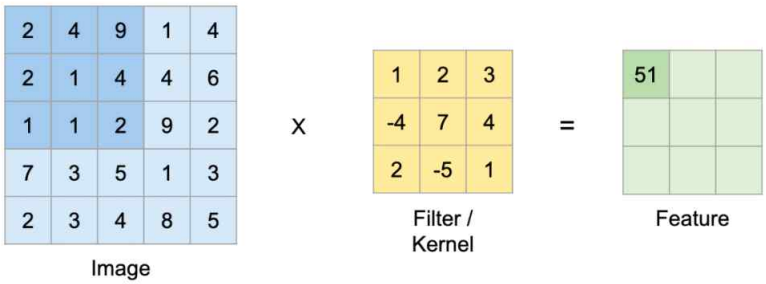
Chap. 3 OpenCV를 이용한 기본 영상처리 기법들

► 수업 내용

- Convolution을 이용한 기본 영상처리 기법들
- OpenCV에서 트랙바(Trackbar)를 사용한 파라미터 변화에 따른 영상처리 효과 확인하기
- Canny 에지 연산, 이진연산

1. Convolution (=이미지 필터링)

- 2차원 이미지에 대하여 매 화소당 커널값을 곱한 후 더하는 방식으로 화소의 새로운 값을 구함으로써 원하는 영상처리를 하는 기법



$$I * K = \sum_i \sum_j I(i,j) * K(i-k, j-l)$$

I: 원력영상. K: kernel

- 커널을 구성하기에 따라 다양한 연산이 가능하고, 병렬처리가 가능하다

- 커널(kernel)은 특정한 연산을 위한 특징 추출기, Highpass/Lowpass 필터로서 역할한다
  - · LowPass Filtering의 대표적인 예: Blurring
  - · HighPass Filtering의 예: Edge Detection

- Convolution을 이용한 연산의 특징
  - · 커널값만 바꾸면 동일한 간단한 방법으로 원하는 영상처리를 할 수 있다
  - · 의미있는 커널값을 구하는 것이 관건
  - · 커널의 크기가 커지면 연산량이 늘어날 수 있다

- 딥러닝 알고리즘에 있어서 핵심적인 연산 방법이다

- Convolution을 이용한 다양한 영상처리의 예 (샤프닝, 엠보싱, 이진화 등 다양)

0	−1	0
−1	5	−1
0	−1	0

−1/9	−1/9	−1/9
−1/9	8/9	−1/9
−1/9	−1/9	−1/9

Sharpening Filter

−1	0	0
0	0	0
0	0	1

or

0	0	−1
0	0	0
1	0	0

Embossing Filter

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

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

Blurring Filter

- 칼라영상(다채널)에 대한 처리시 처리 방법에 유의해야 함
- 영상의 바깥경계(가장자리)영역에 대한 처리 방법이 필요
    · Zero Padding, 경계값 무시, Border Enlarging(복사), Border Wrapping

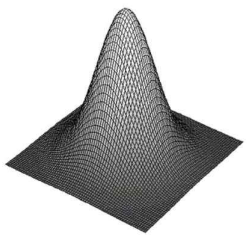
2. 에지 연산

- 윤곽(Edge)는 영상의 많은 정보를 포함하고 있어서 영상처리의 중요한 기법/대상으로 사용된다

- Canny Edge 탐색이 가장 대표적인 탐색법이고,
  연산전에 잡음제거를 위하여 Blurring을 하는 것이 효과적이다

$\frac{1}{273}$

1	4	7	4	1
4	16	26	16	4
7	26	41	26	7
4	16	26	16	4
1	4	7	4	1



(https://webnautes.tistory.com/1255)

- 알고리즘 개요

1. Filter out any noise. The Gaussian filter is used for this purpose. An example of a Gaussian kernel of $size = 5$ that might be used is shown below:

$$K = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

2. Find the intensity gradient of the image. For this, we follow a procedure analogous to Sobel:

   a. Apply a pair of convolution masks (in x and y directions:

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

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

   b. Find the gradient strength and direction with:

$$G = \sqrt{G_x^2 + G_y^2}$$

$$\theta = \arctan\left(\frac{G_y}{G_x}\right)$$

   The direction is rounded to one of four possible angles (namely 0, 45, 90 or 135)

3. *Non-maximum* suppression is applied. This removes pixels that are not considered to be part of an edge. Hence, only thin lines (candidate edges) will remain.

4. *Hysteresis*: The final step. Canny does use two thresholds (upper and lower):

   a. If a pixel gradient is higher than the *upper* threshold, the pixel is accepted as an edge

   b. If a pixel gradient value is below the *lower* threshold, then it is rejected.

   c. If the pixel gradient is between the two thresholds, then it will be accepted only if it is connected to a pixel that is above the *upper* threshold.

Canny recommended a *upper:lower* ratio between 2:1 and 3:1.

(https://docs.opencv.org/2.4/doc/tutorials/imgproc/imgtrans/canny_detector/canny_detector.html)

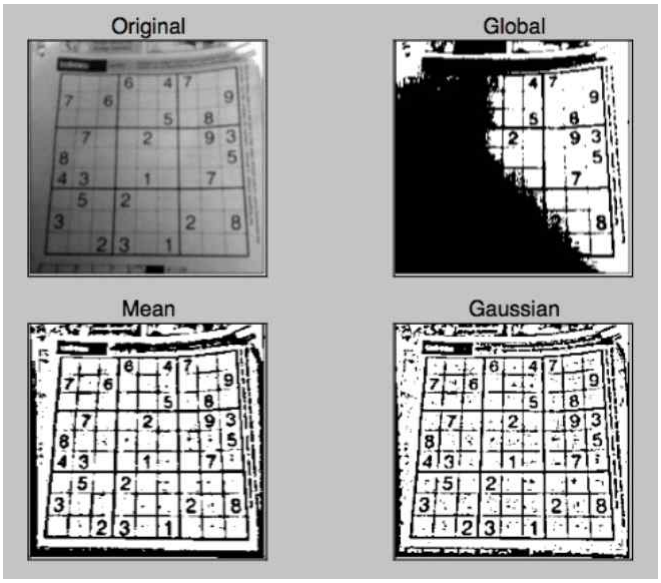
3. 이진화 연산

- OCR과 같은 문서인식이나 정보량을 줄이고 싶을 때 사용
- 전역적 이진화 : 한 개의 임계값 사용 방법 -> cv2.threshold
- 적응적 이진화 : 각 화소 주변의 밝기값 분포를 이용 ->cv2.adaptiveThreshold

`cv2.adaptiveThreshold(src, maxValue, adaptiveMethod, thresholdType, blockSize, C)`

Parameters:

- **src** – grayscale image
- **maxValue** – 임계값
- **adaptiveMethod** – thresholding value를 결정하는 계산 방법
- **thresholdType** – threshold type
- **blockSize** – thresholding을 적용할 영역 사이즈
- **C** – 평균이나 가중평균에서 차감할 값



4. 실습

(ip) d:\ip>python iprocessing_basic.py