Image Sciences in a nutshell

Image Processing
Image to Image

Imaging
Physics to Image

Computer Graphics
Symbols to Image

Computer Vision
Image to Symbols
Images as functions

Continuous

\[ f : \mathbb{R}^2 \rightarrow \mathbb{R}^d \]
\[ x = (h, v) \]

Discrete

\[ f : \mathbb{Z}^2 \rightarrow \mathbb{R}^d \]
\[ n = (n_1, n_2) \]

\( d=1 \): Gray
\( d=3 \): Color
Image Denoising
Image Denoising

Key assumption: clean image is smooth
Moving Average in 2D
Moving Average in 2D
Moving Average in 2D
Moving Average in 2D
Moving Average in 2D
Moving Average in 2D

Slide Source: S. Seitz
Denoising: input
Denoising: first application of averaging filter
Denoising: tenth application of denoising filter
Denoising: application of larger box filter
Weighted averaging

\[ g[n] = \sum_{m} w(|n - m|) f(m) \]

\[ w[k] = \begin{cases} 
  \frac{1}{2l+1}, & k \leq l \\
  0, & \text{otherwise}
\end{cases} \]
Weighting kernel

Gaussian function:

\[ g_\sigma(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \]

Standard deviation, \( \sigma \): determines spatial support

\[ \sigma = 2 \quad \sigma = 5 \quad \sigma = 10 \]
Moving average
Gaussian blur
Image Processing

\[ f \rightarrow \psi \rightarrow \psi(f) \]

image \quad filter \quad image
**Linear Image Processing**

**Linearity**
\[
\psi(\alpha f + \beta h) = \alpha \psi(f) + \beta \psi(h)
\]
\[
\psi \left( \sum_k \alpha_k f_k \right) = \sum_k \alpha_k \psi(f_k)
\]

**Translation Invariance**
\[
f^c(x) = f(x - c)
\]
\[
\psi(f^c) = [\psi(f)]^c
\]

**Linear, Translation-Invariant (LTI) system**
\[
f_k(x) \rightarrow g_k(x)
\]
\[
\sum_k \alpha_k f_k(x - c) \rightarrow \sum_k \alpha_k g_k(x - c)
\]
Linear Image Processing

$\psi(f)$

From time-invariance: useful bases.
Linear Image Processing

\[ f \rightarrow \psi \rightarrow \psi(f) \]

![Diagram](image)

\[ f(x) = \left\{ \begin{array}{c}
a_0 b_0(x) \rightarrow a_0 h_0(x) \\
+a_1 b_1(x) \rightarrow +a_1 h_1(x) \\
\vdots \\
+a_k b_k(x) \rightarrow +a_k h_k(x)
\end{array} \right\} = \psi(f)(x) \]

From time-invariance: useful bases.
Linear algebra reminder

\[ \mathbf{u} \in \mathbb{R}^N \]

Basis: N linearly independent vectors \( \{ \mathbf{v}_i \}, \ i = 1, \ldots, N \)

Expansion on basis:

\[ \mathbf{u} = \sum_i c_i \mathbf{v}_i \]

Orthonormal basis:

\[ \langle \mathbf{v}_i, \mathbf{v}_j \rangle = \begin{cases} 1, & i = j \\ 0, & \text{otherwise} \end{cases} \]

Expansion coefficients:

\[ \langle \mathbf{v}_i, \mathbf{u} \rangle = c_i \]

Expansion:

\[ \mathbf{u} = \sum_i \langle \mathbf{v}_i, \mathbf{u} \rangle \mathbf{v}_i \]
Canonical basis

\[
\begin{bmatrix}
  u_1 \\
  u_2 \\
  u_3
\end{bmatrix} = u_1 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + u_2 \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + u_3 \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}
\]

\[
u = \sum_i u_i e_i = \sum_i \langle e_i, u \rangle e_i
\]
Canonical basis for 2D signals

Kronecker delta

\[ d_k[n] = \begin{cases} 
1, & n = k \\
0, & \text{otherwise}
\end{cases} \]
Canonical basis for 2D signals

Kronecker delta

\[ d_k[n] = \begin{cases} 
1, & n = k \\ 
0, & \text{otherwise} 
\end{cases} \]
Canonical basis for 2D signals

Kronecker delta

\[ d_k[n] = \begin{cases} 
1, & n = k \\
0, & \text{otherwise}
\end{cases} \]
Canonical basis for signals: expansion

Signal expansion:

\[ g[n] = \sum_k c_k d_k[n] \]

Identify terms:

\[ g[k] = c_k \]

Rewrite:

\[ d_k[n] = d[n - k] \]

\[ d[n] = \begin{cases} 
1, & n = 0 \\
0, & \text{otherwise} 
\end{cases} \]

Unit sample function

Sifting property:

\[ g[n] = \sum_k g[k] d[n - k] \]
Canonical basis for signals and LTI filters

**unit sample**

\[ d[n] \rightarrow h[n] \quad \text{impulse response} \]

\[ d[n - k] \rightarrow h[n - k] \quad \text{Translation-invariance} \]

Any signal: \[ g[n] = \sum_{k} g[k] d[n - k] \]

By linearity:

\[ \psi (g) = \sum_{k} g[k] h[n - k] = g[n] \ast h[n] \]

**Convolution sum**

Output of any LSI filter for any input:

convolution of input with filter’s impulse response
Convolution – discrete and continuous

2D convolution sum:

\[ f[n_1, n_2] = \sum_{k_1, k_2} g[k_1, k_2] h[n_1 - k_1, n_2 - k_2] \]

\[ = g[n_1, n_2] * h[n_1, n_2] \]

2D convolution integral:

\[ f(x, y) = \iint g(a, b) h(x - a, y - b) dadb \]

\[ = g(x, y) * h(x, y) \]
Linear Image Processing

\[ f \rightarrow \psi \rightarrow \psi(f) \]

From time-invariance: useful bases.
**Associative property & efficiency**

Associative Property: \[ f * [g * h] = [f * g] * h \]

Separability of Gaussian:

```
Slow  
*   =  

Fast  
*   *  
```
**Associative property & accuracy**

Associative Property: \( f \ast [g \ast h] = [f \ast g] \ast h \)

Derivative of Gaussian:

\[
\begin{align*}
\text{exact} & \quad \ast & \quad \text{approximate} \\
\frac{\partial}{\partial x} & \quad \text{approximate}
\end{align*}
\]
Introduction to Linear Image Processing

**Associative property & multi-scale processing**

Associative Property: \[ f * (g * h) = (f * g) * h \]

Semi-group property of Gaussian:
Denoising: first application of averaging kernel
Denoising: 10th application of denoising kernel
Distributive property & efficiency

Distributive property: \[ f \ast (g + h) = f \ast g + f \ast h \]

Steerable filter: \[ g_\theta(x, y) = \cos(\theta)g_0(x, y) + \sin(\theta)g_{\pi/2}(x, y) \]

\[ I \ast g_\theta = \cos(\theta)(I \ast g_0) + \sin(\theta)(I \ast g_{\pi/2}) \]

Linear algebra reminder: eigenvectors

\( \mathbf{M} : N \times N \)

Eigenvectors: \( \mathbf{M} \mathbf{v}_i = \lambda_i \mathbf{v}_i, \ i = 1, \ldots, N \)

Full-rank, real and symmetric: eigenbasis

\[
\mathbf{u} = \sum_k \langle \mathbf{v}_k, \mathbf{u} \rangle \mathbf{v}_k
\]

\[
\mathbf{M} \mathbf{u} = \sum_k c_k \mathbf{M} \mathbf{v}_k = \sum_k c'_k \lambda_k \mathbf{v}_k
\]

\[
\mathbf{M} (\mathbf{M} \mathbf{u}) = \sum_k c_k \lambda_k^2 \mathbf{v}_k
\]
Eigenvectors and eigenfunctions

Eigenvector: \[ Mv = \lambda v \]

Eigenfunction: \[ \psi (b) = \lambda b \]

Input: \[ f = \sum_{k} a_k b_k \]

Output: \[ \psi (f) = \sum_{k} a_k \psi (b_k) \]

\[ f \leftrightarrow \{ a_k \} \quad \psi (f) \leftrightarrow \{ a_k \lambda_k \} \]
Eigenfunctions for LTI filters

LTI filter: \( \psi (g)[n] = \sum_k h[k]g[n-k] \)

Let's guess: \( b_\omega[n] = \exp(j\omega n) = \cos(\omega n) + j\sin(\omega n) \)

It works: \( \psi (b_\omega)[n] = \sum_k h[n]b_\omega[n-k] \)

\[
= \sum_k h[k] \exp(j\omega[n-k]) \\
= \sum_k h[k] \exp(-j\omega k) \exp(j\omega n) \\
= H(\omega)b_\omega[n]
\]

Frequency response: \( H(\omega) \doteq \sum_k h[k] \exp(-j\omega k) \)
Expansion on harmonic basis

From orthonormality: \[ u = \sum_{k} \langle u, v_k \rangle v_k \]

Inner product for complex functions: \[ \langle f, g \rangle = \sum_{n} f[n] g^*[n] \]

Discrete-time: \[ F(\omega) \triangleq \langle f, b_\omega \rangle = \sum_{n} f[n] e^{-j\omega n} \]

Continuous-time: \[ F(\omega) \triangleq \int_{-\infty}^{\infty} f(t) \exp(-j\omega t) dt \]
Change of basis

Canonical expansion: \( u = \sum_k u_k e_k \)

Eigenbasis expansion: \( u = \sum_k \langle u, v_k \rangle v_k \)

Rotation matrix from eigenbasis:
\( c^T = u^T \begin{bmatrix} v_1 & \cdots & v_N \end{bmatrix} \)

Fourier transform: change of basis
Rotation from canonical basis to eigenfunction basis
Fourier Analysis

\[ F(\omega_1) \cdot e^{j\omega_1 x} + F(\omega_2) \cdot e^{j\omega_2 x} + F(\omega_K) \cdot e^{j\omega_K x} \]
Fourier synthesis equation

Continuous-time:

\[ f(x, y) = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} F(\omega_1, \omega_2) e^{j(\omega_1 x + \omega_2 x)} \, dx \, dy \]

Discrete-time:

\[ f[n, m] = \frac{1}{4\pi^2} \int_{0}^{2\pi} \int_{0}^{2\pi} F(\omega_1, \omega_2) e^{j(\omega_1 n + \omega_2 m)} \, d\omega_1 \, d\omega_2 \]
Convolution theorem of Fourier transform

Input expansion:
\[ f[n] = \int_\omega F(\omega) e^{j\omega n} d\omega \]

Output:
\[ \psi(f)[n] = \int_\omega F(\omega) \psi(e^{\omega n}) d\omega \]
\[ = \int_\omega F(\omega) H(\omega) e^{j\omega n} d\omega \]

\[ H(\omega) \overset{\text{def}}{=} \sum_k h[k] e^{-j\omega k} \]

Expansions:
\[ f[n] \leftrightarrow F(\omega) \]
\[ \psi(f)[n] \leftrightarrow F(\omega) H(\omega) \]
\[ f[n] * h[n] \leftrightarrow F(\omega) \cdot H(\omega) \]
Linear Image Processing

\[ f \rightarrow \psi \rightarrow \psi(f) \]

From time-invariance: useful bases.
Convolution theorem

\[ Y(\omega) = H(\omega)X(\omega) \]
Convolution theorem and efficiency

\[ Y(\omega) = H(\omega)X(\omega) \]

\[ O(NK) \rightarrow O(N \log N) \]
Gaussian blur
Moving average
**Modulation property and Gabor filters**

**Modulation property:**

\[ f(x) \leftrightarrow F(\omega) \]

\[ f(x)e^{j\omega_c x} \leftrightarrow F(\omega - \omega_c) \]

**Gaussian:**

\[ \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}} \leftrightarrow e^{-\frac{(\omega_x^2 + \omega_y^2)\sigma^2}{2}} \]
Modulation property and Gabor filters

Modulation property: \[ f(x) \leftrightarrow F(\omega) \]

\[ f(x)e^{j\omega_c x} \leftrightarrow F(\omega - \omega_c) \]

Gaussian:

\[ \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \leftrightarrow e^{-\frac{(\omega_x^2 + \omega_y^2)\sigma^2}{2}} \]

Gabor:

\[ \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} e^{j(\omega_x^c x + \omega_y^c y)} \leftrightarrow e^{-j\frac{\sigma^2((\omega_x - \omega_x^c)^2 + (\omega_y - \omega_y^c)^2)}{2}} \]
2D Gabor filterbank

Consider many combinations of $|\omega|$ and $\angle \omega$

Increasing $|\omega|$

Increasing $|\omega|$  
Frequency response isocurves
2D Gabor filterbank and texture analysis
2D Gabor filterbank and texture analysis
Summary

- Linear Time-Invariant filters
- Convolution
- Fourier Transform
- (Derivative-of) Gaussian filters
- Steerable filters
- Gabor filters

Thursday’s lecture: Pyramids, Scale-Invariant Blobs/Ridges, SIFT, HOG, Log-polar features, Harmonic analysis on surfaces...

Further reading:

Fast recursive filters:
Recursively implementing the Gaussian and its Derivatives - R. Deriche, 1993
Recursive implementation of the Gaussian filter. I. Young, L. Vliet, 1995
Fast IIR Isotropic 2D Complex Gabor Filters with Boundary Initialization, A Bernardino, J. Santos-Victor, TIP, 2006

Wavelets:
A Wavelet Tour of Signal Processing, S. Mallat, 2008