copyright: M. Dwornik
dwornik@agh.edu.pl

# Dyskretne transformaty obrazu

## 1 Dwuwymiarowa Szybka Transformata Fouriera

Dwuwymiarowa transformata Fouriera (FFT2) służy przejściu z domeny przestrzeni do domeny częstości. W tym przypadku częstotliwość (częstość) możemy definiować w formie dwóch częstotliwości ortogonalnych (częstotliwość pozioma i pionowa) lub jako częstotliwość wypadkową rozumianą jako odległość od środka obrazu transformaty (po przesunięciu widma ku środkowi).

Dwuwymiarowa Szybka Transformata Fouriera obrazu **I** dana jest wzorem:

$$F(nz,nx) = \beta_L \cdot \sum_{kz=0}^{Nz-1} \left[ \sum_{kx=0}^{Nx-1} I(kz,kx) \cdot exp\left(\frac{-2\pi i \cdot kx \cdot nx}{Nx}\right) \right] \cdot exp\left(\frac{-2\pi i \cdot kz \cdot nz}{Nz}\right) \quad (1)$$

Odwrotna transformata Fouriera dla $F_{Obrazu}$ **F** jest dana wzorem:

$$I(kz,kx) = \beta_F \cdot \sum_{nz=0}^{Nz-1} \left[ \sum_{nx=0}^{Nx-1} F(nz,nx) \cdot exp\left(\frac{2\pi i \cdot nx \cdot kx}{Nx}\right) \right] \cdot exp\left(\frac{2\pi i \cdot nz \cdot kz}{Nz}\right) \quad (2)$$

gdzie $\beta_L \cdot \beta_F = \frac{1}{M \cdot N}$. Najczęściej $\beta_L = 1$ i $\beta_F = \frac{1}{MN}$. $i = \sqrt{-1}$.

Transformacje te wykonywane są poleceniem: `F=fft2(I)` oraz `I=ifft2(F)` odpowiednio dla prostej i odwrotnej transformaty.

FFT2 wykonywana jest dwuetapowo. Najpierw dokonywany jest szereg transformat jednowymiarowych (np. w wierszach), a następnie dla ortogonalnego wymiaru wykonywane są transformaty na wynikach otrzymanych w poprzednim kroku. Z tego powodu poniższe polecenia MatLABa są równoważne:
`fft2(I)==fft(fft(I).').';`

## 2 Widma FFT2

Dla dwuwymiarowej FFT, podobnie jak dla FFT 1D możemy policzyć widma amplitudowe (WA) i fazowe (WF). Dane są one wzorami:

$$WA(m,n) = \sqrt{Im(F(m,n))^2 + Re(F(m,n))^2} \quad (3)$$

$$WF(m,n) = arctg\left(\frac{Im(F(m,n))}{Re(F(m,n))}\right) \quad (4)$$

Do policzenia ich służą polecenia MatLABa:
`WA=abs(F)` oraz `WF=angle(F)`.

Ponieważ energia F skupiona jest na brzegach (składowa zero-częstotliwościowa), niezbędne jest przesunięcie jej do centrum. Służy do tego para poleceń: `FS=fftshift(F)` oraz `F=ifftshift(FS)`.
Uwaga! Polecenia te są wykonywane tylko i wyłącznie na wynikach transformaty oraz nie są symetryczne, tzn. poniższe polecenia nie są równoważne:
`fftshift(fftshift(F))` $\neq$ `ifftshift(fftshift(F))`

copyright: M. Dwornik
dwornik@agh.edu.pl

# 3 Filtracja w domenie częstotliwości

Głównym zastosowaniem FFT2 jest filtracja obrazów. Polega to na zamianie kosztownej obliczeniowo operacji splotu na znacznie bardziej szybsze mnożenie widm. Podobnie jak w przypadku filtracji sygnałów jednowymiarowych, możemy podzielić filtry na:

- dolnoprzepustowe - przepuszczające dolne częstotliwości, reprezentowane na ogół poprzez trend na obrazie (np. stopniowe zmiany jasności tła);
- górnoprzepustowe - przepuszczające wysokie częstotliwości, reprezentujące na ogół krawędzie, szum i inne krótkookresowe zmiany jasności na obrazie;
- pasmoprzepustowe i pasmowycięciowe - przepuszczające lub wycinające pewien przedział częstotliwości;
- wycięciowe - eliminujące pojedynczą częstotliwość z obrazu (np. efekt pochodzący od sieci elektrycznej).

Filtracja obrazu I przy użyciu filtra M dana jest wzorem:

$$FF = ifft2(|WA(I).*WA(M)| \cdot e^{i(WF(I)+WF(M))}) \tag{5}$$

Wynika z tego rozmiar maski filtru musi być taki sam jak obrazu. Istnieją dwie metody projektowania filtru: bezpośrednio w 2D oraz poprzez rozszerzenie filtrów 1D w 2D.

## 3.1 Projektowanie filtrów 2D

Idealny, zerofazowy filtr dolnoprzeprzepustowy ma postać centralnie położonego okręgu (o wartościach 1), gdzie długość promienia koła określa pasmo przepuszczania.

Innym przykładem filtru 2D jest dolnoprzepustowy filtr Butterwortha opisany wzorem:

$$H(z,x) = \frac{1}{1+(D(z,x)/D_0)^{2n}} \tag{6}$$

$$D(z,x) = \sqrt{(x-Nx/2)^2+(z-Nz/2)^2}, \tag{7}$$

gdzie $D_0$ odpowiada za częstotliwość odcięcia, $n$ jest rzędem filtru i odpowiada za krzywiznę zbocza filtru, a $[Nz, Nx]$ - rozmiar obrazu.

# 4 Dyskretna Transformacja Cosinusowa (DCT)

Transformacja ta ma zastosowanie przede wszystkim do kompresji obrazów jpg oraz do konwersji mpeg. Para transformat cosinusowych dana jest wzorami:

$$B(p,q) = \alpha_p\alpha_q \sum_{m=0}^{M-1}\sum_{n=0}^{N-1} A_{mn} cos\frac{\pi(2m+1)p}{2M} cos\frac{\pi(2n+1)q}{2N} \tag{8}$$

$$A(m,n) = \sum_{p=0}^{M-1}\sum_{q=0}^{N-1} \alpha_p\alpha_q B_{pq} cos\frac{\pi(2m+1)p}{2M} cos\frac{\pi(2n+1)q}{2N} \tag{9}$$

gdzie:
$0 \leqslant p \leqslant M-1$, $0 \leqslant q \leqslant N-1$

copyright: M. Dwornik
dwornik@agh.edu.pl

$0 \leqslant m \leqslant M-1$, $0 \leqslant n \leqslant N-1$

$$\alpha_p = \begin{cases} \frac{1}{\sqrt{M}} & dla \quad p=0 \\ \frac{2}{\sqrt{M}} & dla \quad p \neq 0 \end{cases} \quad \alpha_q = \begin{cases} \frac{1}{\sqrt{N}} & dla \quad q=0 \\ \frac{2}{\sqrt{N}} & dla \quad q \neq 0 \end{cases}$$

Do implementacji DCT służy polecenie `dct2(obraz, m, n)`, a do transformacji odwrotnej `idct2(obraz, m, n)`. Jeżeli [m,n]<size(obraz), obraz A jest przycinany.

### 4.1 Kompresja JPEG '92

Kompresja JPEG'92 (ang. *Joint Photographic Experts Group*) oparta jest o transformację DCT2 wykonywaną w oknach o rozmiarze 8x8. Istotą procesu jest kwantyfikacja wyników transformaty oraz późniejsze kodowanie. Proces kompresji wygląda następująco:

1. Konwersja RGB do YCbCr połączona z przesunięciem średniej do 0.0:

$$\begin{aligned} Y &= 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B \\ Cr &= 128 - 0.168736 \cdot R - 0.331264 \cdot G + 0.53 \cdot B \\ Cb &= 128 + 0.5 \cdot R - 0.418688 \cdot G - 0.081312 \cdot B \end{aligned} \tag{10}$$

   W MatLABie istnieje polecenie konwertujące `rgb2ycbcr()`. Wymagane jest tylko późniejsze, ręczne przesunięcie średniej.
2. Downsampling (redukcja danych)
   - 4:4:4 - Brak downsamplingu
   - 4:2:2 - Redukcja poziomych składowych Cb i Cr o połowę
   - 4:2:0 - Redukcja poziomych i pionowych składowych barwy o połowę.
3. Wykonywanie DCT2 w oknach 8x8.
4. Kwantyfikacja wyników przy wykorzystaniu tablic kodowania $Q_Y$ i $Q_C$. Polega ona na podzieleniu każdego bloku przez tablicę i zaokrągleniu do najbliższej liczby całkowitej. Dopiero od tego punktu algorytm jest stratny.

$$Q_Y = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix} \quad Q_c = \begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 69 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix}$$

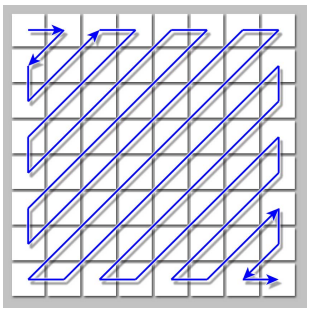
5. Zamiana tablicy na ciąg liczbowy. Wykorzystywana jest ścieżka zygzakowa, by jak najwięcej zer było koło siebie (fig.1).
6. Kompresja. Stosowany jest najczęściej algorytm Huffmana, choć stosowane też może być kodowanie arytmetyczne (bardziej efektywna kompresja, ale kosztowniejsza obliczeniowo).

## 5 Transformacja Radona

Transformacja Radona podaje ilość pikseli obrazu binarnego o wartości logicznej prawda w rzucie na prostą umieszczoną pod kątem theta względem obrazu. Transformacja ta dana jest wzorem:

$$R_o(x') = \int_{-\infty}^{\infty} f(x' \cos\theta - y' \sin\theta, x' \sin\theta + y' \cos\theta) dy \tag{11}$$

copyright: M. Dwornik
dwornik@agh.edu.pl

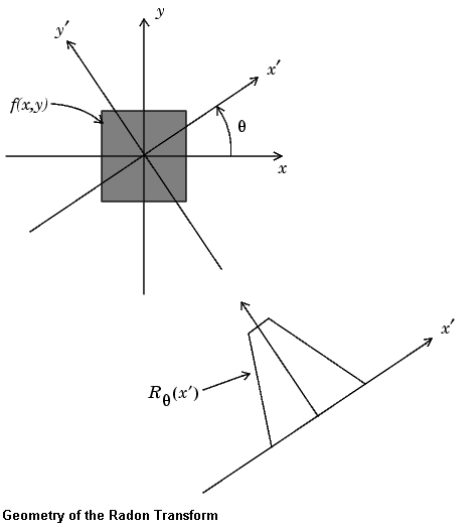


**Rysunek 1:** Ścieżka stosowana w kompresji JPG.Źródło *http://en.wikipedia.org/wiki/Zigzag*

gdzie:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Do transformacji Radona służy polecenie `[R,xp]=radon(obraz, theta)`, gdzie R - wynik transformacji Radona pod kątem theta (kiedy theta - wektor, R - macierz) xp - wektor zawierający współrzędne kątowe odnoszące się do macierzy R




**Rysunek 2:** Idea Transformacji Radona(Źródło: MATLAB help: Radon transform::description of)

Do transformacji odwrotnej służy polecenie `iradon(R, theta, parametry)`. Do parametrów zaliczamy:

- metoda interpolacji: 'nearest', '**linear**', 'spline'
- metoda filtracji: '**Ram-Lak**', 'Shepp-Logan', 'Cosine', 'Hamming', 'Hann'
- współczynnik skali osi częstotliwości: 0-1, domyślnie 1;
- rozmiar wyjściowy: domyślnie: $n = 2 \cdot floor(size(R,1)/(2 * \sqrt(2)))$

Odwrotna Transformata Radona, w wesji monochromatycznej, jest stosowana w tomografii medycznej do inwersji danych pomiarowych.

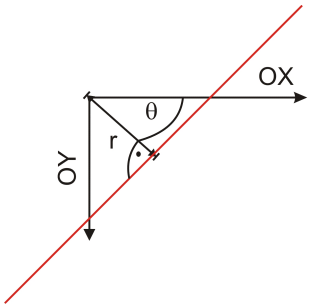
copyright: M. Dwornik
dwornik@agh.edu.pl

# 6 Transformacja Hougha

Standardowa Transformacja Hougha (w pakiecie MatLAB dostępna jest tylko dla obrazów logicznych), wykorzystuje parametryczną postać prostej:

$$\varrho = x \cdot \cos(\theta) + y \cdot \sin(\theta) \tag{12}$$

Na fig.3 przedstawiono interpretację geometryczną postaci parametrycznej prostej. Wartość $r$ jest najkrótszą odległością dowolnego punktu prostej od środka układu współrzędnych (czyli długością linii prostopadłej do prostej przechodzącej przez punkt (0,0)), a kąt $\theta$ jest kątem pomiędzy dodatnią półosią OX, a linią $r$.




**Rysunek 3:** Interpretacja geometryczna parametrycznej postaci prostej

Dzięki temu możemy każdemu punktowi przypisać pewną funkcję sinusoidalną. Linia jest reprezentowana jako punkt przecięcia sinusoid pochodzących od punktów leżących na owej linii. Do implementacji Transformacji Hougha służy polecenie: `[H, theta, rho]=hough(BW, parametry)`. Do parametrów zaliczamy:

- 'ThetaResolution' - Liczba rzeczywista z przedziału 0-90, określająca odległość pomiędzy kolejnymi kątami $\theta$, domyślnie 1;
- 'RhoResolution' - liczba z przedziału: $0 - norm(size(BW))$. Domyślnie 1.

Funkcjami powiązanymi z transformacją Hougha są:

- `peaks = houghpeaks(H, numpeaks, parametry)` - służy ona do lokalizacji punktów przecięć sinusoid na obrazie;
- `houghlines(BW, theta, rho, peaks)` - służy do detekcji linii prostych na obrazie logicznym.

# 7 Transformata Gabora

Inną transformatą czasowo-częstotliwościową jest transformata Gabora. Ma ona zastosowanie głównie do detekcji tekstur na obrazach. Polega ona na splocie zbioru kierunkowych masek (tzw. bank filtrów) z obrazem. Maska filtru ma postać zespoloną i stanowi złożenie dwóch elementów: sinusoidalnej fali nośnej 2D o określonej długości fali $\lambda$ i orientacji $\theta$ oraz dwuwymiarowej funkcji Gaussa o zadanym odchyleniu $\sigma$, współczynniku eliptyczności $\gamma$.

$$G(x, y, \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 \cdot y'^2}{2\sigma^2}\right) \cdot \exp\left(i\left(2\pi\frac{x'}{\lambda} + \psi\right)\right) \tag{13}$$

copyright: M. Dwornik
dwornik@agh.edu.pl

gdzie

$$\begin{cases} x' = x \cdot \cos\theta + y \cdot \sin\theta \\ y' = -x \cdot \sin\theta + y \cdot \cos\theta \end{cases} \tag{14}$$

W programie MatLAB transformację Gabora wykonuje się w dwóch krokach:

1. Tworzymy bank filtrów **g** korzystając z polecenia:
   `g = gabor(długość fali, kąty, 'SpatialFrequencyBand', 1.0, 'SpatialAspectRatio', 0.5);`
   Długość fali podaje w pikselach/okres, natomiast kąty (orientacje filtrów) w stopniach. Ilość filtrów w banku jest iloczynem ilości długości fal i ilości orientacji. Dwa ostatnie parametry są opcjonalne i podane z wartościami domyślnymi. Odpowiadają one za częstotliwość fali nośnej (sinusoidalnej) i za rozciągnięcie funkcji Gaussa (stosunek długości osi prostopadłej do kąta orientacji do długości osi zgodnej z kątem orientacji).

2. Filtrując obraz korzystając z polecenia:
   `Magn = imgaborfilt(obraz, bank filtrów);`
   W wyniku filtracji dostajemy zbiór obrazów odpowiadający każdemu z filtrów dostępnemu w banku (najpierw dla pierwszej orientacji wszystkie długości fali, potem dla każdej kolejnej orientacji odpowiadające im długości).