Uniwersytet Zielonogórski
Instytut Sterowania i Systemów Informatycznych

Rozpoznawanie Obrazów

Semestr VII, Informatyka

mgr inż. Marcin Skobel

2021

Laboratorium nr 7: Transformata Fouriera i falkowa.

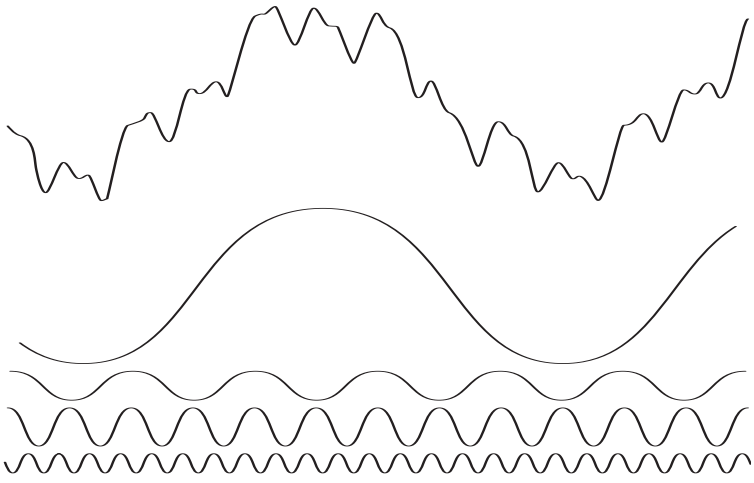
I. Zagadnienia teoretyczne

Wstęp

Transformata Fouriera i transformata falkowa są to funkcje, które pozwalają na wykonanie dekompozycji sygnału pierwotnego. Sygnał w rozumieniu przetwarzania obrazów może stanowić pojedynczy obraz. Wówczas transformacje Fouriera i falkowa mogą zostać użyte do przetworzenia obrazu. Transformacje te mogą być wykorzystywane do detekcji krawędzi, rozmywania obrazów czy też kompresji.

Transformata Fouriera

Transformata Fouriera, w skrócie służy do rozbicia sygnału na grupę sinusoid. Innymi słowy zakłada, że dowolny sygnał można rozbić na prostsze fragmenty. Francuski matematyk Jean



Rysunek 1: Na górze znajduje się funkcja pierwotna a poniżej funkcje na które można ją rozbić

Joseph Fourier opracował transformatę Fouriera, próbując rozwiązać równanie ciepła. W trakcie

tego procesu zauważył, że funkcja okresowa może być wyrażona jako nieskończone sumy sinusów i cosinusów o różnych częstotliwościach, obecnie znanych jako szereg Fouriera. Transformata Fouriera jest rozszerzeniem szeregu Fouriera na funkcje nieokresowe. Przekształcenie Fouriera jest reprezentacją, w której dowolną funkcję można wyrazić jako całkę z sinusów i cosinusów pomnożoną przez funkcję ważoną. Ponadto każda funkcja reprezentowana przez szereg Fouriera lub transformację może być całkowicie zrekonstruowana w procesie odwrotnym. Jest to znane jako odwrotna transformata Fouriera. Transformata Fouriera funkcji ciągłej w jednej zmiennej $f(x)$ wynika z następującego równania:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx \tag{1}$$

gdzie $i = \sqrt{-1}$. Funkcję $f(x)$ można uzyskać, znajdując rozszerzenie odwrotnej transformaty Fouriera funkcji $F(u)$, która jest określona następującym równaniem:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{i2\pi ux} du \tag{2}$$

Transformata Fouriera funkcji dyskretnej jednej zmiennej, $f(x) dla x = 0, 1, ...L-1$ jest określone następującym równaniem:

$$F(u) = \frac{1}{L} \sum_{x=0}^{L-1} f(x) e^{\frac{-i2\pi ux}{L}} \tag{3}$$

dla $u = 0, 1, ...L-1$. Powyższe równanie jest znane jako dyskretna Transformata Fouriera, DFT. Analogicznie, odwrotna dyskretna transformacja Fouriera forma, (IDFT) jest określona następującym równaniem:

$$f(x) = \sum_{x=0}^{L-1} F(u) e^{\frac{i2\pi ux}{L}} \tag{4}$$

dla $x = 0, 1, ...L-1$. Korzystając ze wzoru Eulera $e^{i\theta} = cos\theta + isin\theta$, powyższe równanie upraszcza się do:

$$F(u) = \frac{1}{L} \sum_{x=0}^{L-1} f(x) \left[cos\left(\frac{-2ux\pi}{L}\right) - i\ sin\left(\frac{-i2ux\pi}{L}\right) \right] \tag{5}$$

Teraz, korzystając z faktu, że *cos* jest funkcją parzystą, tj. $cos(-\pi) = cos(\pi)$ i że *sin* jest funkcją nieparzystą, tj. $sin(-\pi) = -sin(\pi)$, równanie powyższe może być uproszczone do:

$$F(u) = \frac{1}{L} \sum_{x=0}^{L-1} f(x) \left[cos\left(\frac{2ux\pi}{L}\right) + i\ sin\left(\frac{2ux\pi}{L}\right) \right] \tag{6}$$

$F(u)$ składa się z dwóch części: część rzeczywista tworząca *cos* jest przedstawiana jako $R(u)$, a część urojona tworząca *sin* jest przedstawiana jako $I(u)$. Każdy człon F jest znany jako współczynnik transformaty Fouriera. Ponieważ u odgrywa kluczową rolę w określaniu częstotliwości współczynników transformaty Fouriera, u jest znane jako zmienna częstotliwości, podczas gdy x jest znana jako zmienna przestrzenna.

Tradycyjnie wielu ekspertów porównywało transformatę Fouriera do szklanego pryzmatu. Gdy szklany pryzmat rozdziela lub rozdziela światło na różne długości fal lub częstotliwości, które tworzą widmo, transformacja Fouriera dzieli lub rozdziela funkcję na jej współczynniki, które zależą od częstotliwości. Te współczynniki Fouriera tworzą widmo Fouriera w dziedzinie częstotliwości.

Z równania 6 wiemy, że transformata Fouriera składa się z liczb zespolonych. Dla celów obliczeniowych wygodnie jest przedstawić transformację Fouriera w postaci biegunowej jako:

$$F(u) = |F(u)|e^{-i\theta(u)}, \tag{7}$$

gdzie $|F(u)| = \sqrt{R^2(u) + I^2(u)}$ nazywana jest wielkością transformaty Fouriera a $\theta(u) = tan^{-1}\left[\frac{I(u)}{R(u)}\right]$ nazywany jest kątem fazowym transformacji. Moc, $P(u)$, definiuje się następująco:

$$P(u) = R^2(u) + I^2(u) = |F(u)|^2 \tag{8}$$

Pierwszą wartość dyskretnej transformaty Fouriera uzyskuje się przez ustawienie $u = 0$ w równaniu 3, a następnie zsumowanie iloczynu po wszystkich x. Zatem $F(0)$ jest niczym innym jak średnią $f(x)$, ponieważ $e^0 = 1$. $F(0)$ ma część rzeczywistą, podczas gdy część urojona wynosi zero. Inne wartości F można obliczyć w podobny sposób.

Transformacja Fouriera dla dwóch zmiennych jest określona następującym równaniem:

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)e^{-i2\pi(ux+vy)}dx\ dy \tag{9}$$

a odwrotna transformata Fouriera to:

$$f(x,y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} F(u,v)e^{i2\pi(ux+vy)}du\ dv \tag{10}$$

Dyskretna transformata Fouriera funkcji 2D, $f(x,y)$ o rozmiarze L i K jest dana następującym równaniem:

$$F(u,v) = \frac{1}{LK}\sum_{x=0}^{L-1}\sum_{x=0}^{K-1} f(x,y)e^{-i2\pi\left(\frac{ux}{L}\frac{vy}{K}\right)} \tag{11}$$

dla $u = 1, 2, ..., L-1$ i $v = 1, 2, ..., K-1$. Podobnie do transformaty Fouriera 1D, $f(x,y)$ można obliczyć z $F(u,v)$, wykonując odwrotną transformatę Fouriera, daną następującym równaniem:

$$F(u,v) = \sum_{x=0}^{L-1}\sum_{x=0}^{K-1} F(u,v)e^{i2\pi\left(\frac{ux}{L}\frac{vy}{K}\right)} \tag{12}$$

dla $x = 1, 2, ..., L-1$ i $y = 1, 2, ..., K-1$. Podobnie jak w przypadku 1D DFT, u i v są zmiennymi częstotliwości, a x i y są zmiennymi przestrzennymi. Wielkość transformaty Fouriera w 2D jest określona następującym równaniem:

$$|F(u,v)| = \sqrt{R^2(u,v) + I^2(u,v)} \tag{13}$$

a kąt fazowy jest określony przez:

$$\theta(u,v) = tan^-1\left[\frac{I(u,v)}{R(u,v)}\right] \tag{14}$$

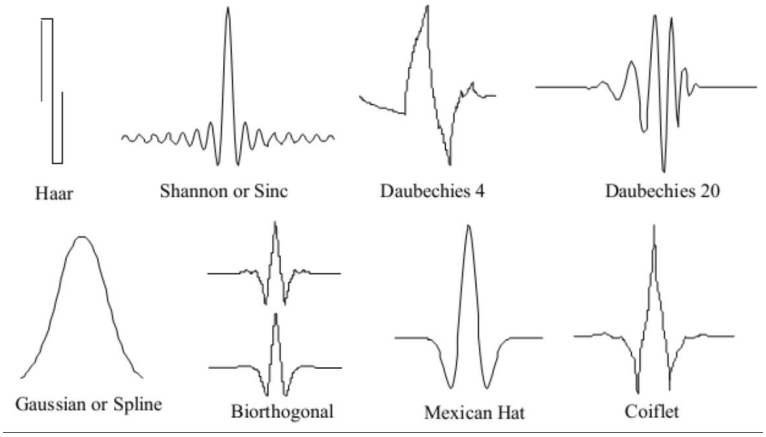
natomiast moc określamy przez:

$$P(u,v) = R^2(u,v) + I^2(u,v) = |F(u,v)|^2 \tag{15}$$

gdzie $R(u,v)$ i $I(u,v)$ są rzeczywistymi i urojonymi częściami 2D DFT. Złożoność DFT wynosi N^2. Dlatego do obliczenia transformaty Fouriera stosuje się zmodyfikowaną metodę zwaną szybką transformacją Fouriera (FFT).

Transformata falkowa

Transformacja falkowa działa na podobnej zasadzie jak transformacja Fouriera z tą różnicą, że w zamiast dekompozycji sygnału zamiast funkcji sinusoidalnych używa się falek. Falka to rodzaj niewielkiej fali skoncentrowanej w stosunkowo krótkim przedziale czasowym. Falka podstawowa stanowi bazę do dekompozycji badanego sygnału. Dekompozycja odbywa się natomiast przy użyciu falek otrzymanych poprzez przekształcenia falki podstawowej. Falka musi spełniać pewne kryteria związane dostosowaniem do analizy wielorozdzielczej i po spełnieniu tych kryteriów może stać się falką. Z tej przyczyny istnieje nieskończenie wiele falek których przykłady można odnaleźć na poniższym rysunku:



Rysunek 2: Różne przykłądy falek

Równanie falkowe aproksymujące funkcję f(x) przyjmuje postać:

$$f(x) = \sum_k \sum_l c_{k,l} \psi_{k,l}(x) \tag{16}$$

gdzie $c_{k,l}$ - to współczynniki falkowe, natomiast $\psi_{k,l}(x)$ to falka w skali l po przesunięciu o k względem falki podstawowej. Ciągłą transformacja falkowa przyjmie formę całki:

$$\tilde{s}_\psi(k,l) = \frac{1}{\sqrt{a}} \int_{-\infty}^{\infty} s(t) \psi \left(\frac{t-l}{k} \right) \, dt \tag{17}$$

gdzie k - parametr skali, l - parametr przesunięcia w dziedzinie czasu t, $s(t)$ - sygnał poddawany analizie, $\psi\left(\frac{t-l}{k}\right)$ - jądro transformacji, $\tilde{s}_\psi(k,l)$ - transformata falkowa. Parametr skali w zależności od wielkości będzie decydował o pseudoczęstotliwości falki czyli obrazowo mówiąc o jej rozciągnięciu. Wraz ze wzrostem wartości parametru skali maleje pseudoczęstotliwość falki. Współczynnik $\frac{1}{\sqrt{a}}$ ma na celu normalizację falki. Proces normalizacji ma na celu utrzymanie stałej energii funkcji falkowej bez względu na skalę. Zastosowania transformacji falkowej to: Lokalizacja anomalii, redukcja szumu, kompresja obrazów, wykrywanie obiektów.

II. Przykład praktyczny

Wstępna konfiguracja

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
from google.colab import drive
drive.mount('/content/drive')
```

Wczytanie pliku

```
path = testImagePath = 'drive/My Drive/Colab Notebooks/Images/Test.tif'
#flaga 0 oznacza, ze obrazek na etapie wczytywania jest konwertowany do skali szarosci
img = cv2.imread(path,0)
```

Transformacja Fouriera - odwrócenie transformacji

```
#deklaracja wielkosci obrazka
plt.figure(figsize=(30, 25))
#wyznaczenie spektrum (transformacja Fouriera)
img2 = np.fft.fft2(img)
# centralizacja spektrum
img3 = np.fft.fftshift(img2)
# decentralizacja spektrum
img4 = np.fft.ifftshift(img3)
# odwrocenie transformaty Fouriera ze zdecentralizowanego obrazu
img5 = np.fft.ifft2(img4)
# wyswietlenie wynikow
plt.subplot(511), plt.imshow(img, "gray"), plt.title("Original Image")
plt.subplot(512), plt.imshow(np.log(1+np.abs(img2)), "gray"), plt.title("Spectrum")
plt.subplot(513), plt.imshow(np.log(1+np.abs(img3)), "gray"), plt.title("Centered Spectrum
plt.subplot(514), plt.imshow(np.log(1+np.abs(img4)), "gray"), plt.title("Decentralized")
plt.subplot(515), plt.imshow(np.abs(img5), "gray"), plt.title("Processed Image")
plt.show()
```

Transformacja Fouriera - filtracja górnoprzepustowa

```
#wyznaczenie spektrum (transformacja Fouriera)
f = np.fft.fft2(img)
# centralizacja spektrum
fshift = np.fft.fftshift(f)
magnitude_spectrum = 20*np.log(np.abs(fshift))

#PROCES FILTRACJI GORNOPRZEPUSTOWEJ (Usuwanie niskich czestotliwosci)
#zmienne rozmiaru obrazu
rows, cols = img.shape
crow,ccol = int(rows/2) , int(cols/2)
# Maskowanie niskich czestotliwosci okienkiem o wymiarach 60x60
fshift[crow-30:crow+30, ccol-30:ccol+30] = 0
#przesuniecie odwrotne
f_ishift = np.fft.ifftshift(fshift)
```

```
#wyznaczenie odwrotnej FFT
img_back = np.fft.ifft2(f_ishift)
# pobranie wartosci bezwzglednych
img_back = np.abs(img_back)
# wyswietlenie wynikow
plt.figure(figsize=(20, 15))
plt.subplot(311),plt.imshow(img, cmap = 'gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(312),plt.imshow(img_back, cmap = 'gray')
plt.title('Image after HPF'), plt.xticks([]), plt.yticks([])
plt.subplot(313),plt.imshow(img_back, cmap = plt.cm.get_cmap("jet"))
plt.title('Result in JET'), plt.xticks([]), plt.yticks([])
plt.show()
```

Transformacja Fouriera - rozmycie obrazu

```
dft = cv2.dft(np.float32(img),flags = cv2.DFT_COMPLEX_OUTPUT)
# centralizacja spektrum
dft_shift = np.fft.fftshift(dft)
magnitude_spectrum = 20*np.log(cv2.magnitude(dft_shift[:,:,0],dft_shift[:,:,1]))
# wyswietlenie wynikow
plt.subplot(121),plt.imshow(img, cmap = 'gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(magnitude_spectrum, cmap = 'gray')
plt.title('Magnitude Spectrum'), plt.xticks([]), plt.yticks([])
plt.show()
rows, cols = img.shape
crow,ccol = rows/2 , cols/2
# najpierw tworzy sie maske, srodkowy kwadrat = 1, pozostale wartosci = 0
mask = np.zeros((rows,cols,2),np.uint8)
mask[int(crow-30):int(crow+30), int(ccol-30):int(ccol+30)] = 1
# zastosowanie maski i odwrocenie DFT
fshift = dft_shift*mask
# przesuniecie odwrotne
f_ishift = np.fft.ifftshift(fshift)
#wyznaczenie odwrotnej FFT
img_back = cv2.idft(f_ishift)
img_back = cv2.magnitude(img_back[:,:,0],img_back[:,:,1])
# wyswietlenie wynikow
plt.subplot(121),plt.imshow(img, cmap = 'gray')
plt.title('Input Image'), plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(img_back, cmap = 'gray')
plt.title('Magnitude Spectrum'), plt.xticks([]), plt.yticks([])
plt.show()
```

Transformacja falkowa

```
#wyswietlenie listy dostepnych falek
for family in pywt.families():
  print("%s family: " % family + ', '.join(pywt.wavelist(family)))
#w tym dyskretne
print(pywt.wavelist(kind='discrete'))
#wybor falki
```

```
waveletname = 'haar'
#konwersja do float
imArray =  np.float32(img)
imArray /= 255;
#obliczenie wspolczynnikow transformacji
coeffs=pywt.wavedec2(imArray,wavelet=waveletname)
#wykonanie wspolczynnikow
coeffs_H=list(coeffs)
coeffs_H[0] *= 0;
#rekonstrukcja
imArray_H=pywt.waverec2(coeffs_H, waveletname);
imArray_H *= 255;
imArray_H =  np.uint8(imArray_H)
#wyniki
plt.imshow(imArray_H)
```

III. Uwagi

Lista dostępnych rodzin falek w bibliotece pywt:
Haar (haar)
Daubechies (db)
Symlets (sym)
Coiflets (coif)
Biorthogonal (bior)
Reverse biorthogonal (rbio)
"Discrete" FIR approximation of Meyer wavelet (dmey)
Gaussian wavelets (gaus)
Mexican hat wavelet (mexh)
Morlet wavelet (morl)
Complex Gaussian wavelets (cgau)
Shannon wavelets (shan)
Frequency B-Spline wavelets (fbsp)
Complex Morlet wavelets (cmor).
Plik z przykładami można pobrać ze strony: `http://staff.uz.zgora.pl/mskobel/lab7.py`

IV. Lista zadań

1. Znajdź w sieci i pobierz obraz z wyraźnymi krawędziami i wykonaj filtrację górnoprzepustową.
2. Dokonaj rozmazania obrazka tak aby rozmycie obrazu było wykonywane przy użyciu okręgu zamiast kwadratu. (Uwaga: chodzi o modyfikację linijki: *mask[int(crow-30):int(crow+30), int(ccol-30):int(ccol+30)] = 1*
3. Utwórz funkcję, która będzie sprawdzać z listy *pywt.wavelist(kind='discrete')* czy nazwa podanego filtra do niej należy. Jeśli nie to program ma wylosować nazwę z listy i poinformować o tym użytkownika.