Signal Processing Background

Biomedical Image Analysis

Prof. Dr. Philippe Cattin

MIAC, University of Basel

Feb 23rd & 29th, 2016
2.3 Properties of the Fourier Transform

- Introduction ......................................................... 23
- Fourier Transform of Even, Odd Functions ............. 24
- Separability .......................................................... 25
- Separability Example ............................................. 26
- Translation ............................................................. 27
- Translation in the Fourier Domain ......................... 28
- Translation in the Fourier Domain Example ............. 29
- Translation in the Fourier Domain Example (2) .... 30
- Translation in the Image Domain .......................... 31
- Translation in the Image Domain Example ............. 32
- Rotation ................................................................. 33
- Rotation Example .................................................... 34
- Distributivity and Scaling ....................................... 35
- Average Value ......................................................... 36
- Laplacian ............................................................... 37
- Convolution and Correlation ................................ 38
- Convolution (1) ......................................................... 39
- Convolution (2) ......................................................... 40
- Convolution with an Impulse Function ................. 41
- Convolution with an Impulse Function (2) .......... 42
- Discrete Convolution .............................................. 43
- Two-Dimensional Continuous Convolution ........... 44
- Two-Dimensional Discretised Convolution ............ 45
- 2D Convolution Example ....................................... 46
- Correlation ............................................................ 47
- Correlation Theorem .............................................. 49
- 2D Correlation Example ......................................... 50

2.4 The Fast Fourier Transform (FFT)

- Computational Complexity .................................... 52
- Computational Complexity (2) ............................... 53
- Derivation of the FFT Algorithm ........................... 54
- The Inverse FFT ....................................................... 55
Abstract

This chapter deals with the signal processing background necessary to understand the underlying mathematics behind many Computer Vision algorithms. In particular the Fourier Transform, the Discrete Fourier Transform, and the Fast Fourier Transform are discussed.

Motivation

Motivation - Fetoscope

The homomorphic filter used for this example uses the Fourier Transform.

Fig 3.1: Homomorphic filter example of a fetoscope image
Motivation Mariner

Fig 3.2: A Fourier Transform based notch filter example

Motivation - Mariner

Fig 3.3: Original Mariner 6 martian image
Fig 3.4: Log Fourier spectra of the image
Fig 3.5: Notch filtered log spectra
Fig 3.6: Notch filtered image
Transformations in the Frequency Domain

Introduction (8)

- A periodic function can be represented by the sum of sines and cosines of different frequencies, multiplied by a different coefficient (Fourier Series).
- Non-periodic functions can also be represented as the integral of sines/cosines multiplied by a weighting function (Fourier Transformation).
Definition of the Fourier Transform

Let $f(x)$ be a continuous function of real variable $x$. The Fourier Transform of $f(x)$, denoted $\mathcal{F}\{f(x)\}$ is defined by the equation

$$\mathcal{F}\{f(x)\} = F(u) = \int_{-\infty}^{\infty} f(x)e^{-j2\pi ux}dx \quad (3.1)$$

where $j = \sqrt{-1}$.

Given $F(u)$, $f(x)$ can be obtained by using the inverse Fourier Transform

$$\mathcal{F}^{-1}\{F(u)\} = f(x) = \int_{-\infty}^{\infty} F(u)e^{j2\pi ux}du \quad (3.2)$$

The Fourier transform pair exists, if $f(x)$ is continuous and integrable and $F(u)$ is integrable, which is almost always satisfied in practice.

In Computer Vision we are mainly concerned with real functions. The Fourier Transform of a real function, however, is generally complex, thus

$$F(u) = \Re(u) + j\Im(u) \quad (3.3)$$

where $\Re(u)$ and $\Im(u)$ are the real and imaginary components of $F(u)$, respectively. Often it is convenient to express it in exponential form

$$F(u) = |F(u)|e^{j\phi(u)} \quad (3.4)$$

where

$$|F(u)| = \sqrt{\Re^2(u) + \Im^2(u)} \quad (3.5)$$

and

$$\phi(u) = \tan^{-1}\frac{\Im(u)}{\Re(u)} \quad (3.6)$$

The magnitude function $|F(u)|$ is called the Fourier Spectrum of $f(x)$ and $\phi(u)$ its Phase Angle.

The square of the spectrum

$$P(u) = |F(u)|^2 = \Re^2(u) + \Im^2(u) \quad (3.7)$$

is commonly referred to as Power Spectrum or Spectral Density.

The variable $u$ appearing in the Fourier Transform is often called the Frequency Variable. The name arises from the exponential term, that can be rewritten using Euler's Formula

$$e^{-j2\pi ux} = \cos(2\pi ux) + j\sin(2\pi ux) \quad (3.8)$$
Fourier Transform

Example

Consider the simple function shown in Fig 3.7. Its Fourier transform is obtained from Eq 3.1 as follows:

\[
F(u) = \int_{-\infty}^{\infty} f(x)e^{-j2\pi ux} dx \\
      = \int_{0}^{X} Ae^{-j2\pi ux} dx \\
      = -\frac{A}{j2\pi u} [e^{-j2\pi ux}]_{0}^{X} \\
      = -\frac{A}{j2\pi u} [e^{-j2\pi ux} - 1] \\
      = \frac{A}{j2\pi u} [e^{j\pi uX} - e^{-j\pi uX}]e^{-j\pi uX} \\
      = \frac{A}{\pi u} \sin(\pi uX)e^{-j\pi uX}
\]

As \( F(u) \) is a complex function, we calculate the Fourier spectrum for visualisation purposes.

\[
|F(u)| = \left| \frac{A}{\pi u} \right| \left| \sin (\pi uX) \right| \left| e^{-j\pi uX} \right| \\
      = AX \left| \frac{\sin (\pi uX)}{\pi uX} \right|
\]

Figure 3.8 shows a plot of \( |F(u)| \).
Extension of the Fourier Transform to 2D Functions

The Fourier Transformation can be easily extended to 2D functions \( f(x, y) \). If the function is continuous and integrable and \( F(u, v) \) is integrable, the Fourier Transform pair exists

\[
\mathcal{F}\{f(x, y)\} = F(u, v) = \int_{x=-\infty}^{\infty} \int_{y=-\infty}^{\infty} f(x, y) e^{-j2\pi(ux+vy)} dx dy \tag{3.11}
\]

and the inverse Fourier Transform

\[
\mathcal{F}^{-1}\{F(u, v)\} = f(x, y) = \int_{u=-\infty}^{\infty} \int_{v=-\infty}^{\infty} F(u, v) e^{j2\pi(ux+vy)} du dv \tag{3.12}
\]

Similar to the 1D case, the Fourier Spectrum, Phase, and Power Spectrum can be defined as follows

\[
|F(u, v)| = \sqrt{\Re^2(u, v) + \Im^2(u, v)} \tag{3.13}
\]

\[
\phi(u, v) = \tan^{-1}\left( \frac{\Im(u, v)}{\Re(u, v)} \right) \tag{3.14}
\]

\[
P(u, v) = |F(u, v)|^2 = \Re^2(u, v) + \Im^2(u, v) \tag{3.15}
\]
As \( F(u, v) \) is a complex function, we calculate the Fourier spectrum for visualisation purposes

\[
|F(u, v)| = AXY \left| \frac{\sin(\pi u X)}{\pi u X} \right| \left| \frac{\sin(\pi v Y)}{\pi v Y} \right| \tag{3.17}
\]

Figure 3.10 shows a plot of \( |F(u)| \).
**Definition of the 1D Discrete Fourier Transform**

In Computer Vision the continuous functions $f(x)$ are generally discretised into a sequence

$$\{f(x_0), f(x_0 + \Delta x), f(x_0 + 2\Delta x), \ldots, f(x_0 + [N - 1]\Delta x)\}$$

by taking $N$ samples $\Delta x$ units apart. The function $f(x)$ can be redefined

$$f(x) = f(x_0 + x\Delta x)$$

where $x$ now assumes the discrete values $0, 1, 2, \ldots, N - 1$.

With this notation, the Discrete Fourier Transform (DFT) can be defined as

$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)e^{-j2\pi ux/N} \quad \forall u = 0, 1, \ldots, N - 1$$

and the Inverse Discrete Fourier Transform

$$f(x) = \sum_{u=0}^{N-1} F(u)e^{j2\pi ux/N} \quad \forall x = 0, 1, \ldots, N - 1$$

The terms $\Delta u$ and $\Delta x$ are related by

$$\Delta u = 1/(N\Delta x).$$
The 2D Discrete Fourier Transform (17)

The definition of the (in Computer Vision more common) 2D Discrete Fourier Transform (DFT) is then given by

\[ F\{f(x,y)\} = F(u,v) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y)e^{-j2\pi(ux/M+vy/N)} \] (3.19)

for \( u = 0, 1, 2, ..., M - 1 \) and \( v = 0, 1, 2, ..., N - 1 \), and the Inverse DFT

\[ F^{-1}\{F(u,v)\} = f(x,y) = \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v)e^{j2\pi(ux/M+vy/N)} \] (3.20)

for \( x = 0, 1, 2, ..., M - 1 \) and \( y = 0, 1, 2, ..., N - 1 \).

Sampling of the continuous function \( f(x,y) \) is in a 2D grid of width \( \Delta x \) and height \( \Delta y \) in the \( x, y \) axis, respectively.

The 2D Discrete Fourier Transform (2)

As in the 1D case, the discrete function \( f(x,y) \) represents samples of the function

\[ f(x_0 + x\Delta x, y_0 + y\Delta y) \] (3.21)

for \( x = 0, 1, 2, ..., M - 1 \) and \( y = 0, 1, 2, ..., N - 1 \).

The sampling increments in the spatial and frequency domain are related by

\[ \Delta u = \frac{1}{M\Delta x} \] (3.22)

and

\[ \Delta v = \frac{1}{N\Delta y} \] (3.23)
Remark 1: The Scaling Terms

Because \( f(x, y) \) and \( F(u, v) \) are a Fourier Transform pair the multiplicative scaling terms can be chosen arbitrary. As images are often digitised in square arrays, thus \( M = N \), the following scaling is often chosen

\[
f(x, y) \quad \rightarrow \quad F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi(ux + vy)/N} \quad (3.24)
\]

and

\[
F(u, v) \quad \rightarrow \quad f(x, y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(ux + vy)/N} \quad (3.25)
\]

Beware, that the scaling term in MATLAB is with the inverse rather than the transform!

\[
F(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} \ldots, \quad f(x, y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} \ldots
\]

Remark 2: Existence of the DFT

Claim

In contrast to the continuous case, existence of the discrete Fourier Transform is of no concern, because both \( F(u) \) and \( F(u, v) \) always exist.

Proof

\[
F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x) e^{-j2\pi ux/N} \quad \text{and} \quad f(x) = \sum_{u=0}^{N-1} F(u) e^{j2\pi ux/N}
\]

\[
F(u) = \frac{1}{N} \sum_{x=0}^{N-1} \left[ \sum_{r=0}^{N-1} F(r) e^{j2\pi rx/N} \right] e^{-j2\pi ux/N}
\]

\[
= \frac{1}{N} \sum_{r=0}^{N-1} F(r) \left[ \sum_{x=0}^{N-1} e^{j2\pi rx/N} e^{-j2\pi ux/N} \right]
\]

The identity follows from the orthogonality condition

\[
\sum_{x=0}^{N-1} e^{j2\pi rx/N} e^{-j2\pi ux/N} = \begin{cases} N & \text{if } r = u \\ 0 & \text{otherwise} \end{cases}
\]
The dynamic range of Fourier spectra usually is much higher than the typical display device can reliably reproduce. The consequence is that only the brightest parts are shown, see Fig 3.13(b). A useful technique that compensates for this difficulty is of displaying the function

\[ D(u, v) = c \cdot \log[1 + |F(u, v)|] \]  

(3.26)

see Fig 3.13(c).
Properties of the Fourier Transform

Fourier Transform of Even, Odd Functions

The fact that the Fourier Transform of a real-valued image yields a complex output might give the impression that information has somehow doubled. **This is of course not the case.** In fact, for real input (such as images) a number of important properties hold for the Fourier Transform:

<table>
<thead>
<tr>
<th>Spatial Domain</th>
<th>Frequency Domain</th>
<th>Hint:</th>
</tr>
</thead>
<tbody>
<tr>
<td>real</td>
<td>real part, even, imaginary part odd</td>
<td>• A function is even if it holds for all real $x$: $f(-x) = f(x)$ thus symmetric to the y-axis.</td>
</tr>
<tr>
<td>real, even</td>
<td>real, even</td>
<td>• A real-valued function is odd if for all real $x$ it holds: $f(-x) = -f(x)$ thus symmetric to the origin.</td>
</tr>
<tr>
<td>real, odd</td>
<td>imaginary, odd</td>
<td></td>
</tr>
</tbody>
</table>

As can be seen in the above table, the relationships between Fourier coefficients are such that the total number of independent variables remains the same.

Separability

The 2D discrete Fourier Transform pair, Eq 3.19 & 3.20, can be expressed in its separable form

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi uv/N}$$ (3.27)

and the Inverse Transformation respectively

$$f(x, y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi uv/N}$$ (3.28)

Advantage:

• The Transformations $F(u, v)$ and $f(x, y)$ can be obtained in two successive applications of 1D Fourier transforms → computationally very efficient.

This becomes evident, when we rewrite the separable discrete Fourier Transform in Eq 3.27 in the form

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} F(x, v)e^{-j2\pi ux/N}$$ (3.29)

where
we get two 1D Fourier Transforms. The same principle applies for the Inverse Fourier Transform. The following figure illustrates this process:

![Figure 3.14: Principle of separability](image)

\[
F(x, v) = N \left[ \frac{1}{N} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi vy/N} \right]
\]

(3.30)

**Separability Example**

MATLAB example using the 2D FFT function

\[
\text{fft2(magic(3))}
\]

ans =

\[
\begin{array}{cccc}
45.0000 & 0 \\
0 & 13.5000 + 7.7942i & 0.0000 - 5.1962i \\
0 - 0.0000i & 0.0000 + 5.1962i & 13.5000 - 7.7942i \\
\end{array}
\]

MATLAB example using two 1D FFT function calls

\[
\text{fft(fft(magic(3)).').}'
\]

ans =

\[
\begin{array}{cccc}
45.0000 & 0 \\
0 & 13.5000 + 7.7942i & 0.0000 - 5.1962i \\
0 - 0.0000i & 0.0000 + 5.1962i & 13.5000 - 7.7942i \\
\end{array}
\]
Translation \( (27) \)

We have to differentiate between two cases

1. Translations in the Fourier (Frequency) Domain
2. Translations in the Image Domain

Let’s assume we know

\[ f(x, y) \iff F(u, v) \quad (3.31) \]

we want to know how a translation in the Fourier Domain by \((u_0, v_0)\) can be expressed in \(f(x, y)\) · · ·

\[ F(u - u_0, v - v_0) \iff f(x, y) \quad (3.32) \]

and similarly for translations in the Image Domain

\[ f(x - x_0, y - y_0) \iff F(u, v) \quad (3.33) \]

Translation in the Fourier \( (28) \) Domain

A Translation of \((u_0, v_0)\) in the Fourier Domain results in

\[ F(u - u_0, v - v_0) \iff f(x, y) e^{j2\pi(u_0x + v_0y)/N} (3.34) \]

A multiplication of \(f(x, y)\) with the exponential term and taking the transform of the product results in a shift of the origin of the frequency plane to the point \((u_0, v_0)\).

The special case were \(u_0 = v_0 = N/2\) is often used and yields

\[ e^{j2\pi(u_0x + v_0y)/N} = e^{j\pi(x+y)} (3.35) \]

and

\[ f(x, y)(-1)^{x+y} \iff F(u - N/2, v - N/2). (3.36) \]

Thus the origin of the Fourier Transform of \(f(x, y)\) can be moved to the centre of its corresponding \(N \times N\) frequency square by multiplying \(f(x, y)\) by the term \((-1)^{x+y} \).

For the 1D case this shift reduces to the term \((-1)^{x}\).
Translation in the Fourier (29) Domain Example

The discrete Fourier Transform is generally calculated using the Fast Fourier Transform. The FFT implementations, however, generally yield the frequency domain unsorted as can be seen in Fig 3.15(b) on the right.

Shifting the frequency domain by $N/2$ would yield the correct sorting order as depicted in Fig 3.15(c). This can be achieved in the spatial domain by multiplying $f(x)$ by

$$f'(x) = f(x) \cdot (-1)^x \text{ (3.37)}$$

followed by the Fourier transform $\mathcal{F}\{f'(x)\}$

![Graph](image1)

**MATLAB Code**

```matlab
img=zeros(256,256);
img(128-32:128+32,128-16:128+16)=1;
IMG=fft2(img);
figure; imshow(log10(1+abs(IMG)));
```

![Graph](image2)

**Log spectrum with the lowest frequencies at the edges**

**MATLAB Code**

```matlab
img=zeros(256,256);
img(128-32:128+32,128-16:128+16)=1;
[x,y]=meshgrid(1:size(img,1),1:size(img,2));
img1xy=img.*(-1).^(x+y);
IMG=fft2(img1xy);
figure; imshow(log10(1+abs(IMG)));
```

![Graph](image3)

**Log spectrum with the lowest frequencies in the centre**

**Translation in the Fourier (30) Domain Example (2)**

![Graph](image4)
Translation in the Image Domain (31)

A Translation of \((x_0, y_0)\) in the Image Domain results in

\[
f(x - x_0, y - y_0) \iff F(u, v)e^{-j2\pi(ux_0 + vy_0)/N} \tag{3.38}
\]

Multiplying \(F(u, v)\) with the exponential term

\[
e^{-j2\pi(ux_0 + vy_0)/N} \tag{3.39}
\]

and taking the inverse Fourier Transform moves the origin of the Image to \((x_0, y_0)\)

Note, that a shift in \(f(x, y)\) does not affect the magnitude of the Fourier Transform (but only its phase), as

\[
|F(u, v)e^{-j2\pi(ux_0 + vy_0)/N}| = |F(u, v)| \tag{3.40}
\]

This is important to know, as the magnitude is often taken to visualise the Fourier Transform.

Translation in the Image Domain Example (32)

Original image

log Fourier spectrum

Phaseangle

Shifted original image

Shifted log Fourier spectrum

Shifted phaseangle
**Rotation**

To investigate the influence of rotation we introduce polar coordinates

\[ x = r \cos \theta \quad y = r \sin \theta \quad u = \omega \cos \phi \quad v = \omega \sin \phi \] \hspace{1cm} (3.41)

then \( f(x, y) \) and \( F(u, v) \) become \( f(r, \theta) \) and \( F(\omega, \phi) \) respectively.

The direct substitution in the discrete Fourier Transform pair yields

\[ f(r, \theta + \theta_0) \leftrightarrow F(\omega, \phi + \theta_0). \] \hspace{1cm} (3.42)

A rotation by \( \theta \) in the Image Domain rotates the Fourier Domain by the same angle and vice versa.
**Distributivity and Scaling (35)**

Form the definition of the discrete Fourier Transform pair follows that

\[
\mathcal{F}\{f_1(x,y) + f_2(x,y)\} = \mathcal{F}\{f_1(x,y)\} + \mathcal{F}\{f_2(x,y)\} (3.43)
\]

However, in general

\[
\mathcal{F}\{f_1(x,y) \cdot f_2(x,y)\} \neq \mathcal{F}\{f_1(x,y)\} \cdot \mathcal{F}\{f_2(x,y)\} (3.44)
\]

In other words, the Fourier Transform and its inverse is distributive over addition but not over multiplication.

For two scalars \(a\) and \(b\)

\[
a f(x,y) \Leftrightarrow a F(u,v) (3.45)
\]

and

\[
f(ax, by) \Leftrightarrow \frac{1}{|ab|} F(u/a, v/b) (3.46)
\]

**Average Value (36)**

The average value of a discrete 2D function can be defined as

\[
\tilde{f}(x,y) = \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) (3.47)
\]

Substituting \(u = v = 0\) in Eq 3.19 and assuming \(M = N\) yields

\[
F(0,0) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M+yv/N)} |_{u=v=0, M=N}
\]

\[
= \frac{1}{N^2} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y)
\]

Therefore the average value \(\tilde{f}(x,y)\) is related to the Fourier transform at the frequency 0 thus \(F(0,0)\).
Laplacian \hspace{1cm} (37)

The Laplacian of a 2-dimensional function \( f(x, y) \) is defined as

\[
\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} (3.49)
\]

With the definition of the Fourier Transform we get

\[
\mathcal{F}\{\nabla^2 f(x, y)\} \leftrightarrow -(2\pi)^2 (u^2 + v^2) F(u, v) (3.50)
\]

The Laplacian operator is useful for outlining edges as will be shown in later sections of this lecture.

Convolution and Correlation \hspace{1cm} (38)

In the next few slides we will introduce two Fourier Transform relationships that connect the spatial and the frequency domain, namely

- Convolution
- Correlation

Convolution and correlation are of fundamental importance in many image processing techniques.
**Convolution (1)**

The convolution of two 1-dimensional functions $f(x)$ and $g(x)$ is generally denoted by $f(x) * g(x)$ and defined by the integral

$$f(x) * g(x) = \int_{-\infty}^{\infty} f(\alpha)g(x - \alpha)\,d\alpha \quad (3.51)$$

where $\alpha$ is a dummy variable.

The importance of convolution in the frequency domain analysis lies in the fact that $f(x) * g(x)$ and $F(u)G(u)$ constitute a Fourier Transformation pair thus

$$f(x) * g(x) \iff F(u)G(u) \quad (3.52)$$

a convolution in the Image domain results in a multiplication in the Frequency domain, and vice versa

$$f(x)g(x) \iff F(u) * G(u) \quad (3.53)$$
Convolultion with an Impulse Function

The special case of convoluting a function $f(x)$ with an Impulse Function $\delta(x - x_0)$ is of particular interest as will be shown later.

Definition:

The Impulse Function (Dirac delta function) is often referred to as the unit impulse function introduced by the physicist Paul Dirac [http://en.wikipedia.org/wiki/Paul_Dirac]. The function $\delta(x - x_0)$ may be viewed as having an area of unity in an infinitesimal small neighbourhood about $x_0$ and zero elsewhere; that is,

Sifting Theorem

$$\int_{-\infty}^{\infty} \delta(x - x_0) \, dx = 1 \quad \text{and thus} \quad \int_{-\infty}^{\infty} f(x) \delta(x - x_0) \, dx = f(x_0)$$

It is common practice to graphically represent the Dirac impulses as arrows at $x_0$ with a height equal to the impulse strength (area).
### Discrete Convolution \((43)\)

Suppose that, instead of being continuous, \(f(x), g(x)\) are discretised into sampled arrays of size \(A\) and \(B\)

\[
f(x) : \{f(0), f(1), f(2), ..., f(A - 1)\} \quad (3.54)
\]

\[
g(x) : \{g(0), g(1), g(2), ..., g(B - 1)\} \quad (3.55)
\]

With \(M \geq A + B - 1\) the discrete convolution can be defined as

\[
f(x) \ast g(x) = \frac{1}{M} \sum_{m=0}^{M-1} f(m)g(x - m) \quad (3.56)
\]

Because \(M\) is bigger than \(A\) and \(B\) they must be padded with zeros, so that both are of length \(M\).

### Two-Dimensional Continuous Convolution \((44)\)

The 2D Convolution is analogous to the 1D, thus

\[
f(x, y) \ast g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\alpha, \beta)g(x - \alpha, y - \beta)\,d\alpha\,d\beta \quad (3.57)
\]

The Convolution Theorem in two dimensions can then be expressed as

\[
f(x, y) \ast g(x, y) \Leftrightarrow F(u, v)G(u, v) \quad (3.58)
\]

and

\[
f(x, y)g(x, y) \Leftrightarrow F(u, v) \ast G(u, v) \quad (3.59)
\]
Two-Dimensional Discretised Convolution

The discretised 2D Convolution is defined by

$$f(x, y) * g(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n)g(x-m, y-n) \quad (3.60)$$

where $A \times B$ and $C \times D$ are the discretised arrays of $f(x, y)$ and $g(x, y)$, respectively.

Wraparound error in the individual convolutions is avoided by choosing

$$M \geq A + C - 1 \quad (3.61)$$

and

$$N \geq B + D - 1 \quad (3.62)$$

Calculating the discrete convolution in the frequency domain is often more efficient than directly using the equation above.

MATLAB snippet

```matlab
IMG=fft2(img);
KERN=fft2(kern);
F=IMG.*KERN;
f=ifft2(F);
```

Fig 3.17: 2D Convolution example (a) Original image of size $A \times B$, (b) filter kern of size $C \times D$, (c) padded original of size $(A + C - 1) \times (B + D - 1)$, (d) padded filter kern of size $(A + C - 1) \times (B + D - 1)$, (e) convolution result of size $(A + C - 1) \times (B + D - 1)$
Correlation

The correlation of two continuous functions \( f(\cdot) \) and \( g(\cdot) \), denoted by \( f(\cdot) \circ g(\cdot) \), is defined by the relation

1D: \[ f(x) \circ g(x) = \int_{-\infty}^{\infty} f^*(\alpha)g(x + \alpha)d\alpha \]

2D: \[ f(x, y) \circ g(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f^*(\alpha, \beta)g(x + \alpha, y + \beta)d\alpha d\beta \]

where \( \ast \) is the complex conjugate.

The discrete equivalent of the correlation is defined as

1D: \[ f(x) \circ g(x) = \frac{1}{M} \sum_{m=n}^{M-1} f^*(m)g(x + m) \]

2D: \[ f(x, y) \circ g(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n)g(x + m, y + n) \]

Correlation Theorem

For both the continuous and discrete cases, the following correlation theorem holds

\[ \quad \]

\[ f(x) \circ g(x) \iff F^*(u)G(u) \quad (3.63) \]

\[ f(x, y) \circ g(x, y) \iff F^*(u, v)G(u, v) \]

and

\[ f^*(x)g(x) \iff F(u) \circ G(u) \quad (3.64) \]

\[ f^*(x, y)g(x, y) \iff F(u, v) \circ G(u, v) \]
Correlation Example  (49)

The principal application for correlation in image processing are

• template matching

However, one has to take into account, that correlation is

• sensitive to lightning changes
• linear bias
The Fast Fourier Transform (FFT)

Computational Complexity (52)

The number of complex multiplications and additions required to implement the Discrete Fourier Transform

\[
F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)e^{-j2\pi ux/N}
\]

is proportional to \( O(N^2) \),

- as for each of the \( N \) values of \( u \) the expansion of \( \sum \) requires \( N \) complex multiplications of \( f(x) \) by \( e^{-j2\pi ux/N} \)
- as the terms \( e^{-j2\pi ux/N} \) can be precalculated and tabulated they are not counted in the complexity analysis
Computational Complexity (53) (2)

Proper decomposition can reduce the number of multiplications and addition proportional to \( O(N \log_2 N) \). This decomposition is called the fast Fourier Transform (FFT) algorithm.

<table>
<thead>
<tr>
<th>( N )</th>
<th>( N^2 )</th>
<th>( N \log_2 N )</th>
<th>( N/\log_2 N )</th>
<th>Example:</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>1'024</td>
<td>160</td>
<td>6.40</td>
<td>Let’s assume that an FFT of size 8'192 takes on one particular machine 1s.</td>
</tr>
<tr>
<td>64</td>
<td>4'096</td>
<td>384</td>
<td>10.67</td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>16'384</td>
<td>896</td>
<td>18.29</td>
<td></td>
</tr>
<tr>
<td>256</td>
<td>65'536</td>
<td>2'048</td>
<td>32.00</td>
<td></td>
</tr>
<tr>
<td>512</td>
<td>262'144</td>
<td>4'608</td>
<td>56.89</td>
<td></td>
</tr>
<tr>
<td>1024</td>
<td>1'048'576</td>
<td>10'240</td>
<td>102.40</td>
<td></td>
</tr>
<tr>
<td>2048</td>
<td>4'194'304</td>
<td>22'528</td>
<td>186.18</td>
<td></td>
</tr>
<tr>
<td>4096</td>
<td>16'777'216</td>
<td>49'152</td>
<td>341.33</td>
<td></td>
</tr>
<tr>
<td>8192</td>
<td>67'108'864</td>
<td>106'496</td>
<td>630.15</td>
<td></td>
</tr>
</tbody>
</table>

Derivation of the FFT (54) Algorithm

The FFT algorithm developed in the next few slides is based on the successive doubling method. We start with the general form of the DFT

\[
F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)e^{-j2\pi ux/N} \quad \forall u = 0, 1, .., N - 1 \tag{3.66}
\]

and rewrite it in the form

\[
F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)W_N^{ux} \tag{3.67}
\]

\[
W_N = e^{-j2\pi/N} \tag{3.68}
\]

and \( N \) is assumed to be of the form \( N = 2^n \) where \( n \) is a positive integer.

The requirement that \( N \) must be a power of 2 does not limit generality of the algorithm, as one can always achieve this requirement by zero-padding the data to the next power of 2.

As \( N \) is a power of 2 we can express it as

\[
N = 2M \tag{3.69}
\]

where \( M \) is also a positive integer. Substitution into Eq. 3.67 yields

\[
F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)W_M^{ux} \tag{3.70}
\]
From Eq. 3.68 we know that \( W_{2M}^{2ux} = W_M^{ux} \), so the previous equation can be expressed as

\[
F(u) = \frac{1}{2M} \sum_{x=0}^{2M-1} f(x)W_{2M}^{2ux} = \frac{1}{2} \left[ \frac{1}{M} \sum_{x=0}^{M-1} f(2x)W_M^{ux} + \frac{1}{M} \sum_{x=0}^{M-1} f(2x+1)W_M^{ux}(2u+1) \right]
\]  

Defining

\[
F_{even}(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(2x)W_M^{ux} \quad \forall \ u = 0, 1, 2, ..., M-1 (3.72)
\]

and

\[
F_{odd}(u) = \frac{1}{M} \sum_{x=0}^{M-1} f(2x+1)W_M^{ux} \quad \forall \ u = 0, 1, 2, ..., M-1 (3.73)
\]

Eq. 3.71 reduces to

\[
F(u) = \frac{1}{2} [F_{even}(u) + F_{odd}(u)W_{2M}^{u}] (3.74)
\]

Also, since \( W_M^{u+M} = W_M^u \) and \( W_{2M}^{u+M} = -W_{2M}^u \) the above equations get

\[
F(u + M) = \frac{1}{2} |F_{even}(u) - F_{odd}(u)W_{2M}^u| (3.75)
\]
The Inverse FFT

(55)

On the previous slide we developed a fast implementation for the Fourier Transform, but what about the Inverse Fourier Transform?

The reason is that any method implementing the forward transform can also be used to compute the inverse. To show this let us repeat the equations for the DFT and inverse DFT

\[ F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)e^{-j2\pi ux/N} \]  
\[ (3.76) \]
\[ f(x) = \sum_{u=0}^{N-1} F(u)e^{j2\pi ux/N} \]  
\[ (3.77) \]

Taking the complex conjugate of Eq. 3.77 and dividing both sides by \(N\) yields

\[ \frac{1}{N}f^*(x) = \frac{1}{N} \sum_{u=0}^{N-1} F^*(u)e^{-j2\pi ux/N} \]  
\[ (3.78) \]

Comparing the result with Eq. 3.76 shows that the right hand side of Eq. 3.78 is in the form of the forward Fourier Transform. Thus inputting \(F^*(u)\) into an algorithm to compute the forward transform gives \(f^*(x)/N\) that can be easily converted to \(f(x)\).