TA session 3 - 8.11.2017

Fourier Transform

Jean Baptiste Joseph Fourier
(1768 – 1830)

Fourier Transform

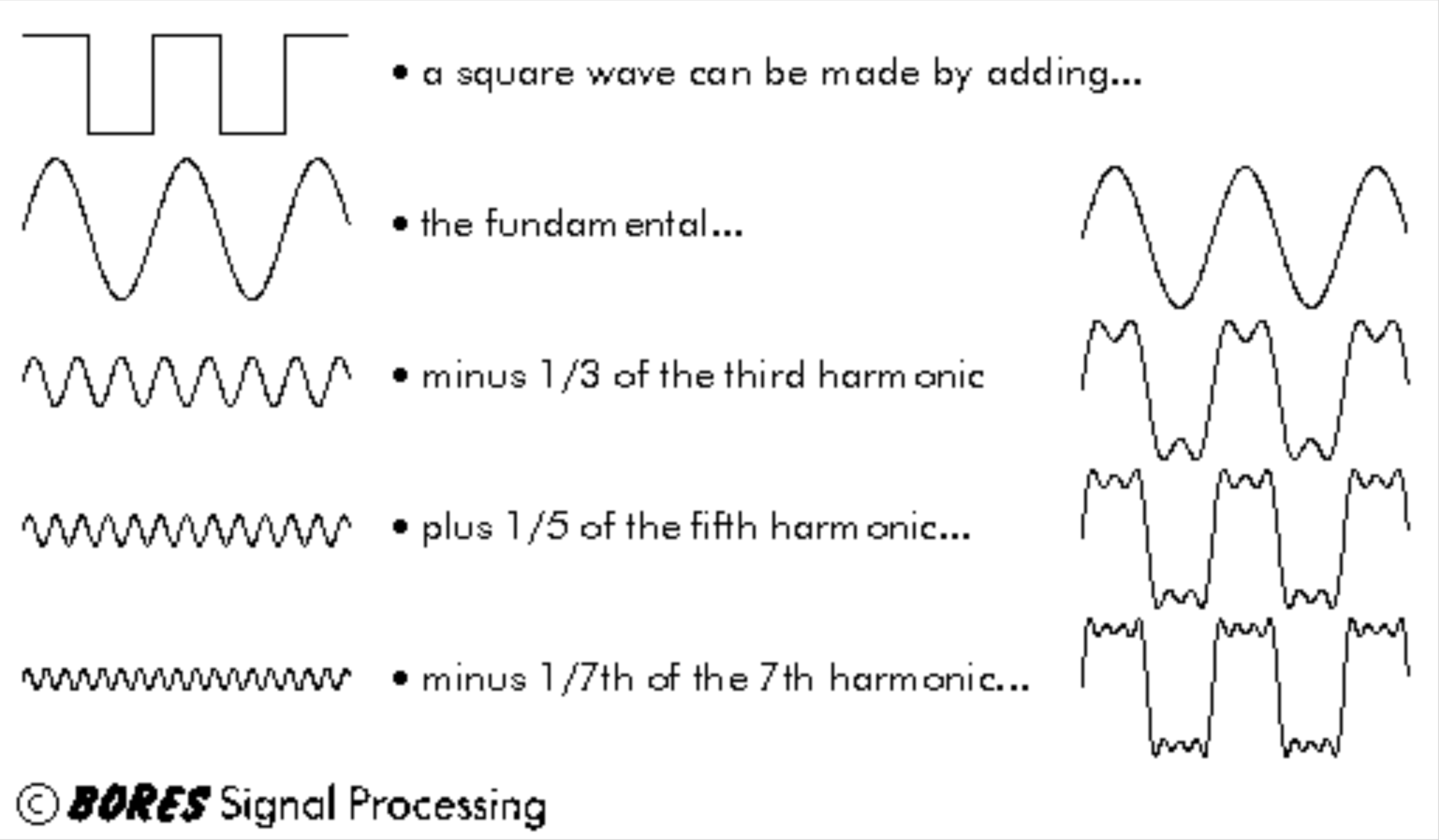
- Any function $f(x)$ can be decomposed into a set of *sin* and *cos* functions of different frequencies,

$$f(x) = \frac{1}{N} \sum_{u=0}^{N-1} F(u) e^{\frac{2\pi i u x}{N}}$$

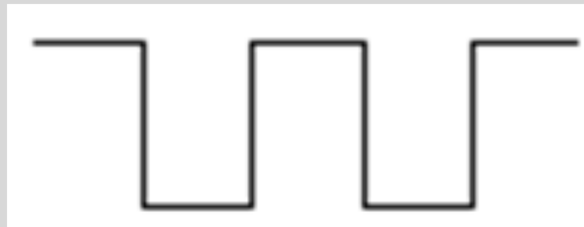
- *f* can be reconstructed from *F* without any loss of data!

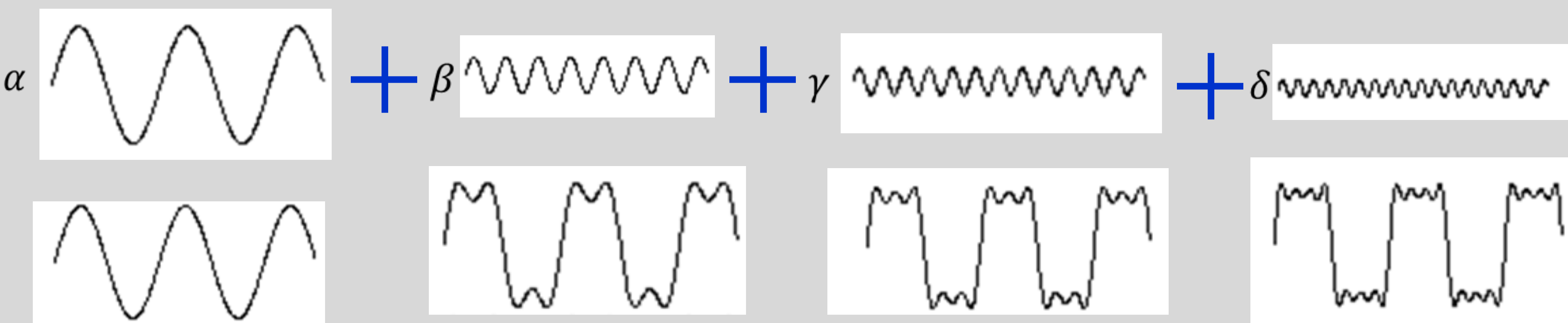
Euler's Formula: $e^{i\theta} = \cos(\theta) + i \cdot \sin(\theta)$

Example


• a square wave can be made by adding...
• the fundamental...
• minus 1/3 of the third harmonic
• plus 1/5 of the fifth harmonic...
• minus 1/7th of the 7th harmonic...
© BORES Signal Processing

Example





Why is that important?

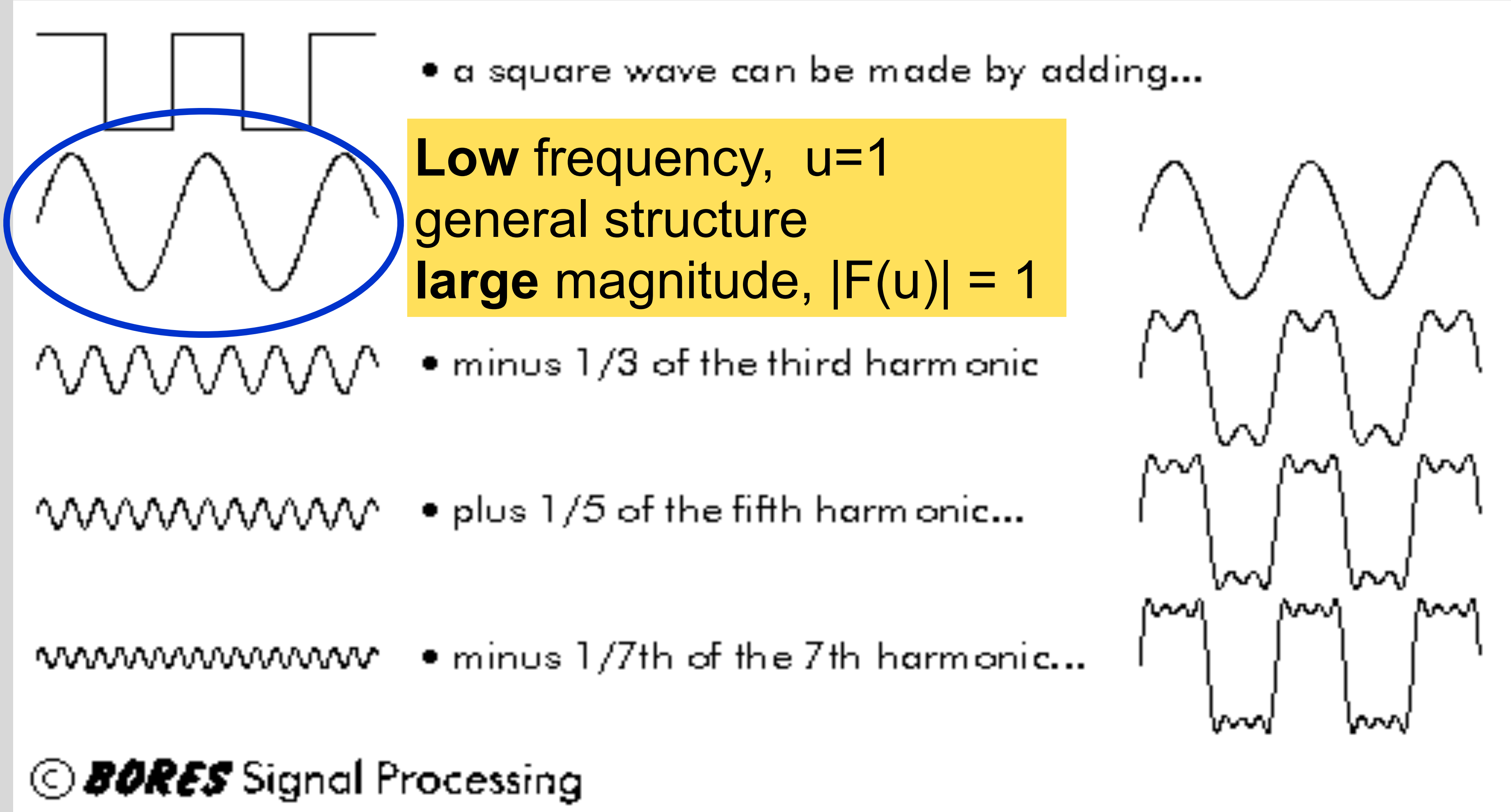
- Decomposition into different resolutions
  - **Low frequencies**: rough general structure
  - **High frequencies**: fine detail

Why is that important?

- Decomposition into different resolutions
  - **Low frequencies**: rough general structure
  - **High frequencies**: fine detail

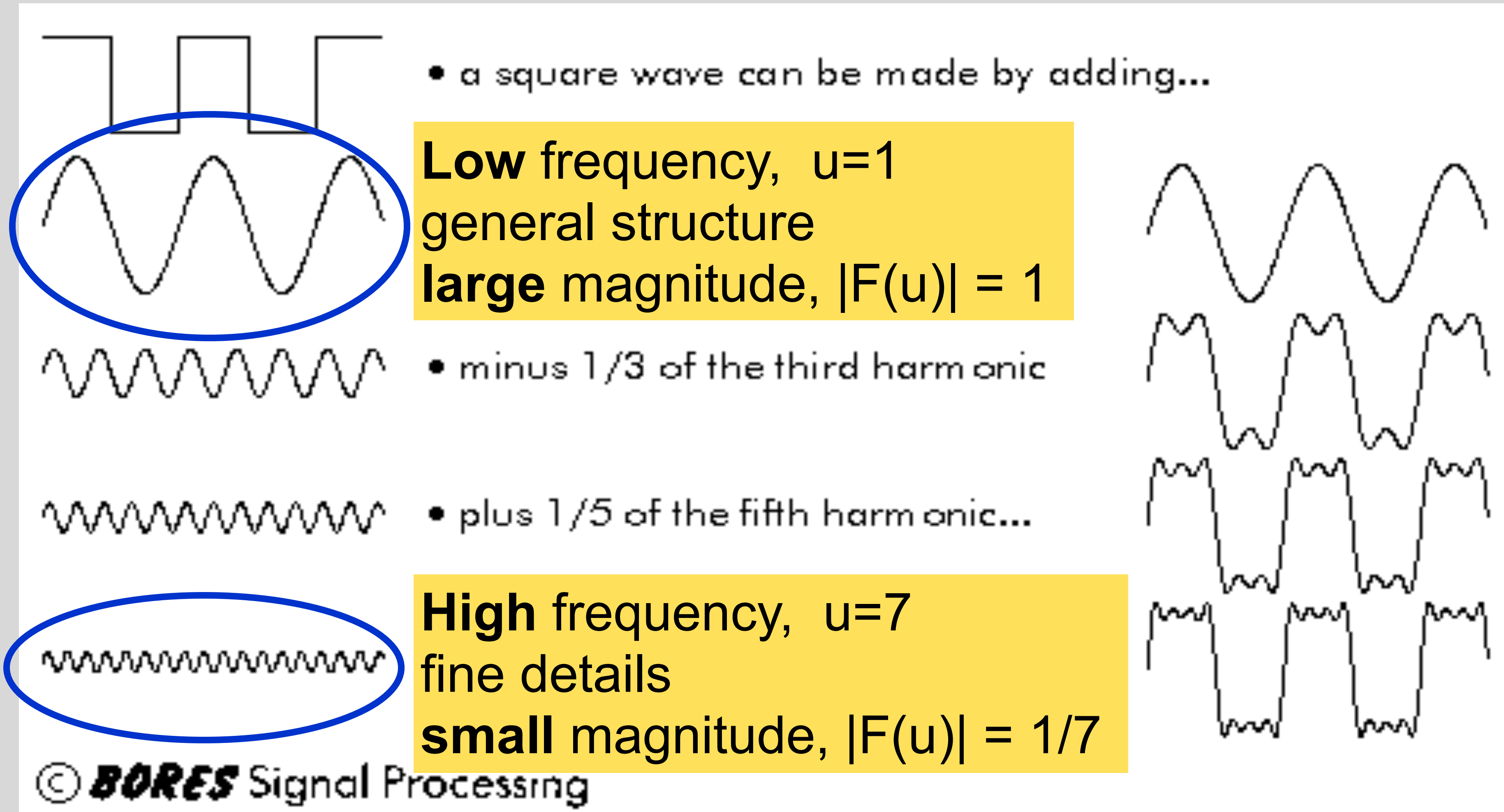
- Very useful for image understanding and processing
  - Filtering, denoising, compression…

Example



• a square wave can be made by adding...
Low frequency, u=1
general structure
large magnitude, |F(u)| = 1
• minus 1/3 of the third harmonic
• plus 1/5 of the fifth harmonic...
• minus 1/7th of the 7th harmonic...
© BORES Signal Processing

Example


• a square wave can be made by adding...
Low frequency, u=1
general structure
large magnitude, |F(u)| = 1
• minus 1/3 of the third harmonic
• plus 1/5 of the fifth harmonic...
High frequency, u=7
fine details
small magnitude, |F(u)| = 1/7
© BORES Signal Processing

Discrete Fourier Transform (DFT)

- Discrete Fourier Transform:

$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)\, e^{\frac{-2\pi i u x}{N}}$$

- Inverse DFT:

$$f(x) = \sum_{u=0}^{N-1} F(u)\, e^{\frac{2\pi i u x}{N}}$$

- Note: f is treated as a **cyclic** function!

Discrete Fourier Transform (DFT)

- Image processing is performed over a discrete world.

- DFT is a **basis transform**:

$$(f(0), f(1), f(2), \ldots, f(N-1))$$

Spatial domain

(Standard basis)

- The frequency domain has an important visual interpretation

Discrete Fourier Transform (DFT)

- Image processing is performed over a discrete world.

- DFT is a **basis transform**:

$$(f(0), f(1), f(2), \ldots, f(N-1))$$

$$f(0) \bullet (1,0,0,\ldots,0) +$$

$$f(1) \bullet (0,1,0,\ldots,0) +$$

$$f(N-1) \bullet (0,0,0,\ldots,1)$$

- The frequency domain has an important visual interpretation

Discrete Fourier Transform (DFT)

- Image processing is performed over a discrete world.

- DFT is a **basis transform**:

$$(f(0), f(1), f(2),\dots, f(N-1)) \xrightarrow[Fourier]{} (F(0), F(1), F(2),\dots, f(N-1))$$

Spatial domain
(Standard basis)
→
Frequency domain
(Fourier basis)

- The frequency domain has an important visual interpretation

Discrete Fourier Transform (DFT)

- Image processing is performed over a discrete world.

- DFT is a **basis transform**:

$$(f(0), f(1), f(2), \ldots, f(N-1)) \xrightarrow[Fourier]{} (F(-\frac{N}{2}), \ldots, F(0), \ldots F(\frac{N}{2}-1))$$

Spatial domain (Standard basis) → Frequency domain (Fourier basis)

- The frequency domain has an important visual interpretation

The DFT Matrix

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\, e^{\frac{-2\pi i u x}{N}} \quad \Leftrightarrow \quad \vec{F} = M_{N\times N}\vec{f}$$

$$\frac{1}{N}\begin{pmatrix} \omega^0 & \omega^0 & \omega^0 & & \omega^0 \\ \omega^0 & \omega^1 & \omega^2 & \cdots & \omega^{N-1} \\ \omega^0 & \omega^2 & \omega^4 & & \omega^{2(N-1)} \\ & \vdots & & & \vdots \\ \omega^0 & \omega^{N-1} & \omega^{2(N-1)} & \cdots & \omega^{(N-1)^2} \end{pmatrix} \begin{pmatrix} f(0) \\ f(1) \\ f(2) \\ \vdots \\ f(N-1) \end{pmatrix} = \begin{pmatrix} F(0) \\ F(1) \\ F(2) \\ \vdots \\ F(N-1) \end{pmatrix}$$

Where $\omega = e^{-\frac{2\pi i}{N}}$

Fourier Spectrum

- Fourier coefficient: $F(u) = R(u) + i \cdot I(u)$
- Power (amplitude): $|F(u)| = \sqrt{R^2(u) + I^2(u)}$
- Phase: $\theta(u) = \tan^{-1}\left(\frac{I(u)}{R(u)}\right)$
- Polar decomposition: $F(u) = |F(u)| \cdot e^{i\theta(u)}$

FT Properties

1. Linearity:

$$\Phi\big(f(x)+g(x)\big)=\Phi\big(f(x)\big)+\Phi\big(g(x)\big)$$
$$\Phi\big(a\cdot f(x)\big)=a\cdot\Phi\big(f(x)\big)$$

2. Periodicity:

$$\forall k\in Z \qquad F(u)=F(u+kN)$$

FT Properties

3. Symmetry*:

$$F(-u) = F^*(u)$$

$$|F(u)| = |F(-u)|$$

4. Scaling: if

$$f(x) \xrightarrow[\text{Fourier}]{} F(u)$$

then

$$f(ax) \xrightarrow[\text{Fourier}]{} \frac{1}{|a|} \cdot F\left(\frac{u}{a}\right)$$

F(0)?

$$F(u) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)\, e^{\frac{-2\pi i u x}{N}}$$

$$F(0) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)\, e^{\frac{-2\pi i 0 x}{N}}$$

$$F(0) = \frac{1}{N} \sum_{x=0}^{N-1} f(x)$$ $\approx$ **Signal average**

Discrete Fourier Transform (2D)

- 2D Fourier Transform:

$$F(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x,y) e^{\frac{-2\pi i(ux+vy)}{N}}$$

Discrete Fourier Transform (2D)

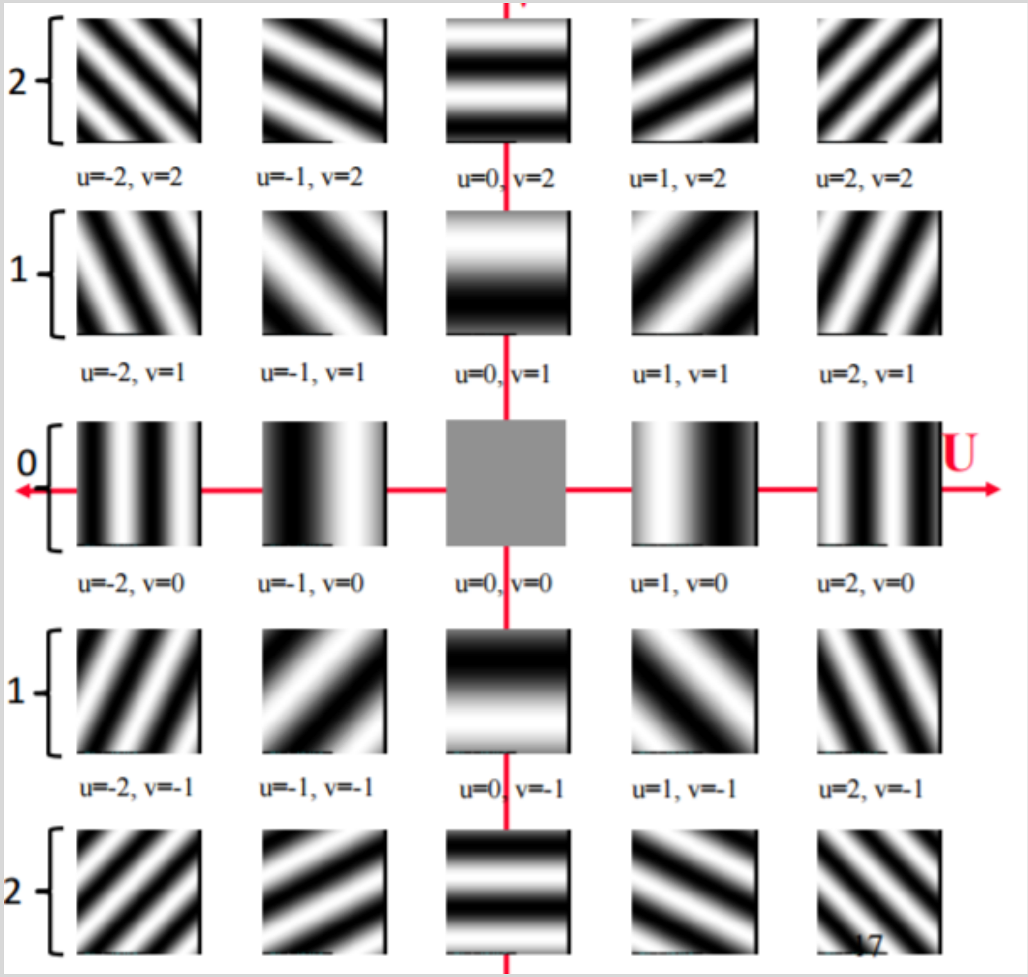
- 2D Fourier Transform:

$$F(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)e^{\frac{-2\pi i(ux+vy)}{N}}$$

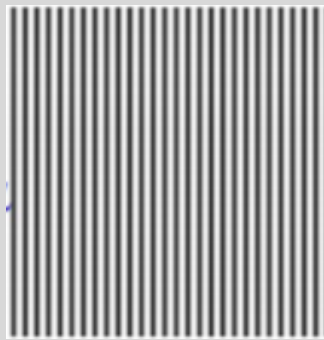
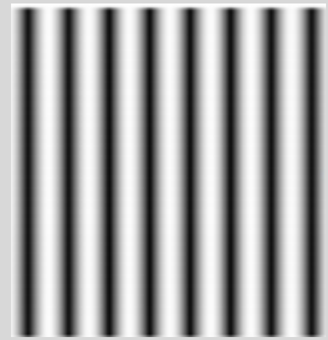
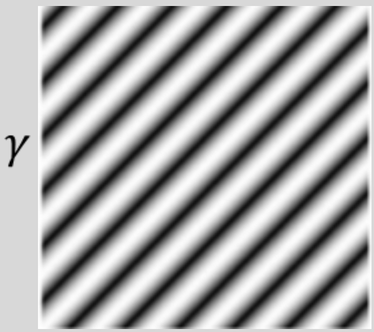
- 2D Inverse Fourier Transform:

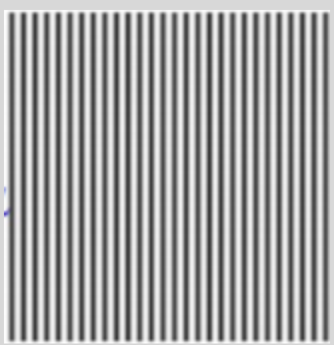
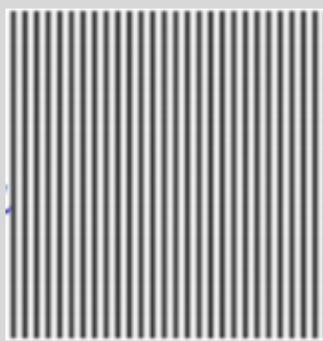
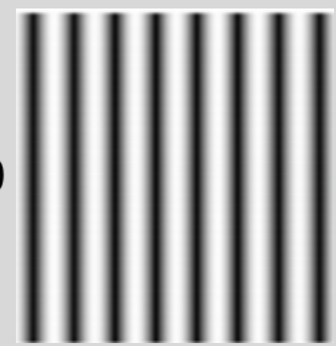
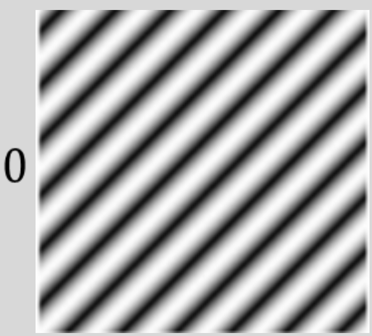
$$f(x,y) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} F(u,v)e^{\frac{2\pi i(ux+vy)}{N}}$$

Discrete Fourier Transform (2D)



Discrete Fourier Transform (2D)

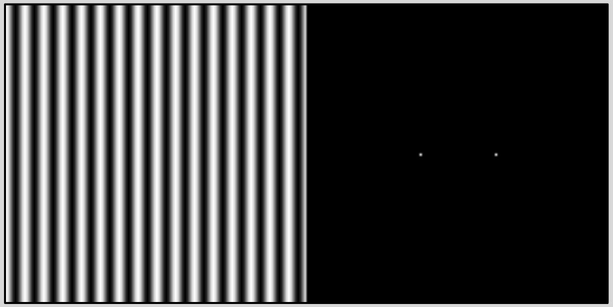
 $=$ α  $+$ β  $+$ γ  $+$

 $=$ 1  $+$ 0  $+$ 0  $+$

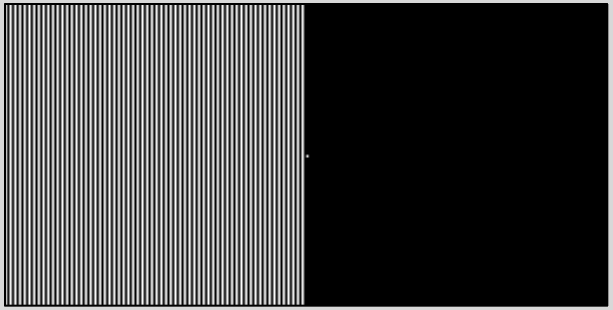
2D DFT – Simple Examples

Low frequency

Medium frequency

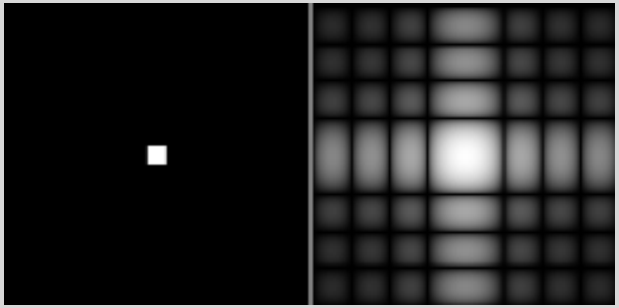


High frequency

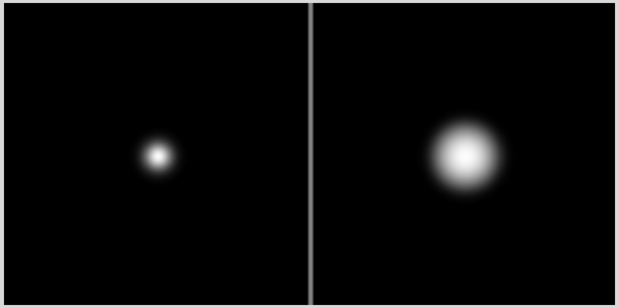


2D DFT – Simple Examples

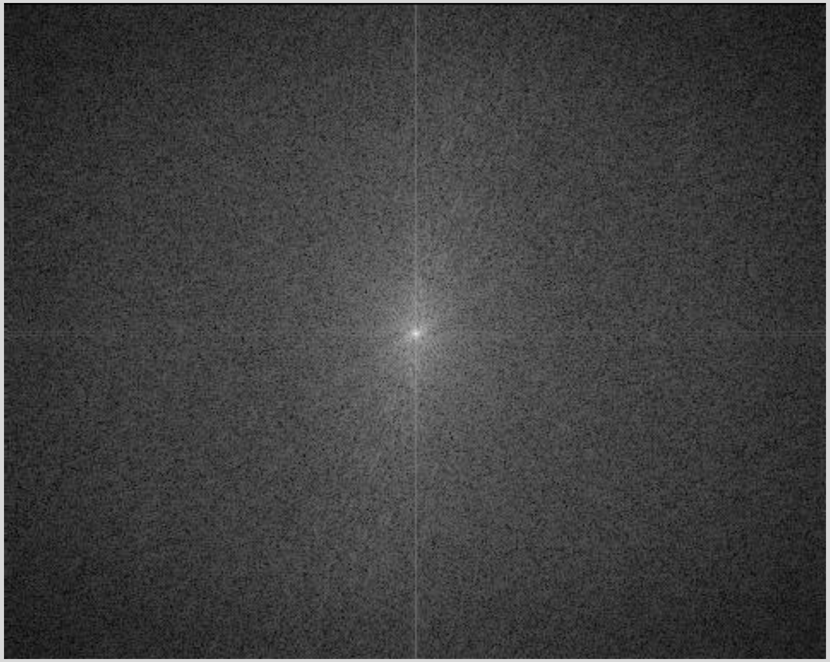
2D rect



Gaussian



Real Example



Computing the 2D Fourier Transform

- Repeat the 1D Fourier twice:
  - Compute the 1D Fourier for each **row**
  - On the result, compute the 1D Fourier for each **column**
  - (Multiply by N , application dependent)

- The 1D Fourier transform is sufficient for computing any multi-dimensional Fourier transform.

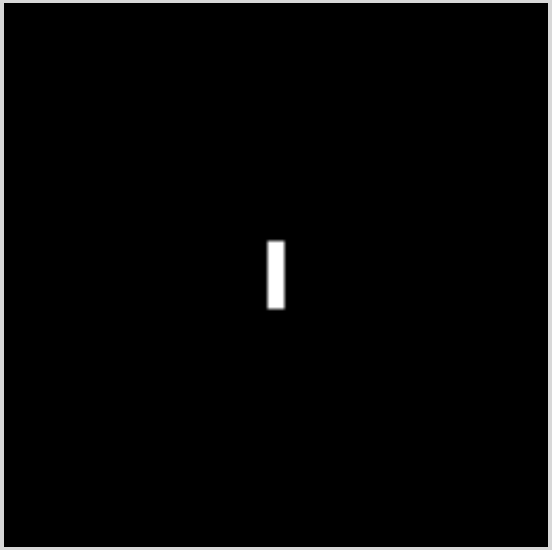
Decomposing 2D DFT to 1D DFT

$$F(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)e^{\frac{-2\pi i(ux+vy)}{N}}$$

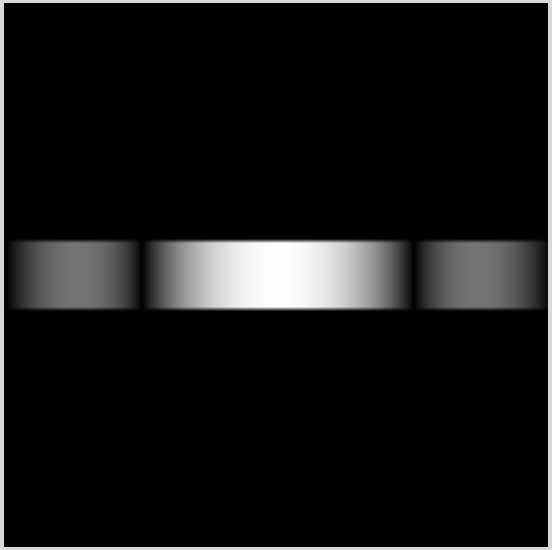
$$F(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\left(e^{\frac{-2\pi iux}{N}}\cdot\sum_{y=0}^{N-1} f(x,y)e^{\frac{-2\pi ivy}{N}}\right)$$

$$F(u,v)=\frac{1}{N}\sum_{x=0}^{N-1}\left(e^{\frac{-2\pi iux}{N}}\cdot F(x,v)\right)$$

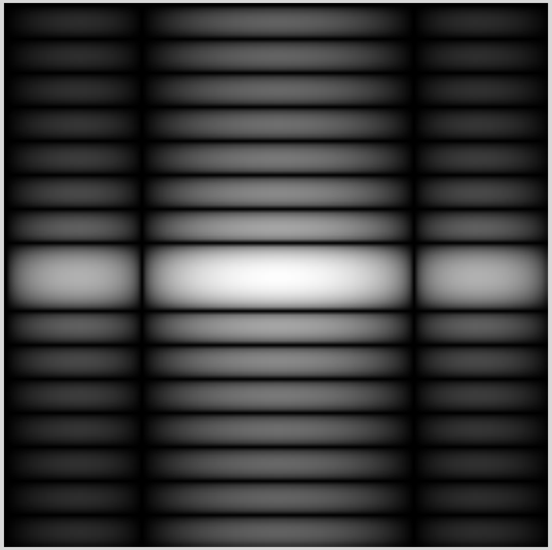
Decomposition Example



(1) Spatial Domain

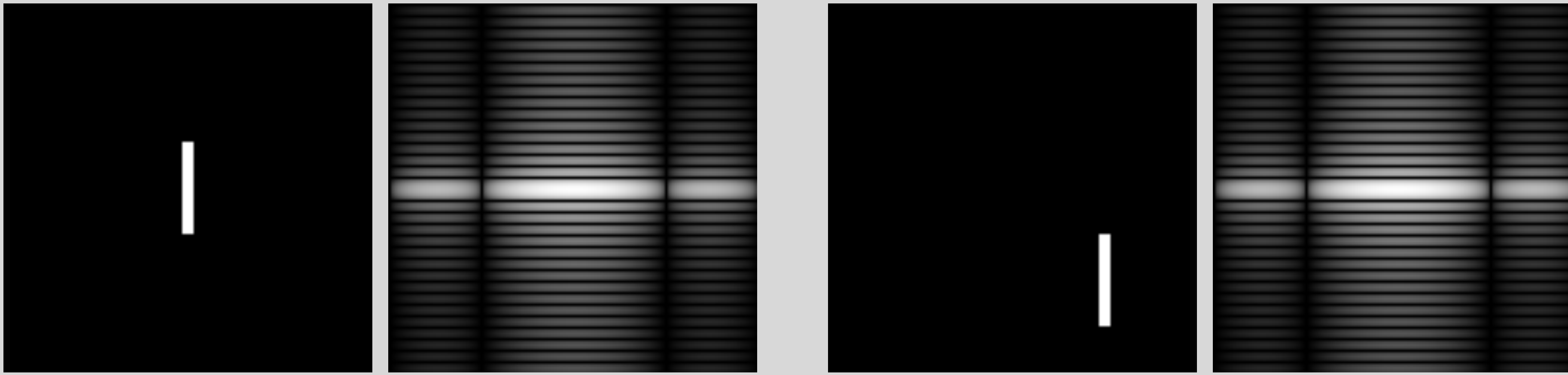


(2) After 1D-DFT on each row



(3) After 1D-DFT on each column

Image Translation



$$f(x-x_0, y-y_0) \Leftrightarrow F(u,v)\cdot e^{\frac{2\pi i(ux_0+vy_0)}{N}}$$

$$F(u-u_0, v-v_0) \Leftrightarrow f(x,y)\cdot e^{-\frac{2\pi i(u_0x+v_0y)}{N}}$$

Fourier is Non Local!

- Fourier Transform supplies a **global representation** of the image in the frequency domain.
- Local objects / features in the image **cannot be assigned** to specific frequencies!

Fourier is Non Local!

- Fourier Transform supplies a **global representation** of the image in the frequency domain.
- Local objects / features in the image **cannot be assigned** to specific frequencies!

- In general:
  - **Low frequencies** represent the coarse structure of the image (large homogenous parts like walls, sky, etc.)
  - **High frequencies** represent the fine details in the image (fine texture, wrinkles, noise, etc.)

Real Example

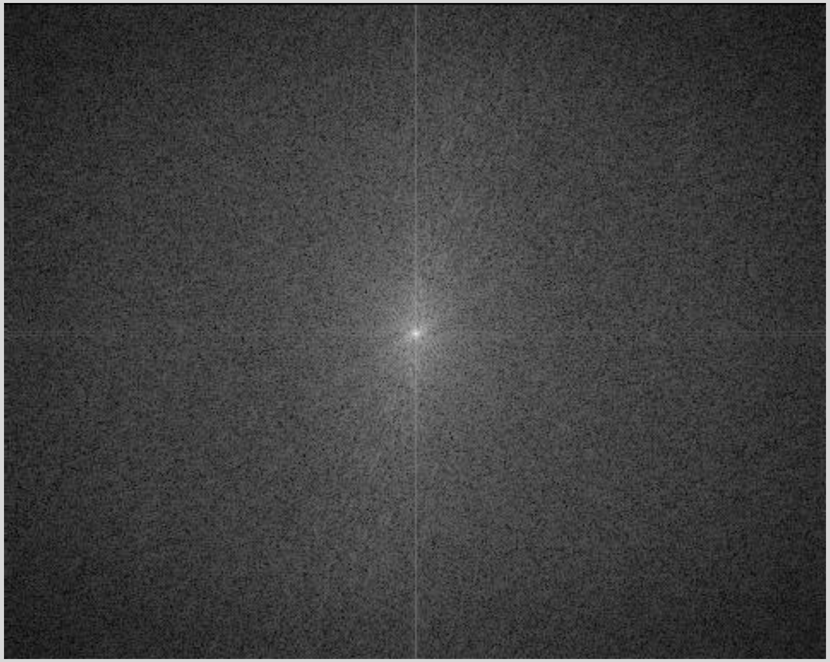


Image Derivatives

Inverse FT: $$f(x,y) = \frac{1}{N}\sum_{u,v} F(u,v)\cdot e^{\frac{2\pi i}{N}(ux+vy)}$$

Derive by *x*: $$\frac{\partial f}{\partial x} = \frac{1}{N}\sum_{u,v} F(u,v)\cdot e^{\frac{2\pi i}{N}(ux+vy)} \boxed{\cdot\left(\frac{2\pi i u}{N}\right)}$$

$$= \frac{2\pi i}{N^2}\cdot\sum_{u,v} F(u,v)\cdot e^{\frac{2\pi i}{N}(ux+vy)}\cdot u$$

Image derivatives

$$\frac{\partial f(x,y)}{\partial x} = \frac{2\pi i}{N} \cdot \Phi^{-1}\left(u \cdot \Phi\left(f(x,y)\right)\right)$$

- Image derivative is the inverse FT of the **weighted** frequency domain.

- **High frequencies** affect the image derivative more than low frequencies.

Image derivatives by FT

- To compute the *x* **derivative** of *f* (up to a constant):
  - Compute the Fourier transform *F*
  - Multiply each Fourier coefficient $F(u,v)$ by u
  - Compute the inverse Fourier transform

- To compute the *y* **derivative** of *f* (up to a constant):
  - Compute the Fourier transform *F*
  - Multiply each Fourier coefficient $F(u,v)$ by v
  - Compute the inverse Fourier transform

Multiplying by u

- To compute the *x* **derivative** of *f* (up to a constant):
  - Compute the Fourier transform *F*
  - *Multiply each Fourier coefficient F(u,v) by u*
  - Compute the inverse Fourier transform

Multiplying by u

- To compute the *x* **derivative** of *f* (up to a constant):
  - Compute the Fourier transform *F*
  - *Multiply each Fourier coefficient F(u,v) by u*
  - Compute the inverse Fourier transform

$$(0,1,2,\ldots,{}^{N}\!/_{2},\ldots,N-1)$$

Multiplying by u

- To compute the *x* **derivative** of *f* (up to a constant):
  - Compute the Fourier transform *F*
  - *Multiply each Fourier coefficient F(u,v) by u*
  - Compute the inverse Fourier transform

$$\cancel{(0,1,2,\ldots,N/2,\ldots,N-1)}$$

The highest frequency is N/2

Multiplying by u

- To compute the *x* **derivative** of *f* (up to a constant):
  - Compute the Fourier transform *F*
  - *Multiply each Fourier coefficient F(u,v) by u*
  - Compute the inverse Fourier transform

(try to use Symmetric + Periodicity)

$$\cancel{(0,1,2,\ldots,{}^{N}\!/_{2},\ldots,N-1)}$$

$$(0,1,\ldots,{}^{N}\!/_{2}-1,-{}^{N}\!/_{2},\ldots,-1)$$

The highest frequency is N/2

Multiplying by u

- To compute the *x* **derivative** of *f* (up to a constant):
  - Compute the Fourier transform *F*
  - <u>*Multiply each Fourier coefficient F(u,v) by u*</u>
  - Compute the inverse Fourier transform

$$\cancel{(0,1,2,\ldots,N/2,\ldots,N-1)}$$

The highest frequency is N/2

$$(0,1,\ldots,N/2-1,-N/2,\ldots,-1)$$

Or

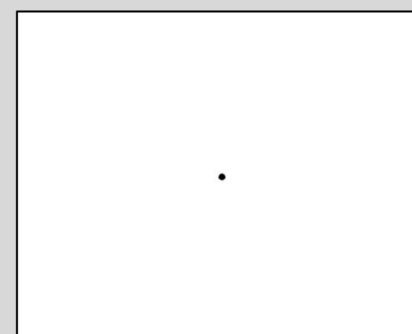
$$(-N/2,\ldots,0,\ldots,N/2-1)$$

In this option: should center Fourier Transform of the image (F) as well

Example

Reminder: $F(0,0) = \bar{f}$

Multiply the Fourier transform by the filter:



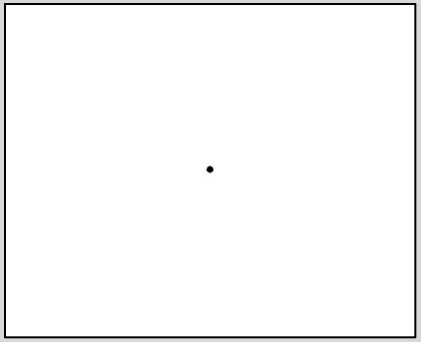
Example

Reminder: $F(0,0) = \bar{f}$

$$\xrightarrow{\quad\quad\quad\quad}$$

$$F(0,0) = 0$$

Multiply the Fourier transform by the filter:



Filtering in the frequency domain: General Scheme

Input image $f(x,y)$

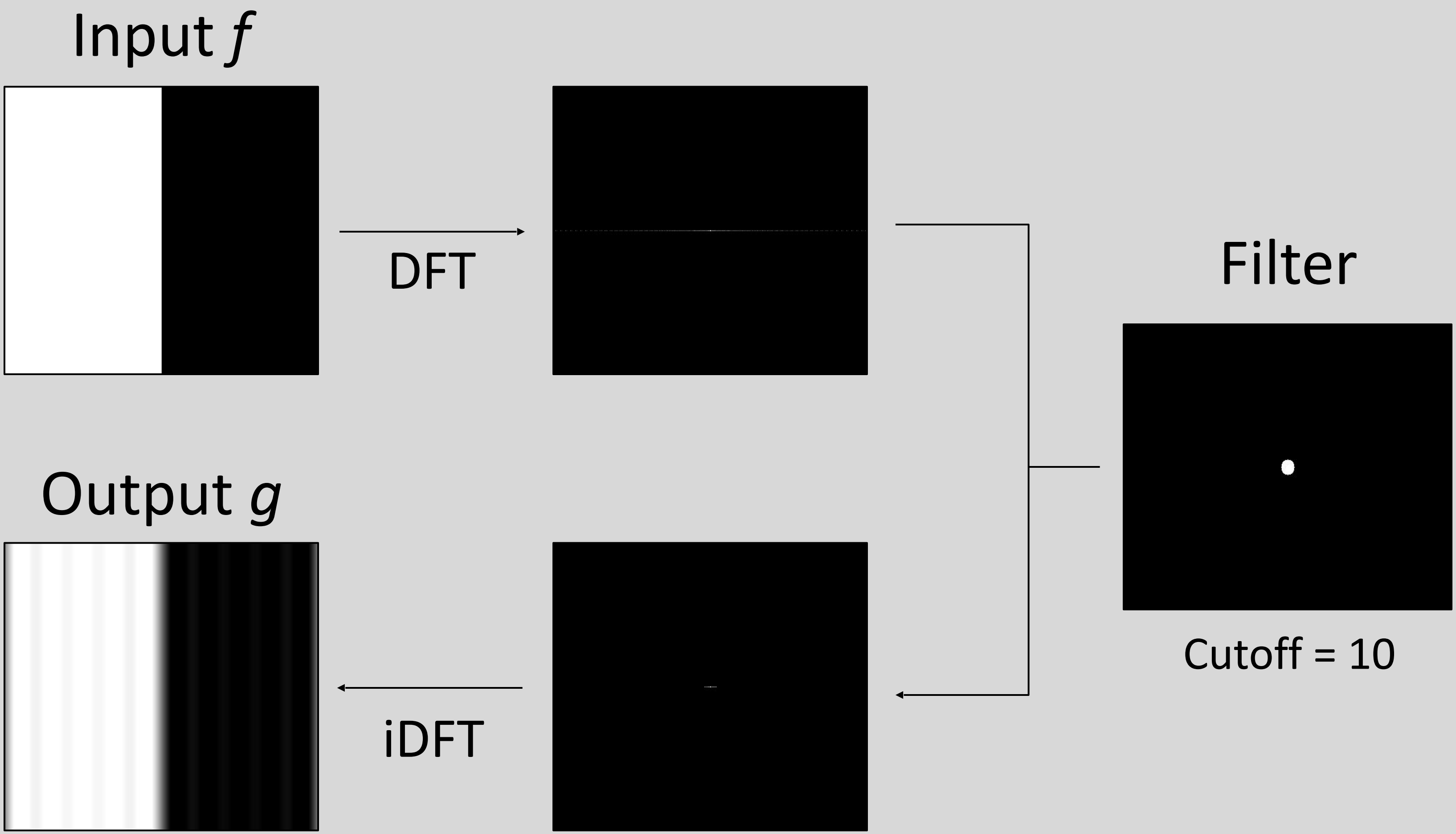
Fourier Transform: $F(u,v)$

Filter function: $H(u,v) \cdot F(u,v)$

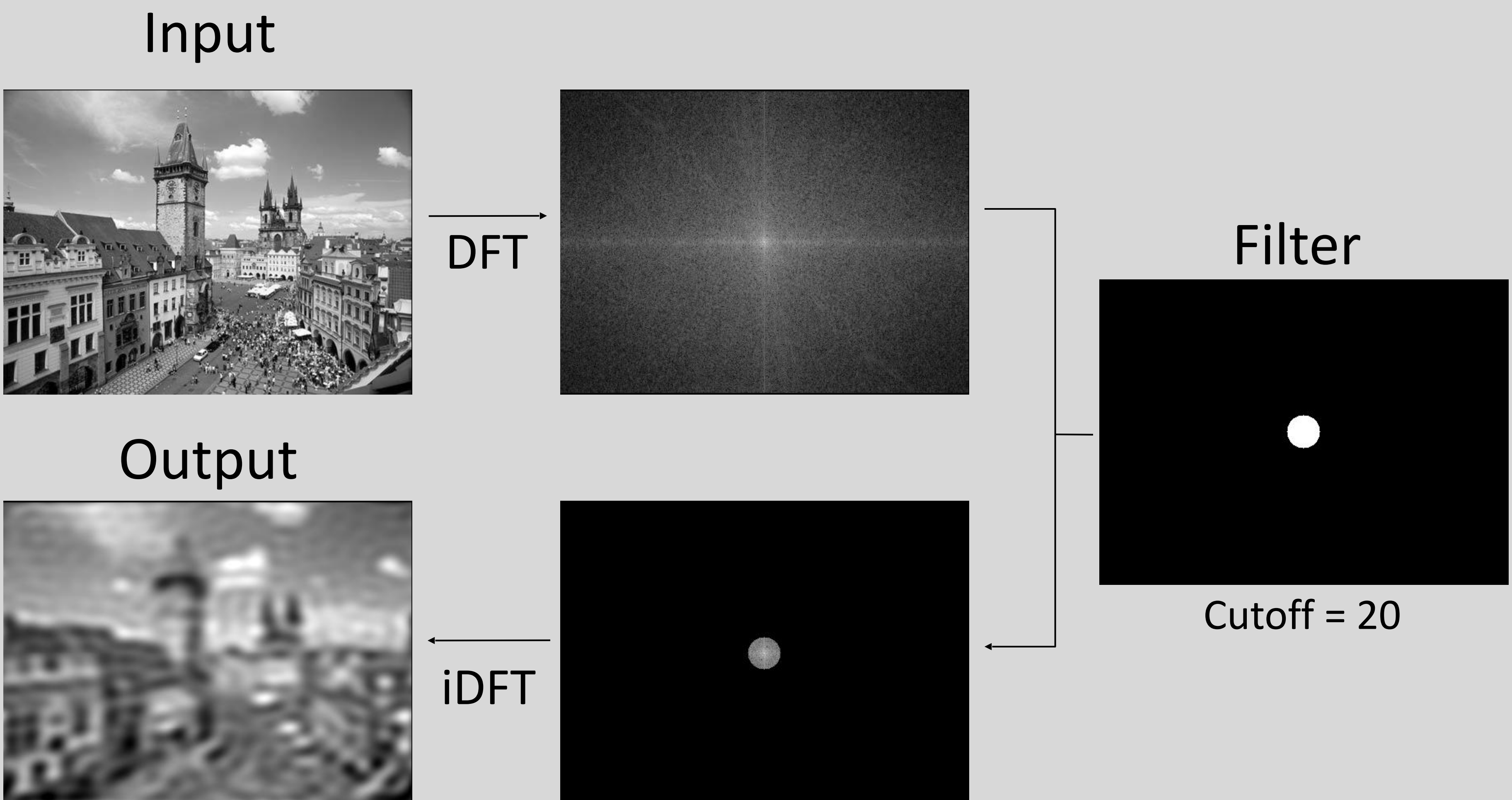
Inverse Fourier Transform

Output image $g(x,y)$

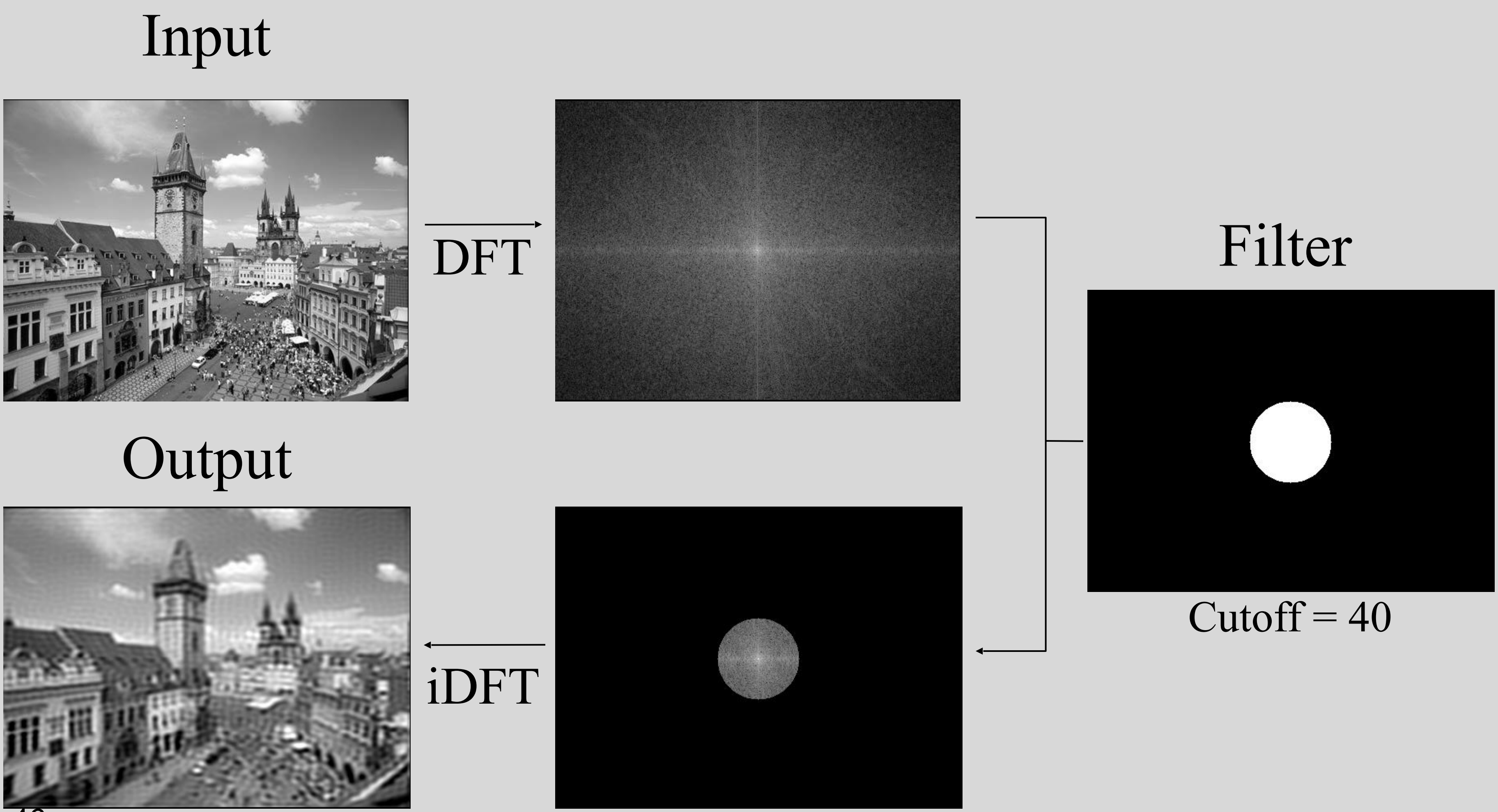
Ideal Low-pass Filters



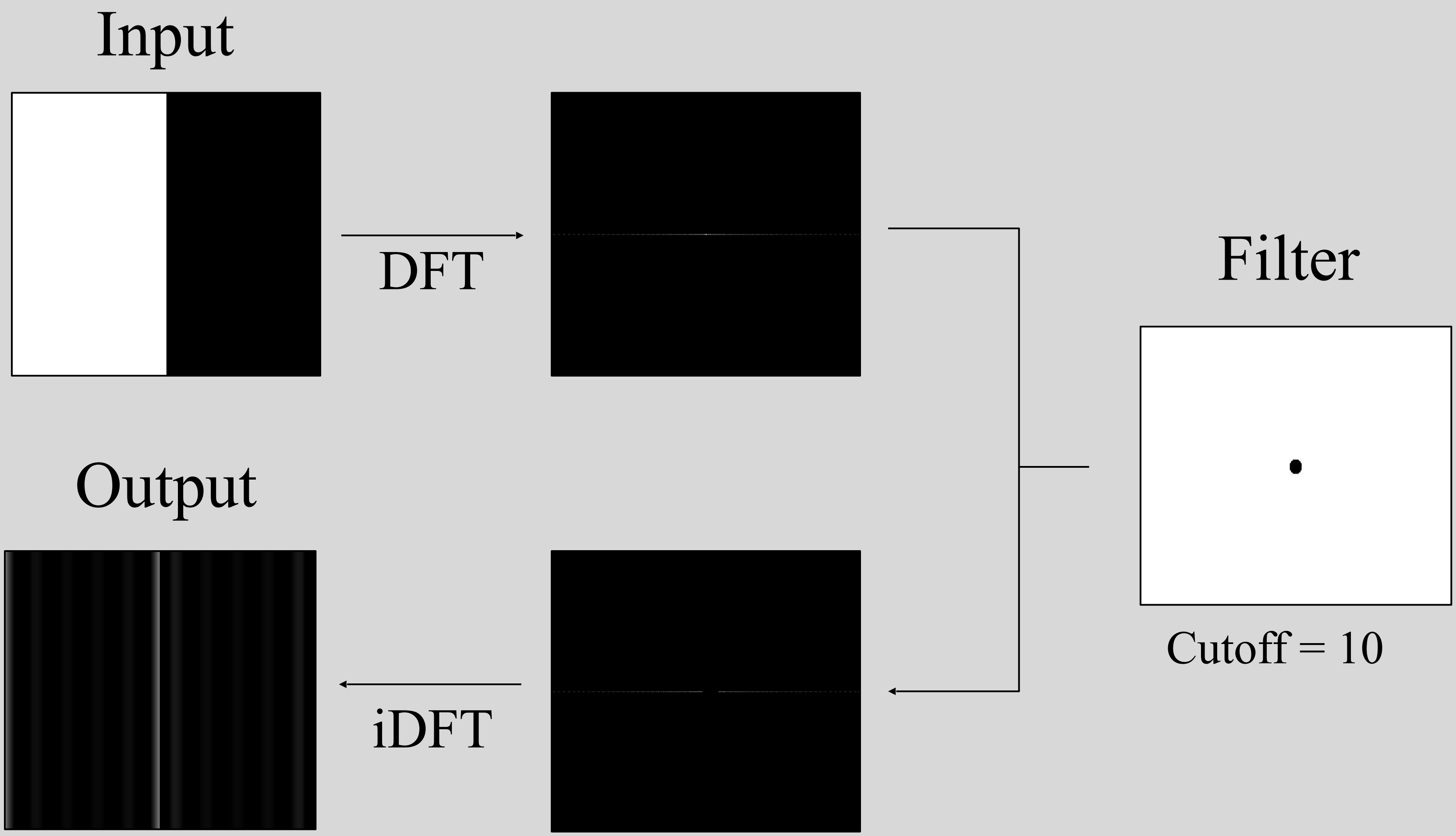
Ideal Low-pass Filters



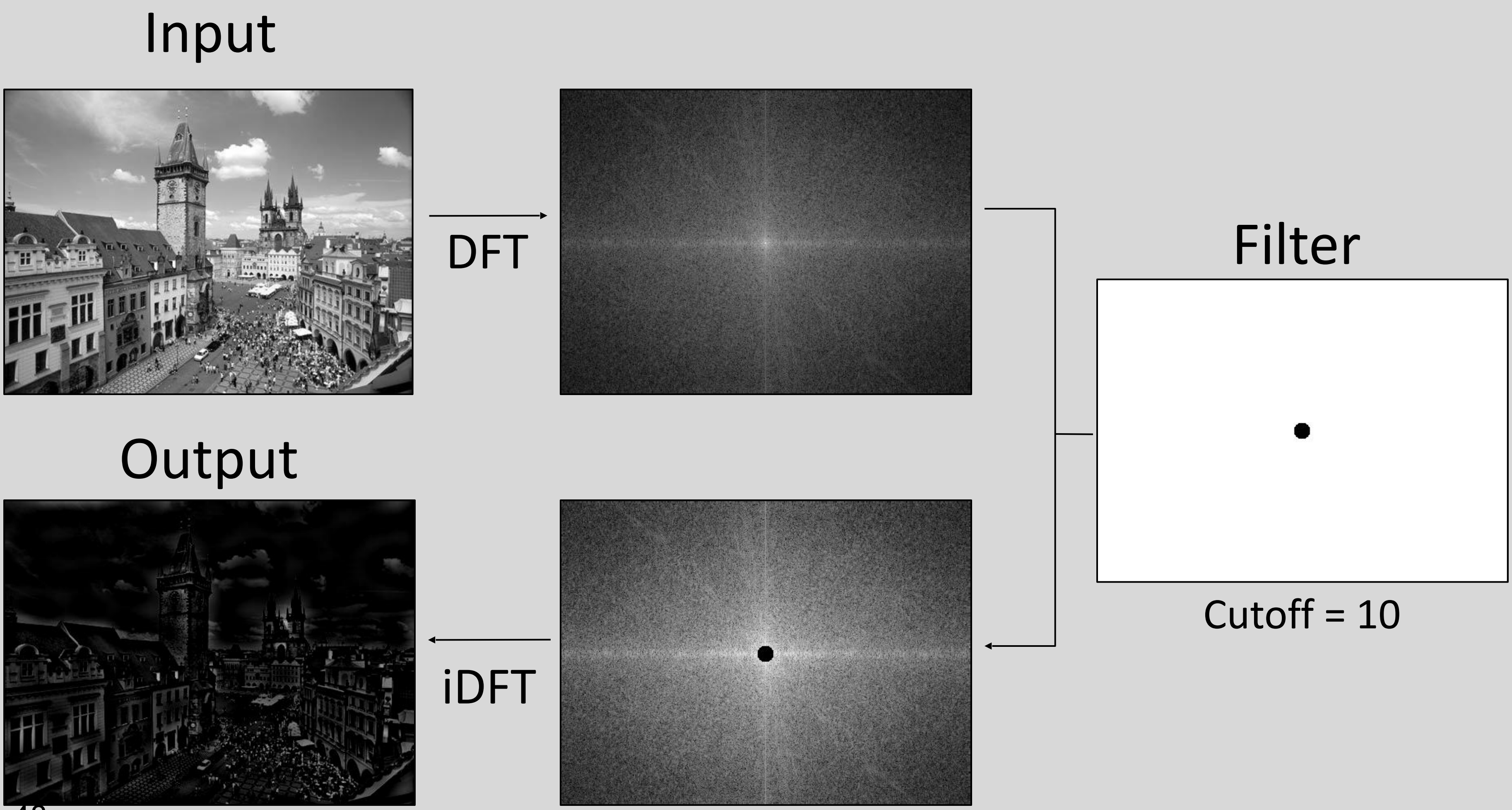
Cutoff = 40



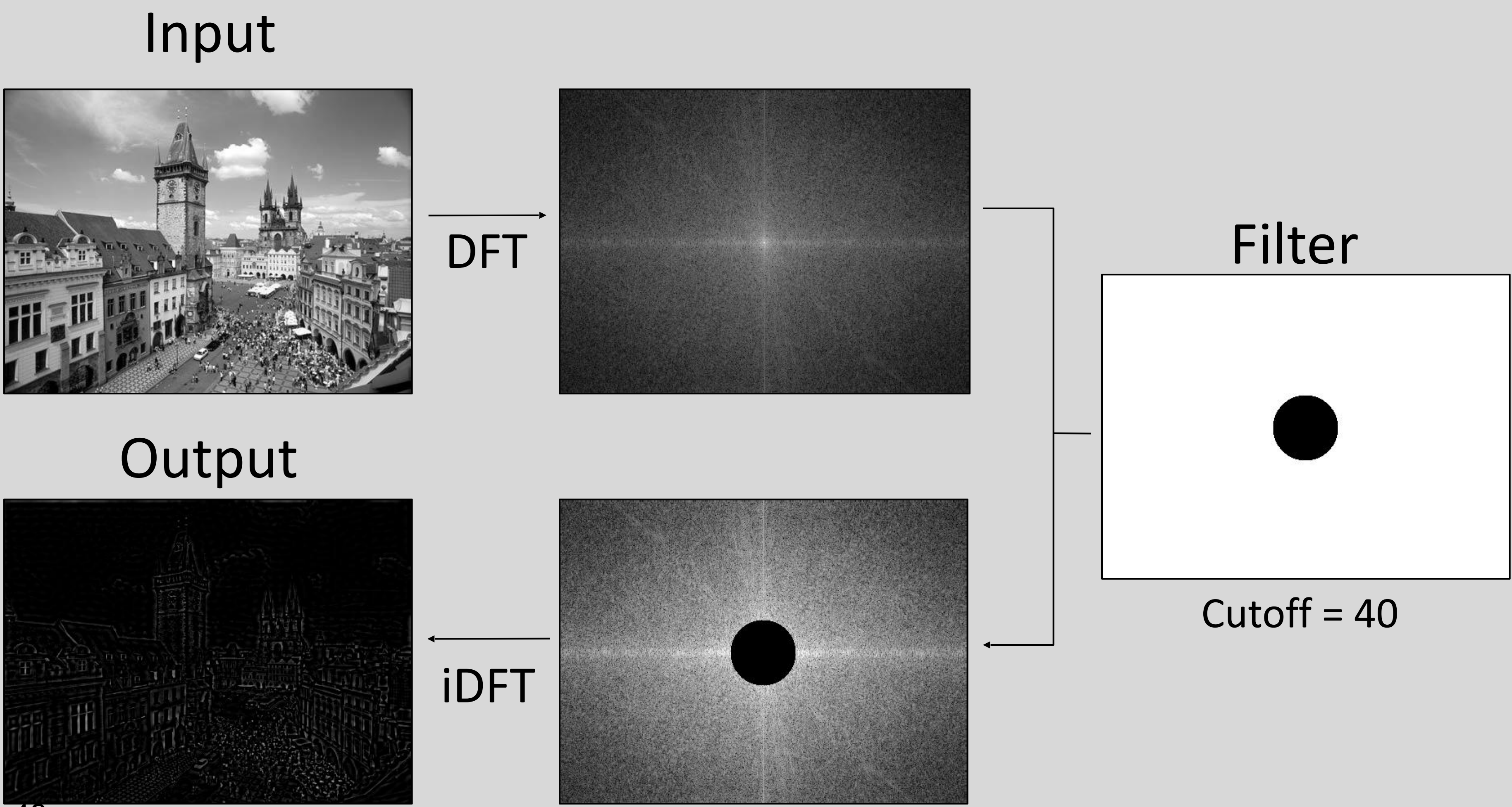
Ideal High-pass Filters



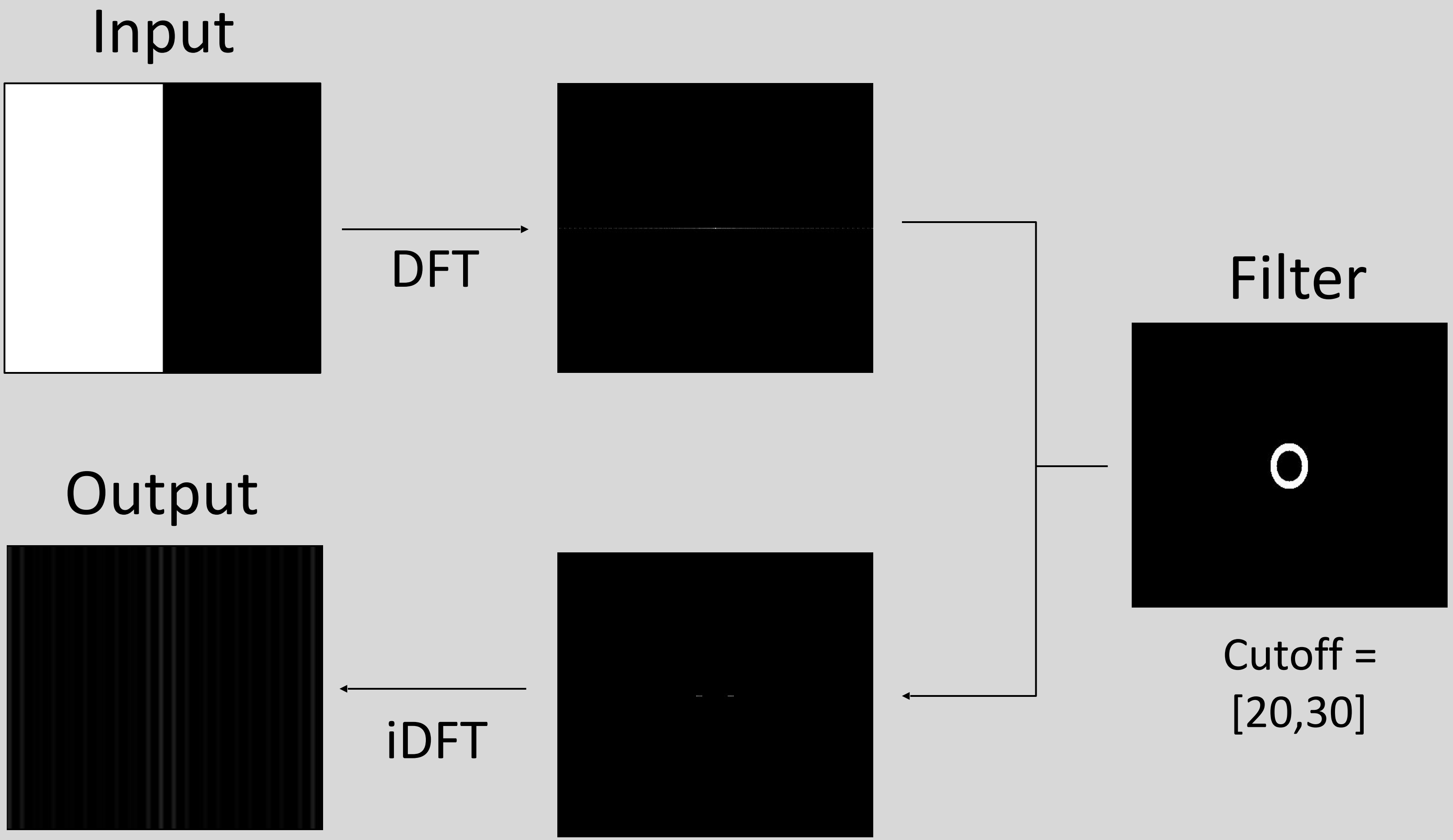
Ideal High-pass Filter



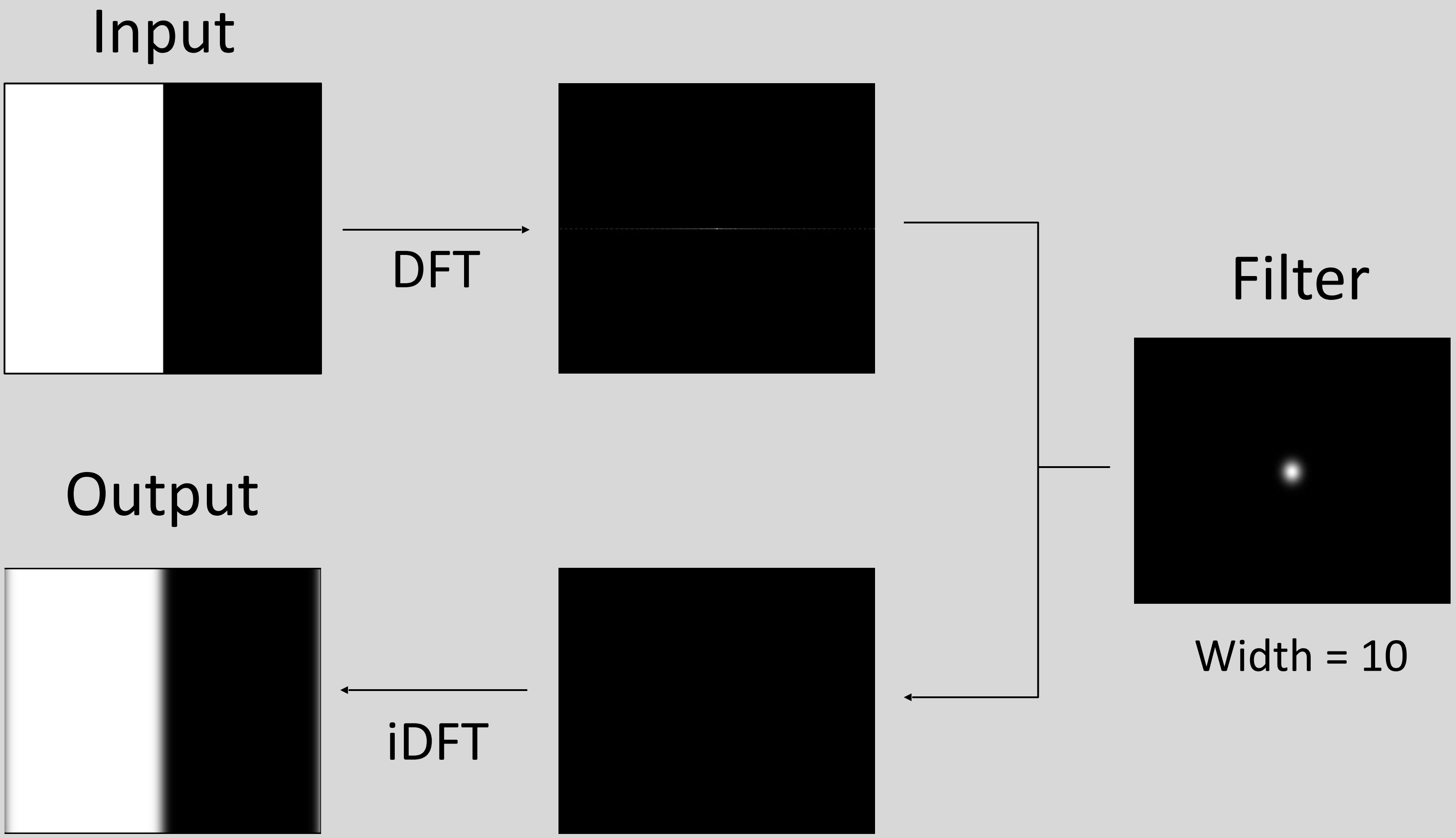
Ideal High-pass Filter



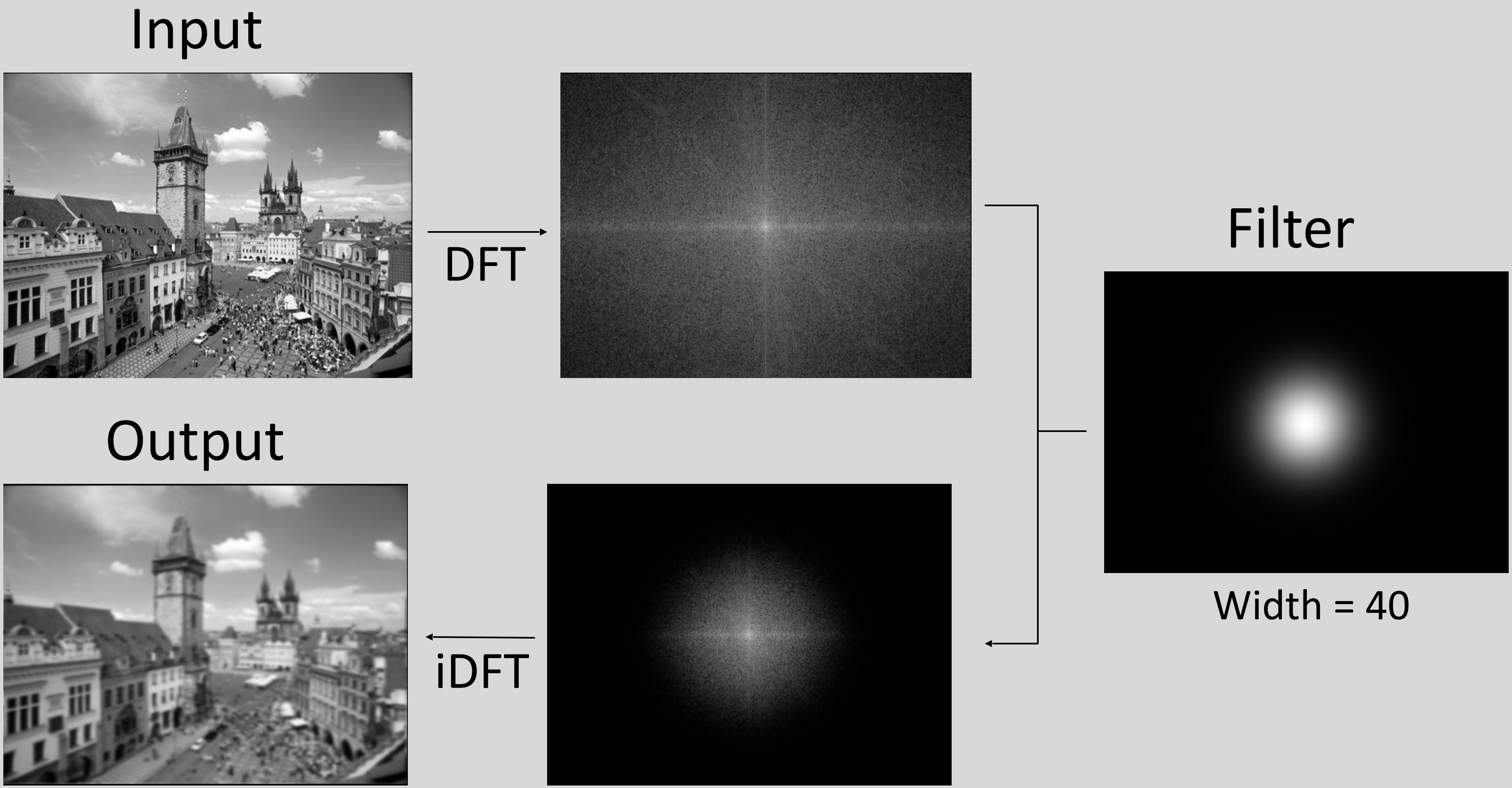
Ideal Band-pass Filter



Gaussian Filter



Gaussian Filter



Resemblance to Convolution

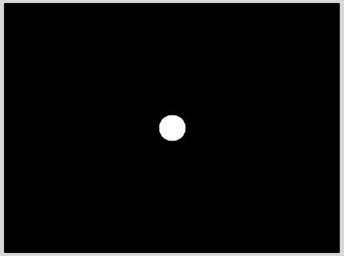
Fourier Filter

$$F(u,v)\cdot H(u,v)$$

Convolution Filter

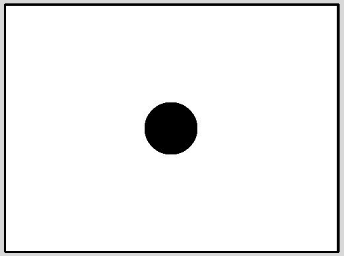
$$f(x,y) * g(x,y)$$

Low-pass filter



$$\frac{1}{16}\begin{pmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{pmatrix}$$

High-pass filter



$$\begin{pmatrix}0 & -1 & 0\\ -1 & 4 & -1\\ 0 & -1 & 0\end{pmatrix}$$

The Convolution Theorem

Reminder: $$(f * g)(x,y) = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} f(x-i, y-j)\; g(i,j)$$

<u>The Convolution Theorem:</u>

$$\Phi(f * g) = F \cdot G$$

$$\Phi(f \cdot g) = F * G$$

Convolution Vs. Fourier

Convolution by Fourier:

$$f * g = \Phi^{-1}(F \cdot G)$$

Complexity (using the **FFT algorithm**): *O*(*N* log *N*), where *N* is the number of pixels in the image.

- Different Fourier transform phenomena can be explained by convolution, and vice versa.
- The Fourier interpretation is used for designing convolution filters.