OPERASI GEOMETRIK

PERTEMUAN KE 8

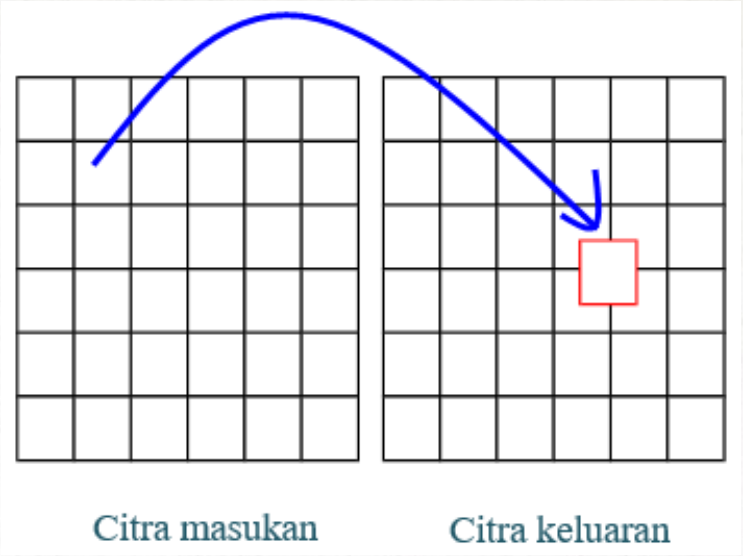
PENGERTIAN OPERASI GEOMETRIK PADA CITRA

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

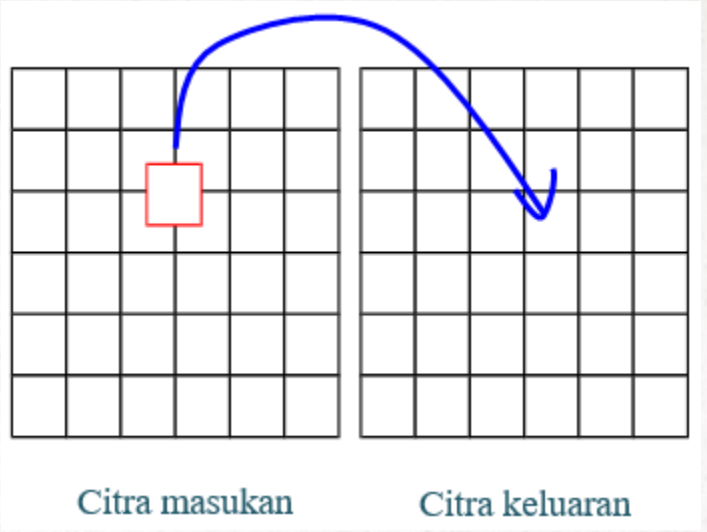


Gambar 8.1 Pemetaan ke depan

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.

MENGGESER CITRA

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

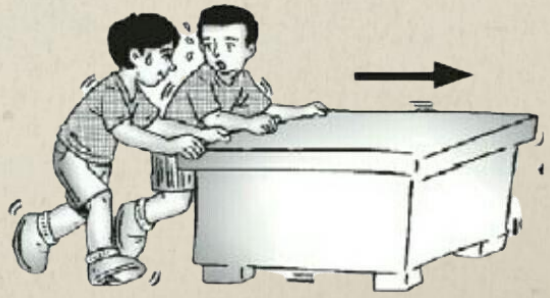
Rumus yang digunakan untuk menggeser citra

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

$$Y_{baru} = Y_{lama} + S_x \quad \text{..... (8.2)}$$

Dimana :

- sx dan sy dianggap bertipe bilangan bulat.



HASIL PENGGESERAN CITRA

Gambar 8.3 Pergeseran pada citra

MEMUTAR 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

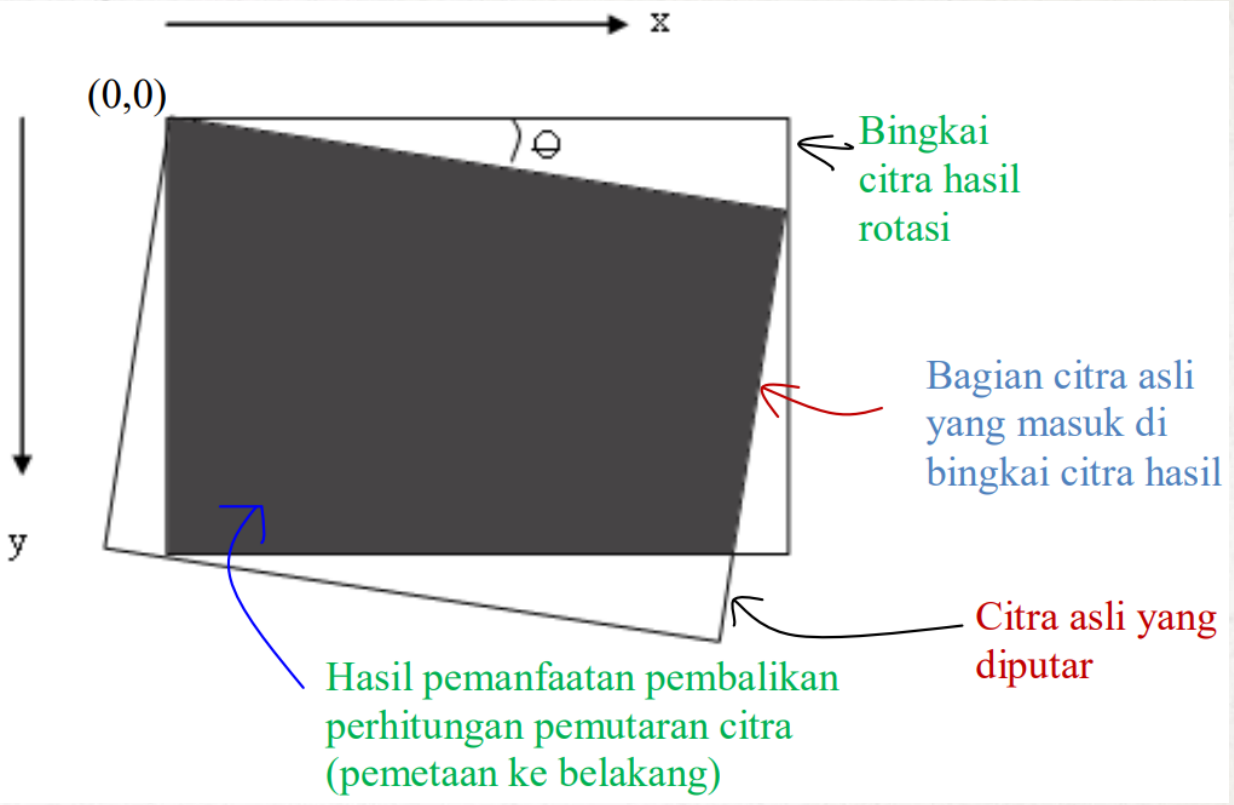
$$x_{baru} = x * Cos(\Theta) + y * Sin(\Theta) \quad \text{..... (8.3)}$$

$$y_{baru} = y * Cos(\Theta) - x * Sin(\Theta) \quad \text{..... (8.4)}$$

MEMUTAR CITRA

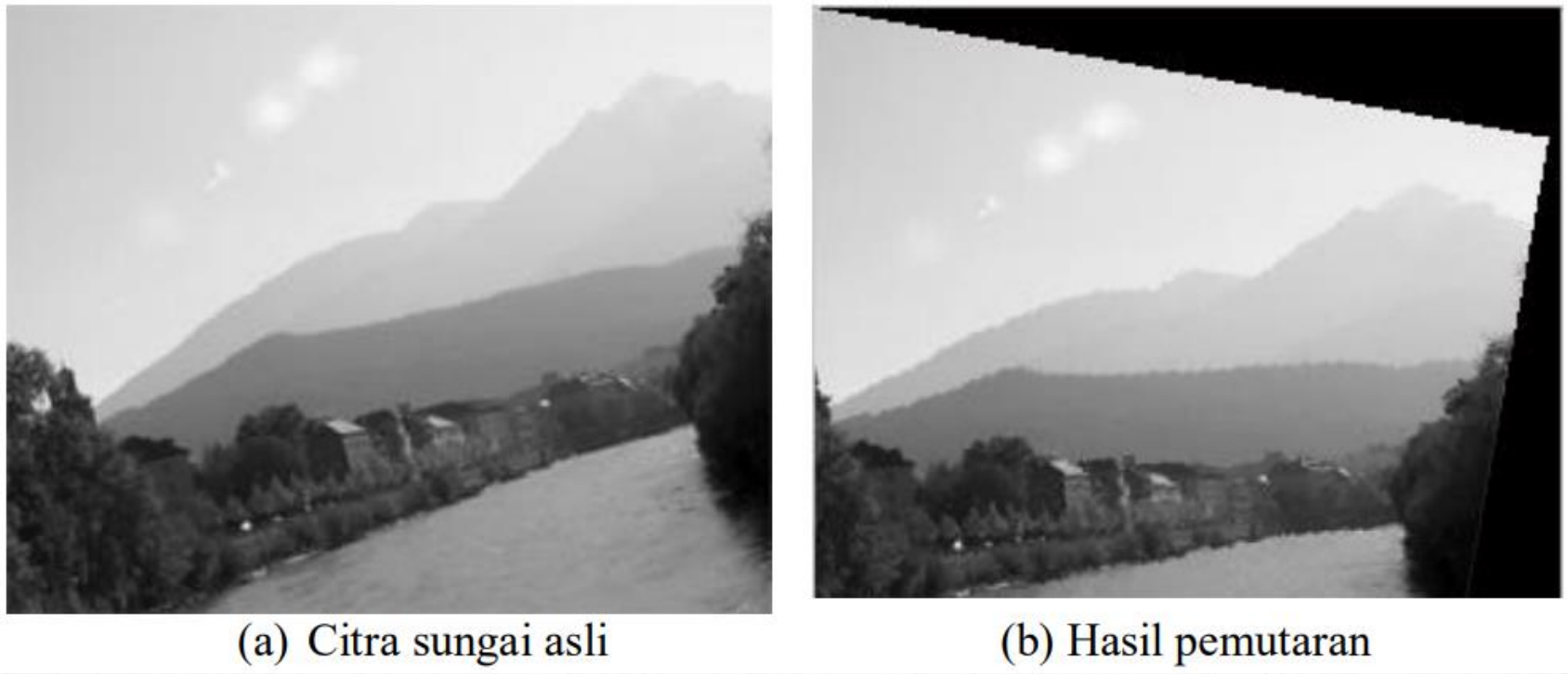
Apabila posisi koordinat (y_{baru} ,x_{baru}) berada di luar area [1, lebar] dan [1, tinggi], intensitas yang digunakan berupa nol.

Cara inilah yang merupakan contoh pemetaan ke belakang.



Gambar 8.4 Pergeseran pada citra

HASIL PEMUTARAN CITRA



(a) Citra sungai asli (b) Hasil pemutaran

Gambar 8.5 Pergeseran pada citra dengan pemetaan ke belakang

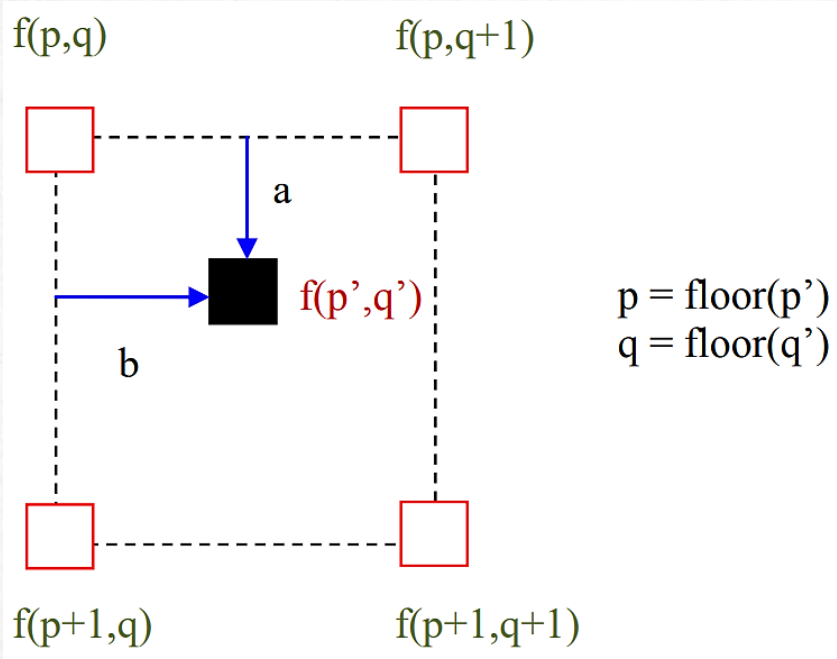
HASIL PEMUTARAN CITRA

(a) Citra gedung asli

(b) Hasil pemutaran citra gedung

Gambar 8.6 Contoh pemutaran citra dengan pemetaan maju

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



Gambar 8.7 Model pendekatan billinier interpollation

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

INTERPOLASI PIKSEL

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

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

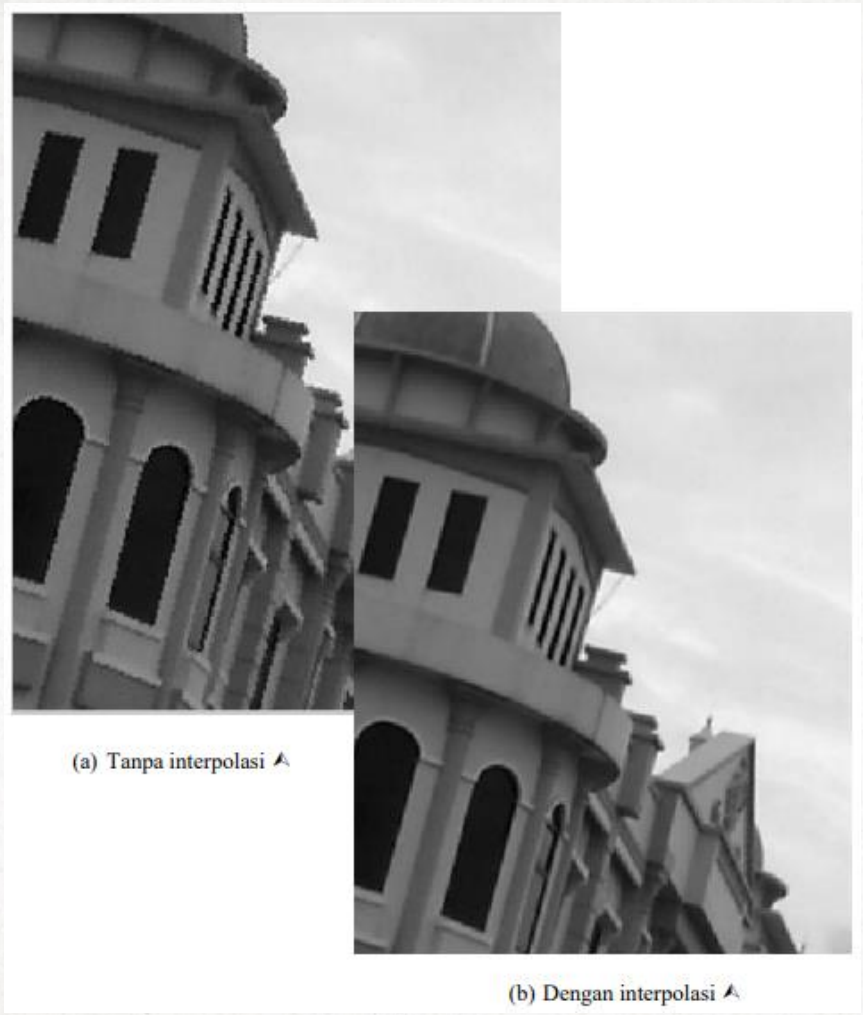
$$a[(1-b)f(p+1,q) + b\,f(p+1,q+1)] \quad \text{..... (8.5)}$$

Dalam hal ini a dan b dihitung melalui:

- $a = p' - p$ .....(8.6)
- $b = q' - q$ .....(8.7)

Rumus dalam Persamaan 8.5 itulah yang disebut sebagai bilinear interpolation

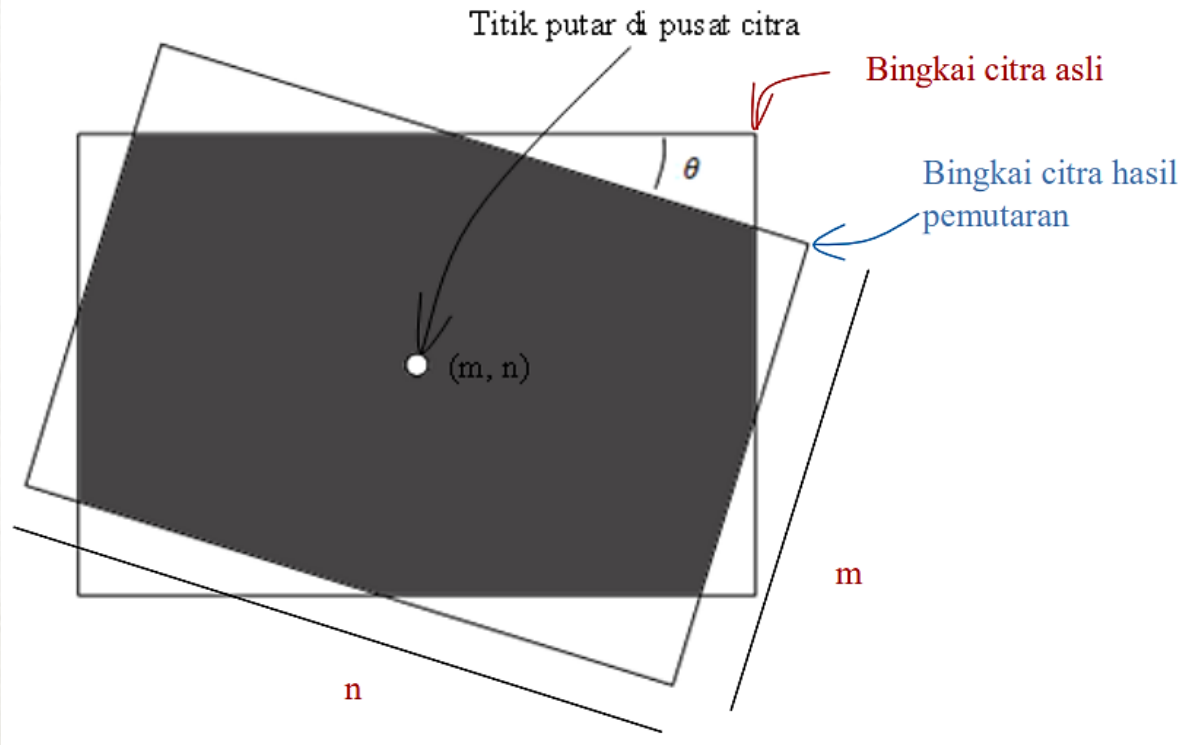
CONTOH HASIL INTERPOLASI



Gambar 8.8 Perbandingan efek interpolasi billinier

MEMUTAR BERDASARKAN SEMBARANG KOODINAT

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



Gambar 8.9 Pemutaran citra melaluli titik pusat citra

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.

$$x_{baru} = (x-n) * Cos(\Theta) + (y-m) * Sin(\Theta) + n \quad \text{..... (8.8)}$$

$$y_{baru} = (y-n) * Cos(\Theta) + (x-n) * Sin(\Theta) + m \quad \text{..... (8.9)}$$

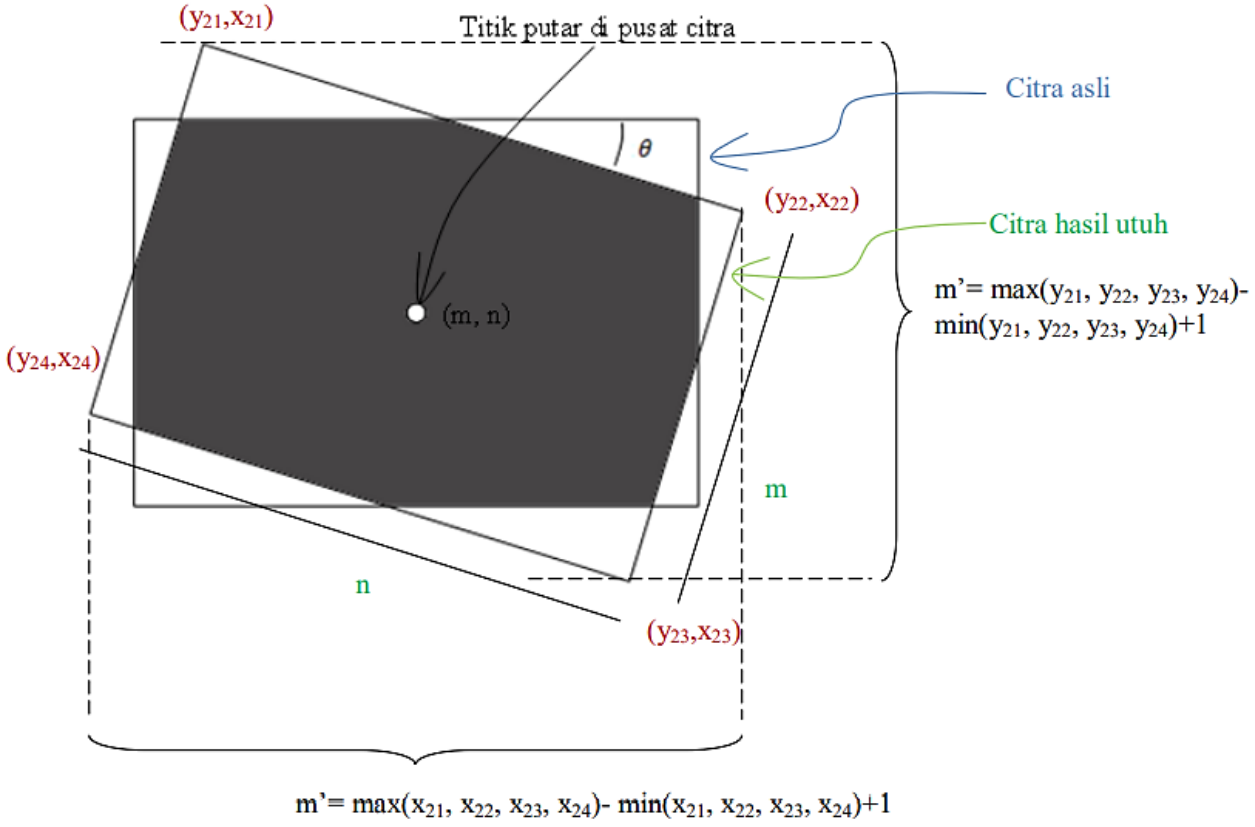
Untuk kepentingan pemutaran citra sejauh Θ searah jarum jam, intensitas piksel (x,y), dapat diperoleh melalui intensitas pada piksel (y_{baru}, x_{baru}), yang tertera pada persamaan 8.8 dan 8.9.

CONTOH HASIL MEMUTAR BERDASARKAN SEMBARANG KOODINAT

(a) Citra gedung asli (b) Hasil pemutaran 5°

Gambar 8.10 Pemutaran melaluli 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

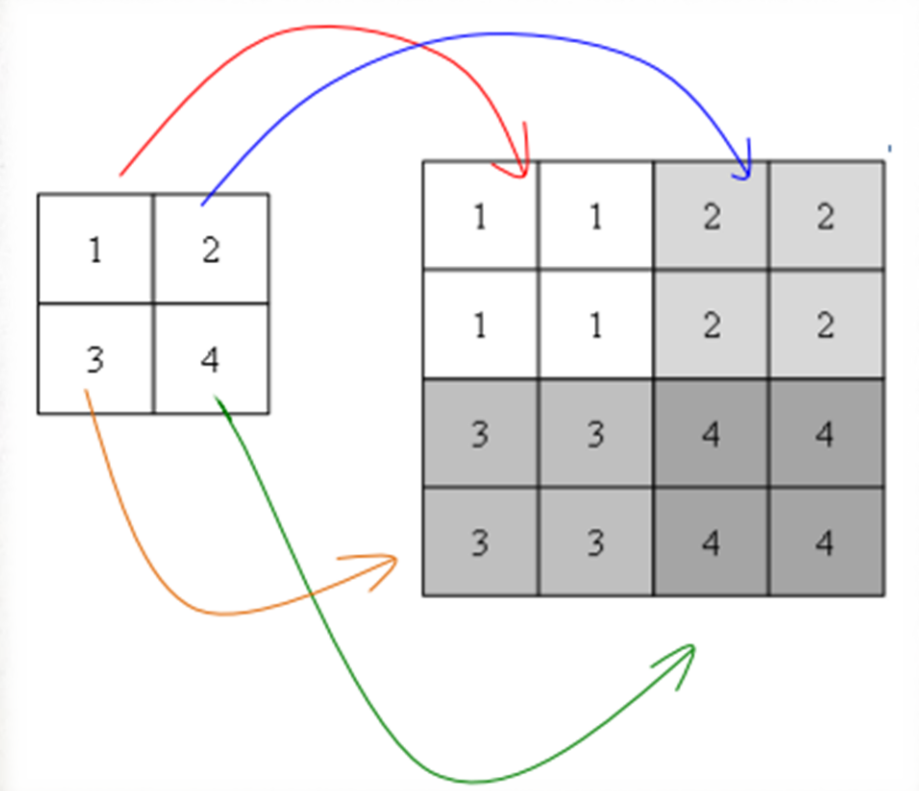
CONTOH HASIL MEMUTAR CITRA SECARA UTUH

(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

Suatu citra dapat diperbesar dengan membuat setiap piksel menjadi beberapa piksel. Seperti ditunjukan 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;
% 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 sy dan sx bilangan pecahan, seperti ½, ¼, 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

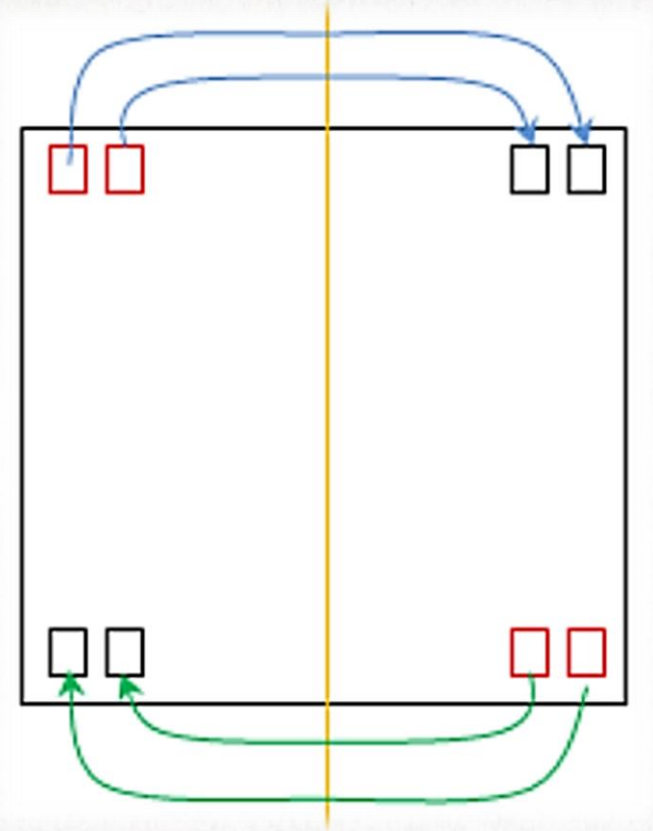
Gambar 8.17 Gedung di perkecil ½ kali pada arah vertikal dan 2,5 kali pada arah horizontal

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 ditunjukan pada Gambar 8.17.

PENCERMINAN CITRA

PENCERMINAN SECARA HORISONTAL



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 ← 1 TO M
2. FOR kolom ← 1 TO N
3. g(baris, kolom) ← f(N – baris + 1, 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 HORISONTAL

(a) Citra boneka.tif

(b) Pencerminan secara horizontal

Gambar 8.19 Contoh hasil pencerminan secara horizontal

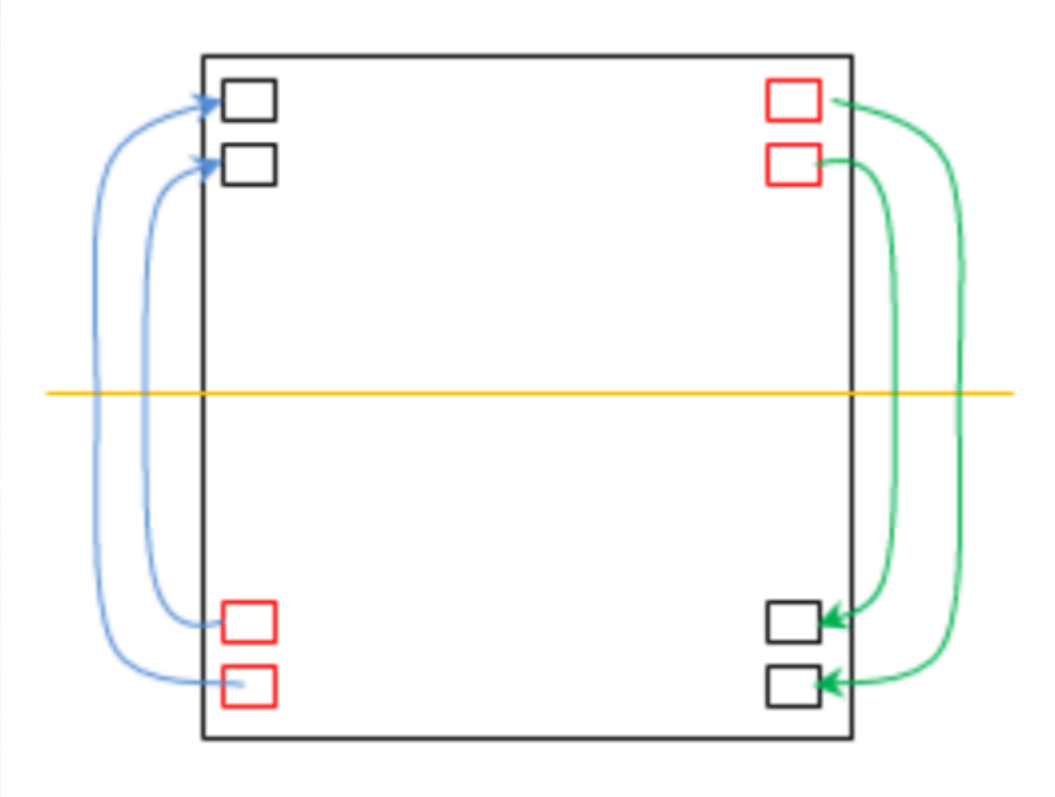
LANJUTAN

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

PENCERMINAN CITRA

LANJUTAN

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 ← 1 TO M
2. FOR kolom ← 1 TO N
3. g(baris, kolom) ← f(baris, N – 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 HORISONTAL CITRA:

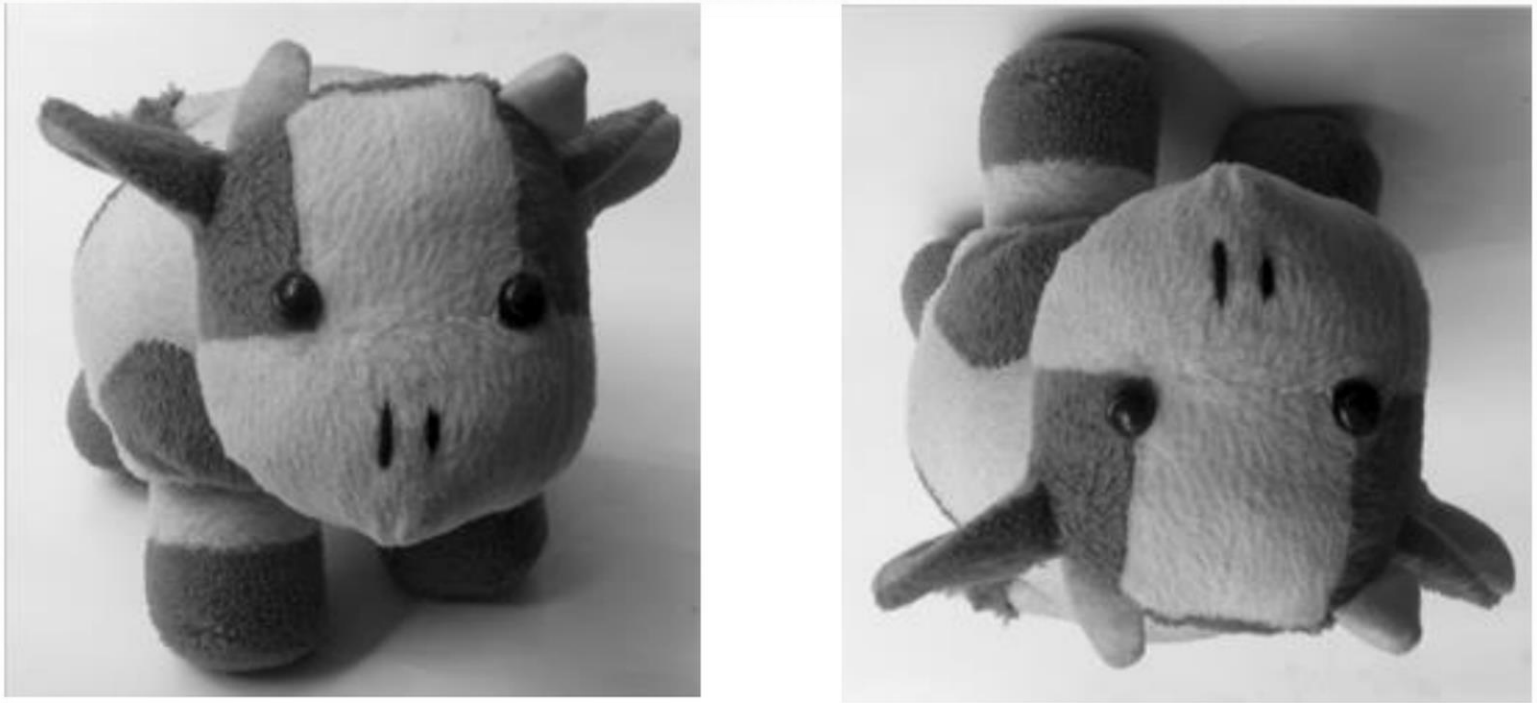
Berdasar fungsi pencerminan secara vertikal di atas, dapat diberikan perintah seperti berikut:

```
>> 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) Pencerminan secara vertikal

Gambar 8.19 Contoh hasil pencerminan secara vertikal

Terimakasih