知能制御システム学

画像処理の基礎 (2)
— OpenCV による実践 —

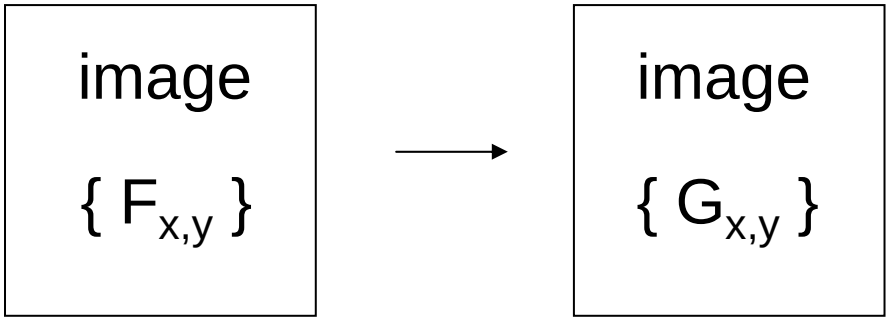
東北大学 大学院情報科学研究科
鏡 慎吾
swk(at)ic.is.tohoku.ac.jp

2007.06.26

画像処理の分類

input	output	
image	image (2-D data)	image to image processing Fourier trans., label image
	1-D data	projection, histogram
	scalar values	position, recognition
image sequence	image (2-D data)	motion image processing
	...	

image to image processing



point operation (点処理)

$G_{i,j}$ depends only on $F_{i,j}$

local operation / neighboring operation (局所処理)

$G_{i,j}$ depends on some neighborhood of the pixel (i, j) in $\{F_{i,j}\}$

global operation (大域処理)

$G_{i,j}$ depends on (nearly) all the pixels in $\{F_{i,j}\}$

Typical Point Operations

2値化 (binarization) [前回の授業でも紹介]

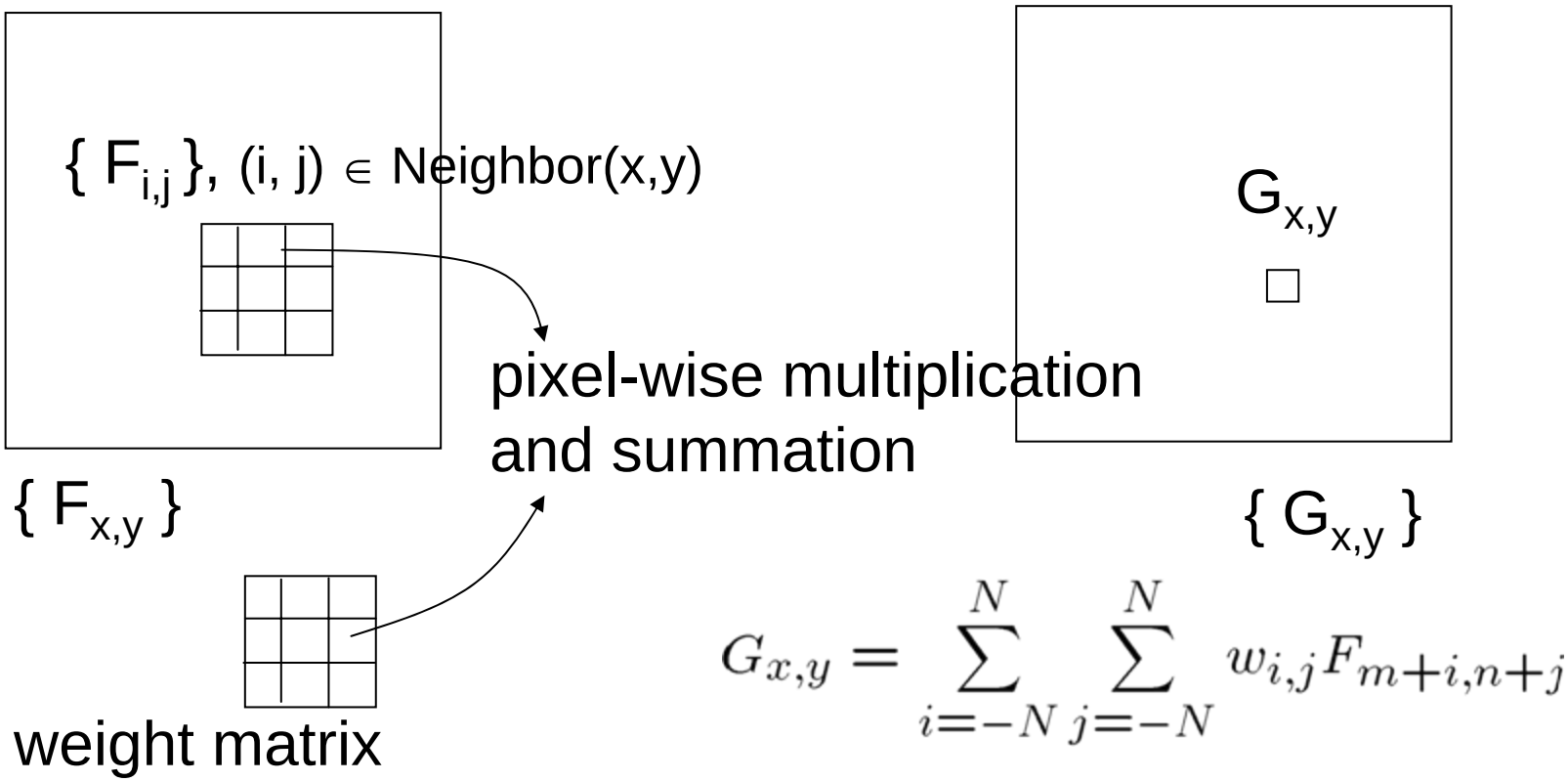
サンプルプログラム: imageproc_binarize.cpp

```
for (j = 0; j < N; j++) {
    for (i = 0; i < M; i++) {
        if (image[M * j + i] >= 128) {
            image[M * j + i] = 255;
        } else {
            image[M * j + i] = 0;
        }
    }
}
```

他の例: 濃度値変換(ヒストグラム平坦化, ガンマ補正など)

Typical Local Operations

Convolution of weight matrices (3x3, 5x5, ...)



$$G_{x,y} = \sum_{i=-N}^{N} \sum_{j=-N}^{N} w_{i,j} F_{m+i,n+j}$$

Often referred to as ``linear spatial filtering'' (線形空間フィルタリング) or simply ``linear filtering'' (線形フィルタリング)

Example Code

```
double w[3 * 3] = { 0.0,  1.0,  0.0,
                    1.0, -4.0,  1.0,
                    0.0,  1.0,  0.0 };
double scaling = 1.0;

for (j = 1; j < N - 1; j++) {
    for (i = 1; i < M - 1; i++) {
        double sum = 0.0;
        for (n = 0; n < 3; n++) {
            for (m = 0; m < 3; m++) {
                sum += PIXVAL(img, i - 1 + m, j - 1 + n)
                       * w[m + 3 * n];
            }
        }
        PIXVAL(result, i, j) = REAL2UCHAR(scaling * sum);
    }
}
```

サンプルプログラム: imageproc_3x3.cpp

- 重み行列は，本当は CvMat などの OpenCV で標準的に用いられる形式で表しておいた方が何かとよい.
- 本当は，周辺画素の処理をきちんと行う必要がある.
- REAL2UCHAR() は，計算後の値が 0 〜 255 の間に収まるように処理する関数. 実装はいい加減なので良い子は真似しないこと.

Examples of Weight Matrices

Smoothing (平滑化)

1	1	1
1	1	1
1	1	1

1	1	1
1	2	1
1	1	1

0	1	0
1	4	1
0	1	0

Sharpening (先鋭化)

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

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

Examples of Weight Matrices

Edge Detection

-1	0	1
-1	0	1
-1	0	1

単純な水平方向
1次微分

-1	-1	-1
0	0	0
1	1	1

単純な垂直方向
1次微分

-1	0	1
-2	0	2
-1	0	1

注目画素の近くに重み
(Sobel フィルタ)

他多数．その他，線形でない（つまり重みつき総和で表せない）
空間フィルタも多数ある．

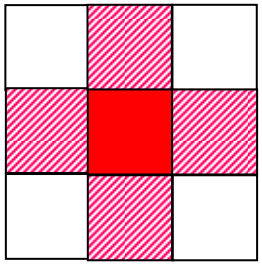
binary image processing

2値化された後の画像 (binary image) の処理は，それだけで一分野をなすほど独特に発展している.

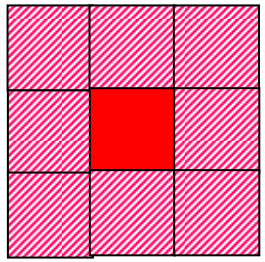
- 実用上重要であった
- 幾何学的に明確な議論がしやすく，体系的な議論が進んだ

Connectivity

近傍 (neighbor): ある画素の近くにある画素の集合. いろいろな定義が可能.



4-neighbor



8-neighbor

互いに n-近傍の関係にある 2 つの画素は「n-隣接している」(n-adjacent) と呼ばれる.

同じ画素値を持つ2つの画素 a, b に対して画素の系列 p_0 (= a), p_1, p_2, ..., p_{n-1}, p_n (= b) が存在し, p_i はすべて同じ画素値を持ち, p_i と p_{i-1} が n-隣接するとき, 画素 a と b は「n-連結している」(n-neighbor connected) という

mathematical morphology

dilation: 近傍の誰かが 1 だったら自分も 1 になる

$G_{i,j} = F_{i,j} \mid F_{i-1,j} \mid F_{i+1,j} \mid F_{i,j-1} \mid F_{i,j+1}$ (4-近傍の場合)

erosion: 近傍の誰かが 0 だったら自分も 0 になる

$G_{i,j} = F_{i,j} \,\&\, F_{i-1,j} \,\&\, F_{i+1,j} \,\&\, F_{i,j-1} \,\&\, F_{i,j+1}$ (4-近傍の場合)

opening: erosion + dilation

closing: dilation + erosion

サンプルプログラム: imageproc_morph.cpp

Example of ``image to 1D data'' processing

projection (射影)

```
for (i = 0; i < img->width; i++) { hx[i] = 0; }
for (j = 0; j < img->height; j++) { hy[j] = 0; }

for (j = 0; j < img->height; j++) {
    for (i = 0; i < img->width; i++) {
        if (PIXVAL(img, i, j) > 128) {
            hx[i]++;
            hy[j]++;
        }
    }
}
```

サンプルプログラム: imageproc_proj.cpp

Example of ``image to scalar values'' processing

moment features

$$m_{p,q} = \sum_i \sum_j i^p j^q F_{i,j}$$

$$m_{0,0} = \sum_i \sum_j F_{i,j}$$ 0次モーメント: 2値画像の場合，面積

$$m_{1,0} = \sum_i \sum_j i F_{i,j}$$ x方向の1次モーメント

$$m_{0,1} = \sum_i \sum_j j F_{i,j}$$ y方向の1次モーメント

0～1次モーメントまでの情報から，面積と重心が分かる.

$(g_x, g_y) = (m_{1,0}/m_{0,0}, m_{0,1}/m_{0,0})$

さらに高次のモーメントから，形状に関するより詳細な情報が得られる.

References

[1] 田村: コンピュータ画像処理, オーム社, 2002.

[2] http://sourceforge.net/projects/opencvlibrary/

[3] http://opencvlibrary.sourceforge.net/