

OPERASI GEOMETRIK

PERTEMUAN KE 9

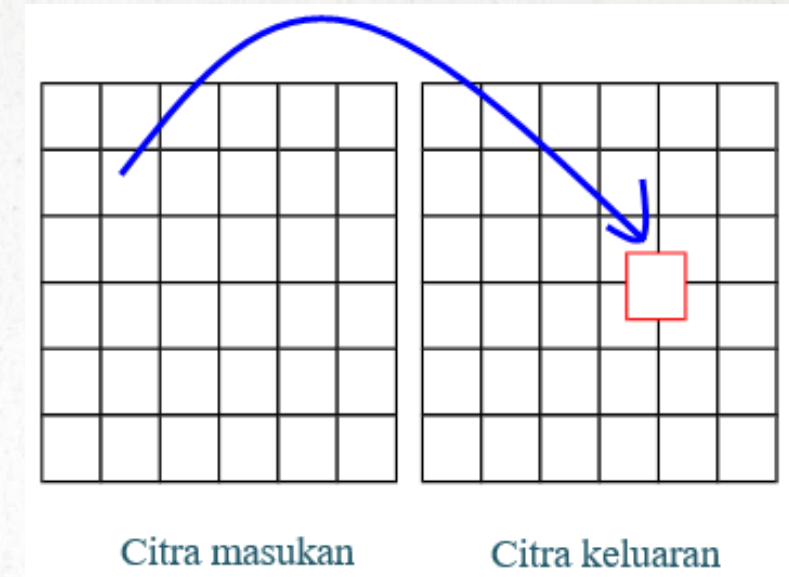
PENGERTIAN OPERASI GEOMETRIK PADA CITRA

Adalah operasi pada citra yang dilakukan secara geometris, seperti translasi, rotasi, dan penyekalan.

Secara prinsip terdapat dua cara yang dapat dipakai, yaitu pemetaan ke depan dan pemetaan ke belakang.

PEMETAAN GEOMETRIK KE DEPAN

Cara pemetaan kedepan, posisi pada citra keluaran ditentukan dengan acuan pemrosesan pada citra masukan

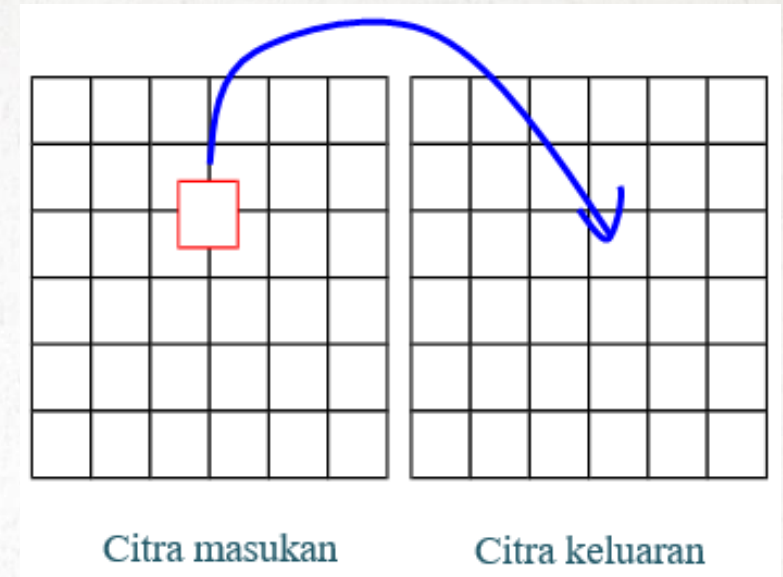


Gambar 8.1 Pemetaan ke depan

Gambar 8.1 di atas menjelaskan bahwa pada cara pemetaan ke depan, posisi pada citra keluaran ditentukan dengan acuan pemrosesan pada citra masukan. Pada gambar tersebut terlihat bahwa kalau piksel keluaran berada pada posisi yang tidak tepat (tidak berupa bilangan bulat), penempatannya dapat berada pada salah satu dari empat kemungkinan.

PEMETAAN GEOMETRIK KE BELAKANG

Pada pemetaan ke belakang, pemrosesan dimulai dari citra keluaran, maka semua piksel pada citra keluaran akan diberi nilai sekali saja berdasarkan piksel masukan.



Gambar 8.2 Pemetaan ke depan

Gambar 8.2 di atas menjelaskan bahwa piksel yang digunakan untuk menentukan piksel keluaran dapat ditentukan oleh salah satu piksel yang tercakup dalam kotak yang menggantung pada keempat piksel. Hal itu merupakan cara tersederhana yang dapat dilakukan dan biasa dinamakan sebagai pemilihan berdasarkan tetangga terdekat.

Penggeseran citra ke arah mendatar dan vertikal dapat dilakukan dengan mudah.

Rumus yang digunakan untuk menggeser citra:

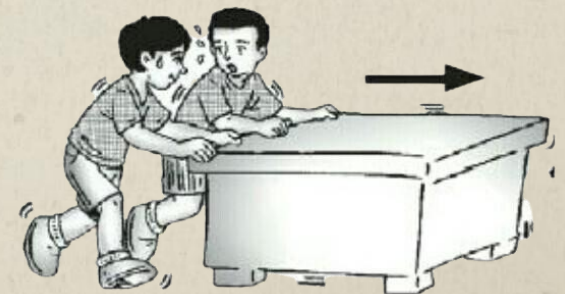
$$X_{\text{baru}} = X_{\text{lama}} + S_x \dots\dots\dots (8.1)$$

$$Y_{\text{baru}} = Y_{\text{lama}} + S_y \dots\dots\dots (8.2)$$

Dimana :

- S_x dan S_y dianggap bertipe bilangan bulat.

MENGGESER CITRA



**LISTING
PROGRAM
MENGGESER
CITRA:**

```
% GESER Melakukan operasi penggeseran citra.
F = imread('c:\Image\gedung.png');
[tinggi, lebar] = size(F);
sx = 45;                                % Penggeseran arah horizontal
sy = -35;                                % Penggeseran arah vertikal
F2 = double(F);
G = zeros(size(F2));
for y=1 : tinggi
    for x=1 : lebar
        xlama = x - sx;
        ylama = y - sy;

        if (xlama>=1) && (xlama<=lebar) && ...
            (ylama>=1) && (ylama<=tinggi)
            G(y, x) = F2(ylama, xlama);
        else
            G(y, x) = 0;                  % Jika Xlama atau Ylama berada pada diluar jangkauan, diisi nol
        end
    end
end
G = uint8(G);
figure(1); imshow(G);
clear all;
```


PENJELASAN PROGRAM

Pada contoh listing program, citra digeser ke kanan sebesar 45 piksel (ditentukan melalui S_x) dan ke atas sebesar 35 piksel (diatur melalui S_y).

Apabila X_{lama} hasil perhitungan di luar jangkauan $[1, lebar]$ atau Y_{lama} hasil perhitungan di luar jangkauan $[1, tinggi]$, intensitas piksel pada posisi (y, x) diisi dengan nol (warna hitam). Posisi yang tidak berada pada posisi koordinat yang valid dalam citra lama akan diisi dengan nilai nol melalui

$$G(y, x) = 0;$$

HASIL PENGGESERAN CITRA



(a) Citra gedung asli



(b) Hasil penggeseran

Gambar 8.3 Pergeseran pada citra

Suatu citra dapat diputar dengan sudut θ seiring arah jarum jam atau berlawanan arah jarum jam dengan pusat putaran pada koordinat (0,0).

Rumus yang digunakan untuk memutar citra

$$X_{\text{baru}} = x * \cos(\theta) + y * \sin(\theta) \dots\dots\dots (8.3)$$

$$y_{\text{baru}} = y * \cos(\theta) - x * \sin(\theta) \dots\dots\dots (8.4)$$

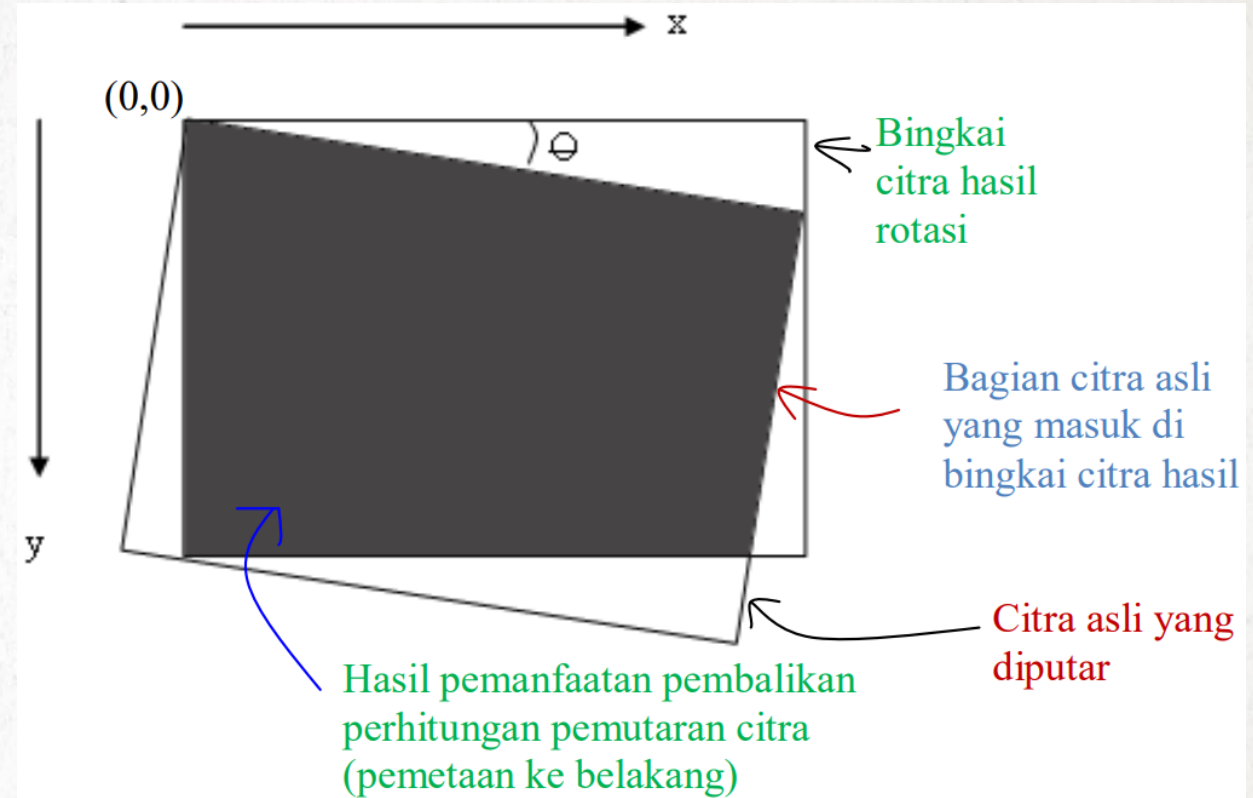
MEMUTAR CITRA



MEMUTAR CITRA

Apabila posisi koordinat (y_{baru} , x_{baru}) berada di luar area [1, lebar] dan [1, tinggi], intensitas yang digunakan berupa nol.

Cara inilah yang merupakan contoh pemetaan ke belakang.



Gambar 8.4 Pergeseran pada citra

**LISTING
PROGRAM
MEMUTAR
CITRA KE
BELAKANG:**

```
% ROTASI Melakukan Operasi pemutaran citra.  
% Menggunakan pendekatan pemetaan ke belakang  
F = imread('c:\Image\sungai.png');  
[tinggi, lebar] = size(F);  
sudut = 10; % Sudut pemutaran  
rad = pi * sudut/180; % dirubah dalam radian  
cosa = cos(rad);  
sina = sin(rad);  
F2 = double(F);  
for y=1 : tinggi  
    for x=1 : lebar  
        x2 = round(x * cosa + y * sina); % diputar searah jarum jam  
        y2 = round(y * cosa - x * sina);  
  
        if (x2>=1) && (x2<=lebar) && ... (y2>=1) && (y2<=tinggi)  
            G(y, x) = F2(y2, x2);  
        else  
            G(y, x) = 0;  
        end  
    end  
end  
G = uint8(G);  
figure(1); imshow(G);  
clear all;
```

PENJELASAN PROGRAM

Berdasarkan Persamaan 8.3 dan 8.4, pemutaran citra dengan sudut θ searah jarum jam dapat dilakukan. Pada saat menghitung dengan rumus di atas, apabila posisi koordinat (y_{baru} , x_{baru}) berada di luar area $[1, \text{lebar}]$ dan $[1, \text{tinggi}]$, intensitas yang digunakan berupa nol. Cara inilah yang merupakan contoh pemetaan ke belakang. Implementasinya seperti ditunjukkan pada Gambar 8.5.

HASIL PEMUTARAN CITRA



(a) Citra sungai asli



(b) Hasil pemutaran

Gambar 8.5 Pergeseran pada citra dengan pemetaan ke belakang

Hasil Pemutaran Citra Dengan Metode Pemetaan Ke Depan

Jika menggunakan rumus pada Persamaan 8.3 dan 8.4 dilaksanakan pemetaan ke depan? cara seperti itu dapat menimbulkan lubang pada citra hasil. Artinya, akan ada piksel yang tidak terisi dengan piksel dari citra masukan.

**LISTING
PROGRAM
MEMUTAR
CITRA KE
DEPAN:**

```
% ROTASI2 Melakukan operasi pemutaran citra.
% Menggunakan pemetaan ke depan
F = imread('c:\Image\gedung.png');
[tinggi, lebar] = size(F);
sudut = 5;
rad = pi * sudut/180;
cosa = cos(rad);
sina = sin(rad);
F2 = double(F);
G=zeros(tinggi, lebar);
for y=1 : tinggi
    for x=1 : lebar
        x2 = round(x * cosa - y * sina);
        y2 = round(y * cosa + x * sina);

        if (x2>=1) && (x2<=lebar) && ... (y2>=1) && (y2<=tinggi)
            G(y2, x2) = F2(y, x);
        end
    end
end
G = uint8(G);
figure(1); imshow(G);
clear all;
```

*% Sudut pemutaran
% dirubah dalam radian*

% diputar berlawanan arah jarum jam

HASIL PEMUTARAN CITRA

Hasil pemutaran yang menimbulkan lubang-lubang (bintik-bintik gelap) pada citra



(a) Citra gedung asli

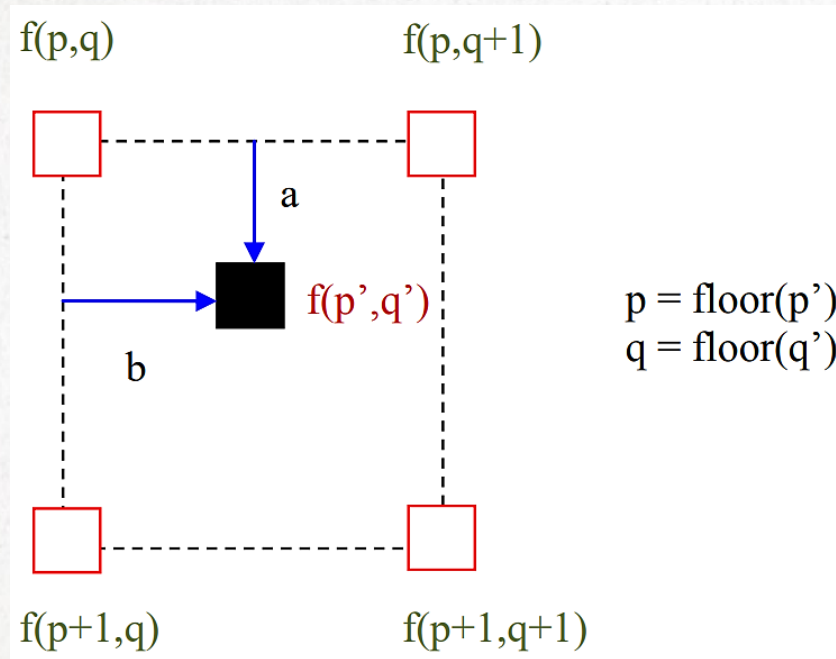
(b) Hasil pemutaran citra gedung

Gambar 8.6 Contoh pemutaran citra dengan pemetaan maju

PENJELASAN PROGRAM

Berdasarkan Persamaan 8.3 dan 8.4, pemutaran citra dengan sudut θ searah jarum jam dapat dilakukan. Pada saat menghitung dengan rumus di atas, apabila posisi koordinat (y_{baru} , x_{baru}) berada di luar area $[1, lebar]$ dan $[1, tinggi]$, intensitas yang digunakan berupa nol. Cara inilah yang merupakan contoh pemetaan ke belakang. Implementasinya seperti ditunjukkan pada Gambar 8.5.

Penggunaan piksel terdekat dengan menggunakan teknik fungsi **round** (pembulatan ke atas) atau **floor** (pembulatan kebawah), guna menghilangkan efek bergerigi pada obyek citra.



Gambar 8.7 Model pendekatan bilinear interpolation

Perhatikan bahwa $f(p', q')$ mempunyai empat piksel terdekat berupa $f(p,q)$, $f(p,q+1)$, $f(p+1,q)$, dan $f(p+1,q+1)$.

INTERPOLASI PIKSEL



Pratt (2001) menunjukkan cara menghitung nilai intensitas yang digunakan untuk suatu piksel berdasarkan empat piksel. Rumusnya sebagai berikut:

$$f(p', q') = (1 - a)[(1 - b)f(p, q) + b f(p, q + 1)] + a[(1 - b)f(p + 1, q) + b f(p + 1, q + 1)] \dots\dots\dots (8.5)$$

Dalam hal ini a dan b dihitung melalui:

- $a = p' - p \dots\dots\dots (8.6)$

- $b = q' - q \dots\dots\dots (8.7)$

Rumus dalam Persamaan 8.5 itulah yang disebut sebagai bilinear interpolation

INTERPOLASI PIKSEL



% ROTASI3 Melakukan operasi pemutaran citra.
% Versi 3 - menggunakan bilinear interpolation

```
F = imread('c:\Image\gedung.png');  
[tinggi, lebar] = size(F);  
sudut = 15;                                % Sudut pemutaran  
rad = pi * sudut/180;                       % dirubah dalam radian  
cosa = cos(rad);  
sina = sin(rad);  
F2 = double(F);  
for y=1 : tinggi  
    for x=1 : lebar  
        x2 = x * cosa + y * sina;  
        y2 = y * cosa - x * sina;  
        if (x2>=1) && (x2<=lebar) && ... (y2>=1) && (y2<=tinggi)  
            % Lakukan interpolasi bilinear  
            p = floor(y2);  
            q = floor(x2);  
            a = y2-p;  
            b = x2-q;
```

**LISTING
PROGRAM
INTERPOLASI
CITRA :**


```

if (x2 == lebar) || (y2 == tinggi)
    G(y, x) = F(floor(y2), floor(x2));
else
    intensitas = (1-a)*((1-b)*F(p,q) + ...
        b * F(p, q+1)) + ...
        a *((1-b)* F(p+1, q) + ...
        b * F(p+1, q+1));
    G(y, x) = intensitas;
end
else
    G(y, x) = 0;
end
end

end
G = uint8(G);
figure(1); imshow(G);
clear all;

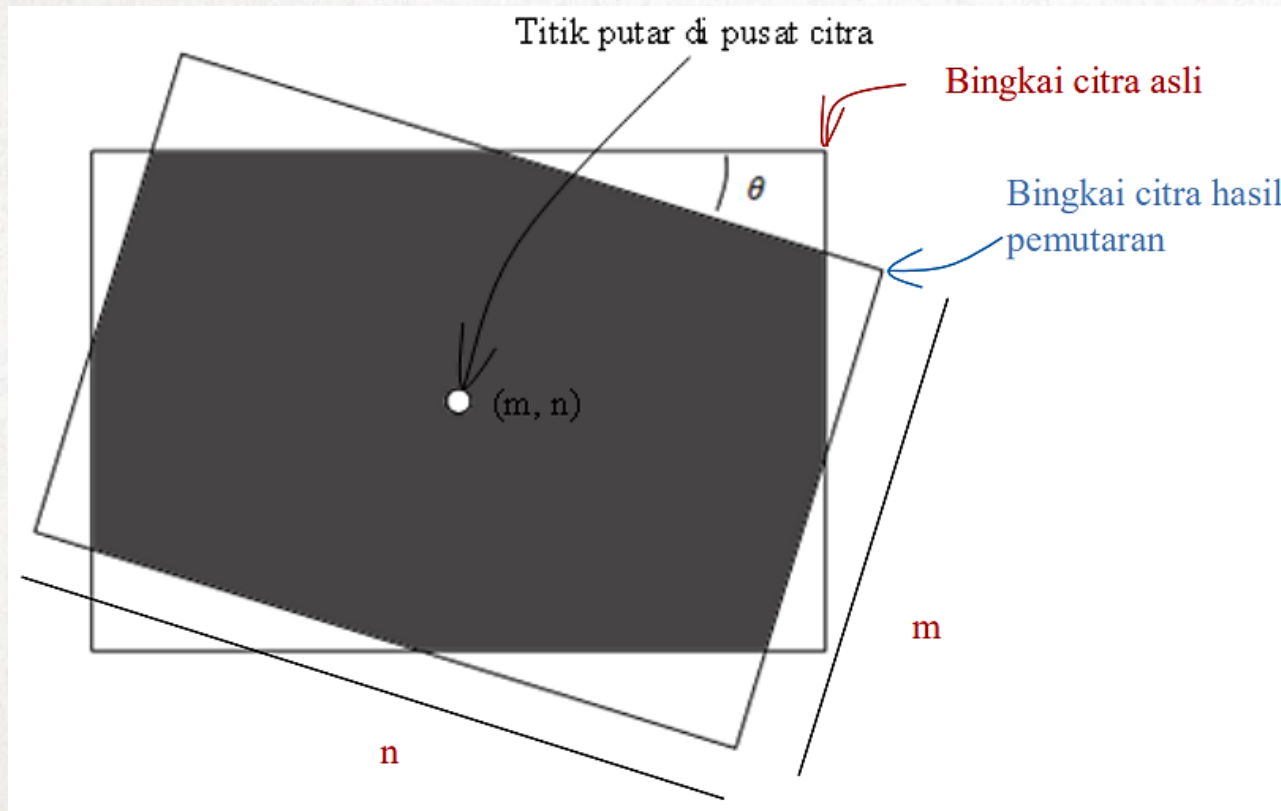
```

CONTOH HASIL INTERPOLASI



Gambar 8.8 Perbandingan efek interpolasi billinier

Pemutaran citra dapat dilakukan dengan pusat dimana saja, tidak harus dari (1,1). Seperti ditunjukkan pada Gambar 8.9.



Gambar 8.9 Pemutaran citra melalui titik pusat citra

MEMUTAR BERDASARKAN SEMBARANG KOODINAT



RUMUS MEMUTAR BERDASARKAN SEMBARANG KOODINAT

Rumus untuk melakukan pemutaran berlawanan arah jarum jam, sebesar θ yang diperlihatkan di Gambar 8.9, diperoleh dari modifikasi persamaan 8.3 dan 8.4.

$$x_{\text{baru}} = (x-n) * \cos (\theta) + (y-m) * \sin (\theta) + n \dots\dots\dots (8.8)$$

$$y_{\text{baru}} = (y-n) * \cos (\theta) + (x-n) * \sin (\theta) + m \dots\dots\dots (8.9)$$

Untuk kepentingan pemutaran citra sejauh θ searah jarum jam, intensitas piksel (x,y) , dapat diperoleh melalui intensitas pada piksel $(y_{\text{baru}}, x_{\text{baru}})$, yang tertera pada persamaan 8.8 dan 8.9.

**LISTING
PROGRAM
MEMUTAR
CITRA
SEMBARANG
KOORDINAT :**

```
% Versi 4 - pusat putaran pada pusat citra
F = imread('c:\Image\gedung.png');
[tinggi, lebar] = size(F);
sudut = 5; % Sudut pemutaran
rad = pi * sudut/180;
cosa = cos(rad);
sina = sin(rad);
F2 = double(F);
m = floor(tinggi / 2);
n = floor(lebar / 2);
for y=1 : tinggi
    for x=1 : lebar
        x2 = (x-n) * cosa + (y-m) * sina + n;
        y2 = (y-m) * cosa - (x-n) * sina + m;
        if (x2>=1) && (x2<=lebar) && ... (y2>=1) &&
(y2<=tinggi)
            % Lakukan interpolasi bilinear
            p = floor(y2);
            q = floor(x2);
            a = y2-p;
            b = x2-q;
```

**LISTING
PROGRAM
MEMUTAR
CITRA
SEMBARANG
KOORDINAT :**

```

if (x2==lebar) || (y2 == tinggi)
    G(y, x) = F(y2, x2);
else
    intensitas = (1-a)*((1-bF(p,q) + ...
        b * F(p, q+1)) + ...
        a *((1-b)* F(p+1, q) + ...
        b * F(p+1, q+1));
    G(y, x) = intensitas;
end
else
    G(y, x) = 0;
end

end
end
G = uint8(G);
figure(1); imshow(G);
clear all;

```


CONTOH HASIL MEMUTAR BERDASARKAN SEMBARANG KOODINAT



(a) Citra gedung asli

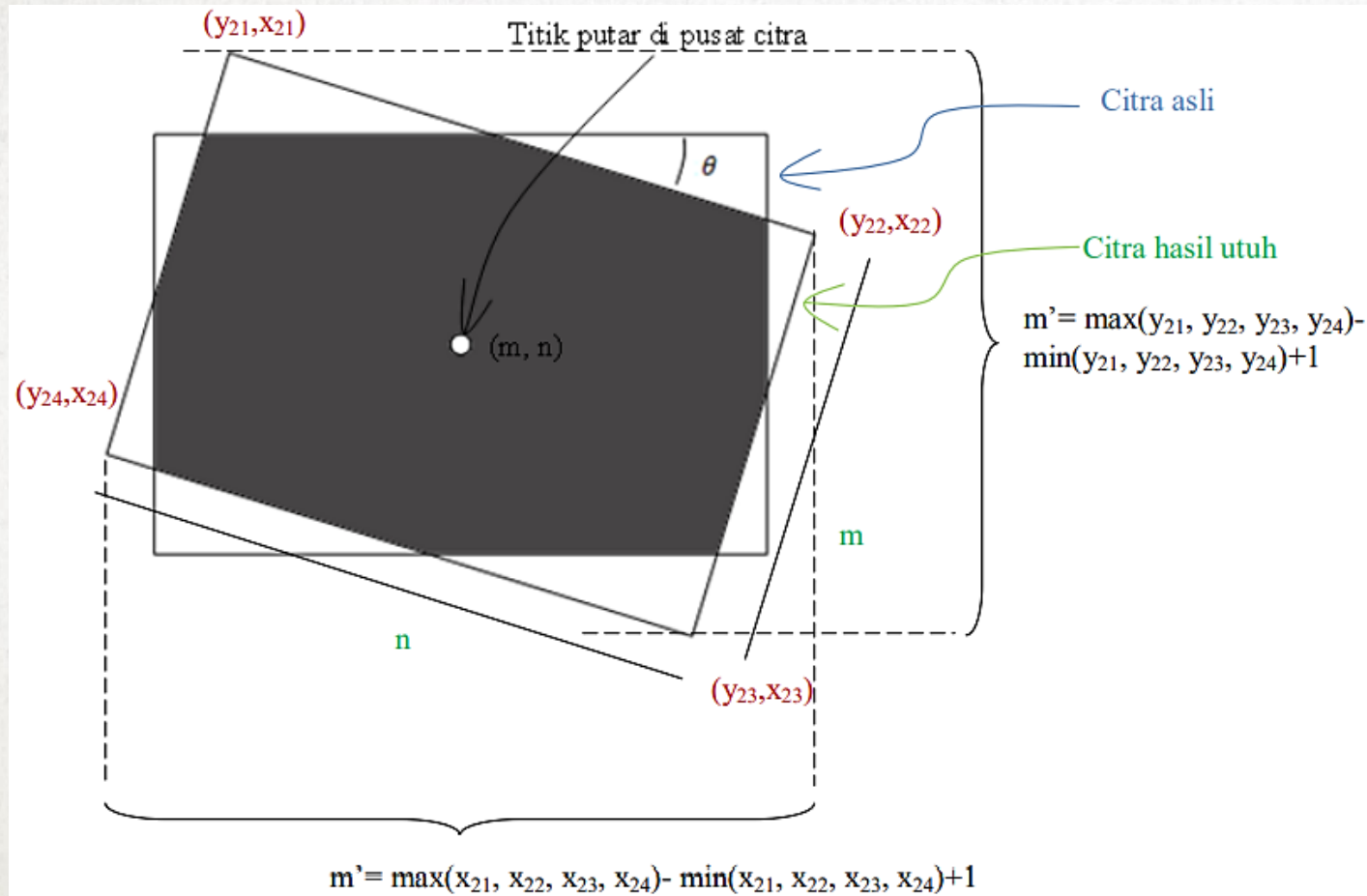


(b) Hasil pemutaran 5°

Gambar 8.10 Pemutaran melalui titik pusat citra

Pratt (2001) menunjukkan cara menghitung nilai intensitas yang digunakan untuk suatu piksel berdasarkan empat piksel.

Rumusnya sebagai berikut:



Gambar 8.11 Penentuan lebar dan tinggi citra hasil pemutaran

MEMUTAR CITRA SECARA UTUH



**LISTING
PROGRAM
MEMUTAR
CITRA
SECARA
UTUH:**

```
% ROTASI5 Melakukan operasi pemutaran citra.  
% Memutar dengan hasil utuh
```

```
F = imread('c:\Image\gedung.png');  
[tinggi, lebar] = size(F);  
sudut = 45; % Sudut pemutaran  
rad = pi * sudut/180;  
cosa = cos(rad);  
sina = sin(rad);  
x11 = 1; y11 = 1;  
x12 = lebar; y12 = 1;  
x13 = lebar; y13 = tinggi;  
x14 = 1; y14 = tinggi;  
m = floor(tinggi/2);  
n = floor(lebar/2);
```

LISTING

PROGRAM

MEMUTAR

CITRA

SECARA

UTUH:

```
% Menentukan pojok
x21 = ((x11-n) * cosa + (y11-m) * sina + n);
y21 = ((y11-m) * cosa - (x11-n) * sina + m);
x22 = ((x12-n) * cosa + (y12-m) * sina + n);
y22 = ((y12-m) * cosa - (x12-n) * sina + m);
x23 = ((x13-n) * cosa + (y13-m) * sina + n);
y23 = ((y13-m) * cosa - (x13-n) * sina + m);
x24 = ((x14-n) * cosa + (y14-m) * sina + n);
y24 = ((y14-m) * cosa - (x14-n) * sina + m);
ymin = min([y21 y22 y23 y24]);
xmin = min([x21 x22 x23 x24]);
ymak = max([y21 y22 y23 y24]);
xmak = max([x21 x22 x23 x24]);
lebar_baru = xmak - xmin + 1;
tinggi_baru = ymak - ymin + 1;
tambahan_y = floor((tinggi_baru-tinggi)/2);
tambahan_x = floor((lebar_baru-lebar)/2);
F2=zeros(tinggi_baru, lebar_baru);
    for y=1 : tinggi
        for x=1 : lebar
            F2(y+tambahan_y, x+tambahan_x) = F(y, x);
        end
    end
figure(1);
imshow( uint8(F2));
```


LISTING

PROGRAM

MEMUTAR

CITRA

SECARA

UTUH:

```
% Putar citra

m = floor(tinggi_baru/2);
n = floor(lebar_baru/2);
for y=1 : tinggi_baru
    for x=1 : lebar_baru
        x2 = round((x-n) * cosa + (y-m) * sina + n);
        y2 = round((y-m) * cosa - (x-n) * sina + m);
        if (x2>=1) && (x2<=lebar_baru) && ... (y2>=1) &&
(y2<=tinggi_baru)
            G(y, x) = F2(y2,x2);
        else
            G(y,x) = 0;
        end
    end
end
figure(2);
G = uint8(G);
imshow(G);
clear all;
```

CONTOH HASIL MEMUTAR CITRA SECARA UTOH



(a) Pengubahan ukuran citra
Supaya kalau diputar tidak
ada yang hilang



(b) Hasil pemutaran citra
dengan sudut 45°

Gambar 8.12 Pemutaran citra secara utuh

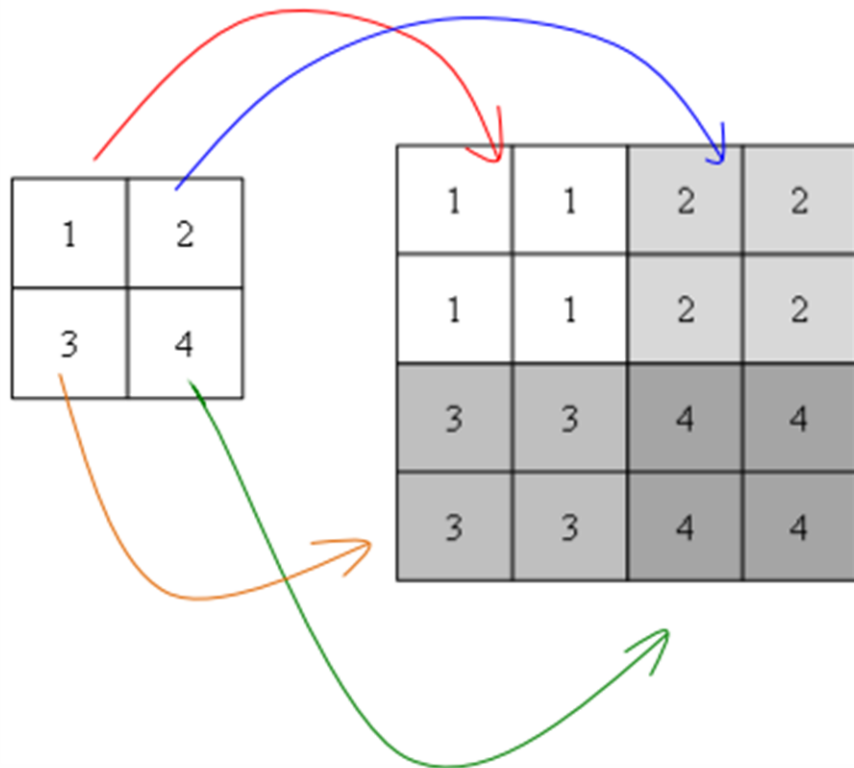


TERIMAKASIH

OPERASI GEOMETRIK

PERTEMUAN KE 10

Suatu citra dapat diperbesar dengan membuat setiap piksel menjadi beberapa piksel. Seperti ditunjukkan pada Gambar 8.13, memberikan contoh cara memperbesar citra.



Gambar 8.13 Cara memperbesar citra

MEMPERBESAR CITRA



LISTING PROGRAM MEMPERBESAR CITRA:

```
function G = perbesar(berkas, sy, sx)
% PERBESAR Melakukan operasi pembesaran citra.
% Masukan: berkas = nama berkas image
% sy : skala pembesaran pada sumbu Y
% sx : skala pembesaran pada sumbu X
%
% Versi 1

F = imread(berkas);
[tinggi, lebar] = size(F);
tinggi_baru = tinggi * sy;
lebar_baru = lebar * sx;
F2 = double(F);
for y=1 : tinggi_baru
    y2 = ((y-1) / sy) + 1;
    for x=1 : lebar_baru
        x2 = ((x-1) / sx) + 1;
        G(y, x) = F(floor(y2), floor(x2));
    end
end
G = uint8(G);    % untuk mengatur nilai piksel agar berada
                  pada jangkauan [0 -255]
```


LISTING PROGRAM MEMPERBESAR CITRA :

Berdasar fungsi memperbesar citra di atas, dapat diberikan perintah seperti berikut:

```
>> Img = perbesar('C:\Image\lena128.png', 3, 3);
```

Selanjutnya, hasil perbesaran ditampilkan melalui:

```
>> imshow(Img);
```

CONTOH HASIL MEMPERBESAR GAMBAR



(a) Citra lena 128x128



(b) Pembesaran 3x tanpa interpolasi

Gambar 8.14 Pembesaran citra 3 kali tanpa interpolasi

CONTOH HASIL MEMPERBESAR GAMBAR DENGAN DIBERIKAN PERINTAH INTERPOLASI



(a) Citra lena 128x128



(b) Pembesaran 3x

Gambar 8.15 Pembesaran citra 3 kali dengan interpolasi

LISTING PROGRAM MEMPERBESAR :

```
function G = perbesar2(berkas, sy, sx)
% PERBESAR2 Melakukan operasi pembesaran citra
% dengan interpolasi.
% Masukan: berkas = nama berkas image
% sy : skala pembesaran pada sumbu Y
% sx : skala pembesaran pada sumbu X

F = imread(berkas);
[tinggi, lebar] = size(F);
tinggi_baru = round(tinggi * sy);
lebar_baru = round(lebar * sx);
F2 = double(F);
for y=1 : tinggi_baru
    y2 = (y-1) / sy + 1;
    for x=1 : lebar_baru
        x2 = (x-1) / sx + 1;
```


LISTING PROGRAM MEMPERBESAR :

```

% Lakukan interpolasi bilinear
    p = floor(y2);
    q = floor(x2);
    a = y2-p;
    b = x2-q;
    if (floor(x2)==lebar) || ...
        (floor(y2) == tinggi)
        G(y, x) = F(floor(y2), floor(x2));
    else
        intensitas = (1-a)*((1-b)*F(p,q) + ...
            b * F(p, q+1)) + ...
            a *((1-b)* F(p+1, q) + ...
            b * F(p+1, q+1));
        G(y, x) = intensitas;
    end
end
end
G = uint8(G);

```

LISTING PROGRAM MEMPERBESAR DENGAN INTERPOLASI:

Berdasar fungsi memperbesar citra dengan interpolasi di atas, dapat diberikan perintah seperti berikut:

```
>> Img = perbesar2('C:\Image\lena128.png', 4, 4);
```

Selanjutnya, hasil perbesaran citra dengan interpolasi ditampilkan melalui:

```
>> imshow(Img);
```


Pengecilan citra berarti mengurangi jumlah piksel. Algoritma pembesaran tanpa interpolasi dan pembesaran dengan interpolasi, dapat digunakan untuk kepentingan memperkecil citra. Adapun cara yang digunakan adalah dengan memberikan s_y dan s_x bilangan pecahan, seperti $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, dan seterusnya.

MEMPERKECIL CITRA



LISTING PROGRAM MEMPERKECIL DENGAN INTERPOLASI:

Berdasar fungsi memperkecil citra di atas, dapat diberikan perintah seperti berikut:

```
>> Img = perbesar2('C:\Image\lena256.png', 0.5, 0.5);
```

Selanjutnya, hasil memperkecil citra, ditampilkan melalui:

```
>> imshow(Img);
```


CONTOH HASIL MEMPERKECIL CITRA



(a) Citra lena 256x256



(b) Hasil pengecilan 0,5 x
pada arah vertikal dan horisontal

Gambar 8.16 Pengecilan citra lena256.png dengan interpolasi

Algoritma pembesaran tanpa interpolasi dan pembesaran dengan interpolasi, juga dapat digunakan untuk memperbesarkan / memperkecil citra, dengan skala horizontal dan vertikal yang berbeda. Adapun cara yang digunakan adalah dengan memberikan **Sy** dan **Sx** bilangan bilangan yang berbeda.

MEMPERBESAR CITRA DENGAN SKALA VERTIKAL DAN HORIZONTAL BERBEDA



LISTING PROGRAM MEMPERBESAR CITRA DENGAN SKALA VERTIKAL DAN HORIZONTAL BERBEDA:

Berdasar fungsi perbesar di atas, dapat diberikan perintah seperti berikut:

```
>> Img = perbesar2('C:\Image\gedung.png', 0.5, 2.5);
```

Selanjutnya, hasil perbesaran ditampilkan melalui:

```
>> imshow(Img);
```

CONTOH HASIL MEMPERBESAR CITRA DENGAN SKALA VERTIKAL DAN HORIZONTAL BERBEDA



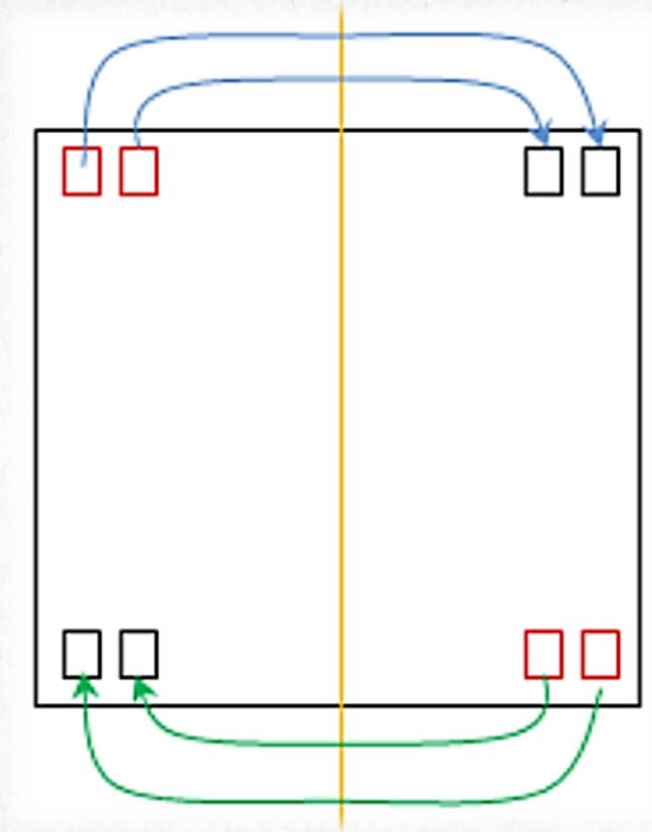
Gambar 8.17 Gedung di perkecil $\frac{1}{2}$ kali pada arah vertikal dan 2,5 kali pada arah horizontal

Pada proses pencerminan yang umum dilakukan adalah pencerminan secara horisontal dan pencerminan secara vertikal.

Pencerminan secara horisontal dilakukan dengan menukar dua piksel yang berseberangan kiri – kanan. Pencerminan secara horisontal ditunjukkan pada Gambar 8.17.



PENCERMINAN SECARA HORIZONTAL



Gambar 8.18 Pencerminan secara horizontal

ALGORITMA PENCERMINKAN GAMBAR SECARA HORIZONTAL

Masukan:

- $f(M, N)$: Citra masukan berukuran M baris dan N kolom

Keluaran:

- $g(M, N)$: Hasil citra yang telah dicerminkan secara horizontal

1. FOR baris \leftarrow 1 TO M
2. FOR kolom \leftarrow 1 TO N
3. $g(\text{baris}, \text{kolom}) \leftarrow f(N - \text{baris} + 1, \text{kolom})$
4. END-FOR
5. END-FOR

LISTING PROGRAM PECERMINAN CITRA SECARA HORISONTAL:

```
function G = cerminh(F)
% CERMINH Berfungsi untuk mencerminkan citra
% secara horizontal
% Masukan: F = Citra berskala keabuan
[tinggi, lebar] = size(F);
for y=1 : tinggi
    for x=1 : lebar
        x2 = lebar - x + 1;
        y2 = y;
        G(y, x) = F(y2, x2);
    end
end
G = uint8(G);    % untuk mengatur nilai piksel agar berada
                  pada jangkauan [0 -255]
```


LISTING PROGRAM PECERMINAN SECARA HORISONTAL CITRA:

Berdasar fungsi pencerminan secara horisontal di atas, dapat diberikan perintah seperti berikut:

```
>> F = imread('C:\Image\boneka.png');
```

```
>> G = cerminh(F);
```

Selanjutnya, hasil pencerminan horisontal citra ditampilkan melalui:

```
>> imshow(Img);
```

CONTOH HASIL PENCERMINAN CITRA SECARA HORIZONTAL



(a) Citra boneka.tif



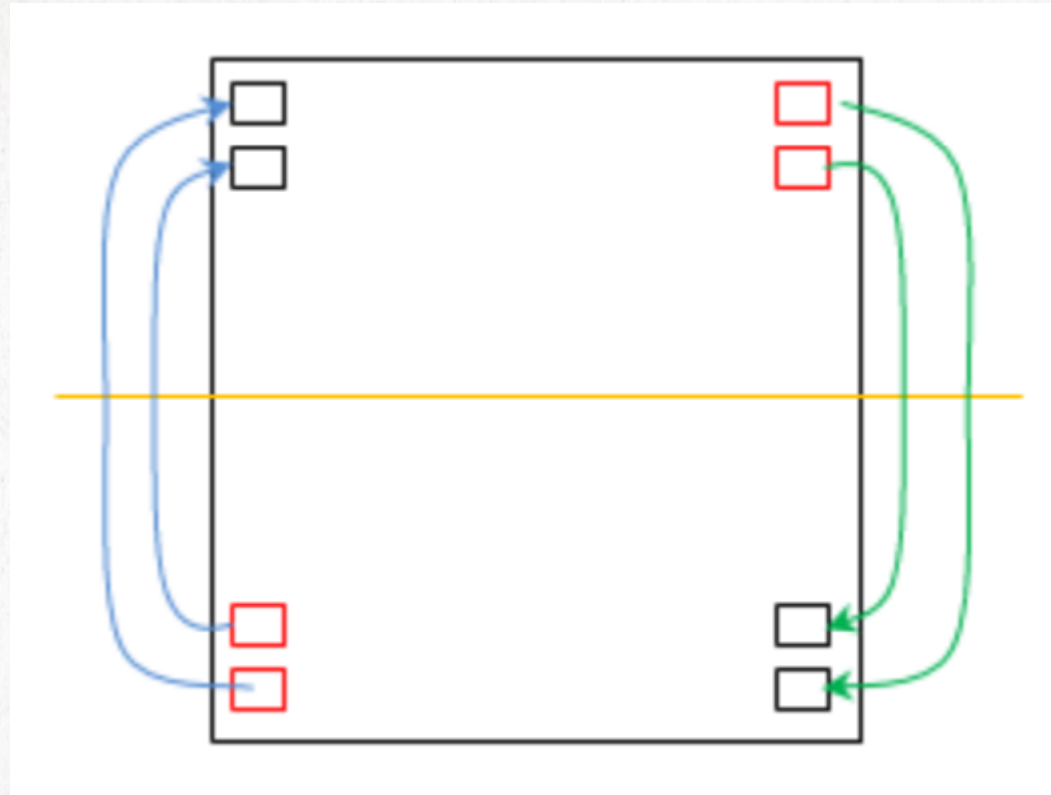
(b) Pencermian secara horizontal

Gambar 8.19 Contoh hasil pencerminan secara horizontal

Pencerminan secara vertikal dilakukan dengan menukar dua piksel yang berseberangan atas – bawah. Pencerminan secara horisontal ditunjukkan pada Gambar 8.20.



PENCERMINAN SECARA VERTIKAL



Gambar 8.20 Pencerminan secara vertikal

ALGORITMA PENCERMINKAN GAMBAR SECARA HORIZONTAL

Masukan:

- $f(M, N)$: Citra masukan berukuran M baris dan N kolom

Keluaran:

- $g(M, N)$: Hasil citra yang telah dicerminkan secara vertikal

1. FOR baris \leftarrow 1 TO M
2. FOR kolom \leftarrow 1 TO N
3. $g(\text{baris}, \text{kolom}) \leftarrow f(\text{baris}, N - \text{kolom} + 1)$
4. END-FOR
5. END-FOR

LISTING PROGRAM PECERMINAN CITRA SECARA VERTIKAL:

```
function G = cerminv(F)
% CERMINV Berfungsi untuk mencerminkan citra
% secara vertikal
% Masukan: F = Citra berskala keabuan
[tinggi, lebar] = size(F);
for y=1 : tinggi
    for x=1 : lebar
        x2 = x;
        y2 = tinggi - y + 1;
        G(y, x) = F(y2, x2);
    end
end
G = uint8(G); % untuk mengatur nilai piksel agar berada
              pada jangkauan [0 -255]
```


LISTING PROGRAM PECERMINAN SECARA HORIZONTAL CITRA:

Berdasar fungsi pencerminan secara vertikal di atas, dapat diberikan perintah seperti berikut:

```
>> F = imread('C:\Image\boneka.png');  
>> G = cerminv(F);
```

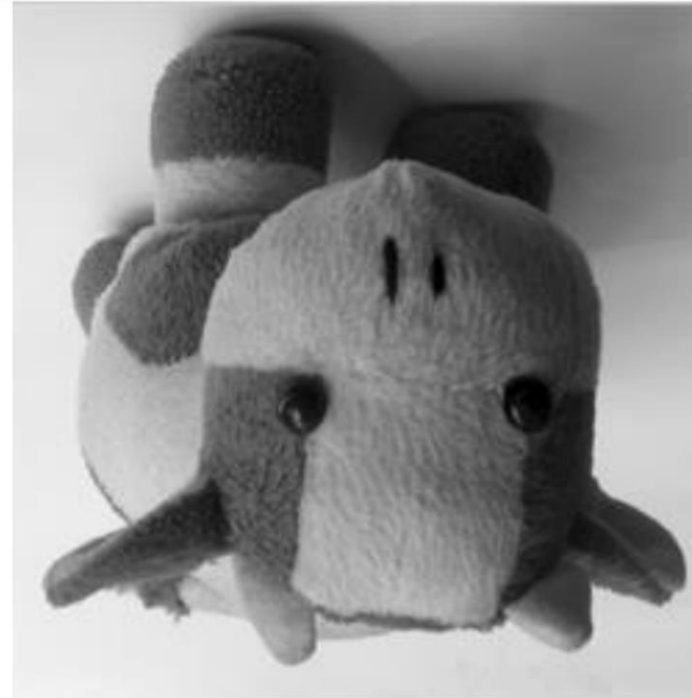
Selanjutnya, hasil pencerminan vertikal citra ditampilkan melalui:

```
>> imshow(Img);
```

CONTOH HASIL PENCERMINAN CITRA SECARA VERTIKAL



(a) Citra boneka.tif



(b) Pencermianan secara vertikal

Gambar 8.19 Contoh hasil pencerminan secara vertikal



Terimakasih