TEKNIK PENGOLAHAN CITRA

Kuliah 10 – Removal of Periodic Noise Dan Segmentasi

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

Program Studi Teknik Informatika/Sistem Informasi

Fakultas Teknologi Informasi

Universitas Mercu Buana Yogyakarta

2014

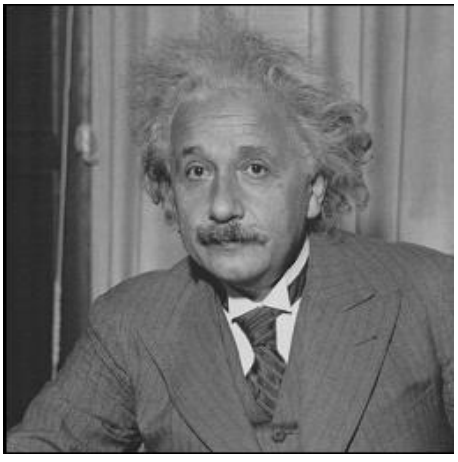
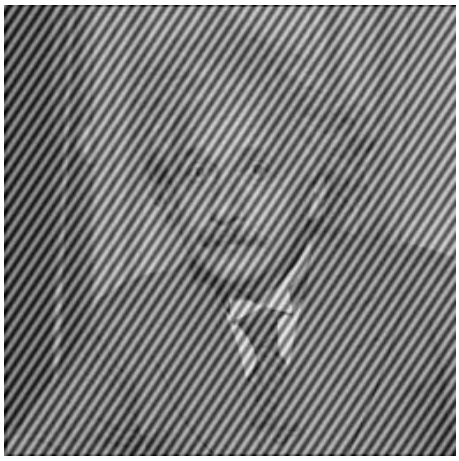
KULIAH 10

TEKNIK PENGOLAHAN CITRA

REMOVAL OF PERIODIC NOISE

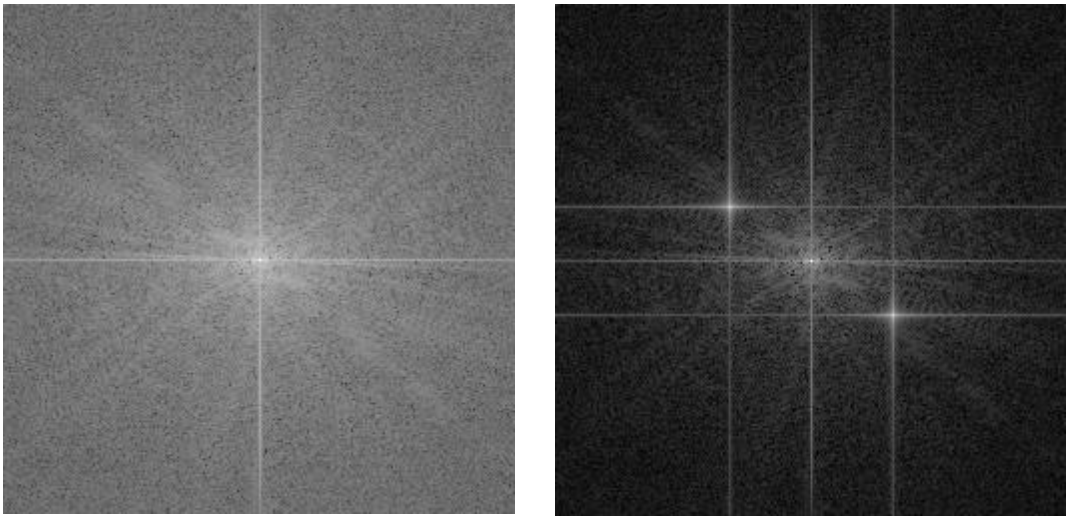
Derau periodik dapat terjadi peralatan yang digunakan dalam proses akuisisi citra atau perangkat keras jaringan (dalam proses transmisi citra) terpengaruh oleh gangguan elektronis yang sifatnya berulang (repetitif). Misalnya akibat pengaruh adanya motor listrik. Berikut contoh pengaruh derau periodik pada citra rekam medis dan citra satelit.


Untuk memperlihatkan cara mengurangi atau menghilangkan pengaruh derau periodik pada citra, akan digunakan simulasi pengaruh derau periodik pada citra Einstein.jpg sebagai berikut.

Citra asli dan citra dengan derau periodik

Jika dikenakan DFT pada kedua citra tersebut, maka hasilnya pada gambar berikut.



DFT citra Einstein.jpg asli dan berderau periodik

Terlihat pada gambar di atas, terdapat dua "*spike*" yang muncul selain pada komponen DC citra; ini merupakan derau yang sebelumnya disimulasikan (ditambahkan). Semakin tinggi frekuensi derau yang mengganggu citra maka letaknya akan semakin jauh dari titik pusat citra hasil DFT.

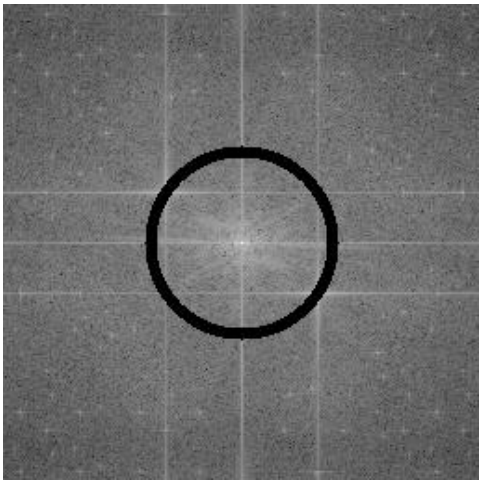
Band Reject Filtering

Dapat dgunakan *tools* dari Matlab yaitu **pixval, impixelinfo**, atau **imdistline** untuk mengetahui nilai piksel pada posisi (x, y) tertentu dan jarak antara dua titik pada citra. Dengan menggunakan *tools* impixelinfo, maka diketahui bahwa dua "*spike*" yang muncul pada citra Einstein.jpg berderau berada pada koordinat (102, 88) dan (156, 170). Kedua "*spike*" ini jaraknya dari pusat citra (dimana komponen DC berada) adalah 49,0918. Komponen DC citra tentu saja berada di pusat yaitu (128, 128).

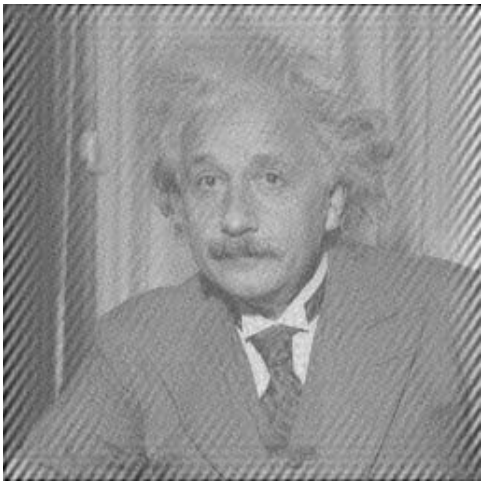
Oleh karena derau periodik bersesuaian dengan dua "*spike*" tersebut, maka artinya derau periodik pada citra dapat dihilangkan atau dikurangi dengan cara menghilangkan kedua "*spike*" yang bersangkutan. Salah satunya adalah dengan cara *band reject filtering*.

Band reject filter dapat dibangun dengan membuat cincin bernilai nol (berwarna hitam); untuk citra di atas maka cincin bernilai nol ini berada pada radius kira-kira 49 dari pusat citra.

```
clear all;
clc;
t = imread ('einstein_pn.jpg');
tf = fft2(t);
tf = fftshift(tf);
% ------------------------------------------
% Membuat band reject filter
% ------------------------------------------
s = size (t);
[x, y] = meshgrid (1:s(1), 1:s(2));
z = sqrt ((x-129).^2 + (y-129).^2);
br = (z < 46 | z > 52);
% ------------------------------------------
% Filtering
% ------------------------------------------
tbr = tf.*br;
% ------------------------------------------
% IDFT hasil filtering
% ------------------------------------------
tbr_inv = ifft2 (tbr);
% ------------------------------------------
% Menampilkan hasil
% ------------------------------------------
fftshow (tbr, 'log')
figure
fftshow (tbr_inv, 'abs')
```



Filter *band reject*



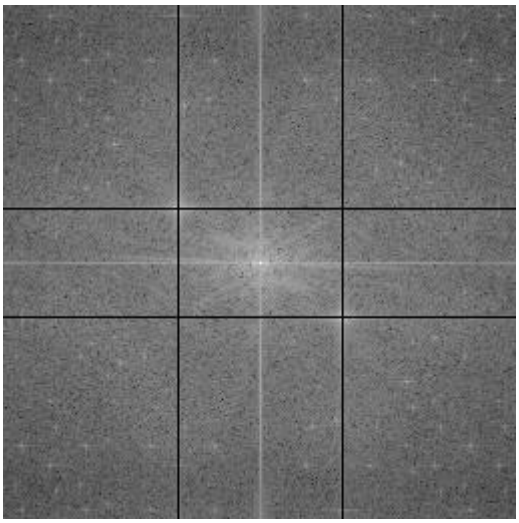
Citra output

Citra hasil filtering dengan *band reject filter* menunjukkan bahwa tidak semua derau dapat dihilangkan namun telah cukup mengurangi pengaruh derau tersebut terutama di bagian tengah citra.

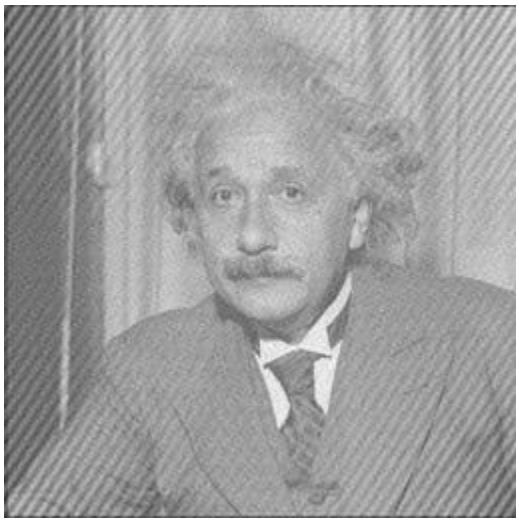
Notch Filtering

Notch filter membuat nilai piksel pada kolom dan baris pada "*spike*" menjadi nol.

```
clear all; clc;
t = imread ('einstein_pn.jpg');
tf = fft2(t);
tf = fftshift(tf);
% ------------------------------------------
% Notch filtering
% ------------------------------------------
tf(156,:) = 0; tf(102,:) = 0;
tf(:,170) = 0; tf(:,88) = 0;
% ------------------------------------------
% IDFT hasil filtering
% ------------------------------------------
tf_inv = ifft2 (tf);
% ------------------------------------------
% Menampilkan hasil
% ------------------------------------------
fftshow (tf, 'log')
figure, fftshow (tf_inv, 'abs')
```



Filter Notch



Citra output

SEGMENTASI CITRA

Segmentasi mengacu pada operasi pemisahan sebuah citra menjadi bagian-bagian atau komponen-komponennya, atau memisahkan objek-objek yang ada pada citra tersebut; sebagian besar kegiatan segmentasi citra melakukan pemisahan obyek (yang menjadi pusat perhatian) dari latar belakangnya. Akan dibahas dua cara segmentasi yang paling banyak dipakai, yaitu *thresholding* (pengambangan) dan *edge detection* (deteksi tebing).

Thresholding

Citra aras keabuan dapat diubah menjadi citra biner (hitam putih, *black and white*, BW) dengan terlebih dulu memilih suatu nilai aras keabuan T (dari citra asli) dan kemudian mengubah setiap piksel menjadi hitam atau putih bergantung apakah nilai piksel asli tersebut lebih besar atau lebih kecil dari nilai T. Piksel akan diubah menjadi putih jika nilai aras keabuannya lebih besar daripada T, dan akan diubah menjadi hitam jika nilai aras keabuannya lebih kecil atau sama dengan T. Atau dinyatakan,

$$y = \begin{cases} \text{putih jika } x > T \\ \text{hitam jika } x \le T \end{cases}$$

Dengan x adalah nilai aras keabuan dari citra input (asli), T adalah nilai ambang yang dipilih, dan y adalah keluaran. *Thresholding* merupakan bagian yang penting dalam segmentasi citra, misalnya saat dikehendaki untuk mengisolasi suatu obyek tertentu dari latar belakangnya. Dewasa ini juga digunakan sebagai bagian dari penglihatan robot (*robot vision*).

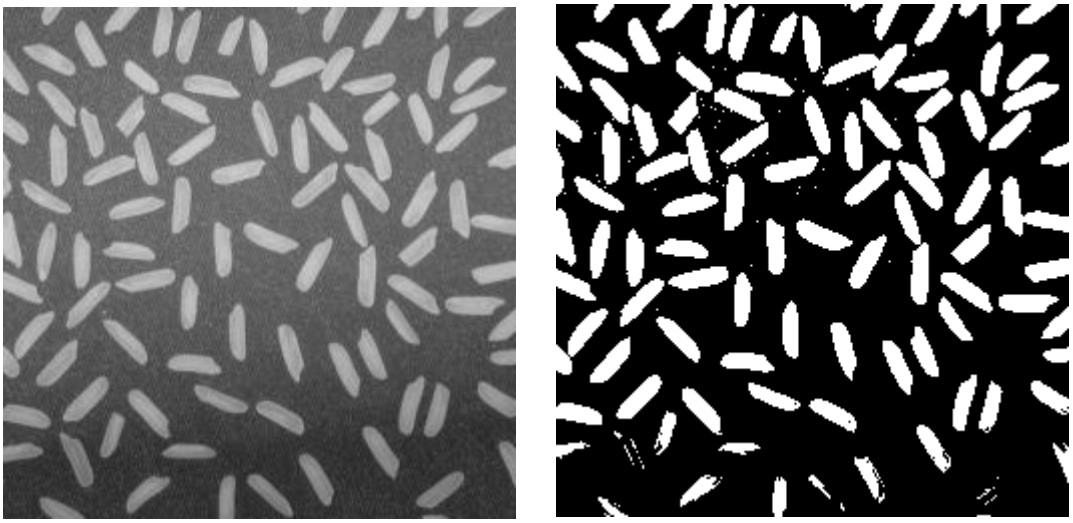
Thresholding dapat dilakukan dalam bahasa Matlab maupun Scilab dengan sederhana. Jika terdapat citra 8 bit yang tersimpan dalam variabel x, maka *thresholding* dapat dilakukan dengan perintah

x > T

dengan T adalah nilai ambang yang dikehendaki, dan hasilnya dapat ditampilkan dengan perintah **imshow**.

Contoh

```
>> r = imread ('rice.png');
>> imshow (r)
>> figure, imshow (r > 130)
```



Citra rice.png dan hasil *threshoding* dengan T = 130

Perintah x > 130 akan menghasilkan keluaran 1 (benar) jika masukan lebih besar daripada 130 dan akan menghasilkan keluaran 0 (salah) jika masukan lebih kecil atau

sama dengan 130. Ini akan menghsilkan citra yang disebut citra biner yang juga dapat ditampilkan menggunakan **imshow**.

Double Thresholding

Jika dipilih dua nilai T_1 dan T_2 dan digunakan untuk operasi pengambangan atau *thresholding*, maka operasi *double thresholding* diimplementasikan dengan cara

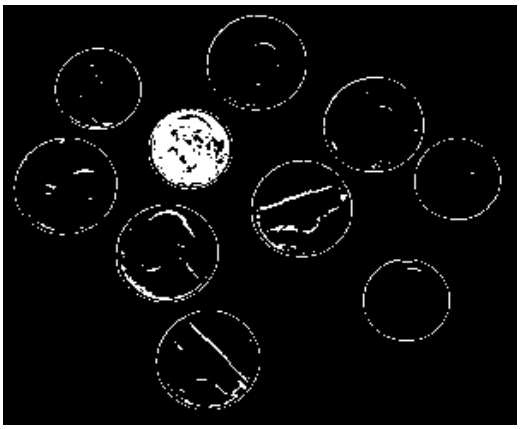
$$y = \begin{cases} \text{putih jika nilai x berada diantara } T_1 \text{ dan } T_2 \\ \text{hitam untuk nilai-nilai x yang lain} \end{cases}$$

Dengan x adalah nilai aras keabuan dari citra input (asli), T_1 dan T_2 adalah nilai ambang yang dipilih, dan y adalah keluaran. Implementasi dalam bahasa Matlab dapat dilakukan dengan sintaks

$$x > T_1 \ \& \ x < T_2$$

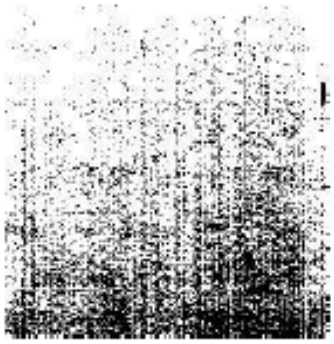
Contoh

```
>> r = imread ('coins.png');
>> imshow (r)
>> figure,imshow ( r > 100  &  r < 150 )
```

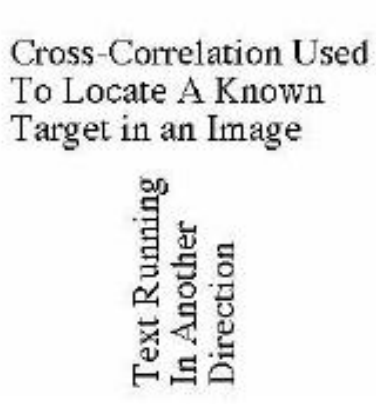


Aplikasi *thresholding* misalnya adalah:

1. Menghilangkan detail yang tidak dikehendaki pada citra sehingga dapat berfokus pada bagian yang dikehendaki saja. Hal ini tampak pada contoh *thresholding* citra rice.png. Informasi yang didapat dari citra hasil *thresholding* dapat digunakan untuk mengetahui ukuran, bentuk, atau jumlah obyek.
2. Memunculkan detail yang sebelumnya tersembunyi. Contohnya adalah gambar berikut. Jika dinyatakan dalam citra aras keabuan 8 bit maka mata manusia tidak mampu membedakan perbedaan yang kecil pada nilai aras keabuan citra, namun setelah dilakukan *thresholding* maka tampak detail yang sebelumnya tersembunyi.



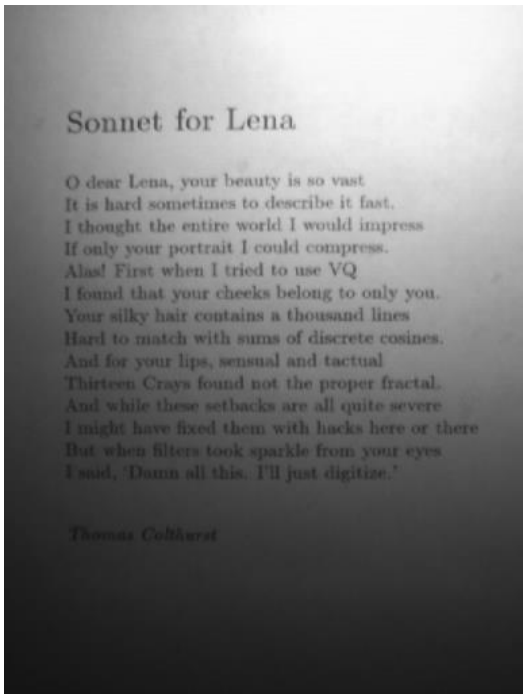
3. Menghilangkan latar belakang yang bervariasi pada teks atau gambar. Misalnya pada gambar berikut.

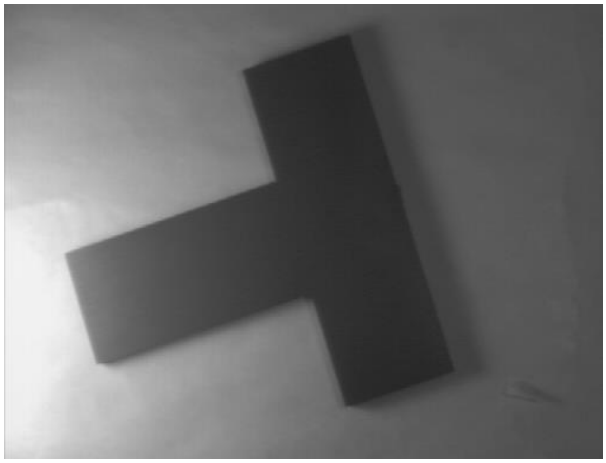


Adaptive Thresholding

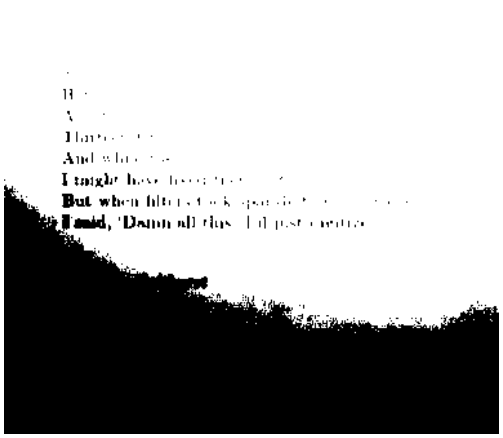
Pada beberapa kasus tidak dimungkinkan untuk memilih satu atau dua nilai ambang yang dapat mengisolasi objek secara keseluruhan. Hal ini misalnya terjadi jika

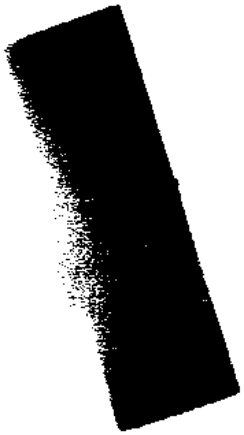
baik objek maupun latar belakang cita sangat bervariasi. Berikut satu contoh *adaptive thresholding*.





Thresholding non adaptif dengan T = 50





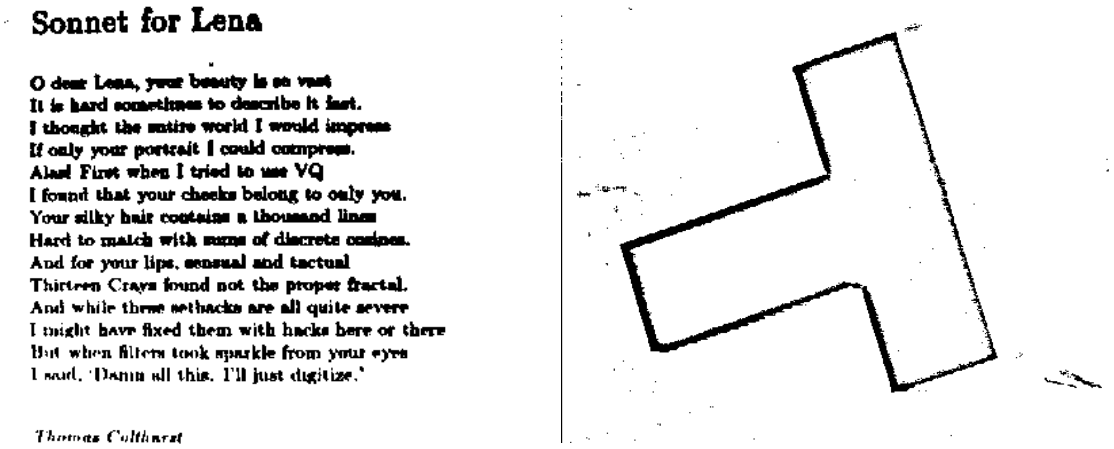
Terlihat bahwa tidak semua objek yang diinginkan dapat tersegmentasi, hal ini akibat iluminasi yang tidak seragam. Penyelesaian yang dapat digunakan adalah dengan menggunakan nilai ambang yang berbeda-beda bergantung pada lokasi citra atau sering disebut pengolahan lokal (*localized*). Misalnya dengan menggunakan parameter nilai rerata (*mean*) atau *median* dari suatu jendela (*window*) pada citra yang bersangkutan.

```
function bw = adaptivethreshold(IM,ws,C,tm)
% Adaptivethreshold adl algorithma yg memisahkan latar blkg
% dan objek dengan iluminasi yang tidak seragam
% Sintaks:
%
%   bw = adaptivethreshold(IM,ws,C,tm)
%
% Outputs bw merupakan citra biner
% C digunakan utk threshold local dg rumus mean-C atau
% median-C
% ws adalah ukuran window yang digunakan
% tm bernilai 0 or 1, (mean atau median)
% tm = 0 mean(default); tm = 1 median
% Contributed by GuangleiXiong

if (nargin<3)
    error('Masukkan citra IM, ukuran window ws, dan C');
elseif (nargin==3)
    tm=0;
elseif (tm~=0 && tm~=1)
    error('tm harus 0 atau 1');
end

IM=mat2gray(IM);

if tm==0
    mIM=imfilter(IM,fspecial('average',ws),'replicate');
else
    mIM=medfilt2(IM,[ws ws]);
end
sIM=mIM-IM-C;
bw=im2bw(sIM,0);
bw=imcomplement(bw);
```

Penggunaan fungsi untuk threshold pada citra page.png dan tshape.png.

```
clear;
close all;
im1=imread('page.png');
im2=imread('tshape.png');
bwim1=adaptivethreshold(im1,20,0.04,1);
%rata = fspecial('average');
%bwim1 = filter2 (rata, bwim1);
bwim2=adaptivethreshold(im2,15,0.02,0);
%rata = fspecial('average');
%bwim2 = filter2 (rata, bwim2);
imshow(im1);
figure, imshow(bwim1);
figure, imshow(im2);
figure, imshow(bwim2);
```

Hasil yang diperoleh adalah sbb.

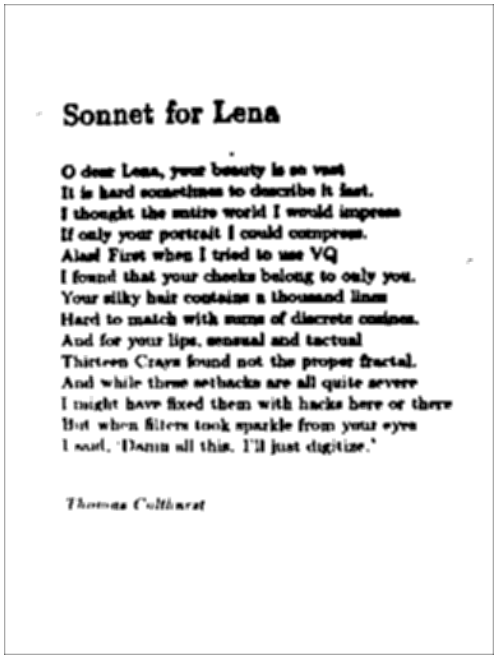


Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

Thomas Colthurst

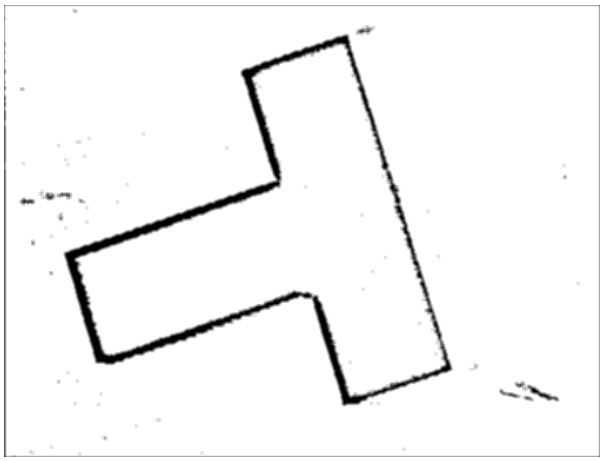
Jika hasilnya dikenakan filter rerata, maka hasilnya sebagai berikut.



Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactual
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this, I'll just digitize.'

Thomas Colthurst



Edge Detection (Deteksi Tebing)

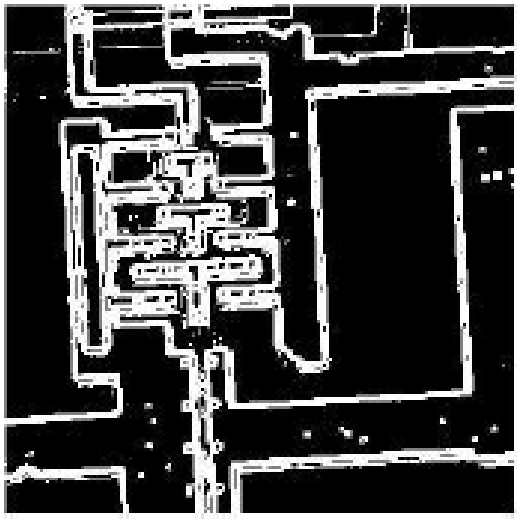
Tebing (*edge*) citra dapat digunakan untuk memperoleh informasi dari citra seperti halnya pada *threshloding*. Tebing citra juga dapat digunakan untuk mengetahui ukuran objek, mengisolasi objek tertentu pada citra, atau untuk mengenali dan mengklasifikasikan objek (misalnya berdasarkan bentuknya).

Tebing dapat didefinisikan sebagai suatu diskontinuitas lokal dalam nilai pikselnya, yaitu bahwa nilai pikselnya dapat dengan jelas ‘terlihat’ berubah. Misalnya pada potongan citra berikut.

51	52	53	59
54	52	53	62
50	52	53	68
55	52	53	55

50	53	155	160
51	53	160	170
52	53	167	190
51	53	162	155

Pada potongan citra yang pertama, mata manusia kemungkinan tidak dapat megikuti dengan jelas perubahan nilai piksel pada citra karena perubahannya cukup kecil. Sedangkan pada potongan yang kedua, antara kolom kedua dan ketiga terjadi perubahan piksel yang cukup besar (kira-kira 100 satuan) sehingga mata manusia kemungkinan besar dapat melihat hal ini sebagai suatu tebing (misalnya tebing yang membatasi suatu warna gelap ke warna yang terang). Perhatikan gambar berikut.



Tebing pada citra circuit

Filter Pendeteksi Tebing (Edge Detection Filters)

Filter Prewitt

Filter Prewitt menggunakan dua operator untuk mendeteksi tebing vertikal dan tebing horisontal.

Operator pendeteksi tebing vertikal

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

Operator pendeteksi tebing horisontal

$$P_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

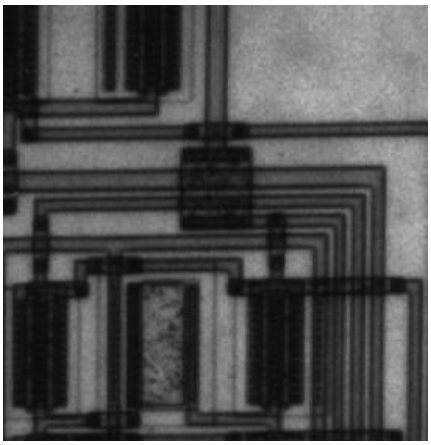
Contoh: Mendeteksi tebing vertikal

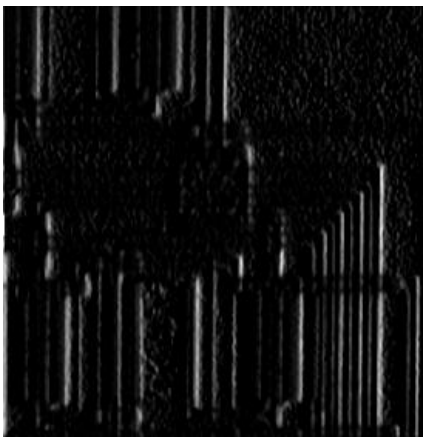
```
clear all; clc;
c = imread ('circuit.tif');
px = [-1 0 1;-1 0 1;-1 0 1];
c_px = filter2 (px, c);
imshow (c)
figure, imshow (c_px/255)
```

Contoh: Mendeteksi tebing horisontal

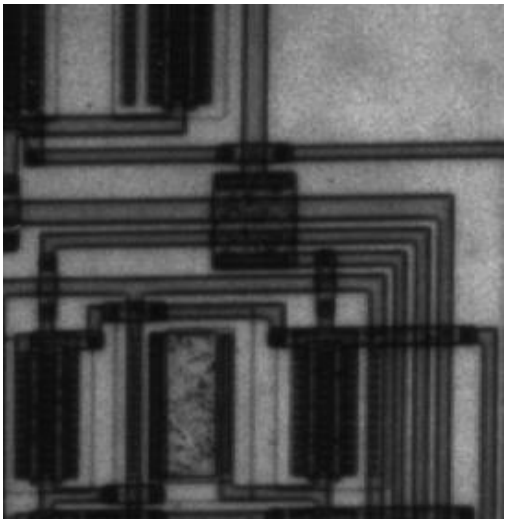
```
clear all; clc;
c = imread ('circuit.tif');
py = [-1 -1 -1;0 0 0;1 1 1];
c_py = filter2 (py, c);
imshow (c)
figure, imshow (c_py/255)
```

Hasil deteksi tebing vertikal dengan operator filter Prewitt





Hasil deteksi tebing horisontal dengan operator filter Prewitt



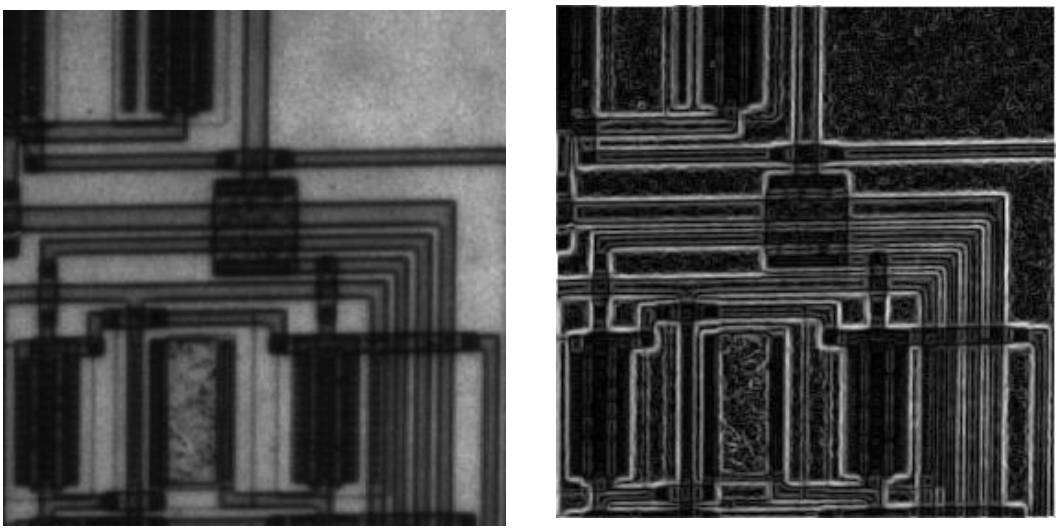
Jika dengan menerapkan P_x dan P_y akan diperoleh nilai p_x dan p_y maka *magnitude gradien*-nya diperoleh dengan cara

$$\sqrt{p_x^2 + p_y^2}$$

Besaran ini dapat digunakan sebagai pendeteksi tebing vertikal sekaligus horisontal.

Contoh

```
clear all; clc;
c = imread ('circuit.tif');
px = [-1 0 1;-1 0 1;-1 0 1];
py = [-1 -1 -1;0 0 0;1 1 1];
c_px = filter2 (px, c);
c_py = filter2 (py, c);
tebing = sqrt(c_px.^2 + c_py.^2);
imshow (c)
figure, imshow (tebing/255)
```

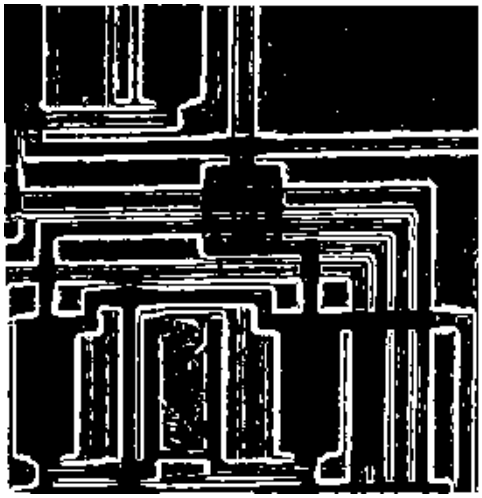


Citra circuit.tif asli dan hasil deteksi tebing vertikal dan horisontal Prewitt

Hasil deteksi tebing yang terakhir di atas masih berupa citra aras keabuan, untuk mengubahnya menjadi citra biner dapat digunakan operasi *thresholding*, atau dapat juga menggunakan fungsi yang disediakan Matlab yaitu im2bw.m.

```
>> tebing = im2bw (tebing/255, 0.3)
```

Dan hasilnya adalah pada gambar berikut.



Robert Cross-Gradient Filters

Filter Robert menggunakan dua operator sebagai berikut.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Filter Sobel

Filter Sobel menggunakan dua operator untuk mendeteksi tebing vertikal dan tebing horisontal.

Operator pendeteksi tebing vertikal

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Operator pendeteksi tebing horisontal

$$\begin{bmatrix} -1 & -2 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Matlab menyediakan fungsi untuk deteksi tebing dengan pilihan beberapa jenis filter, termasuk diantaranya filter Prewitt, Robert, dan Sobel.

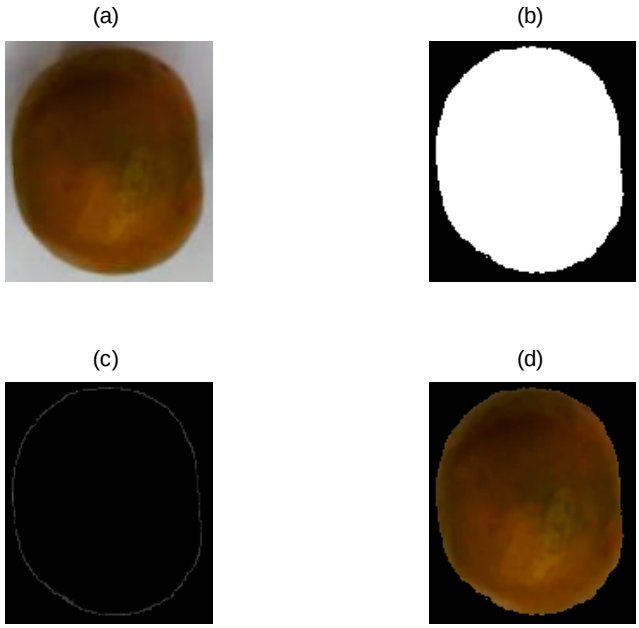
Sintaks:

edge (image, ‘prewitt’)

edge (image, ‘robert’)

edge (image, ‘sobel’)

Contoh hasil penggunaan (aplikasi) *thresholding*, deteksi tepi, dan segmentasi pada citra jeruk.



a. Citra asli
b. Hasil *thresholding*
c. Hasil deteksi tepi
d. Hasil segmentasi

Homework

Lakukan *thresholding*, deteksi tepi, dan segmentasi pada citra yang anda pilih sendiri menggunakan algoritma yang telah dipelajari di atas. Pada pekerjaan anda, tentukanlah obyek apa yang akan anda pisahkan (segmentasi) dari obyek-obyek lain (latar belakang) pada gambar/citra tersebut, kemudian ubahlah latar belakang tsb menjadi hitam atau puith). Kumpulkan via email susilawati.indah@gmail.com paling lambat Selasa, 16 – 12 – 2014 pukul 16.00 WIB.

NB :

1. Nama email pengirim harus sesuai nama yang ada pada presensi
2. Subject harus ditulis : Homework TPC TM10
3. Tidak sesuai ketentuan 1 dan 2 maka pekerjaan anda tidak akan dikoreksi
4. Jika pekerjaan anda sama dengan mahasiswa lain, maka yang pertama dikirim dianggap sebagai yang original.