Operasi-operasi Dasar Pengolahan Citra

Dra. Hernawati, M.T

Bab 4. Operasi-operasi Dasar Pengolahan Citra

1. Operasi Aljabar
2. Operasi Aritmatika
3. Operasi Geometri
4. Zoom
5. Aspect Ratio
6. Rotasi
7. Flipping
8. Cut & Paste
9. Warping Indikator

Pencapaian Mahasiswa dapat melakukan proses tranformasi pada citra

Aras(level) operasi

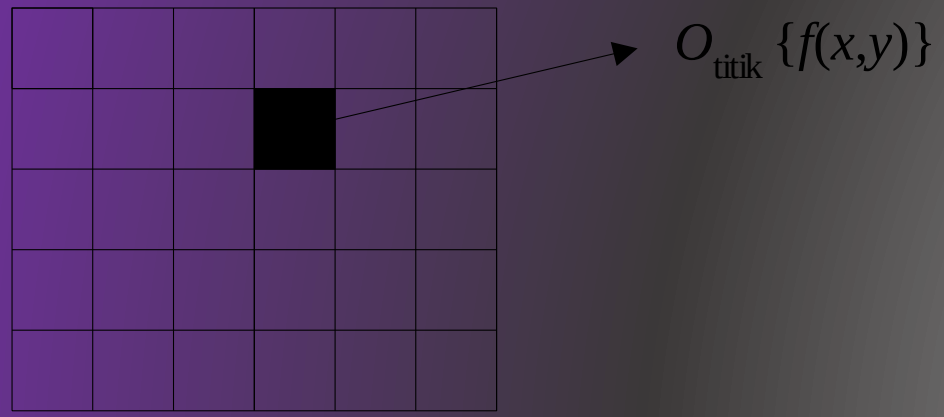
- Operasi-operasi yang dilakukan pada pengolahan citra dapat dikelompokkan ke dalam empat aras (*level*) komputasi:
  - aras titik
  - aras local
  - aras global
  - aras objek

Aras Titik

- Operasi pada aras titik hanya dilakukan pada *pixel* tunggal di dalam citra.

$$f_B(x, y) = O_{\text{titik}}\{f_A(x, y)\}$$

- Operasi ini diulangi untuk keseluruhan *pixel* di dalam citra.



Contoh-contoh operasi titik:

1. Pengambangan (*thresholding*)

$$f(x,y)' = \begin{cases} a_1, & f(x,y) < T \\ a_2, & f(x,y) \geq T \end{cases}$$

Jika $a_1 = 0$ dan $a_2 = 1$, → transformasi citra ke **citra biner**

Contoh T = 128:

$$f(x,y)' = \begin{cases} 0, & f(x,y) < 128 \\ 1, & f(x,y) \geq 128 \end{cases}$$

```
void biner(citra A, citra_biner B, int T, int N, int M)
/* Membuat citra biner dari citra A berdasarkan nilai ambang
(threshold) T yang dispesifikasikan. Ukuran citra adalah N x M.
citra_biner adalah tipe data untuk citra biner).
*/
{ int i, j;
  citra_biner B;

  for (i=0; i<=N-1; i++)
    for (j=0; j<=M-1; j++)
    {
      if (A[i][j] < T)
         B[i][j] = 0;
      else
         B[i][j] = 1;
    }
}
```

Algoritma *mengubah citra A menjadi citra biner.*

2. Membuat citra negatif

- Seperti film negatif pada fotografi.
- Caranya: kurangi nilai intensitas *pixel* dari nilai keabuan maksimum.

Contoh pada citra *grayscale* 8-bit:

$$f(x, y)' = 255 - f(x, y)$$

```
void negatif(citra A, citra B, int N, int M)
/* Membuat citra negatif dari citra A. Hasilnya disimpan di
dalam citra B. Ukuran citra adalah N x M.
*/
{ int i, j;

  for (i=0; i<=N-1; i++)
    for (j=0; j<=M-1; j++)
    {
      B[i][j] = 255 - A[i][j];
    }
}
```

3. Pencerahan Citra (*image brightening*)

- Kecerahan citra dapat diperbaiki dengan menambahkan/mengurangkan sebuah konstanta kepada (atau dari) setiap *pixel.*

$$f(x, y)' = f(x, y) + b$$

- Jika *b* positif, kecerahan citra bertambah,

  Jika *b* negatif kecerahan citra berkurang

- Perlu operasi clipping jika nilai f(x,y)' berada di bawah nilai intensitas minimum atau d atas nilai intensitas maksimum:

$$f(x, y)' = \begin{cases} 255, & f(x, y) > 255 \\ f(x, y), & 0 \le f(x, y) \le 255 \\ 0, & f(x, y) < 0 \end{cases}$$

Gambar Kiri: citra Zelda (agak gelap); kanan: citra Zelda setelah operasi pencerahan citra, $b = 100$

```
void image_brightening(citra A, int b, citra B, int N, int M)
/* Pencerahan citra dengan menjumlahkan setiap pixel di dalam citra A dengan
sebuah skalar b. Hasil disimpan di dalam citra B. Citra berukuran N x M. */
{ int i, j, temp;

  for (i=0; i<=N-1; i++)
    for (j=0; j<=M-1; j++)
    {
       temp = A[i][j] + b;

       /* clipping */
       if (temp < 0)
         B[i][j] = 0;
       else
         if (temp > 255)
            B[i][j]=255;
         else
            B[i][j]=temp;
    }
  }
```

4. Konversi citra berwarna ke citra *grayscale*

- Konversi citra dengan kanal RGB menjadi satu kanal Y (*luminance*)
- Cara paling sederhana: $Y = (R + G + B)/3$
- Hasil yang lebih baik: $Y = 0.299R + 0.587G + 0.144B$

$Y = (R + G + B)/3$

Kurang bagus!

$$Y = 0.299R + 0.587G + 0.144B$$

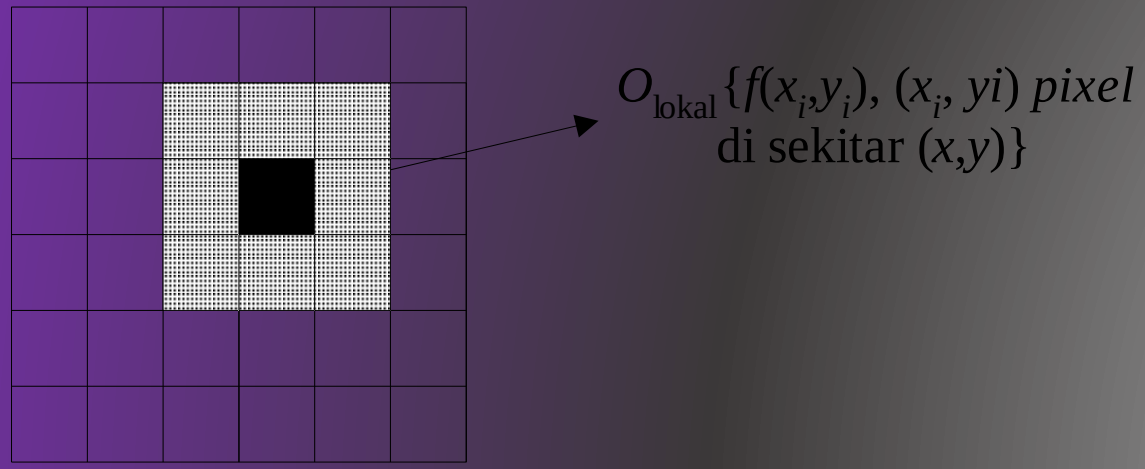
Lebih bagus!

Aras Lokal

- Operasi pada aras lokal menghasilkan citra luaran yang intensitas *pixel-nya* bergantung pada intensitas *pixel-pixel* tetangganya

$$f_B(x, y)' = O_{\text{lokal}}\{f_A(x_i, y_j);\ (x_i, y_j) \in N(x, y)\ \}$$

(keterangan: N = *neighborhood*, yaitu *pixel-pixel* yang berada di sekitar (x, y))

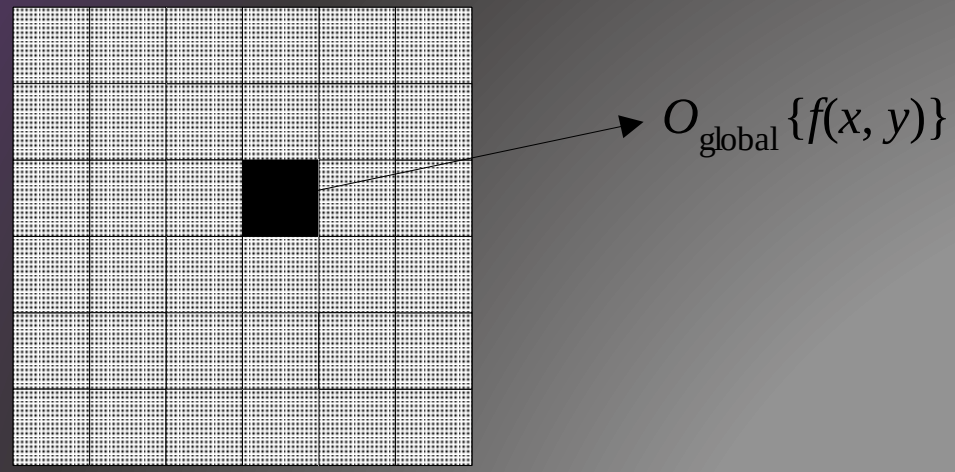


- Contoh operasi beraras lokal adalah operasi konvolusi untuk mendeteksi tepi (*edge detection*) dan pelembutan citra (*image smoothing*). Akan dijelaskan pada kuliah selanjutnya.

Aras Global

- Operasi pada aras global menghasilkan citra keluaran yang intensitas suatu *pixel* bergantung pada intensitas keseluruhan *pixel.*

$$f_B(x, y)' = O_{\text{global}}\{f_A(x, y)\}$$



- Contoh operasi beraras global adalah operasi penyetaraan histogram untuk meningkatkan kualitas citra → akan dibahas pada kuliah selanjutnya.

Aras Objek

- Operasi jenis ini hanya dilakukan pada objek tertentu di dalam citra.
-
- Tujuan dari operasi pada aras objek adalah untuk mengenali objek tersebut, misalnya dengan menghitung ukuran, bentuk, dan karakteristik lain dari objek.

- Operasi aras objek adalah operasi yang kompleks, karena sebelumnya kita harus dapat menjawab: apakah objek itu, bagaimana menemukannya?

Operasi Aritmetika

Karena citra digital adalah matriks, maka operasi-operasi aritmetika matriks juga berlaku pada citra. Operasi matriks yang dapat dilakukan adalah:

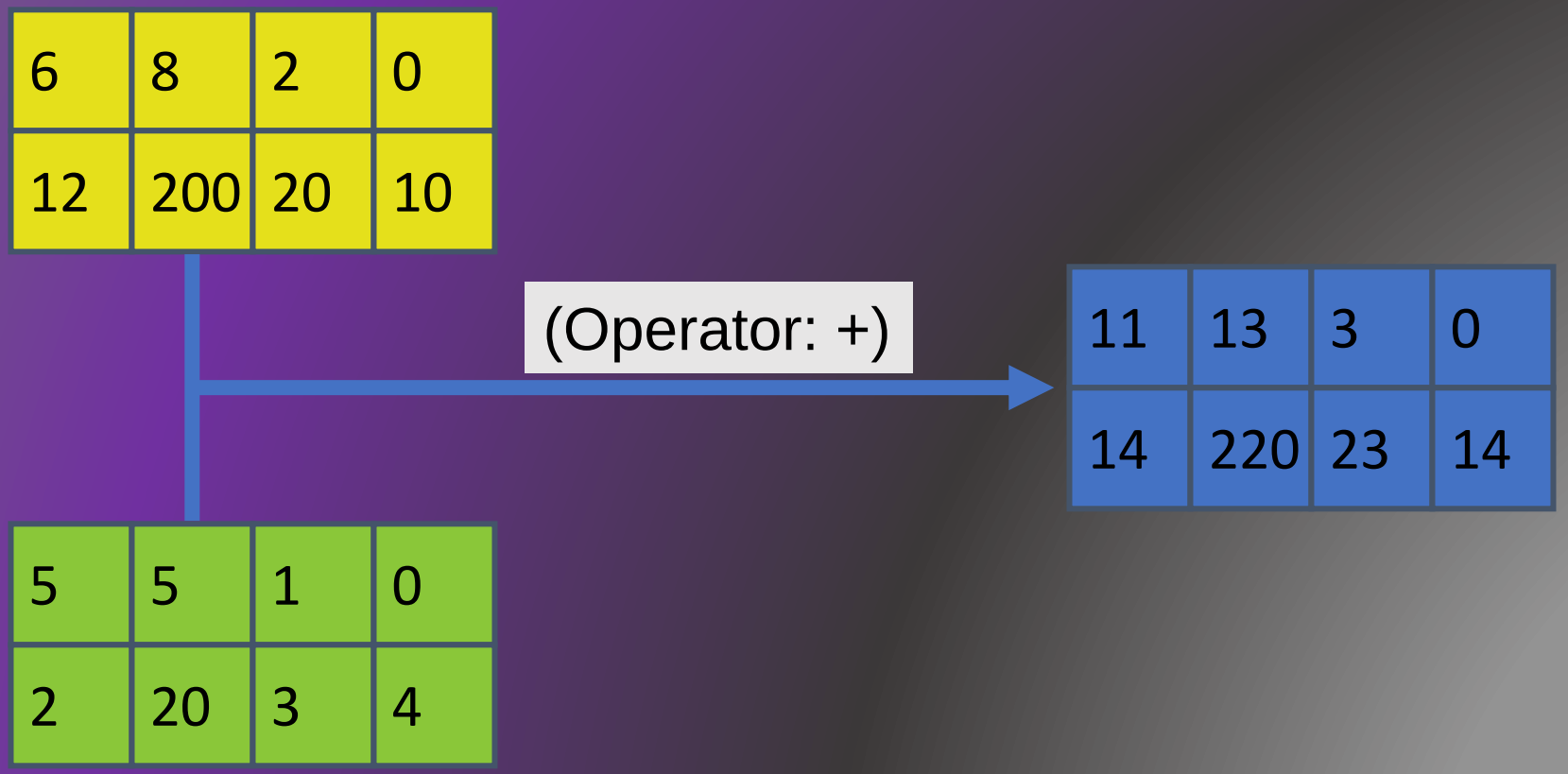
- Penjumlahan/pengurangan dua buah citra: $C(x, y) = A(x, y) \pm B(x, y)$
- Perkalian dua buah citra: $C(x, y) = A(x, y)\, B(x, y)$
- Penjumlahan/pengurangan citra *A* dengan skalar *c*: $B(x, y) = A(x, y) \pm c$
- Perkalian/pembagian citra *A* dengan sebuah skalar *c*: $B(x, y) = c \cdot A(x, y)$

Penjumlahan dua buah citra

$$C(x, y) = A(x, y) + B(x, y)$$

```
void addition(citra A, citra B, citra C, int N, int M)
/* Menjumlahkan dua buah citra A dan B menjadi citra baru, C.
   Citra A, B, dan C masing-masing berukuran N x M.
*/
{ int i, j, temp;

  for (i=0; i<=N-1; i++)
    for (j=0; j<=M-1; j++)
    {
       temp=A[i][j] + B[i][j];
       if (temp > 255) C[i][j]=255; else C[i][j]=temp;
    }
}
```



Sumber gambar: Dr. Rolf Lakaemper, CIS 601 Image ENHANCEMENT in the SPATIAL DOMAIN

- Contoh penjumlahan dua buah citra: mengurangi pengaruh derau (*noise*) di dalam data, dengan cara merata-ratakan derajat keabuan setiap *pixel* dari citra yang sama yang diambil berkali-kali.

- Misalnya untuk citra yang sama direkam dua kali, f_1 dan f_2, lalu dihitung intensitas rata-rata untuk setiap *pixel*

$$f'(x,y) = \frac{1}{2}\{f_1(x,y) + f_2(x,y)\}$$

Pengurangan dua buah citra

$$C(x, y) = A(x, y) - B(x, y)$$

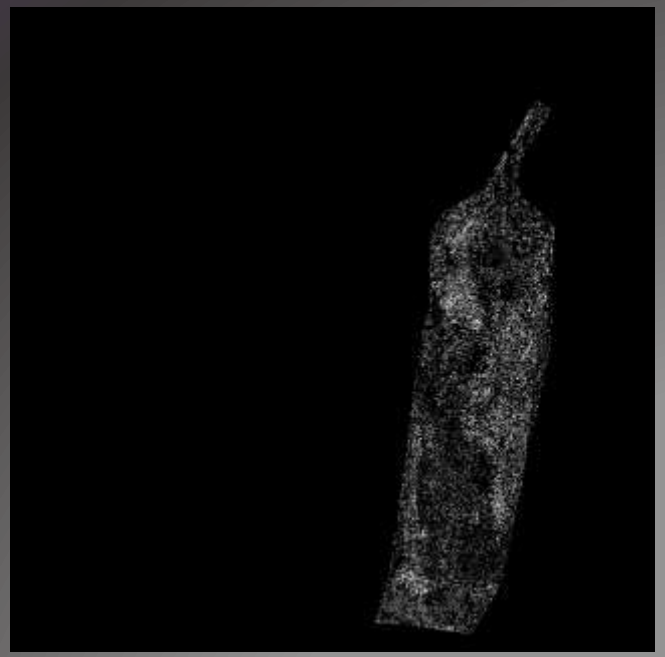
```
void substraction (citra A, citra B, citra C, int N, int M)
/* Mengurangkan dua buah citra A dan B menajdi citra baru, C.
   Citra A, B, dan C berukuran N x M.
*/
{ int i, j;

  for (i=0; i<=N-1; i++)
    for (j=0; j<=M-1; j++)
    {
      C[i][j]=A[i][j] - B[i][j];
      if (C[i][j] != 0) C[i][j]=255;  /* nyatakan objek berwarna putih */
    }
}
```

- Contoh operasi pengurangan citra: Misalkan ada dua buah citra dengan latar belakang yang sama. Citra yang satu memiliki objek, citra yang kedua tanpa ada objek. Hasil pengurangan citra kedua dengan gambar pertama menghasilkan citra yang hanya mengandung objek tersebut. Latar belakangnya hitam.

−

=



Perkalian dua buah citra

$$C(x, y) = A(x, y)\, B(x, y)$$

```
void multiplication(citra A, matriks_riil B, citra C, int N)
/* Mengalikan buah citra A dengan matriks koreksi B menjadi citra C.
   Citra A, matriks B, dan hasil perkalian C berukuran N x N.
*/
{ int i, j, temp;

  for (i=0; i<=N-1; i++)
    for (j=0; j<=N-1; j++)
     { temp=0;
       for (k=0; k<=N-1; k++)
       { temp = temp + A[i][k]*B[k][j];
         /* clipping */
         if (temp < 0)
           C[i][j] = 0;
         else
           if (temp > 255)
              C[i][j]=255;
           else
              C[i][j]=temp;
       }
     }
}
```

- Contoh perkalian citra: mengoreksi ketidaklinieran sensor dengan cara mengalikan matriks citra dengan matrik koreksi

$$\begin{bmatrix} 0 & 12 & 142 & 255 \\ 1 & 6 & 40 & 254 \\ 24 & 0 & 20 & 255 \\ 30 & 2 & 10 & 240 \end{bmatrix} \begin{bmatrix} 0.3 & 0.4 & 0.1 & 0.1 \\ 0.3 & 0.0 & 0.0 & 0.1 \\ 0.3 & 0.0 & 0.0 & 0.0 \\ 0.4 & 0.1 & 0.0 & 0.1 \end{bmatrix} = \begin{bmatrix} 0 & 17 & 157 & 255 \\ 2 & 6 & 40 & 255 \\ 32 & 0 & 20 & 255 \\ 42 & 3 & 10 & 255 \end{bmatrix}$$

Matriks citra A Matriks koreksi B Matriks keluaran C

Penjumlahan/pengurangan citra dengan skalar

$$B(x, y) = A(x, y) \pm c$$

Contoh: *Image brightening*

$$B(x, y) = A(x, y) + 100$$

Nilai sebagian *pixel* sebelum pencerahan:

108 108 108 107 107 107 107 106 106 106
107 107 107 107 107 107 107 106 106 106
107 107 107 107 106 106 106 106 106 106
107 107 107 107 106 106 105 106 107 106
107 108 108 107 107 107 107 106 106 107
108 108 108 108 106 106 106 107 107 107
108 108 108 108 106 106 106 107 107 107
108 108 108 108 106 106 106 107 107 106
108 108 108 107 106 106 107 107 106 106
108 108 108 108 107 107 106 106 107 107

Nilai sebagian *pixel* setelah pencerahan (+ 100):

208 208 208 207 207 207 207 206 206 206
207 207 207 207 207 207 207 206 206 206
207 207 207 207 206 206 206 206 206 206
207 207 207 207 206 206 205 206 207 206
207 208 208 207 207 207 207 206 206 207
208 208 208 208 206 206 206 207 207 207
208 208 208 208 206 206 206 207 207 207
208 208 208 208 206 206 206 207 207 206
208 208 208 207 206 206 207 207 206 206
208 208 208 208 207 207 206 206 207 207

Perkalian/pembagian Citra dengan Skalar

$$B(x, y) = c \cdot A(x, y), \text{ dan } \; B(x, y) = A(x, y) / c$$

- Operasi perkalian citra dengan skalar dipakai untuk kalibrasi kecerahan (*callibration of brightness*).

- Operasi pembagian citra dengan skalar dipakai untuk normalisasi kecerahan (*normalization of brightness*).

Operasi Boolean

- Operasi AND: $C(x, y) = A(x, y)$ **and** $B(x, y)$
- Operasi OR: $C(x, y) = A(x, y)$ **or** $B(x, y)$
- Operasi NOT: $C(x, y) =$ **not** $A(x, y)$

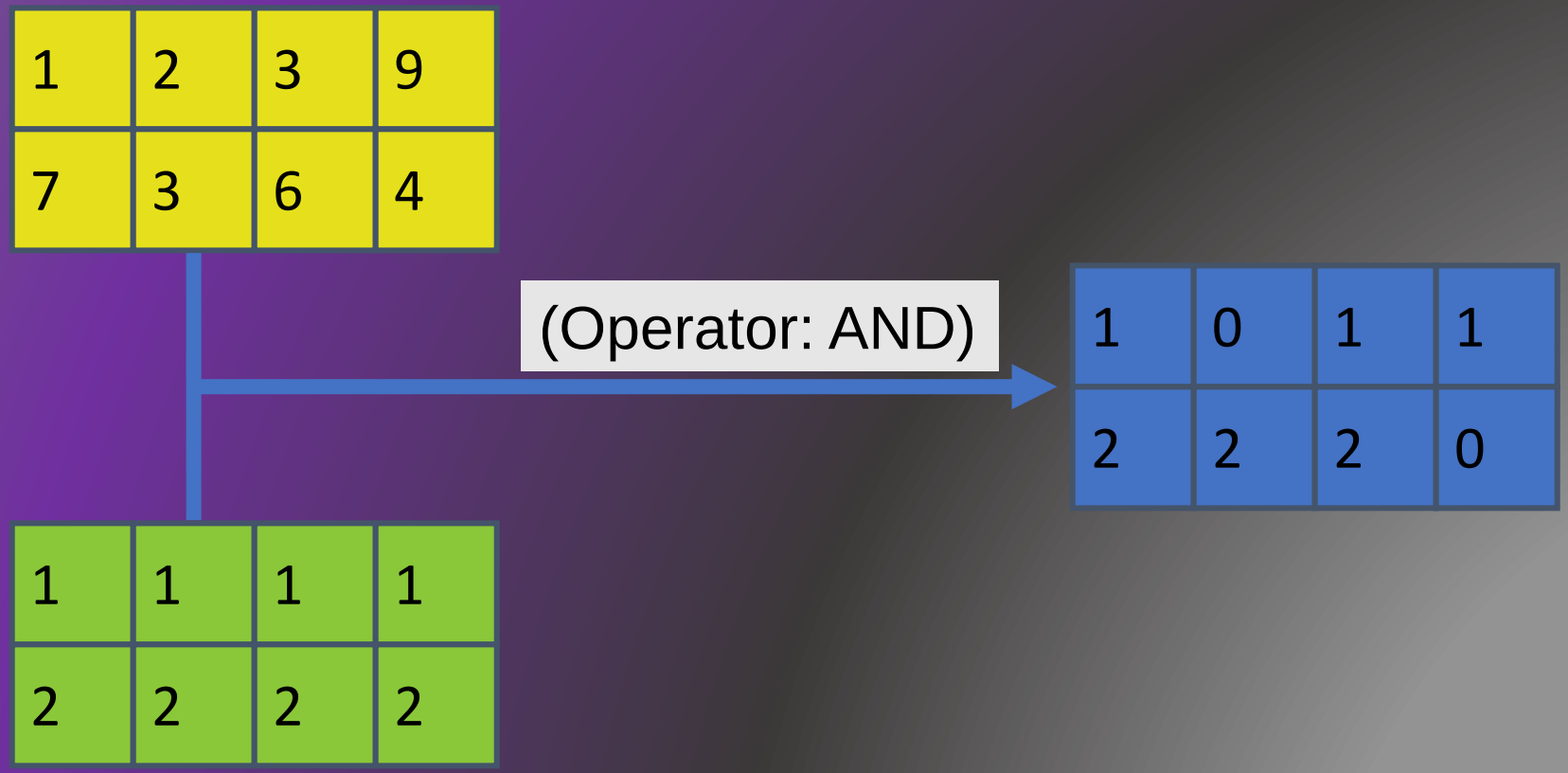
- Contoh operasi NOT:

Ganesha

NOT Ganesha

```
void not(citra_biner A, citra_biner B, int N, int M)
/* Membuat citra komplemen dari citra biner A.
Komplemennya disimpan di dalam B. Ukuran citra A
adalah N x M.
*/
{ int i, j;

  for (i=0; i<=N-1; i++)
    for (j=0; j<=M-1; j++)
    {
      B[i][j] = !A[i][j];
    }
}
```



Sumber gambar: Dr. Rolf Lakaemper, CIS 601 Image ENHANCEMENT in the SPATIAL DOMAIN

Operasi Geometri

- Operasi geometri: translasi, rotasi, penskalaan citra, pencerminan citra (*flipping*).
- Pengubahan geometri dari citra $f(x, y)$ menjadi citra baru $f'(x, y)$ dapat ditulis sbb:

$$f'(x', y') = f(g_1(x, y), g_2(x, y))$$

yang dalam hal ini, $g_1(x)$ dan $g_2(y)$ adalah fungsi transformasi geometrik. Dengan kata lain,

$$x' = g_1(x, y)$$
$$y' = g_2(x, y)$$

1. Translasi

Rumus translasi sejauh m dalam arah x dan *n* dalam arah y:

$$x' = x + m$$
$$y' = y + n$$

Translasi citra A menjadi citra B sejauh *m* dalam arah x dan *n* dalam arah y:

$$B[x][y] = A[x + m][y + n]$$

```
void translation(citra A, citra B, int N, int M, int m, int n)
/* Mentranslasi citra A sejauh m, n menjadi citra B. Ukuran citra N x M. */
{ int i, j;

  for (i=0; i<=N-1; i++)`
    for (j=0; j<=M-1; j++)
    {
       B[i][j]=A[i+m][j+n];
    }
}
```

Translasi pada citra *camera*: (a) citra semula, (b) citra hasil translasi dengan m = 30 dan n = 25.

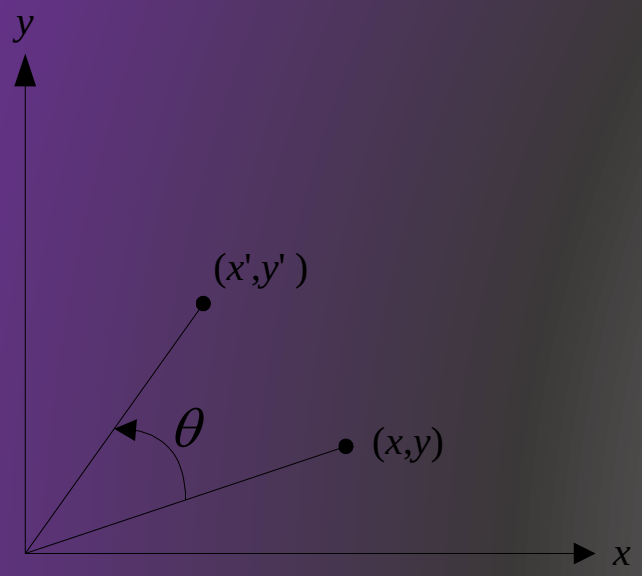
2. Rotasi

Rumus rotasi citra sejauh θ radian berlawanan arah jarum jam :

$$x' = x\cos(\theta) - y\sin(\theta)$$
$$y' = x\sin(\theta) + y\cos(\theta)$$

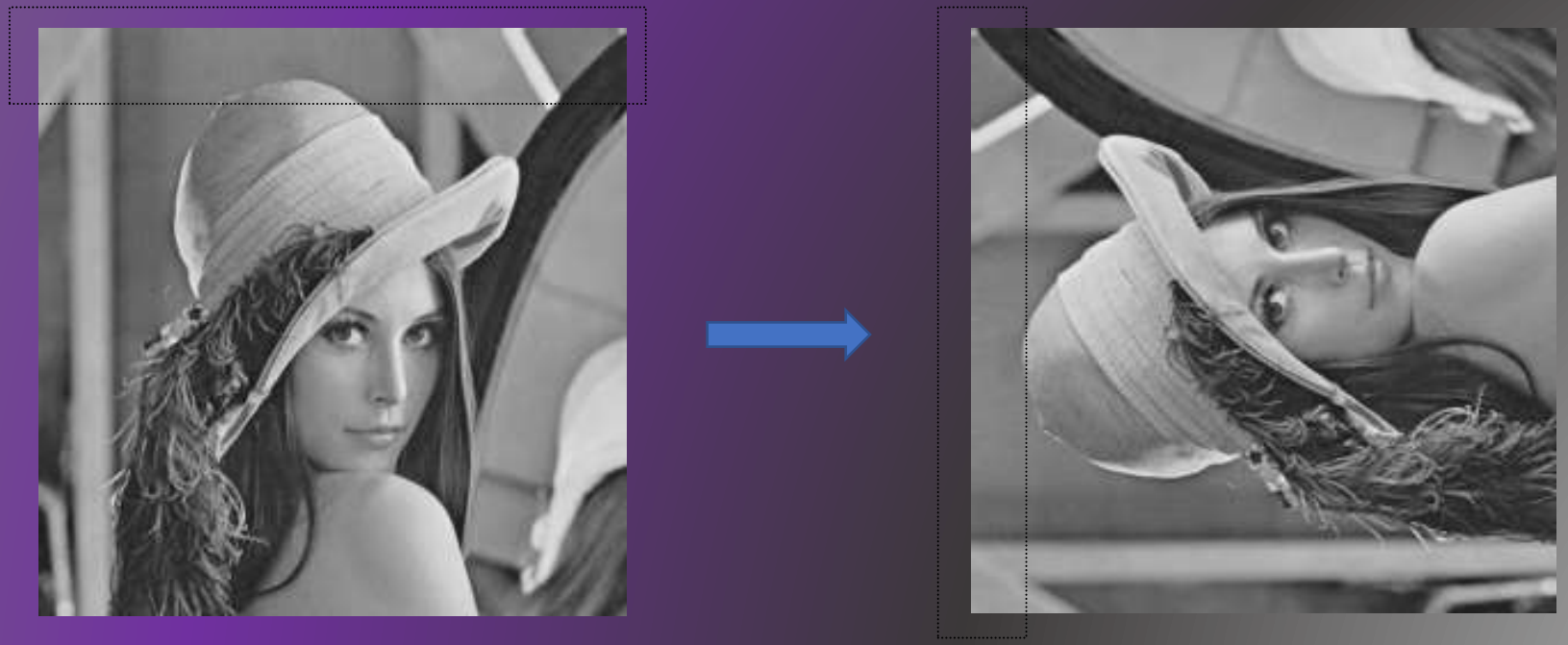
Rotasi citra A menjadi citra B sejauh θ radian berlawanan arah jarum jam :

$$B[x'][y'] = B[x\cos(\theta) - y\sin(\theta)][x\cos(\theta) + y\cos(\theta)] = A[x][y]$$



- Jika sudut rotasinya 90°, maka implementasinya lebih mudah dilakukan dengan cara menyalin *pixel-pixel* baris ke *pixel-pixel* kolom pada arah rotasi.

- Rotasi 180° diimplementasikan dengan melakukan rotasi 90° dua kali.



Gambar *rotasi citra Lena sejauh 90° berlawanan arah jarum jam*

```
void rotation90CCW(citra A, citra B, int N, int M)
/* Rotasi citra A sejauh 90° berlawanan arah jarum jam (CCW = Clock Counter-
wise). Ukuran citra adalah N x M. Hasil rotasi disimpan di dalam citra B.
*/
{ int i, j, k;

  for (i=0; i<=N-1; i++)
  {
    k=M-1;
    for (j=0; j<=M-1; j++)
    {
      B[k][i]=A[i][j];
      k--;
    }
  }
}
```

```
void rotation90CW(citra A, citra B, int N, int M)
/* Rotasi citra A sejauh 90° searah jarum jam (CW = Clock-wise).
   Ukuran citra adalah N x M. Hasil rotasi disimpan di dalam cira B.
*/
{ int i, j, k;

  k=M-1;
  for (i=0; i<=N-1; i++)
  {
   for (j=0; j<=M-1; j++)
   {
      B[j][k]=A[i][j];
    }
    k--;
  }
}
```

3. Flipping

- *Flipping* adalah operasi geometri yang sama dengan pencerminan (*image reflection*).
- Ada dua macam *flipping*: horizontal dan vertikal

- *Flipping* horizontal: pencerminan pada sumbu-*Y* (*cartesian*) dari citra *A* menjadi citra *B* :

  $B[x][y] = A[N - x][y]$

- *Flipping* vertical: pencerminan pada sumbu-*X* (*cartesian*) dari citra *A* menjadi citra *B*:

  $B[x][y] = A[x][M - y]$

- Pencerminan pada titik asal (*cartesian*) dari citra *A* menjadi citra *B:*

  $B[x][y] = A[N - x][M - y]$

- Pencerminan pada garis $x = y$ dari citra *A* menjadi citra *B*:

  $B[x][y] = A[y][x]$

```
void vertical_flip(citra A, citra B, int N, int M)
/* Flipping vertikal (pencerminan terhadap sumbu-X) terhadap citar A. */
   Ukuran citra adalah N x M. Hasil flipping disimpan di dalam citra B.
*/
{ int i, j, k;

  k=M-1;
  for (i=0; i<=N-1; i++)
  {
   for (j=0; j<=M-1; j++)
   {
     B[k][j]=A[i][j];
   }
   k--;
   }

}
```

4. Penskalaan citra (*image zooming*)

- Pengubahan ukuran citra (membesar/*zoom out* atau mengecil/*zoom in*).
- Rumus penskalaan citra:

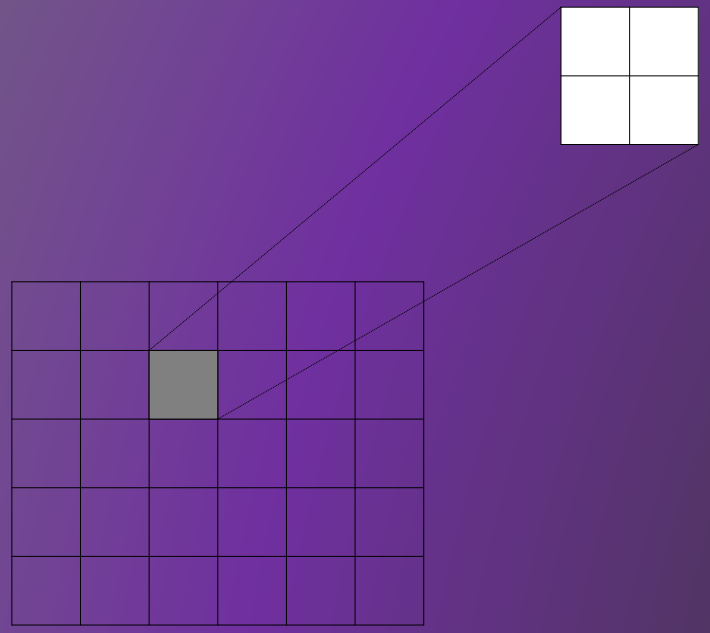
$$x' = s_x \cdot x$$
$$y' = s_y \cdot y$$

s_x dan s_y adalah faktor skala masing-masing dalam arah x dan arah y.

- Penskalaan citra A menjadi citra B:

$$B[x'][y'] = A[s_x \cdot x][\, s_y \cdot y]$$

- Operasi *zoom in* dengan faktor 2 (yaitu, $s_x = s_y = 2$) diimplementasikan dengan menyalin setiap *pixel* sebanyak 4 kali.

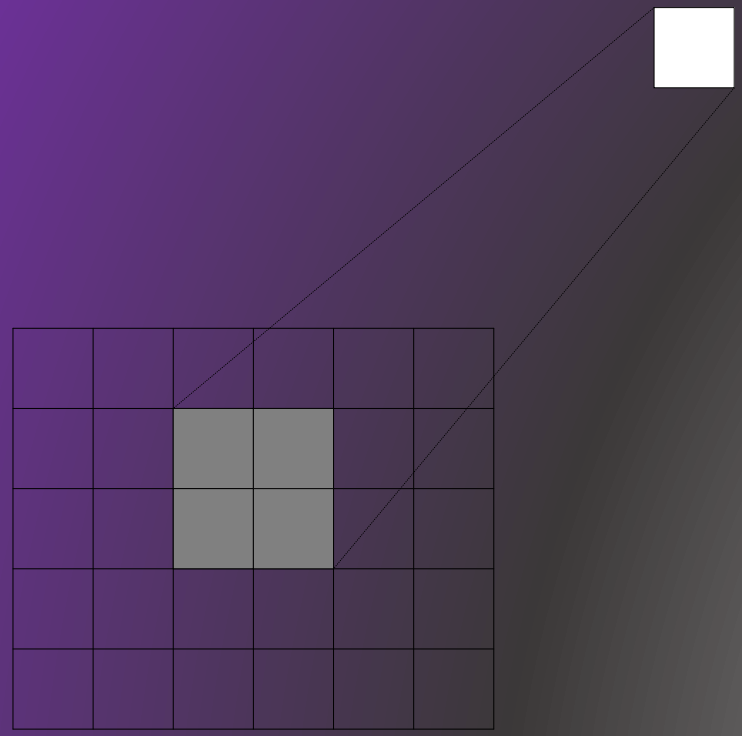


Gambar Kiri: citra kota San Fransisco (ukuran normal),
Kanan: citra kota San Fransisco setelah diperbesar 2 kali ($s_x = s_y = 2$):

```
void zoom_in(citra A, citra B, int N, int M)
/* perbesaran citra A dengan faktor skala 2
   Ukuran citra adalah N x M. Hasil perbesaran disimpa d dalam citra B.
*/
{ int i, j, k, m, n;

  m=0; n=0;
  for (i=0; i<=N-1; i++)
   {
     for (j=0; j<=M-1; j++)
     {
       B[m][n]= A[i][j];
       B[m][n+1]= A[i][j];
       B[m+1][n]= A[i][j];
       B[m+1][n+1]= A[i][j];
       n=n+2;
      }
      m=m+2;
      n=0;
   }
}
```

- Operasi *zoom out* (pengecilan) dengan faktor skala = ½ dilakukan dengan mengambil rata-rata dari 4 *pixel* yang bertetangga menjadi 1 *pixel*



Tugas

- Implementasikan menjadi program dalam Bahasa C/C++ ke dalam aplikasii Photoshop mini++ yang anda buat:
  1. Membuat citra negatif
  2. Mengubah citra berwarna menjadi citra *grayscale*
  3. *Image brightening*
  4. Operasi aritmetika dua buah citra
  5. Operasi boolean pada citra
  6. Operasi geometri (translasi, rotasi, flipping, zooming)