



Filtering dan Masking

Pengolahan Citra Digital

1. Filtering --> Menggunakan Kernel → Konvolusi
 2. Masking --> menggunakan Mask
- **Filtering** → tujuan/proses untuk menyaring atau memodifikasi citra.
 - **Kernel** → matriks kecil yang digunakan dalam filtering.
 - **Konvolusi** → operasi matematis untuk menerapkan kernel pada citra.
 - **Masking** → proses memilih atau mempertahankan area tertentu dari citra menggunakan mask.

Konvolusi

- Konvolusi (convolution) adalah operasi untuk mengolah suatu citra dengan menggunakan sebuah kernel/filter yang digeser pada seluruh bagian citra.

Tujuannya adalah untuk menghasilkan informasi tertentu dari citra, misalnya:

- mendeteksi tepi (edge),
- mendeteksi garis horizontal/vertikal,
- memperjelas pola,
- menghaluskan citra,
- mendeteksi tekstur,
- atau pada CNN, mengekstraksi fitur.

Secara sederhana: $\text{Citra} * \text{Kernel} \rightarrow \text{Feature Map}$

Teori Konvolusi

- Untuk konvolusi dalam fungsi diskrit, didefinisikan sbb:

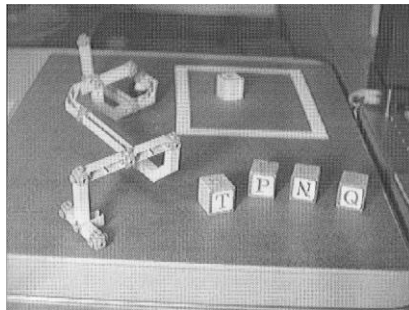
$$h(x) = f(x) * g(x) = \sum_{a=-\infty}^{\infty} f(a)g(x-a)$$

- $g(x)$ disebut sebagai kernel konvolusi atau kernel penapis (filter).
- Kernel $g(x)$ merupakan suatu jendela yang dioperasikan secara bergeser pada sinyal masukan $f(x)$
- Jumlah perkalian kedua fungsi pada setiap titik merupakan hasil konvolusi yang dinyatakan dengan keluaran $h(x)$.

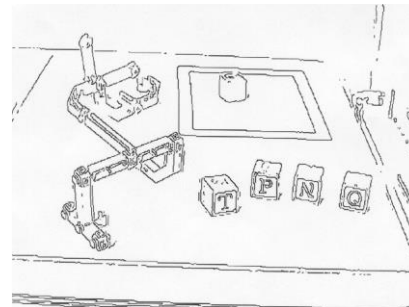
- Di dalam pengolahan citra, sebuah citra sering dilakukan proses penapisan (*filtering*) untuk memperoleh citra sesuai dengan tujuan yang diinginkan.



filtering



filtering



Sumber gambar: *Image Fitering*, CS485/685 Computer Vision, Prof. George Bebis

- Dalam ranah diskrit kernel konvolusi dinyatakan dalam bentuk matriks kernel (3x3, 2x2 dll).
- Kernel biasanya merupakan sebuah matriks kecil yang berisi angka yang digunakan utk konvolusi
- Kernel dengan ukuran berbeda yang berisi pola angka berbeda menghasilkan hasil berbeda
- ukuran kernel bisa berubah-ubah tetapi 3x3 sering digunakan

Example kernel:

0	1	0
1	1	1
0	1	0

Catatan: Lakukan operasi *clipping* Jika hasil konvolusi menghasilkan nilai *pixel* negatif atau lebih besar dari nilai keabuan maksimum → Misal dijadikan 0 jika hasil konvolusi negatif, dijadikan 255 jika hasil konvolusi lebih besar dari 255.

Misalkan:

$$I = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

dan:

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

Kernel pertama kali ditempatkan pada bagian kiri atas citra:

$$\begin{bmatrix} \boxed{1} & \boxed{2} & 3 & 4 \\ \boxed{5} & \boxed{6} & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Ambil bagian:

$$\begin{bmatrix} 1 & 2 \\ 5 & 6 \end{bmatrix}$$

Kemudian kalikan **elemen per elemen** dengan kernel:

$$\begin{aligned} & \begin{bmatrix} 1 & 2 \\ 5 & 6 \end{bmatrix} * \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \\ &= (1 \times 1) + (2 \times 0) + (5 \times 0) + (6 \times -1) \\ &= 1 + 0 + 0 - 6 \\ & \quad \boxed{-5} \end{aligned}$$

Jadi piksel pertama feature map adalah **-5**.

Kernel digeser

Kemudian kernel digeser satu langkah ke kanan:

$$\begin{bmatrix} 2 & 3 \\ 6 & 7 \end{bmatrix}$$

Hitung:

$$\begin{aligned} & (2 \times 1) + (3 \times 0) + (6 \times 0) + (7 \times -1) \\ &= 2 - 7 \\ &= -5 \end{aligned}$$

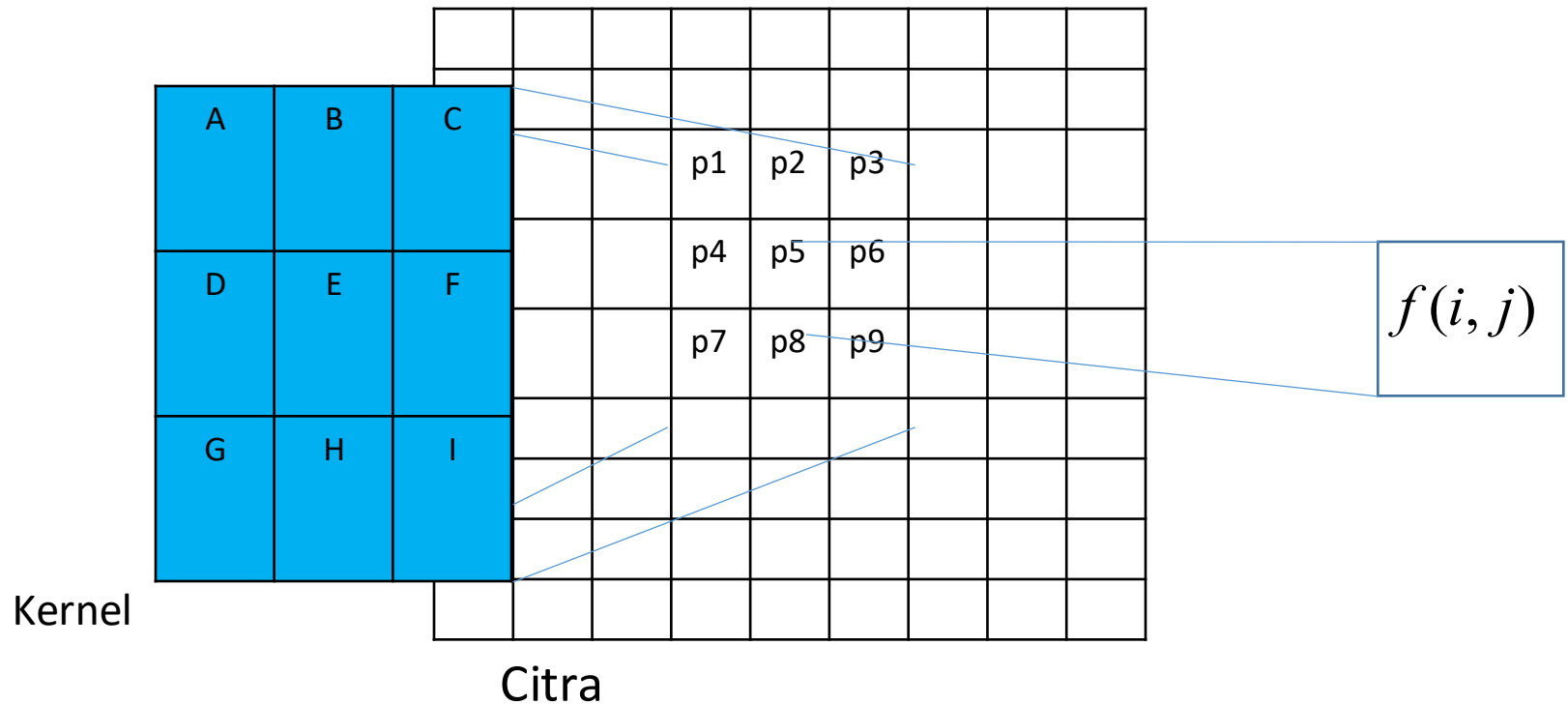
Kemudian terus digeser ke kanan dan ke bawah sampai seluruh citra diproses.

Hasil akhirnya:

$$\begin{bmatrix} -5 & -5 & -5 \\ -5 & -5 & -5 \\ -5 & -5 & -5 \end{bmatrix}$$

Itulah **feature map**.

Konvolusi pada fungsi 2 dimensi



$$f(i, j) = Ap1 + Bp2 + Cp3 + Dp4 + Ep5 + Fp6 + Gp7 + Hp8 + Ip9$$

Contoh : Misalkan citra $f(x,y)$ berukuran 5x5 dengan kernel atau mask 3x3 masing-masing sbb :

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

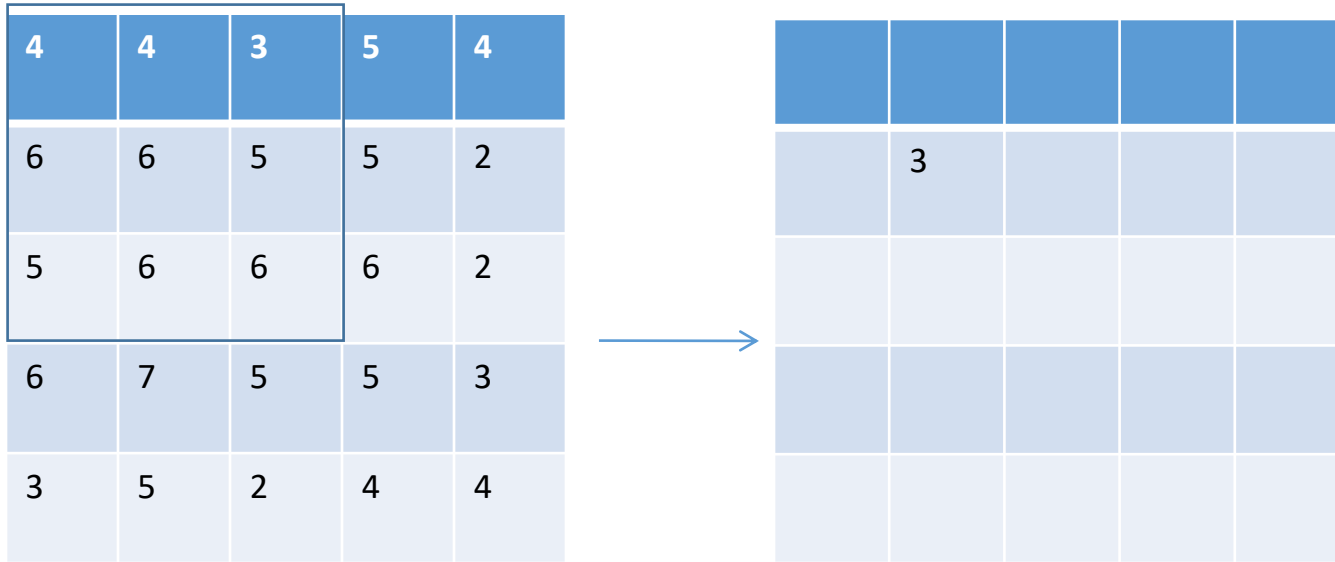
(Keterangan : tanda $\bullet$ menyatakan posisi (0,0) dari kernel)

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

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

dapat diilustrasikan sbb:

- a) Tempatkan kernel pada sudut kiri atas, kemudian hitung nilai pixel pada posisi (0,0) dari kernel.
- b) Geser kernel satu pixel ke kanan, kemudian hitung nilai pixel pada posisi (0,0) kernel, begitu seterusnya hingga geser satu pixel ke bawah, lalu mulai lagi melakukan konvolusi dari sisi kiri citra.



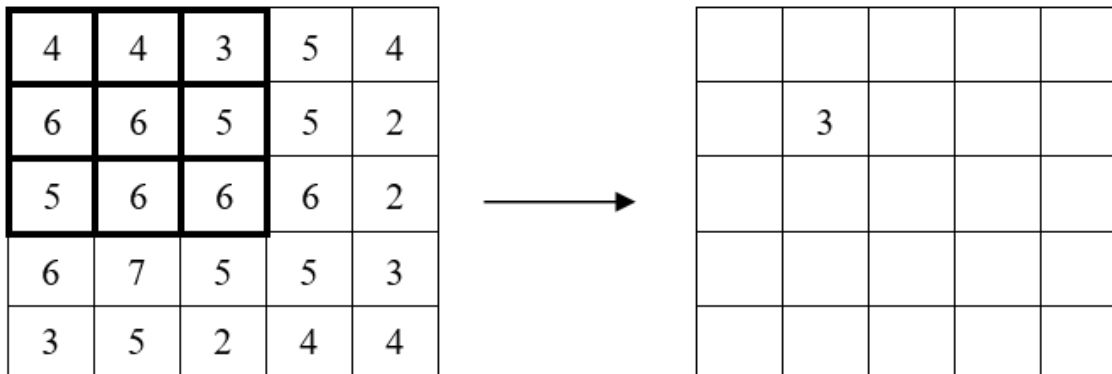
Hasil konvolusi = 3. Nilai ini dihitung sbb:

$$(0 \times 4) + (-1 \times 4) + (0 \times 3) + (-1 \times 6) + (4 \times 6) + (-1 \times 5) + (0 \times 5) + (-1 \times 6) + (0 \times 6) = 3$$

$$g(x, y) = \begin{bmatrix} 0 & -1 & 0 \\ -1 & \bullet 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

- Operasi konvolusi $f(x, y) * g(x, y)$ adalah sebagai berikut:

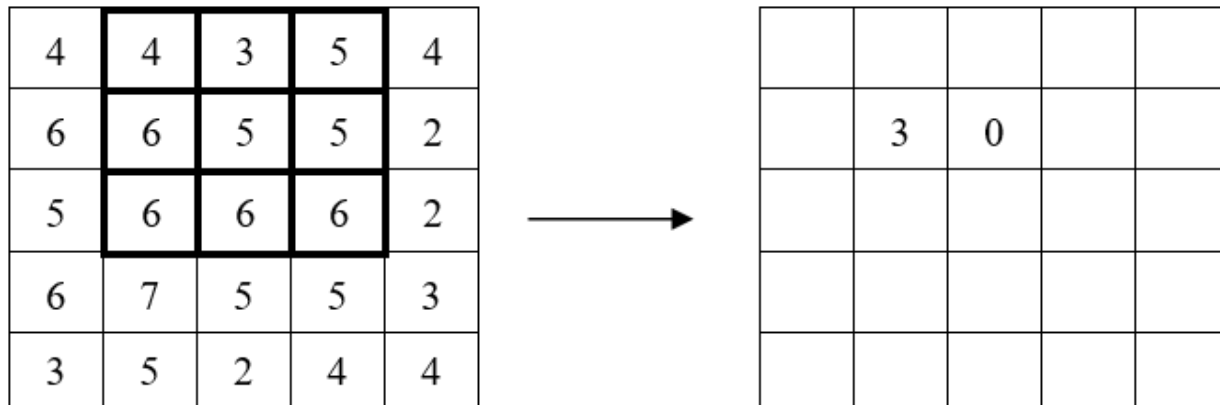
(1) Tempatkan *kernel* pada sudut kiri atas, kemudian hitung nilai *pixel* pada posisi (0, 0) dari *kernel*:



Hasil konvolusi = 3. Nilai ini dihitung dengan cara berikut:

$$(0 \times 4) + (-1 \times 4) + (0 \times 3) + (-1 \times 6) + (4 \times 6) + (-1 \times 5) + (0 \times 5) + (-1 \times 6) + (0 \times 6) = 3$$

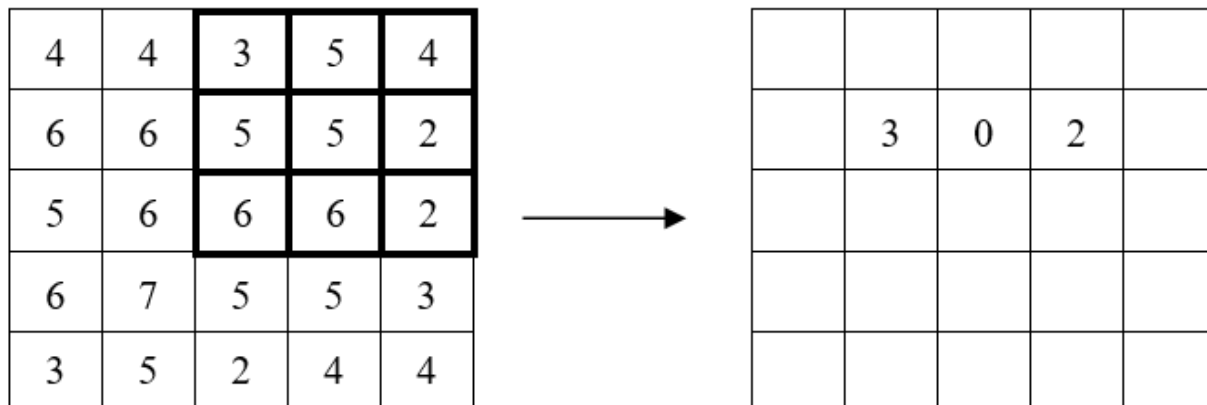
(2) Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (0, 0) dari *kernel*:



Hasil konvolusi = 0. Nilai ini dihitung dengan dengan cara berikut:

$$(0 \times 4) + (-1 \times 3) + (0 \times 5) + (-1 \times 6) + (4 \times 5) + (-1 \times 5) + (0 \times 6) + (-1 \times 6) + (0 \times 6) = 0$$

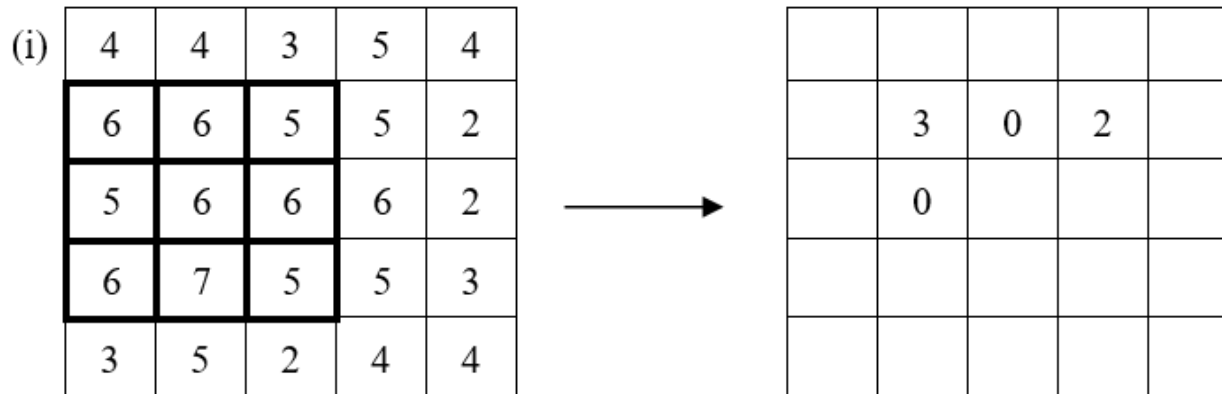
- (3) Geser *kernel* satu *pixel* ke kanan, kemudian hitung nilai *pixel* pada posisi (0, 0) dari *kernel*:



Hasil konvolusi = 2. Nilai ini dihitung dengan cara berikut:

$$(0 \times 3) + (-1 \times 5) + (0 \times 4) + (-1 \times 5) + (4 \times 5) + (-1 \times 2) + (0 \times 6) + (-1 \times 6) + (0 \times 2) = 2$$

- (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:

$$(0 \times 6) + (-1 \times 6) + (0 \times 5) + (-1 \times 5) + (4 \times 6) + (-1 \times 6) + (0 \times 6) + (-1 \times 7) + (0 \times 5) = 0$$

(ii)

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

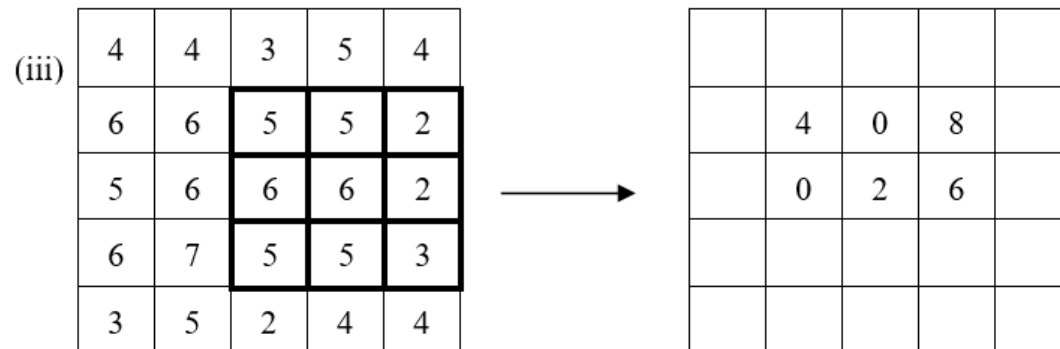


	4	0	8	
	0	2		



Hasil konvolusi = 2. Nilai ini dihitung dengan cara berikut:

$$(0 \times 6) + (-1 \times 5) + (0 \times 5) + (-1 \times 6) + (4 \times 6) + (-1 \times 6) + (0 \times 7) + (-1 \times 5) + (0 \times 5) = 2$$



Hasil konvolusi = 6. Nilai ini dihitung dengan cara berikut:

$$(0 \times 5) + (-1 \times 5) + (0 \times 2) + (-1 \times 6) + (4 \times 6) + (-1 \times 2) + (0 \times 5) + (-1 \times 5) + (0 \times 3) = 6$$

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

	4	0	8	
	0	2	6	
	6	0	2	

■

Masalah timbul bila pixel yang dikonvolusi adalah pixel pinggir, karena beberapa koefisien konvolusi tidak dapat diposisikan pada pixel-pixel citra ('efek menggantung'), seperti contoh dibawah 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	

Solusi untuk masalah ‘menggantung’ ini adalah :

1. *Piksel* pinggir di abaikan, tidak di Konvolusi. Solusi ini banyak di pakai di dalam pustaka fungsi – fungsi pengolahan citra. *Piksel* tepi tidak digunakan sebagai pusat kernel sehingga hanya bagian citra yang memiliki pasangan *piksel* lengkap yang diproses. Akibatnya, ukuran citra hasil konvolusi menjadi lebih kecil.
2. Dupliaksi elemen citra, misalnya elemen kolom pertama disalin ke kolom $M+1$, begitu juga sebaliknya, lalu konvolusi *piksel* pinggir tersebut.
3. Padding → penambahan *piksel* di sekeliling tepi citra sebelum proses konvolusi. Tujuannya agar kernel tetap bisa memproses *piksel*-*piksel* di bagian tepi.
4. Ambil bagian yang tidak punya pasangan dengan bagian lain dari citra. Indeks melingkar dilaksanakan dengan mengambil data pada posisi di seberang citra, sedangkan indeks tercermin diambilkan dari baris/kolom yang ada di dekatnya.

Piksel pinggir diabaikan

Ini disebut juga **valid convolution**.

Karena kernel 3×3 tidak bisa ditempatkan dengan pusatnya di tepi, hanya memproses bagian yang lengkap.

Citra:

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Maka hasilnya:

$$\begin{bmatrix} 54 & 63 \\ 90 & 99 \end{bmatrix}$$

Ukuran piksel berubah:

$$4 \times 4 \rightarrow 2 \times 2$$

Kelebihan: tidak perlu membuat nilai tambahan.

Kekurangan: informasi pada bagian tepi tidak digunakan sebagai pusat kernel.

Duplikasi piksel luar

Citra awal:

$$\begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

Setelah dilakukan duplikasi satu piksel:

$$\begin{bmatrix} 1 & 1 & 2 & 3 & 4 & 4 \\ 1 & 1 & 2 & 3 & 4 & 4 \\ 5 & 5 & 6 & 7 & 8 & 8 \\ 9 & 9 & 10 & 11 & 12 & 12 \\ 13 & 13 & 14 & 15 & 16 & 16 \\ 13 & 13 & 14 & 15 & 16 & 16 \end{bmatrix}$$

Sekarang kernel 3×3 dapat bergerak pada seluruh piksel asli.

Misalnya pada piksel pojok kiri atas:

$$\begin{bmatrix} 1 & 1 & 2 \\ 1 & 1 & 2 \\ 5 & 5 & 6 \end{bmatrix}$$

Dengan cara ini, ukuran output dapat dipertahankan:

$$4 \times 4 \rightarrow 4 \times 4$$

Padding

Ini yang paling sering digunakan dalam pembahasan CNN.

Misalnya kita menggunakan **zero padding**.

Ditambahkan nilai 0 di sekeliling citra:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 4 & 0 \\ 0 & 5 & 6 & 7 & 8 & 0 \\ 0 & 9 & 10 & 11 & 12 & 0 \\ 0 & 13 & 14 & 15 & 16 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Karena kernel 3×3 , digunakan:

$$\boxed{Padding = 1}$$

Sekarang kernel bisa ditempatkan pada piksel tepi.

Misalnya piksel pojok kiri atas

Kernel melihat:

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 2 \\ 0 & 5 & 6 \end{bmatrix}$$

Hasil akhirnya:

$$\boxed{\begin{bmatrix} 14 & 24 & 30 & 22 \\ 33 & 54 & 63 & 45 \\ 57 & 90 & 99 & 69 \\ 46 & 72 & 78 & 54 \end{bmatrix}}$$

Sehingga:

$$\boxed{4 \times 4 \rightarrow 4 \times 4}$$

Jadi **padding** memungkinkan piksel tepi tetap diproses.

Indeks melingkar / Circular Padding

Pada metode ini, jika kernel keluar dari sisi citra, maka dia kembali mengambil nilai dari sisi yang berlawanan.

Citra awal:

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Jika kernel membutuhkan piksel **di sebelah kiri kolom pertama**, maka mengambil:
kolom terakhir

Jadi sebelah kiri dari:

1

dianggap:

4

Sedangkan sebelah kanan dari:

4

dianggap:

1

Begitu juga bagian atas dan bawah.

Metode ini juga mempertahankan ukuran:

$$\boxed{4 \times 4 \rightarrow 4 \times 4}$$

Hitung Konvolusi Citra berikut

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Dengan kernel berikut

$$\begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix}$$

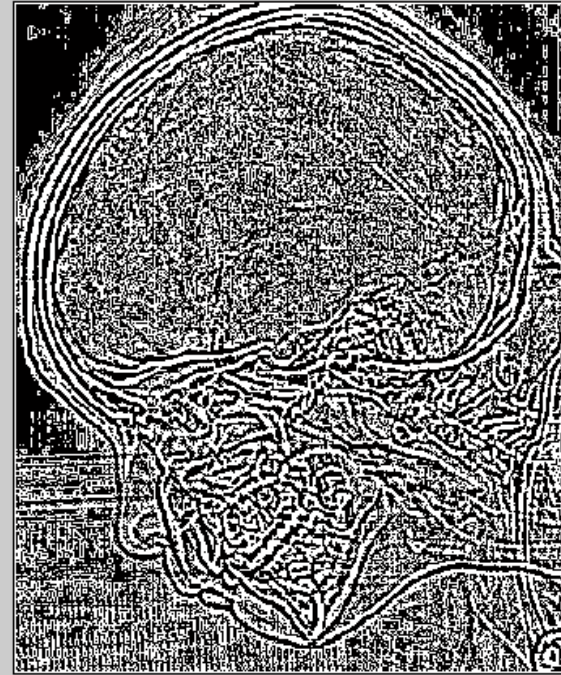
Konvolusi dg kernel

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

image asli



hasil konvolusi



Contoh: Untuk mempertajam citra, sebuah penapis Gaussian digunakan. Penapis Gaussian adalah sebuah *mask* berukuran 3 x 3:

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



(a) Citra Lena semula

$$* \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} =$$



(b) Citra Lena sesudah konvolusi

Some other kernel examples

1	1	1
1	1	1
1	1	1

Unweighted 3x3 smoothing kernel

0	1	0
1	4	1
0	1	0

Weighted 3x3 smoothing kernel with Gaussian blur

0	-1	0
-1	5	-1
0	-1	0

Kernel to make image sharper

-1	-1	-1
-1	9	-1
-1	-1	-1

Intensified sharper image



Gaussian Blur



Sharpened image

Sumber: Image Convolution, Jamie Ludwig, Satellite Digital Image Analysis, 581
Portland State University

Filtering

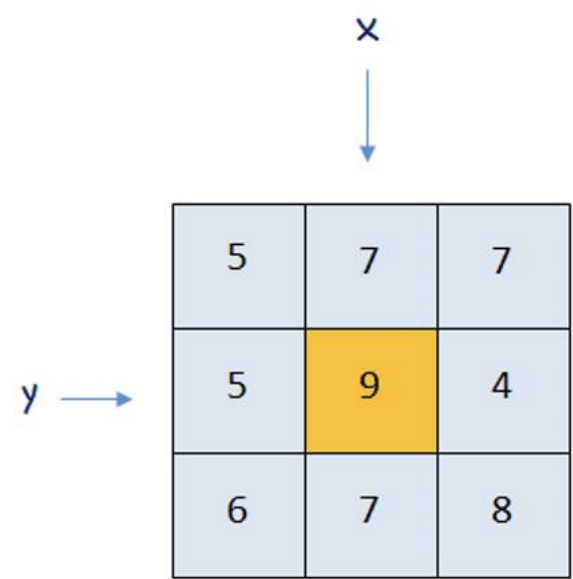
- Filtering adalah proses memodifikasi nilai piksel pada citra menggunakan suatu filter/kernel untuk memperbaiki kualitas citra, mengurangi noise, atau menonjolkan karakteristik tertentu.
- Sederhananya, filtering dapat dianggap seperti “menyaring” informasi pada citra agar bagian yang diinginkan lebih terlihat.

Aplikasi Ketetanggaan Piksel pada Filter

- Ada tiga jenis filter yang menggunakan operasi ketetanggaan piksel : filter batas, filter pererataan, dan filter median.
- Sebagai filter, operasi ketetanggaan piksel berfungsi untuk menyaring atau paling tidak mengurangi gangguan atau penyimpangan pada citra.
- Idenya adalah mencegah piksel yang intensitasnya di luar intensitas piksel-piksel tetangga.

filter batas

- $\min = \text{minimum}(5, 7, 7, 5, 4, 6, 7, 8) = 4$;
- $\text{maks} = \text{maksimum}(5, 7, 7, 5, 4, 6, 7, 8) = 8$;
- 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.



5	7	7
5	9	4
6	7	8

Perlu diketahui, pemrosesan hanya dilakukan selain baris pertama, baris terakhir, kolom pertama, dan kolom terakhir. Keempat area tersebut tidak diproses karena tidak mempunyai tetangga yang lengkap (sebanyak 8).

Efek hasil filter batas



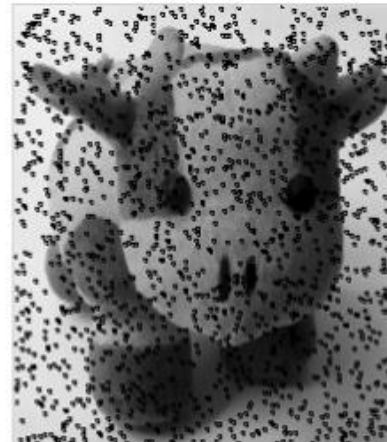
(a) Citra mobil yang telah diberi bintik-bintik putih



(b) Hasil pemfilteran gambar (a)



(c) Citra bird dengan derau



(d) Hasil pemfilteran gambar (c)

Filter Pererataan

- *Filter pererataan menggunakan Rumus*

$$g(y, x) = \frac{1}{9} \sum_{p=-1}^1 \sum_{q=-1}^1 f(y + p, x + q)$$

Sebagai contoh, piksel pada $f(y, x)$ dan kedelapan tetangganya memiliki nilai-nilai kecerahan seperti berikut.

65	50	55
78	68	60
60	60	62

Contoh

Pada contoh di atas, yang menjadi point (yang bernilai 68) merupakan nilai pada $f(y, x)$.

Nilai rerata pengganti untuk $g(y, x)$ dihitung dengan cara seperti berikut:

$$g(y, x) = 1/9 \times (65+50+55+78+68+60+60+60+62) = 61,7778$$

$$\cong 62$$

Jadi, nilai 68 pada $f(y, x)$ diubah menjadi 62 pada $g(y, x)$.

Hasil dari filter pererataan



(a) Citra mobil dengan bintik-bintik putih



(b) Hasil pemrosesan mobil



(c) Citra boneka berbintik dengan derau



(d) Hasil pemrosesan boneka

efek pemrosesan dengan filter pererataan. Dibandingkan dengan filter batas, hasil pemrosesan filter pererataan tidak menghilangkan bintik-bintik putih pada citra mobil, tetapi hanya agak menyamarkan. Pada citra boneka derau lebih dihaluskan

Filter Median

- Filter median sangat populer dalam pengolahan citra.
- Filter ini dapat dipakai untuk menghilangkan derau bintik-bintik.
- Nilai yang lebih baik digunakan untuk suatu piksel ditentukan oleh nilai median dari setiap piksel dan kedelapan piksel tetangga pada 8-ketetanggaan.

$$g(y, x) = \text{median}(\begin{aligned} &f(y - 1, x - 1), f(y - 1, x), f(y - 1, x + 1), \\ &f(y, x - 1), f(y, x), f(y, x + 1), \\ &f(y + 1, x - 1), f(y + 1, x), f(y + 1, x + 1)) \end{aligned})$$

10	13	10
10	10	12
12	12	12



10 10 10 10 12 12 12 12 13 ← Diurutkan

Nilai di tengah
(median)

untuk mendapatkan median,
diperlukan pengurutan (*sorting*)
terlebih dulu.

FILTER; Median

136	127	142	135	143	152	172	173	189	184
143	138	125	145	138	145	137	171	175	185
157	145	139	133	146	139	136	147	157	164
165	154	157	145	143	147	147	133	146	158
174	158	167	162	158	147	146	140	142	149
185	169	167	159	180	160	146	143	137	145
190	180	161	162	172	178	149	141	141	139
192	166	177	158	172	156	161	144	137	141
192	186	170	172	146	159	147	137	140	140
205	184	184	152	160	142	157	142	140	141

136	127	142	135	143	152	172	173	189	184
143	139	138	139	143	143	147	171	173	185
157	145	145	143	145	143	145	147	158	175
165	157	154	146	146	146	146	146	147	158
174	167	159	159	158	147	146	143	143	146
185	169	162	162	162	158	146	142	141	145
190	177	166	167	162	161	149	143	141	141
192	180	170	170	162	159	149	141	140	141
192	184	172	170	158	157	147	142	140	140
205	186	175	170	158	154	145	145	141	140

Matriks citra semula
Sampel 10x10 kiri atas

125 133 138 139 145 145 145 154 157



median

Matriks citra hasil penapisan
Sampel 10x10 kiri atas

Hasil dari Filter Median



(a) Citra mobil dengan bintik-bintik putih



(b) Hasil pemrosesan terhadap gambar (a)



(c) Citra boneka dengan derau



(d) Hasil pemrosesan terhadap gambar (c)

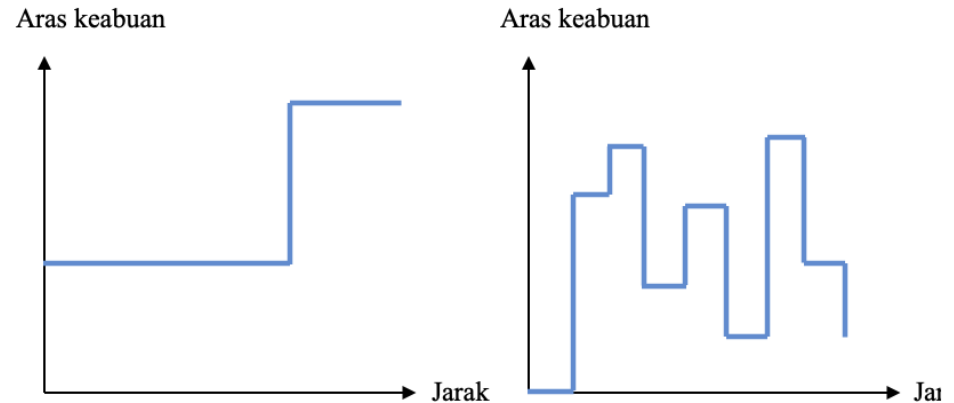
Hasilnya terlihat bahwa derau dapat dihilangkan, detail pada citra tetap dipertahankan.

Namun, hal ini tentu saja didapat dengan tambahan beban komputasi karena adanya sorting

Domain Frekuensi

Frekuensi domain pada citra menunjukkan seberapa sering suatu perubahan aras keabuan terjadi dari suatu posisi ke posisi berikutnya. Gambar disamping menunjukkan secara visual perbedaan antara frekuensi rendah dan frekuensi tinggi. Pada citra berfrekuensi tinggi, perubahan aras sering terjadi seiring dengan pergeseran jarak.

Jenis filter frekuensi : Low-pass filter dan high pass-filter

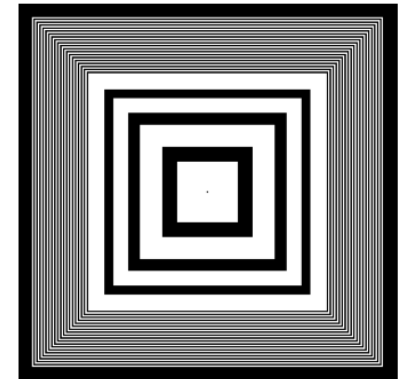


(a) Frekuensi rendah

(b) Frekuensi tinggi



(c) Citra dengan frekuensi rendah



(d) Citra berfrekuensi tinggi

Lowpass filter

- Filter lolos-bawah (*low-pass filter*) adalah filter yang mempunyai sifat dapat meloloskan yang berfrekuensi rendah dan menghilangkan yang berfrekuensi tinggi.
- Efek filter ini membuat perubahan aras keabuan menjadi lebih lembut
- Adapun yang melibatkan konvolusi menggunakan kernel $1/6$, $1/9$, $1/10$, $1/16$, Tujuannya agar setelah filtering, nilai rata-rata intensitas citra tidak berubah secara drastis.

1/6	0	1	0
	1	2	1
	0	1	0
#1			

1/9	1	1	1
	1	1	1
	1	1	1
#2			

1/10	1	1	1
	1	2	1
	1	1	1
#3			

1/16	1	2	1
	2	4	2
	1	2	1
#4			

Kernel #1

Kernel ini memberikan bobot lebih besar pada piksel tengah:

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

Artinya piksel tengah lebih dipertahankan dibandingkan piksel tetangganya. Kernel ini melakukan smoothing, tetapi hanya menggunakan tetangga atas, bawah, kiri, dan kanan.

Kernel #3

$$K_3 = \frac{1}{10} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Filter masih melakukan smoothing, tetapi lebih mempertahankan informasi pada piksel pusat dibandingkan mean filter #2.

Kernel #2

$$K_2 = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Ini merupakan mean/average filter yang sangat umum. Semua piksel memiliki bobot yang sama. Semua piksel tetangga dianggap sama penting. Akibatnya citra menjadi lebih halus, tetapi detail/edge juga dapat menjadi lebih kabur.

Kernel #4

$$K_4 = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Piksel tengah mempunyai bobot 4, sedangkan piksel diagonal hanya mempunyai bobot 1.

Dengan demikian:

Piksel yang lebih dekat dengan pusat mempunyai pengaruh lebih besar.

Kernel #4 merupakan bentuk kernel menyerupai Gaussian smoothing.

Contoh

40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128

(a) Citra dengan frekuensi rendah

128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40

(b) Citra dengan frekuensi tinggi

Dengan menggunakan kernel

$1/9$

1	1	1
1	1	1
1	1	1



(c) Citra dengan frekuensi rendah



(d) Citra dengan frekuensi tinggi

Hasil lowpass Filter

40	40	40
40	40	40
40	40	40
40	40	40
40	40	40
40	40	40
40	40	40
69	69	69
99	99	99
128	128	128
128	128	128
128	128	128
128	128	128
128	128	128
128	128	128
128	128	128
128	128	128
128	128	128

(a) Citra dengan frekuensi rendah

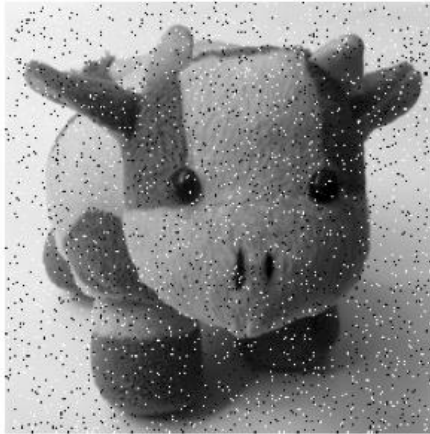
99	99	99
69	69	69
99	99	99
69	69	69
99	99	99
69	69	69
99	99	99
69	69	69
99	99	99
69	69	69
99	99	99
69	69	69
99	99	99
69	69	69
99	99	99
69	69	69

(b) Citra dengan frekuensi tinggi

Gambar (a). filter tidak membuat perubahan yang sangat berarti pada citra kecuali perubahan pada baris yang berisi 69 dan 99.

Gambar (b), frekuensi memang tidak berubah, tetapi terjadi penghalusan perubahan aras (128 menjadi 99 dan 40 menjadi 69).

Melalui filter lowpass, hal-hal yang menyatakan frekuensi tinggi akan diredupkan, sedangkan bagian berfrekuensi rendah hampir tidak berubah.



(a) Citra boneka yang dilengkapi derau



(b) Hasil penapisan citra boneka

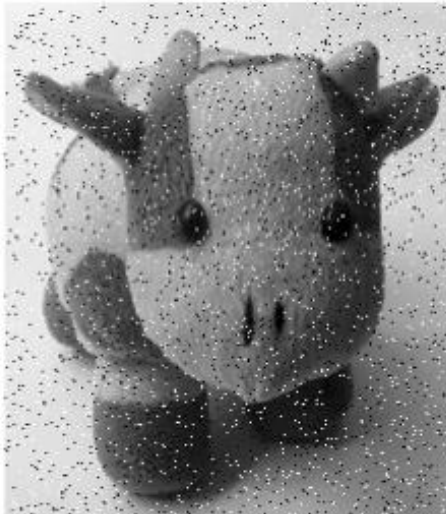


(c) Citra goldhill



(b) Hasil penapisan citra goldhill

filter lowpass mampu menghaluskan perubahan-perubahan yang drastis. Perhatikan ketajaman genting pada *goldhill* menjadi diperhalus setelah melalui penapisan. Begitu pula derau pada boneka.



(a) Citra boneka yang dilengkapi derau



(b) Hasil dengan kernel #1



(c) Hasil dengan kernel #2



(b) Hasil dengan kernel #4

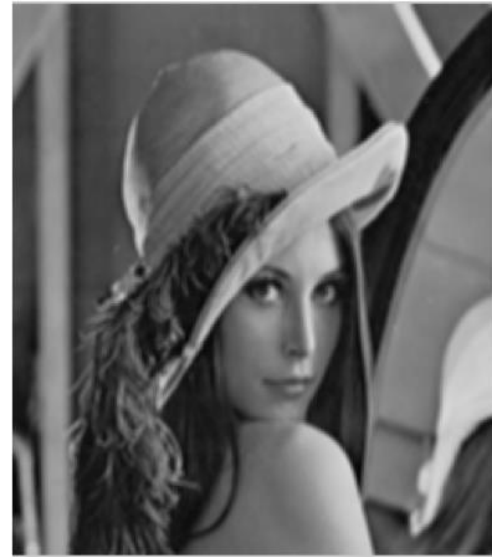
Hasil Low pass filter
dengan perbedaan kernel

- Low-pass filter merupakan filter yang digunakan untuk mengurangi komponen frekuensi tinggi pada citra, seperti noise dan perubahan intensitas yang tajam.
- Proses filtering dapat dilakukan melalui operasi konvolusi dengan menggunakan kernel tertentu.
- Kernel memberikan bobot pada piksel pusat dan piksel di sekitarnya.
- Setelah kernel digeser pada seluruh area citra, diperoleh citra baru dengan perubahan intensitas yang lebih halus.

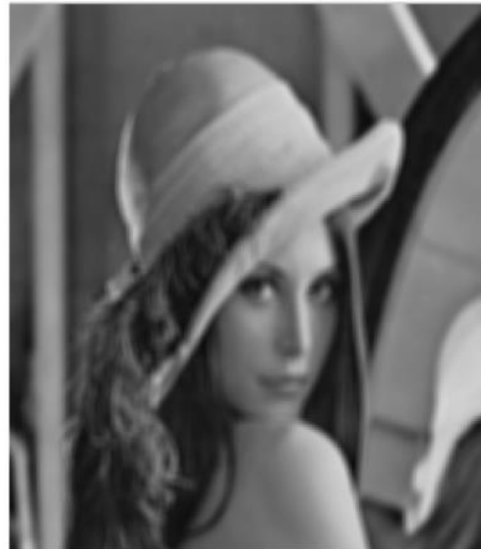
**Efek pemakaian
filter low-pass
dengan berbagai
ukuran kernel.**



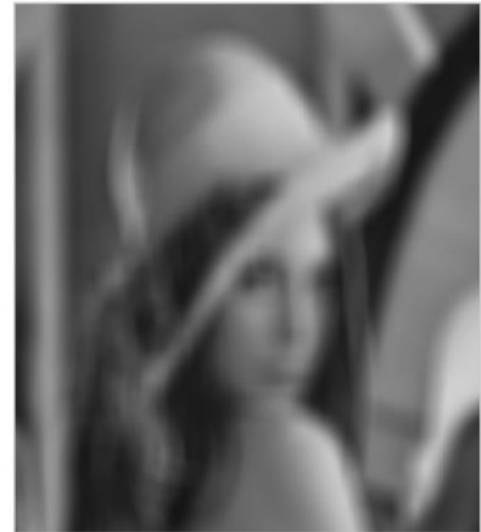
(a) Citra lena256



(b) Hasil dengan kernel 3x3



(c) Hasil dengan kernel 5x5



(b) Hasil dengan kernel 13x13

Kernel	Karakteristik	Bobot pusat	Efek
#1	Tetangga atas/bawah/kiri/kanan	2	Smoothing
#2	Semua piksel sama	1	Average/blur
#3	Pusat sedikit lebih penting	2	Smoothing dengan mempertahankan pusat
#4	Pusat paling dominan	4	Gaussian-like smoothing

Highpass filter

- Filter highpass adalah 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
- Filter lolos-tinggi mempunyai sifat yaitu jumlah seluruh koefisien adalah nol

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

#1

-1	-1	-1
-1	8	-1
-1	-1	-1

#2

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

#3

Contoh

40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
40	40	40	40	40
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128
128	128	128	128	128

(a) Citra dengan frekuensi rendah

128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40
128	128	128	128	128
40	40	40	40	40

(b) Citra dengan frekuensi tinggi

Jika kernel seperti berikut

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

Hasil

0	0	0	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
88	88	88	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
0	0	0	0	0	0
0	0	0	176	176	176
(a) Citra dengan frekuensi rendah			(b) Citra dengan frekuensi tinggi		

Gambar (a) menunjukkan bahwa hanya pada perbatasan antara perubahan aras keabuan yang ditonjolkan (baris berisi 88) dan nilai yang lain bernilai rendah (nol). Dengan demikian, akan muncul garis putih.

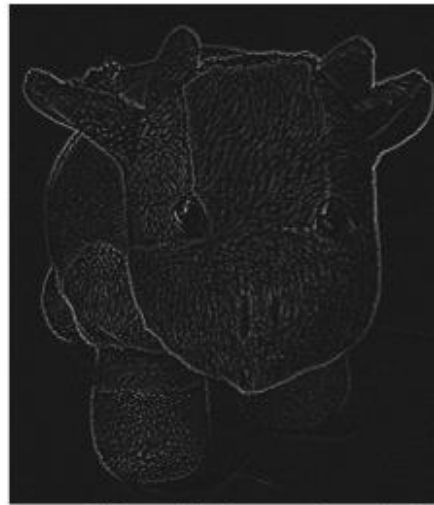
Gambar

(b) menunjukkan bahwa citra yang berfrekuensi tinggi hampir tidak mengalami perubahan, kecuali nilainya saja yang berefek pada penajaman perbedaan aras keabuan (nilai 150 menjadi 176 dan nilai 40 menjadi 0).

Hasil high pass
filter



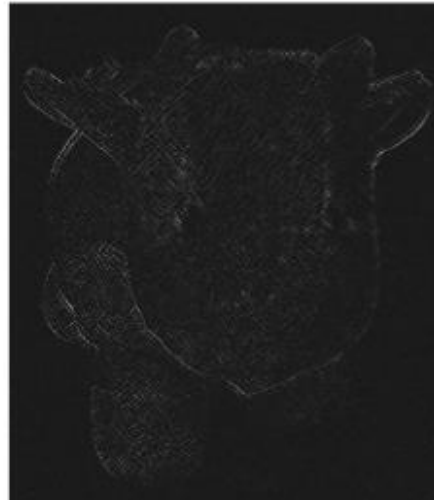
(a) Citra boneka.png



(b) Hasil dengan kernel #1

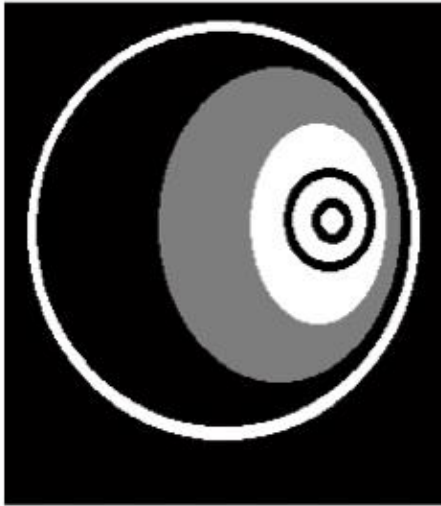


(c) Hasil dengan kernel #2

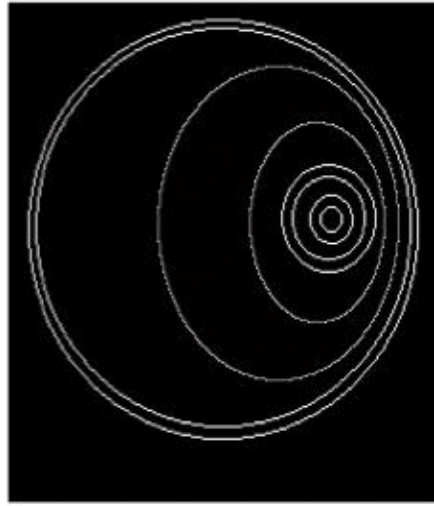


(b) Hasil dengan kernel #3

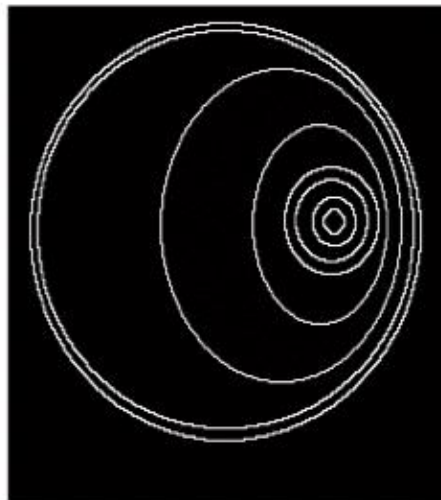
Hasil high pass
filter



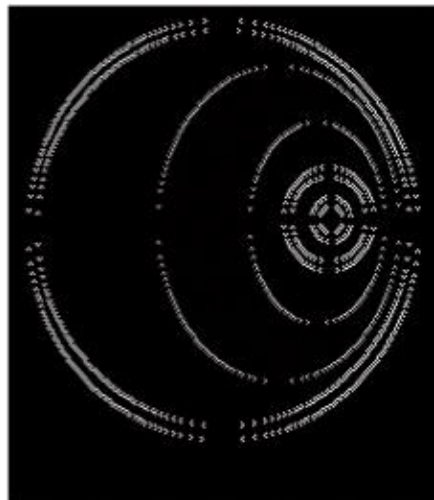
(a) Citra bulat.png



(b) Hasil dengan kernel #1



(c) Hasil dengan kernel #2



(b) Hasil dengan kernel #3

LATIHAN

Hitunglah nilai citra dibawah ini dengan

- teknik konvolusi
- Filter median
- Filter lowpass

$$f(x,y) = \begin{pmatrix} 4 & 4 & 3 & 5 & 4 \\ 6 & 6 & 5 & 5 & 2 \\ 5 & 6 & 6 & 6 & 2 \\ 6 & 7 & 5 & 5 & 3 \\ 3 & 5 & 2 & 4 & 4 \end{pmatrix}$$

Citra 5 x 5

$$g(x,y) = \begin{pmatrix} 0 & -1 & 0 \\ -2 & \bullet 3 & -2 \\ 0 & -1 & 0 \end{pmatrix}$$

Kernel 3 x 3

Tanda  menyatakan posisi (0,0) dari kernel