6- edge detection

goal= identify sudden changes (discontinuities) in an image

edge= an image contour across which the image's brightness or hue changes abruptly

↳ surface-normal discontinuities = top vs side

↳ depth discontinuities= side of an object

↳ surface color discontinuities= text / ink

↳ illumination discontinuities= shadows

image gradients

1-D discrete derivatives: — kernels

↳ backward filter= $f(x) - f(x-1) = f'(x)$ — $[0 \ 1 \ -1]$

↳ forward filter= $f(x) - f(x+1) = f'(x)$ — $[-1 \ 1 \ 0]$

↳ central filter= $f(x+1) - f(x-1) = f'(x)$ — $[1 \ 0 \ -1]$

example= $f(x) = 10 \ 15 \ 10 \ 10 \ 25 \ 20 \ 20$

$-f(x-1) = -10 \ -10 \ -15 \ -10 \ -10 \ -25 \ -20$

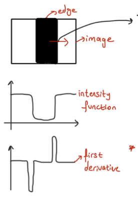
$f'(x) = 0 \ 5 \ -5 \ 0 \ 15 \ -5 \ 0$

2-D discrete derivatives: $f(x,y) \longrightarrow \nabla f(x,y) = \begin{bmatrix} \frac{\partial f(x,y)}{\partial x} \\ \frac{\partial f(x,y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$ → gradient vector

example= $I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix}$

to detect vertical lines: $F_1 = \frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ $\quad I * F_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 10 & 0 & 0 \\ 0 & 10 & 10 & 0 & 0 \\ 0 & 10 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$

to detect horizontal lines: $F_2 = \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$ $\quad I * F_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$



$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$

* the gradient vector points in the direction of most rapid increase in the intensity. $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$
* the edge strength is the magnitude of the gradient $\|\nabla f\|$
* edges correspond to extrema of derivatives

• if the image is noisy, the gradient of it would give no information (instant changes everywhere)

↳ smoothing the image= forcing pixels different to their neighbors to look more like neighbors

↳ mean smoothing = $\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ $[1 \ 1 \ 1]$

↳ gaussian smoothing = $\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$ $[1 \ 2 \ 1]$ $\qquad \frac{d}{dx}(f * g) = f * \frac{dg}{dx}$

convolving with smooth filter and then taking derivative = convolving with derivative of smoothing filter

* Just convolve with the derivative of gaussian filter to detect edges

optimal edge detector

↳ good detection= minimum false positives (gives edges when there is none) and false negatives (missing real edges)

↳ good localization= detected edges must be so close the actual edge pixels

↳ silent response= must return one point only for each true edge point (not too many extra responses)

gradient operators = small ones like 2x2 have good localization but they are sensetive to noise and poor in detection

edge thresholding (hysteresis) = $\|\nabla f(x,y)\| < T_0$ definitely not an edge

↳ $\|\nabla f(x,y)\| \geq T_1$ definitely an edge

↳ $T_0 \leq \|\nabla f(x,y)\| < T_1$ is an edge if a neighboring pixel is definitely an edge

sobel edge detector = 3x3 kernels (grayscale filter, since it is small it is noisy)

↳ convolved with the image to calculate approximations of the derivative

$\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}$ G_x horizontal change $\qquad \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$ G_y vertical change

$G = \sqrt{G_x^2 + G_y^2}$ $\qquad$ orientation $\theta = \tan^{-1}\left(\frac{G_y}{G_x}\right)$

★ sobel operator is obtained by calculating the derivative of the gaussian filter

$G_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} = \underset{\text{gaussian smoothing}}{\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}} \overbrace{[+1 \ 0 \ -1]}^{\text{derivative}}$

canny edge detector =

1- noise reduction= gaussian filter

2- gradient calculation= sobel filters for both direction → then the magnitude and orientation of the gradient is calculated

3- non- maximum suppression = finds the pixel with max value in the edge direction (thinner edges)

4- double threshold= strong, weak and non-relavant pixels are identified

5- edge tracking by hysteresis = transforming weak pixels into strong ones

σ = gaussian kernel spread/size (large σ detects large scale edges)