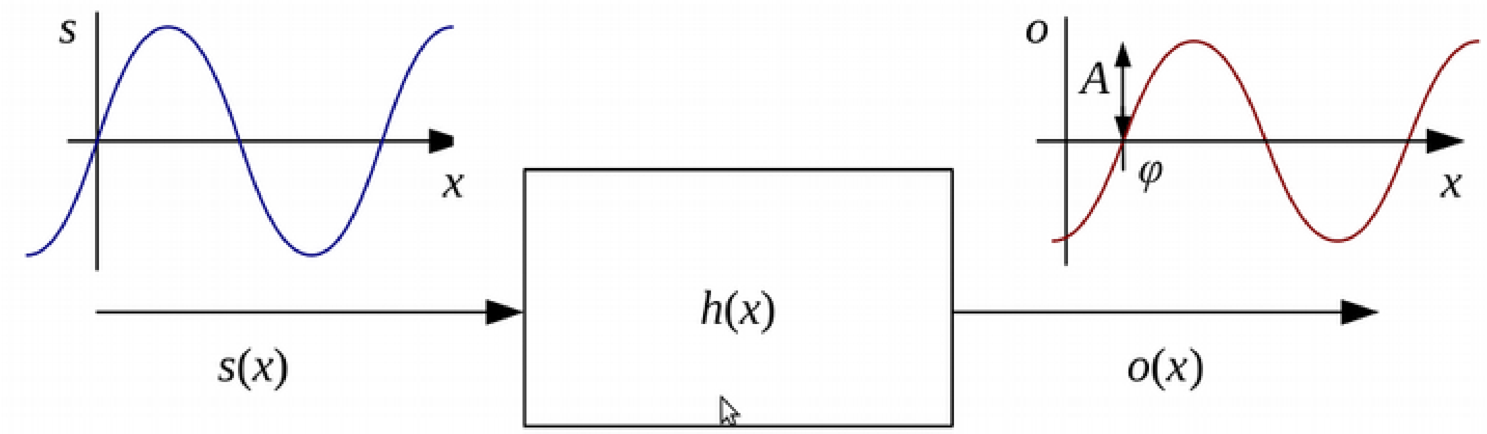
Wizja Maszynowa

Metody częstotliwościowe w przetwarzaniu obrazu - wykład 10

Adam Szmigielski
aszmigie@pjwstk.edu.pl
materiały: $ftp(public)://aszmigie/WM$

Transformata Fouriera (powtórzenie z kursu SYC)



- Transformacja Fouriera jako odpowiedź filtru $h(x)$ na sinusoidę wejściową $s(x) = e^{j\omega x}$ dając sinusoidę wyjściową $o(x) = h(x) \star s(x) = Ae^{j\omega x + \phi}$.
- Jeśli spleciemy sygnał sinusoidalny $s(x)$ z filtrem, którego odpowiedź impulsowa to $h(x)$, otrzymujemy kolejną sinusoidę o tej samej częstotliwości, ale różnej wielkości A i fazie ϕ_0,

$$o(x) = h(x) \star s(x) = A \sin(\omega x + \phi_0) = Ae^{j\omega x + \phi_0}$$

Transformacja Fouriera - definicja

Przekształcenie Fouriera to po prostu zestawienie amplitudy i fazy dla każdej częstotliwości,

$$H(\omega) = \mathcal{F}(h(x)) = Ae^{j\phi}$$

tj. jest odpowiedzią na złożoną sinusoidę o częstotliwości $omega$ przechodzącą przez filtr h (x). Para transformacji Fouriera jest również często zapisywana jako

$$h(x) \leftrightarrow^{\mathcal{F}} H(\omega)$$

Transformacja Fouriera istnieje zarówno w dziedzinie ciągłej,

$$H(\omega) = \int_{-\infty}^{\infty} h(x) \cdot e^{-j\omega x} dx$$

i w dziedzinie dyskretnej.

Dyskretna transformata Fouriera (DFT) **i** *Szybka transformata Fouriera (FFT)*

Dyskretna postać transformaty Fouriera jest znana jako *Dyskretna transformata Fouriera (DFT)*:

$$H(k) = \sum_{x=0}^{N-1} h(x) e^{-j\frac{2\pi kx}{N}}$$

gdzie N jest długością sygnału lub obszaru analizy. Te formuły mają zastosowanie zarówno do filtrów, takich jak $h(x)$, jak i do sygnałów lub obrazów, takich jak $s(x)$ lub $g(x)$.

- Formuła może być oceniana dla dowolnej wartości k, ma to sens tylko dla wartości z zakresu $k \in < -\frac{N}{2}, \frac{N}{2} >$. Jest to spowodowane większymi wartościami aliasu k przy niższych częstotliwościach,
- DFT wymaga operacji $O(N^2)$ do obliczenia. Na szczęście istnieje szybszy algorytm zwany Fast Fourier Transform (FFT), który wymaga tylko $O(N \log_2 N)$ - obejmuje serię $\log_2 N$ etapów.

Dwuwymiarowe transformaty Fouriera

Zamiast tylko określać częstotliwość poziomą lub pionową $omega_x$ lub $omega_y$, możemy utworzyć zorientowaną sinusoidę o częstotliwości $\{\omega_x, \omega_y\}$

$$s(x, y) = \sin(\omega_x x + \omega_y y).$$

Odpowiednie dwuwymiarowe transformaty Fouriera są wtedy

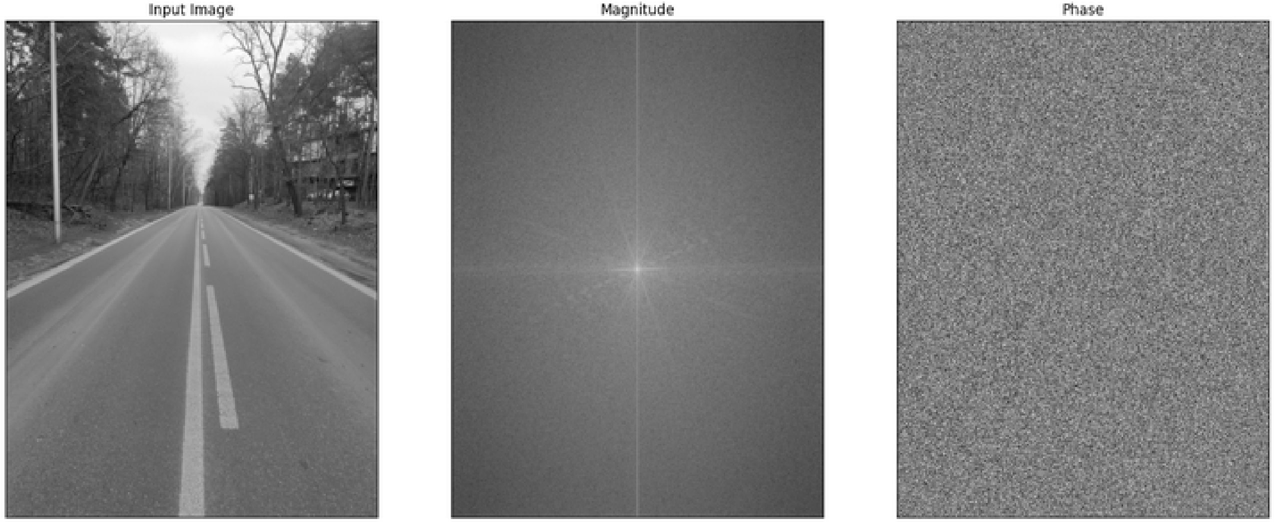
$$H(\omega_x, \omega_y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} h(x, y) \cdot e^{-j(\omega_x x + \omega_y y)} dx dy$$

a w dziedzinie dyskretnej,

$$H(k_x, k_y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} h(x, y) e^{-j2\pi \frac{k_x x + k_y y}{MN}}$$

gdzie M i N to szerokość i wysokość obrazu.

Transformacja obrazu - przykład

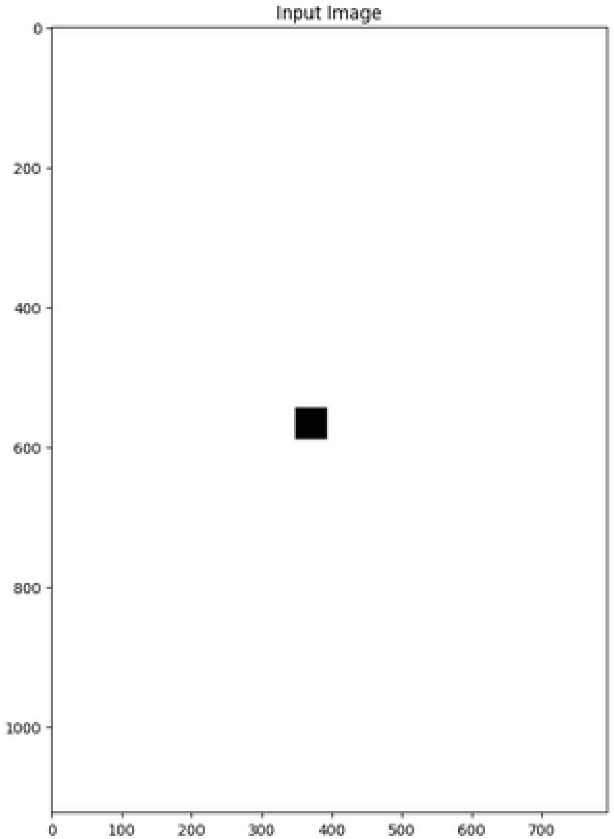


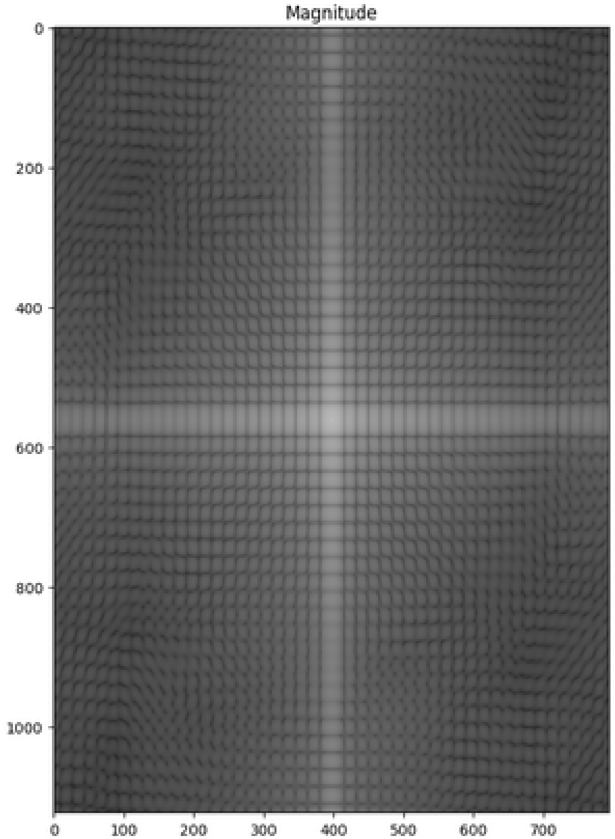

```
img = cv.imread('droga4.jpg',0)
f = np.fft.fft2(img)
fshift = np.fft.fftshift(f)
faza = np.arctan(np.abs(f.imag)/ f.real)
magnitude_spectrum = 20*np.log(np.abs(fshift))

plt.subplot(131),plt.imshow(img, cmap = 'gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(132),plt.imshow(magnitude_spectrum, cmap = 'gray')
plt.title('Magnitude'), plt.xticks([]), plt.yticks([])
plt.subplot(133),plt.imshow(faza, cmap = 'gray')
plt.title('Phase'), plt.xticks([]), plt.yticks([])
plt.show()
```

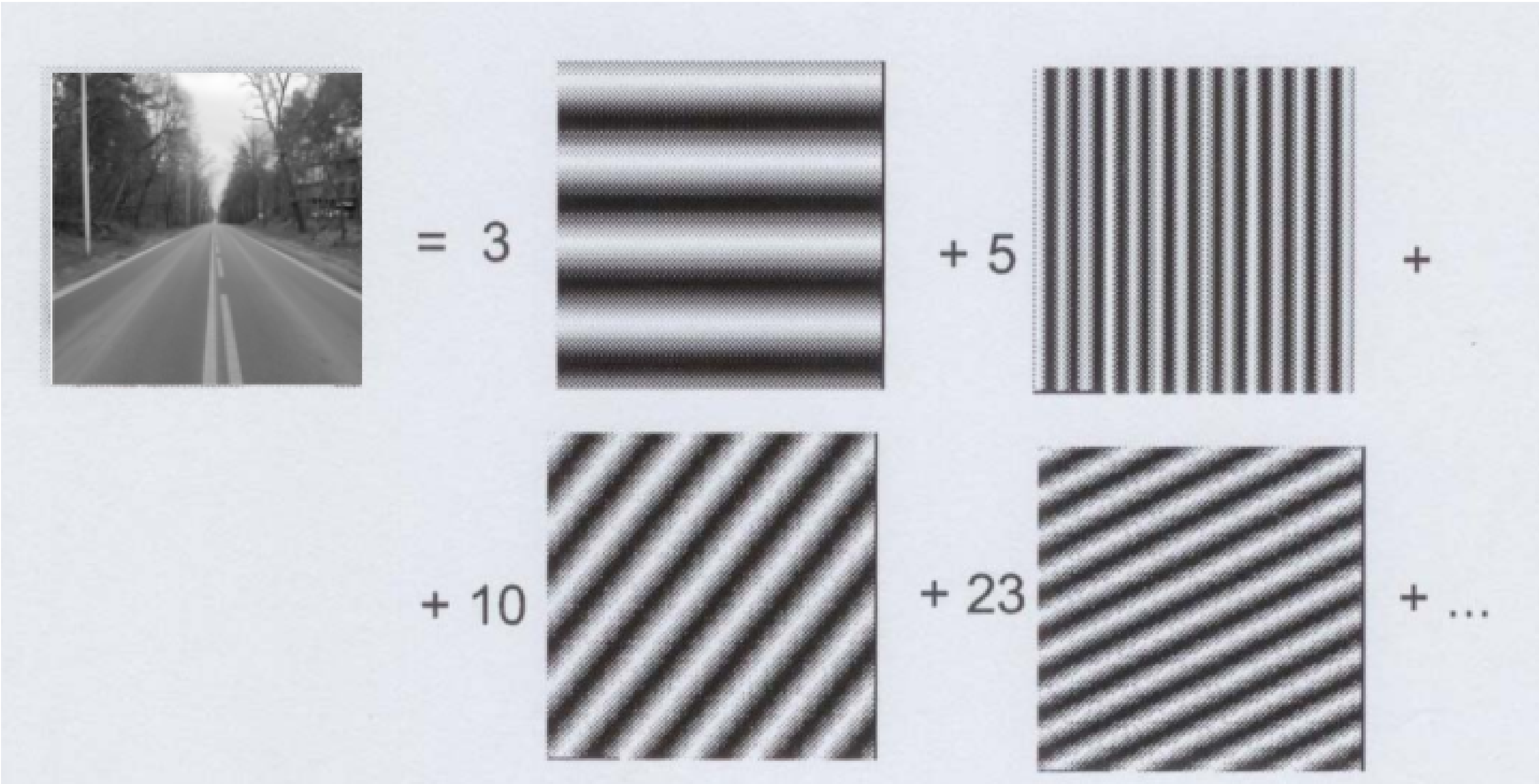
Transformacja obrazu - przykład

$$H(k_x, k_y) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} h(x,y) e^{-j2\pi \frac{k_x x + k_y y}{MN}}$$

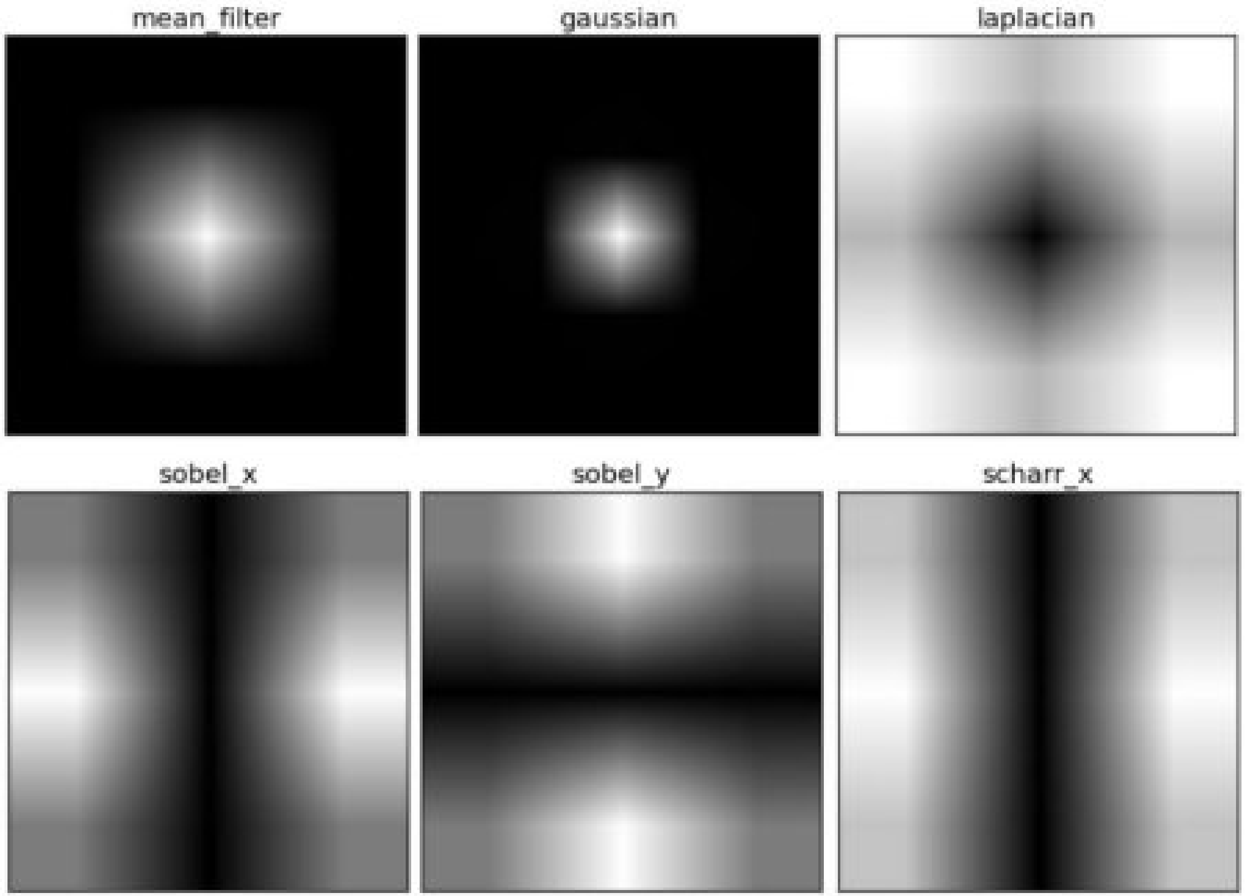




Dekompozycja obrazu

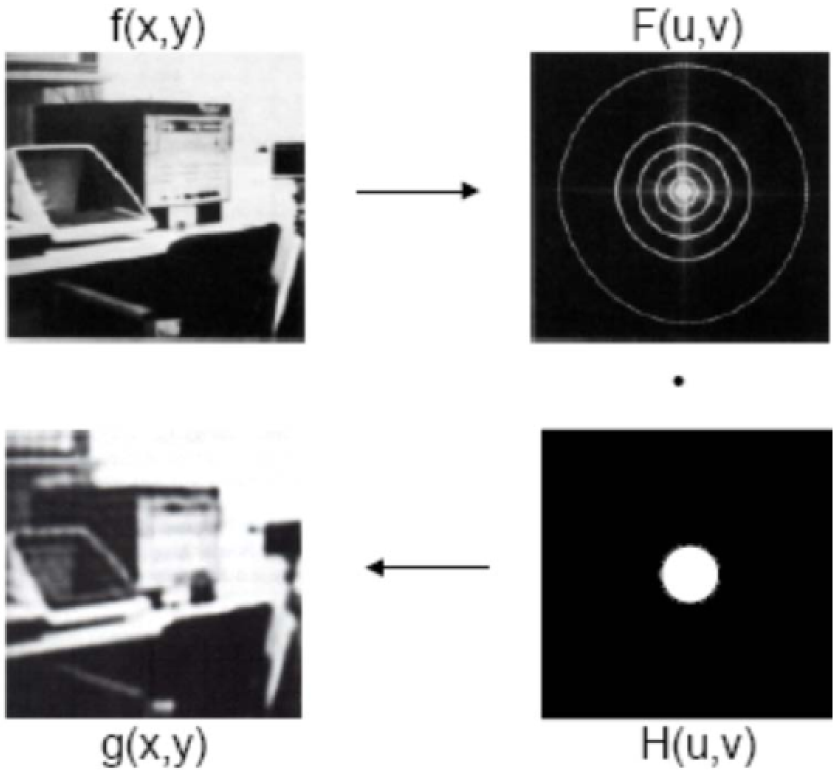


Widmo filtrów



https://docs.opencv.org/master/de/dbc/tutorial_py_fourier_transform.html

Filtrowanie Fourierowskie



spatial domain frequency domain

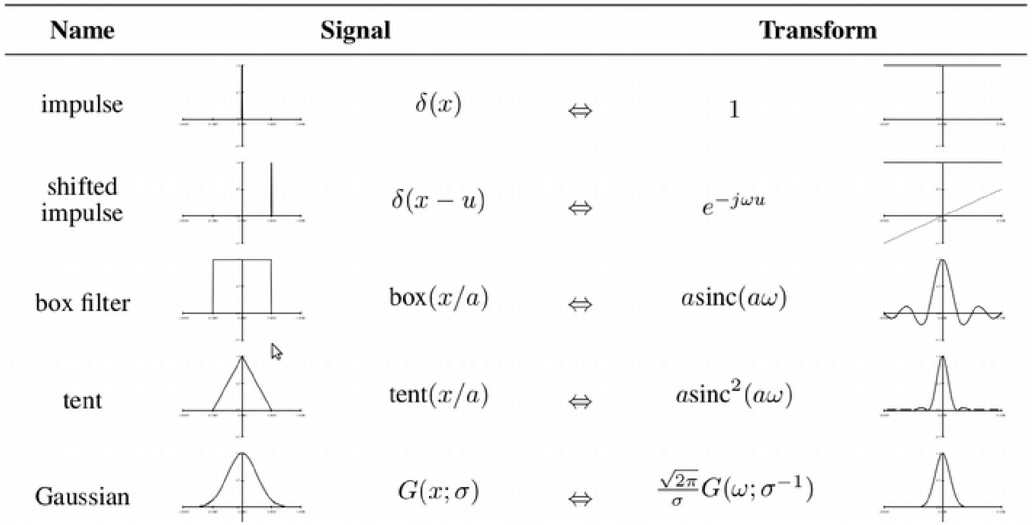
$f(x,y) \longrightarrow F(u,v)$

filter

$G(u,v) = F(u,v) \cdot H(u,v)$

$g(x,y) \longleftarrow G(u,v)$

Pary transformacji Fouriera



Name	Signal		Transform
impulse	$\delta(x)$	$\Leftrightarrow$	1
shifted impulse	$\delta(x-u)$	$\Leftrightarrow$	$e^{-j\omega u}$
box filter	$\mathrm{box}(x/a)$	$\Leftrightarrow$	$a\,\mathrm{sinc}(a\omega)$
tent	$\mathrm{tent}(x/a)$	$\Leftrightarrow$	$a\,\mathrm{sinc}^2(a\omega)$
Gaussian	$G(x;\sigma)$	$\Leftrightarrow$	$\frac{\sqrt{2\pi}}{\sigma}G(\omega;\sigma^{-1})$

- *Impulse*: Odpowiedź impulsowa ma stałą (wszystkie częstotliwości) transformację.
- *Przesunięty impuls*: Przesunięty impuls ma jednostkę wielkości i fazę liniową.
- *Filtr pudełkowy*: Filtr *box* (średnia ruchoma).

$$box(x) = \begin{cases} 1 & \text{if } |x| \le 1 \\ 0 & \text{else} \end{cases}$$

ma $sinc$ Transformata Fouriera,

$$sinc(\omega) = \frac{\sin(\omega)}{\omega}$$

który ma nieskończoną liczbę płatków bocznych. I odwrotnie, filtr $sinc$ jest idealnym filtrem dolnoprzepustowym. Dla prostokąta niejednostkowego szerokość prostokąta a i odstęp zerowy przecięcia w $sinc$ $\frac{1}{a}$ są odwrotnie proporcjonalne.

- *Tent*: Odcinkowo liniowa funkcja namiotu,

  $$tent(x) = \max\{0, 1 - |x|\},$$

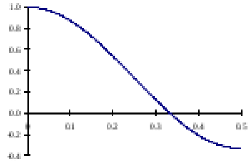
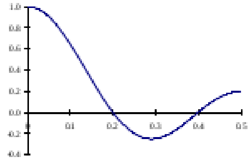
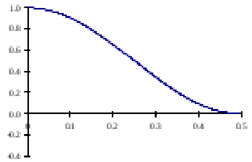
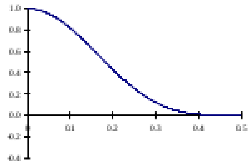
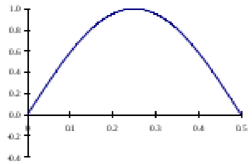
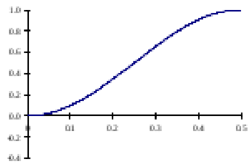
  ma $sinc^2$ transformatę Fouriera.

- *Gaussowski*: Gaussian (obszar jednostki) szerokości σ,

  $$G(x, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma^2}}$$

  ma (wysokość jednostkową) Gaussa z szerokością σ^{-1} jako transformatę Fouriera.

Transformaty Fouriera separowalnych kerneli

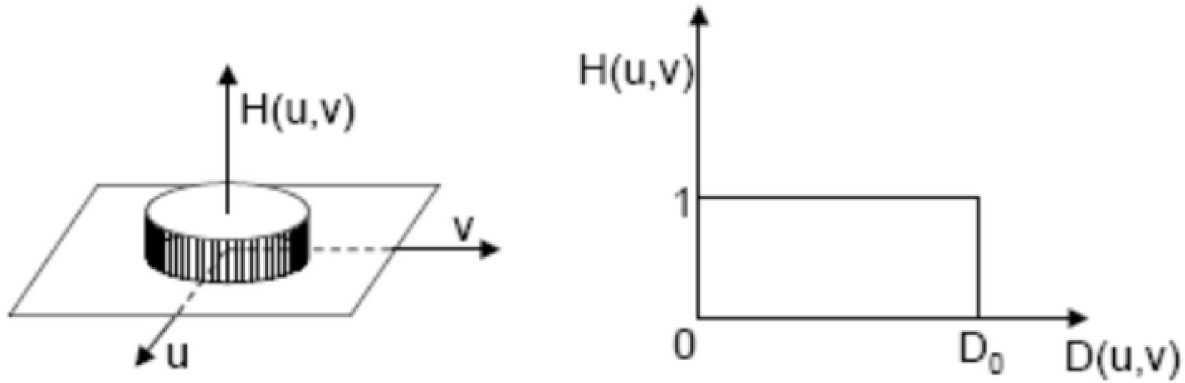
Name	Kernel	Transform	Plot
box-3	$\frac{1}{3}\begin{array}{\|c\|c\|c\|}\hline 1 & 1 & 1 \\ \hline\end{array}$	$\frac{1}{3}(1+2\cos\omega)$	
box-5	$\frac{1}{5}\begin{array}{\|c\|c\|c\|c\|c\|}\hline 1 & 1 & 1 & 1 & 1 \\ \hline\end{array}$	$\frac{1}{5}(1+2\cos\omega+2\cos 2\omega)$	
linear	$\frac{1}{4}\begin{array}{\|c\|c\|c\|}\hline 1 & 2 & 1 \\ \hline\end{array}$	$\frac{1}{2}(1+\cos\omega)$	
binomial	$\frac{1}{16}\begin{array}{\|c\|c\|c\|c\|c\|}\hline 1 & 4 & 6 & 4 & 1 \\ \hline\end{array}$	$\frac{1}{4}(1+\cos\omega)^2$	
Sobel	$\frac{1}{2}\begin{array}{\|c\|c\|c\|}\hline -1 & 0 & 1 \\ \hline\end{array}$	$\sin\omega$	
corner	$\frac{1}{2}\begin{array}{\|c\|c\|c\|}\hline -1 & 2 & -1 \\ \hline\end{array}$	$\frac{1}{2}(1-\cos\omega)$	

$H(u,v)$ – Idealny filtr dolnoprzepustowy

$$H(u,v) = \begin{cases} 1 & D(u,v) \leq D_0 \\ 0 & D(u,v) > D_0 \end{cases}$$

$$D(u,v) = \sqrt{u^2 + v^2}$$

D_0 = cut off frequency

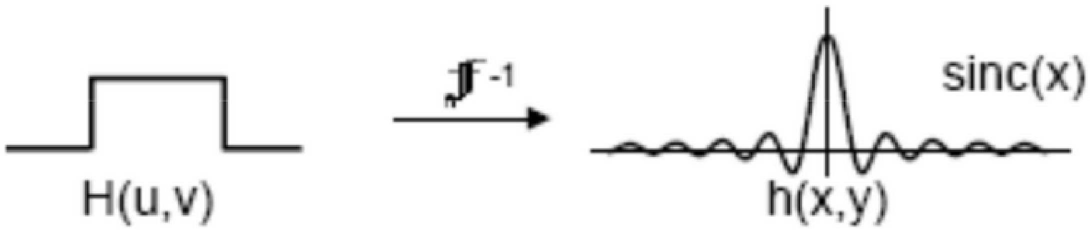


Problem z “dzwonieniem”

$$G(u,v) = F(u,v) \cdot H(u,v)$$

$$\Big\downarrow \text{Convolution Theorm}$$

$$g(x,y) = f(x,y) * h(x,y)$$



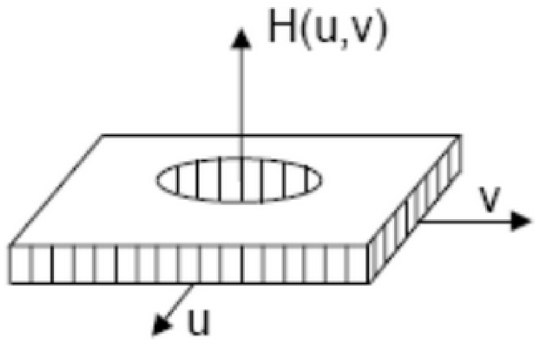
$\uparrow D_0 \longrightarrow \downarrow$ Ringing radius + blur

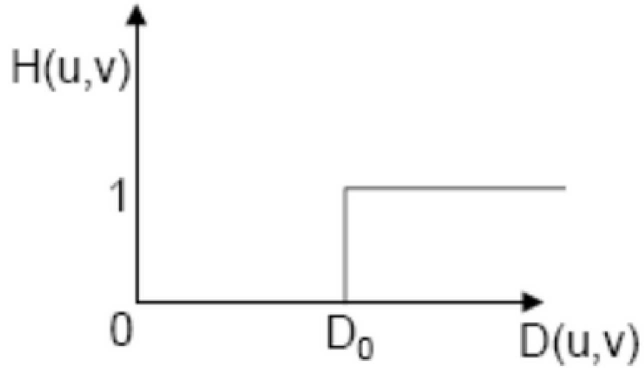
Wyostrzanie obrazu - filtr górnoprzepustowy

$$H(u,v) = \begin{cases} 0 & D(u,v) \le D_0 \\ 1 & D(u,v) > D_0 \end{cases}$$

$$D(u,v) = \sqrt{u^2 + v^2}$$

D_0 = cut off frequency





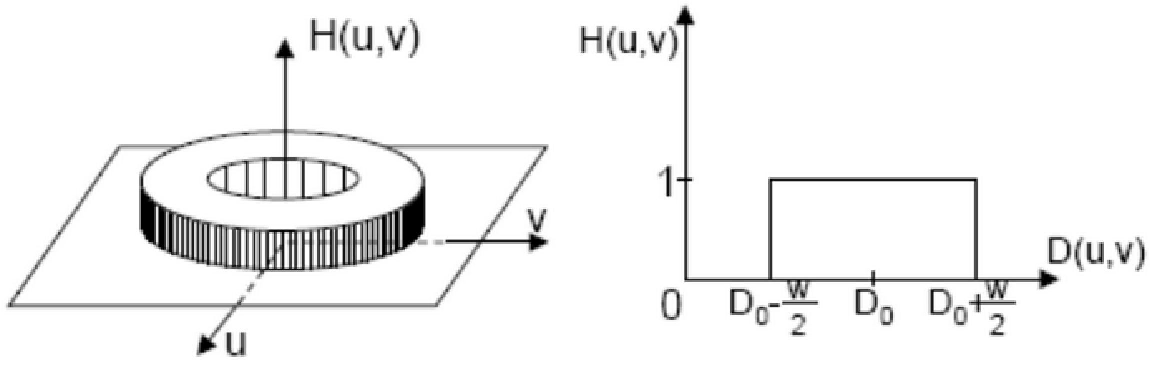
Filtrowanie średnio-pasmowe

$$H(u,v) = \begin{cases} 0 & D(u,v) \le D_0 - \frac{w}{2} \\ 1 & D_0 - \frac{w}{2} \le D(u,v) \le D_0 + \frac{w}{2} \\ 0 & D(u,v) > D_0 + \frac{w}{2} \end{cases}$$

$$D(u,v) = \sqrt{u^2 + v^2}$$

D_0 = cut off frequency

w = band width



Własności transformaty Fouriera $F(\omega) = \mathcal{F}\{f(x)\}$

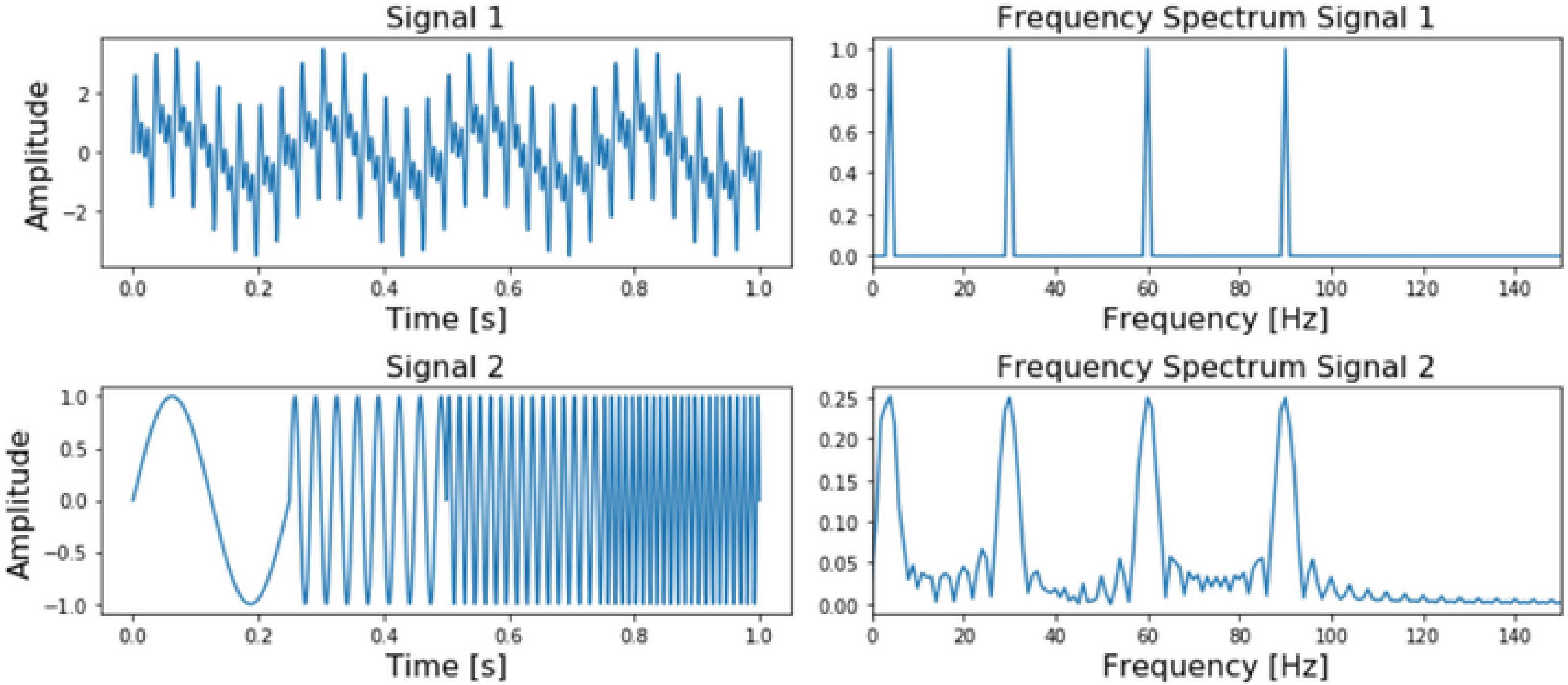
Property	Signal		Transform
superposition	$f_1(x) + f_2(x)$		$F_1(\omega) + F_2(\omega)$
shift	$f(x - x_0)$		$F(\omega)e^{-j\omega x_0}$
reversal	$f(-x)$		$F^*(\omega)$
convolution	$f(x) * h(x)$		$F(\omega)H(\omega)$
correlation	$f(x) \otimes h(x)$		$F(\omega)H^*(\omega)$
multiplication	$f(x)h(x)$		$F(\omega) * H(\omega)$
differentiation	$f'(x)$		$j\omega F(\omega)$
domain scaling	$f(ax)$		$1/aF(\omega/a)$
real images	$f(x) = f^*(x)$	$\Leftrightarrow$	$F(\omega) = F(-\omega)$
Parseval's Theorem	$\sum_x [f(x)]^2$	$=$	$\sum_\omega [F(\omega)]^2$

- *Superpozycja*: Transformacja Fouriera sumy sygnałów jest sumą ich transformacji Fouriera (transformata Fouriera jest operatorem liniowym).
- *Shift*: Transformacja Fouriera przesuniętego sygnału jest transformacją pierwotnego sygnału pomnożoną przez liniowe przesunięcie fazowe.
- *Reversal*: Transformata Fouriera odwróconego sygnału jest sprzężoną liczbą zespoloną transformacji sygnału.
- *Convolution*: Transformacja Fouriera pary splecionych sygnałów jest

iloczynem ich transformacji.

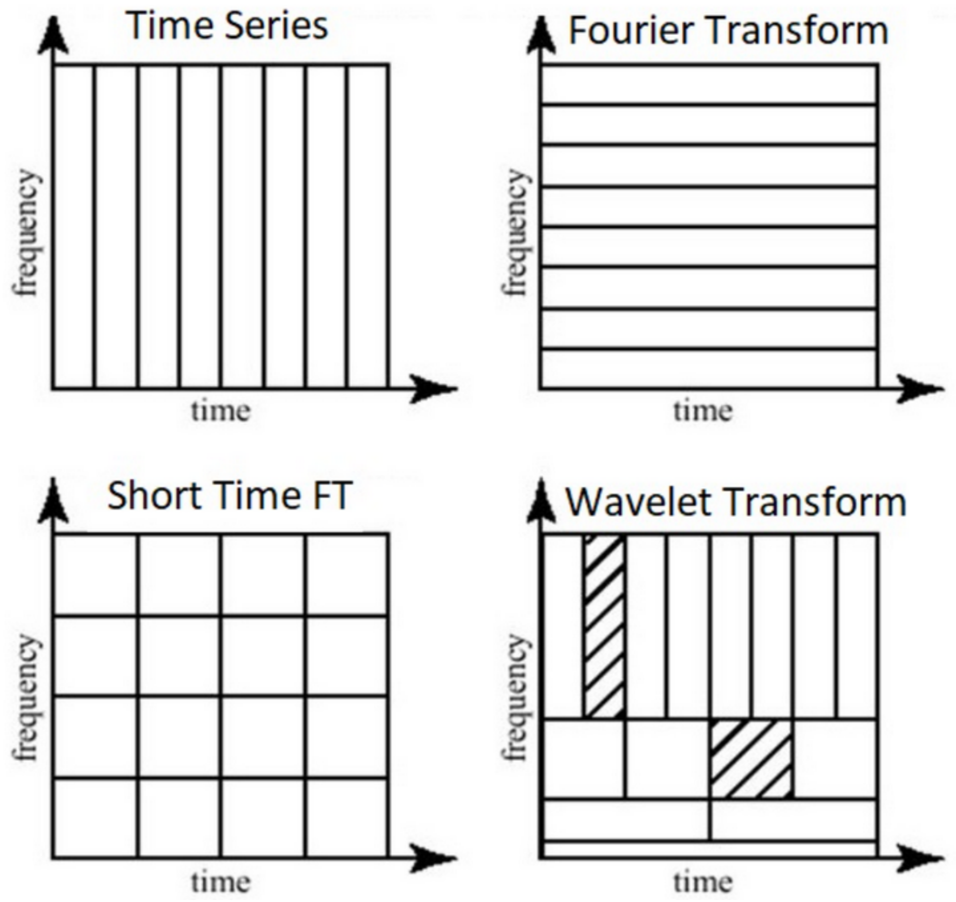
- *Korelacja*: Transformacja Fouriera korelacji jest iloczynem pierwszej transformacji razy zespolona sprzężona druga.
- *Multiplication*: Transformacja Fouriera iloczynu dwóch sygnałów jest splotem ich transformacji.
- *Differentiation*: Transformata Fouriera pochodnej sygnału to transformata sygnału pomnożona przez częstotliwość (różnicowanie liniowo uwydatnia (powiększa) wyższe częstotliwości)
- *Domain scaling*: Transformacja Fouriera rozciągniętego sygnału jest równoważnie skompresowaną (i przeskalowaną) wersją oryginalnej transformacji i odwrotnie.
- *Rzeczywiste obrazy*: Transformacja Fouriera sygnału o wartościach rzeczywistych jest symetryczna,
- *Twierdzenie Parsevala*: Energia (suma kwadratów wartości) sygnału jest taka sama jak energia jego transformaty Fouriera.

Opis przestrzenny i częstotliwościowy



- Góra - sygnał zawierający cztery różne częstotliwości (4 , 30, 60 $i 90 Hz$), które są obecne przez cały czas,
- Dół - te same cztery częstotliwości, tylko pierwsza występuje w pierwszej ćwiartce sygnału.

Krótkotrwała transformata Fouriera

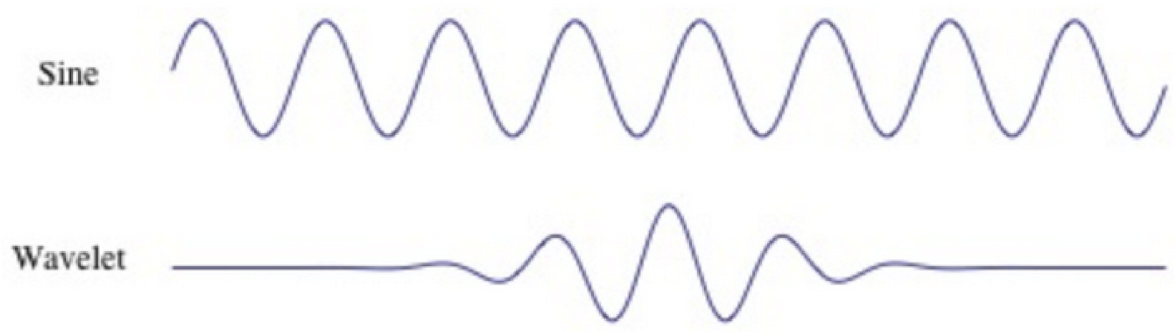


- Pierwotny sygnał jest dzielony na kilka części o jednakowej długości przy użyciu przesuwanego okna przed zastosowaniem transformaty Fouriera.

Granice transformaty Fouriera - zasada nieoznaczoności

- Im mniejszy rozmiar okna, tym więcej będziemy wiedzieć o tym, gdzie wystąpiła częstotliwość w sygnale, ale mniej o samej wartości częstotliwości.
- Im większy rozmiar okna, tym więcej będziemy wiedzieć o wartości częstotliwości, a mniej o czasie.

Transformacja falkowa

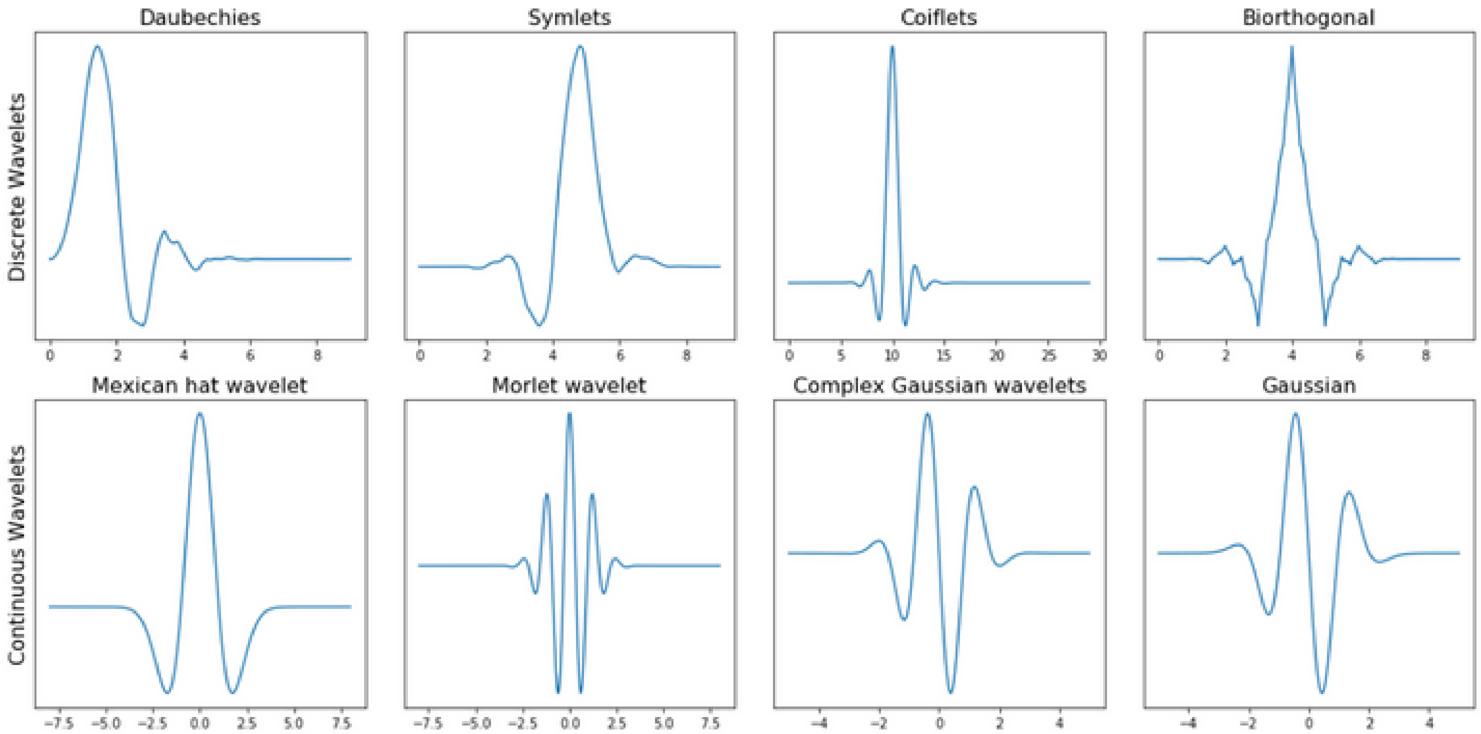


- **Transformata Fouriera** wykorzystuje szereg fal sinusoidalnych o różnych częstotliwościach do analizy sygnału.
- **Transformacja falkowa** używa szeregu funkcji zwanych falkami, z których każda ma inną skalę.
- Fala sinusoidalna jest nieskończenie długa, a Falka jest zlokalizowana w czasie.

Procedura falkowa

- Ponieważ falka jest zlokalizowana, możemy pomnożyć nasz sygnał za pomocą falki w różnych lokalizacjach,
- Zaczynamy od początku naszego sygnału i przesuwamy falkę w kierunku końca sygnału (splot),
- możemy go przeskalować tak, aby stał się większy i powtórzyć proces.

Rodziny falek



Ciągła transformacja falkowa

Ciągła transformacja falkowa jest opisana następującym równaniem:

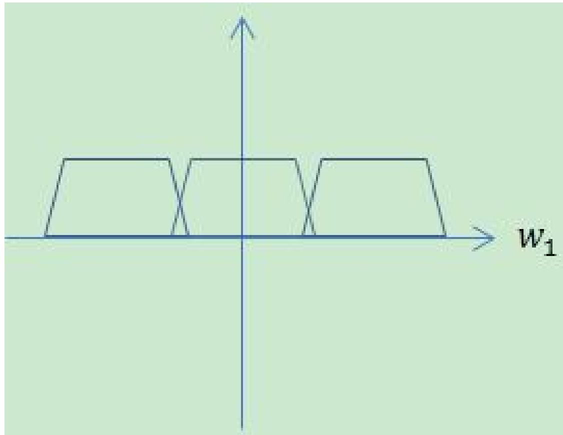
$$X_w(\tau, s) = \frac{1}{\sqrt{|s|}} \int_{-\infty}^{\infty} x(t)\psi(\frac{t-\tau}{s})dt$$

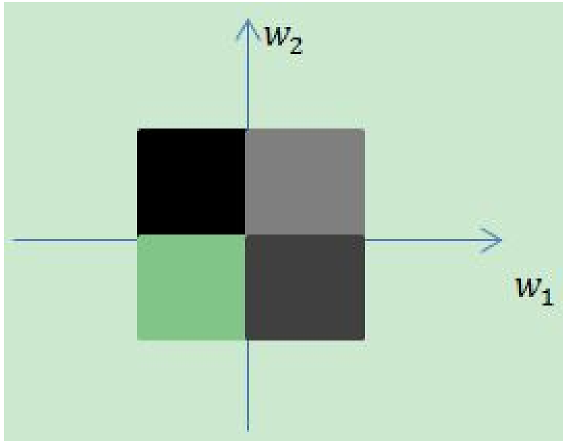
gdzie $\psi(t)$ jest ciągłą falką matki, która jest skalowana przez współczynnik s i przesunięta o współczynnik τ.

Falka musi spełniać:

- **Skończona energia** - iloczyn falki z sygnałem zawsze istnieje.
- **zerowa średnia** - można obliczyć odwrotność transformaty falkowej.

Dyskretna transformata falkowa jako bank filtrów.





- Dyskretna transformata falkowa jest zawsze implementowana jako bank filtrów,
- Oznacza to, że jest zaimplementowany jako kaskada filtrów górnoprzepustowych i dolnoprzepustowych.

Piramidy i falki

- Może służyć do zmniejszania rozmiaru obrazu (w celu przyspieszenia wykonania algorytmu lub zaoszczędzenia miejsca w pamięci lub czasu transmisji),
- Czasami nawet nie wiemy, jaka powinna być odpowiednia rozdzielczość obrazu.
- Ponieważ nie znamy skali, w jakiej pojawi się obiekt, musimy wygenerować całą piramidę obrazów o różnych rozmiarach i przeskanować każdy z nich w poszukiwaniu możliwego obiektu.

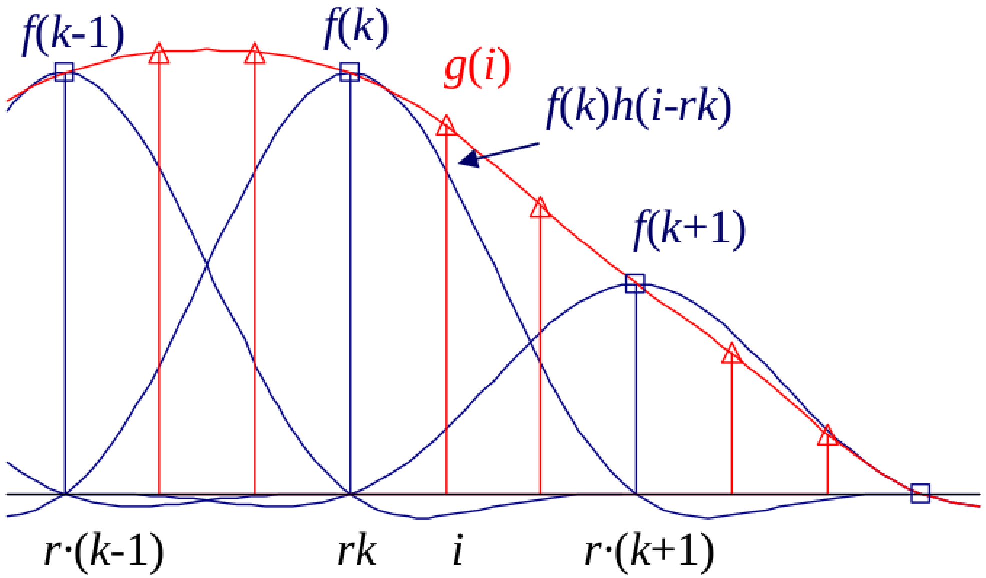
Interpolacja

Aby to interpolować (lub próbkować) obraz do wyższej rozdzielczości, musimy wybrać jądro interpolacji, za pomocą którego będziemy konwertować obraz,

$$g(i,j) = \sum_{k,l} f(k,l)h(i-rk, j-rl)$$

- Ta formuła jest związana z *dyskretnym splotem* z tym wyjątkiem, że zamieniamy k i l w $h()$ na rk i rl, gdzie r to *upsampling rate*

Interpolacja - przykład



- Interpolacja sygnału: $g(i) = \sum_k f(k)h(i - rk)$ - ważone sumowanie wartości wejściowych, gdzie $f(k)$ to próbki, a $h(i - rk)$ to kernel.
- Interpolację można postrzegać jako superpozycję kerneli interpolacji ważonej próbką, po jednym wyśrodkowanym na każdej próbce wejściowej k.
- Jakie kernele są dobrymi interpolatorami?

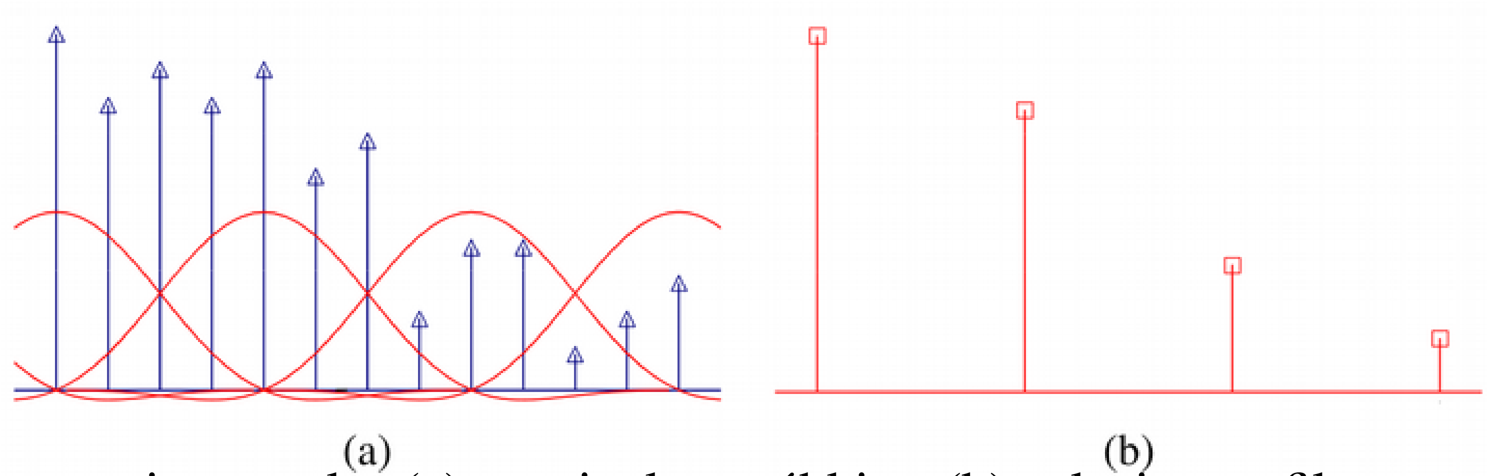
Decymacja (dziesiątkowanie)

Aby dokonać decymacji, najpierw splatamy obraz z filtrem dolnoprzepustowym (aby uniknąć aliasingu), a następnie zachowujemy każdą próbkę r.

$$g(i,j) = \sum_{k,l} f(k,l)h(ri-k, rj-l)$$

- W praktyce zwykle oceniamy splot tylko w każdej r próbce,
- O ile interpolacja może być użyta do zwiększenia rozdzielczości obrazu, to decymacja (próbkowanie w dół) jest wymagana w celu zmniejszenia rozdzielczości.

Decymacja - przykład

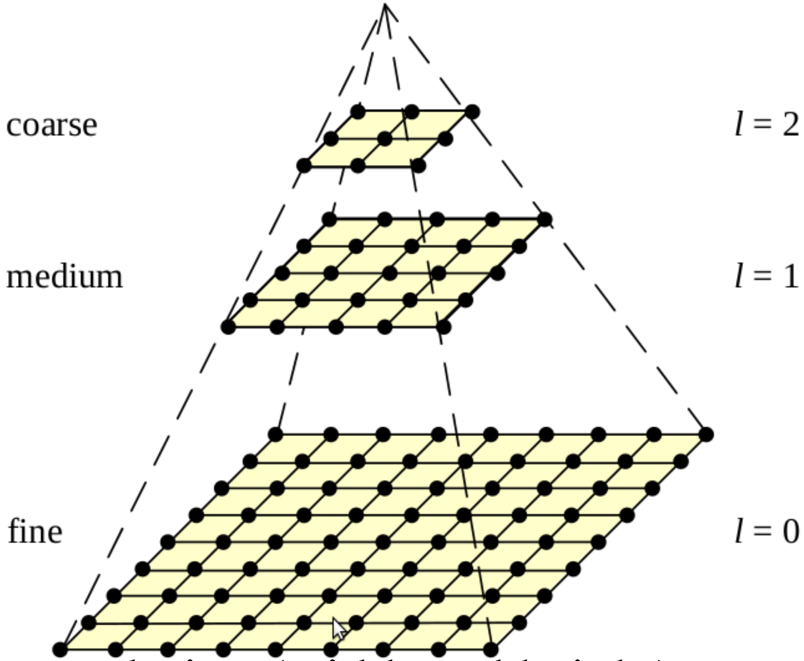


Decymacja sygnału: (a) oryginalne próbki są (b) splecione z filtrem dolnoprzepustowym przed próbkowaniem w dół.

Reprezentacje w wielu rozdzielczościach

- Piramidy mogą być używane do przyspieszania algorytmów wyszukiwania od zgrubnego do dokładnego, do wyszukiwania obiektów lub wzorców w różnych skalach oraz do wykonywania operacji mieszania w wielu rozdzielczościach.
- Ponieważ sąsiednie poziomy w piramidzie są powiązane częstotliwością próbkowania $r = 2$, ten rodzaj piramidy jest znany jako piramida oktawowa

Piramida obrazu

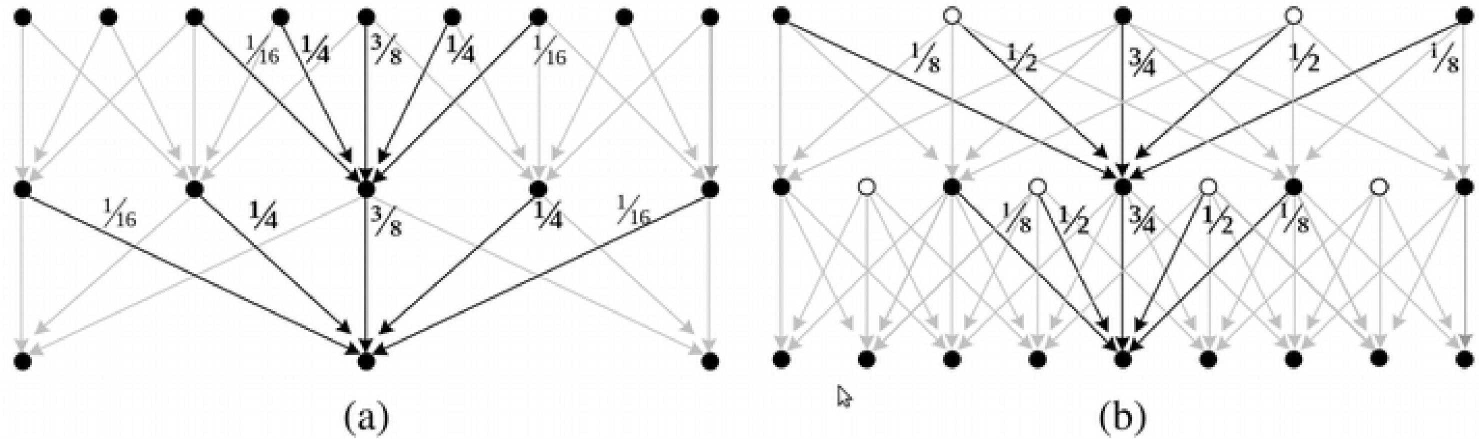


Each level has half the resolution (width and height)

- Kernel z pięcioma elementami postaci | c | b | a | b | c |, gdzie b = 1/4 ic = 1/4-a / 2, co daje w wyniku znane dwumianowe jądro:

$$\frac{1}{16}|1|4|6|4|1|$$

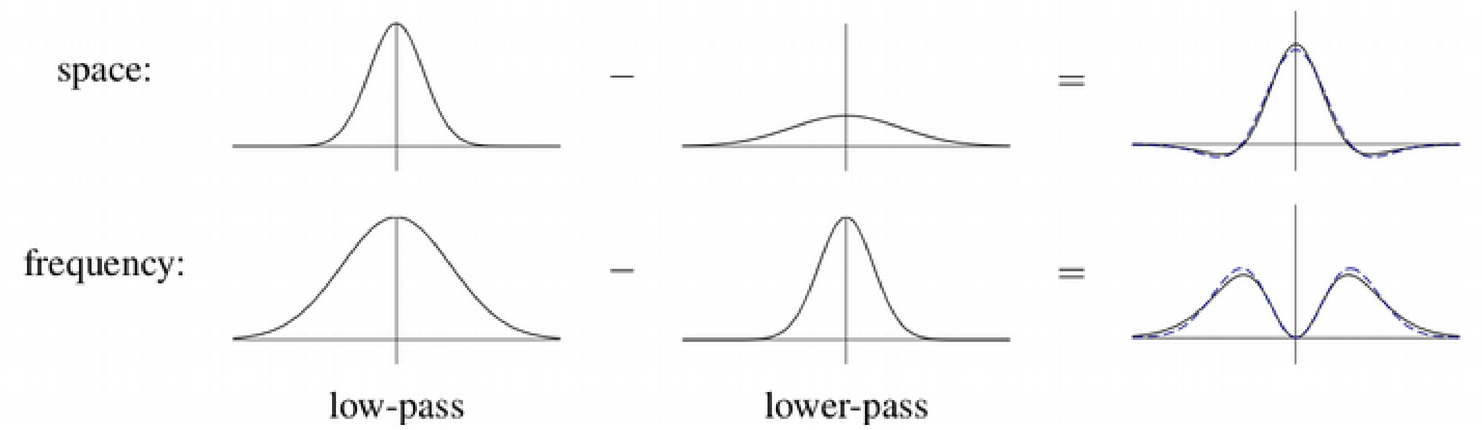
Piramida Gaussa



Piramida Gaussa jako schemat przetwarzania sygnału:

- (a) **Analiza**
- (b) **Etapy ponownej syntezy** są pokazane przy użyciu podobnych obliczeń. Białe kółka oznaczają wartości zerowe wstawione przez operację upsamplingu $\uparrow 2$.
- Współczynniki filtru rekonstrukcji są dwukrotnie większe od współczynników analizy. Obliczenie jest wyświetlane jako spływające w dół, niezależnie od tego, czy przechodzimy od zgrubnego do dokładnego, czy odwrotnie.

Laplasjan i filtr Gaussa



- Różnica dwóch filtrów dolnoprzepustowych skutkuje filtrem pasmowo-przepustowym. Przerywane niebieskie linie pokazują dopasowanie zbliżone do półoktawowego Laplacianu z Gaussa,
- Różnice w wyglądzie Gaussa i Laplasjanu z rozkładu Gaussa zarówno w przestrzeni, jak i częstotliwości,
- Termin Laplasjan jest trochę mylący, ponieważ ich obrazy pasmowo-przepustowe są w rzeczywistości różnicami (przybliżonych)

rozkładów Gaussa.

$$DoG\{I;\sigma_1,\sigma_2\} = G_{\sigma_1} \star I - G_{\sigma_2} \star I = (G_{\sigma_2} - G_{\sigma_2}) \star I.$$

Laplasjan w stosunku do rozkładu Gaussa jest właściwie jego drugą pochodną,

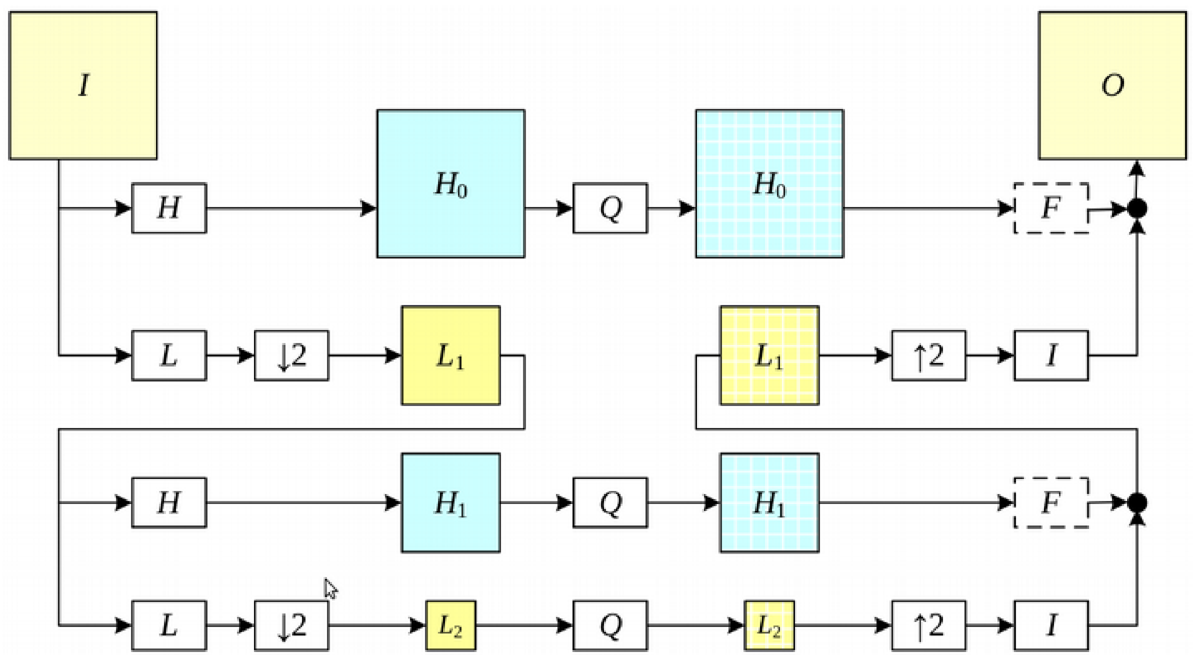
$$LoG\{I;\sigma\} = \nabla^2(G_\sigma \star I) = (\nabla^2 G_\sigma) \star I$$

gdzie

$$\nabla^2 = \frac{\partial}{\partial x^2} + \frac{\partial}{\partial y^2}$$

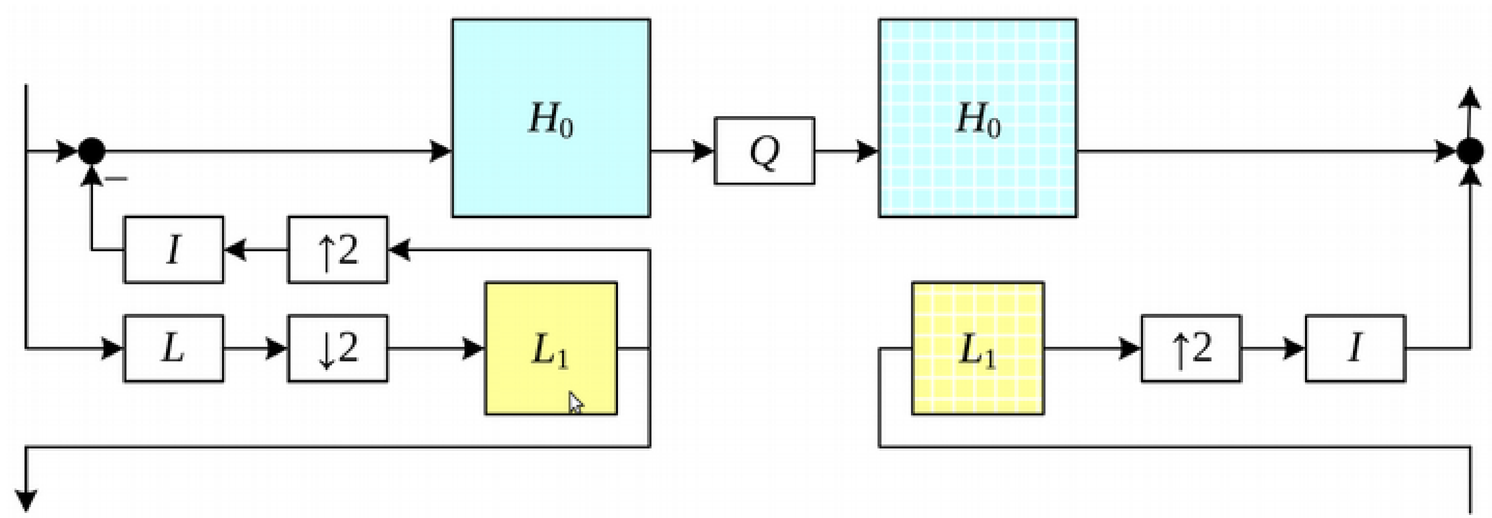
jest operatorem jest “Laplasowskim” (operatorem) funkcji.

Piramida Laplasjanów - konceptualny przepływ



- Koncepcyjny przepływ obrazów przez etapy przetwarzania: obrazy są filtrowane górnoprzepustowo i dolnoprzepustowo, a obrazy filtrowane dolnoprzepustowo są przetwarzane w kolejnym etapie piramidy.
- Podczas rekonstrukcji interpolowany obraz i (opcjonalnie filtrowany) obraz górnoprzepustowy są ponownie dodawane. Pole Q wskazuje kwantyzację.

Piramida Laplasjanów - konceptualny przepływ

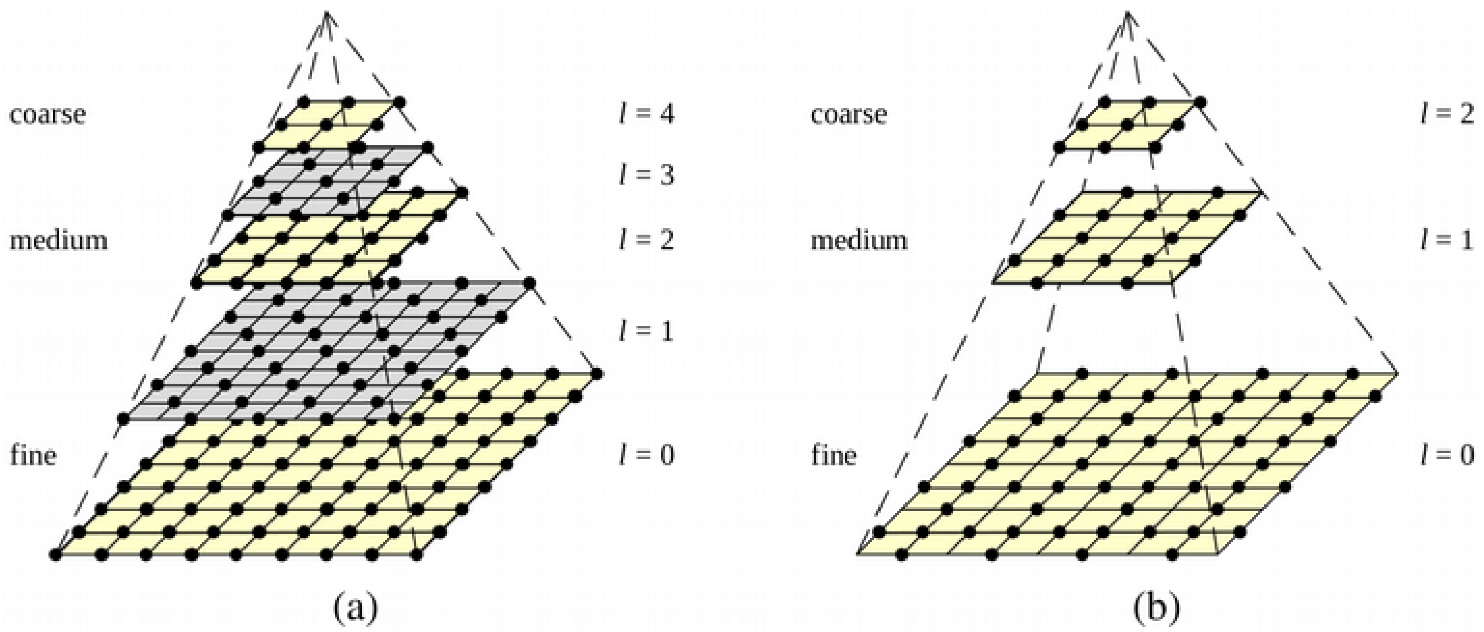


- Rzeczywiste obliczenia filtra górnoprzepustowego obejmują najpierw interpolację próbkowanego w dół obrazu dolnoprzepustowego, a następnie odjęcie go.
- Daje to doskonałą rekonstrukcję, gdy Q jest tożsamością.
- Obrazy górnoprzepustowe (lub pasmowo-przepustowe) są zwykle nazywane obrazami Laplace'a, podczas gdy obrazy dolnoprzepustowe nazywane są obrazami Gaussa.

Falki (Wavelety)

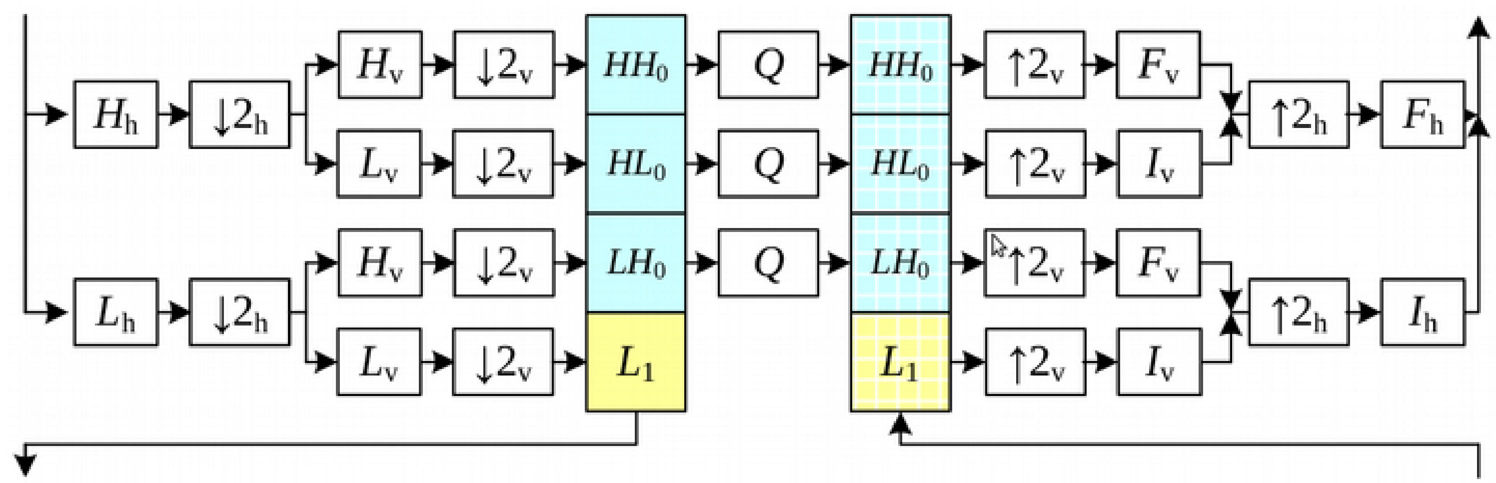
- Wavelety to filtry, które lokalizują sygnał zarówno w przestrzeni, jak i częstotliwości i są zdefiniowane w hierarchii skal,
- Wavelety zapewniają płynny sposób dekompozycji sygnału na składowe częstotliwościowe bez blokowania i są ściśle związane z piramidami,
- Wavelety zostały pierwotnie opracowane w społecznościach matematyki stosowanej i przetwarzania sygnałów i zostały wprowadzone do społeczności wizji komputerowej,
- Wavelety są szeroko stosowane w społeczności grafików komputerowych do wykonywania geometrycznego przetwarzania w wielu rozdzielczościach.

Piramidy w wielu skalach rozdzielczości



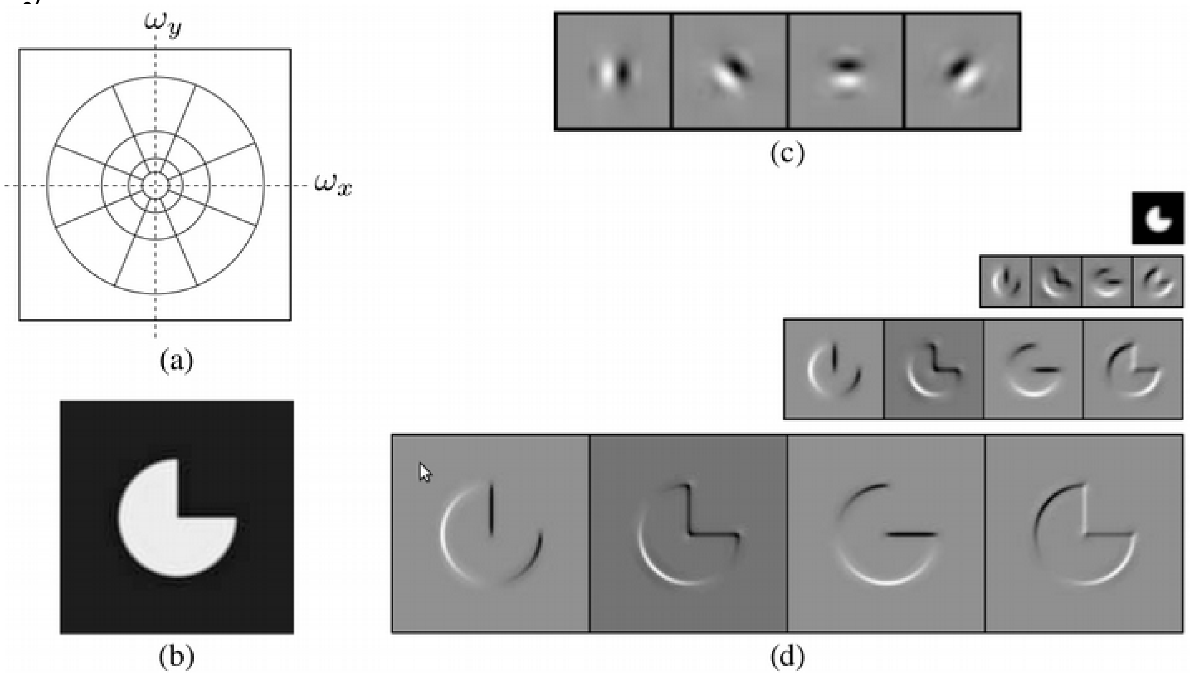
- (a) Piramida z próbkowaniem półoktawowym (kwinkunks) (nieparzyste poziomy są w kolorze szarym dla przejrzystości),
- (b) Piramida falkowa - każdy poziom falki przechowuje $\frac{3}{4}$ oryginalnych pikseli (zwykle gradienty poziome, pionowe i mieszane).

Dwuwymiarowy rozkład falkowy



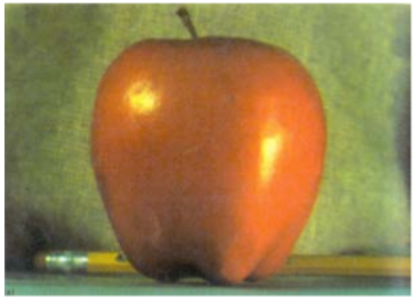
- Oddzielna implementacja, która polega najpierw na wykonaniu transformacji falkowej w poziomie, a następnie w pionie.
- Pola I i F to pola interpolacji i filtrowania wymagane do ponownej syntezy obrazu z jego składników falkowych.

Sterowalne, przesuwne transformacje wieloskalowe - przykład



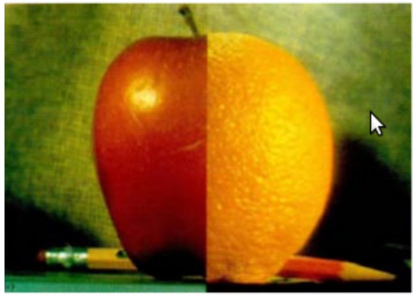
- (a) radialny wieloskalowy rozkład w dziedzinie częstotliwości,
- (b) oryginalny obraz,
- (c) zestaw czterech sterowalnych filtrów,
- (d) radialna wieloskalowa dekompozycja falkowa.

Zastosowanie: mieszanie obrazu

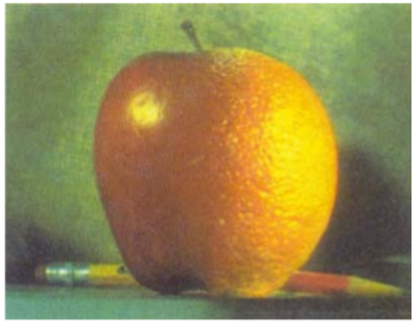


(a)

(b)



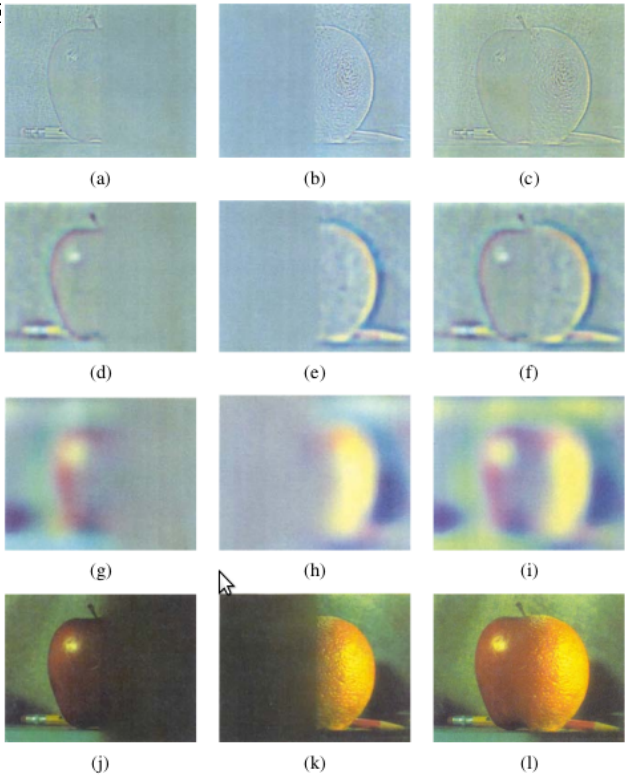
(c)



(d)

- (a) oryginalny obraz jabłka,
- (b) oryginalny obraz pomarańczy,
- (c) zwykłe połączenie,
- (d) mieszanka piramid

Piramida Laplasjanów łącząca szczegóły



Pierwsze trzy rzędy pokazują części piramidy Laplacian o wysokiej, średniej i niskiej częstotliwości (wzięte z poziomów 0, 2 i 4). Lewa i środkowa kolumna pokazują oryginalne obrazy jabłek i pomarańczy ważone przez funkcje płynnej interpolacji, podczas gdy prawa kolumna przedstawia uśrednione wkłady.

Kod biblioteki OpenCV - procedura

`https://docs.opencv.org/master/dc/dff/tutorial_py_pyramids.html`

1. Załaduj dwa obrazy jabłka i pomarańczy
2. Znajdź piramidy Gaussa dla jabłka i pomarańczy (w tym konkretnym przykładzie liczba poziomów to 6)
3. W piramidach Gaussa znajdź ich piramidy laplaciańskie
4. Teraz dołącz do lewej połowy jabłka i prawej połowy pomarańczy na każdym poziomie piramid Laplacian
5. Na koniec z tych połączonych piramid obrazów zrekonstruuj oryginalny obraz.

Zadania na laboratoria

1. Zastosuj transformatę Fouriera FFT (lub DFT) do przykładu. Wyświetla odebraną składową fazową i amplitudową,
2. Sprawdź empirycznie twierdzenie Pasevala dla obrazu i jego modułu,
3. Sprawdź działanie algorytmu piramidy dla przykładu w opencv,
4. Aby zobaczyć przykładowe zdjęcie morza i łodzi, spróbuj użyć algorytmu piramidy do umieszczenia łodzi na morzu (fotomontaż)