Bab 4

Konvolusi dan Transformasi Fourier

Bab ini berisi konsep matematis yang melandasi teori pengolahan citra. Dua operasi matematis penting yang perlu dipahami dalam mernpelajari pengo1ahan citra digital adalah operasi konvolusi dan Transformasi Fourier. Konvolusi terdapat pada operasi pengolahan citra yang mengalikan sebuah citra dengan sebuah *mask* atau *kernel* (akan dijelaskan kemudian), sedangkan Transforrnasi Fourier dilakukan bila citra dimanipulasi dalam ranah (domain) frekuensi ketimbang dalam ranah spasial.

Teori Konvolusi

Operasi yang mendasar dalam pengolahan citra adalah operasi konvolusi. Konvolusi 2 buah fungsi *f(x)* dan *g(x)* didefinisikan sebagai berikut:

$$h(x) = f(x) * g(x) = \int_{-\infty}^{\infty} f(a)g(x - a)da$$

yang dalam hal ini, tanda * menyatakan operator konvolusi, dan peubah *(variable)* a adalah peubah bantu *(dummy variable)*.

Untuk fungsi diskrit, konvolusi didefinisikan sebagai

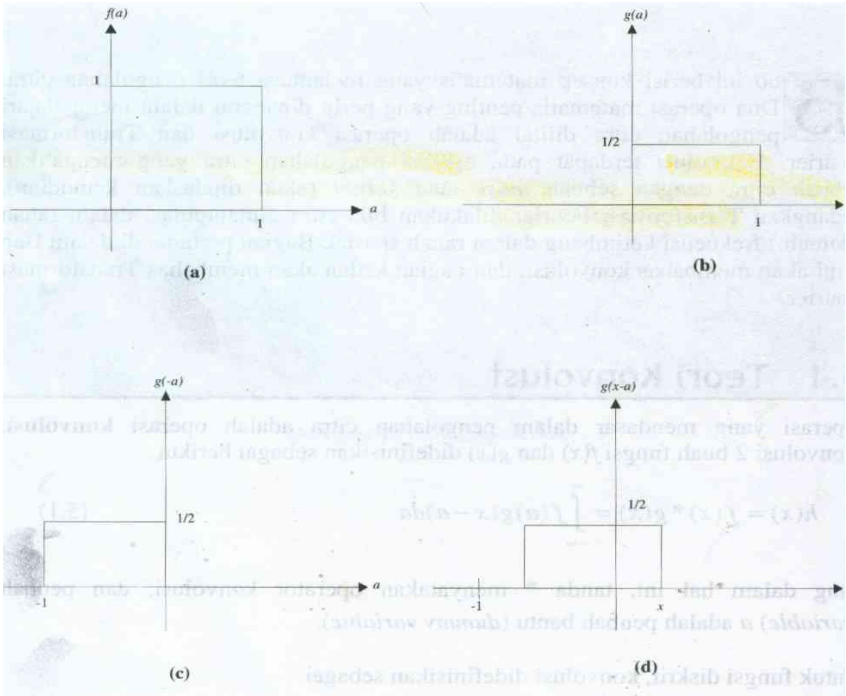
$$h(x) = f(x) * g(x) = \sum_{a=-\infty}^{\infty} f(a)g(x - a)$$

Pada operasi konvolusi di atas, *g(x)* disebut kernel konvolusi atau kernel penapis *(filter)*. Kernel *g(x)* merupakan suatu jendela yang dioperasikan secara bergeser pada sinyal masukan *f(x)*, yang dalam hal

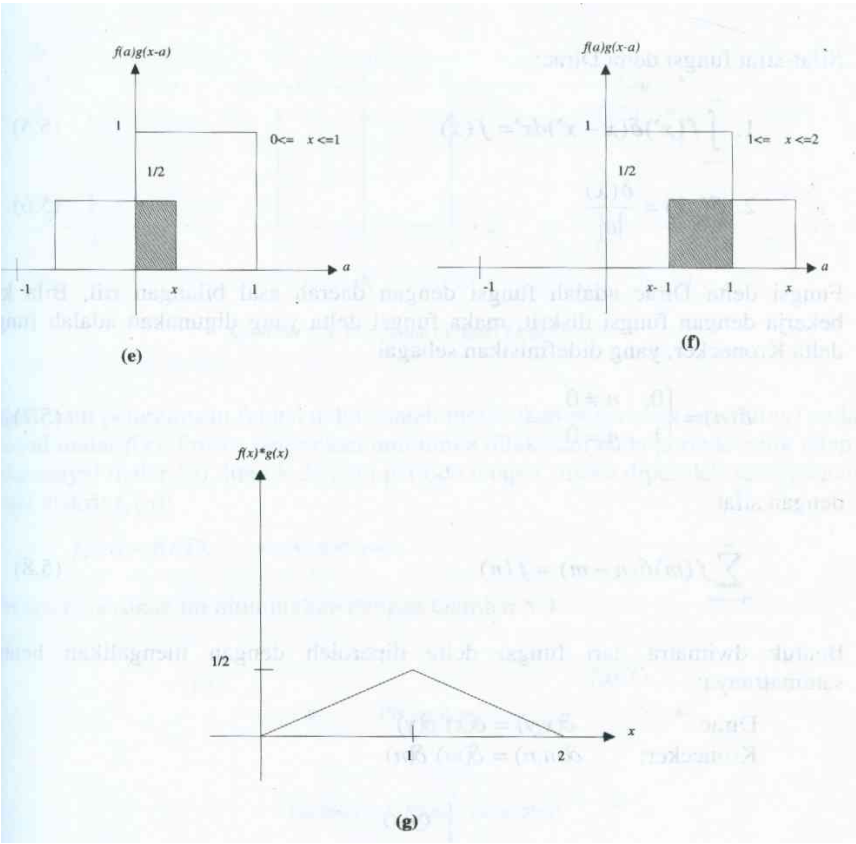
ini, jumlah perkalian kedua fungsi pada setiap titik merupakan hasil konvolusi yang dinyatakan dengan keluaran *h(x)*

Ilustrasi konvolusi adalah sebagai berikut. Misailkan fungsi *f(x)* dan *g(x)* diperlihatkan pada Gambar 4.1(a) dan 4.1(b). Langkah-langkah perhitungan hasil konvolusi ditunjukkan mulai dan Gambar 4.1(c) sampai 4.1(f). Hasil konvolusi ditunjukkan pada Gambar 4.1(g), yaitu:

$$f(x)*g(x) = \begin{cases} x/2, 0 \le x < 1 \\ 1 - x/2, 1 \le x \le 2 \\ 0, lainnya \end{cases}$$



Gambar 4.1 Ilustrasi prose konvolusi



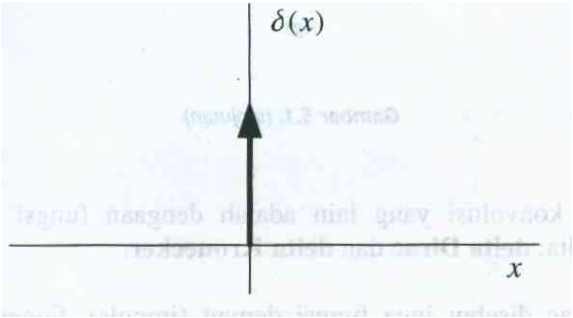
Gambar 4.1. (lanjutan)

Contoh ilustrasi konvolusi yang lain adalah dengan fungsi delta.Ada dua macam fungsi delta: **delta Dirac** dan delta **Kronecker.**

Fungsi delta Dirac disebut juga fungsi denyut (impuls). Fungsi ini bernilai 0 untuk x 0, dan "lebar" denyutnya sama dengan 1. Secara matematis fungsi delta Dirac definisikan sebagai

$$\delta(x) = 0, x \neq 0$$

$$\lim_{\varepsilon \to 0} \int_{-\varepsilon}^{\varepsilon} \delta(x)dx = 1$$



Gambar 4.2 Fungsi delta Dirac

Gambar 4.2 memperlihatkan bentuk fungsi delta Dirac.

Sifat-sifat fungsi delta Dirac:

1. $\int f(x)\delta(x-x')dx = f(x)$

2. $\delta(ax) = \dfrac{\delta(x)}{|a|}$

Fungsi delta Dirac adalah fungsi dengan daerah asal bilangan nil. Bila kita bekerja dengan fungsi diskrit, maka fungsi delta yang digunakan adalah fungsi delta Kronecker, yang didefinisikan sebagai

$$\delta(n) = \begin{cases} 0, n \neq 1 \\ 1, n = 0 \end{cases}$$

dengan sifat

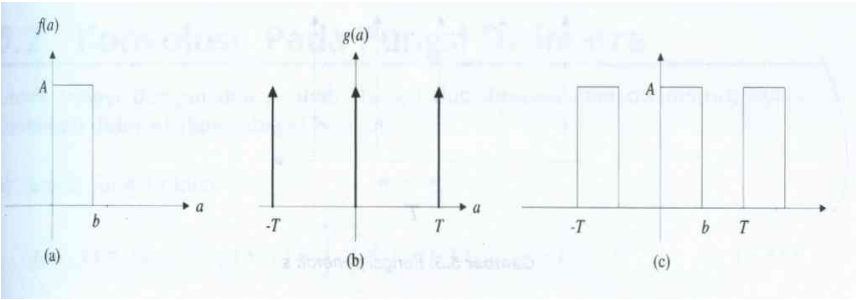
$$\sum_{m=-\infty}^{\infty} f(m)\delta(n-m) = f(n)$$

Bentuk dwimatra dari fungsi delta diperoleh dengan mengalikan bentuk satumatranya :

Dirac : $\delta(x, y) = \delta(x)\delta(y)$

Kronecker : $\delta(m,n) = \delta(m)\delta(n)$

Hasil konvolusi (fungsi f(x) pada gambar 4.3(a) dengan fungsi g(x) = $\delta(x+T)+\delta(x)+\delta(x-T)$ pada gambar 4.3(b) ditunjukkan pada gambar 4.3(c).

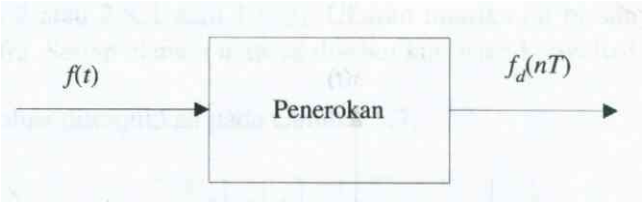


Gambar 4.3 Konvolusi dengan fungsi impuls

Salah satu penggunaan fungsi delta adalah melakukan penerokan *(sampling)* pada sinyal malarf(x). Proses penerokan umumnya dilakukan pada periode yang tetap. Jika sinyal malar *f(t)* diterok dengan periode tetap *T,* maka diperoleh serangkaian nilai diskrit $f_d(n)$:

$f_d(n) = f(nT), \quad -\infty < n < +\infty$

Proses penerokan ini ditunjukkan dengan Gambar *4.4.*



Gambar 4.4 Proses penerokan

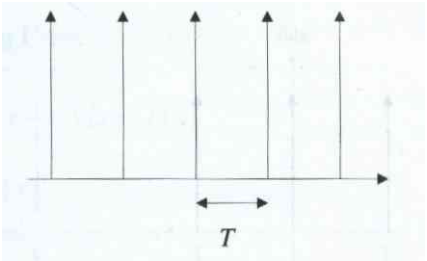
Secara matematis, proses penerokan dinyatakan sebagai perkalian sinyal malar *f(t)* dengan fungsi penerok berupa rentetan sinyal delta sejarak *T* satu sama lain (Gambar *4.5)*. Fungsi penerok itu dapat dinyatakan sebagai

$$s(t) = \sum_{-\infty}^{\infty} \delta(t - nT)$$

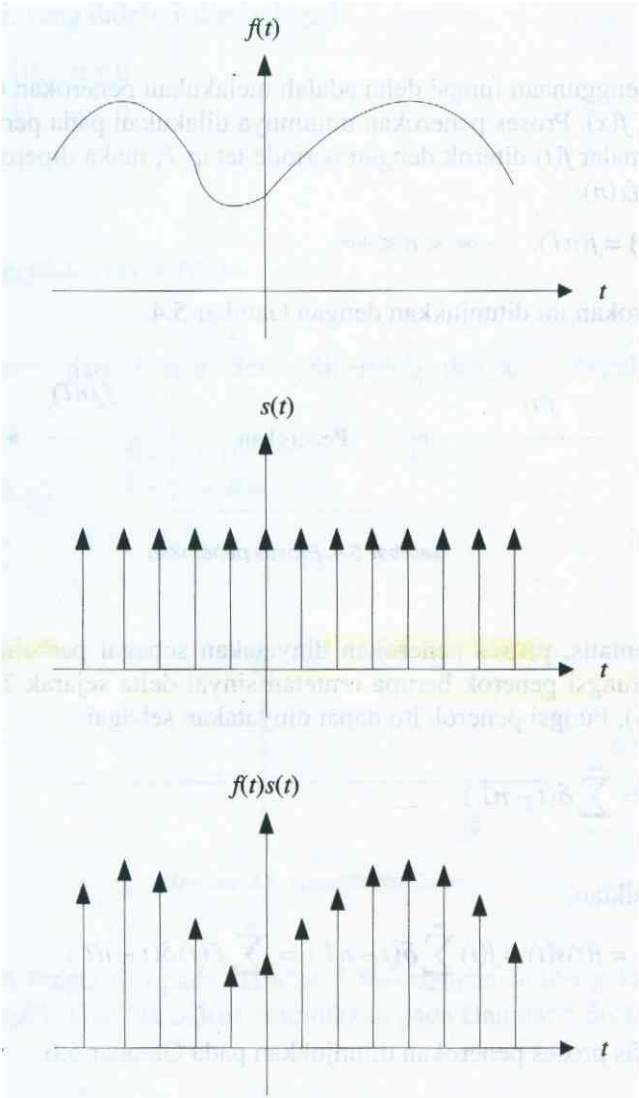
Dengan demikian,

$$f_d(t) = f(t)s(t0 = f(t) \sum_{-\infty}^{\infty} \delta(t - nT) = \sum_{-\infty}^{\infty} f(t)\delta(t - nT)$$

Ilustrasi grafis proses penerokan rlitunjukkan pada Gambar *4.6*



Gambar 4.5 Fungsi penerokan s



Gambar 4.6 Ilustrasi grafis proses penerokan

Konvolusi Pada Fungsi Dwimatra

Untuk fungsi dengan dua peubah (fungsi dua dimensi atau dwimatra), operasi konvolusi didefinisikan sebagai berikut:

a) untuk fungsi malar

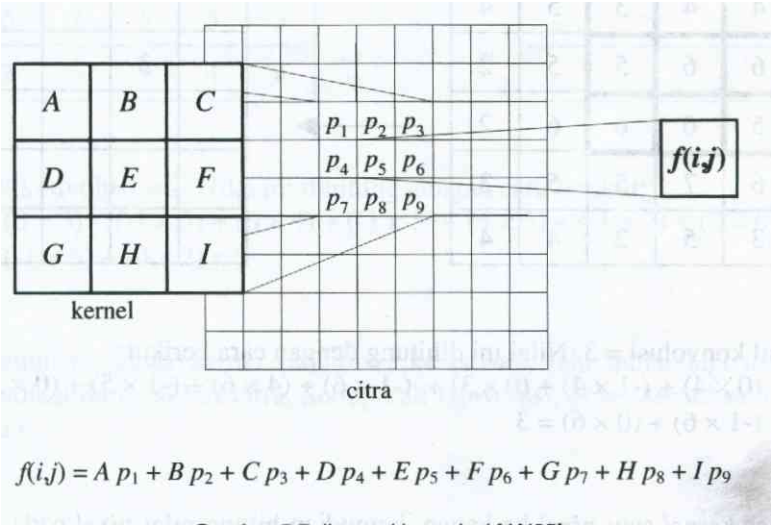
$$h(x, y) = f(x, y) * g(x, y) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(a,b)g(x-a, y-b)dadb$$

b) untuk fungsi diskrit

$$h(x, y) = f(x, y) * g(x, y) = \sum_{a=-\infty}^{\infty}\sum_{b=-\infty}^{\infty} f(a,b)g(x-a, y-b)$$

Fungsi penapis *g(x,y)* disebut juga *convolution filter, convolution mask, convolution kernel,* atau *template.* Dalam ranah diskrit kernel konvolusi dinyatakan dalam bentuk matriks (umumnya 3 x 3, namun ada juga yang berukuran 2 x 2 atau 2 x 1 atau 1 x 2). Ukuran mathks im biasanya lebih kecil dan ukuran citra. Setiap elemen matriks disebut koefisien konvolusi.

Ilustrasi konvolusi ditunjukkan pada Gambar 4.7.



Gambar 4.7 Ilustrasi konvolusi

Operasi konvolusi dilakukan dengan menggeser kernel konvolusi pixel per pixel. Hasil konvolusi disimpan di dalam matriks yang baru.

Contoh 4.1. Misalkan citra f(x,y) yang berukuran 5 x 5 dan sebuah kernel atau mask yang berukuran 3 x 3 masing-masing adalah sebagai berikut :

$$f(x,y) = \begin{bmatrix} 4 & 4 & 3 & 5 & 4 \\ 6 & 6 & 5 & 5 & 2 \\ 5 & 6 & 6 & 6 & 2 \\ 6 & 7 & 5 & 5 & 3 \\ 3 & 5 & 2 & 4 & 4 \end{bmatrix} \qquad g(x,y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & \bullet 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

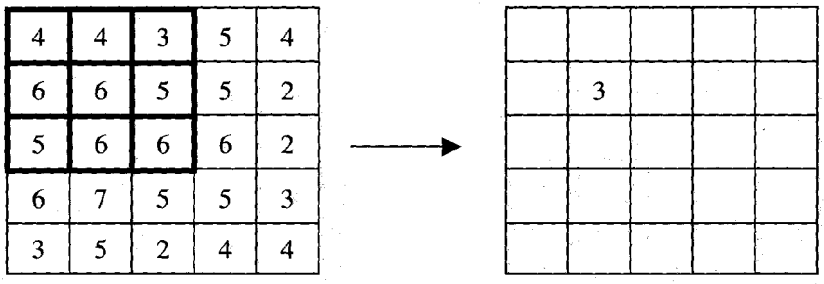
(keterangan : tanda ● menyatakan posisi (0,0) dari kernel)

Operasi konvolusi antara citra f(x,y) dengan kernel g(x,y),

f(x,y)*g(x,y)

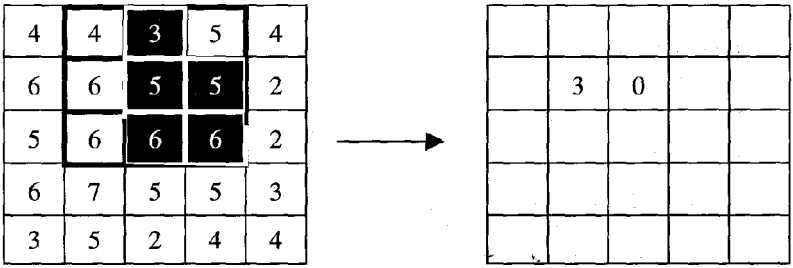
dapat diilustrasikan sebagai berikut :

1. Tempatkan kernel pada sudut kiri atas, kemudian hitung nilai pixel pada posisi (0,0) dari kernel :



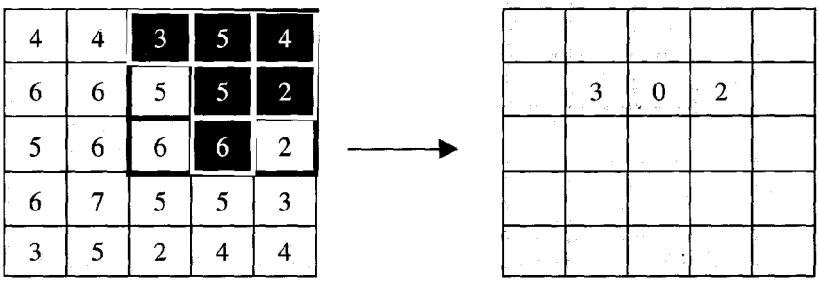
Hasil konvolusi = 3. Nilai dihitung dengan cara berikut :
(0x4) + (-1x4)+(0x3)+(-1x6)+(4x6)+(-1x5)+(0x5)+(-1x6)+(0x6) = 3

2. Geser kernel satu pixel ke kanan, kemudian hitung nilai pixel pada posisi (0,0) dari kernel :



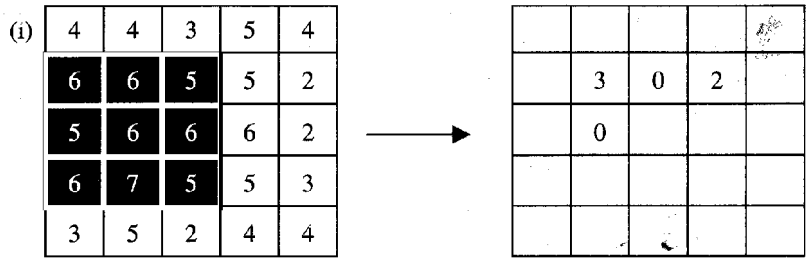
Hasil konvolusi = 0. Nilai ini dihitung dengan cara berikut :
(0x4)+(-1x3)+(0x5)+(-1x6)+(4x5)+(-1x5)+(0x6)+(-1x6)+(0x6)=0

3. Geser kernel saru pixel ke kanan, kemudian hitung nilai pixel pada posisi (0,0) dari kernel :

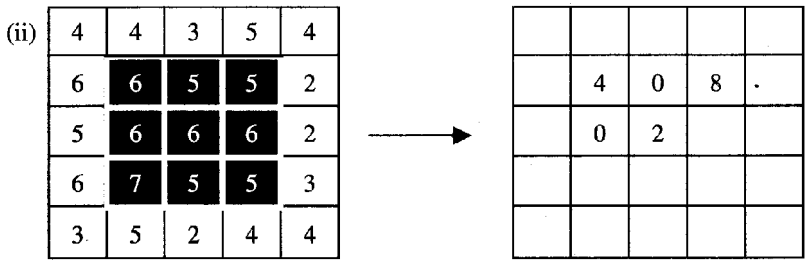


Hasil konvolusi = 2. Nilai ini dihitung dengan cara berikut :
(0x3)+(-1x5)+(0x4)+(-1x5)+(4x5)+(-1x2)+(0x6)+(-1x6)+(0x2)=2

4. Selanjutnya, geser kernel satu pixel ke bawah, lalu mulai lagi melakukan konvolusi dari sisi kiri citra. Setiap kali konvolusi, geser kernel satu pixel ke kanan :

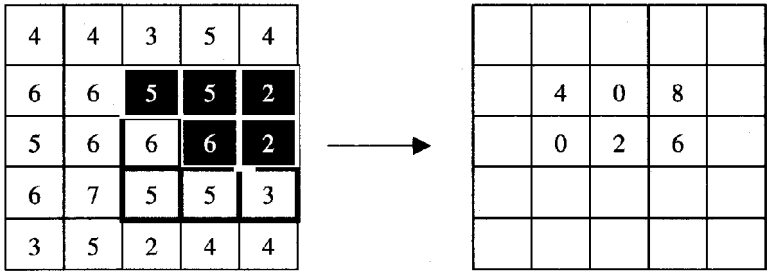


Hasil konvolusi = 0. Nilai ini dihitung dengan cara berikut :
(0x6)+(-1x6)+(0x5)+(-1x5)+(4x6)+(-1x6)+(0x6)+(-1x7)+(0x5)=0



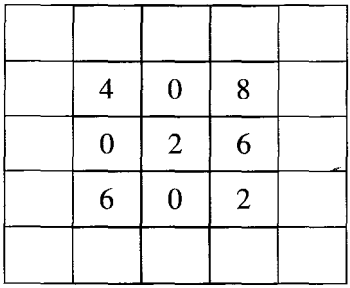
Hasil konvolusi = 2. Nilai ini dihitung dengan cara berikut :
(0x6)+(-1x5)+(0x5)+(-1x6)+(4x6)+(-1x6)+(0x7)+(-1x5)+(0x5)=2

(iii)



Hasil konvolusi = 6. Nilai ini dihitung dengan cara berikut :
(0x5)+(-1x5)+(0x2)+(-1x6)+(4x6)+(-1x2)+(0x5)+(-1x5)+(0x3)=6

Dengan cara yang sama seperti tadi, maka *pixel-pixel* pada baris ketiga dikonvolusi sehingga menghasilkan:



Sebagai catatan, jika hasil konvolusi menghasilkan nilai *pixel* negatif, maka nilai tersebut dijadikan 0, sebaliknya jika hasil konvolusi menghasilkan nilai *pixel* lebih besar dan nilai keabuan maksimum, maka nilai tersebut dijadikan ke nilai keabuan maksimum (ingat operasi *clipping).*

Masalah timbul bila *pixel* yang dikonvolusi adalah *pixel* pinggir *(border),* karena beberapa koefisien konvolusi tidak dapat dapat diposisikan pada *pixel-pixel* citra (efek “menggantung”), seperti contoh di bawah ini:

4	4	3	5	4	?
6	6	5	5	2	?
5	6	6	6	2	?
6	7	5	5	3	
3	5	2	4	4	

Masalah "menggantung" seperti ini selalu terjadi pada *pixel-pixel* pinggir kiri, kanan, atas, dan bawah. Solusi untuk masalah ini adalah :

1. *Pixel-pixel* pinggir diabaikan, tidak di-konvolusi. Solusi mi banyak dipakai di dalam pustaka fungsi-fungsi pengolahan citra. Dengan cara seperti ini, maka *pixel-pixel* pinggir nilainya tetap sama seperti citra asal. Gambar 4.8 memperlihatkan hasil konvolusi pada Contoh 4.1, yang dalam hal im nilai *pixel-pixel* pinggir sama dengan nilai *pixel* semula.

2. Duplikasi elemen citra, misalnya elemen kolom pertama disalin ke kolom $M+1$, begitu juga sebaliknya, lalu konvolusi dapat dilakukan terhadap *pixel- pixel* pinggir tersebut.

3. Elemen yang ditandai dengan "?" diasumsikan bemilai 0 atau konstanta yang lain, sehingga konvolusi *pixel-pixel* pinggir dapat dilakukan.

Solusi dengan ketiga pendekatan di atas mengasumsikan bagian pinggir citra lebamya sangat kecil (hanya satu *pixel)* relatif dibandingkan denagn ukuran citra, sehingga *pixel-pixel* pinggir tidak memperlihatkan efek yang kasat mata.

4	4	3	5	4
6	4	0	8	2
5	0	2	6	2
6	6	0	2	3
3	5	2	4	4

Gambar 4.8 Pixel-pixel pinggir (yang tidak diarsir) tidak dikonvolusi (dan Contoh 4.1)

Pixel yang dikonvolusi adalah elernen *(i, j).* Delapan buah *pixel* yang bertetangga dengan *pixel (i,j)* diperlihatkan pada Gambar *4.9.*

i-1, j-1	i-1, j	i-1, j+1
i, j-1	i,j	i, j+1
i+1, j-1	i+1,j	i+1, j+1

Gambar 4.9 mask 3 x 3

Anda dapat *melihat* bahwa operasi konvolusi merupakan komputasi pada aras lokal, karena komputasi untuk suatu *pixel* pada citra keluaran melibatkan *pixel- pixel* tetangga pada citra masukannya.

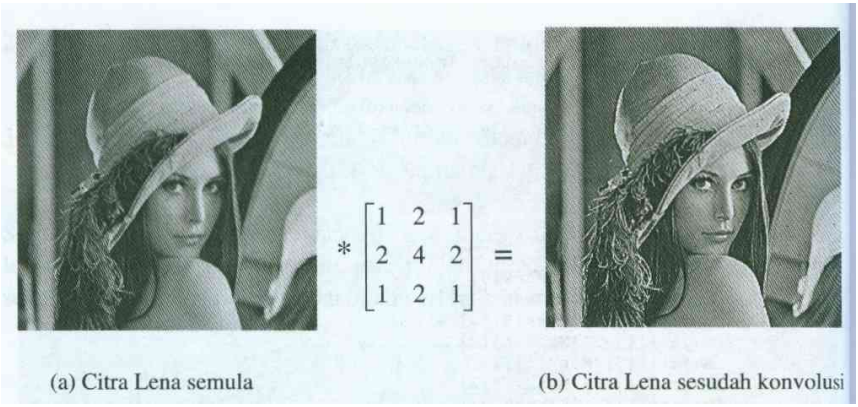
Konvolusi berguna pada proses pengolahan citra seperti:
- perbaikan kualitas citra (image enhancement)

- penghilangan derau

- mengurangi erotan
- penghalusan / pelembutan citra
- deteksi tepi, penajaman tepi
- dll

Sebagai contoh, Gambar 4.10 memperlihatkan konvolusi citra Lena dengan penapis Gaussian untuk mempertajam tepi-tepi di dalam citra. Penapis Gaussian adalah sebuah mask yang berukuran 3 x 3:

$$g(x,y) = \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$



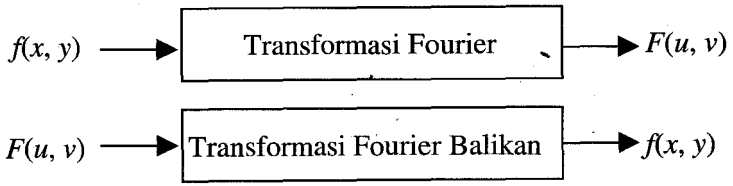
Gambar 4.10 Konvolusi citra Lena dengan penapis Gaussian untuk mempertajam gambar.

Karena konvolusi dilakukan per *pixel* dan untuk setiap *pixel* dilakukan operasi ,erkalian dan penjumlahan, maka jelas konvolusi mengkonsumsi banyak waktu. ika citra berukuran $N \times N$ dan *kernel* berukuran $m \times m$, maka jumlah perkalian tdalah dalam orde N^2m^2.

Sebagai contoh jika citra berukuran 512 x 512 dan kernel berukuran 16 x 16, maka ada sekitar 32 juta perkalian yang dibutuhkan. mi elas tidak cocok untuk proses yang *real time* tanpa perangkat keras yang d*edicated.*

*S*atu cara mengurangi waktu komputasi adalah mentransformasi citra dan *kernel* ke dalam ranah frekuensi, selanjutnya konvolusi dilakukan dalam ranah waktu. Keuntungan utama dari penggunaan ranah frekuensi adalah proses konvolusi dapat diterapkan dalam bentuk perkalian langsung.

Proses perubahan fungsi dan ranah ranah spasial ke ranah frekuensi dilakukan nelalui **Transformasi Fourier.** Sedangkan perubahan fungsi dan ranah rekuensi ke ranah spasial dilakukan melalui Transformasi **Fourier Balikan (***invers).*



Dengan demikian, operasi konvolusi dua buah fungsi dalam ranah frekuensi menjadi:

$$h(x, y) = f(x, y) * g(x, y) \leftrightarrow \quad H(u, v) = F(u, v)\, G(u, v)$$

Transformasi Fourier

Transformasi Fourier merupakan transformasi paling penting di dalam bidang pengolahan sinyal *(*signal *processing),* khususnya pada bidang pengolahan citra.

Umumnya sinyal dinyatakan sebagai bentuk plot amplitudo versus waktu (pada fungsi satu matra) atau plot amplitudo versus posisi spasial (pada fungsi dwimatra). Pada beberapa aplikasi pengolahan sinyal, terdapat kesukaran melakukan operasi karena fungsi dalam ranah waktu/spasial, misalnya pada operasi konvolusi di atas. Operasi konvolusi dapat diterapkan sebagai bentuk perkalian langsung bila fungsi berada dalam ranah frekunsi.

Transformasi Fourier adalah kakas *(tool)* untuk mengubah fungsi dari ranah waktu/spasial ke ranah frekuensi. Untuk perubahan sebaliknya digunakan Transformasi Fourier Balikan. Intisari dan Transformasi Fourier adalah menguraikan sinyal atau gelombang menjadi sejumlah sinusoida dari berbagai frekuensi, yang jumlahnya ekivalen dengan gelombang asal.

Di dalam pengolahan citra, transformasi Fourier digunakan untuk menganalisis Irekuensi pada operasi seperti perekaman citra, perbaikan kualitas citra, restorasi citra, pengkodean, dan lain-lain. Dan analisis frekuensi, kita dapat melakukan perubahan frekuensi pada gambar. Perubahan frekuensi berhubungan dengan spektrum antara gambar yang kabus kontrasnya samapi gambar yang kaya akan rincian visualnya. Sebagai contoh, pada proses perekaman citra mungkin terjadi

pengaburan kontras gambar. Pada gambar yang mengalami kekaburan kontras ierjadi perubahan intensitas secara perlahan, yang berarti kehilangan informasi frekuensi tinggi. Untuk meningkatkan kualitas gambar, kita menggunakan penapis frekuensi tinggi sehingga *pixel* yang berkontras kabur dapat dinaikkan intensitasnya.