TEKNIK PENGOLAHAN CITRA

Kuliah 5 – Edge Sharpening

Indah Susilawati, S.T., M.Eng.

Program Studi Teknik Elektro

Program Studi Teknik Informatika

Fakultas Teknik dan Ilmu Komputer

Universitas Mercu Buana Yogyakarta

2009

KULIAH 5

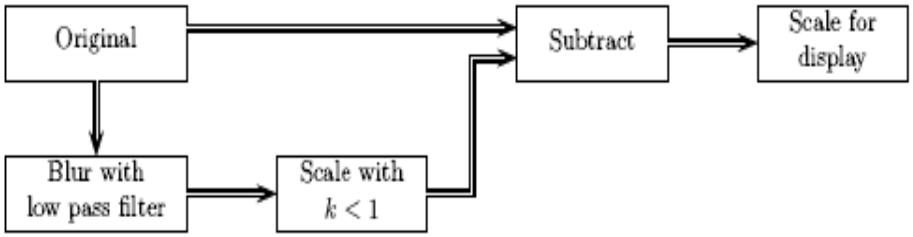
TEKNIK PENGOLAHAN CITRA

PENAJAMAN TEBING (EDGE SHARPENING)

Spatial filtering dapat digunakan untuk membuat tebing-tebing dalam sebuah citra menjadi terlihat lebih tajam (*sharper*), sehingga lebih menyenangkan dalam pandangan manusia. Operasi ini sering disebut "*edge enhancement*" atau "*edge crispening*" atau "*unsharp masking*".

Unsharp Masking

Ide dasar dari *unsharp masking* adalah untuk mengurangkan versi kabur terskala (*scaled unsharp version*) dari sebuah citra terhadap citra aslinya. Secara praktis, dapat dilakukan dengan mengurangkan citra yang telah dikaburkan dan diskala (*scaled blurred image*) terhadap citra aslinya. Perhatikan skema *unsharp masking* berikut.



Citra asli (original)

Citra kabur (blur)

Misalkan ada sebuah citra, maka *unsharp masking* dapat diterapkan menggunakan perintah-perintah berikut.

>> f = fspecial (‘average’);

>> xf = filter2 (f, x);

>> xu = double (x) – xf/1.5;

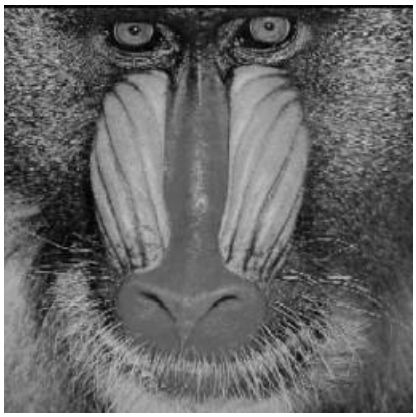
>> imshow (xu/70);

Perintah pada baris terakhir digunakan untuk *scalling* hasil sehingga fungsi imshow.m dapat menampilkan citra; nilai skala mungkin harus diubah bergantung citra input.

Contoh implementasi.

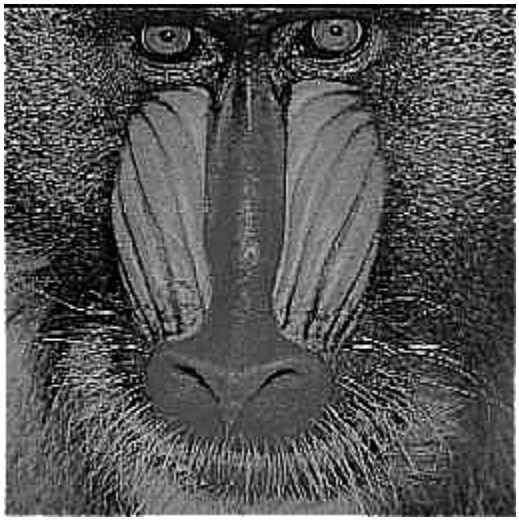
```
clear all;
clc;
I = imread ('mandrill2.jpg');
f = fspecial('average');
If = filter2(f,I);
Iu = double(I)-If/1.25
figure, imshow (I)
title ('citra input')
figure, imshow (mat2gray(If))
title ('citra blurred')
figure, imshow (Iu/65)
title ('citra unsharp masking')
```

citra input

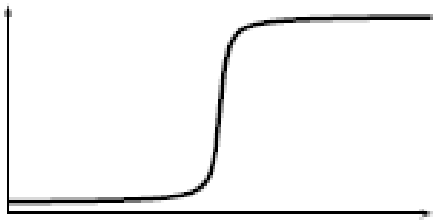


citra blurred

citra unsharp masking

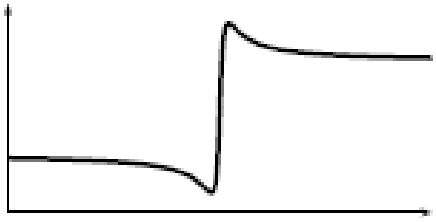


Gambaran secara grafis proses *unsharp masking* diperlihatkan pada gambar berikut.



(a) Pixel values over an edge

(b) The edge blurred



(c) (a) − k(b)

Operasi *filtering* dan pengurangan dapat dilakukan sekaligus dengan menggunakan linearitas filter 3 x 3 berikut, yang disebut filter identitas.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Dengan demikian, *unsharp masking* dapat diterapkan menggunakan filter tersebut dengan formula sbb.

$$f = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{k}\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

Dengan k adalah konstanta yang dipilih untuk memperoleh hasil yang terbaik. Alternatif lain adalah dengan mendefinisikan filter *unsharp masking* sebagai berikut.

$$f = k\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

Pada formula yang kedua tsb, dilakukan pengurangan citra yang dikaburkan terhadap versi terskala citra aslinya. Faktor skala juga dapat dipecah di antara filter identitas dan filter *blurring* (kabur).

Pilihan *unsharp* pada fungsi matlab fspecial, dibuat menggunakan formula

$$\frac{1}{\alpha+1}\begin{bmatrix} -\alpha & \alpha-1 & -\alpha \\ \alpha-1 & \alpha+5 & \alpha-1 \\ -\alpha & \alpha-1 & -\alpha \end{bmatrix}$$

Dengan α adalah parameter opsional dengan default 0,2. Jika α = 0,5 maka filter yang terbentuk menjadi

$$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ -1 & 11 & -1 \\ -1 & -1 & -1 \end{bmatrix} = 4\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} - 3\begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

Meskipun pada pembahasan ini digunakan filter pererata, namun pada dasarnya dapat digunakan sebarang LPF untuk proses *unsharp masking*.

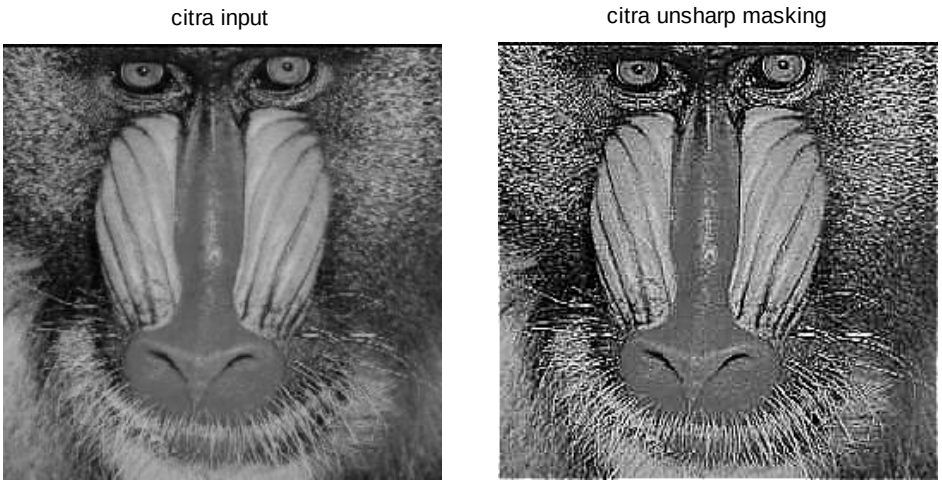
Matlab menyediakan opsi atau pilihan *unsharp* pada fungsi fspecial untuk *unsharp filtering.*

Sintaks : H = FSPECIAL ('unsharp', ALPHA)

FSPECIAL membuat *unsharp filter* menggunakan *Laplacian filter* dengan parameter ALPHA berada pada jangkauan 0.0 hingga 1.0. Default nilai ALPHA adalah 0.2.

Contoh implementasi.

```
clear all;
clc;
I = imread ('mandrill2.jpg');
f = fspecial('unsharp');
If = filter2(f,I);
figure, imshow (I)
title ('citra input')
figure, imshow (If/255)
title ('citra unsharp masking')
```



HIGH BOOST FILTERING

Sejenis dengan *filter unsharp masking* adalah filter *high boost*, yang diperoleh dengan menggunakan formula

high boost = A x (original) – (*low pass*)

Dengan A adalah faktor penguatan. Jika A = 1, maka filter *high boost* menjadi filter HPF sederhana. Jika diambil filter pererata 3 x 3 sebagai filter LPF, maka filter *high boost* akan menjadi berbentuk sbb.

$$\frac{1}{9}\begin{bmatrix} -1 & -1 & -1 \\ -1 & z & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Dengan z > 8. Jika z = 11, diperoleh proses *filtering* yang sangat mirip dengan *unsharp masking*, namun dengan faktor skala yang berbeda. Formula untuk *high boost* juga dapat dinyatakan sebagai

high boost = A *(original) – (low pass)*

= A *(original) – [(original) – (high pass)]*

= (A – 1) *(original) + (high pass)*

Hasil terbaik untuk *high boost filtering* dapat diperoleh dengan mengalikan persamaan dengan faktor w sehingga

$$wA - w = 1$$

atau

$$w = \frac{1}{A-1}$$

Sehingga formula *unsharp masking* secara umum dapat dinyatakan sebagai

$$\frac{A}{A-1}(original) - \frac{1}{A-1}(low\ pass)$$

Versi lain dari formula ini juga dinyatakan dengan

$$\frac{A}{2A-1}(original) - \frac{1-A}{2A-1}(low\ pass)$$

Untuk hasil terbaik, A dipilih sedemikian sehingga

$$\frac{3}{5} \le A \le \frac{5}{6}$$

Jika A = 3/5 maka

$$\frac{3/5}{2(3/5)-1}(original) - \frac{1-(3/5)}{2(3/5)-1}(low\ pass) \quad = \quad 3(original) - 2(low\ pass)$$

Jika A = 5/6 maka

$$\frac{5}{4}(original) - \frac{1}{4}(low\ pass)$$

Filter *high boost* juga dapat dibuat menggunakan filter identitas dan filter pererata. Silakan mencoba untuk implementasinya dalam Matlab.

Filter Non Linear

Filter yang telah dipelajari sebelumnya adalah filter-filter linear, cukup mudah dipelajari, dapat diaplikasikan dengan cepat dan efisien menggunakan Matlab.

Filter non linear diperoleh dengan menggunakan fungsi non linear yang dikenakan pada nilai-nilai piksel dalam “mask”. Contoh sederhana dari filter non linear adalah filter maksimum (*maximum filter*) dan filter minimum (*minimum filter*). Filter maksimum mengasilkan output berupa nilai piksel maksimum dalam *mask*, dan sebaliknya filter minimum menghasilkan output berupa nilai piksel minimum dalam *mask*.

Baik filter maksimum maupun filter minimum termasuk dalam golongan *rank-order filters*. Dalam filter yang demikian, elemen-elemen yang berada di dalam “*mask*” akan diurutkan (*ordered*) dan satu nilai tertentu dikeluarkan sebagai output. Dengan demikian, jika nilai-nilai tersebut diurut menaik, maka filter minimum adalah *rank-order filter* dimana elemen pertamanya dikeluarkan sebagai output; sedangkan filter maksimum merupakan *rank-order filter* dimana elemen terakhirnya yang dikeluarkan sebagai output.

Untuk implementasi filter non linear dalam Matlab dapat digunakan fungsi nlfilter.m, yang akan menerapkan filter pada suatu citra menurut suatu fungsi yang telah

ditentukan sebelumnya. Berikut adalah salah satu contoh implementasi filter non linear dengan menerapkan fungsi maksimum.

Sintaks: B = NLFILTER(A, [M N], FUN)

Dengan A adalah matriks citra, [M, N] adalah ukuran filter, dan FUN adalah fungsi yang akan diterapkan.

```
clear all;
clc;
I = imread ('cameraman.tif');
If = nlfilter (I, [3,3],'max(x(:))');
subplot (1,2,1),, imshow (I)
title ('citra input')
subplot (1,2,2),, imshow (If)
title ('citra filter maks')
```

citra input

citra filter maks

```
clear all;
clc;
I = imread ('cameraman.tif');
If = nlfilter (I, [3,3],'min(x(:))');
subplot (1,2,1),, imshow (I)
title ('citra input')
subplot (1,2,2),, imshow (If)
title ('citra filter minimum')
```

citra input

citra filter minimum

Menggunakan fungsi nlfilter.m dapat menjadi sangat lambat terutama untuk citra dengan ukuran besar. Alternatif yang lain adalah menggunakan fungsi colfilt.m, dengan fungsi ini proses menjadi lebih cepat. Mengapa?

Rank-order filter lain yang sangat penting adalah filter median. Pada filter median maka nilai tengah dari nilai-nilai piksel yang telah diurutkan akan dikeluarkan sebagai output. Filter ini banyak sekali digunakan sehingga Matlab menyediakan fungsi untuk filter median secara khusus, yaitu medfilt2.m.

Contoh implementasi

```
clear all;
clc;
I = imread ('cameraman.tif');
Imed = medfilt2(I);
subplot (1,2,1),, imshow (I)
title ('citra input')
subplot (1,2,2),, imshow (Imed)
title ('citra filter median')
```

citra input

citra filter median

Silakan dicoba apakah dapat menerapkan filter median menggunakan fungsi nlfilter.m atau fungsi yang lain.