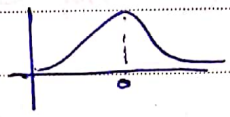
lec 2 Image Processing

① linear Filter in **Spatial** domain
Image deverative , Box filter , Gaussian filter

② $I(x,y)$ → true pixels $n(x,y)$ → noise pixel $\hat{I}(x,y) = I(x,y) + n(x,y)$ (Actual img)

③ Gaussian noise
→ $n(x,y) = e^{\frac{-n^2}{2\sigma^2}}$  $\mu = 0$, $\sigma = 1$

④ Image processing transformation
① Point_operators → Point to point mapping (Pexel-Pexel)
brightness , Contrast , Correction

② Filters in Spatial domain → neighbourhood pixels
mathematical operations → Smooth , Sharp , texture

③ Filters in Frequency domain → Frequinces
denoise , Sampling , Compression

④ Templates , img Pyramids → match a template
detection , Coarse_to_fine registiration

⑤ Point operator → Input pixel : output pixel
$g(i,j)$ → Brightness
Contrast → multiplicative gain → ⓐ
$g(i,j)$ → addative offset → ⓑ

output img ← $g(x) = h(f(x))$ ← input img → Continous domain
↓ Function

$g(i,j) = h(f(i,j))$ → discrete domain

/ /

Common Point Process

⑥ multiplication } $g(x) = af(x) + b$ → $a > 0$ → gain → Contrast
addition → b → bias → bright

⑦ adding Same value to each Color not only increase the intensity of each pixel but also affect hue, Saturation

⑧ Color balancing → ضرب X each channel with Scale factor

⑨ matting → Process of extracting object from image
Compositing → insert img into another image

↑ الـ استخدم → alpha matted Color img for foreground
matting , ↓ , Compositing

⑩ Compositing equation ⇒ $C = (1 - \alpha)B + \alpha F$

⑪ Alphamatted image → Contain 4th alpha channel
(A) → relative amount of opacity or Fractional Coverage
Alpha (opacity) is opposite of transparency
$\alpha = 0$ → transparent
$\alpha = 1$ → opaque → كله ظاهر

⑫ histogram equalization
histogram for individual Color Channel
لكل منهم ⇒ Calculate min, max, Avg intensity Values

histogram equalization ⇒ Find intensity mapping function $f(I)$ Such as histogram is flat

smooth Sharpeing

(13) Linear Filtering in Spatial domain

- neighborhood Filtering (Convolution)

window ⊗ weight (Filter) = X → output

f(x,y) X h(x,y) = g(x,y)

- Image Filtering ⇒ Compute function of Local neighborhood at each position

used in

1. Enhance img → Denoise, resize, Contrast
2. Extract info → text, edge, Points
3. detect patterns → template matching

(14) Derevative (تفاضل) → rate of change

(15) Average (mean) = divide sum of N values by N

(16) discrete derivative

f(x) − f(x−1) → backward difference

f(x) − f(x+1) → Forward difference

f(x+1) − f(x−1) → Central difference

(17) direvative in 2D

Function = f(x,y)

Gradient vector = $\nabla f(x,y)$

Gradient magnitude = $|\nabla f(x,y)|$

Gradien direction

$\theta = \tan^{-1} \frac{f_x}{f_y}$

Gaussian
1. Smooth function , infinite num of derivatives
2. Forier transform of Gaussian → gaussian
3. Convolution of Gaussian → gaussian

---

(18) Derevative mask

[-1, 1] backward
[1, -1] Forward
[-1, 0, 1] Central

(19) Corelation ⇒ $f \otimes h = \sum\sum f(k,l)\, h(i+k, j+l)$

(20) Convolution ⇒ $f * h = \sum\sum f(k,l)\, h(i-k, j-l)$

(21) Box Filter = Average Filter → Smoothing
Replace each pixel with an average of its neighbors

(22) Gaussian Filter ⇒ Smoothness
weight Contribution of neighbour pixels by nearness

$$G = \frac{1}{2\pi\sigma^2} \times e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

- Remove high Frequency = low Pass Filter
- Convolution ↓ Gaussian with itself → Gaussian

Convolving 2 times Gaussian kernel of width σ is Same as Convolving once with width $\sigma\sqrt{2}$

Seprable kernel → Factors into product 2 1-D Gaussians
Complexity → $2n$

العادي Complexity → n^2