Dasar Teori Operasi Piksel

# Operasi Brightness

Operasi Brightness pada digital image processing merupakan operasi yang digunakan untuk meningkatkan / mengurangi intensitas kecerahan pada suatu citra tanpa merubah bentuk dasar citra tersebut. Kecerahan citra dapat diperbaiki dengan menambahkan (atau mengurangkan) sebuah konstanta kepada (atau dari) setiap pixel di dalam citra.

Pada operasi brightness menggunakan alogaritma sebagai berikut :

**Citra Hasil = Citra Asli + Nilai Intensitas Kecerahan**

Secara matematis operasi ini ditulis sebagai :

**f(x, y)’ = f(x, y) + b**

Jika b positif, kecerahan citra bertambah, sebaliknya jika b negatif kecerahan citra berkurang.

>> Dalam aplikasi yang kami buat, kami menggunakan nilai intensitas sebesar -255 hingga 255 artinya :

* Jika Citra Asli di tambah dengan intensitas sebesar 0 maka tidak akan terjadi perubahan pada citra,
* Jika Citra Asli ditambah dengan intensitas diatas 0 (positif) atau mendekati nilai 255 maka intensitas kecerahan pada Citra akan bertambah dan seolah olah mendekati warna putih.
* Sebaliknya jika Citra Asli ditambah dengan intensitas dibawah nilai 0 (negative) atau mendekati nilai - 255 maka intensitas kecerahan pada Citra akan berkurang dan seolah olah mendekati warna hitam.
* Jika Citra Asli ditambah dengan intensitas sebesar 255 maka Citra akan berubah menjadi Putih keseluruhan, sebaliknya jika Citra Asli ditambah dengan intensitas sebesar -255 maka Citra akan berubah menjadi Hitam keseluruhan.

# Operasi Kontras

Kontras dalam suatu citra menyatakan distribusi warna terang dan warna gelap. Suatu citra berskala keabuan dikatakan memiliki kontras rendah apabila distribusi warna cenderung pada jangkauan aras keabuan yang sempit. Sebaliknya, citra mempunyai kontras tinggi apabila jangkauan aras keabuan lebih terdistribusi secara melebar. Kontras dapat diukur berdasarkan perbedaan antara nilai intensitas tertinggi dan nilai intensitas terendah yang menyusun piksel-piksel dalam citra.

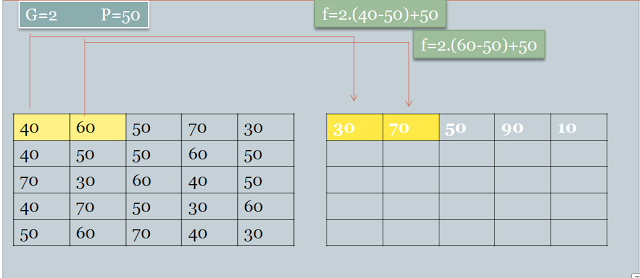
Ada beberapa cara untuk melakukan perhitungan kontras, namun yang saya tampilkan sekarang adalah dengan rumus berikut:

[](http://1.bp.blogspot.com/-WTweZ40n5MA/Up679WG1-3I/AAAAAAAAAIo/ugv44lgy2KY/s1600/rumus.png)

Dimana:   
G = koefisien penguatan kontras

P = nilai grayscale yang dipakai sebagai pusat pengkontrasan

 contoh perhitungan:

[](http://3.bp.blogspot.com/-svw2NWct-Ks/Up68fE2l2lI/AAAAAAAAAIw/Z9tdcohH-vA/s1600/hasil.png)

# Operasi Invers

Invers dalam ilmu matematika adalah fungsi yang merupakan kebalikan aksi dari suatu fungsi. (<https://id.wikipedia.org/wiki/Fungsi_invers>). Dalam pengolahan citra digital, invers merupakan penerapan fungsi Boolean not (operasi not) pada sautu citra (*Pengolahan Citra Digital Bab 4 :* *Operasi-operasi Dasar Pengolahan Citra Dijital* dari Bapak Hari Wibawanto). Oleh karena itu, invers hanya bisa digunakan pada citra biner saja.

Pada mulanya citra yang kita dapat, harus diubah dulu ke citra biner, kemudian akan kita *not*-kan citra tersebut.

Dalam mengubah dari citra apapun yang kita dapat menuju citra biner, kita perlu titik ambang (treshold point) yang menjadi acuan, dalam hal ini titik acuan kita ambil dari nilai rata-rata intensitas tiap – tiap pixel sehingga titik ambang berada di tengah intensitas seluruh pixel dari citra tersebut. Setelah citra biner didapat, kita invers menggunakan simbol ~.

uk = size(citra\_asli); %mencari ukuran pixel

pxl = uk(1)\*uk(2); %menghitung luas citra

Kam = rgb2gray(citra\_asli); %mengubah jadi grayscale

suG = sum(sum(Kam)); %menjumlah nilai intensitas

Rat = suG/pxl; %merata intensitas

Prc = Rat/255; %mengubah rata-rata ke 0:1

invk = ~(im2bw(Kam,prc)); %mengubah ke biner & invers

|  |  |
| --- | --- |
| Citra asli | Sesudah diinvers |
| C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\unnes.jpg | C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\unnes inv.jpg |

# Operasi Thresholding

Thresholding adalah metode dengan seluruh pixel pada citra dikonversi menjadi hitam dan putih dengan satu nilai ambang (T) tertentu. Proses thresholding dilakukan pada citra grayscale f(x,y) yang bertujuan untuk menghasilkan citra biner g(x,y), secara matematis dapat ditulis sebagai berikut



T menyatakan nilai ambang. Nilai T ditentukan dengan menggunakan metode thresholding lokal dan thresholding global. Nilai T pada thresholding lokal dapat diatur sesuai kebutuhan pengguna. Sementara pada thresholding global, niali T dicari dengan menggunakan fungsi otomatis metode otsu.

Metode otsu melakukan analisis diskriminan dengan menentukan suatu variable dengan membedakan antara dua atau lebih kelompok secara alami (Purba, D. 2010). Metode otsu dimulai dengan normalisasi histogram citra sebagai fungsi probability discrete density sebagai :



Dimana :

n = total jumlah piksel dalam citra.

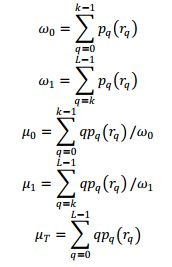
𝑛𝑞 = jumlah pixel 𝑟𝑞

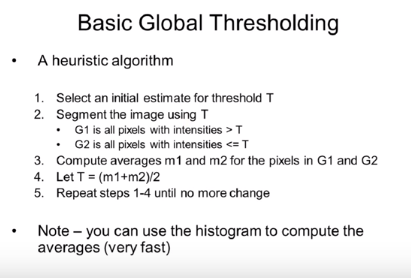
L = total jumlah level intensitas citra.

Kemudian menentukan nilai T dengan memaksimalkan between class variance yang didefenisikan sebagai berikut :



dimana :





# Operasi Histogram

Histogram adalah representasi grafis untuk distribusi warna dari citra digital. Ekualisasi histogram merupakan metode pengolahan citra tentang penyesuaian kontras menggunakan histogram dari citra tersebut. (istogram equalization ­ Wikipedia <https://en.wikipedia.org/wiki/Histogram_equalization>).

Ekualisasi histogram pada dasarnya hanya bisa dilakukan pada citra skala keabuan. Teknik perataan histogram merupakan gabungan antara penggeseran dan pelebaran histogram. Tujuan yang akan dicapai pada teknik ini adalah untuk mendapatkan citra dengan daerah tingkat keabuan yang penuh dan dengan distribusi pixel pada setiap tingkat keabuan yang merata. Dengan kata lain, Tujuan dari perataan histogram ini adalah untuk memperoleh penyebaran histogram yang merata, sedemikian sehingga setiap derajat keabuan memiliki jumlah pixel yang relative sama. Dengan demikian, citra yang dihasilkan tidak terlalu gelap ataupun terlalu terang.

Pada MatLab, sudah disediakan fungsi ekualisasi histogram yaitu histeq.

out = histeq(citra\_asli);

fungsi diatas hanya untuk citra skala keabuan. Untuk citra berwarna, kita ubah dulu ke format HSV kemudian kita ambil nilai Value-nya saja. Kemudian nilai Value tadi diratakan histogramnya.

%mengubah RGB ke HSV

HSV = rgb2hsv(citra\_asli);

%mengambil nilai Value & diekualisasi histogramnya

Heq = histeq(HSV(:,:,3));

%membuat variabel baru untuk output

HSV\_mod = HSV;

%memasukan Value yang sudah diekualisasi

HSV\_mod(:,:,3) = Heq;

%mengubah HSV ke RGB

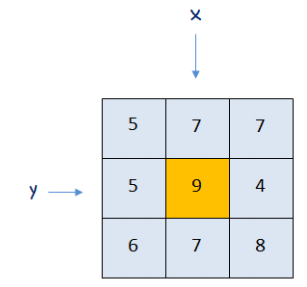
RGB = hsv2rgb(HSV\_mod);

|  |  |
| --- | --- |
| Citra asli | Setelah ekualisasi histogram |
| C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\bunga.png | C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\bunga ekual.png |
| C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\gedung2.jpg | C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\gedung2 ekual.jpg |
| C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\lena.png | C:\Users\Cahyo\Documents\GitHub\matlab-operasi_piksel\gambar-contoh\lena ekual.png |

# Operasi Filter Batas

Filter batas dikemukan oleh Davies (1990). Fungsi dari filter batas untuk mencegah piksel yang intensitasnya di luar intensitas piksel-piksel tetangga. Algoritma untuk menghitung piksel dengan filter batas.

Contoh piksel dengan tetangga



Berdasarkan keadaan tersebut,

minInt = minimum(5, 7, 7, 5, 4, 6, 7, 8) = 4;

maksInt = maksimum(5, 7, 7, 5, 4, 6, 7, 8) = 8;

mengingat f(y, x) bernilai 9 dan lebih besar daripada 8 (maksInt) maka g(y, x) bernilai 8;

seandainya f(y, x) pada keadaan di atas bernilai 2 (bukan 9), g(y,x) akan bernilai 4.

Contoh program filter batas.m

* FILBATAS Melakukan operasi ketetanggan piksel
* menggunakan filter batas

F = imread('c:\Image\mobil.png'); Ukuran = size(F);

tinggi = Ukuran(1); lebar = Ukuran(2);

G = F;

|  |  |  |  |
| --- | --- | --- | --- |
| for baris=2 : tinggi-1 | |  |  |
| for kolom=2 : lebar-1 | |  |  |
| minPiksel = min([F(baris-1, kolom-1) | | | ... |
| F(baris-1, kolom) | | F(baris, kolom+1) | ... |
| F(baris, | kolom-1) |  | ... |
| F(baris, | kolom+1) | F(baris+1, kolom-1) ... | |
| F(baris+1, kolom) | | F(baris+1, kolom+1)]); | |
| maksPiksel = | min([F(baris-1, kolom-1) | | ... |
| F(baris-1, kolom) | | F(baris, kolom+1) | ... |
| F(baris, | kolom-1) |  | ... |
| F(baris, | kolom+1) | F(baris+1, kolom-1) ... | |
| F(baris+1, kolom) | | F(baris+1, kolom+1)]); | |

if F(baris, kolom) < minPiksel G(baris, kolom) = minPiksel;

else

if F(baris, kolom) > maksPiksel G(baris, kolom) = maksPiksel;

else

G(baris, kolom) = F(baris, kolom);

end

end

end

end

figure(1);

imshow(G);

clear;

filter batas hanya bisa dilakukan selain baris pertama, baris terakhir, kolom pertama dan kolom terakhir. Keempat area tersebut tidak dapat diproses karena tidak mempunyai tetangga yang lengkap.

# Operasi Filter Rerata

Mean filter merupakan filter yang bekerja dengan cara membuat window/kernel, misalnya 3x3. Titik tengah window/kernel tersebut kemudian diletakan diatas piksel dari gambar yang akan diproses. Selanjutnya untuk semua piksel yang berada dibawah window/kernel tersebut dihitung nilai reratanya. Nilai rerata ini digunakan untuk mengganti nilai piksel yang tepat berada pada titik tengah window/kernel. Prosedur ini d *loop* sampai semua piksel terproses.

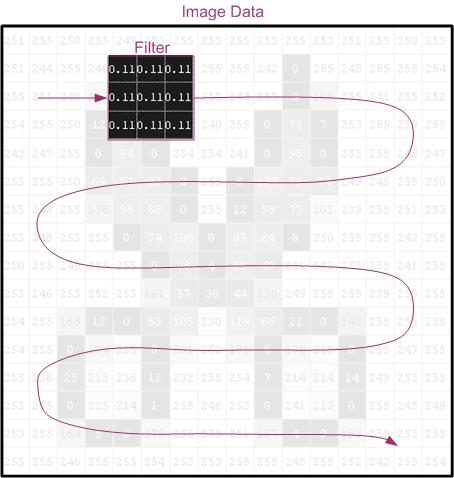
|  |  |  |
| --- | --- | --- |
| Sebelum difilter | | |
| 5 | 3 | 6 |
| 2 | **1** | 9 |
| 8 | 4 | 7 |

|  |  |  |
| --- | --- | --- |
| Setelah difilter | | |
| \* | \* | \* |
| \* | **5** | \* |
| \* | \* | \* |

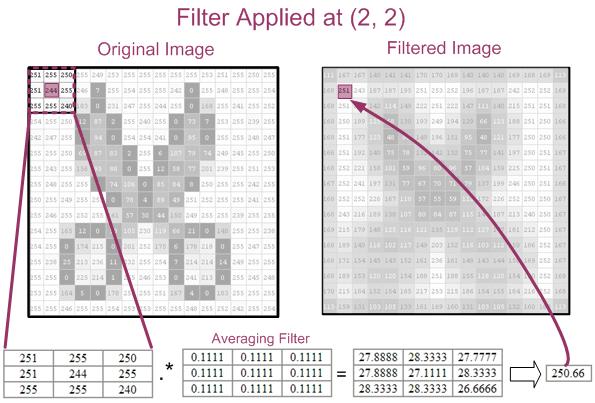
Jumlah = 5 + 3 + 6 + 2 + 1 + 9 + 8 + 4 + 7 = 45  
**Rerata = 45 / 9 = 5**

Nilai piksel ditengah awalnya adalah 1, kemudian diganti dengan nilai rerata yaitu 5.

Ilustrasi:

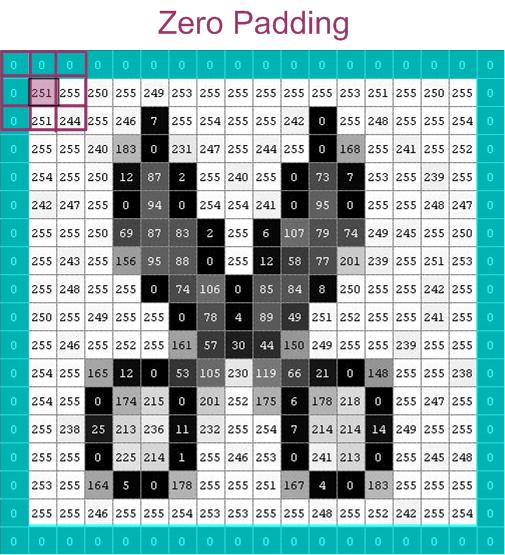


Gambar 1: Kernel dijalankan pada keseluruhan piksel pada gambar

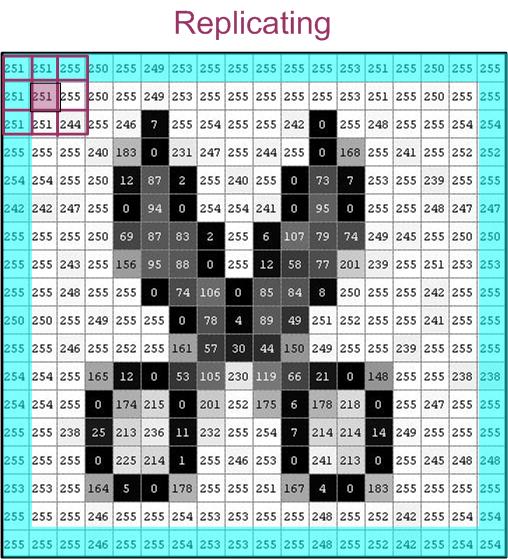


Gambar 2: Filter diterapkan pada f(2,2)

Kasus khusus untuk piksel tanpa tetangga misalnya f(1,1), maka solusinya adalah *zero padding* dan replikasi.



Gambar 3: Zero padding



Gambar 4: Replicating

# Operasi Filter Median

Filter ini bekerja dengan menggantikan nilai tengah dari piksel yang dicakup oleh area filter (3x3) dengan sebuah nilai tengah (median) setalah diurutkan terlebih dahulu dari yang terkecil hingga yang terbesar. Biasanya ukuran filter adalah ganjil karena akan memberikan nilai tengah, sehingga akan lebih mudah dalam mengolah citra.

Proses Pengolahan

1. Misal akan mengolah citra dengan matrik 6x6

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

1. Matrik filter adalah 3x3

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

1. Area cakupan matrik filter pada matrik citra yang akan diolah

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 |  |  |  |
|  | x |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | x |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

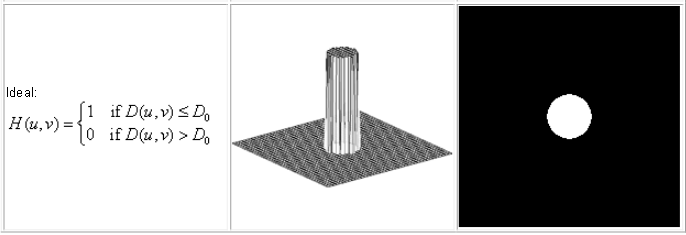
1. Data piksel kemudian diurutkan dari yang terkecil, kemudian dicari nilai tengahnya untuk ditempatka di posisi (x).

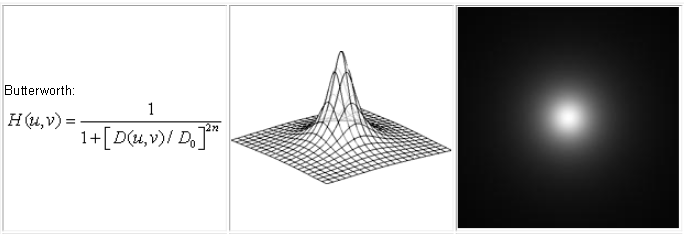
# Operasi Filter Low Pass

Low pass filter membuat gambar blur, atau diperhalus dengan cara menahan frekuensi tinggi dan meloloskan frekuensi rendah.

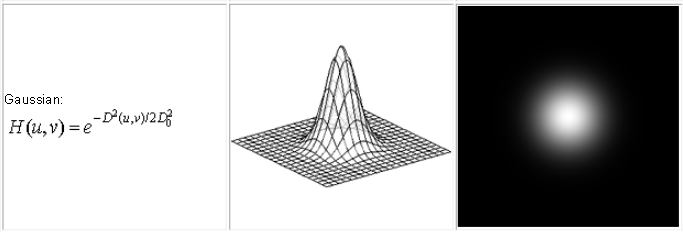
Low pass filter atau sering dikenal dengan istilah *smoothing* merupakan upaya untuk menghilangkan *noise* berfrekuensi tinggi dari citra digital. Cara kerjanya yaitu menggunakan window/kernel berukuran tertentu, window/kernel tersebut kemudian diterapkan pada semua piksel satu demi satu. Window/kernel ini akan mengubah nilai dari piksel sesuai dengan fungsi yang ada dalam window/kernel tersebut.

Dalam matlab, ada tiga low pass filter yang digunakan untuk pengolahan citra digital yaitu:

1. Idela lowpass filter (ilpf) 
2. Butterworth low pass filter (blbf)



1. Gaussian low pass filter (glpf)



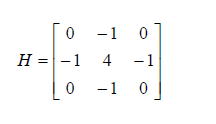
# Operasi Filter High Pass

High-pass filteradalah filter yang ditujukan untuk melewatkan frekuensi tinggi dan menghalangi yang berfrekuensi rendah. Hal ini biasa dipakai untuk mendapatkan tepi objek dalam citra atau menajamkan citra. High-pass filter biasa digunakan untuk menpertajam (sharping) suatu citra atau gambar. High-pass filter akan memperkuat komponen yang memiliki frekuensi tinggi dan akan menurunkan atau menghilangkan komponen yang memiliki frekuensi rendah.

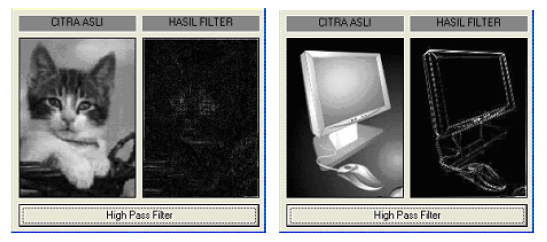
Ciri-ciri dari fungsi High-pass filter adalah sebagai berikut:



Sebagai contoh dibuat program High Pass Filter dengan fungsi filter rata-rata sebagai berikut:



Hasil dari program High Pass Filter, ini untuk beberapa macam gambar adalah sebagai berikut:



Gambar Hasil HPF untuk gambar kucing dan computer

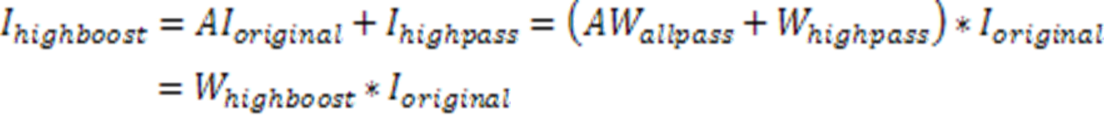
Dari kedua hasil di atas dapat dilihat bahwa High Pass Filter menyebabkan gambar hanya diambil atau ditampilkan pada daerah-daerah yang berbeda misalkan pada tepi-tepi gambar. Pada gambar kucing perbedaan yang muncul tidak begitu jelas karena gambarnya mempunyai gradiasi yang tinggi (halus), sedangkan pada gambar komputer tepi-tepi gambar tampak jelas karena perbedaannya tinggi.

# Operasi Filter High Boost

Operasi High Boost Filter dalam digital image processing merupakan sebuah operasi dimana sebuah citra di filter untuk di kuatkan frekuensi tinggi nya, namun tanpa menghilangkan frekuensi rendah nya. Mudah nya adalah mempertajam komponen citra yang mewakili detil gambar. Dalam High Boost Filter terdiri dari semua komponen pass filter dan deteksi tepi (edge detection), jadi filter ini menekankan batas tepi dan penajaman gambar.

High Boost Filter adalah operator penajaman sederhana dalam pengolahan sinyal dan citra. Dalam pengolahan citra, kita dapat mempertajam tepi gambar melalui amplifikasi dan mendapatkan gambar yang lebih jelas. Unsharp masking Filter (High-boost filter) menghilangkan bagian-bagian kabur dan meningkatkan tepi

High Boost Filter dinyatakan dalam bentuk persamaan sebagai berikut :

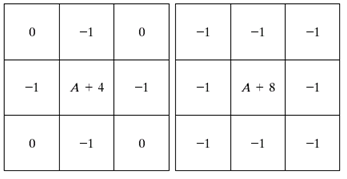


Dengan arti :

 adalah high boost convolution kernel,.

dan A adalah konstanta.

Teknik High-boost filter dapat diimplementasikan dengan menggunakan mask yang diberikan di bawah untuk A ≥ 1





Jadi untuk mendapatkan hasil citra highboost filter yaitu dengan cara meng-konvolusikan citra asli dengan Mask highboost ( W*highboost* ) yang telah diberi konstanta tertentu.

# Operasi Level Clipping

Level Clipping (pemotongan) dilakukan jika nilai intensitas piksel hasil suatu operasi pengolahan citra terletak dibawah nilai intensitas minimum atau diatas nilai intensitas maksimum.

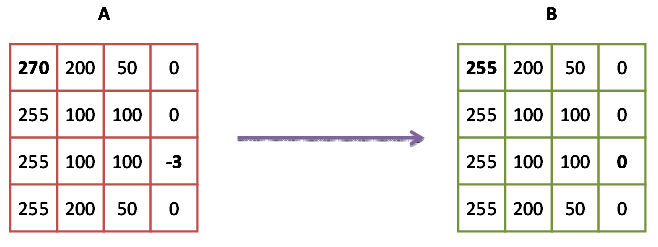
Aljabar level clipping

255 f (x,y) > 255

F (x,y)’ = f (x,y) 0 ≤ f (x,y) ≤ 255

1. f (x,y) < 0

Contoh Clipping untuk 8 bit



/\* clipping \*/

If (temp<0);

B [I] [J] = 0;

else

If (temp>255);

B [I] [J] = 255;

else

B [I] [J] = tetap;