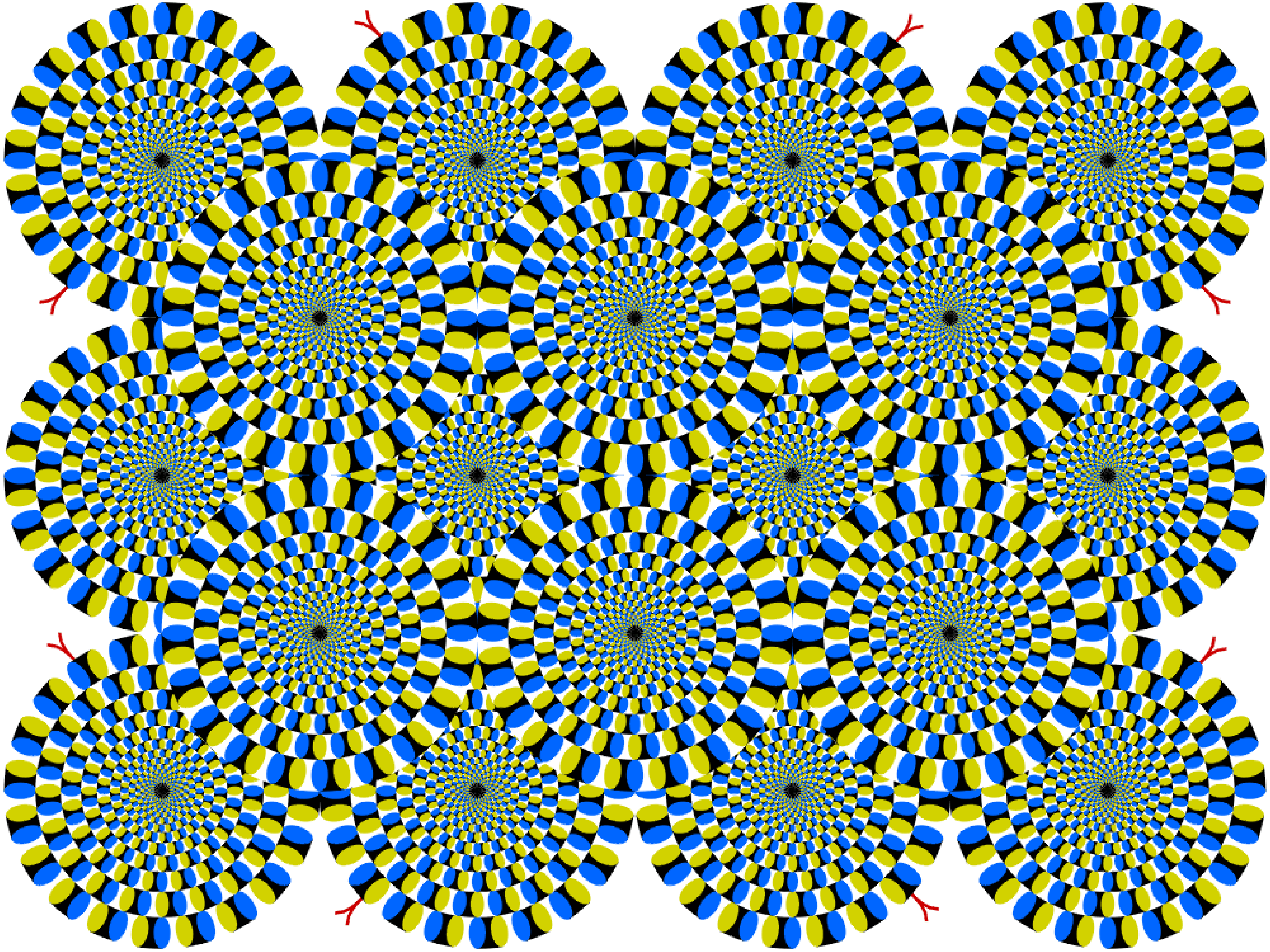
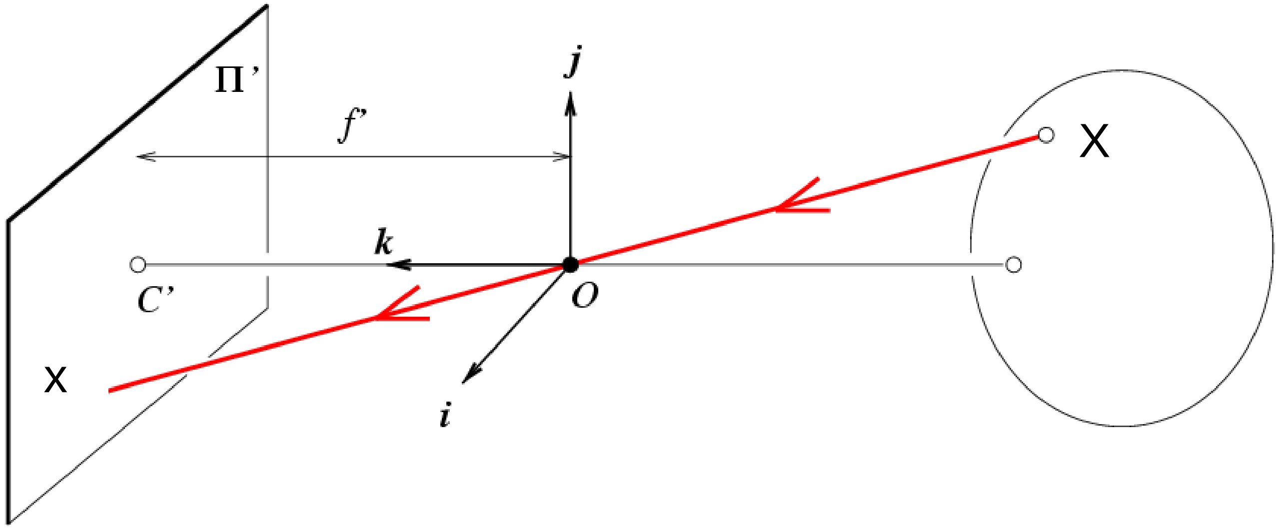


Recap: projection



$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

$$w\begin{bmatrix}u\\v\\1\end{bmatrix} = \begin{bmatrix}\alpha & s & u_0\\0 & \beta & v_0\\0 & 0 & 1\end{bmatrix}\begin{bmatrix}r_{11} & r_{12} & r_{13} & t_x\\r_{21} & r_{22} & r_{23} & t_y\\r_{31} & r_{32} & r_{33} & t_z\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

Relating multiple views

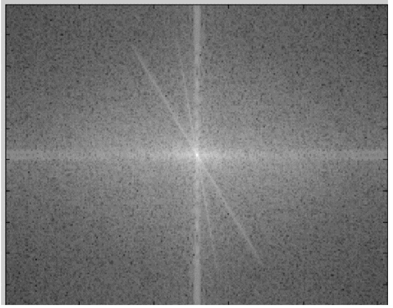
Figure Credit: Bundler: Structure from Motion (SfM) for Unordered Image Collections

Recap of Filtering

- Linear filtering is dot product at each position
  - Not a matrix multiplication
  - Can smooth, sharpen, translate (among many other uses)

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- We can use the Fourier transform to represent images in the frequency domain.
  - Filtering in the spatial domain is multiplication in the frequency domain.

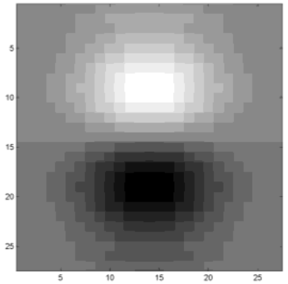


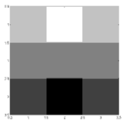
Canvas Quiz

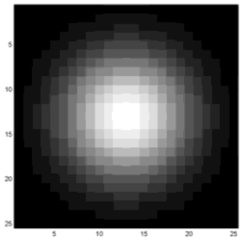
Fill in the blanks:

Filering Operator

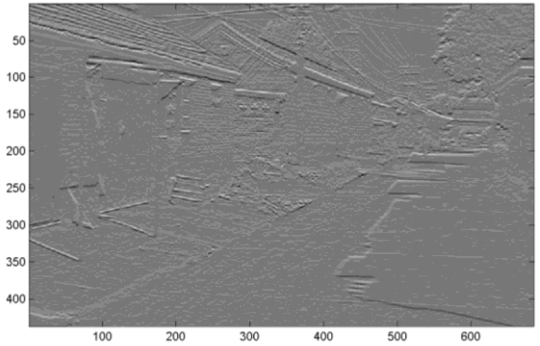
1) _ = D * B
2) A = _ * C
3) F = D * _
4) _ = D * D

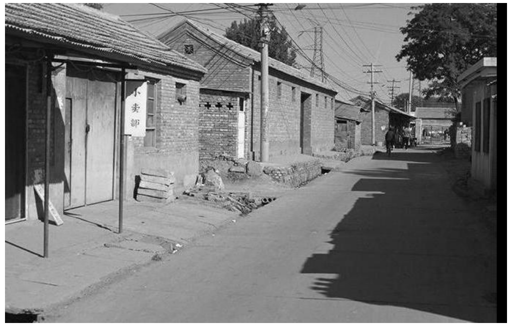
A


B


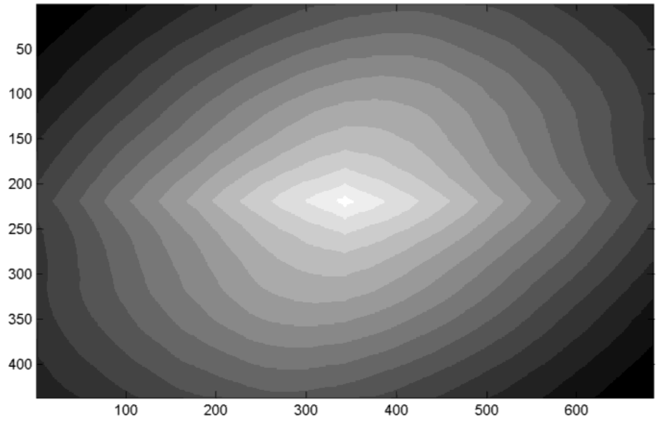
C


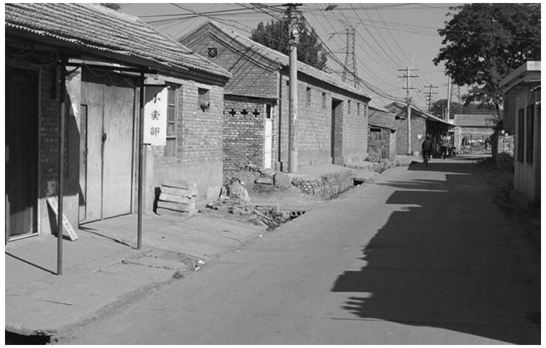
E


G


F


H

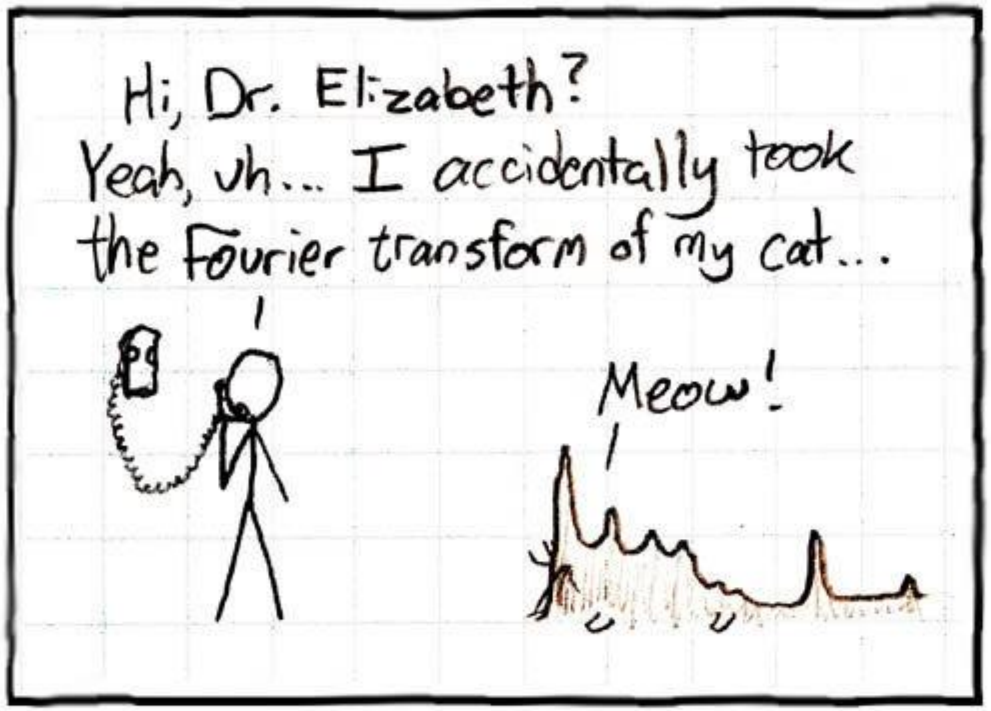

I


D


Slide: Hoiem

Other signals

- We can also think of all kinds of other signals the same way

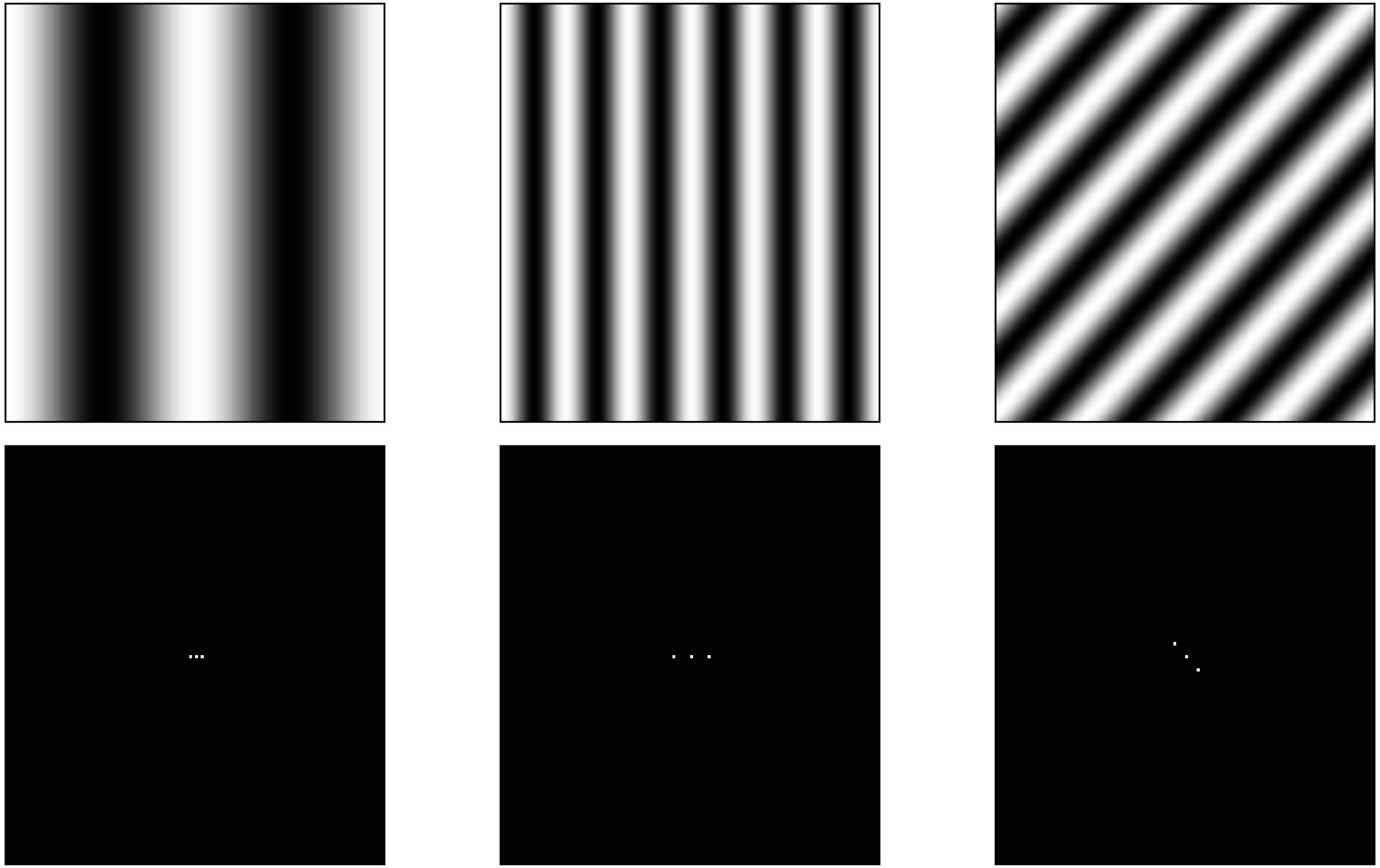


xkcd.com

Fourier analysis in images

Intensity Image

Fourier Image



http://sharp.bu.edu/~slehar/fourier/fourier.html#filtering

Fourier Transform

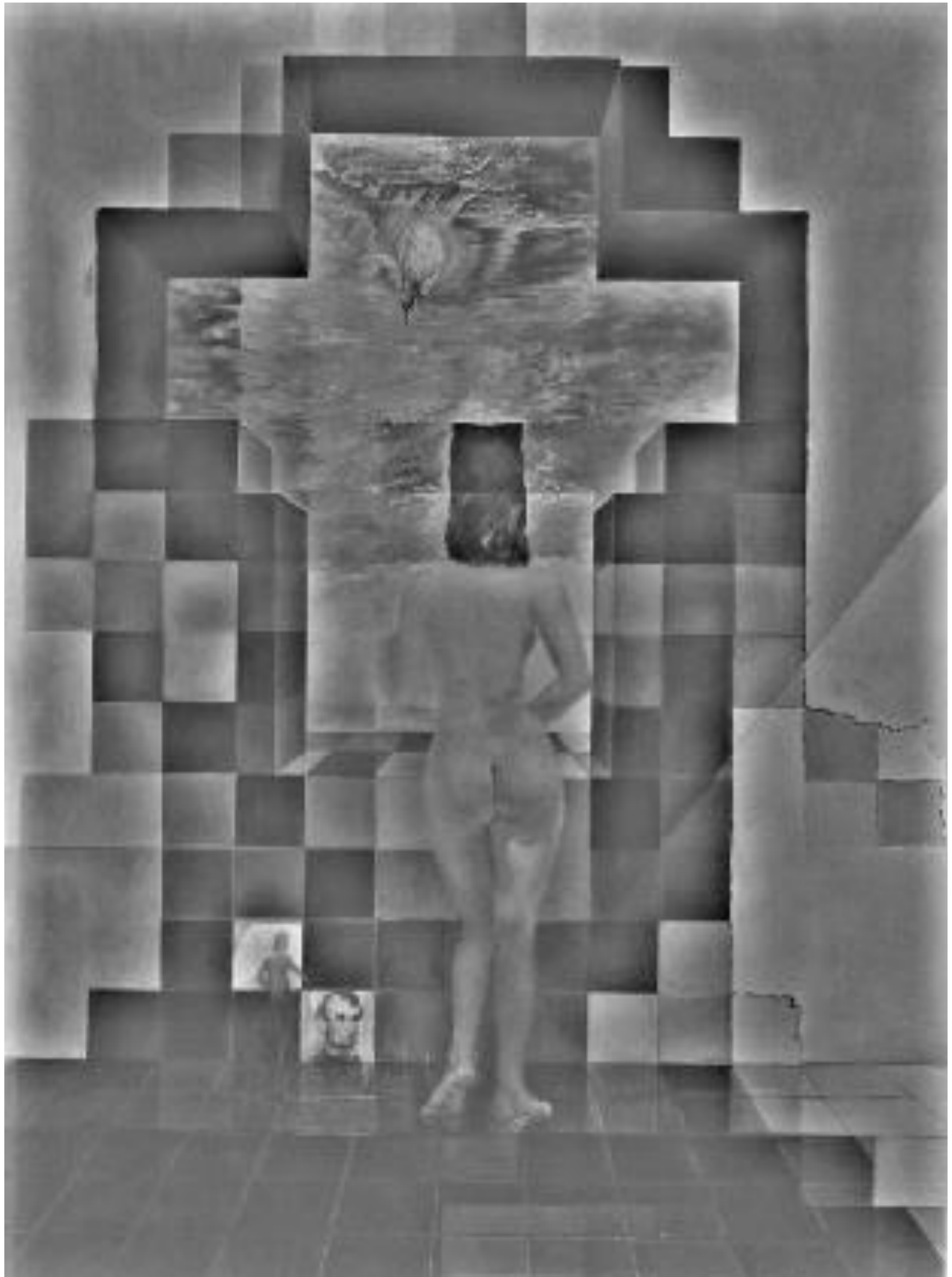
- Fourier transform stores the magnitude and phase at each frequency
  - Magnitude encodes how much signal there is at a particular frequency
  - Phase encodes spatial information (indirectly)
  - For mathematical convenience, this is often notated in terms of real and complex numbers

Amplitude: $A = \pm\sqrt{R(\omega)^2 + I(\omega)^2}$

Phase: $\phi = \tan^{-1} \frac{I(\omega)}{R(\omega)}$

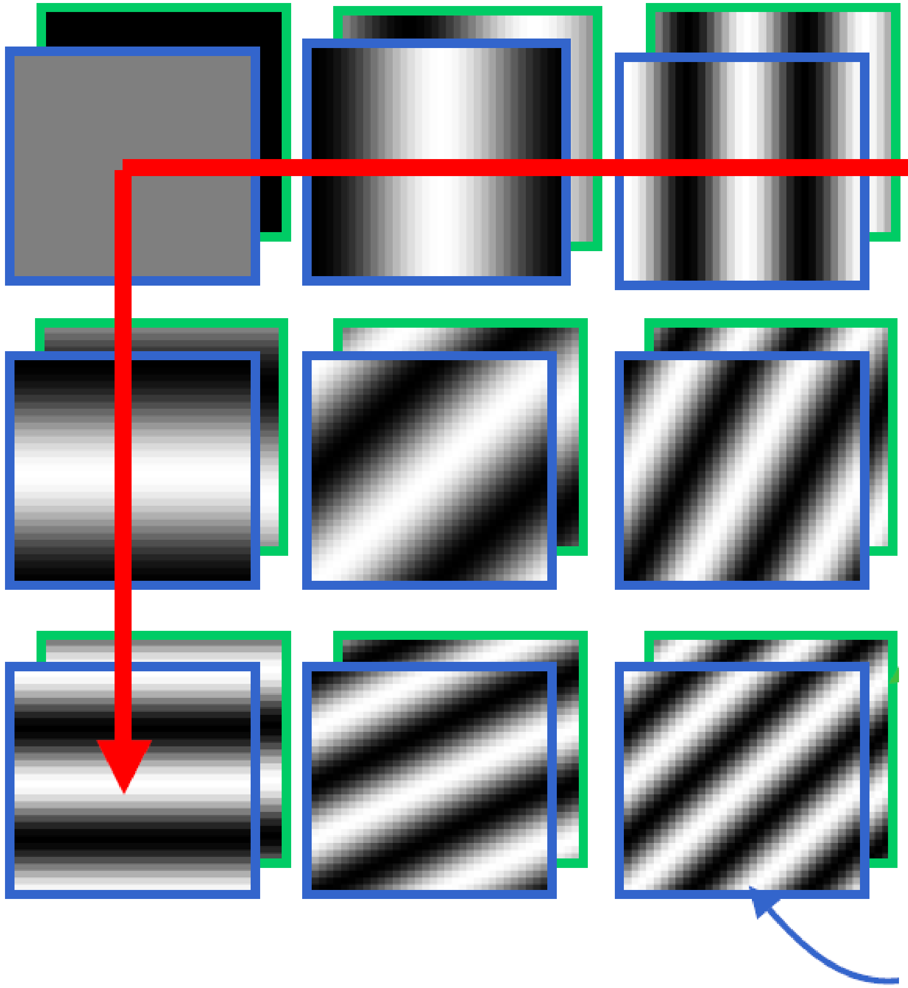
Salvador Dali invented Hybrid Images?

Salvador Dali
“Gala Contemplating the Mediterranean Sea, which at 20 meters becomes the portrait of Abraham Lincoln”, 1976



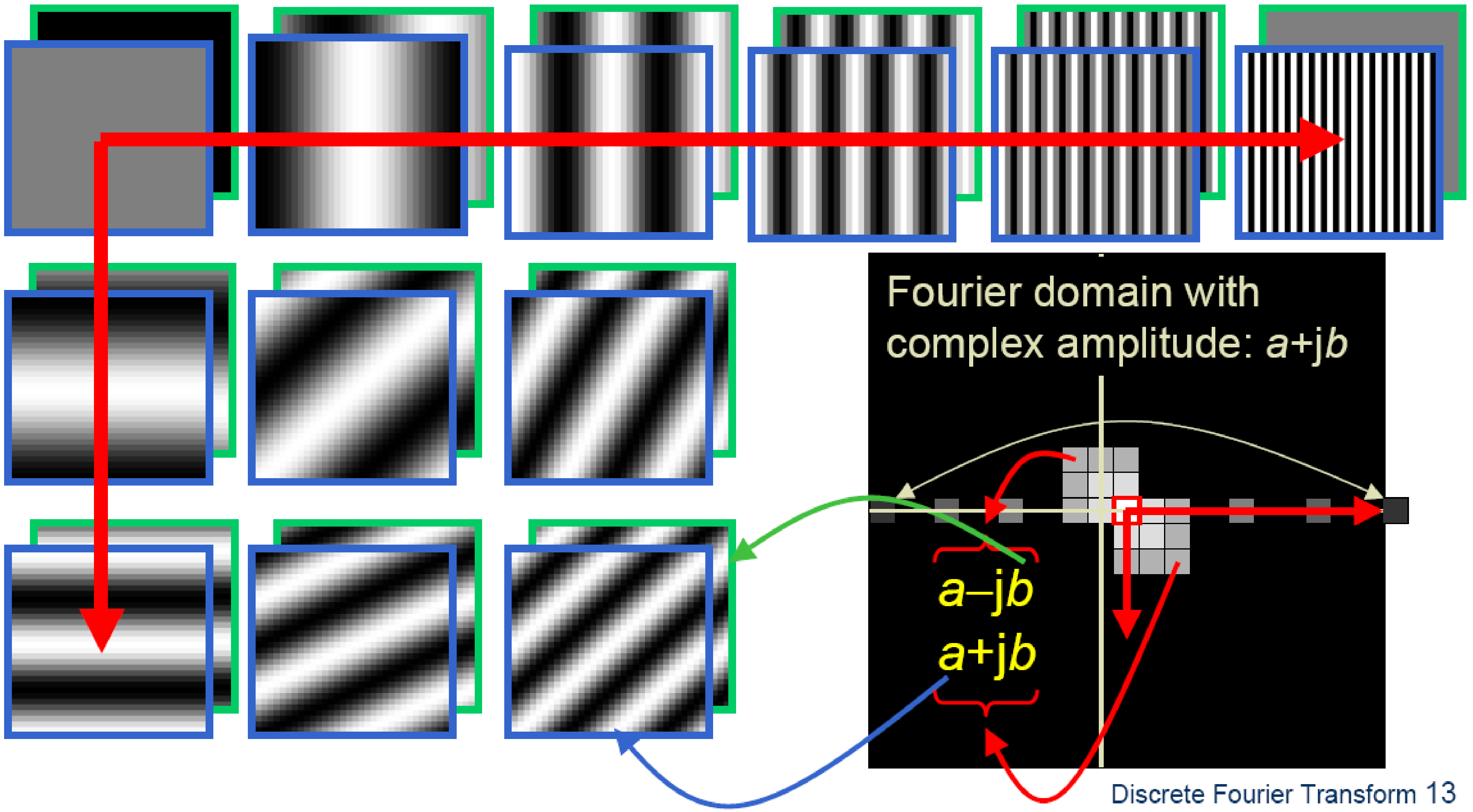
Fourier Bases

Teases away fast vs. slow changes in the image.

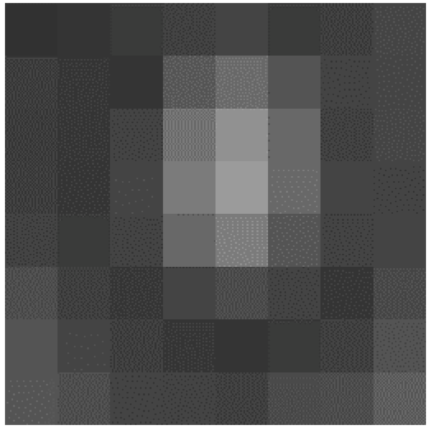


This change of basis is the Fourier Transform

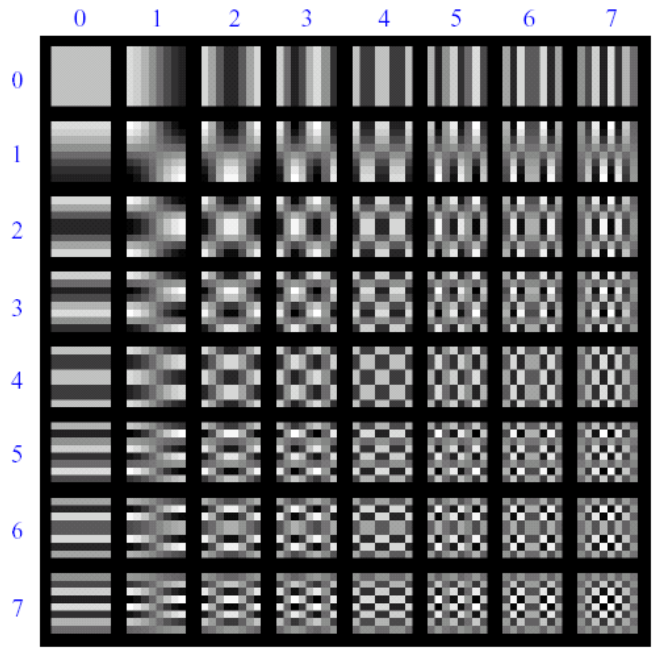
Fourier Bases



This looks a lot like DCT in JPEG compression



8x8 image patch

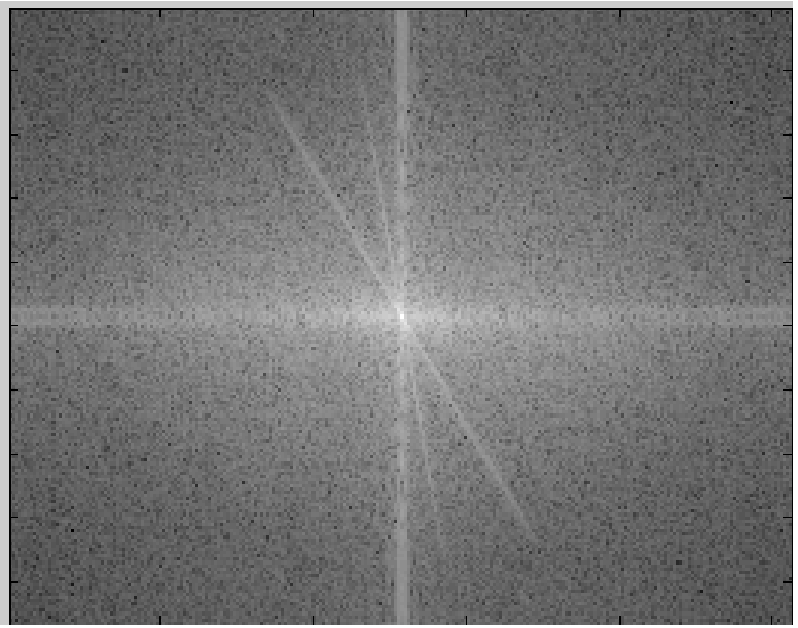


DCT bases

$$G = \begin{array}{c} \xrightarrow{u} \\ \begin{bmatrix} -415.38 & -30.19 & -61.20 & 27.24 & 56.13 & -20.10 & -2.39 & 0.46 \\ 4.47 & -21.86 & -60.76 & 10.25 & 13.15 & -7.09 & -8.54 & 4.88 \\ -46.83 & 7.37 & 77.13 & -24.56 & -28.91 & 9.93 & 5.42 & -5.65 \\ -48.53 & 12.07 & 34.10 & -14.76 & -10.24 & 6.30 & 1.83 & 1.95 \\ 12.12 & -6.55 & -13.20 & -3.95 & -1.88 & 1.75 & -2.79 & 3.14 \\ -7.73 & 2.91 & 2.38 & -5.94 & -2.38 & 0.94 & 4.30 & 1.85 \\ -1.03 & 0.18 & 0.42 & -2.42 & -0.88 & -3.02 & 4.12 & -0.66 \\ -0.17 & 0.14 & -1.07 & -4.19 & -1.17 & -0.10 & 0.50 & 1.68 \end{bmatrix} \end{array} \Big\downarrow v$$

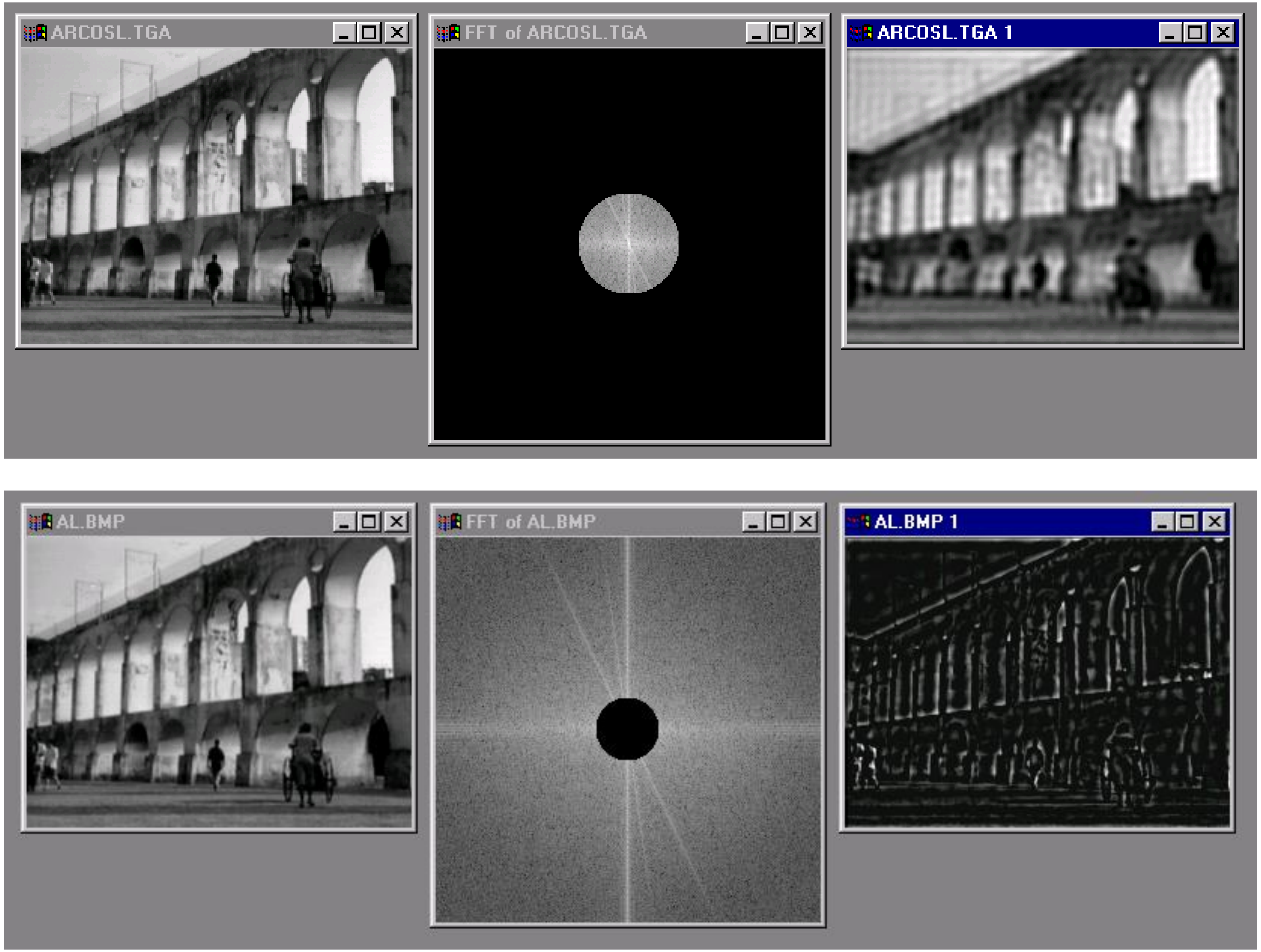
Patch representation after projecting on to DCT bases

Man-made Scene



Can change spectrum, then reconstruct

Low and High Pass filtering



Computing the Fourier Transform

$$H(\omega) = \mathcal{F}\left\{h(x)\right\} = Ae^{j\phi}$$

Continuous

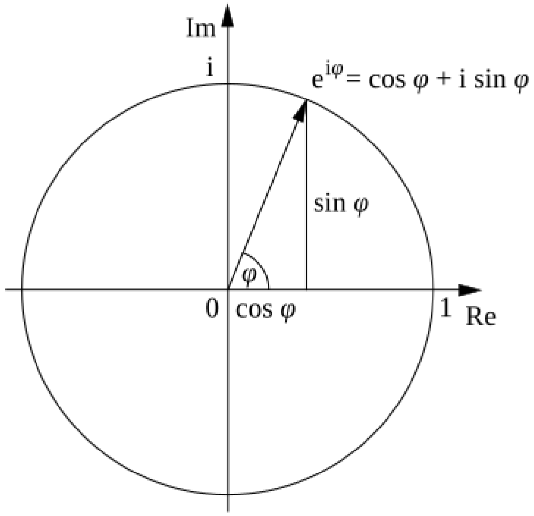
$$H(\omega) = \int_{-\infty}^{\infty} h(x)e^{-j\omega x}dx$$

Discrete

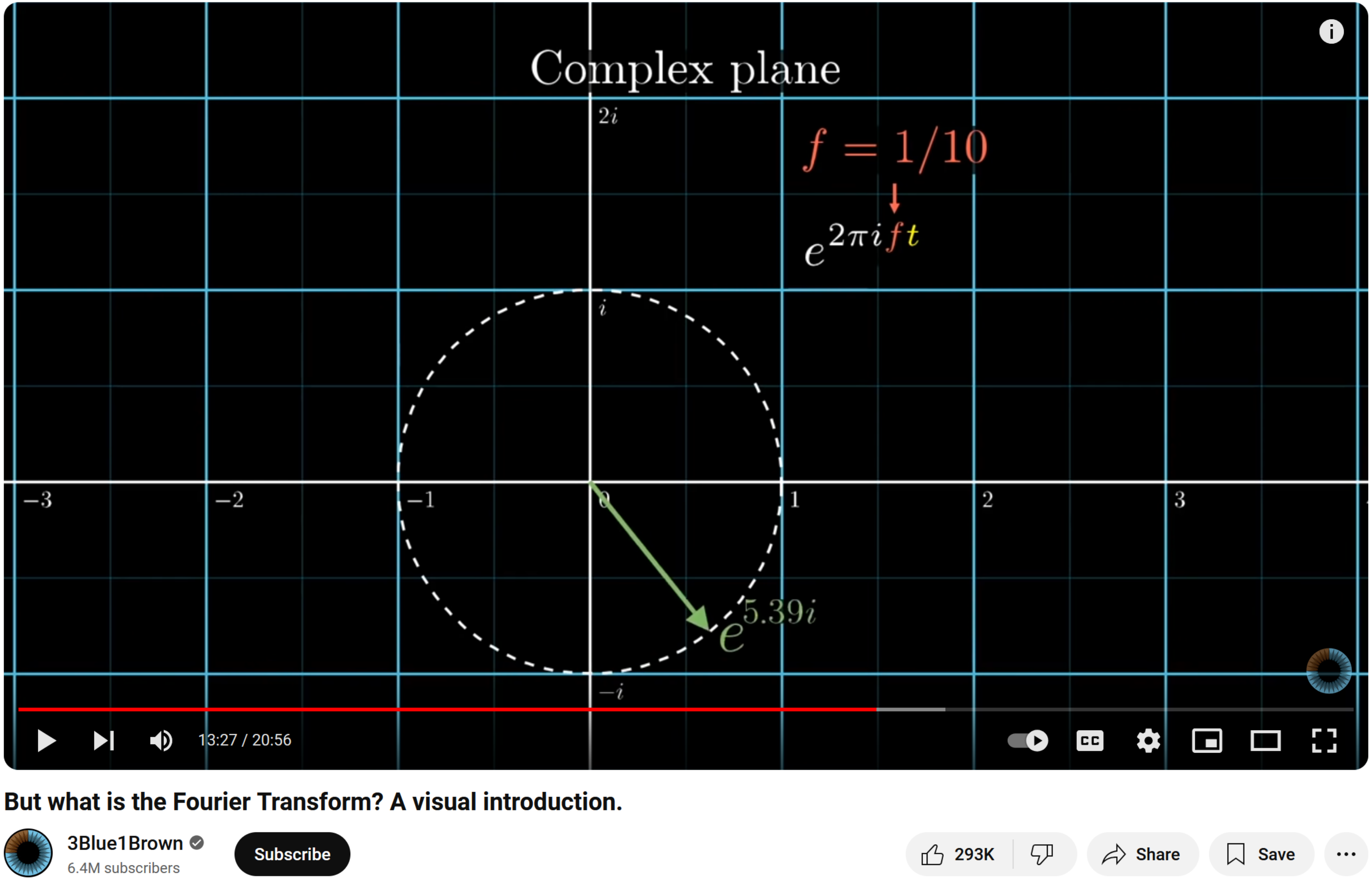
$$H(k) = \frac{1}{N}\sum_{x=0}^{N-1} h(x)e^{-j\frac{2\pi kx}{N}}$$

k = -N/2..N/2

Fast Fourier Transform (FFT): NlogN



Euler's Formula



https://youtu.be/spUNpyF58BY?si=93x8YxT5n45OA3CD

The Convolution Theorem

- The Fourier transform of the convolution of two functions is the product of their Fourier transforms

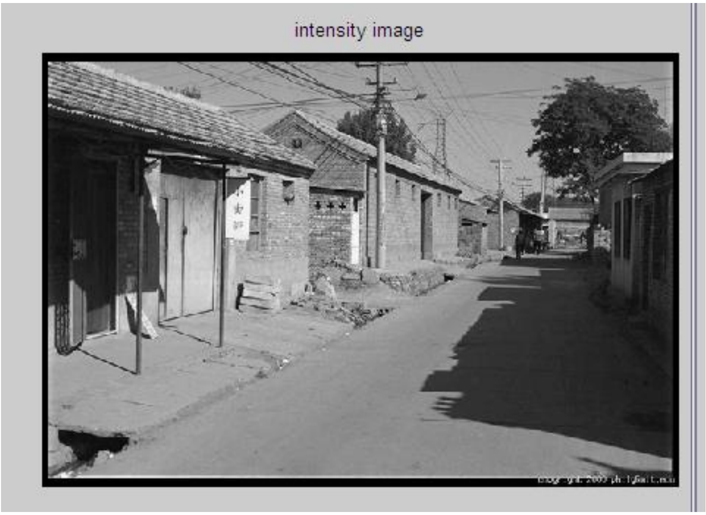
$$\mathrm{F}[g * h] = \mathrm{F}[g]\,\mathrm{F}[h]$$

- **Convolution** in spatial domain is equivalent to **multiplication** in frequency domain!

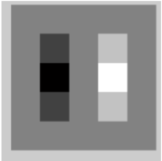
$$g * h = \mathrm{F}^{-1}[\mathrm{F}[g]\,\mathrm{F}[h]]$$

Filtering in spatial domain

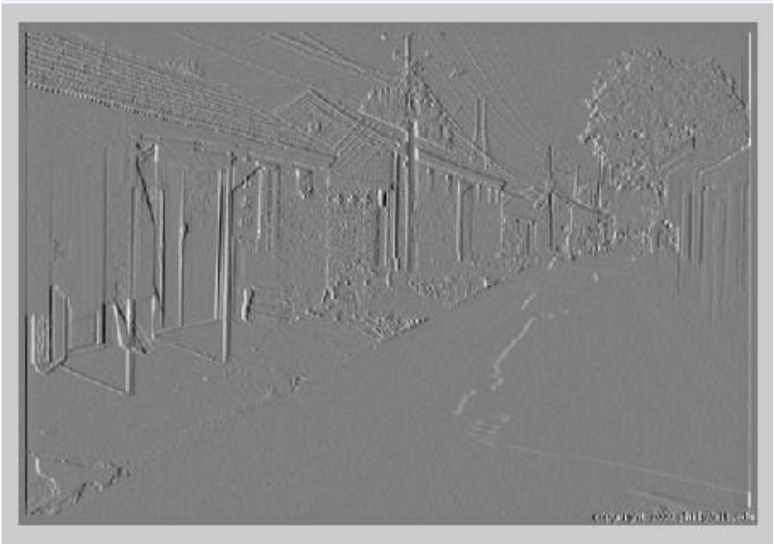
1	0	-1
2	0	-2
1	0	-1



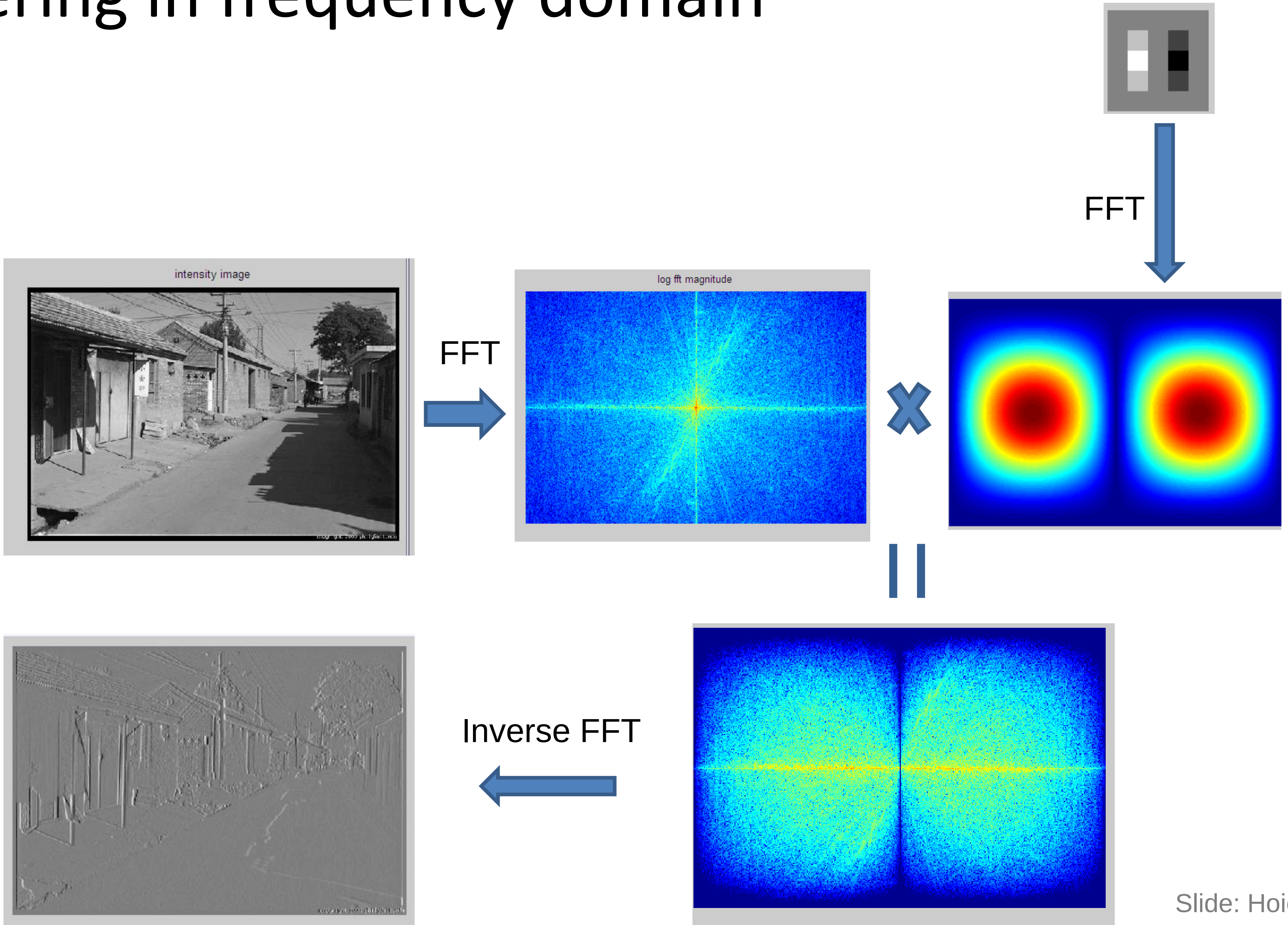
*



=



Filtering in frequency domain

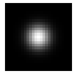


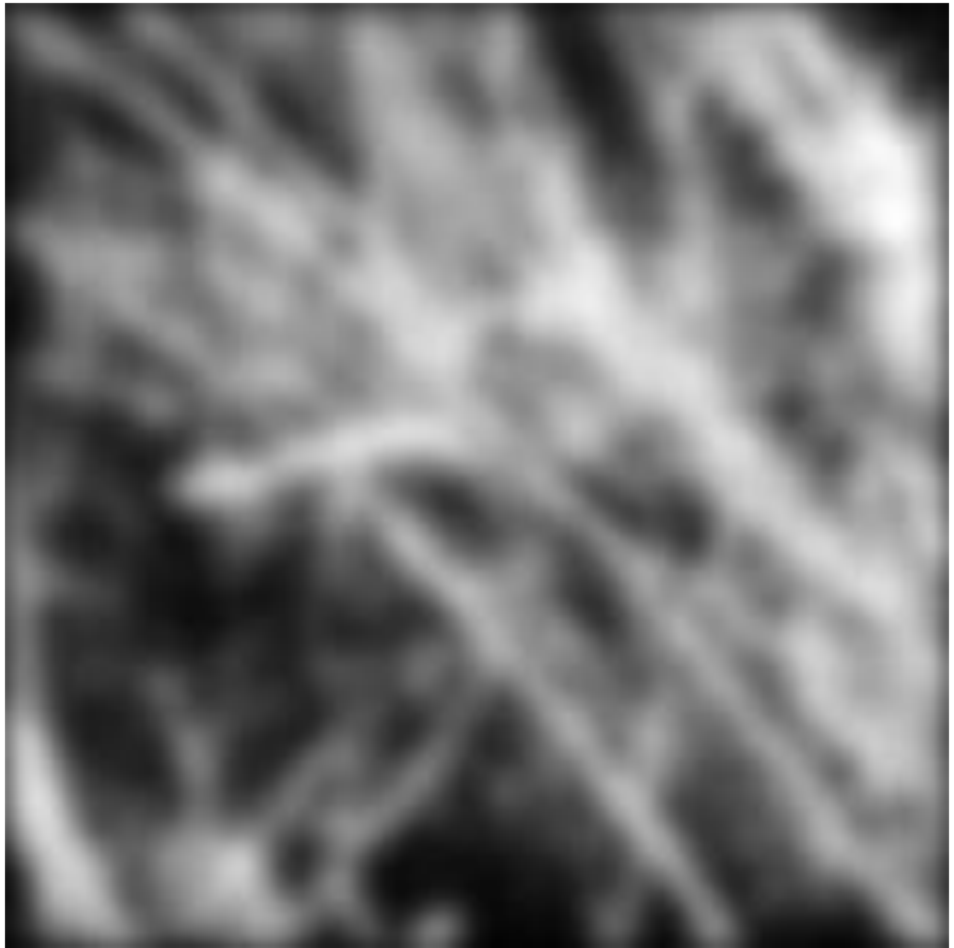
Slide: Hoiem

Filtering

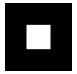
Why does the Gaussian give a nice smooth image, but the square filter give edgy artifacts?

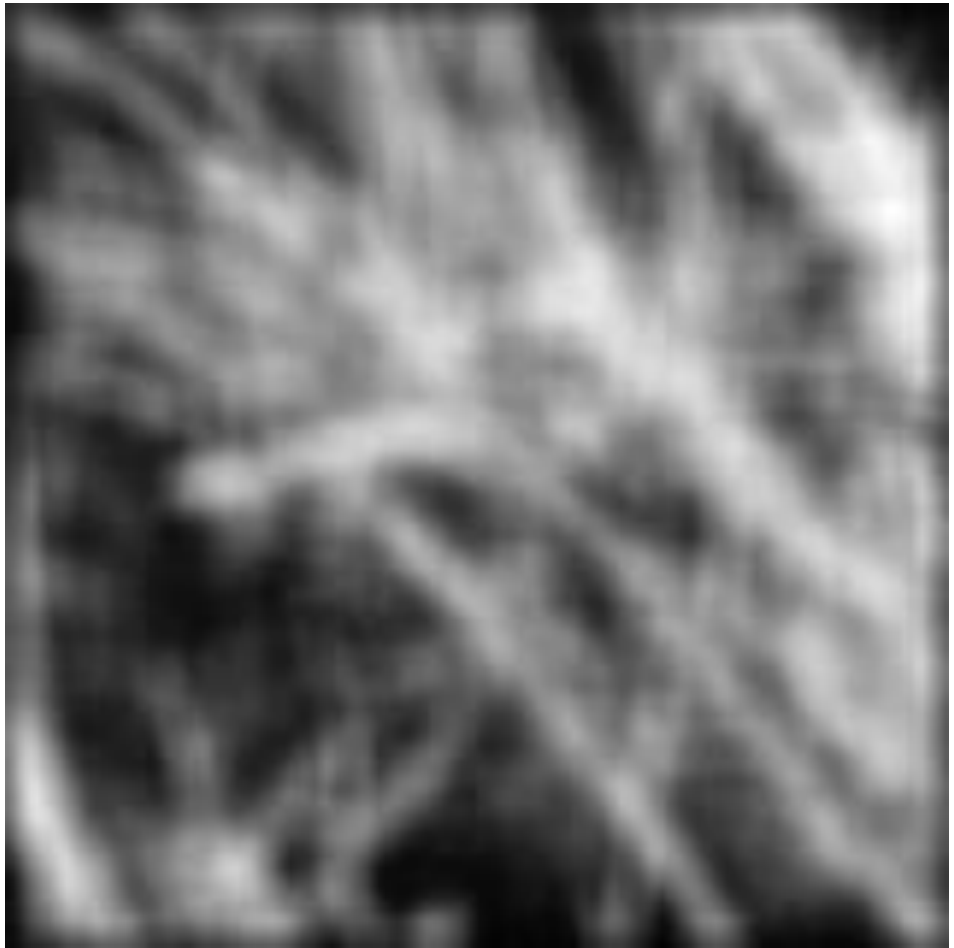
Gaussian



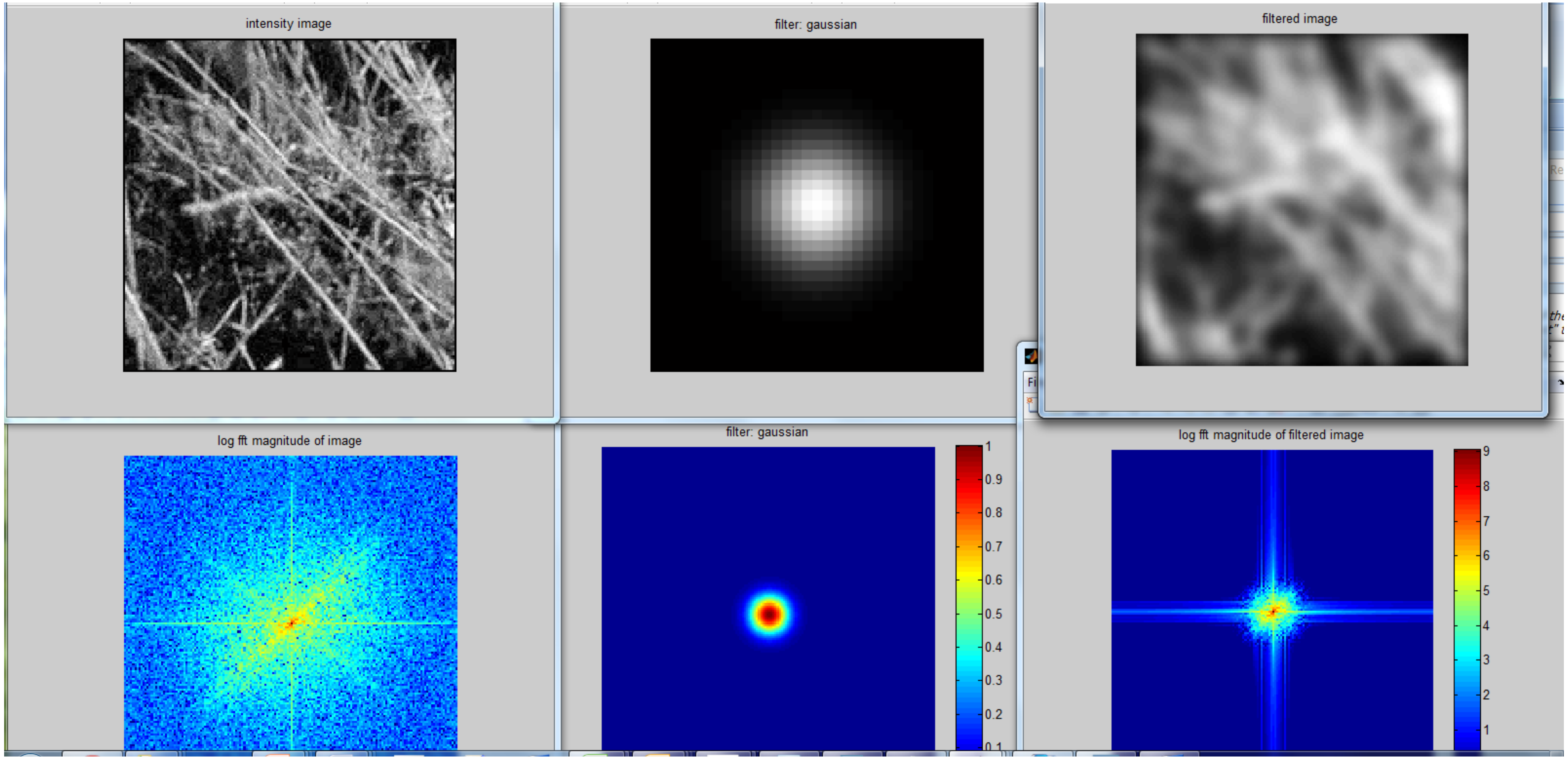


Box filter



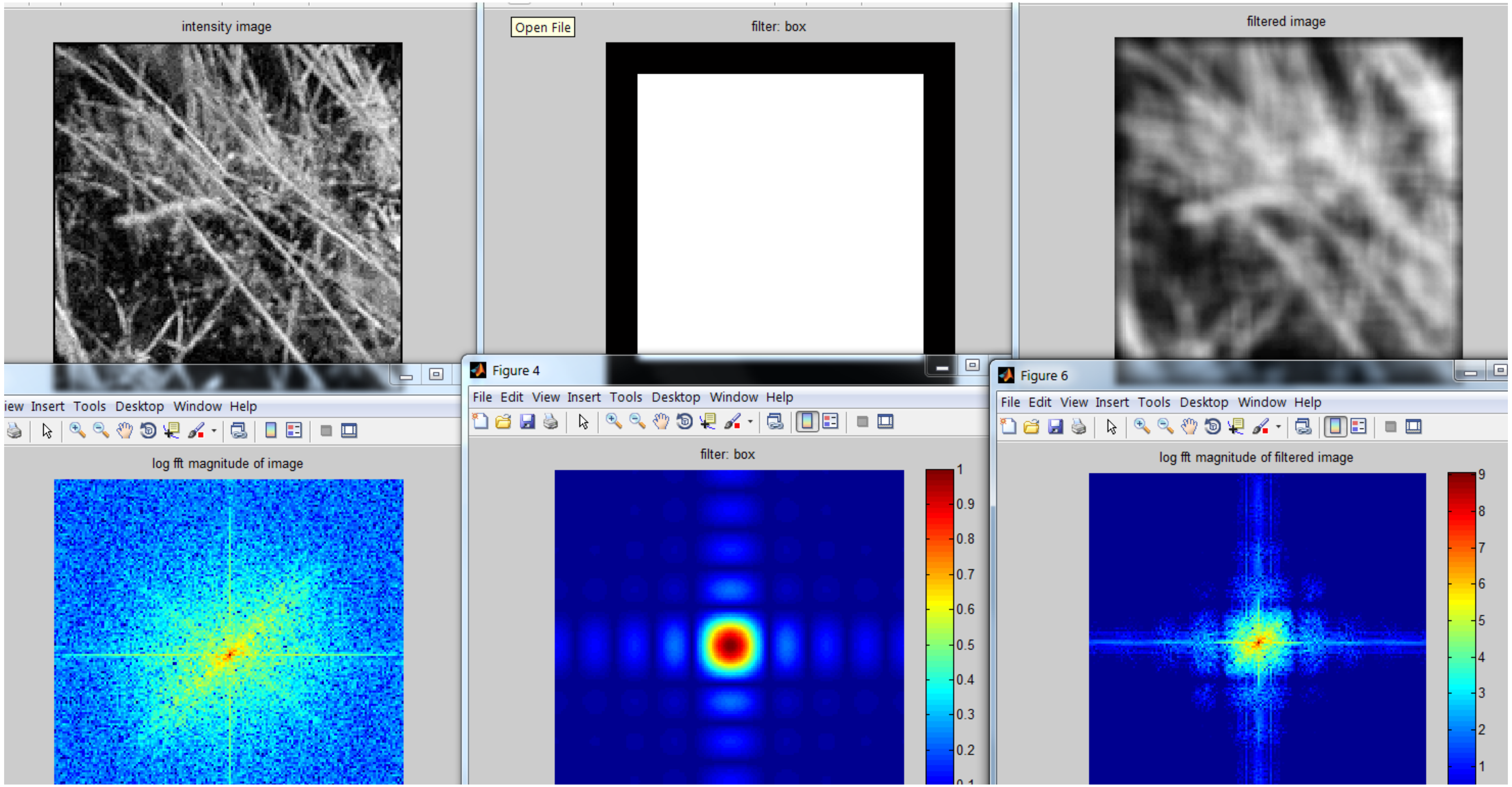


Gaussian



intensity image
filter: gaussian
filtered image
log fft magnitude of image
filter: gaussian
log fft magnitude of filtered image

Box Filter

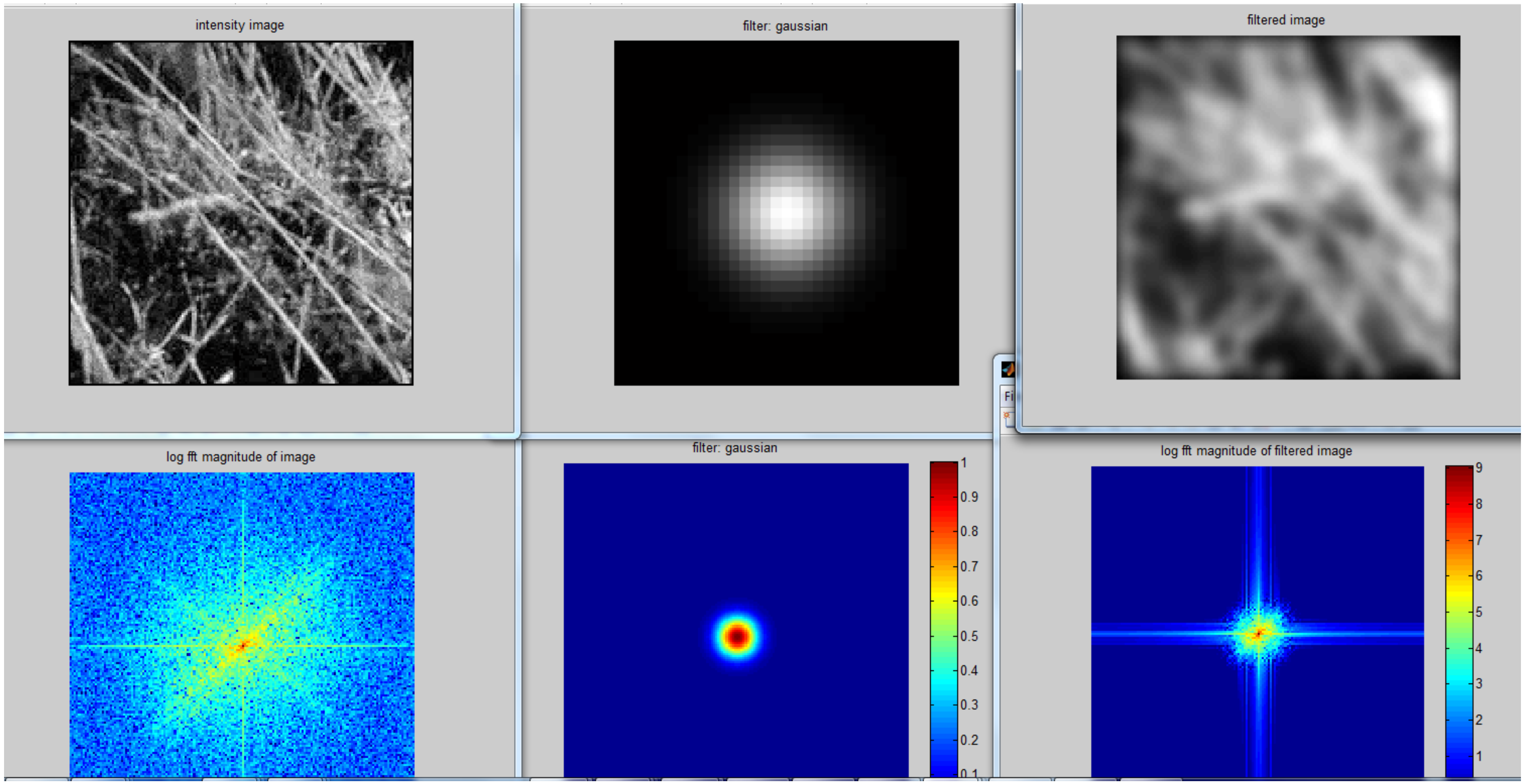

intensity image
Open File
filter: box
filtered image
Figure 4
Figure 6
File Edit View Insert Tools Desktop Window Help
log fft magnitude of image
filter: box
log fft magnitude of filtered image

Is convolution invertible?

- If convolution is just multiplication in the Fourier domain, isn't deconvolution just division?
- Sometimes, it clearly is invertible (e.g. a convolution with an identity filter)
- In one case, it clearly isn't invertible (e.g. convolution with an all zero filter)
- What about for common filters like a Gaussian?

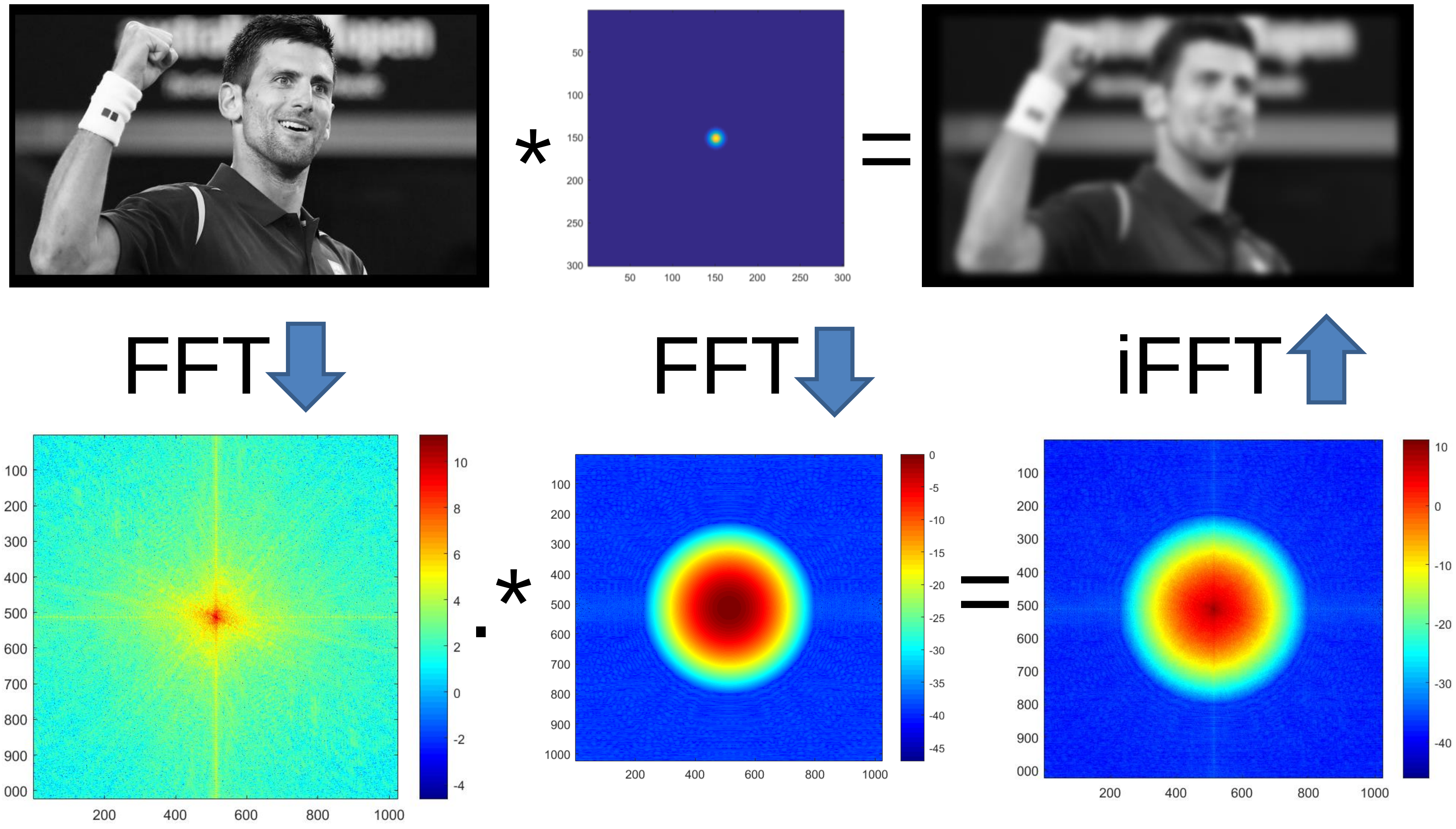
But you can't invert multiplication by 0

- But it's not quite zero, is it…

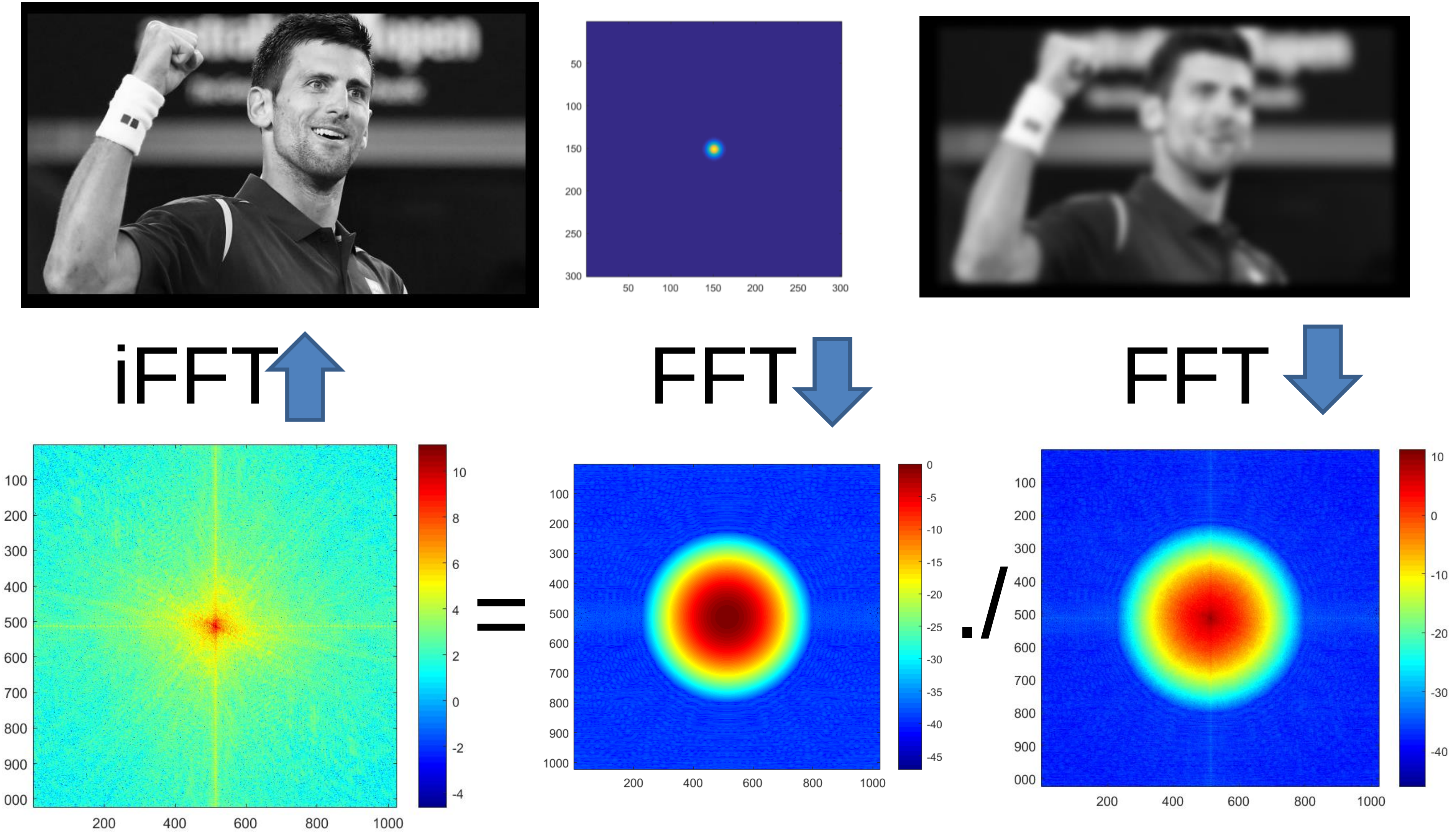


Let’s experiment on Novak

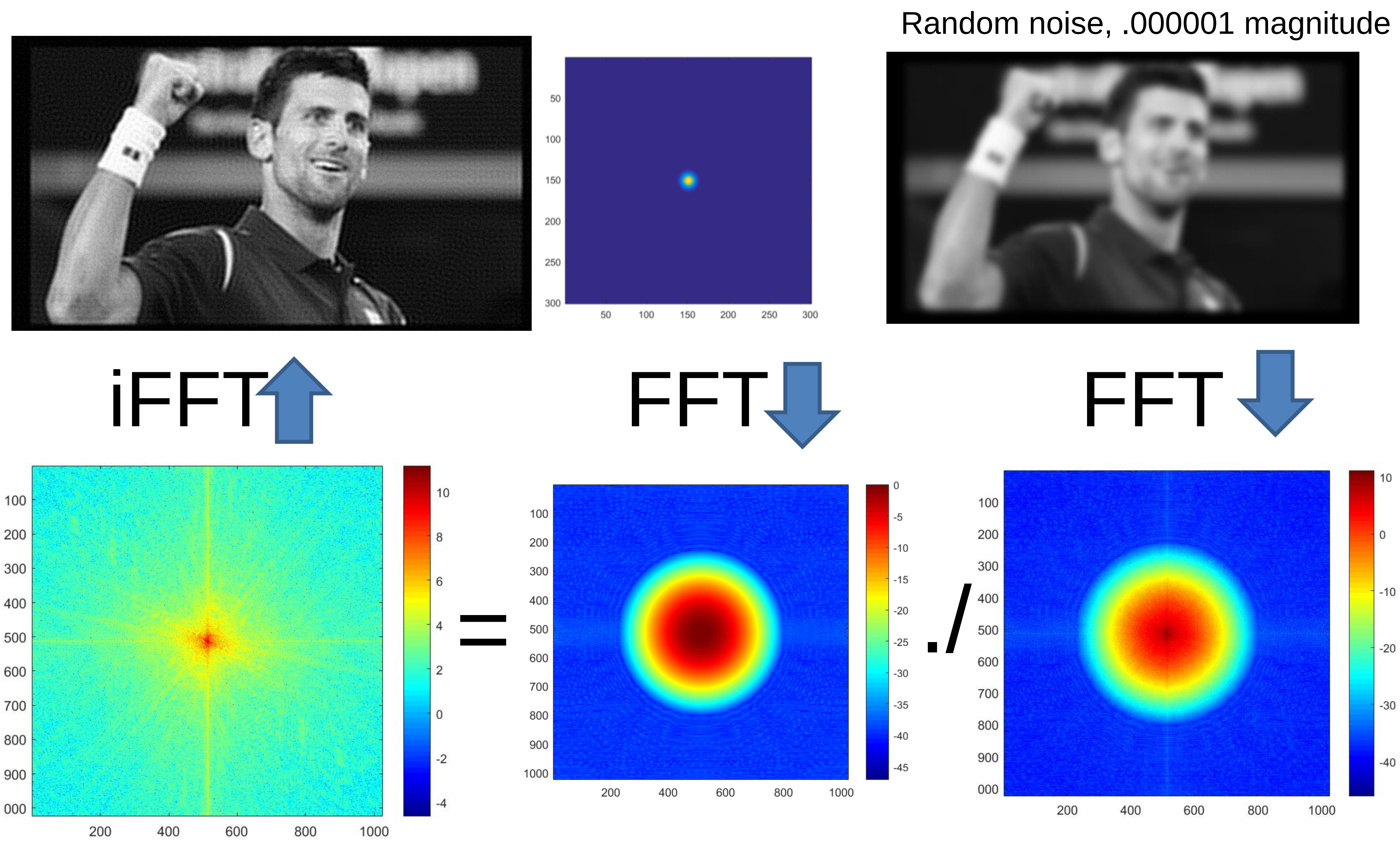
Convolution



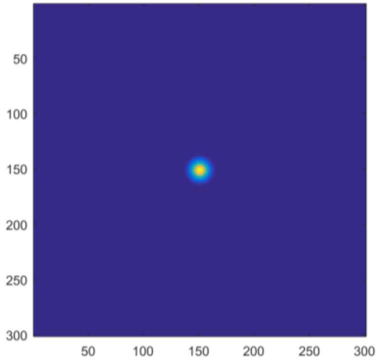
Deconvolution?



But under more realistic conditions



But under more realistic conditions

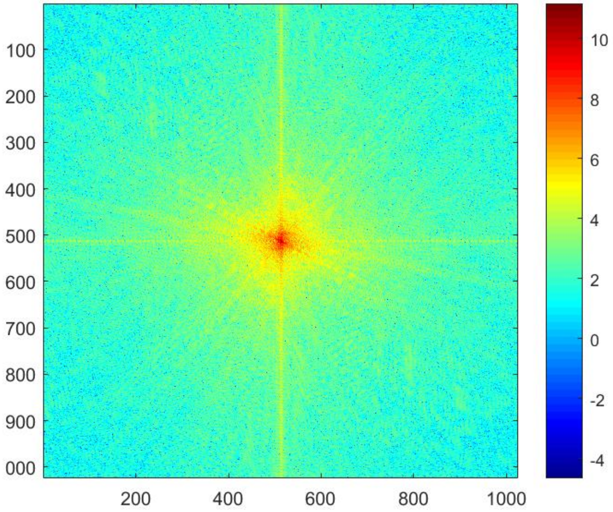


Random noise, .0001 magnitude

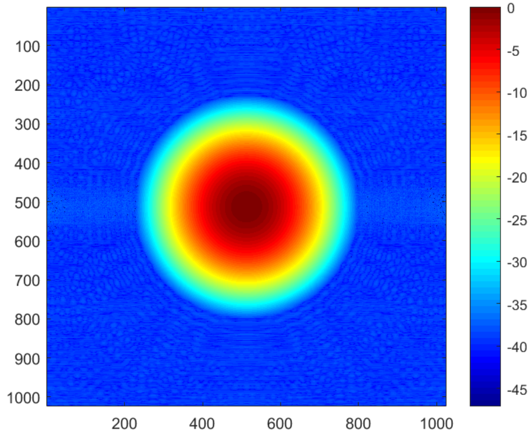
iFFT

FFT

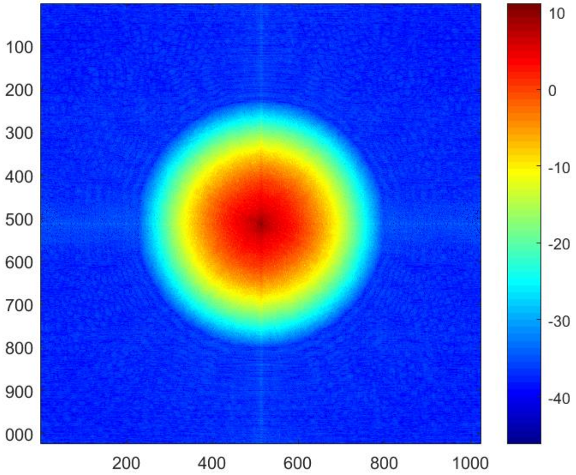
FFT



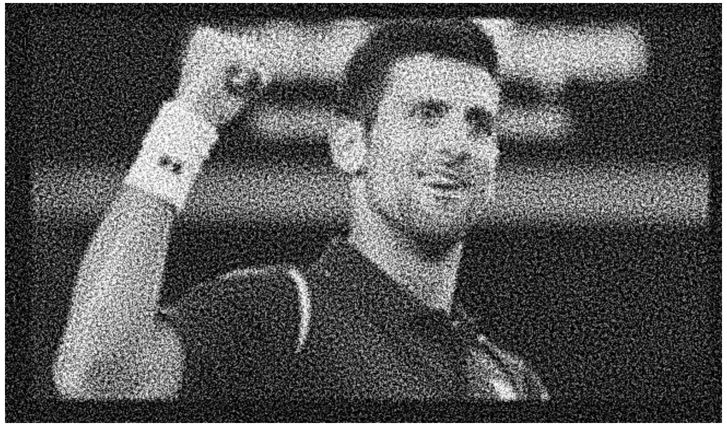
=

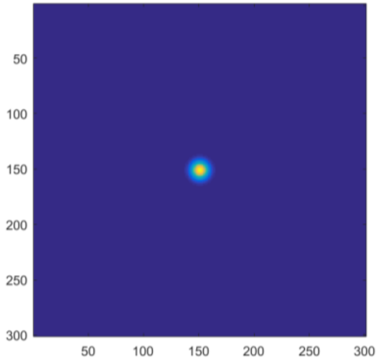


./



But under more realistic conditions



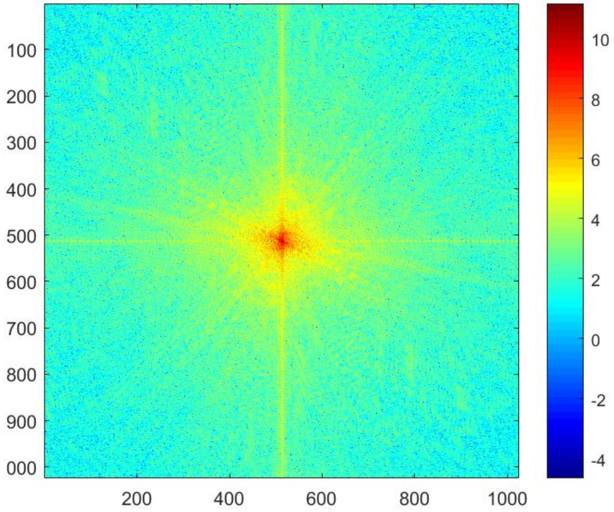


Random noise, .001 magnitude

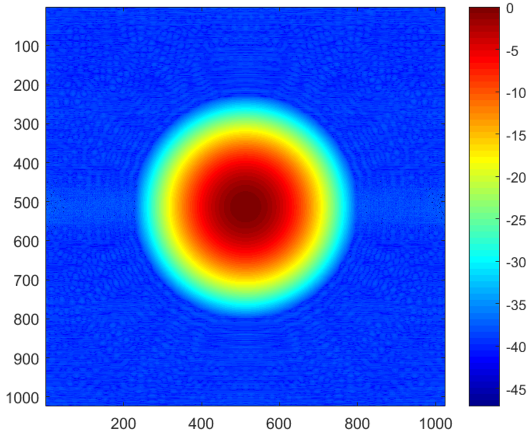
iFFT

FFT

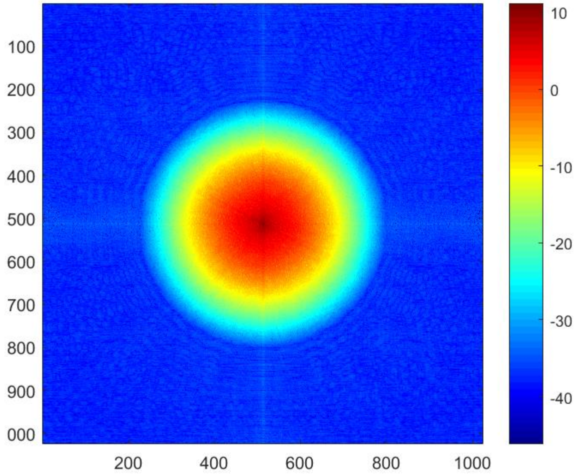
FFT



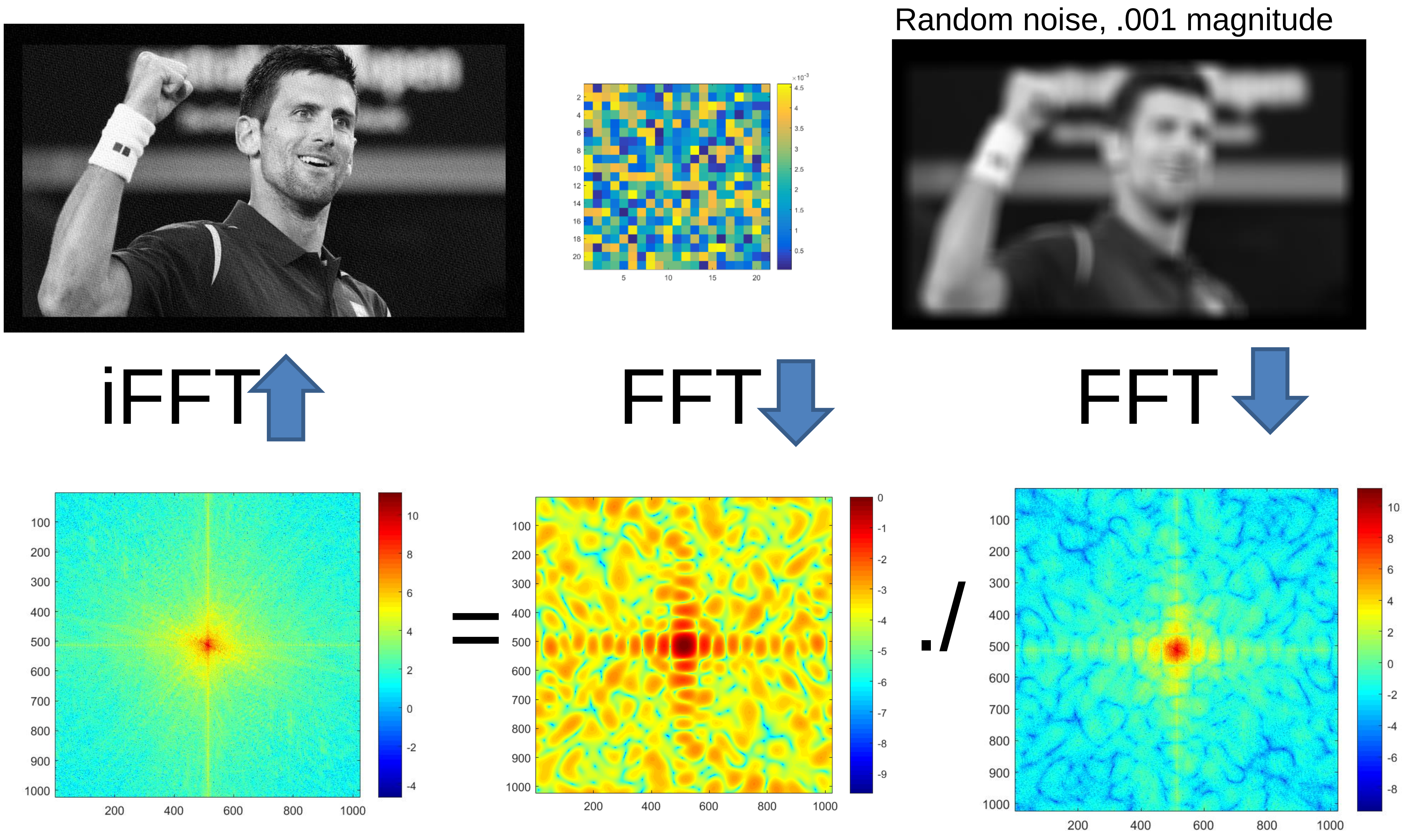
=



./



With a random filter…



Deconvolution is hard

- Active research area.
- Even if you know the filter (non-blind deconvolution), it is still very hard and requires strong *regularization.*
- If you don’t know the filter (blind deconvolution) it is harder still.

Blind Deconvolution Example

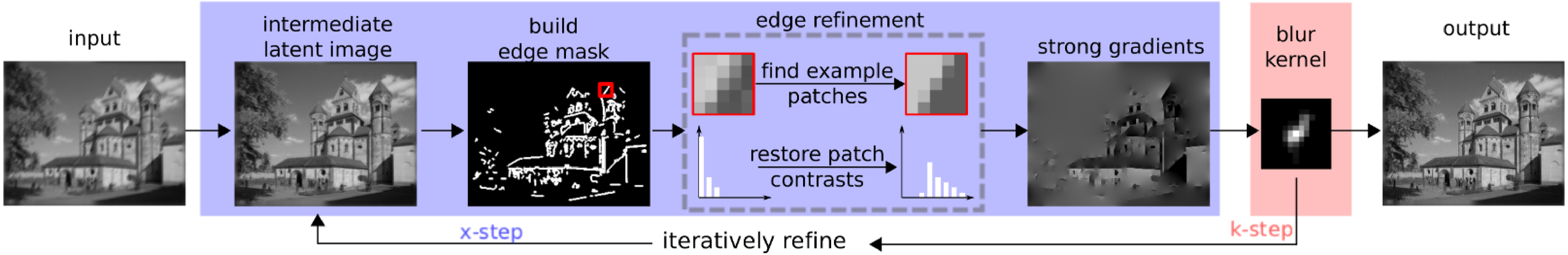
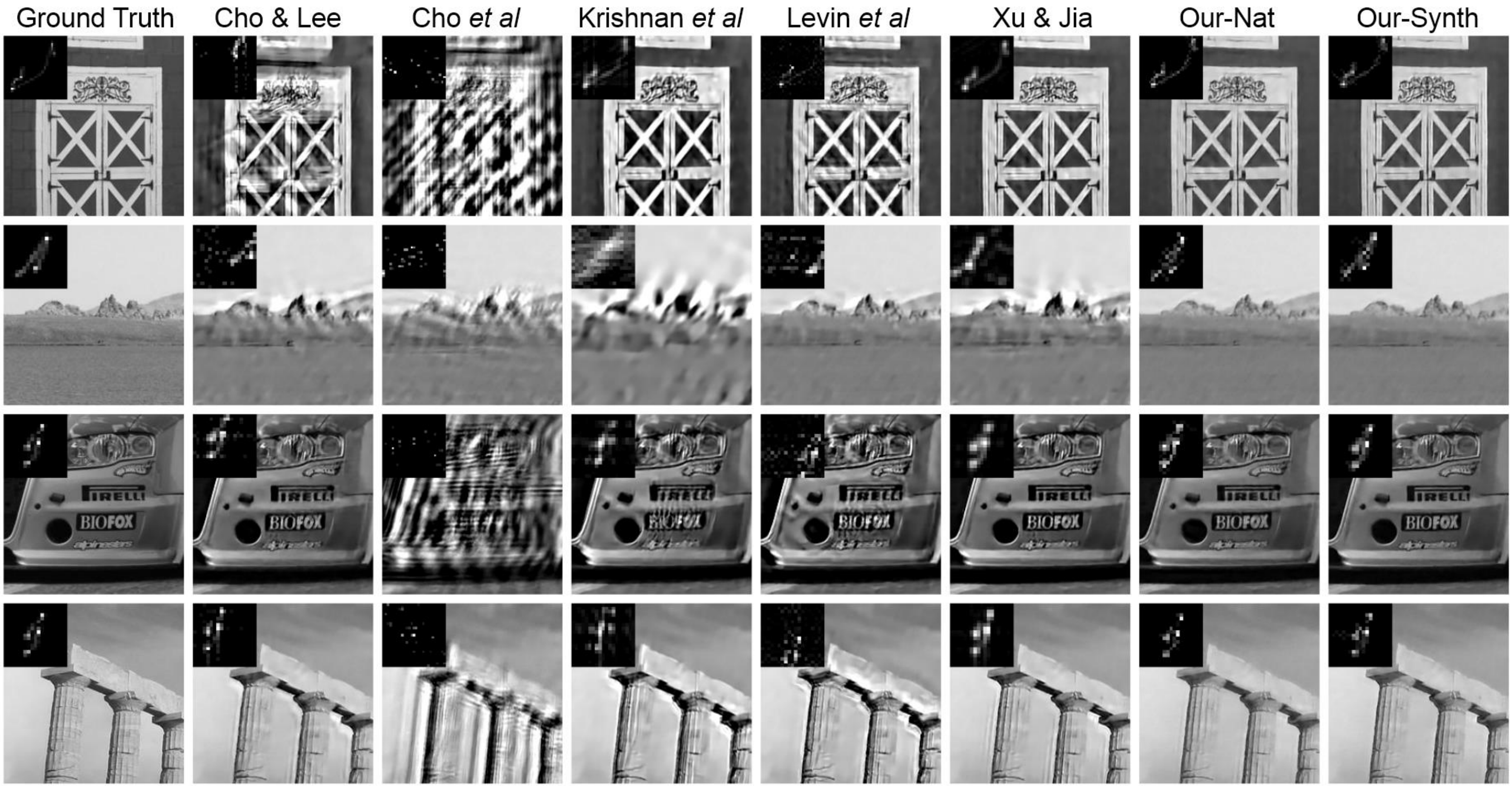


Figure 1. Algorithm pipeline. Our algorithm iterates between x-step and k-step with the help of a patch prior for edge refinement process. In particular, we coerce edges to become sharp and increase local contrast for edge patches. The blur kernel is then updated using the strong gradients from the restored latent image. After kernel estimation, the method of [20] is used for final non-blind deconvolution.

Edge-based Blur Kernel Estimation Using Patch Priors.
Libin Sun, Sunghyun Cho, Jue Wang, and James Hays.
IEEE International Conference on Computational Photography 2013.



Edge-based Blur Kernel Estimation Using Patch Priors.
Libin Sun, Sunghyun Cho, Jue Wang, and James Hays.
IEEE International Conference on Computational Photography 2013.