HW 03 – REPORT

소속 : 정보컴퓨터공학부

학번 : 201925111

이름 : 김건호

1. 서론

1. 실습 목표 – canny edge detection을 이해할 수 있다.

1) gaussian filter를 사용해 image의 noise를 줄일 수 있다.

2) sobel filter를 사용해 image의 gradient 크기와 방향을 이해할 수 있다.

3) Non-Maximum Suppression을 통해, edge의 크기를 줄일 수 있다.

4) Hysteresis thresholding를 통해, weak edge를 strong edge로 강화할 수 있다.

2. 이론적 배경

Edge는 image의 많은 정보를 담고 있습니다. Computer vision에서, 이러한 edge를 detection한 후, pattern을 matching하거나, recognization하기 위해 edge를 사용합니다. Canny edge detection란, edge detection을 하기 위한 알고리즘 중 하나입니다.

Edge는 흑백 사진에서 image intensity가 갑자기 변하는 지점이라고 생각할 수 있습니다. 갑작스레 변하는 지점을 찾기 위해, image의 gradient를 계산합니다.

먼저 gaussian filter와 image를 convolution해서 image의 noise를 줄입니다. image에서 noise를 줄이지 않는다면, speckle pixel을 edge로 판별할 가능성이 있기 때문입니다.

이후, sobel filter를 사용해 image의 gradient 크기와 방향을 계산합니다. Sobel filter는 derivatived gaussian filter의 근사입니다. 아래는 sobel x filter와 sobel y filter입니다.

$\frac{1}{8}$

-1	0	1
-2	0	2
-1	0	1

s_x

$\frac{1}{8}$

1	2	1
0	0	0
-1	-2	-1

s_y

Sobel filter와 image를 convolution한 후, 아래 수식을 사용해서 image의 gradient 크기와 방향을 얻습니다.

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2} \qquad \theta = \tan^{-1}\left(\frac{\partial f}{\partial y} \Big/ \frac{\partial f}{\partial x}\right)$$

이렇게 얻어진 image의 edge는 굵습니다. Edge는 하나의 선들로 이루어져 있기에, Non-maximum

suppression 방식을 통해 edge size를 조절합니다. 어떤 pixel의 gradient 방향으로 두 이웃들을 고려합니다. 만약 현재 pixel의 intensity값이 local maximum이라면 intensity value를 유지, 아니라면 0을 선택합니다.

마지막으로 Double threshold방식을 통해 image를 strong, weak, no edge part로 나누고, strong edge 주변 weak edge는 strong edge로 강화합니다. 이 때, 2개의 threshold value(t_high, t_low)를 사용합니다. 만약 어떤 pixel의 intensity value가 t_high보다 크다면 strong edge로 판단하고, t_low보다 작다면 no edge라고 판단합니다. 만약 t_high와 t_low 사잇값이라면 weak edge로 판단합니다. Strong edge는 255, weak edge는 80, no edge는 0의 intensity value로 mapping합니다.

아래 수식을 통해, t_high와 t_low를 계산할 수 있습니다.

$$diff = \max(image) - \min(image)$$
$$T_{high} = \min(image) + diff * 0.15$$
$$T_{low} = \min(image) + diff * 0.03$$

이렇게 mapping된 image를 linking (hysteresis)합니다. 본 실습에선, linking을 위해 depth-first search 방식을 선택하였습니다. 이러한 일련의 과정을 통해, canny edge detection을 성공할 수 있습니다.

2. 본론

1. Noise reduction

Canny edge detection을 하기 위해서 첫 번째로 image의 noise를 감소시켜야 합니다. 따라서 gaussian filter(sigma=1.6)을 사용해서 image를 smooth하게 만듭니다.

결과 :

2. Finding the intensity gradient of the image

Sobel filter를 사용해서, image와 convolution합니다. 다음과 같은 sobel filter를 사용합니다.

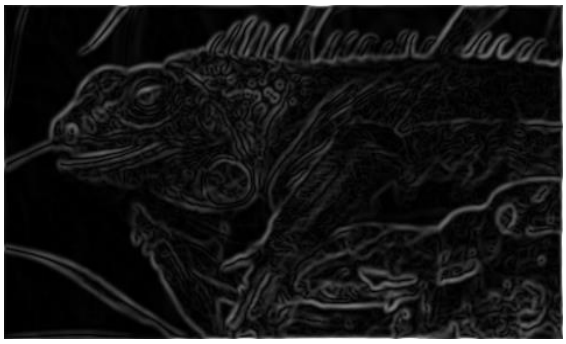
$$Sx = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$$

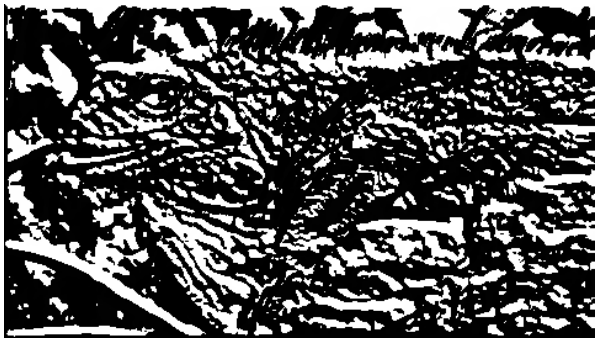
$$Sy = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}$$

Image와 sobel filter를 convolution한 후, gradient_x와 gradient_y를 얻을 수 있습니다.

Np.hypot과 , np.arctan2를 활용해서, gradient의 크기와 gradient의 각도를 얻을 수 있습니다.

Gradient의 크기와 각을 나타낸 결과 :



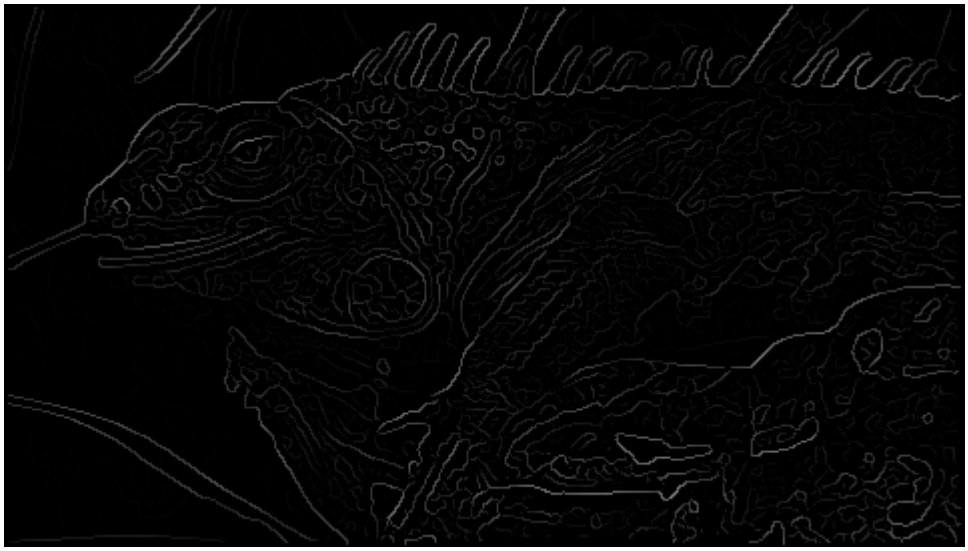


3. Non-Maximum Suppression

현재 pixel에서 계산된 gradient의 방향 부근 이웃한 두 pixel에 대해 local maximum이 된다면 값을 선택하고, local maximum이 되지 않는다면 0을 선택합니다. 최종적으로 좀 더 얇은 edge를 얻을 수 있는 과정입니다. Np.arctan2()의 반환값은 부호를 포함한 라디안입니다. 아래 라디안 값들을 기준으로 하고, 유격을 $\pi/8$으로 설정해서 neighbor의 intensity value와 현재 pixel의 intensity value를 비교합니다.

$$45° = \frac{\pi}{4}, 90° = \frac{\pi}{2}, 135° = \frac{3\pi}{4}, 180° = \pi$$

결과 :

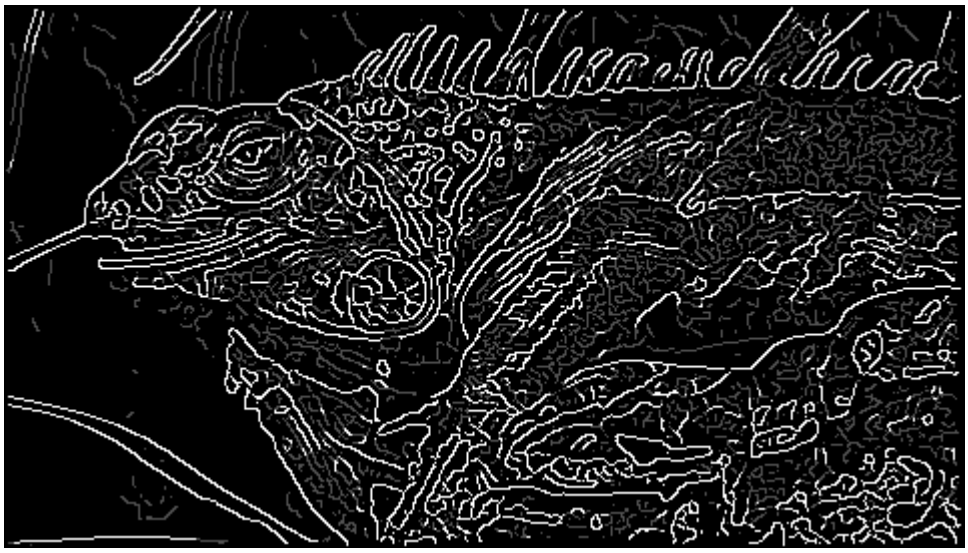


4. Double threshold

이미지의 최댓값과 최솟값을 사용하여, threshold의 최대, 최솟값을 지정합니다.

Threshold 값을 기준으로 strong, weak, no edge를 판단합니다. Strong edge라면 255, weak edge라면 80, no edge라면 0을 mapping합니다.

결과 :

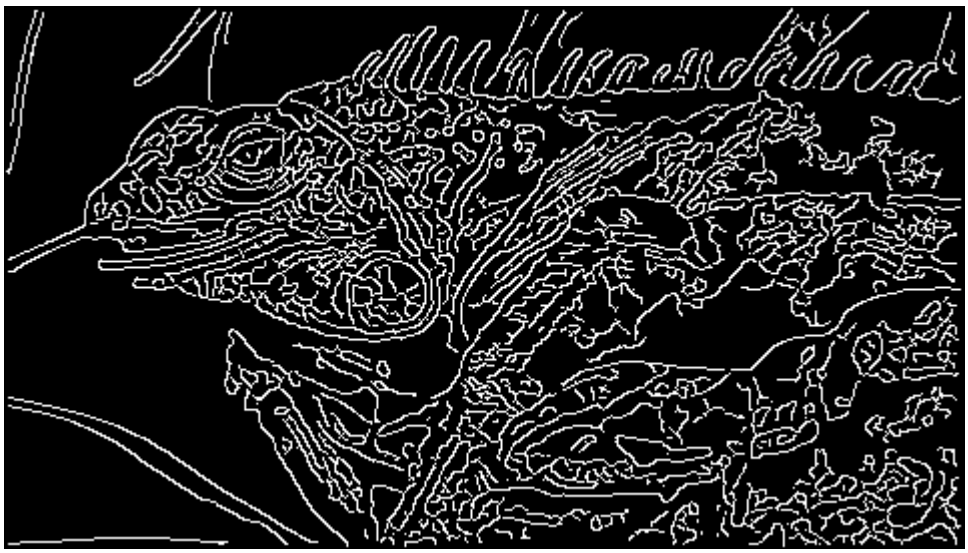


5. Edge Tracking by hysteresis

Weak edge가 만약 strong edge와 연결되어 있다면 strong edge로 강화하는 단계입니다.

Strong edge를 기준으로, 주변 weak edge를 depth-first search로 탐색합니다.

결과 :



3. 결론

Canny edge detection을 통해, image의 edge를 획득할 수 있었습니다. Sobel operator 과정에서 일반적으로는 3x3 matrix를 사용합니다. 하지만 특정 상황에선, 5x5 sobel filter를 사용하거나, Robert cross filter를 사용할 수도 있습니다. 아래는 Robert cross filter입니다.

$$\begin{bmatrix} +1 & 0 \\ 0 & -1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 0 & +1 \\ -1 & 0 \end{bmatrix}.$$

출처 – 위키피디아

또한, sobel operator가 아닌, 기준 pixel에서 이웃한 pixel에 동일한 가중치를 두는 prewitt operator도 존재합니다. 아래는 prewitt operator에서 사용되는 filter를 사용해 gradient를 얻는 방식입니다.

$$\mathbf{G_x} = \begin{bmatrix} +1 & 0 & -1 \\ +1 & 0 & -1 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G_y} = \begin{bmatrix} +1 & +1 & +1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} * \mathbf{A}$$

출처 – 위키피디아

또한, noise를 줄일 때, gaussian filter를 사용했습니다. 조금 더 성능을 개선하기 위해, edge에는 덜 smoothing하고, noise는 더 smoothing한 효과를 주도록, noise reduction 과정에서 filter를 교체할 수 있습니다.

1. K = 1, set the iteration n and the coefficient of the amplitude of the edge h.
2. Calculate the gradient value $G_x(x, y)$ and $G_y(x, y)$
3. Calculate the weight according to the formulae below:

$$d(x, y) = \sqrt{G_x(x, y)^2 + G_y(x, y)^2}$$

$$w(x, y) = \exp\left(-\frac{\sqrt{d(x, y)}}{2h^2}\right)$$

4. The definition of the adaptive filter is:

$$f(x, y) = \frac{1}{N} \sum_{i=-1}^{1} \sum_{j=-1}^{1} f(x + i, y + j) w(x + i, y + j)$$

to smooth the image, where

$$N = \sum_{i=-1}^{1} \sum_{j=-1}^{1} w(x + i, y + j)$$

5. When K = n, stop the iteration, otherwise, k = k+1, keep doing the second step

출처 – 위키피디아

실습을 통해서, canny edge detection을 구현해 볼 수 있었습니다. 이렇게 추출한 image의 edge는 pattern matching 등에 사용될 수 있습니다.