(6abc)

Local Operators

⇒ Local operators are also called neighborhood operators.

⇒ We will look in several local operators
- Noise reduction
- Gaussian filter
- Gradients
- Edge detection

★ Moving Average / Box Filter

⇒ Replace a pixel value by the mean intensity value of the neighborhood.

$$g(i,j) = \frac{1}{KL} \sum_{K,l} f(i-K, j-l)$$

★ Kernel

⇒ We can formulate the box filter by using a weighting function w.

$$g(i,j) = \sum_{K,l} w(K,l)\, f(i-K, j-l)$$

⇒ This weighting function is called Kernel.

⇒ Linear filtering operators involve weighted combinations of pixels in (small) neighborhood.

★ Linear Filter

⇒ A filter L which transforms

$$g(i,j) = L(f(i,j))$$

is called linear and shift invariant if

$$L(\alpha_1 f_1 + \alpha_2 f_2) = \alpha_1 g_1 + \alpha_2 g_2$$

&

$$L(f(i-K, j-l)) = g(i-k, j-l)$$

* Convolution

⇒ Filters of the form

$$g(i,j) = \sum_{k,l} f(i-k, j-l)\,\omega(k,l)$$

are Convolutions of the function f with a Kernel function ω.

$$\boxed{g = f * \omega}$$

$$g = f * R_3^{(2)}$$

$(2) \rightarrow$ dimensionality

$R_3 \rightarrow$ Neighborhood

Example ⇒ $\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & \underline{1} & 1 \\ 1 & 1 & 1 \end{bmatrix}$

Element with index zero

* Noise

⇒ The Convolution with the described Kernels changes the noise of the signal.

$$\boxed{\sigma_{ng}^{2} = \sum_{i} (\omega(i))^2\, \sigma_{nf}^{2}}$$

{ Variance of Output Image }

{ Variance of Input Image }

⇒ For box filter, we have

$$\sigma_{n_g}(R_m^{(2)}) = \frac{1}{m}\sigma_{n_f}$$

★ **Median Filter**

⇒ Robust against outliers

⇒ No linear filter anymore.

★ **How to deal with the Borders**

⇒ **Padding options:**

- → Constant zero {0 for all outside pixels.}
- → Cyclic wrap {loop around the image}
- → Clamp {Repeat edge pixels indefinitely}
- → Mirror {Reflect pixel across the image edge}

★ **Binomial / Gaussian filter**

⇒ Performs a smoothing.

⇒ Smoothing using a Gaussian as the kernel function.

⇒ Discrete approximation due to pixels.

⇒ The decay of the weights approximates a Gaussian using the coefficients of a Binomial distribution $B(0.5, m)$

★ **Noise Reduction**

⇒ For the Gaussian filter, we obtain for the noise

$$\boxed{\sigma_{n_g}(B_m^{(2)}) = \frac{1}{\sqrt{\pi m}}\sigma_{n_f}}$$

★ **Degree of Smoothing** $= \frac{\sigma_{ng}}{\sigma_{nf}}$

⇒ Box filter smoothes more...

★ Convolution

$$\boxed{g = f * \omega}$$

Output Image   Input Image   Kernel

★ Defination

⇒ The discrete convolution of the functions $a(i)$ and $b(i)$ is defined as

$$C(i) = \sum_{K=-\infty}^{+\infty} a(K)\, b(i-K)$$

⇒ and in 2D as

$$C(i,j) = \sum_{K=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} a(K,l)\, b(i-K, j-l)$$

⇒ and is written as $C = a * b$

★ Commutative property

⇒ The Convolution is Commutative.

$$\boxed{a * b = b * a}$$

★ Associative property

$$a * b * c = (a * b) * c = a * (b * c)$$

* Distributive Property

$(a+b)*c = a*c + b*c$

* Scalar multiplication

$\lambda(a*b) = (\lambda a)*b = a*(\lambda b)$

* Neutral Element

$\boxed{a*\delta = a}$ {Unit impulse}

$$\delta = \begin{bmatrix} \cdots & \vdots & \cdots \\ \vdots & 1 & \vdots \\ \cdots & \vdots & \cdots \end{bmatrix}$$

* Multiplication as a Convolution

$121 * 121$

```
121
 242
  121
-----
14641
```

* De-Convolution

⇒ Given $c(x) = a(x) * b(x)$

we can recover $b(i)$ given $a^{-1}(i)$ by

$b(x) = a^{-1}(x) * c(x)$

* Separable Kernels

⇒ If a multi-dimensional Kernel, can be split into the individual dimensions, it is called separable.

→ Example:

$$B_2^{(2)} = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \frac{1}{4}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \frac{1}{4}[1\ 2\ 1] = B_2^{[i]} * B_2^{[j]}$$

★ Separable Kernels Allow for Efficient Computations

⇒ Two 1D convolutions are more efficient to compute than one 2D convolution.

$$g = \underbrace{R_n^{(2)} * f}_{O(n^2)\text{ operation}} = \underbrace{R_n^{[1]} * (R_n^{[1]} * f)}_{O(2n)\text{ operations}}$$

★ Multiple Convolutions

⇒ Smoothing filters have the property

$$\sum_i \omega(i) = 1$$

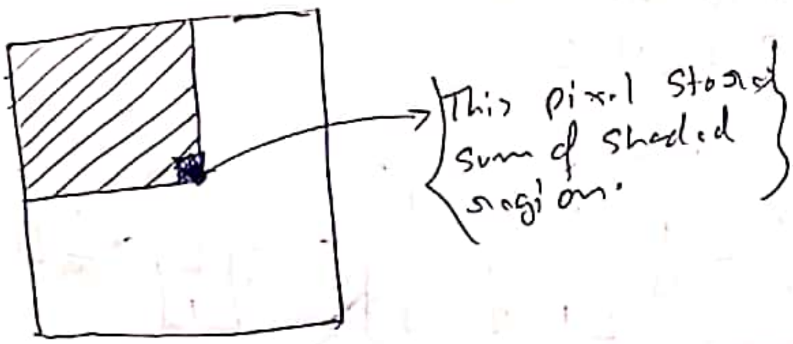
⇒ Thus, the concatenation of smoothing filters is again a smoothing filter.

$$\omega_n = \underbrace{\omega_1 * \cdots \omega_1}_{n \text{ times}}$$

★ Integral Image

⇒ An integral image is an image in which each pixel store a sum of intensity values of the form

$$\boxed{S(i,j) = \sum_{i'=1}^{i} \sum_{j'=1}^{j} f(i',j')}$$



⇒ Integral Image allows for effective computing the sum over intensities in any rectangle.

$$\boxed{S([i_0, i_1] \times [j_0, j_1]) = S(i_1, j_1) + S(i_0 - 1, j_0 - 1) - S(i_0 - 1, j_1) - S(i_1, j_0 - 1)}$$

★ Gradient Filters

$$f'(i) = \frac{\Delta f}{\Delta x} = \frac{f(i+1) - f(i-1)}{2\Delta x}$$

$$\Delta f = \frac{f(i+1) - f(i-1)}{2}$$

$$\text{So} \quad \Delta = \frac{1}{2}\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix}$$

$$\boxed{\Delta f(i) = \Delta * f}$$

⇒ In contrast to before, the weight vector contains negative weights and <u>sums up to 0.</u>

⇒ Gradient of the Image function

$$\nabla g = \nabla * g = \begin{bmatrix} g_x \\ g_y \end{bmatrix}$$

⇒ With the magnitude of the gradient

$$|\nabla g| = \sqrt{g_x^2 + g_y^2}$$

⇒ and the direction

$$\alpha = \arctan\left(\frac{g_y}{g_x}\right) = \operatorname{atan}(g_y, g_x)$$

★ Sobel Operator

⇒ The Sobel operator is a standard operator for gradients using a 3×3 window.

⇒ It is a combination of a Gaussian filter and the gradient

$$\Delta x = B_2^{[;]} * \frac{1}{2}\begin{bmatrix}1\\0\\-1\end{bmatrix} = \frac{1}{8}\begin{bmatrix}1 & 2 & 1\\0 & 0 & 0\\-1 & -2 & -1\end{bmatrix}$$

$$\Delta y = \frac{1}{8}\begin{bmatrix}1 & 0 & -1\\2 & 0 & -2\\1 & 0 & 1\end{bmatrix}$$

★ Scharr operator

⇒ Improved Sobel operator

$$\Delta x = \frac{1}{16}[3\ 10\ 3] * \frac{1}{2}\begin{bmatrix}1\\0\\-1\end{bmatrix}$$

$$\Delta x = \frac{1}{32}\begin{bmatrix}3 & 10 & 3\\0 & 0 & 0\\-3 & 10 & -3\end{bmatrix}$$

$$\Delta y = \frac{1}{32}\begin{bmatrix}3 & 0 & -3\\10 & 0 & -10\\3 & 0 & -3\end{bmatrix}$$

⇒ 10-time more accurate than Sobel in determining the gradient direction.

★ 2nd Derivative

$$\frac{\delta^2 f}{\delta x^2} = \frac{\delta^2}{\delta x^2} * f = \left(\frac{\delta}{\delta x} * \frac{\delta}{\delta x}\right) * f$$

$$\begin{bmatrix}1\\-1\end{bmatrix} * \begin{bmatrix}1\\-1\end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix}1\\-2\\1\end{bmatrix}$$

Scanned by CamScanner

⇒ The second derivative are given through the Hessian matrix of f.

$$H(f) = [h(f)_{i,j}] = \begin{bmatrix} \frac{\delta^2 f}{\delta x^2} & \frac{\delta^2 f}{\delta x \delta y} \\ \frac{\delta^2 f}{\delta y \delta x} & \frac{\delta^2 f}{\delta y^2} \end{bmatrix}$$

$$\frac{\delta^2}{\delta x^2} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} * \frac{1}{4}[1\ 2\ 1]$$

$$\frac{\delta^2}{\delta x \delta y} = \frac{1}{2}\begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * \frac{1}{2}[1\ 0\ -1]$$

$$\frac{\delta^2}{\delta y^2} = \frac{1}{4}\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * [1\ -2\ 1]$$

★ <u>Laplace Operator</u>

⇒ The Laplace operator is useful for edge detection and is defined as

$$\Delta_L = \frac{\delta^2}{\delta x^2} + \frac{\delta^2}{\delta y^2} = \frac{1}{2}\begin{bmatrix} 1 & 0 & 1 \\ 0 & -4 & 0 \\ 1 & 0 & 1 \end{bmatrix}$$

$$\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ -2 & -4 & -2 \\ 1 & 2 & 1 \end{bmatrix} \qquad \frac{1}{4}\begin{bmatrix} 1 & -2 & 1 \\ 2 & -4 & 2 \\ 1 & -2 & 1 \end{bmatrix}$$

⇒ A smoother variant of the Laplace operator is

$$\Delta_L = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & -12 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Scanned by CamScanner

* Edge Detection

=> An edge is given by a change from dark to bright (or Vice-Versa)

=> There are points with

$$\frac{\delta^2 f(x)}{\delta x^2} = f'' = 0$$

and $f''' \neq 0$