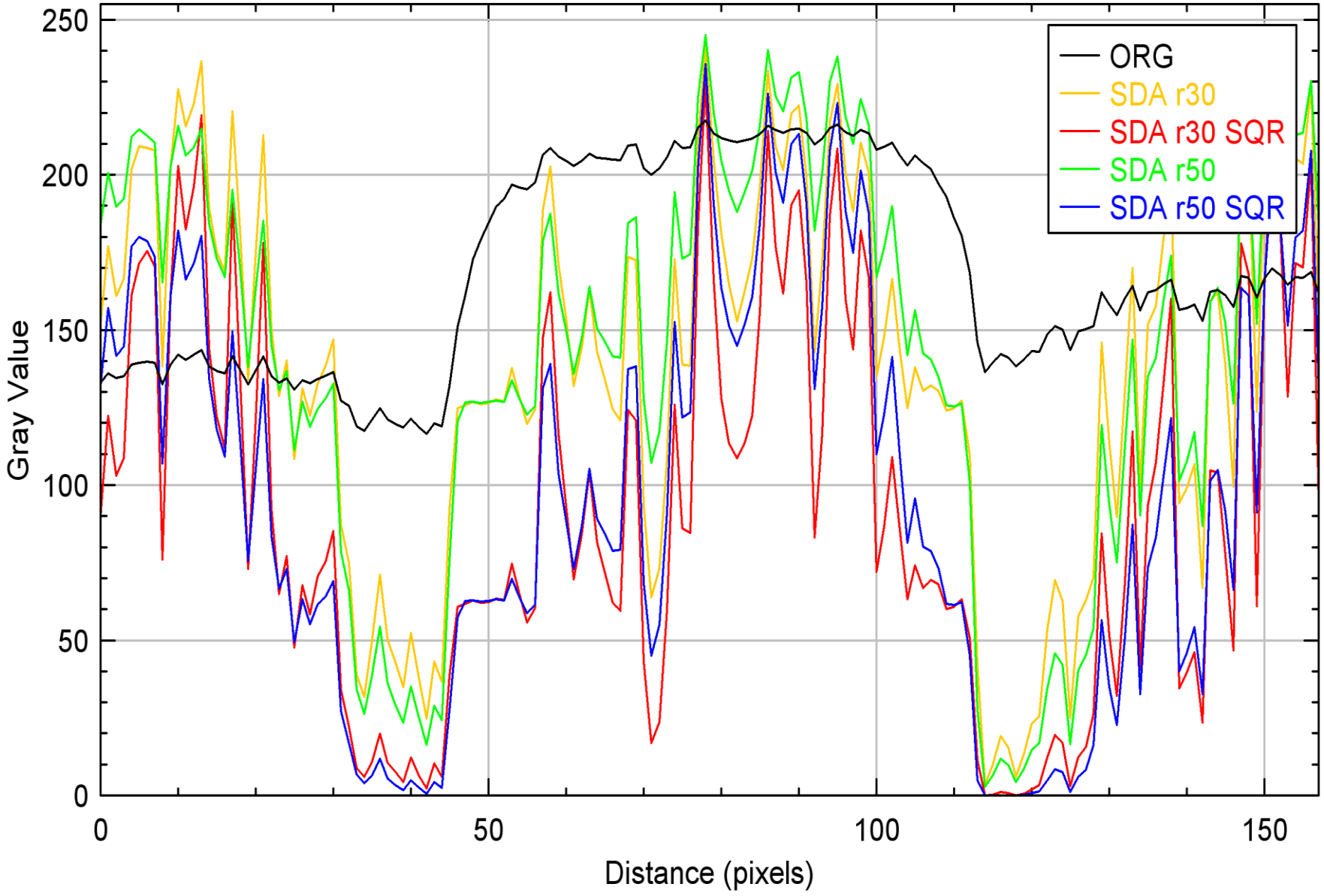


N: 192000
Mean: 89.716
StdDev: 73.110
Value: ---

Min: 0
Max: 255
Mode: 0 (32954)
Count: ---



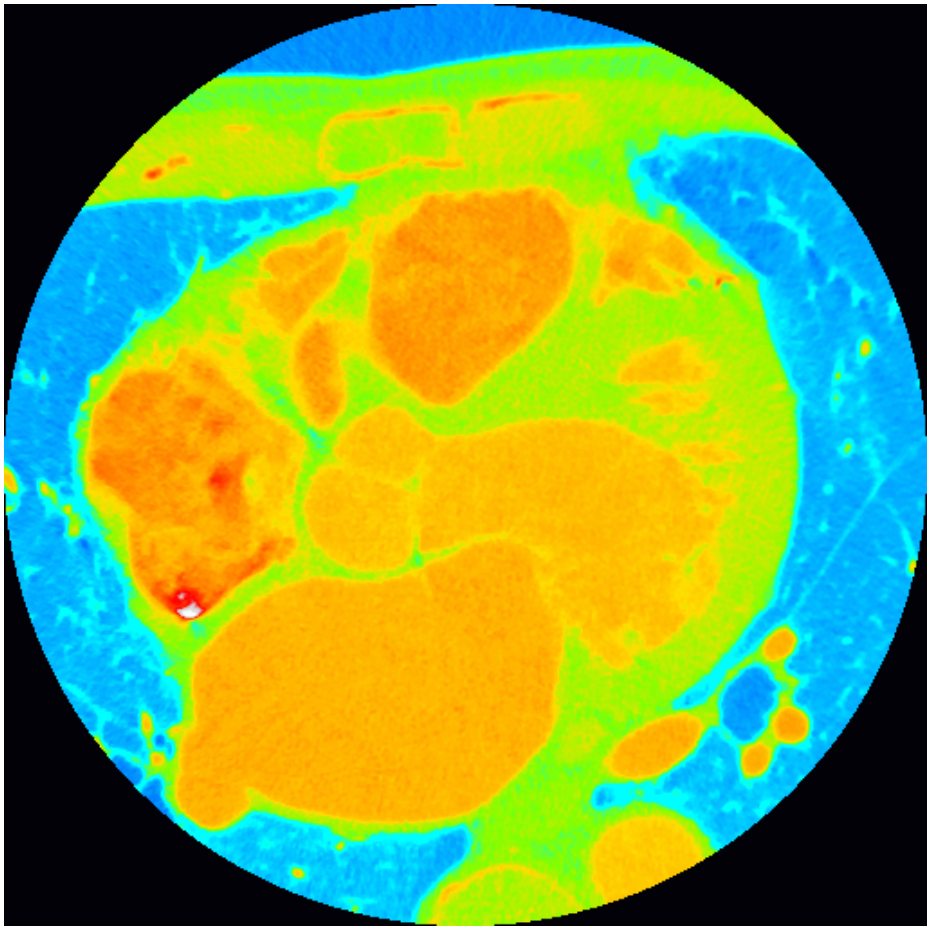
potęgowanie

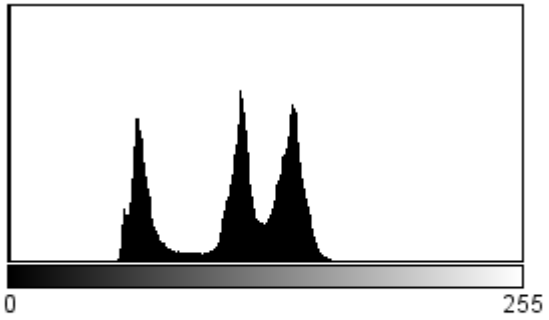

ORG
SDA r30
SDA r30 SQR
SDA r50
SDA r50 SQR
Gray Value
Distance (pixels)
0
50
100
150
200
250
0
50
100
150

Histogram

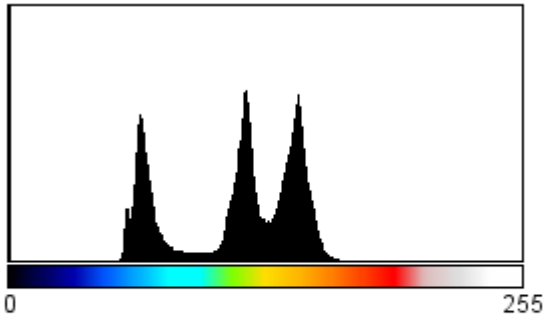
- histogram to informacja statystyczna prezentująca rozkład liczby pikseli o danej jasności, należącej do zakresu głębi obrazu
- histogram jest tabelą, wygodną formą prezentacji jest wykres (skala liniowa lub logarytmiczna)
- dla obrazów składających się z więcej niż jednego kanału (np. barwnych) histogramy można wyznaczać osobno dla każdego kanału lub dla kombinacji kanałów

$$H(i) = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \begin{cases} 1 & p(x,y) = i \\ 0 & p.p. \end{cases}$$



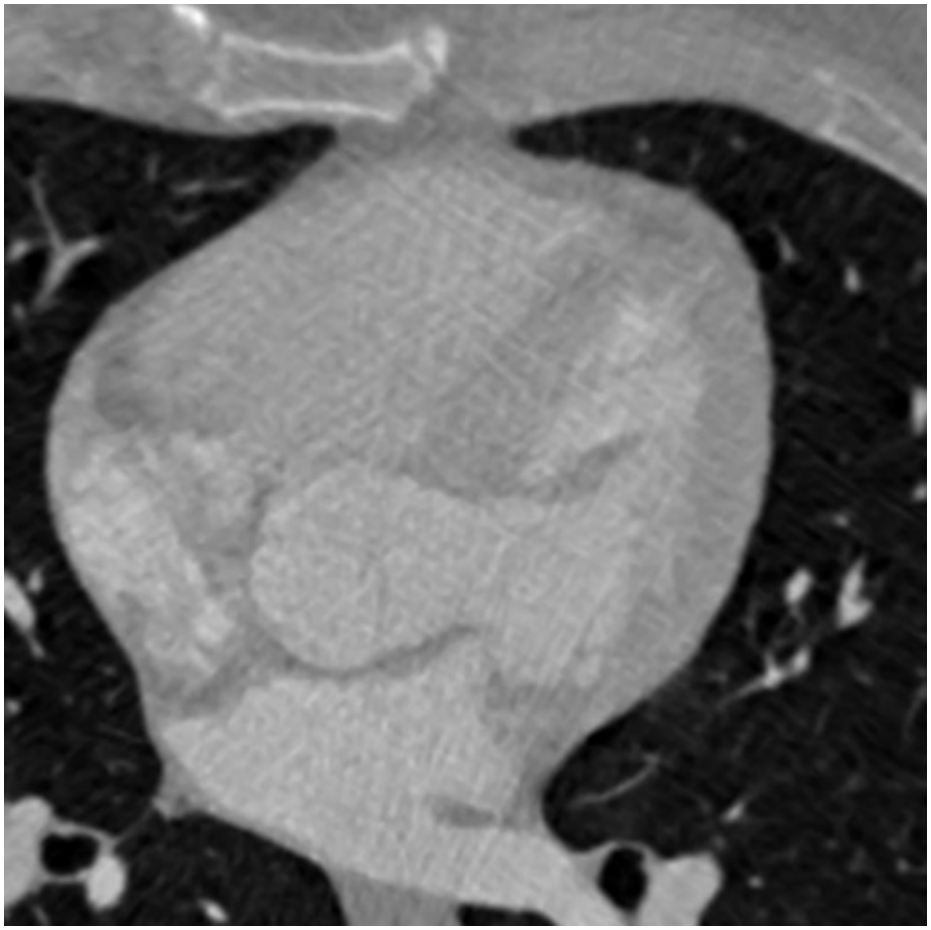


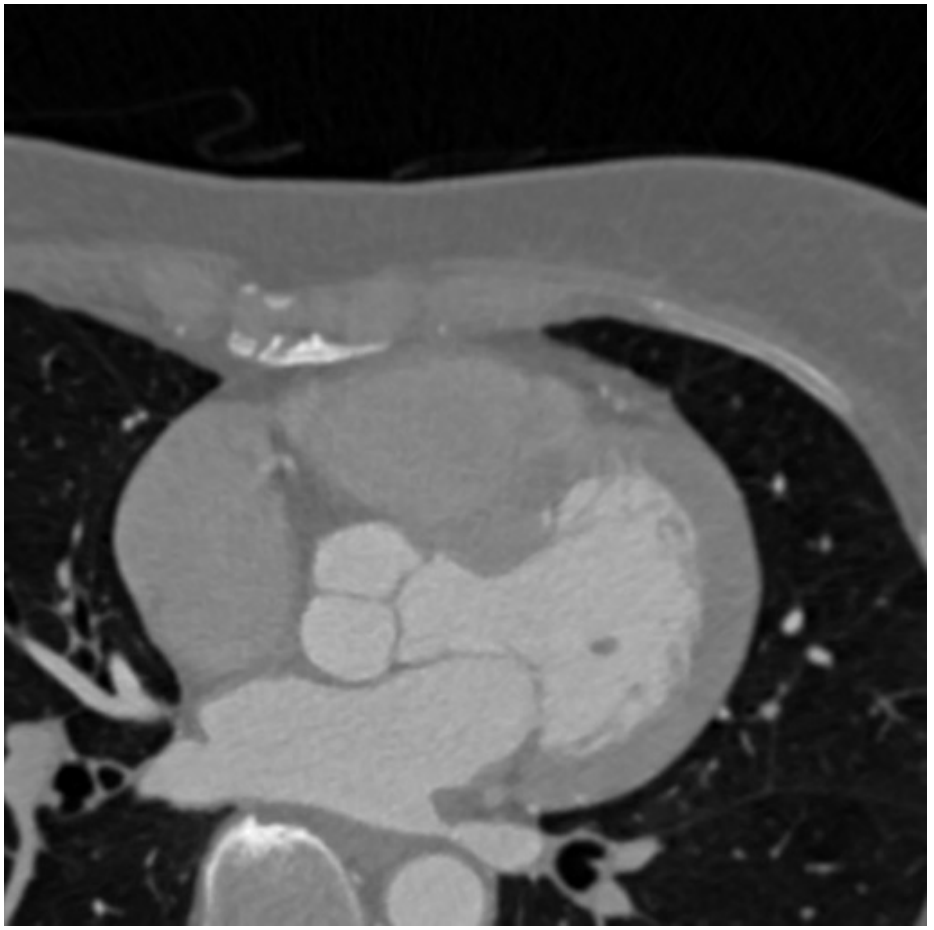
N: 262144
Mean: 87.135
StdDev: 53.055
Min: 0
Max: 252
Mode: 0 (56284)

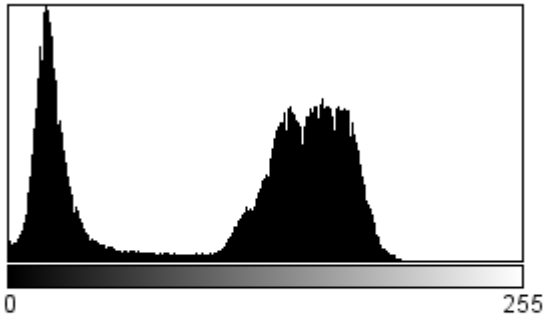


N: 262144
Mean: 88.751
StdDev: 53.961
Value: 132
Min: 0
Max: 255
Mode: 0 (56252)
Count: 1567

LUT: royal

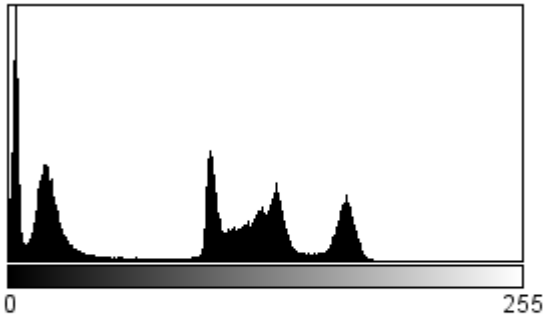






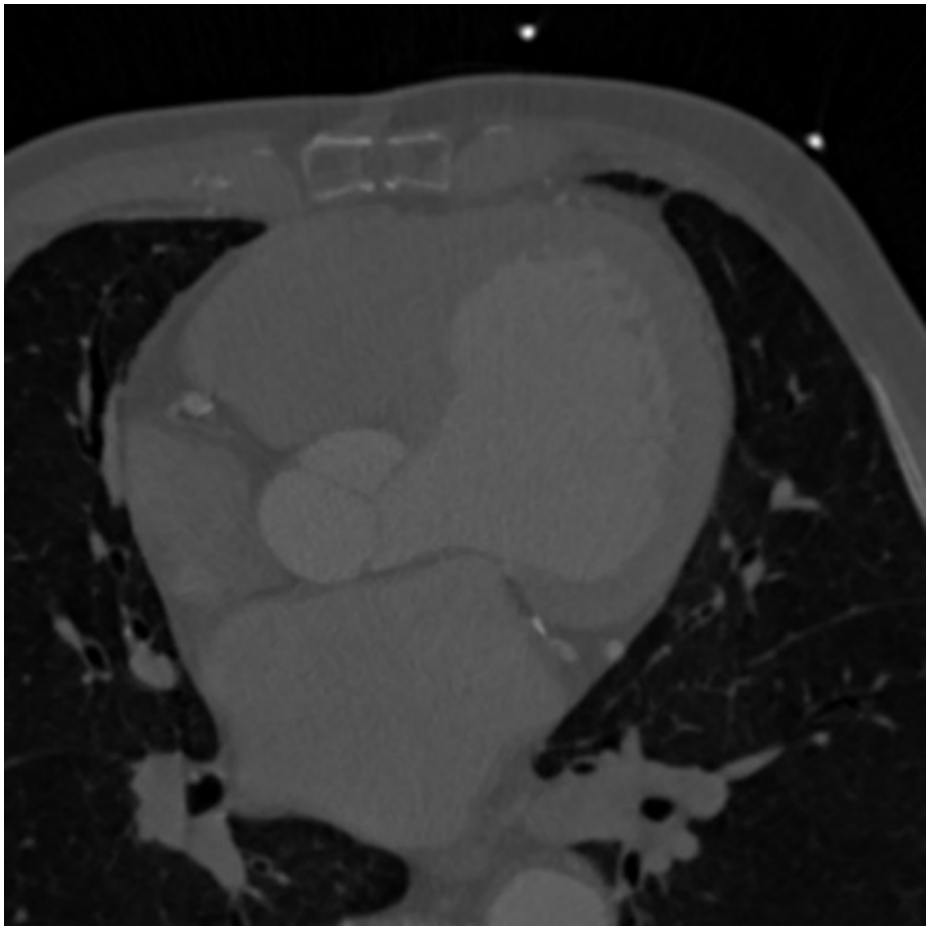
N: 262144
Mean: 104.288
StdDev: 62.546
Value: ---

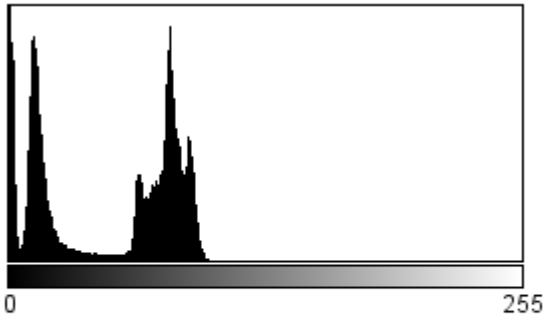
Min: 0
Max: 255
Mode: 18 (5135)
Count: ---



N: 262144
Mean: 80.511
StdDev: 60.611
Value: ---

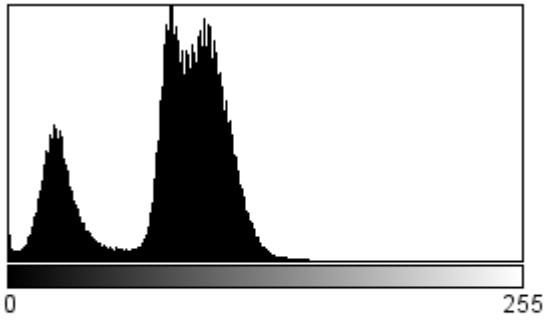
Min: 0
Max: 255
Mode: 3 (12092)
Count: ---





N: 262144
Mean: 48.450
StdDev: 34.435
Value: ---

Min: 0
Max: 253
Mode: 0 (10218)
Count: ---



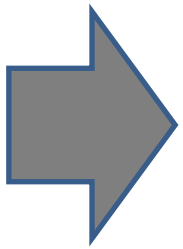
N: 262144
Mean: 77.472
StdDev: 32.357
Value: ---

Min: 0
Max: 253
Mode: 80 (5597)
Count: ---

LUT

- Look-Up Table

255	200
254	200
…	…
…	…
…	…
128	128
127	100
…	…
…	…
…	…
1	10
0	10



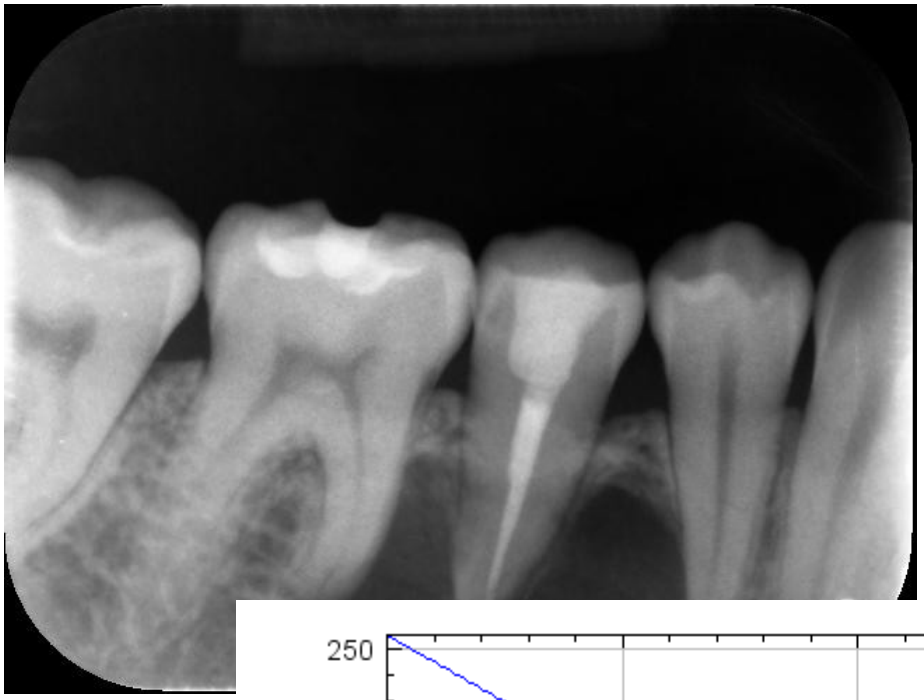
modyfikacja zawartości obrazu

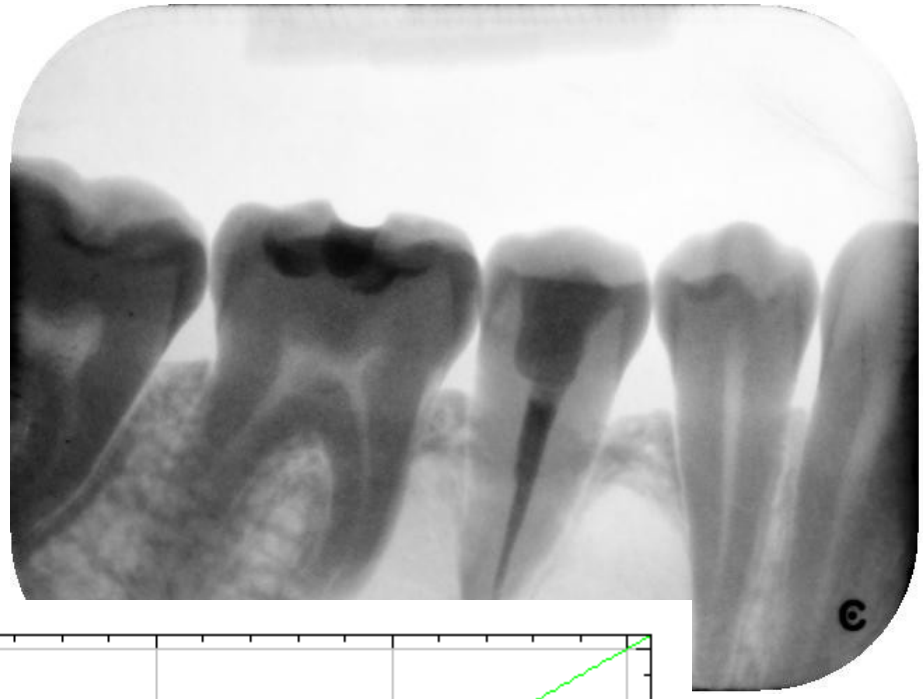
LUT do wyświetlania tylko!

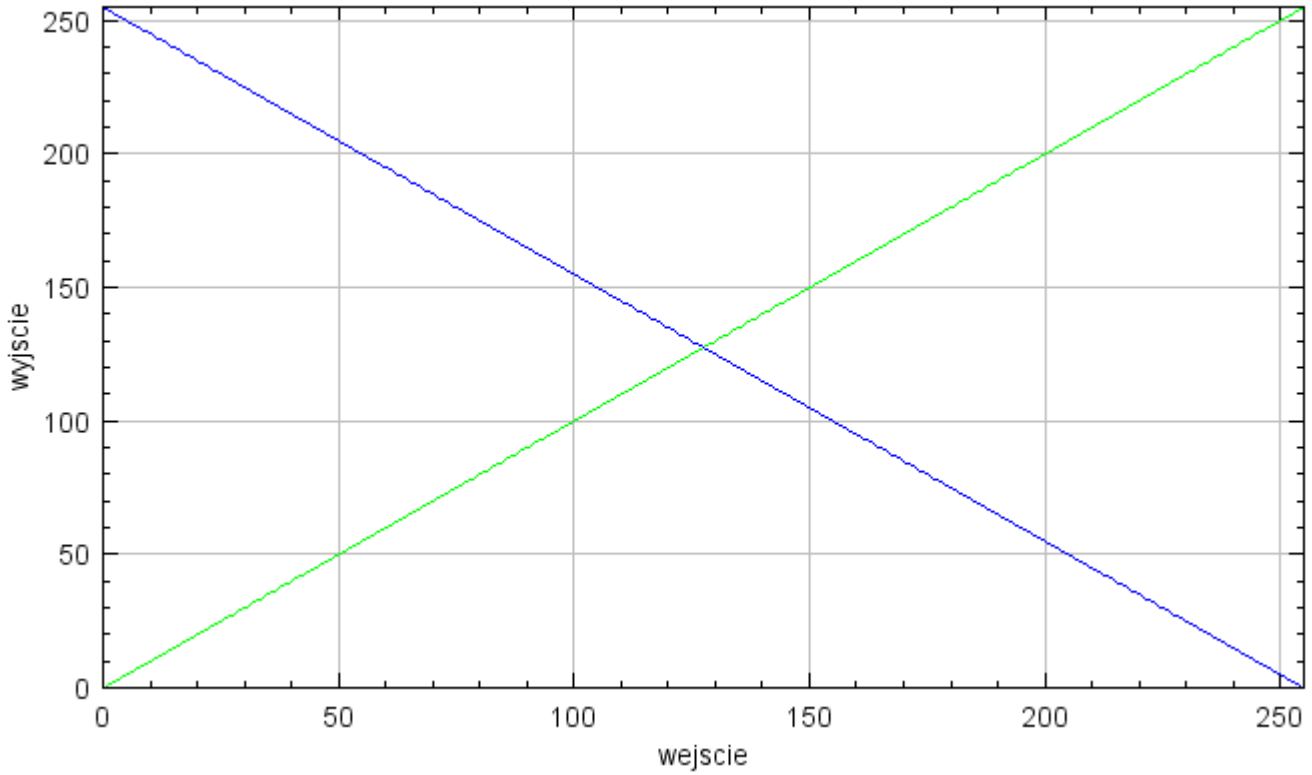
	R	G	B
255	200	255	200
254	200	255	200
…	…	…	…
…	…	…	…
…	…	…	…
128	128	255	128
127	100	255	100
…	…	…	…
…	…	…	…
…	…	…	…
1	10	255	10
0	10	255	10

ImageJ używa dodatkowych LUT podczas wyświetlania

```
run("8-bit");    // na wszelki wypadek wymuszenie 8-bit (0-255)

w = getWidth();
h = getHeight();

tablica_LUT = newArray(256);     // ”wlasna” tablica dla 8-bit

Plot.create("Przyklad LUT", "wejscie", "wyjscie");

// tablica tozsamosciowa
for(i=0; i < 256; i++)
        tablica_LUT[i] = i;

Plot.setColor("Green");
Plot.add("line", tablica_LUT);

// tablica odwracajajaca (Invert LUT)
// (poprzednia zostala nadpisana)
for(i=0; i < 256; i++)
        tablica_LUT[i] = 255 - i;

Plot.setColor("Blue");
Plot.add("line", tablica_LUT);

// (poprzednia zostala nadpisana)
for(x=0;x<w;x++)
        for(y=0;y<h;y++)
                setPixel(x, y, tablica_LUT[ getPixel(x, y) ] );
```

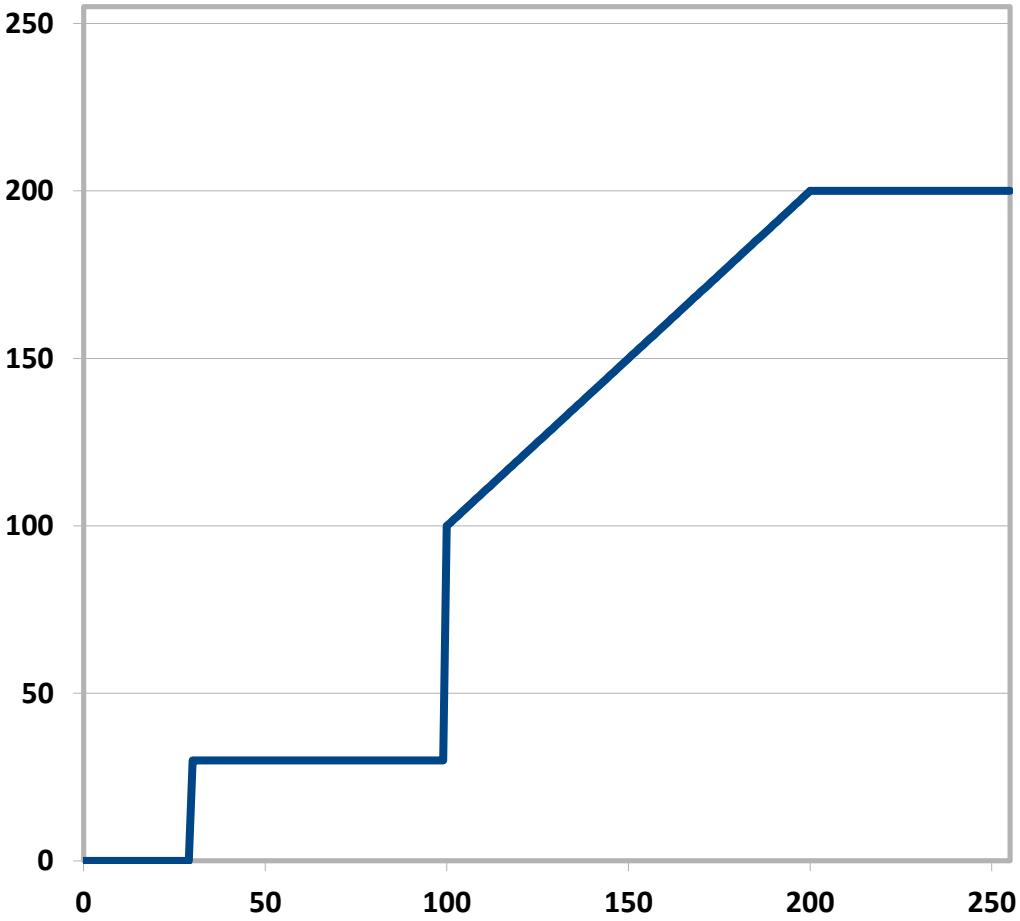


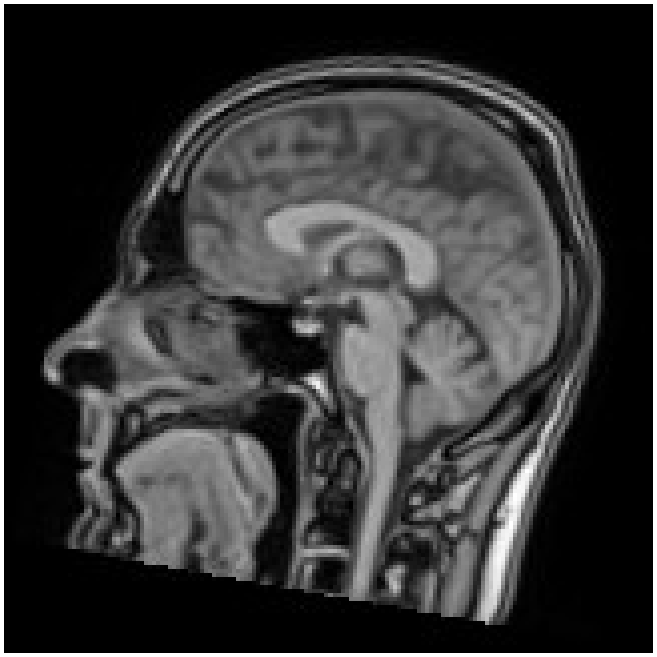


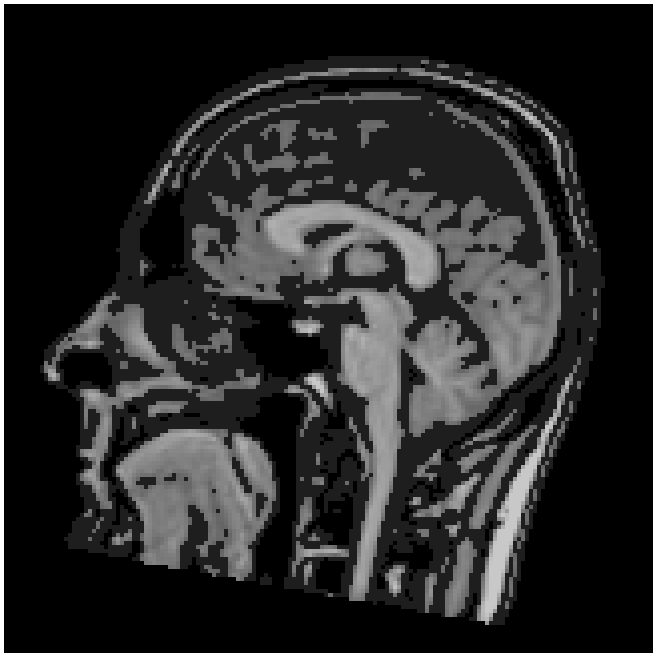

250
200
150
100
50
0
wyjscie
0
50
100
150
200
250
wejscie

LUT

transformacja odcinkowo liniowa

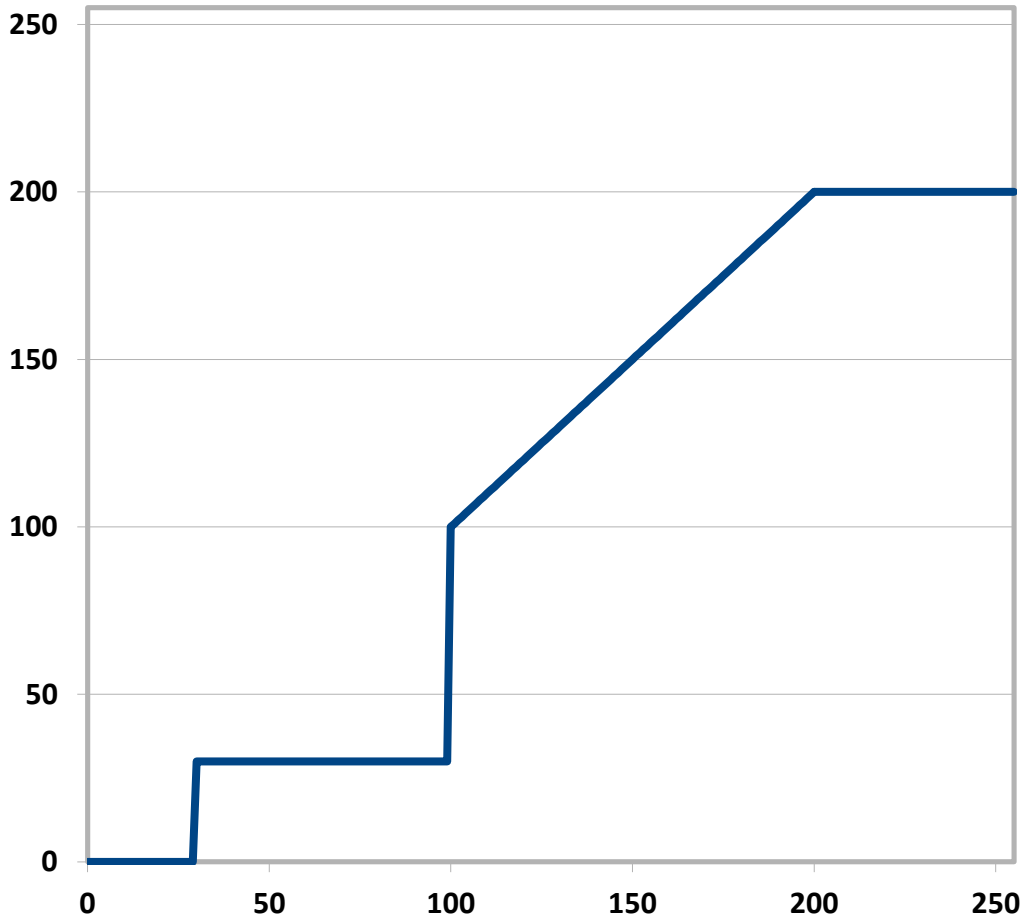


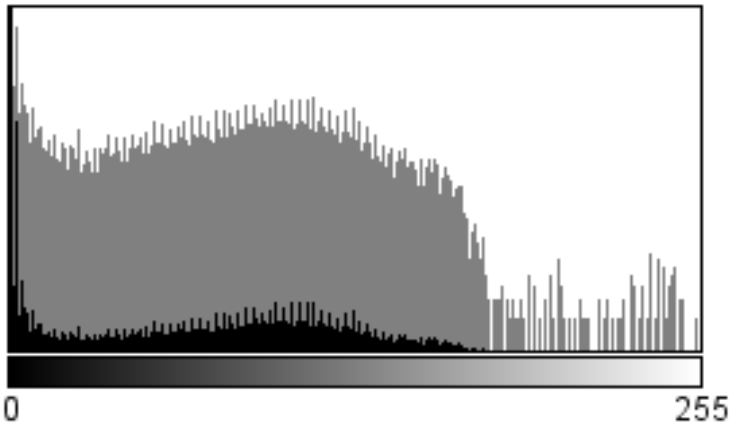




LUT

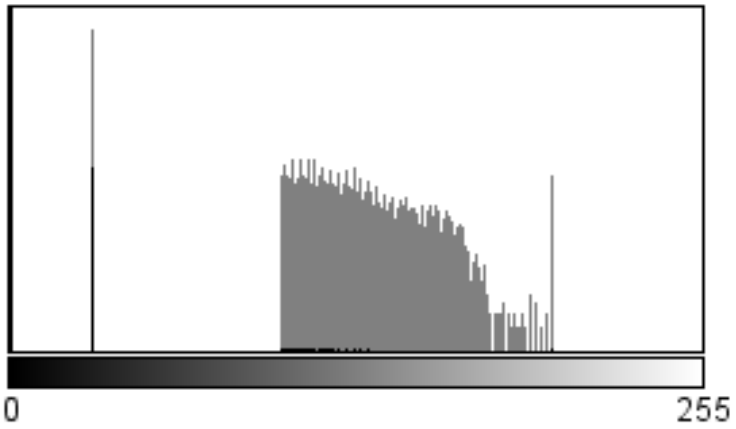
transformacja odcinkowo liniowa





N: 26244
Mean: 50.105
StdDev: 53.995
Value: 70

Min: 0
Max: 254
Mode: 0 (9361)
Count: 137

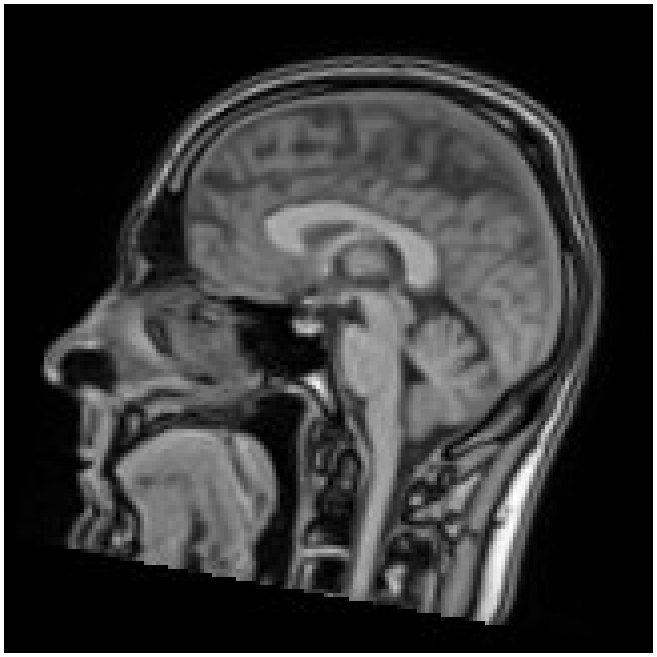


N: 26244
Mean: 38.113
StdDev: 52.349
Value: ---

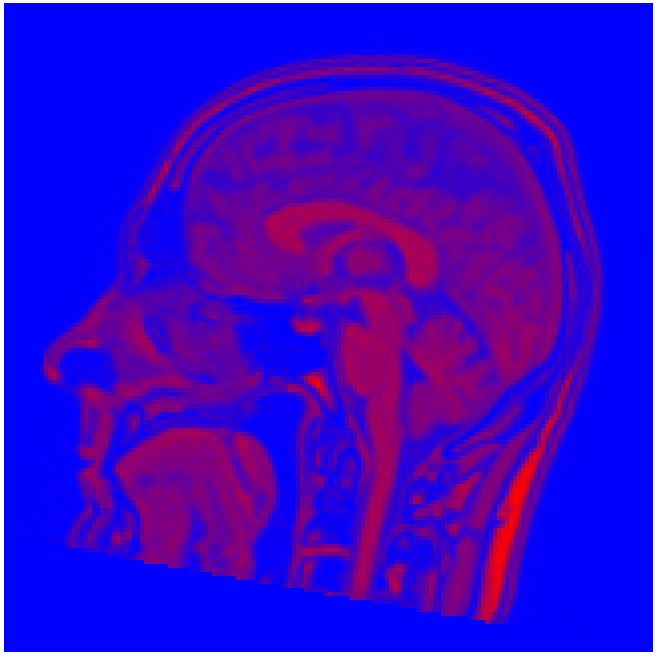
Min: 0
Max: 200
Mode: 0 (13050)
Count: ---

LUT w ImageJ

```
// stworzenie LUT w ImageJ

LUT_R = newArray(256);
LUT_G = newArray(256);
LUT_B = newArray(256);

for (i=0; i<256; i++)
{
        LUT_R[i] = i;
        LUT_G[i] = 0;
        LUT_B[i] = 255-i;
}

setLut(LUT_R, LUT_G, LUT_B);
```

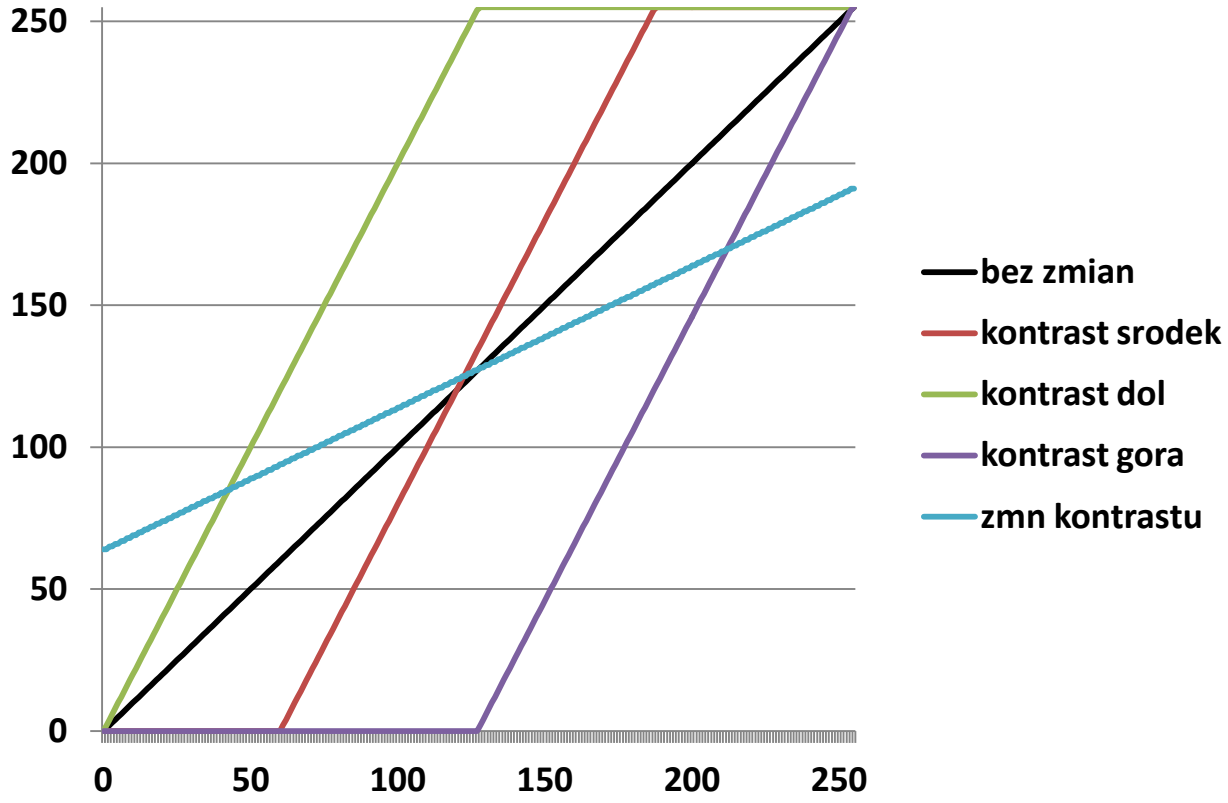


Number of unique colors: 246

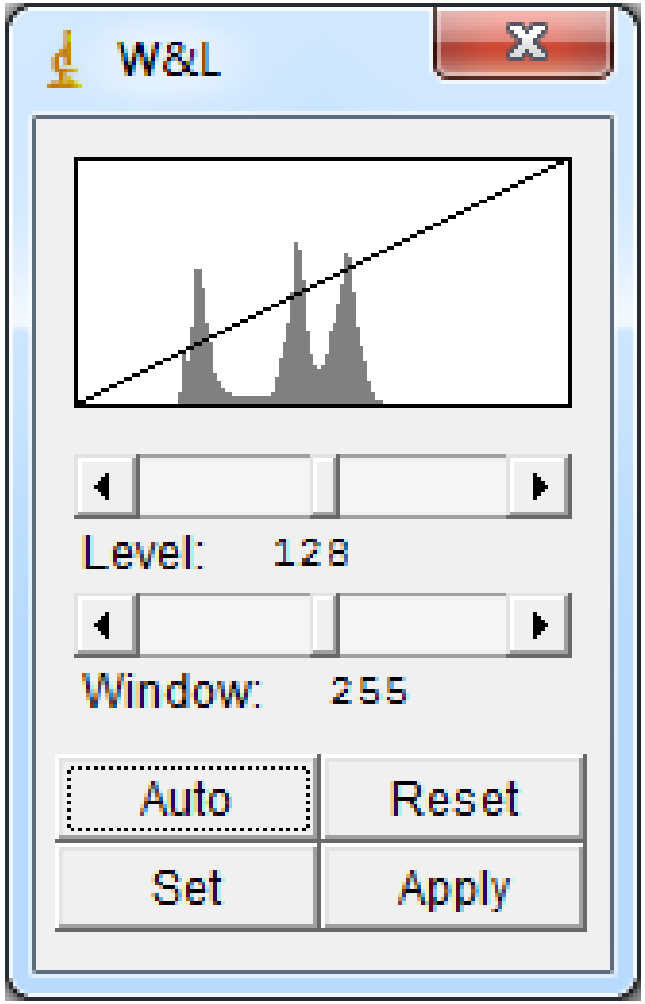


Number of unique colors: 246

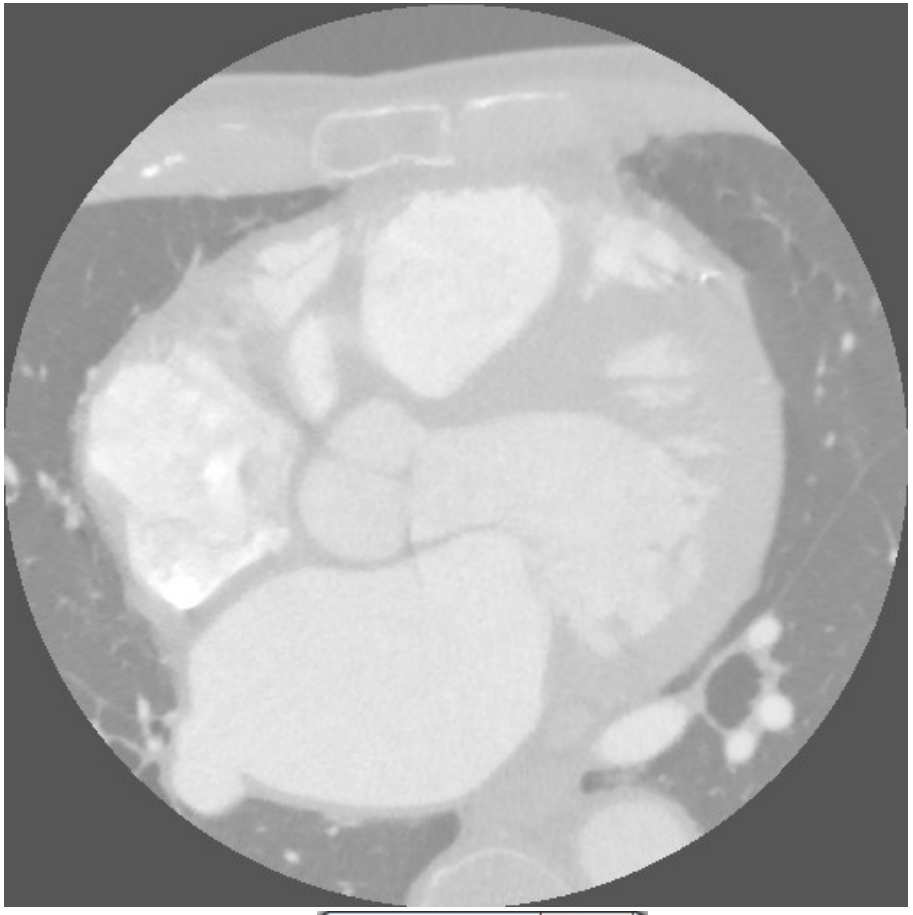
Kontrast i jasność (windowing)

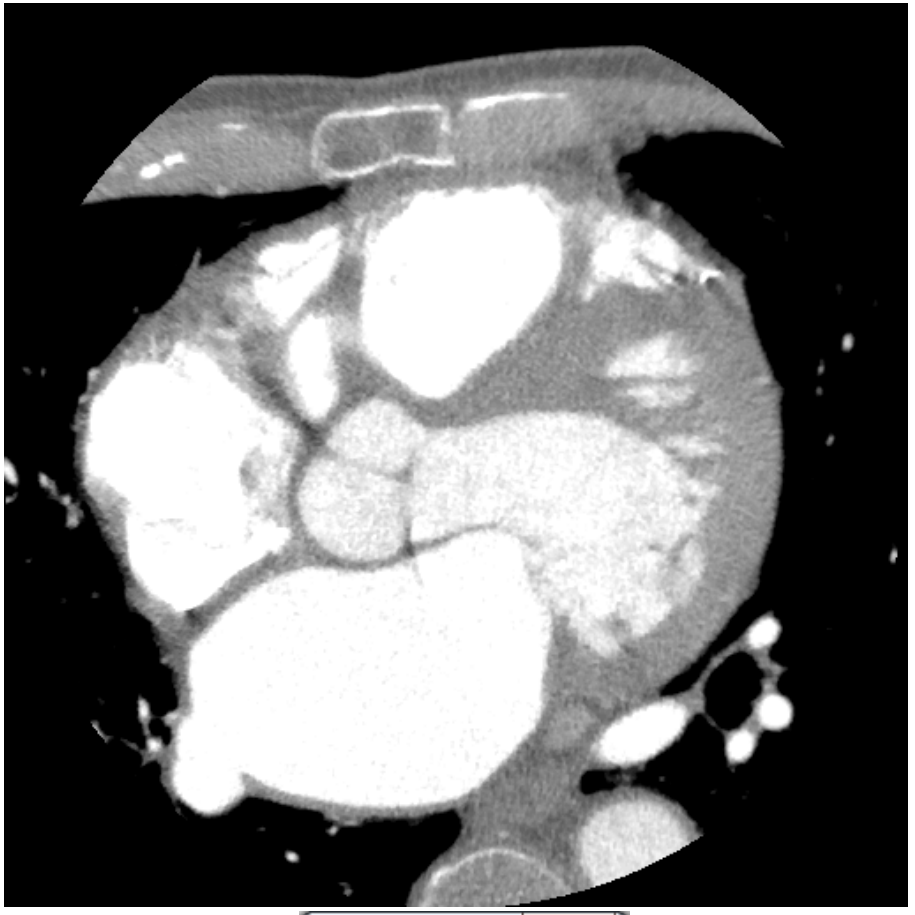


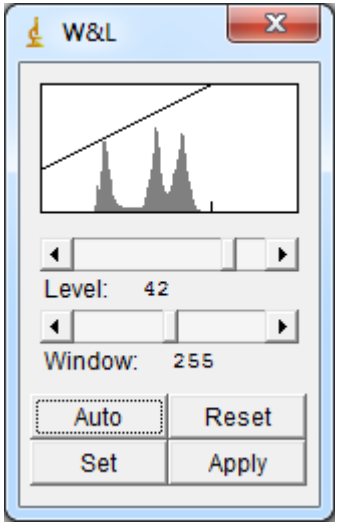
(0028,1050)	Window Center	80
(0028,1051)	Window Width	600

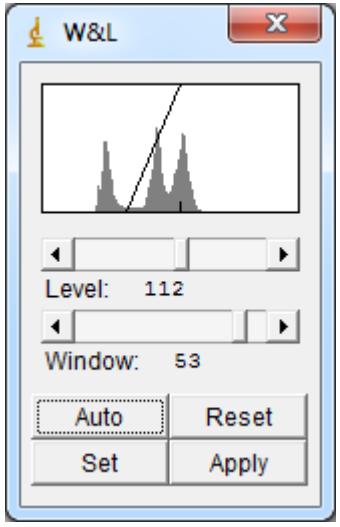


Wersja po wcześniejszym skonwertowaniu do 8 bit

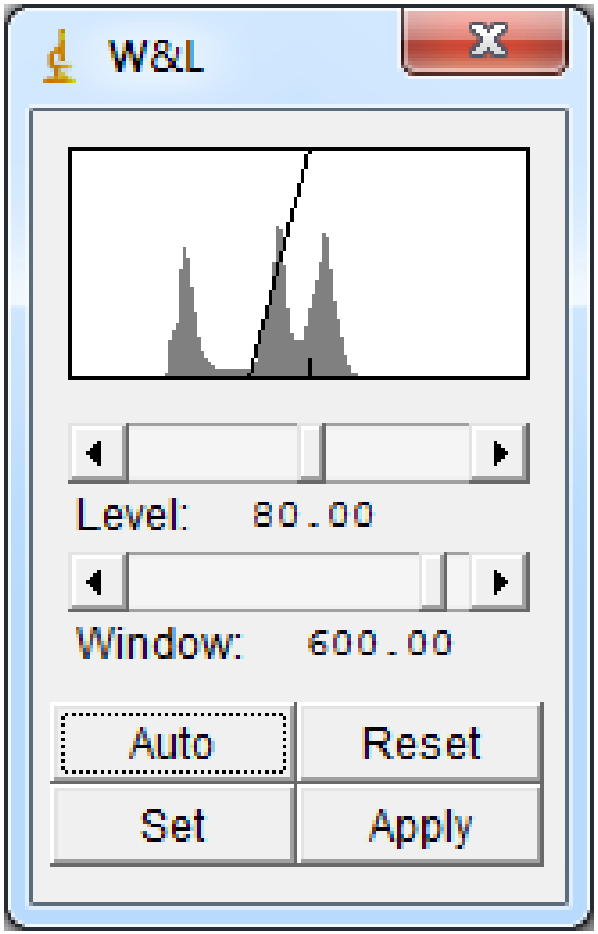




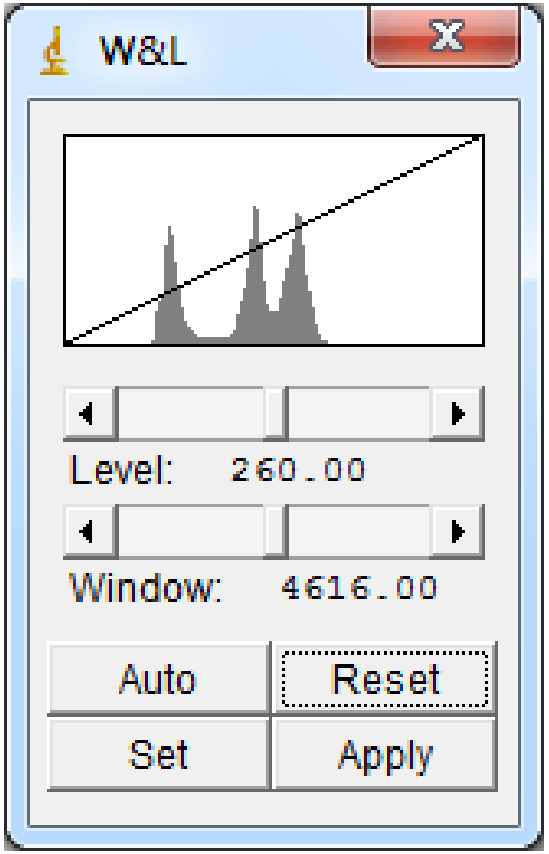

W&L
Level: 42
Window: 255
Auto
Reset
Set
Apply

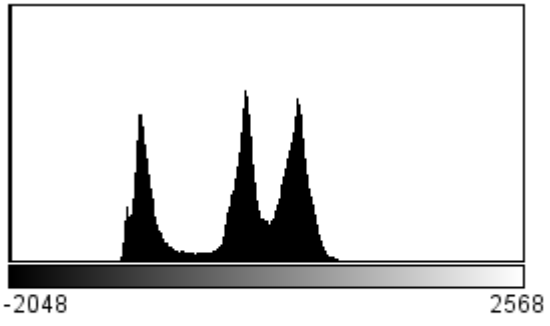

W&L
Level: 112
Window: 53
Auto
Reset
Set
Apply

Oryginalny plik DICOM, Toshiba, zapis ze znakiem (skala HU)



(0028,1050)	Window Center	80
(0028,1051)	Window Width	600

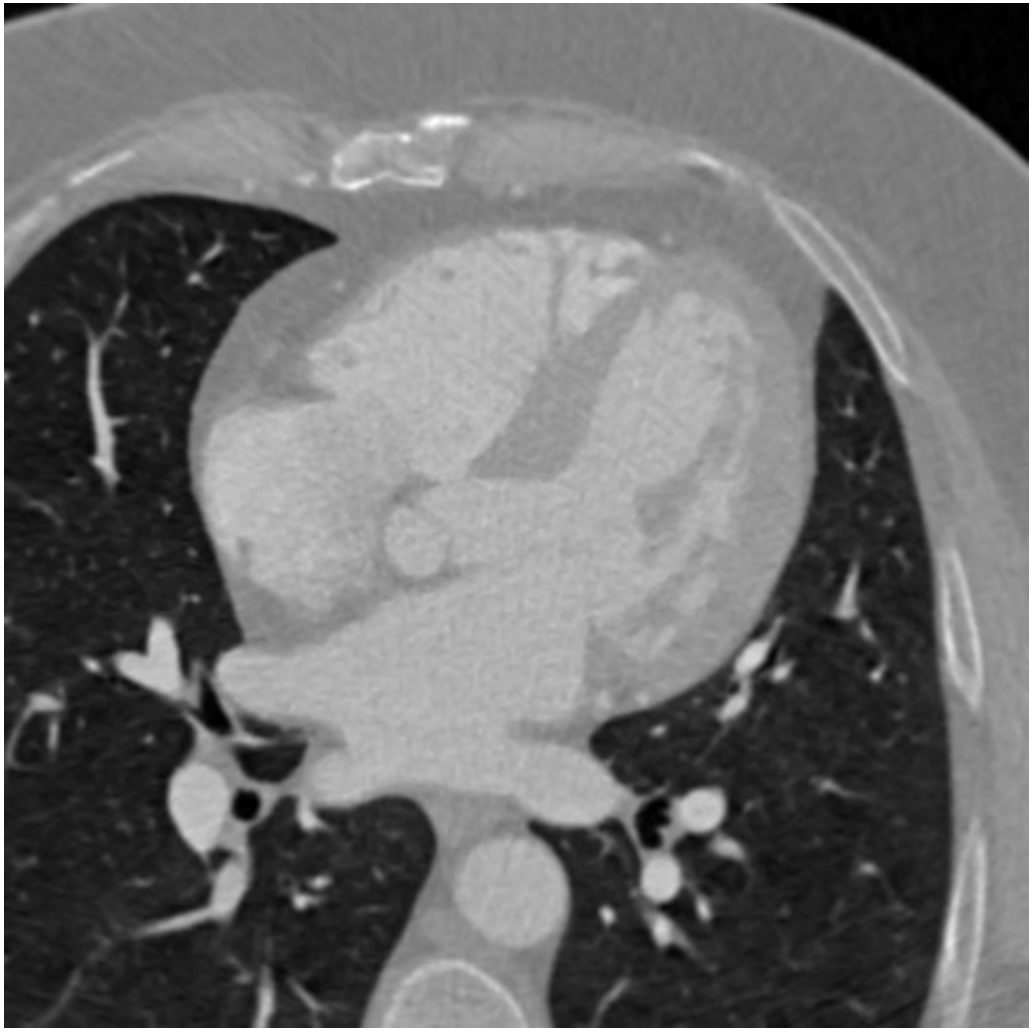


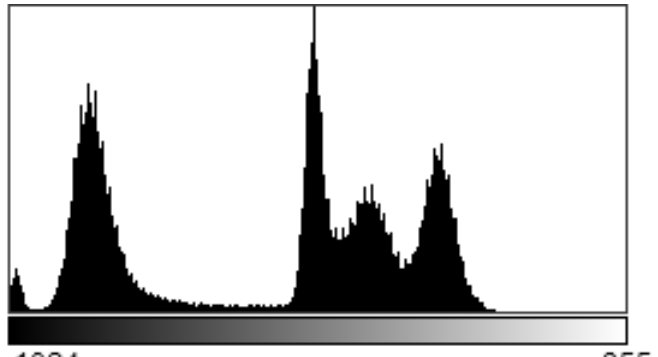


N: 262144 Min: -2048
Mean: -447.286 Max: 2568
StdDev: 973.193 Mode: -2048 (56252)
Bins: 256 Bin Width: 18.031
Value: 151.812

```
resetMinAndMax();
```

Problem skalowania danych CT 16-bit (DICOM) do obrazu 8-bit





N: 262144 Min: -1024
Mean: -224.530 Max: 855
StdDev: 431.507 Mode: -99 (6276)

Info for 092_14180_T8.dcm

File Edit Font

0028,0002 Samples per Pixel: 1
0028,0004 Photometric Interpretation: MONOCHROME2
0028,0010 Rows: 512
0028,0011 Columns: 512
0028,0030 Pixel Spacing: 0.42578125\0.42578125
0028,0100 Bits Allocated: 16
0028,0101 Bits Stored: 12
0028,0102 High Bit: 11
0028,0103 Pixel Representation: 0
0028,0106 Smallest Image Pixel Value: 0
0028,0107 Largest Image Pixel Value: 1879
0028,1050 Window Center: -600
0028,1051 Window Width: 1300
0028,1052 Rescale Intercept: -1024
0028,1053 Rescale Slope: 1
0028,1055 Window Center & Width Explanation: WINDO

v HU= v*RescaleSlope + RescaleIntercept

Problem skalowania danych 16-bit (CT) do obrazu 8-bit

1. Metoda min-max -> 0-255 (**B08**)

Zachowane liniowe zależności, ale dla każdego obrazu inne – w zależności od wartości min i max, grupa obrazów może mieć różną jasność

8-bit https://imagej.net/ij/docs/menus/image.html

Converts to 8-bit grayscale. The active image must be 16-bit grayscale, 32-bit grayscale, 8-bit color or RGB color.

ImageJ converts 16-bit and 32-bit images and stacks to 8-bits by linearly scaling from min-max to 0-255, where min and max are the two values displayed in the *Image>Adjust>Brightness>Contrast* tool. *Image>Show Info* displays these two values as the "Display range". Note that this scaling is not done if "Scale When Converting" is not checked in *Edit>Options>Conversions*.

2. Dzielenie przez 16 (redukcja z 12 bit do 8 bit poprzez dzielenie przez 2^4 (**B16_divb04_B08**)

<0,4095>/16

Bardzo duża redukcja szczegółów w tkankach, szczególnie o niewielkim zakresie jasności (HU), jak np. chrząstka,

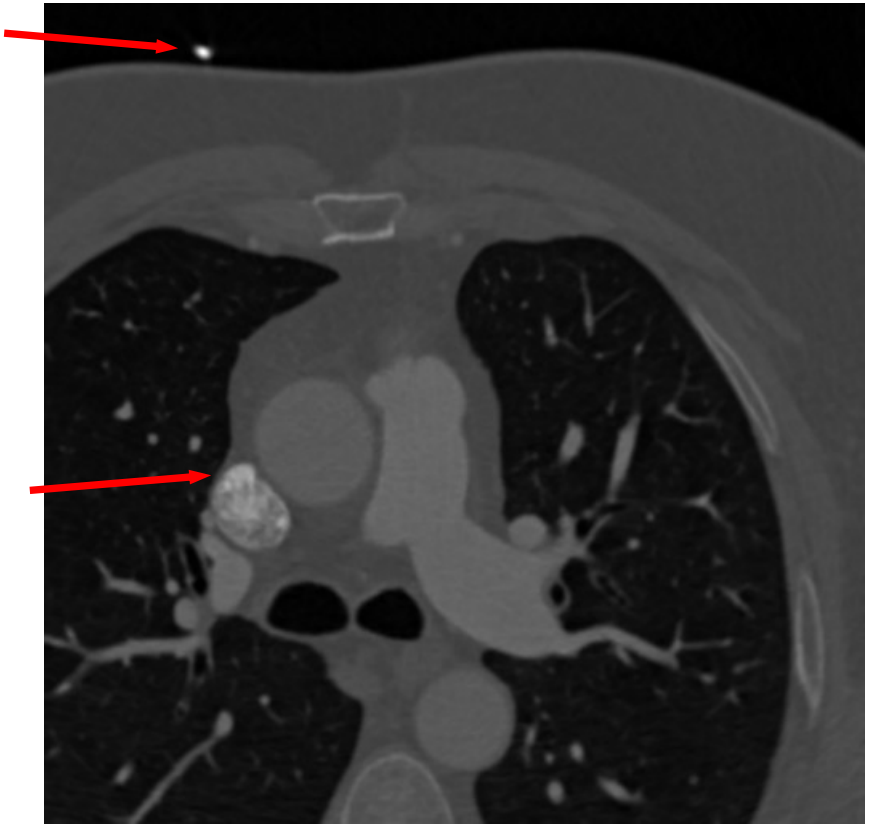
3. Dzielenie przez 8 (redukcja z 12 bit do 8 bit poprzez dzielenie przez 2^3 i limitowanie do 255 (w oryginale powyżej 2048 (**B16_divb03_L-B08**)

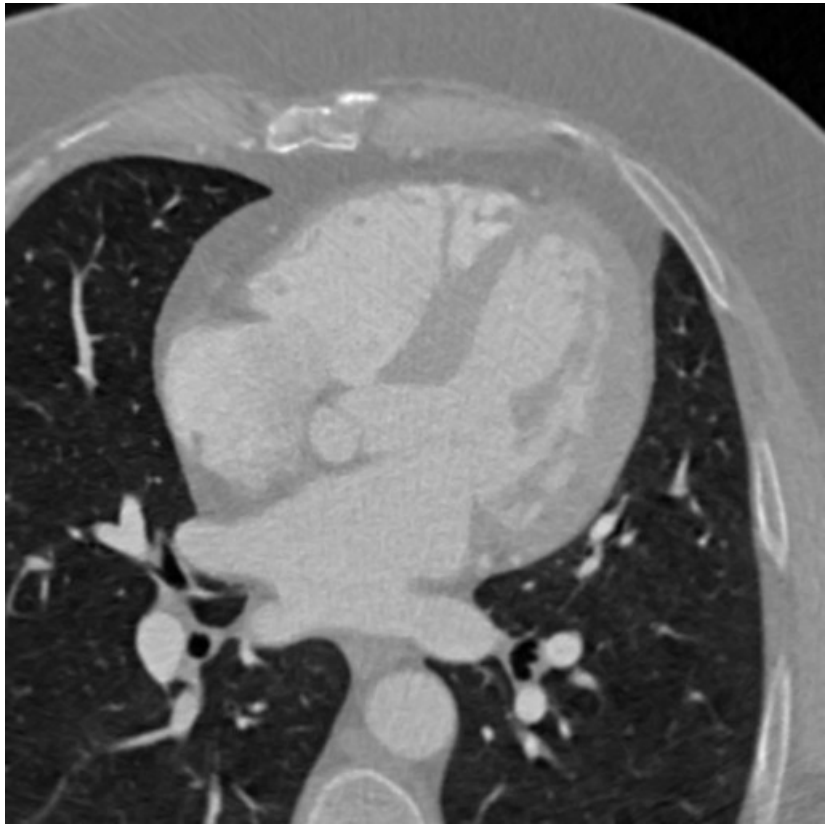
MIN (<0,4095>/8 , 255) albo: MIN (<0,4095 >, 2047) /8

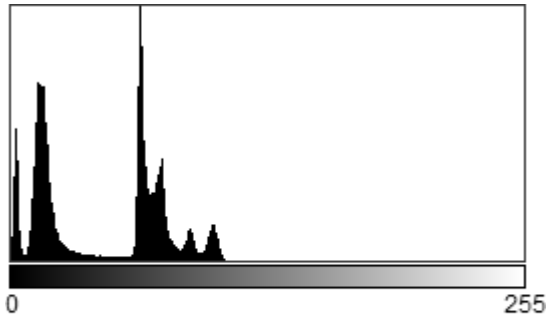
Redukcja szczegółów w tkankach 2x mniejsza niż w metodzie 2, większy kontrast, ale zniekształcenia obiektów jasnych – dopuszczalne, jeśli nie są istotne

Nurzynska, K., et al.:.Differentiating age and sex in vertebral body CT scans–Texture analysis versus deep learning approach. *Biocybernetics and Biomedical Engineering*, (2024), *44*(1), 20-30.

Problem skalowania danych 16-bit (CT) do obrazu 8-bit

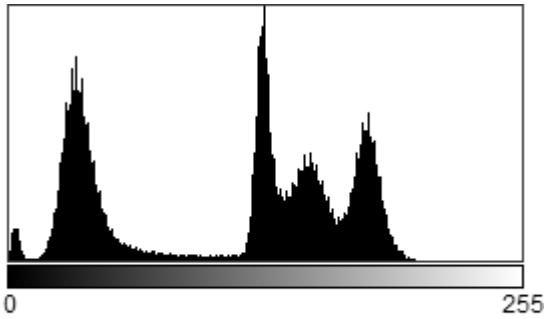






N: 262144
Mean: 45.589
StdDev: 31.864
Min: 0
Max: 255
Mode: 64 (14707)

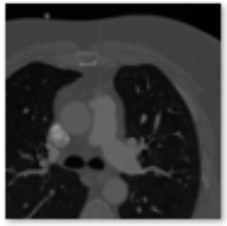
B08

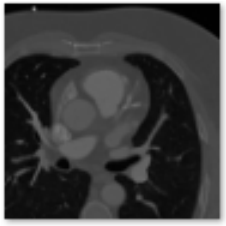


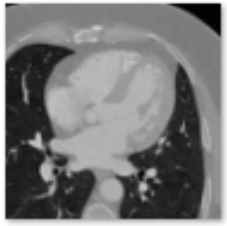
N: 262144
Mean: 108.865
StdDev: 58.762
Min: 0
Max: 255
Mode: 127 (6097)

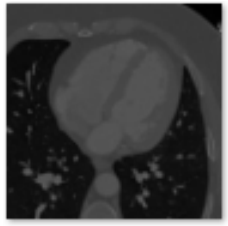
Problem skalowania danych 16-bit (CT) do obrazu 8-bit

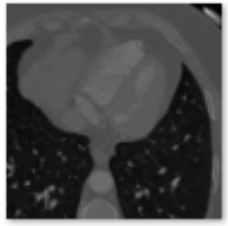
B08

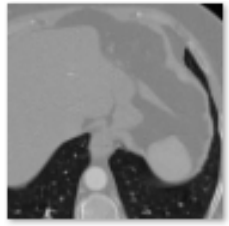

092_14095_T6_B08.png

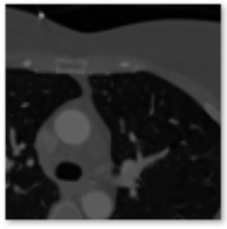

092_14128_T7_B08.png

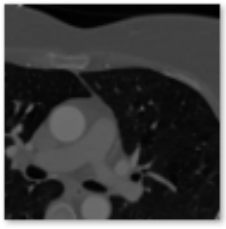

092_14180_T8_B08.png

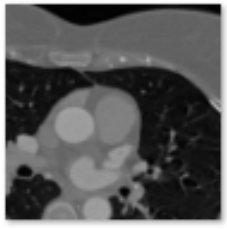

092_14226_T9_B08.png

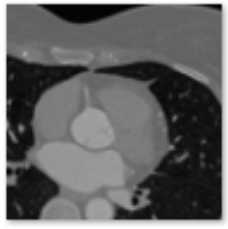

092_14267_T10_B08.png

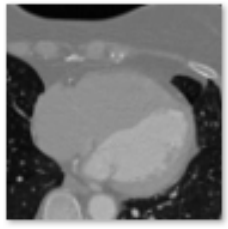

092_14317_T11_B08.png

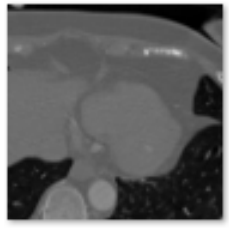

096_32776_T6_B08.png

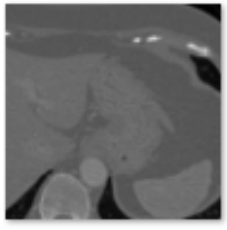

096_32815_T7_B08.png

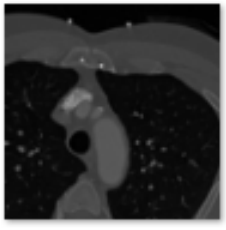

096_32847_T8_B08.png

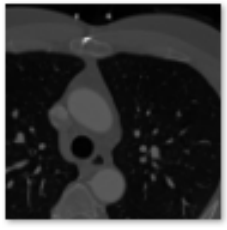

096_32887_T9_B08.png

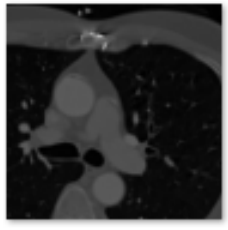

096_32956_T10_B08.png

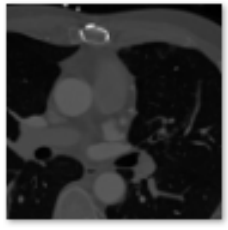

096_33003_T11_B08.png

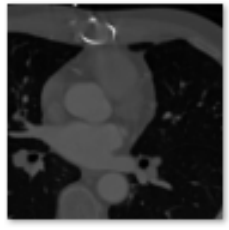

096_33063_T12_B08.png

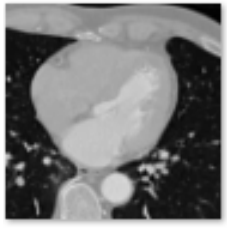

098_22012_T4_B08.png

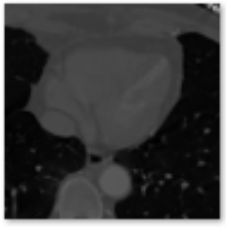

098_22059_T5_B08.png

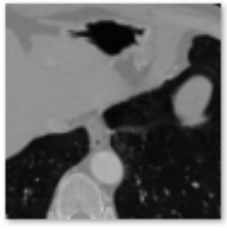

098_22105_T6_B08.png

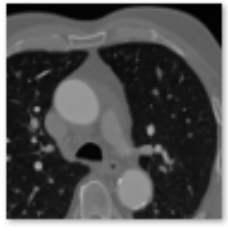

098_22153_T7_B08.png

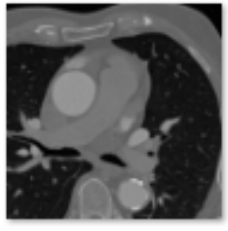

098_22189_T8_B08.png

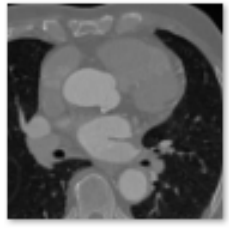

098_22249_T9_B08.png


098_22310_T10_B08.png


098_22358_T11_B08.png

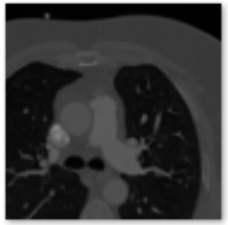

099_15361_T6_B08.png

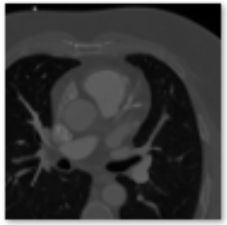

099_15407_T7_B08.png

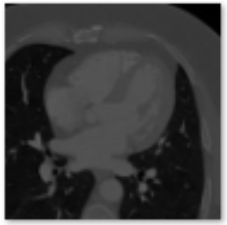

099_15450_T8_B08.png

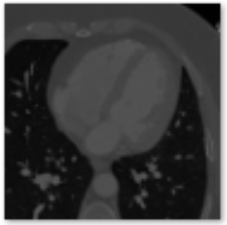
Problem skalowania danych 16-bit (CT) do obrazu 8-bit

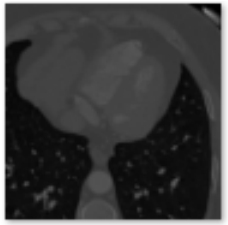
B16-divb04-B08

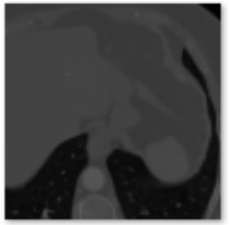

092_14095_T6_B1
2divB04B08.png

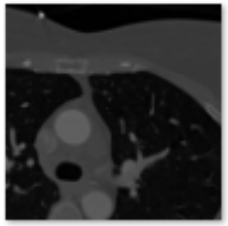

092_14128_T7_B1
2divB04B08.png

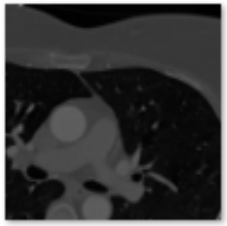

092_14180_T8_B1
2divB04B08.png

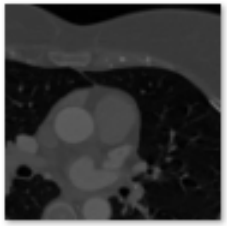

092_14226_T9_B1
2divB04B08.png

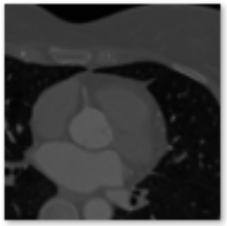

092_14267_T10_B
12divB04B08.png

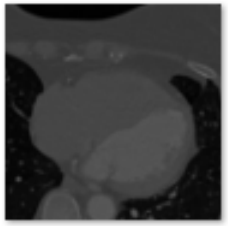

092_14317_T11_B
12divB04B08.png

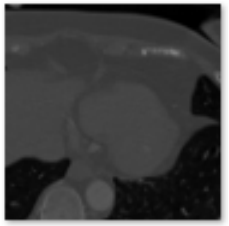

096_32776_T6_B1
2divB04B08.png

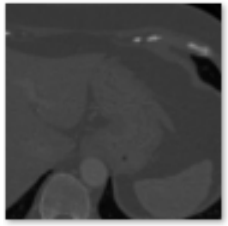

096_32815_T7_B1
2divB04B08.png

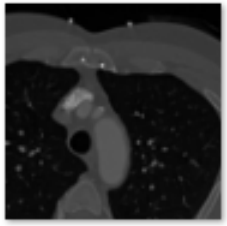

096_32847_T8_B1
2divB04B08.png

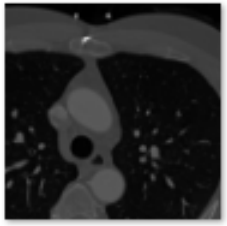

096_32887_T9_B1
2divB04B08.png

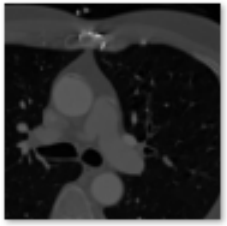

096_32956_T10_B
12divB04B08.png

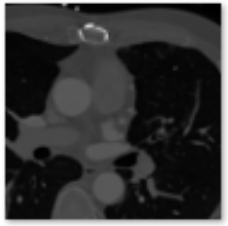

096_33003_T11_B
12divB04B08.png

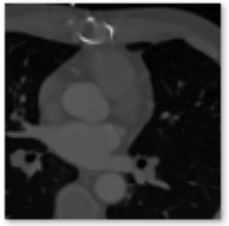

096_33063_T12_B
12divB04B08.png

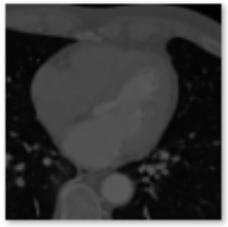

098_22012_T4_B1
2divB04B08.png

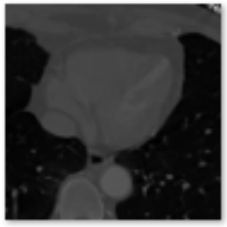

098_22059_T5_B1
2divB04B08.png

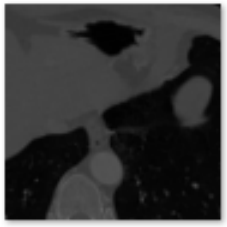

098_22105_T6_B1
2divB04B08.png

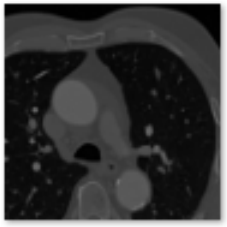

098_22153_T7_B1
2divB04B08.png

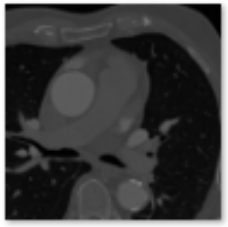

098_22189_T8_B1
2divB04B08.png

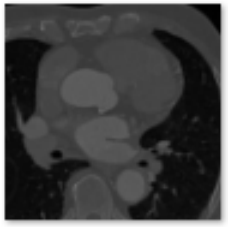

098_22249_T9_B1
2divB04B08.png


098_22310_T10_B
12divB04B08.png


098_22358_T11_B
12divB04B08.png

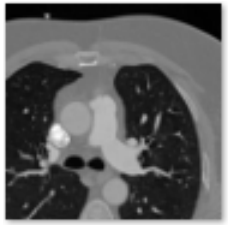

099_15361_T6_B1
2divB04B08.png

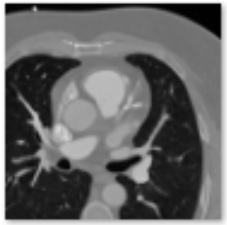

099_15407_T7_B1
2divB04B08.png

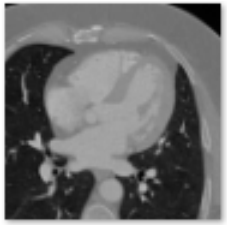

099_15450_T8_B1
2divB04B08.png

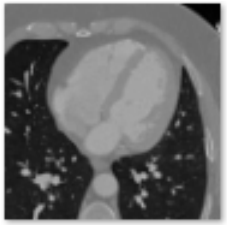
Problem skalowania danych 16-bit (CT) do obrazu 8-bit

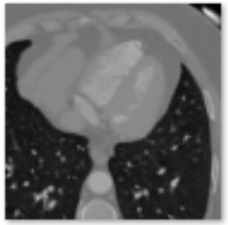
B16-divb03_L-B08

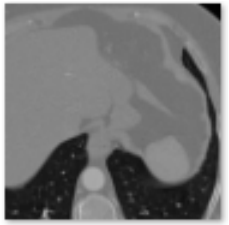

092_14095_T6_div B03L_B08.png

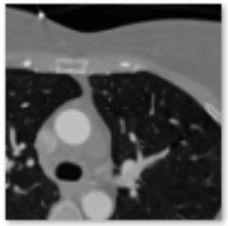

092_14128_T7_div B03L_B08.png

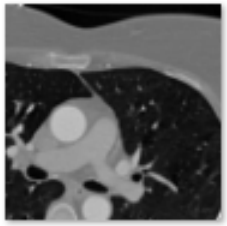

092_14180_T8_div B03L_B08.png

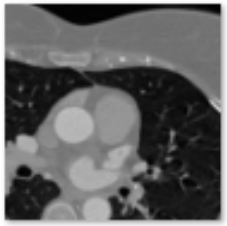

092_14226_T9_div B03L_B08.png

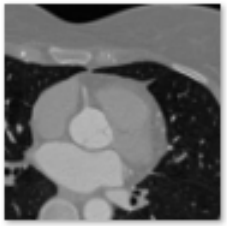

092_14267_T10_di vB03L_B08.png

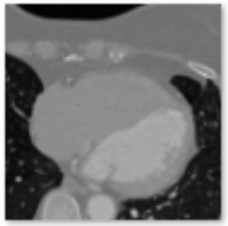

092_14317_T11_di vB03L_B08.png

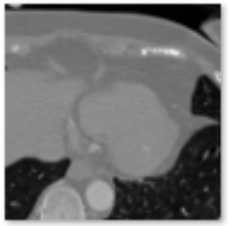

096_32776_T6_div B03L_B08.png

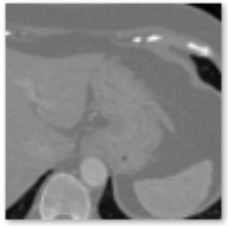

096_32815_T7_div B03L_B08.png

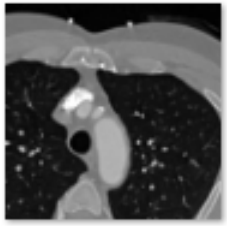

096_32847_T8_div B03L_B08.png

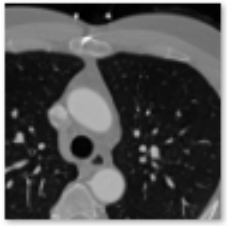

096_32887_T9_div B03L_B08.png

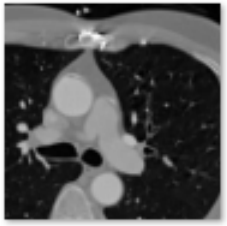

096_32956_T10_di vB03L_B08.png

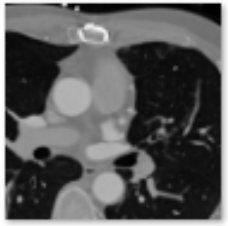

096_33003_T11_di vB03L_B08.png

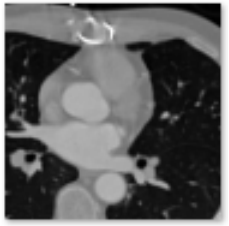

096_33063_T12_di vB03L_B08.png

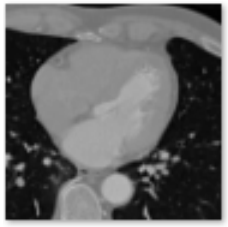

098_22012_T4_div B03L_B08.png

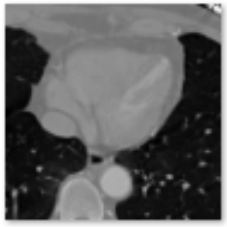

098_22059_T5_div B03L_B08.png

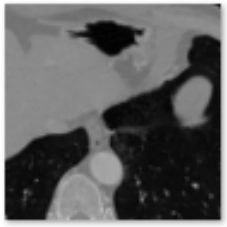

098_22105_T6_div B03L_B08.png

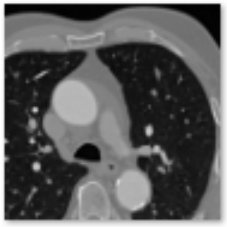

098_22153_T7_div B03L_B08.png

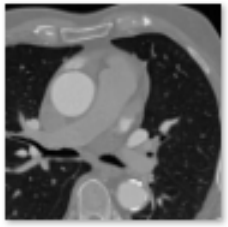

098_22189_T8_div B03L_B08.png

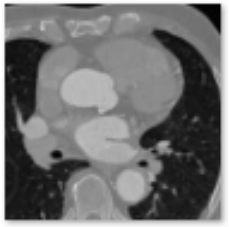

098_22249_T9_div B03L_B08.png


098_22310_T10_di vB03L_B08.png

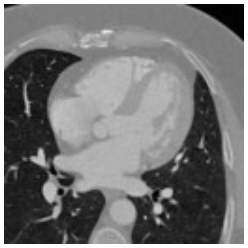

098_22358_T11_di vB03L_B08.png


099_15361_T6_div B03L_B08.png

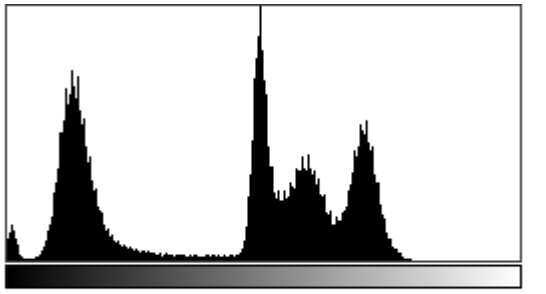

099_15407_T7_div B03L_B08.png


099_15450_T8_div B03L_B08.png

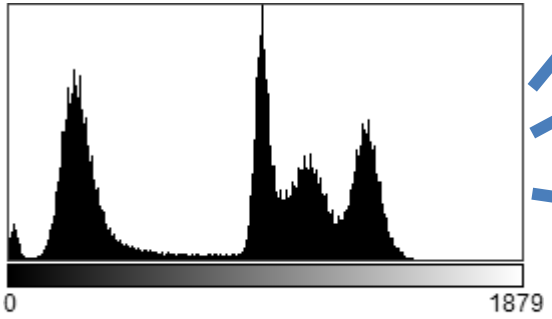
Problem skalowania danych 16-bit (CT) do obrazu 8-bit



DICOM



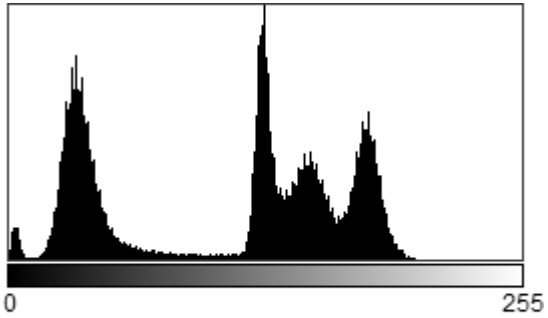
N: 262144
Mean: -224.530
StdDev: 431.507
Bins: 256
Min: -1024
Max: 855
Mode: -99 (6276)
Bin Width: 7.340



N: 262144
Mean: 799.470
StdDev: 431.507
Bins: 256
Min: 0
Max: 1879
Mode: 925 (6276)
Bin Width: 7.340

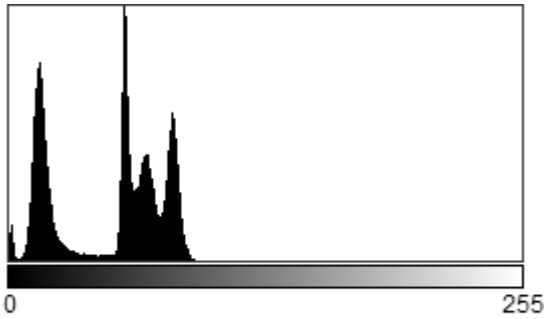
B16

B08



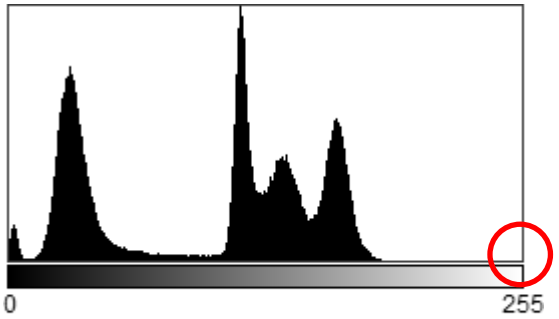
N: 262144
Mean: 108.865
StdDev: 58.762
Min: 0
Max: 255
Mode: 127 (6097)

B16-divb04-B08



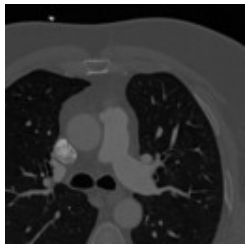
N: 262144
Mean: 49.498
StdDev: 26.970
Min: 0
Max: 117
Mode: 57 (12266)

B16-divb03_L-B08

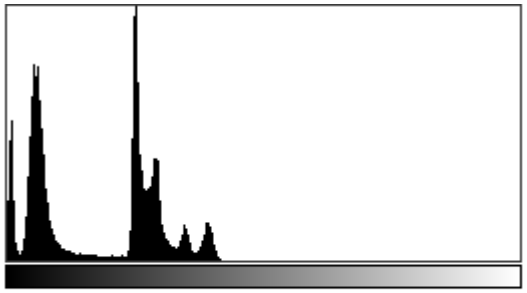


N: 262144
Mean: 99.495
StdDev: 53.940
Min: 0
Max: 234
Mode: 115 (6409)

Problem skalowania danych 16-bit (CT) do obrazu 8-bit

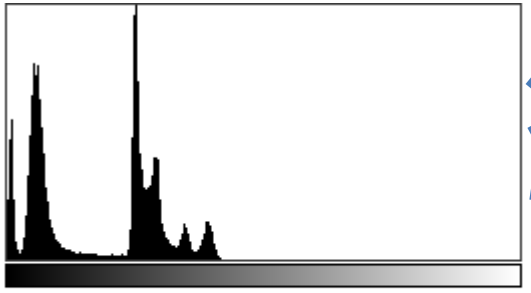


DICOM



N: 262144
Mean: -368.160
StdDev: 458.400
Bins: 256

Min: -1024
Max: 2658
Mode: -105 (13901)
Bin Width: 14.383

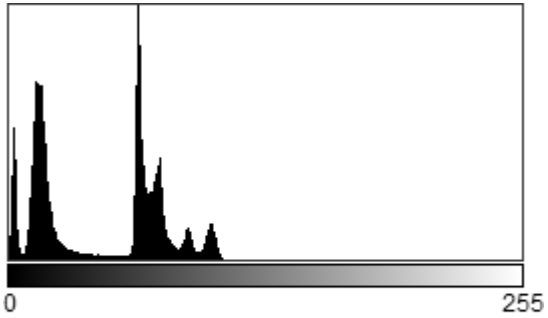


N: 262144
Mean: 655.840
StdDev: 458.400
Bins: 256

Min: 0
Max: 3682
Mode: 919 (13901)
Bin Width: 14.383

B16

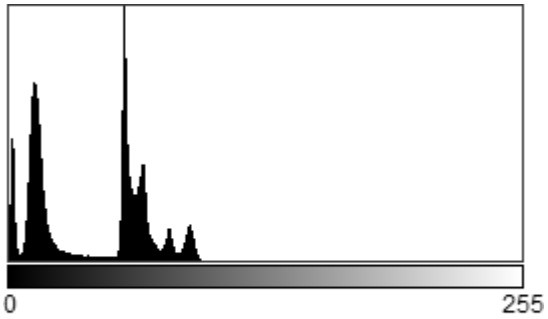
B08



N: 262144
Mean: 45.589
StdDev: 31.864

Min: 0
Max: 255
Mode: 64 (14707)

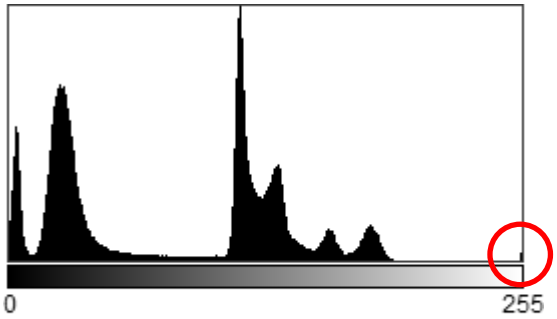
B16-divb04-B08



N: 262144
Mean: 40.522
StdDev: 28.650

Min: 0
Max: 230
Mode: 57 (16493)

B16-divb03_L-B08

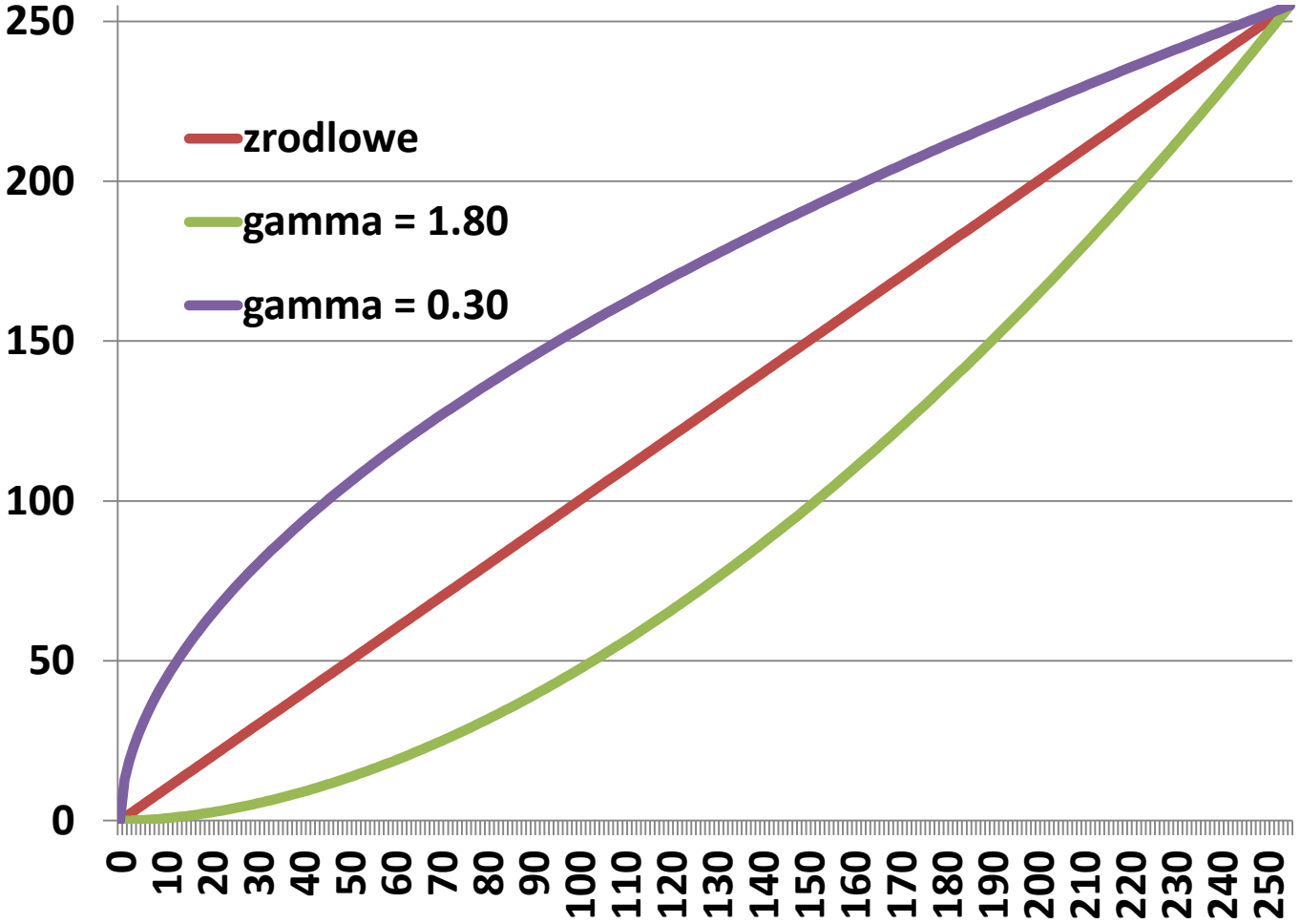


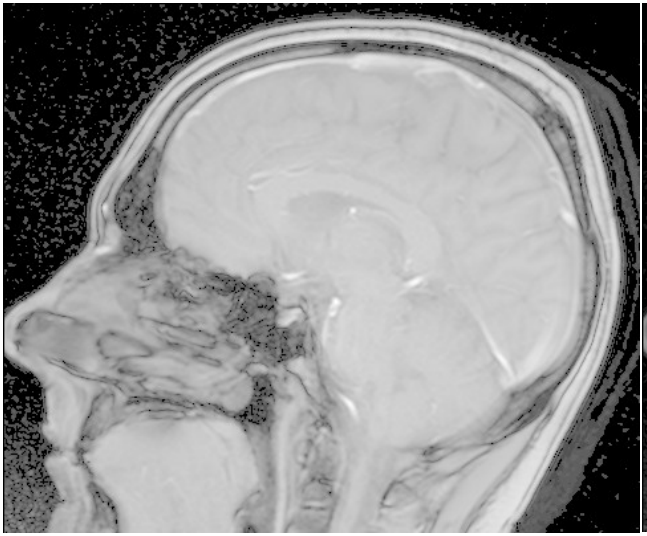
N: 262144
Mean: 81.511
StdDev: 57.183

Min: 0
Max: 255
Mode: 115 (8464)

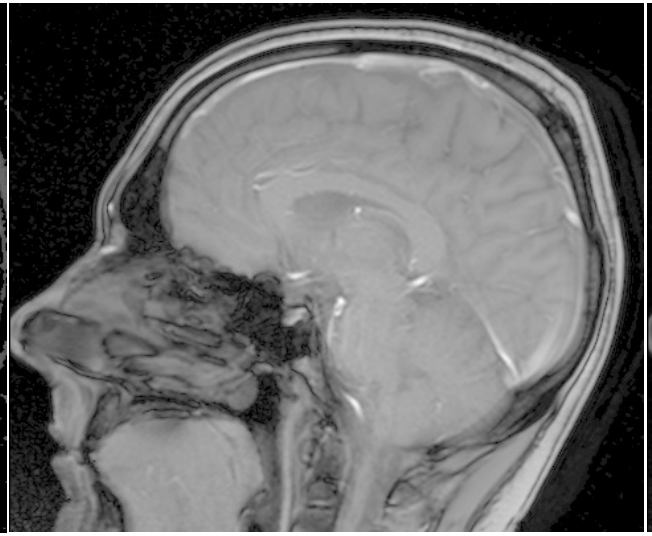
$$P'(x,y) = \left(\frac{P(x,y)}{255}\right)^{\gamma} * 255$$

Korekcja gamma

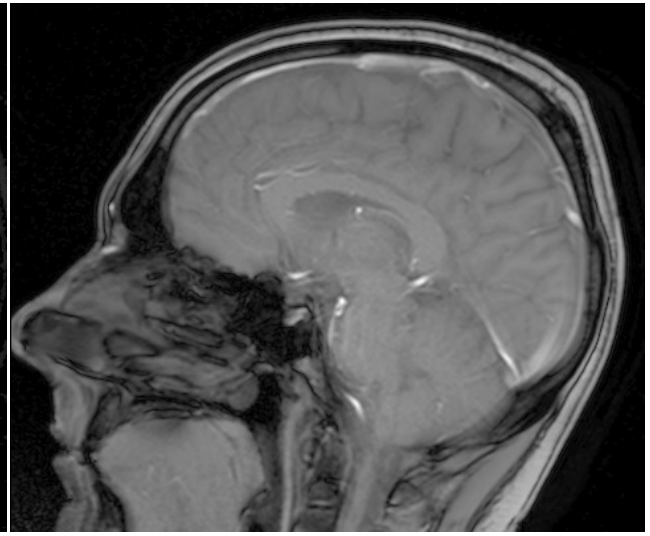




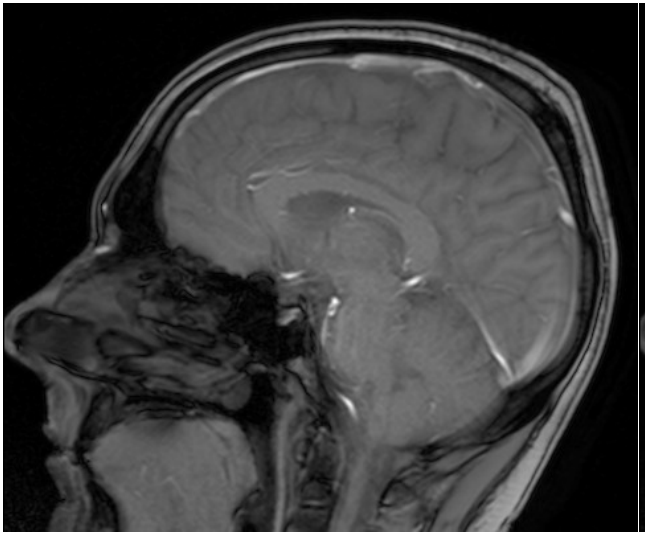
0.2



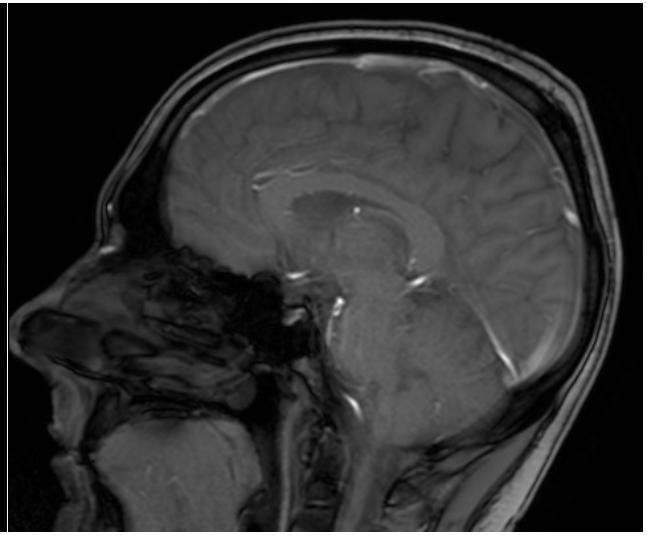
0.4



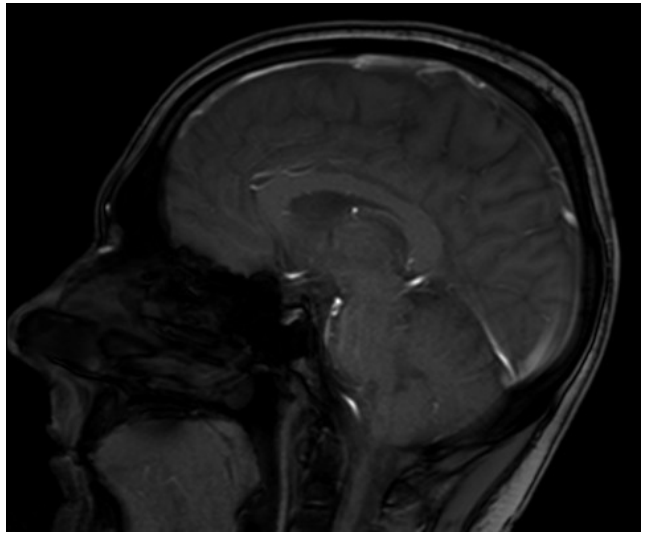
0.6



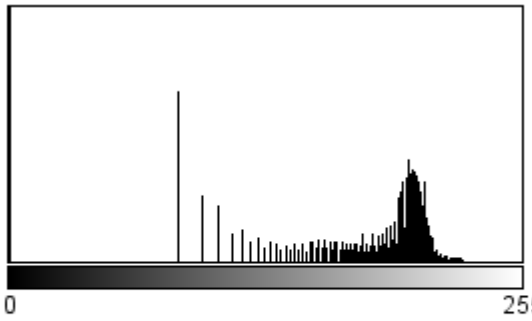
0.8



SRC

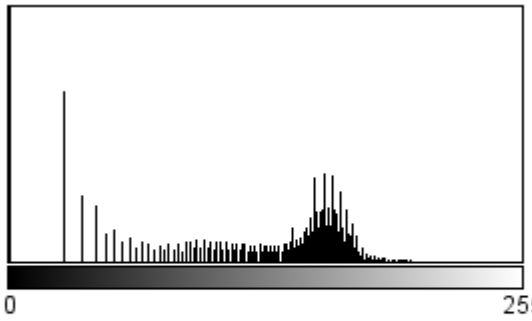


1.5



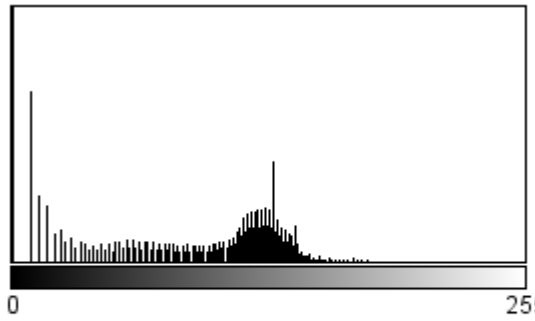
N: 192000 Min: 0
Mean: 144.816 Max: 255
StdDev: 74.560 Mode: 0 (32954)

0.2



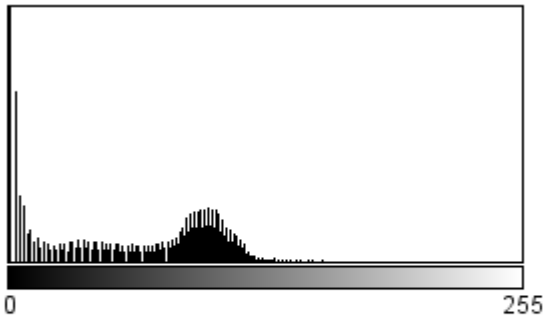
N: 192000 Min: 0
Mean: 104.133 Max: 255
StdDev: 63.388 Mode: 0 (32954)

0.4



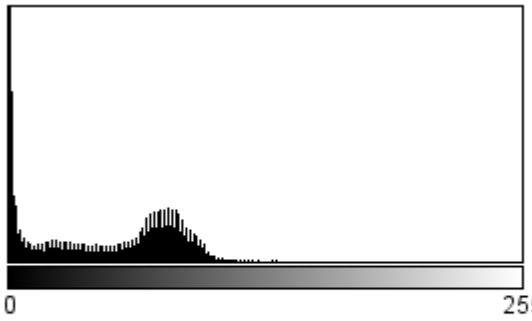
N: 192000 Min: 0
Mean: 77.231 Max: 255
StdDev: 53.393 Mode: 0 (32954)

0.6



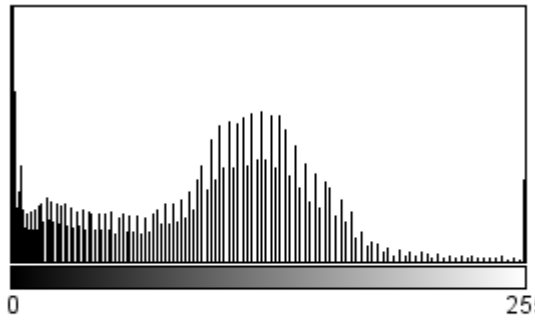
N: 192000 Min: 0
Mean: 58.225 Max: 255
StdDev: 44.321 Mode: 0 (32954)
Value: --- Count: ---

0.8



N: 192000 Min: 0
Mean: 45.004 Max: 255
StdDev: 36.960 Mode: 0 (32954)
Value: 142 Count: 50

SRC

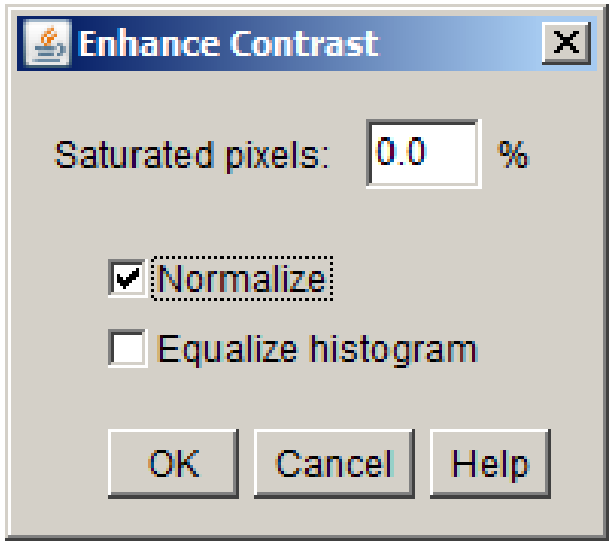


N: 192000 Min: 0
Mean: 67.145 Max: 255
StdDev: 63.648 Mode: 0 (50424)

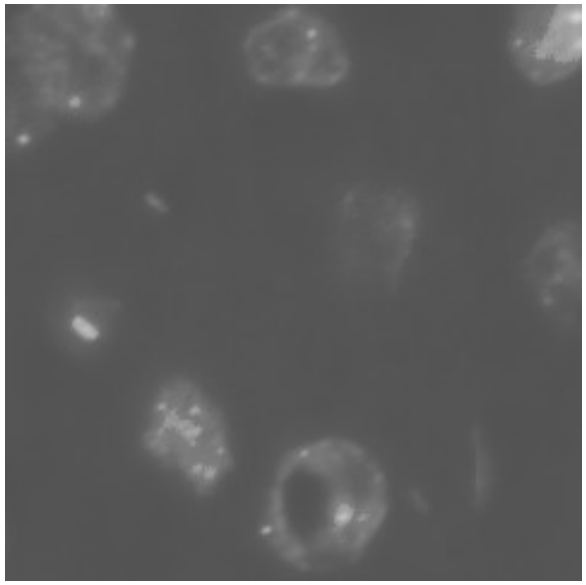
1.5

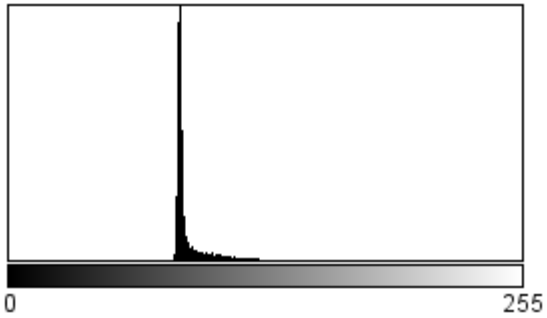
Normalizacja (rozciągnięcie histogramu)

$$P'(x,y) = \left(\frac{P(x,y) - min}{max - min} \right) * 255$$



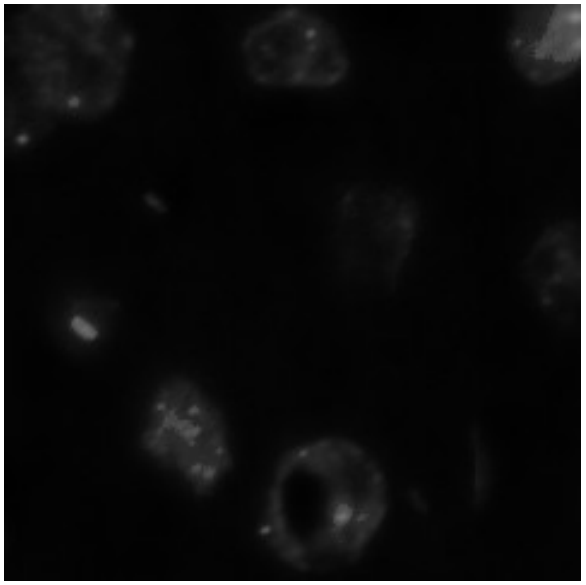
- najmniejsza wartość przesunięta do wartości 0
- największa wartość przesunięta do wartości 255
- wartości pośrednie proporcjonalnie rozciągnięte do przedziału 0-255





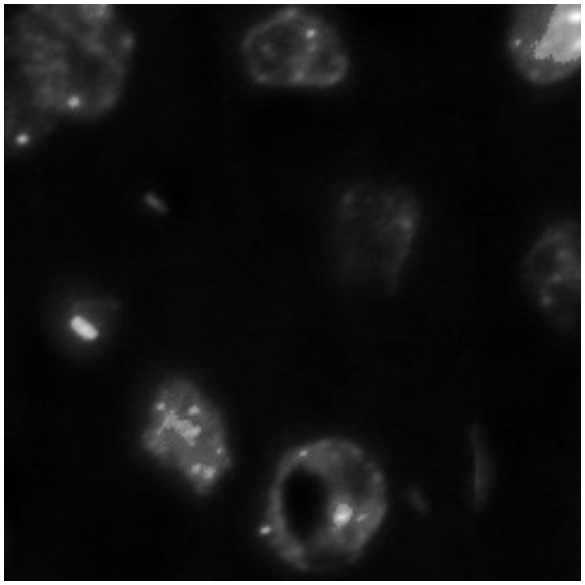
N: 57600
Mean: 89.562
StdDev: 11.567
Value: ---

Min: 81
Max: 210
Mode: 85 (14531)
Count: ---



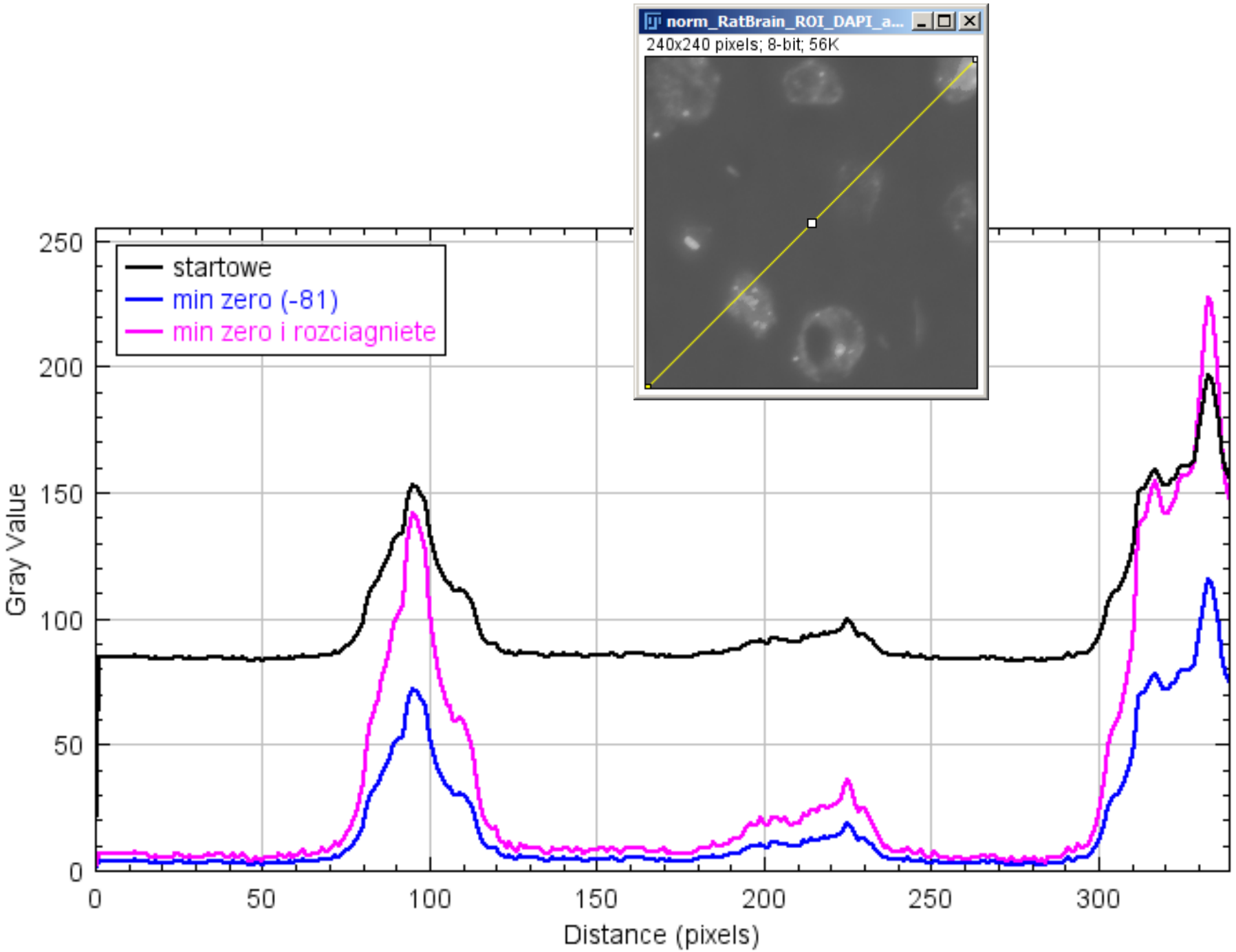
N: 57600
Mean: 8.562
StdDev: 11.567
Value: 172

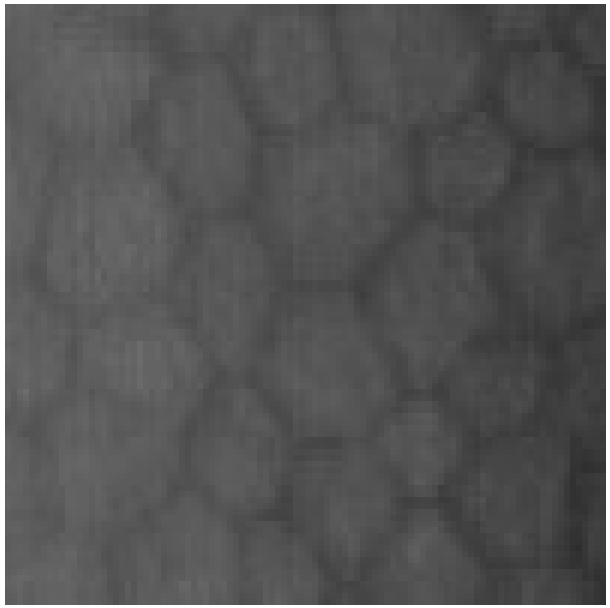
Min: 0
Max: 129
Mode: 4 (14531)
Count: 0

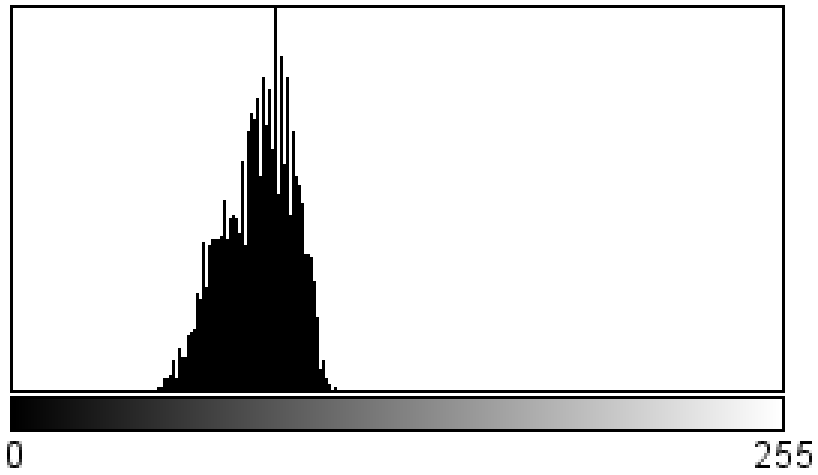


N: 57600
Mean: 16.105
StdDev: 23.028
Value: ---

Min: 0
Max: 255
Mode: 7 (14531)
Count: ---

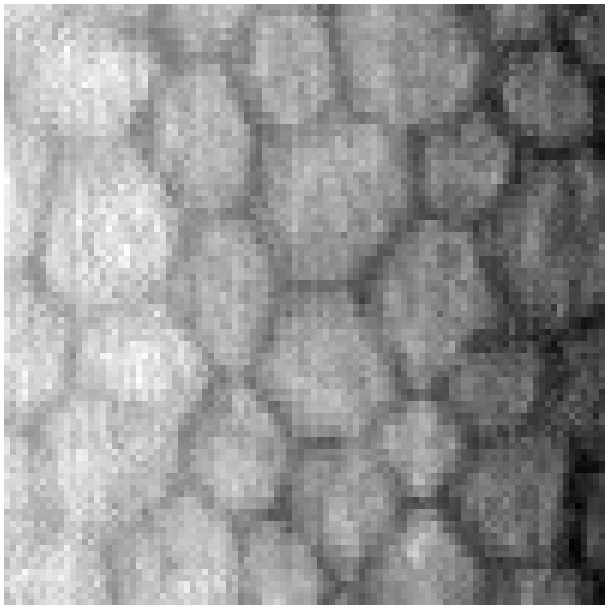

norm_RatBrain_ROI_DAPI_a...
240x240 pixels; 8-bit; 56K
startowe
min zero (-81)
min zero i rozciagniete
Gray Value
Distance (pixels)
0
50
100
150
200
250
300

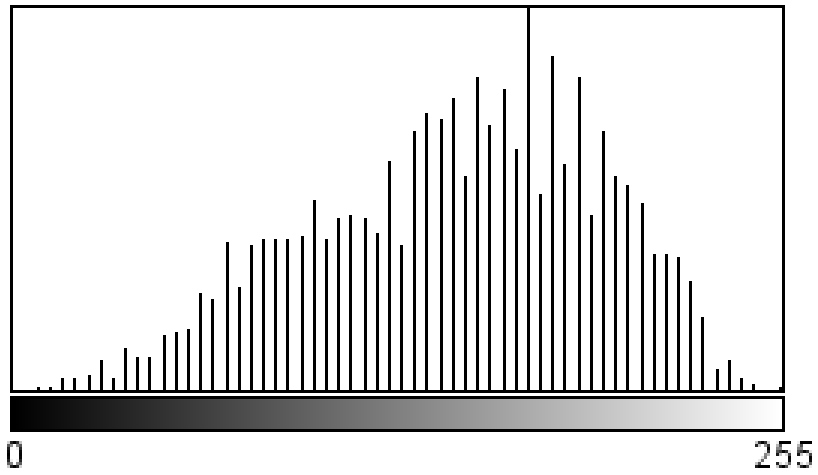




N: 10000
Mean: 81.131
StdDev: 11.547
Value: 80

Min: 46
Max: 107
Mode: 87 (447)
Count: 315





N: 10000
Mean: 146.367
StdDev: 48.281
Value: 252

Min: 0
Max: 255
Mode: 171 (447)
Count: 0

Wyrównanie histogramu

Cel: równomierny (o ile to możliwe) rozkład reprezentacji jasności pikseli w założonym zakresie
- zwiększanie różnic jasności między pikselami, których jest dużo

$$H(i) = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} \begin{cases} 1 & p(x,y) = i \\ 0 & p.p. \end{cases}$$

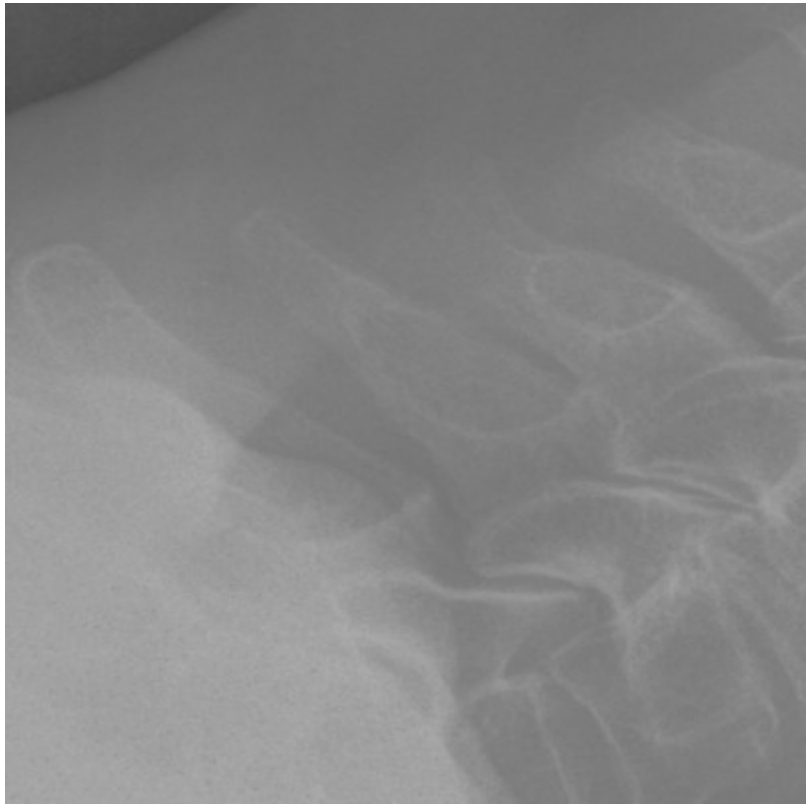
$$D(i) = \sum_{j=0}^{i} H(i)$$

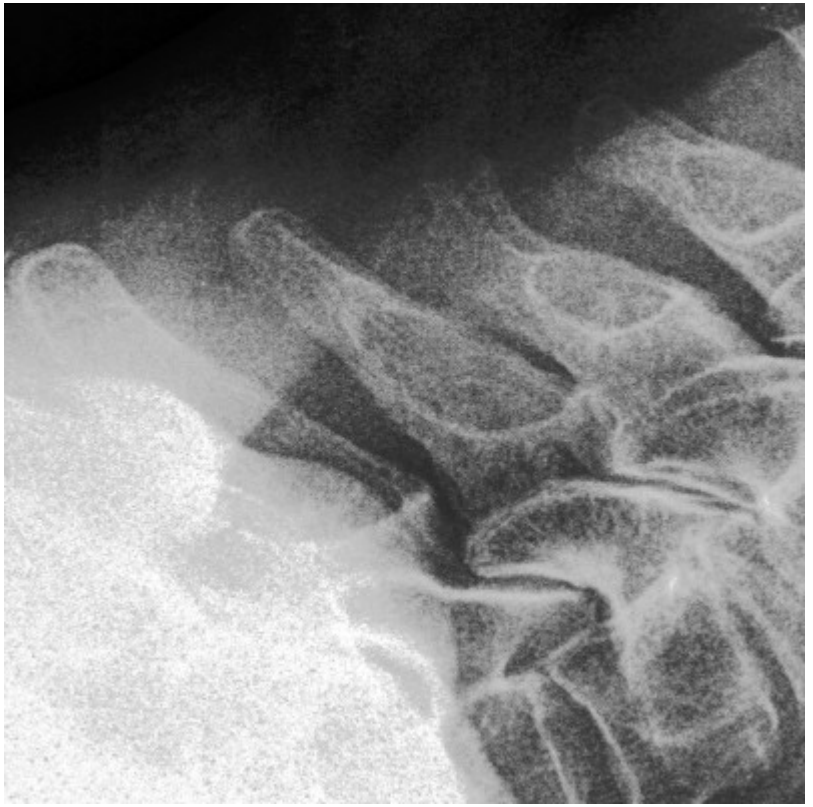
$$W(i) = \frac{D(i)}{w * h} * 255$$

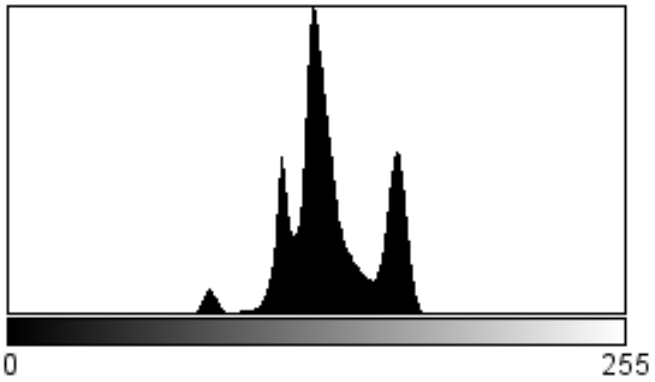
Wersja uproszczona – bez wyrównywania do najmniejszej niezerowej wartości D(i)

```
run("8-bit");          // na wszelki wypadek - 0-255
w = getWidth();
h = getHeight();
liczba_pixeli = w*h;

tablica_histogram = newArray(256);        // tablica dla 8-bit

for(x=0;x<w;x++)                          // wyliczenie histogramu
        for(y=0;y<h;y++)
                tablica_histogram[ getPixel(x, y) ] +=1;

tablica_dystrybuanta = newArray(256);     // tablica dla 8-bit

for(i=0;i<256;i++)                        // wyliczenie dystrybuanty (ze wzoru)
        for(j=0;j<=i;j++)
                tablica_dystrybuanta[i] += tablica_histogram[j];

// mozna policzyc szybciej
//tablica_dystrybuanta[0] = tablica_histogram[0];
//for(i=1;i<256;i++)
//        tablica_dystrybuanta[i] = tablica_dystrybuanta[i-1] + tablica_histogram[i];

tablica_LUT = newArray(256);              // tablica dla 8-bit

for(i=0;i<256;i++)                        // wyznaczenie tablicy LUT
        tablica_LUT[i] = (255 * tablica_dystrybuanta[i])/liczba_pixeli;

for(x=0;x<w;x++)                          // zaaplikowanie tablicy LUT
        for(y=0;y<h;y++)
                setPixel(x, y, tablica_LUT[ getPixel(x, y) ] );
```

```
[…]

tablica_histogram = newArray(256);

Plot.create("Histogram", "jasnosc", "czestotliwosc");

for(x=0;x<w;x++)
          for(y=0;y<h;y++)
                    tablica_histogram[ getPixel(x, y) ] +=1;

Plot.setColor("Blue");
Plot.add("line", tablica_histogram);

tablica_dystrybuanta = newArray(256);

for(i=0;i<256;i++)
          for(j=0;j<=i;j++)
                    tablica_dystrybuanta[i] += tablica_histogram[j];

Plot.create("Dystrybuanta", "wejscie", "dystrybucja");
Plot.setColor("Red");
Plot.add("line", tablica_dystrybuanta);

tablica_LUT = newArray(256);

for(i=0;i<256;i++)
          tablica_LUT[i] = (255 * tablica_dystrybuanta[i])/liczba_pixeli;

for(x=0;x<w;x++)
          for(y=0;y<h;y++)
                    setPixel(x, y, tablica_LUT[ getPixel(x, y) ] );

Plot.create("tablica LUT", "wejscie", "wyjscie");
Plot.setColor(„Green");
Plot.add("line", tablica_LUT);
```

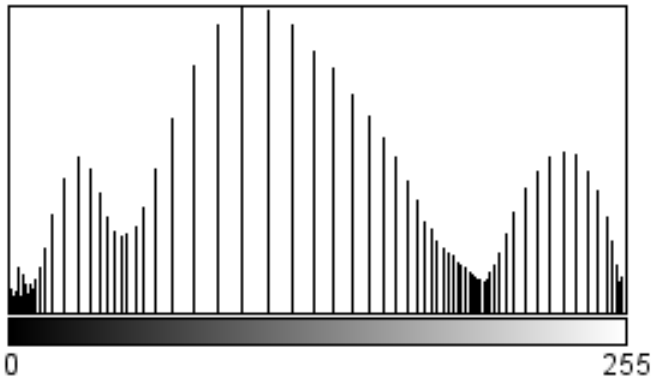






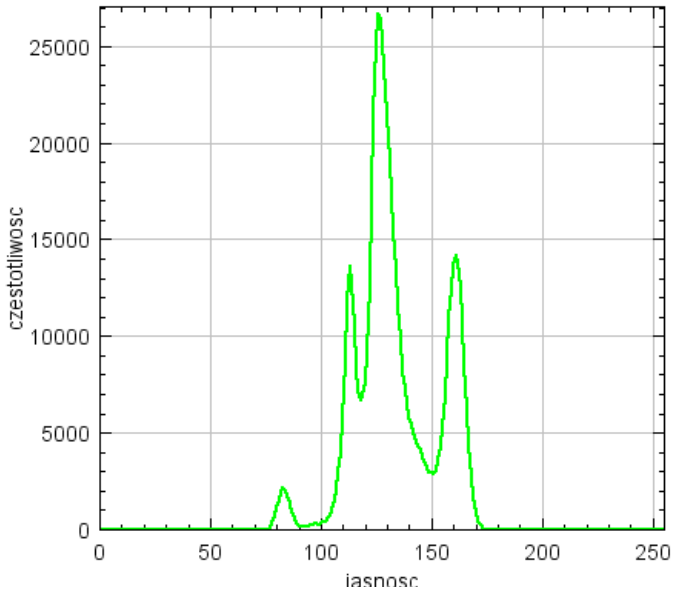
N: 640000
Mean: 133.403
StdDev: 18.372
Value: ---

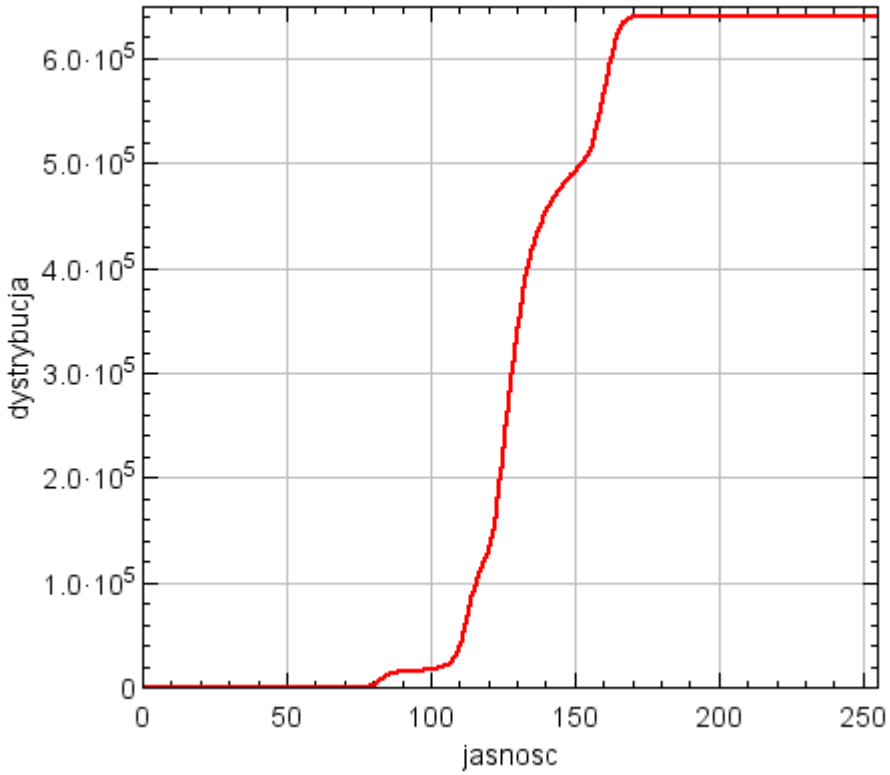
Min: 76
Max: 176
Mode: 126 (26647)
Count: ---

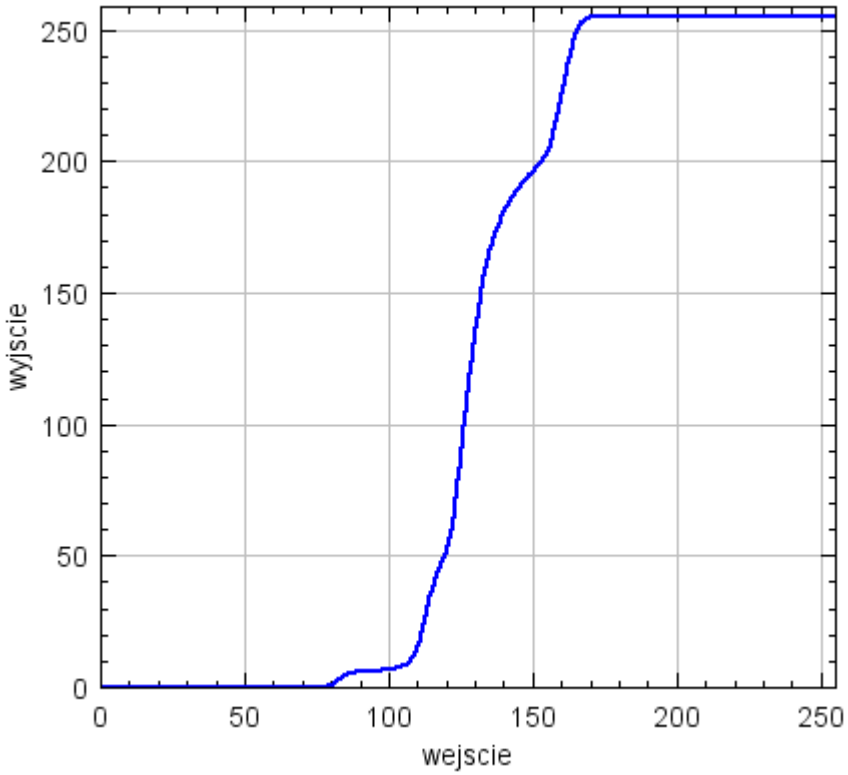


N: 640000
Mean: 129.791
StdDev: 73.342

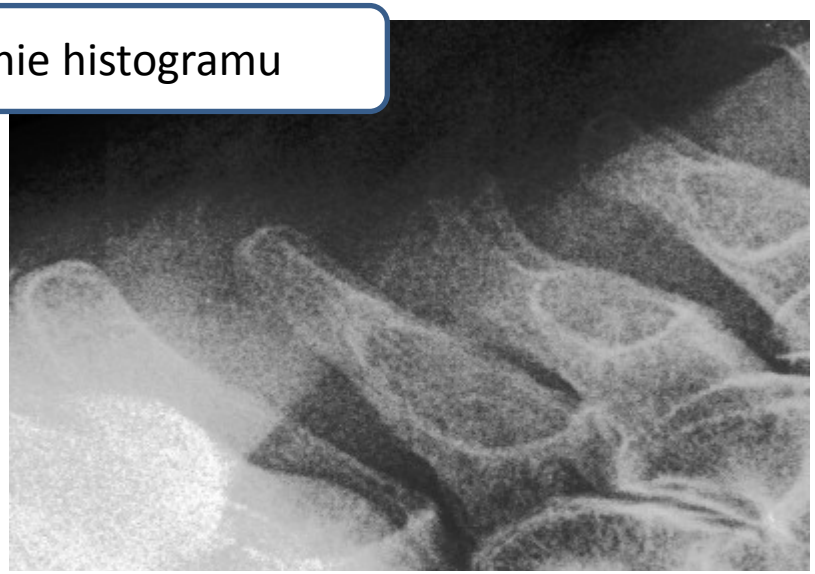
Min: 0
Max: 255
Mode: 96 (26647)

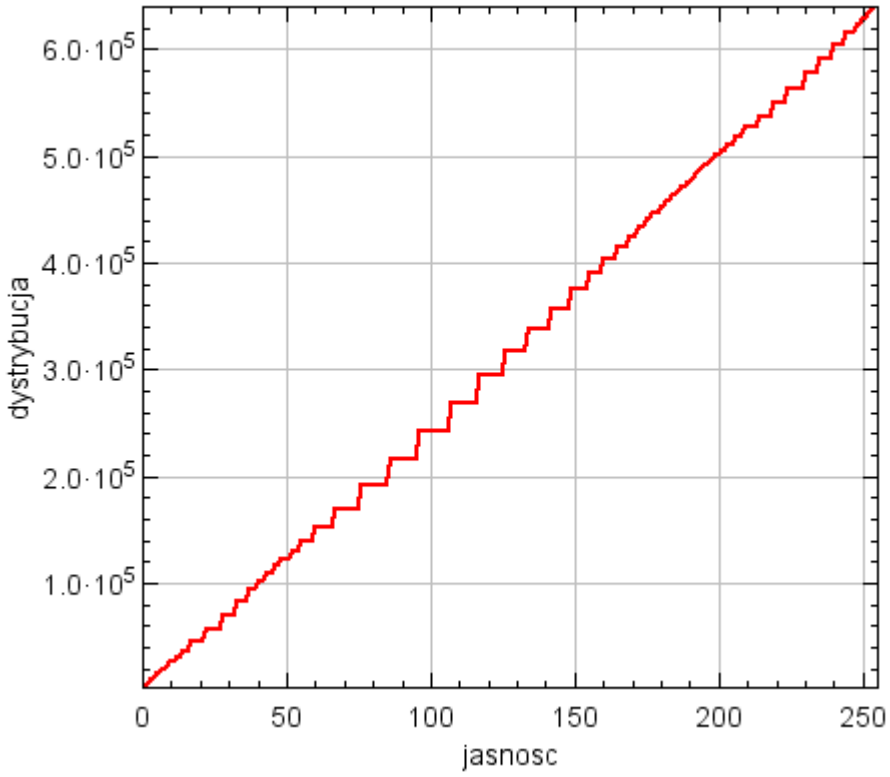

25000
20000
15000
10000
5000
0
czestotliwosc
0
50
100
150
200
250
jasnosc

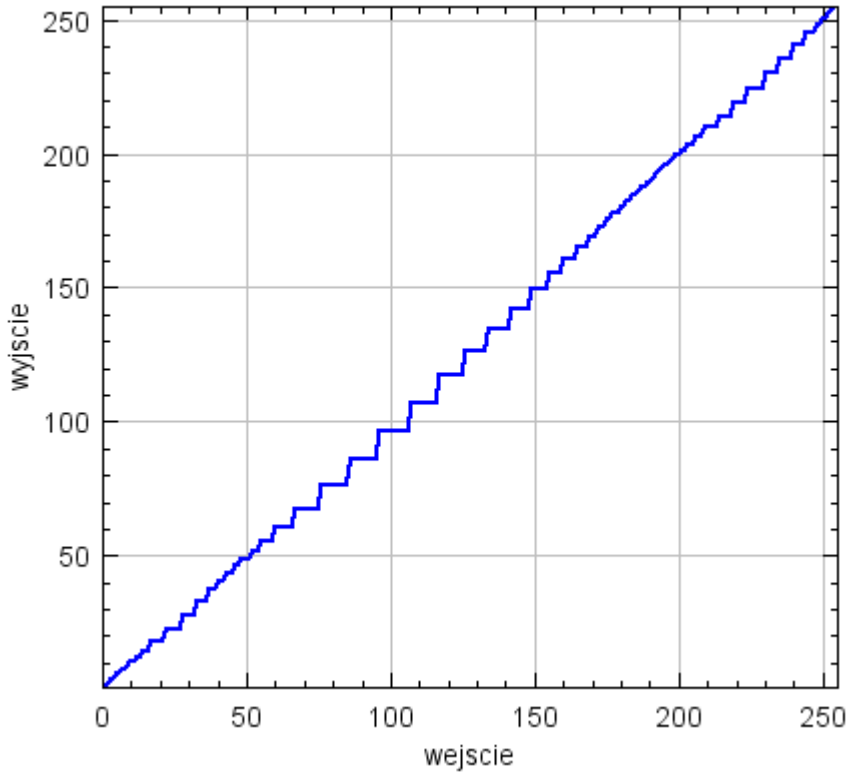

6.0·10^5
5.0·10^5
4.0·10^5
3.0·10^5
2.0·10^5
1.0·10^5
0
dystrybucja
0
50
100
150
200
250
jasnosc

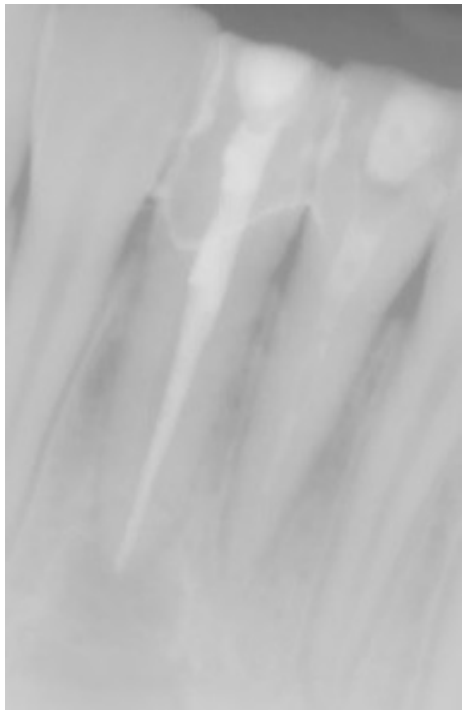

250
200
150
100
50
0
wyjscie
0
50
100
150
200
250
wejscie

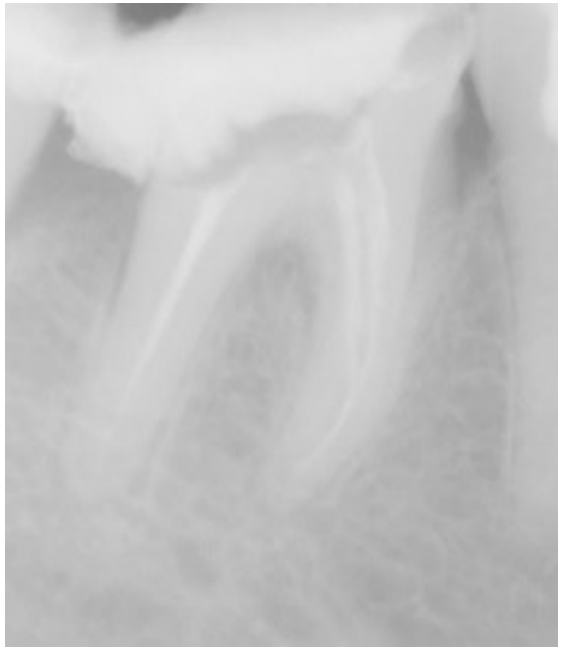
ponowne wyrównanie histogramu

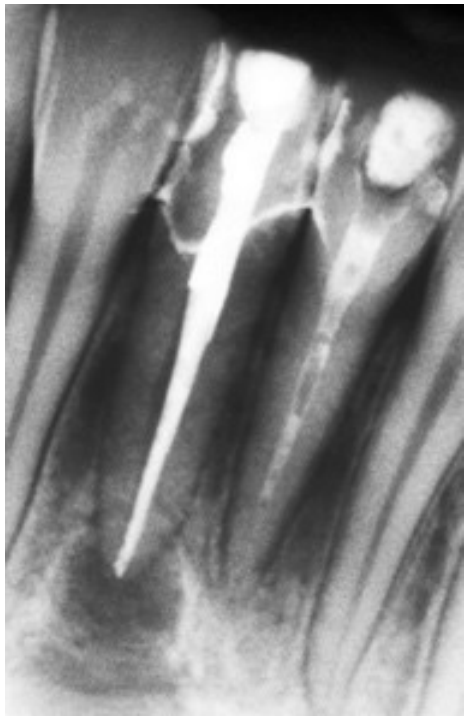


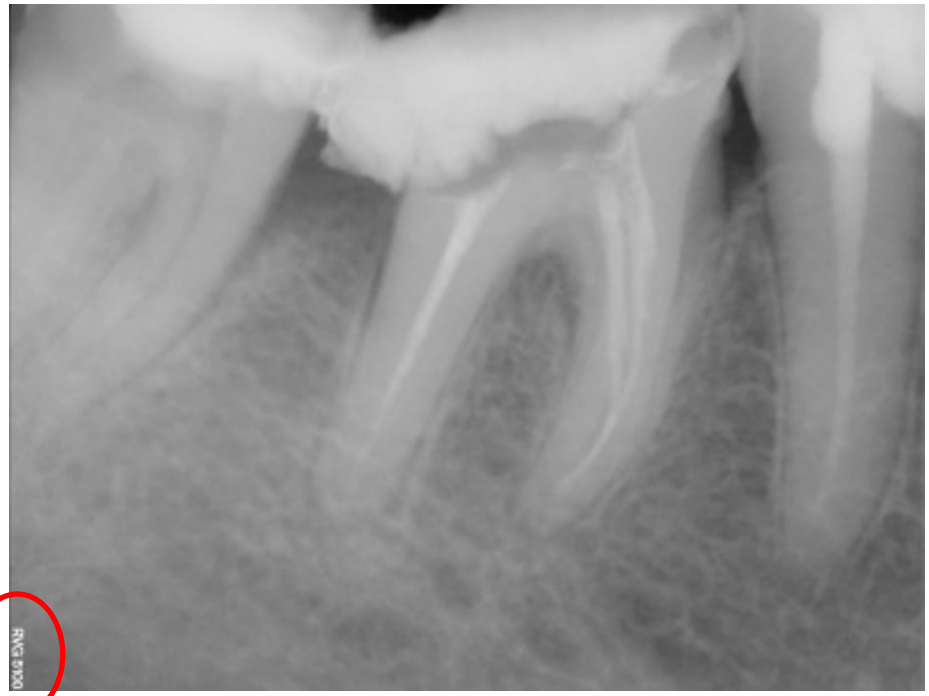






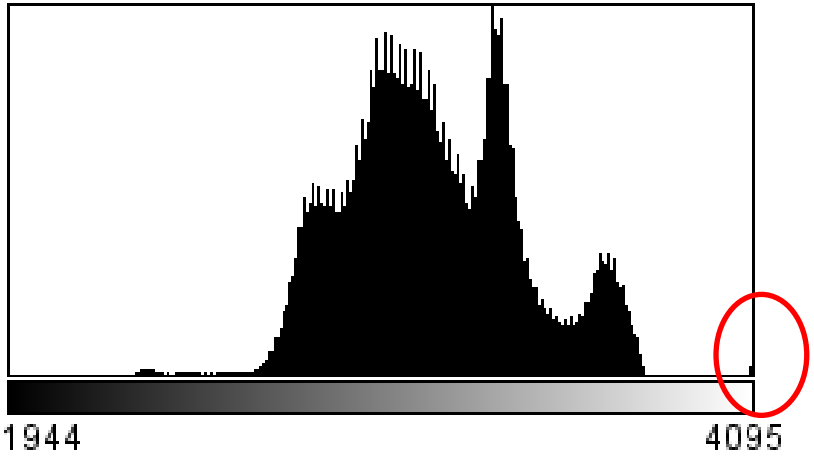






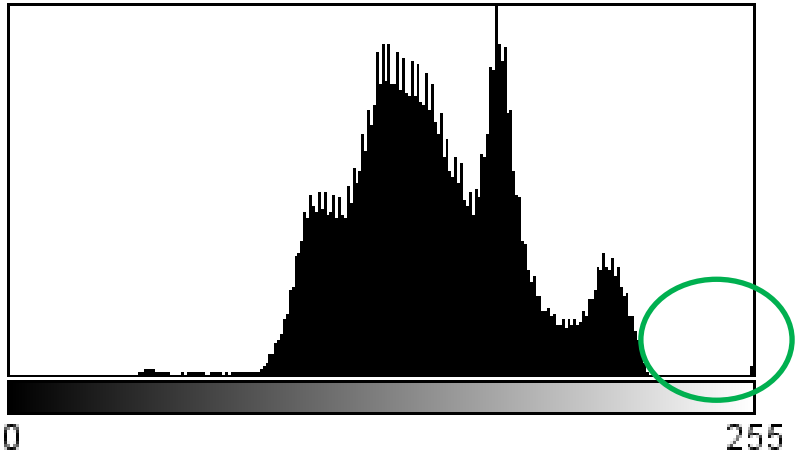
B08

B16



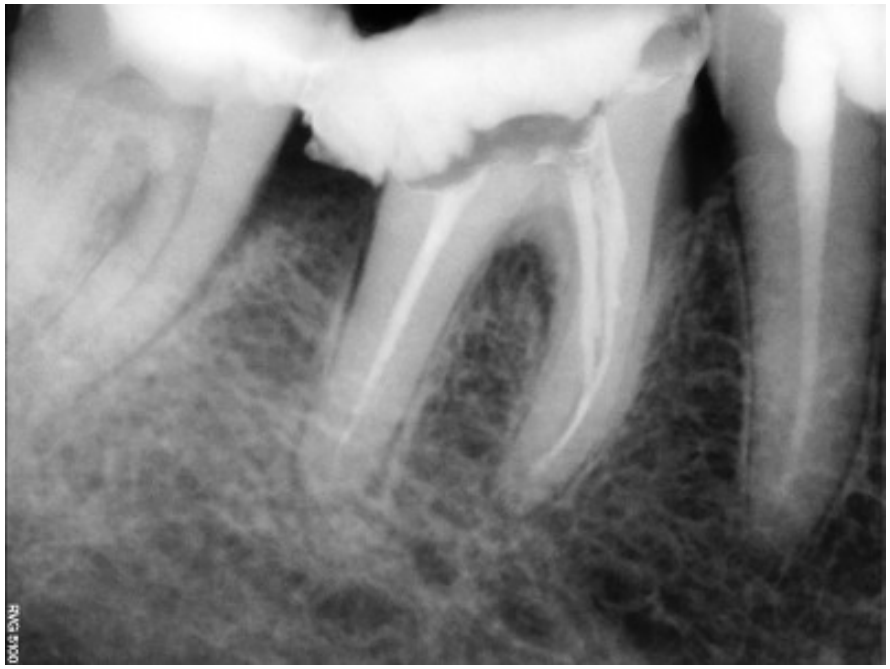
N: 1920000
Mean: 3168.500
StdDev: 258.242
Bins: 256
Value: 3767.309

Min: 1944
Max: 4095
Mode: 3348 (31153)
Bin Width: 8.402
Count: 1856

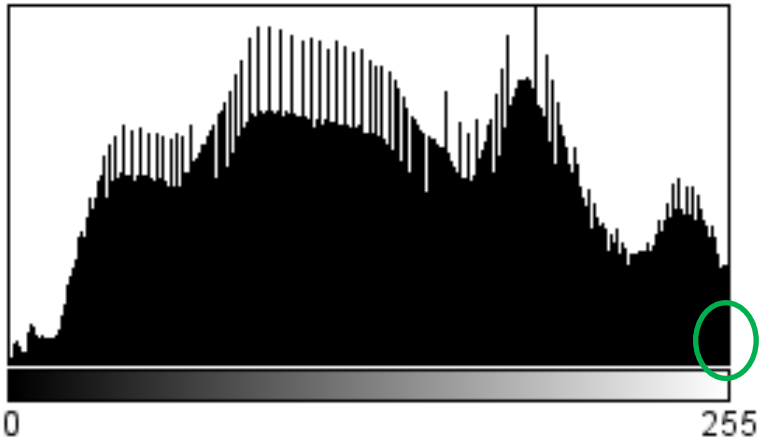


N: 1920000
Mean: 145.664
StdDev: 30.721
Value: ---

Min: 0
Max: 255
Mode: 167 (32281)
Count: ---

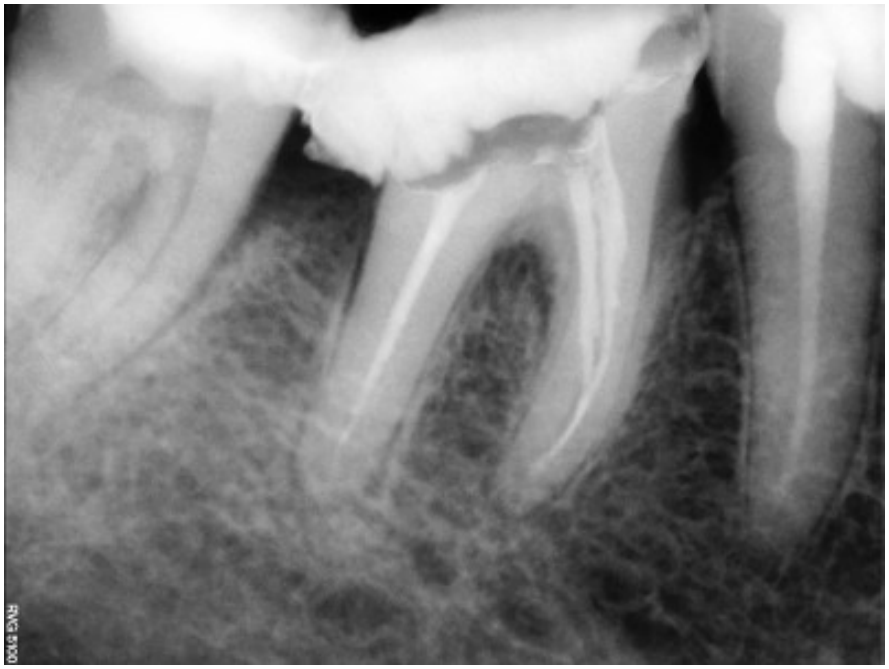


B16 -> HEQ -> B08

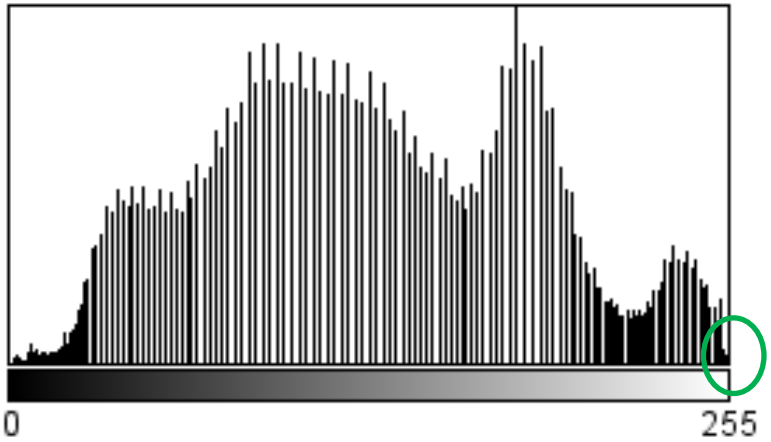


N: 1920000
Mean: 130.718
StdDev: 62.507

Min: 0
Max: 255
Mode: 187 (13636)

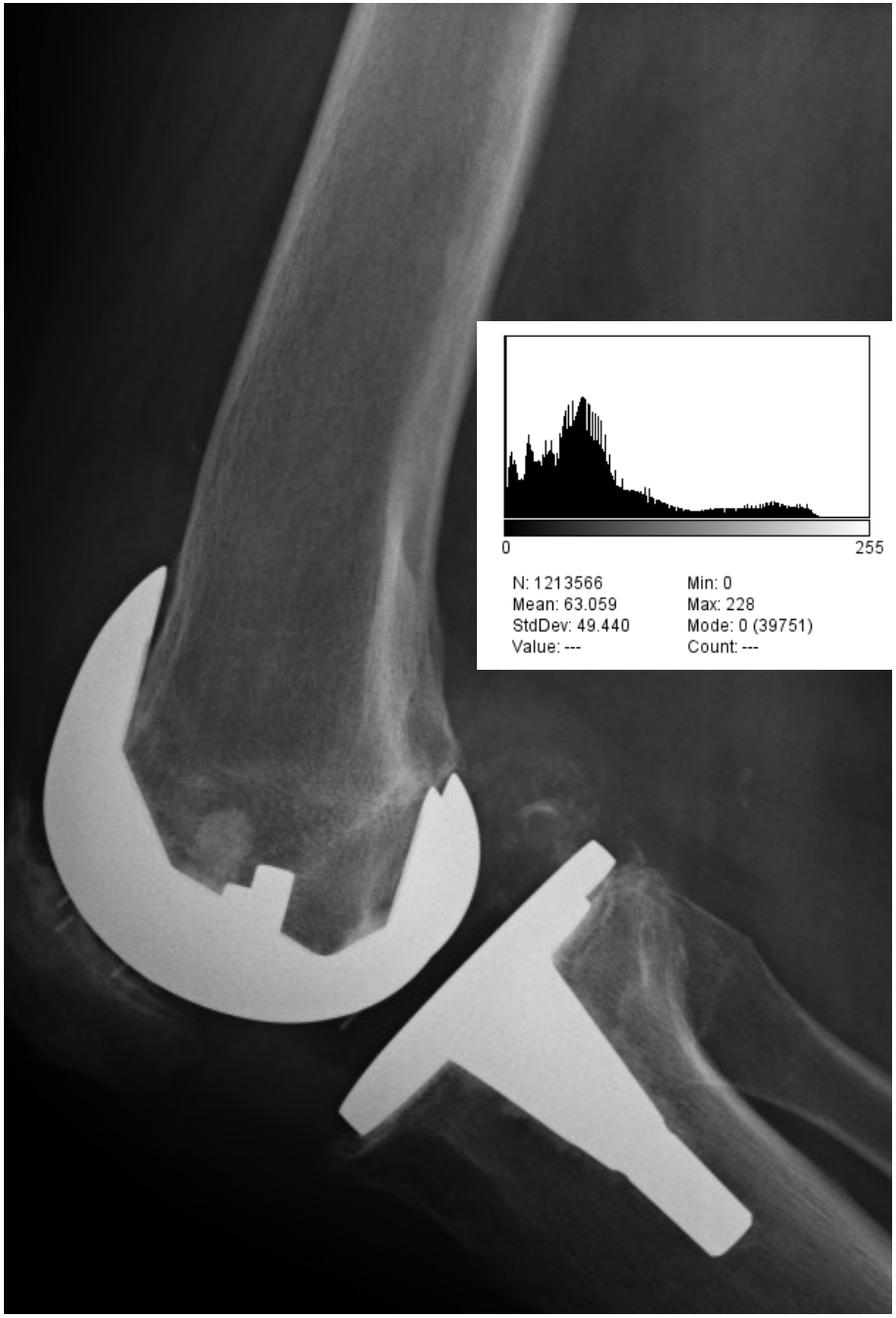


B16 -> B08 -> HEQ

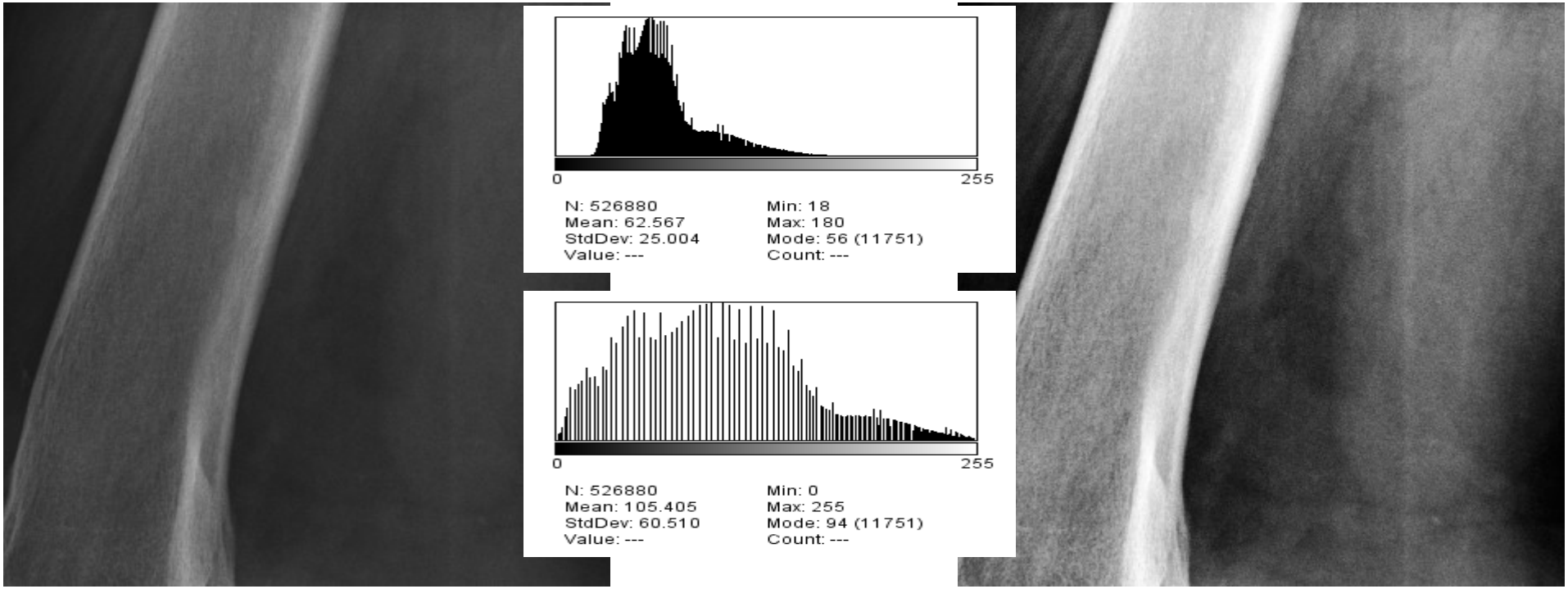


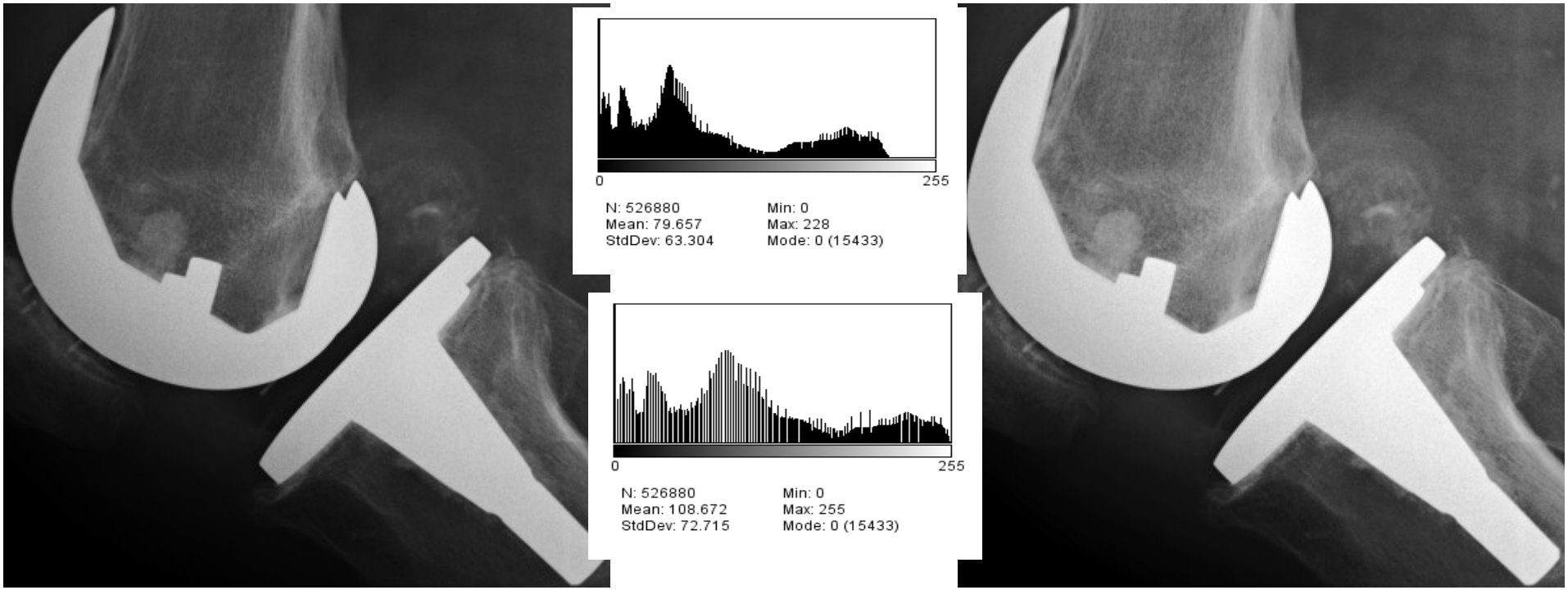
N: 1920000
Mean: 130.247
StdDev: 62.165

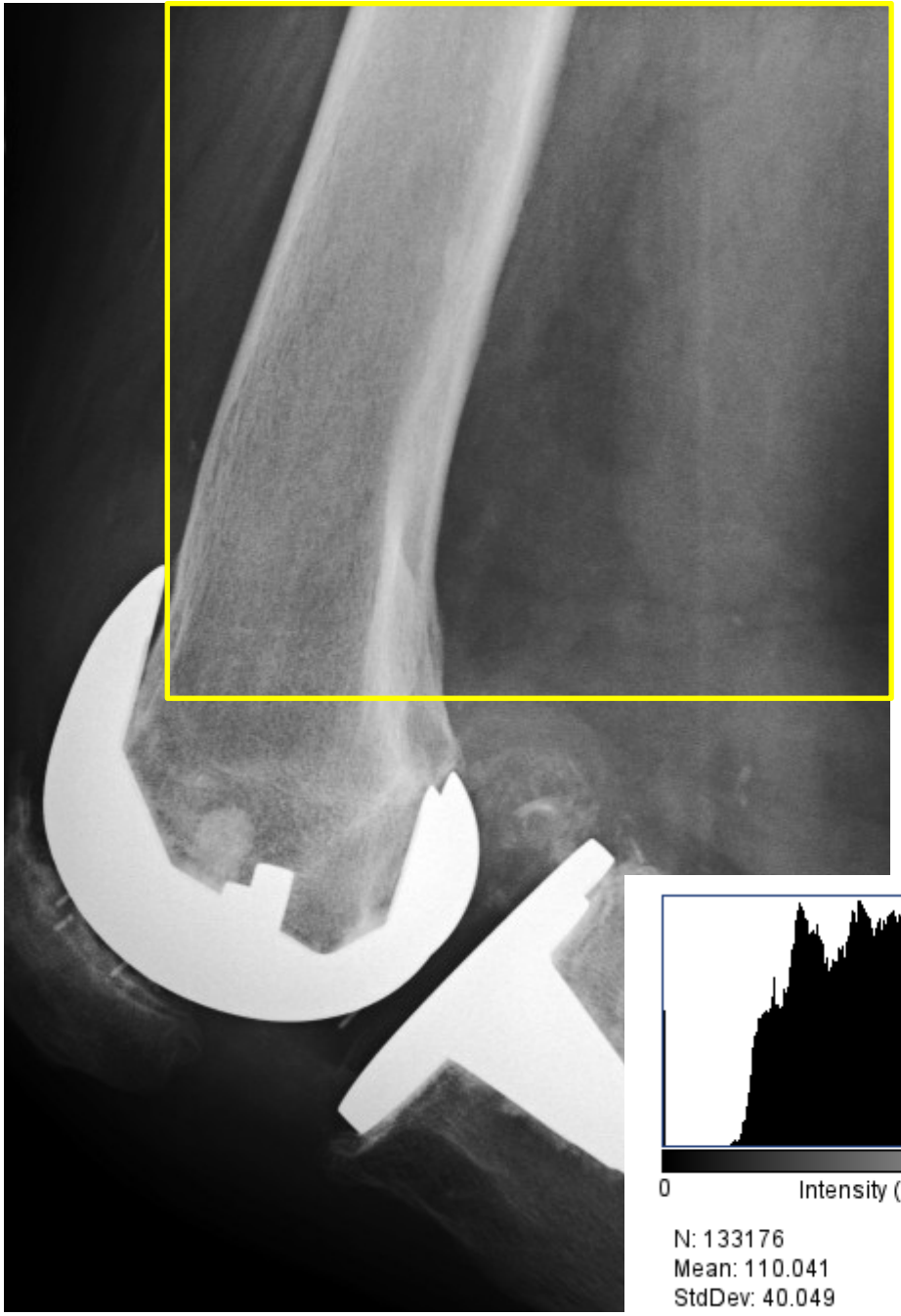
Min: 0
Max: 255
Mode: 180 (32281)

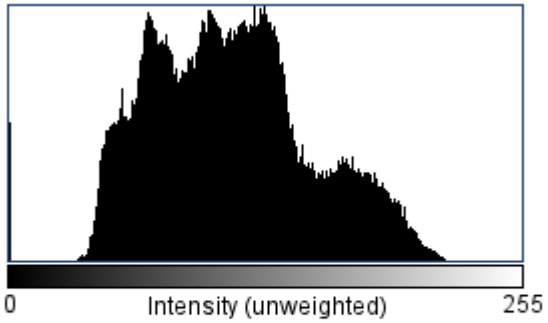

0
255
N: 1213566
Mean: 63.059
StdDev: 49.440
Value: ---
Min: 0
Max: 228
Mode: 0 (39751)
Count: ---


0
255
N: 1213566
Mean: 101.898
StdDev: 65.045
Value: ---
Min: 0
Max: 255
Mode: 0 (39751)
Count: ---


0
255
N: 526880
Mean: 62.567
StdDev: 25.004
Value: ---
Min: 18
Max: 180
Mode: 56 (11751)
Count: ---
0
255
N: 526880
Mean: 105.405
StdDev: 60.510
Value: ---
Min: 0
Max: 255
Mode: 94 (11751)
Count: ---

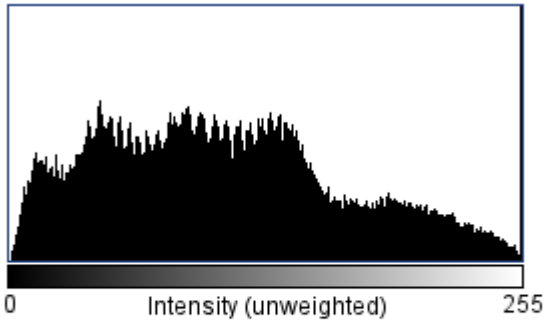

0
255
N: 526880
Mean: 79.657
StdDev: 63.304
Min: 0
Max: 228
Mode: 0 (15433)
0
255
N: 526880
Mean: 108.672
StdDev: 72.715
Min: 0
Max: 255
Mode: 0 (15433)





N: 133176
Mean: 110.041
StdDev: 40.049
Value: 180

Min: 0
Max: 220
Mode: 127 (1341)
Count: 452



N: 133176
Mean: 107.524
StdDev: 61.718
Value: ---

Min: 0
Max: 255
Mode: 255 (1463)
Count: ---