1 Spatial Smoothing Filters

Spatial smoothing filters are a concept in image processing used to reduce noise and smooth out variations in an image by averaging or combining pixel values in a local neighborhood.

Types of Filters:

1. **Averaging Filter:** Replaces each pixel with the average of its neighbours.

$$\omega_1 = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

2. **Gaussian Filter:** Weights the neighbors according to a Gaussian distribution, giving higher weight to pixels closer to the center.

$$h(m,n) = K\, e^{-(m^2+n^2)}, \qquad m,n \in \{-1,0,1\}.$$

K is chosen such that

$$\sum_{m,n} h(m,n = 1$$

or

$$K = \left(\sum_{m=-1}^{1}\sum_{n=-1}^{1} e^{-(m^2+n^2)}\right)^{-1}.$$

3. **Median Filter:** A nonlinear filter that replaces each pixel with the median of its neighborhood. Useful for removing *salt-and-pepper noise.*

4. **Weighted Average Filter:** Similar to the averaging filter, but assigns more importance to the central pixel. Example:

$$\omega_3 = \frac{1}{10}\begin{bmatrix}1 & 1 & 1\\ 1 & 2 & 1\\ 1 & 1 & 1\end{bmatrix}$$

2 Applications of Spatial Smoothing Filters

Spatial smoothing filters are widely used in image processing tasks. Some common applications include:

1. **Denoising:** By averaging neighboring pixels, smoothing filters reduce the impact of random noise, making the image cleaner for further processing.
2. **Preprocessing before Binarization:** Smoothing helps suppress small intensity variations so that threshold-based binarization produces cleaner object boundaries.
3. **Edge Detection Preprocessing:** Many edge detection algorithms (such as Canny) first apply Gaussian smoothing to reduce noise. This ensures that false edges caused by random pixel fluctuations are minimized.
4. **Image Pyramid / Downsampling:** Before reducing image resolution, Gaussian smoothing is applied to avoid aliasing. This is a fundamental step in building image pyramids for multi-scale analysis (e.g., object detection and image compression).

3 Spatial Sharpening Filters

$$\frac{\partial f}{\partial x} = f(m+1,n) - f(m,n) = \nabla f_x(m,n) \rightarrow \textit{Image gradient in x-direction}$$

$$\frac{\partial f}{\partial y} = f(m,n+1) - f(m,n) = \nabla f_y(m,n) \rightarrow \textit{Image gradient in y-direction}$$

$$\begin{aligned}\frac{\partial^2 f}{\partial x^2} &= \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial x}\right) = (f(m+1,n) - f(m,n)) - (f(m,n) - f(m-1,n)) \\ &= f(m+1,n) - 2f(m,n) + f(m-1,n) \\ \frac{\partial^2 f}{\partial y^2} &= \frac{\partial}{\partial y}\left(\frac{\partial f}{\partial y}\right) = f(m,n+1) - 2f(m,n) + f(m,n-1)\end{aligned}$$

4 Laplacian of an Image

The Laplacian is a second-order derivative operator that highlights regions of rapid intensity change in an image. It works by comparing a pixel's value with its neighbors using convolution masks, making edges stand out while smooth regions stay near zero. At edges, it produces strong positive or negative responses, and zero-crossings indicate the actual edge locations. Because it amplifies rapid changes, the Laplacian is highly sensitive to noise. To reduce this effect, it is often combined with Gaussian smoothing, leading to the Laplacian of Gaussian (LoG) operator. The equation for the Laplacian is given by

$$\nabla^2 f = \nabla^2 f_x + \nabla^2 f_y$$

$$= \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$= f(m+1,n) + f(m-1,n) + f(m,n+1) + f(m,n-1) - 4f(m,n)$$

- 3×3 **filter (Isotropic):**

$$h(m,n) \quad = \quad \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

- **Another variant:**

$$h(m,n) \quad = \quad \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad \rightarrow \quad \textit{Smooth regions will have zero convolved value}$$

Application of Laplacian

- Highlight intensity discontinuity.
- Deemphasize regions with slowly varying intensity levels.

Image Sharpening using Laplacian Filters

The sharpened image is given by:

$$g(x,y) = f(x,y) + c\,\nabla^2 f(x,y)$$

where $c = -1$ if the central coefficient of the Laplacian filter is negative.

In terms of convolution:

$$g(m,n) = f(m,n) + c\,[f(m,n) * h(m,n)]$$

$$= f(m,n) + f(m,n) * h'(m,n)$$

Expanding the discrete Laplacian:

$$g(m,n) = f(m,n) + \Big[4f(m,n) - \big(f(m+1,n) + f(m-1,n) + f(m,n+1) + f(m,n-1)\big)\Big]$$

Unsharp Masking and High-Boost Filtering

Let $\bar{f}(x,y)$ be a blurred (unsharp/smoothed) version of the original image $f(x,y)$.

$$g_{\text{mask}}(x,y) = f(x,y) - \bar{f}(x,y)$$

The sharpened image is obtained as:

$$g(x,y) = f(x,y) + k\, g_{\text{mask}}(x,y)$$

where

$$k = 1 \quad \Longrightarrow \quad \text{Unsharp Masking (basic sharpening)}$$

$$k > 1 \quad \Longrightarrow \quad \text{High-Boost Filtering (stronger sharpening effect)}$$

5 Geometric Operation

We have an input image $I(i,j)$ and the output image is defined as

$$J(i,j) = I(a_1(i,j), a_2(i,j)).$$

- **Image Translation**

  $$a_1(i,j) = i - b_1, \quad a_2(i,j) = j - b_2$$
  $$J(i,j) = I(i - b_1,\, j - b_2)$$

  For example, if $b_1 = b_2 = 2$:

  $$J(2,2) = I(0,0), \quad J(0,0) = I(-2,-2) = 0 \quad \text{(outside image} \to 0\text{)}.$$

- **Image Rotation**
  A counter-clockwise rotation by angle θ is given by:

  $$a_1(i,j) = i\cos\theta - j\sin\theta$$

  Rotation of axis in counter-clockwise direction

  $$a_2(i,j) = i\sin\theta + j\cos\theta$$