Projekt II

Procesy Losowe - Teoria dla Praktyka

Klaudia Klębowska 160820
Bartosz Filipów 160488

09.05.2019

1 Power spectral subtraction

Metoda odejmowania widmowego (ang. Spectral Subtraction) służy do oczyszczania zakłóceń oraz rekonstrukcji sygnałów mowy. Każdy nagrywany sygnał, oprócz użytecznej części posiada także pewien poziom szumu. Na wejściu programu mamy zaszumiony dyskretny sygnał $x[t]$.

1.1 DFT i IDFT

Na początku zdefiniujmy dyskretną transformatę Fouriera (DFT) oraz odwrotną dyskretną transformatę Fouriera (IDFT), które będą potrzebne w dalszych krokach algorytmu.

$$\mathrm{dft}(x[t]) = X[\omega_i] = \sum_{t=1}^{N-1} x[t] e^{-j\omega_i t}, \tag{1}$$

gdzie:
$\omega_i = \frac{2\pi i}{M}$,
$i = 0, 1, ..., M-1$
M - długość sygnału.

$$\mathrm{idft}(X[\omega]) = x[t] = \frac{1}{M} \sum_{i=1}^{M-1} X[\omega_i] e^{j\omega_i t}, \tag{2}$$

gdzie:
$t = 0, 1, ..., M-1$,
M - długość sygnału.

1.2 Estymacja mocy widmowej szumu.

Nasz sygnał należy podzielić na ramki o szerokości N. Powinna być ona tak dobrana, aby nie była dłuższa, niż 10-20 ms w przypadku sygnału mowy. W pierwszym kroku należy wykorzystać ramkę, która zawiera sam szum. Oznaczmy ją jako $z[t]$ i estymujmy jej widmową gęstość mocy:

$$\widehat{S_Z[\omega_i]} = \frac{1}{N} |Z[\omega_i]|^2, \tag{3}$$

gdzie:
$i = 0, 1, ..., \frac{N}{2}$,
$Z[\omega_i] = \mathrm{dft}(z[t])$.

Jeżeli dostępne jest wiele ramek zawierających tylko szum, to można zastosować uśrednianie:

$$\widehat{S_Z[\omega_i]} = \frac{1}{U}\sum_{i=1}^{N}\sum_{j=1}^{U}\widehat{S_{Z_j}[\omega_i]}, \tag{4}$$

gdzie:
U - ilość ramek zawierających tylko szum.

1.3 Estymacja mocy widmowej sygnału.

Drugim krokiem jest estymacja mocy widmowej sygnału. W tym celu bierzemy pierwszą ramkę sygnału wejściowego. Oznaczmy ją jako $y[t]$.

$$\widehat{S_Y[\omega_i]} = \frac{1}{N}|Y[\omega_i]|^2, \tag{5}$$

gdzie:
$i = 0, 1, ..., \frac{N}{2}$,
$Y[\omega_i] = \mathrm{dft}(y[t])$.

1.4 Estymacja funkcji gęstości widmowej mocy sygnału bezszumowego.

Trzecim krokiem algorytmu jest estymacja funkcji gęstości widmowej mocy sygnału bezszumowego. W tym celu wykorzystujemy wyniki z równań (4) i (5):

$$\widehat{S_X[\omega_i]} = \begin{cases} \widehat{S_Y[\omega_i]} - \widehat{S_Z[\omega_i]} & \text{gdy } \widehat{S_Y[\omega_i]} - \widehat{S_Z[\omega_i]} \geqslant 0 \\ 0 & \text{w pozostałych przypadkach} \end{cases}, \tag{6}$$

gdzie:
$i = 0, 1, ... \frac{N}{2}$.

1.5 Filtr odszumiający

Czwartym krokiem jest zaprojektowanie filtra, który odszumi nasz sygnał wejściowy:

$$\widehat{A}[\omega_i] = \sqrt{\frac{\widehat{S_X[\omega_i]}}{\widehat{S_Y[\omega_i]}}}, \tag{7}$$

gdzie:
$i = 0, 1, ..., \frac{N}{2}$. Filtr ten jest symetryczny, więc drugą część próbek wystarczy odbić względem próbki $\frac{N}{2}$: Czwartym krokiem jest zaprojektowanie filtra, który odszumi nasz sygnał wejściowy:

$$\widehat{A}[\omega_i] = \widehat{A}[\omega_{N-1-i}], \tag{8}$$

gdzie:
$i = \frac{N}{2} + 1, \frac{N}{2} + 2, ..., N - 1$.

1.6 Odszumianie

Piątym krokiem jest wykorzystanie filtra zaprojektowanego w poprzednim etapie:

$$\widehat{X_{czysty}}[\omega_i] = \widehat{A}[\omega_i]\widehat{Y}[\omega_i] \tag{9}$$

Na koniec wystarczy przeprowadzić odwrotną dyskretną transformatę Fouriera:

$$\widehat{x_{czysty}}[t] = \mathrm{idft}(\widehat{X_{czysty}}[\omega_i]) \tag{10}$$

W efekcie otrzymujemy pierwszą ramkę pozbawioną szumu. Teraz należy wczytać kolejną ramkę i wrócić do drugiego kroku. Czynności te powtarzamy do ostatniej ramki sygnału wejściowego.

1.7 Rozszerzenie metody o technikę overlap-add

W przypadku, gdy na końcach lub początkach każdej z ramki słychać nieciągłość sygnału można zastosować technikę overlap-add. W tym celu należy pobierać kolejne ramki o długości N z przesunięciem $\frac{N}{2}$ oraz wykorzystać współczynnik $w[t]$:

$$w[t] = 1 - \frac{|t - N|}{N} \tag{11}$$

$$\widehat{x_{overlap_i}}[t] = \widehat{x_{czysty_i}}[t + \frac{N}{2}]w[t] + \widehat{x_{czysty_{i+1}}}[t]w[t], \tag{12}$$

gdzie:
N - szerokość ramki,
$t = 1, 2, ..., \frac{N}{2}$,
$i = 1, 2, ..., R$,
R - liczba ramek na które został podzielony sygnał wejściowy.

Dzięki temu otrzymujemy kolejne ramki pozbawione szumu oraz nieciągłości sygnału na początku lub końcu ramki.

2 Listing programu

Program został napisany w środowisku Matlab.

```
clear; clc;
format compact
format long

% Parametry
koniec       = 2^18;
dl_ramka     = koniec/256;
skal         = 10;
ram_szum     = 30;

% Czytanie z pliku audio
[y,Fs]   = audioread('Randka_w_ciemno.wav');
info     = audioinfo('Randka_w_ciemno.wav');
y1       = y(1:1:koniec);

% Generowanie bialego szumu
szum = wgn(koniec,1,-40);

y2 = y1+szum;
y3 = zeros(length(y2)+dl_ramka*ram_szum,1);

for i=1:1:(koniec+dl_ramka*ram_szum)
  if i <= dl_ramka*ram_szum
    y3(i) = szum(i);
  else
    y3(i) = y2(i-dl_ramka*ram_szum);
  end
end

```

```
% Krok 1
ramki    = ram_szum;
S_z      = zeros(dl_ramka/2,1);
N        = dl_ramka;
szum_usr= zeros(dl_ramka,1);
szum2    = zeros(N,1);

for i=1:1:ramki
  for j=1:1:N
    szum2(j) = szum(j+(N*(i-1)));
  end
  Z = fft(szum2);
  for j=1:1:(N/2)
    S_z(j) = S_z(j) + (1/N)*(abs(Z(j))^2);
  end
end

S_z = S_z./ram_szum;

% Krok 2
ramki    = (length(y3)/dl_ramka);
y4       = y3(1:1:length(y3));
S_y      = zeros(dl_ramka/2,1);
N        = dl_ramka;
S_x      = zeros(dl_ramka/2,1);
A        = zeros(N,1);
X        = zeros(N,1);
sygnal   = zeros(N,1);
wyjsciowy = zeros(length(y4),1);


for i=1:0.5:ramki
  for j=1:1:N
    sygnal(j) = y4(j+dl_ramka*(i-1));
  end
  Y = fft(sygnal);

  for j=1:1:(N/2)
    S_y(j) = (1/N)*(abs(Y(j))^2);
  end

  % Krok 3
  for j=1:1:N/2
    if (S_y(j) - skal*S_z(j) >= 0)
      S_x(j) = S_y(j) - skal*S_z(j);
    else
      S_x(j) = 0;
    end
  end

  % Krok 4
```

```
81    for j=1:1:N
82      if j <= N/2
83        A(j) = sqrt(S_x(j)/S_y(j));
84      else
85        A(j) = A(N-j+1);
86      end
87    end
88
89    for j=1:1:length(Y)
90      X(j) = (A(j))*(Y(j));
91    end
92
93    Xodw = real(ifft(X));
94
95    for j=1:1:dl_ramka
96      k = j+(N*(i-1));
97      wp = 1-(abs(j-N*0.5)/(N*0.5));
98      wyjsciowy(k) = wyjsciowy(k)+wp*Xodw(j);
99    end
100 end
101 sound(wyjsciowy,Fs)
```