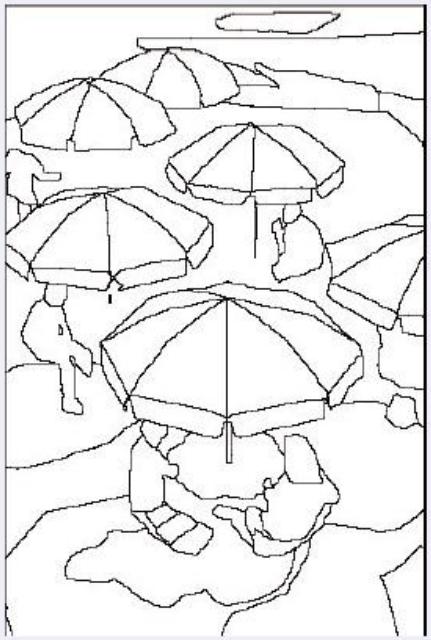
에지와 특징(1)

4.1 에지

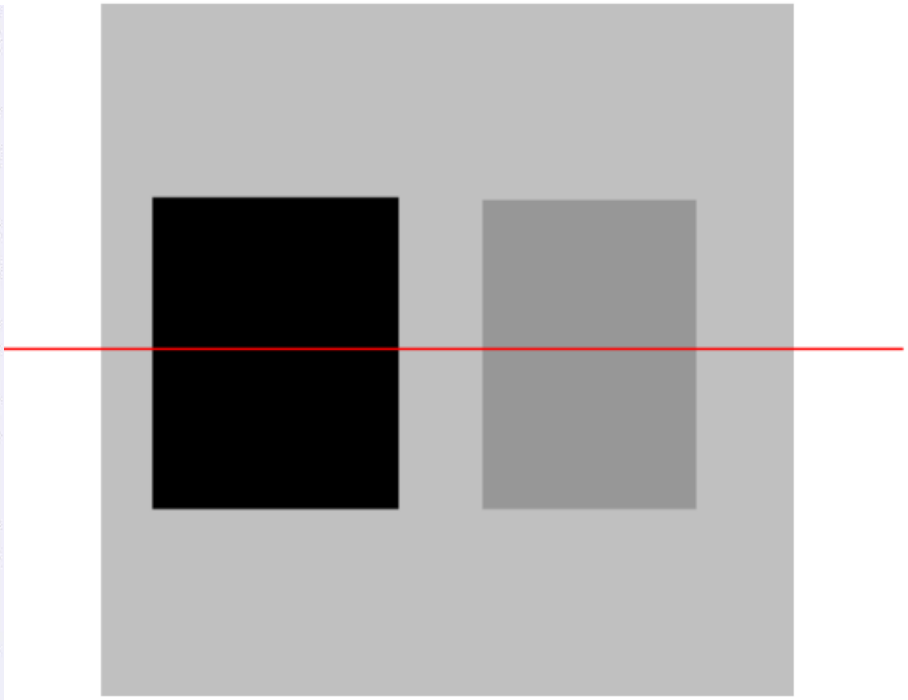
에지(edge, 경계선)

- 에지(edge)는 물체의 가장 바깥 부분의 둘레를 의미하며, 객체의 테두리에 해당함
- 경계는 전경(foreground)과 배경(background)이 구분되는 지점이며, 전경과 배경 사이에서 밝기가 큰 폭으로 변하는 지점이 객체의 경계선
- 에지는 픽셀의 밝기가 급격하게 변하는 부분으로 간주하면, 에지를 찾기 위해 미분(derivative)과 기울기(gradient) 연산을 수행하며, 영상 상에서 픽셀의 밝기 변화율이 높은 부분을 에지로 찾아냄

(a) 원래 영상

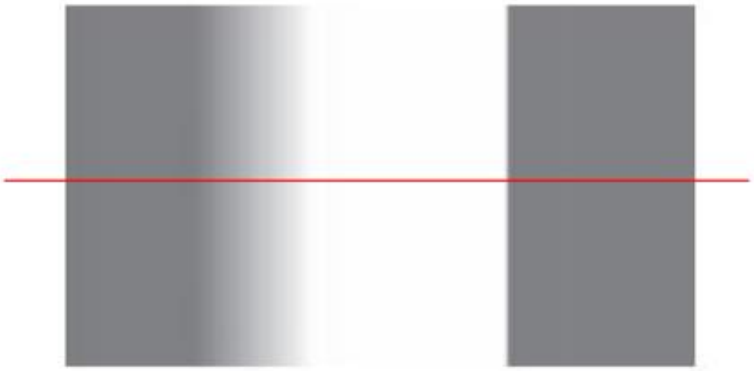


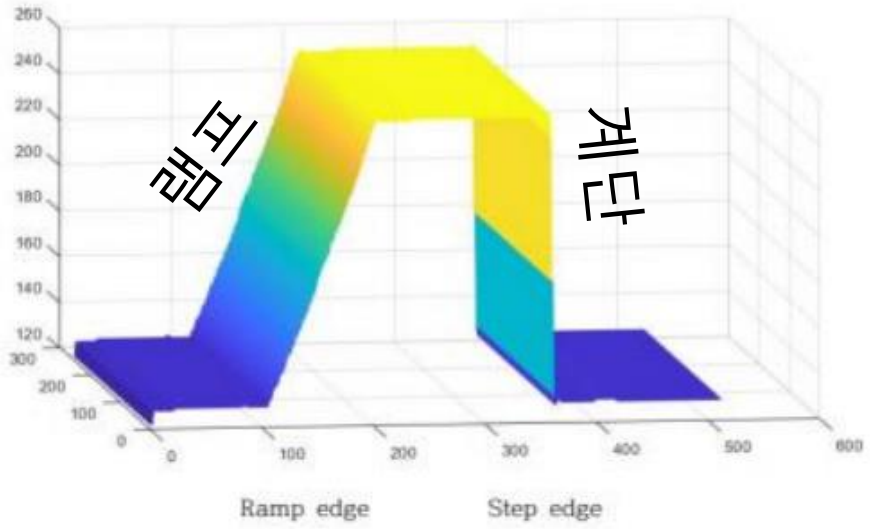
(b) 영역 영상(사람이 분할)



4.1 에지

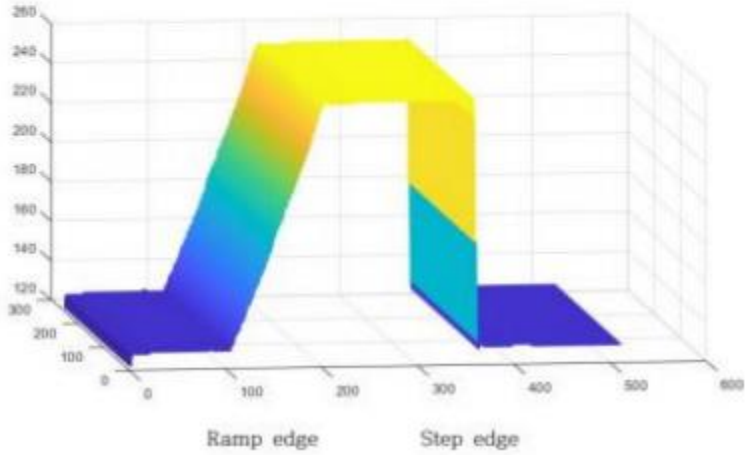
- 에지 종류
  - 밝기 값이 완만히 변하는 램프(ramp) 에지와 급격히 변하는 계단(step) 에지로 구분
  - 자연 영상에서는 주로 램프 에지가 나타남

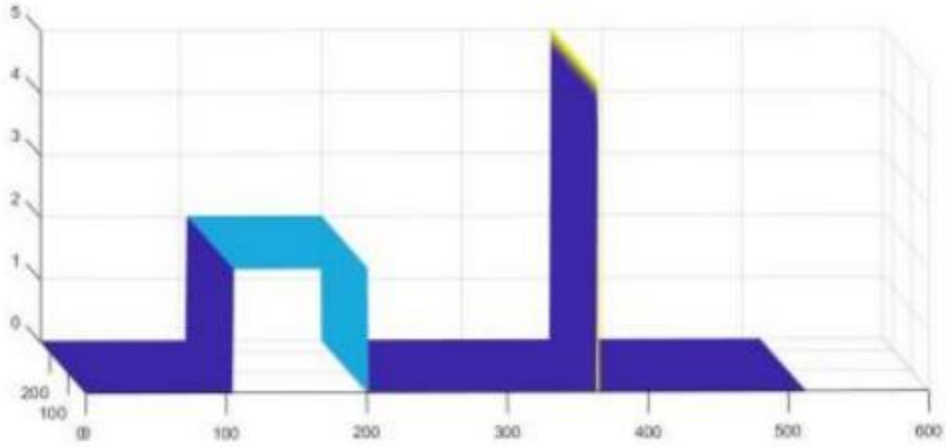




4.1 에지

- 에지 미분
  - 입력 영상을 x 축의 방향으로 1차 미분하면 램프 에지는 경사 부분에서 미분값이 높게 나타나고, 스텝 에지는 밝기값이 변하는 부분에서 미분값이 매우 높게 나타남





4.1 에지

미분

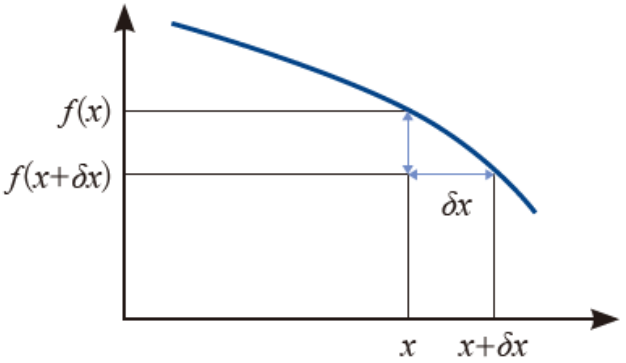
- 변수 x가 미세하게 증가했을 때 함수 변화량을 측정하는 수학 기법

$$f'(x)=\lim_{\delta x\to 0}\frac{f(x+\delta x)-f(x)}{\delta x}$$

- 미분을 디지털 영상에 적용하면,

$$f'(x)=\frac{f(x+\delta x)-f(x)}{\delta x}=f(x+1)-f(x)$$

- 실제 구현은 필터 u로 컨볼루션 (u를 에지 연산자라 부름)



(a) 연속 함수의 미분

(b) 디지털 영상의 미분(필터 u로 컨볼루션)

4.1 에지

- 디지털 영상의 1차 미분 정의
  - 입력 영상에 x축 방향과 y축 방향에서의 1차 미분의 정의

$$\nabla s(i,j) = \frac{\partial s(i,j)}{\partial x} = \frac{s(i+\Delta x, j) - s(i,j)}{\Delta x}$$

$$\nabla s(i,j) = \frac{\partial s(i,j)}{\partial y} = \frac{s(i,j+\Delta y) - s(i,j)}{\Delta y}$$

  - 일반적으로 영상에서 Δx = 1 (1픽셀 기준) : 우측/하단 픽셀 – 현재 픽셀

4.2 Roberts & Prewitt 에지

- 로버츠(Roberts)와 프리윗(Prewitt) 필터
  - 에지 검출을 위한 1차 미분 마스크 형태를 정의
  - 미분은 공간 기반 처리의 대표적인 컨볼루션을 통하여 수행

Roberts

-1	0
0	1

0	-1
1	0

2x2 형태 Original Roberts Mask

-1	0	0
0	1	0
0	0	0

0	0	-1
0	1	0
0	0	0

3x3 형태 컨볼루션 마스크

Prewitt

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

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

y 방향 및 x 방향 마스크

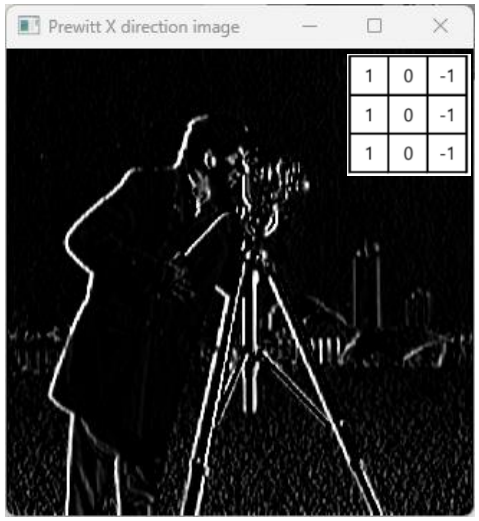
4.2 Roberts & Prewitt 에지

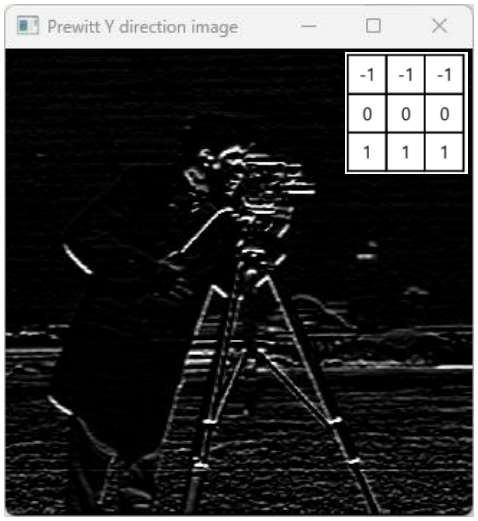
- 회색조 영상에 대하여 로버츠(Roberts)와 프리윗(Prewitt) 필터 적용

```
import numpy as np
import cv2

img = cv2.imread('./images/cameraman.jpg',cv2.IMREAD_REDUCED_GRAYSCALE_2)

roberts_x = np.array([[0, 0, -1], [0, 1, 0], [0, 0, 0]])
roberts_y = np.array([[-1, 0, 0], [0, 1, 0], [0, 0, 0]])

prewitt_x = np.array([[1, 0, -1], [1, 0, -1], [1, 0, -1]])
prewitt_y = np.array([[-1, -1, -1], [0, 0, 0], [1, 1, 1]])

r_imageX = cv2.convertScaleAbs(cv2.filter2D(img, -1, roberts_x))
r_imageY = cv2.convertScaleAbs(cv2.filter2D(img, -1, roberts_y))

p_imageX = cv2.convertScaleAbs(cv2.filter2D(img, -1, prewitt_x))
p_imageY = cv2.convertScaleAbs(cv2.filter2D(img, -1, prewitt_y))

cv2.imshow('Original image', img)
cv2.imshow('Roberts X direction image', r_imageX)
cv2.imshow('Roberts Y direction image', r_imageY)
cv2.imshow('Prewitt X direction image', p_imageX)
cv2.imshow('Prewitt Y direction image', p_imageY)

cv2.waitKey(0)
```

4.2 Roberts & Prewitt 에지

- 회색조 영상에 대하여 로버츠(Roberts)와 프리윗(Prewitt) 필터 적용





4.3 소벨 필터

- 소벨(Sobel filter) 필터
  - 영상 처리에 있어서 가장 많이 사용되는 1차 미분 필터
  - 프리윗 필터와 유사하지만, 중심 픽셀에 가까운 점에서 영향력을 더 높게 고려하도록 설계
  - 중앙 부분의 기울기 변화를 더 크게 반영하면서, 주변 픽셀의 잡음(노이즈) 영향은 덜 받도록 설계

Sobel	-1	-2	-1	1	0	-1	y 방향 및 x 방향 마스크
	0	0	0	2	0	-2	
	1	2	1	1	0	-1	

4.3 소벨 필터

- 소벨 필터링 함수: Sobel()함수
  - 입력된 영상 데이터에 대하여 소벨 필터를 적용

함수명	cv2.Sobel(src, Ddepth, dx, dy[, dst[, ksize[, scale[, delta[, borderType]]]]])
매개변수	- Src 입력 영상 데이터 - Ddepth 출력 영상의 깊이 정밀도 (-1이면 입력 영상과 동일) - dx x 방향 미분 차수 - dy y 방향 미분 차수 - Dst 출력 영상 데이터 - ksize 소벨 마스크 커널 크기 (홀수, 1, 3, 5, 또는 7) - scale 계산된 미분 값에 적용할 스케일 요소 (비율) - delta 오프셋 - borderType 픽셀 외삽(extrapolation) 방법 BORDER_CONSTANT 또는 BORDER_REPLICATE
리턴값	출력 영상 데이터 (numpy.ndarray)

4.3 소벨 필터

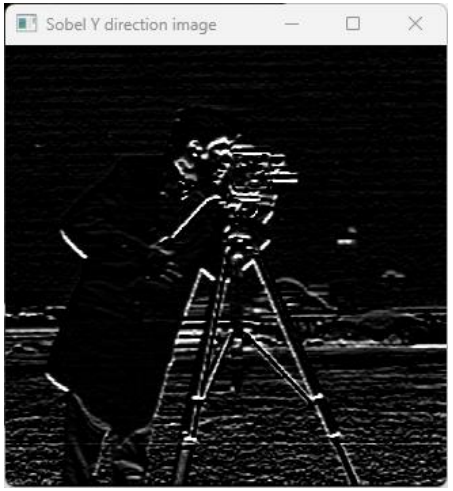
- 회색조 영상에 대하여 소벨 필터(마스크)를 이용하여 x방향 변화율, y방향 변화율, x-y방향 변화율을 계산

```
import cv2

img = cv2.imread('./images/person_dark.jpg',cv2.IMREAD_REDUCED_GRAYSCALE_2)

s_imageX = cv2.Sobel(img, cv2.CV_8U, 1, 0, ksize=3)
s_imageY = cv2.Sobel(img, cv2.CV_8U, 0, 1, ksize=3)
s_imageXY = cv2.Sobel(img, cv2.CV_8U, 1, 1, ksize=3)

cv2.imshow('Original image', img)
cv2.imshow('Sobel X direction image', s_imageX)
cv2.imshow('Sobel Y direction image', s_imageY)
cv2.imshow('Sobel X-Y direction image', s_imageXY)
cv2.waitKey(0)
```

4.3 소벨 필터

- 회색조 영상에 대하여 소벨 필터(마스크)를 이용하여 x방향 변화율, y방향 변화율, x-y방향 변화율을 계산

x 방향 변화율 | y 방향 변화율 | x-y 방향 변화율



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

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

4.4 라플라시안 필터

- 라플라시안 필터의 수학적 계산
  - 2차 미분에 해당
  - x축에 대해 1차 미분 결과를 다시 미분하고, y축에 대해 1차 미분 결과를 다시 미분

$$\nabla^2 s(i,j) = \frac{\partial^2 s(i,j)}{\partial x^2} + \frac{\partial^2 s(i,j)}{\partial y^2}$$

$$s(i+1,j) + s(i-1,j) + s(i,j+1) + s(i,j-1) - 4 \cdot s(i,j)$$

$$\frac{\partial^2 s(i,j)}{\partial x^2}$$
$$= \{s(i+1,j) - s(i,j)\} - \{s(i,j) - s(i-1,j)\}$$
$$= s(i+1,j) + s(i-1,j) - 2 \cdot s(i,j)$$

$$\frac{\partial^2 s(i,j)}{\partial y^2}$$
$$= \{s(i,j+1) - s(i,j)\} - \{s(i,j) - s(i,j-1)\}$$
$$= s(i,j+1) + s(i,j-1) - 2 \cdot s(i,j)$$

4.4 라플라시안 필터

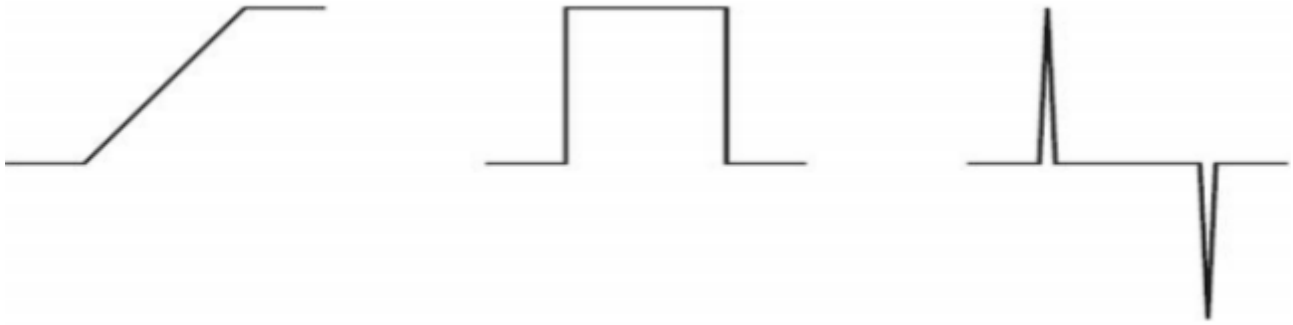
- 라플라시안 필터의 형태
  - 수학적으로 정리하면 마스크로 표현되며, 좌우/상하가 같은 동형(isotropic) 필터에 해당
  - 주변 픽셀과 중심 픽셀의 차이를 계산하는 것으로서 잡음의 경우 오히려 강조되는 특성이 있어서 잡음에 민감한 필터

Laplacian	0	1	0
	1	-4	1
	0	1	0

- Laplacian of Gaussian (LoG)
  - 라플라시안 필터를 적용하기 전에 가우시안 블러링과 같은 처리를 통하여 잡음에 대한 영향을 최소화

4.4 라플라시안 필터

- 라플라시안 필터의 장점
  - 램프 에지는 1차 미분을 수행하면 두껍게 표현되어 정확한 위치를 설정하기 어렵지만, 2차 미분을 수행하면 램프 에지의 시작과 끝 부분에서 차이가 크게 나타나서 정확한 위치를 추정이 용이



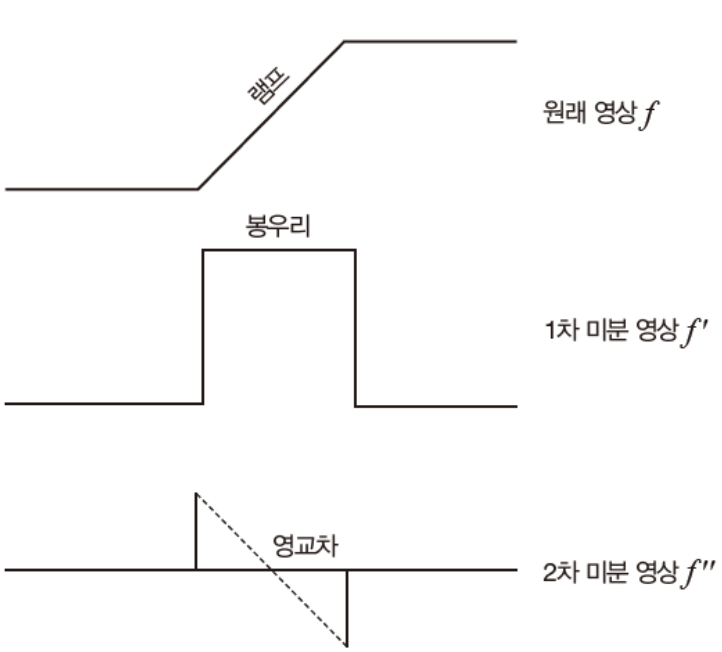
램프 에지 1차 미분 2차 미분

	0	1	2	3	4	5	6	7	8	9	0	1
원래 영상 f	3	3	3	3	5	7	7	7	7	3	3	3
필터 $u=$	-1	1	로 컨볼루션									
1차 미분 f'	0	0	0	2	2	0	0	0	-4	0	0	-
필터 $u=$	-1	1	로 컨볼루션									
2차 미분 f''	0	0	2	0	-2	0	0	-4	4	0	-	-

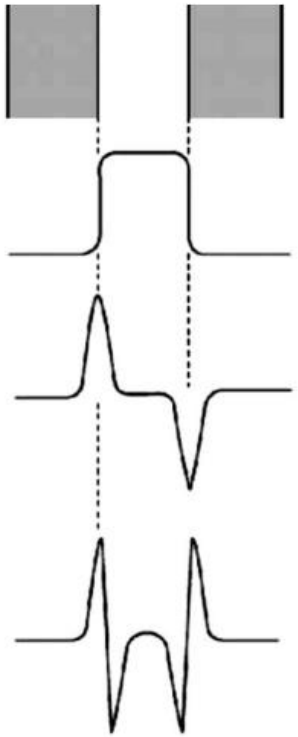
두꺼운 에지로 인해 위치 찾기 문제 발생

4.4 라플라시안 필터

- 라플라시안 필터와 영교차 (Zero-crossing)
  - 램프 에지에 대해서 2차 미분을 수행하게 되면 에지의 시작과 끝 부분에서 피크(peak)가 높게 나타남
  - 하나의 에지에 대해서 2개의 피크가 나타나서 에지의 정확한 위치를 선정하는데 어려움
  - 영교차점은 필터링된 결과에서 +에서 –로 또는 –에서 +로 부호가 변경되는 위치로서, 2차 미분의 결과에서 영교차점을 검출하면 에지의 정확한 위치를 식별



램프 에지에서 발생하는 봉우리와 영교차



4.4 라플라시안 필터

- 라플라시안 필터링 함수: Laplacian() 함수
  - 입력된 영상 데이터에 대하여 라플라시안 필터를 적용

함수명	cv2.Laplacian(src, ddepth[, dst[, ksize[, scale[, delta[, borderType]]]]])
매개변수	- Src 입력 영상 데이터 - ddepth 출력 영상의 깊이 정밀도 (-1이면 입력 영상과 동일) - dst 출력 영상 데이터 - ksize 2차 미분 필터 계산에 사용할 커널 크기 (양수, 홀수) - scale 계산된 미분 값에 적용할 스케일 요소 (비율) - delta 오프셋 - borderType 픽셀 외삽(extrapolation) 방법 BORDER_CONSTANT 또는 BORDER_REPLICATE
리턴값	출력 영상 데이터 (numpy.ndarray)

4.4 라플라시안 필터

- 회색조 영상에 라플라시안 필터(마스크)를 이용하여 경계선을 검출

```
import cv2

img = cv2.imread('../images/person_dark.jpg',cv2.IMREAD_REDUCED_GRAYSCALE_2)

l_image = cv2.Laplacian(img, cv2.CV_8U, ksize=3)

cv2.imshow('Original image', img)
cv2.imshow('Laplacian image', l_image)

cv2.waitKey(0)
```

4.4 라플라시안 필터

- 회색조 영상에 라플라시안 필터(마스크)를 이용하여 경계선을 검출

4.5 샤르 필터

- 샤르(Scharr) 필터
  - 1차 미분에 해당
  - Sobel 필터와 동일하지만 커널 사이즈가 3x3로 고정되어 있으며 Sobel 보다 좀 더 정확하게 적용

Scharr	-3	0	3
	-10	0	10
	-3	0	3

4.5 샤르 필터

- Scharr 필터링 함수: Scharr() 함수
  - 입력된 영상 데이터에 대하여 Scharr 필터를 적용

함수명	cv2.Scharr(src, ddepth, dx, dy[, dst[, scale[, delta[, borderType]]]])
매개변수	- src 입력 영상 데이터 - ddepth 출력 영상의 깊이 정밀도 (-1이면 입력 영상과 동일) - dx x 방향 미분 차수 - dy y 방향 미분 차수 - dst 출력 영상 데이터 - scale 계산된 미분 값에 적용할 스케일 요소 (비율) - delta 오프셋 - borderType 픽셀 외삽(extrapolation) 방법 BORDER_CONSTANT 또는 BORDER_REPLICATE
리턴값	출력 영상 데이터 (numpy.ndarray)

4.5 샤르 필터

- 회색조 영상에 대하여 Scharr 필터(마스크)를 이용하여 x 방향 변화율, y 방향 변화율을 계산하여 출력

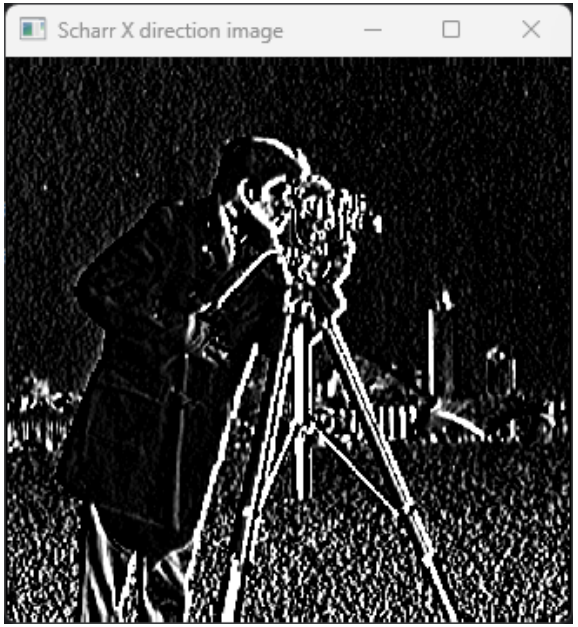
```
import cv2

img = cv2.imread('../images/person_dark.jpg',cv2.IMREAD_REDUCED_GRAYSCALE_2)

s_imageX = cv2.Scharr(img, cv2.CV_8U, 1,0)
s_imageY = cv2.Scharr(img, cv2.CV_8U, 0,1)

cv2.imshow('Original image', img)
cv2.imshow('Scharr X direction image', s_imageX)
cv2.imshow('Scharr Y direction image', s_imageY)

cv2.waitKey(0)
```

4.5 샤르 필터

- 회색조 영상에 대하여 Scharr 필터(마스크)를 이용하여 x 방향 변화율, y 방향 변화율을 계산하여 출력

x 방향 변화율

y 방향 변화율



4.6 케니 에지

- Canny 경계선 검출기(edge detector)
  - 가장 대표적인 경계선 검출(edge detection) 방법
  - 에지를 잘못 찾는 경우를 최소화하기 위한 최소 오류율(거짓 긍정과 거짓 부정이 최소), 검출된 에지의 높은 위치 정확도, 실제 에지에 해당하는 곳의 얇은 에지 두께를 갖는 기준을 충족할 수 있도록 설계한 알고리즘
  - Canny 경계선 검출의 세 가지 목적
    - 낮은 에러율, 경계선의 이동 최소화, 하나만의 경계선만 검출

IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. PAMI-8, NO. 6, NOVEMBER 1986

A Computational Approach to Edge Detection

JOHN CANNY, MEMBER, IEEE

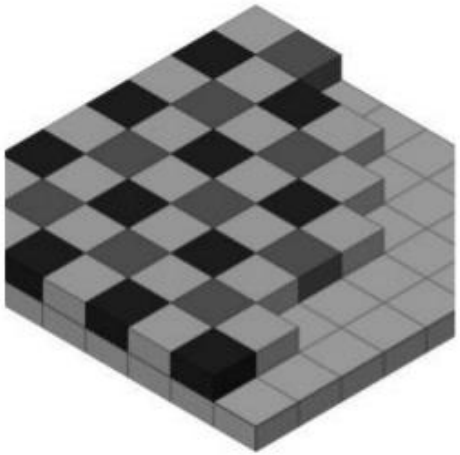
John Canny (S'81–M'82) was born in Adelaide, Australia, in 1958. He received the B.Sc. degree in computer science and the B.E. degree from Adelaide University in 1980 and 1981, respectively, and the S.M. degree from the Massachusetts Institute of Technology, Cambridge, in 1983.

He is with the Artificial Intelligence Laboratory, M.I.T. His research interests include low-level vision, model-based vision, motion planning for robots, and computer algebra.

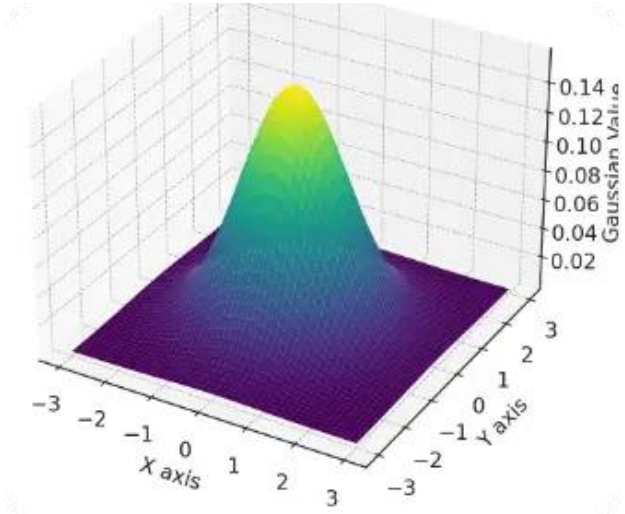
Mr. Canny is a student member of the Association for Computing Machinery.

4.6 케니 에지

- Canny 경계선 검출기(edge detector)
  - 입력 영상에 대하여 여러 단계의 처리 과정을 통해서 경계선 추출
    - 일련의 처리 과정을 통해서 경계값만 남겨두고 나머지를 제거
  - 1단계. Noise Reduction
    - 영상에 존재하는 Noise를 제거 (5x5 크기의 Gaussian filter를 이용)



원본 영상



$$\mathbf{B} = \frac{1}{159}\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} * \mathbf{A}$$

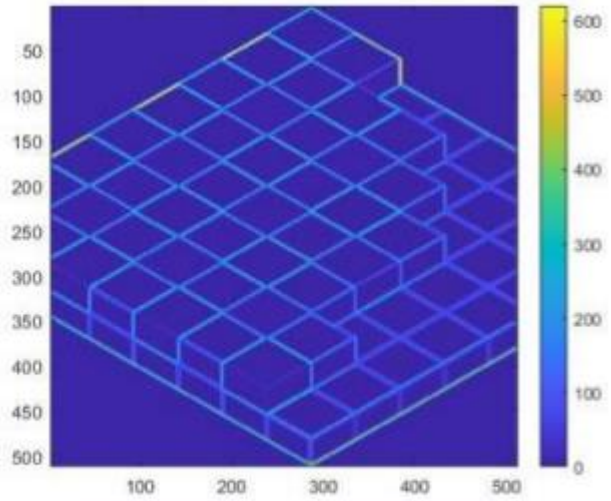
4.6 케니 에지

- Canny 경계선 검출기(edge detector)
  - 2단계. Edge Gradient Detection
    - 영상에서 Gradient의 방향과 강도를 계산

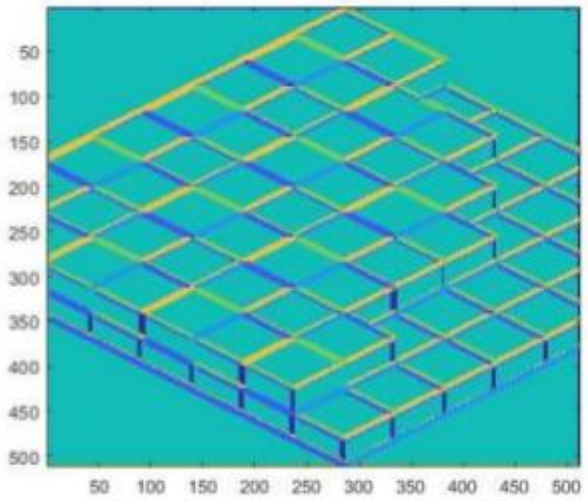
$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

$$\mathbf{\Theta} = \operatorname{atan2}(\mathbf{G}_y, \mathbf{G}_x)$$

    - 경계에서는 주변과 색이 다르기 때문에 미분값이 급속도로 변하게 되므로, 이를 통해 경계값 후보군을 선별



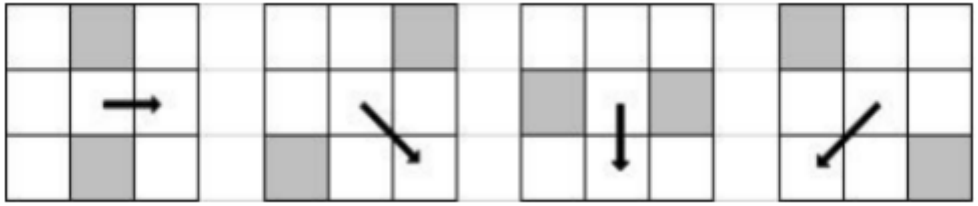
Gradient 강도



Gradient 방향

4.6 케니 에지

- Canny 경계선 검출기(edge detector)
  - 3단계. 비-최대 억제(Non-maximum Suppression)
    - 영상 픽셀을 전체 탐색(scan)하여 경계선(edge)에 해당하지 않는 픽셀 제거
    - 이웃 두 화소보다 에지의 강도(크기)가 크지 않으면 억제됨



※화살표: 에지의 방향

  - 4단계. Hysteresis Thresholding (이력, 과거 거쳐온 상황)
    - 경계선으로 판단된 픽셀이 진짜 경계선인지 판별
    - 최대 임계값 T_{high} 및 최소 임계값 T_{low}를 설정하여 최대 임계값 이상은 강한 경계선, 최소와 최대 임계값 사이는 약한 경계선으로 설정
    - 약한 경계선이 진짜 경계선인지 확인하기 위해서 강한 경계선에서 추적하여 연결이 되어 있으면 경계선으로 판단하고, 그렇지 않은 경우 제거

4.6 케니 에지

- Canny 경계선 검출 함수: Canny() 함수

함수명	cv2.Canny(src, threshold1, threshold2[, edges[, apertureSize[, L2gradient]]])
매개변수	- src 입력 영상 데이터 - threshold1 hysteresis 절차에 사용되는 1번째 임계값 (최소 임계값) - threshold2 hysteresis 절차에 사용되는 2번째 임계값 (최대 임계값) - edges 출력 영상 데이터 (경계선 데이터) - apertureSize Sobel 필터에 사용되는 커널 크기 - L2gradient 영상 Gradient 계산에 L1 norm을 사용할지 L2 norm을 사용할지 설정
리턴값	출력 영상 데이터 (numpy.ndarray)

4.6 케니 에지

- 회색조 영상에 대하여 Canny 경계선 검출기를 이용하여 경계선을 검출

```
import cv2

img = cv2.imread('../images/person_dark.jpg',cv2.IMREAD_REDUCED_GRAYSCALE_2)

c_image1 = cv2.Canny(img, 10, 50)
c_image2 = cv2.Canny(img, 150, 300)

cv2.imshow('Original image', img)
cv2.imshow('Canny image 1', c_image1)
cv2.imshow('Canny image 2', c_image2)

cv2.waitKey(0)
```

4.6 케니 에지

- 회색조 영상에 대하여 Canny 경계선 검출기를 이용하여 경계선을 검출