Linear Filters

DD2423 Image Analysis and Computer Vision

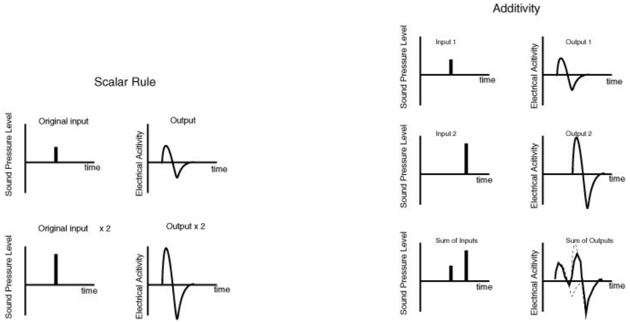
Mårten Björkman

Robotics, Perception and Learning Division
School of Electrical Engineering and Computer Science

November 5, 2021

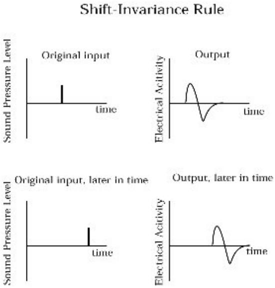
Image processing can be modeled by utilizing linear systems theory. A linear operator obeys the principle of superposition:

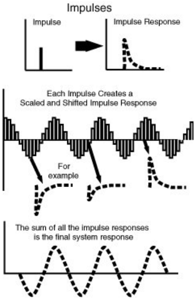
- Homogeneity (scalar rule): an increase in strength of the input, increases the output/response for the same amount.
- Additivity: if the input consists of two signals, the output/response is equal to the sum of the individual responses.



Additional properties:

- Shift-invariance: If a system is given two impulses with a time delay, the response remains the same except for time difference.
- Signals can be represented as sums of impulses of different strenghts (image intensities), shifted in time (image space).
- If we know how system responds to an impulse, we know how it reacts to combination of impulses: **impulse-response function**.





Notation

- Assume f and f' are 2D images, then $f \overset{\mathcal{L}}{\Longrightarrow} f' = \mathcal{L}(f)$, where $\mathcal{L}$ is an operator that "converts" the input f into the output f'.

Linear operator $\mathcal{L}$ satisfies

- Homogeneity: $\mathcal{L}(\alpha\, f(x,y)) = \alpha\mathcal{L}(f(x,y))$; $\alpha \in \mathbb{R}$
- Additivity: $\mathcal{L}(f(x,y) + g(x,y)) = \mathcal{L}(f(x,y)) + \mathcal{L}(g(x,y))$; $x, y \in \mathbb{R}$

Given

- $g \rightarrow \boxed{\quad \mathcal{L} \quad} \rightarrow \mathcal{L}(g)$
- $f \rightarrow \boxed{\quad \mathcal{L} \quad} \rightarrow \mathcal{L}(f)$

we have

- $(\alpha f + \beta g) \rightarrow \boxed{\quad \mathcal{L} \quad} \rightarrow \alpha\mathcal{L}(f) + \beta\mathcal{L}(g)$

Linear Shift Invariant Systems

$\mathcal{L}$ is called shift-invariant, if and only if a shift (translation) of the input causes the same shift of the output:

$$f(x,y) \rightarrow \boxed{\quad \mathcal{L} \quad} \rightarrow \mathcal{L}(f(x,y))$$

$$f(x-x_0, y-y_0) \rightarrow \boxed{\quad \mathcal{L} \quad} \rightarrow \mathcal{L}(f(x-x_0, y-y_0))$$

Alternative formulation: $\mathcal{L}$ commutes with a shift operator $\mathcal{S}$

$$\rightarrow \boxed{\quad \mathcal{L} \quad} \rightarrow \boxed{\quad \mathcal{S} \quad} \rightarrow \text{ same as } \rightarrow \boxed{\quad \mathcal{S} \quad} \rightarrow \boxed{\quad \mathcal{L} \quad} \rightarrow$$

$$\begin{tikzcd}
\mathbf{S}(\mathbf{f}) \arrow[r, "\boldsymbol{L}"] & \textbf{same!} \\
\mathbf{f} \arrow[u, "\mathbf{S}"] \arrow[r, "\boldsymbol{L}"] & \boldsymbol{L}(\mathbf{f}) \arrow[u, "\mathbf{S}"']
\end{tikzcd}$$

Linear filtering in image processing

Using digital linear filters to modify pixel values based on some pixel neighborhoods. Linear means linear combination of neighbors.

- Linear methods simplest.
- Can combine linear methods in any order to achieve same result.
- May be easier to invert.

Useful to:

- Integrate information over larger regions.
- Blur images to get rid of noise.
- Detect changes (edge detection).

Linear image filtering

- Estimate an output image by modifying pixels in the input image using a function of a local pixel neighborhood.
- The neighborhood and the corresponding linear weights per pixel is called a convolution kernel.

$$\begin{array}{|c|c|c|} \hline 9 & 5 & 3 \\ \hline 4 & 5 & 1 \\ \hline 1 & 1 & 7 \\ \hline \end{array} \quad \xrightarrow{\text{some function}} \quad \begin{array}{|c|c|c|} \hline & & \\ \hline & 9 & \\ \hline & & \\ \hline \end{array}$$

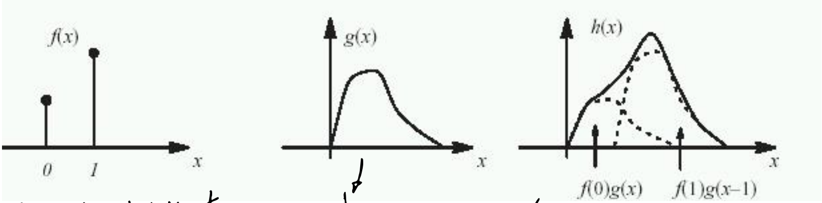
$$\begin{array}{|c|c|c|} \hline 9 & 5 & 3 \\ \hline 4 & 5 & 1 \\ \hline 1 & 1 & 7 \\ \hline \end{array} * \begin{array}{|c|c|c|} \hline 0 & -1 & 0 \\ \hline -1 & 4 & -1 \\ \hline 0 & -1 & 0 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & & \\ \hline & 9 & \\ \hline & & \\ \hline \end{array}$$

Convolution

- Convolution is a tool to build linear shift invariant (LSI) filters.
- Mathematically, a convolution is defined as the integral over space of one function at α, times another function at $x-\alpha$.

$$\mathrm{f}(x) * \mathrm{g}(x) = \int_{\alpha \in \mathbb{R}^n} \mathrm{f}(\alpha)\mathrm{g}(x-\alpha)d\alpha = \mathrm{g}(x) * \mathrm{f}(x) = \int_{\alpha \in \mathbb{R}^n} \mathrm{g}(\alpha)\mathrm{f}(x-\alpha)d\alpha$$

- Convolution operation is commutative!

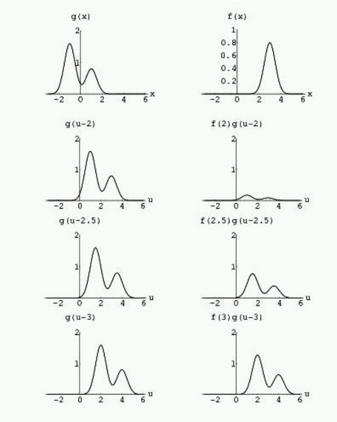


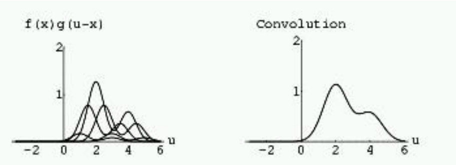
(two "points" at different space and intensity)

function s(x)

↳ s(x) at each point on x with the intensity told by f(x).

Way of considering convolution: weighted sum of shifted copies of one function, with weights given by the function value of the second function at the shift vector.





Theorem 1

Every shift invariant linear operator can be written as a convolution

$$\mathcal{L}(f) = g * f$$

- Continuous case

$$\mathcal{L}(f(x)) = \int_{\alpha \in \mathbb{R}^n} g(\alpha) f(x - \alpha) d\alpha$$

- Discrete case

$$\mathcal{L}(f(x)) = \sum_{\alpha \in \mathbb{R}^n} g(\alpha) f(x - \alpha)$$

Convolution (discrete case)

- The convolution of an image $f(x,y)$ with a kernel $g(x,y)$ is

$$f'(x,y) = g(x,y) * f(x,y) = \sum_{m=-M}^{M} \sum_{n=-N}^{N} g(m,n)\, f(x-m, y-n)$$

- Convolution kernel $g(x,y)$ represented as a matrix and is also called:
  - impulse response,
  - point spread function,
  - filter kernel,
  - filter mask,
  - template...

Convolution (filtering)

- Frame mask over image - multiply mask values by image values and sum up the results - a sliding dot product.

- For mathematical correctness: From the definition, the kernel first has to be flipped x-wise and y-wise. People are sloppy though.

Convolution: 1D example

If

$$F_1 = [1 \quad 2 \quad 3 \quad 4 \quad 5]$$
$$F_2 = [1 \quad 2 \quad 1 \quad 2 \quad 1]$$
$$G_1 = [-1 \quad 2 \quad -1]$$
$$G_2 = [1 \quad 2 \quad 3]$$

then

$$F_1 * G_1 = [-1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 6 \quad -5]$$
$$F_2 * G_1 = [-1 \quad 0 \quad 2 \quad -2 \quad 2 \quad 0 \quad -1]$$
$$F_1 * G_2 = [1 \quad 4 \quad 10 \quad 16 \quad 22 \quad 22 \quad 15]$$
$$F_2 * G_2 = [1 \quad 4 \quad 8 \quad 10 \quad 8 \quad 8 \quad 3]$$

Note1: outside the windows, values are assumed to be zero.
Note2: normally you assume $x = 0$ at center of filter kernel.

Convolution: 1D example

$$F_1 = [1 \quad 2 \quad 3 \quad 4 \quad 5]$$
$$G_2 = [1 \quad 2 \quad 3]$$

$$\begin{array}{cccccccc}
 & & & 1 & 2 & 3 & 4 & 5 \\
* & & & & & 1 & 2 & 3 \\
\hline
 & & & 3 & 6 & 9 & 12 & 15 \\
 & & 2 & 4 & 6 & 8 & 10 & \\
+ & 1 & 2 & 3 & 4 & 5 & & \\
\hline
 & 1 & 4 & 10 & 16 & 22 & 22 & 15
\end{array}$$

An easier way of doing it! Almost like regular multiplication.

Convolutions in Matlab and Python

In `Matlab` (and `SciPy.signal`) there are essentially two functions that can be used for image filtering.

- SciPy. `conv2 (convolve2d)` – a proper convolution according to the theory.

$$f'(x) = \sum_m g(m)\, f(x-m)$$

  Use it when you want to use stick as close as possible to the theory and exploit known mathematical properties of convolutions.

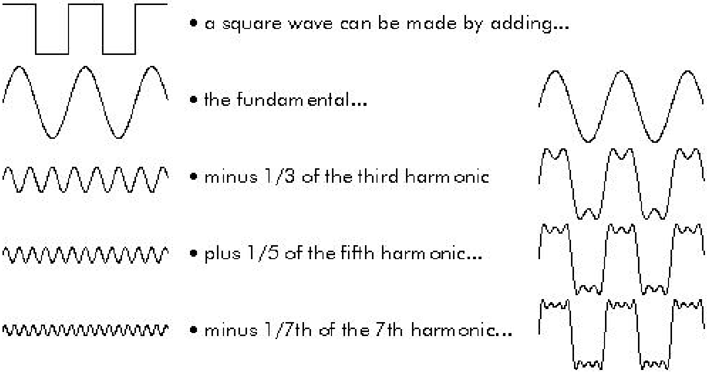
- MATLAB. `filter2 (correlate2d)` – does not flip the kernel before applying it.

$$f'(x) = \sum_m g(m)\, f(x+m)$$

  Use it when you want to match a known shape to the image data to see where in the image you have the best correlation.

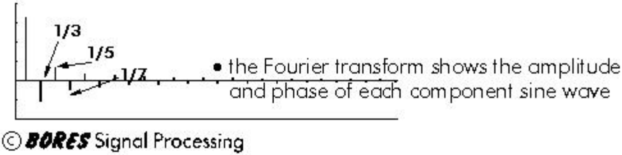
Signal decomposition

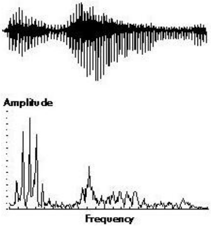
- In 1807 Jean Baptiste Fourier showed that any periodic signal could be represented by a series/sum of sine waves with appropriate amplitude, frequency and phase.



The Fourier transform

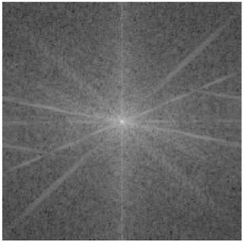
- The Fourier transform is a function that calculates the frequency, amplitude and phase of each sine wave needed to make up any given signal.
- The Fourier transform *converts* a signal (image) between its spatial and frequency domain representations.



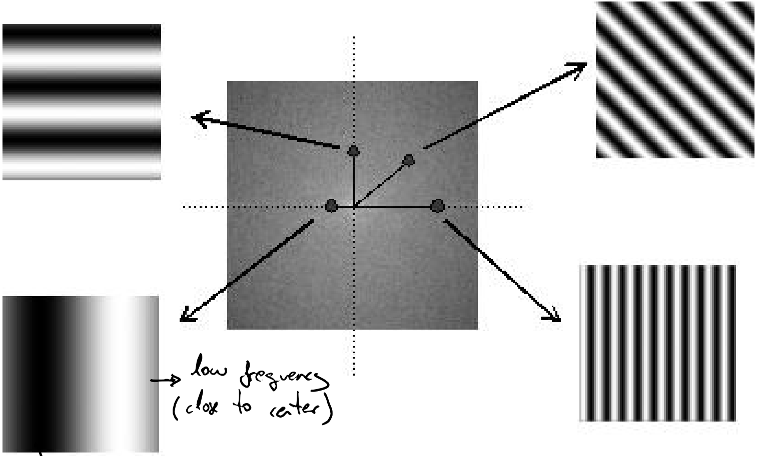


The Fourier transform

- The output of the Fourier transform represents the image in the frequency space.
- In the Fourier space image, each point represents a particular frequency contained in the original spatial domain image.
- The Fourier Transform is used in a wide range of applications, such as image analysis, image filtering, and image compression.

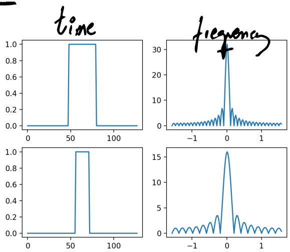
Example

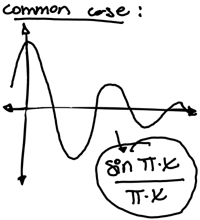


low frequency
(close to center)

starting black
(x axis, negative direction).

- The spatial frequency of an image refers to the rate at which the pixel intensities change.



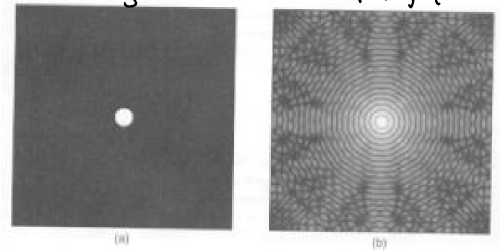


- A 1D pulse (left) and its Fourier transform (right). The center peak represents all uniform image areas. The second peak represents the width of the pulse.
- On the second row, the pulse is narrower and the rate of change is higher, and the Fourier transform is spread to higher frequencies.

2D example

- An image of a spot (left) and the Fourier Transform (right).

image $FT\{\cdot\}$



- The origin of the Fourier Transform is in the center of the image.
- Higher frequencies are heavily affected by image noise.

Fourier transform

$$\mathcal{F}(\mathrm{f}(x)) = \int_{x \in \mathbb{R}^n} \mathrm{f}(x) e^{-i\omega^T x} dx = \hat{\mathrm{f}}(\omega)$$

$$\mathcal{F}^{-1}(\hat{\mathrm{f}}(\omega)) = \frac{1}{(2\pi)^n} \int_{\underline{\underline{\omega \in \mathbb{R}^n}}} \hat{\mathrm{f}}(\omega) e^{\underline{\underline{i\omega^T}} x} d\omega = f(x)$$

frequency

$\oplus$ sign $\Rightarrow$ counter-clockwise.

$$e^{i\omega^T x} = \cos \omega^T x + i \sin \omega^T x$$

Terminology:

Frequency spectrum : $\hat{\mathrm{f}}(\omega) = Re(\omega) + i\ Im(\omega) = \mid \hat{\mathrm{f}}(\omega) \mid e^{i\phi(\omega)}$

Fourier spectrum: $\mid \hat{\mathrm{f}}(\omega) \mid = \sqrt{Re^2(\omega) + Im^2(\omega)}$

Power spectrum: $\mid \hat{\mathrm{f}}(\omega) \mid^2$

Phase angle: $\phi(\omega) = \arg \hat{\mathrm{f}}(\omega) = \tan^{-1} \frac{Im(\omega)}{Re(\omega)}$

- Angular frequency $\omega = (\omega_1 \;\; \omega_2)^T$

$$\omega_1 = \text{angular frequency in } x \text{ direction}$$
$$\omega_2 = \text{angular frequency in } y \text{ direction}$$

- Frequency

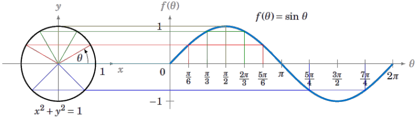
$$f = \frac{\omega}{2\pi}$$

- Wavelength

$$\lambda = \frac{2\pi}{\|\omega\|} = \frac{2\pi}{\sqrt{\omega_1^2 + \omega_2^2}}$$

space between peaks (between black/white bars).

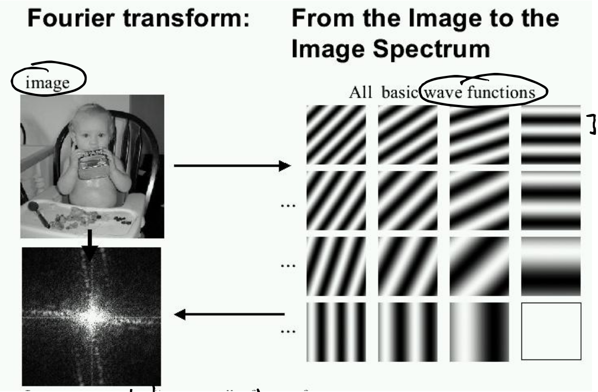
- Think of something spinning around the unit circle.



Basis functions - complex exponential functions

$$e^{i\omega^T x} = e^{i(\omega_1 x_1 + \omega_2 x_2)} = \cos\omega^T x + i\sin\omega^T x \quad \text{(Euler's formula)}$$

$$Re(e^{i\omega^T x}) = \cos(\omega^T x) \text{ and } Im(e^{i\omega^T x}) = \sin(\omega^T x), \; e_\omega : \mathbb{R}^2 \to \mathbb{C}$$



(all sine waves representation in 2D)

Image decomposition

Gradually reconstruct the image by summing up waveforms in the order to decreasing magnitudes

Change of basis functions

- An image can be viewed as a spatial array of gray level values, but can also thought of as a spatially varying function.
- Decompose the image into a set of orthogonal basis functions.
- When basis functions are combined (linearly) the original function will be reconstructed.
- Spatial domain: basis consists of shifted Dirac functions.
  Fourier domain: basis consists of complex exponential functions.
- The Fourier transform is “just” a change of basis functions.

Change of basis functions

- Assume you have a vector

$$\begin{pmatrix}1\\2\\3\\4\end{pmatrix} = 1\begin{pmatrix}1\\0\\0\\0\end{pmatrix} + 2\begin{pmatrix}0\\1\\0\\0\end{pmatrix} + 3\begin{pmatrix}0\\0\\1\\0\end{pmatrix} + 4\begin{pmatrix}0\\0\\0\\1\end{pmatrix}$$

- This may be expressed with another basis (e.g. Haar wavelet).

$$\begin{pmatrix}1\\2\\3\\4\end{pmatrix} = 2.5\begin{pmatrix}1\\1\\1\\1\end{pmatrix} - 1\begin{pmatrix}1\\1\\-1\\-1\end{pmatrix} - 0.5\begin{pmatrix}1\\-1\\0\\0\end{pmatrix} - 0.5\begin{pmatrix}0\\0\\1\\-1\end{pmatrix}$$

- The only condition is that the basis vectors are orthogonal.

Magnitude and Phase

- The Fourier coefficients $\hat{f}(\omega_1, \omega_2)$ are complex numbers, but it is not obvious what the real and imaginary parts represent.
- Another way to represent the data is with phase and magnitude.
- Magnitude:

$$|\hat{f}(\omega_1, \omega_2)| = \sqrt{Re^2(\omega_1, \omega_2) + Im^2(\omega_1, \omega_2)}$$

- Phase:

$$\phi(\omega_1, \omega_2) = \tan^{-1} \frac{Im(\omega_1, \omega_2)}{Re(\omega_1, \omega_2)}$$

Magnitude and Phase

Figure 4a

Original

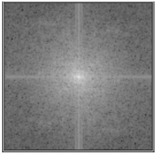


Figure 4b

$\log(|A(\Omega,\Psi)|)$

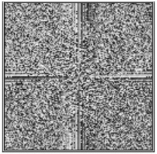


Figure 4c

$\varphi(\Omega,\Psi)$

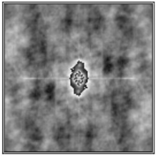


Figure 5a

$\varphi(\Omega,\Psi) = 0$

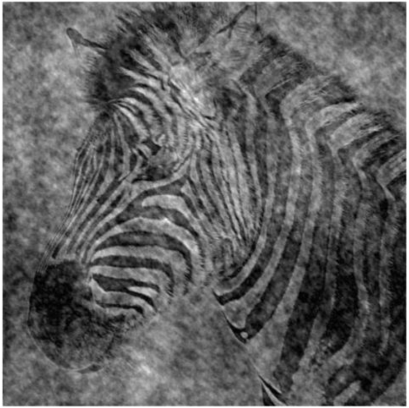
Figure 5b

$|A(\Omega,\Psi)| = constant$

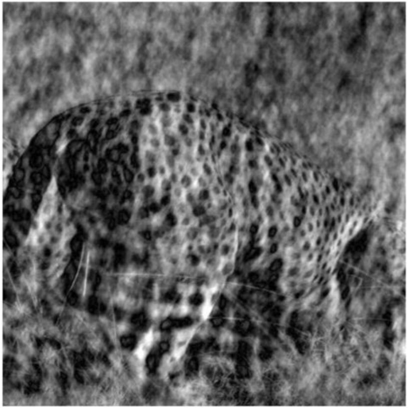
→ Keep phase (detect edges).

- Phase defines how waveforms are shifted along its direction
  Where edges will end up in the image (most important)
- Magnitude defines how large the waveforms are
  What grey-levels are on either side of edge (less important)

Magnitude and Phase



Phase of zebra - magnitude of tiger



Phase of tiger - magnitude of zebra

9

↳ FT'ing back allows to see zebra when phase (edges) are kept!

Why Fourier transforms?

- Why are we interested in a decomposition of an image into complex exponential functions?
- Sinusoids and cosinusoids are eigenfunctions of convolutions!

$$e^{i\omega t} \overset{\mathcal{L}}{\Longrightarrow} A(\omega)e^{i\omega t}$$

- Note: $A(\omega)$ is complex (change in magniture and phase).
- Thus we can understand what the filter does to the different frequencies of the image.

Theorem 2

- Convolution in the spatial domain is same as multiplication in the Fourier (frequency) domain

$$\mathcal{F}(h * f) = \mathcal{F}(h)\mathcal{F}(f)$$

$$f \rightarrow \boxed{\;*h\;} \rightarrow g=h*f \qquad \hat{f} \rightarrow \boxed{\;\hat{h}\;} \rightarrow \hat{g} = \hat{h}\,\hat{f}$$

Usage:

- For analysis and understanding of convolution operators.
- Some filters may be easily represented in the Fourier domain.
- Implementation: when size of the filter is too large it is more effective to use multiplication in the Fourier domain.

(convolution) (multiplication)

$$\mathcal{F}(h^*f) = \mathcal{F}(h)\mathcal{F}(f)$$

Proof:

$$\mathcal{F}(h^*f)(\omega) = \int_{x\in\mathbb{R}^n}(\int_{\alpha\in\mathbb{R}^n} h(x-\alpha)f(\alpha)d\alpha)e^{-i\omega^T x}dx \qquad \{\text{rewrite}\}$$

$$= \int_{\alpha\in\mathbb{R}^n}(\int_{x\in\mathbb{R}^n} h(x-\alpha)e^{-i\omega^T(x-\alpha)}dx)f(\alpha)e^{-i\omega^T\alpha}d\alpha \quad \{\text{with} \quad (x-\alpha)=\gamma\}$$

$$= \int_{\alpha\in\mathbb{R}^n}(\int_{\gamma\in\mathbb{R}^n} h(\gamma)e^{-i\omega^T\gamma}d\gamma)f(\alpha)e^{-i\omega^T\alpha}d\alpha \quad \{\text{separate}\}$$

$$= (\int_{z\alpha\in\mathbb{R}^n} h(\gamma)e^{-i\omega^T\gamma}d\gamma)(\int_{\alpha\in\mathbb{R}^n} f(\alpha)e^{-i\omega^T\alpha}d\alpha) = \mathcal{F}(h)(\omega)\mathcal{F}(f)(\omega)$$

Spatial separability

- Given

$$h(x,y) = h_1(x)h_2(y)$$

$$(h*f)(x,y) = \int_{\alpha\in\mathbb{R}^n}\int_{\gamma\in\mathbb{R}^n} h(\alpha,\gamma)f(x-\alpha,y-\gamma)\,d\alpha\, d\gamma$$

$$= \int_{\alpha\in\mathbb{R}^n} h_1(\alpha)\underbrace{\left(\int_{\gamma\in\mathbb{R}^n} h_2(\gamma)f(x-\alpha,y-\gamma)\,d\gamma\right)}_{***} d\alpha$$

  *** convolution of a column (fixed value of x) in y-direction
- If convolution mask $h(x,y)$ can be separated as above $\Rightarrow$ 2D convolution can be performed as a series of 1D convolutions.
- Discrete case: If the mask is m^2 in size $\Rightarrow$ $2m$ operations / pixel instead of m^2 operations per pixel.

In the Fourier domain

$$\hat{f} = \int_{\omega_1 \in \mathbb{R}^n} \int_{\omega_2 \in \mathbb{R}^n} f(x,y) e^{-i(\omega_1 x + \omega_2 y)} dxdy$$
$$= \int_{\omega_1 \in \mathbb{R}^n} e^{-i\omega_1 x} \left(\int_{\omega_2 \in \mathbb{R}^n} f(x,y) e^{-i\omega_2 y} dy \right) dx$$

- A Fourier transform in 2D can always be performed as a series of two 1D Fourier transforms.

Applications

Filtering techniques typically modify frequency characteristics:

- Remove noise (decrease high frequencies)
- Smooth (decrease high frequencies, increase low frequencies)
- Enhance edges (increase medium frequencies)

Images can be converted to their frequency component prior to filtering to facilitate direct manipulation of image frequency characteristics.

Spatial versus frequency domain

The Fourier Transform converts spatial image data into a frequency representation. Both representations contain equivalent information.

Spatial Domain

+ Intuitive Representation
- Designing filters can be hard
- Filtering with large kernels may result in long processing times.
+ Kernels applied directly to spatial data.

Frequency Domain

- Non-intuitive representation
+ Designing filters often easier.
+ Filtering with large kernels can be performed very quickly
- Image and Kernel must first be converted to frequency domain, modified, then reconverted.

Discrete Fourier Transform in 2D

$$\hat{f}(u,v) = \frac{1}{\sqrt{MN}} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m,n) e^{-2\pi i (\frac{mu}{M} + \frac{nv}{N})} \tag{1}$$

$$f(m,n) = \frac{1}{\sqrt{MN}} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \hat{f}(u,v) e^{+2\pi i (\frac{mu}{M} + \frac{nv}{N})} \tag{2}$$

Terminology:

- Fourier spectrum: $|F(u,v)| = \sqrt{Re^2(u,v) + Im^2(u,v)}$
- Phase angle: $\phi(u,v) = \tan^{-1} \frac{Im(u,v)}{Re(u,v)}$
- Power spectrum: $P(u,v) = |F(u,v)|^2 = Re^2(u,v) + Im^2(u,v)$
- The magnitude is simply the peak value, and the phase determines where the origin is, or where the sinusoid starts.

Relation continuous/discrete Fourier transform

- Continuous

$$\hat{f}(\omega) = \int_{x \in \mathbb{R}^n} f(x) e^{-i\omega^T x} dx$$

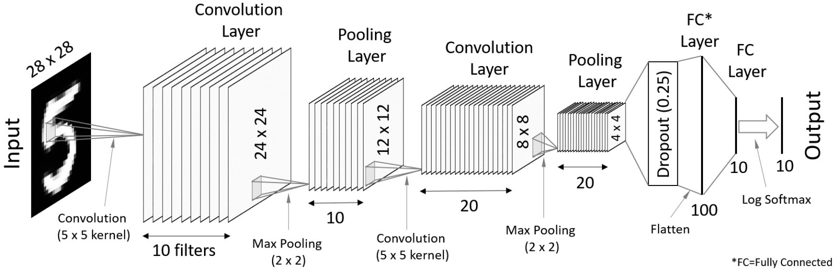
- Discrete

$$\hat{f}(u) = \frac{1}{\sqrt{M}^n} \sum_{x \in I^n} f(x) e^{-\frac{2\pi i u^T x}{M}}$$

- Frequency variables are related (in 1D) by

$$\omega = \frac{2\pi u}{M}$$

- Note: u assumes values $0 \ldots M-1 \Rightarrow \omega \in [0, 2\pi)$. → periodic (every 2π).
- By periodic extension, we can map this integral to $[-\pi, \pi)$.



- Convolution layers are based on convolutions

$$z_{n+1}^{c'} = f\,(\sum_{c} w_n^{c,c'} * z_n^{c} + b_n^{c'}\,)$$

with filter kernels $w_n^{c,c'}$ and neurons z_n^{c} organized in channels c.

Speeding up convolutional neural networks (CNN)

- Convolution layers are based on convolutions

$$z_{n+1}^{c'} = f\,(\sum_c w_n^{c,c'} * z_n^c + b_n^c)$$

- Computational cost is $\mathcal{O}(C'CN^2m^2)$ with image size $N \times N$, filter size $m \times m$, number of input channels C and output channels C'.
- This can be speeded up with convolution using Fourier transform, so that complexity becomes $\mathcal{O}(C'CN^2 \log N)$.
- Approximate speed-ups for 32×32 pixel images:

$$\times 1\ (m = 3), \times 2\ (m = 5), \times 5\ (m = 7)$$

Summary of good questions

- What properties does a linear system have?
- What does shift-invarience mean in terms of image filtering?
- How do you define a convolution?
- Why are convolutions important in linear filtering?
- How do you define a 2D Fourier transform?
- If you apply a Fourier transform to an image, what do you get?
- What information does the phase contain? And the magnitude?
- What is the Fourier transform of a convolution? Why important?
- What does separability of filters mean?

Readings

- Gonzalez and Woods: Chapters 3.4, 4.3-4.5
- Szeliski: Chapter 3.2
- Introduction to Lab 1