CSC496: Deep learning in computer vision

Prof. Bei Xiao

Lecture 7: Fourier transform, Temporal Processing

Lecture Forecast

- Today (September 23): FT, Convolution in FT, Start temporal processing
- Thursday (September 26): Temporal processing, spatial-temporal gaussians, Very important to understand CNN.
- Monday (September 30th) : Multiscale Pyramids. Quiz 2.
- Thursday (October 3): Introduction to Machine learning: Linear models.
- Monday (October 7th): Supervised learning.
- Thursday (October 10th): Neural networks
- Monday (October 14th): Tensorflow tutorials
- Thursday (October 18th): Back propagation

Take-home reading

- SignalProcessing.pdf up to page 46

The Discrete Fourier transform

Discrete Fourier Transform (DFT) transforms an image
$f[m, m]$ into the complex image Fourier transform $F[u, v]$ as:

$$F[u,v] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n,m] \exp\left(-2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

The inverse Fourier transform is:

$$f[n,m] = \frac{1}{NM} \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F[u,v] \exp\left(+2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

2D Discrete Fourier Transform

$$F[u,v] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n,m] \exp\left(-2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

Using the real and imaginary components:

$$F[u,v] = Re\{F[u,v]\} + j\,Imag\{F[u,v]\}$$

Or using a polar decomposition:

$$F[u,v] = A[u,v] \exp(j\theta[u,v])$$

2D Discrete Fourier Transform

Real and imaginary parts of FT will have the symmetries:

$$Re\{F[u,v]\} = Re\{F[-u,-v]\}$$

$$Imag\{F[u,v]\} = -Imag\{F[-u,-v]\}$$

Proof: shift in FT

- How do we proof the following statement:

- If we displace a signal in the spatial domain, it results in multiplying its Fourier transform by a complex exponential?

Properties for the DFT

- Shift in space

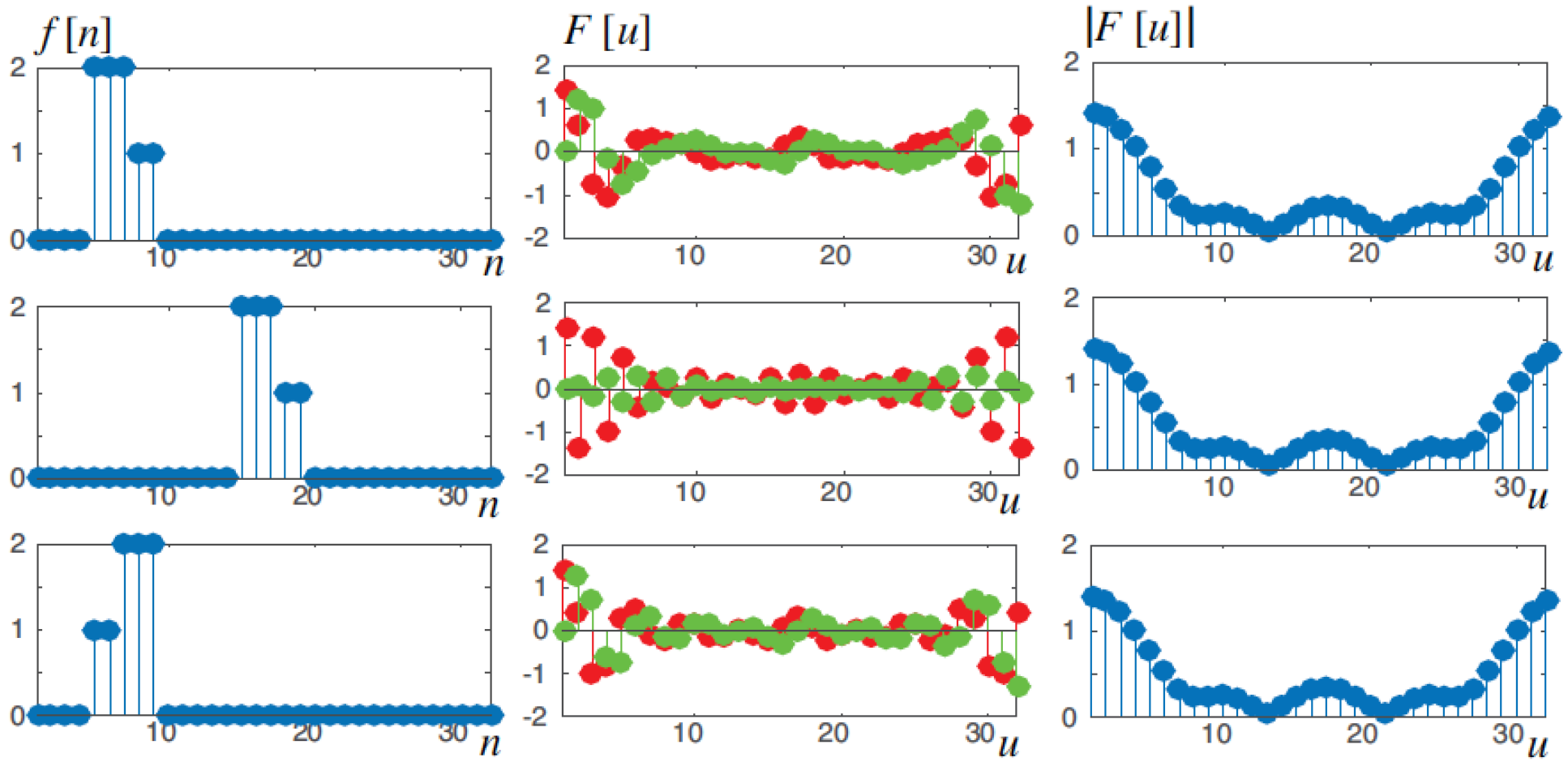
$$DFT\{f[n-n_0, m-m_0]\} =$$

$$= \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n-n_0, m-m_0] \exp\left(-2\pi j \left(\frac{u\,n}{N} + \frac{v\,m}{M}\right)\right) =$$

$$= \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n, m] \exp\left(-2\pi j \left(\frac{u\,(n+n_0)}{N} + \frac{v\,(m+m_0)}{M}\right)\right) =$$

$$= F[u, v] \exp\left(-2\pi j \left(\frac{u\,n_0}{N} + \frac{v\,m_0}{M}\right)\right)$$

Properties for the DFT: translation invariant



Only the phase changes! The magnitude is translation invariant.

Properties for the DFT: shift in space is modulation in frequency domain

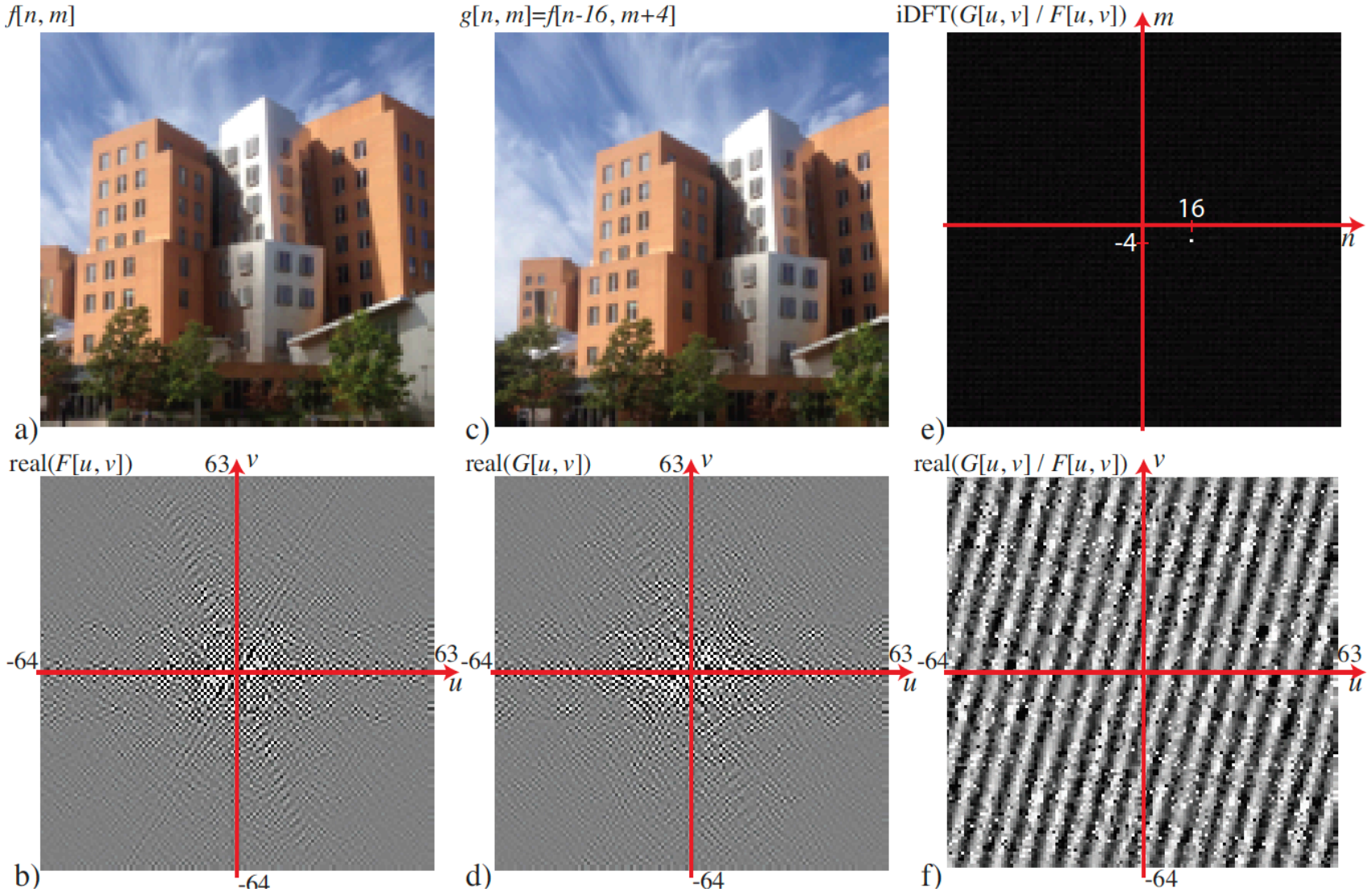
- Modulation

$$f[n,m]\cos\left(2\pi j\left(\frac{u_0 n}{N}+\frac{v_0 m}{M}\right)\right)$$

$$f[n,m]\exp\left(-2\pi j\left(\frac{u_0 n}{N}+\frac{v_0 m}{M}\right)\right)$$

Multiplying by a complex exponential results in a translation of the DFT

$$DFT\left\{f[n,m]\exp\left(-2\pi j\left(\frac{u_0 n}{N}+\frac{v_0 m}{M}\right)\right)\right\}=F[u-u_0,v-v_0]$$



Close to an impulse

$$f[n,m]\cos\left(2\pi j\left(\frac{u_0 n}{N}+\frac{v_0 m}{M}\right)\right)$$

Some useful transforms

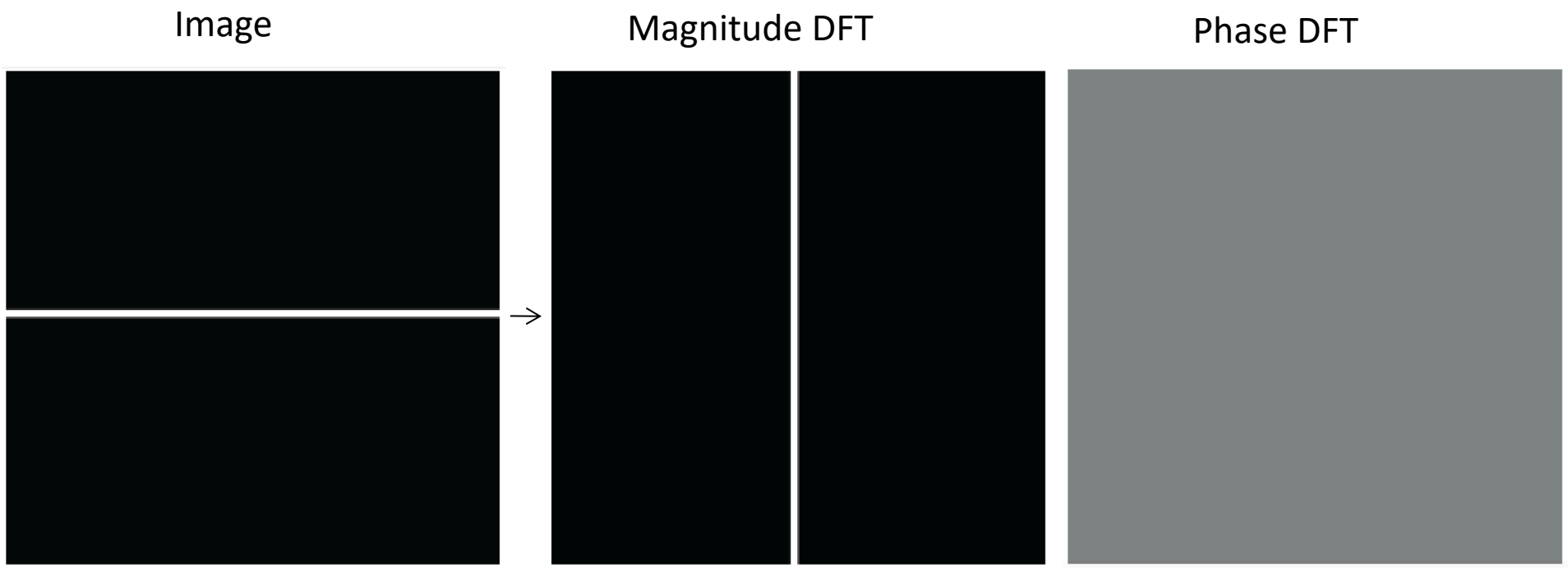
Fourier transform of the Delta function $\delta[n, m]$:

$$F[u,v] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \delta[n,m] \exp\left(-2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right) = 1$$

Observation: if we think in terms of the inverse DFT, this means that:

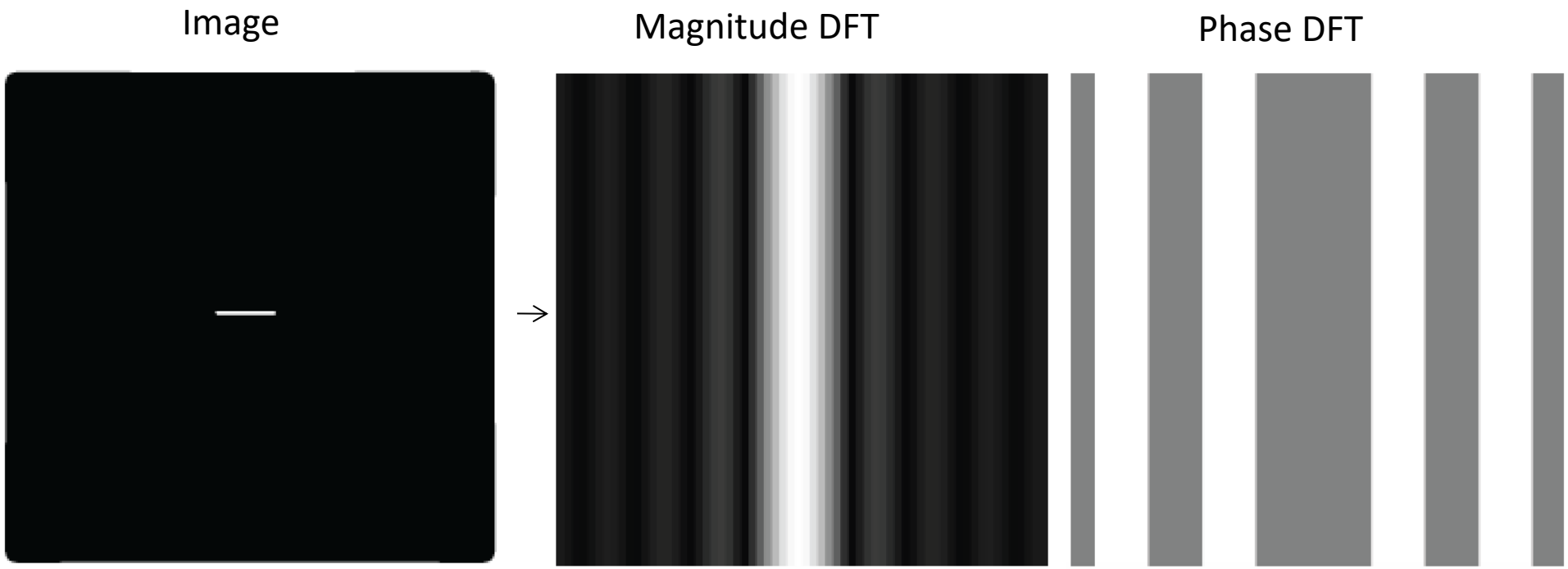
$$\delta[n,m] = \frac{1}{NM} \sum_{u=-N/2}^{N/2} \sum_{v=-M/2}^{M/2} \exp\left(2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

Some important Fourier transforms

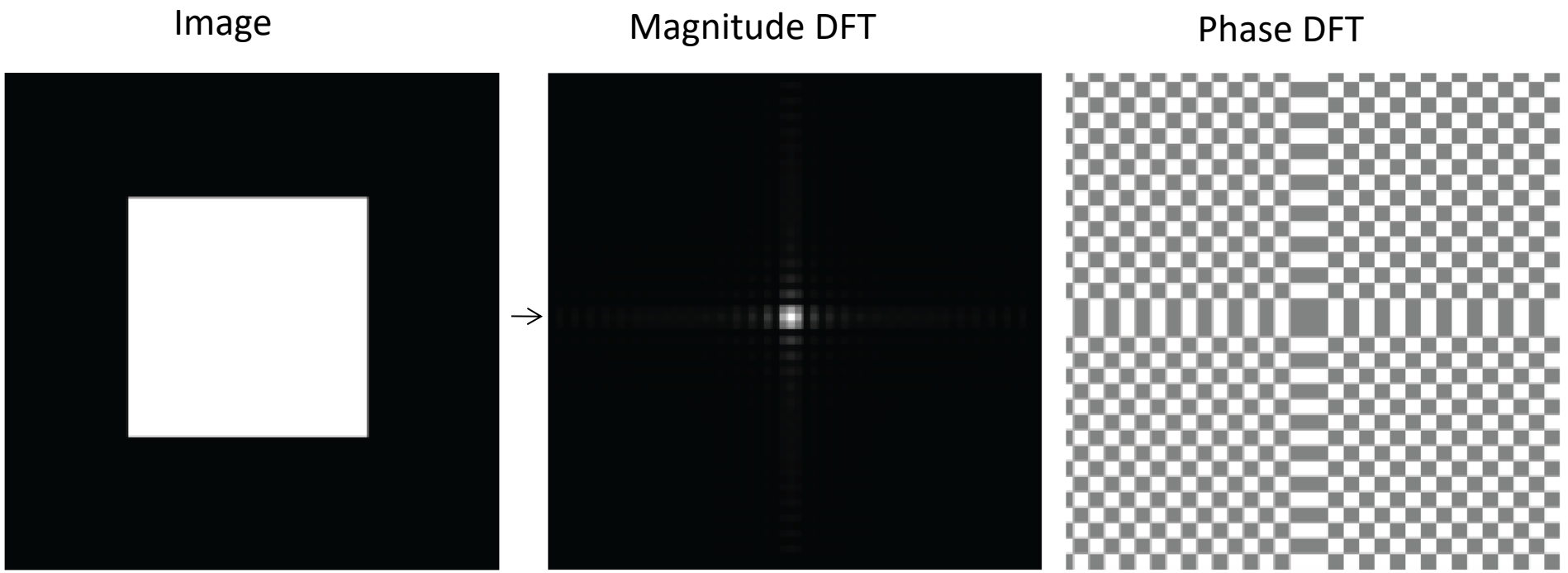


Images are 64x64 pixels.

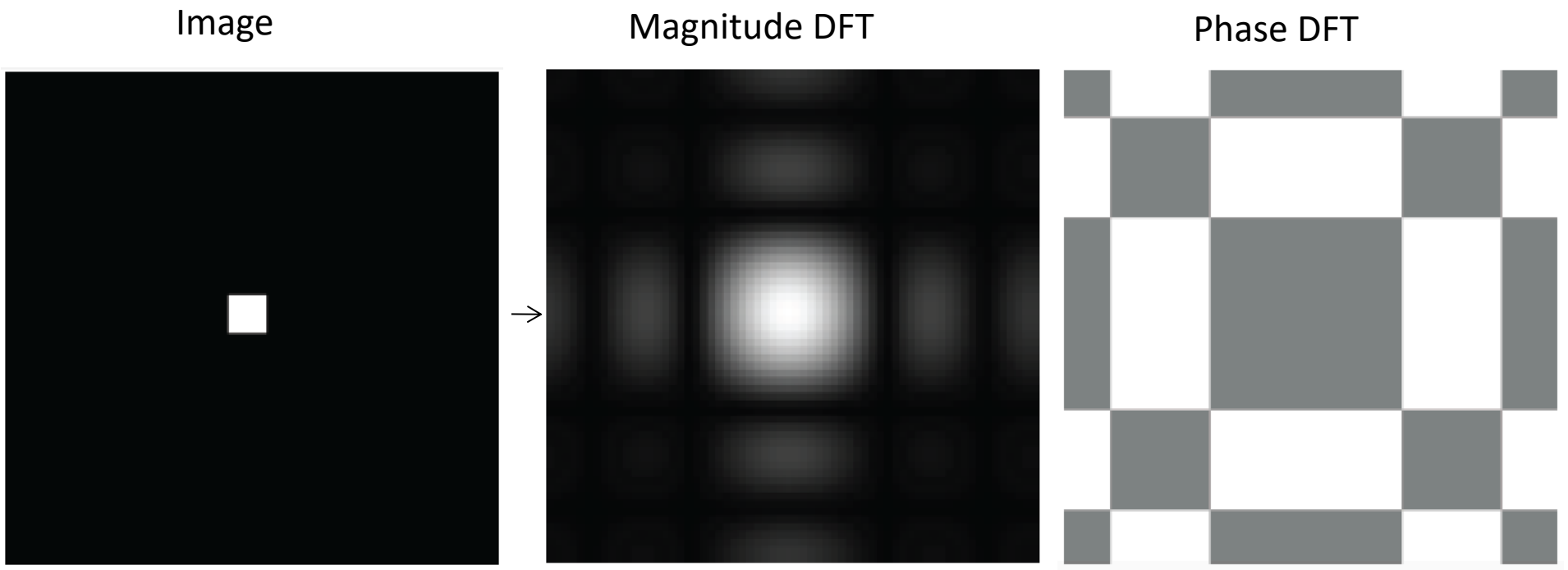
Some important Fourier transforms



Some important Fourier transforms

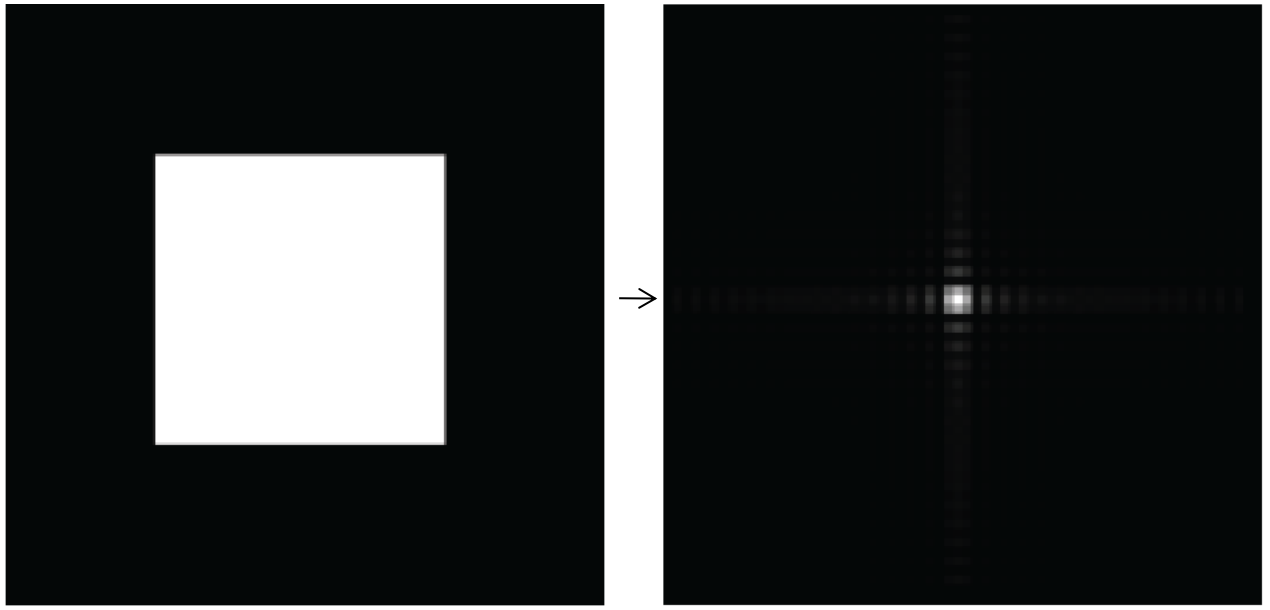


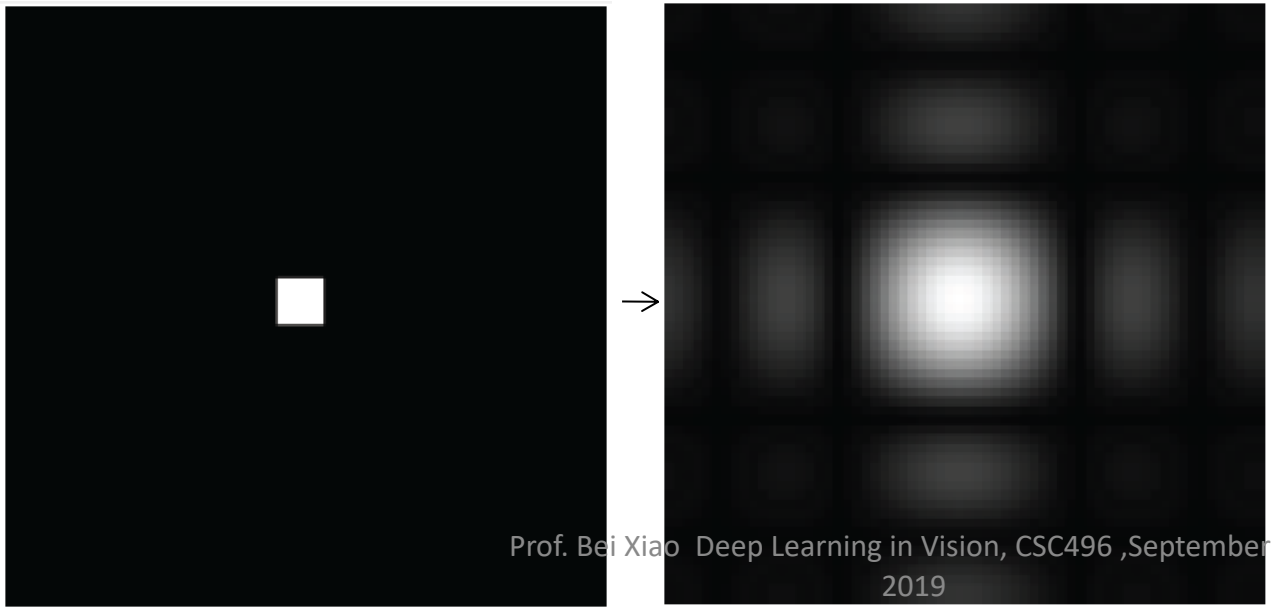
Some important Fourier transforms



Image

Magnitude DFT

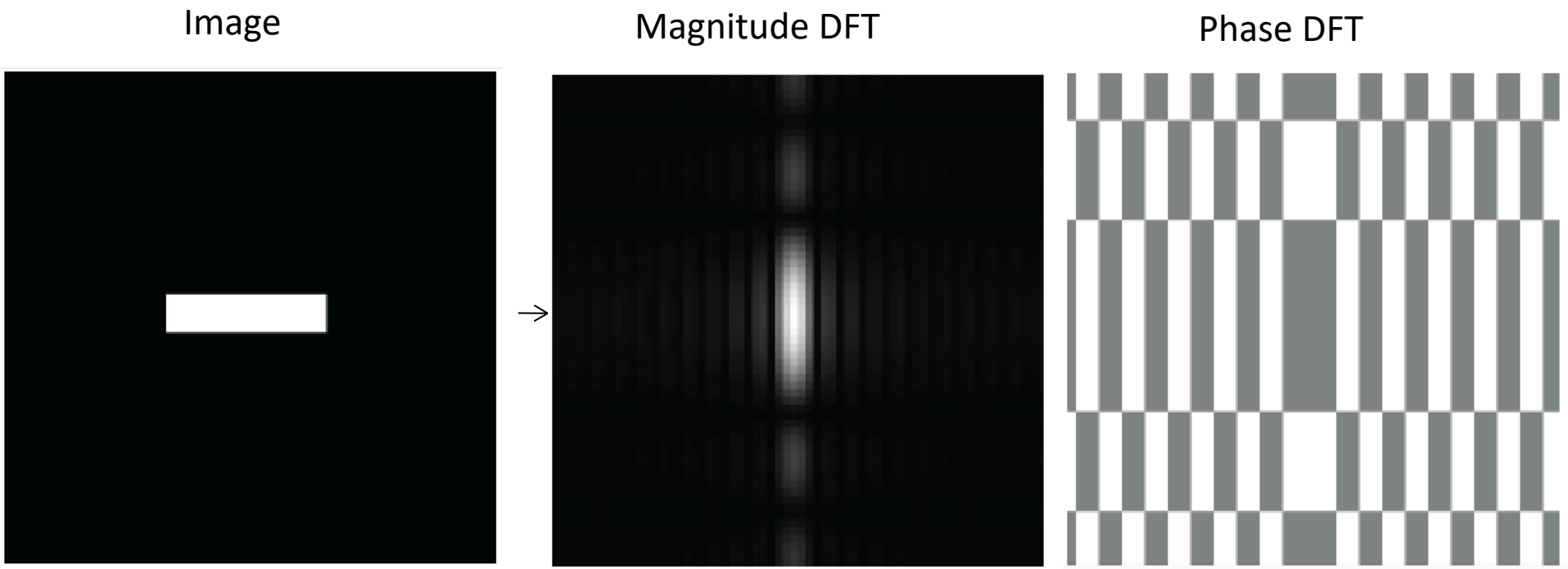




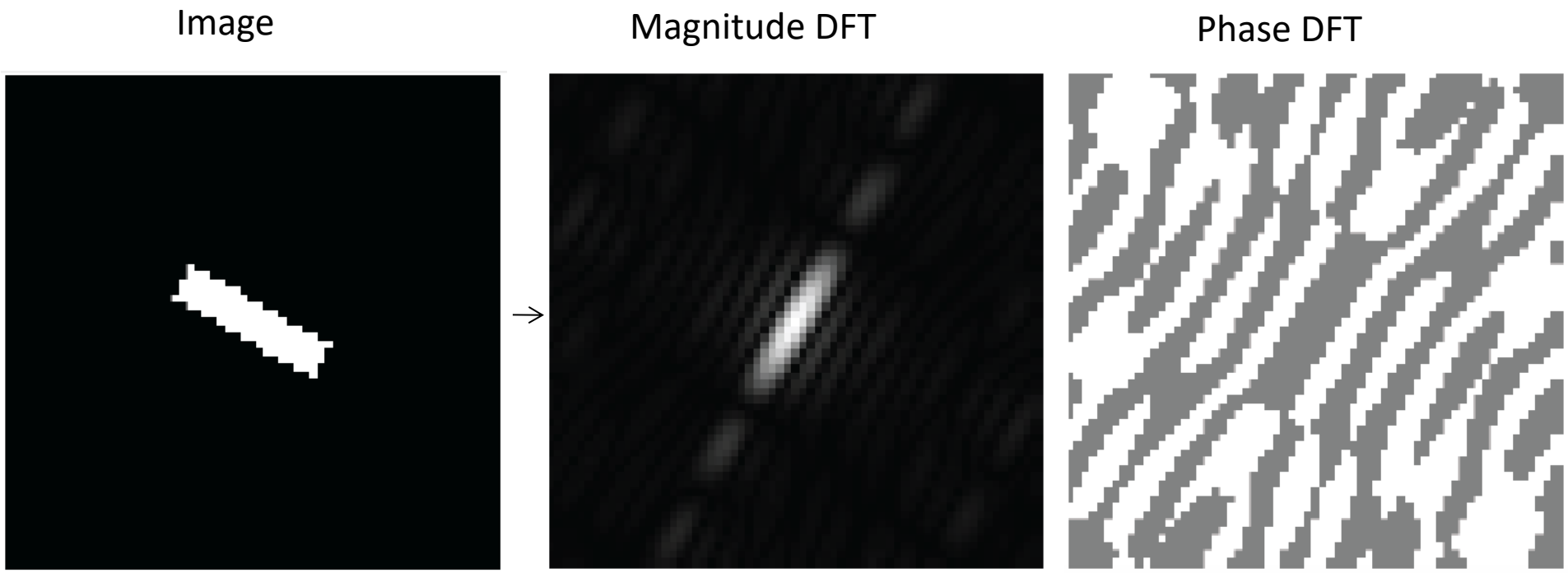
Scale

Small image
details produce content in
high spatial frequencies

Some important Fourier transforms

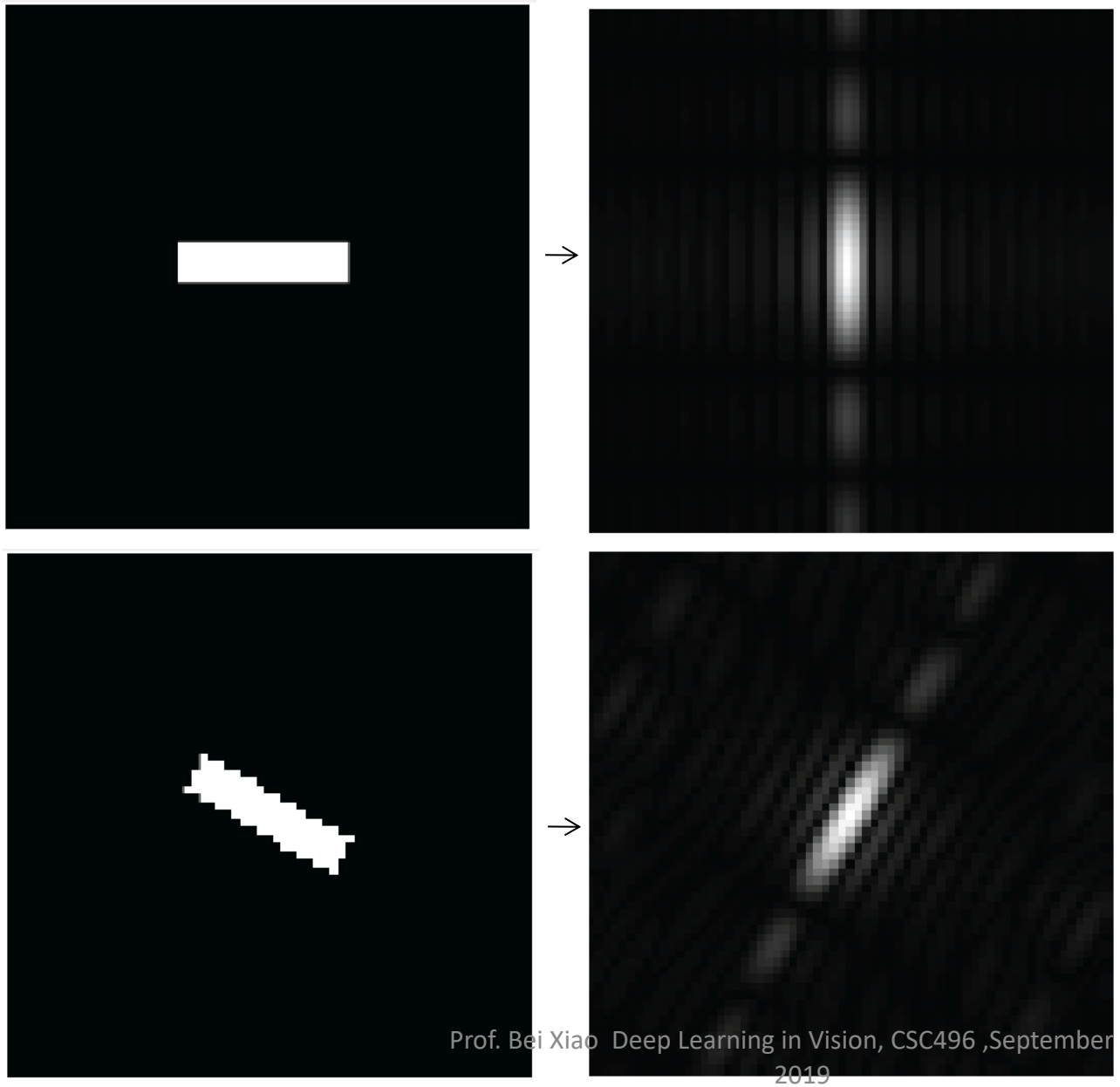


Some important Fourier transforms



Image

Magnitude DFT

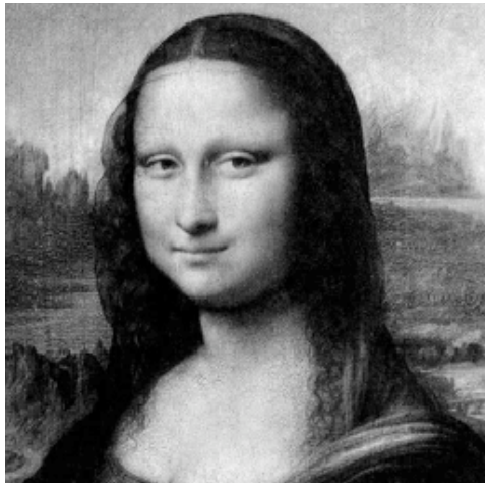


Orientation

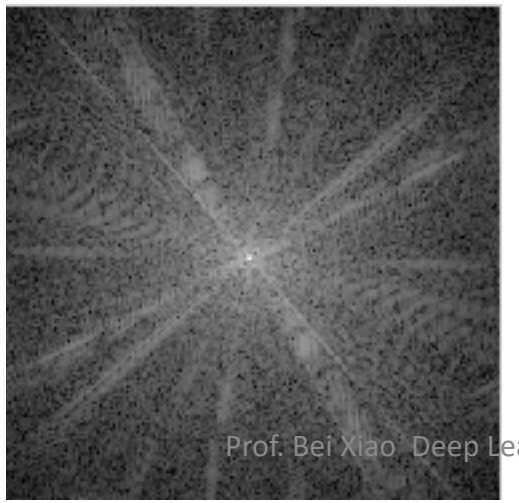
A line transforms to a line oriented perpendicularly to the first.

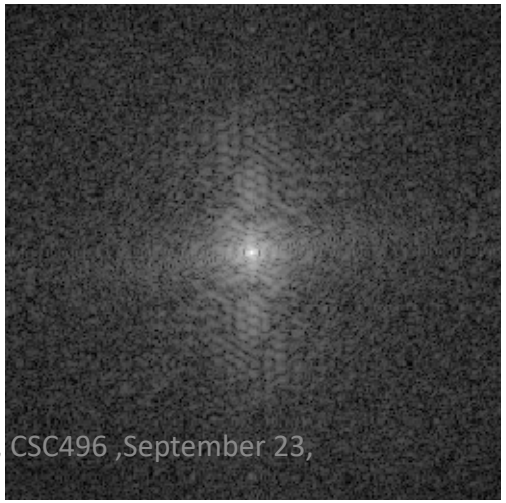
The Fourier Transform of some important images

Image



Log(1+Magnitude FT)





More properties for the DFT

$$F[u,v] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} f[n,m] \exp\left(-2\pi j \left(\frac{un}{N} + \frac{vm}{M}\right)\right)$$

DFT of the convolution

$$f = g \circ h \longleftrightarrow F[u,v] = G[u,v]H[u,v]$$

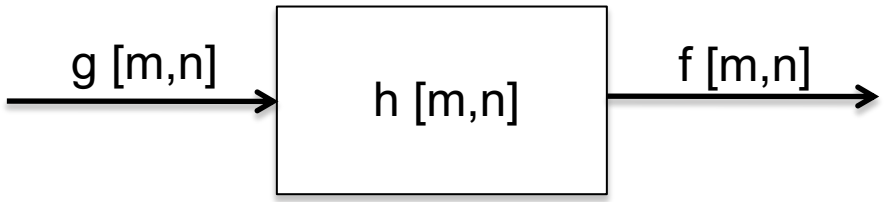
$$F[u,v] = DFT\{g \circ h\}$$

$$= \sum_{m=0}^{M-1}\sum_{n=0}^{N-1}\sum_{k=0}^{M-1}\sum_{l=0}^{N-1} g[m-k,n-l]\,h[k,l]\exp\left(-2\pi j\left(\frac{mu}{M}+\frac{nv}{N}\right)\right)$$

$$F[u,v] = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} h[k,l]\sum_{m'=-k}^{M-k-1}\sum_{n'=-l}^{N-l-1} g[m',n']\exp\left(-2\pi j\left(\frac{(m'+k)u}{M}+\frac{(n'+l)v}{N}\right)\right)$$

$$F[u,v] = \sum_{k=0}^{M-1}\sum_{l=0}^{N-1} G[u,v]\exp\left(-2\pi j\left(\frac{ku}{M}+\frac{lv}{N}\right)\right)h[k,l]$$

Linear filtering



In the spatial domain:

$$f[m,n] = h \circ g = \sum_{k,l} h[m-k, n-l]\, g[k,l]$$

In the frequency domain:

$$F[u,v] = G[u,v]\, H[u,v]$$

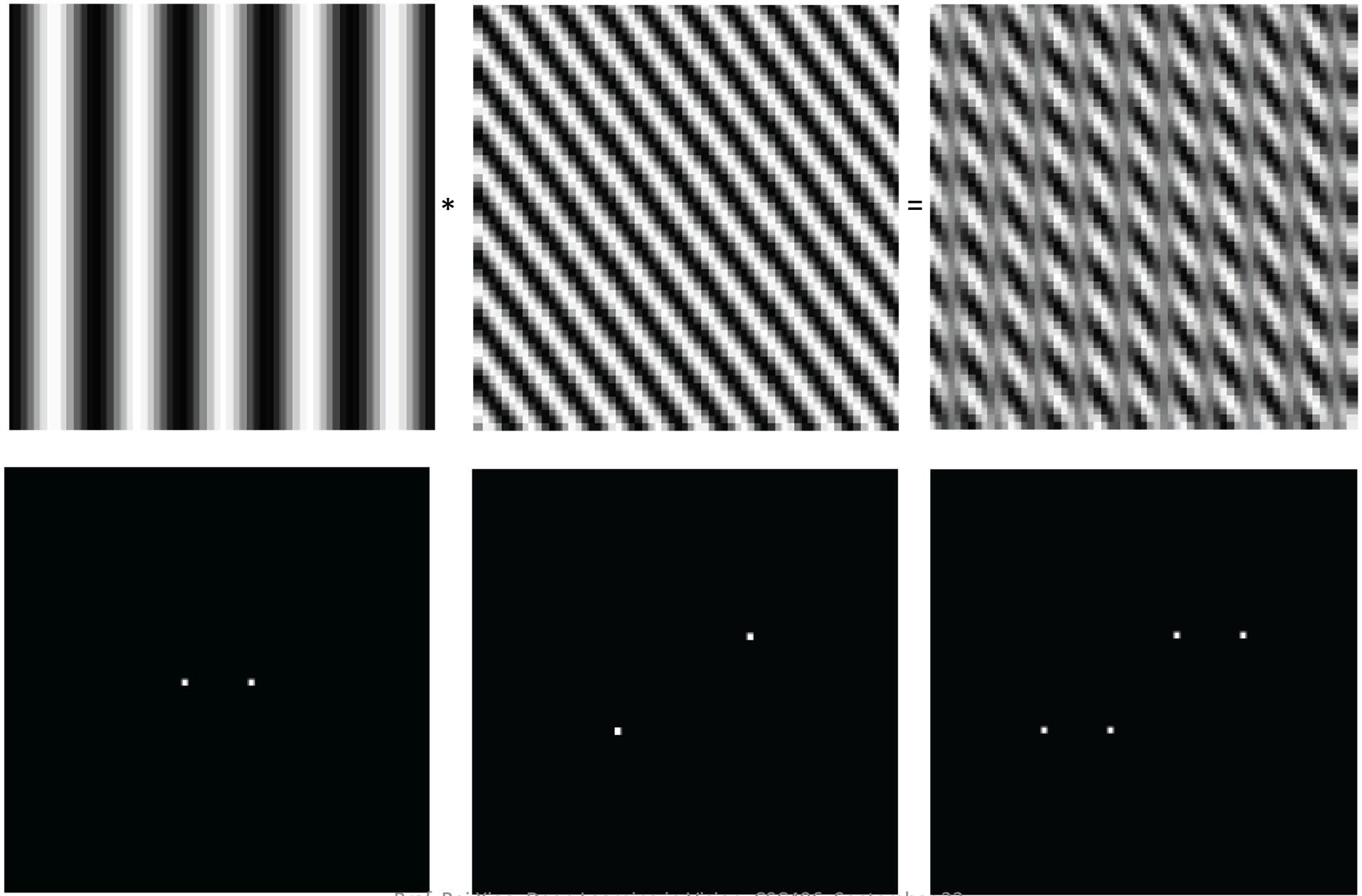
Product of images

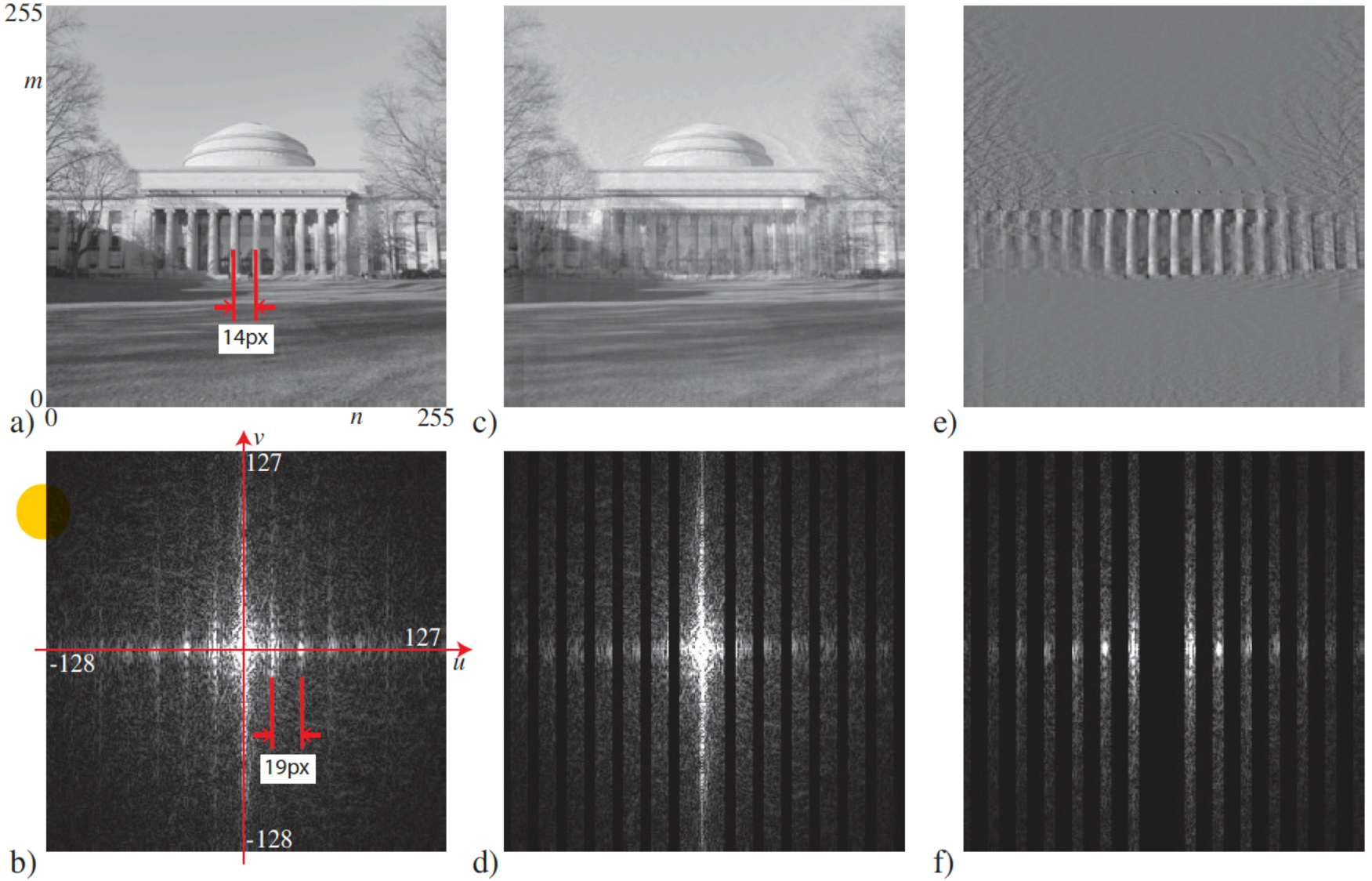
The Fourier transform of the product of two images

$$f[n,m] = g[n,m]\,h[n,m]$$

is the convolution of their DFTs:

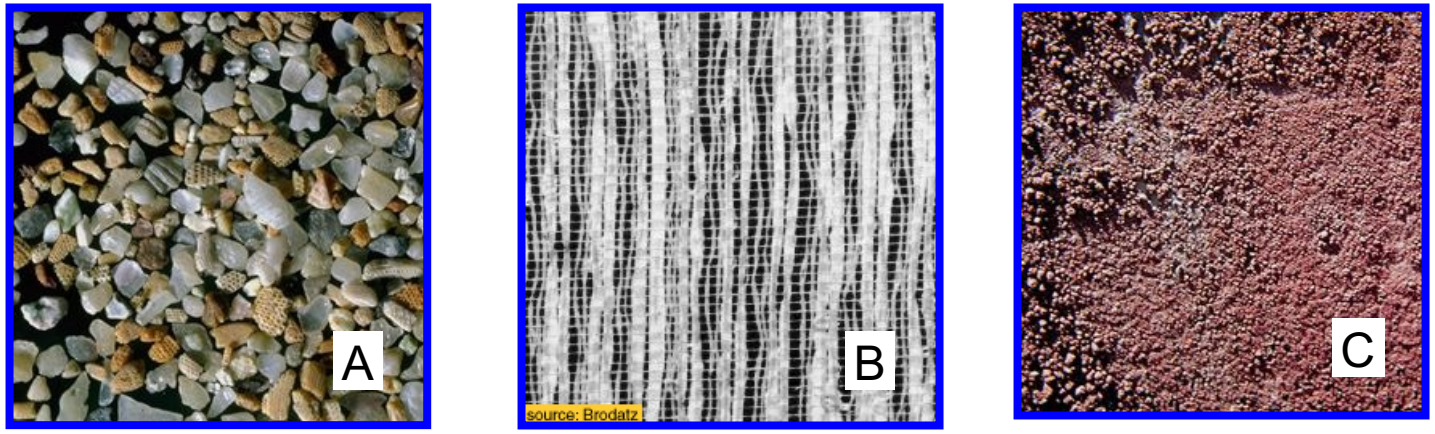
$$F[u,v] = \frac{1}{NM} G[u,v] \circ H[u,v]$$



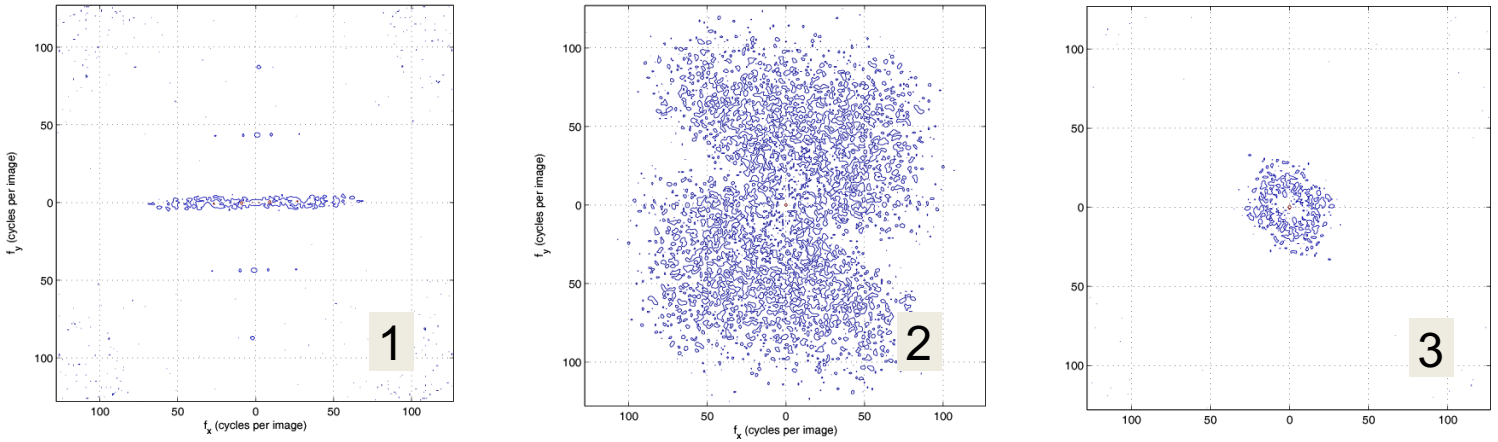

255
m
0
14px
0
n
255
a)
c)
e)
v
127
-128
127
u
19px
-128
b)
d)
f)

Game: find the right pairs

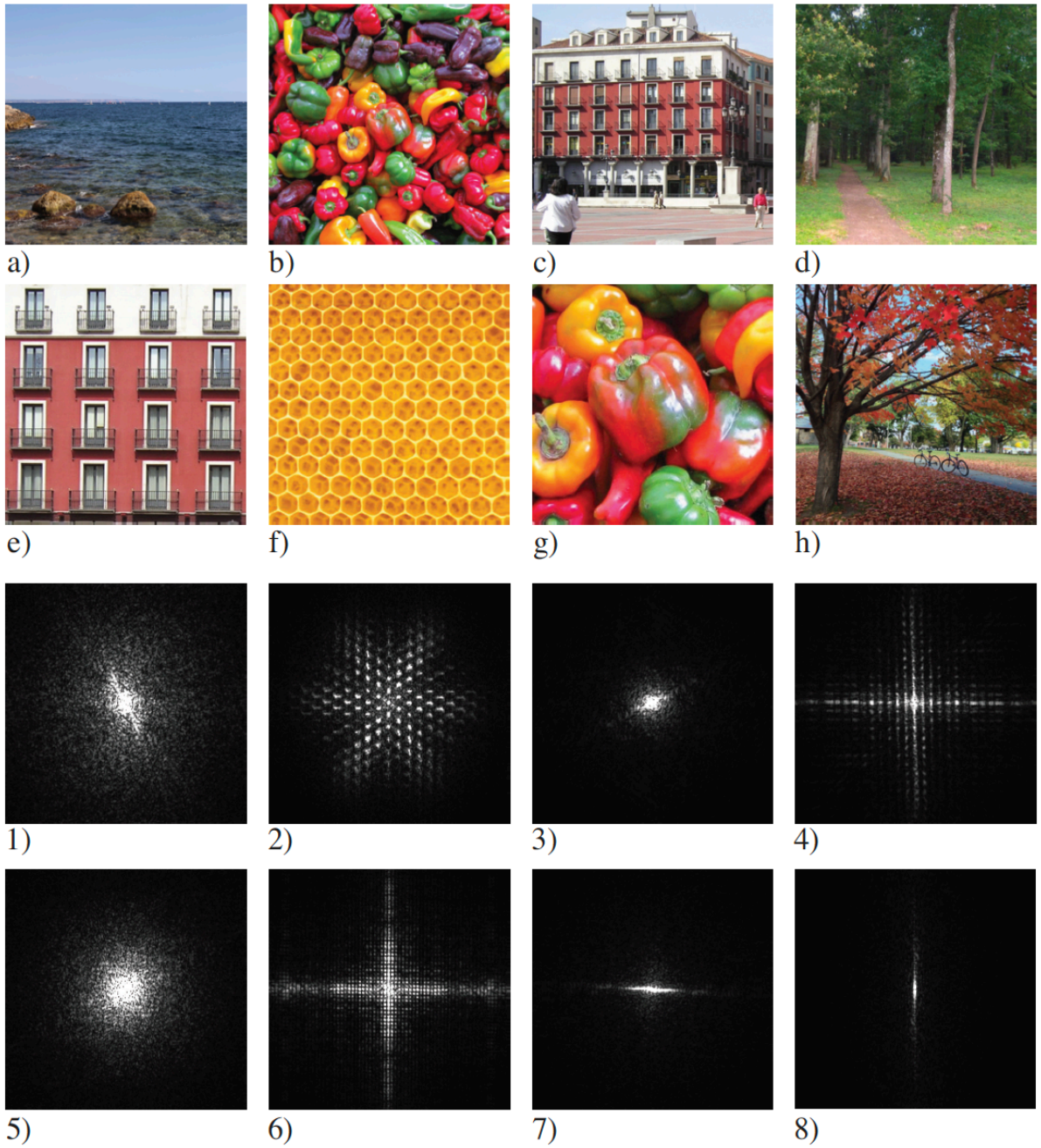
Images



DFT magnitude



fx(cycles/image pixel size) fx(cycles/image pixel size) fx(cycles/image pixel size)

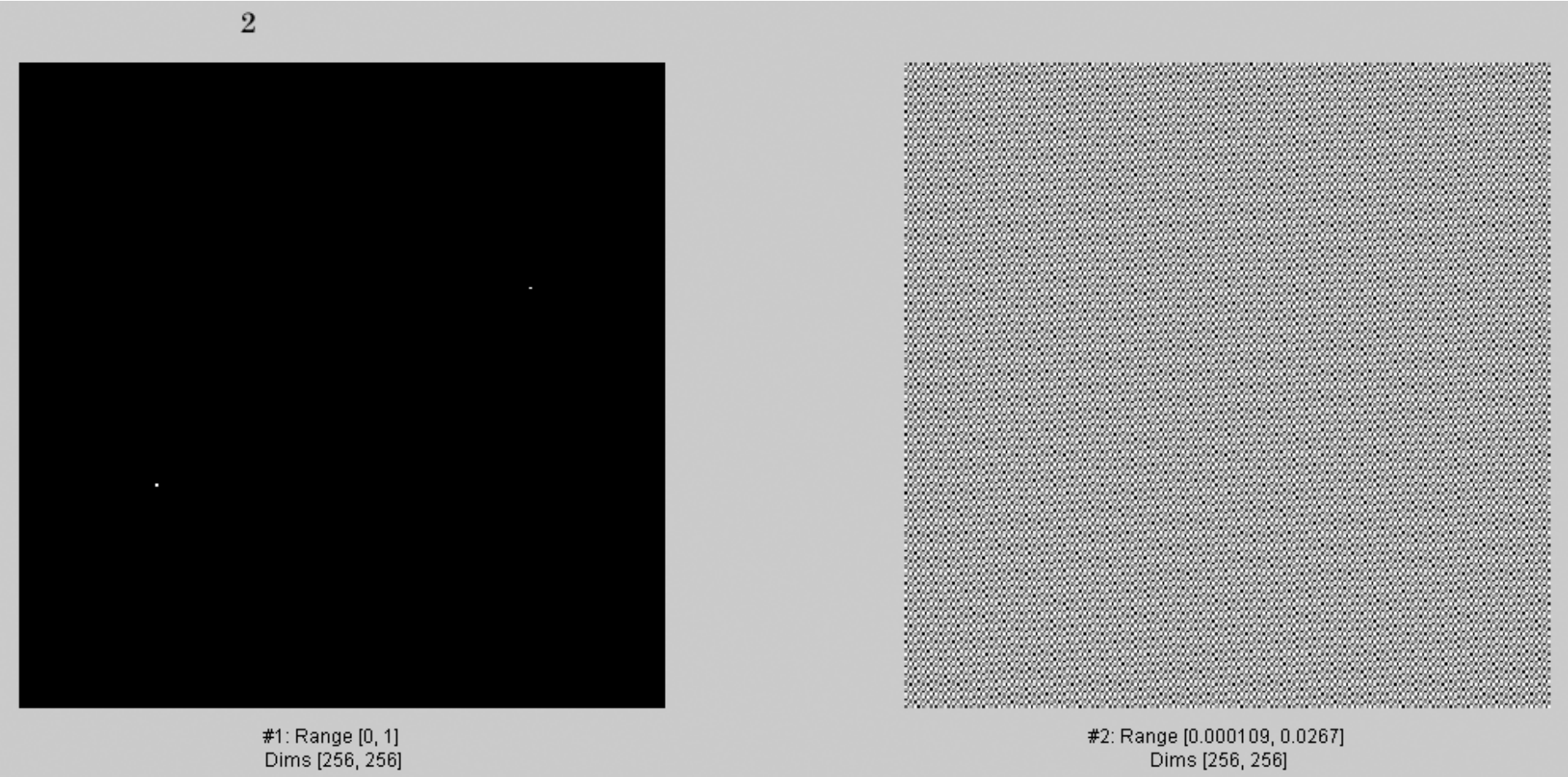

a)
b)
c)
d)
e)
f)
g)
h)
1)
2)
3)
4)
5)
6)
7)
8)

The inverse Discrete Fourier transform

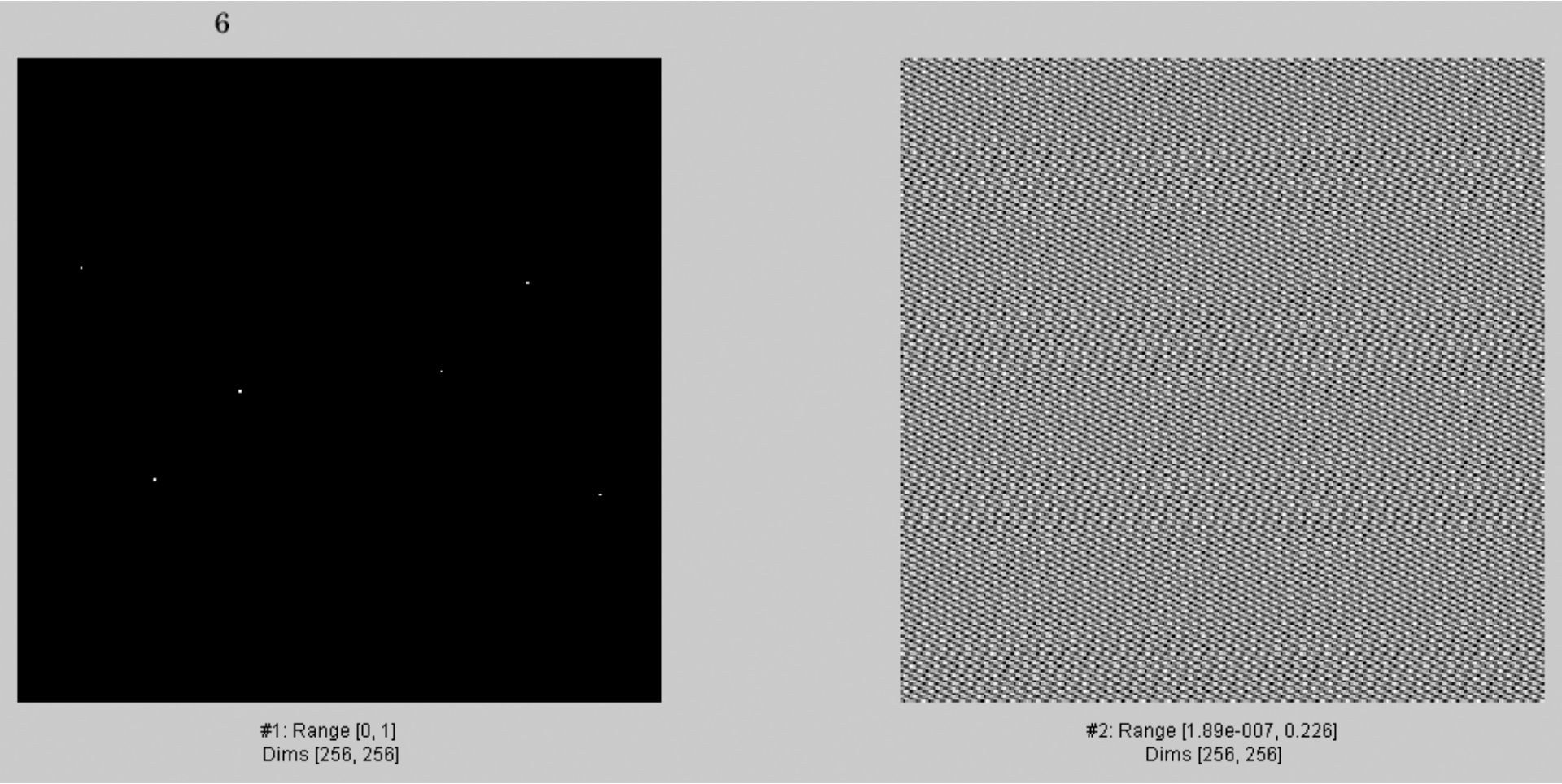
$$f[n,m] = \frac{1}{NM} \sum_{u=0}^{N-1} \sum_{v=0}^{M-1} F[u,v] \exp\left(+2\pi j \left(\frac{u\,n}{N} + \frac{v\,m}{M}\right)\right)$$

How does summing waves ends up giving back a picture?

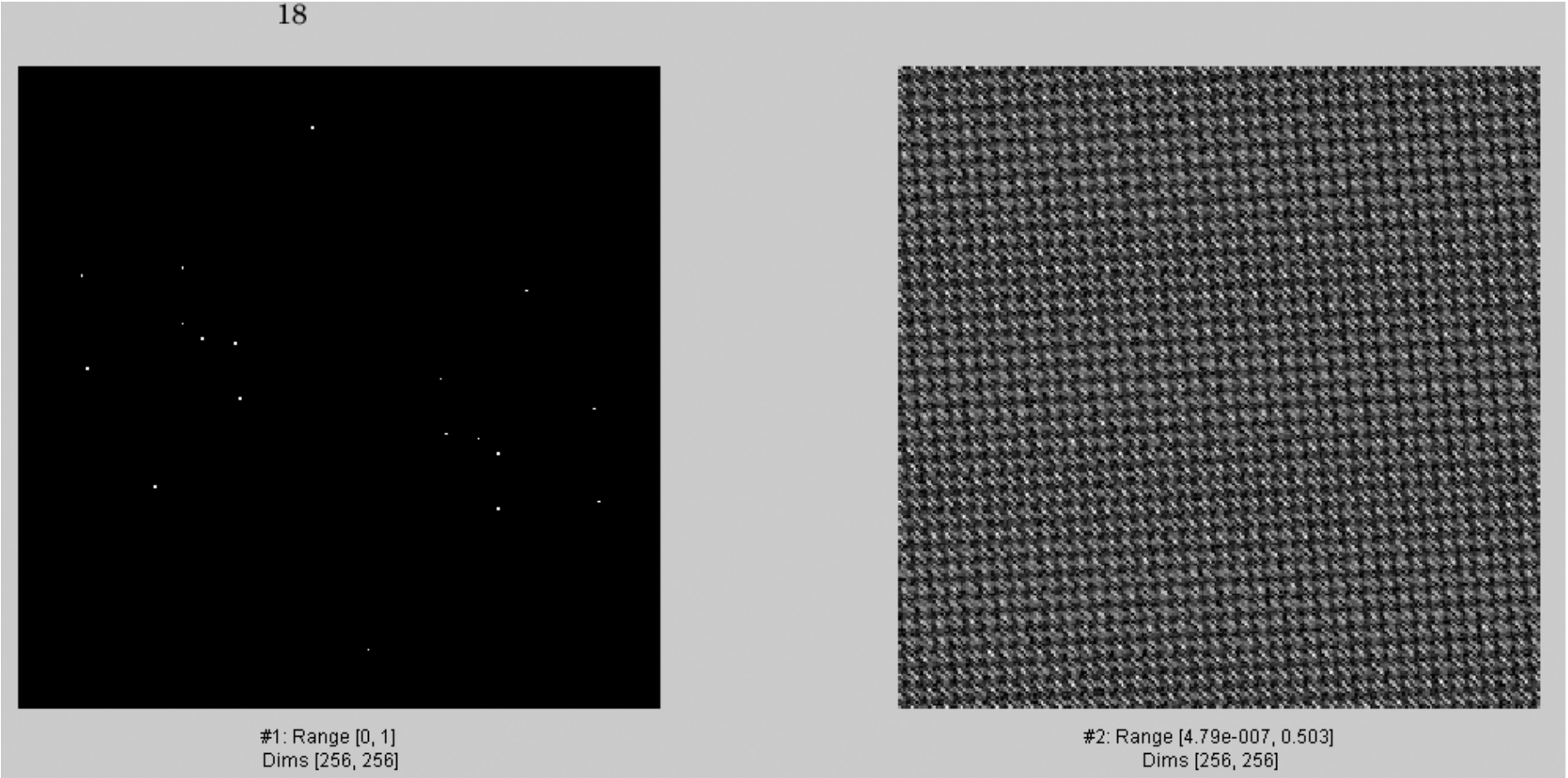
2



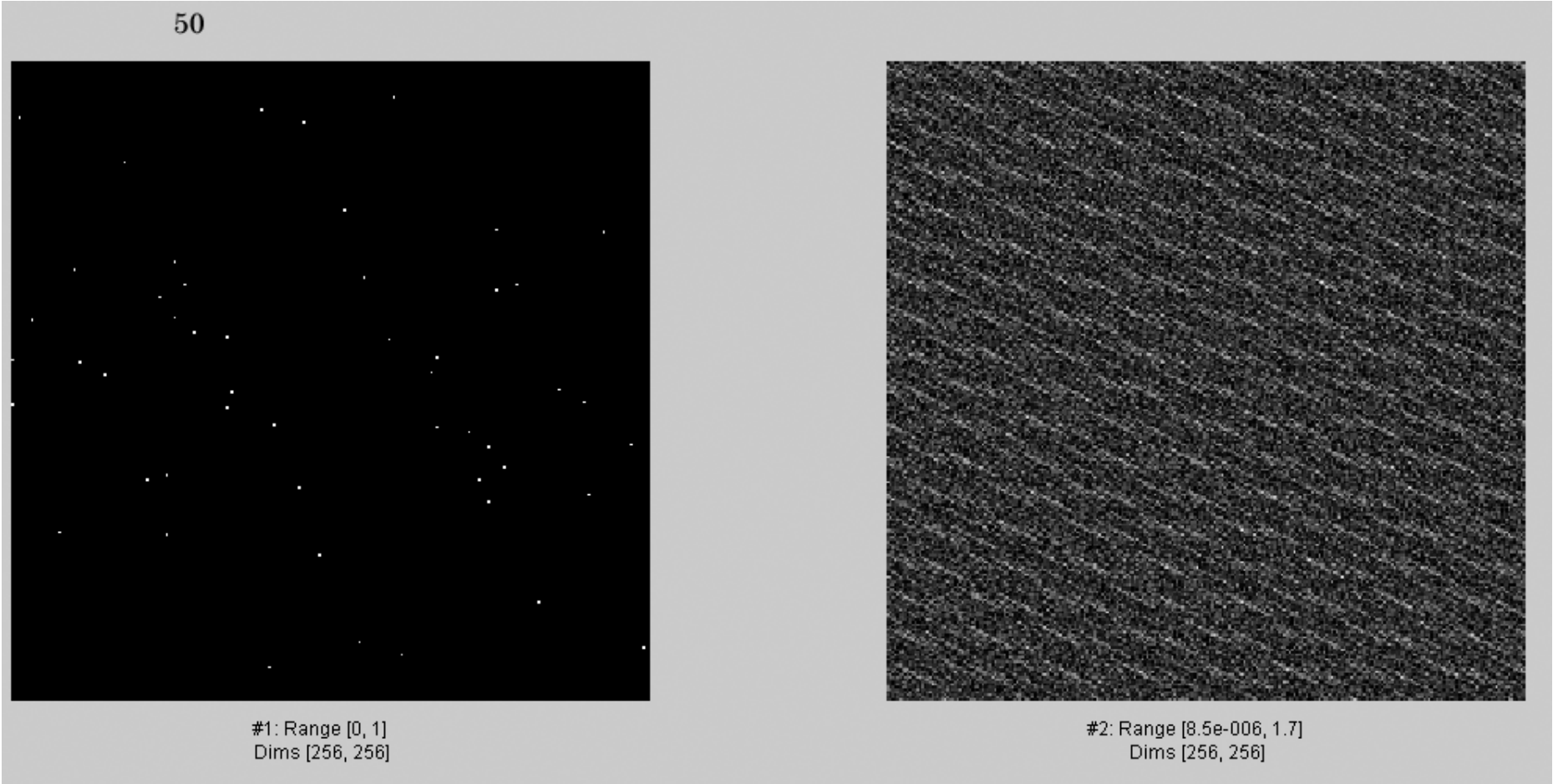
6



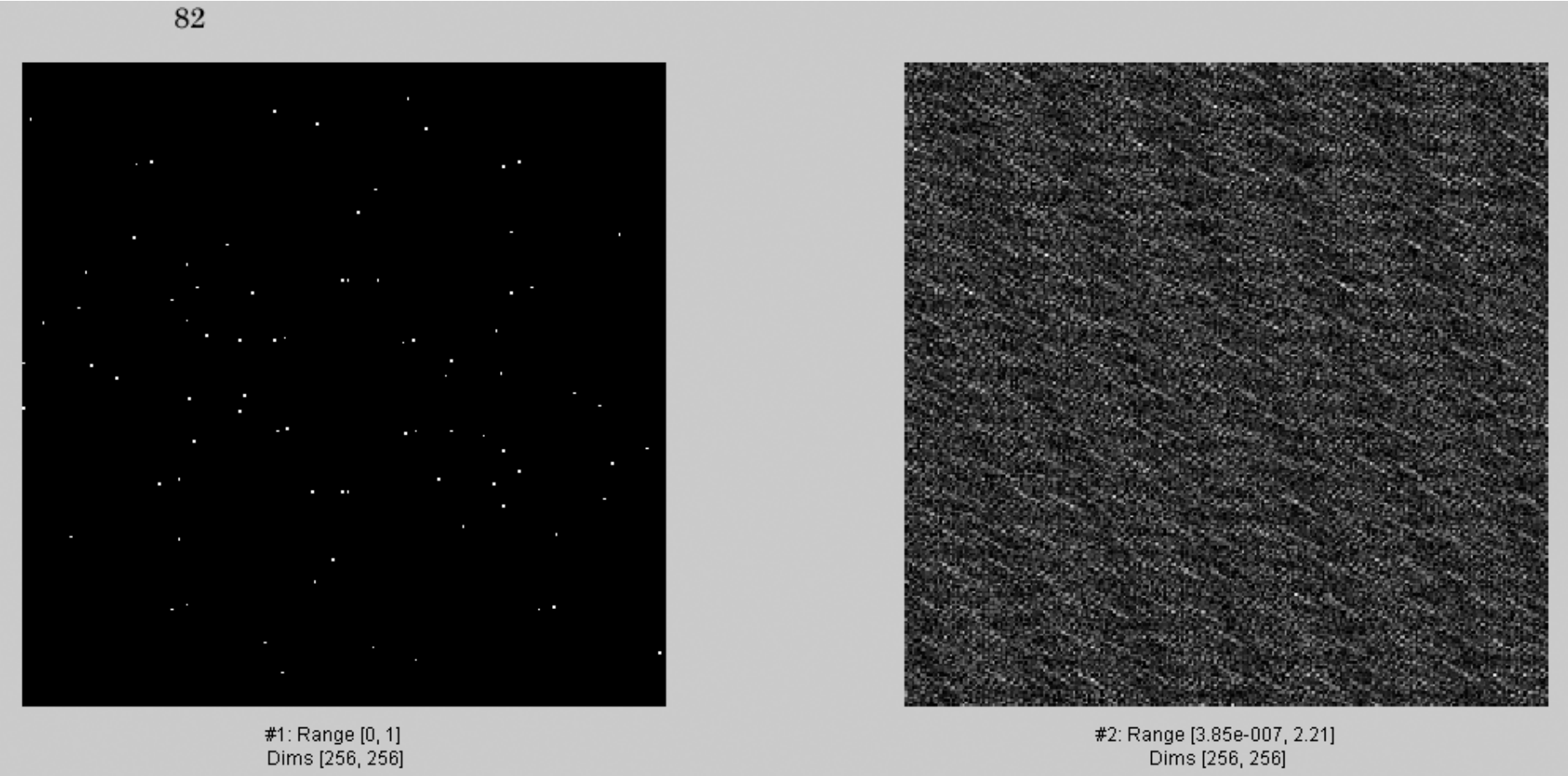
18



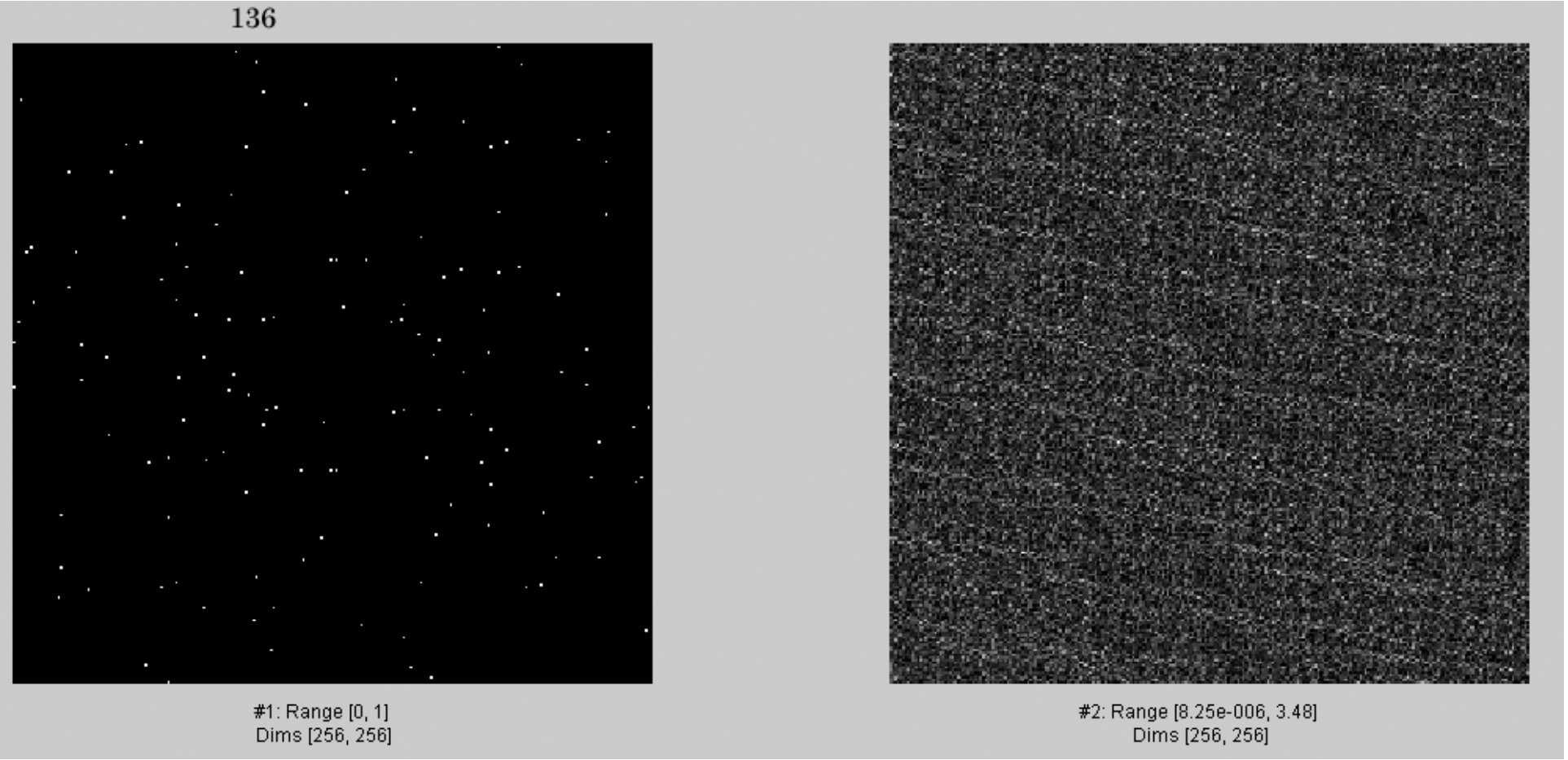
50



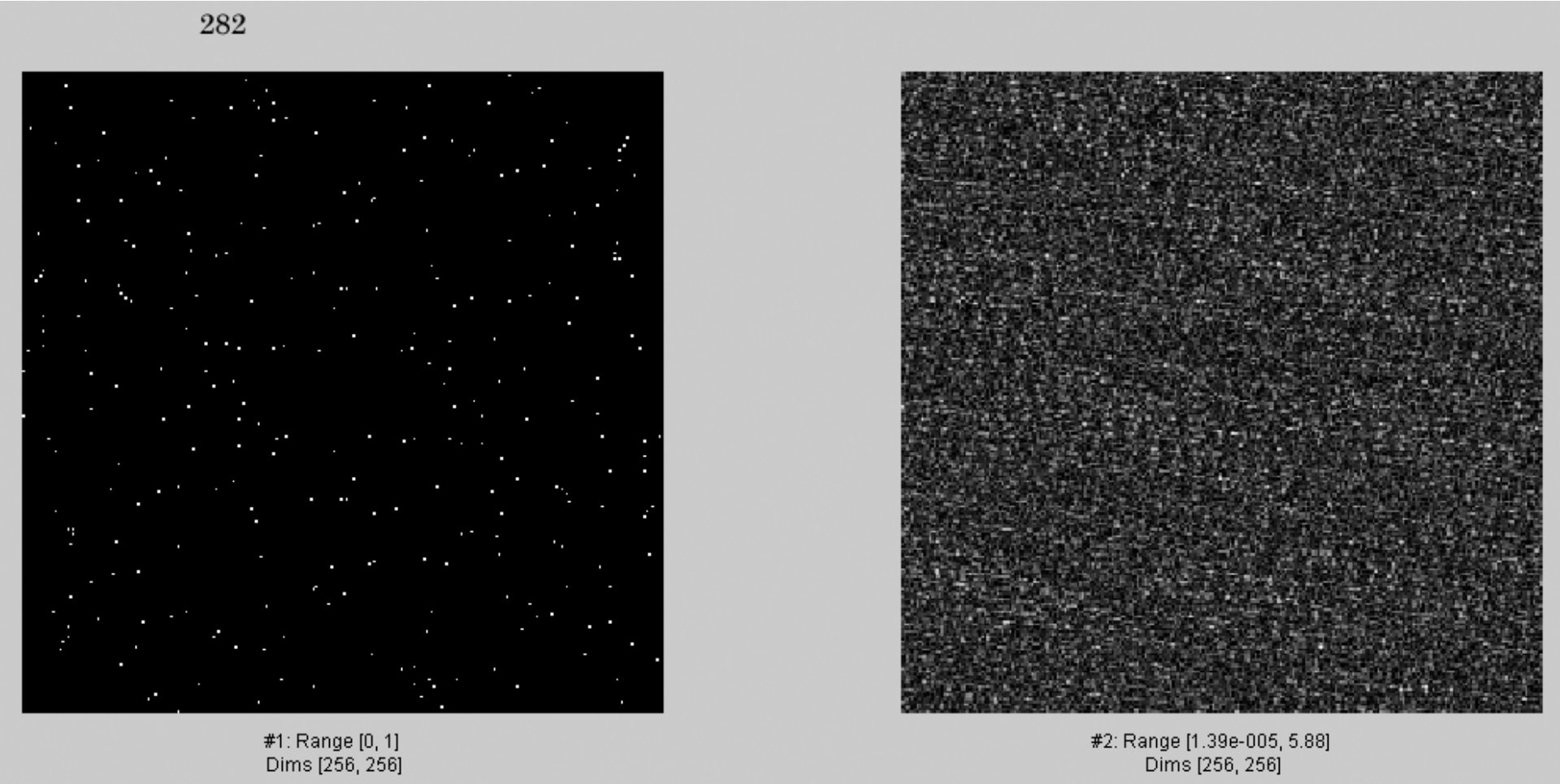
82



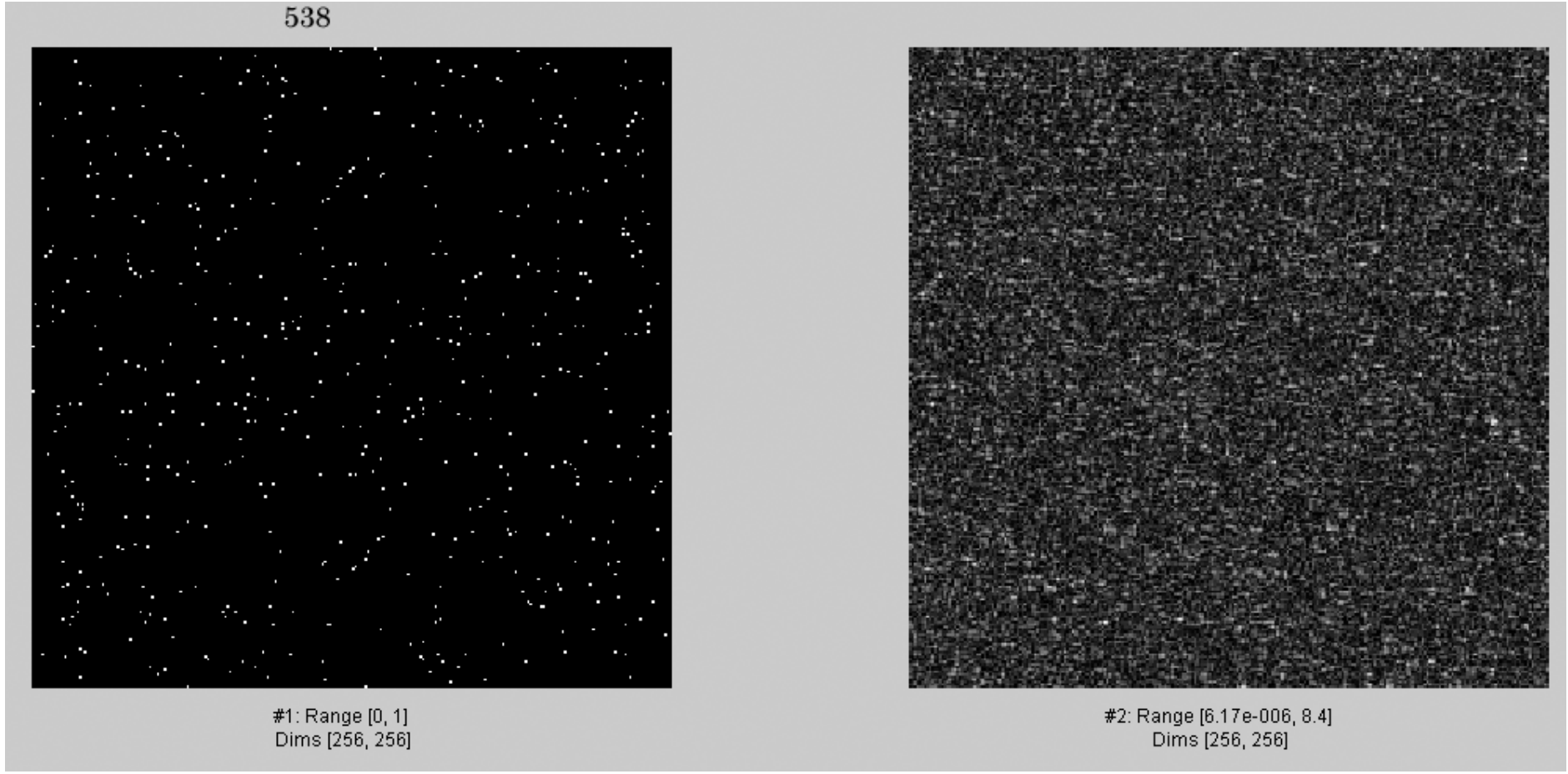
136



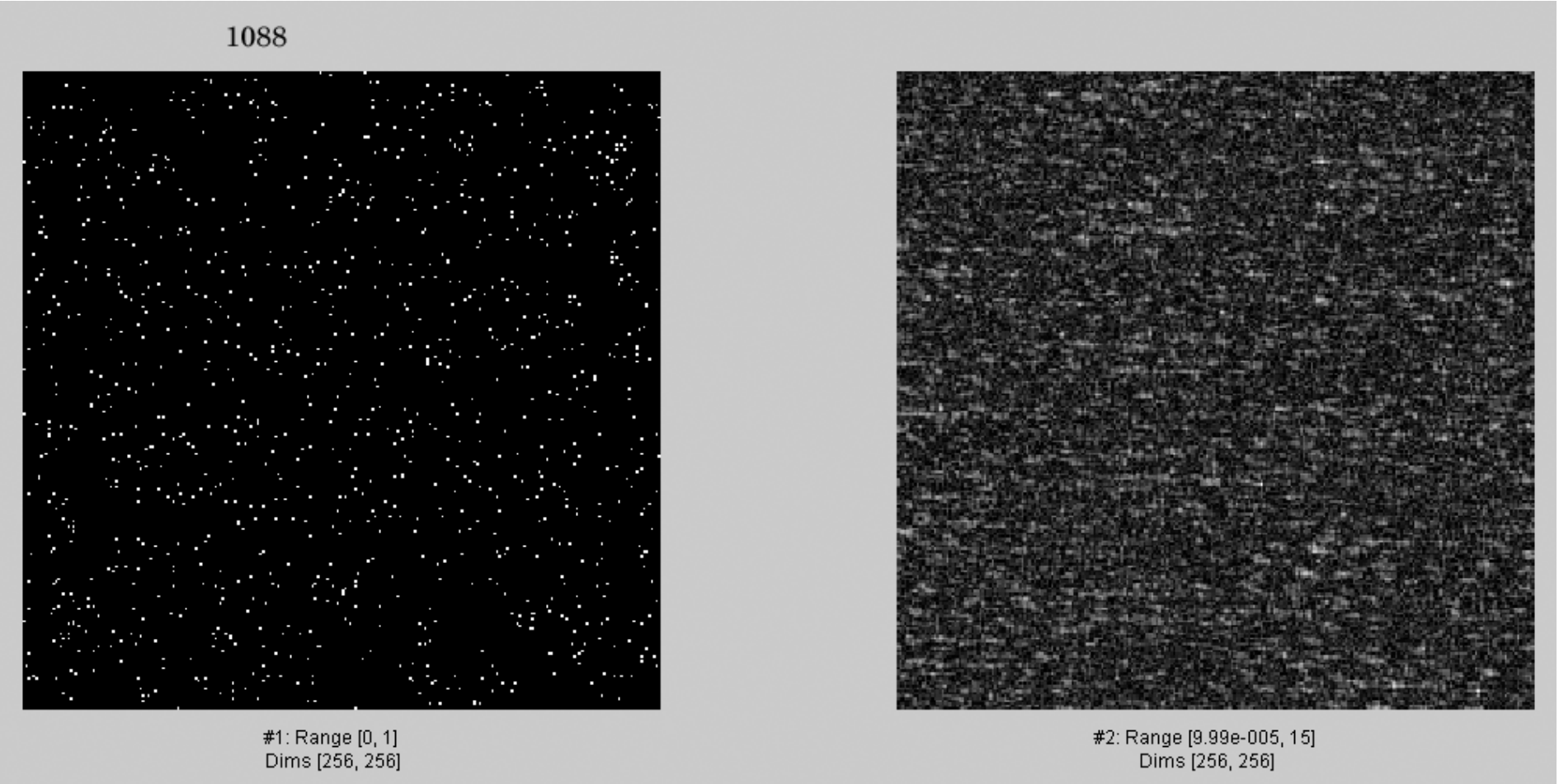
282



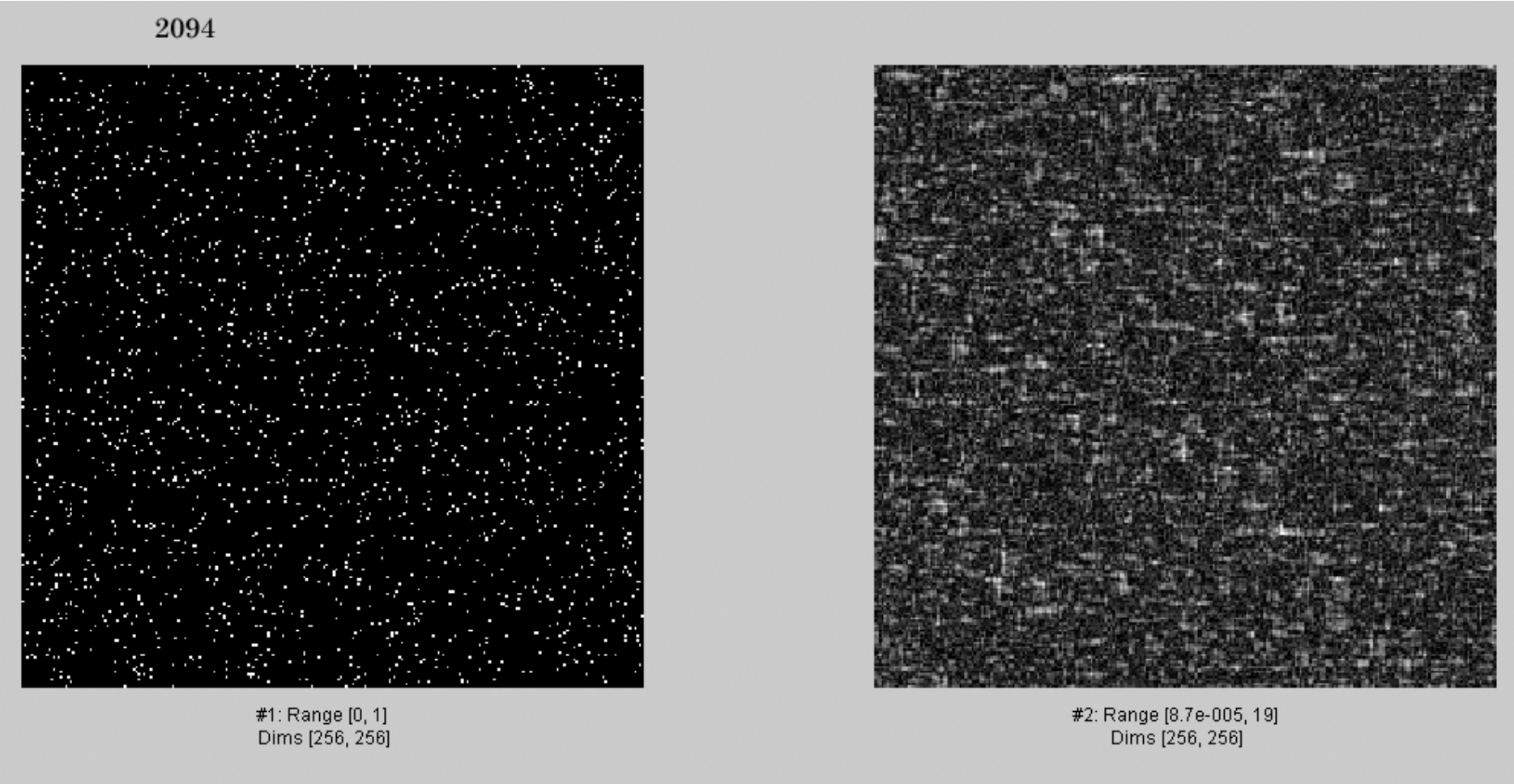
538



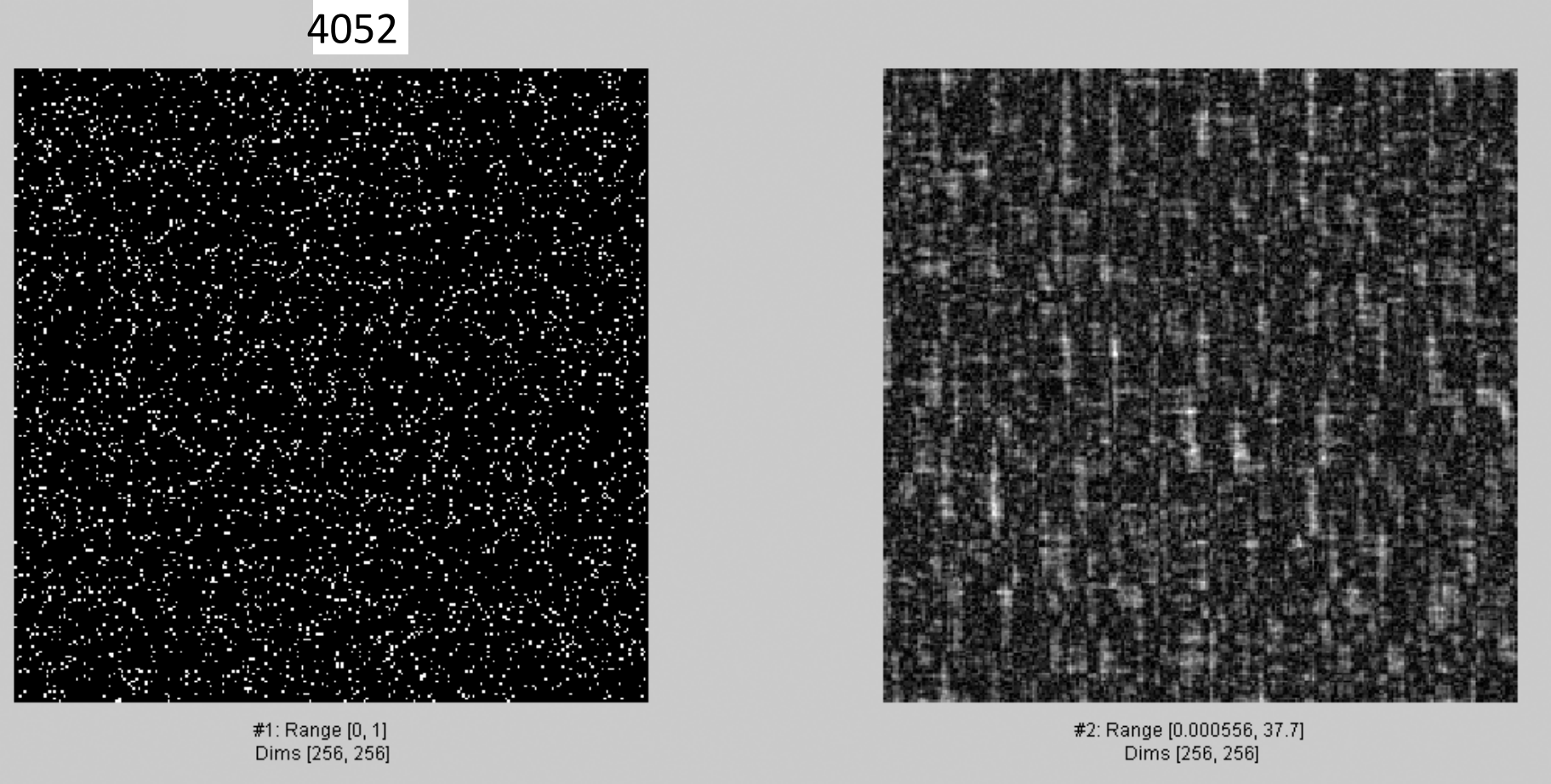
1088



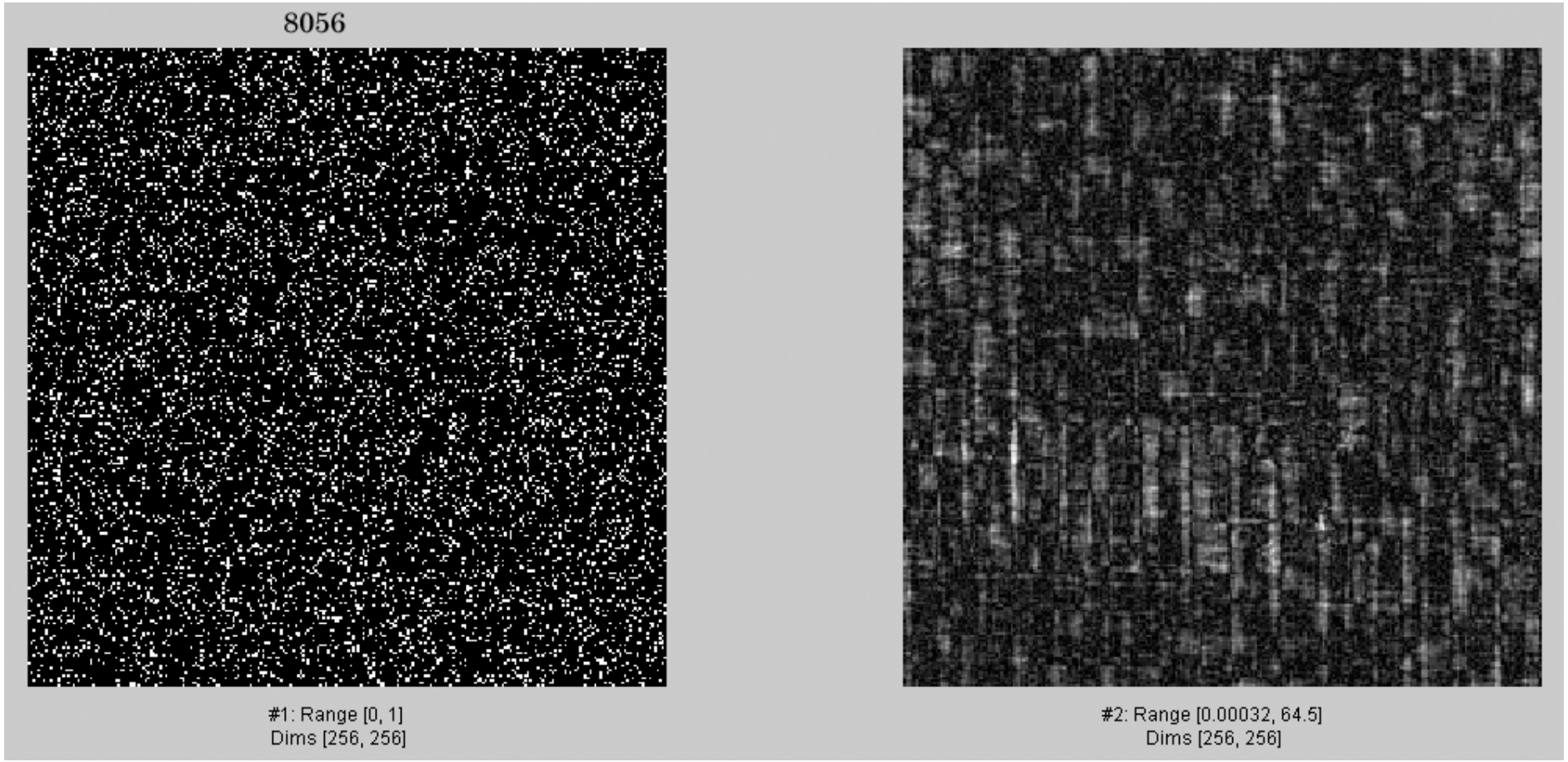
2094



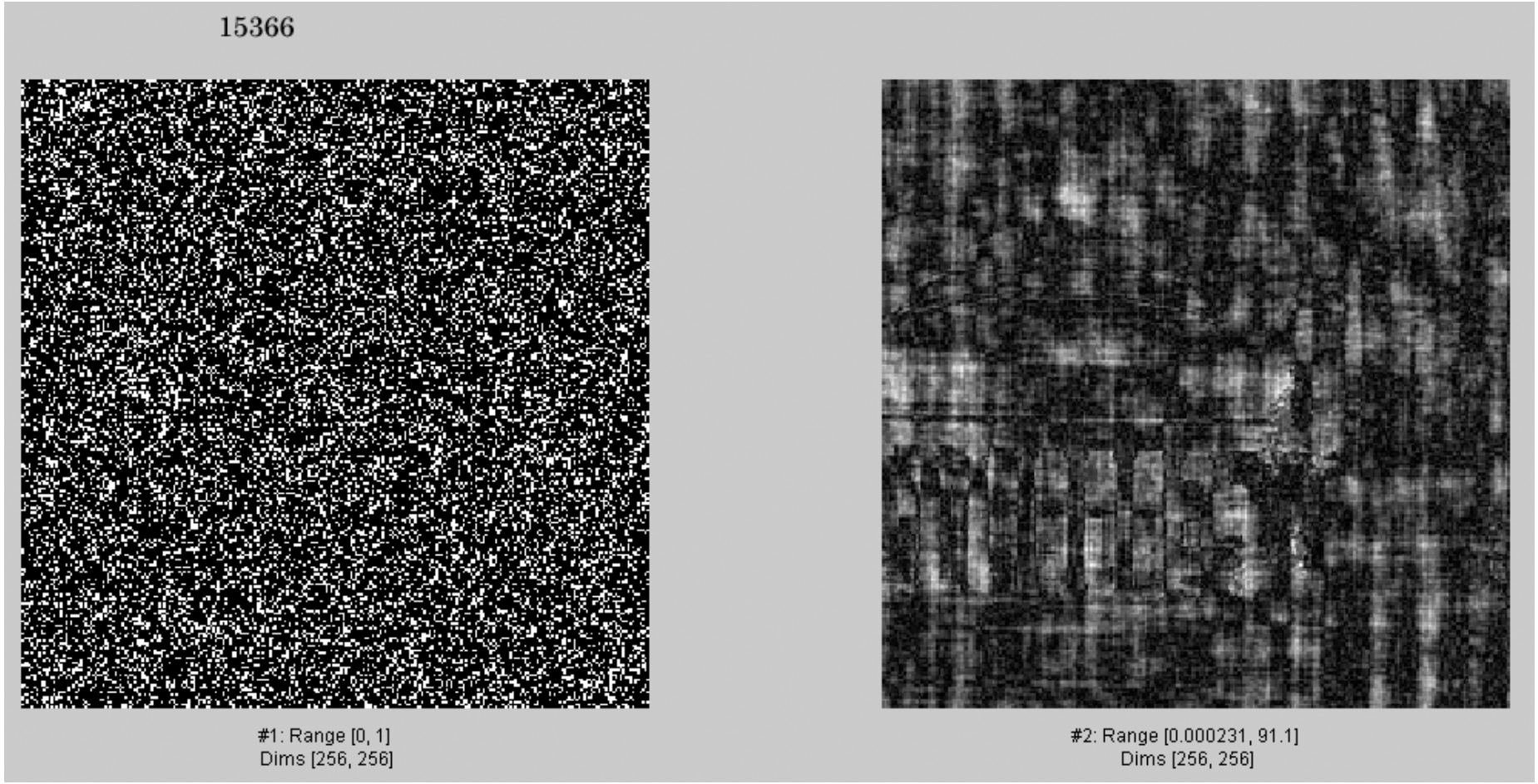
4052.



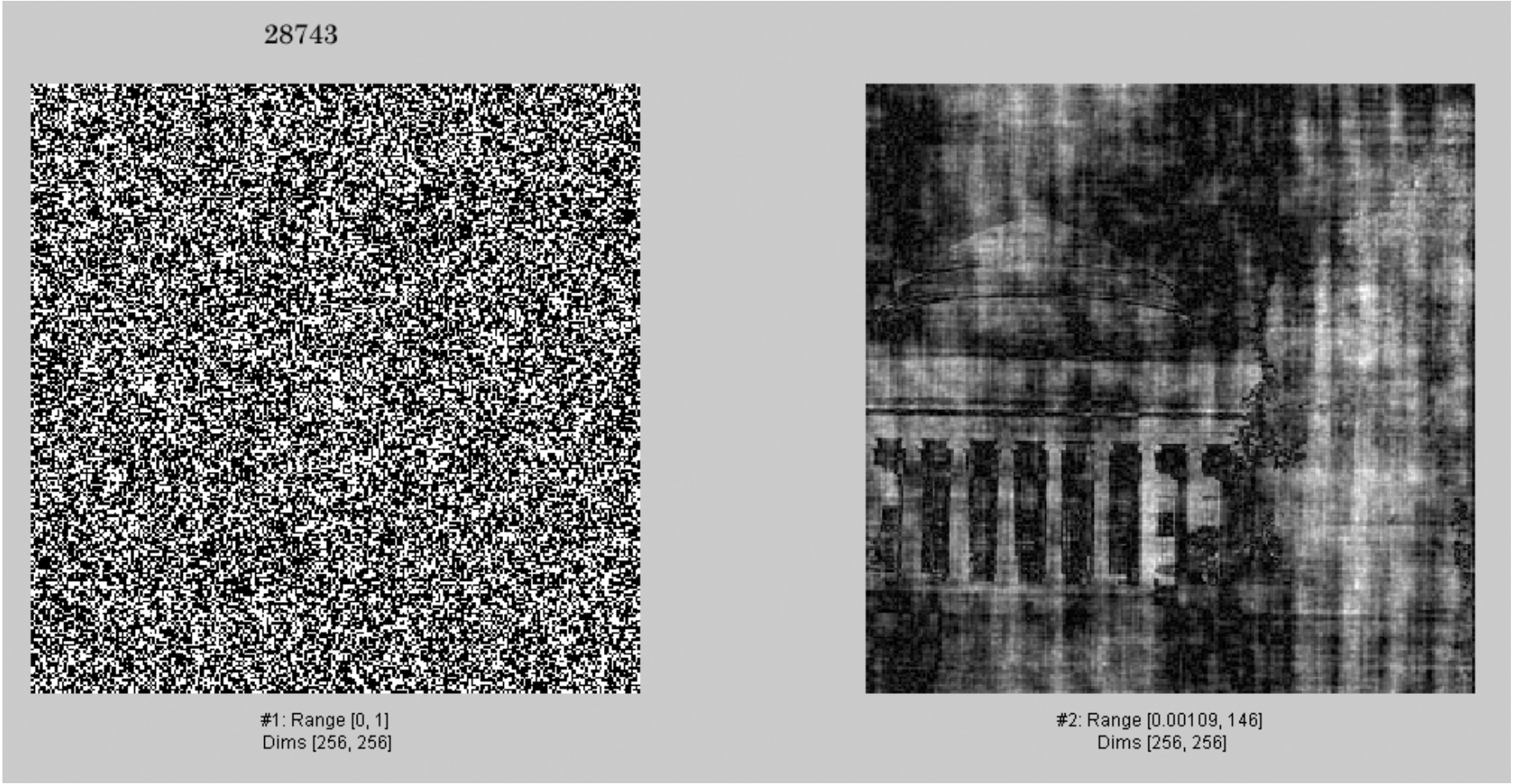
8056.



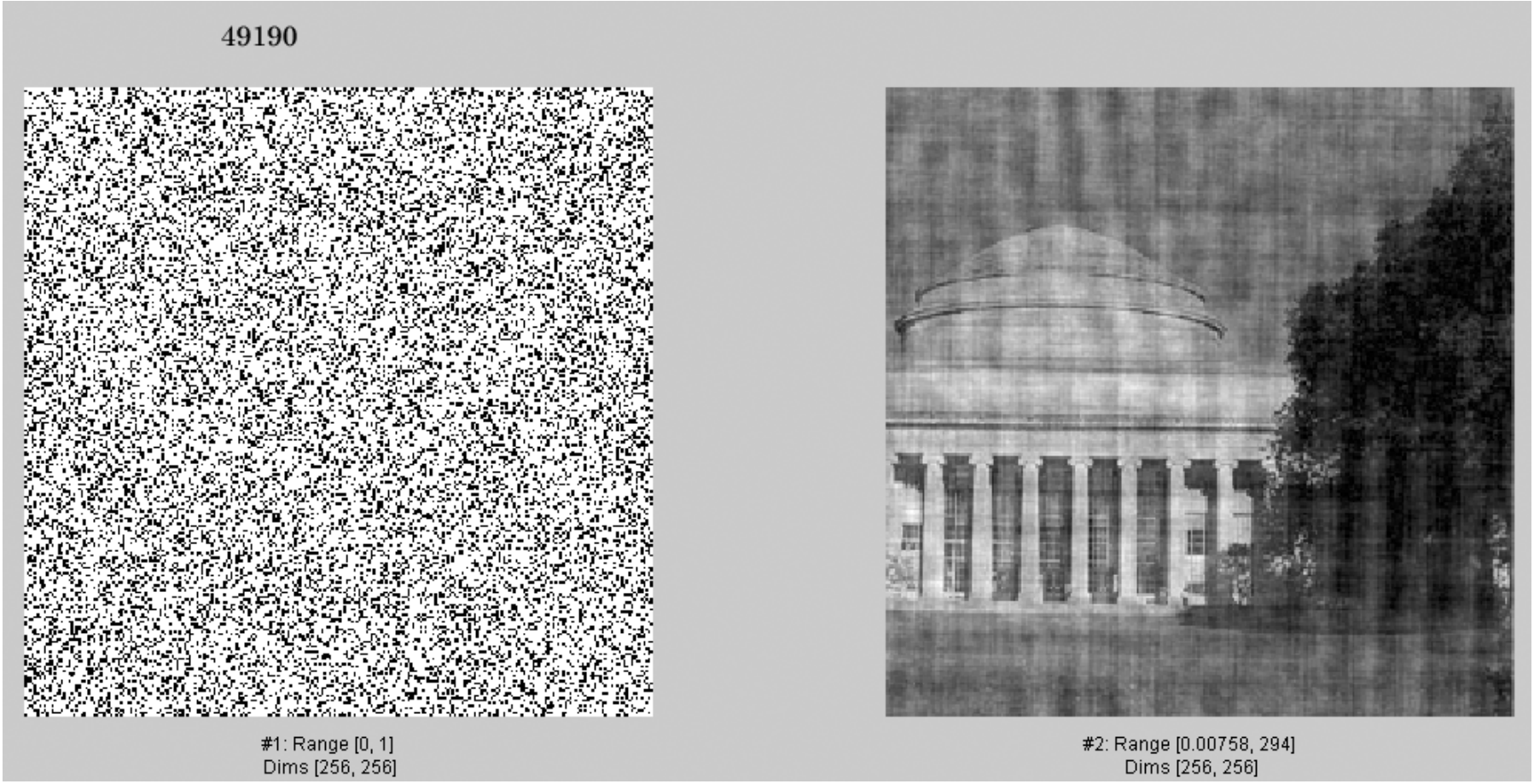
15366



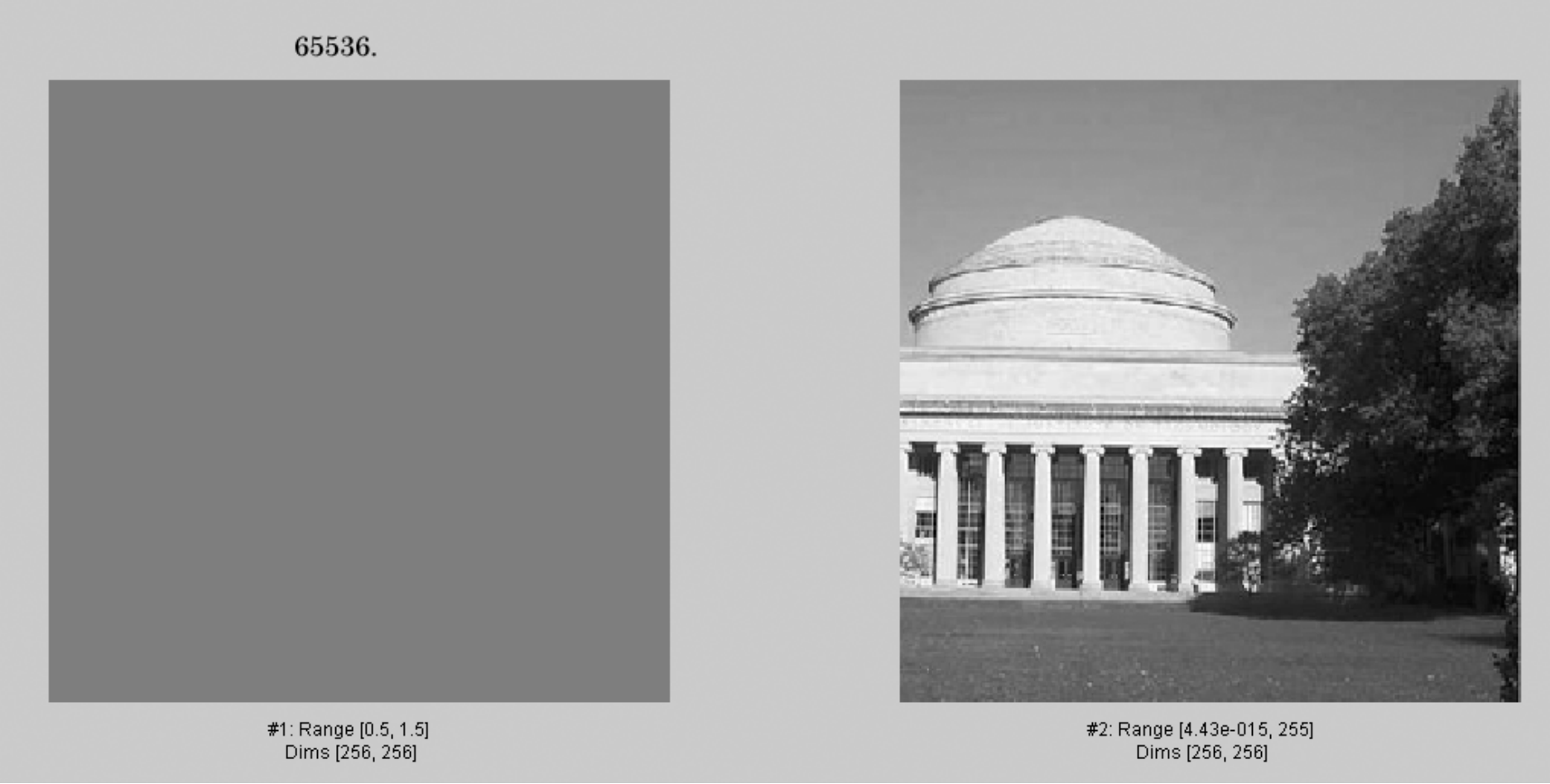
28743



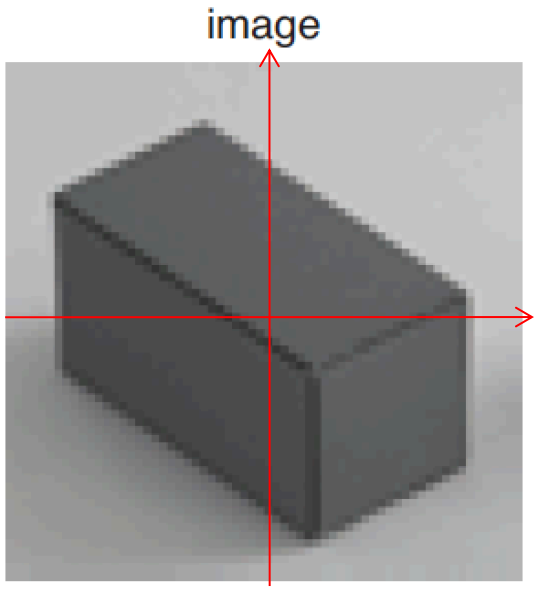
49190.

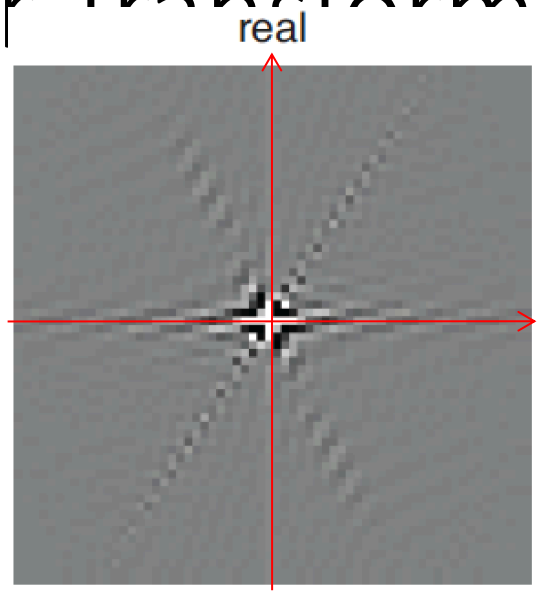


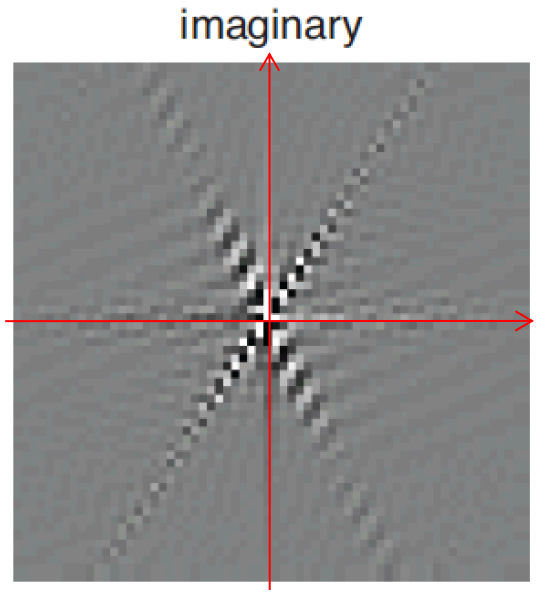
65536.

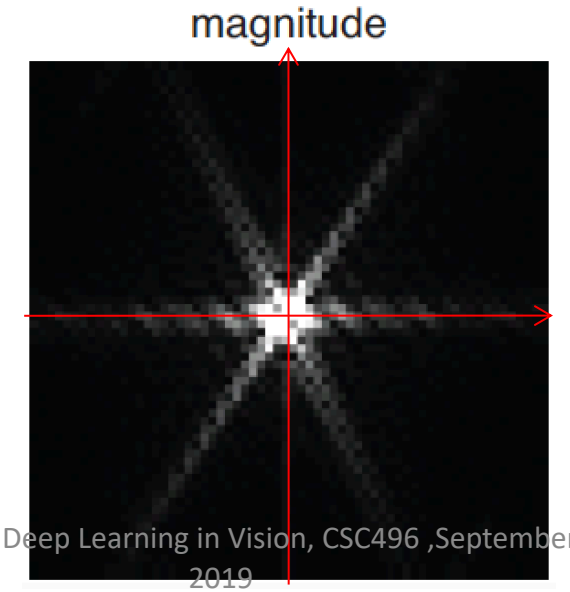

65536.
#1: Range [0.5, 1.5]
Dims [256, 256]
#2: Range [4.43e-015, 255]
Dims [256, 256]

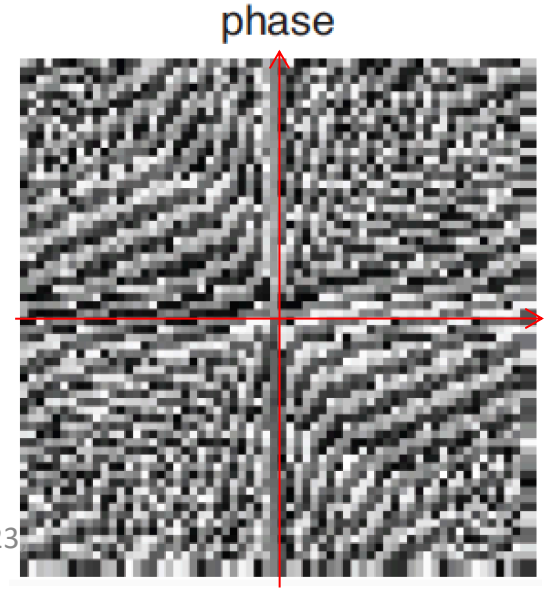
2D Discrete Fourier Transform









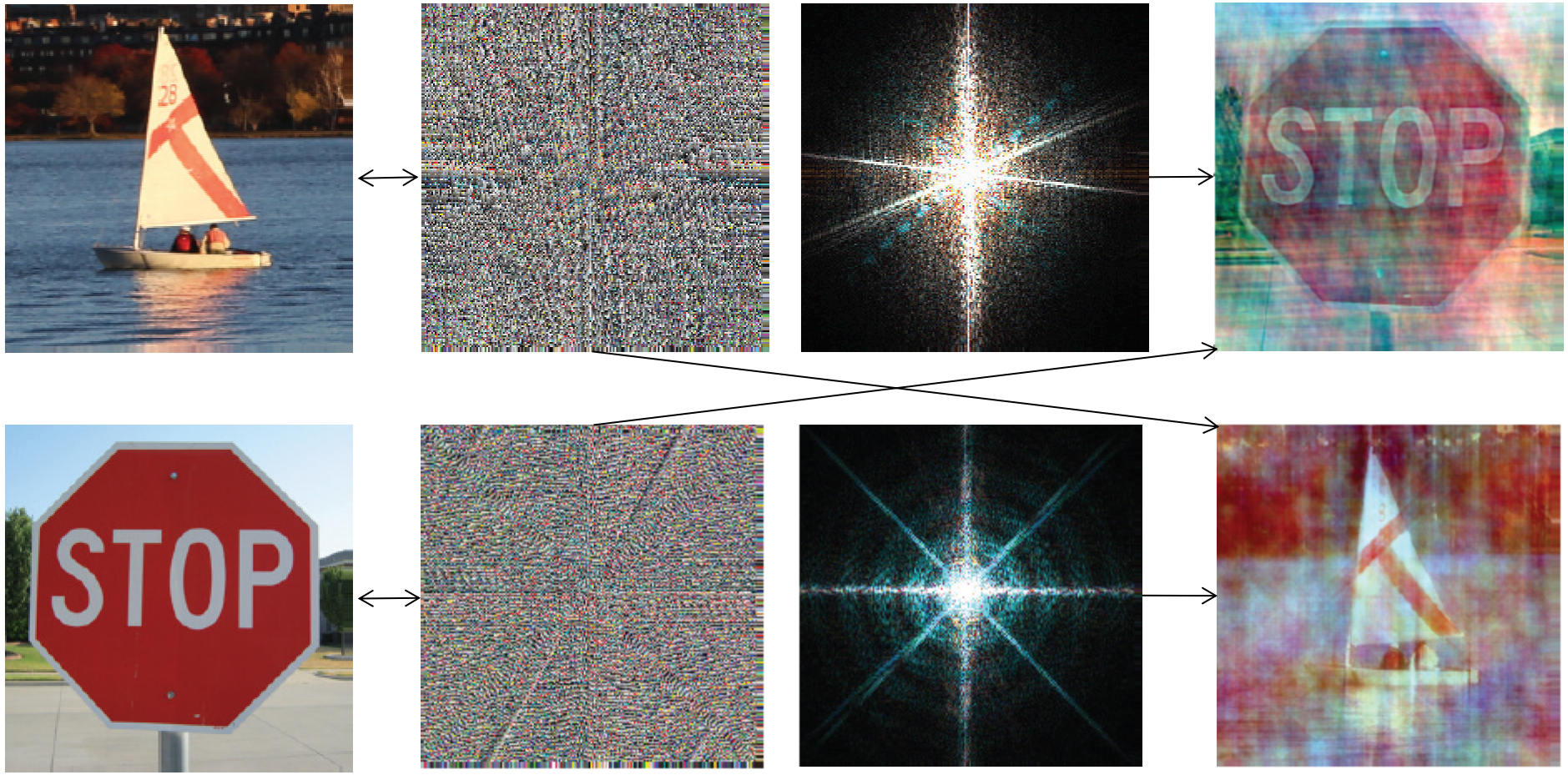


Phase and Magnitude

- Curious fact
  - all natural images have about the same magnitude transform
  - hence, phase seems to matter, but magnitude largely doesn't
- Demonstration
  - Take two pictures, swap the phase transforms, compute the inverse - what does the result look like?

Phase and Magnitude

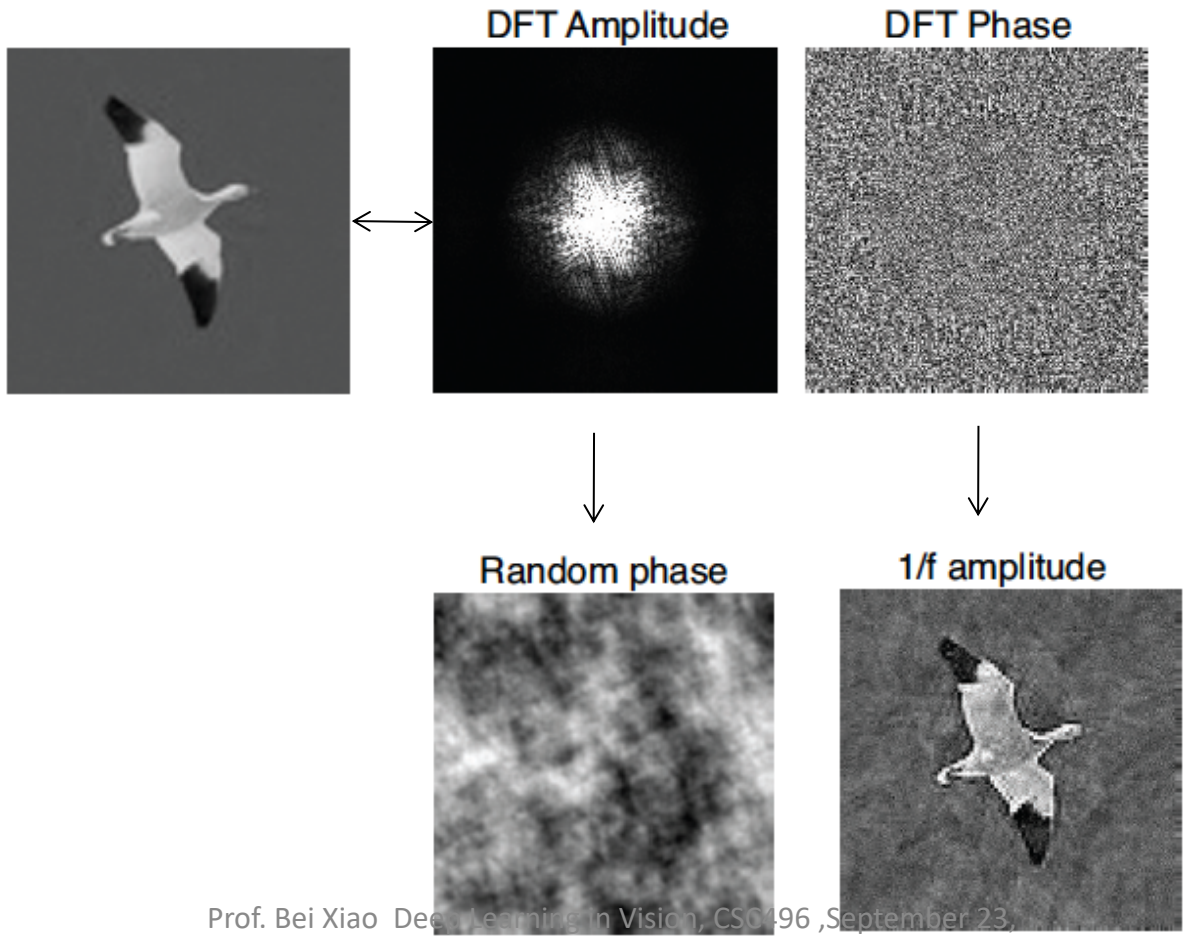
$$F[u,v] = A[u,v]\exp(j\theta[u,v])$$



Each color channel is processed in the same way.

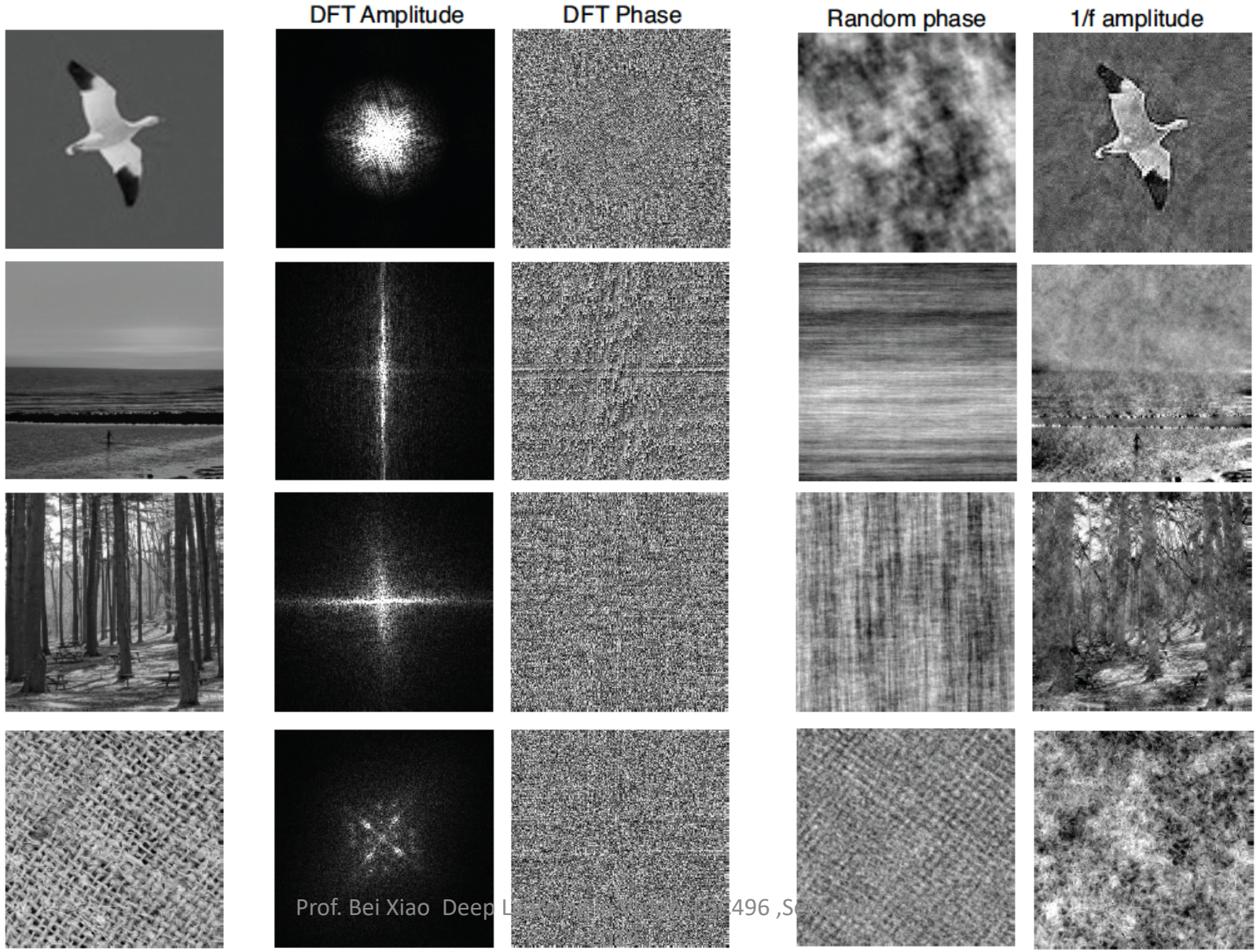
Phase and magnitude

$$F[u,v] = A[u,v]\exp(j\theta[u,v])$$

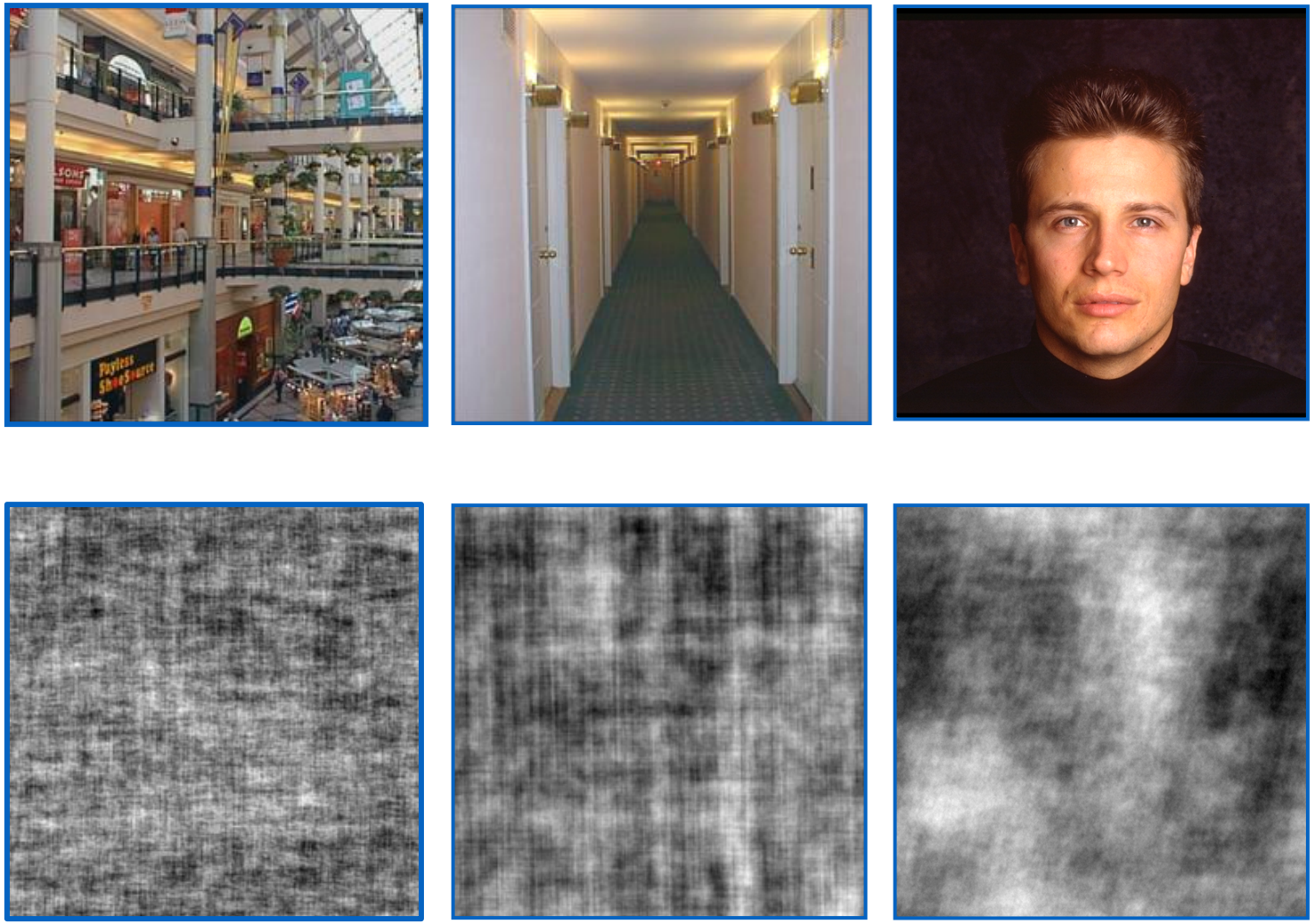


Using random amplitude does not look good.

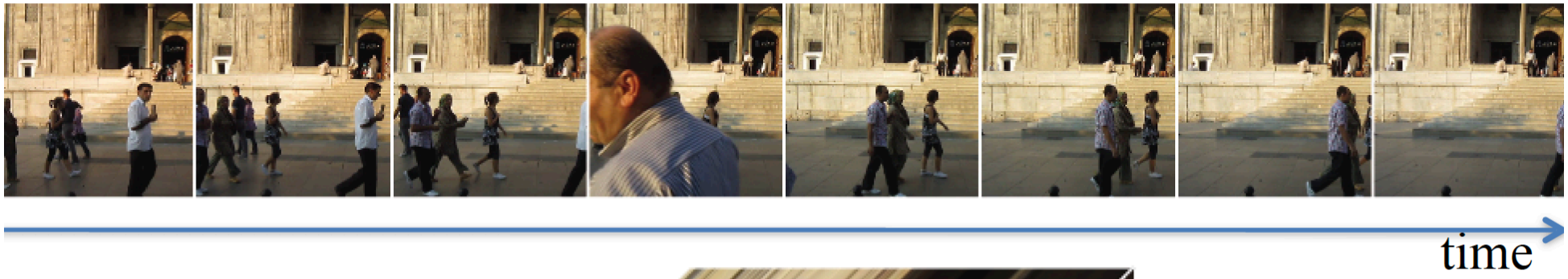
Does phase always win?

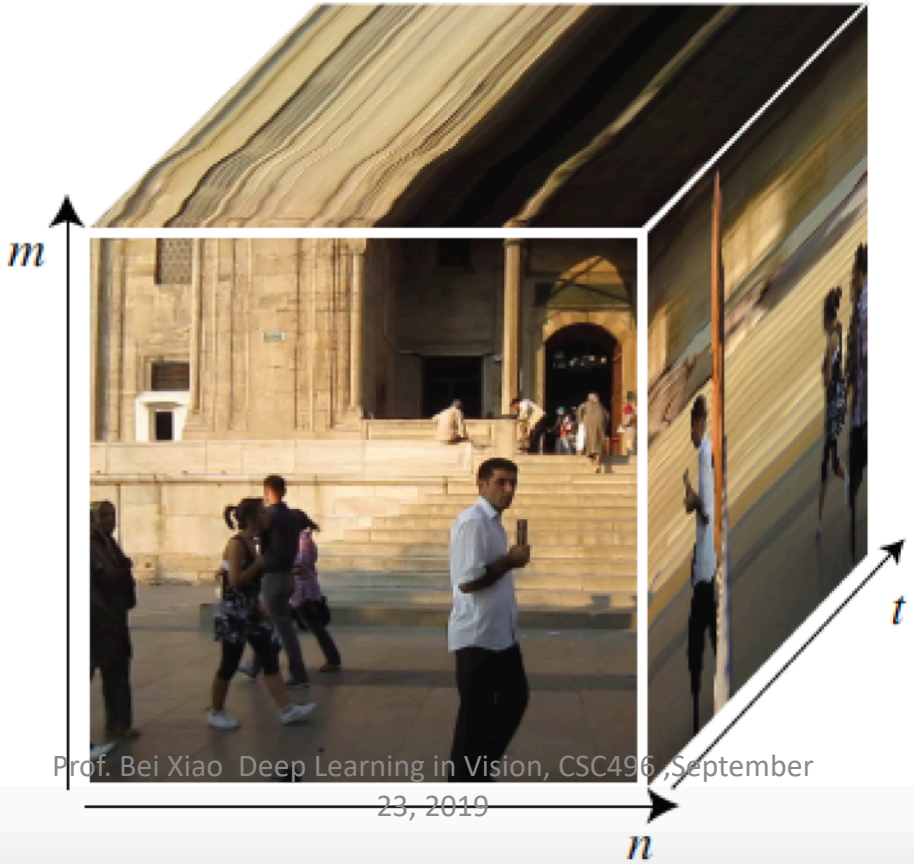


Randomizing the phase

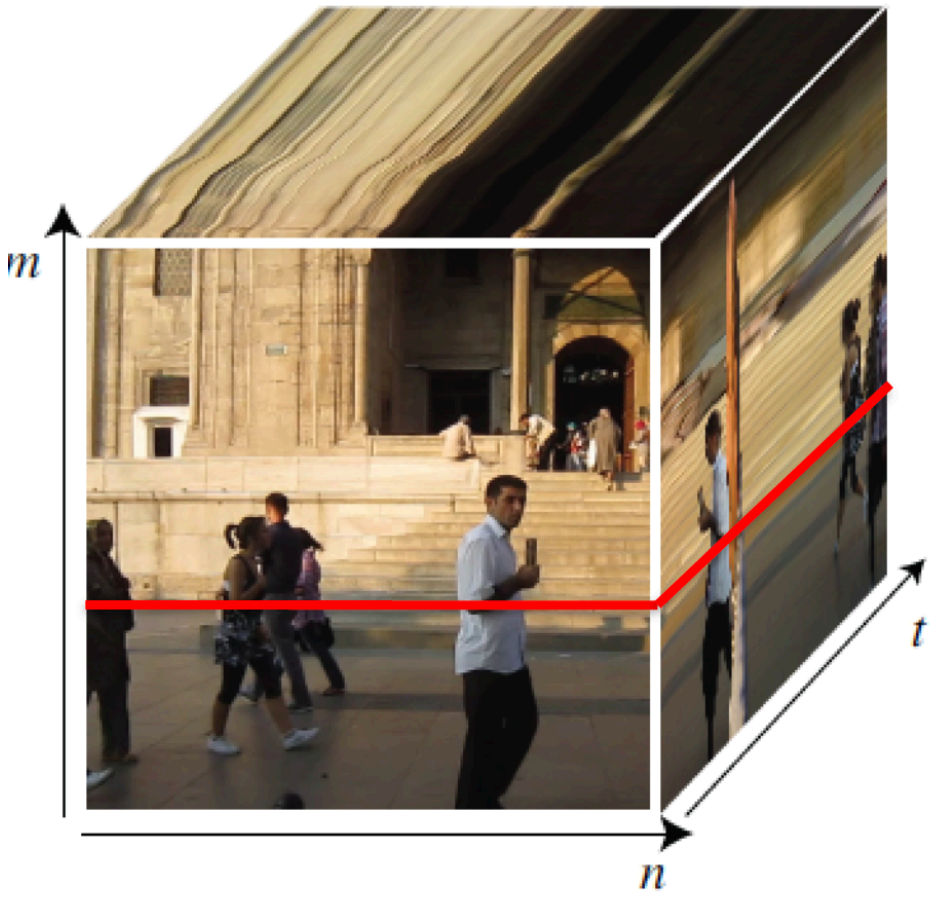


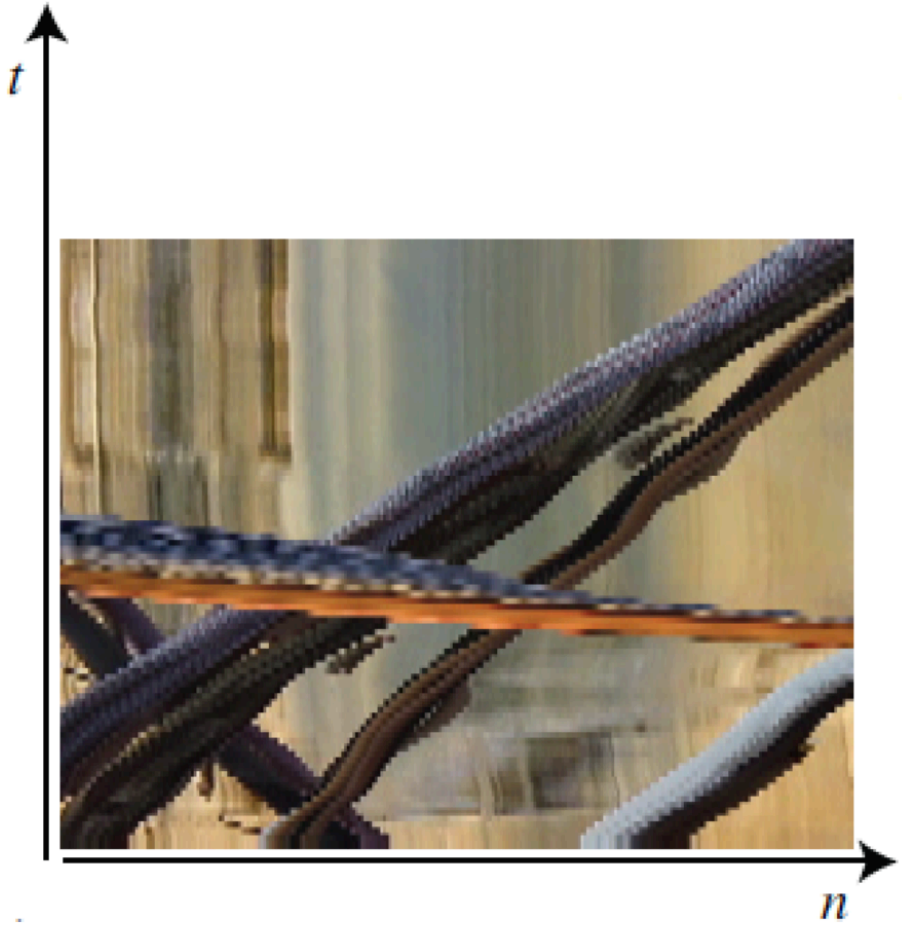
Sequences





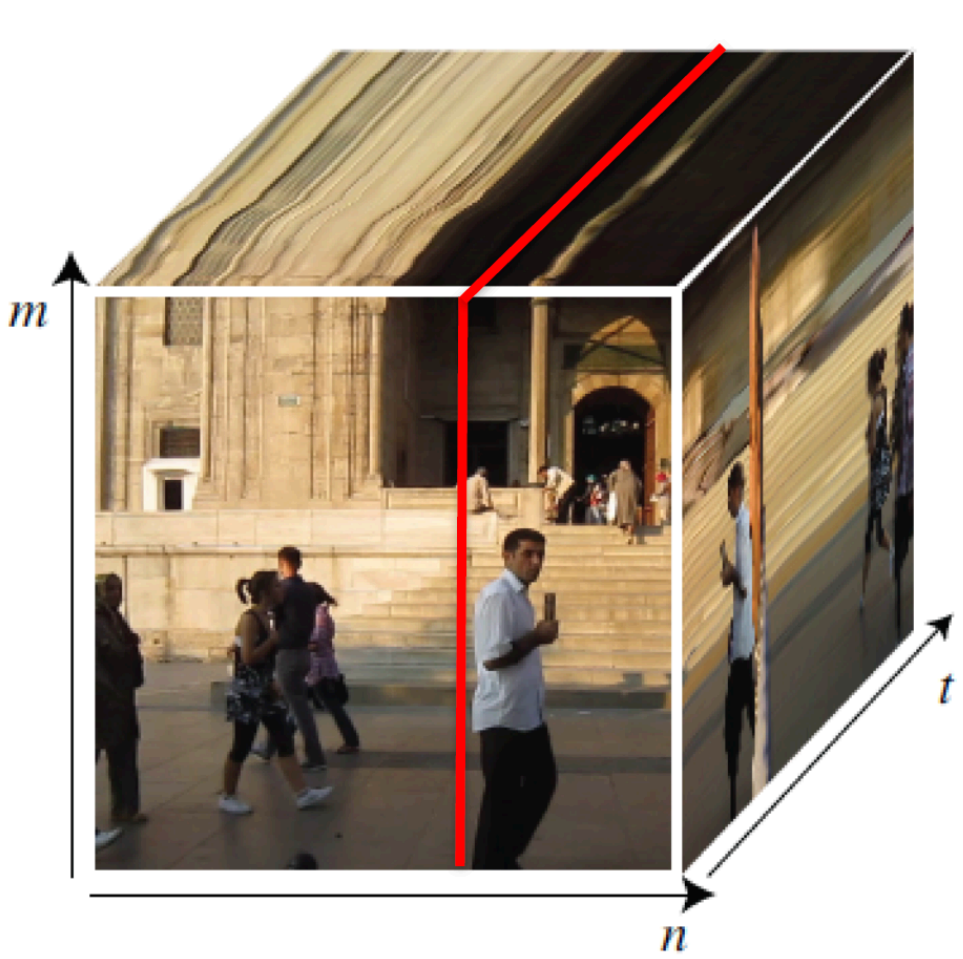
Sequences





Cube size = 128x128x90

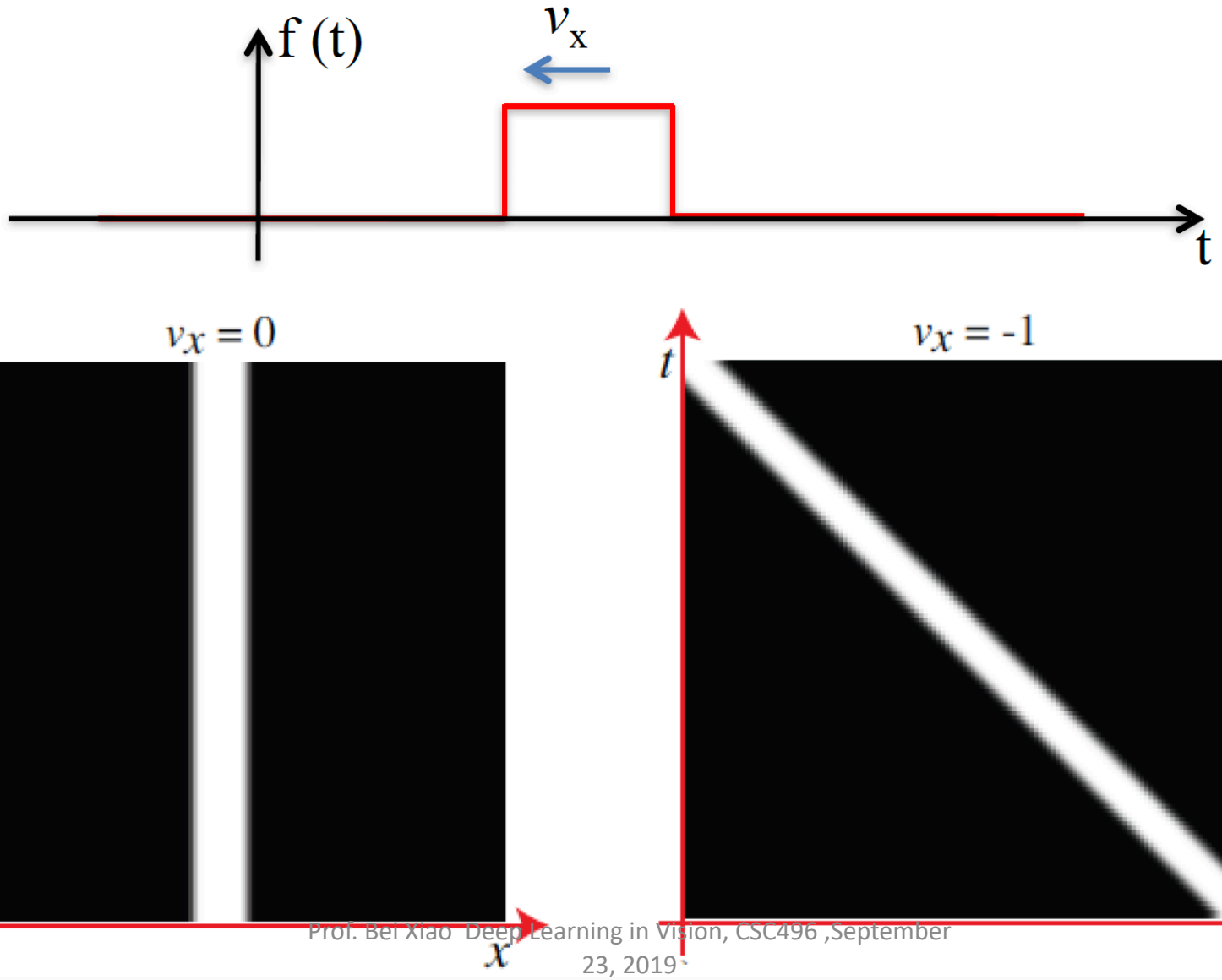
Sequences



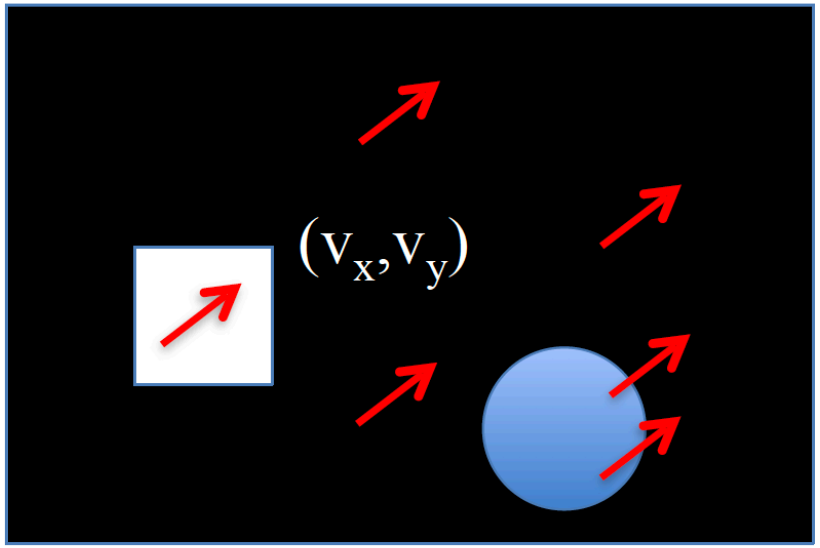
Cube size = 128x128x90

Globally constant motion

Let's work on the continuous space-time domain…



Global constant motion



A global motion can be written as:

$$f(x, y, t) = f_0(x - v_x t, y - v_y t)$$

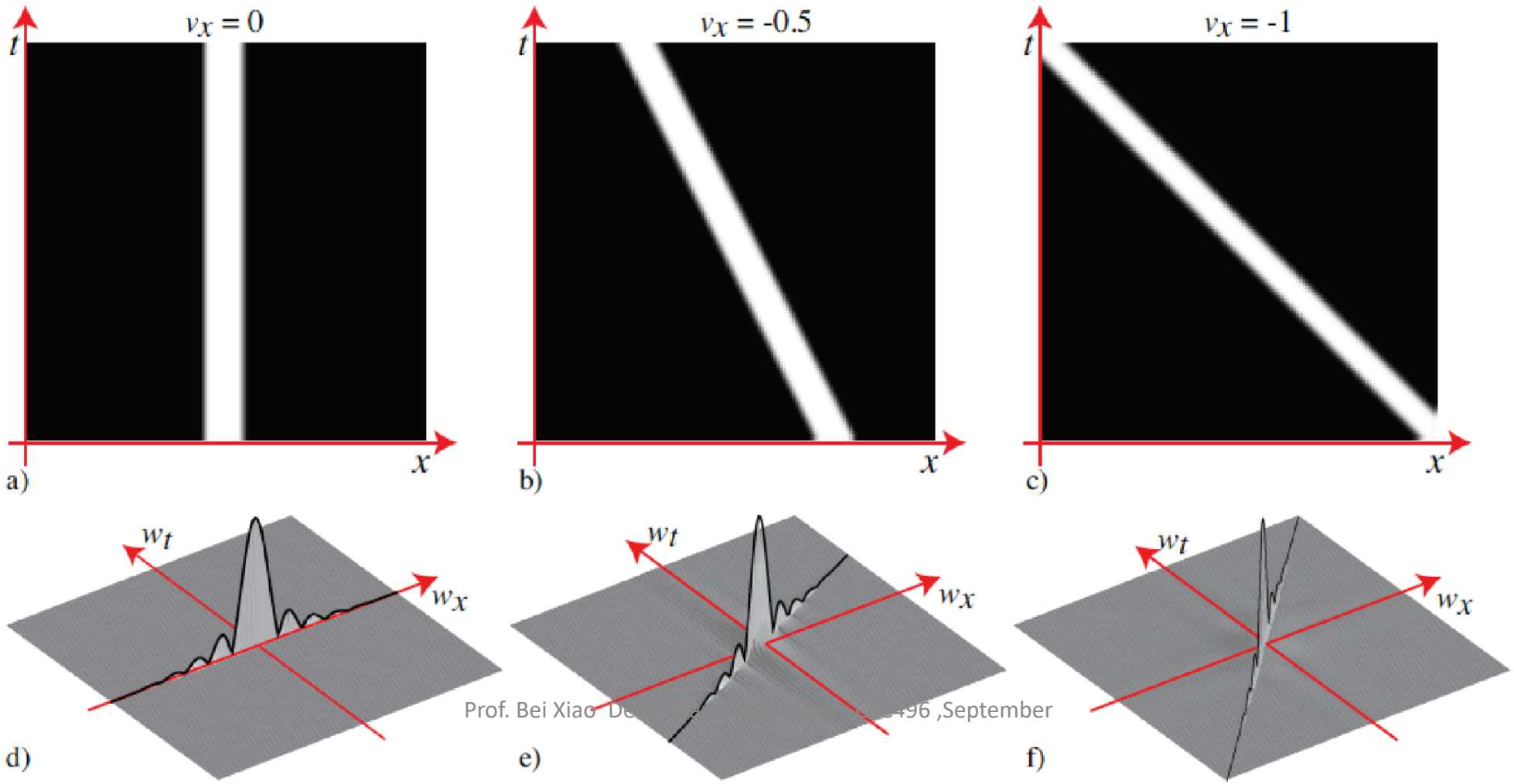
Where:

$$f_0(x, y) = f(x, y, 0)$$

$$f(x,y,t) = f_0(x - v_x t, y - v_y t)$$

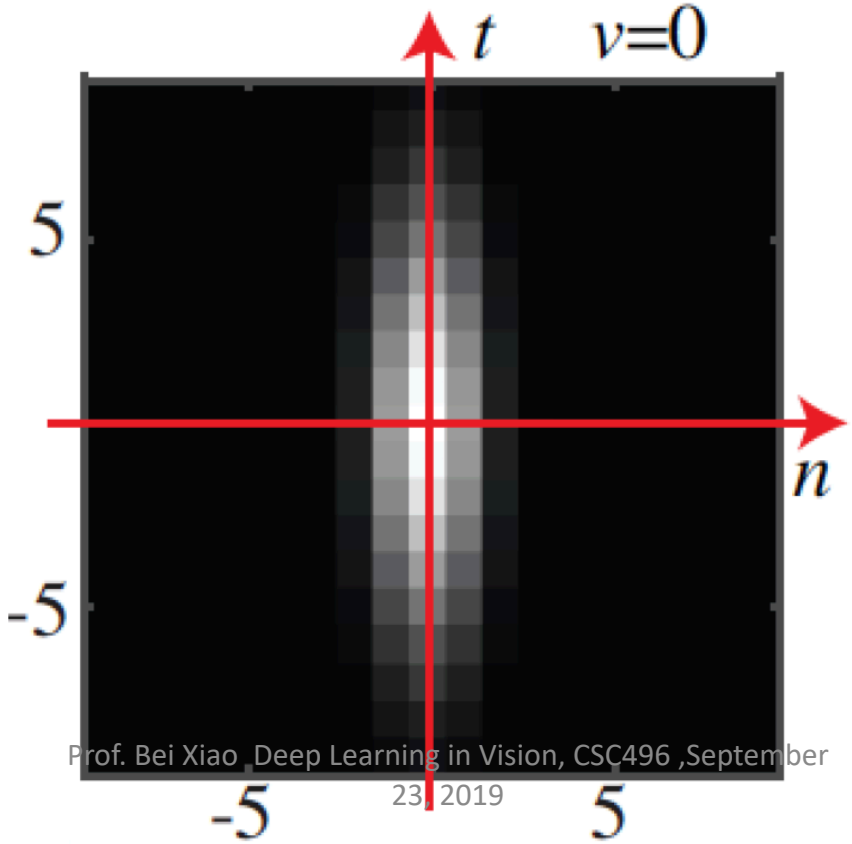
$$\updownarrow$$

$$F(w_x, w_y, w_t) = F_0(w_x, w_y)\delta(w_t + v_x w_x + v_y w_y)$$

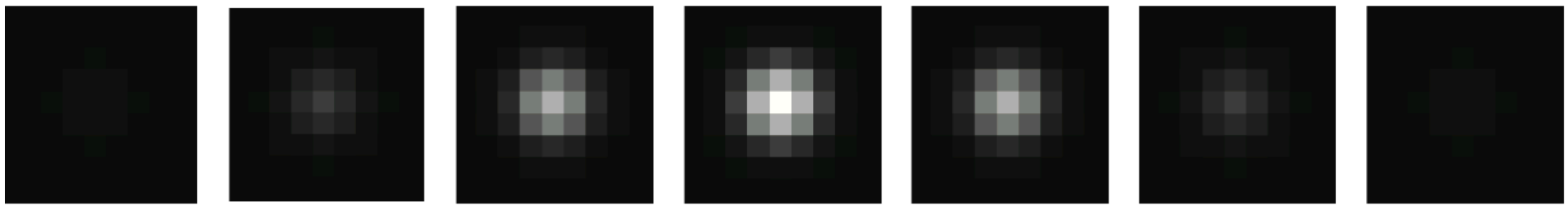


Temporal Gaussian

$$g(x, y, t; \sigma_x, \sigma_t) = \frac{1}{(2\pi)^{3/2}\sigma_x^2\sigma_t} \exp -\frac{x^2 + y^2}{2\sigma_x^2} \exp -\frac{t^2}{2\sigma_t^2}$$

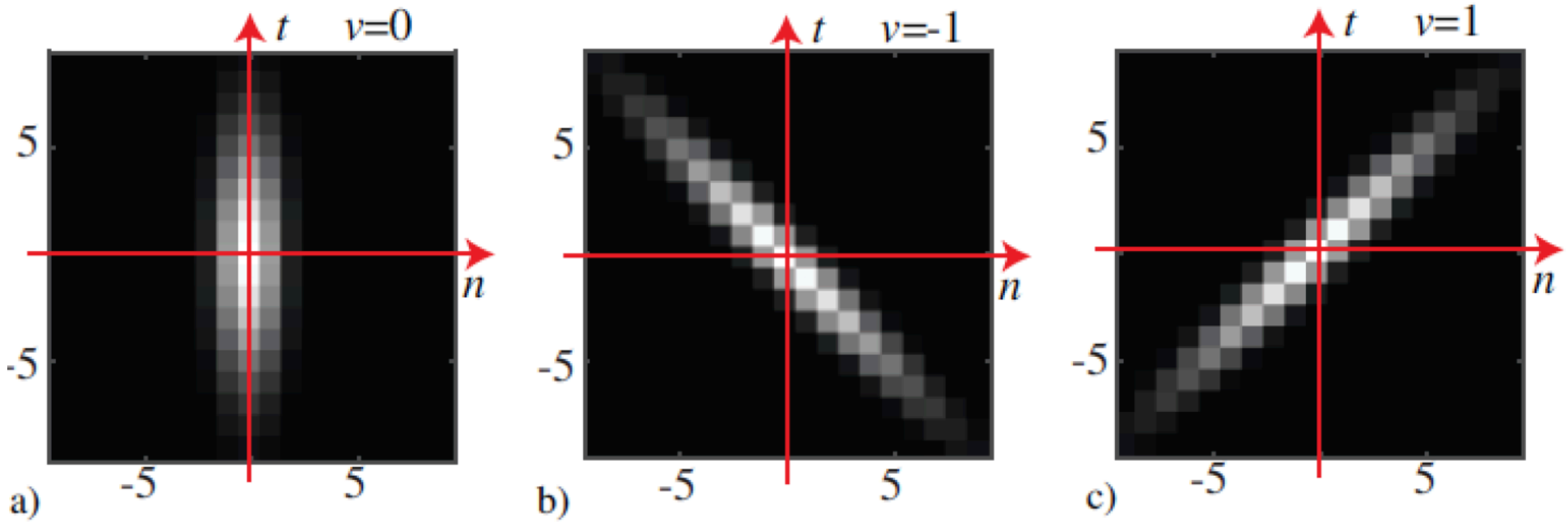


Spatio-temporal Gaussian



Spatio-temporal Gaussian

How could we create a filter that keeps sharp objects that move at some velocity **(vx , vy)** while blurring the rest?

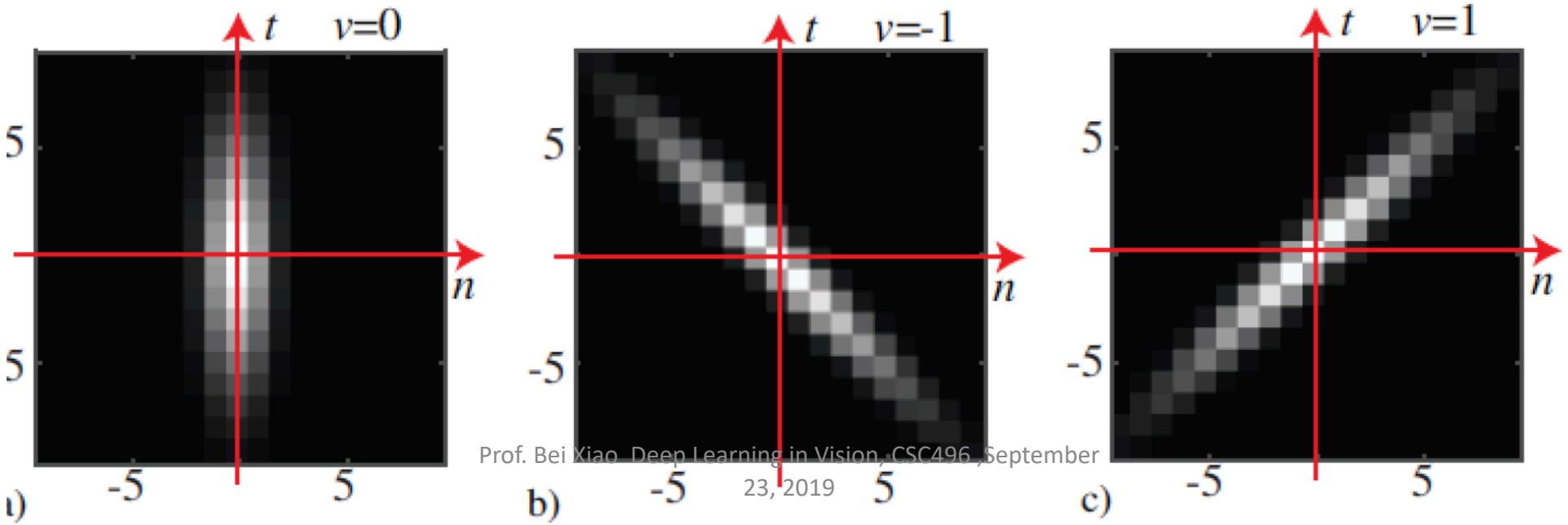


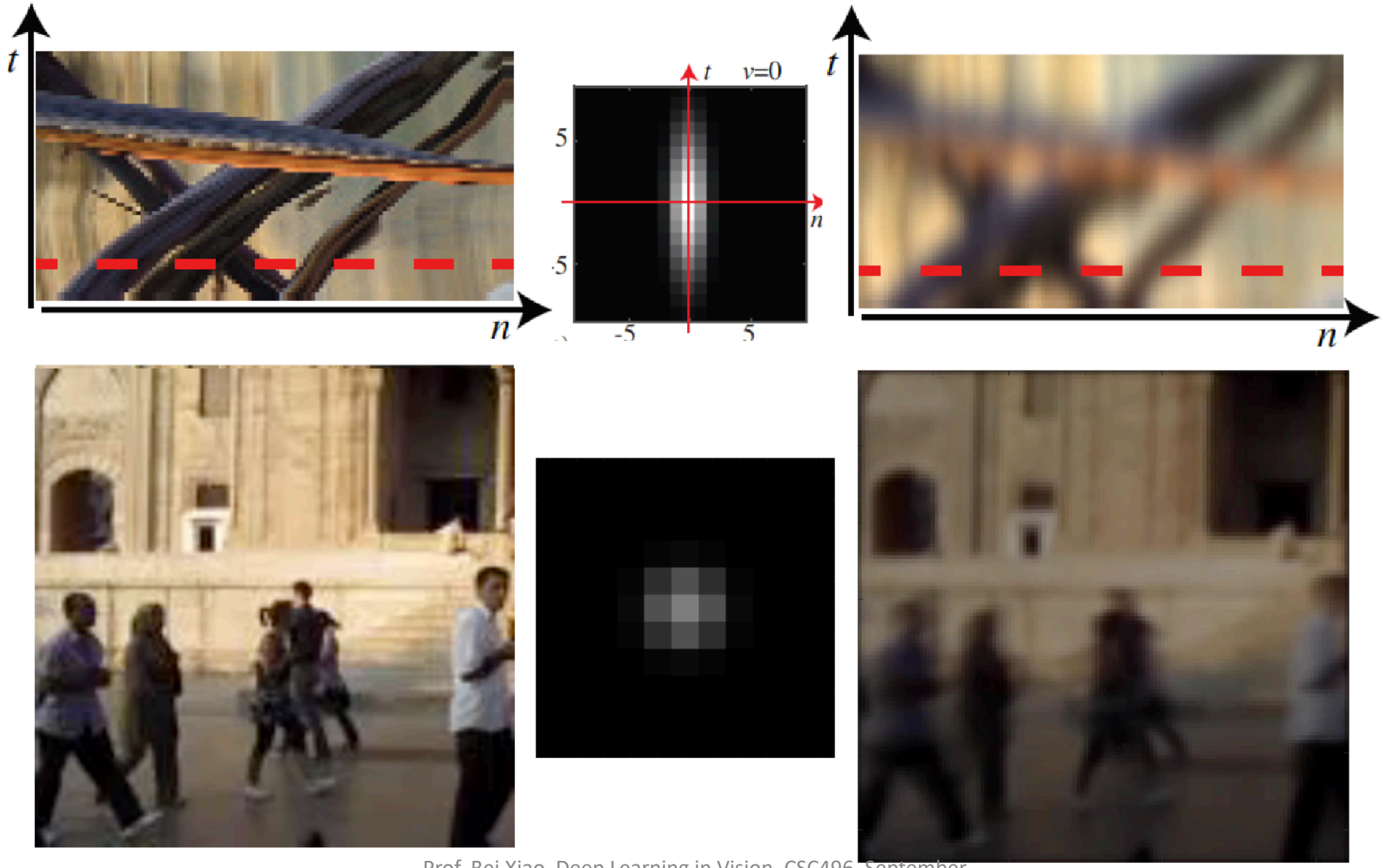
(Note: although some of the analysis is done on continuous variables, the processing is on done on the discrete domain)

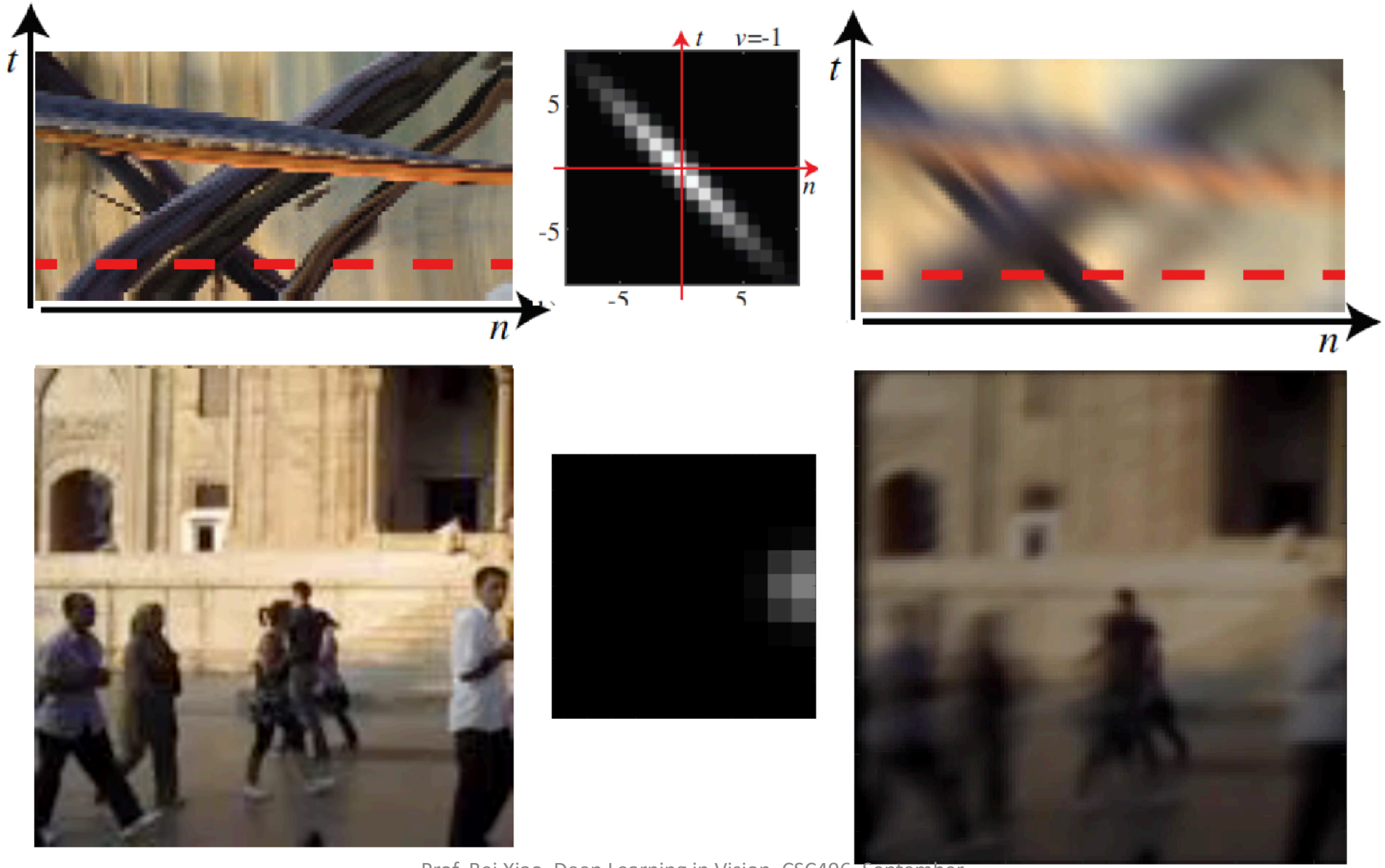
Spatio-temporal Gaussian

How could we create a filter that keeps sharp objects that move at some velocity **(vx , vy)** while blurring the rest?

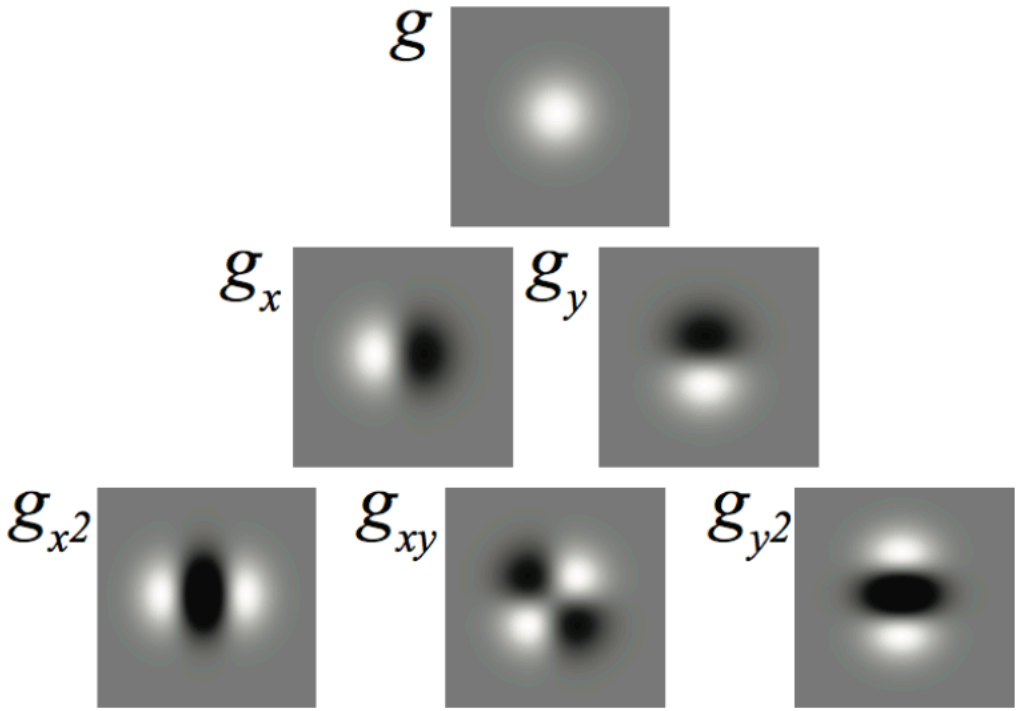
$$g_{v_x,v_y}(x,y,t) = g(x - v_x t, y - v_y t, t)$$



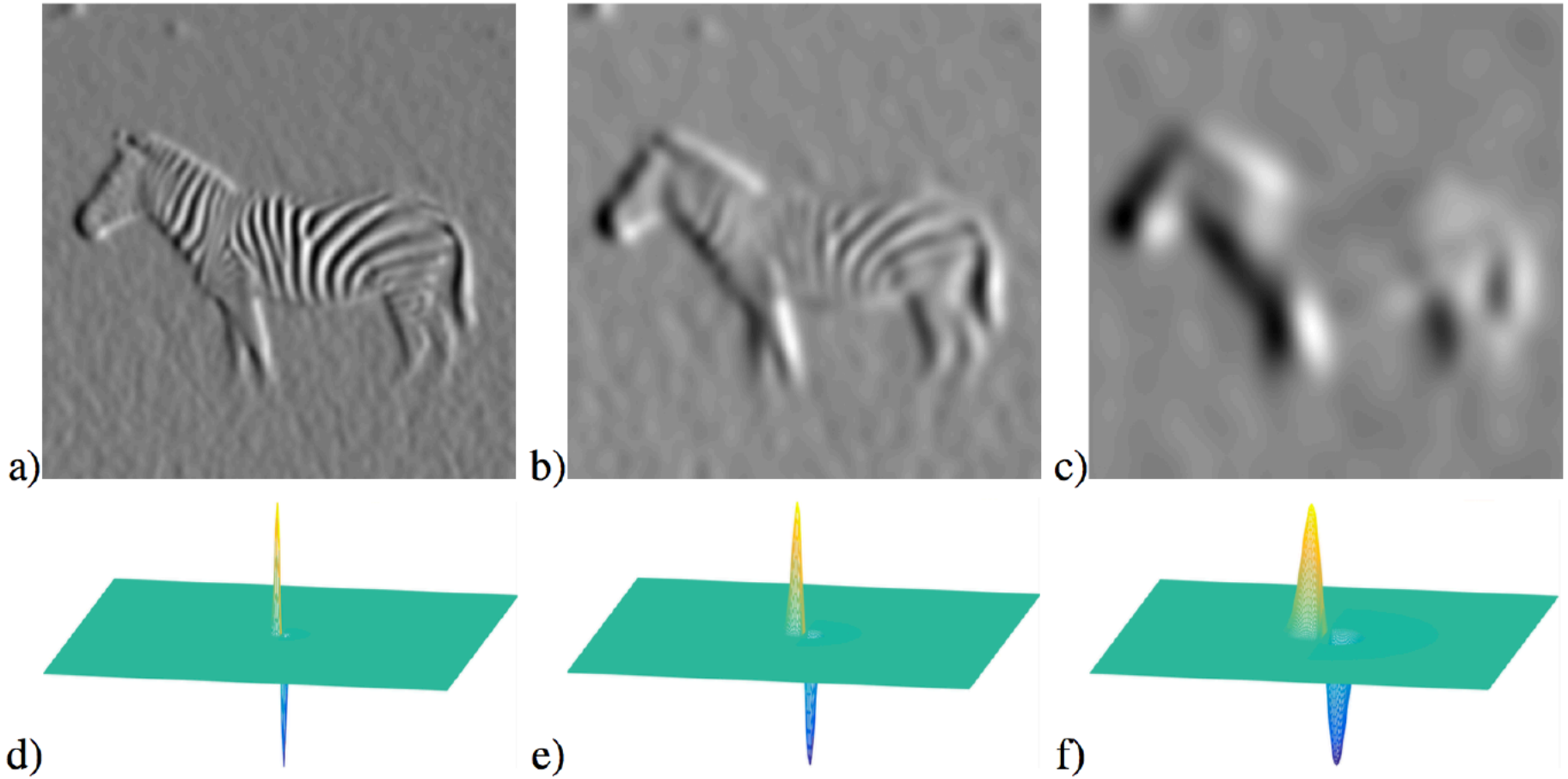

t
n
t
v=0
5
-5
n
-5
5
t
n


t
n
t
v=-1
5
-5
n
-5
5
t
n

derivatives of Gaussians



derivatives of Gaussians



Space-time Gaussian derivatives

$$\frac{\partial g}{\partial t} = \frac{-t}{\sigma_t^2} g(x, y, t)$$

$$\nabla g = \left(g_x(x, y, t), g_y(x, y, t), g_t(x, y, t)\right) =$$
$$= \left(-x/\sigma^2, -y/\sigma^2, -t/\sigma_t^2\right) g(x, y, t)$$

Note: we can discretize time derivatives in the same way we discretized spatial derivatives. For instance:

$$f[m, n, t] - f[m, n, t-1]$$

Cancelling moving objects

Can we create a filter that *removes* objects that move at some velocity **(vx , vy)** while keeping the rest?

Space-time Gaussian derivatives

For a global translation, we can write:

$$f(x, y, t) = f_0(x - v_x t, y - v_y t)$$

Therefore, we can write the temporal derivative of f as a function of the spatial derivatives of f_0 :

$$\frac{\partial f}{\partial t} = \frac{\partial f_0}{\partial t} = -v_x \frac{\partial f_0}{\partial x} - v_y \frac{\partial f_0}{\partial y}$$

And from here (using derivatives of f):

$$\frac{\partial f}{\partial t} + v_x \frac{\partial f}{\partial x} + v_y \frac{\partial f}{\partial y} = 0$$

This relation is known as the "Brightness change constraint equation", introduced by Horn & Schunck in 1981

Space-time Gaussian derivatives

Can could we create a filter that removes objects that move at some velocity **(vx , vy)** while keeping the rest?

Yes, we could create a filter that implements this constraint:

$$\frac{\partial f}{\partial t} + v_x \frac{\partial f}{\partial x} + v_y \frac{\partial f}{\partial y} = 0$$

We can create this filter as a combination of Gaussian derivatives:

$$h(x, y, t; v_x, v_y) = g_t + v_x g_x + v_y g_y$$
$$= \nabla g \left(1, v_x, v_y\right)^T$$

Take-home reading and next class

- Multiscale pyramids

- Lecturenotes: SignalProcessing.pdf

- Linear filtering: page 111-120
- http://szeliski.org/Book/drafts/SzeliskiBook_20100903_draft.pdf