메디치소프트 기술연구소

2018.08.08

CH14. OpenCV: 이미지 Gradient를 이용한 경계 찾기

MEDICISOFT

14. 이미지 Gradient를 이용한 경계 찾기

- 14.1 테스트 환경

순 번	제 목	설치 버전
1	운영체제	Windows 10 64bit
2	프로그래밍언어	Python3.6.5

▪ 14.2 [14-01 example.py] - 이미지 Gradient

- OpenCV는 Sobel, Scharr, Laplacian 이 세가지 타입의 Gradient 필터(High-pass filters;HPF)를 제공한다.

- Sobel, Scharr 미분 (Sobel and Sharr Derivatives)
  obel 오퍼레이터는 가우스 스무딩(Gaussian Smoothing)과 미분연산을 결합한 형태의 연산을 수행함으로써 노이즈에 보다 강력한 저항성을 제공한다.
  세로 방향 또는 가로 방향으로 연산 수행이 가능하다.
  cv2.Sobel() 함수는 이미지에 sobel 연산을 수행하는 함수이다.

▪ 14.2 [14-01 example.py] - 이미지 Gradient

- cv2.Sobel(src, ddepth, dx, dy, ksize)

  src: Sobel 미분을 적용할 원본 이미지

  ddepth: 결과 이미지 데이터 타입

  CV_8U: 이미지 픽셀값을 uint8 로 설정

  CV_16U: 이미지 픽셀값을 uint16으로 설정

  CV_32F: 이미지 픽셀값을 float32로 설정

  CV_64F: 이미지 픽셀값을 float64로 설정

  dx, dy: 각각 x방향, y방향으로 미분 차수 (eg. 1, 0 이면, x 방향으로 1차 미분 수행, y 방향으로 그대로 두라는 의미)

  ksize: 확장 Sobel 커널의 크기. 1, 3, 5, 7 중 하나의 값으로 설정. -1로 설정되면 3x3 Soble 필터 대신 3x3 Scharr 필터를 적용하게 됨

▪ 14.2 [14-01 example.py] - 이미지 Gradient

- 다음은 14-01 example.py 예제의 code이다.

```
import numpy as np
import cv2
import matplotlib.pyplot as plt

def grad() :
    img = cv2.imread('images/keyboard.jpg', cv2.IMREAD_GRAYSCALE)
    laplacian = cv2.Laplacian(img, cv2.CV_64F)
    sobelx = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=3)
    sobely = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=3)

    plt.subplot(2,2,1), plt.imshow(img, cmap='gray')
    plt.title('original'), plt.xticks([]), plt.yticks([])
```

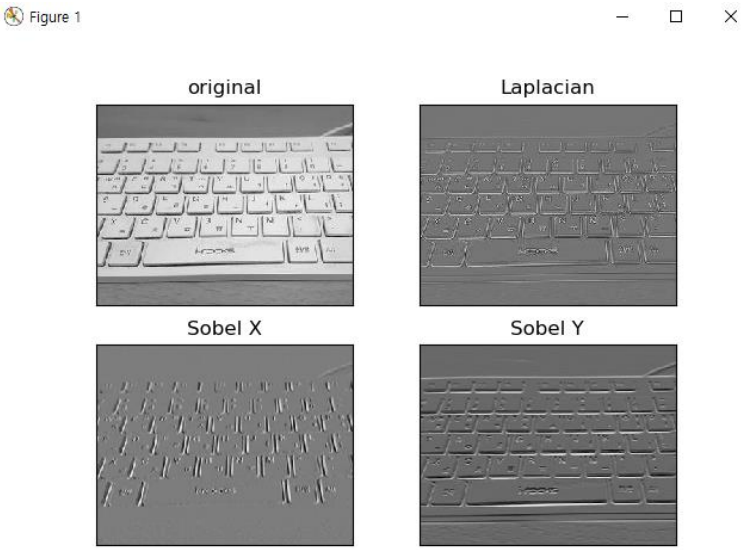
▪ 14.2 [14-01 example.py] - 이미지 Gradient

- 다음은 14-01 example.py 예제의 code이다.

```
    plt.subplot(2,2,2), plt.imshow(laplacian, cmap='gray')
    plt.title('Laplacian'), plt.xticks([]), plt.yticks([])

    plt.subplot(2,2,3), plt.imshow(sobelx, cmap='gray')
    plt.title('Sobel X'), plt.xticks([]), plt.yticks([])

    plt.subplot(2,2,4), plt.imshow(sobely, cmap='gray')
    plt.title('Sobel Y'), plt.xticks([]), plt.yticks([])

    plt.show()

grad()
```

▪ 14.2 [14-01 example.py] - 이미지 Gradient

- 다음 그림은 14-01 example.py를 실행한 결과 생성되는 image 사진이다.
  뒷 장부터 14-01 example.py code의 주요 함수를 line 순서대로 분석해보도록 한다



[그림] 14-01 example.py의 실행 결과

▪ 14.2 [14-01 example.py] - 이미지 Gradient

>> LINE 7~9)

```
laplacian = cv2.Laplacian(img, cv2.CV_64F)
sobelx = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=3)
sobely = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=3)
```

- Laplacian, Sobel의 결과 데이터 타입은 모두 CV_64F로 정의했다.

- obelx는 x방향으로만 1차 미분하고, 커널 크기는 3, soblex는 y방향으로만 1차 미분하고, 커널 크기는 3으로 정한다.

▪ 14.3 [14-02 example.py] - 이미지 Gradient

- cv2.Sobel() 함수를 사용할 때, 인자로 입력되는 데이터타입은 결과물에 영향을 미치는 요소이므로 주의해야한다.

  Sobel 알고리즘은 이미지가 검정색에서 흰색으로 변화될 때 양수 값을 취하고, 흰색에서 검정색으로 변화될 때 음수 값을 취한다.

  만약 데이터 타입을 양수만 취급하는 np.uint8 또는 cv2.CV_8U로 지정하면 흰색에서 검정색으로 변화될 때 취한 음수값을 모두 0으로 만든다.

  다시 말하면, 흰색에서 검정색으로의 경계를 찾지 못하게 되는 결과를 가져온다.

  이와 같은 경우를 예제에 적용하여 분석해보도록 한다.

▪ 14.3 [14-02 example.py] - 이미지 Gradient

- 다음은 14-02 example.py 예제의 code이다.

```
import numpy as np
import cv2
import matplotlib.pyplot as plt

def grad() :
    img = cv2.imread('images/box.jpg', cv2.IMREAD_GRAYSCALE)

    sobelx8u = cv2.Sobel(img, cv2.CV_8U, 1, 0, ksize=5)

    tmp = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
    sobel64f = np.absolute(tmp)
    sobelx8u2 = np.uint8(sobel64f)
```

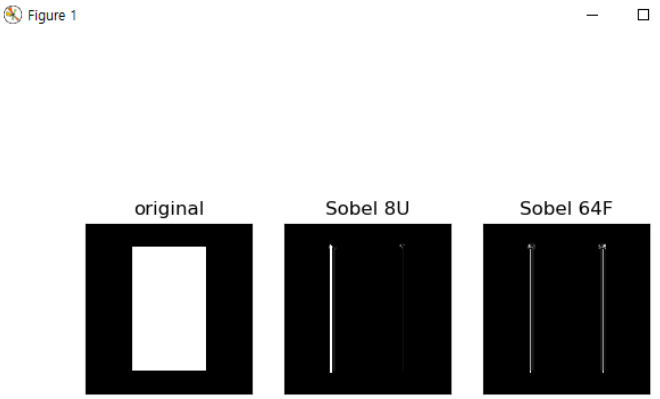
▪ 14.3 [14-02 example.py] - 이미지 Gradient

- 다음은 14-02 example.py 예제의 code이다.

```
    plt.subplot(1,3,1), plt.imshow(img, cmap='gray')
    plt.title('original'), plt.xticks([]), plt.yticks([])

    plt.subplot(1,3,2), plt.imshow(sobelx8u, cmap='gray')
    plt.title('Sobel 8U'), plt.xticks([]), plt.yticks([])

    plt.subplot(1,3,3), plt.imshow(sobelx8u2, cmap='gray')
    plt.title('Sobel 64F'), plt.xticks([]), plt.yticks([])

    plt.show()

grad()
```

▪ 14.3 [14-02 example.py] - 이미지 Gradient

- 다음 그림은 14-02 example.py를 실행한 결과 생성되는 image 사진이다.
  뒷 장부터 14-02 example.py code의 주요 함수를 line 순서대로 분석해보도록 한다



[그림] 14-02 example.py의 실행 결과

▪ 14.3 [14-02 example.py] - 이미지 Gradient

>> LINE 8) sobelx8u = cv2.Sobel(img, cv2.CV_8U, 1, 0, ksize=5)

- Sobel() 함수에 인자 cv2.CV_8U를 적용하였습니다. 이 함수의 결과물은 sobelx8u에 담는다.

>> LINE 10~12)

```
tmp = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
sobel64f = np.absolute(tmp)
sobelx8u2 = np.uint8(sobel64f)
```

- Sobel() 함수에 인자 cv2.CV_64F를 적용하여 얻은 결과물을 tmp에 담고, tmp의 음수 부분을 모두 양수로 변환한 다음 uint8 형식으로 전환한 결과를 sobelx8u2에 담는다.

```python
import numpy as np
import cv2
import matplotlib.pyplot as plt

def grad() :
    img = cv2.imread('images/keyboard.jpg', cv2.IMREAD_GRAYSCALE)

    laplacian = cv2.Laplacian(img, cv2.CV_64F)
    sobelx = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=3)
    sobely = cv2.Sobel(img, cv2.CV_64F, 0, 1, ksize=3)

    plt.subplot(2,2,1), plt.imshow(img, cmap='gray')
    plt.title('original'), plt.xticks([]), plt.yticks([])

    plt.subplot(2,2,2), plt.imshow(laplacian, cmap='gray')
    plt.title('Laplacian'), plt.xticks([]), plt.yticks([])

    plt.subplot(2,2,3), plt.imshow(sobelx, cmap='gray')
    plt.title('Sobel X'), plt.xticks([]), plt.yticks([])

    plt.subplot(2,2,4), plt.imshow(sobely, cmap='gray')
    plt.title('Sobel Y'), plt.xticks([]), plt.yticks([])

    plt.show()

grad()
```

```python
import numpy as np
import cv2
import matplotlib.pyplot as plt

def grad() :
    img = cv2.imread('images/box.jpg', cv2.IMREAD_GRAYSCALE)

    sobelx8u = cv2.Sobel(img, cv2.CV_8U, 1, 0, ksize=5)

    tmp = cv2.Sobel(img, cv2.CV_64F, 1, 0, ksize=5)
    sobel64f = np.absolute(tmp)
    sobelx8u2 = np.uint8(sobel64f)

    plt.subplot(1,3,1), plt.imshow(img, cmap='gray')
    plt.title('original'), plt.xticks([]), plt.yticks([])

    plt.subplot(1,3,2), plt.imshow(sobelx8u, cmap='gray')
    plt.title('Sobel 8U'), plt.xticks([]), plt.yticks([])

    plt.subplot(1,3,3), plt.imshow(sobelx8u2, cmap='gray')
    plt.title('Sobel 64F'), plt.xticks([]), plt.yticks([])

    plt.show()

grad()
```