

Image sharpening in the frequency domain

Goal: Enhance image so that it shows more obvious edges.

Method 1: Laplacian masking

Recall that : $\Delta f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$.

In the discrete case, $\Delta f(x, y) \approx f(x+1, y) + f(x, y+1) + f(x, y-1) + f(x-1, y) - 4f(x, y)$
or $\Delta f \approx p * f$ where $p = \begin{pmatrix} 1 & 1 & 1 \\ 1 & -4 & 1 \\ 1 & 1 & 1 \end{pmatrix}$

We can observe that $-\Delta f$ captures the edges of the image
(leaving other region zero)

i.e. Sharpen image = $f + \underbrace{(-\Delta f)}_{\text{add more edges}}$ $\stackrel{p * f}{\approx}$

In the frequency domain: $DFT(g) = DFT(f) - DFT(\Delta f)$
 $= DFT(f) - c DFT(p).DFT(f)$

$\therefore DFT(g) = [1 - H_{\text{Laplacian}}(u, v)] DFT(f)(u, v)$
 $c DFT(p)$

Method 2: Unsharp masking

Idea: Add high-frequency component

Definition: Let f = input image (blurry)

Let f_{smooth} = smoother image (using mean filter / Gaussian filter etc)

Define a sharper image as:

$$g(x, y) = f(x, y) + k(f(x, y) - f_{\text{smooth}}(x, y))$$

When $k=1$, the method is called unsharp masking.

When $k>1$, the method is called highboost filtering.

In the frequency domain, let $\text{DFT}(f_{\text{smooth}})(u, v) = \underbrace{H_{LP}(u, v)}_{\text{Low-pass filter}} \text{DFT}(f)(u, v)$

$$\text{Then: } \text{DFT}(g) = [1 + k(1 - H_{LP}(u, v))] \text{DFT}(f)(u, v)$$

Image denoising in the spatial domain

Definition: Linear filter = modify pixel value by a linear combination of pixel values of local neighbourhood.

Example 1: Let f be an $N \times N$ image. Extend the image periodically. Modify f to \tilde{f} by:

$$\tilde{f}(x, y) = f(x, y) + 3f(x - 1, y) + 2f(x + 1, y).$$

This is a linear filter.

Example 2: Define

$$\tilde{f}(x, y) = \frac{1}{4} (f(x + 1, y) + f(x - 1, y) + f(x, y + 1) + f(x, y - 1))$$

This is also a linear filter.

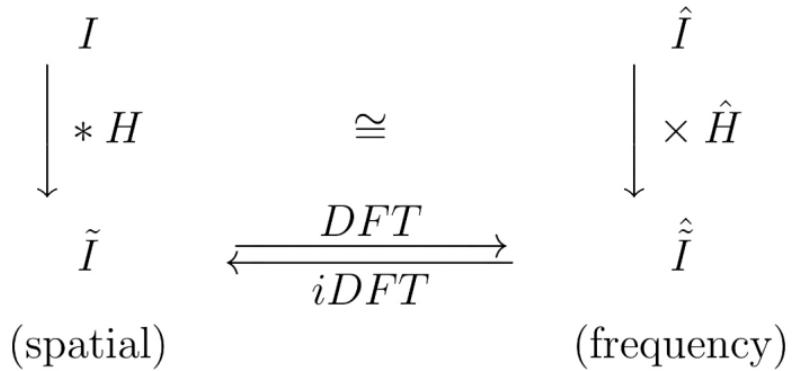
Recall: The discrete convolution is defined as:

$$I * H(u, v) = \sum_{m=-M}^M \sum_{n=-N}^N I(u-m, v-n)H(m, n)$$

(Linear combination of pixel values around (u, v))

Therefore, **Linear filter is equivalent to a discrete convolution.**

Geometric illustration



Example 3: In Example 1, if f is defined on $[-M, M] \times [-N, N]$, then:

$$\tilde{f} = f * H$$

where

$$H = \begin{pmatrix} 0 & 2 & 0 \\ 0 & 1 & 0 \\ 0 & 3 & 0 \end{pmatrix}$$

In Example 2, $\tilde{f} = f * H$ where

$$H = \frac{1}{4} \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

H is called the filter

Commonly used filter (linear)

- Mean filter:

$$H = \frac{1}{9} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

(Here, we only write down the entries of the matrix for indices $-1 \leq k, l \leq 1$ for simplicity. All other matrix entries are equal to 0.)

This is called the *mean filtering with window size 3×3* .

- **Gaussian filter:** The entries of H are given by the Gaussian function $g(r) = \exp\left(-\frac{r^2}{2\sigma^2}\right)$, where $r = \sqrt{x^2 + y^2}$.

Properties of linear filtering

- **Associativity:** $A * (B * C) = (A * B) * C$
- **Commutativity:** $I * H = H * I$
- **Linearity:**

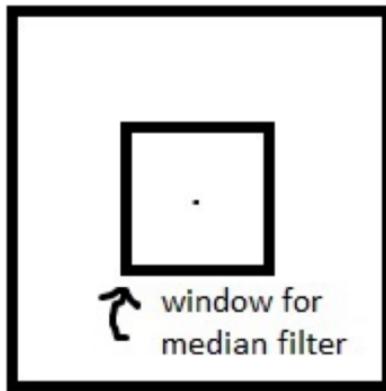
$$(s \cdot I) * H = I * (s \cdot H) = s \cdot (I * H)$$

$$(I_1 + I_2) * H = (I_1 * H) + (I_2 * H)$$

Remark: Convolution of Gaussian with a Gaussian is also a Gaussian
 \therefore Successive Gaussian filter = Gaussian filter with larger σ .

Non-linear spatial filter

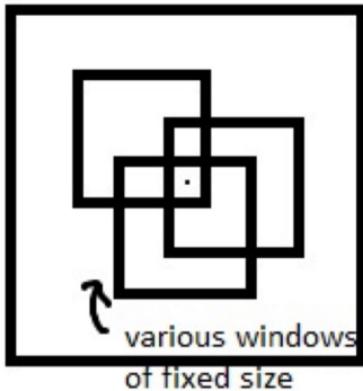
- Median filter



Take a window with center at pixel (x_0, y_0) . Update the pixel value at (x_0, y_0) from $I(x_0, y_0)$ to $\tilde{I}(x_0, y_0) = \text{median}(I \text{ within the window})$

Example 4: If pixel values within a window is 0, 0, 1, 2, 3, 7, 8, 9, 9, then the pixel value is updated as 3 (median).

- Edge-preserving filter



- **Step 1:** Consider all windows with certain size around pixel (x_0, y_0) (not necessarily be centered at (x_0, y_0));
- **Step 2:** Select a window with minimal variance;
- **Step 3:** Do a linear filter (mean filter, Gaussian filter and so on).

Lecture 18:

- Non-local mean filter

Let g be a $N \times N$ image.

$X = (x, y)$ } Two pixels.
 $X' = (x', y')$

Define: $S_x = \{(x+s, y+t) : -a \leq s, t \leq a\}$; $S_{x'} = \{(x'+s, y'+t) : -a \leq s, t \leq a\}$

Define: $g_x = g|_{S_x}$ and $g_{x'} = g|_{S_{x'}}$.

$\underbrace{(2a+1)}_m \times \underbrace{(2a+1)}_m$ image

Let \tilde{g}_x = smoothed image of g_x by Gaussian smoothing

$\tilde{g}_{x'}$ = smoothed image of $g_{x'}$ by Gaussian smoothing.

Define the weight: $w(x, x') = e^{-\frac{\|\tilde{g}_x - \tilde{g}_{x'}\|_F^2}{\sigma^2}}$

(small when x and x' are far away) noise level parameter

Non-local mean filter of g :

$$\hat{g} = \frac{\sum_{x' \in \text{image domain}} w(x, x') g(x')}{\sum_{x' \in \text{image domain}} w(x, x')}$$

far away in
term of small
images

Image denoising by solving Anisotropic heat diffusion

Consider the PDE:

$$(*) \quad \frac{\partial I(x, y, \sigma)}{\partial \sigma} = \sigma \left[\frac{\partial^2 I(x, y, \sigma)}{\partial x^2} + \frac{\partial^2 I(x, y, \sigma)}{\partial y^2} \right] = \sigma \nabla \cdot (\nabla I)$$

$$(\nabla \cdot = \text{divergence}; \nabla \cdot (v_1, v_2) = \frac{\partial v_1}{\partial x} + \frac{\partial v_2}{\partial y}) \quad (\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right))$$

Then: $g(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2}$ satisfies (*).

Observation: We'll see that Gaussian filter is approximately solving (*).

Given an image $I(x, y)$ (Assume I is continuously defined on the whole 2D domain)

Gaussian filter = convolution of I with the Gaussian function:

$$\tilde{I}(x, y, \sigma) = I * g(x, y, \sigma) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(x-u, y-v) I(u, v) du dv$$

(Analogous to discrete convolution = $\int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(u, v; \sigma) I(x-u, y-v) du dv$)

$$\begin{aligned}
 \therefore \frac{\partial \tilde{I}}{\partial \sigma} &= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\partial g(u, v; \sigma)}{\partial \sigma} I(x-u, y-v) du dv \\
 &= \sigma \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\partial^2 g(u, v; \sigma)}{\partial u^2} I(x-u, y-v) du dv + \sigma \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \frac{\partial^2 g}{\partial v^2}(u, v; \sigma) \frac{\partial I}{\partial u}(x-u, y-v) du dv \\
 &= \sigma \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(u, v; \sigma) \frac{\partial I}{\partial x^2}(x-u, y-v) du dv + \sigma \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(u, v; \sigma) \frac{\partial I}{\partial y^2}(x-u, y-v) du dv \\
 &= \sigma \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \right) \iint_{-\infty}^{\infty} g(u, v; \sigma) I(x-u, y-v) du dv. \quad \begin{matrix} \text{(Using the fact:)} \\ \frac{\partial}{\partial x}(g * f) = \frac{\partial g}{\partial x} * f \end{matrix} \\
 &= \sigma \nabla \cdot (\nabla \tilde{I})(x, y, \sigma)
 \end{aligned}$$

\therefore Gaussian filter = solving "diffusion" eqt !!

Anisotropic diffusion for edge-preserving Image denoising

General "diffusion" eqn :

$$\frac{\partial I(x,y;\sigma)}{\partial \sigma} = \nabla \cdot (\underbrace{K(x,y)}_{\text{Controls the rate of diffusion}} \nabla I(x,y;\sigma))$$

• Controls the rate of diffusion

• Smaller K = smaller diffusion at (x,y)

Edge detector : Edge of an image can be detected by: $|\nabla I(x,y)|$.

If (x,y) is on the edge, $|\nabla I(x,y)|$ is big.

If (x,y) is in the interior region, $|\nabla I(x,y)| \approx 0$

\therefore Suitable $K(x,y)$ to preserve edge: 1. $K(x,y) = \frac{1}{|\nabla I(x,y)| + \epsilon^2}$ ↑
Avoid Singularity

2. $K(x,y) = e^{-|\nabla I(x,y;\sigma)|/b}$

i.e. The denoising problem can be written as:

$$\frac{\partial I(x, y; \sigma)}{\partial \sigma} = \nabla \cdot \left(e^{-\frac{|\nabla I(x, y; \sigma)|}{b}} \nabla I(x, y; \sigma) \right)$$

In the discrete case, we solve:

$$I^{n+1}(x, y) - I^n(x, y) = D_1 \left(e^{-\frac{|\nabla I^n(x, y)|}{b}} D_2 I^n(x, y) \right)$$

D_1 = matrix approximating $\nabla \cdot$.

D_2 = matrix approximating ∇

i.e. Starting with $I^0(x, y) = I(x, y)$,

we iteratively modify $I^n(x, y)$.

Recall:
 $\frac{\partial I}{\partial x}(x, y) \approx$

$I(x+1, y) - I(x, y)$ etc
underbrace
linear operator

Such a process is called Anisotropic diffusion image denoising.