Jacek Misiurewicz
Krzysztof Kulpa
Piotr Samczyński
Mateusz Malanowski
Piotr Krysik
Łukasz Maślikowski
Damian Gromek
Artur Gromek
Marcin K. Bączyk

Zakład Teorii Obwodów i Sygnałów
Instytut Systemów Elektronicznych
Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Laboratorium Cyfrowego Przetwarzania Sygnałów

Wersja do wydruku - bez części teoretycznej

Warszawa 2019

Ćwiczenie 9

Przetwarzanie obrazów

Część teoretyczną w tej wersji opuszczono.

9.2. Zadania do pracy własnej studenta

Podobne zadania mogą znaleźć się na wejściówce. Nie dotyczy to zadań oznaczonych tu jako „trudne".

1) Przetwarzamy na bieżąco (w czasie rzeczywistym) sygnał dwuwymiarowy z anteny wieloelementowej $x(m, n)$, gdzie indeks m odpowiada numerowi anteny, a indeks n – numerowi próbki czasowej. Czy możemy zastosować filtr nieprzyczynowy? (Uwaga, pytanie podchwytliwe!)[1].

2) Oblicz dwuwymiarową DTF obrazu o rozmiarze 16×16 pikseli:

a) całkowicie czarnego (wypełnionego zerami),
b) całkowicie białego (wypełnionego jedynkami),
c) zawierającego tylko jeden biały piksel w pozycji $(0, 0)$,
d) zawierającego osiem pionowych białych pasów i osiem pionowych czarnych pasów,
e) to samo, tylko pasy poziome,

(zakładamy, że czarny oznaczamy zerem, biały – jedynką). **Wskazówka:** Zauważ, że 2D-DTF można obliczać kolejno – najpierw poziomo, potem pionowo (lub odwrotnie).

3) Jakiemu obrazowi będzie odpowiadać widmo o jednej niezerowej próbce w punkcie $(0, 0)$? Przyjmij wartość próbki taką, aby uzyskać w obrazie tylko czerń lub biel (uwaga na współczynnik skalujący w ODTF).

4) Obraz (sygnał 2D) o rozmiarze 16×16 pikseli zawierający tylko jeden biały piksel w pozycji $(8, 8)$ przefiltrowano filtrem liniowym o odpowiedzi impulsowej

```
lp2=[ 1 1 1;...
      1 2 1;...
      1 1 1];
```

[1] Przyczynowość można badać oddzielnie dla każdego wymiaru!

Zakładając, że jest to filtr nieprzyczynowy (o zerowej fazie):
a) zastanów się, która próbka odpowiedzi impulsowej filtru jest próbką $h(0,0)$,
b) oblicz obraz po filtracji (czyli sygnał wyjściowy),
c) uzasadnij, że rzeczywiście charakterystyka fazowa jest zerowa.

5) Oblicz ręcznie wynik filtracji obrazu o rozmiarze 16×16 pikseli filtrem wykrywającym krawędzie:
a) obraz czarny z jednym białym pikselem, filtr (nieprzyczynowy),
```
edg1=[0  0  0;...
      -1  1  0;...
       0  0  0];
```
b) obraz z jedną pionową białą linią, filtr jak wyżej,
c) ... (wymyśl sobie inne kombinacje).

6) Narysuj linię prostą w obrazie, której odpowiada podany punkt w przeciwdziedzinie transformacji Hougha:
a) $(0,0)$,
b) $(0,\pi/2)$,
c) $(0,-\pi/4)$,
d) $(10,0)$,
e) ... (wymyśl jeszcze kilka).

7) Przefiltruj podany sygnał filtrem medianowym rzędu 5 (dla uproszczenia rozpatrzymy sygnały jednowymiarowe):
a) impuls jednostkowy $\delta(n)$,
b) skok jednostkowy $u(n)$,
c) skok jednostkowy z dodanym impulsem: $u(n) + \delta(n-2)$,
d) skok jednostkowy z dziurką: $u(n) - \delta(n-3)$.

9.3. Dostępny sprzęt i oprogramowanie

9.3.1. Wczytywanie i przygotowanie obrazów

W wielu zadaniach trzeba będzie skądś pozyskać obraz, na którym będziesz ćwiczyć różne algorytmy. Zalecamy tu używać obrazów naturalnych („fotograficznych"), a nie grafiki przygotowanej przez człowieka – lepiej będzie widać większość efektów.

Jeśli używasz obrazu zapisanego na dysku (np. z jakiegoś uniwersalnego narzędzia do obsługi kamery), obraz trzeba wczytać. Obrazy wczytujemy poleceniem `imread`.
```
myimageRGB=imread('SciezkaDoPliku.png');
```
Można też przeciągnąć plik myszką na okno „workspace" Matlaba (nazwa zmiennej będzie identyczna z nazwą pliku, bez rozszerzenia).

Kamerę podłączoną do komputera można obsłużyć za pomocą drivera vcapg[2]. – po podłączeniu kamery wystarczy wywołać z poziomu Matlaba poniższy kod.

[2] VCAPG nie jest standardowym składnikiem Matlaba, został opracowany w Japonii: `http://www.ikko.k.hosei.ac.jp/~matlab/matkatuyo/`

```
myimageRGB=LCPS_getsnapshot();% calls VCAPG.MEXW64 loadable function
imshow(myimageRGB);
```

Może zdarzyć się, że driver się „zawiesi” – wtedy można go zresetować poprzez wywołanie `LCPS_getsnapshot(1)`.

Jeśli obraz jest barwny, otrzymasz trójwymiarową tablicę $M \times N \times 3$ z trzema kanałami barw R, G, B; wydobądź jeden z kanałów wczytanego obrazu lub transformuj go do odcieni szarości procedurą `rgb2gray`, aby uzyskać tablicę $M \times N$.

```
myimageGray=rgb2gray(myimageRGB);
```

Obrazy typu JPG i PNG wczytywane są w postaci z całkowitoliczbową skalą barwy lub jasności (`uint8`). Aby uniknąć ograniczeń przy obliczeniach, dokonaj konwersji (rzutowania) na typ liczb zmiennoprzecinkowych.

```
myimage=double(myimageGray);
```

Wskazówka: W ćwiczeniu przy każdym wczytaniu nowego obrazu trzeba będzie powtórzyć powyższe operacje.

9.4. Eksperymenty do wykonania w laboratorium

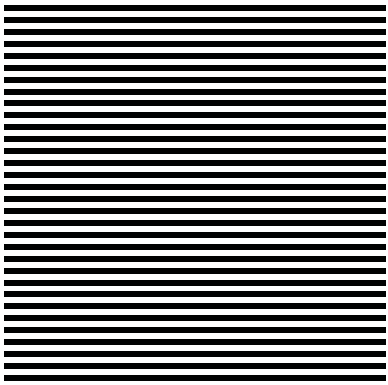
9.4.1. Widmo sygnału dwuwymiarowego

Uwaga: w prezentowanym przykładowym kodzie staramy się konsekwentnie pisać nazwy zmiennych małymi literami, jeśli oznaczają one sygnał, a wielkimi – jeśli oznaczają widmo.

9.4.1.1. Widma prostych obrazów

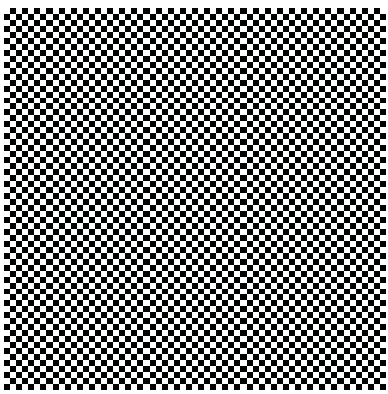
- Utwórz obraz o jednolitej jasności o rozmiarze 32×16 pikseli.
  ```
  img1=ones(32,16);
  ```
- Wyświetl obraz, oblicz i wyświetl jego widmo.
  ```
  imagesc(img1)
  IMG1=fft2(img1);
  figure;imagesc(abs(IMG1));
  ```
  Zanotuj rozmiar tablicy z transformatą, wartość maksimum widma i lokalizację maksimum (jako parę indeksów tablicy). Zanotuj
  Wskazówka: Użyj polecenia `colorbar` – zorientujesz się wtedy jaka jest skala wartości.
- Łatwiej interpretuje się widmo, gdy jest ono wyświetlone symetrycznie (zero częstotliwości w środku). Użyj polecenia `fftshift`, które zamienia miejscami dwie „połówki” wektora lub cztery „ćwiartki” macierzy.
  ```
  imagesc(fftshift(abs(IMG1)));
  ```
  Wskazówka: `fftshift` użyty drugi raz przywróci oryginalny układ ćwiartek widma (będzie to potrzebne, gdy w następnych zadaniach będziesz odtwarzać sygnał z jego widma).
  ***Odpowiedz na pytanie*:** Jakie wielkości reprezentowane są na osiach? Jak Odpowiedz

powinny być wyskalowane osie (podaj jedną z kilku możliwości)?

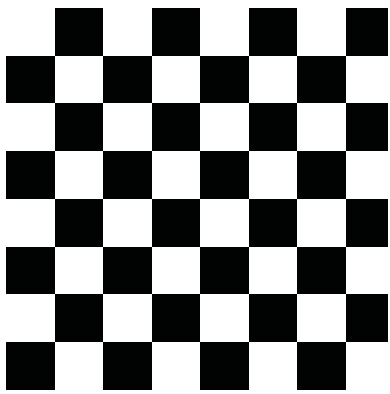


Rysunek 9.1. Poziome linie o wysokiej częstotliwości

Rysunek 9.2. Poziome linie o niskiej częstotliwości



Rysunek 9.3. Szachownica o wysokiej częstotliwości



Rysunek 9.4. Szachownica o niskiej częstotliwości

Wygeneruj obrazy podobne do przedstawionych na rys. 9.1 – 9.4. Dokładne parametry obrazów wybierz zgodnie z numerem stanowiska z tabeli 9.1.

> W porozumieniu z prowadzącym można obrazy takie wczytać, fotografując za pomocą kamery tablice kartonowe dostępne w laboratorium. Różnice parametrów linii między stanowiskami niech wtedy wynikają z fotografowania z różnej odległości – oczywiście nie uda się dokładnie wybrać ich z tabeli.

Jest wiele sprytnych sposobów utworzenia takich obrazów – jedną z możliwości jest przygotowanie wzorca, który będzie powielony odpowiednią liczbę razy.

Tabela 9.1

Nr stan.	1	2	3	4	5	6	7	8	9	10	11	12	13
Liczba wierszy w obrazie M	32	64	48	16	32	64	60	40	24	32	48	64	16
Liczba kolumn w obrazie N	16	32	48	48	32	24	60	48	64	16	36	48	32
Rozmiar szerokiego pasa/kratki [px]	4	8	8	4	8	16	5	8	6	8	6	8	4
Rozmiar wąskiego pasa/kratki [px]	1	2	3	2	4	4	3	1	4	1	3	4	2

`wzor=[0 0; 0 0; 1 1; 1 1]`

Aby uzyskać obraz 8×16, wzorzec 4×2 powielimy 2 razy w pionie i 8 razy w poziomie.

`IleKopiiPionowo=2; IleKopiiPoziomo=8;`

Matlab dostarcza specjalne polecenie do powielania macierzy.

`img2=repmat(wzor, IleKopiiPionowo, IleKopiiPoziomo);`

Innymi eleganckimi sposobami będzie: przemnożenie dwóch odpowiednio uformowanych wektorów, albo też utworzenie zerowej macierzy i wypełnienie jedynkami sprytnie zaindeksowanych fragmentów.

Dla każdego z 4 przypadków oblicz i obejrzyj widmo dwuwymiarowego sygnału – wyświetl je w układzie symetrycznym (zero pośrodku – czyli po użyciu `fftshift`). Aby zrozumieć, co oznaczają kolory – użyj `colorbar`.

***Odpowiedz na pytania*:** Odpowiedz

– Jakie składowe widoczne są na widmie? Opisz je, używając pojęcia częstotliwości przestrzennej (cz. pozioma, cz. pionowa).
– Czym różni się widmo sygnału „pasiastego" od widma sygnału „kraciastego"?
– Jakiemu sygnałowi (obrazowi) odpowiadają częstotliwości reprezentowane na skraju wykresu?

Odpowiedzi możesz udzielić słowami, albo poprzez naszkicowanie czterech widm i zaznaczenie na nich istotnych elementów z krótkim podpisem.

Obraz szachownicy o małej częstotliwości zachowaj w zmiennej o wybranej nazwie – będzie jeszcze potrzebny.

9.4.1.2. Widmo obrazu rzeczywistego

Przygotowanie obrazu do przetwarzania

W zadaniu tym zaleca się pracę na obrazie własnoręcznie uzyskanym z kamery.

Tematyka zdjęcia może być dowolna, natomiast zalecane jest, aby studenci na sąsiadujących stanowiskach wybrali różne tematy (twarz, widok sali, scenka na biurku, widok za oknem[3]) i następnie porównywali między sobą wyniki.

[3] Widok za oknem najwygodniej będzie wykorzystać studentowi siedzącemu koło okna. Jeśli okno jest brudne, będzie problem, bo w CS202 okna się nie otwierają, tylko uchylają.

Jeśli nie ma możliwości pozyskania „ciekawszego” obrazu, można posłużyć się obrazem znajdującym się w katalogu ćwiczenia – np. `Lenna.png`.

- Wczytaj obraz do zmiennej środowiska Matlab (np. `myimage`). Przygotuj go do przetwarzania (patrz 9.3.1).
- Wyświetl obraz w skali szarości (`gray` zamiast domyślnego mapowania wartości pikseli w skalę fałszywych barw `jet`).

```
imagesc(myimage);colormap(gray);colorbar;
```

Widmo obrazu

- Oblicz i obejrzyj widmo rzeczywistego dwuwymiarowego sygnału. Pamiętaj o użyciu `fftshift`, żeby częstotliwości zerowe były wyświetlane na środku widma.

Odpowiedz ? ***Odpowiedz na pytanie*:** Jakiej częstotliwości odpowiada dominująca próbka widma? Jakiej składowej odpowiada ona w badanym obrazie?
Widmo wyświetlone w skali liniowej jest zazwyczaj nieczytelne – zawiera jedną dominującą próbkę, przy której pozostałe są niewidoczne.

- Wyświetl widmo w skali logarytmicznej (wyskaluj je w dB).

Zawsze przy użyciu miary decybelowej należy zastanowić się, czy mamy do czynienia z amplitudą (lub stosunkiem amplitud) czy też z mocą (lub kwadratem amplitudy albo wariancją, ew. ich stosunkiem); w pierwszym przypadku obliczamy $20 \cdot \log_{10}(x)$, w drugim $10 \cdot \log_{10}(x)$.

Wskazówka: W Matlabie istnieje funkcja `dB`, zwracająca domyślnie `20*log10(x)`.

Odpowiedz ? ***Odpowiedz na pytanie*:** Czym różni się widmo badanego obrazu rzeczywistego od widm obrazów z rys. 9.1 – 9.4?

9.4.2. Filtrowanie sygnałów dwuwymiarowych

9.4.2.1. Przykładowe filtry liniowe

Filtry dolnoprzepustowe przepuszczają tylko składowe obrazu o niskiej częstotliwości. Składowe o wyższej częstotliwości są tłumione. Przykłady dwuwymiarowych filtrów dolnoprzepustowych:

```
lp1=[ 1, 1, 1;...
      1, 1, 1;...
      1, 1, 1];
```

```
lp2=[ 1, 1, 1;...
      1, 2, 1;...
      1, 1, 1];
```

```
lp3=[ 1, 2, 1;...
      2, 4, 2;...
      1, 2, 1];
```

```
lp4=[ 1, 1, 2, 1, 1;...
      1, 2, 4, 2, 1;...
      2, 4, 8, 4, 2;...
      1, 2, 4, 2, 1;...
      1, 1, 2, 1, 1];
```

Filtry górnoprzepustowe przepuszczają tylko składowe obrazu o wysokiej częstotliwości. Składowe o niższych częstotliwościach są tłumione. Przykłady dwuwymiarowych filtrów górnoprzepustowych:

```
hp1=[ -1, -1, -1;...
      -1,  9, -1;...
      -1, -1, -1];

hp2=[  0, -1,  0;...
      -1,  5, -1;...
       0, -1,  0];

hp3=[  1, -2,  1;...
      -2,  5, -2;...
       1, -2,  1];
```

Piksele obrazu zasadniczo reprezentowane są przez liczby dodatnie. Wskutek odejmowania liczb dodatnich mogą jednak powstać wartości mniejsze od zera. Decyzja, jak postąpić w takiej sytuacji (obciąć wartości ujemne, przesunąć wszystkie wartości w górę, zastąpić wartością bezwzględną,...) powinna zależeć od interpretacji danych w konkretnym przypadku. Jeśli obraz wyświetlany jest funkcją `imagesc`, jest on automatycznie przesuwany i skalowany co do wartości tak, aby najmniejszej wartości odpowiadał czarny piksel, a największej – biały.

Filtry wykrywające krawędzie polegają na odejmowaniu przesuniętych kopii obrazu. Liczba tych kopii i przesunięcie zależy od konkretnego filtru. Matematycznie jest to przybliżone obliczanie wartości pochodnej cząstkowej w danym kierunku.

Przykładowe filtry do wykrywania krawędzi:

```
edg1=[0,  0,  0;...
     -1,  1,  0;...
      0,  0,  0];

edg2=[-1,  0,  0;...
       0,  1,  0;...
       0,  0,  0];

edg3=[-1,  1,  1;...
      -1, -2,  1;...
      -1,  1,  1];

edg4=[-1, -1,  1;...
      -1, -2,  1;...
       1,  1,  1];

edg5=[0, -1,  0;...
     -1,  4, -1;...
      0, -1,  0];

edg6=[-1,  0, -1;...
       0,  4,  0;...
      -1,  0, -1];

edg7=[1,  2,  1;...
      0,  0,  0;...
     -1, -2, -1];
```

Jeżeli zależy nam na wykryciu krawędzi niezależnie od ich kierunku, można zastosować kombinację wyników filtracji w wielu kierunkach – np. sumować moduły (albo kwadraty) wyników filtracji.

9.4.2.2. Filtracja obrazu

Tabela 9.2

Nr stan.	1	2	3	4	5	6	7	8	9	10	11	12	13
Filtr dolnoprzepustowy	1	2	3	4	1	2	3	4	1	2	3	4	1
Filtr górnoprzepustowy	1	2	3	1	2	3	1	2	3	1	2	3	1
Filtr krawędziowy	1	2	3	4	5	6	7	1	2	3	4	5	6

Dla wczytanego w poprzednim zadaniu obrazu wykonaj filtrację, wykorzystując filtry przedstawione w punkcie 9.4.2.1, wybrane wg tabeli 9.2 dla Twojego stanowiska. ***Dla każdego filtru zanotuj jego współczynniki.***

Zanotuj

Dwuwymiarową filtrację wykonuje się, używając polecenia `filter2(h,x)`.

Polecenie `filter2` w Matlabie realizuje filtrację filtrem nieprzyczynowym – zero opóźnienia odpowiada środkowemu elementowi macierzy współczynników filtru.

Wyświetl obraz po filtracji, znajdź jego wartość maksymalną i minimalną. **Wskazówka:** `max(max(x))` i `min(min(x))` pomogą Ci znaleźć maksimum/minimum w dwuwymiarowej macierzy `x`.

Odpowiedz

***Odpowiedz na pytania*:** (dla każdego z 3 filtrów)

- Opisz słowami efekt filtracji (możesz, lecz nie musisz naszkicować w protokole wynik filtracji).
- Jak zmieniła się maksymalna wartość próbek obrazu? Pamiętaj o użyciu `colorbar`, bo `imagesc` skaluje obraz.
- Jak można by zmodyfikować współczynniki filtru, aby uniknąć drastycznej zmiany zakresu wartości próbek?

Dla jednego wybranego filtru wykonaj filtrację metodą mnożenia transformaty obrazu i transformaty odpowiedzi impulsowej filtru. Porównaj oba wyniki.
Wskazówka: Do mnożenia potrzebujesz obu transformat w tym samym rozmiarze. Gdy odpowiedź impulsowa filtru jest krótsza niż sygnał, musisz ją uzupełnić zerami do rozmiaru $M \times N$ (*zero-padding*), wywołując funkcję `fft2` z dodatkowymi parametrami:

```
X=fft2(x,M,N);
```

Zanotuj

Zanotuj zauważone różnice.

Zmierz czasy wykonania filtracji dla tych dwóch sposobów, powtarzając je w pętli (dla poprawy dokładności pomiaru czasu) np. po 10 razy. Pomiar wykonaj dla badanego obrazu rzeczywistego oraz dla zapamiętanej szachownicy z zadania 9.4.1.1 – zobaczysz jak czasy wykonania zależą od rozmiaru obrazu.

Zanotuj

Zanotuj te czasy oraz zapisz wybrany do testu filtr.

Wskazówka: Użyj funkcji `tic()` do uruchomienia „stopera” i `toc()` do jego zatrzymania.

***Odpowiedz na pytanie*:** Czy wyniki filtracji w obu metodach różnią się, czy też są jednakowe? Czy znajdujesz zastosowania gdzie jedna bądź druga metoda jest lepsza? Odpowiedz

9.4.3. Wykrywanie krawędzi i detekcja prostych

W praktyce automatycznego rozpoznawania elementów obrazów – np. w układzie „wzroku" robota poruszającego się w przestrzeni – często konieczne jest rozróżnianie kształtów. Podstawową operacją wykorzystywaną w tym celu jest identyfikacja odcinków prostych, stanowiących krawędzie kształtów.

W tym zadaniu użyjemy filtrów potrafiących wykrywać (wzmacniać) krawędzie, poznanych w poprzedniej części ćwiczenia, a następnie zastosujemy transformację Hougha do odnalezienia odcinków prostych w tak wzmocnionym obrazie.

Alternatywnie, student może (w porozumieniu z prowadzącym) zamiast 9.4.3.1 wykonać zadanie 9.4.3.2, gdzie porównuje się obrazy naczyń krwionośnych.

9.4.3.1. Wykrywanie krawędzi. Miniprojekt

W ramach miniprojektu przygotujemy obraz do wykrywania linii, wzmacniając wyrazistość krawędzi, a następnie zastosujemy gotowe procedury Matlaba implementujące:
- transformację Hougha (`hough()`),
- wizualizację wyniku (`plot_hough()`),
- wykrywanie maksimów w przekształconym obrazie (`houghpeaks()`),
- odnajdowanie w oryginalnym obrazie odcinków, odpowiadających odnalezionym maksimom (`houghlines()`),

i na koniec narysujemy odnalezione odcinki na oryginalnym obrazie (`plot_lines()`).

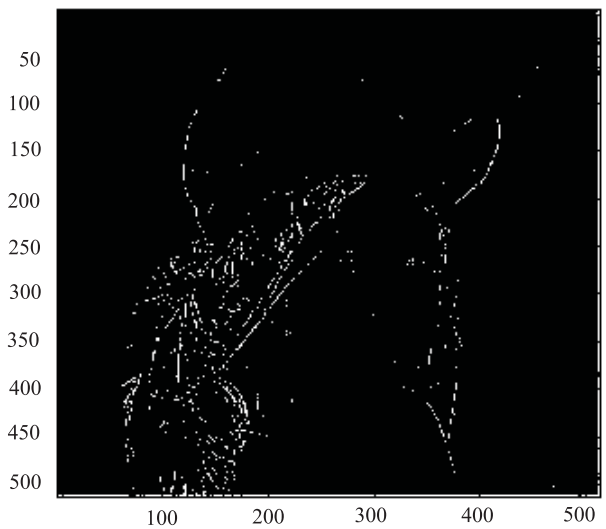
Procedura środowiska Matlab `hough()`, która implementuje transformację Hougha wymaga na wejściu macierzy binarnej (wypełnionej zerami i jedynkami), a więc obrazu dwubarwnego (piksele czarne i białe). W poniższym kodzie założono, że w zmiennej `BW` jest obraz oryginalny sprogowany do postaci binarnej (0 – czarny, 1 – biały).

Sprawdź, jak wygląda transformacja pojedynczego punktu i prostej:
- przygotuj obraz binarny z jednym jasnym punktem (a potem – z jedną jasną prostą pionową)
- Oblicz i wyświetl transformatę

```
[H,T,R] = hough(BW,'RhoResolution',0.5,'Theta',-90:0.5:89.5);
plot_hough(H,T,R);
```

Zanotuj parametry użytej linii (ρ i θ) i lokalizację odpowiadającego im punktu w transformacie. Zanotuj

- Wczytaj obraz, który będzie zawierał linie: zrób zdjęcie odpowiedniej sceny, ewentualnie – jeśli nie ma możliwości pozyskania indywidualnego zdjęcia – skorzystaj ze zdjęcia w katalogu ćwiczenia `'bariera.JPG'`.
  Jeśli zdjęcie jest kolorowe, wybierz jeden z kanałów lub przetwórz je na czarno-białe (jak w 9.4.1.2).
- Korzystając z wiedzy na temat filtrów detekujących krawędzie przetwórz obraz, i następnie wykonaj operację progowania do wymaganej macierzy binarnej (przykład – rys. 9.5).



Rysunek 9.5. Przykład obrazu przygotowanego do wywołania na nim funkcji `hough()`

Wskazówka: Konstrukcja `Ab=boolean(A>0.5)` zwróci macierz binarną z wartościami 1 dla indeksów, dla których wyrażenie logiczne jest prawdziwe. Obejrzyj wynik przetwarzania

```
figure(1)
imagesc(BW); colormap(gray); colorbar
```

Spróbuj dobrać sposób filtracji i parametr progowania, aby otrzymać możliwie najlepszy obraz binarny. ***Zanotuj parametry użytego filtru (oraz ewentualne inne parametry przetwarzania obrazu).***

Zanotuj

- Gdy już uzyskasz macierz BW, w której pozostały głównie linie, to wywołaj następujące polecenia w celu detekcji linii i ich wyświetlenia

```
[H,T,R] = hough(BW,'RhoResolution',0.5,'Theta',-90:0.5:89.5);
P = houghpeaks(H,9,'threshold',ceil(0.3*max(H(:))));
% współczynnik 0.3 w powyższej linii
%należy dobrać empirycznie
L = houghlines(BW,T,R,P,'FillGap',5,'MinLength',7);
figure(2)
```

```
plot_hough(H,T,R);
figure(3)
imshow(BW);
hold on
plot_lines(L);
hold off
```

Spróbuj ocenić, jak dobrze udało się wykryć linie:
– Czy wszystkie linie, które ludzkie oko dostrzega na obrazie zostały wykryte procedurą Hougha?
– Czy może pojawiły się fałszywe linie?
– W jakich przypadkach powstały błędy?

Spróbuj skomentować skuteczność tej metody wykrywania linii w obrazie. Zanotuj

9.4.3.2. Zadanie extra Analiza obrazu naczyń krwionośnych – miniprojekt

Przebieg naczyń krwionośnych (np. po wierzchniej stronie dłoni lub przedramienia) jest jedną z cech używanych do identyfikacji biometrycznej.

Spróbuj wykonać zdjęcie (jedno lub kilka – poeksperymentuj z oświetleniem!), a następnie przygotować obraz do wykrywania przebiegu naczyń:
– wybrać najlepszą warstwę barwną (R, G, B, a może jakaś kombinacja...),
– usunąć szumy i pozostałości skóry, włosów, zmarszczek... (patrz następne zadania!),
– wzmocnić linie o sensownej grubości i wybranych kierunkach (zastanów się nad ideą filtru dopasowanego),
– zastosować transformację Hougha.

Jeśli uda Ci się chociaż część z tych zadań, będzie to sukces. Jeśli dobrze wykona się transformacja Hougha, dostaniesz parametry, którymi wystarczy „nakarmić" jakieś algorytmy sztucznej inteligencji – i gotowe! (Oczywiście w naszym laboratorium tą częścią problemu się nie zajmujemy...).

Możesz też spróbować użyć idei filtracji dopasowanej do porównywania obrazów (ale nie jest to łatwe, są problemy np. ze skalowaniem itd. – dlatego chętniej przekształca się obraz do postaci parametrycznej i porównuje zestawy parametrów).

9.4.4. Usuwanie szumów

W tym zadaniu porównamy wyniki filtracji liniowej i nieliniowej (konkretnie – medianowej) w zastosowaniu do usuwania szumów różnego typu.

Filtrację medianową obrazu w środowisku Matlab można wykonać poleceniem `medfilt2`. Natomiast filtry oparte na innych statystykach porządkowych (w tym filtr maksymalny i minimalny) mogą być zrealizowane poleceniem `ordfilt2`.

9.4.4.1. Usuwanie addytywnego szumu typu „sól i pieprz"

Do wczytanego obrazu `image` dodaj szum addytywny poleceniem:

```
NoisyImage=double(imnoise(uint8(image),'salt & pepper',0.2));
```

Uwaga: Konwersje typów wynikają z założeń poczynionych przez programistów Matlaba, że obrazy będą typu `uint8`, i naszej potrzeby operowania na wartościach `double`.

- Zmierz średniokwadratowy stosunek sygnału do szumu dla otrzymanego obrazu

$$SNR_{ms} = \frac{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}[F'(i,j)]^2}{\sum_{i=0}^{N-1}\sum_{j=0}^{M-1}[F(i,j)-F'(i,j)]^2} \tag{9.1}$$

Zanotuj

gdzie F to obraz oryginalny, a F' – zaszumiony ***Zanotuj otrzymaną wartość***.
Wskazówka: Sumę kwadratów elementów macierzy (występującą we wzorze (9.1)) można oczywiście liczyć w pętli `for`, ale taka pętla jest nieefektywna z punktu widzenia zarówno programisty, jak i komputera. Znacznie szybciej obliczy się ją poprzez `sum(A(:).^2)` – zwróć uwagę na rozwijanie macierzy do wektora `(:)` i na podnoszenie do kwadratu „element po elemencie". Natomiast zapis `A(:)'*A(:)` będzie poprawny także dla liczb zespolonych, i na dokładkę bardzo elegancki[4].

- Spróbuj usunąć zakłócenie, używając zarówno filtrów dolnoprzepustowych, jak i filtru medianowego. Zmierz średniokwadratowy stosunek sygnału do szumu dla otrzymanych obrazów. ***Zanotuj otrzymane wartości i wykorzystane filtry***. ***Odpowiedz na pytanie*:** Który filtr lepiej usuwa szumy addytywne?

Zanotuj

Odpowiedz

9.4.4.2. Usuwanie szumu multiplikatywnego

Szum multiplikatywny polega na tym, że wartości próbek sygnału mnożone są przez wartość próbki szumu. Takie zjawisko może zachodzić, gdy wzmocnienie w torze sygnału zmienia się losowo (np. losowa zmiana czułości pikseli w kamerze). Również efekty nierównomiernego oświetlenia (światło i cień) są zakłóceniami multiplikatywnymi.

- Do wczytanego obrazu dodaj szum multiplikatywny poleceniem:

```
MulNoisyImage=double(imnoise(uint8(image),'speckle'));
```

- Spróbuj usunąć zakłócenie, używając zarówno filtrów dolnoprzepustowych, jak i filtru medianowego. Zmierz średniokwadratowy stosunek sygnału do szumu dla otrzymanych obrazów. ***Zanotuj otrzymane wartości i wykorzystane filtry***. ***Odpowiedz na pytanie*:** Który filtr lepiej usuwa szumy multiplikatywne?
Wskazówka: Szum multiplikatywny można też usuwać po **zlogarytmowaniu obrazu** – wtedy szum multiplikatywny zmienia się w szum addytywny[5].

Zanotuj

Odpowiedz

[4] W zagadnieniach matematycznych „elegancja" jest pojęciem oznaczającym mniej więcej „uniwersalność i prostota zapisu".

[5] Tę technikę nazywa się w literaturze przetwarzaniem homomorficznym, i stosuje także do usuwania efektów oświetlenia.

Zadanie extra Spróbuj zastosować przetwarzanie homomorficzne, ale nie zapomnij przed wyświetleniem dokonać operacji odwrotnej do logarytmowania!

9.4.5. Proste metody kompresji obrazu

9.4.5.1. Kompresja obrazu przez obcinanie widma. Miniprojekt

- Dla wczytanego obrazu wyznacz widmo.
- Wybierz współczynnik kompresji k z przedziału 0 do 1. Dla wybranej wartości współczynnika kompresji wytnij ze środka widma koło, tak by stosunek jego pola powierzchni do pola powierzchni całego widma wynosił k (zakładamy nadal, że środek widma odpowiada składowej stałej – pamiętaj o użyciu `fftshift`). Kołową (lub eliptyczną – dla $M \neq N$) maskę zerojedynkową o promieniu `r` i rozmiarach `M, N` można uzyskać za pomocą funkcji `LCPS_image_mask(r,M,N);`, gdzie $r \in (0, 1)$. Funkcja ta została napisana na potrzeby tego ćwiczenia (rys. 9.6).

```
>> image_mask(0.5,16,16)
ans =
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   1   1   1   1   1   0   0   0   0   0
0   0   0   0   0   1   1   1   1   1   1   1   0   0   0   0
0   0   0   0   0   1   1   1   1   1   1   1   0   0   0   0
0   0   0   0   0   1   1   1   1   1   1   1   0   0   0   0
0   0   0   0   0   1   1   1   1   1   1   1   0   0   0   0
0   0   0   0   0   1   1   1   1   1   1   1   0   0   0   0
0   0   0   0   0   0   1   1   1   1   1   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
>>
```

Rysunek 9.6. Wynik działania polecenia image_mask

Sprawdź, czy rzeczywiście pozostawiłeś w masce $k \cdot M \cdot N$ jedynek.
`sum(maska(:))/numel(maska)`

- Odtwórz obraz z uciętego widma.
- Wyznacz średniokwadratowy stosunek sygnału do szumu SNR_{ms}. Wyraź go w decybelach[6].
- Powtórz procedurę dla trzech różnych różnych wartości współczynnika kompresji k i ***zanotuj wyniki***.

Zanotuj

[6] Zastanów się, jakiego wyrażenia użyć: $20 \cdot \log_{10}(x)$, czy $10 \cdot \log_{10}(x)$.

Zadanie extra Możliwe jest uruchomienie procedury w pętli, zebranie informacji do wektora i wyświetlenie wyniku w postaci wykresu $\mathrm{SNR}_{ms}(k)$.

Zanotuj

Na wykresie znajdź i ***zanotuj wartość współczynnika kompresji***, dla której stosunek sygnału do szumu wynosi w przybliżeniu K dB, gdzie $K = 10 + 2 \cdot numerstanowiska$.

Odpowiedz

***Odpowiedz na pytanie*:** Jak można by ocenić błąd kompresji *bez* odtwarzania obrazu? **Wskazówka:** Przypomnij sobie twierdzenie Parsevala.

9.4.5.2. Zadanie extra Kompresja obrazu z wykorzystaniem transformacji falkowej

- Przygotuj filtry odpowiadające falce Daubechies stopnia 8
```
[LoD, HiD, LoR, HiR] = LCPS_daubechies(8);
```
- Dokonaj dekompozycji
```
[wv idx] = LCPS_dwt2(img, 1, LoD, HiD);
```
- Wyświetl zdekomponowany obraz
```
imagesc(wv);
```
Wskazówka: Obraz zdekomponowany może być wyraźniejszy, jeśli wyświetlimy jego moduł.
- Zrekonstruuj obraz idealnie oraz po usunięciu składowych szybkozmiennych:
```
img_idrec = LCPS_idwt2(wv, idx, LoR, HiR);

wvc=zeros(size(wv));
wvc(1:end/2,1:end/2)=wv(1:end/2,1:end/2);
img_crec = LCPS_idwt2(wvc, idx, LoR, HiR);
```
- Wyświetl oba obrazy

Zanotuj

Oblicz i zanotuj średniokwadratowy stosunek sygnału do szumu po rekonstrukcji.

9.4.5.3. Zadanie extra Miniprojekt – wyznaczenie progu dla zadanego stopnia kompresji

Analizując histogram wartości współczynników, dla zadanego stopnia kompresji wyznacz wartość progową `lambda`, poniżej której współczynniki będą zerowane.

Wyzeruj te współczynniki:

```
cH(abs(cH) <= lambda) = 0.0;
cV(abs(cV) <= lambda) = 0.0;
cD(abs(cD) <= lambda) = 0.0;
```

a następnie odtwórz obraz.

Zanotuj

Oblicz i zanotuj średniokwadratowy stosunek sygnału do szumu po rekonstrukcji.

Przeanalizuj oddzielnie histogramy wartości współczynników w 4 grupach składowych; spróbuj zastosować oddzielne progi dla każdej grupy. Sprawdź, na ile zmniejszyło to błąd rekonstrukcji.

Jeżeli nie uda się uzyskać zadanego stopnia kompresji, użyj dekompozycji w dwóch lub więcej skalach (`Kscales=2`).

```
[wv idx] = LCPS_dwt2(img, Kscales, LoD, HiD);
```

9.4.6. Zakres dynamiczny obrazów

9.4.6.1. Rozciąganie zakresu dynamicznego poszczególnych kanałów

- Dla wczytanego kolorowego obrazu wyznacz histogramy dla poszczególnych kanałów. Można do tego wykorzystać gotowe narzędzie w postaci polecenia `imhist`.
  Wskazówka: Dobrze jest wykreślić wszystkie histogramy na jednym oknie, wykorzystując polecenie `subplot`.
- Rozciągnij zakres dynamiczny poszczególnych kanałów obrazu tak, by wykorzystany był ich cały zakres dynamiczny.
- Wyświetl poprawiony obraz. Porównaj z obrazem oryginalnym.
  ***Odpowiedz na pytanie*:** Opisz swoje wrażenia po rozciągnięciu zakresu dynamicznego obrazu. Czy taki zabieg ma wpływ na wierność odtworzenia kolorów? Odpowiedz ?