CS - 512 - AS2

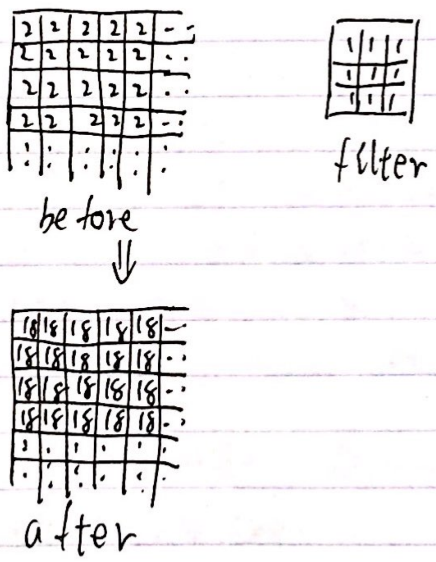
1. (a) $SNR = \frac{E_s}{E_n} = \frac{\sigma_s^2}{\sigma_n^2}$

The signal to noise ratio is use varience of pixels in a sequence of images to divide by varience in uniform area.

(b) Gaussian noise having a probablity density function(PDF) equal to that of the normal distribution. However, impulsive doesn't fit the normal distribution. It has outliers or unwanted noises.

median filter handles better. Because average is much sensitive to extrme value than median.

(c)



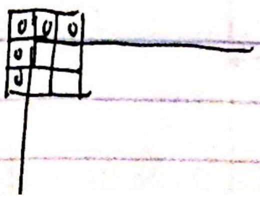
for boundaries use replication

(d) Because convolution is a linear filter, it can be derivative before combine with image.

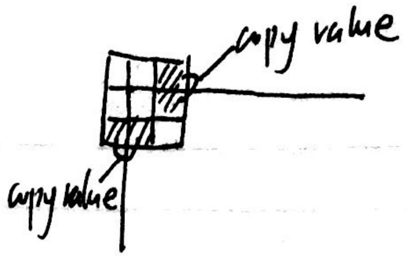
$$\frac{d}{dx}(f * g) = (\frac{d}{dx}f) * g$$

(e) ways to handle boundaries :

1). zero padding



2) replication



3). ignore the boundaries, if image is large and filter is small.

(f) basic smoothing filter

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

the sum of all entries in this filter is 1.

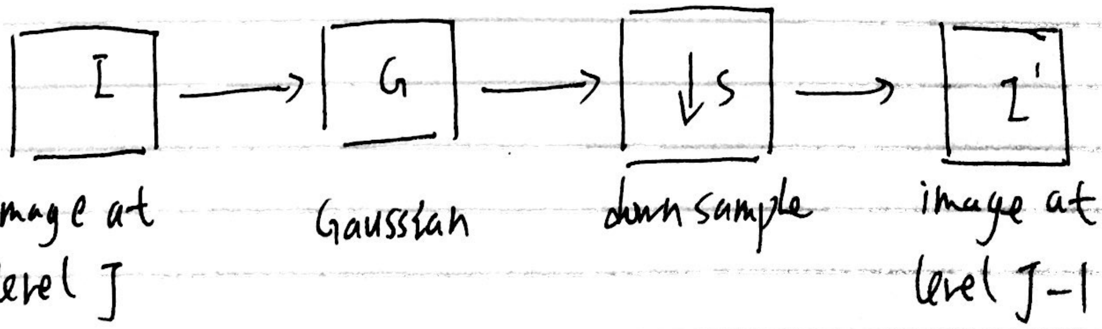
Because the entry in the filter means the weight of each pixel in this filter. therefore the sum should be 1.

(g) $I * G(x,y) = G(y) * (I * G(x))$
first use $G(x)$ filter then apply $G(y)$ filter to implement a 2D convolution.
Because 2D need compute $M \times N \times m^2$ pixels, but 1D only compute $2 \times M \times N \times m$. Use 1D filter to implement 2D convolution requires less computation.

No, it's impossible. Because only linear filter can be implement like this.

(h) Because of $m \geqslant 5\sigma$, $\sigma = 2$ then $m \geqslant 10$.

(i)



For an $m \times m$ image,
the total level of it is $J = \log m$.

The reason for producing such pyramid is to reduce the scale of image.

For this process it's total analysis pixels is:

$$m^2 + \frac{1}{4}m^2 + \frac{1}{16}m^2 \cdots + 1 < \frac{4}{3}m^2$$

Therefore, the additional processing less than $\frac{4}{3}m^2 - m^2 = \frac{1}{3}m^2$

(j).

$$\begin{tikzcd}
\boxed{I} \arrow[r] & \boxed{G} \arrow[r] & \boxed{\downarrow s} \arrow[d] & \text{down sample} \\
\text{image at level } J & \text{Gaussian} & \boxed{\uparrow s} \arrow[d] & \text{up sample} \\
 & & \boxed{G} \arrow[d] & \text{Gaussian} \\
 & & \boxed{I'} & \text{image at level } J-1
\end{tikzcd}$$

For an $m \times m$ image, the total level of image is $J = \log m$.

This technique can be used in image compression.

2.(a). Edge means the change in image, the detection of edge can help us get the feature and important detail of image. It also can segment the image and extract textrue data of image.

desired properties of edge detection:

- correspond to scene elements
- invarient
- reliable consistant detection.

(b) Edge detection:

1. smoothing (without affecting edges)
2. enhance edges
3. localize edges. (look at derivative between pixel)

need: 1. improve the performance of edge detector which are sensetive to noise.

2. Enhance emphasizes pixels where there is a significant change in image.

3. localize the edges with strong edge contents, use threshold to find which point are edge points.

(c) Sobel filter: use two 3x3 kernel one for horizontal change, one for vetical, calculate approximate derivatives.

Canny edge detector: detect edges using ~~directive~~ directional derivative with multi-stages. which is non-maximam supression and hysteresis thresholding.

image gradient: It is a directional change in the intensity or color in an image.

It's used for the fundamental building block in image processing. It also used for gradual blend of color which can be consider as an even gradation from low to high values.

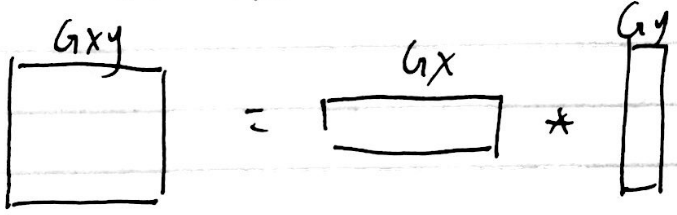
(d) $I_x = I * (G * \frac{\partial}{\partial x})$ $I_y = I * (G * \frac{\partial}{\partial y})$

smooth before taking derivatives.

$(G * \frac{\partial}{\partial x}) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

$(G * \frac{\partial}{\partial y}) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} * \begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$

(e) More accurate derivative $= f[x] * G'[x]$

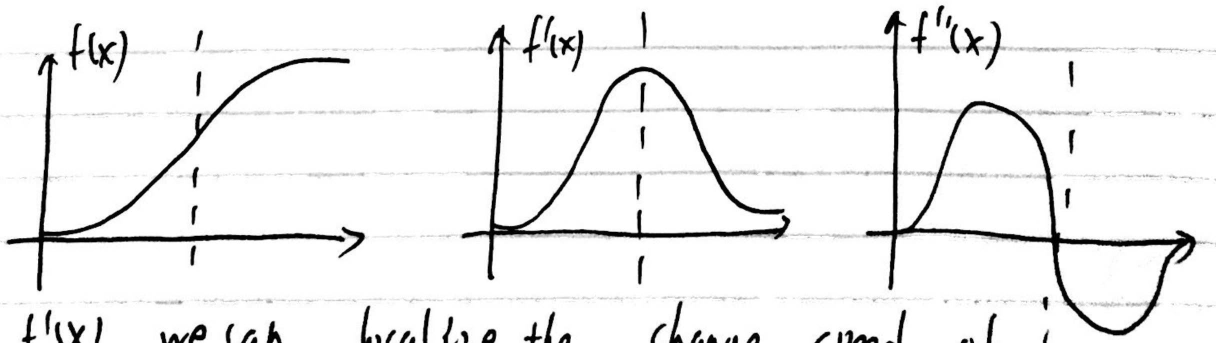


$I_x = I * G_y * G'_x$ $I_y = I * G_x * G_y'$

$G(x) = e^{-\frac{x^2}{2\sigma^2}}$ $G'(x) = -\frac{x}{2\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$ $G'(y) = -\frac{y}{2\sigma^2} e^{-\frac{y^2}{2\sigma^2}}$

$\sigma = 2$ then $\begin{cases} I_x = I * -\frac{x}{8} e^{-\frac{x^2}{8}} * e^{-\frac{y^2}{8}} = I * -\frac{x}{8} e^{\frac{-(x^2+y^2)}{8}} \\ I_y = I * -\frac{y}{8} e^{-\frac{y^2}{8}} * e^{-\frac{x^2}{8}} = I * \frac{x}{8} e^{-\frac{(x^2+y^2)}{8}} \end{cases}$

(f)



for f'(x), we can localize the change speed of edges. and use threshold to get edges.

for f''(x), edge is located at zero-crossing.

(g). $LOG = \Delta G = \frac{r^2 - 2\sigma^2}{\sigma^4} e^{\frac{-r^2}{2\sigma^2}}$ where $r = \sqrt{x^2+y^2}$

$\sigma = 1$ $LoG = (r^2 - 2) e^{-\frac{r^2}{2}}$

edge detection with log:

1). compute LOG (convolve with LOG)

2) threshold $\begin{cases} \rightarrow 1 & \text{where } I * LoG > 0 \\ \rightarrow 0 & \text{where } I * LoG \le 0 \end{cases}$

3) mark edges at transitions $(0 \to 1, 1 \leftarrow 0)$

(h). Canny edge detection using directional derivative to detect edges.

$$\begin{cases} n = \nabla I \\ \text{if } |n| > \tau \text{ (threshold)} \\ \text{detect edges as max of } \frac{\partial I}{\partial n}(I * G) \end{cases}$$

(i). Non-maximum suppression: if $|\nabla I| > \tau$, find the local max of $|\nabla I * G|$ in direction of gradient and set all other values to 0.

hysteresis thresholding:

1) make array of visited pixels $V[i,j] = 0$

2). scan image left to right, top to bottom. if [illegible] $V(i,j) != 0$, and $\nabla(I * G) > \tau_H$, track an edge using τ_L, and set it $V(i,j) = 1$.