Intelligent Control Systems

Image Processing (2)

— Point Operations and Local Spatial Operations —

Shingo Kagami
Graduate School of Information Sciences,
Tohoku University
swk(at)ic.is.tohoku.ac.jp

http://www.ic.is.tohoku.ac.jp/ja/swk/

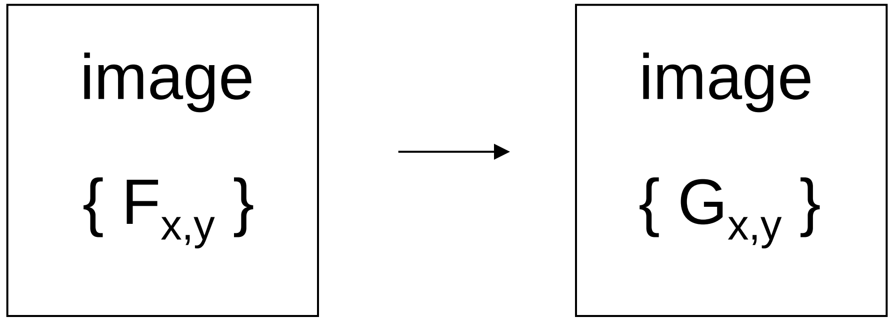
Image Processing Classification

input	output	example
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 sequence	motion image processing
	image	
	1-D data	
	scalar	

Outline

- Image to Image Processing
  - Point Operations
  - Local Operations

Image to Image



point operation

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

local operation / neighboring operation

$G_{i,j}$ depends on pixels within some neighborhood of $F_{i,j}$

global operation

$G_{i,j}$ depends on almost all the pixels in { $F_{i,j}$ }

Point Operation Examples

pixel value conversion, color conversion

- e.g.: binarization, pixel value inversion, gamma correction

```
cv::Mat input, output1, output2;

...

cv::threshold(input, output1, 128, 255, THRESH_BINARY);
cv::equalizeHist(input, output2);
```

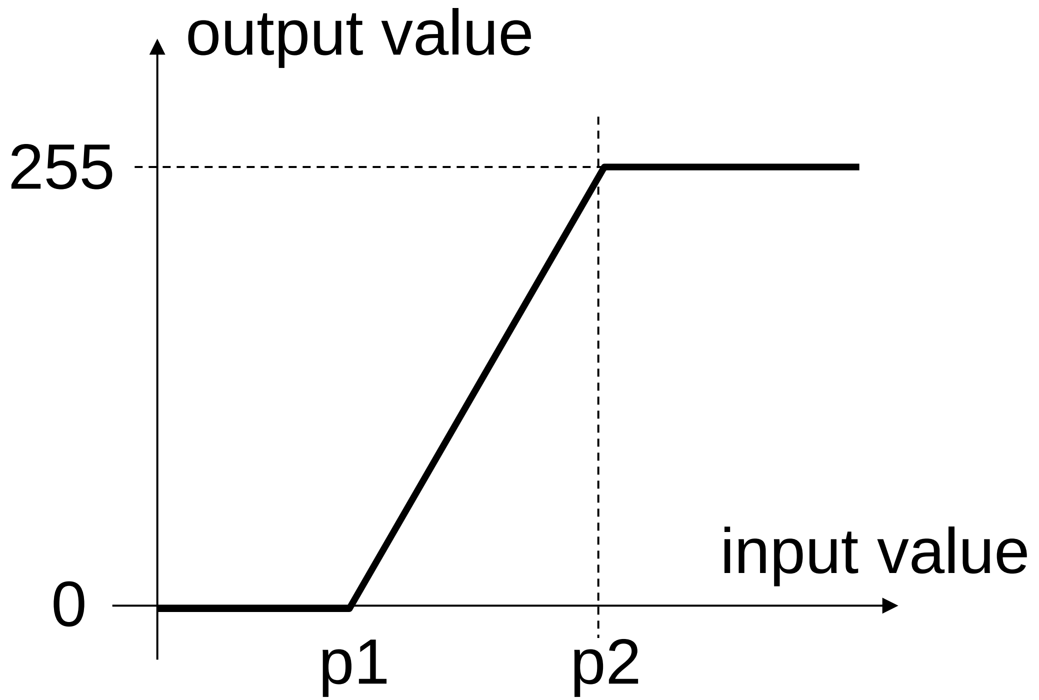
(Handling of color images will be explained next week)

Implementation

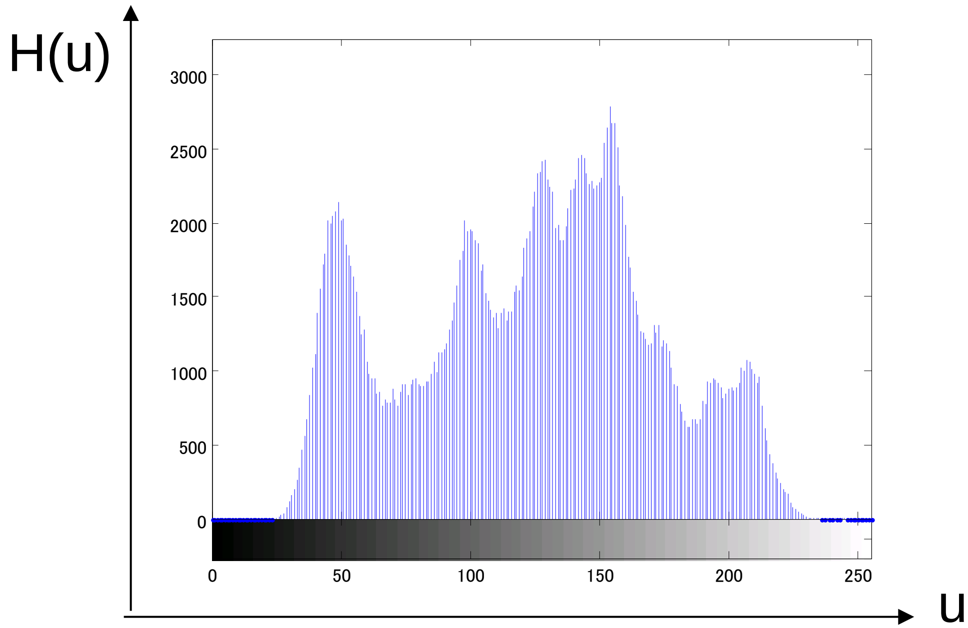
```
for (j = 0; j < height; j++) {
    for (i = 0; i < width; i++) {
        output.at<uchar>(j, i)
            = some_func(input.at<uchar>(j, i));
    }
}
…

uchar some_func(uchar val) {
    …
}
```

Note again: `img.at<uchar>(j, i)` is an 8-bit value of the pixel at x = i, y = j

Pixel value conversion example



Histogram (of pixel values)



$$\boldsymbol{H} = \{H_u\}_{u=1,2,\cdots,m}, \;\; H_u = \sum_{\boldsymbol{x} \in S(u)} 1$$

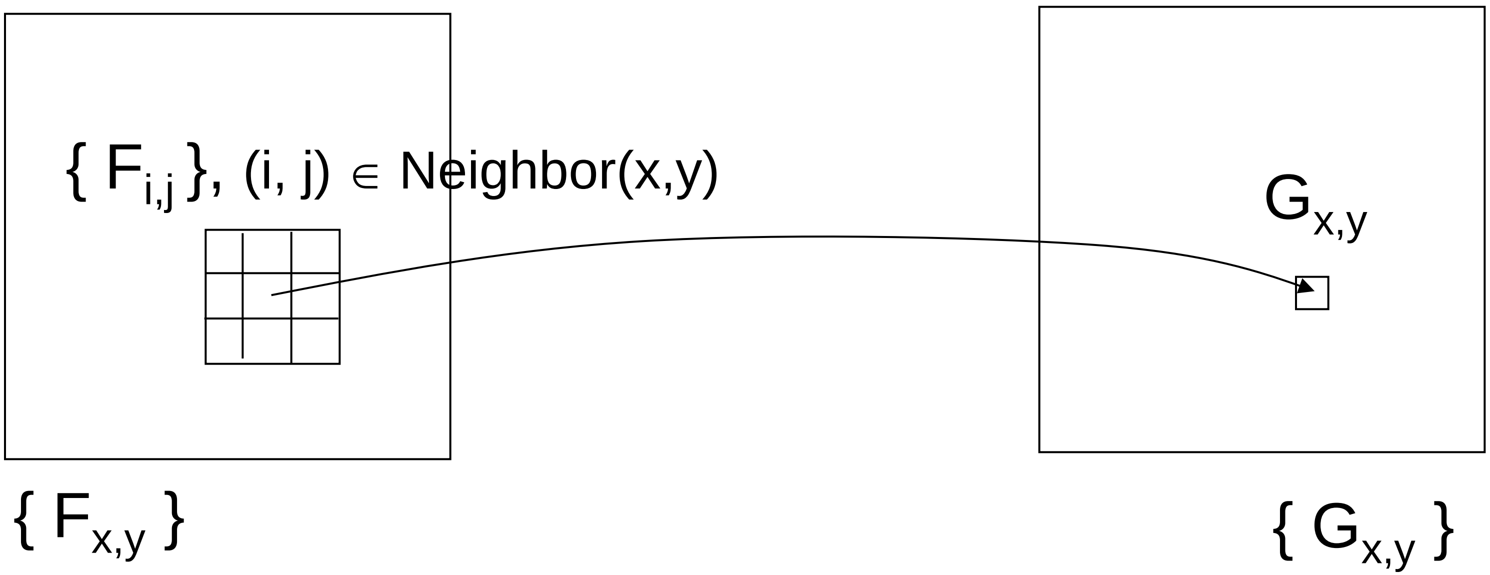
where S(u) is a set of pixels having values belonging to the bin u

Outline

- Image to Image Processing
  - Point Operations
  - Local Operations

Local operation example: Spatial Filter

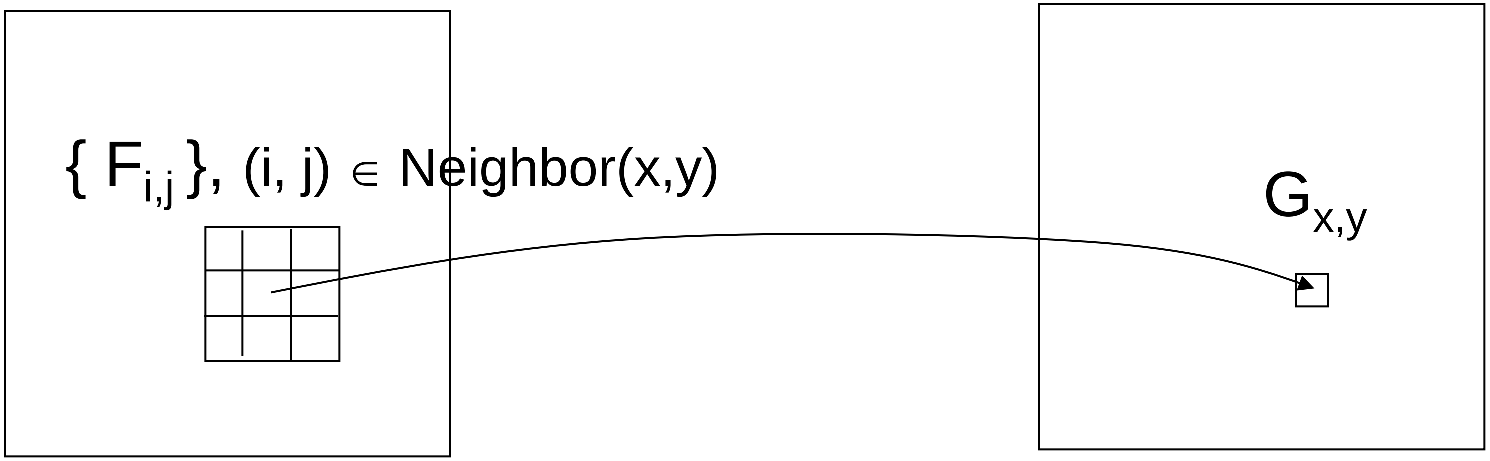
$G_{x,y}$ depends on some neighborhood (e.g. 3×3, 5×5 pixels, etc.) of the point of interest (x,y)



Typical examples: smoothing, edge detection

Important Example: Smoothing

- Output at (x, y): some representative value of the set of neighbor pixels around (x, y), e.g. mean, weighted mean, median
- Used for: e.g. noise reduction, scale-space processing



1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

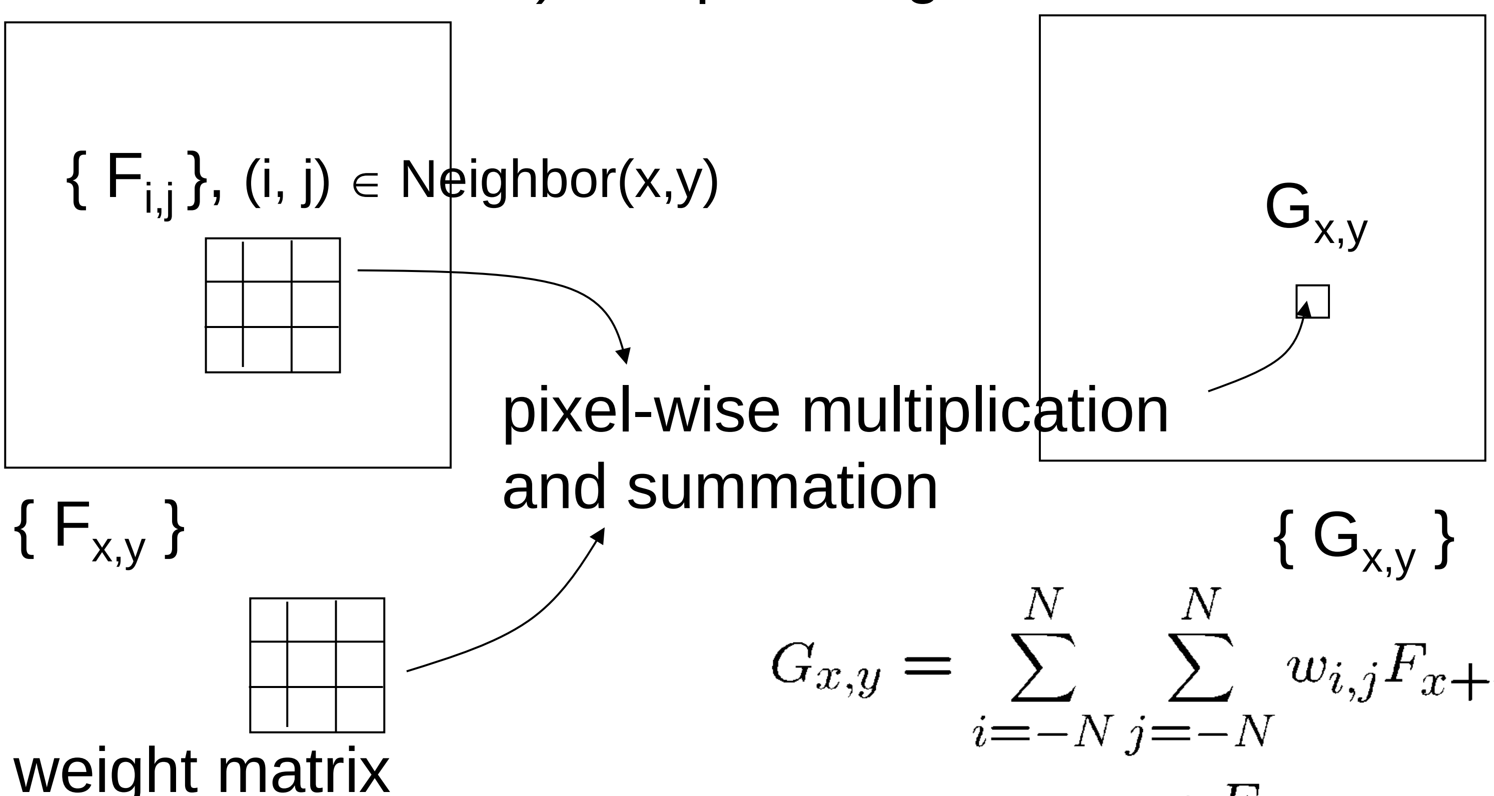
(mean)

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

(weighted mean)

Linear Spatial Filtering

- Smoothing with (weighted) mean is an example of linear spatial filtering (while smoothing with median is nonlinear)
- Computed by convolving a weight matrix (filter coefficients, filter kernel, or mask) to input image



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

Examples of 3x3 smoothing weight matrices

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

=

1/9

1	1	1
1	1	1
1	1	1

1/10	1/10	1/10
1/10	1/5	1/10
1/10	1/10	1/10

=

1/10

1	1	1
1	2	1
1	1	1

0	1/8	0
1/8	1/2	1/8
0	1/8	0

=

1/8

0	1	0
1	4	1
0	1	0

Linear filtering example

```
cv::Mat weight = (cv::Mat_<float>(3,3) <<
                      0, 1, 0,
                      1, 4, 1,
                      0, 1, 0);
weight = weight / 8.0f;
```

convenient way of matrix initialization

```
…
cv::filter2D(input, output, CV_8U, weight);
```

Or, very common filters are readily available

```
cv::GaussianBlur(input, output, cv::Size(0, 0), 10.0);
cv::Sobel(input, output, CV_8U, 1, 0);
cv::Laplacian(input, output, CV_8U);
```

Implementation of 3x3 linear filtering

```
cv::Mat input, output;
cv::Mat weight = (cv::Mat_<int>(3,3) <<
    0, 1, 0,
    1, 4, 1,
    0, 1, 0);
int normalizer = 8;

for (j = 1; j < height - 1; j++) {
    for (i = 1; i < width - 1; i++) {
        int sum = 0;
        for (n = 0; n < 3; n++) {
            for (m = 0; m < 3; m++) {
                sum += weight.at<int>(n, m)
                      * input.at<uchar>(j - 1 + n, i - 1 + m);
            }
        }

        output.at<uchar>(j, i) = saturate(sum / normalizer);
    }
}
```

boundary handling (just untouched)

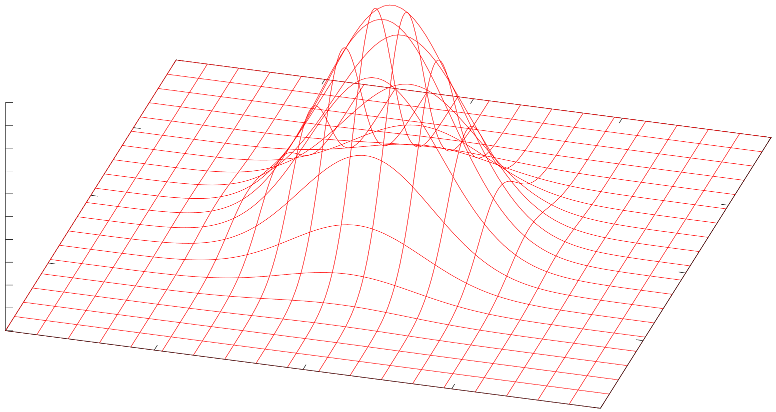
Note: center coordinate of weight is not (0, 0) but (1, 1)

make sure the pixel value is in [0, 255]

Gaussian: most widely used smoothing kernel

$$g_\sigma(x,y) = \frac{1}{\sqrt{2\pi}\sigma}\exp\{-\frac{x^2}{2\sigma^2}\} \cdot \frac{1}{\sqrt{2\pi}\sigma}\exp\{-\frac{y^2}{2\sigma^2}\}$$
$$= \frac{1}{2\pi\sigma^2}\exp\{-\frac{x^2+y^2}{2\sigma^2}\}$$

separable in x and y



- Discretized in space for computation
- Coefficient values are sometimes rounded to integer (for efficiency)
- Amount of smoothing can be controlled by parameter σ (large σ requires large matrix size)

Frequency-domain understanding

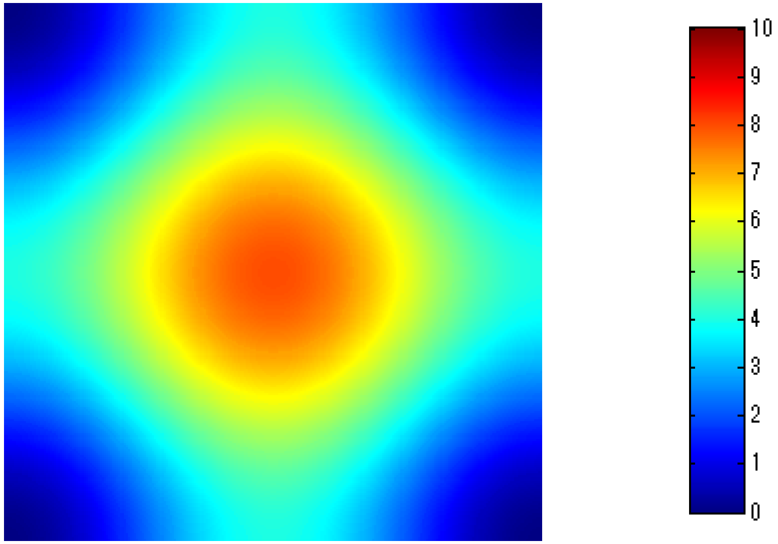
$$G_{x,y} = \sum_{i=-N}^{N} \sum_{j=-N}^{N} w_{i,j} F_{x+i,y+j}$$

$$= w_{-x,-y} * F_{x,y} \xrightarrow{\mathcal{F}} \mathcal{F}\left[w_{-x,-y}\right] \cdot \mathcal{F}\left[F_{x,y}\right]$$

$\mathcal{F}[\cdot]$: 2-D discrete Fourier transform

0	1	0
1	4	1
0	1	0

(zero-padded to 256x256 and) $\xrightarrow{\mathcal{F}}$



Recall: Fourier transform of Gaussian function is Gaussian

Why Gaussian is preferred for smoothing

Several explanations are possible. For example:

- Receptive fields in mammalian retina and visual cortex have similar characteristics
- Gaussian is ideal in that the product of standard deviations in spatial and frequency domains is minimal [Marr and Hildreth 1980]
- When convolved to 1D signals, Gaussian is the unique kernel that never creates new local extrema and satisfies some other good properties (but not in 2D) [Babaud 1986] [Lindeberg 1994]
- For 2D or more dimensional signals, Gaussian is the unique kernel that is linear, shift invariant, scale invariant, isotropic, separable and has a semi-group structure [Florack et al. 1992]

Florack's proof (simplified)

p(**x**): original image, p(**x**, t): smoothed image by amount t = t(σ),
σ: scale parameter with dimension of length

Linear, shift invariant $\Rightarrow p(\boldsymbol{x};t) = h(\boldsymbol{x};t) * p(\boldsymbol{x}),\ \ P(\boldsymbol{\omega};t) = H(\boldsymbol{\omega};t)P(\boldsymbol{\omega})$

Scale invariant $\Rightarrow$ H must be described by normalized frequency, and t must be homogeneous function of σ: $H(\boldsymbol{\omega};t) = \widehat{H}(\sigma\boldsymbol{\omega}),\ \ t = \sigma^p$

Istropic $\Rightarrow$ described by the magnitude (Euclidean norm) of frequency

$$\widehat{H}(\sigma\boldsymbol{\omega}) = \tilde{H}(\sigma^p||\boldsymbol{\omega}||^p)$$

Semi-group structure $\Rightarrow$ successive smoothing by two filters is represented by another smoothing filter

$$H(\boldsymbol{\omega};t_1)H(\boldsymbol{\omega};t_2) = H(\boldsymbol{\omega};t_1+t_2)$$

$$\tilde{H}(v_1)\tilde{H}(v_2) = \tilde{H}(v_1+v_2) \text{ where } v_i = \sigma_i^p||\boldsymbol{\omega}||^p$$

$$\tilde{H}(v_1) = \exp(-av_1) = \exp(-a\sigma^p||\boldsymbol{\omega}||^p) \text{ where } a > 0$$

Separable $\Rightarrow \exp(-a\sigma^p||\boldsymbol{\omega}||^p) = \prod_i \exp(-a\sigma^p|\omega_i|^p) = \exp(-a\sigma^p\sum_i|\omega_i|^p)$

$$(\sum_i \omega_i^2)^{p/2} = \sum_i |\omega_i|^p, \text{ thus } p \doteq 2$$

Edge Detection

- Spatial differentiation (approximated by finite difference)

0	0	0
-1	0	1
0	0	0

1st order diff. in x direction

0	-1	0
0	0	0
0	1	0

1st order diff. in y direction

- Often combined with smoothing:

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

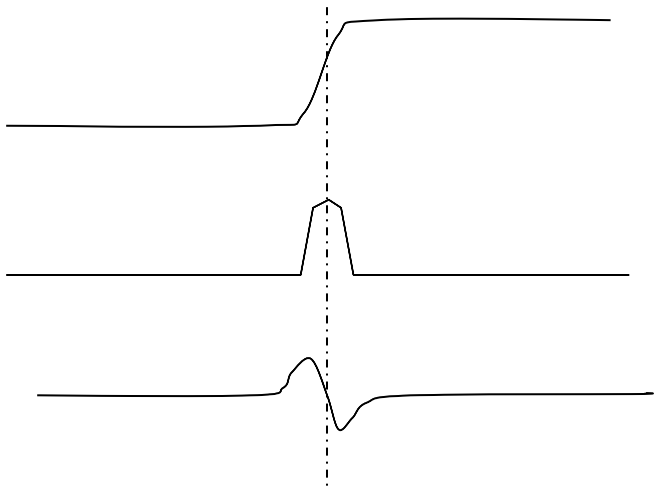
Sobel filter in x direction

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

Sobel filter in y direction

Edge detection by 2nd order derivative

- Edge = zero crossing of 2nd order derivative
- Laplacian $\partial^2/\partial x^2 + \partial^2/\partial y^2$ is the lowest-order isotropic differential operator
  - does not depend on direction of edges

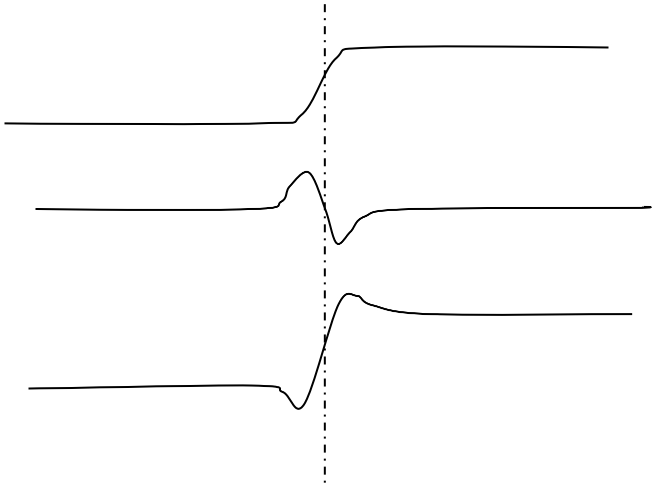


- Laplacian operator is realized by adding 2nd order differentials $f_{i+1} - 2 f_i + f_{i-1}$ of x and y directions

0	1	0
1	–4	1
0	1	0

Sharpening

Subtract the Laplacian image from the original image to yield an edge-enhanced image



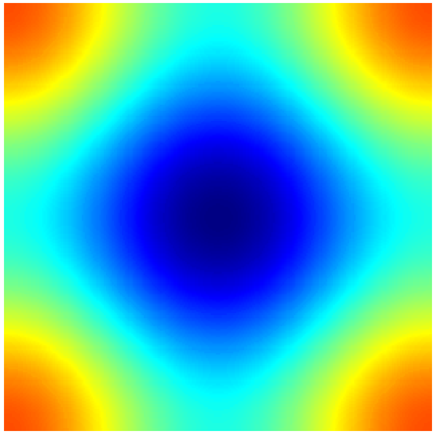
0	0	0		0	1	0		0	-1	0
0	1	0	–	1	–4	1	=	-1	5	-1
0	0	0		0	1	0		0	-1	0

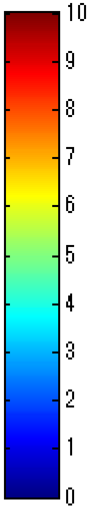
Frequency-domain visualization

Laplacian:

0	1	0
1	-4	1
0	1	0

$\xrightarrow{\mathcal{F}}$

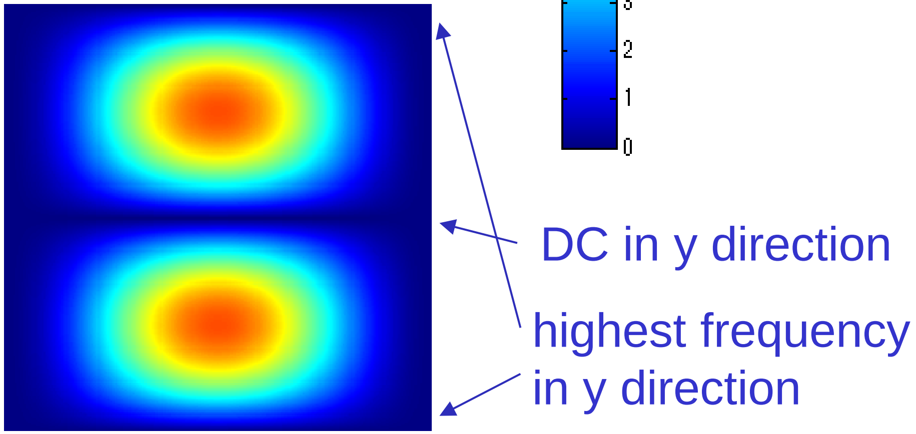




Sobel:

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

$\xrightarrow{\mathcal{F}}$



Q: Why is Sobel a band-pass filter instead of high-pass?

Summary

- Image to Image processing
  - Point operations
    - e.g.: pixel value conversion
    - concept of histogram
  - Local operations
    - linear spatial filters (cf. nonlinear filters)
      - smoothing
      - edge detection
      - sharpening
    - frequency domain understanding
    - uniqueness of Gaussian

References

- R. Szeliski: Computer Vision: Algorithms and Applications, Springer, 2010.
- A. Hornberg eds.: Handbook of Machine Vision, Wiley-VCH, 2006.
- G. Bradski and A. Kaebler: Learning OpenCV, O‘Reilly, 2008.
- OpenCV Documentation: http://docs.opencv.org/index.html

- T. Lindeberg: Scale-space theory: A basic to ol for analysing structures at different scales, J. Applied Statistics, vol. 21, no. 2, pp.225-270, 1994.
- L. M. J. Florack, B. M. T. Romeny, J. J. Koenderink and M. A. Viergever: Scale and the Differential Structure of Images, Image and Vision Computing, vol.10, no.6, pp.376-388, 1992.

(in Japanese)
- ディィジタル画像処理編集委員会, ディィジタル画像処理, CG-ARTS協会, 2015.
- 田村: コンピュータ画像処理, オーム社, 2002.

Sample codes are available at
http://www.ic.is.tohoku.ac.jp/~swk/lecture/