Image Processing & Vision Homework 2: Edge Detection

예술공학대학 컴퓨터예술학부

20190807 민정우

Implement 3x3 Sobel edge filters in horizontal and vertical directions

이미지에서 Edge는 두 영역의 경계로, 픽셀 값의 변화량이 가장 큰 지점을 의미한다. 픽셀 값의 변화량은 미분을 통해 계산하며 주로 오차가 적은 중앙 차분법을 사용한다.

$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h}$	$[-1 \quad 0 \quad 1]$	$\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$
Central difference	중앙 차분을 이용한 x축과 y축 방향의 미분 마스크	

이미지에 노이즈가 있으면 변화량을 통해 경계를 파악하기 힘드므로 Sobel edge detection은 경계를 추출하기 위해 미분 마스크로 2차원 이미지의 수직 및 수평 방향에 대한 변화량을 계산하며, 스무딩 마스크를 통해 노이즈를 억제한다.

$G_x = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * ([-1 \quad 0 \quad 1] * I) = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I$	$G_y = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * ([1 \quad 2 \quad 1] * I) = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I$
Get gradient with derivative mask and smoothing mask	

따라서, 수직 및 수평 방향에 대한 Sobel Filter는 미분 마스크와 스무딩 마스크의 합성곱과 같다.

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

```
################################################
#                Sobel Filter                  #
################################################
def edgeSobel_x():
  output_filter = np.array([[-1, 0, 1],
                            [-2, 0, 2],
                            [-1, 0, 1]])

  return output_filter


def edgeSobel_y():
  output_filter = np.array([[1, 2, 1],
                            [0, 0, 0],
                            [-1, -2, -1]])

  return output_filter
```

Perform edge detection with the implemented Sobel edge filters

2차원 이미지 I에 대해 Sobel filter를 통한 수직 및 수평 방향의 이미지 그라디언트는 다음과 같다.

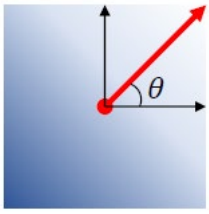
$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} * I, G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * I$$

	
Horizontal Edge Map by Sobel Filter	Vertical Edge Map by Sobel Filter

각 축에 대한 이미지 그라디언트를 계산함으로써 Horizontal Edge Map에서는 수평 방향의 경계가 검출됐고, Vertical Edge Map에서는 수직 방향의 경계가 검출됐다.

With the horizontal and vertical edges (i.e., image gradients), calculate the magnitude and orientation (angle) of the gradients

각 축에 대한 2개의 Sobel Filter를 적용한 결과 이미지는 수직, 수평에 대한 픽셀 값의 변화량이며, 두 결과 이미지를 통해 2차원 이미지 그라디언트의 크기와 방향을 구할 수 있다.



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

그라디언트의 크기는 다음과 같이 계산한다.

$$||\nabla f|| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

```
##############################################
#              Gradient Magnitude            #
##############################################
def magGrad(edge_x, edge_y):
  magnitude_grad = np.sqrt(edge_x * edge_x + edge_y * edge_y)

  return magnitude_grad
```

Gradient Magnitude Map

수직 방향의 그라디언트와 수평 방향의 그라디언트를 이용해 이미지의 2차원 그라디언트의 크기를 계산함으로써 이미지에서 경계를 감지한다. 적절한 임계값을 지정해 임계값보다 작은 그라디언트를 무시해 변화량이 큰 경계를 강조 표시하면 보다 확실한 경계만을 추출할 수 있다.

```
##############################################
#            Set Gradient Threshold          #
##############################################
def setThreshold(image, threshold):
  output_grad = np.where(image < threshold, 0, image)

  return output_grad
```

Threshold = 0	Threshold = 50	Threshold = 100	Threshold = 150

임계값이 클수록 작은 변화를 무시하면서 확실한 경계를 추출할 수 있으나 임계값이 너무 크다면 필요한 경계가 무시될 수 있어 적절한 임계값을 설정하는 것이 중요하다.

그라디언트의 방향은 다음과 같이 계산한다.

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial y}\right)$$

```
##############################################
#             Gradient Orientation           #
##############################################
def angleGrad(edge_x, edge_y):
  angle_grad = np.arctan2(edge_y, edge_x) * 180 / np.pi

  return angle_grad
```

numpy.arctan2 함수의 반환값은 라디안이므로 $\frac{180}{\pi}$을 곱해 일반각으로 변환한다.

Gradient Orientation Map

그라디언트의 각도는 이미지에서 경계의 방향을 나타내며, 그라디언트 각도에 수직인 방향으로 경계를 추적할 수 있다. Canny Edge Detection은 경계의 방향을 분석함으로써 경계의 방향을 따라 겹쳐 있는 경계 중 최댓값이 아닌 경계를 억제해 경계를 얇게 만들어 Sobel Edge Detection보다 더 정교하게 이미지의 경계를 감지한다.