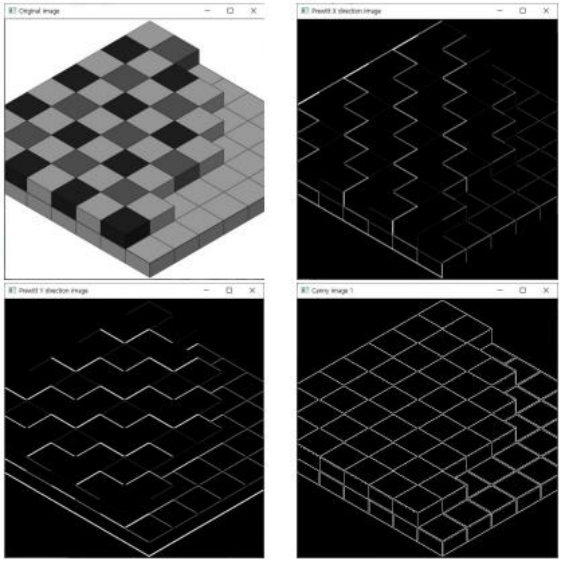
OpenCV 에지 (경계선) 검출

파이썬을 이용한 컴퓨터비전 프로그래밍



Hae-Yeoun Lee

라플라시안 필터 (Laplacian filter)

1. 라플라시안 필터
   A. 2차 미분에 해당
   B. x축에 대해 1차 미분 결과를 다시 미분하고,
      y축에 대해 1차 미분 결과를 다시 미분

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

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

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

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

라플라시안 필터 (Laplacian filter)

1. 라플라시안 필터
   A. 수학적으로 정리하면 마스크로 표현되며, 동형(isotropic) 필터에 해당

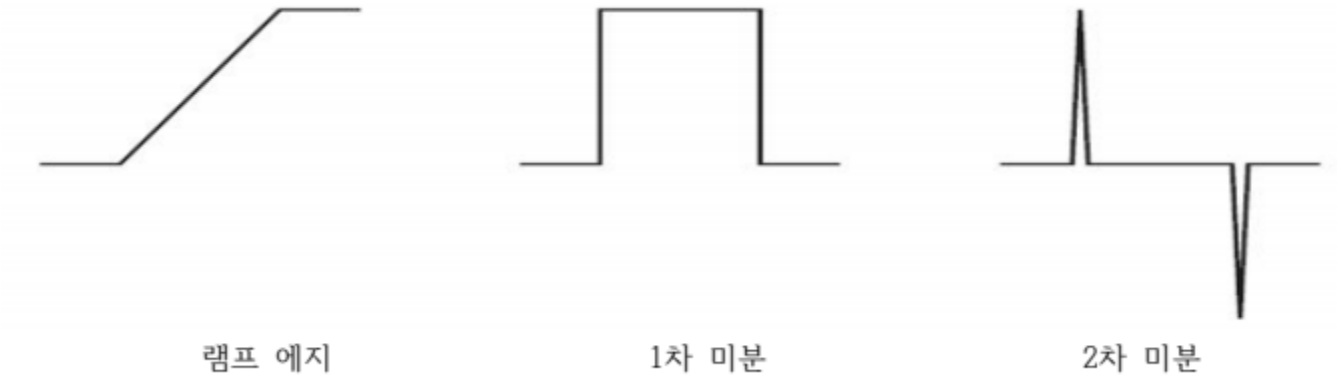
Laplacian	0	1	0
	1	-4	1
	0	1	0

   B. 주변 픽셀과 중심 필셀의 차이를 계산하는 것으로서 잡음의 경우 오히려 강조되는 특성이 있어서 잡음에 민감한 필터

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

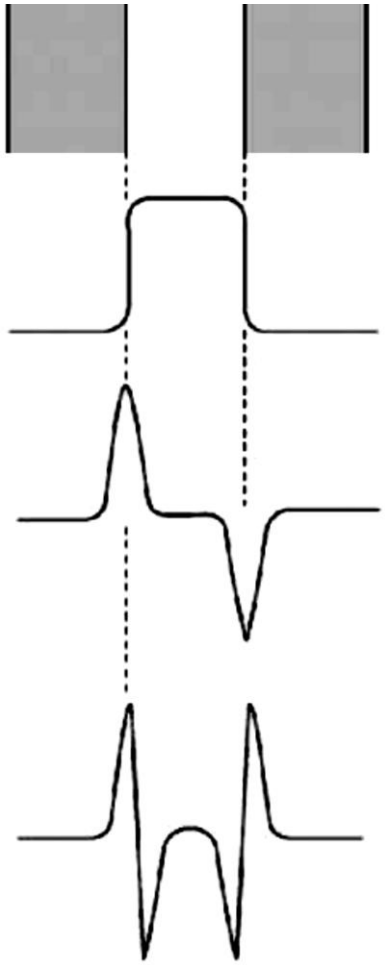
라플라시안 필터 (Laplacian filter)

1. 라플라시안 필터
   A. 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



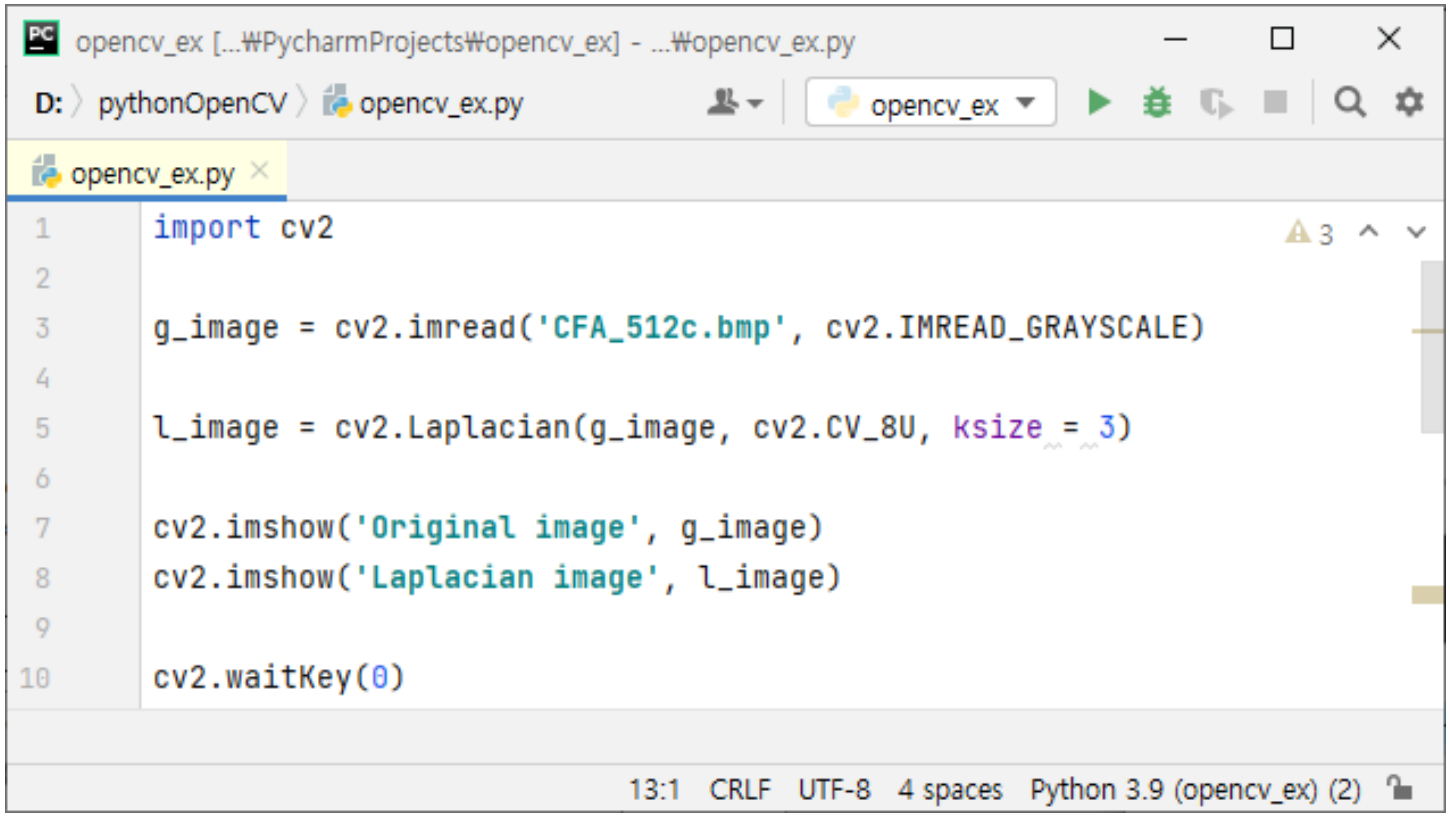
라플라시안 필터 (Laplacian filter)

1. 영교차 (Zero-crossing)
   A. 램프 에지에 대해서 2차 미분을 수행하게 되면 에지의 시작과 끝 부분에서 피크(peak)가 높게 나타남
   B. 따라서 1개 에지에 대해서 2개의 피크가 나타나서 에지의 정확한 위치를 선정하는데 어려움
   C. 영교차점은 필터링된 결과에서 +에서 –로 또는 –에서 +로 부호가
      변경되는 위치로서, 2차 미분의 결과에서 영교차점을 검출하면
      에지의 정확한 위치를 식별



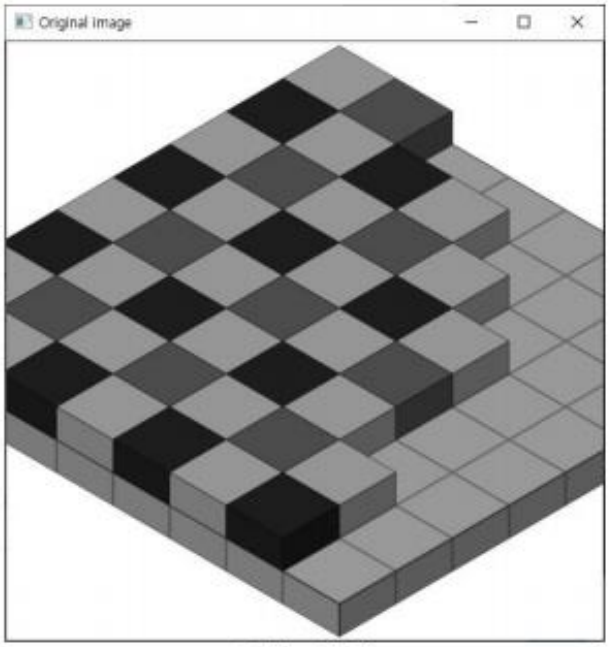
라플라시안 필터 (Laplacian filter)

1. 회색조 영상에 라플라시안 필터(마스크)를 이용하여 경계선을 검출
2. 소스코드 (CodeCV8X_2.py)

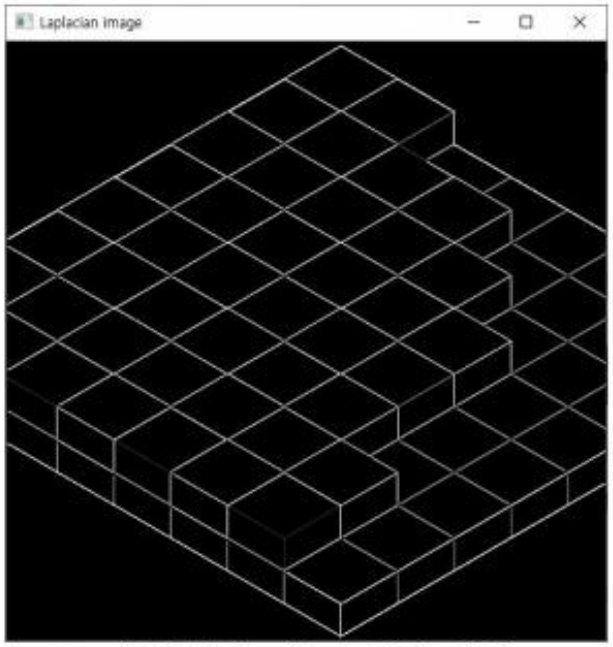


라플라시안 필터 (Laplacian filter)

1. 실행결과



원본 영상



라플라시안 마스크 적용 영상

라플라시안 필터 (Laplacian filter)

1. 라플라시안 필터링 함수: Laplacian() 함수

A. 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 라플라시안 값에 적용할 스케일 요소 (비율) • delta 오프셋 • borderType 픽셀 외삽(extrapolation) 방법 BORDER_CONSTANT 또는 BORDER_REPLICATE
리턴값	출력 영상 데이터 (numpy.ndarray)

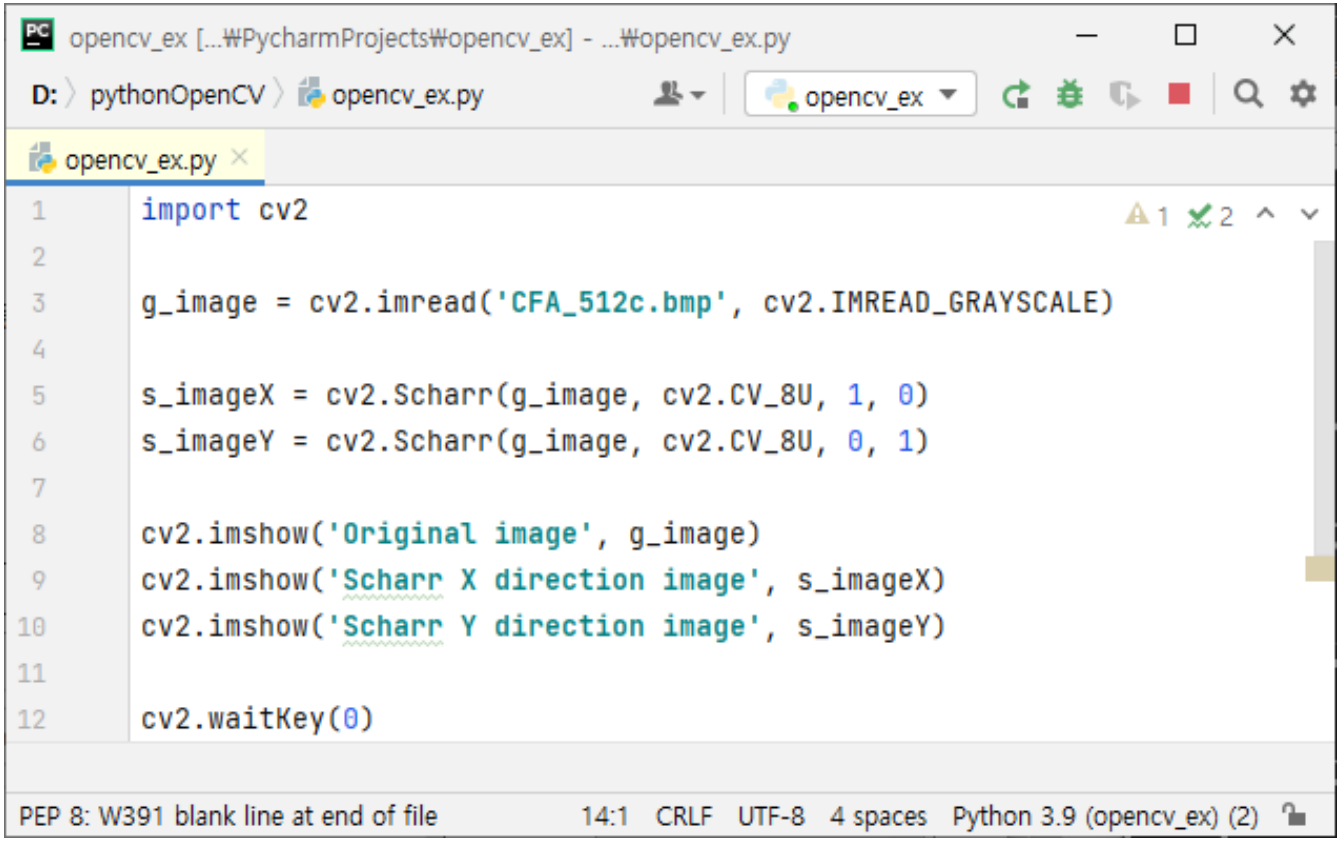
Scharr 필터

1. Scharr 필터
   A. 1차 미분에 해당
   B. Sobel 필터와 동일하지만 ksize가 3x3로 고정되어 있으며 Sobel 보다 좀 더 정확하게 적용

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

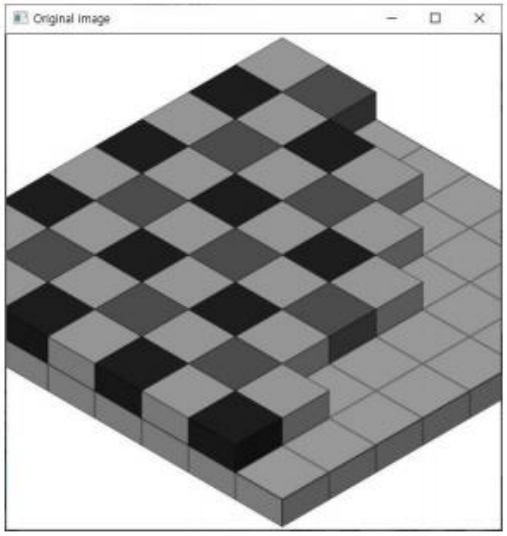
Scharr 필터

1. 회색조 영상에 대하여 Scharr 필터(마스크)를 이용하여 x 방향 변화율, y 방향 변화율, x-y 방향 변화율을 계산하여 출력
2. 소스코드 (CodeCV8X_3.py)

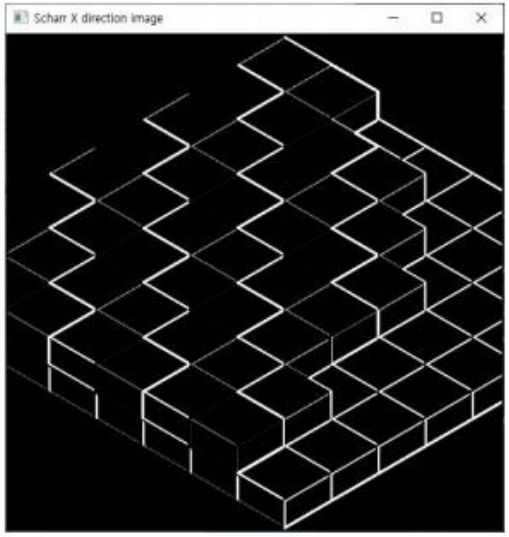


Scharr 필터

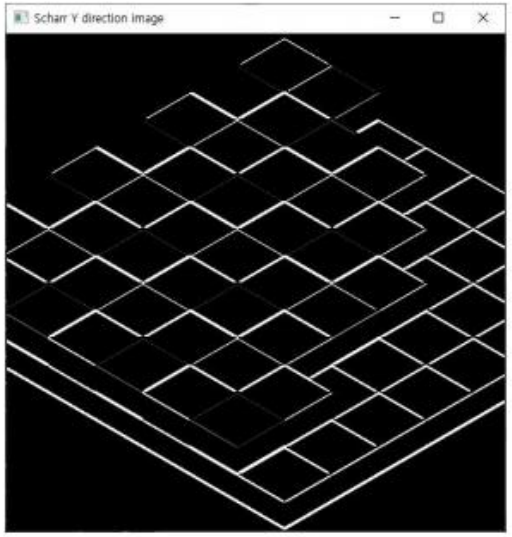
1. 실행결과



원본 영상



Scharr X 방향 마스크 적용 영상



Scharr Y 방향 마스크 적용 영상

Scharr 필터

1. Scharr 필터링 함수: Scharr() 함수

A. 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)

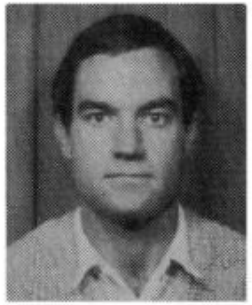
Canny 경계선 검출기

1. Canny 경계선 검출기(edge detector)
   A. 가장 대표적인 경계선 검출(edge detection) 방법
   B. 에지를 잘못찾는 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

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.

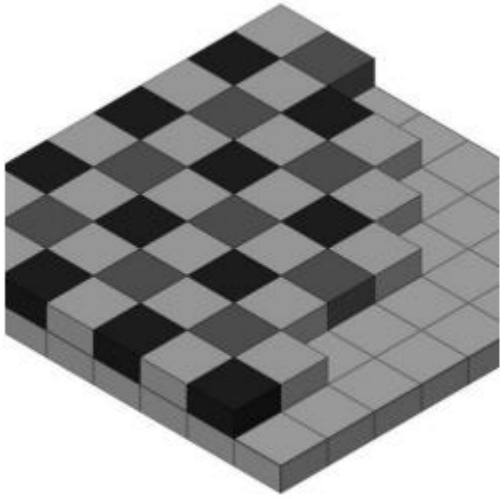
He is with the Artificial Intelligence Laboratory, M.I.T. His research interests include low-level vision, model-based vision, motion planning for robots, and computer algebra.

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

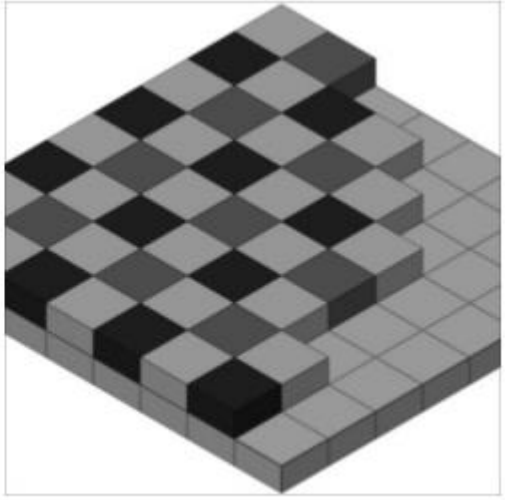
Canny 경계선 검출기

1. 입력 영상에 대하여 여러 단계의 처리 과정을 통해서 경계선 추출
   A. 일련의 처리 과정을 통해서 경계값만 남겨두고 나머지를 제거

2. 단계 1. Noise Reduction
   A. 영상에 존재하는 Noise를 제거 (5x5 크기의 Gaussian filter를 이용)



원본 영상



Gaussian 필터링 영상

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

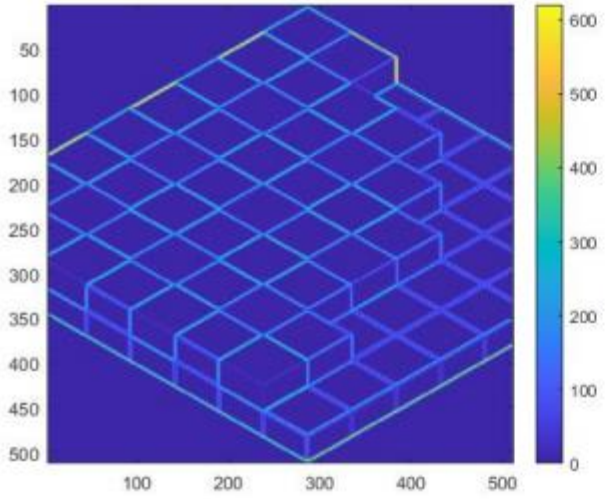
Canny 경계선 검출기

1. 단계 2. Edge Gradient Detection
   A. 영상에서 Gradient의 방향과 강도를 확인

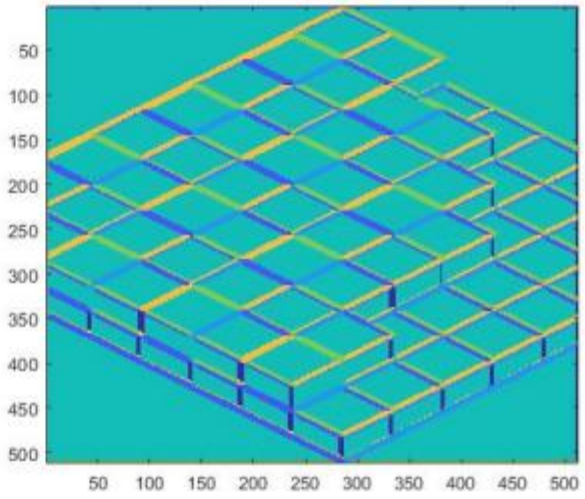
$$\mathbf{G} = \sqrt{\mathbf{G}_x{}^2 + \mathbf{G}_y{}^2}$$

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

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



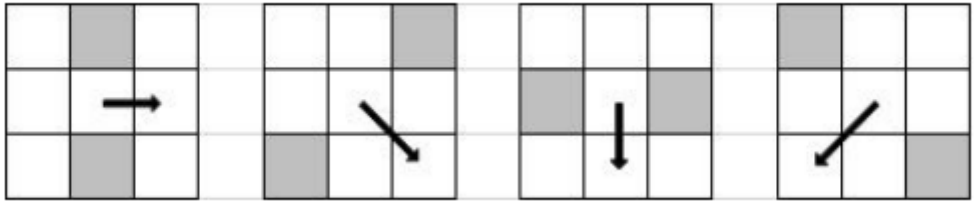
Gradient 강도



Gradient 방향

Canny 경계선 검출기

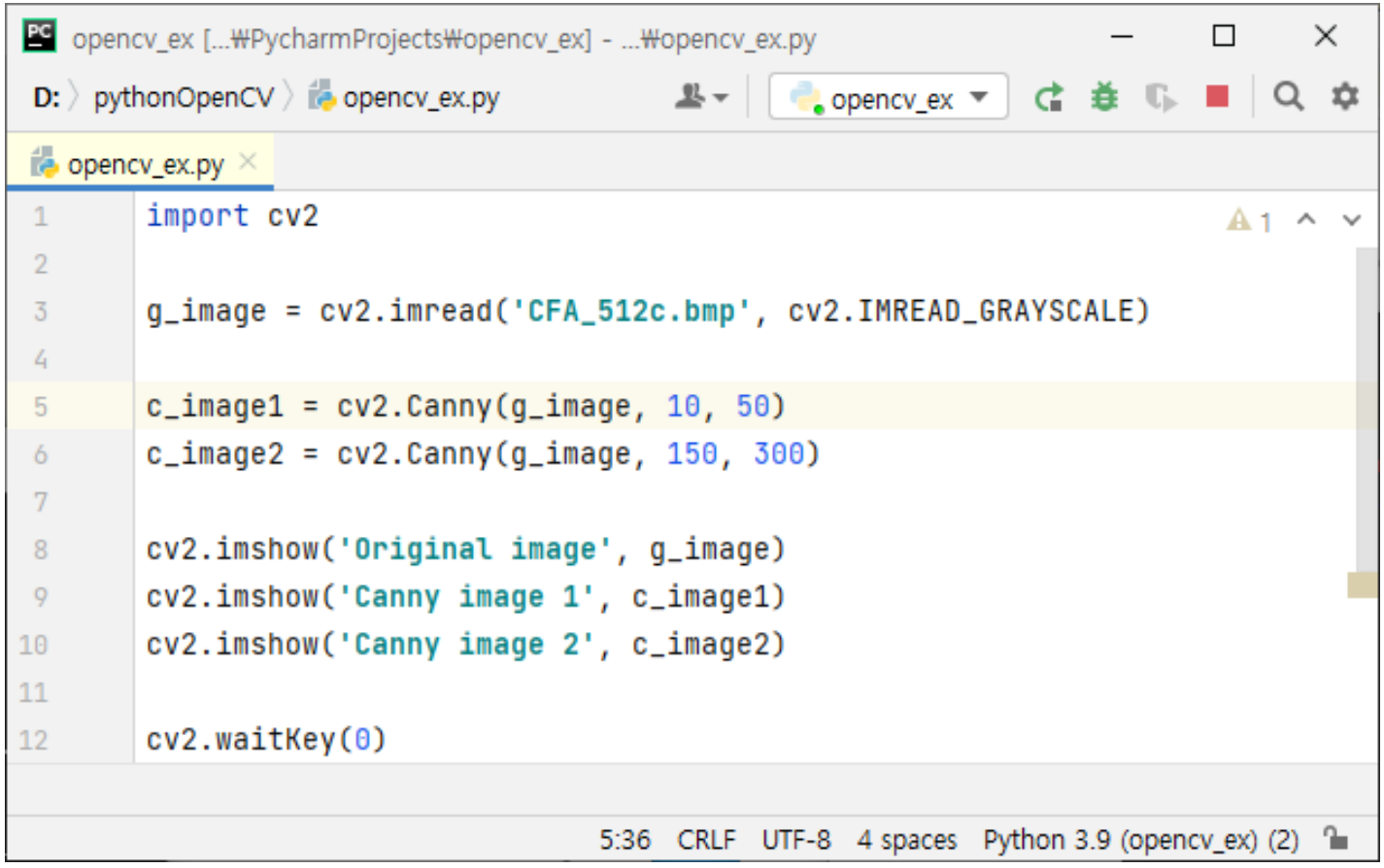
1. 단계 3. Non-maximum Suppression
   A. 영상 픽셀을 전체 탐색(scan)하여 경계선(edge)에 해당하지 않는 픽셀 제거



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

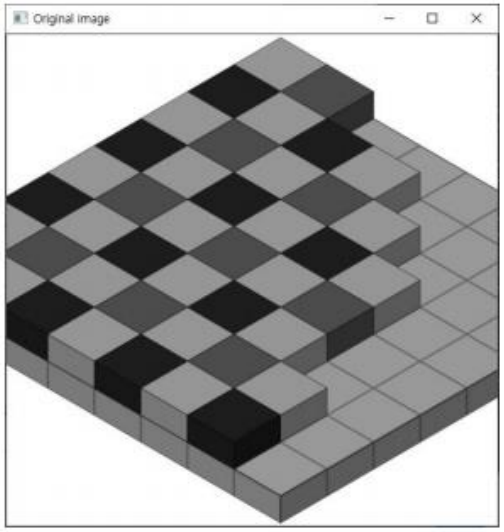
Canny 경계선 검출기

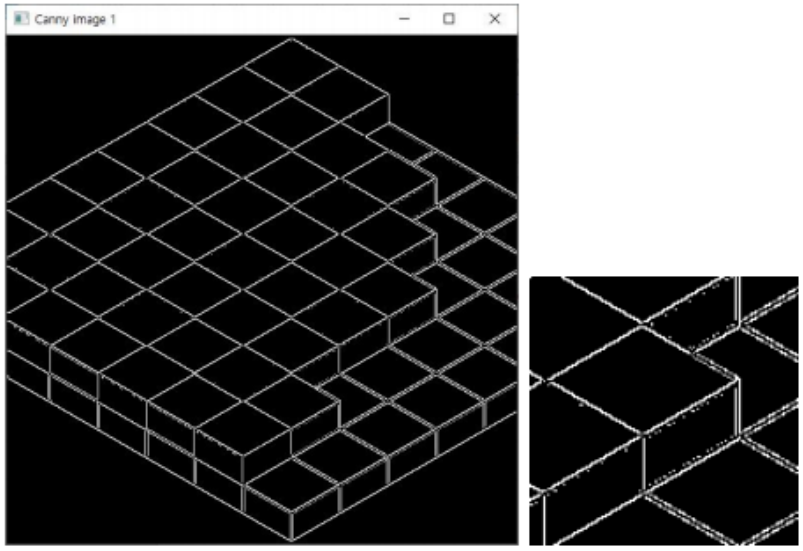
1. 회색조 영상에 대하여 Canny 경계선 검출기를 이용하여 경계선을 검출
2. 소스코드 (CodeCV8X_4.py)



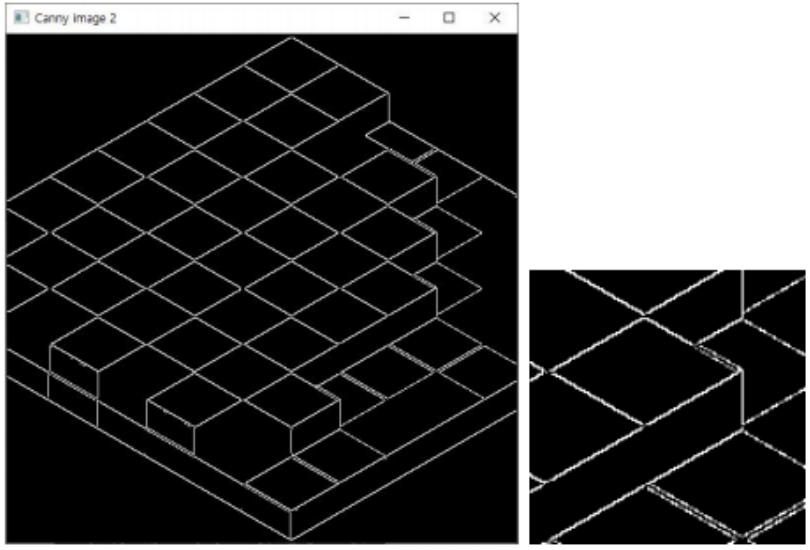
Canny 경계선 검출기

1. 실행결과





Canny 경계선 영상 10, 50
- 대부분의 선들이 검출됨
- 짧은 선들이 다수 검출됨



Canny 경계선 영상 150, 300
- Gradient 강도가 낮기 때문에, 미검출된 선들도 존재함 (좌측 하단 검정색 블록 영역, 우측 영역)
- 짧은 선들은 검출되지 않고, 긴 선들만 검출됨

Canny 경계선 검출기

1. Canny 경계선 검출 함수: Canny() 함수

A. 입력된 영상 데이터에 대하여 Canny 경계선을 검출

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

다양한 응용 프로그램 개발

1. 웹 카메라로 촬영한 영상에 대하여 사용자가
   0을 누르면 원본,
   1를 누르면 Canny Edge 검출
   이미지 출력하는 프로그램 개발