Summary of previous lecture

• Virtually all filtering is a local neighbourhood operation
• Convolution = linear and shift-invariant filters
  – e.g. mean filter, Gaussian weighted filter
  – kernel can sometimes be decomposed
• Many non-linear filters exist also
  – e.g. median filter, bilateral filter

• The Fourier transform decomposes a function (image) into trigonometric basis functions (sines & cosines).
• The Fourier transform is used to analyse frequency components of an image.
Linear neighbourhood operation

- For each pixel, multiply the values in its neighbourhood with the corresponding weights, then sum.
Convolution properties

- **Linear:**
  - Scaling invariant: \((C f) \otimes h = C (f \otimes h)\)
  - Distributive: \((f + g) \otimes h = f \otimes h + g \otimes h\)

- **Time Invariant:**
  \((= shift \ invariant)\)
  \(\text{shift} (f) \otimes h = \text{shift} (f \otimes h)\)

- **Commutative:**
  \(f \otimes h = h \otimes f\)

- **Associative:**
  \(f \otimes (h_1 \otimes h_2) = (f \otimes h_1) \otimes h_2\)
Convolution properties

- Convolving a function with a unit impulse yields a copy of the function at the location of the impulse.
- Convolving a function with a series of unit impulses “adds” a copy of the function at each impulse.
Fourier transform

\[ F(\omega) = \int_{-\infty}^{\infty} f(x) \, e^{-i\omega x} \, dx \]

\[ f(x) = F(\omega_1) e^{i\omega_1 x} + F(\omega_2) e^{i\omega_2 x} + F(\omega_3) e^{i\omega_3 x} + \ldots \]

\[ F(-\omega_1) = F^*(\omega_1) \]
Fourier transform pairs

Spatial
impulse

Frequency
1
cosine

2 impulses
sine

2 impulses
box

sinc

Notice the symmetry!
Gaussian

Gaussian
white noise

white noise
Properties of the Fourier transform

- **Spatial scaling**
- **Amplitude scaling**
- **Addition**
- **Translation**
- **Convolution**

\[
\mathcal{F}\{f \otimes h\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{h\}
\]

\[
\mathcal{F}\{f \cdot h\} = \mathcal{F}\{f\} \otimes \mathcal{F}\{h\}
\]
Today’s lecture

- The Discrete Fourier transform (DFT)
- The Fourier transform in 2D
- The Fast Fourier Transform (FFT) algorithm

- Designing filters in the Fourier domain
  - filtering out structured noise

- Sampling, aliasing, interpolation
Sampling

spatial domain  frequency domain

continuous function

sampling function

sampled function

\[ \Delta T \]

\[ 1/\Delta T \]
Discrete Fourier transform

- **Spatial domain**
  - Continuous function
  - Sampled function
  - Continuous image
  - Discrete image

- **Frequency domain**
Discrete Fourier transform

Continuous FT: \[ F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} \, dx \]

Discrete FT: \[ F[k] = \sum_{n=0}^{N-1} f[n] e^{-i \frac{2\pi}{N} kn} \]

- \( k \) is the spatial frequency, \( k \in [0, N-1] \)
- \( \omega = 2\pi k / N \)
- \( \omega \in [0, 2\pi) \)
Discrete Fourier transform

$F[k]$ is defined on a limited domain ($N$ samples), these samples are assumed to repeat periodically:

$$F[k] = F[k+N]$$

In the same way, $f[n]$ is defined by $N$ samples, assumed to repeat periodically:

$$f[n] = f[n+N]$$

$$F[k] = \sum_{n=0}^{N-1} f[n] e^{-i\frac{2\pi}{N} kn}$$
Discuss why the DFT only has positive frequencies!
What is the zero frequency?

Write out the value of \( F[0] \) for an input function \( f[n] \).

What does it mean?

\[
F[k] = \sum_{n=0}^{N-1} f[n] e^{-i \frac{2\pi}{N} kn}
\]
**Inverse DFT**

\[ F[k] = \sum_{n=0}^{N-1} f[n] e^{-i \frac{2\pi}{N} kn} \]

\[ f[n] = \frac{1}{N} \sum_{k=0}^{N-1} F[k] e^{i \frac{2\pi}{N} kn} \]

**Normalization**

**No minus sign**
Fourier transform in 2D, 3D, etc.

- Simplest thing there is! — the FT is separable:
  - Perform transform along x-axis,
  - Perform transform along y-axis of result,
  - Perform transform along z-axis of result, (etc.)

\[
F[u,v] = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f[n,m] e^{-i2\pi \left(\frac{un}{N} + \frac{vm}{M}\right)} = \sum_{m=0}^{M-1} \left(\sum_{n=0}^{N-1} f[n,m] e^{-i2\pi \frac{un}{N}}\right) e^{-i2\pi \frac{vm}{M}}
\]
2D Fourier transform pairs

sine

box
2D Fourier transform pairs

pillbox

Gauss

FT
2D Fourier transform pairs

lines

dots

FT
2D transform example 1
2D transform example 1
2D transform example 2
2D transform example 2
2D transform example 3
2D transform example 3
2D transform example 4
2D transform example 4
2D transform example 5
2D transform example 5
What is more important?

Jean Baptiste Joseph Fourier

magnitude

phase
What is more important?

magnitude

phase

[Image of a portrait of a person]

Filip Malmberg – 1TD396 – fall 2017 – filip.malmberg@it.uu.se
Computing the DFT

• For an image with N pixels, the DFT contains N elements.
• Each element of the DFT can be computed as a sum of all N elements in the image.
• A naive implementation of the DFT the requires $O(N^2)$ time.
• This is impractical!
The Fast Fourier Transform (FFT)

- Clever algorithm to compute the DFT.
- Runs in $O(N \log N)$ time, rather than $O(N^2)$ time.
- Because of symmetry of the forward and inverse Fourier transforms, FFT can also compute the IDFT.

$$F[k] = F_{\text{even}}[k] + F_{\text{odd}}[k]e^{-i\frac{2\pi}{N}k} \quad N = 2M$$

$$F[k+M] = F_{\text{even}}[k] - F_{\text{odd}}[k]e^{-i\frac{2\pi}{N}k}$$

$N = 2^n$
Convolution in the Fourier domain

- The Convolution property of the Fourier transform:
  \[ \mathcal{F} \{ f \otimes h \} = \mathcal{F} \{ f \} \cdot \mathcal{F} \{ h \} \]

- Thus we can calculate the convolution through:
  - \( F = \text{FFT}(f) \)
  - \( H = \text{FFT}(h) \)
  - \( G = F \cdot H \)
  - \( g = \text{IFFT}(G) \)

- Convolution is an operation of \( O(NM) \)
  - \( N \) image pixels, \( M \) kernel pixels

- Through the FFT it is an operation of \( O(N \log N) \)
  - Efficient if \( M \) is large!
Low-pass filtering

- Linear smoothing filters are all low-pass filters.
  - Mean filter (uniform weights)
  - Gauss filter (Gaussian weights)

- Low-pass means low frequencies are not altered, high frequencies are attenuated
High-pass filtering

- The opposite of low-pass filtering: low frequencies are attenuated, high frequencies are not altered.
- The “unsharp mask” filter is a high-pass filter.
- The Laplace filter is a high-pass filter.

![High-pass filter diagram](image-url)
Band-pass filtering

- You can choose any part of the frequency axis to preserve (band-pass filter).
- Or you can attenuate a specific set of frequencies (band-stop filter).
Example: low-pass filtering

input image $f$

Fourier transform $F$
Example: frequency domain filtering

Fourier filter $H$

$G = F \cdot H$

filtered image $g$
Why the ringing?

Spatial domain

Frequency domain

FT
What is the solution?

Spatial domain

Frequency domain

FT
Example: frequency domain filtering

Fourier filter $H$

$G = F \cdot H$

filtered image $g$
Structured noise
Structured noise
Filtering structured noise

Notch filter
Filtering structured noise

Notch filter, Gaussian
Filtering structured noise

Notch filter, Gaussian
Fourier analysis of sampling

\[ F(\omega) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} \, dx \]

- Smooth function
- Band limit (cutoff frequency)
  \[ F(\omega) = 0, \omega > \omega_c \]
Fourier analysis of sampling

- Spatial domain
- Frequency domain

Continuous function

Sampling function

Sampled function
Fourier analysis of interpolation

Spatial domain

Sampled function

Reconstruction function

Continuous function

Frequency domain
Aliasing

spatial domain vs. frequency domain

- Continuous function
- Sampling function
- Sampled function
Aliasing

Sampling distance
Aliasing
Avoid aliasing

Frequency domain

\[ F(\omega) = 0, \quad \omega > \omega_c \]

\[ \omega_s > 2\omega_c \]

Minimum sampling frequency
Nyquist frequency
Example: aliasing
Example: aliasing
Example: aliasing

When we downsample, we only keep this part!
Example: aliasing

The spectrum is replicated, higher frequencies being duplicated as lower frequencies.
Example: Moire
Example: Moire
Summary of today’s lecture

- The Fourier transform
  - decomposes a function (image) into trigonometric basis functions (sines & cosines)
  - is used to analyse frequency components
  - is computed independently for each dimension

- The DFT can be computed efficiently through the FFT algorithm

- Convolution can be studied through the FT
  - and filters can be designed in the Fourier domain
  - $\mathcal{F}\{f \otimes h\} = \mathcal{F}\{f\} \cdot \mathcal{F}\{h\}$

- Aliasing can be understood through the FT
Reading assignment

- The Fourier transform and the DFT
  - Sections 4.2, 4.4, 4.5, 4.6, 4.11.1
- Filtering in the Fourier domain
  - Sections 4.7, 4.8, 4.9, 4.10, 5.4
- Sampling and aliasing
  - Sections 4.3, 4.5.4
- The FFT
  - Section 4.11.3
- Exercises:
  - 4.14, 4.21, 4.22, 4.42, 4.43
  - 4.27, 4.29
  (feel free to solve these in MATLAB)