Uniwersytet Rzeszowski

Wydział Nauk Ścisłych i Technicznych

Instytut Informatyki

BIOMETRYCZNE SYSTEMY ZABEZPIECZEŃ

INSTRUKCJA DO ĆWICZEŃ LABORATORYJNYCH

TREŚCI KSZTAŁCENIA: DETEKCJA KRAWĘDZI W OBRAZIE METODAMI SPLOTOWYMI

Spis treści

1. Cele laboratorium

Filtracja splotowa to jedno z podstawowych narzędzi przetwarzania obrazów, wykorzystywane do wygładzania, wykrywania krawędzi, redukcji szumów oraz innych operacji poprawiających jakość obrazu. Splot (ang. convolution) polega na przekształceniu obrazu poprzez nałożenie na niego maski (jądra) filtrującego. Celem laboratorium jest implementacja algorytmu splotowego w MATLAB-ie oraz zbadanie wpływu różnych filtrów na obraz.

2. Wprowadzenie

Zadanie wyodrębnienia w obrazie obszarów, w których istnieje gradient jasności o pewnej określonej i dużej wartości jest jednym z podstawowych działań w ramach wstępnego przetwarzania obrazu. Inaczej mówiąc są to punkty rozdzielające np. dwa obszary, z których każdy posiada inną średnią wartość jasności. Krawędzie w obrazie, będą więc zwykle stanowiły linie zawierające istotne informacje o kształcie rozpoznawanego obiektu. Takie nieciągłości jasności zwykle powstają w naturalnych obrazach z powodu zmian właściwości powierzchni, zmian w oświetleniu czy też cieni. Ogólnie, filtry wykrywające krawędzie, dzieli się na dwie podstawowe grupy:

- lokalnego dopasowania, (local template fitting).
- wykorzystujące operatory gradientowe, (gradient operators)

W pierwszej grupie obok charakterystycznych krawędziowych filtrów kierunkowych Sobela, Prewitta czy też Kirscha, wymienić należy operator Frei-Chen. Jak nazwa kategorii tej grupy filtrów wskazuje, wykrywanie linii polega tutaj na projekcji maski filtru na wybrany punkt obrazu wraz z jego sąsiedztwem zwykle 8-mio lub 24-ro pikslowym. Z matematycznego punktu widzenia, jest to klasyczny zabieg konwolucji (inaczej splotu-zamiennie wykorzystywane są oba określenia) dwóch funkcji $g(x)$ i $h(y)$ zmiennej x i y odpowiednio.

$$(g * h)(x) = \int_{-\infty}^{\infty} g(x-y)h(y)dy \tag{1}$$

Zależność tę możemy zapisać w formie dyskretnej jako proces filtracji sygnału wejściowego $g(m)$ przy pomocy filtru opisanego współczynnikami $h(n)$:

$$f(n) = \sum_{m=-\infty}^{\infty} g(n-m)h(m) \tag{2}$$

Zwykle oznacza się krótko splot jako:

$$f = g * h \tag{3}$$

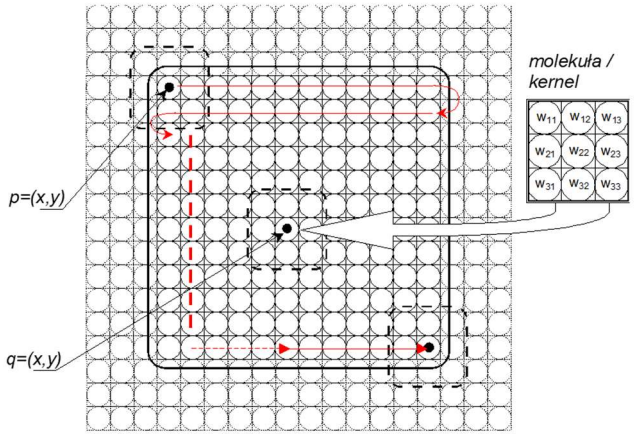
W dziedzinie przetwarzania obrazów liniową konwolucję opisuje się jako proces mapowania dwóch obrazów w jeden obraz wyjściowy. I tak, dla obrazu g, o rozmiarach $M \times N$ splot z kernelem h (lub inaczej molekułą, ang. Computational molecule or convolution kernel) o rozmiarach $m \times n$, gdzie zwykle:

$$m = n = 2k + 1, \text{ gdzie } k \in C^{+} \tag{4}$$

zapiszemy następująco:

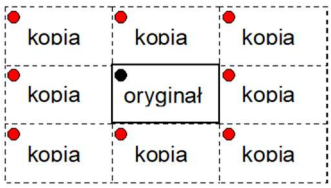
$$(g * h)(x, y) = \frac{1}{mn} \sum_{i=-k}^{k} \sum_{j=-k}^{k} g(x-i, y-j) \cdot h(i, j) \tag{5}$$

Rozmiar obrazu wyjściowego jest $M \times N$, zaś $1 \leq x \leq M$ oraz $1 \leq y \leq N$, przy czym zwykle $M = N$. Procedurę konwolucji można przedstawić graficznie jak na Rys. 1:



Rys. 1. Graficzna prezentacja konwolucji.

Obraz wejściowy to piksele otoczone linią ciągłą. Linią przerywaną zaznaczono te obszary obrazu, które powstają poprzez tzw. zawijanie obrazu (image wraping). Jest to niezbędny zabieg umożliwiający przetworzenie punktów brzegowych obrazu.



Rys. 2. Sposób zawijania obrazu.

Mechanizm zawijania przedstawiono na Rys. 2. Symbol ● oznacza piksel w lewym górnym narożniku obrazu. Linią przerywaną zaznaczono kopie oryginału. W praktyce można również uzupełnić potrzebne do konwolucji brzegi obrazu poprzez zwierciadlane odbicie oryginału lub dokonując ekstrapolacji w zakresie niezbędnym do przeprowadzania procedury splotu. Wracając do realizacji operacji splotu umieśćmy dla przykładu wartość jasności wybranego piksela w obrazie wyjściowym $f_q(x,y)$. Można wyliczyć ją następująco:

$$f_q(x,y) = w_{11}g(x-1,y-1) + w_{12}g(x-1,y) + w_{13}g(x-1,y+1) + \\ +w_{21}g(x,y-1) + w_{22}g(x,y) + w_{23}g(x,y+1) + \\ +w_{31}g(x+1,y-1) + w_{32}g(x+1,y) + w_{33}g(x+1,y+1) \quad (6)$$

Jak łatwo można zauważyć, zależnie od wartości poszczególnych wag maski filtru $w_{i,j}$, zaprojektowany filtr będzie realizował detekcję różnych cech obrazu, na przykład wyszukiwanie krawędzi albo linii.

Druga grupa detektorów tzw. gradientowych opiera się na popularnych filtrach Marr-Hildreth, Canny, Deriche czy też Mallat. Otóż jak zauważyli w swojej pracy Marr i Hildreth, natura obrazów, a mianowicie różnorodność zmian jasności od przejść łagodnych poprzez odbicia cienie szum i ostre krawędzie, sprawia, że spektrum częstotliwościowe obrazu jest bardzo szerokie. Konsekwencją tego jest szeroki rozkład widma częstotliwości takiego obrazu. Wniosek, który sformułowali był następujący: obraz należy wstępnie wygładzić tak, by jak najmniej szczegółów zostało zniekształconych i dalej drogą projektowania filtrów pracujących w poszczególnych pasmach częstotliwości analizować obraz. W etapie końcowym byłaby zaś dokonywana synteza informacji zebranej w poszczególnych kanałach częstotliwościowych. Ta teoria znalazła swoje podobne zastosowanie w transformacie falkowej

chętnie wykorzystywanej w częstotliwościowej analizie obrazów i w nowoczesnych technikach kompresji obrazów.

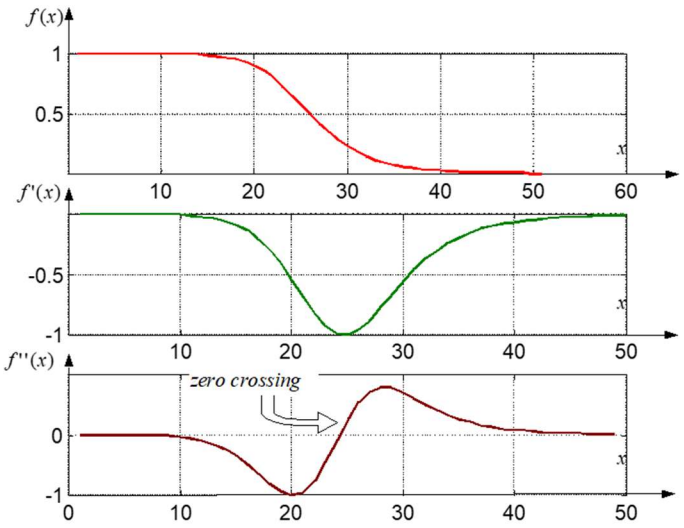
Jak zauważyli Marr i Hildreth filtr, który realizowałby wstępne wygładzenie powinien redukować zakres częstotliwości do wybranego kanału. Optymalnym rozkładem spełniającym te warunki jest Gaussian:

$$G(x) = \left(\frac{1}{\sigma\sqrt{2\Pi}}\right) exp\left(\frac{-x^2}{2\sigma^2}\right) \tag{7}$$

gdzie parametr σjest standardową dewiacją. Konsekwentnie dopełnieniem tej postaci filtru jest jego laplasjan:

$$G''(x) = \left(\frac{1}{\sigma^3\sqrt{2\Pi}}\right)\left(1 - \frac{x^2}{\sigma^2}\right) exp\left(\frac{-x^2}{2\sigma^2}\right) \tag{8}$$

Jest to klasyczny operator laplasjanowy wykorzystywany w przetwarzaniu obrazów. Na Rys. 3 przedstawiono przykładowy wynik symulacji demonstrującej mechanizm wykrywania przejść przez zero w drugiej pochodnej sygnału wejściowego.



Rys. 3. Mechanizm powstawania zero-crossings.

Jak łatwo zauważyć obraz, który zostanie poddany działaniu takiego filtru, umożliwi wyszukanie krawędzi w sygnale wejściowym $f(x)$, poprzez detekcję przejść przez zero w sygnale wyjściowym $f''(x)$ (ang. *zero crossings detection*).

Operację tę można by zapisać uwzględniając wstępne wygładzanie obrazu jako:

$$f(x,y) = \Delta\big(G * I(x,y)\big) \tag{9}$$

gdzie: Δ - operator laplasjanowy, $I(x,y)$ – obraz wejściowy.

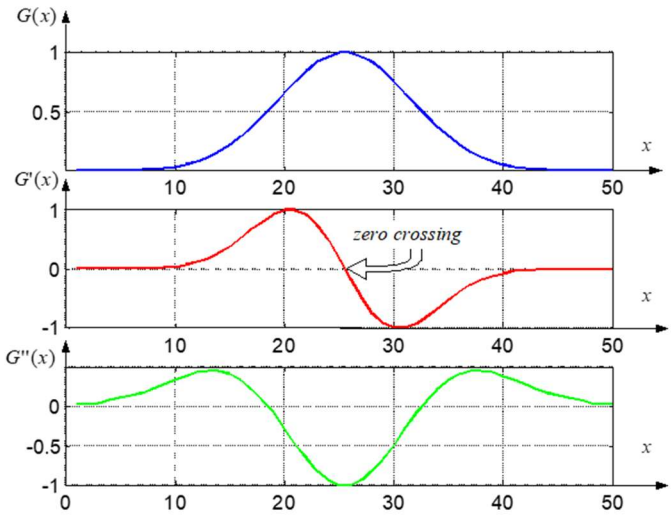
Przepisując tę zależność otrzymujemy postać:

$$f(x,y) = \Delta\mathrm{G} * I(x,y) \tag{10}$$

W formie cylindrycznej operator ΔG moglibyśmy wyrazić następująco:

$$\Delta\mathrm{G} = \mathrm{G}''(x,y) = \left(\frac{-1}{\sigma^3\sqrt{2\Pi}}\right)\left(1 - \frac{x^2+y^2}{\sigma^2}\right) exp\left(\frac{-x^2-y^2}{2\sigma^2}\right) \tag{11}$$

Opierając się na takim zapisie filtru, w literaturze wprowadzono określenie LOG (od ang. skrótu Laplacian Of Gaussian). Jak wyglądać będzie odpowiedź tego filtru na sygnał wejściowy zaprezentowano na Rys. 4.



Rys. 4. Profile filtrów Gaussianowych

Algorytm filtracji splotowej

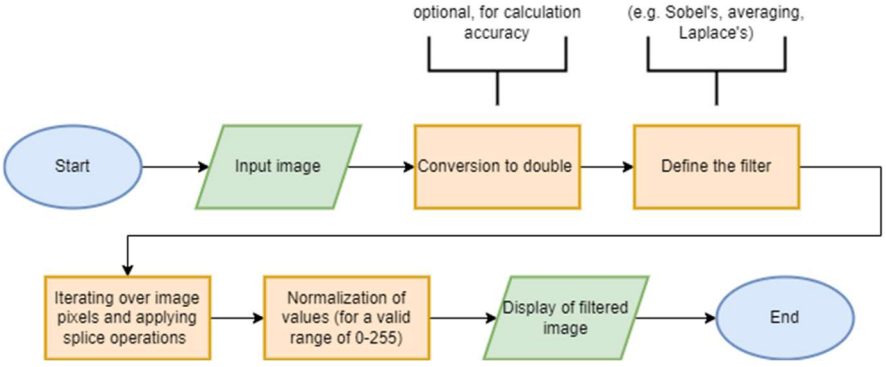
Splot 2D (dla obrazów) jest definiowany jako:

$$g(x,y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(x-i, y-j) \cdot h(i,j) \tag{12}$$

gdzie: $f(x,y)$ – obraz wejściowy, $h(i,j)$ – jądro (maska) filtra o wymiarach $(2k+1) \times (2k+1)$, g(x,y)– obraz przefiltrowany. Operacja ta polega na przesuwaniu maski filtra po obrazie i obliczaniu wartości nowych pikseli jako ważonej sumy otoczenia. Podstawowe kroki algorytmu dotyczą:

- Wczytanie obrazu rastrowego
- Definicja maski filtra (np. filtr uśredniający, Sobela, Laplace'a).
- Iteracyjne stosowanie splotu na każdym pikselu obrazu.
- Zapis i wizualizacja przefiltrowanego obrazu.

Na Rys. 5 przedstawiono schemat blokowy algorytmu filtracji splotowej.



Rys. 5. Schemat blokowy algorytmy filtracji splotowej.

Przykładowa implementacja

```
clc; clear; close all;

% Wczytanie obrazu
I = imread('cameraman.tif'); % Przykładowy obraz w MATLAB-ie
I = double(I); % Konwersja na double dla obliczeń
```

```
% Definicja filtru (np. filtr uśredniający 3x3)
h = ones(3,3) / 9;

% Wykonanie filtracji splotowej
filtered_image = conv2(I, h, 'same');

% Wyświetlenie oryginalnego i przefiltrowanego obrazu
figure;
subplot(1,2,1);
imshow(uint8(I)); title('Oryginalny obraz');

subplot(1,2,2);
imshow(uint8(filtered_image)); title('Obraz po filtracji');
```

Otrzymany rezultat:

Rys. 6. Otrzymany wynik filtracji.

3. Zadania do samodzielnego rozwiązania

Zadanie 1.

- Wykorzystując dowolny obraz biometryczny tęczówki oka wykonaj program dokonujący filtracji splotowej filtrami Robertsa, Prewitta, Sobela oraz Laplace'a. Uwzględnij w programie możliwość zadawania rozmiaru maski kernela 3x3 oraz 5x5.
- Wyświetl wyniki filtracji w oknie wykorzystując funkcję subplot.
- Porównaj efekty filtracji dla różnych filtrów. Sprawdź wpływ zmiany rozmiaru maski filtra na wynik końcowy.
- Wprowadź do obrazu zadaną porcję szumu *Salt&Pepper* wykorzystując funkcję imnoise.
- Przeprowadź na obrazie wejściowym filtrację medianową i uśredniającą a następnie porównaj otrzymane wyniki. Ponów procedurę wykrycia krawędzi w obrazie; przedstaw w sprawozdaniu dyskusję otrzymanych wyników.
- Wczytaj obraz kolorowy (peppers.png) i zastosuj filtrację na każdej składowej R, G, B osobno. Połącz przefiltrowane składowe w nowy obraz i wyświetl wynik.
- Sprawdź, jak działa imfilter(I, h, 'same') w porównaniu do conv2(). Czy wyniki są identyczne? Jakie są różnice?

Implementacja masek:

Maska Robertsa

0	0	0
-1	0	0
0	1	0

Maska Prewitta

-1	-1	-1
0	0	0
1	1	1

Maska Sobela

-1	-2	-1
0	0	0
1	2	1

Maski Laplace'a

0	-1	0
-1	4	-1
0	-1	0

-1	-2	-1
-2	4	-2
-1	-2	-1