WASEDA UNIVERSITY

Practical work of Image Processing

Hideyuki Sawada, Ph.D.
Professor
Waseda University
sawada@waseda.jp

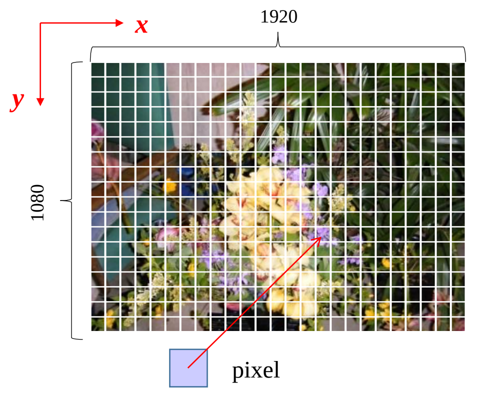
Disciplinary Background

1. Linear Algebra
2. Signals and Systems
3. Digital Signal Processing
4. Probability Theory and Random Process
5. C/C++, Python Programming Skill, Matlab

Computational Intelligence
Machine Learning
Data Science .....

Description of Image Data

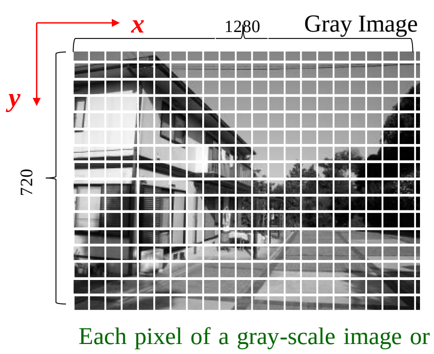
Image Size: 1920 × 1080, 1280 × 720, 640 × 480, 320 × 240 .....

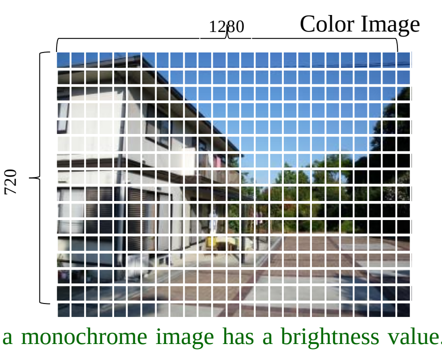


When you take a picture using a digital camera or a mobile phone, the image size is described such as 1280x720 and 1920x1080. It means that the image taken by the camera consists of the 1920x1080 picture elements. A picture is divided into 1920x1080 small pixels.

WASEDA UNIVERSITY
Physics, Informatics and Robotics

Description of Image Data



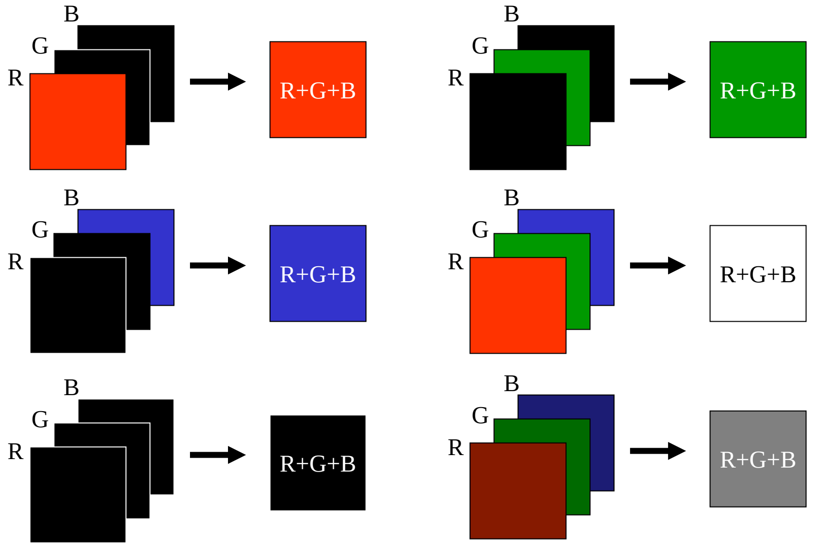


Each pixel of a gray-scale image or a monochrome image has a brightness value. In a computer, the brightness value is expressed by an integer from 0 to 255, where 0 is black and 255 is white.

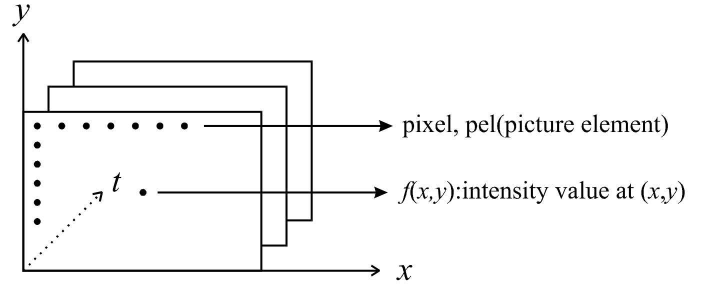
In a color image, each pixel consists of 3 data. There are different ways to express color data, however RGB expression is widely used. R means red color, G means green color and B means blue color, which are the three primary colors of light. The three values of RGB colors are expressed also by the number from 0 to 255. Any color that can be displayed in a color display is expressed by the combination of RGB data.

Japanese French Week on Mechatronics, Sep. 16th 2024

Expression of Color Image Data



Description of Image Data



picture size $\Rightarrow$ picture resolution ; 640×480, 1980×1080

$0 \leq f(x, y) \leq L\ (= 255)$; gray level, 8bit/pixel

(x, y) ; spatial coordinate

t ; temporal coordinate

Algebraic Description

$$f(x,y)=\begin{bmatrix} f(0,0) & f(0,1) & \cdots & f(0,N-1) \\ f(1,0) & f(1,1) & \cdots & f(1,N-1) \\ \vdots & & & \\ f(M-1,0) & \cdots & \cdots & f(M-1,N-1) \end{bmatrix} \qquad M \times N \text{ matrix}$$

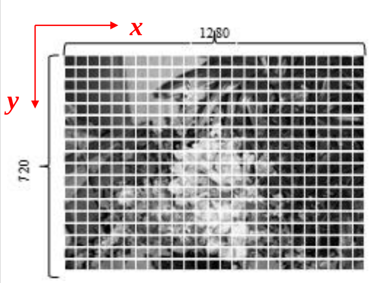
$$\mathbf{A}=\begin{bmatrix} a_{00} & a_{01} & \cdots & a_{0,N-1} \\ \vdots & & & \\ \vdots & & & \\ a_{M-1,0} & \cdots & \cdots & a_{M-1,N-1} \end{bmatrix} \qquad a_{ij}= f\,(x=i,\, y=j)$$

WASEDA UNIVERSITY
Physics, Informatics and Robotics

Image Description in C-program

Image data are described as 2 dimensional array, which elements have values from 0 to 255. As a gray image, typical expression is as follows;

unsigned char image-name[y-pixel-size][x-pixel-size];



Left image presents 1280x720 pixels picture in gray scale values, and the description of the image is;

unsigned char flower[720][1280];

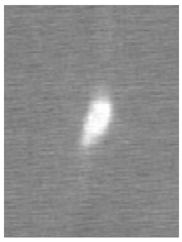
or

unsigned char image_name[HEIGHT][WIDTH];

where HEIGHT is 720 and WIDTH is 1280.

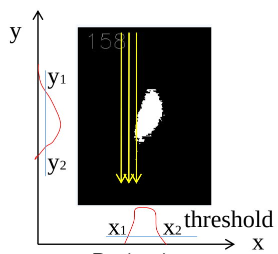
For color images, each pixel consists of three colors, which are R, G and B that represent Red, Green and Blue, respectively.

Projection for the Localization of an Object



Original image

Binarized image



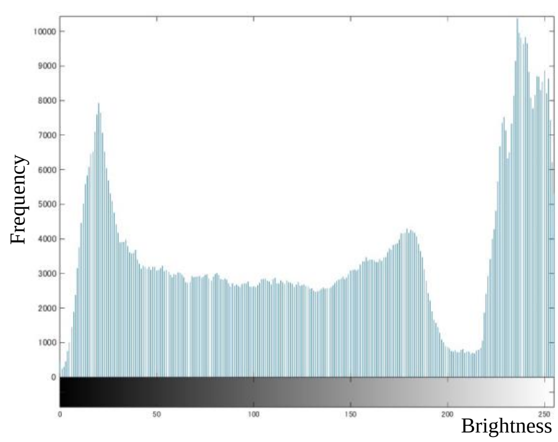
Projection

"Projection" is the operation to count white pixels along x-axis and y-axis. The counting result shows the distribution of white pixels. The area that exceeds a certain threshold should be the position, or the region of the white blob.

We can say that a white blob exists in the area between (x1 and x2) and (y1 and y2).

Histogram

I = imread("Temple.jpg");
imshow(I);



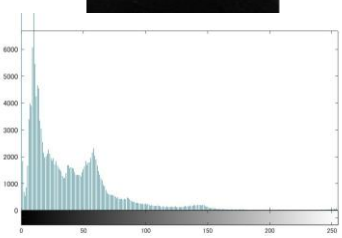
imhist(I);

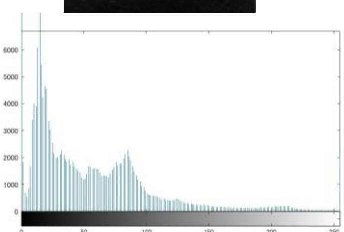
imwrite(I, "Histogram.png")

Adjusting Image Contrast

imadjust(gs);

Non-linear adjustment



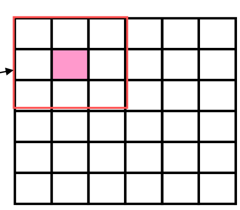


WASEDA UNIVERSITY
Physics, Informatics and Robotics

Image Filtering by Convolutional Operations

Averaging/Smoothing/Uniform filter

$$h = \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix} = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



H = fspecial("average", 3)

Image data

$W_{mn}=$

$W_{-1,-1}$	$W_{0,-1}$	$W_{+1,-1}$
$W_{-1,0}$	$W_{0,0}$	$W_{+1,0}$
$W_{-1,+1}$	$W_{0,+1}$	$W_{+1,+1}$

Kernel

$$F_{ij} = \sum_{m=-k}^{+k} \sum_{n=-k}^{+k} W_{mn} * f_{i+m,j+n}$$

Operation

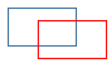
Japanese French Week on Mechatronics, Sep. 16th 2024

Geometric translation

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} S_{11} & S_{12} & S_{13} \\ S_{21} & S_{22} & S_{23} \\ S_{31} & S_{32} & S_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

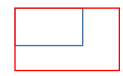
Shift in *x*- or *y*-direction

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{pmatrix}$$



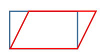
Scale translation

$$\begin{pmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



Shear deformation

$$\begin{pmatrix} 1 & \tan\theta_y & 0 \\ \tan\theta_x & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



Rotation

$$\begin{pmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

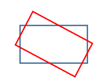
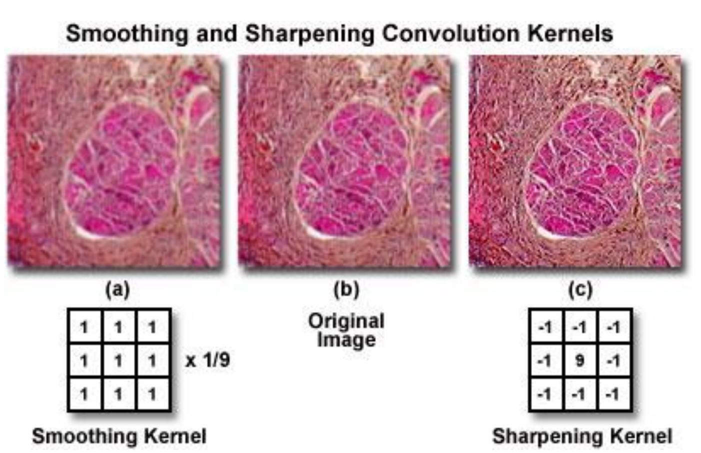


Image Filtering by Convolutional Operations



microscopy.fsu.edu

Edge Detection

WASEDA UNIVERSITY
Physics, Informatics and Robotics

Sobel operator

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$h_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$h_{xy} = \left(h_x^2 + h_y^2\right)^{\frac{1}{2}}$$

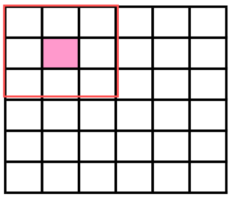


Image data

H = fspecial(“sobel”)

Edge Detection

WASEDA UNIVERSITY
Physics, Informatics and Robotics

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad h_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50

$$h_x = \begin{array}{l} -1*10 + 0*10 + 1*10 \\ +(-2*10) + 0*10 + 2*10 \\ -1*10 + 0*10 + 1*10 \end{array}$$

$$h_x = \begin{array}{l} -1*10 + 0*10 + 1*50 \\ +(-2*10) + 0*10 + 2*50 \\ -1*10 + 0*10 + 1*50 \end{array}$$

$$h_{xy} = \left(h_x^2 + h_y^2\right)^{\frac{1}{2}}$$

0	160	160	0
0	160	160	0
0	160	160	0
0	160	160	0
0	160	160	0
0	160	160	0

h_x

EdgeDetect = imfilter(GrayImage, H);

Edge Detection

$$h_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad h_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50
10	10	10	50	50	50

$$h_x = \begin{array}{l} -1*10+0*10+1*10 \\ +(-2*10)+0*10+2*10 \\ -1*10+0*10+1*10 \end{array}$$

$$h_x = \begin{array}{l} -1*10+0*10+1*50 \\ +(-2*10)+0*10+2*50 \\ -1*10+0*10+1*50 \end{array}$$

$$h_{xy} = \left(h_x^2 + h_y^2\right)^{\frac{1}{2}}$$

0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

h_y

EdgeDetect = imfilter(GrayImage, H);

Edge Detection

Original picture

Grayscale

Edge detection

Practical work with Matlab

Image Processing Toolbox
Statistics and Machine Learning Toolbox

WASEDA UNIVERSITY

https://www.mathworks.com/products.html

Physics, Informatics and Robotics

(A) Find the location of an object in a picture.

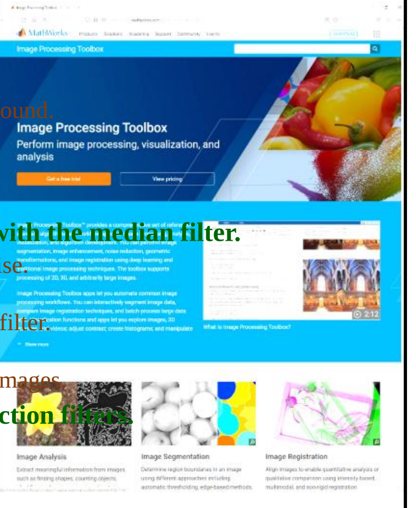
1) Take a picture of some object using your camera.
2) Send the picture to the programing environment.
3) Set a threshold to separate the object from background.
4) Binarize the image.
5) Execute the projection to x- and y- direction.
6) Calculate the center coordinate of the object.

(B) Compare the effect of the averaging filter with the median filter.

1) Take a picture of some texture or with spiking noise.
2) Send the picture to the programing environment.
3) Find the averaging/uniform filter and the median filter.
4) Apply the filters to the picture.
5) Examine the difference between the two filtered images.

(C) Examine the effects of different edge-extraction filters.

1) Take a picture of some objects having textures.
2) Send the picture to Matlab.
3) Find different edge-detection filters.
4) Apply the filters to the picture.
5) Examine the difference among the filtered images.



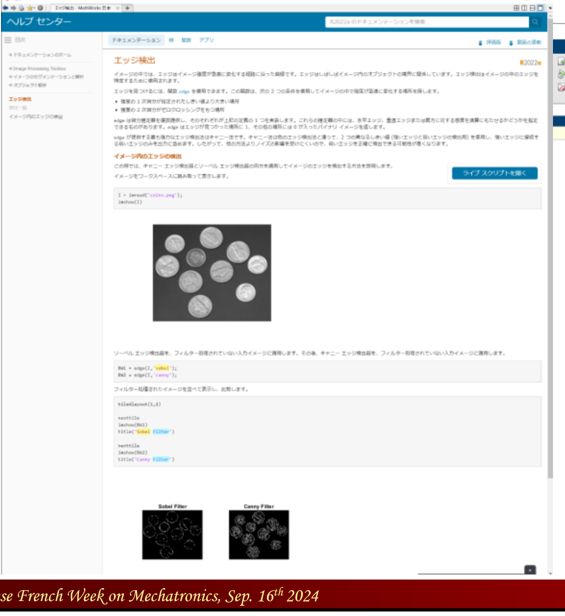
Practical work with Matlab

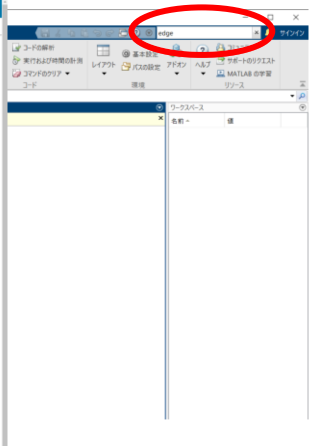
Image Processing Toolbox
Statistics and Machine Learning Toolbox

WASEDA UNIVERSITY

https://www.mathworks.com/products.html

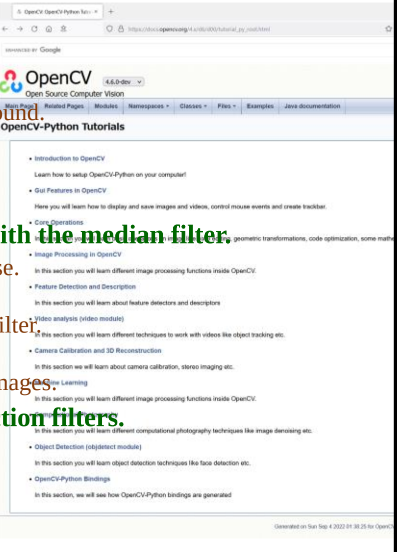
Physics, Informatics and Robotics





Practical work with OpenCV-Python

https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html



(A) Find the location of an object in a picture.

1) Take a picture of some object using your camera.
2) Send the picture to the programing environment.
3) Set a threshold to separate the object from background.
4) Binarize the image.
5) Execute the projection to *x*- and *y*- direction.
6) Calculate the center coordinate of the object.

(B) Compare the effect of the averaging filter with the median filter.

1) Take a picture of some texture or with spiking noise.
2) Send the picture to the programing environment.
3) Find the averaging/uniform filter and the median filter.
4) Apply the filters to the picture.
5) Examine the difference between the two filtered images.

(C) Examine the effects of different edge-extraction filters.

1) Take a picture of some objects having textures.
2) Send the picture to Matlab.
3) Find different edge-detection filters.
4) Apply the filters to the picture.
5) Examine the difference among the filtered images.

Presented by
Hideyuki Sawada Laboratory
Waseda University