photo by unsplash user @evablue

COMP527

COMPUTATIONAL IMAGING

Lecture #08 – Deconvolution and Coded Photography

KOÇ UNIVERSITY

Aykut Erdem // Koç University // Spring 2023

Previously on COMP527

- Quick reminder: pinhole vs lens cameras
- Focal stack
- Confocal stereo
- Lightfield
- Measuring lightfields
- Plenoptic camera
- Images from lightfields
- Some notes on (auto-)focusing

photo by unsplash user @ekamelev

Today’s Lecture

- Deconvolution
  - Sources of blur
  - Blind deconvolution
  - Non-blind deconvolution
- Coded photography
  - The coded photography paradigm
  - Dealing with depth blur
  - Dealing with motion blur

Disclaimer: The material and slides for this lecture were borrowed from
—Ioannis Gkioulekas’ 15-463/15-663/15-862 “Computational Photography” class
—Seungyong Lee and Sunghyun Cho’s “Recent Advances in Image Deblurring” course at SIGGRAPH Asia 2013

Today's Lecture

- Deconvolution
  - Sources of blur
  - Blind deconvolution
  - Non-blind deconvolution
- Coded photography
  - The coded photography paradigm
  - Dealing with depth blur
  - Dealing with motion blur

Sources of Blur

blur [bl3:(r)]

- Long exposure
- Moving objects
- Camera motion
  - panning shot
- Lens imperfections
- Depth defocus

blur [bl3:(r)]

- Often degrades image/video quality severely
- Unavoidable under dim light circumstances

Various Kinds of Blurs

Camera shake (Camera motion blur)

Object movement (Object motion blur)

Out of focus (Defocus blur)

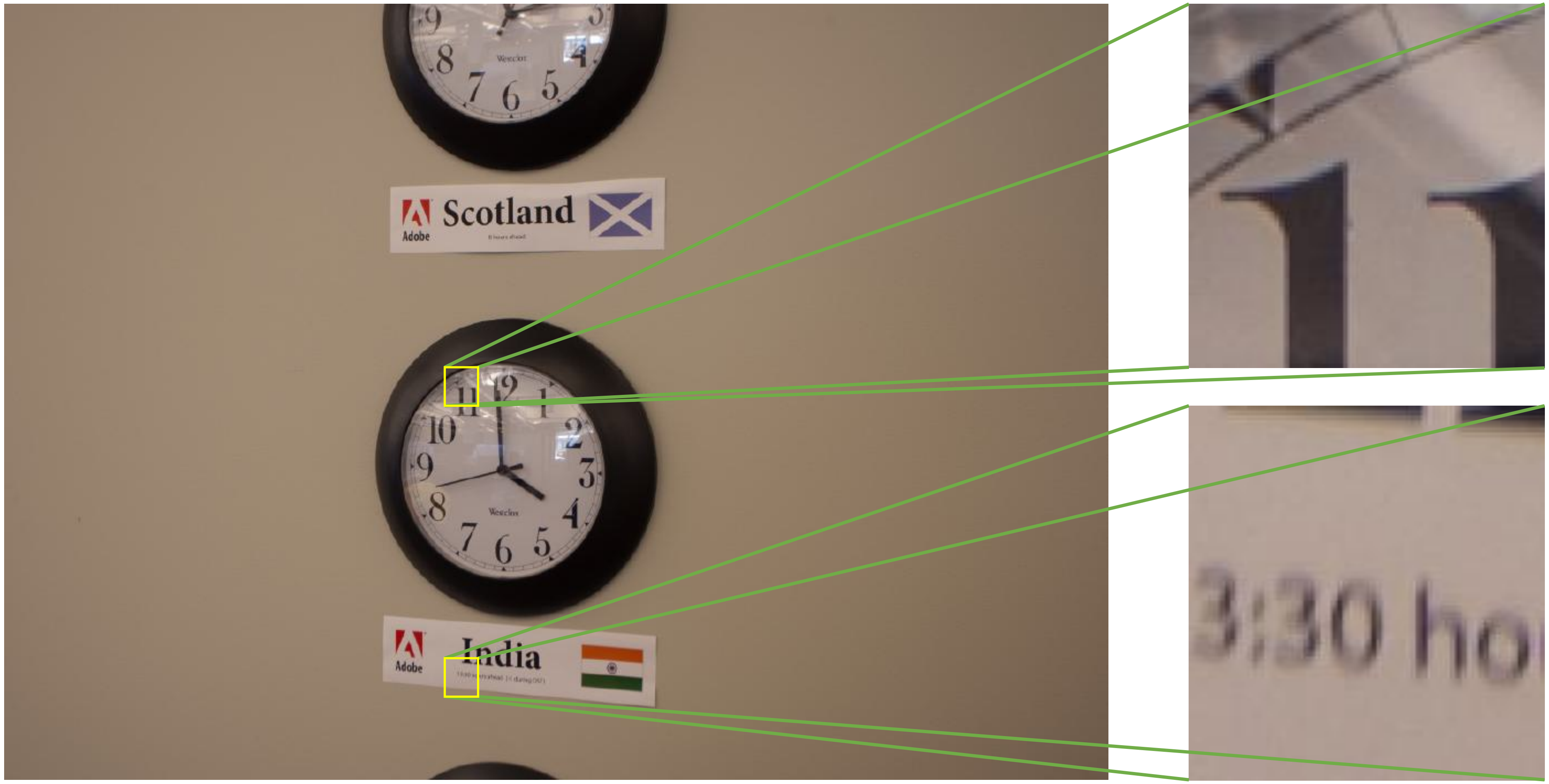
Combinations (vibration & motion, …)

Object Motion Blur

- Caused by object motions during exposure time

Optical Lens Blur

- Caused by lens aberration

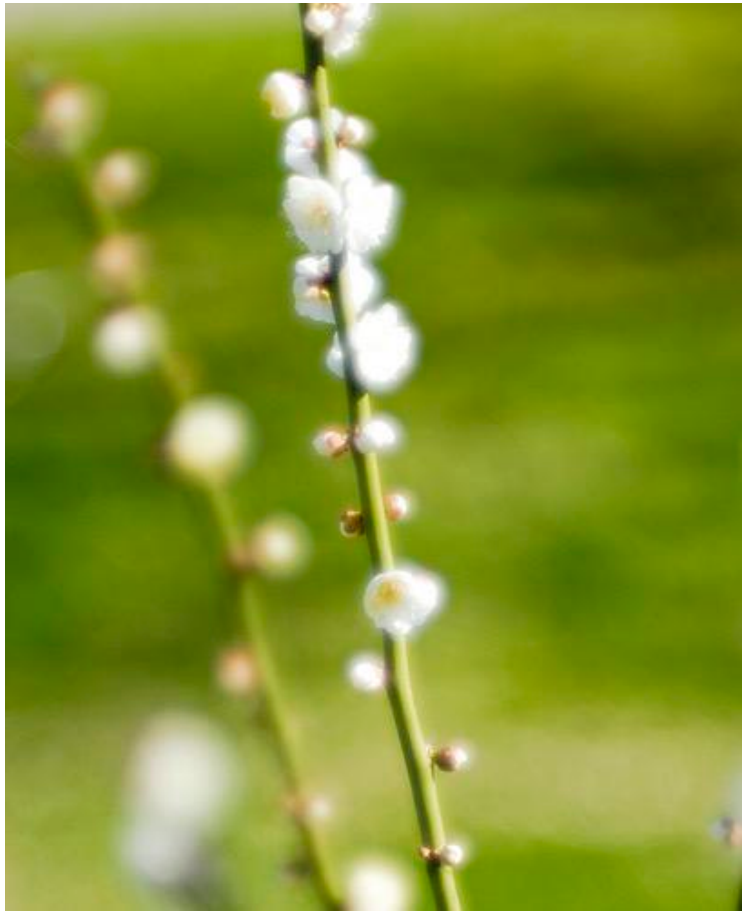


Camera Motion Blur

- Caused by camera shakes during exposure time
  - Motion can be represented as a camera trajectory

Defocus Blur

- Caused by the limited depth of field of a camera



More on coded photography part

Deblurring?

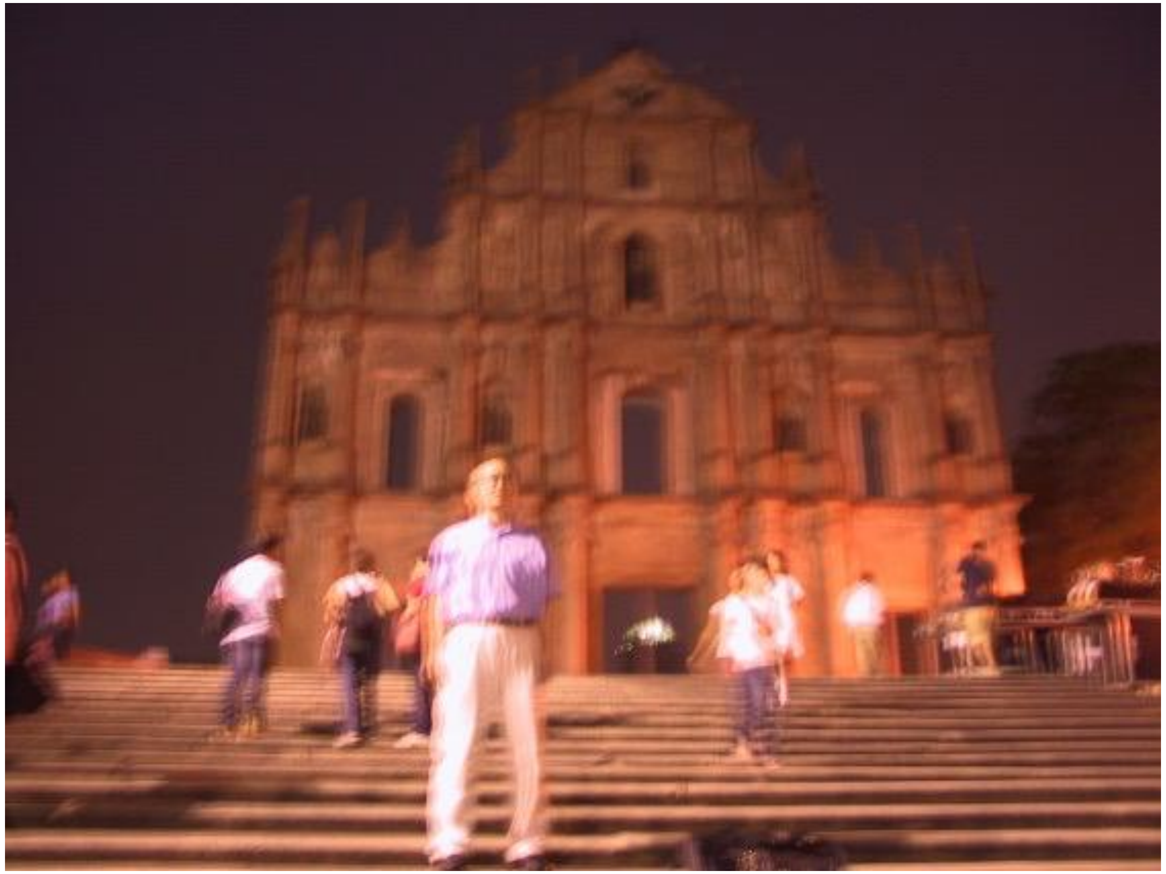
- Remove blur and restore a latent sharp image

from a given blurred image

find its latent sharp image

Why is it important?

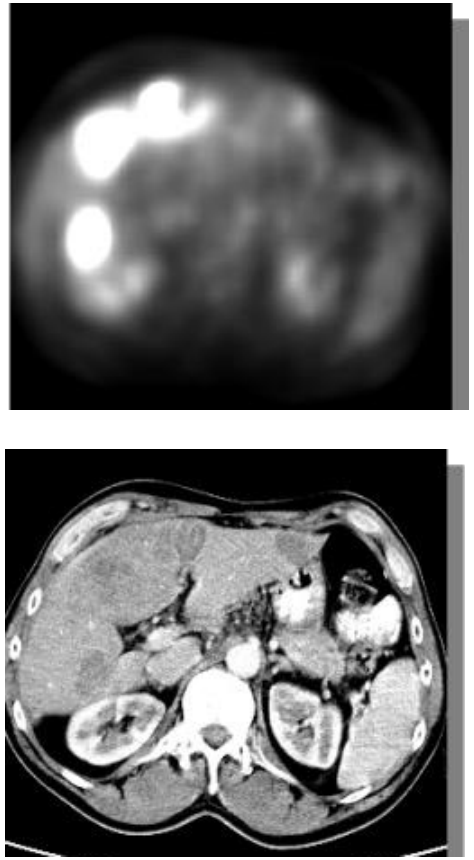
- Image/video in our daily lives
  - Sometimes a retake is difficult!



Why is it important?

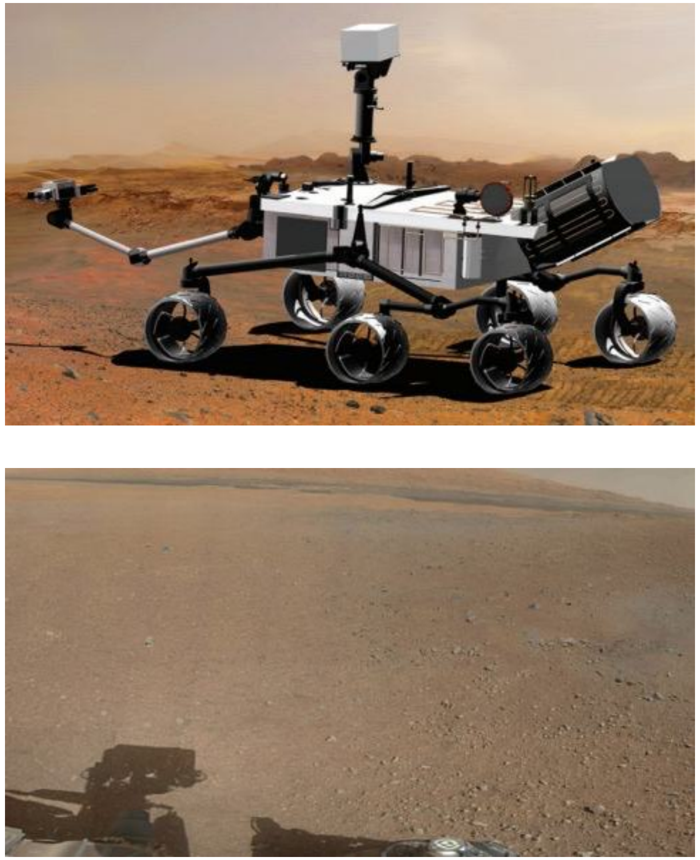
- Strong demand for high quality deblurring

CCTV, car black box



Medical imaging

Aerial/satellite photography



Robot vision

Deblurring

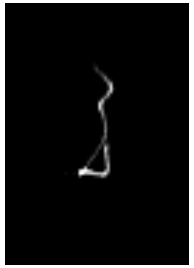
from a given blurred image

find its latent sharp image

Commonly Used Blur Model

Blurred image

$=$



$*$

Latent sharp image

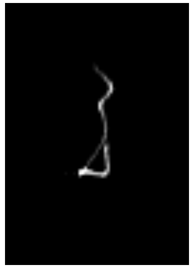
Blur kernel or
Point Spread
Function (PSF)

Convolution
operator

Blind Deconvolution

Blurred image

=



Blur kernel or Point Spread Function (PSF)

*

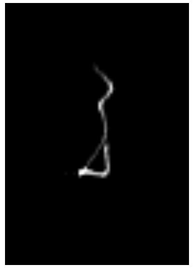
Convolution operator

Latent sharp image

Non-blind Deconvolution

Blurred image

=



*

Latent sharp image

Blur kernel or Point Spread Function (PSF)

Convolution operator

Uniform vs. Non-uniform Blur

Uniform blur

- Every pixel is blurred in the same way
- Convolution based blur model

Uniform vs. Non-uniform Blur

Non-uniform blur

- Spatially-varying blur
- Pixels are blurred differently
- More faithful to real camera shakes

Most Blurs Are Non-Uniform

Camera shake (Camera motion blur)

Object movement (Object motion blur)

Out of focus (Defocus blur)

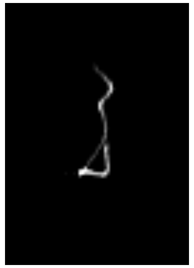
Combinations (vibration & motion, …)

Blind Deconvolution

Blind Deconvolution (Uniform Blur)

Blurred image

=



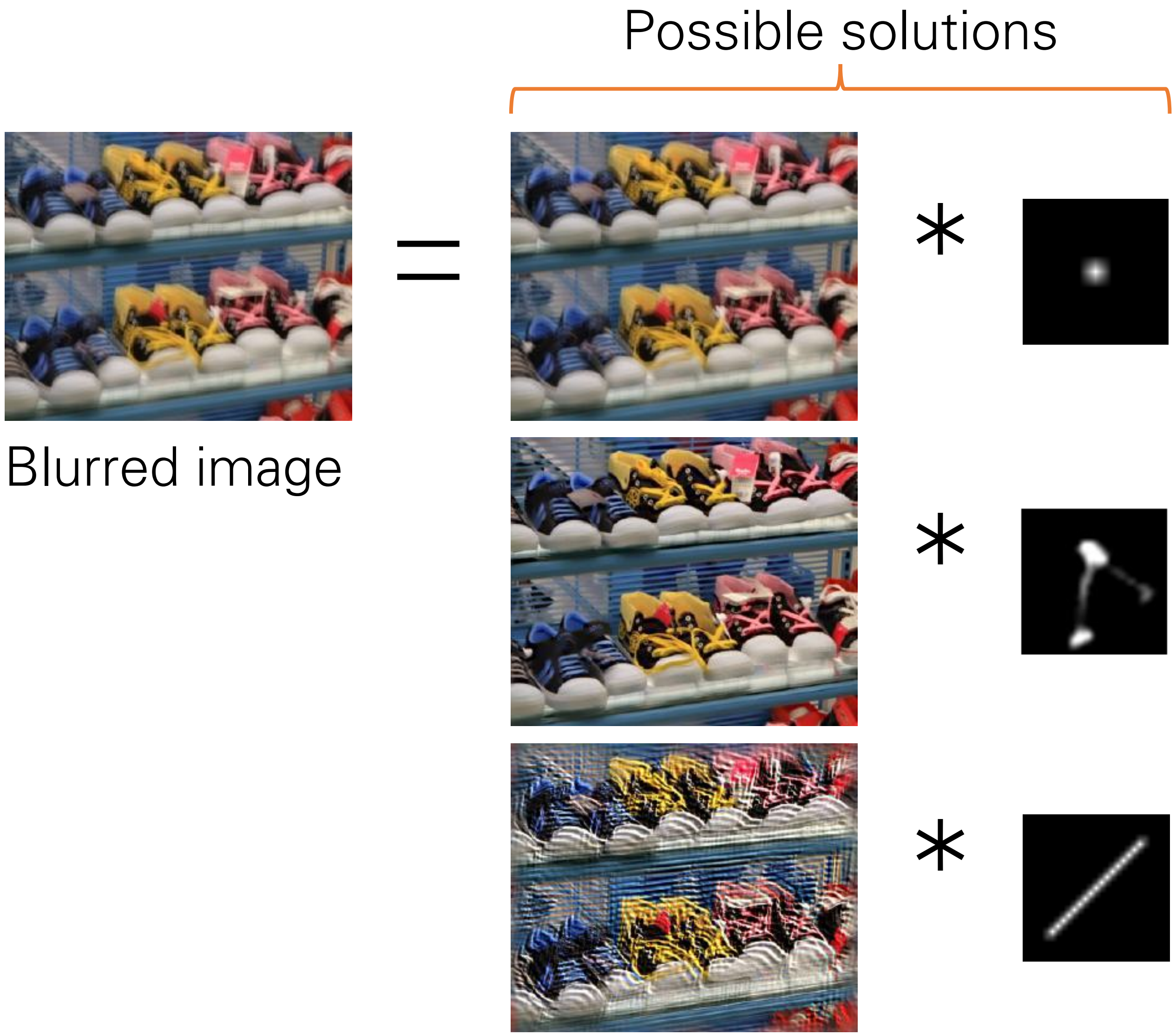
Blur kernel or Point Spread Function (PSF)

*

Convolution operator

Latent sharp image

Key challenge: Ill-posedness!



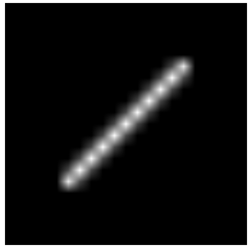
- Infinite number of solutions satisfy the blur model
- Analogous to

$$100 = \begin{cases} 2\times 50 \\ 4\times 25 \\ 3\times 33.333\ldots \end{cases}$$

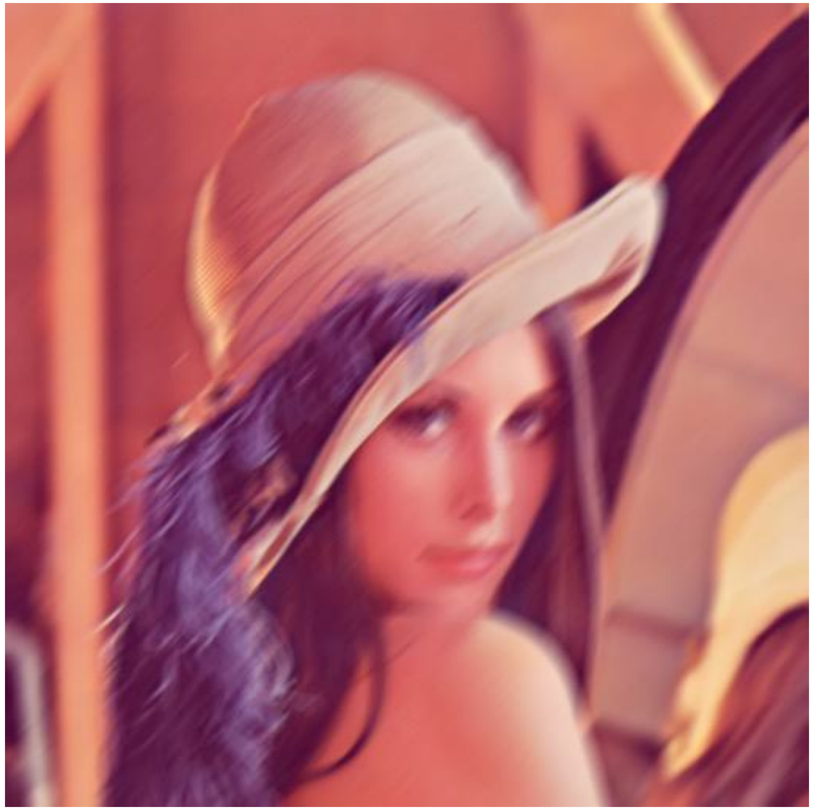
Early approaches

- Parametric blur kernels
  - [Yitzhakey et al. 1998], [Rav-Acha and Peleg 2005], …
  - Directional blur kernels defined by (length, angle)

*

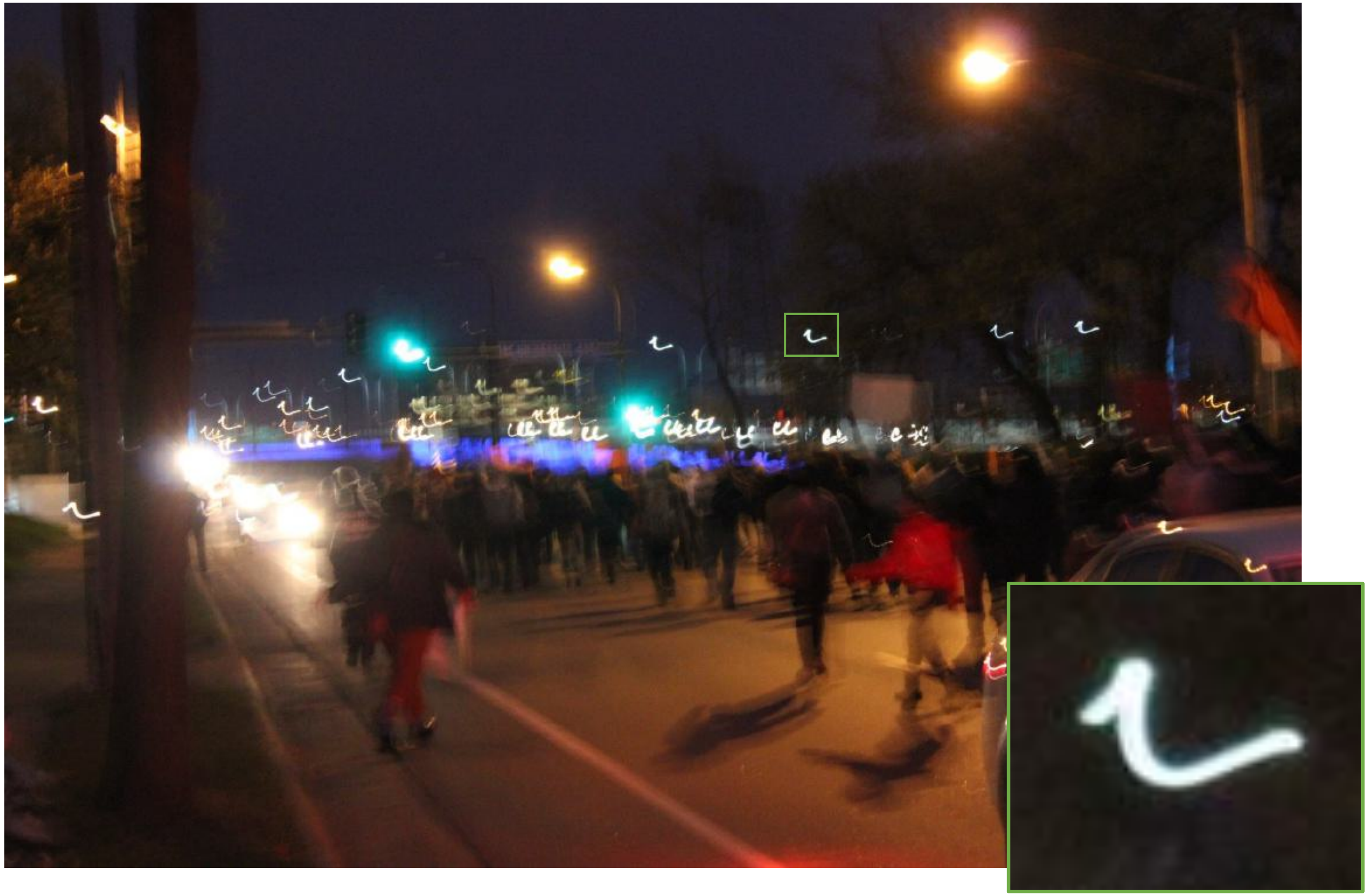


→



Early approaches

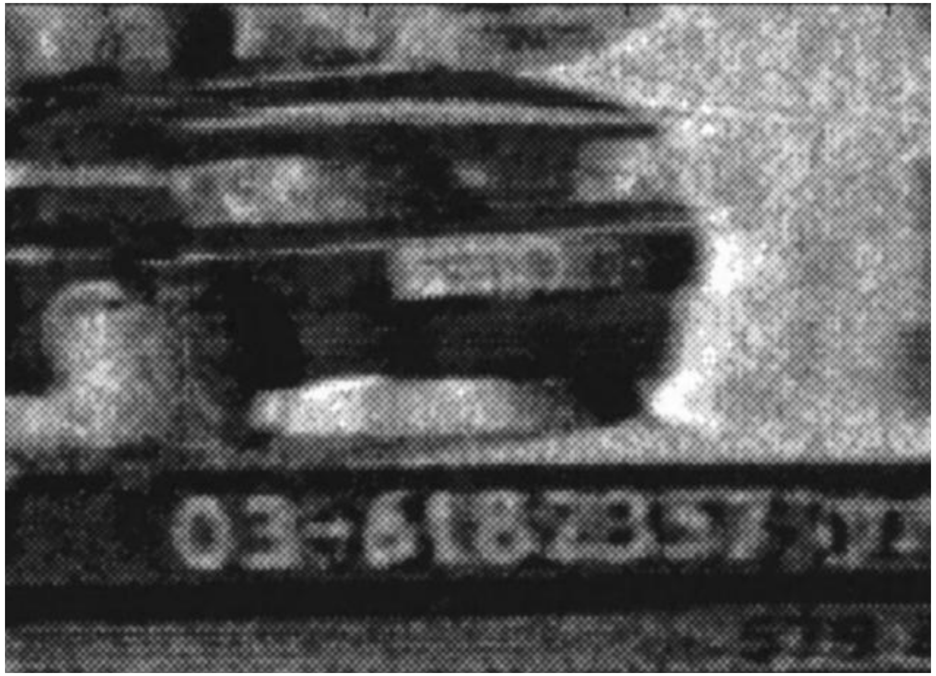
- But real camera shakes are much more complex



Early approaches

- Parametric blur kernels
  - Very restrictive assumption
  - Often failed, poor quality

Blurred image

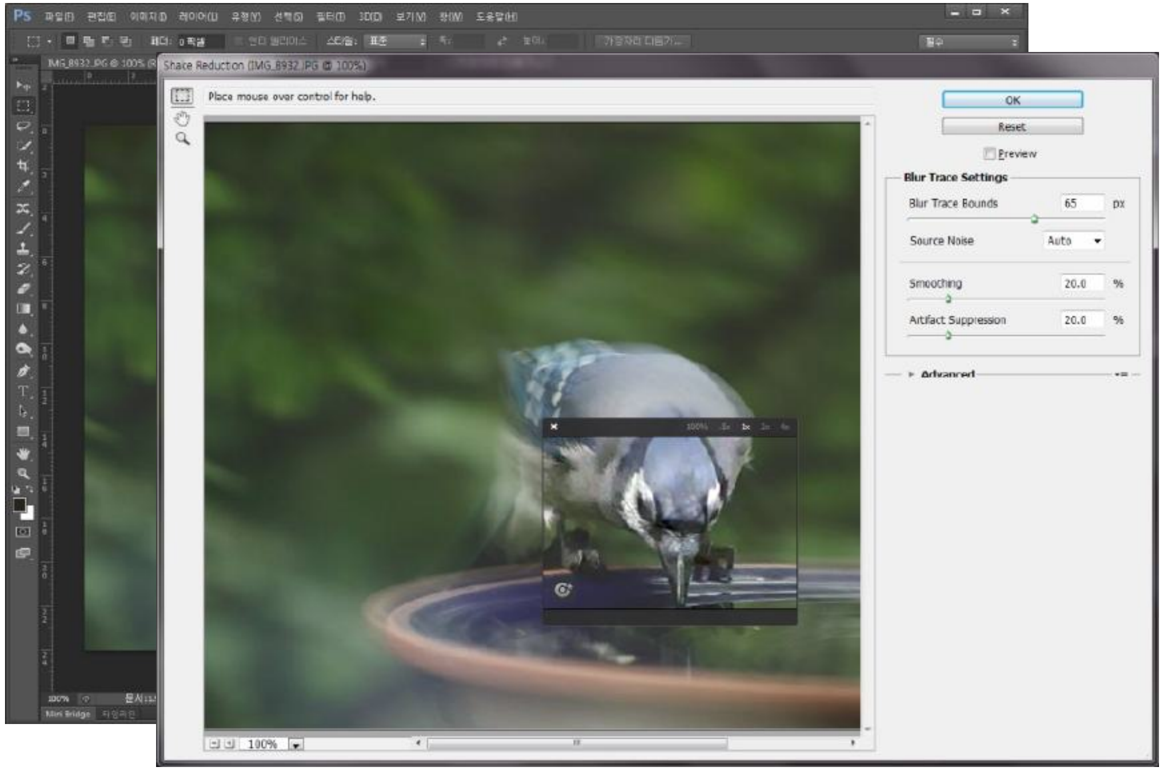


Latent sharp image

Yitzhaky et al. 1998]

More recent work

- Some successful approaches have been introduced…
  - [Fergus et al. SIGGRAPH 2006], [Shan et al. SIGGRAPH 2008], [Cho and Lee, SIGGRAPH Asia 2009], …
  - More realistic blur kernels
  - Better quality
  - More robust



- Commercial software
  - Photoshop CC Shake reduction

Popular Approaches

- Maximum Posterior (MAP) based
- Variational Bayesian based
- Edge Prediction based

Popular Approaches

- Maximum Posterior (MAP) based
- Variational Bayesian based
- Edge Prediction based
- Deep-Learning based (not now, later on)

Popular Approaches

- Maximum Posterior (MAP) based
- Variational Bayesian based
- Edge Prediction based

- [Shan et al. SIGGRAPH 2008], [Krishnan et al. CVPR 2011], [Xu et al. CVPR 2013], …
- Seek the most probable solution, which maximizes a posterior distribution
- Easy to understand
- Convergence problem

MAP based Approaches

Maximize a joint posterior probability with respect to k and l

Posterior distribution

$$p(k, l \mid b)$$

Blur kernel k

Latent image l

Blurred image b

MAP based Approaches

Bayes rule:

Posterior distribution Likelihood Prior on l Prior on k

$$p(k, l \mid b) \propto p(b \mid l, k)\, p(l)\, p(k)$$

Blur kernel k

Latent image l

Blurred image b

MAP based Approaches

Negative log-posterior:

$$-\log p(k, l|b) \Rightarrow -\log p(b|k, l) - \log p(l) - \log p(k)$$

$$\Rightarrow \underbrace{\|k * l - b\|^2}_{\text{Data fitting term}} + \underbrace{\rho_l(l)}_{\text{Regularization on latent image } l} + \underbrace{\rho_k(k)}_{\text{Regularization on blur kernel } k}$$

MAP based Approaches

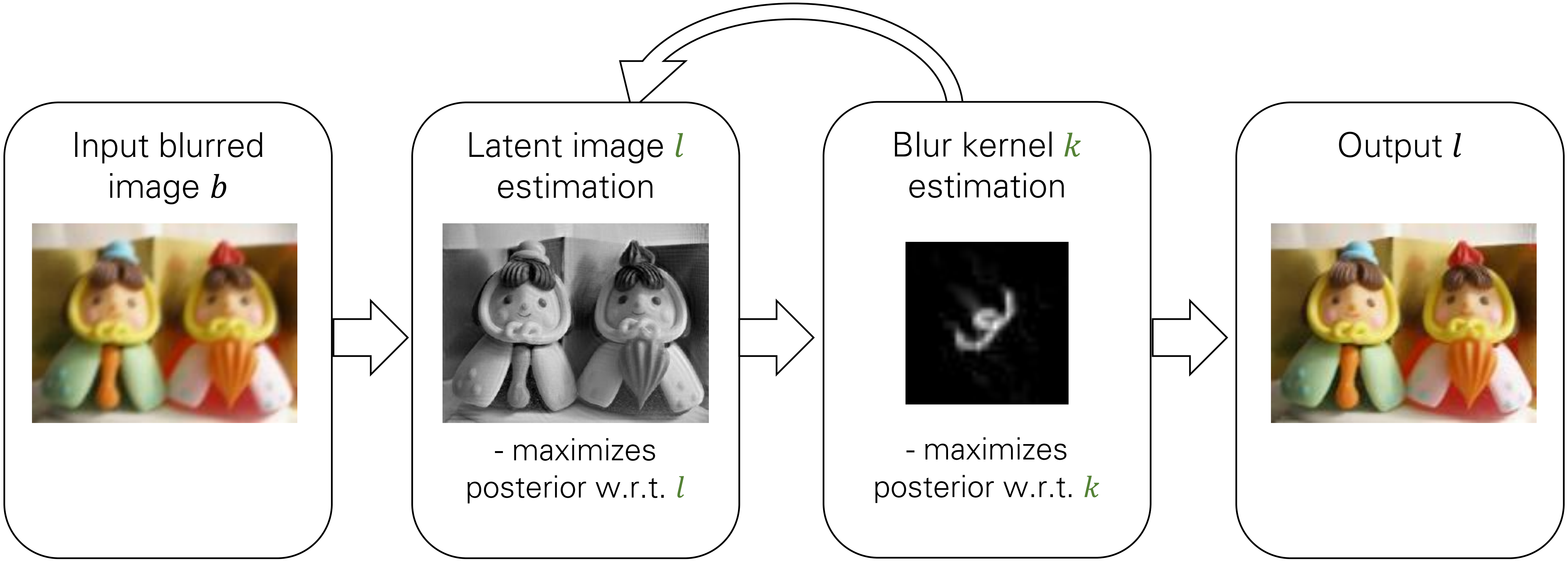
Negative log-posterior:

$$-\log p(k,l|b) \Rightarrow -\log p(b|k,l) - \log p(l) - \log p(k)$$

$$\Rightarrow \underbrace{\|k * l - b\|^2}_{\text{Data fitting term}} + \underbrace{\rho_l(l)}_{\text{Regularization on latent image } l} + \underbrace{\rho_k(k)}_{\text{Regularization on blur kernel } k}$$

Alternatingly minimize the energy function w.r.t. k and l

MAP based Approaches



MAP based Approaches

- Chan and Wong, TIP 1998
  - Total variation based priors for estimating a parametric blur kernel
- Shan et al. SIGGRAPH 2008
  - First MAP based method to estimate a nonparametric blur kernel
- Krishnan et al. CVPR 2011
  - Normalized sparsity measure, a novel prior on latent images
- Xu et al. CVPR 2013
  - L0 norm based prior on latent images

Shan et al. SIGGRAPH 2008

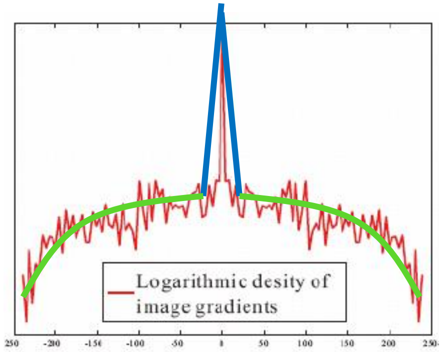
- Carefully designed likelihood & priors

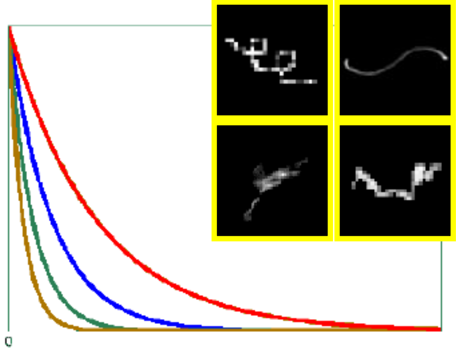
$$p(k,l|b) \propto p(b|l,k)p(l)p(k)$$

Likelihood based on intensities & derivatives

Natural image statistics based prior on l

Kernel statistics based prior on k





Shan et al. SIGGRAPH 2008

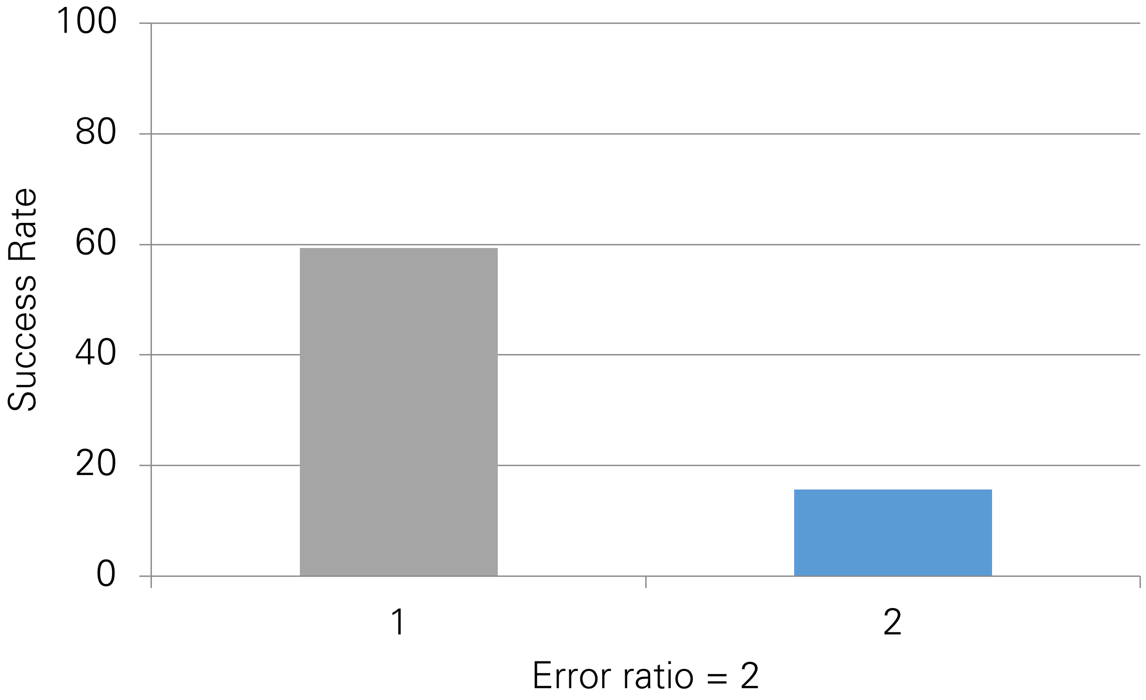
- A few minutes for a small image
- High-quality results

Shan et al. SIGGRAPH 2008

- Convergence problem
  - Often converge to the no-blur solution [Levin et al. CVPR 2009]
  - Natural image priors prefer blurry images

Shan et al. SIGGRAPH 2008

Fergus et al. SIGGRAPH 2006
(variational Bayesian based)



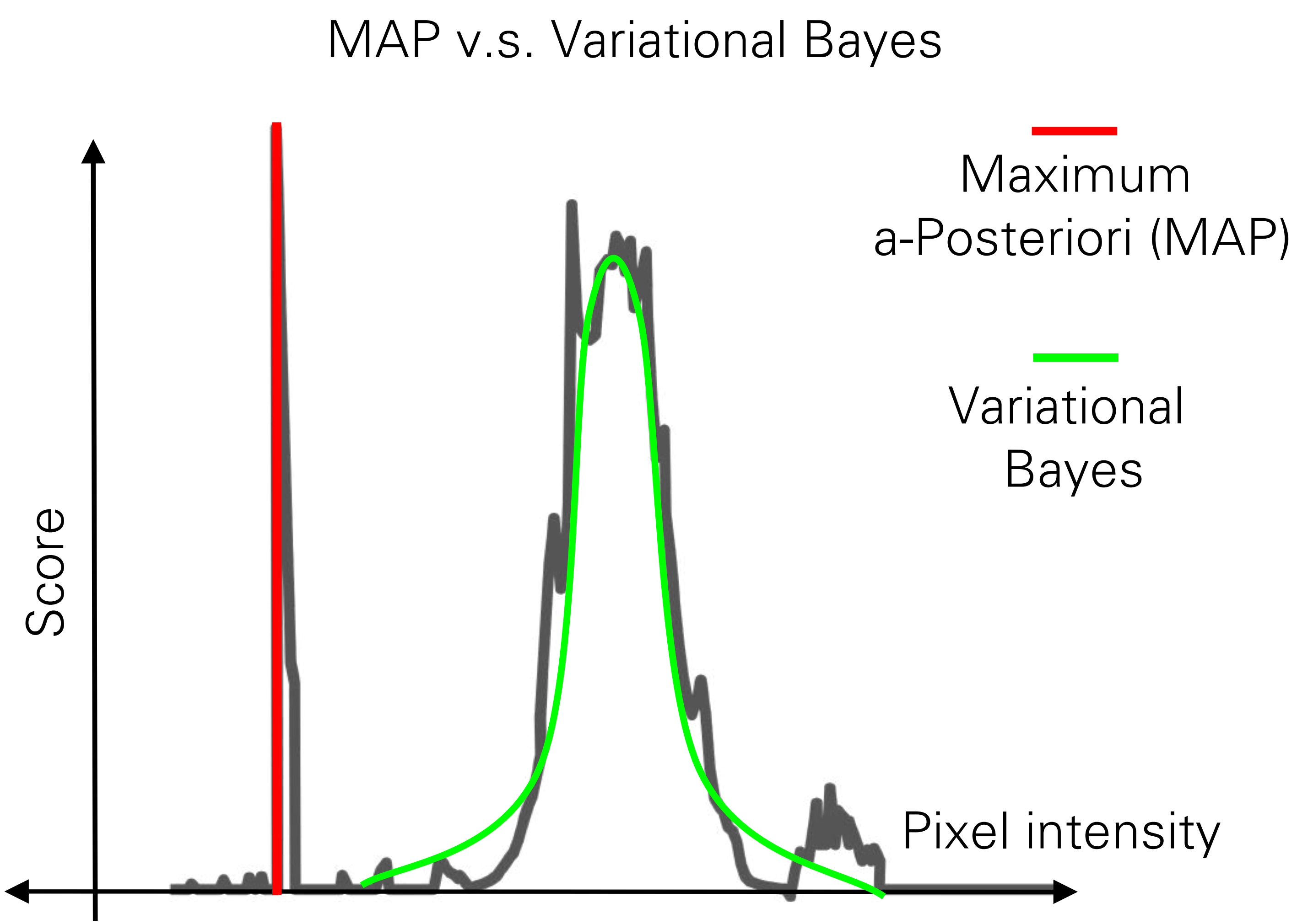
Popular Approaches

- Maximum Posterior (MAP) based
- Variational Bayesian based
- Edge Prediction based

- [Fergus et al. SIGGRAPH 2006],
  [Levin et al. CVPR 2009],
  [Levin et al. CVPR 2011], …

- Not seek for one most probable solution, but consider all possible solutions
- Theoretically more robust
- Slow

Variational Bayesian



- MAP
  - Find the most probable solution
  - May converge to a wrong solution
- Variational Bayesian
  - Approximate the underlying distribution and find the mean
  - More stable
  - Slower

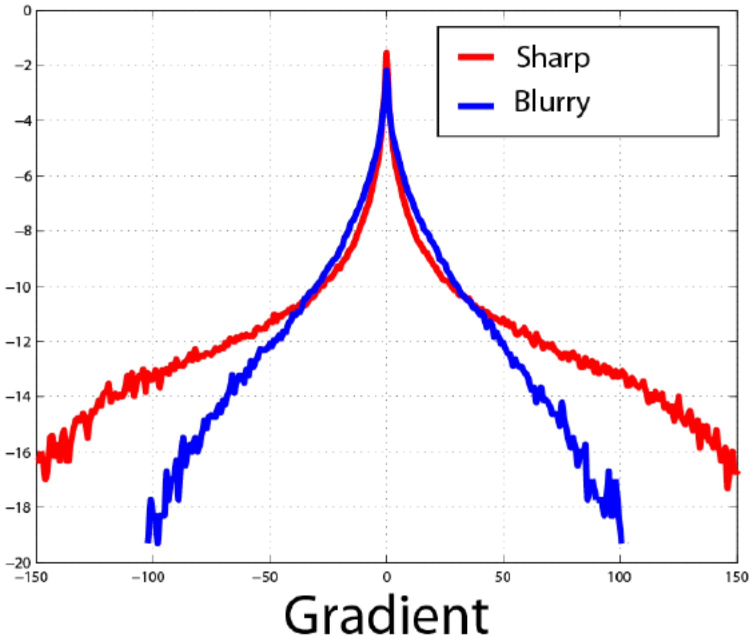
Variational Bayesian

- Fergus et al. SIGGRAPH 2006
  - First approach to handle non-parametric blur kernels
- Levin et al. CVPR 2009
  - Show that variational Bayesian approaches can perform more robustly than MAP based approaches
- Levin et al. CVPR 2010
  - EM based efficient approximation to variational Bayesian approach

Fergus et al. SIGGRAPH 2006

- Posterior distribution

$$p(k, l|\mathrm{b}) \propto p(b|k, l)p(l)p(k)$$



Fergus et al. SIGGRAPH 2006

- Find an approximate distribution by minimizing Kullback-Leibler (KL) divergence

$$\underset{q(k),q(l),q(\sigma^{-2})}{\arg\min} KL\left(\underbrace{q(k)q(l)q(\sigma^{-2})}_{\text{approximate distributions for blur kernel } k,\ \text{latent image } l,\ \text{and noise variance } \sigma^2}\middle\| p(k,l|b)\right)$$

- cf. MAP based approach:

$$\underset{k,l}{\arg\min}\, p(k,l|\mathrm{b})$$

Fergus et al. SIGGRAPH 2006

- First method to estimate a nonparametric blur kernel
- Complex optimization
- Slow: more than an hour for a small image



Popular Approaches

- Maximum Posterior (MAP) based
- Variational Bayesian based
- Edge Prediction based

- [Cho & Lee. SIGGRAPH Asia 2009],
  [Xu et al. ECCV 2010],
  [Hirsch et al. ICCV 2011], …

- Explicitly try to recover sharp edges using heuristic image filters
- Fast
- Proven to be effective in practice, but hard to analyze because of heuristic steps

Edge Prediction based Approaches

- Joshi et al. CVPR 2008
  - Proposed sharp edge prediction to estimate blur kernels
  - No iterative estimation, limited to small scale blur kernels
- Cho & Lee, SIGGRAPH Asia 2009
  - Proposed sharp edge prediction to estimate large blur kernels
  - Iterative framework, very fast
- Cho et al. CVPR 2010
  - Applied Radon transform to estimate a blur kernel from blurry edge profiles
  - Small scale blur kernels
- Xu et al. ECCV 2010
  - Proposed a prediction scheme based on structure scales as well as gradient magnitudes
- Hirsch et al. ICCV 2011
  - Applied a prediction scheme to estimate spatially-varying camera shakes

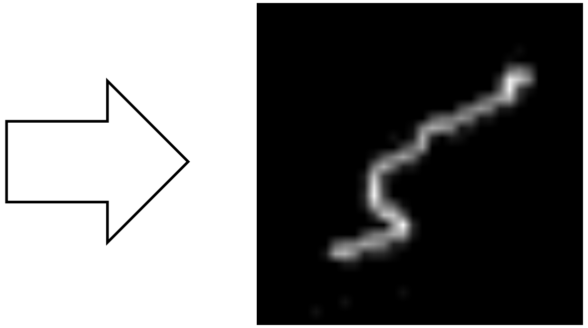
Cho & Lee, SIGGRAPH Asia 2009

- Key idea: blur can be estimated from a few **edges**

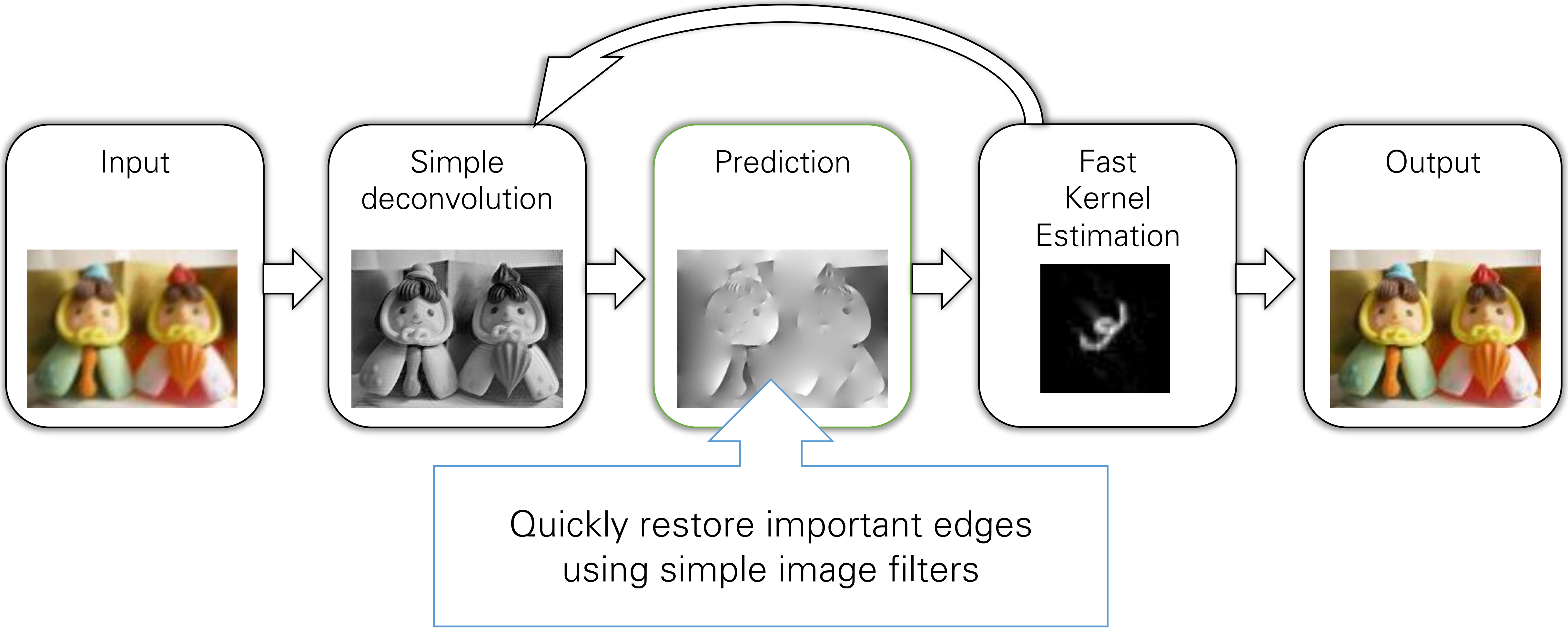
  → No need to restore every detail for kernel estimation

Blurred image

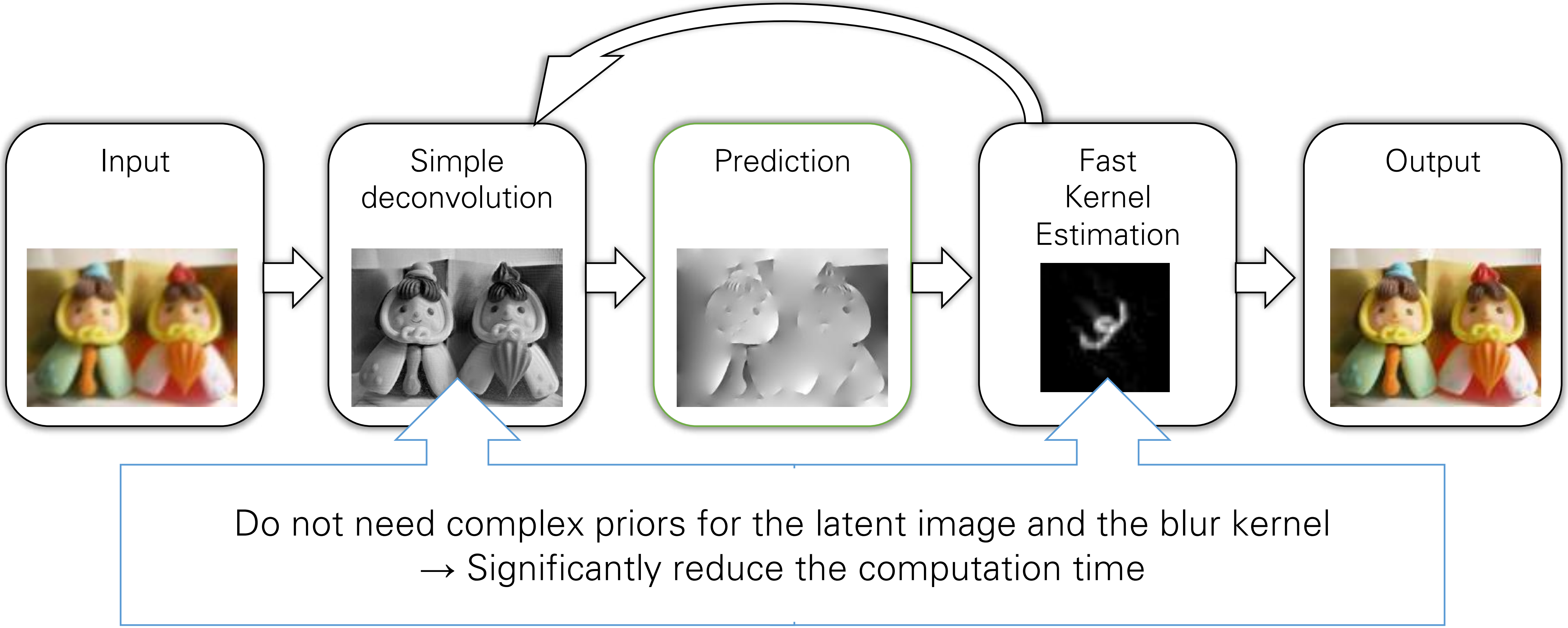
Latent image with only a few edges and no texture



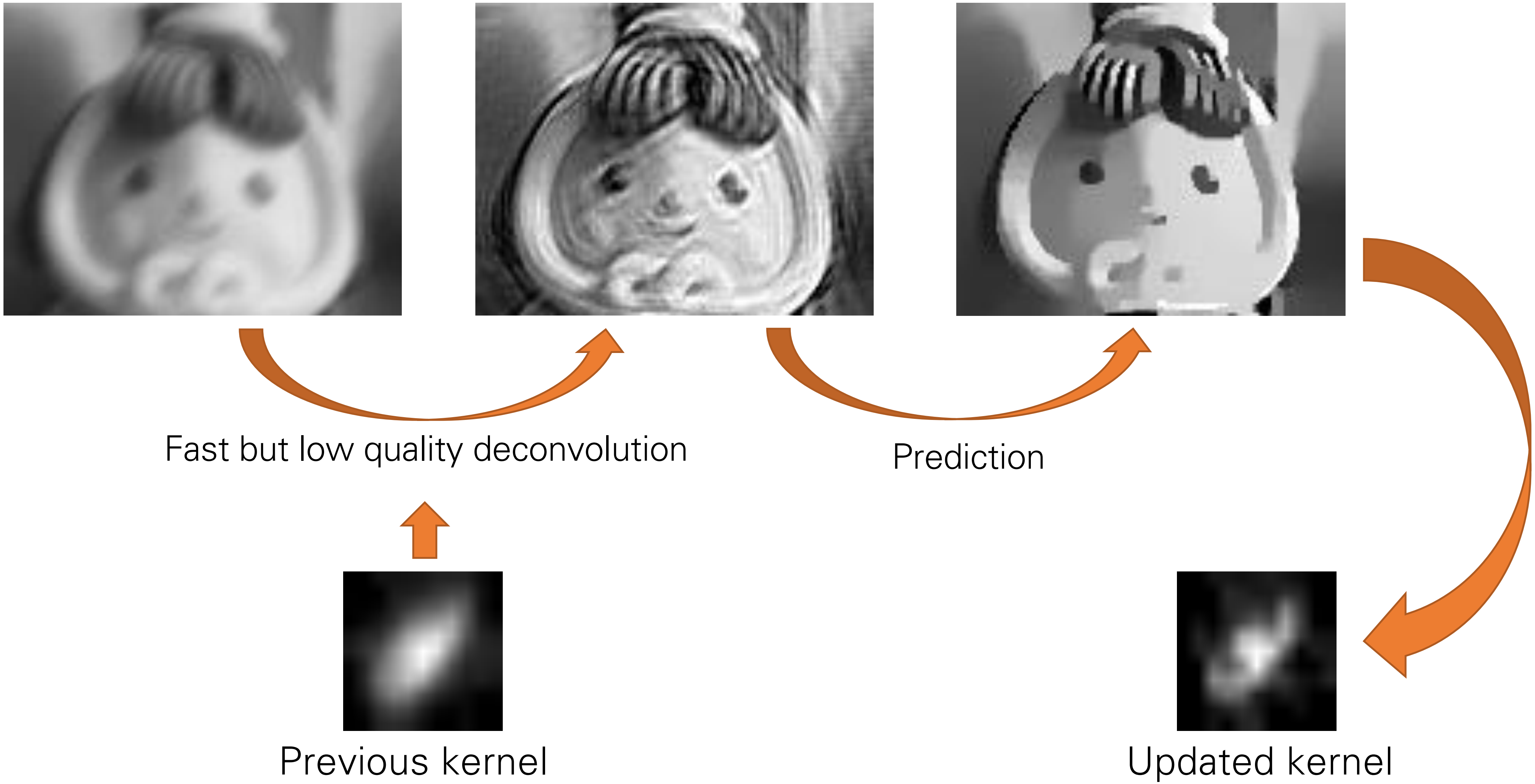
Cho & Lee, SIGGRAPH Asia 2009



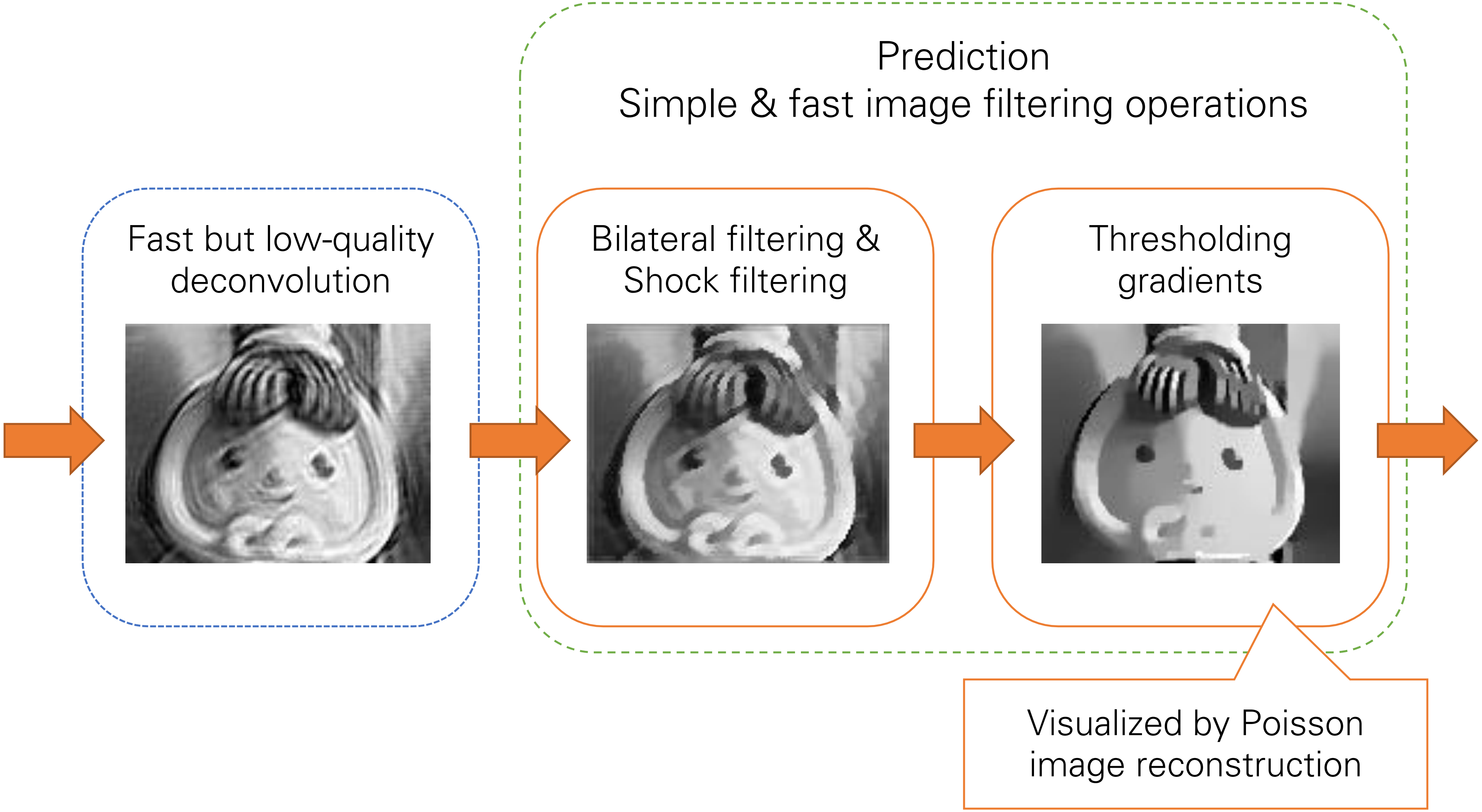
Cho & Lee, SIGGRAPH Asia 2009



Cho & Lee, SIGGRAPH Asia 2009



Cho & Lee, SIGGRAPH Asia 2009



Cho & Lee, SIGGRAPH Asia 2009


Blurry input


Deblurring result

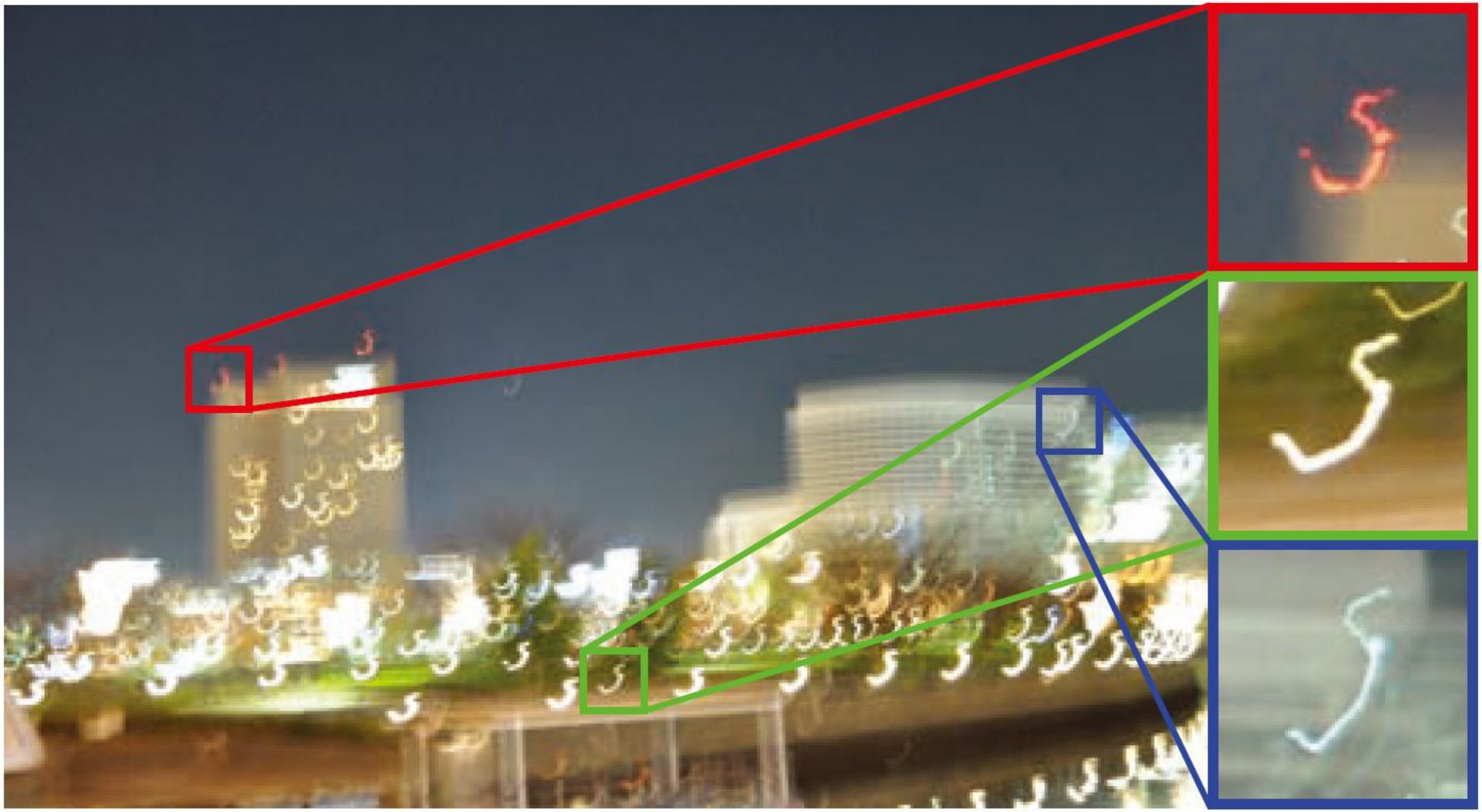
- State of the art results
- A few seconds
- 1Mpix image
- in C++


Blur kernel

Convolution based Blur Model

- Uniform and spatially invariant blur

Real Camera Shakes: Spatially Variant!



Uniform Blur Model Assumes

x & y translational
camera shakes

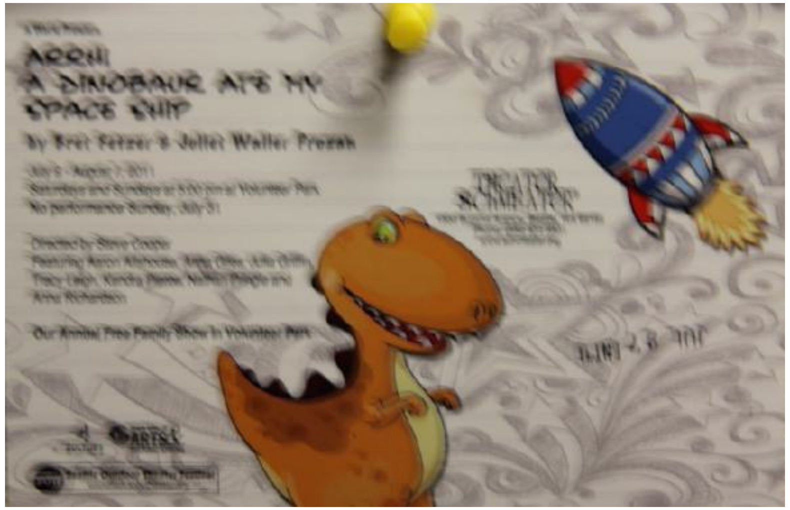
Planar scene

Real Camera Shakes

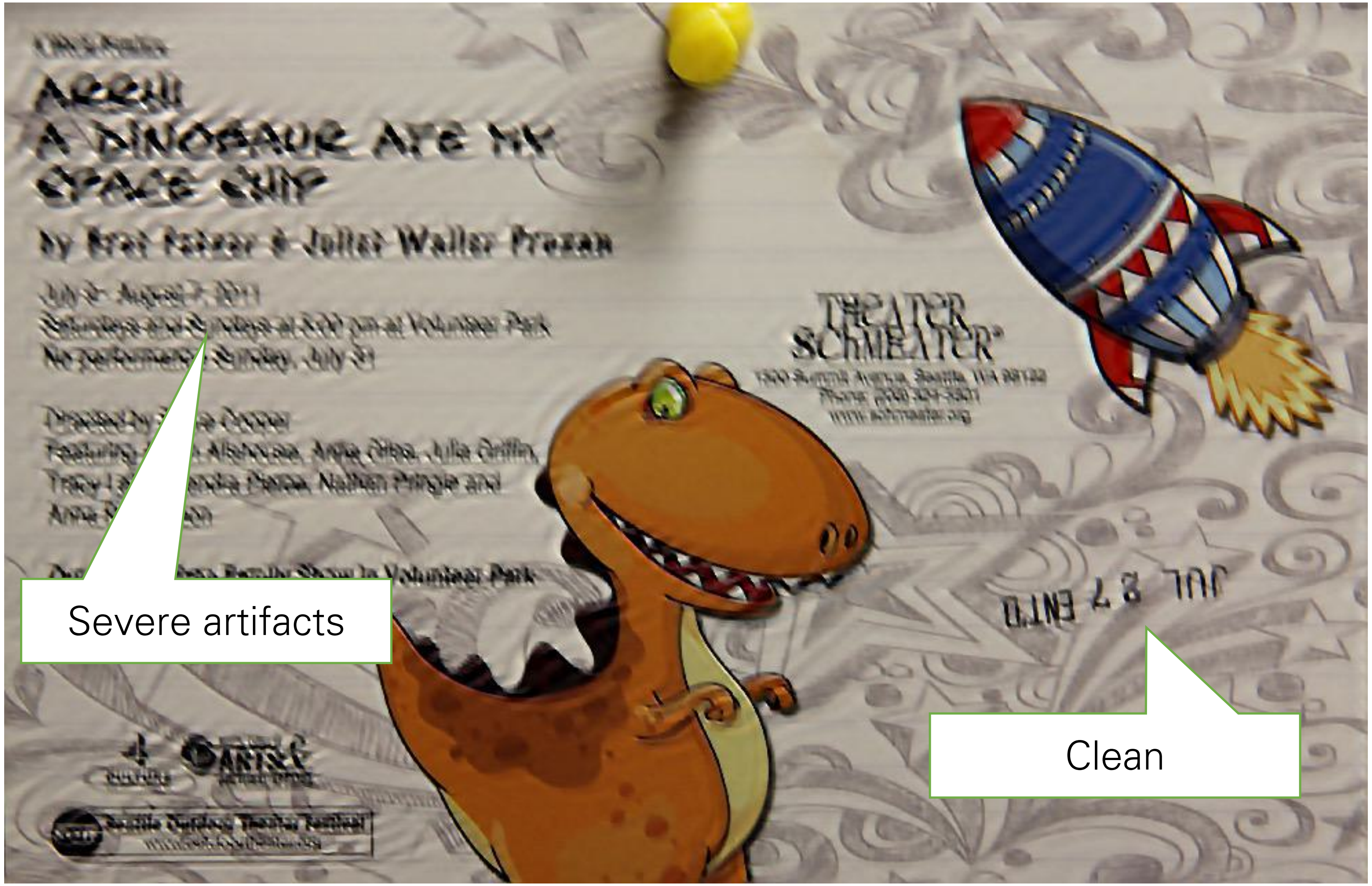
6D real camera motion

Different depths

Real Blurred Image



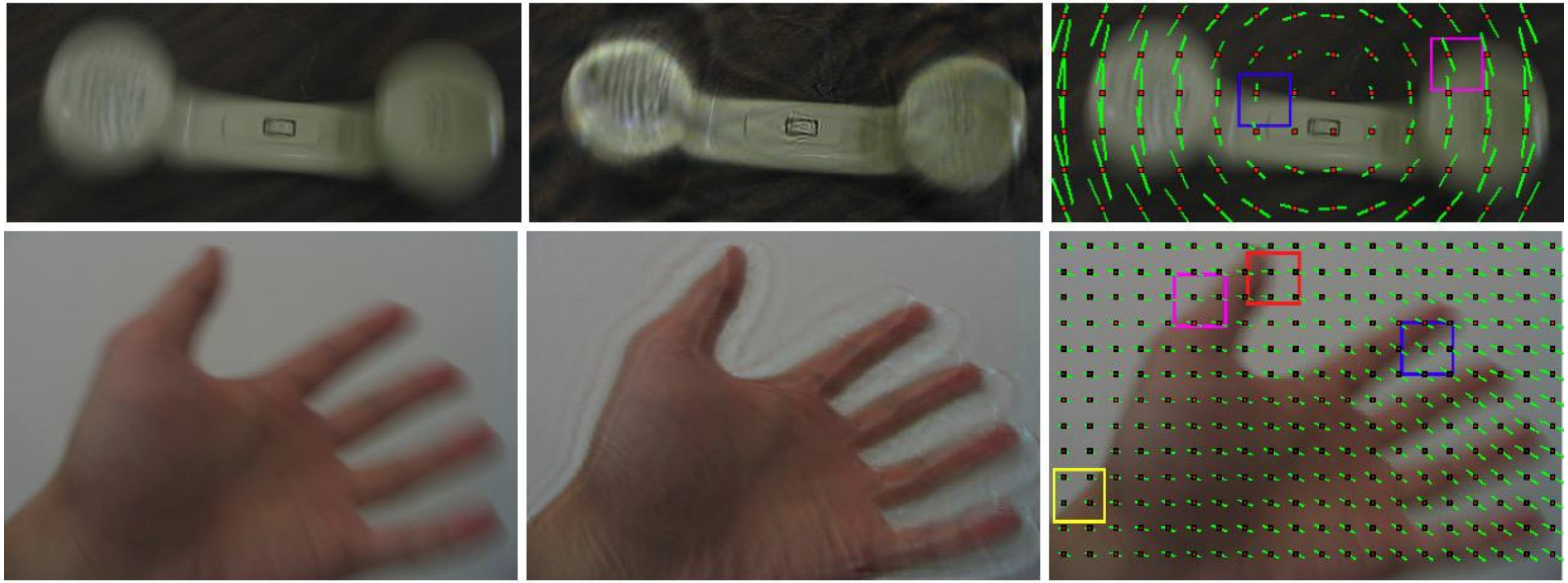
Non-uniformly blurred image



Uniform deblurring result

Pixel-wise Blur Model

- Dai and Wu, CVPR 2008
  - Estimate blur kernels for every pixel from a single image
  - Severely ill-posed
  - Parametric blur kernels

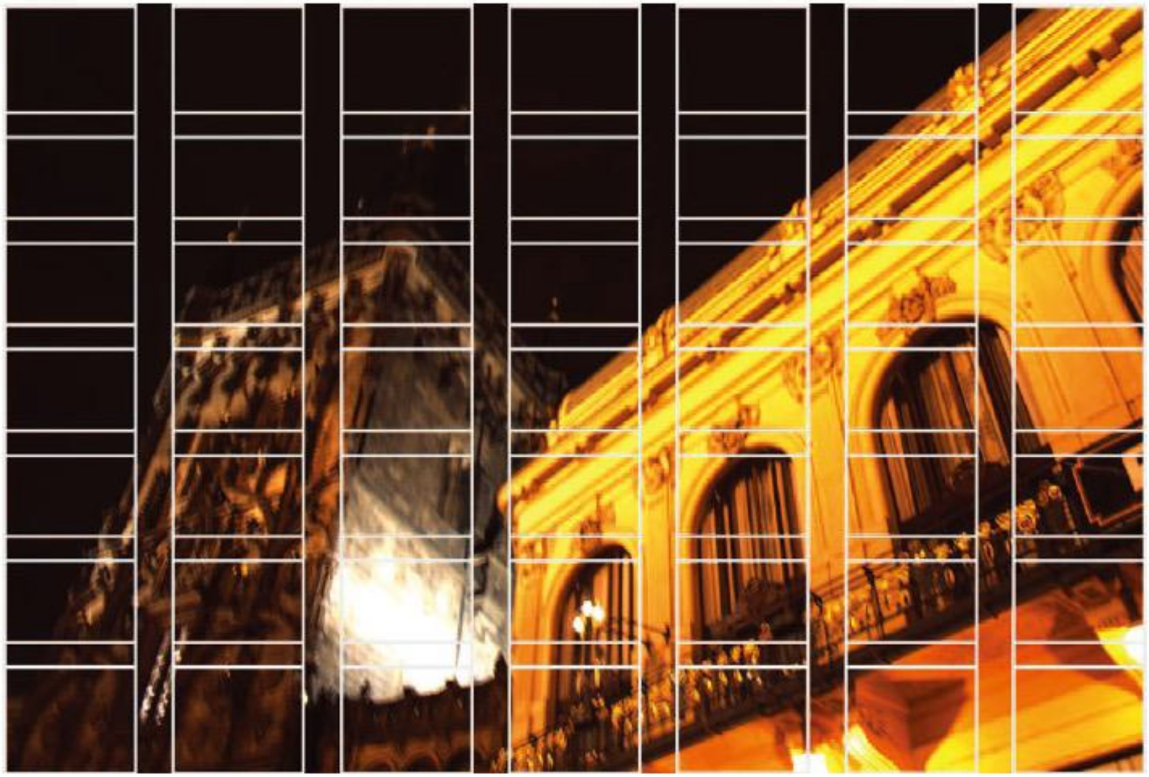


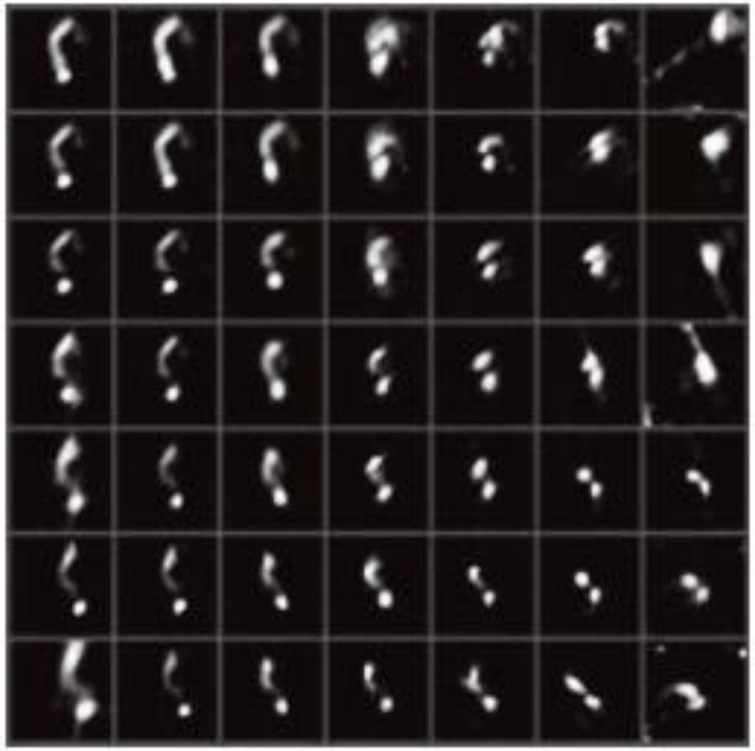
Pixel-wise Blur Model

- Tai et al. CVPR 2008
  - Hybrid camera to capture hi-res image & low-res video
  - Estimate per-pixel blur kernels using low-res video

Patch-wise Blur Model

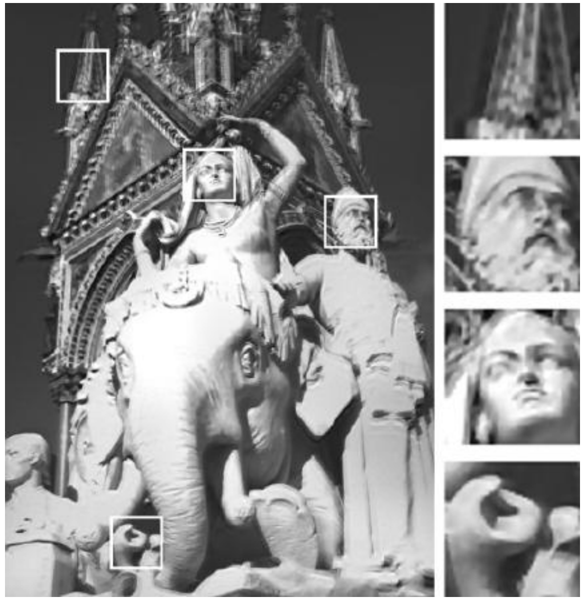
- Sorel and Sroubek, ICIP 2009
  - Estimate per-patch blur kernels from a blurred image and an underexposed noisy image





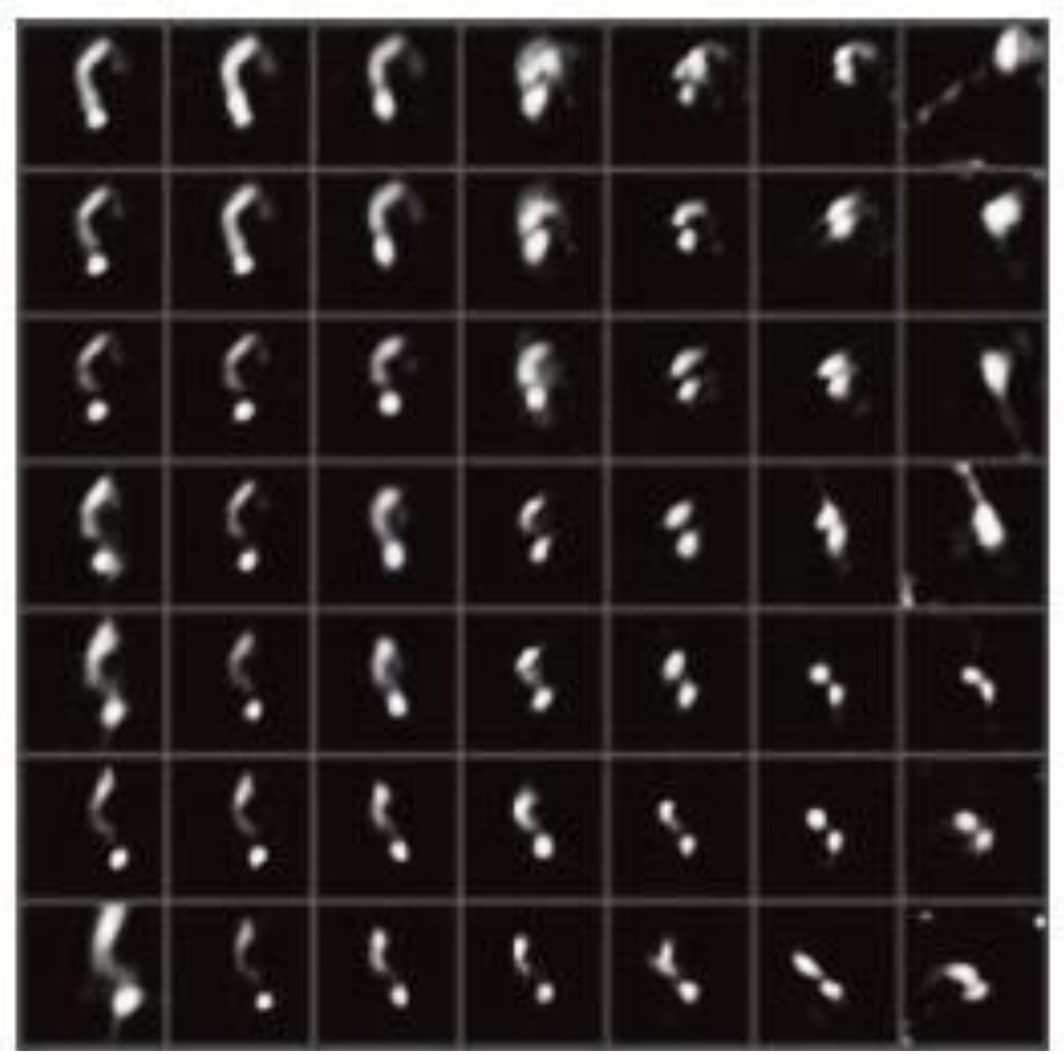
Patch-wise Blur Model

- Hirsch et al. CVPR 2010
  - Efficient filter flow (EFF) framework
  - More accurate approximation than the naïve patch-wise blur model
- Harmeling et al. NIPS 2010
  - Estimate per-patch blur kernels based on EFF from a single image



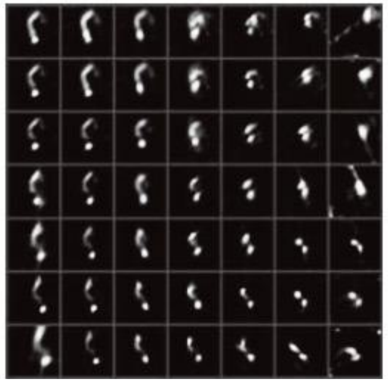
Patch-wise Blur Model

- Approximation
  - More patches → more accurate
- Computationally efficient
  - Patch-wise uniform blur
  - FFTs can be used
- Physically implausible blurs
  - Adjacent blur kernels cannot be very different from each other

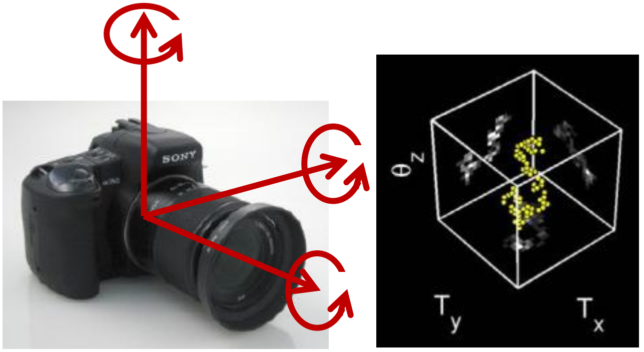


Summary

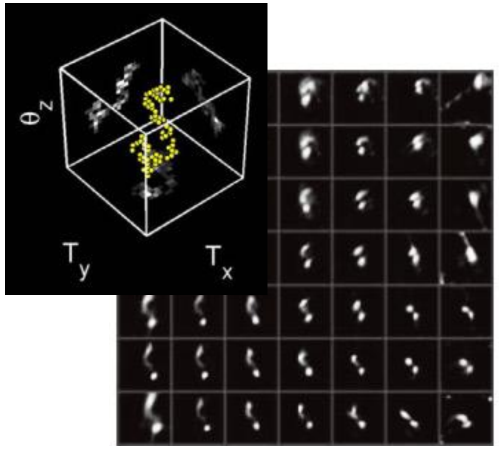
- Different blur models



Patch based
Efficient but no global constraint



Projective Motion Path
Globally consistent but inefficient



Hybrid
Efficient & globally consistent

- More realistic than uniform blur model
- Still approximations
  – Real camera motions: 6 DoF + more (zoom-in, depth, etc…)
- High dimensionality
  – Less stable & slower than uniform blur model

Remaining Challenges

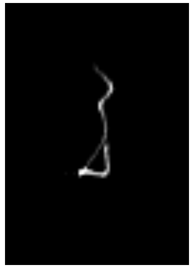
- All methods still fail quite often
- Noise
- Outliers
- Non-uniform blur
- Limited amount of edges
- Speed…
- Etc…

Non-blind Deconvolution

Non-blind Deconvolution (Uniform Blur)

Blurred image

=



*

Latent sharp image

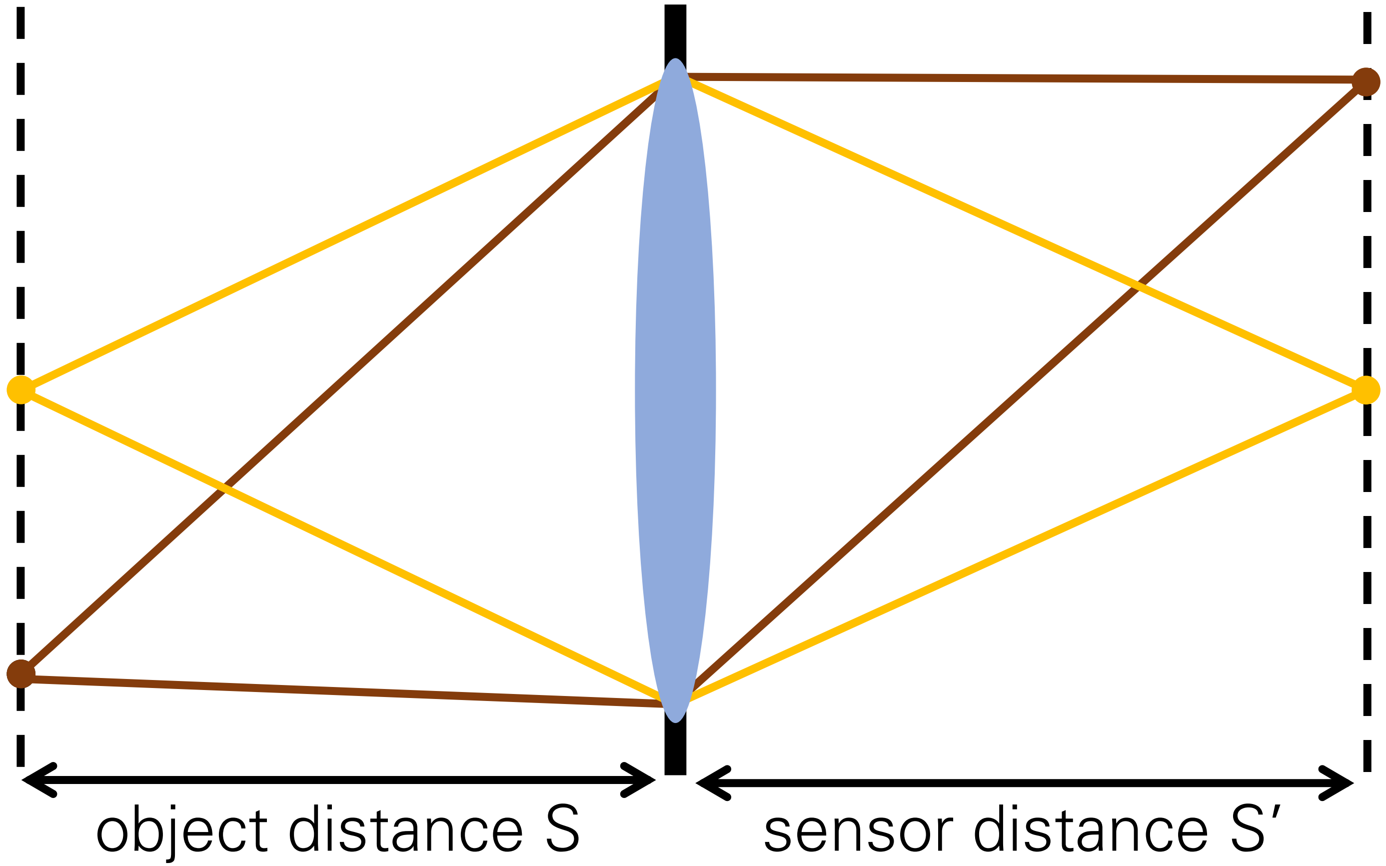
Blur kernel

Convolution operator

Lens imperfections

- Ideal lens: A point maps to a point at a certain plane.
- Real lens: A point maps to a circle that has non-zero minimum radius among all planes.

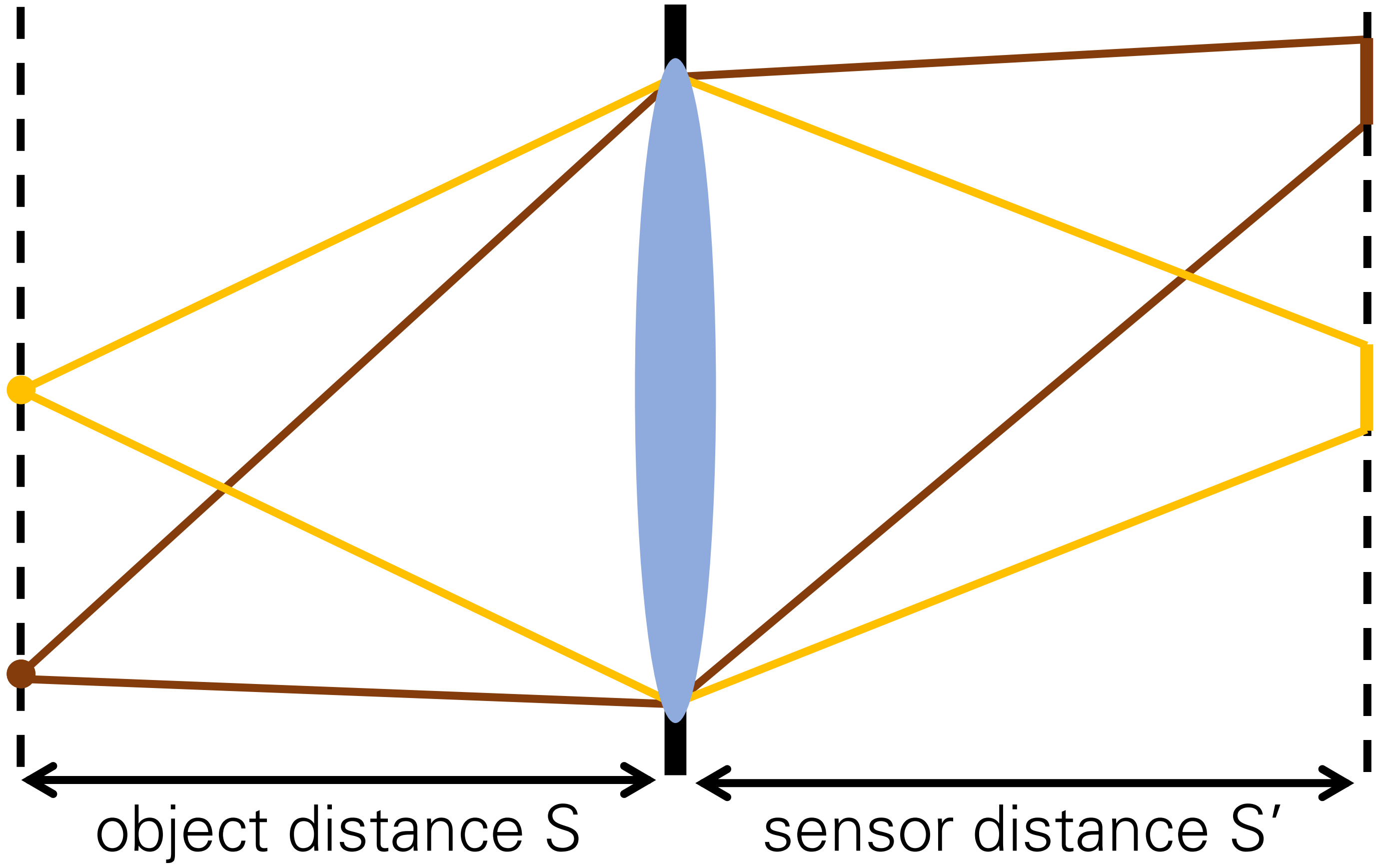
$$\boxed{\frac{1}{S'} + \frac{1}{S} = \frac{1}{f}}$$



Lens imperfections

- Ideal lens: A point maps to a point at a certain plane.
- Real lens: A point maps to a circle that has non-zero minimum radius among all planes.

$$\boxed{\frac{1}{S'} + \frac{1}{S} = \frac{1}{f}}$$

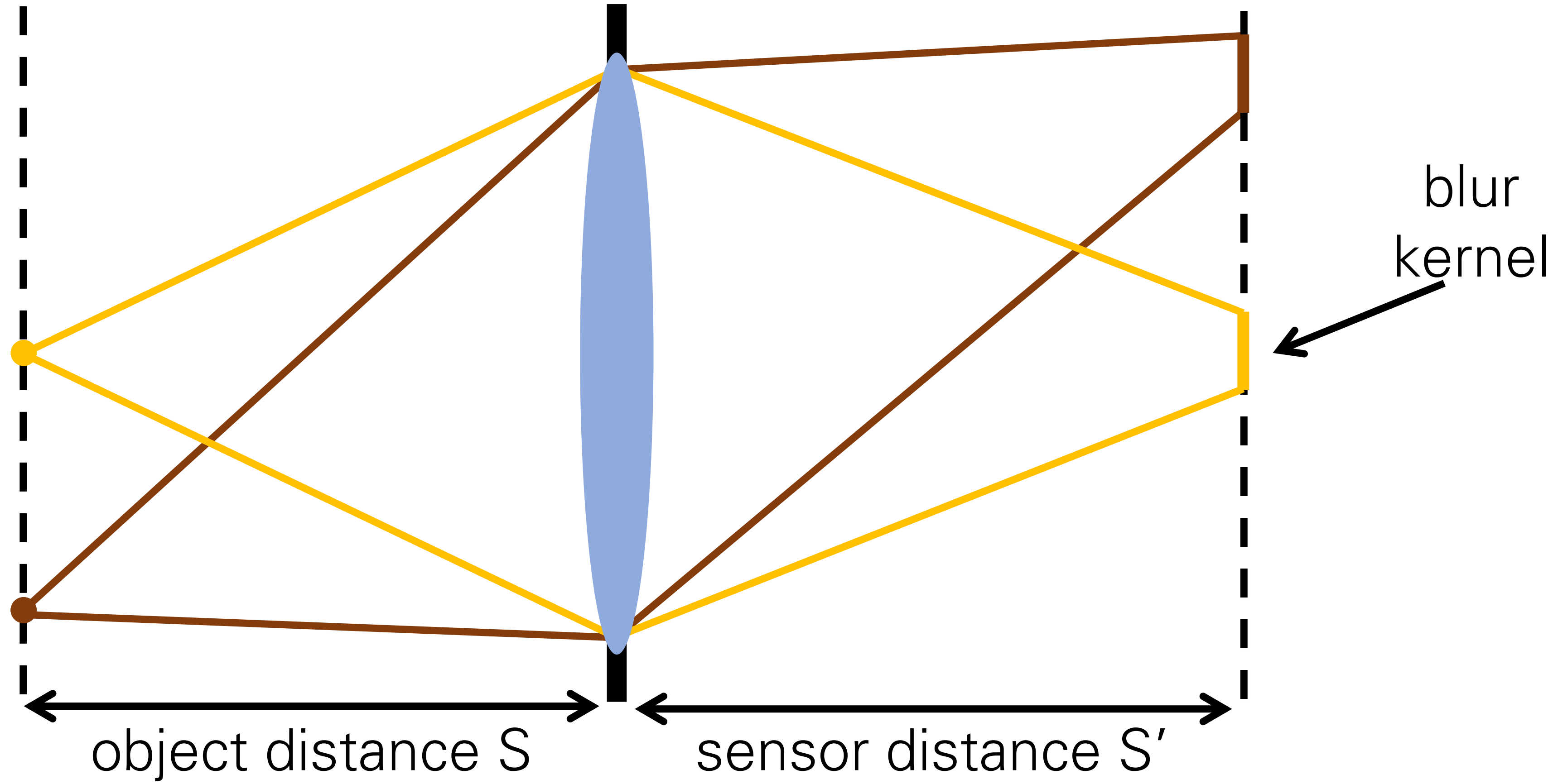


What is the effect of this on the images we capture?

Lens imperfections

- Ideal lens: A point maps to a point at a certain plane.
- Real lens: A point maps to a circle that has non-zero minimum radius among all planes.

$$\boxed{\frac{1}{S'} + \frac{1}{S} = \frac{1}{f}}$$



Shift-invariant blur.

Lens imperfections

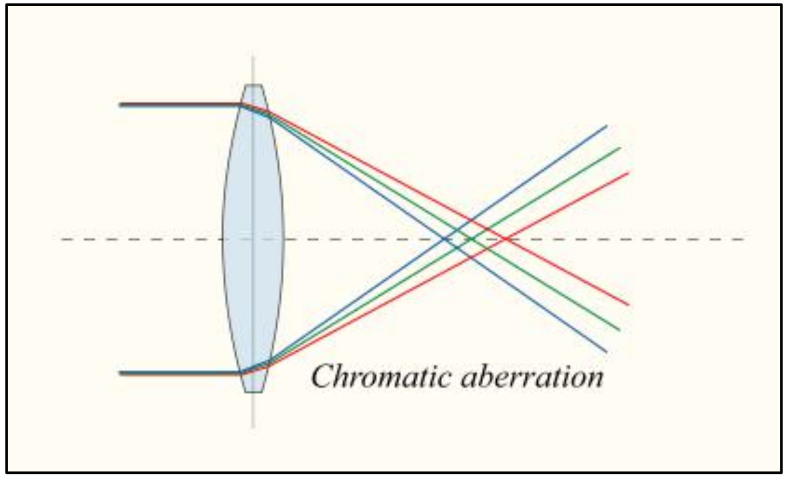
What causes lens imperfections?

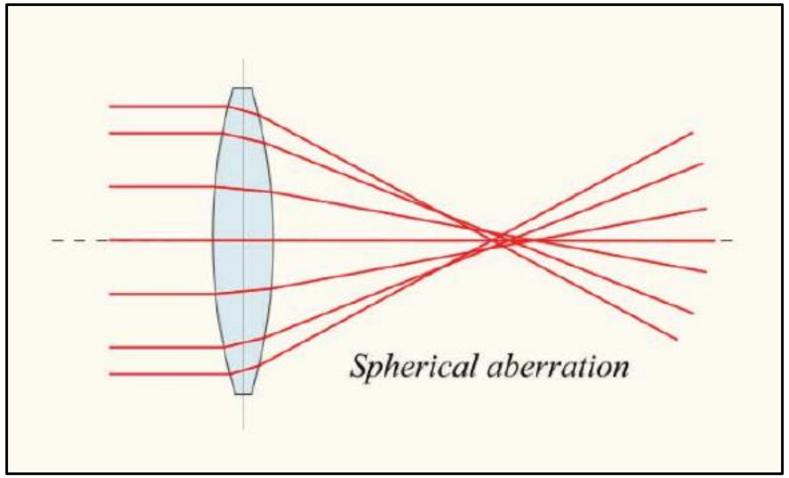
Lens imperfections

What causes lens imperfections?

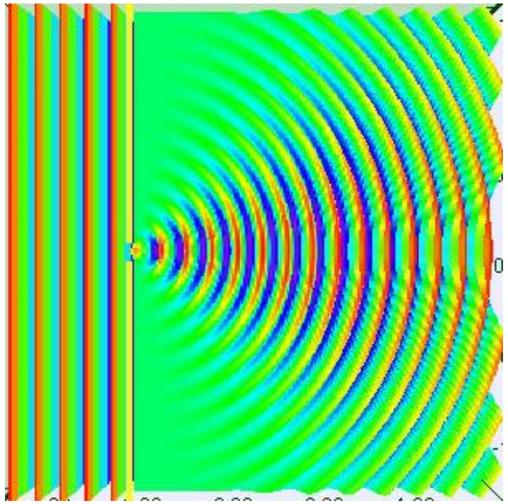
- Aberrations.

(Important note: Oblique aberrations like coma and distortion <u>are not shift-invariant</u> blur and we do not consider them here!)

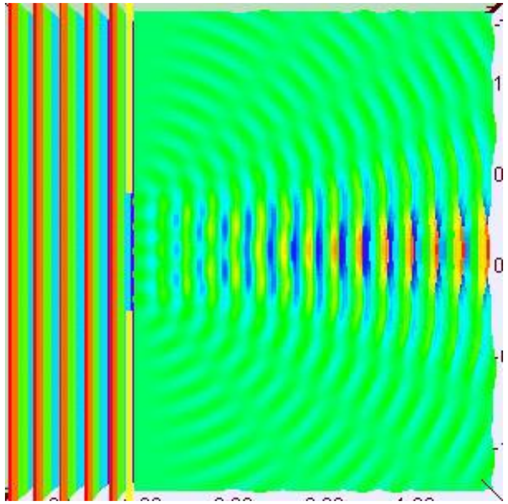




- Diffraction.



small aperture



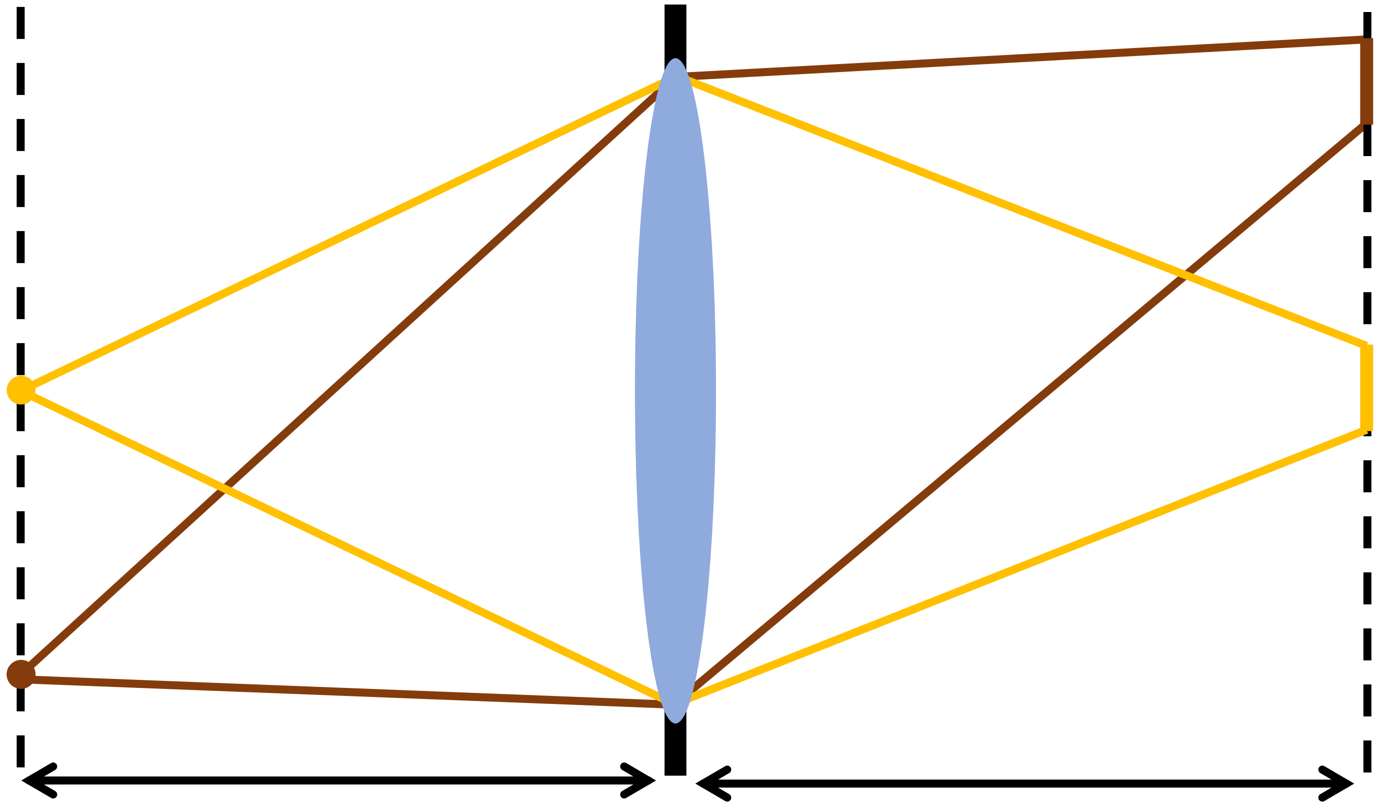
large aperture

Lens as an optical low-pass filter

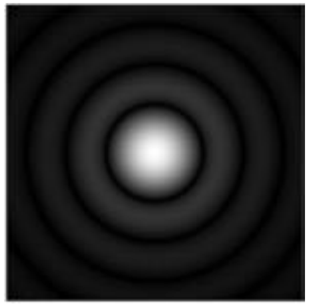
Point spread function (PSF): The blur kernel of a lens.

- "Diffraction-limited" PSF: No aberrations, only diffraction. Determined by aperture shape.

$$\boxed{\frac{1}{S'} + \frac{1}{S} = \frac{1}{f}}$$



blur kernel



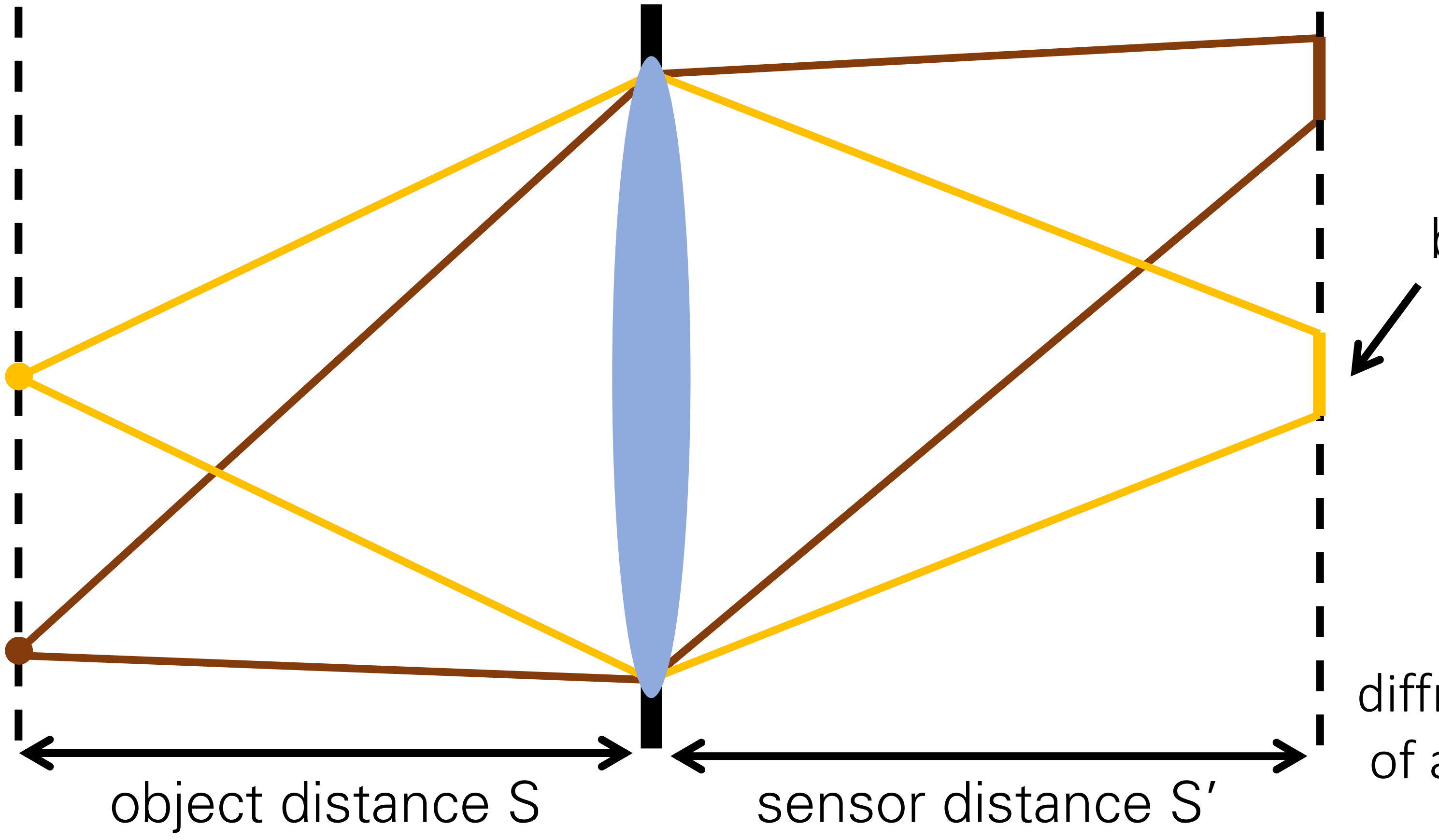
diffraction-limited PSF of a circular aperture (Airy pattern)

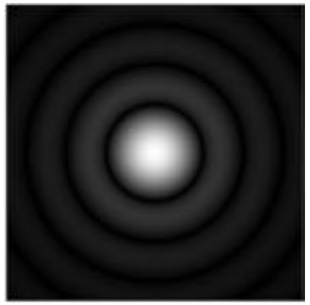
Lens as an optical low-pass filter

Point spread function (PSF): The blur kernel of a lens.

- "Diffraction-limited" PSF: No aberrations, only diffraction. $\boxed{\text{Determined by aperture shape.}}$

$$\boxed{\frac{1}{S'} + \frac{1}{S} = \frac{1}{f}}$$





diffraction-limited PSF of a circular aperture (*Airy pattern)*

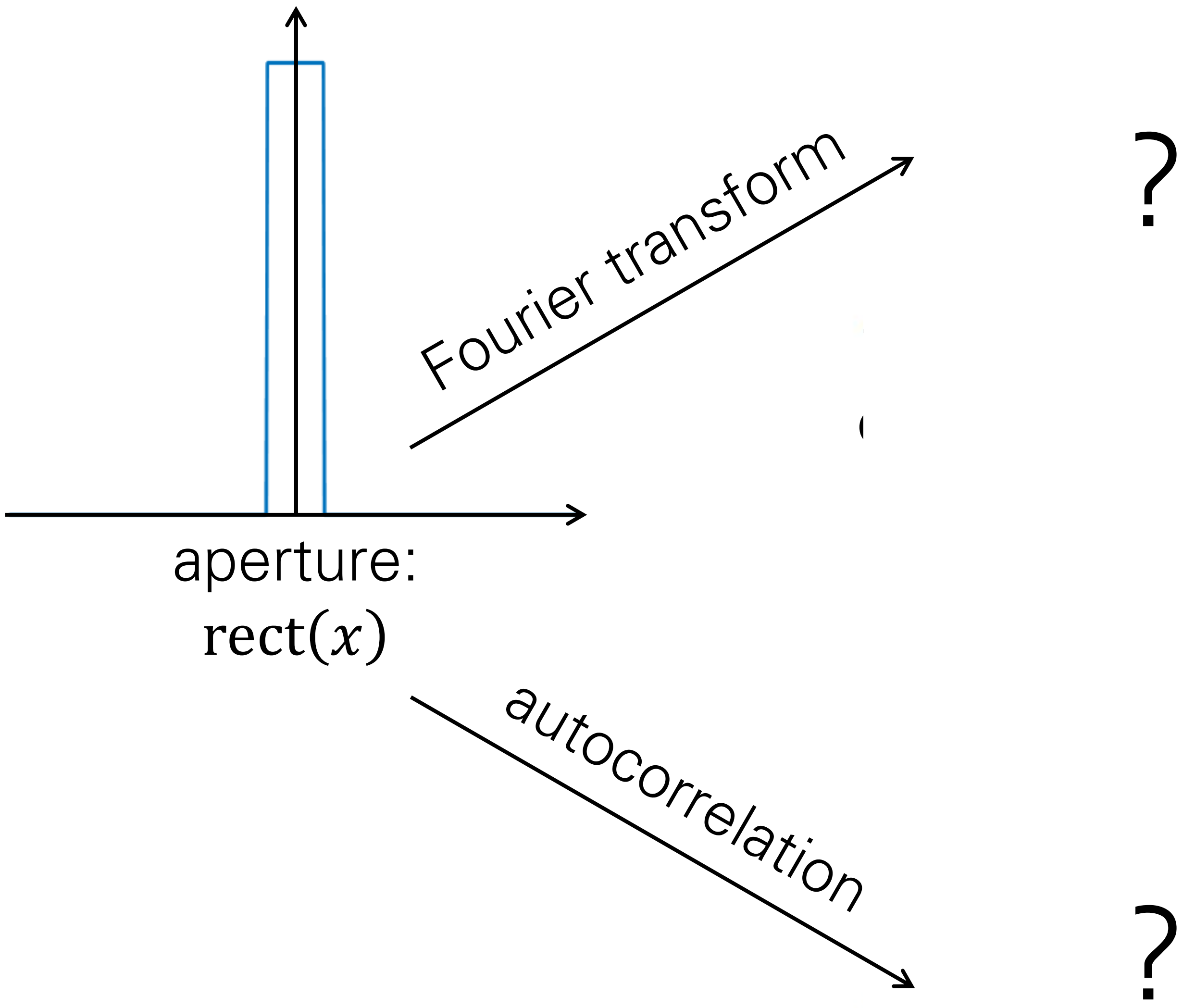
Some basics of diffraction theory

We will assume that we can use:

- Fraunhofer diffraction (i.e., distance of sensor and aperture is large relative to wavelength).
- incoherent illumination (i.e., the light we are measuring is not laser light).

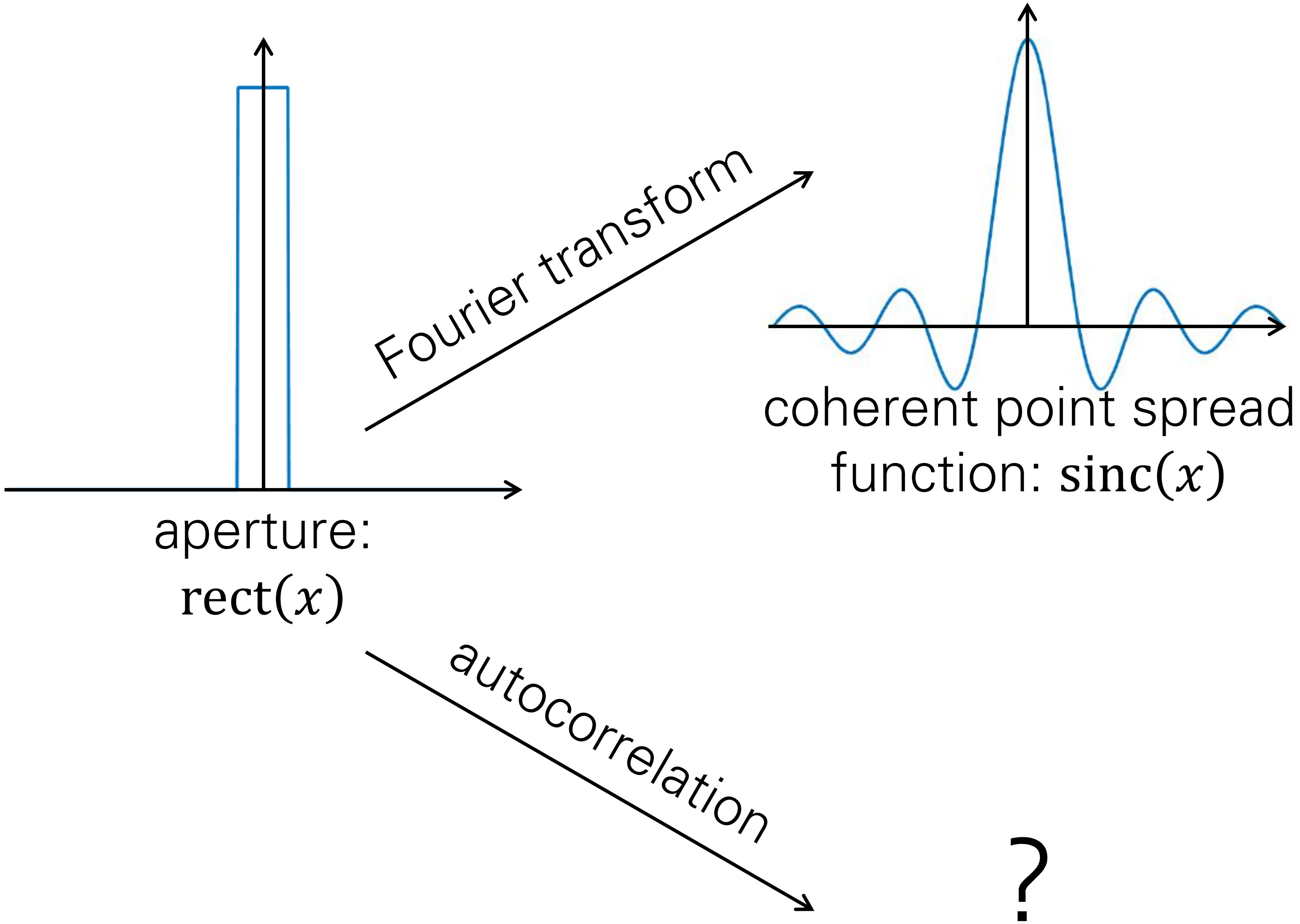
We will also be ignoring various scale factors. Different functions are <u>not</u> drawn to scale.

Some basics of diffraction theory



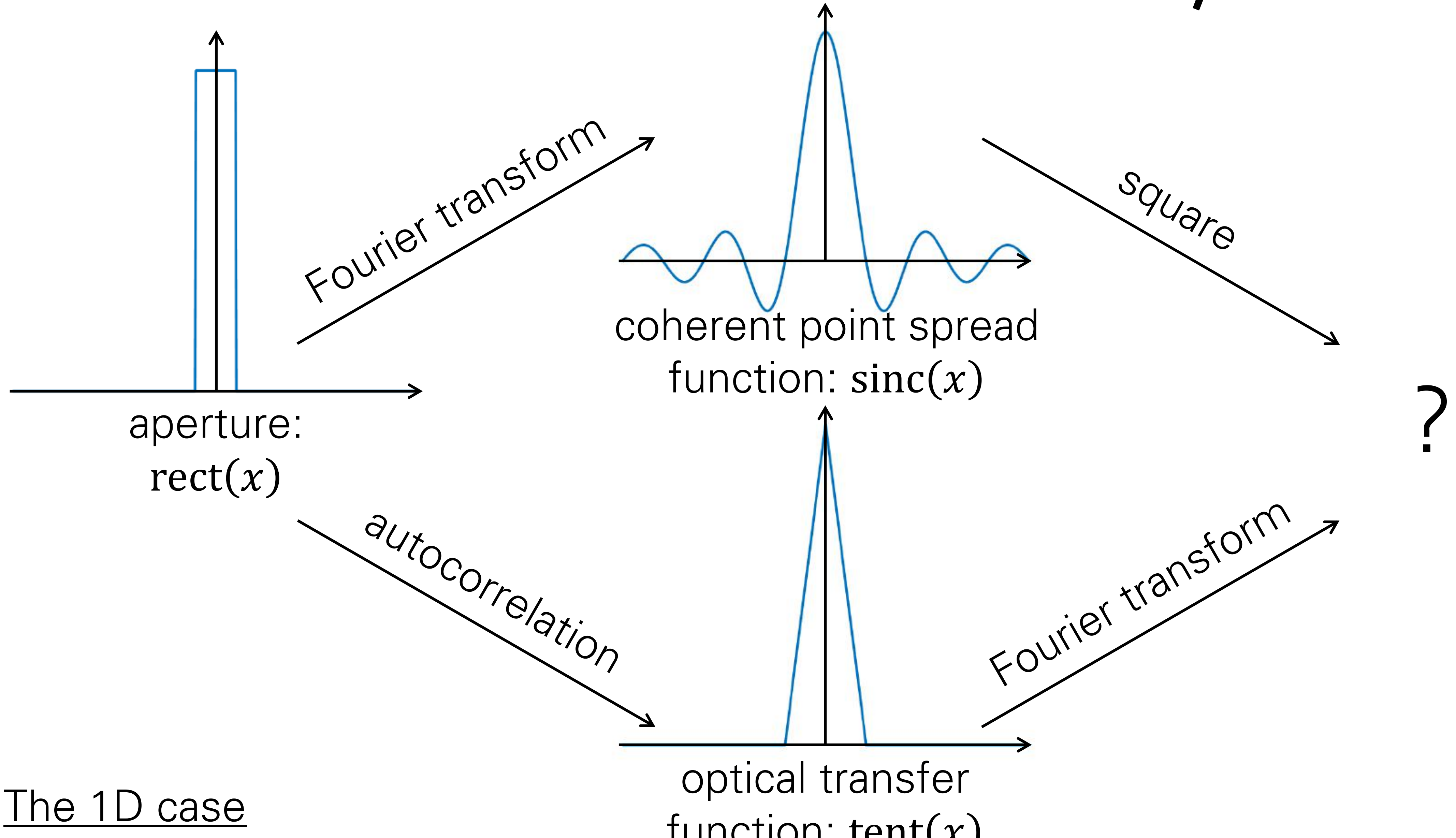
The 1D case

Some basics of diffraction theory



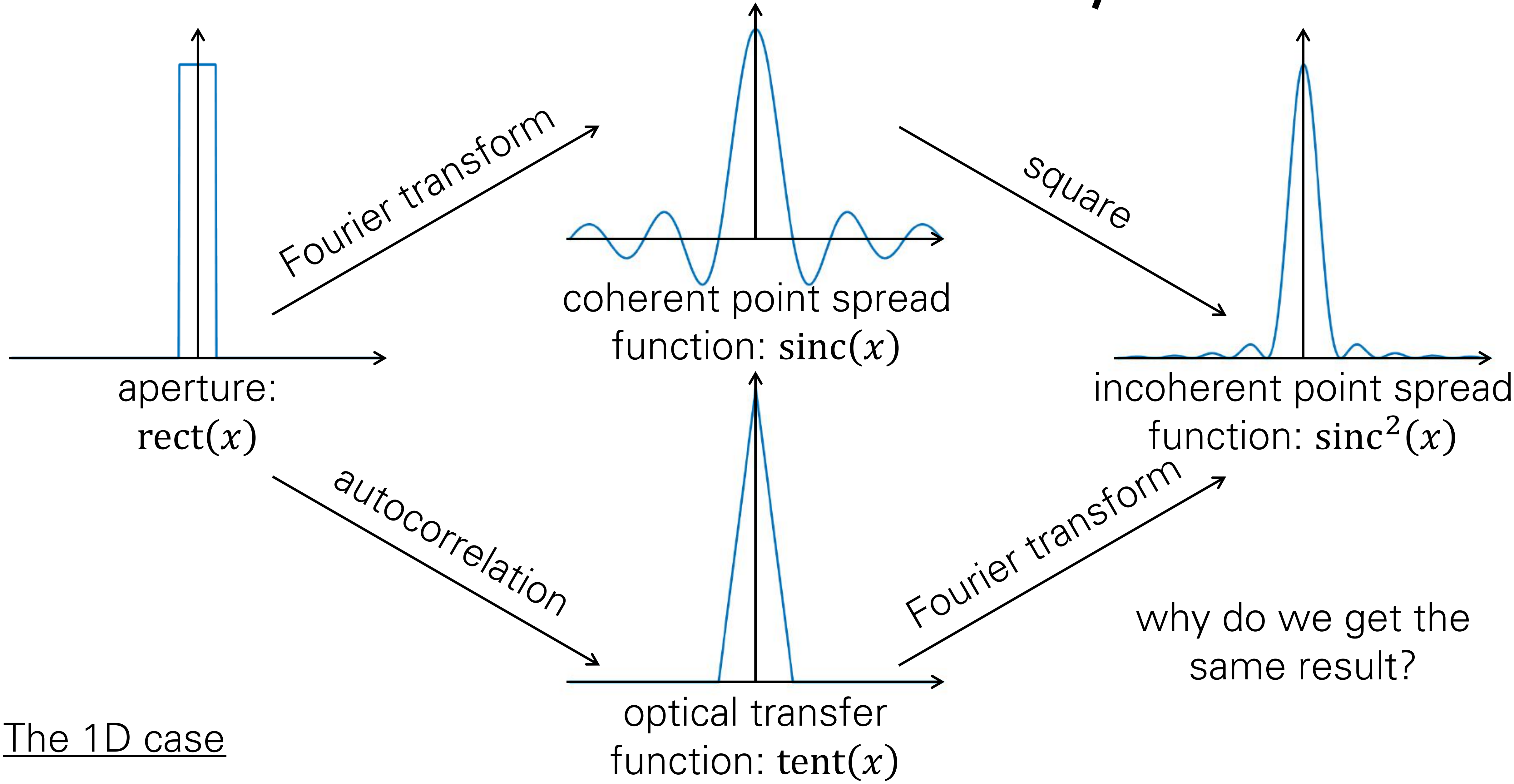
The 1D case

Some basics of diffraction theory

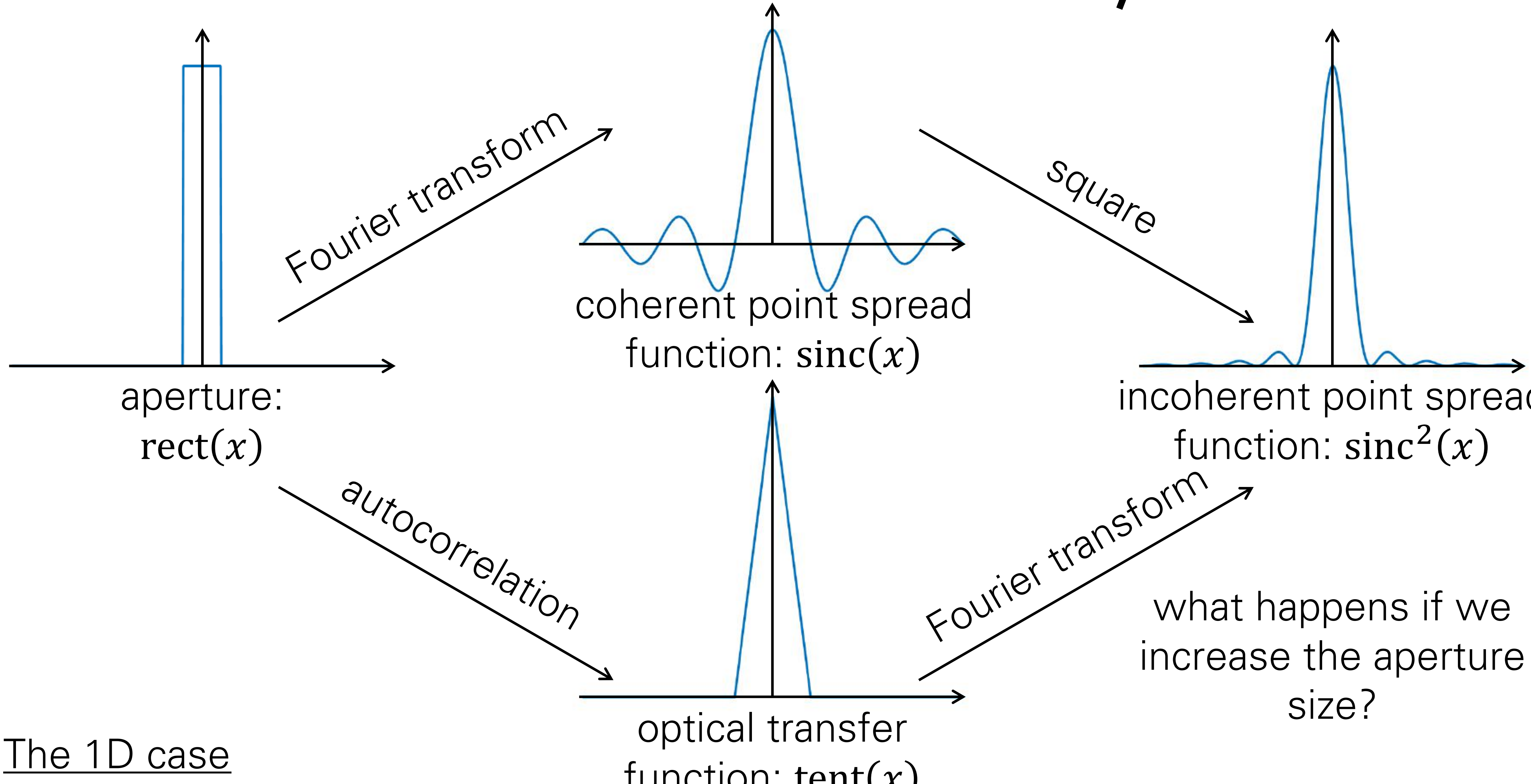


The 1D case

Some basics of diffraction theory



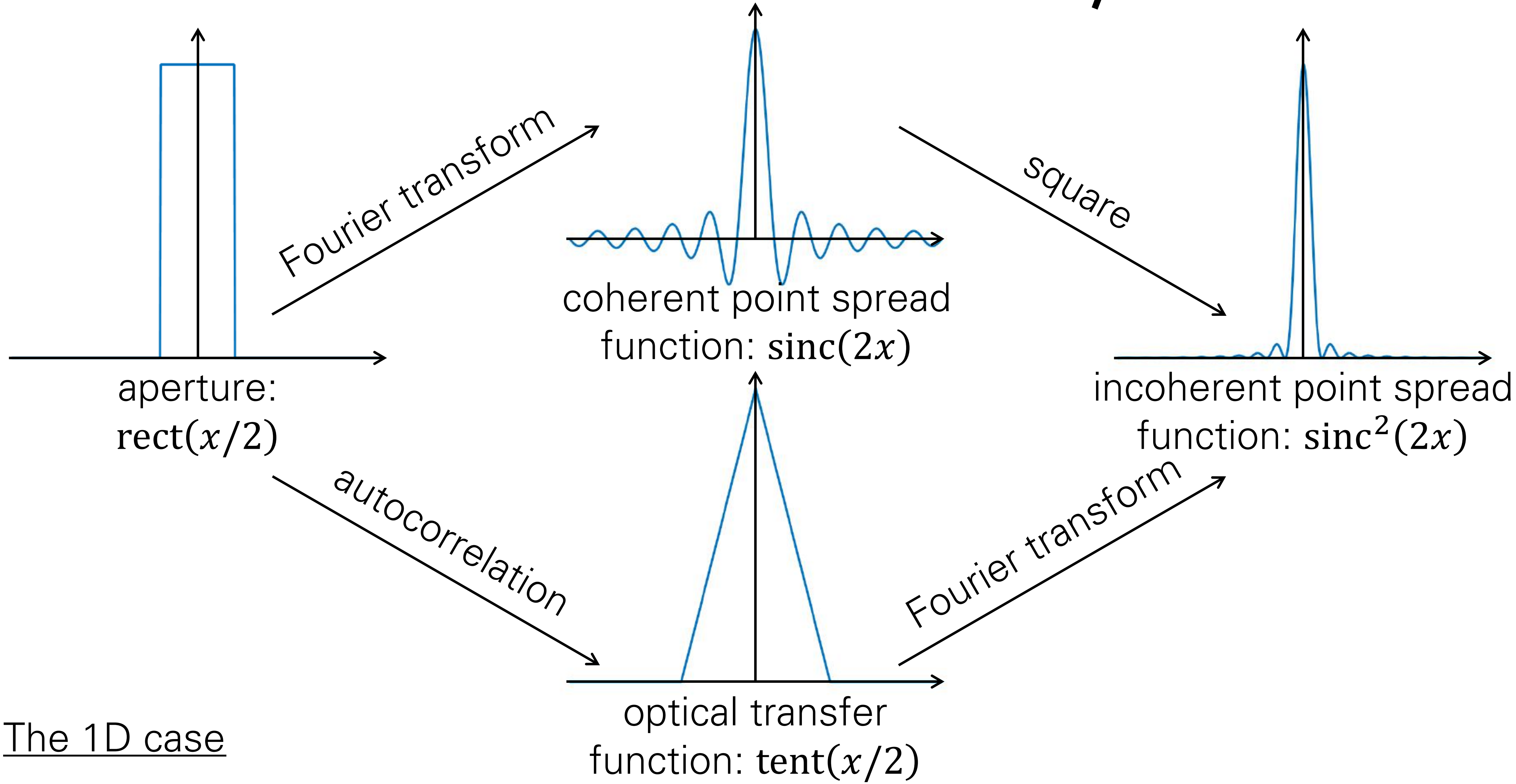
Some basics of diffraction theory



what happens if we increase the aperture size?

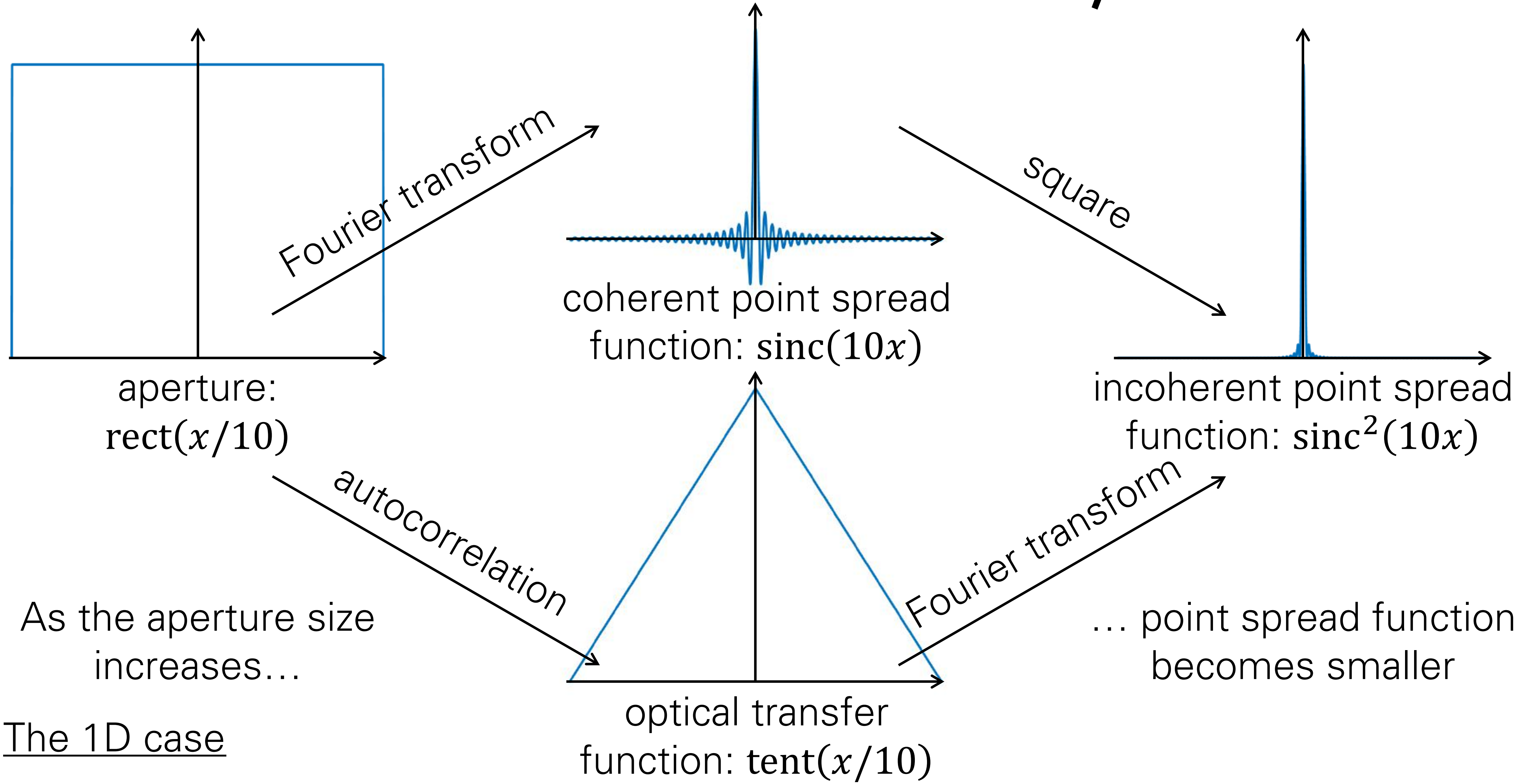
The 1D case

Some basics of diffraction theory



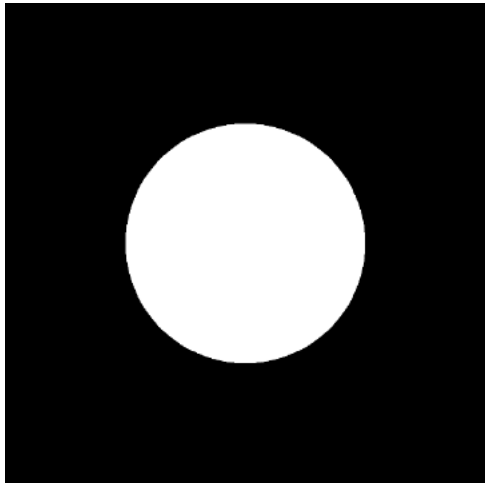
<u>The 1D case</u>

Some basics of diffraction theory



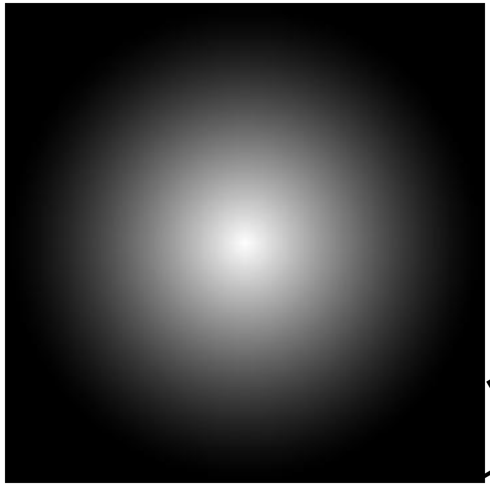
<u>The 1D case</u>

Some basics of diffraction theory



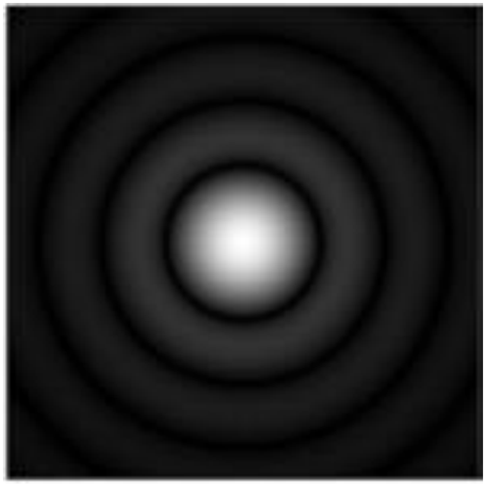
aperture

autocorrelation



optical transfer function

Fourier transform



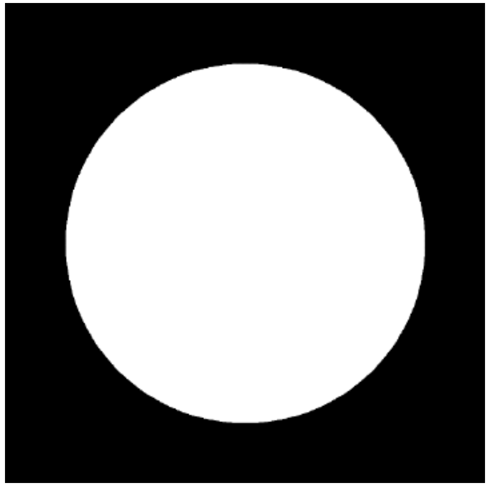
incoherent point spread function

As the aperture size increases…

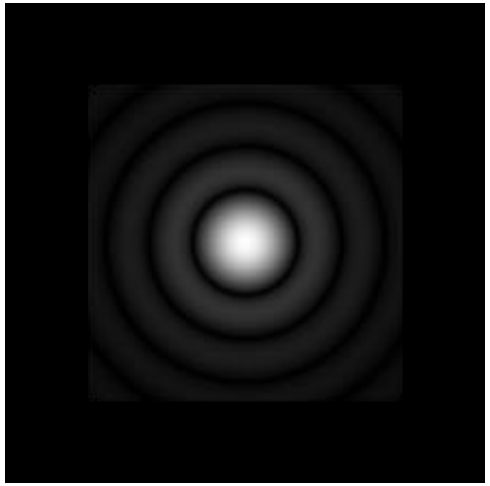
… point spread function becomes smaller

The 2D case

Some basics of diffraction theory

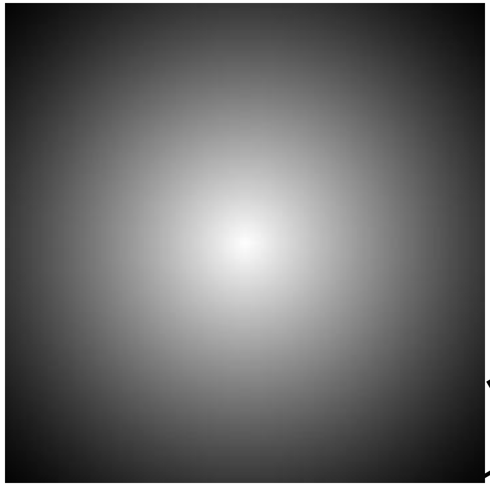


aperture



incoherent point spread function

autocorrelation



optical transfer function

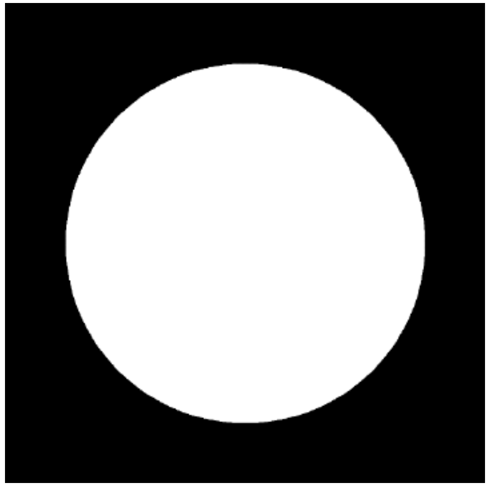
Fourier transform

As the aperture size increases…

… point spread function becomes smaller

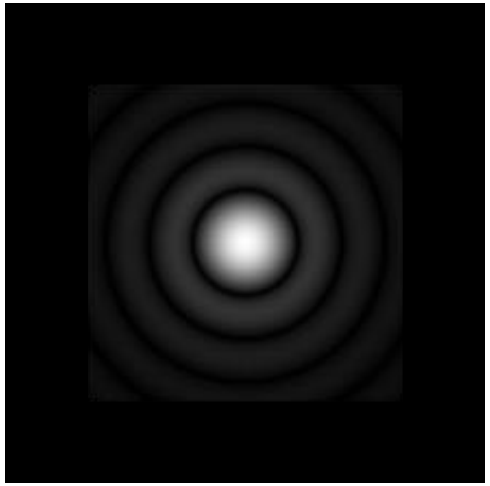
The 2D case

Some basics of diffraction theory



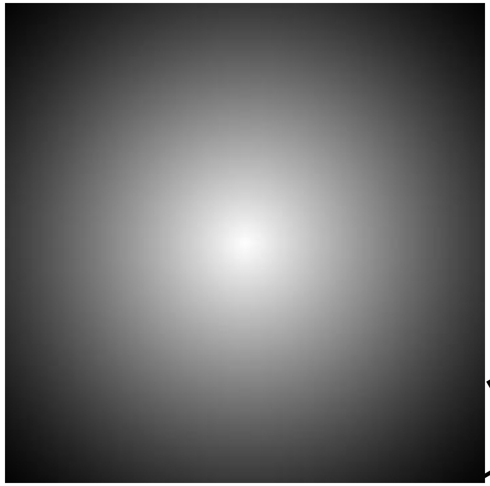
aperture

Why do we prefer circular apertures?



incoherent point spread function

autocorrelation



optical transfer function

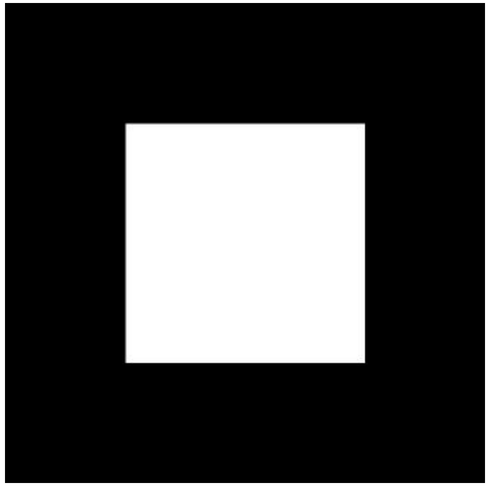
Fourier transform

As the aperture size increases…

… point spread function becomes smaller

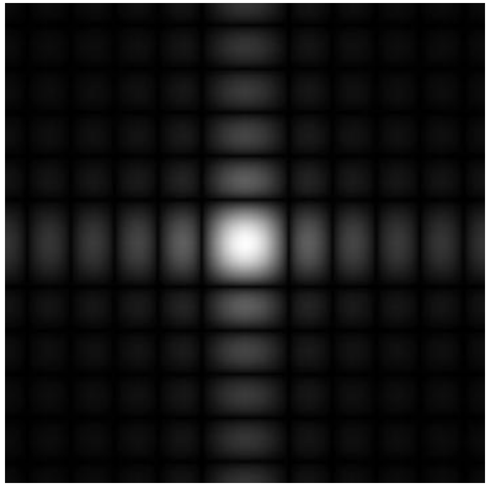
The 2D case

Some basics of diffraction theory



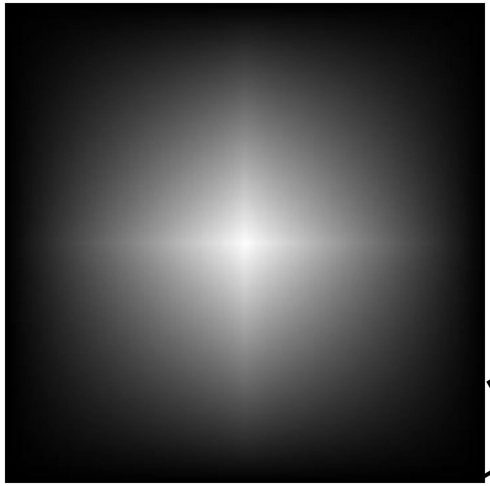
aperture

Other shapes produce very anisotropic blur.



incoherent point spread function

autocorrelation



optical transfer function

Fourier transform

As the aperture size increases…

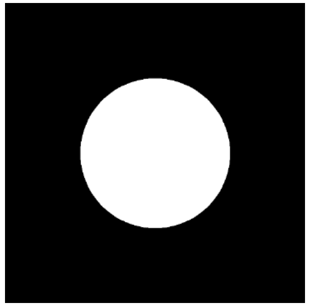
… point spread function becomes smaller

The 2D case

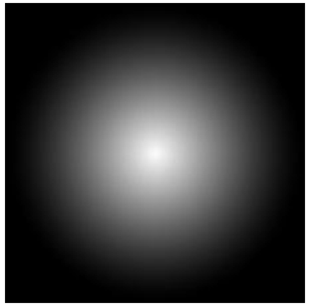
Lens as an optical low-pass filter

Point spread function (PSF): The blur kernel of a lens.

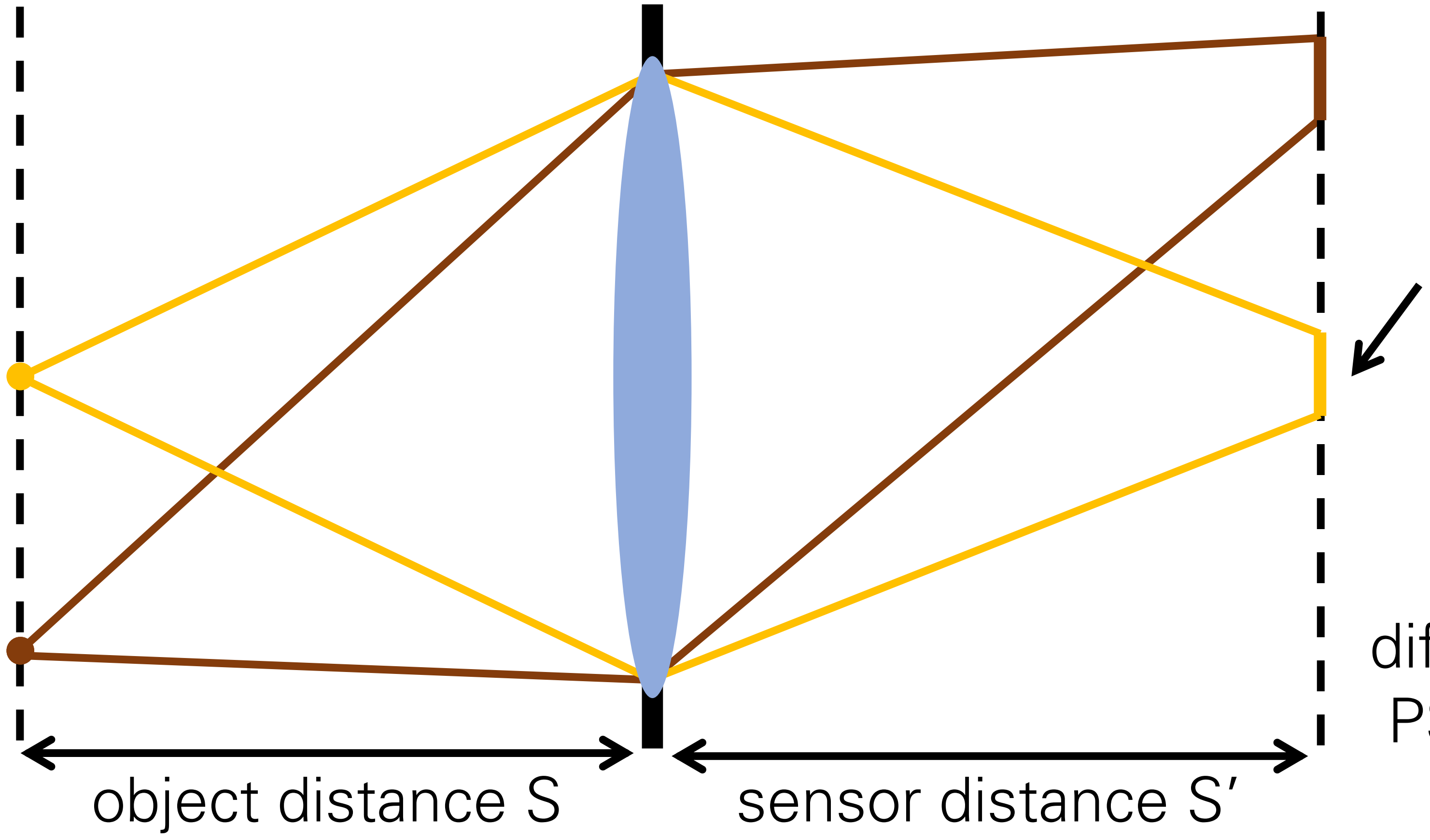
- "Diffraction-limited" PSF: No aberrations, only diffraction. Determined by aperture shape.

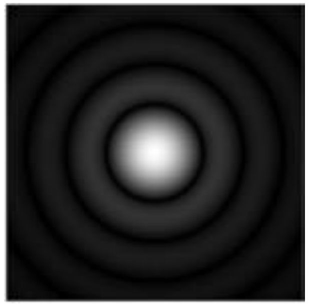


aperture



optical transfer function (OTF)

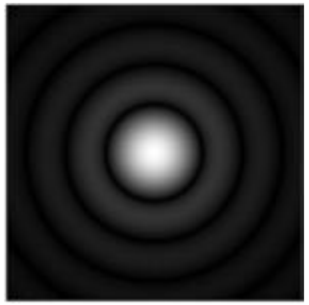




diffraction-limited PSF of a circular aperture (Airy pattern)

Lens as an optical low-pass filter

$*$



$=$

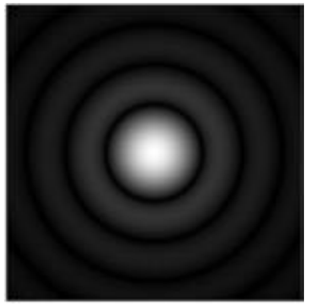
image from a perfect lens | imperfect lens PSF | image from imperfect lens

$$i \quad * \quad k \quad = \quad b$$

Lens as an optical low-pass filter

If we know b and k, can we recover i?

$*$



$=$

image from a perfect lens

imperfect lens PSF

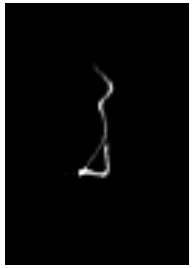
image from imperfect lens

$$i \quad * \quad k \quad = \quad b$$

Non-blind Deconvolution (Uniform Blur)

Blurred image

=



*

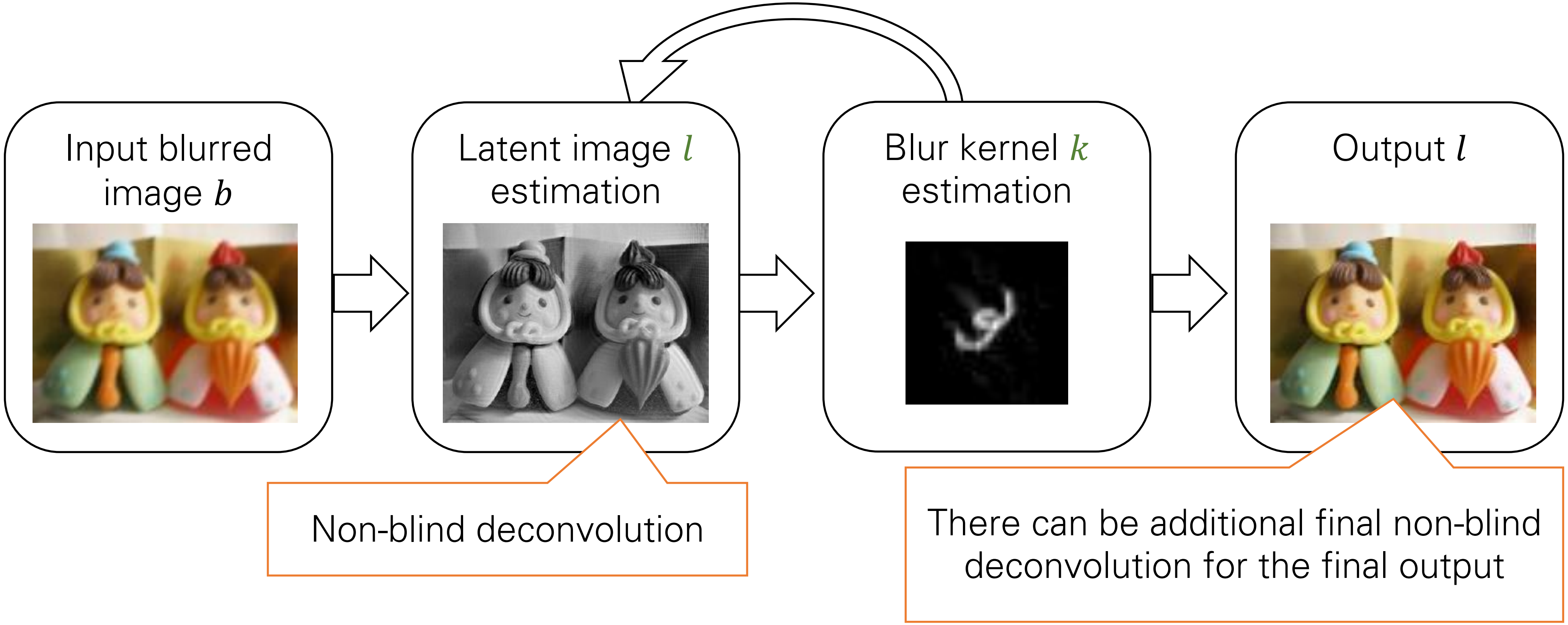
Latent sharp image

Blur kernel

Convolution operator

Non-blind Deconvolution

- Key component in many deblurring systems
  - For example, in MAP based blind deconvolution:



Non-blind Deconvolution

- Wiener filter
- Richardson-Lucy deconvolution
- Rudin et al. Physica 1992
- Bar et al. IJCV 2006
- Levin et al. SIGGRAPH 2007
- Shan et al. SIGGRAPH 2008
- Yuan et al. SIGGRAPH 2008
- Harmeling et al. ICIP 2010
- etc…

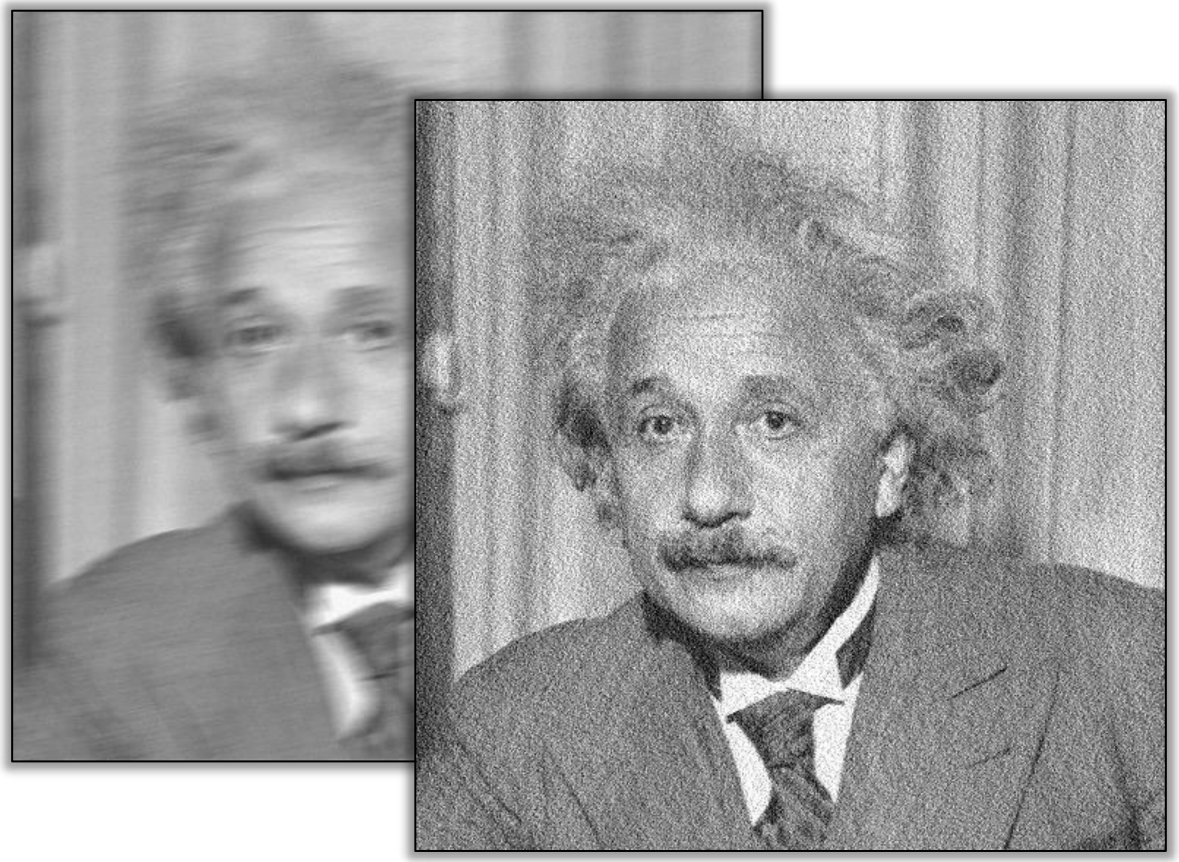
Ill-Posed Problem

- Even if we know the true blur kernel, we cannot restore the latent image perfectly, because:

- Loss of high-freq info & noise ≈ denoising & super-resolution

Ill-Posed Problem

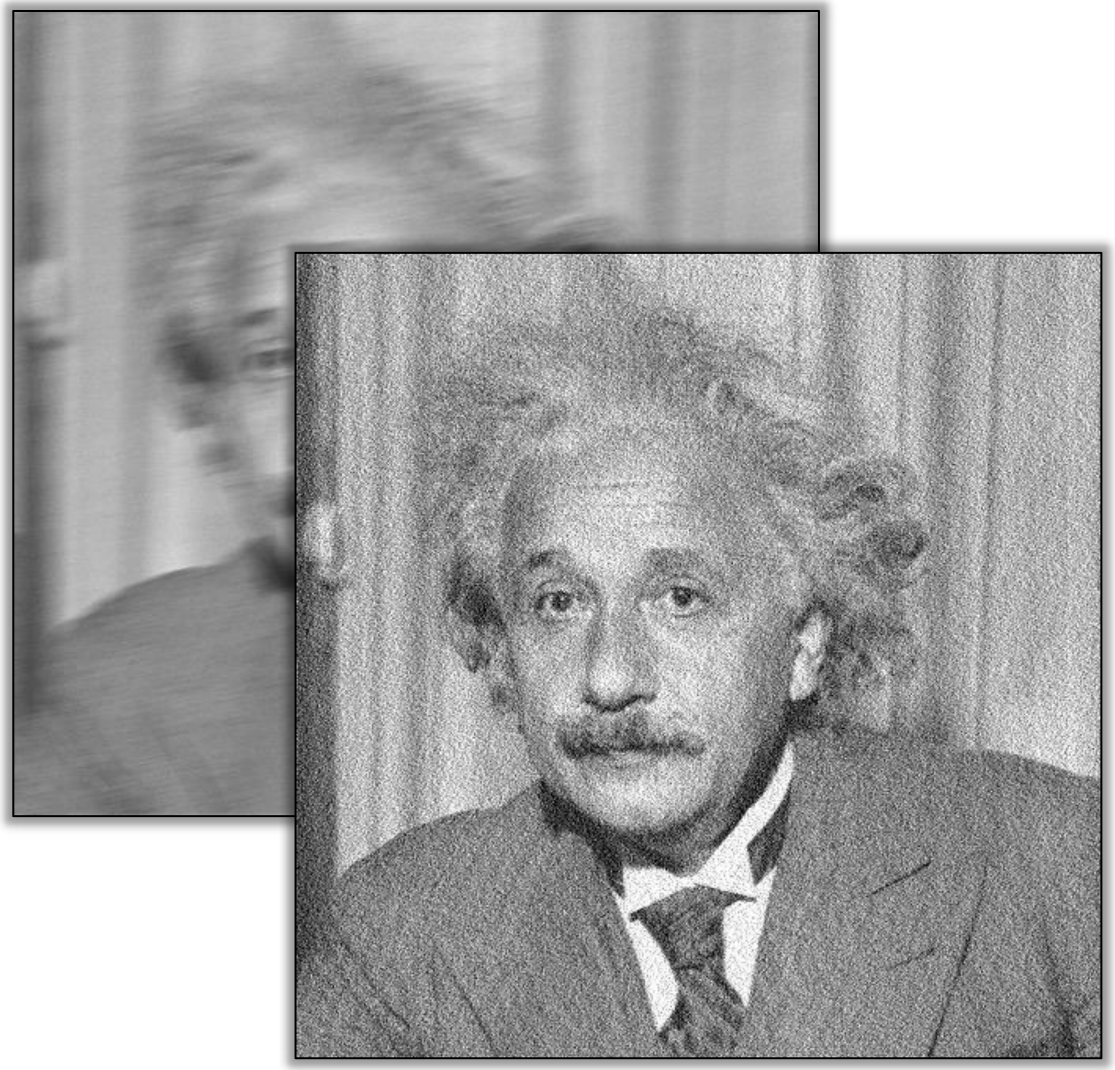
- Deconvolution amplifies noise as well as sharpens edges



- Ringing artifacts
  - Inaccurate blur kernels, outliers cause ringing artifacts

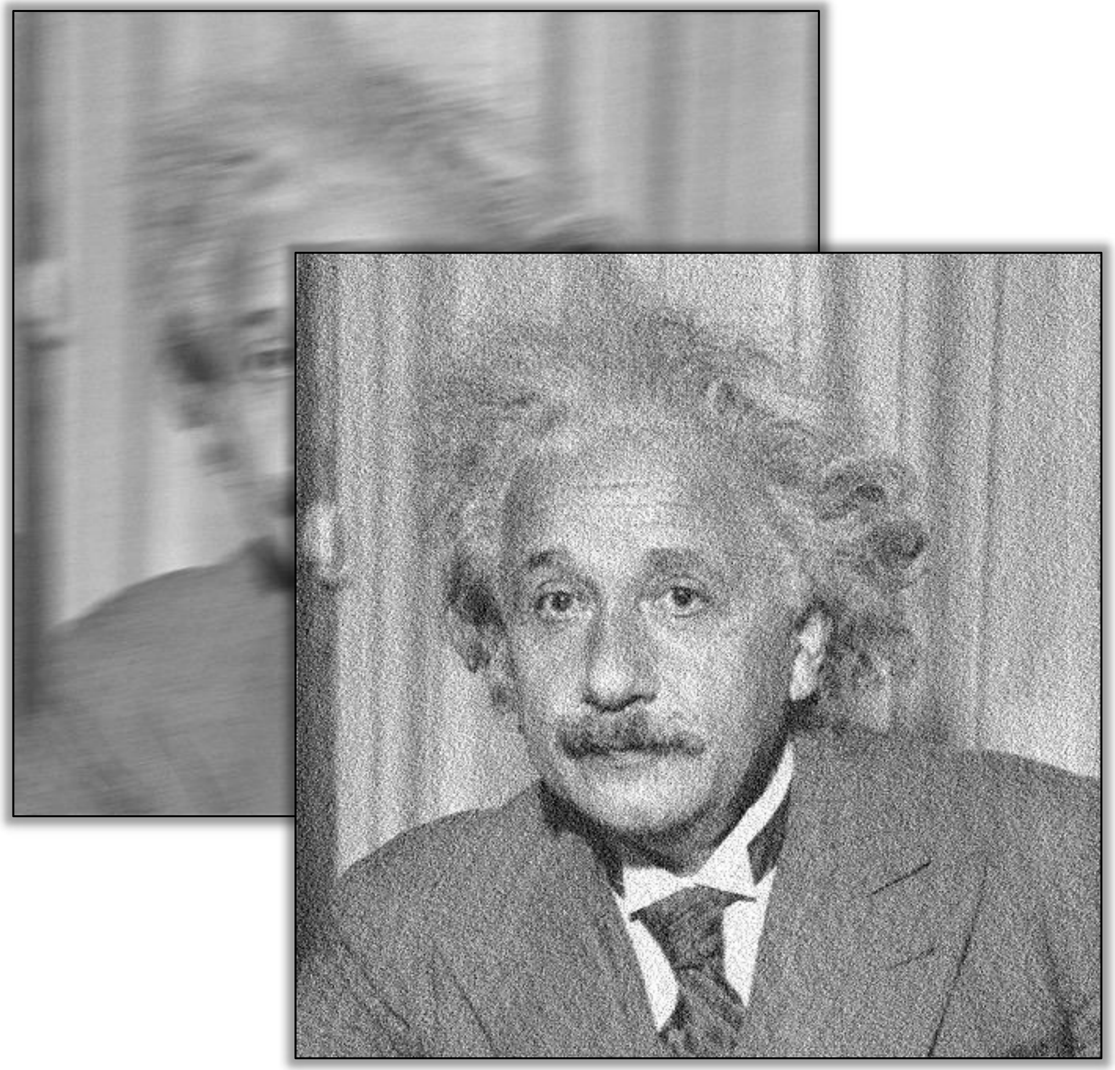
Classical Methods

- Popular methods
  - Wiener filtering
  - Richardson-Lucy deconvolution
  - Constrained least squares
- Matlab Image Processing Toolbox
  - deconvwnr, deconvlucy, deconvreg
- Simple assumption on noise and latent images
  - Simple & fast
  - Prone to noise & artifacts



Classical Methods

- Popular methods
  - **Wiener filtering**
  - Richardson-Lucy deconvolution
  - Constrained least squares
- Matlab Image Processing Toolbox
  - deconvwnr, deconvlucy, deconvreg
- Simple assumption on noise and latent images
  - Simple & fast
  - Prone to noise & artifacts



Deconvolution

$$i \ast k = b$$

If we know k and b, can we recover i?

Deconvolution

$$i \ast k = b$$

Reminder: convolution is multiplication in Fourier domain:

$$F(i) \cdot F(k) = F(b)$$

If we know k and b, can we recover i?

Deconvolution

$$i \ast k = b$$

Reminder: convolution is multiplication in Fourier domain:

$$F(i) \cdot F(k) = F(b)$$

Deconvolution is division in Fourier domain:

$$F(i_{est}) = F(b) \setminus F(k)$$

After division, just do inverse Fourier transform:

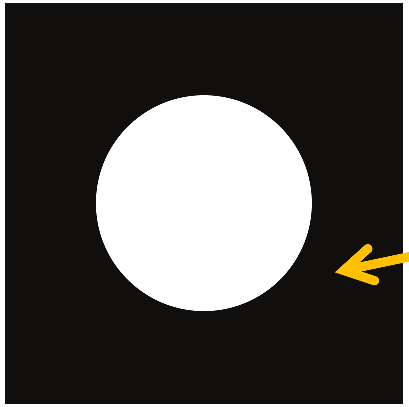
$$i_{est} = F^{-1} (F(b) \setminus F(k))$$

Deconvolution

Any problems with this approach?

Deconvolution

- The OTF (Fourier of PSF) is a low-pass filter



zeros at high frequencies

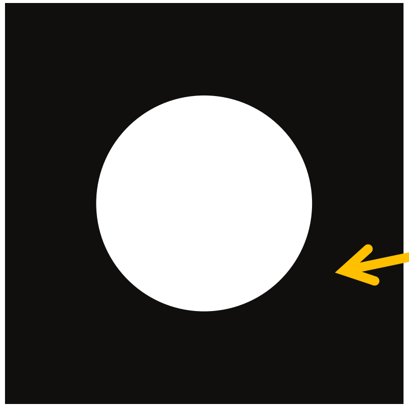
- The measured signal includes noise

$$b = k * i + n$$

noise term

Deconvolution

- The OTF (Fourier of PSF) is a low-pass filter



zeros at high frequencies

- The measured signal includes noise

$$b = k * i + n$$

noise term

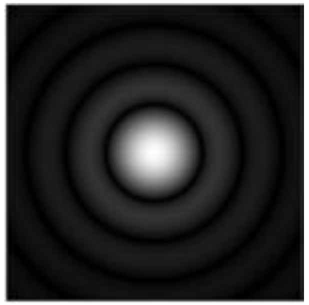
- When we divide by zero, we amplify the high frequency noise

Naïve deconvolution

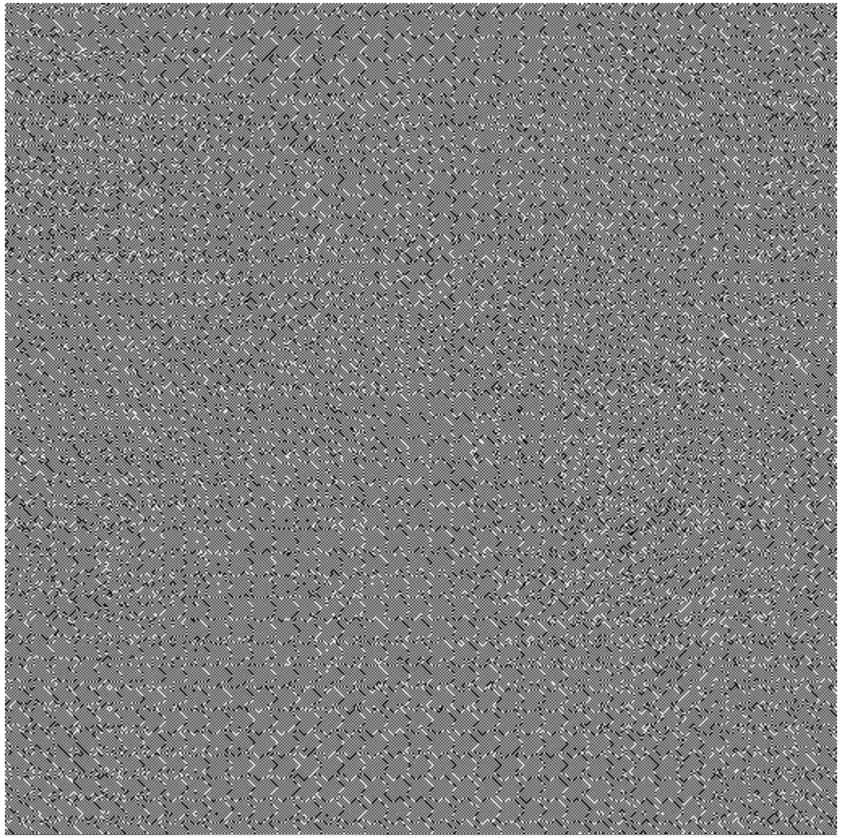
Even tiny noise can make the results awful.

- Example for Gaussian of $\sigma = 0.05$

$* \left(\right)^{-1}$



$=$



$$b \quad * \quad k^{-1} \quad = \quad i_{est}$$

Wiener Deconvolution

Apply inverse kernel and do not divide by zero:

$$i_{est} = F^{-1}\left(\boxed{\frac{|F(k)|^2}{|F(k)|^2 + 1/SNR(\omega)}} \cdot \frac{F(b)}{F(k)}\right)$$

noise-dependent damping factor

- Derived as solution to maximum-likelihood problem under Gaussian noise assumption
- Requires noise of signal-to-noise ratio at each frequency

$$SNR(\omega) = \frac{\text{signal variance at } \omega}{\text{noise variance at } \omega}$$

Wiener Deconvolution

Apply inverse kernel and do not divide by zero:

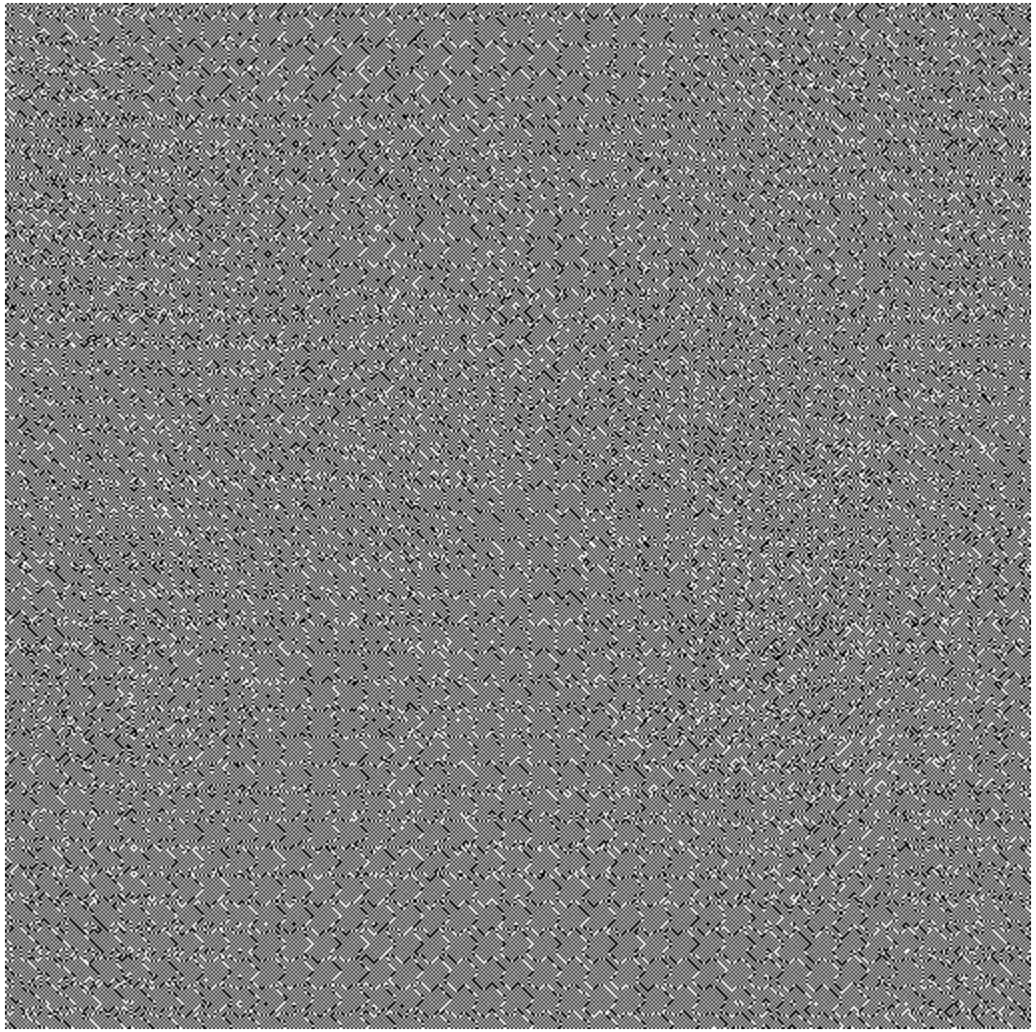
$$i_{est} = F^{-1}\left(\boxed{\frac{|F(k)|^2}{|F(k)|^2 + 1/SNR(\omega)}} \cdot \frac{F(b)}{F(k)} \right)$$

noise-dependent damping factor

Intuitively:

- When SNR is high (low or no noise), just divide by kernel.
- When SNR is low (high noise), just set to zero.

Deconvolution comparisons



naïve deconvolution

Wiener deconvolution

Deconvolution comparisons

σ = 0.01

σ = 0.05

σ = 0.01

Derivation

Sensing model:

$$b = k * i + n$$

Noise $\mathbf{n}$ is assumed to be zero-mean and independent of signal $\mathbf{i}$.

Derivation

Sensing model:

$$b = k * i + n$$

Noise $\mathbf{n}$ is assumed to be zero-mean and independent of signal $\mathbf{i}$.

Fourier transform:

$$B = K \cdot I + N$$

Why multiplication?

Derivation

Sensing model:

$$b = k * i + n$$

Noise $\mathbf{n}$ is assumed to be zero-mean and independent of signal $\mathbf{i}$.

Fourier transform:

$$B = K \cdot I + N$$

Convolution becomes multiplication.

Problem statement: Find function $\mathrm{H}(\omega)$ that minimizes expected error in Fourier domain.

$$\min_H E[\|I - HB\|^2]$$

Derivation

Replace B and re-arrange loss:

$$\min_H E[\|(1 + HK)I - HN\|^2]$$

Expand the squares:

$$\min_H \|1 - HK\|^2 E[\|I\|^2] - 2(1 - HK)E[IN] + \|H\|^2 E[\|N\|^2]$$

Derivation

When handling the cross terms:

- Can I write the following?

$$E[IN] = E[I]E[N]$$

Derivation

When handling the cross terms:

- Can I write the following?

$$E[IN] = E[I]E[N]$$

Yes, because I and N are assumed independent.

- What is this expectation product equal to?

Derivation

When handling the cross terms:

- Can I write the following?

$$E[IN] = E[I]E[N]$$

Yes, because I and N are assumed independent.

- What is this expectation product equal to?

Zero, because N has zero mean.

Derivation

Replace B and re-arrange loss:

$$\min_H E[\|(1 + HK)I - HN\|^2]$$

Expand the squares:

$$\min_H \|1 - HK\|^2 E[\|I\|^2] - 2(1 - HK)E[IN] + \|H\|^2 E[\|N\|^2]$$

cross-term is zero

Simplify:

$$\min_H \|1 - HK\|^2 E[\|I\|^2] + \|H\|^2 E[\|N\|^2]$$

How do we solve this optimization problem?

Derivation

Differentiate loss with respect to H, set to zero, and solve for H:

$$\frac{\partial \text{loss}}{\partial H} = 0$$

$$\Rightarrow -2(1 - HK)E[\|I\|^2] + 2HE[\|N\|^2] = 0$$

$$\Rightarrow H = \frac{KE[\|I\|^2]}{K^2E[\|I\|^2] + E[\|N\|^2]}$$

Divide both numerator and denominator with $E[\|I\|^2]$, extract factor $1/\mathrm{K}$, and done!

Wiener Deconvolution

Apply inverse kernel and do not divide by zero:

$$i_{est} = F^{-1}\left(\frac{|F(k)|^2}{|F(k)|^2 + 1/SNR(\omega)} \cdot \frac{F(b)}{F(k)} \right)$$

noise-dependent damping factor

- Derived as solution to maximum-likelihood problem under Gaussian noise assumption
- Requires estimate of signal-to-noise ratio at each frequency

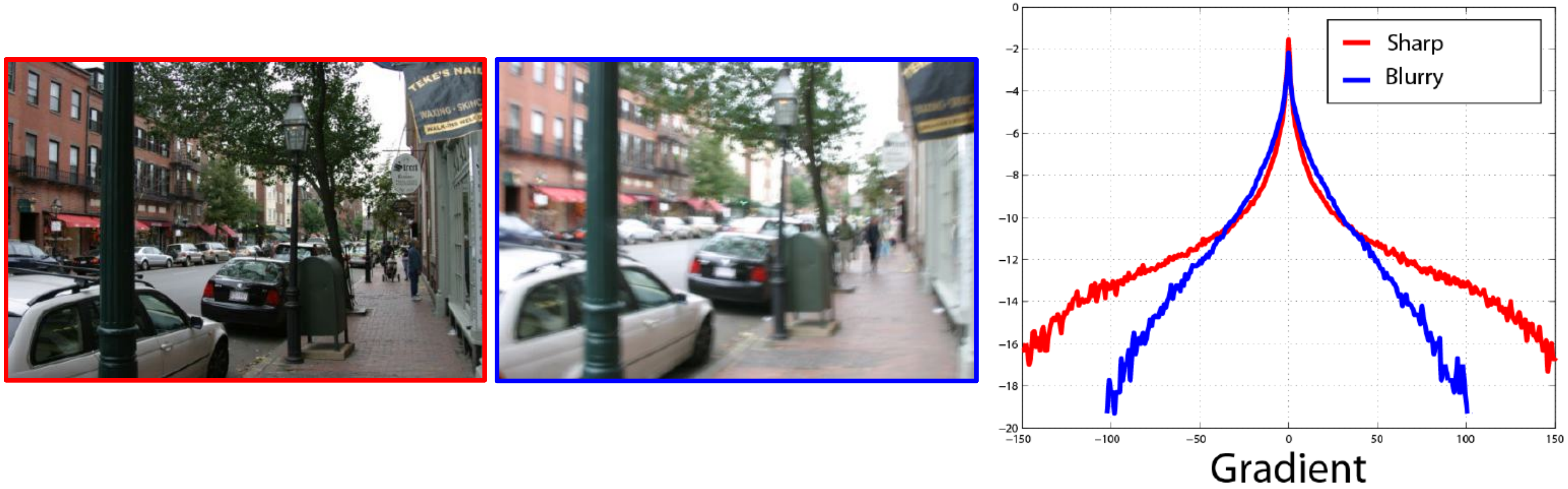
$$SNR(\omega) = \frac{\text{signal variance at } \omega}{\text{noise variance at } \omega}$$

Natural Image Statistics

- Non-blind deconvolution: ill-posed problem
- We need to assume something on the latent image to constrain the problem.

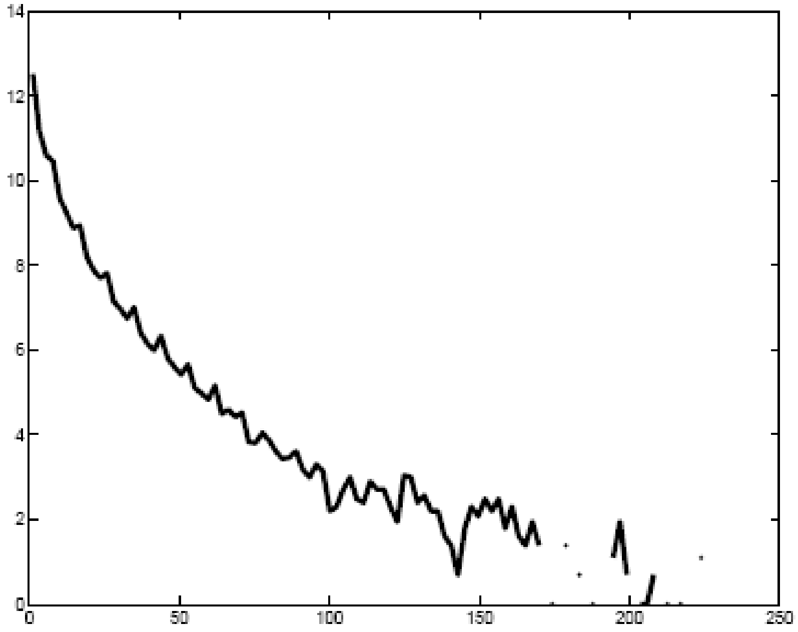
Natural Image Statistics

- Natural images have a heavy-tailed distribution on gradient magnitudes
  - Mostly zero & a few edges
  - Levin et al. SIGGRAPH 2007, Shan et al. SIGGRAPH 2008, Krishnan & Fergus, NIPS 2009

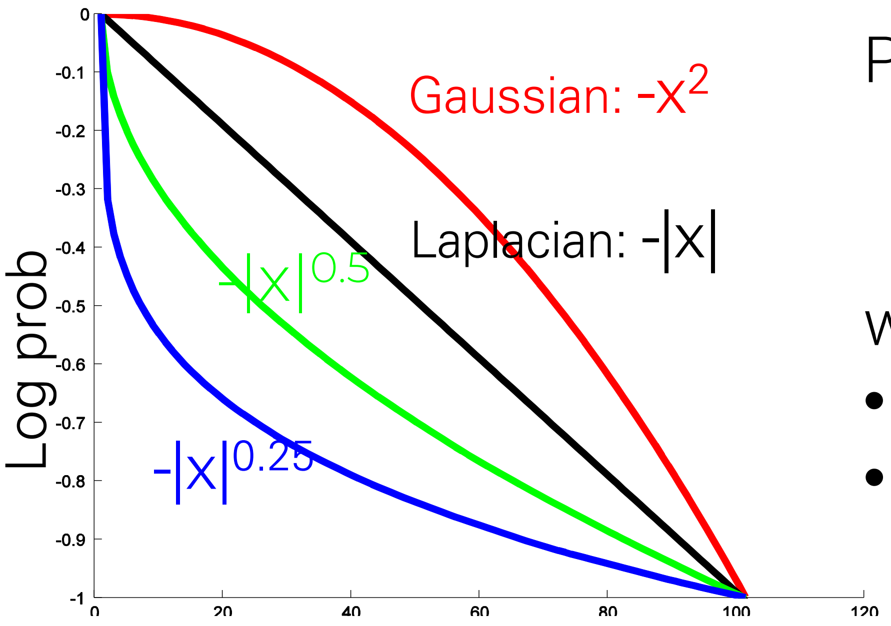


Natural Image Statistics

- Levin et al. SIGGRAPH 2007
  - Propose a parametric model for natural image priors based on image gradients



Derivative histogram from a natural image



Parametric models

Proposed prior

$$\log p(x) = -\sum_i |\nabla x_i|^\alpha$$

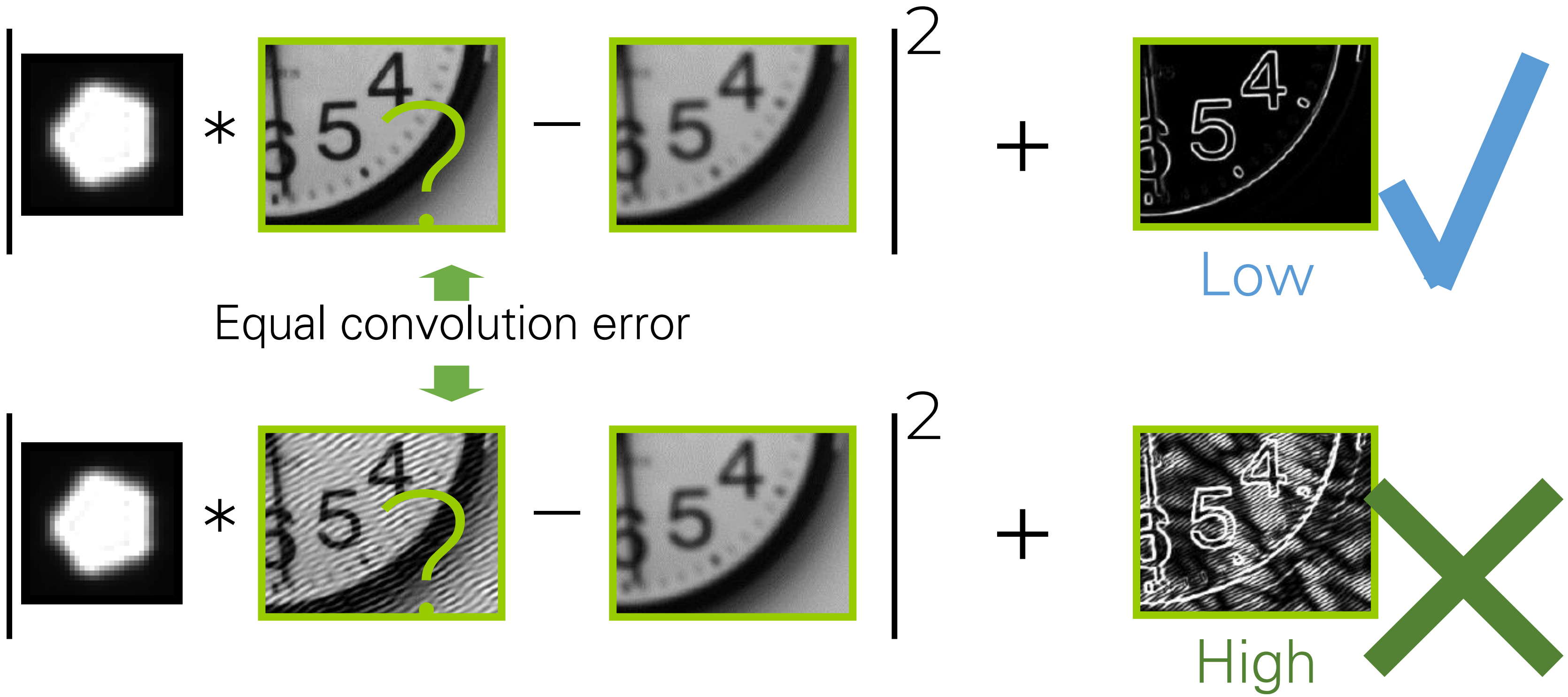
where:

- x: image
- α: model parameter, $\alpha < 1$

Natural Image Statistics

- Levin et al. SIGGRAPH 2007

$$l = \arg\min_{l} \{\underbrace{\|k * l - b\|^2}_{\text{Data term}} + \underbrace{\lambda \textstyle\sum_i |\nabla l_i|^{\alpha}}_{\text{Prior}}\} \qquad (\alpha < 1)$$



Natural Image Statistics

- Levin et al. SIGGRAPH 2007

Input

Richardson-Lucy

"spread" gradients

Gaussian prior

$$\sum_i |\nabla l_i|^2$$

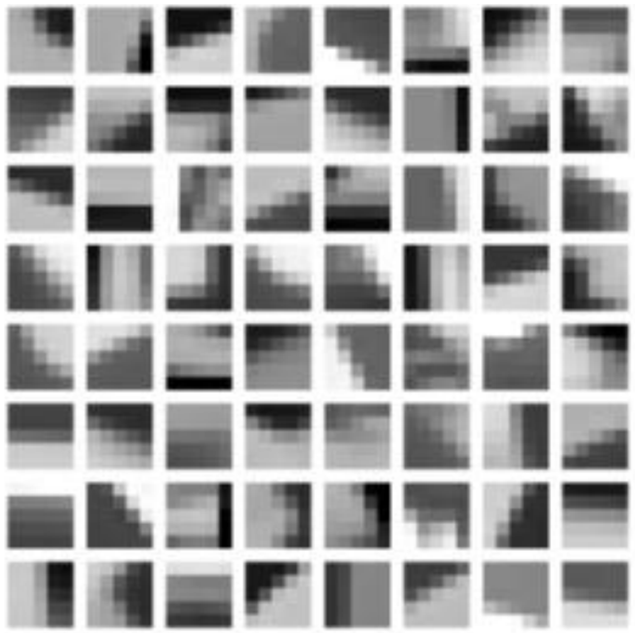
"localizes" gradients

Sparse prior

$$\sum_i |\nabla l_i|^{0.8}$$

High-order Natural Image Priors

- Patches, large neighborhoods, …
- Effective for various kinds of image restoration problems
  - Denoising, inpainting, super-resolution, deblurring, …

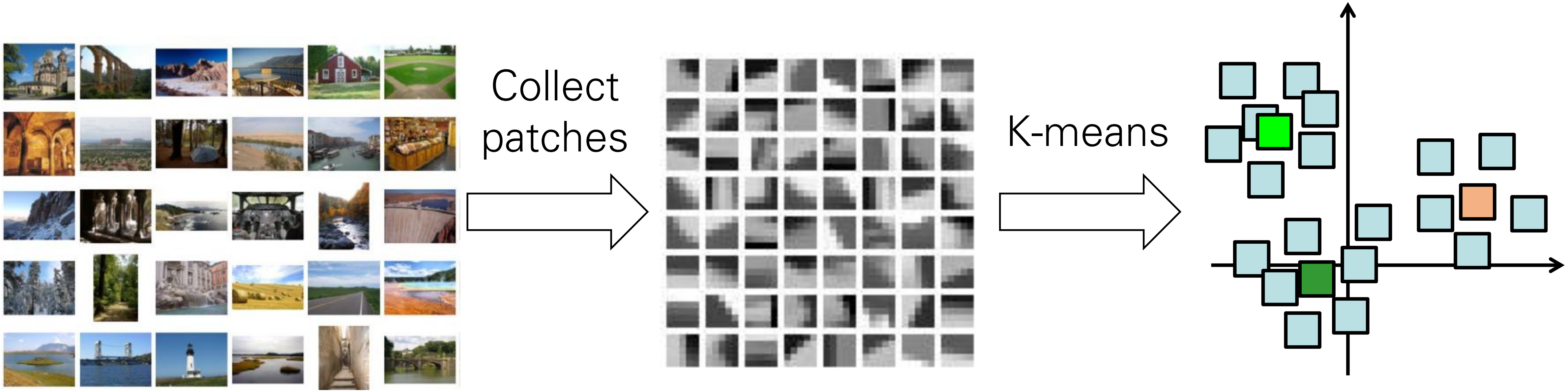


High-order Natural Image Priors

- Schmidt et al. CVPR 2011
  - Fields of Experts
- Zoran & Weiss, ICCV 2011
  - Trained Gaussian mixture model for natural image patches
- Schuler et al. CVPR 2013
  - Trained Multi-layer perceptron to remove artifacts and to restore sharp patches
- Schmidt et al. CVPR 2013
  - Trained regression tree fields for 5x5 neighborhoods

High-order Natural Image Priors

- Zoran & Weiss, ICCV 2011
  - Gaussian Mixture Model (GMM) learned from natural images



Natural images | Collected patches | GMM

High-order Natural Image Priors

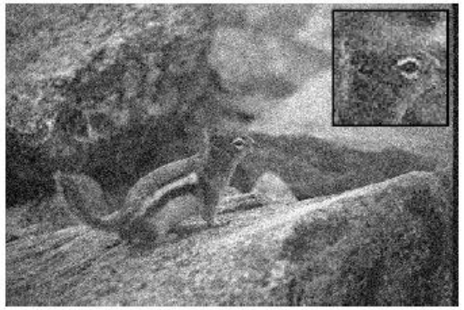
- Zoran & Weiss, ICCV 2011
  - Given a patch, we can compute its likelihood based on the GMM.
  - Deconvolution can be done by solving:

$$\arg\min_{l}\left\{\|k * l - b\|^2 - \lambda \sum_i \underbrace{\log p(l_i)}_{\text{Log-likelihood of a patch } l_i \text{ at } i\text{-th pixel based on GMM}}\right\}$$

High-order Natural Image Priors

- Zoran & Weiss, ICCV 2011

Denoising



(a) Noisy Image - PSNR: 20.17

(b) KSVD - PSNR: 28.72

(c) LLSC - PSNR: 29.30

(d) EPLL GMM - PSNR: 29.39

Deblurring

Blurred image

Krishnan & Fergus
PSNR: 26.38

Zoran & Weiss
PSNR: 27.70

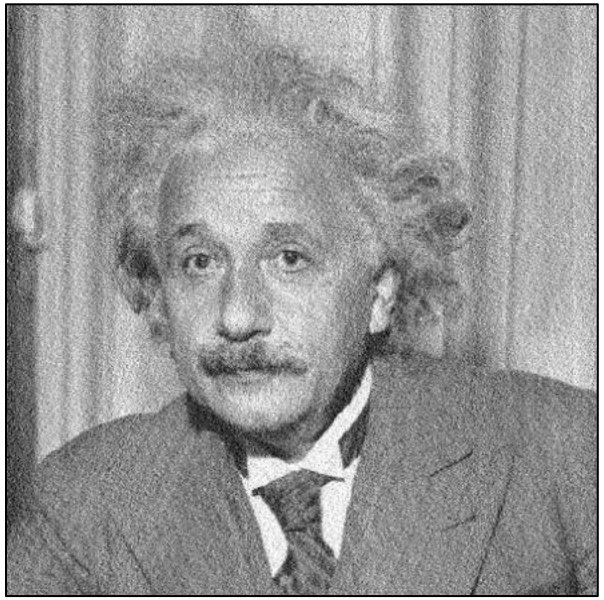
Ringing Artifacts

- Wave-like artifacts around strong edges
- Caused by
  - Inaccurate blur kernels
  - Nonlinear response curve
  - Etc…

Ringing Artifacts

- Noise
  - High-freq
  - Independent and identical distribution
  - Priors on image gradients work well

- Ringing
  - Mid-freq
  - Spatial correlation
  - Priors on image gradients are not very effective



Ringing Artifacts

- Yuan et al. SIGGRAPH 2007
  - Residual deconvolution & de-ringing
- Yuan et al. SIGGRAPH 2008
  - Multi-scale deconvolution framework based on residual deconvolution

Blurred image | Richardson-Lucy | Yuan et al. SIGGRAPH 2008

Residual Deconvolution [Yuan et al. SIGGRAPH 2007, 2008]

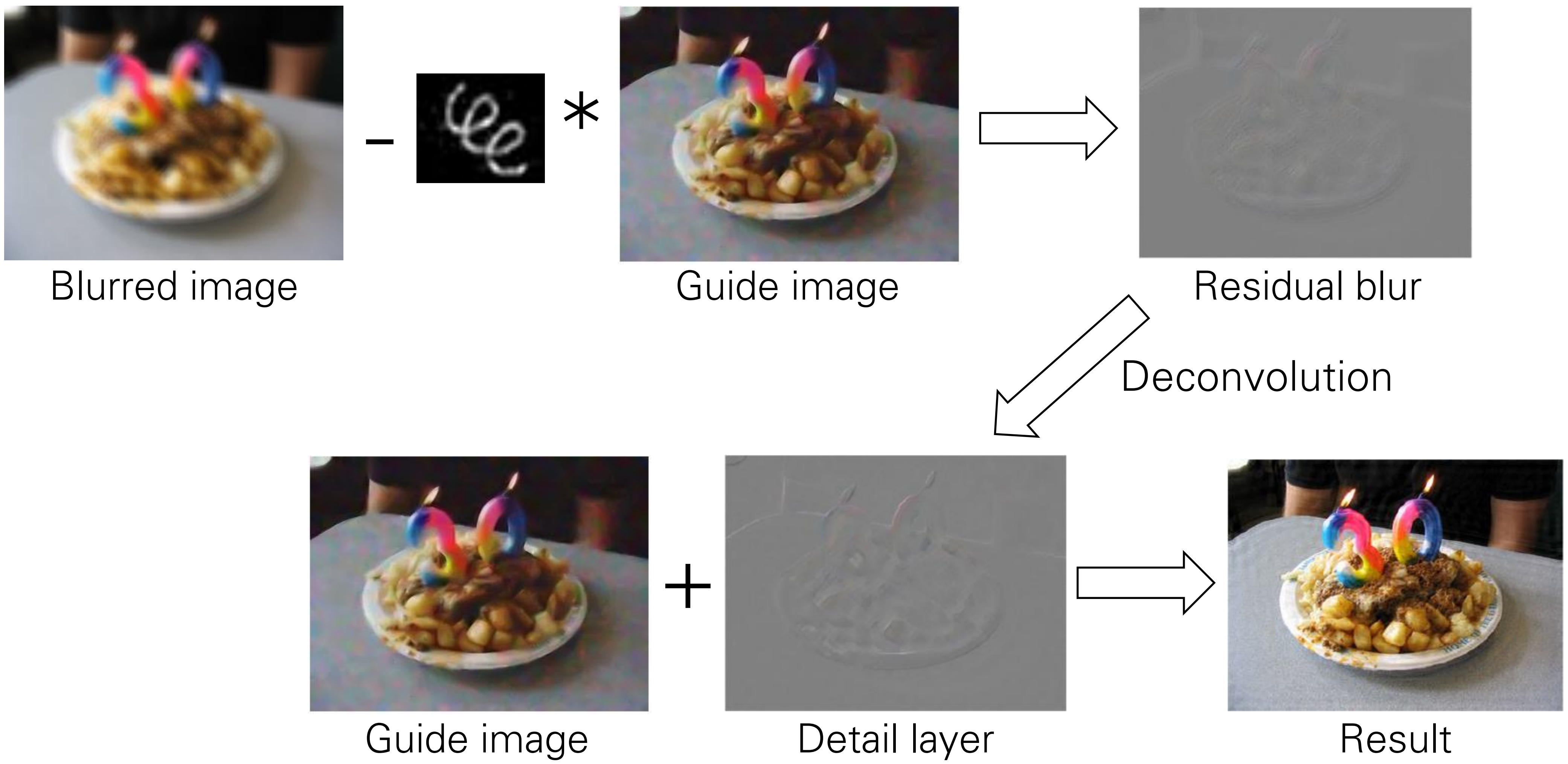
Blurred image

Guide image

Residual deconvolution result
with less ringing artifacts

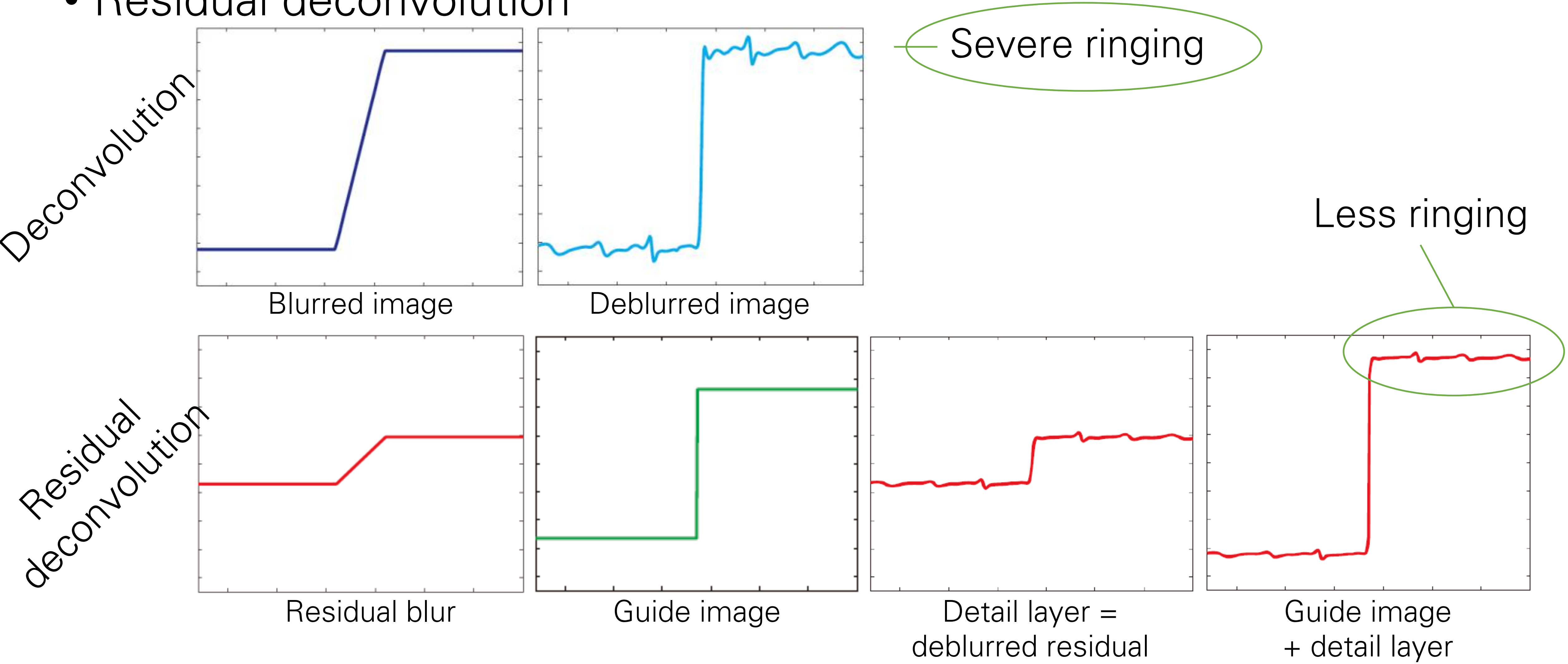
- Relatively accurate edges, but less details
- Obtained from a deconvolution result from a smaller scale

Residual Deconvolution [Yuan et al. SIGGRAPH 2007, 2008]



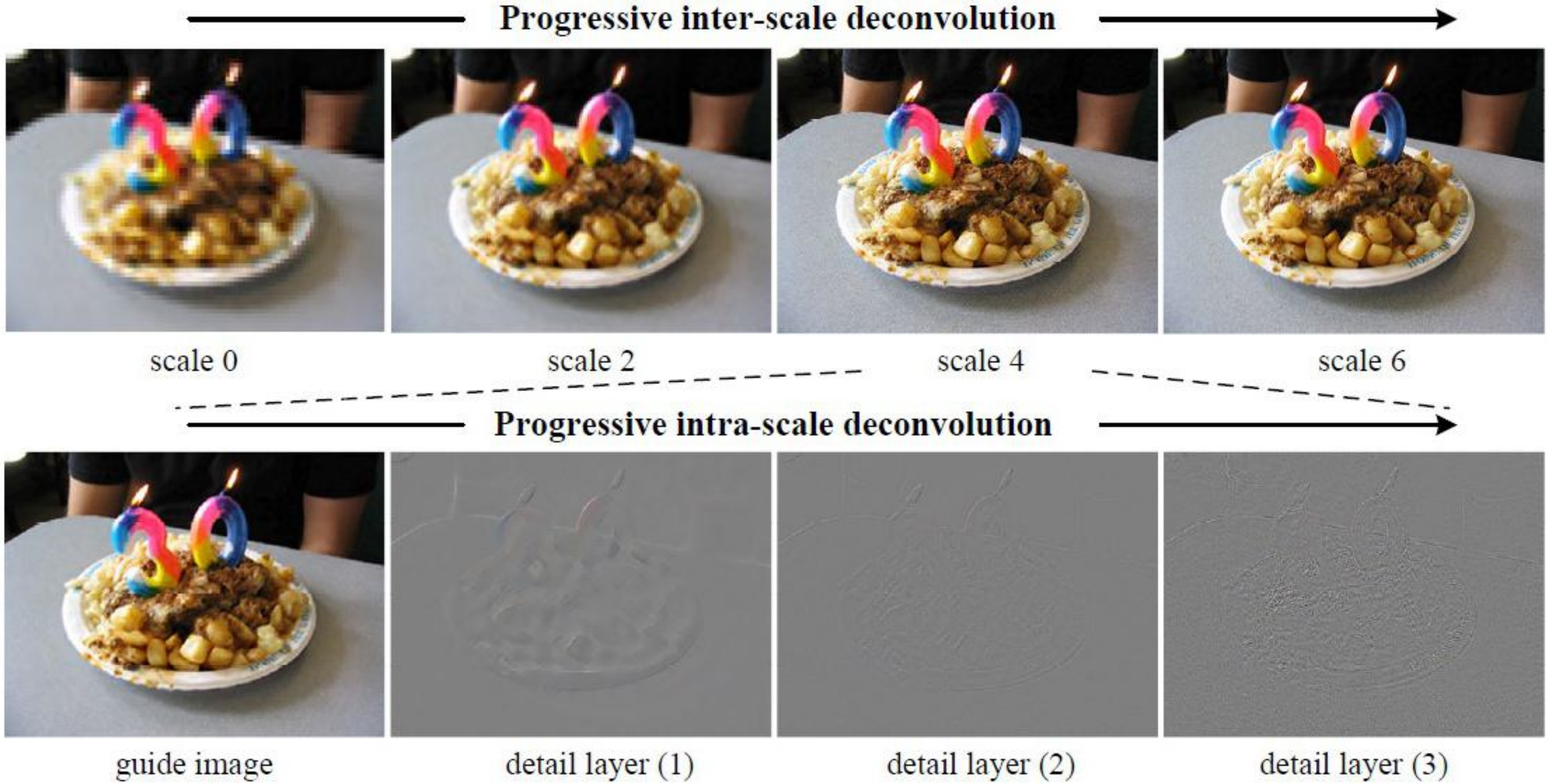
Residual Deconvolution [Yuan et al. SIGGRAPH 2007, 2008]

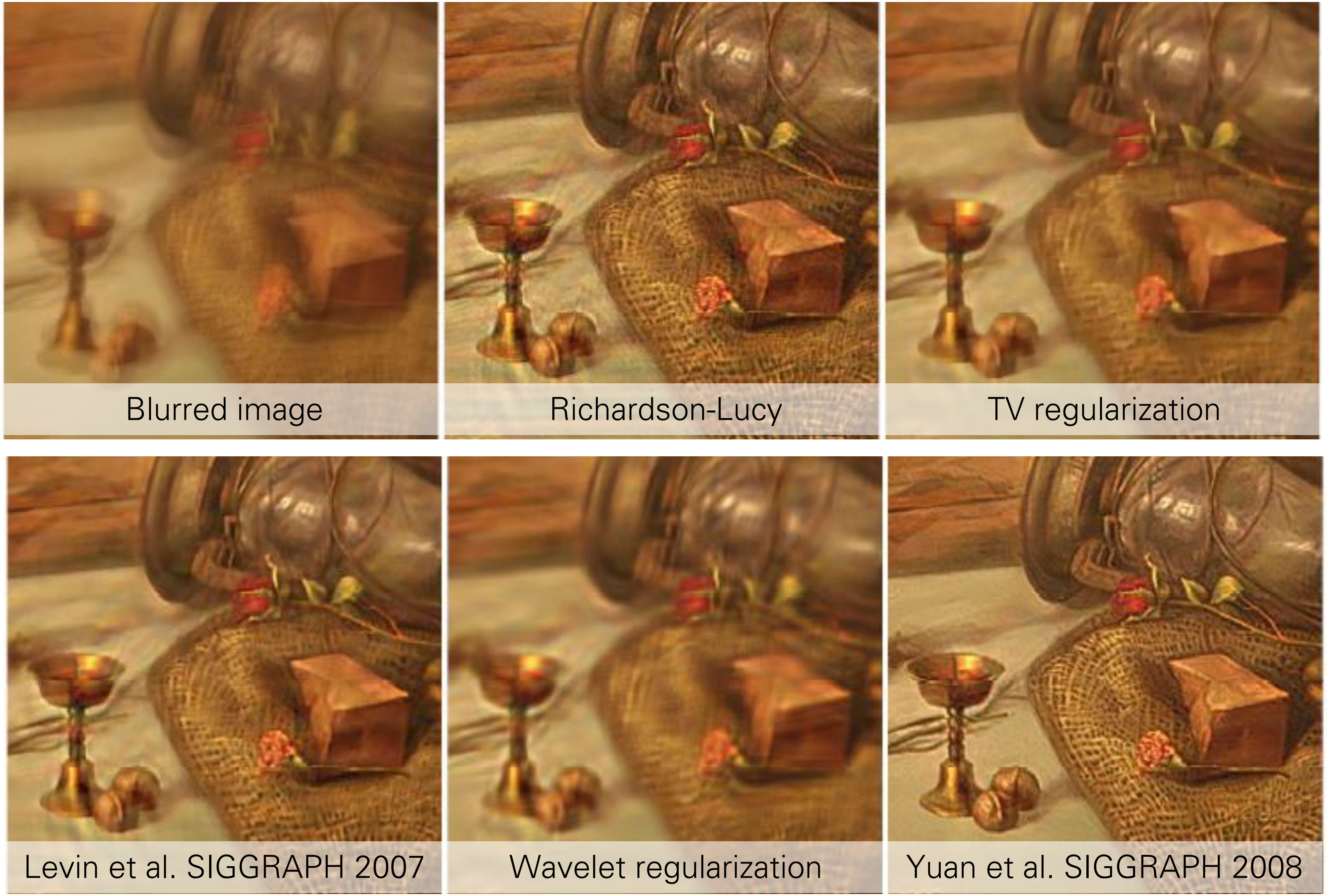
- Residual deconvolution



Progressive Inter-scale & Intra-scale Deconvolution [Yuan et al. SIGGRAPH 2008]

- Progressive inter-scale & intra-scale deconvolution





Blurred image | Richardson-Lucy | TV regularization

Levin et al. SIGGRAPH 2007 | Wavelet regularization | Yuan et al. SIGGRAPH 2008

Outliers

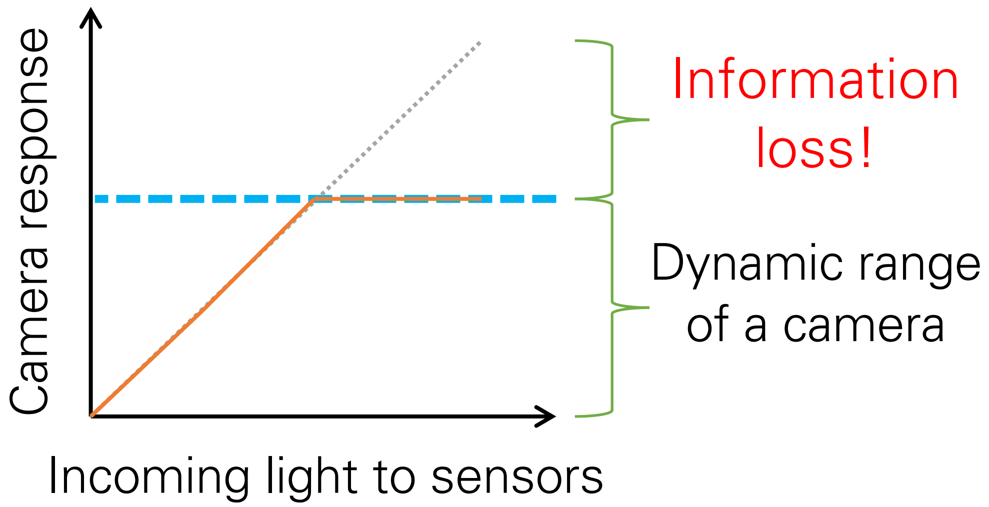
- A main source of severe ringing artifacts

Blurred image with outliers

Deblurring result
[Levin et al. SIGGRAPH 2007]

Outliers

- Saturated pixels caused by limited dynamic range of sensors



Blurred image

[Levin et al. 2007]

Outliers

- Hot pixels, dead pixels, compression artifacts, etc…

Hot pixel

Blurred image with outliers

[Levin et al. 2007]

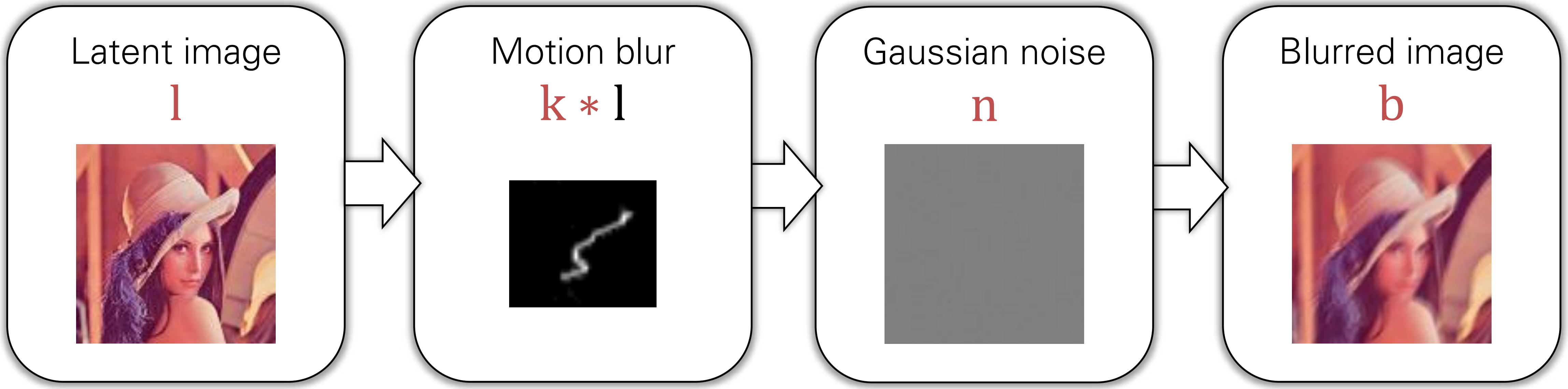
Outlier Handling

- Most common blur model:

$$b = k * l + n$$

small amount of Gaussian noise

Equivalent to



Outlier Handling

- An energy function derived from this model:

$$E(l) = \underbrace{\|k * l - b\|^2}_{\substack{L^2\text{-norm based data term:}\\ \text{known to be vulnerable to}\\ \text{outliers}}} + \underbrace{\rho(l)}_{\substack{\text{Regularization term on}\\ \text{a latent image } l}}$$

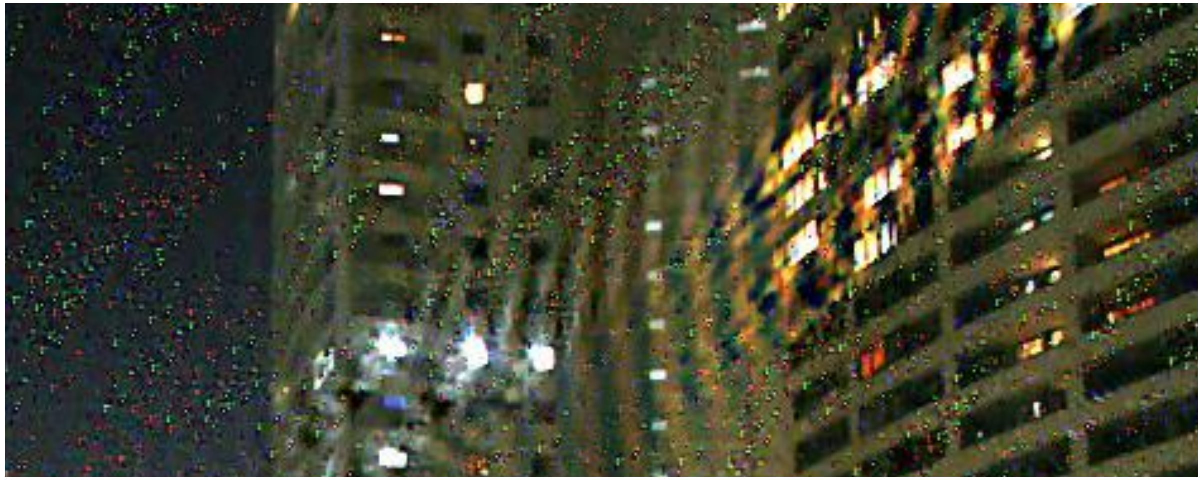
- More robust norms to outliers
  - L^1-norm, other robust statistics…

$$E(l) = \|k * l - b\|_1 + \rho(l)$$

  - Bar et al. IJCV 2006, Xu et al. ECCV 2010, …

Outlier Handling

- L^1-norm based data term
  - Simple & efficient
  - Effective on salt & pepper noise
  - Not effective on saturated pixels



L^2-norm based data term

L^1-norm based data term

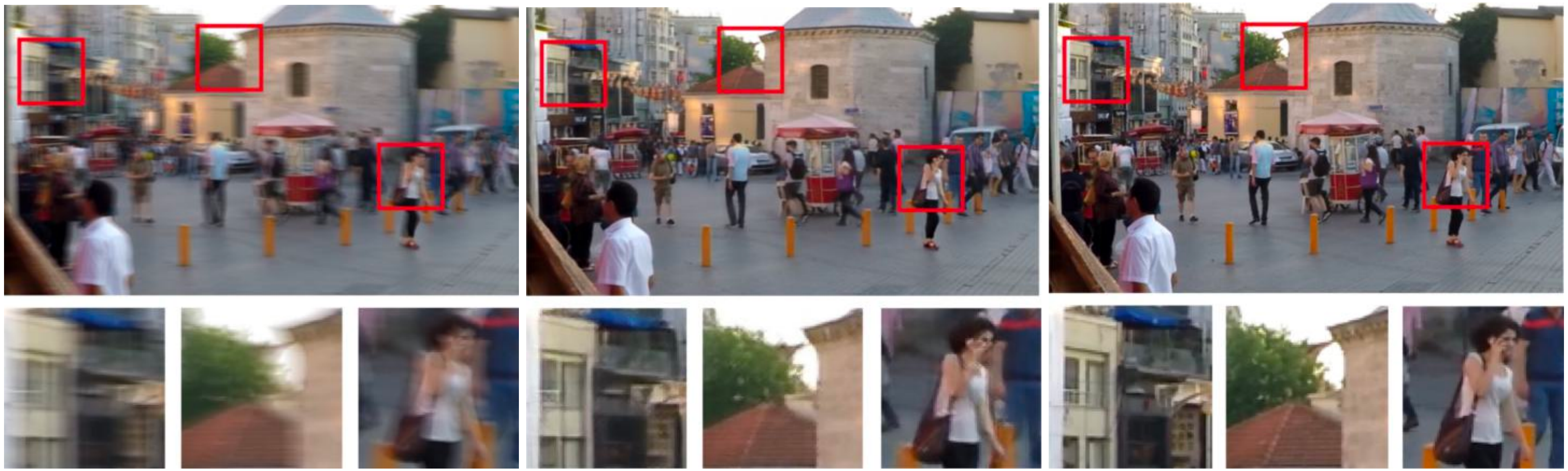
Modern Approaches

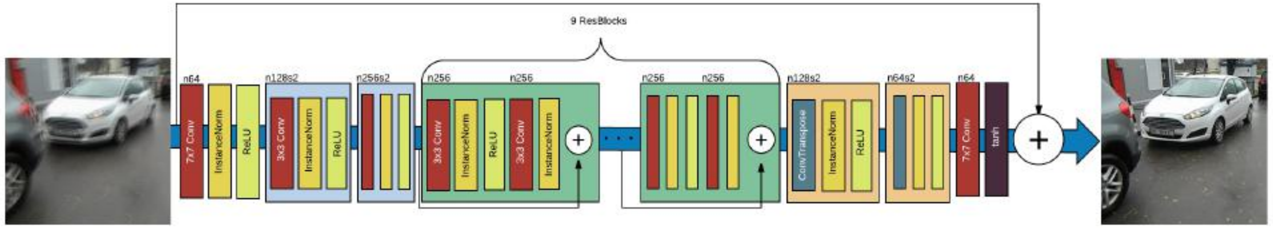
DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks

Orest Kupyn[1,3], Volodymyr Budzan[1,3], Mykola Mykhailych[1], Dmytro Mishkin[2], Jiři Matas[2]

[1] Ukrainian Catholic University, Lviv, Ukraine
{kupyn, budzan, mykhailych}@ucu.edu.ua

[2] Visual Recognition Group, Center for Machine Perception, FEE, CTU in Prague
{mishkdmy, matas}@cmp.felk.cvut.cz
[3] ELEKS Ltd.



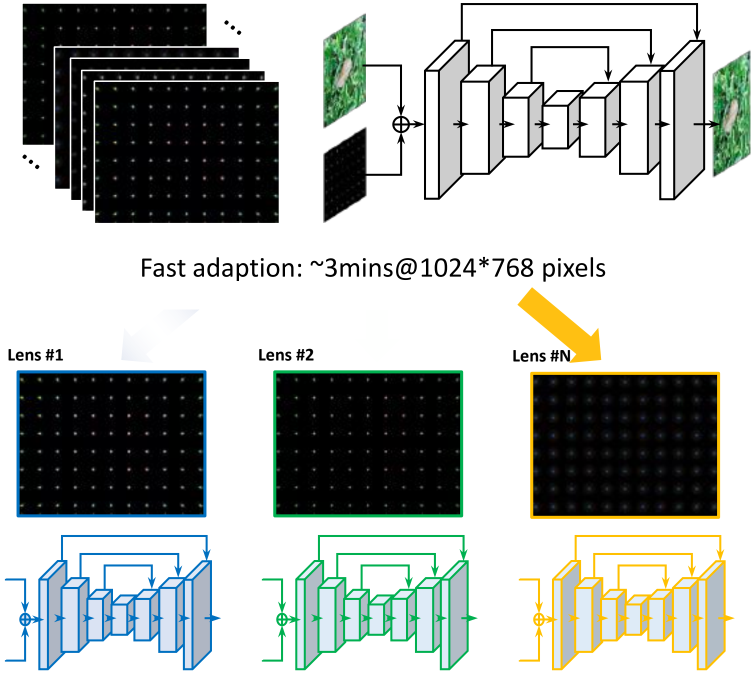


Modern Approaches

Universal and Flexible Optical Aberration Correction Using Deep-Prior Based Deconvolution

Xiu Li[1*], Jinli Suo[1], Weihang Zhang[1], Xin Yuan[2], Qionghai Dai[1]

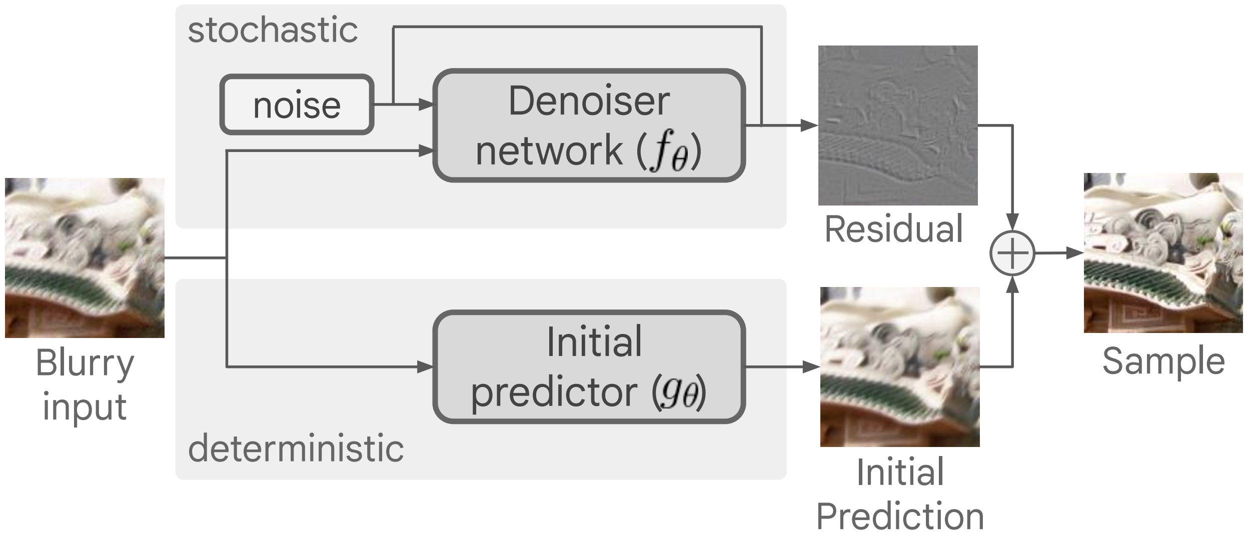
[1]Tsinghua University, [2]Westlake University

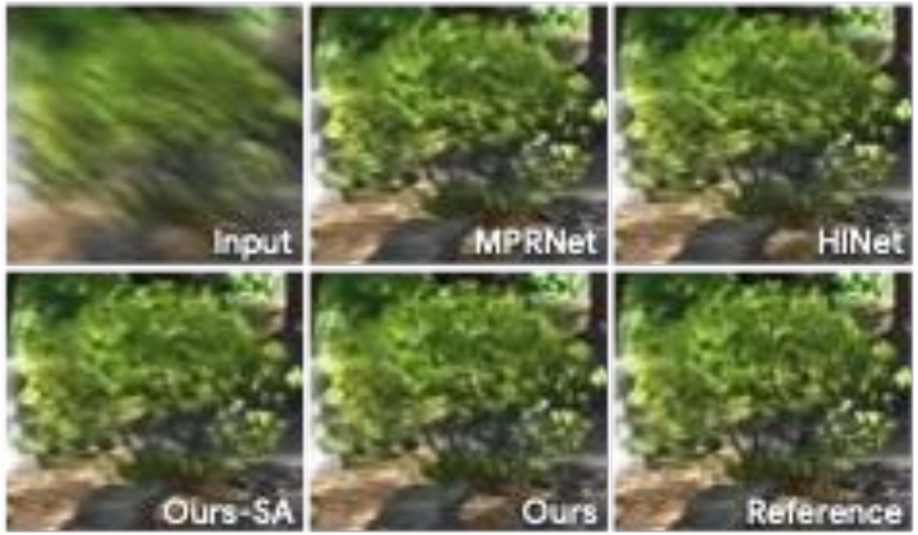


Modern Approaches

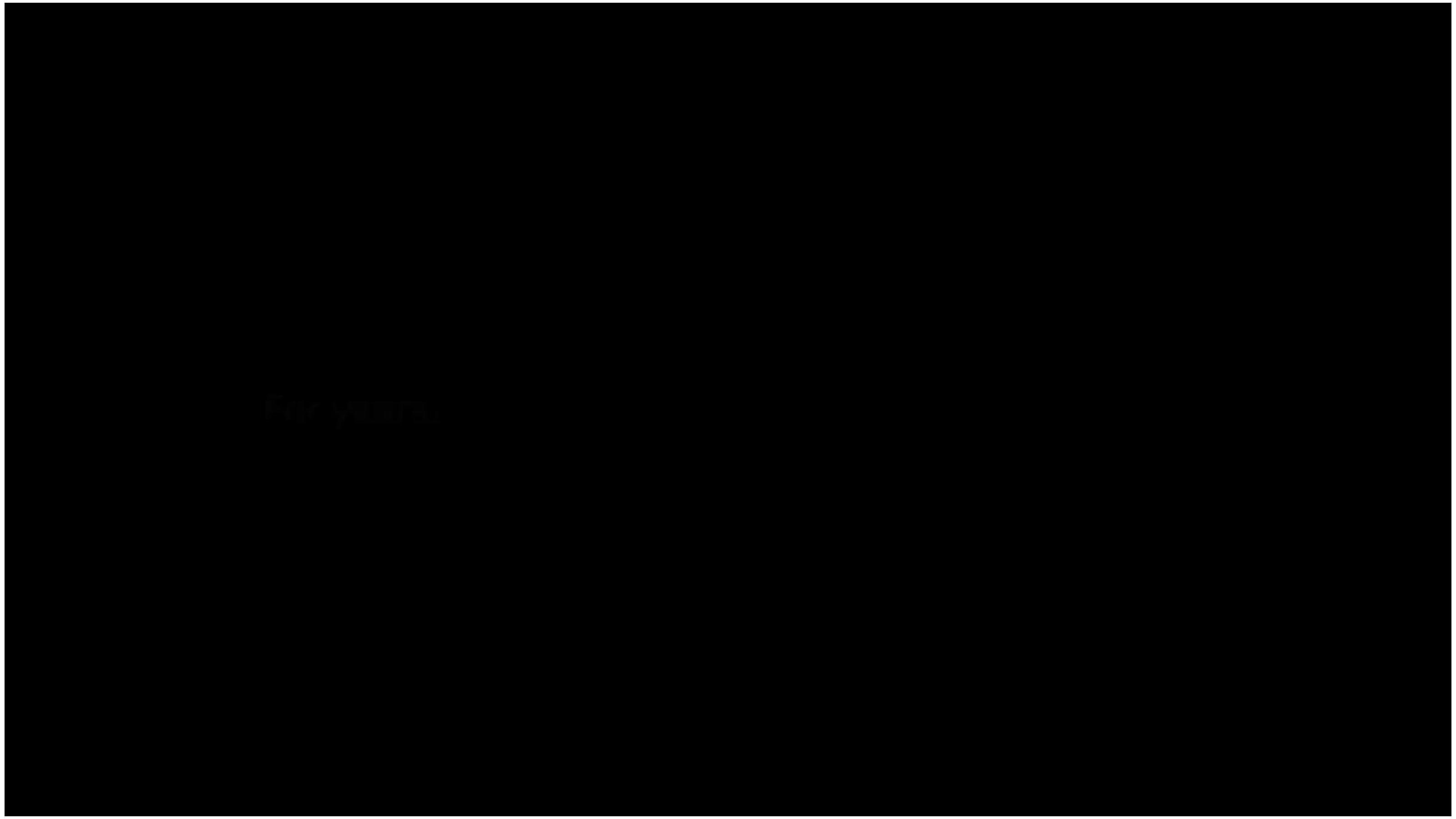
Deblurring via Stochastic Refinement

Jay Whang[†*] Mauricio Delbracio[‡] Hossein Talebi[‡] Chitwan Saharia[‡]
Alexandros G. Dimakis[†] Peyman Milanfar[‡]
[†]University of Texas at Austin [‡]Google Research





Modern Technology

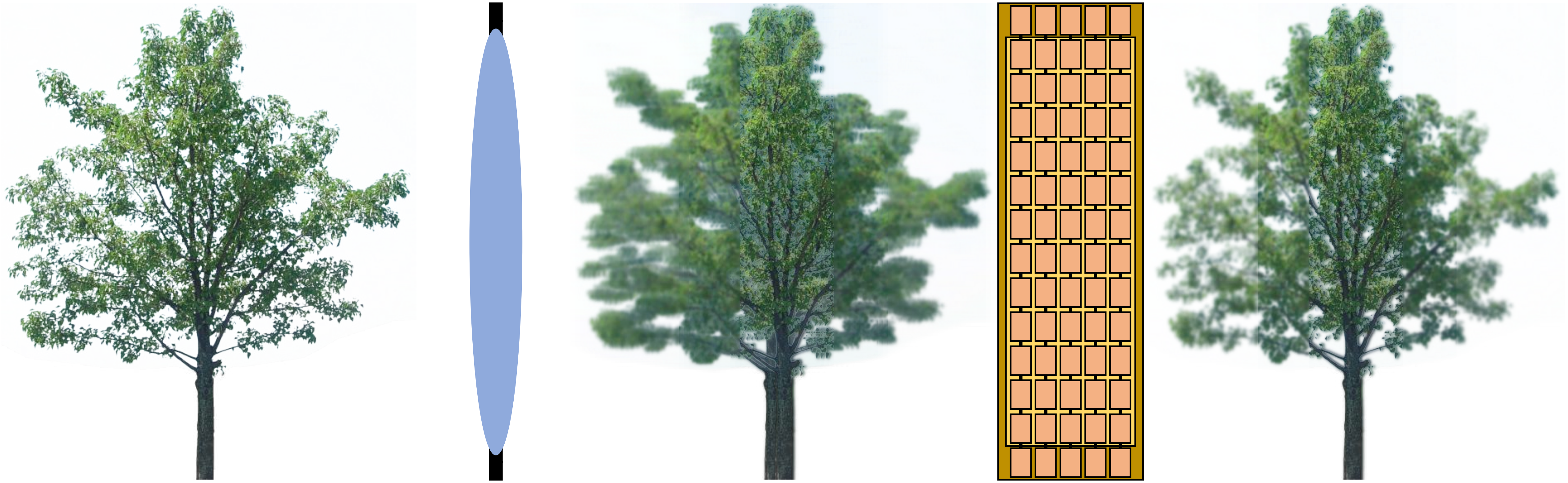


Today’s Lecture

- Deconvolution
  - Sources of blur
  - Blind deconvolution
  - Non-blind deconvolution
- Coded photography
  - The coded photography paradigm
  - Dealing with depth blur
  - Dealing with motion blur

The coded photography paradigm

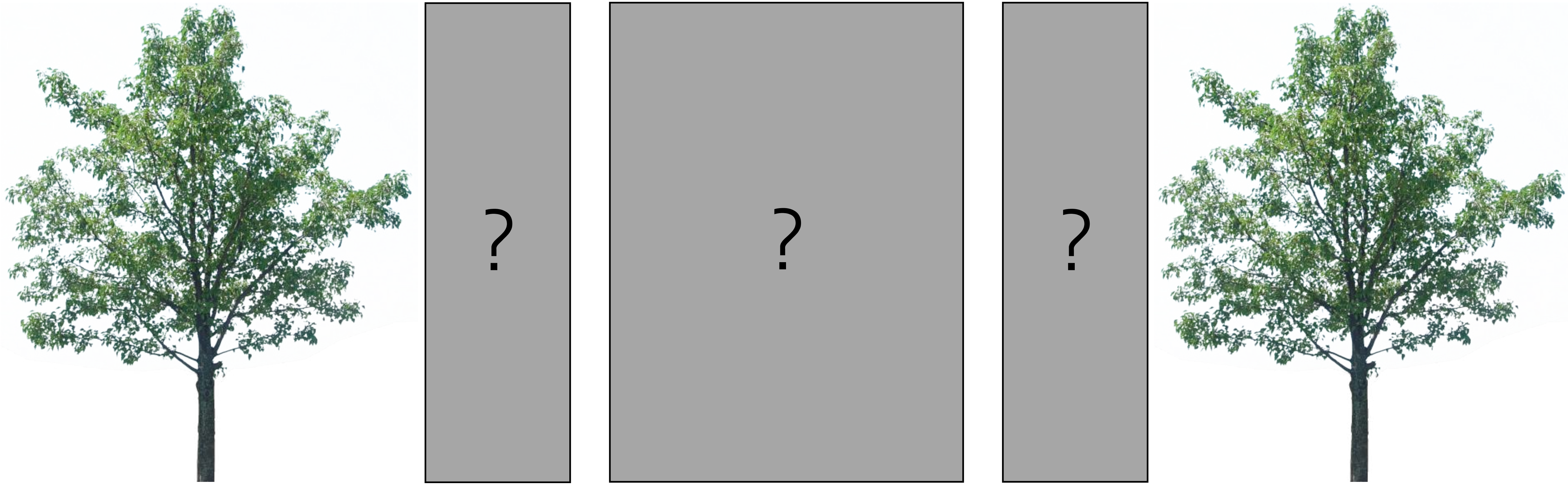
Conventional photography



real world optics captured image computation enhanced image

- Optics capture something that is (close to) the final image.
- Computation mostly "enhances" captured image (e.g., deblur).

Coded photography



real world | generalized optics | coded representation of real world | generalized computation | final image(s)

- Generalized optics encode world into intermediate representation.
- Generalized computation decodes representation into multiple images.

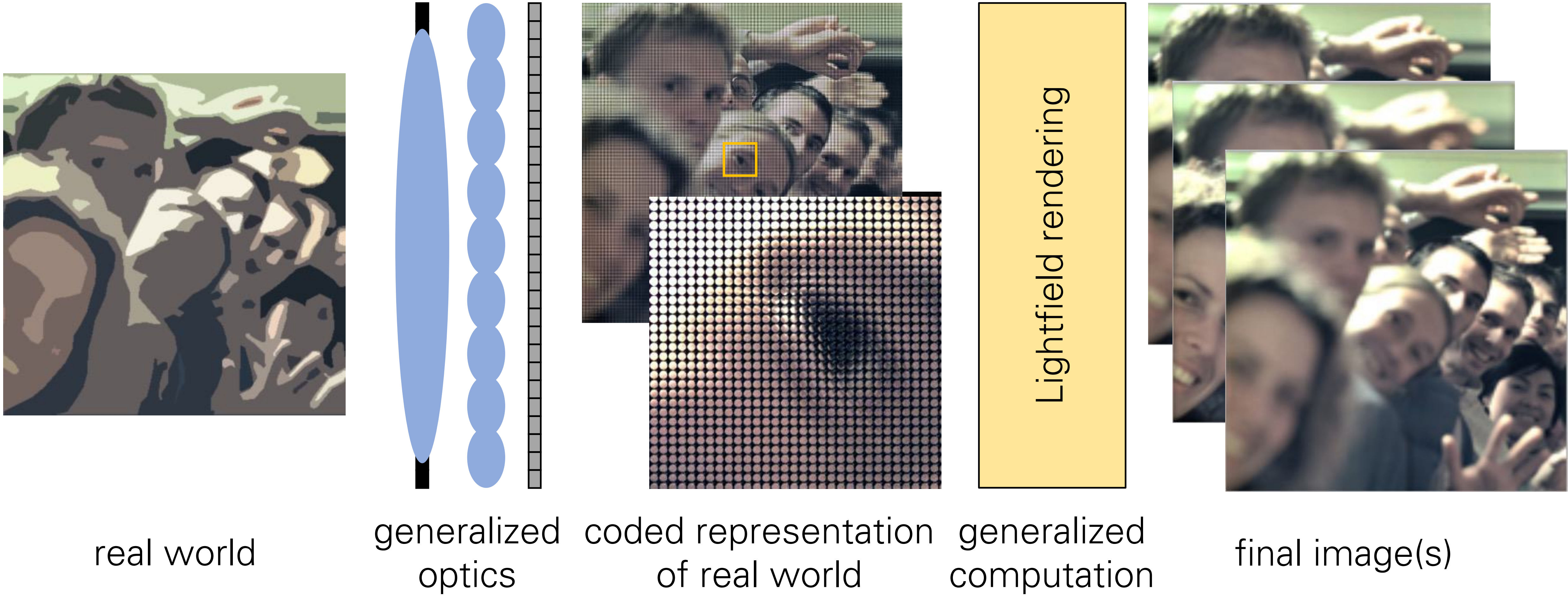
Can you think of any examples?

Early example: mosaicing

real world | generalized optics | coded representation of real world | generalized computation | final image(s)

- Color filter array encodes color into a mosaic.
- Demosaicing decodes color into RGB image.

Recent example: plenoptic camera



- Plenoptic camera encodes world into lightfield.
- Lightfield rendering decodes lightfield into refocused or multi-viewpoint images.

Why are our images blurry?

• Lens imperfections.	⟵	non-blind deconvolution	conventional photography
• Camera shake.	⟵	blind deconvolution	
• Scene motion.	⟵	flutter shutter, motion-invariant photo	coded photography
• Depth defocus.	⟵	coded aperture, focal sweep, lattice lens	

Why are our images blurry?

- Lens imperfections. ⟵ non-blind deconvolution
- Camera shake. ⟵ blind deconvolution
- Scene motion. ⟵ flutter shutter, motion-invariant photo
- Depth defocus. ⟵ coded aperture, focal sweep, lattice lens

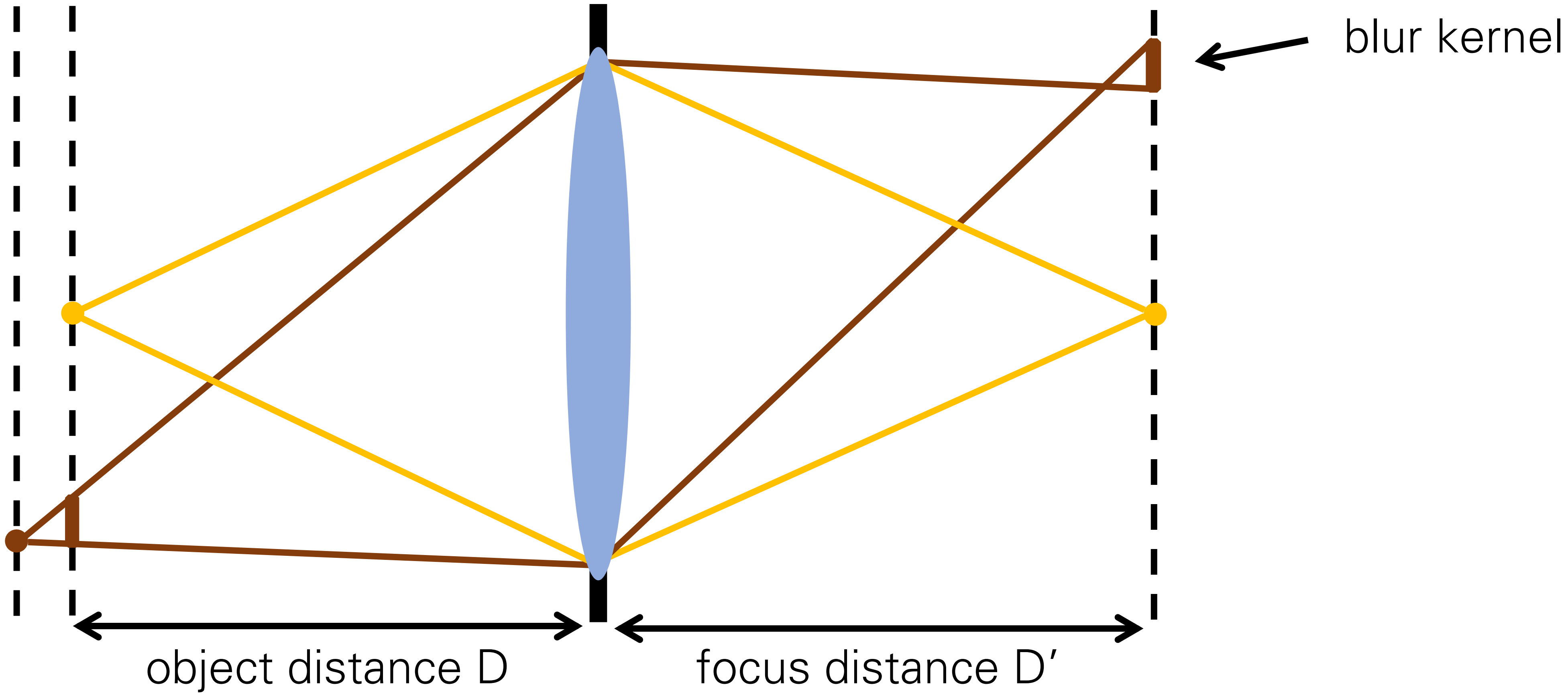
conventional photography

coded photography

Dealing with depth blur: coded aperture

Defocus blur

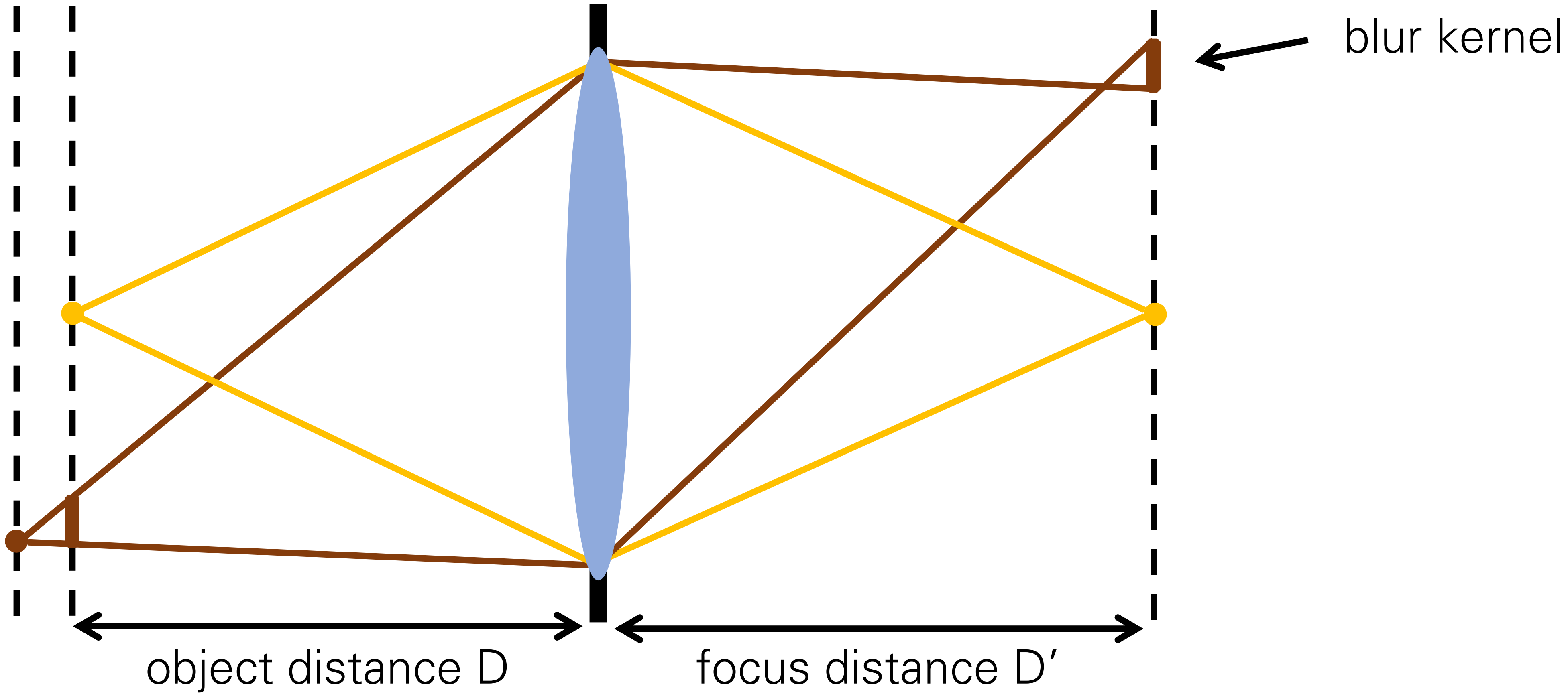
Point spread function (PSF): The blur kernel of a (perfect) lens at some out-of-focus depth.



What does the blur kernel depend on?

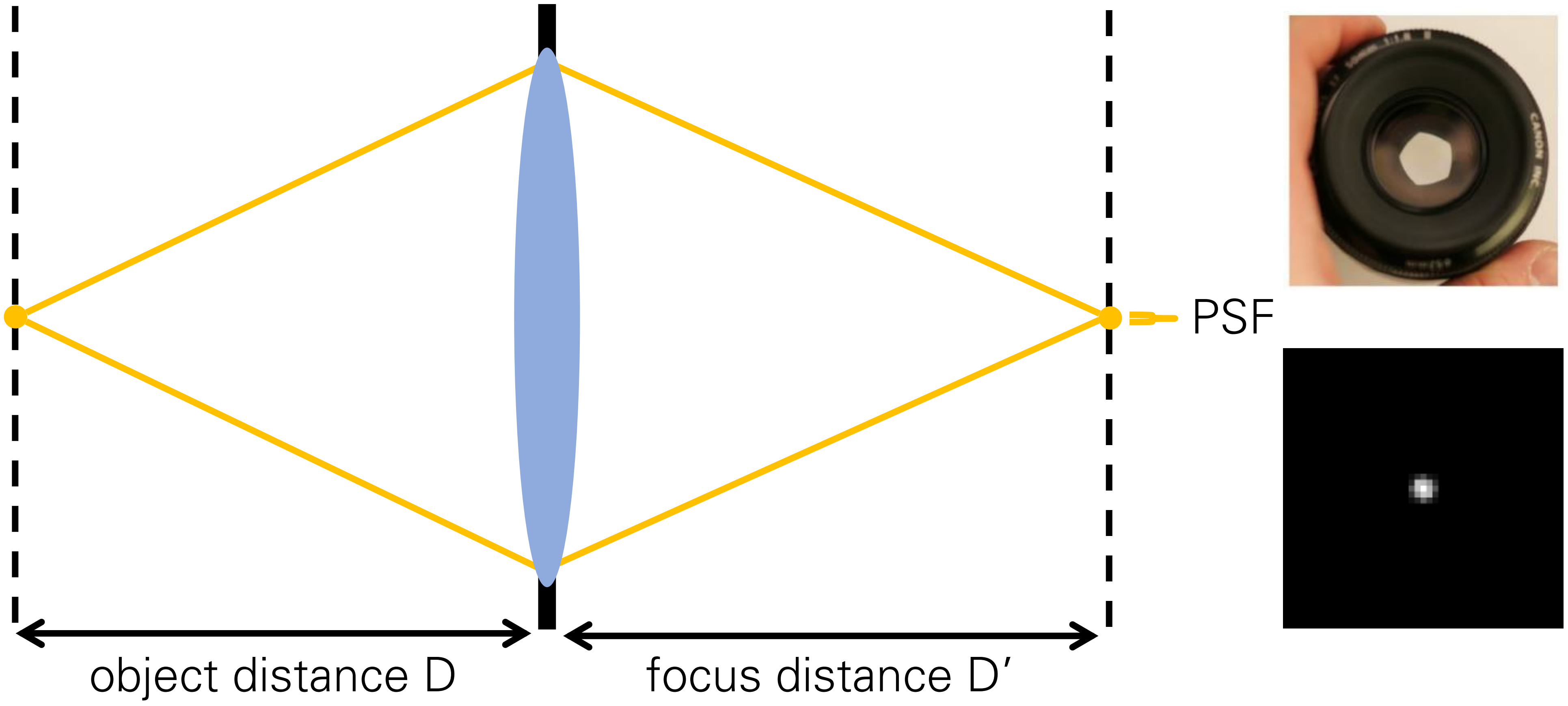
Defocus blur

Point spread function (PSF): The blur kernel of a (perfect) lens at some out-of-focus depth.

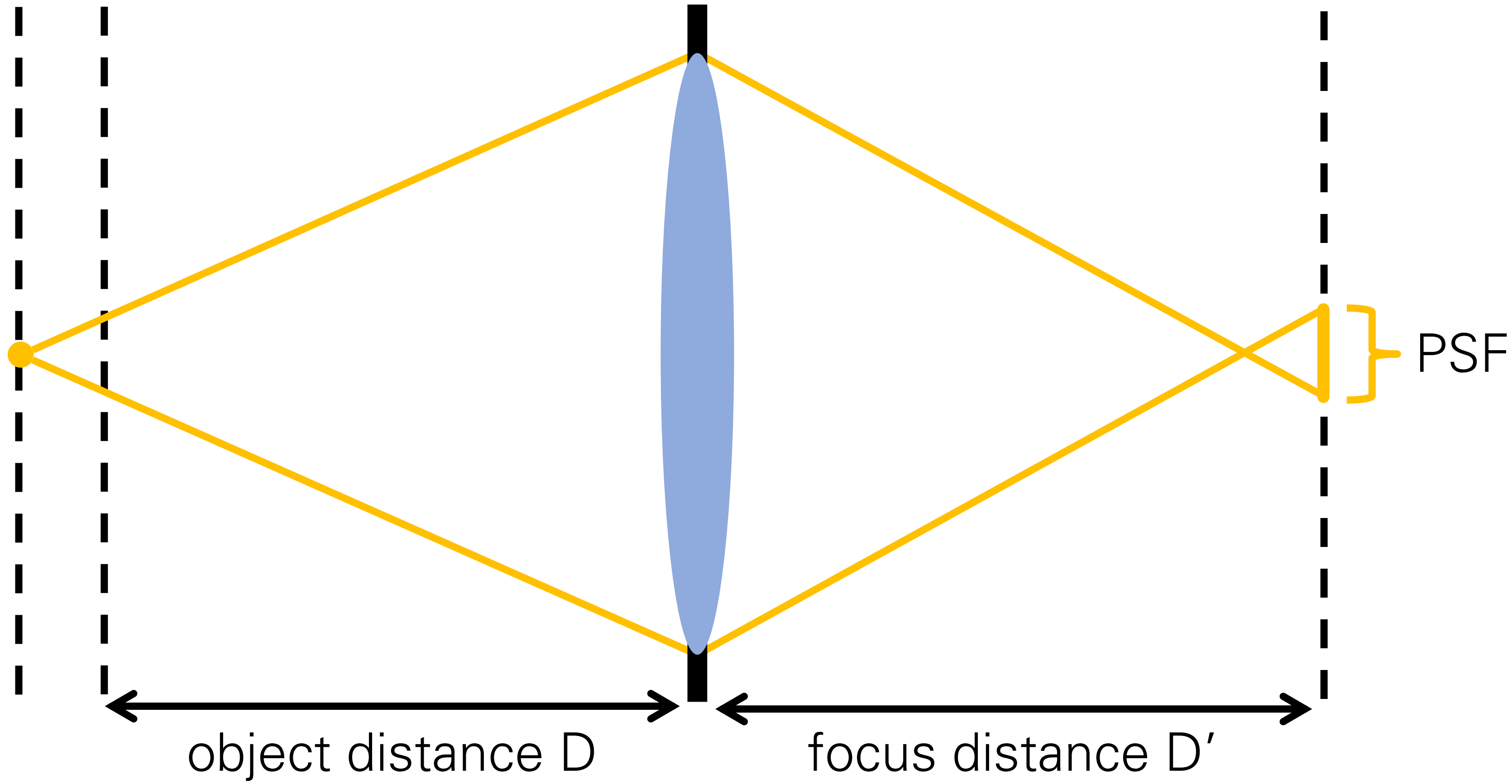


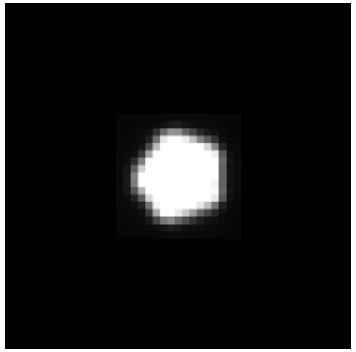
- Aperture determines shape of kernel.
- Depth determines scale of blur kernel.

Depth determines scale of blur kernel

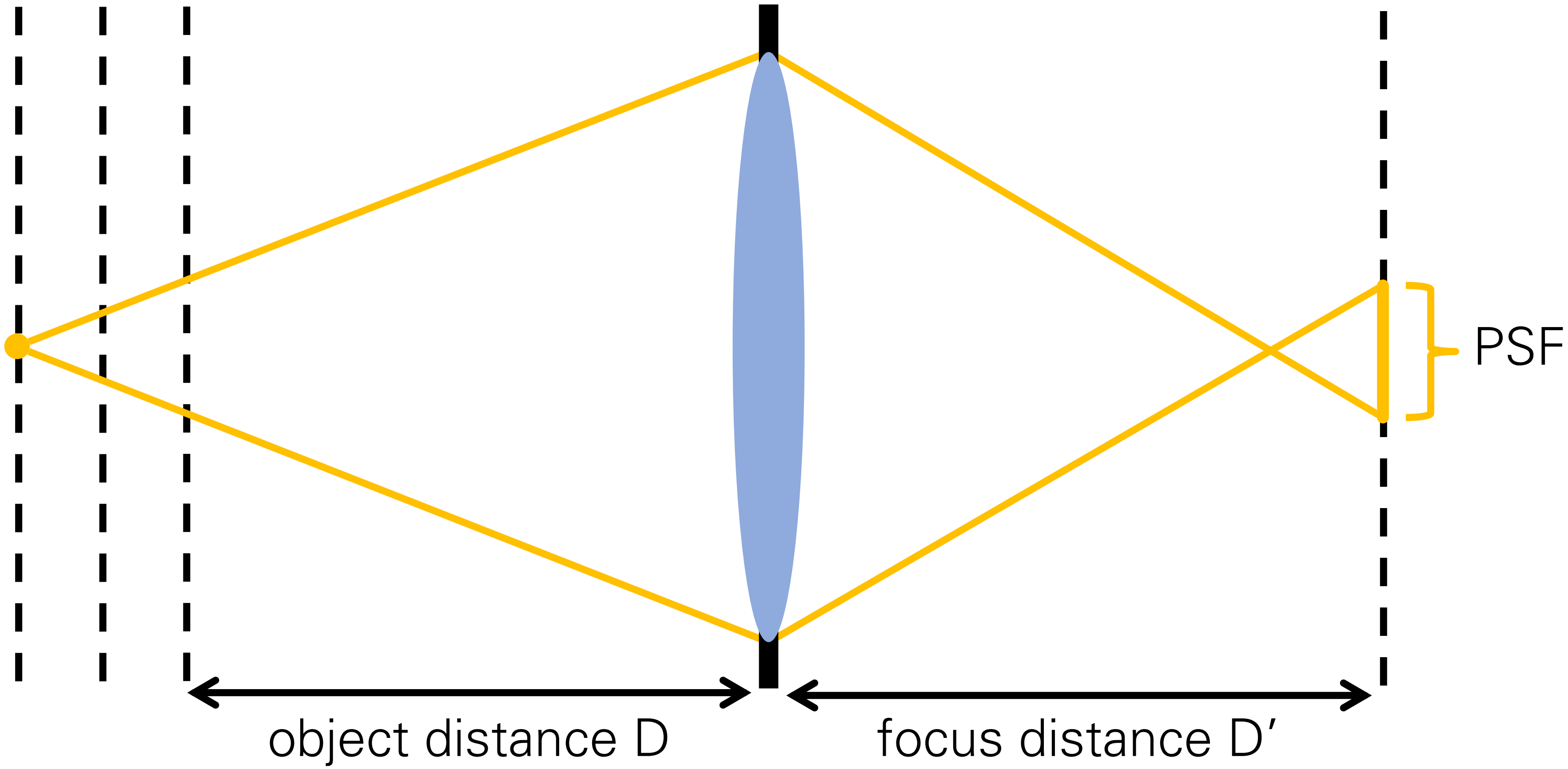


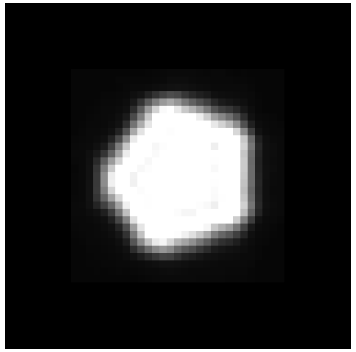
Depth determines scale of blur kernel



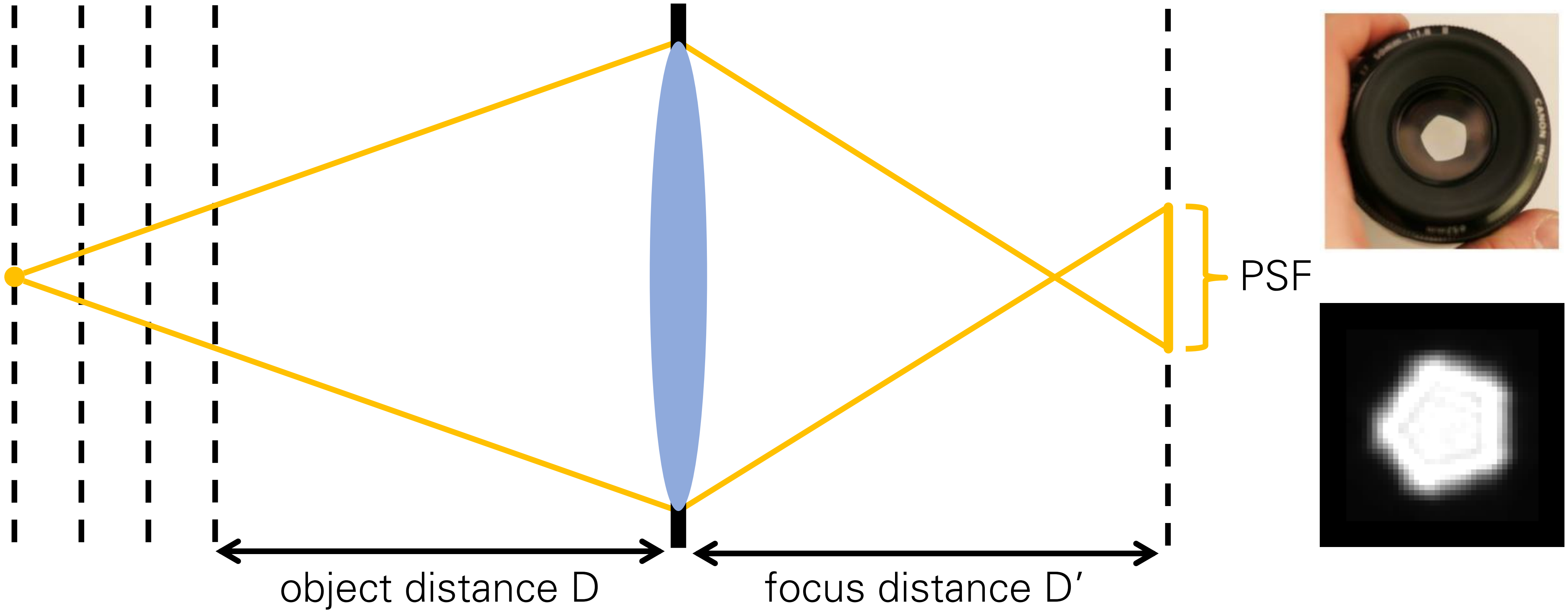


Depth determines scale of blur kernel

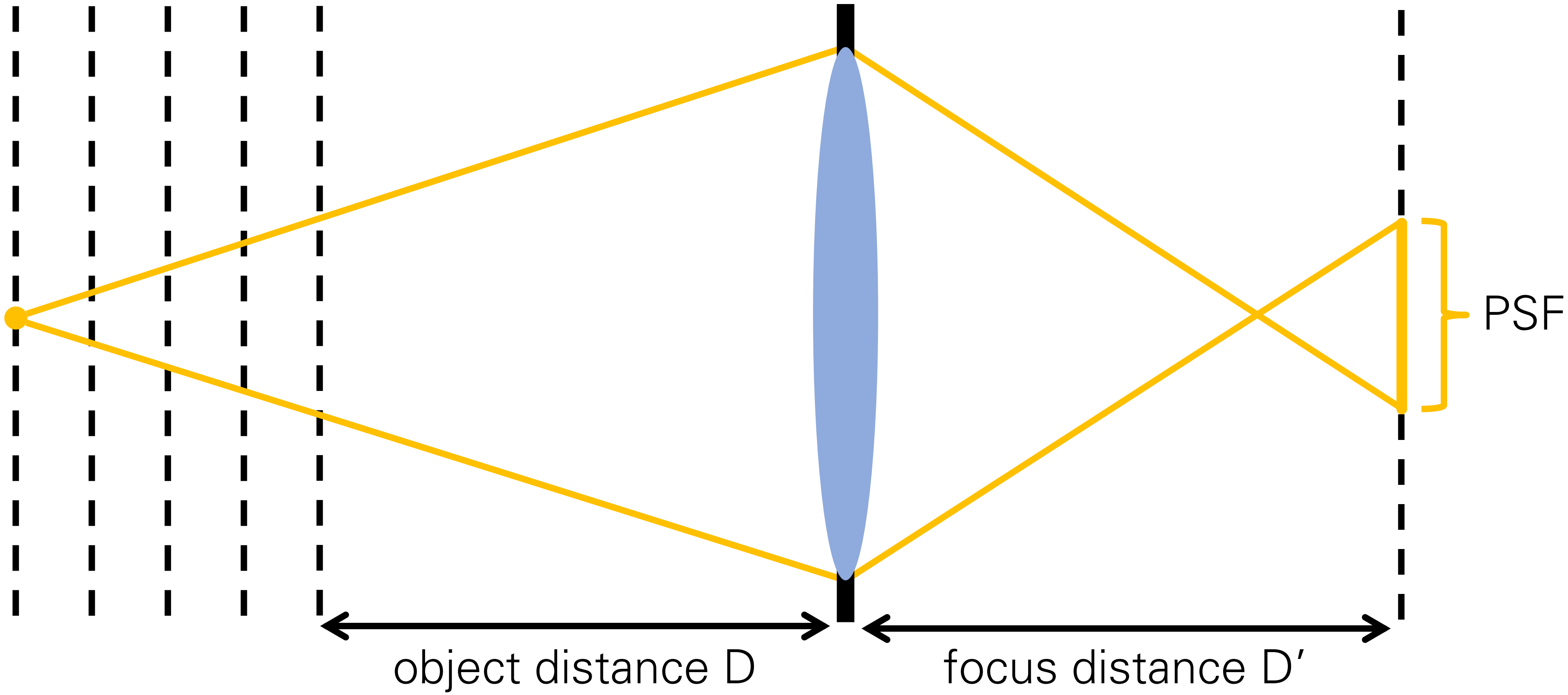


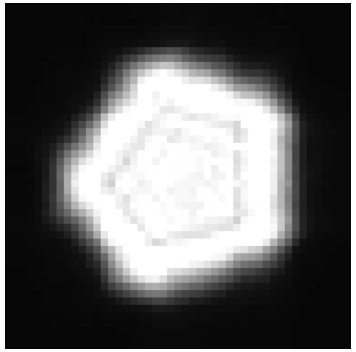


Depth determines scale of blur kernel

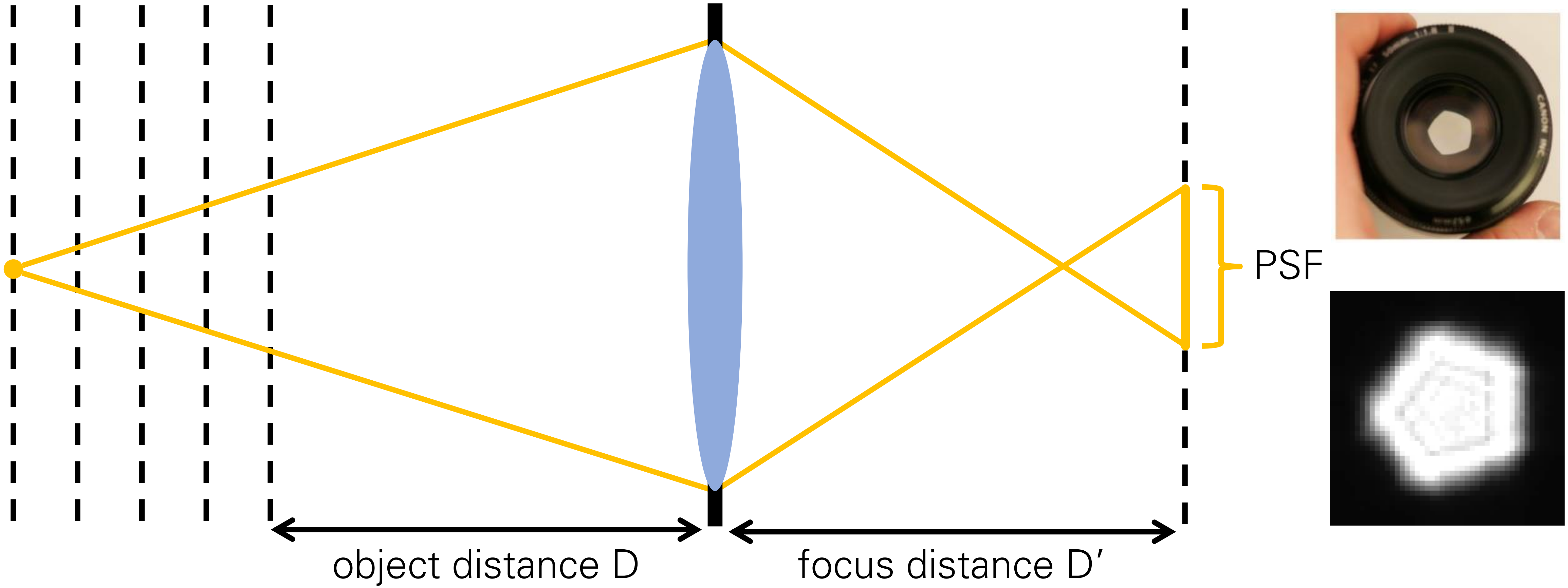


Depth determines scale of blur kernel





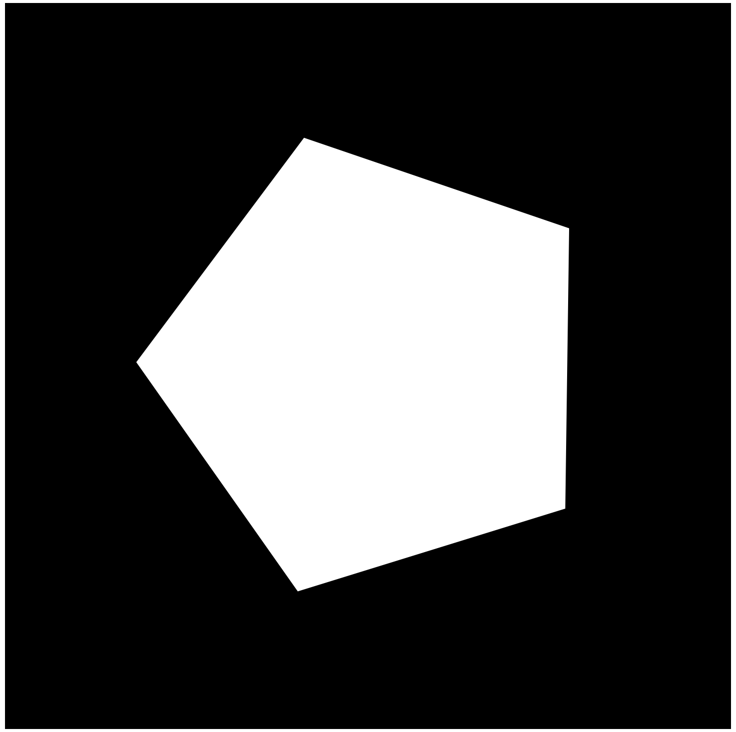
Aperture determines shape of blur kernel



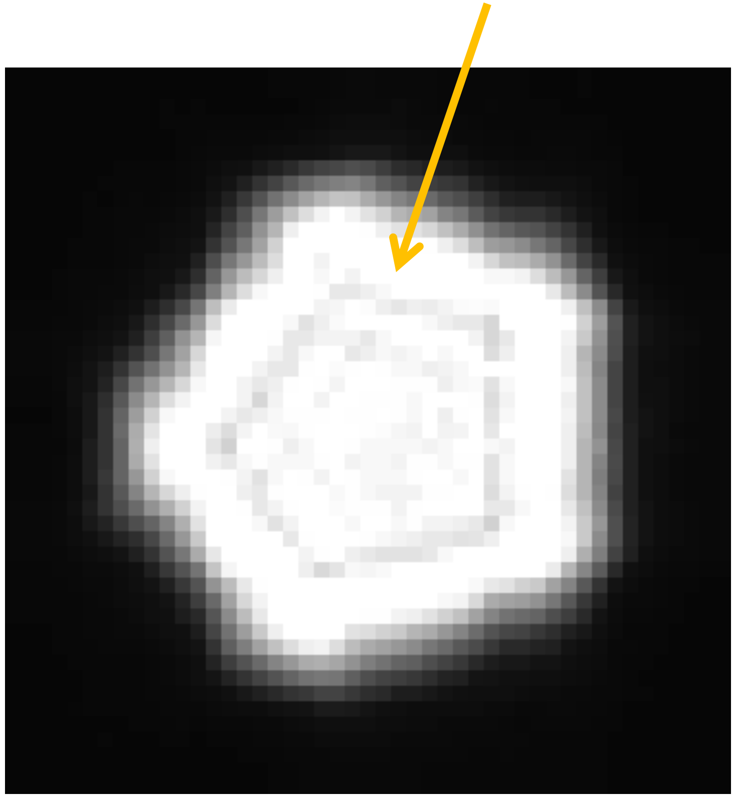
Aperture determines shape of blur kernel

What causes these lines?

photo of aperture



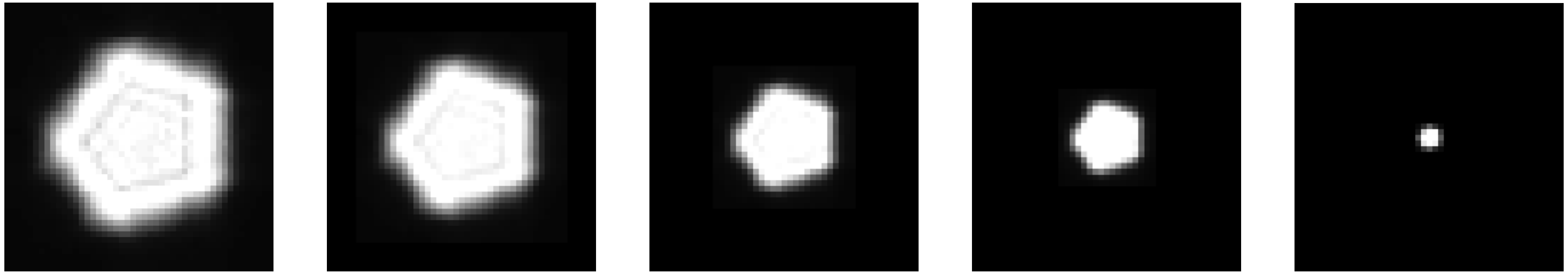
shape of aperture
(optical transfer function, OTF)



blur kernel
(point spread function, PSF)

How do the OTF and PSF relate to each other?

Removing depth defocus



measured PSFs at different depths

input defocused image

How would you create an all in-focus image given the above?

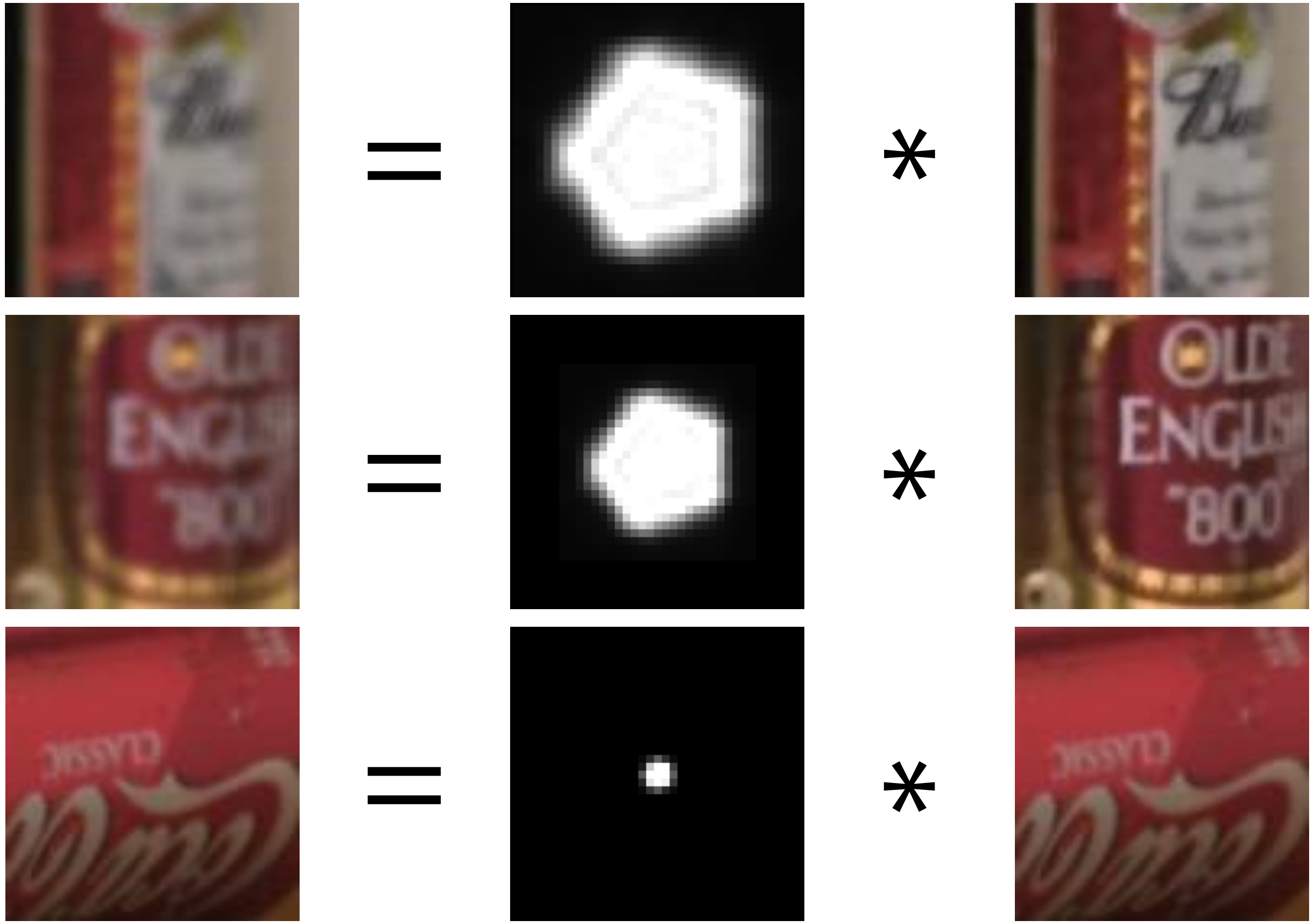
Removing depth defocus

Defocus is local convolution with a depth-dependent kernel

depth 3

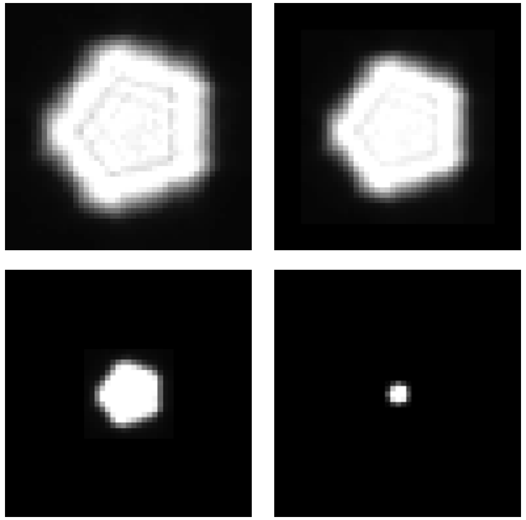
depth 2

depth 1



How would you create an all in-focus image given the above?

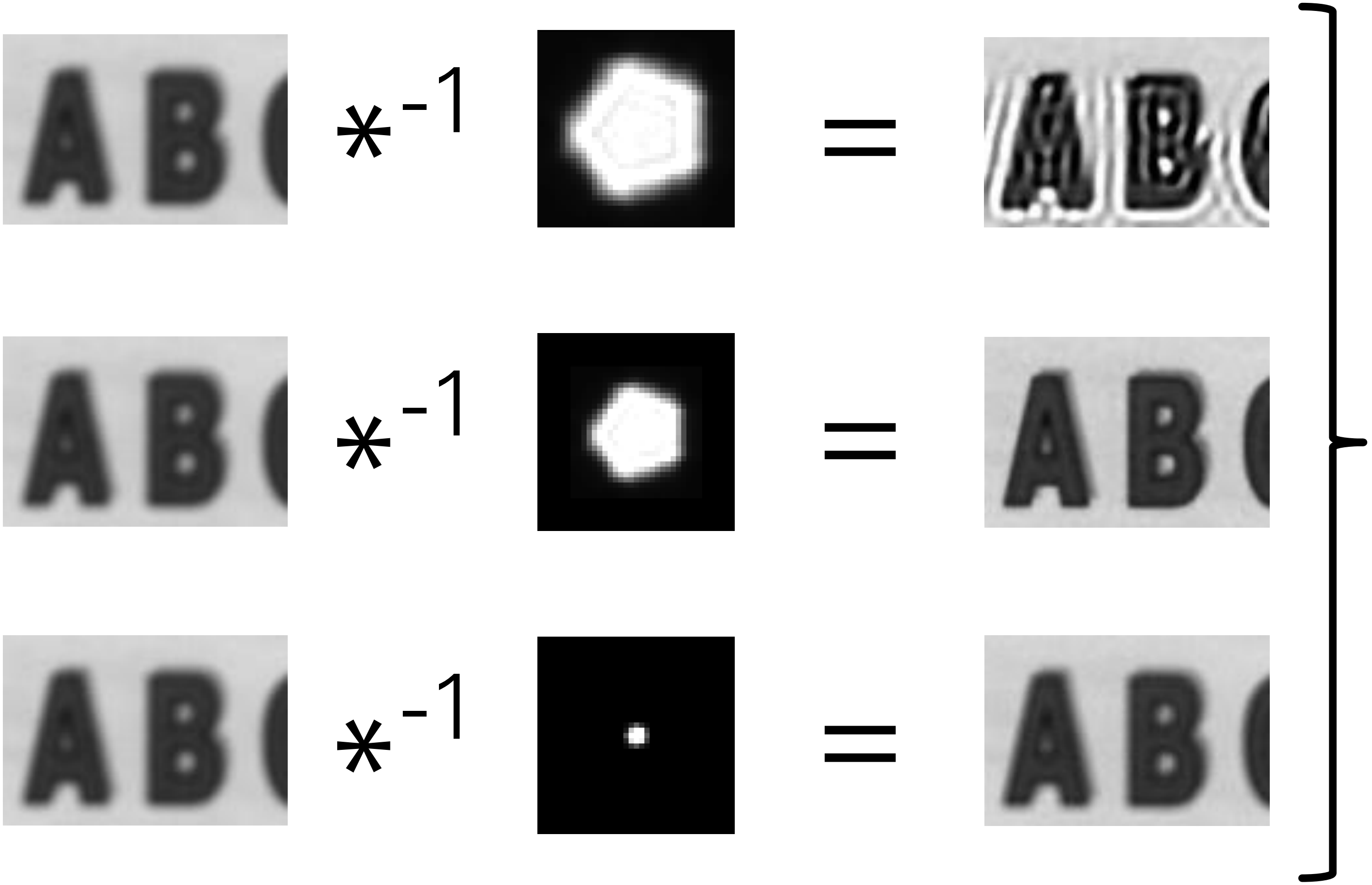
input defocused image



measured PSFs at
different depths

Removing depth defocus

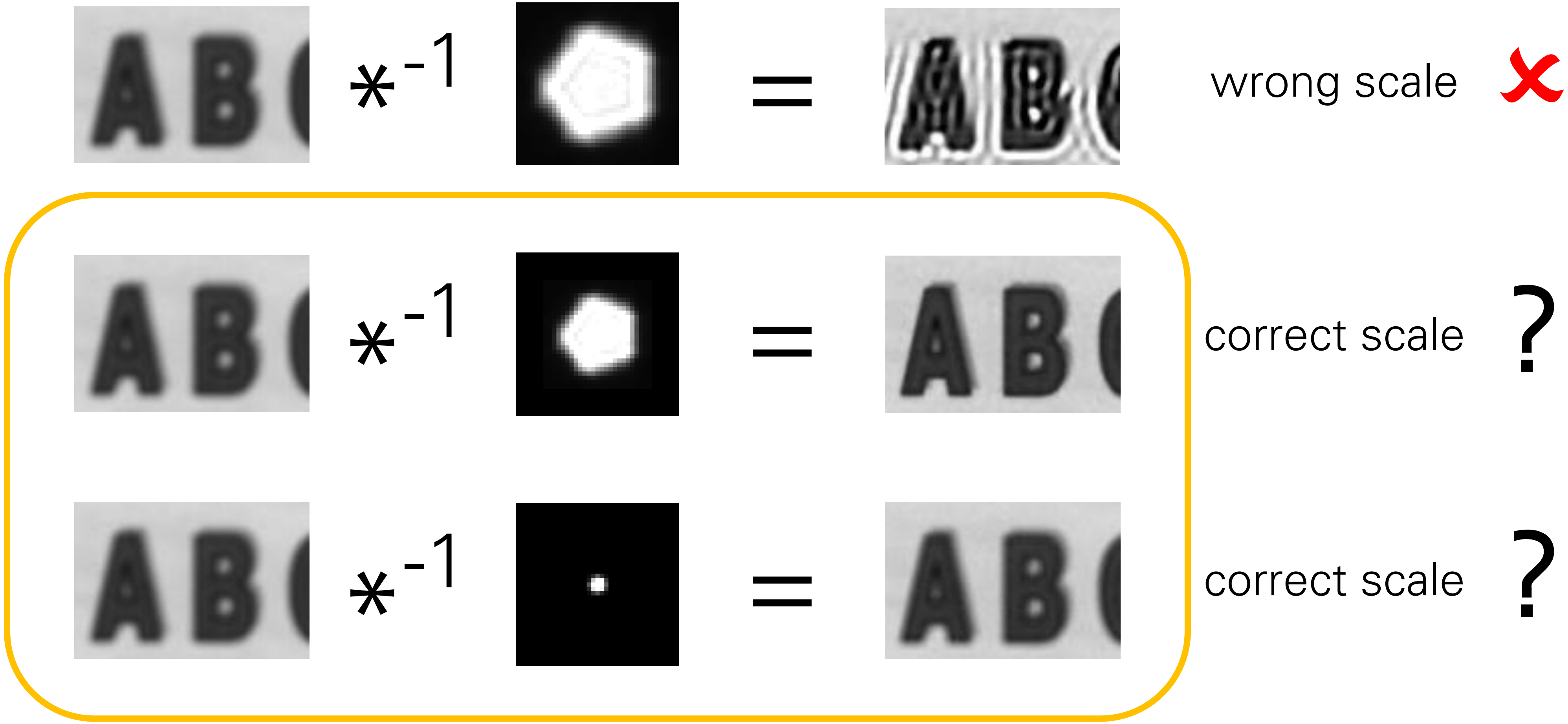
- Deconvolve each image patch with all kernels
- Select the right scale by evaluating the deconvolution results



How do we select the correct scale?

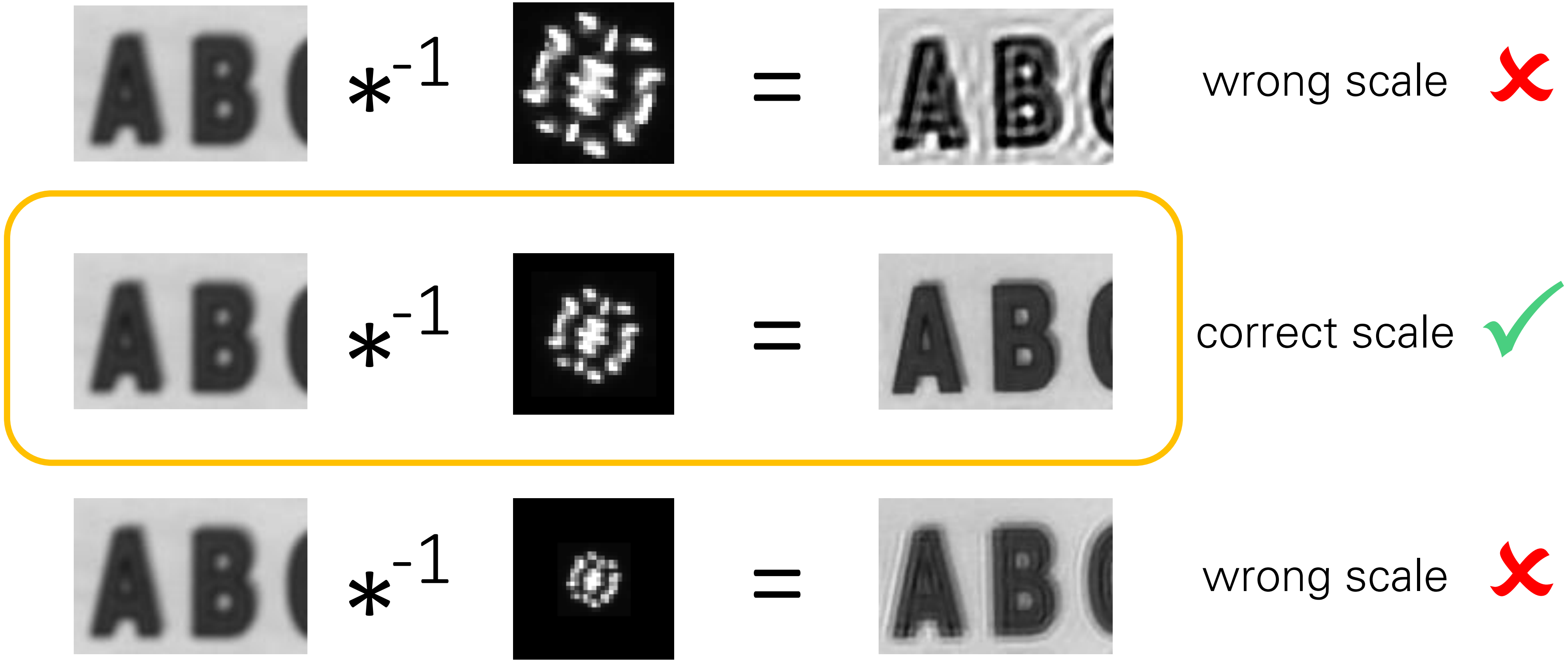
Removing depth defocus

Problem: With standard aperture, results at different scales look very similar.

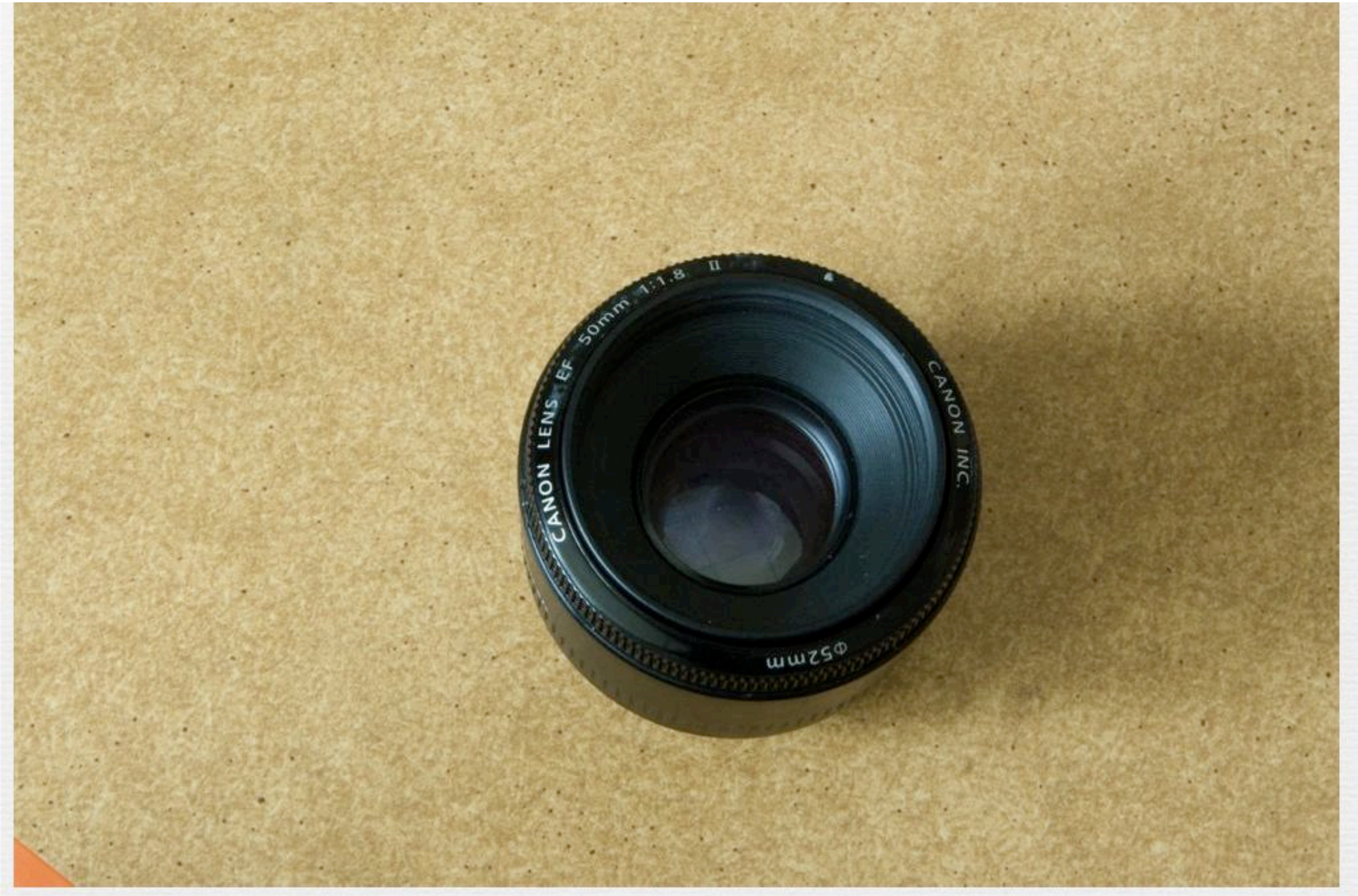


Coded aperture

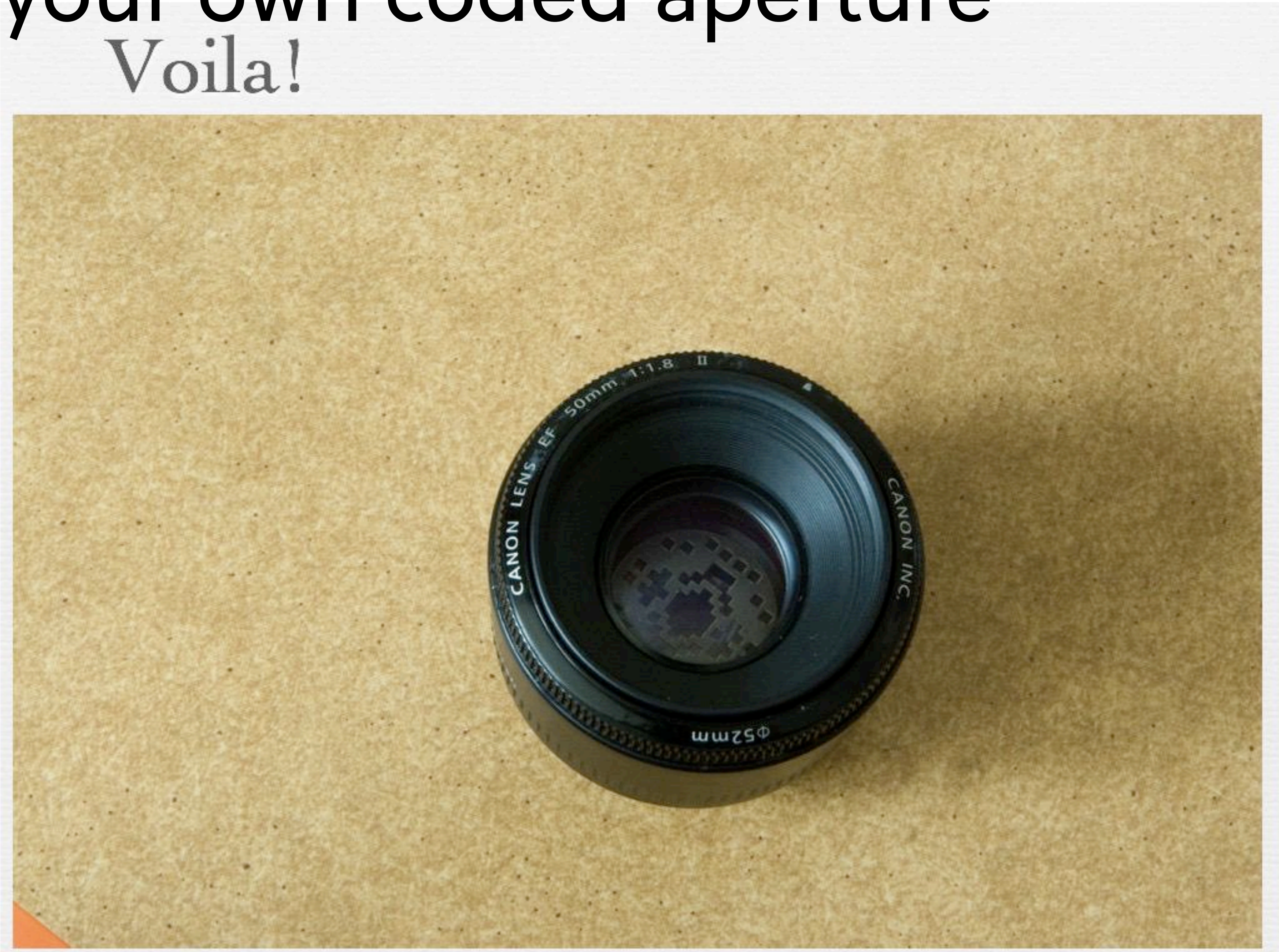
Solution: Change aperture so that it is easier to pick the correct scale



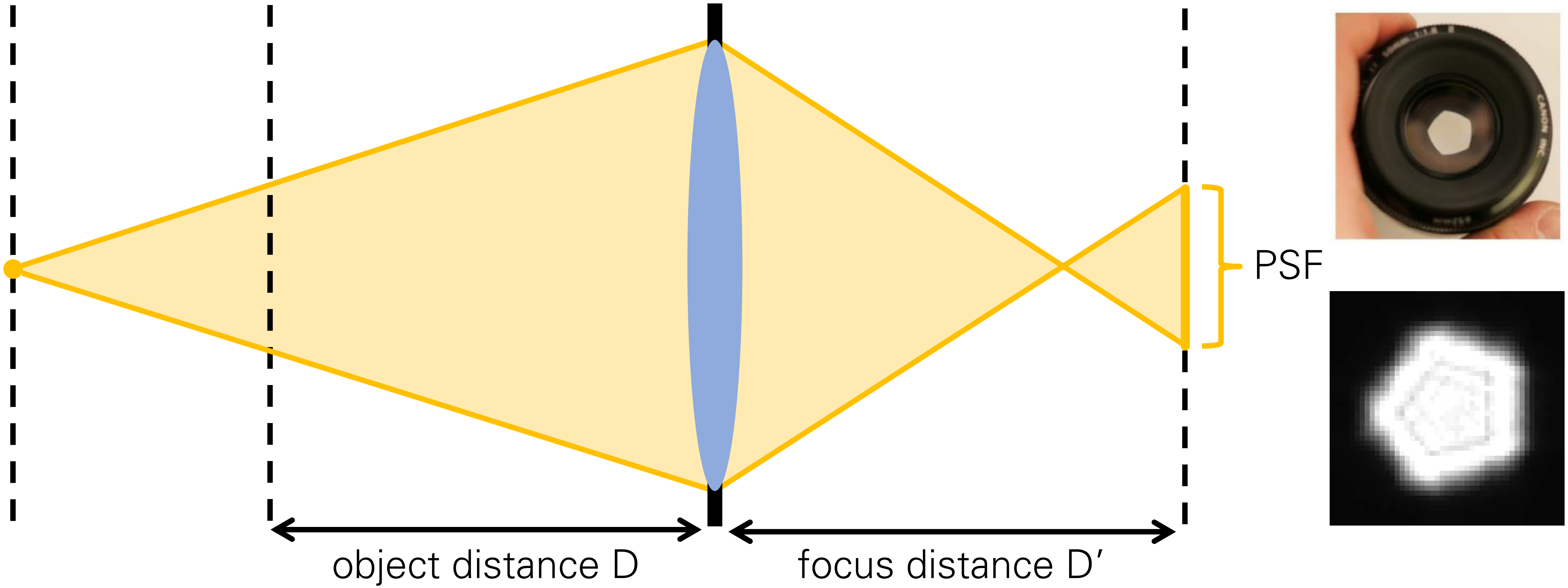
Build your own coded aperture



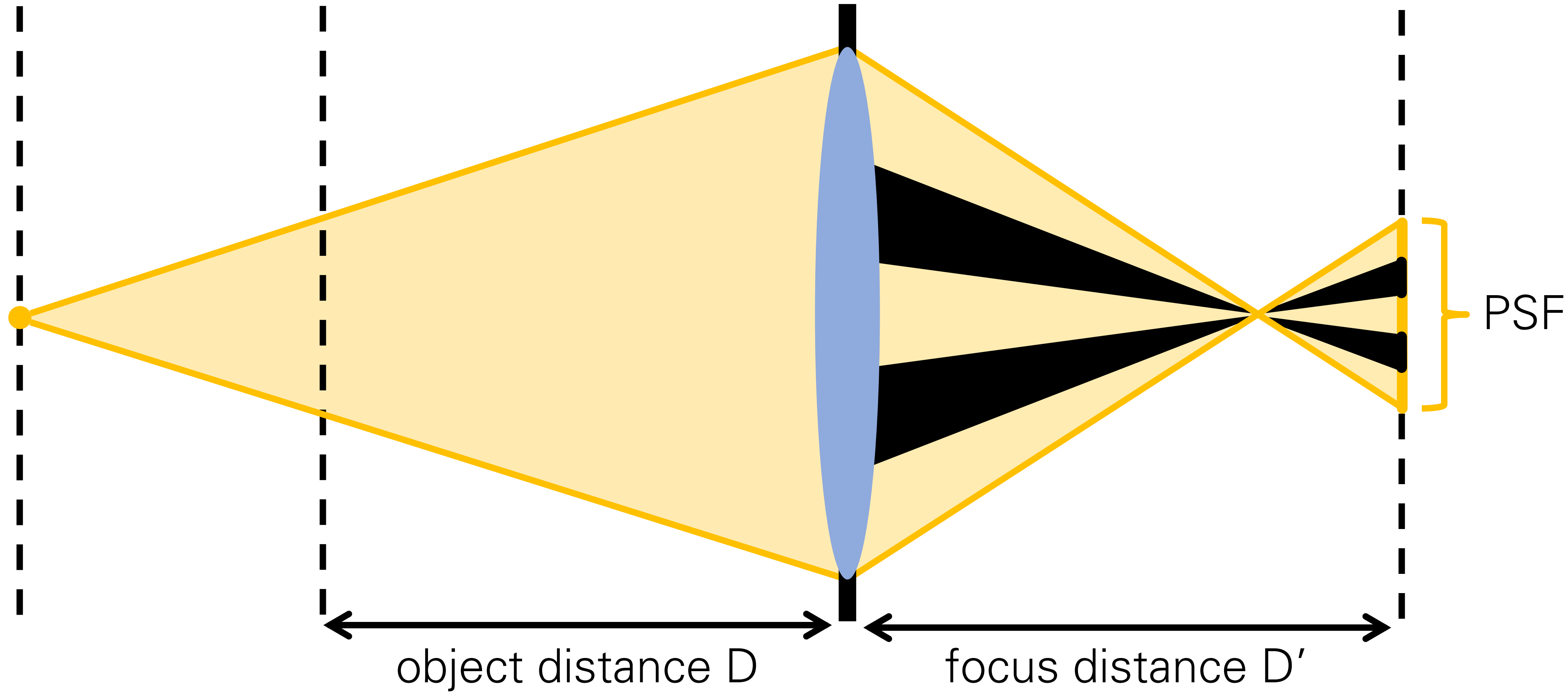
Build your own coded aperture

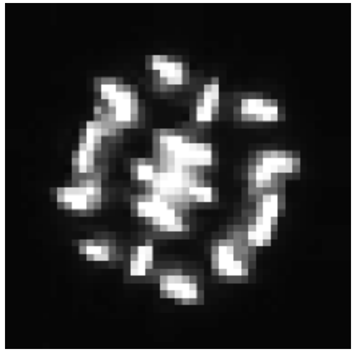


Coded aperture changes shape of kernel

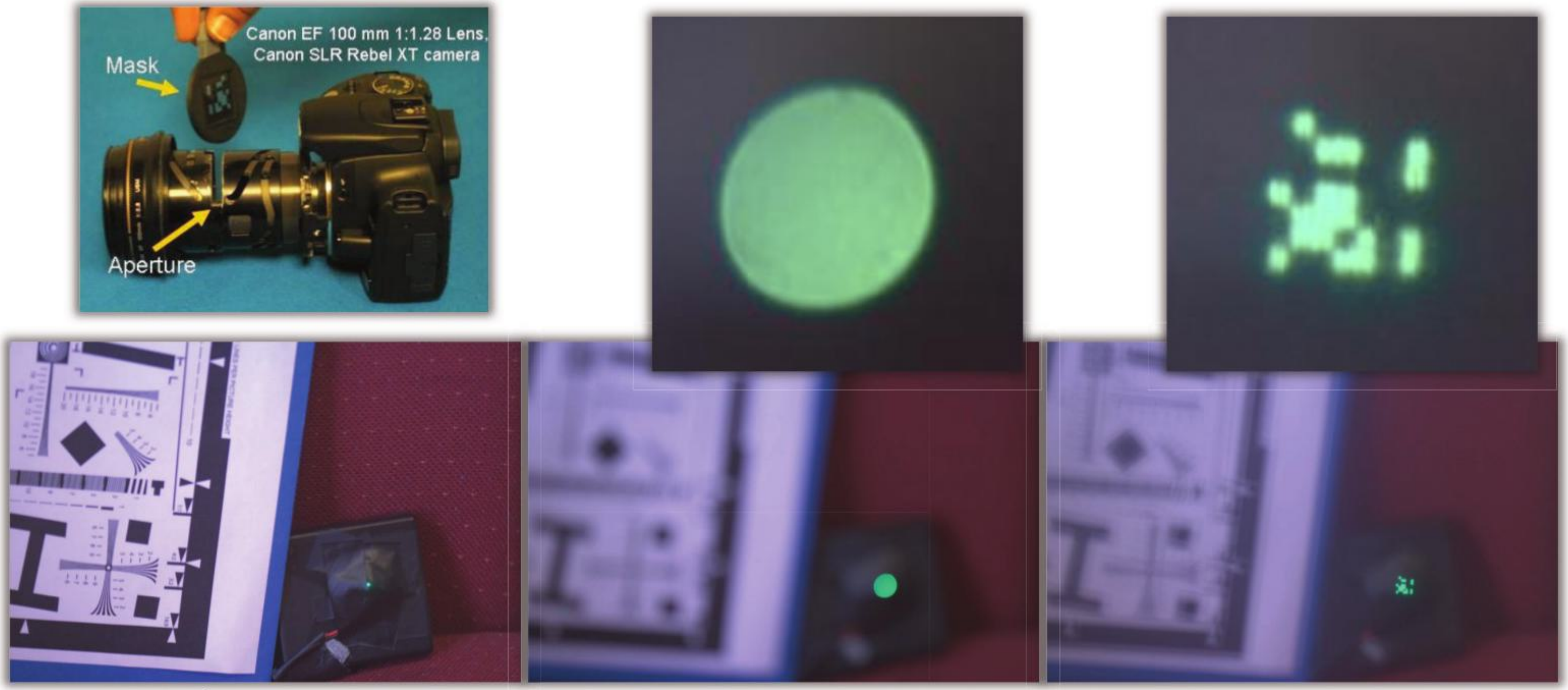


Coded aperture changes shape of kernel





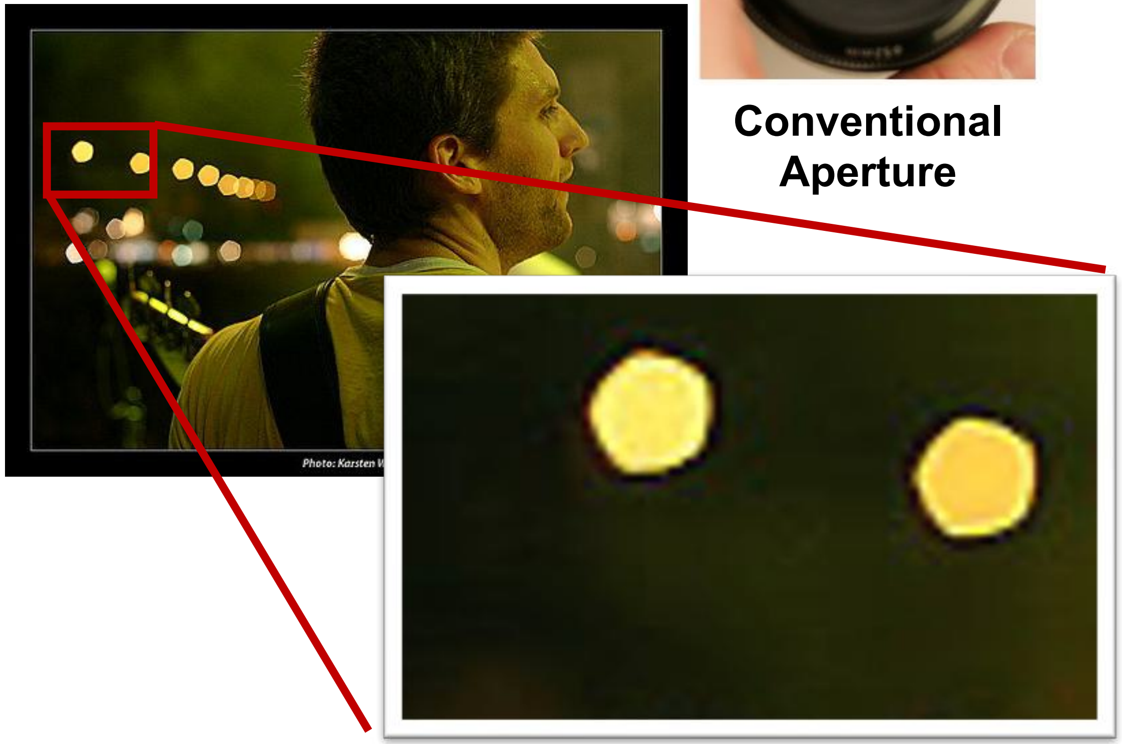
Coded aperture changes shape of PSF



in-focus photo | out-of-focus, circular aperture | out-of-focus, coded aperture

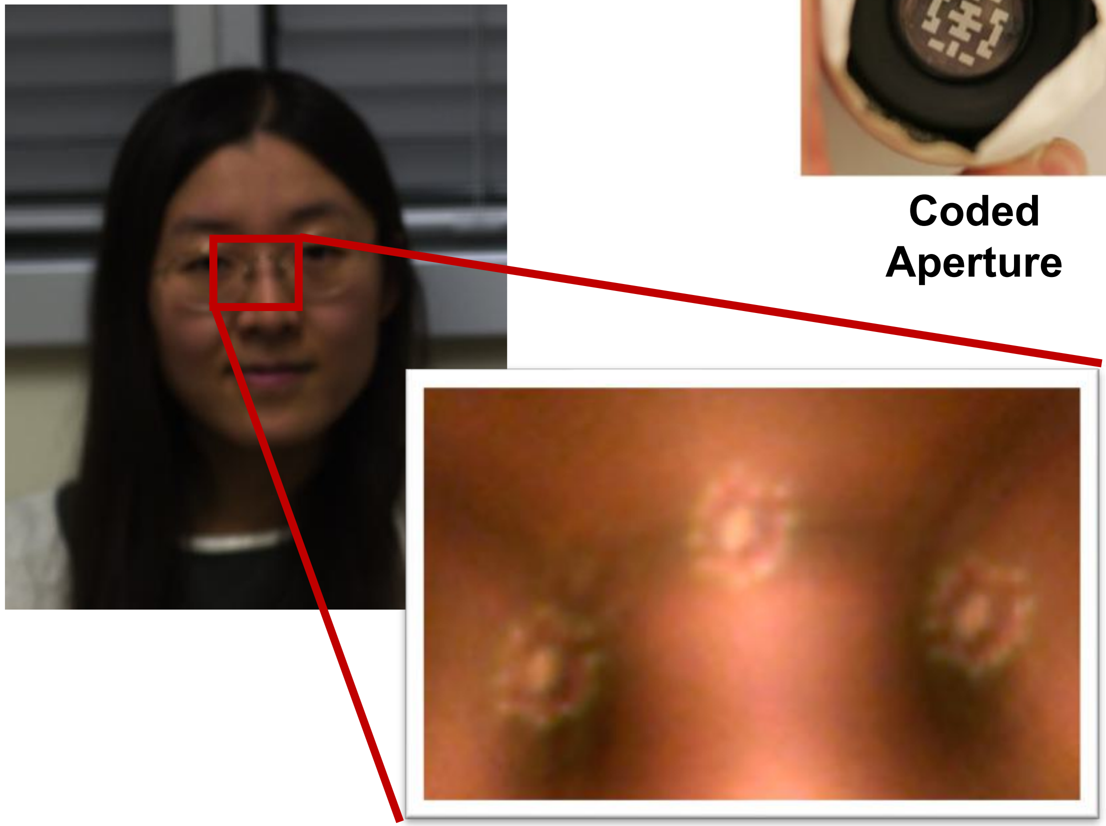
Image of a point light source

Conventional
Aperture



Captured Image

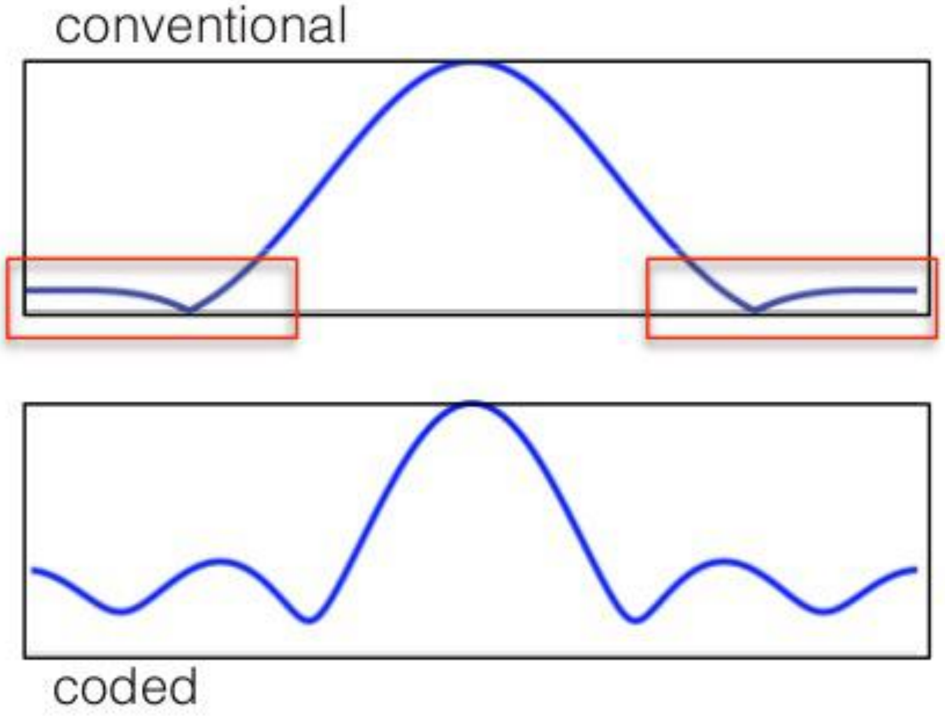
Coded
Aperture

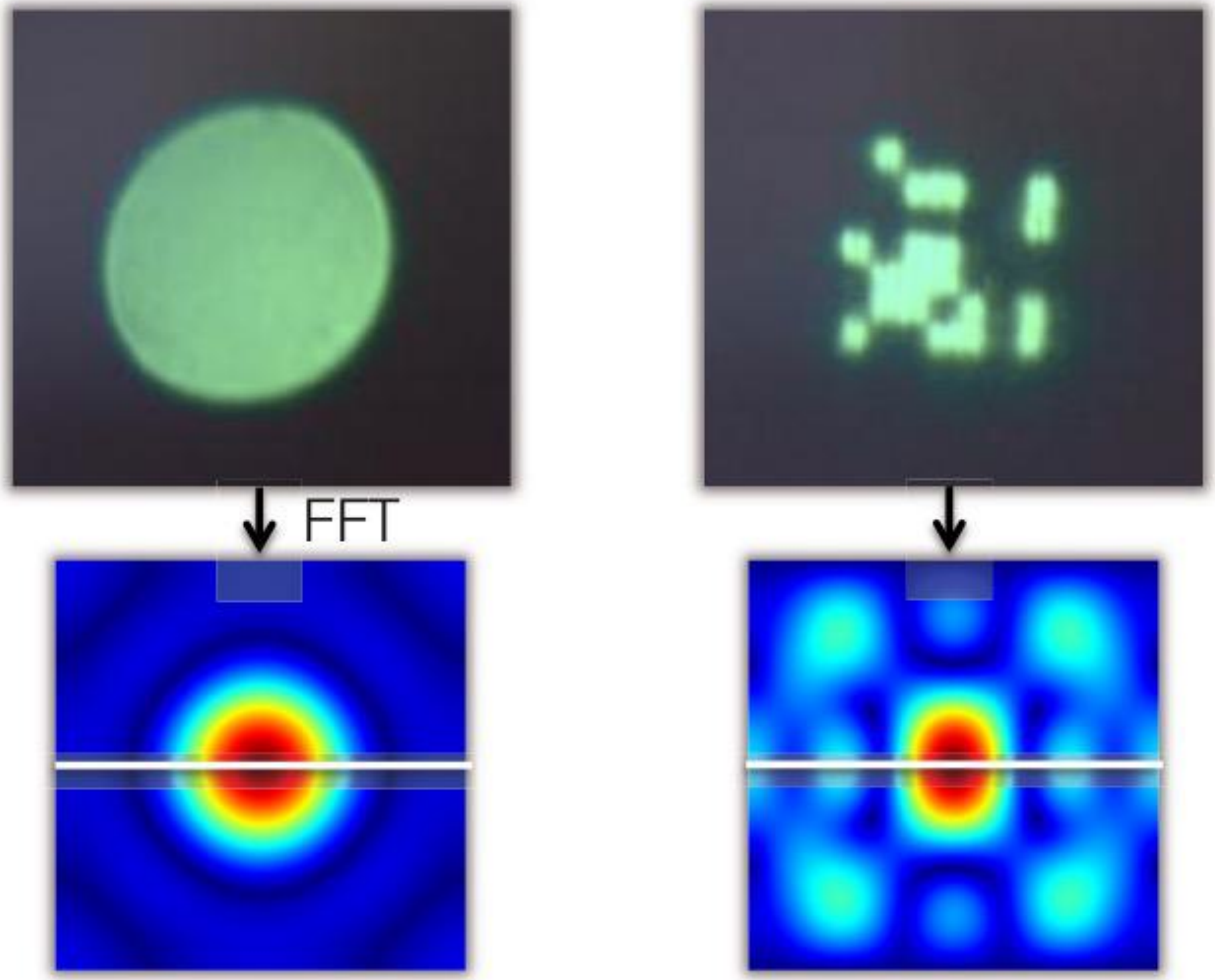


Captured Image

Coded aperture changes shape of PSF

New PSF preserves high frequencies
- More content available to help us determine correct depth

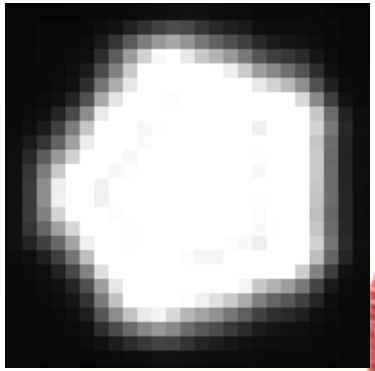




Input

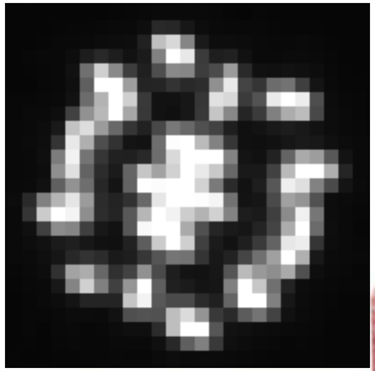
All-focused (deconvolved)

Comparison between standard and coded aperture



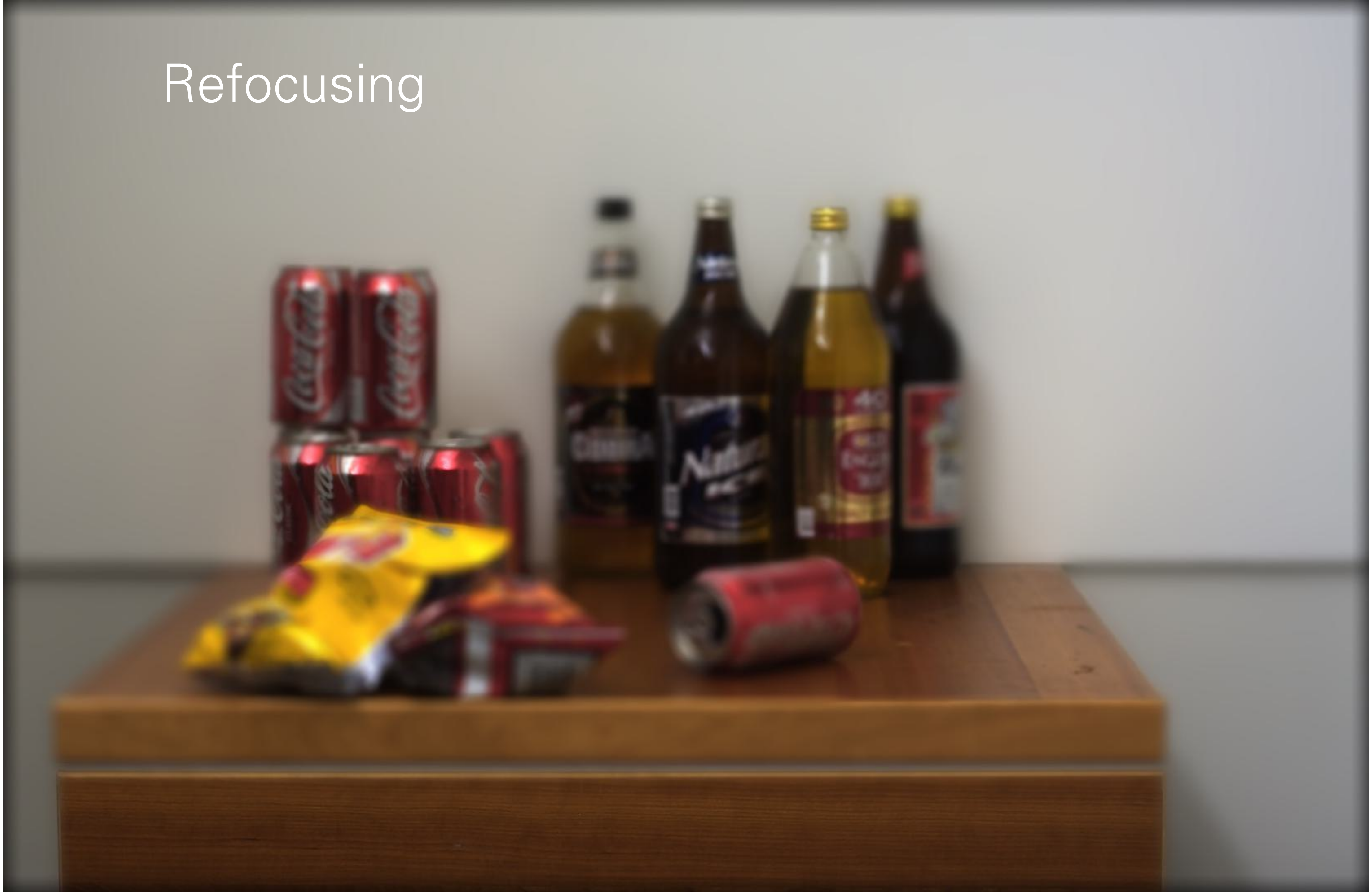
Ringing due to wrong scale estimation

Comparison between standard and coded aperture

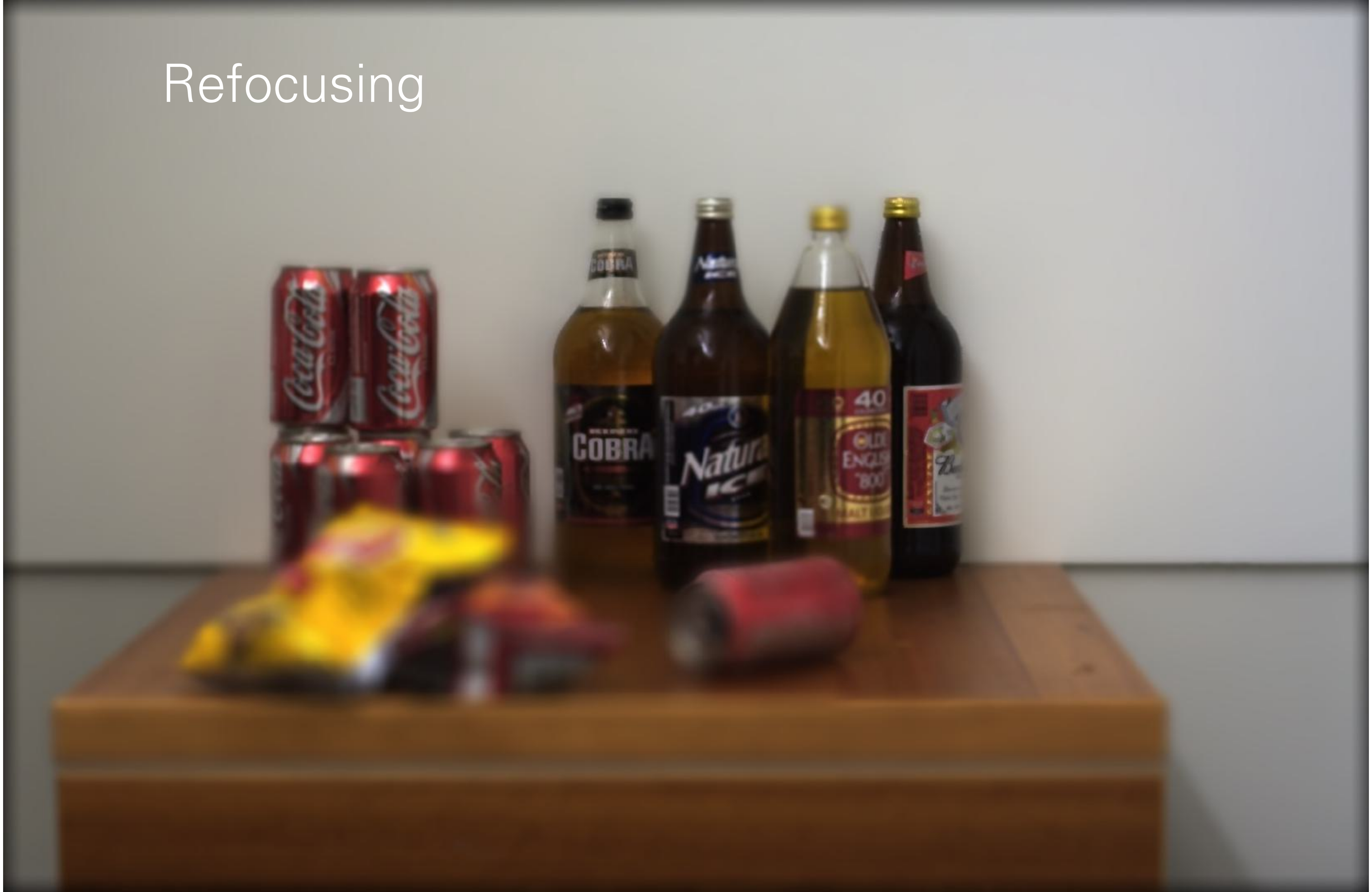


Refocusing

Refocusing



Refocusing



Depth estimation


Input


All-focused
(deconvolved)

Refocusing

Refocusing

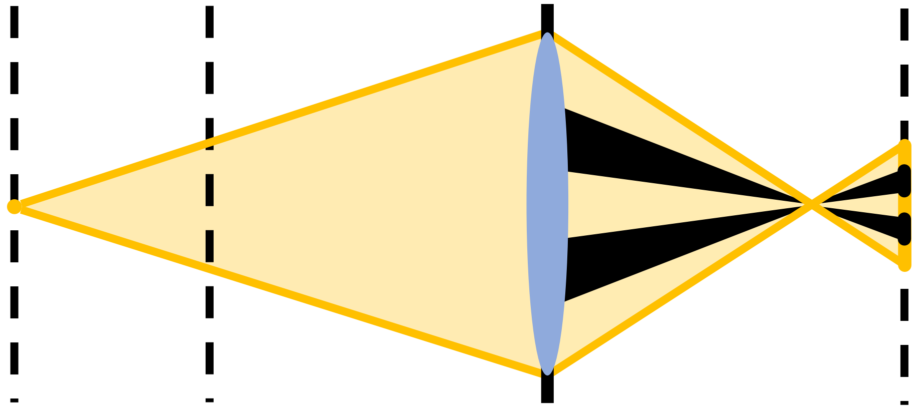
Refocusing

Depth estimation

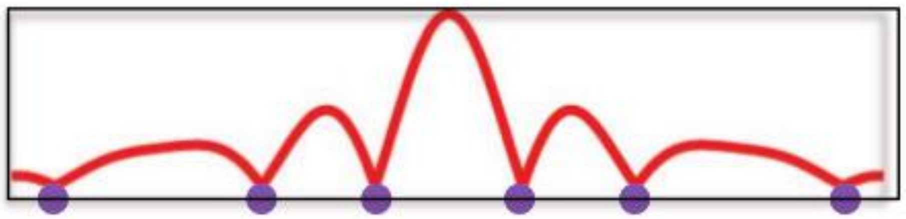
Any problems with using a coded aperture?

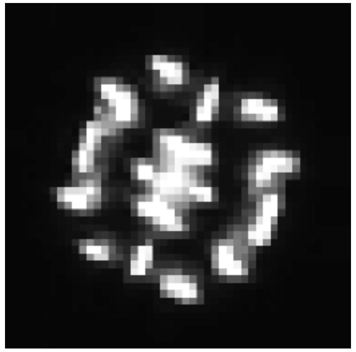
Any problems with using a coded aperture?

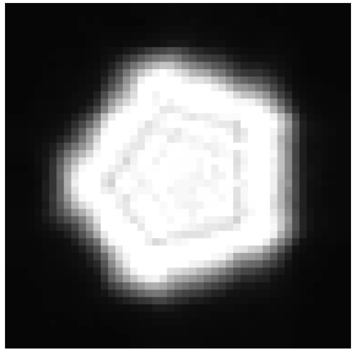
- We lose a lot of light due to blocking.



- The deconvolution becomes harder due to more diffraction/zeros in frequency domain.



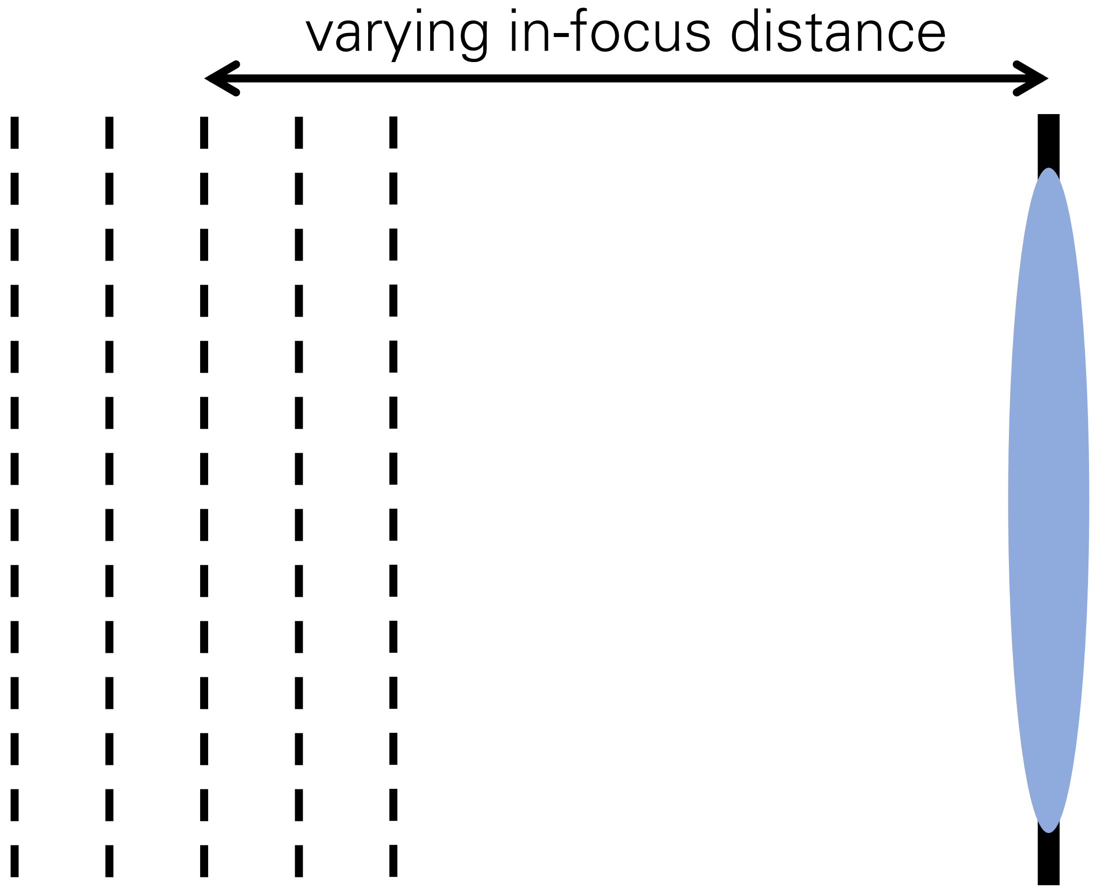




- We still need to select correct scale.

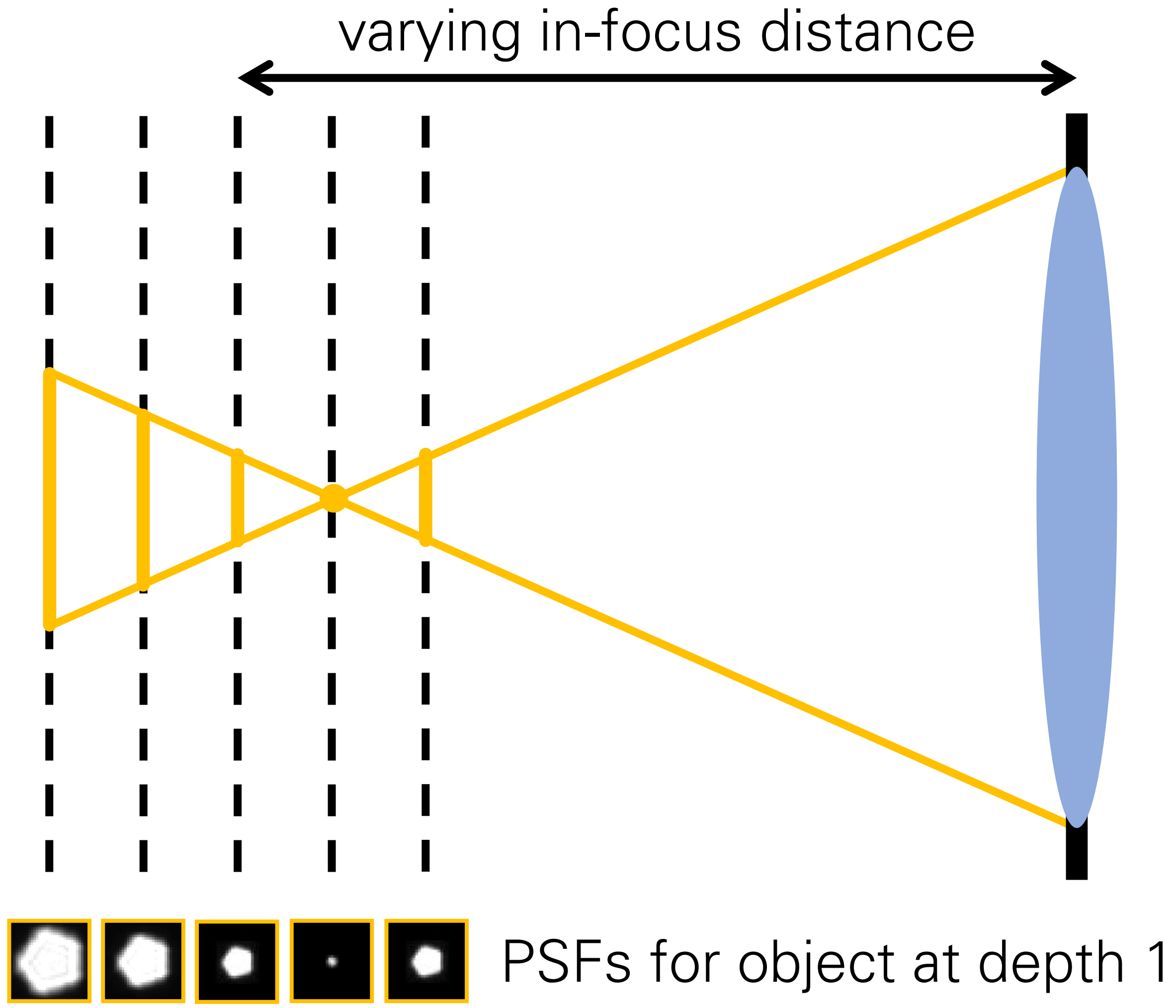
Dealing with depth blur: focal sweep

The difficulty of dealing with depth defocus



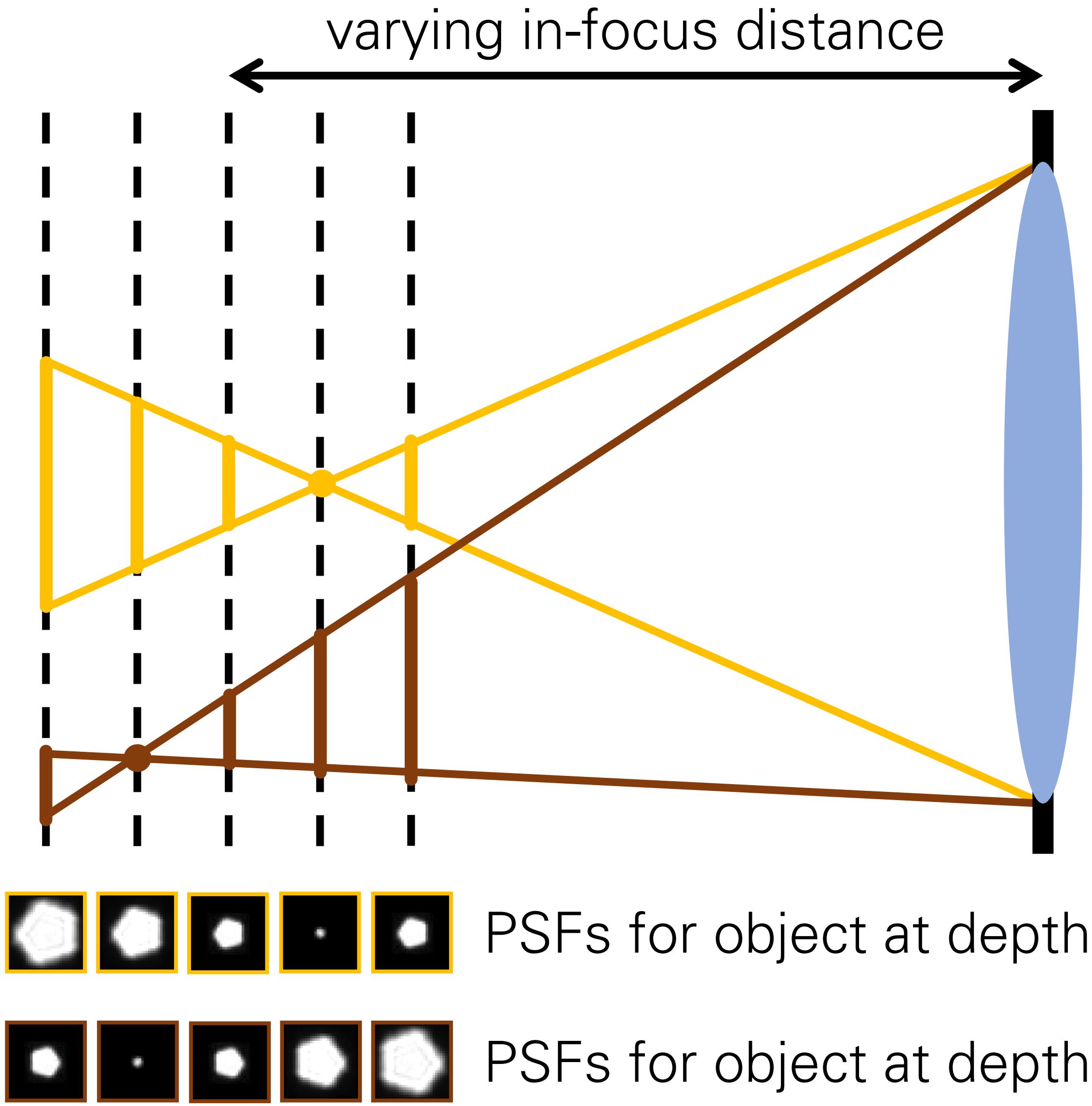
At every focus setting, objects at different depths are blurred by different PSF

The difficulty of dealing with depth defocus



At every focus setting, objects at different depths are blurred by different PSF

The difficulty of dealing with depth defocus

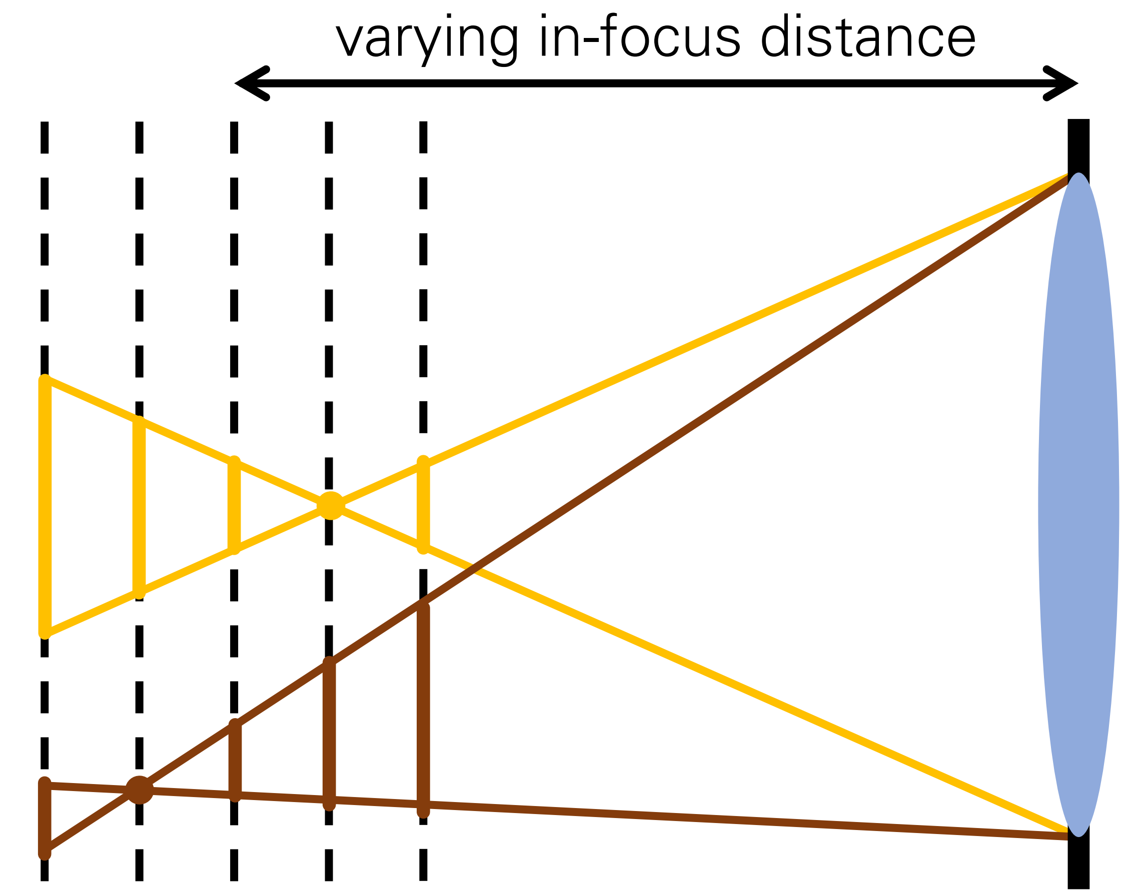


PSFs for object at depth 1

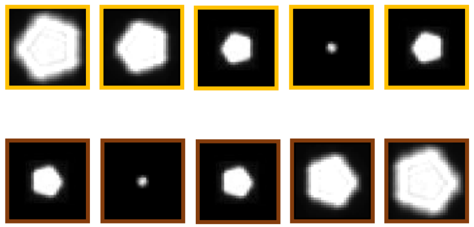
PSFs for object at depth 2

At every focus setting, objects at different depths are blurred by different PSF

The difficulty of dealing with depth defocus



At every focus setting, objects at different depths are blurred by different PSF

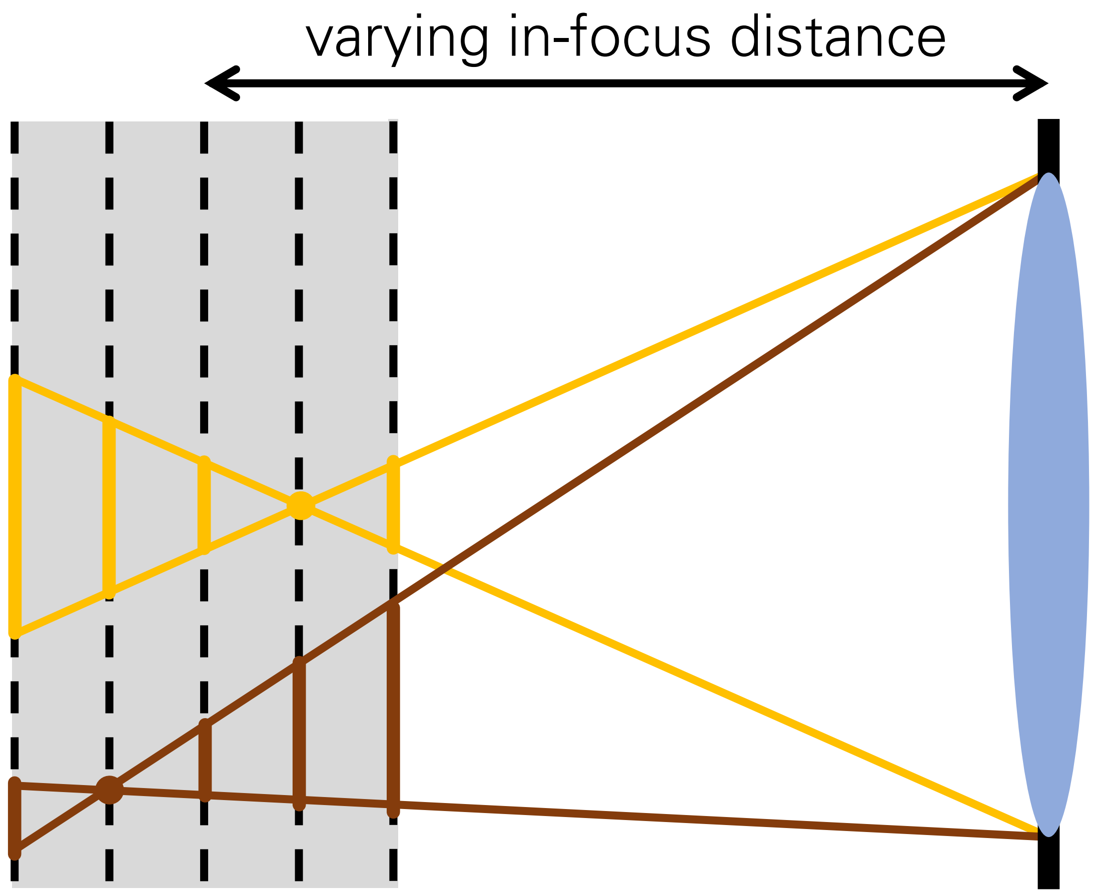


PSFs for object at depth 1

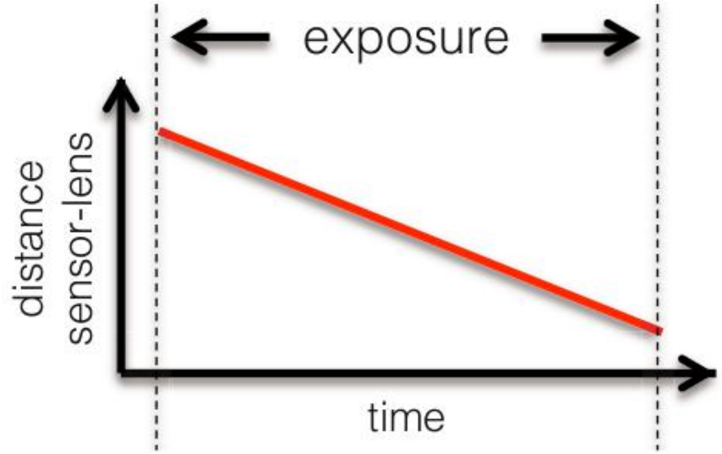
PSFs for object at depth 2

As we sweep through focus settings, each point every object is blurred by all possible PSFs

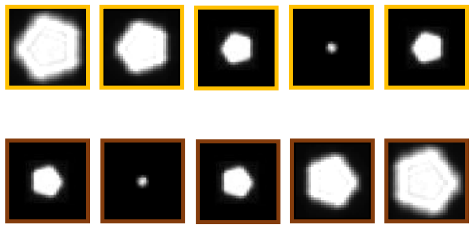
Focal sweep



linear motion:



Go through all focus settings during a single exposure

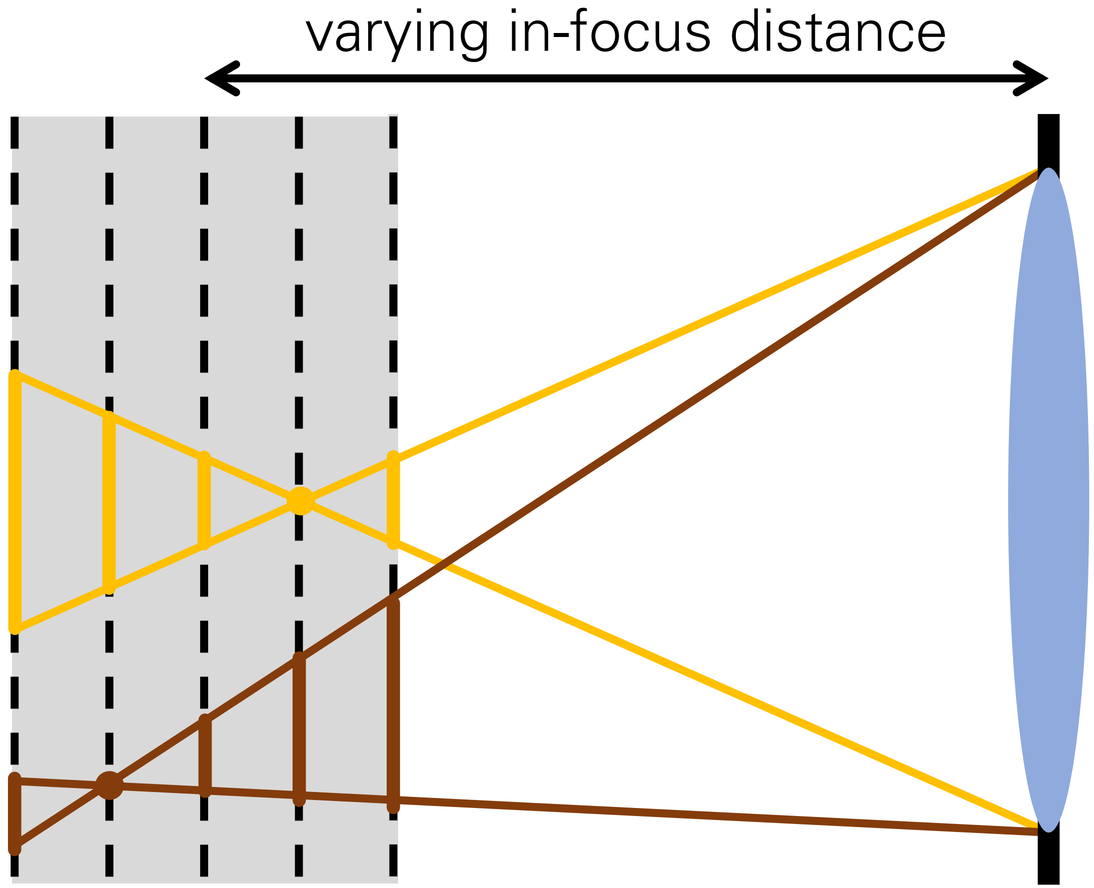


PSFs for object at depth 1

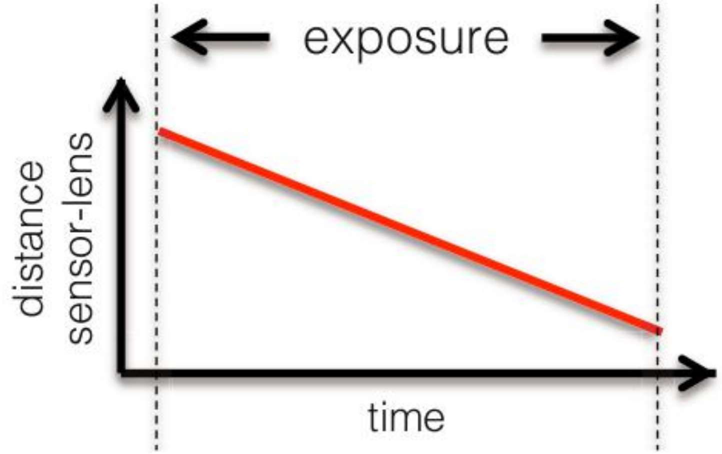
PSFs for object at depth 2

What is the effective PSF in this case?

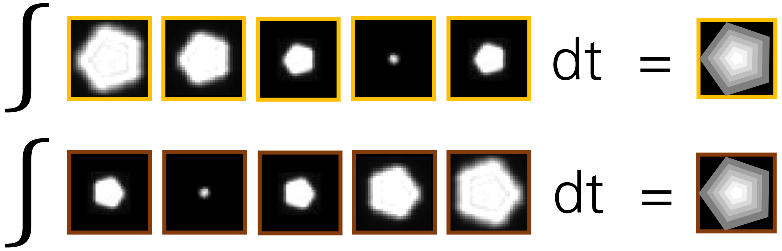
Focal sweep



linear motion:



Go through all focus settings during a single exposure



effective PSF for object at depth 1

effective PSF for object at depth 2

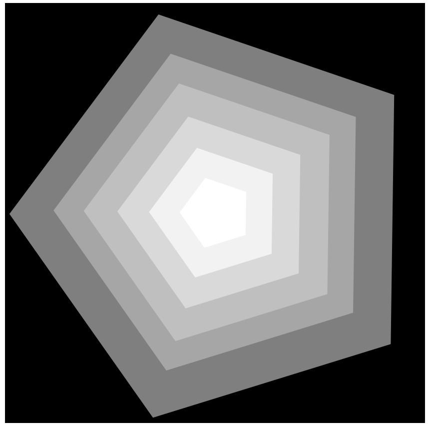
Anything special about these effective PSFs?

Focal sweep

The effective PSF is:

1. Depth-invariant – all points are blurred the same way regardless of depth.
2. Never sharp – all points will be blurry regardless of depth.

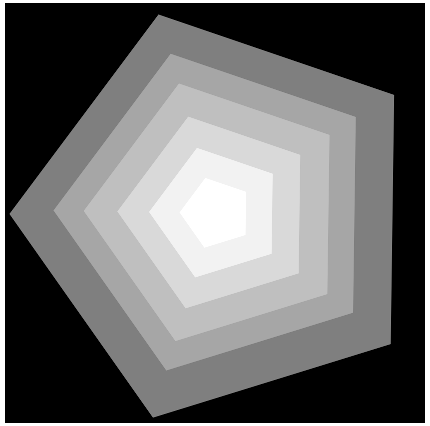
What are the implications of this?



Focal sweep

The effective PSF is:

1. Depth-invariant – all points are blurred the same way regardless of depth.
2. Never sharp – all points will be blurry regardless of depth.



What are the implications of this?

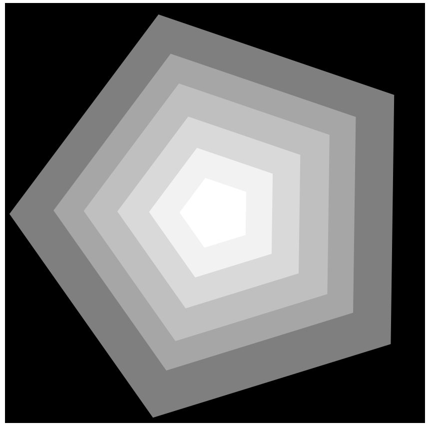
1. The image we capture will not be sharp anywhere; but
2. We can use simple (global) deconvolution to sharpen parts we want

1. Can we estimate depth from this?
2. Can we do refocusing from this?

Focal sweep

The effective PSF is:

1. Depth-invariant – all points are blurred the same way regardless of depth.
2. Never sharp – all points will be blurry regardless of depth.



What are the implications of this?

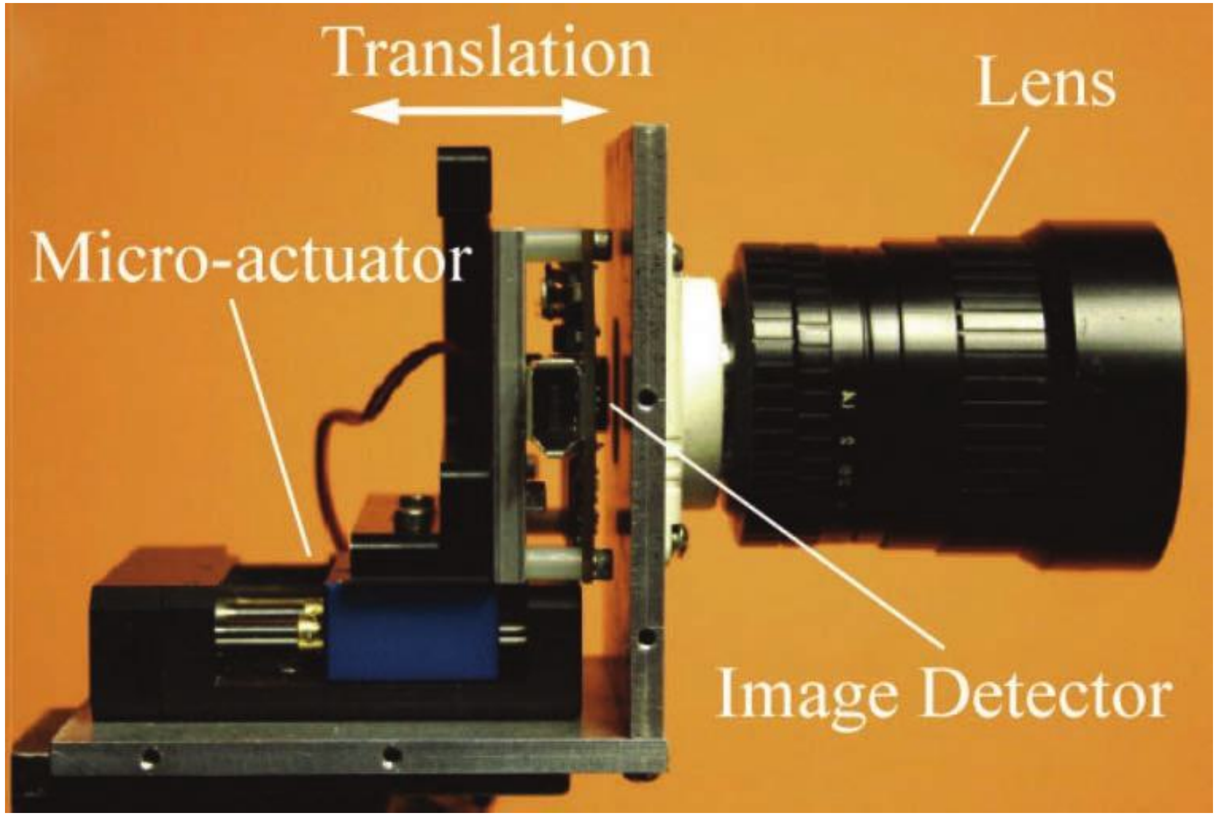
1. The image we capture will not be sharp anywhere; but
2. We can use simple (global) deconvolution to sharpen parts we want

1. Can we estimate depth from this?
2. Can we do refocusing from this?

Depth-invariance of the PSF means that we have lost all depth information

How can you implement focal sweep?

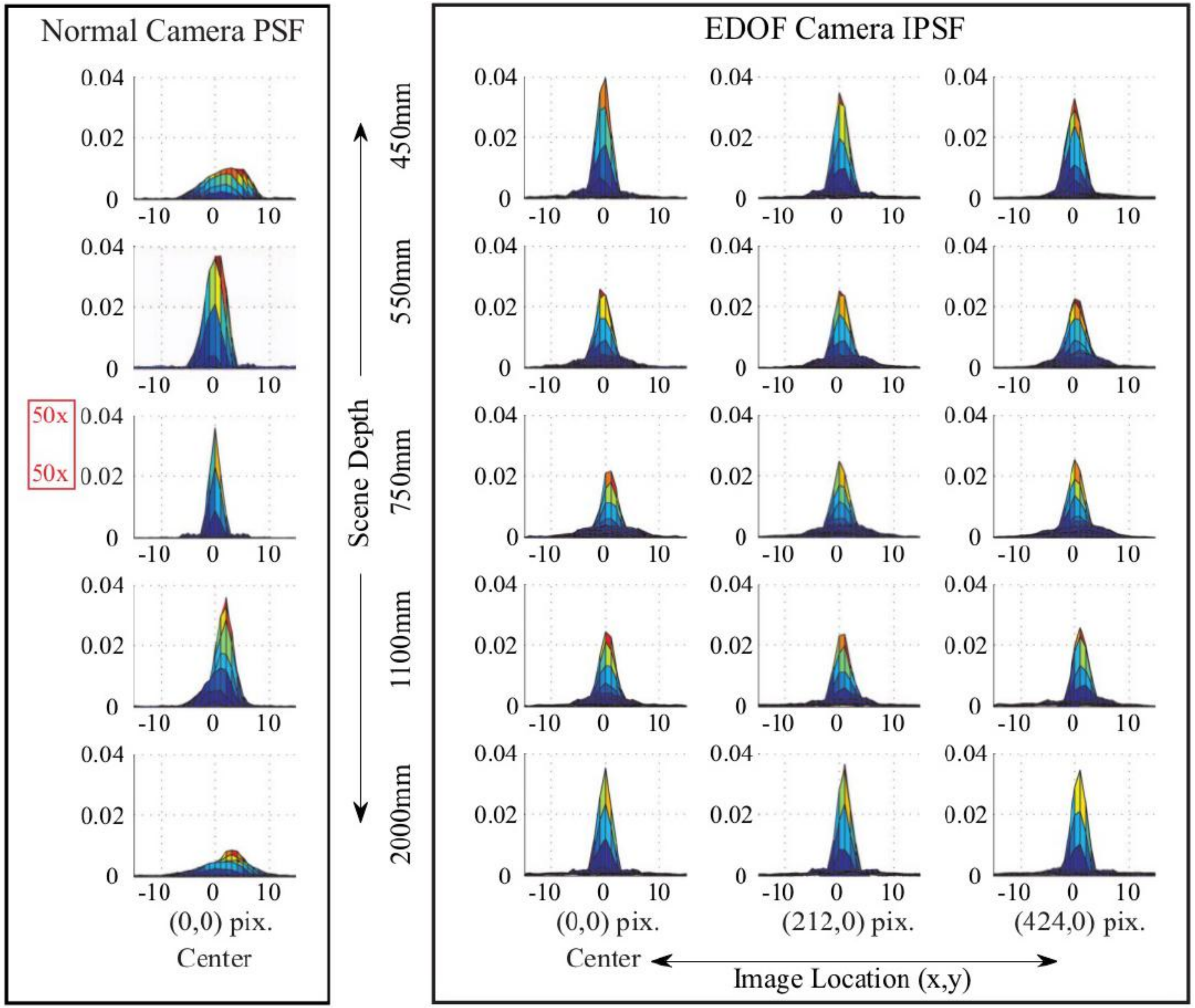
How can you implement focal sweep?



Use translation stage to move sensor relative to fixed lens during exposure

Rotate focusing ring to move lens relative to fixed sensor during exposure

Comparison of different PSFs



Depth of field comparisons

captured focal sweep
always blurry!

EDOF image

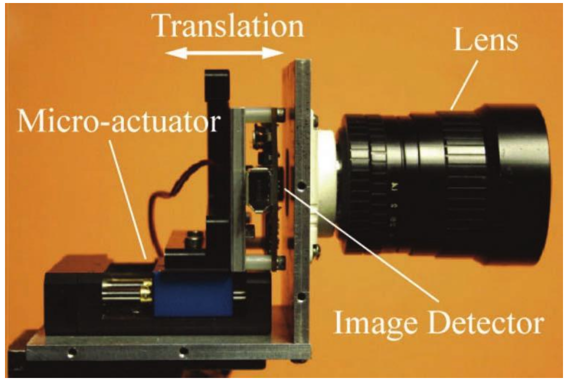
conventional photo
(small DOF)

conventional photo
(large DOF, noisy)

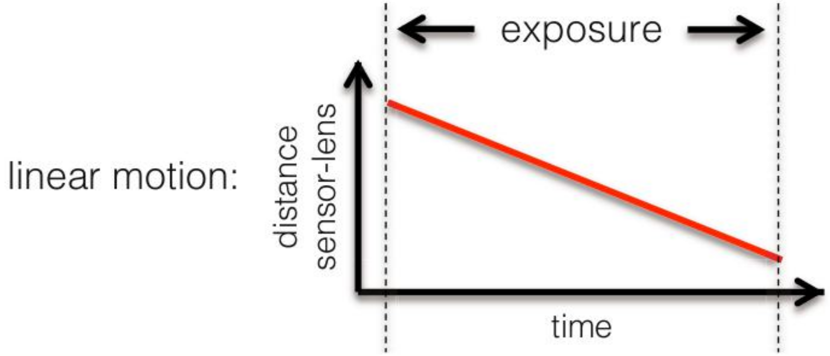
Any problems with using focal sweep?

Any problems with using focal sweep?

- We have moving parts (vibrations, motion blur).



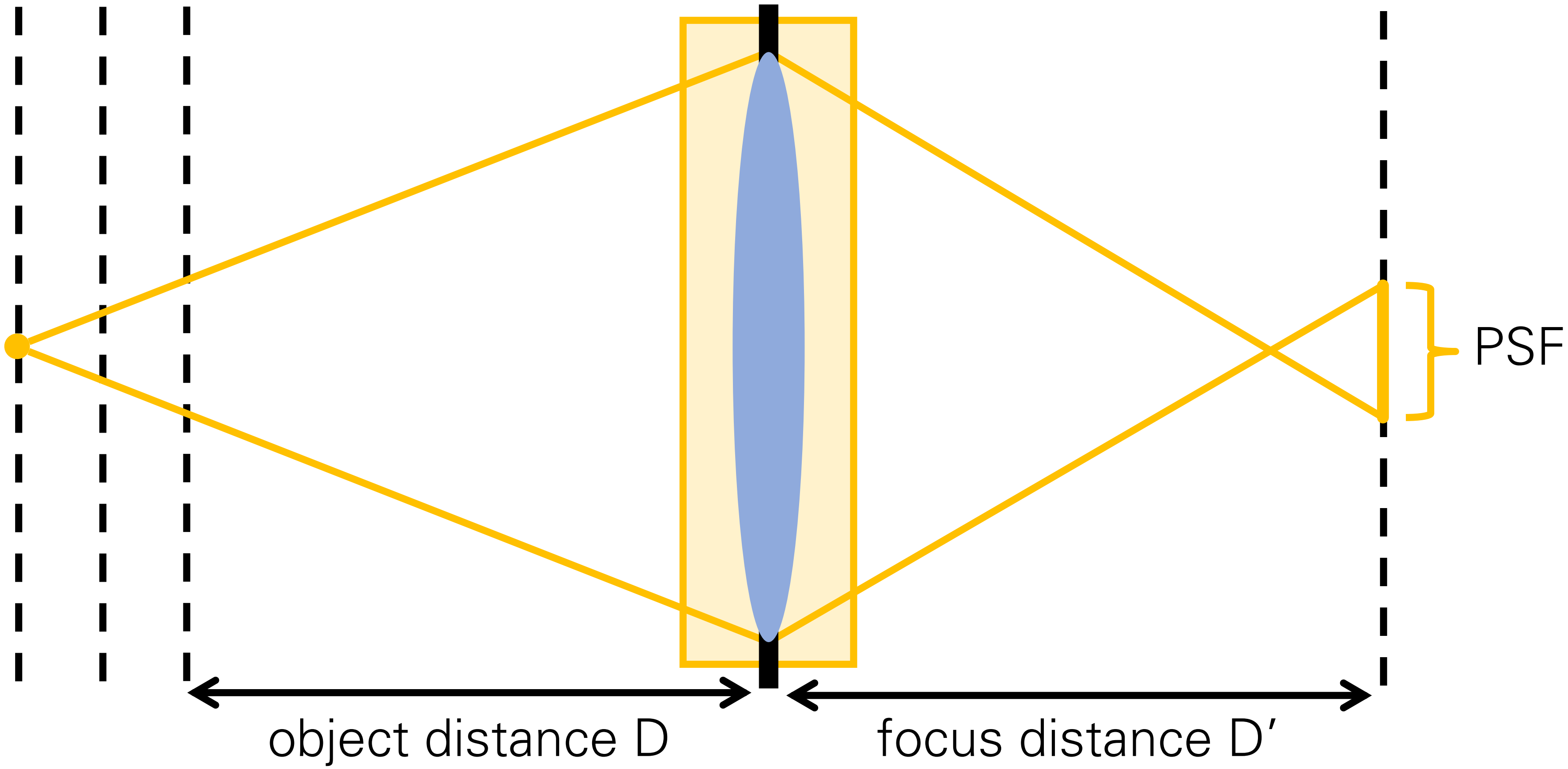
- Perfect depth invariance requires very constant speed.



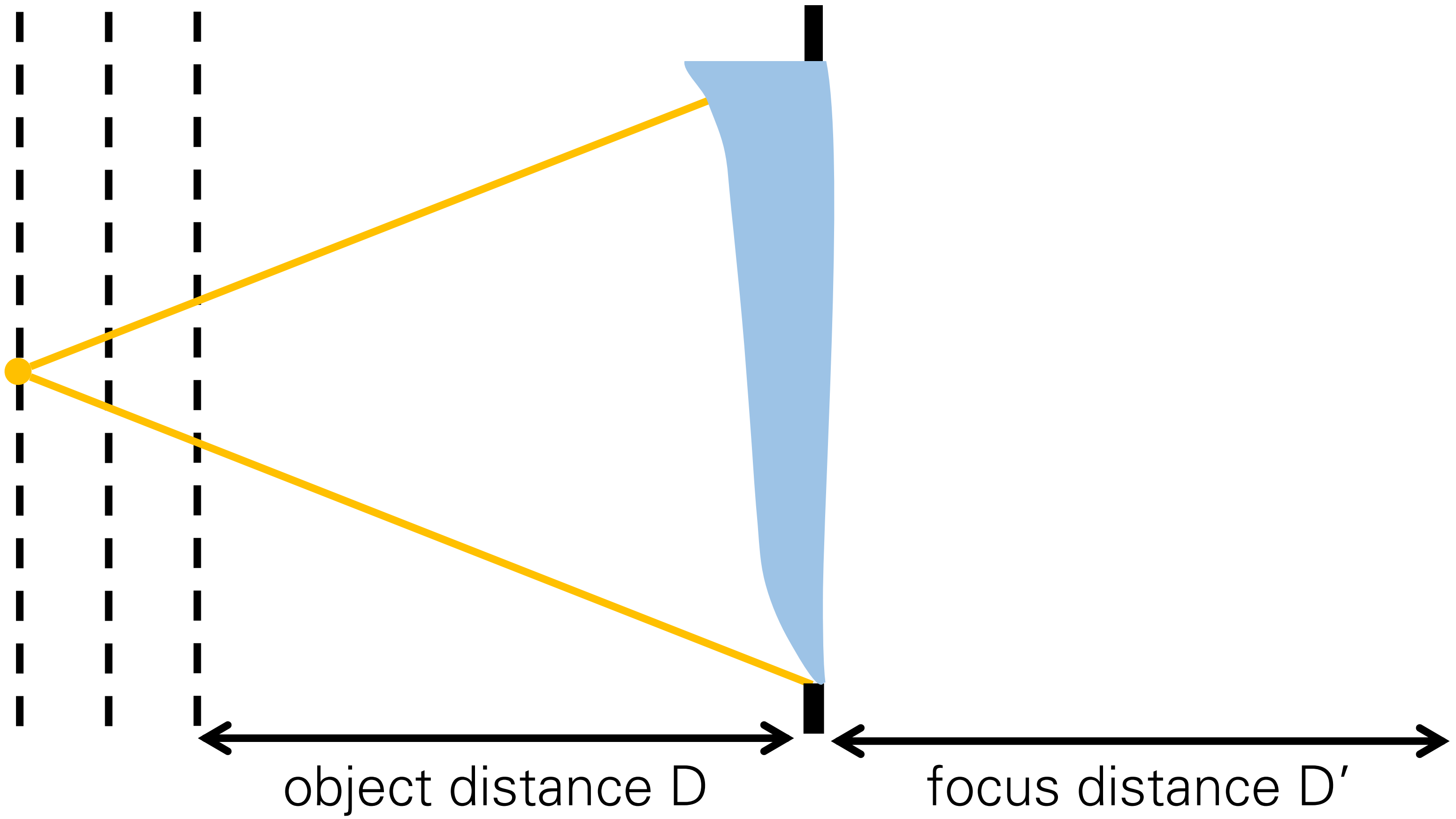
- We lose depth information.

Dealing with depth blur: generalized optics

Change optics, not aperture



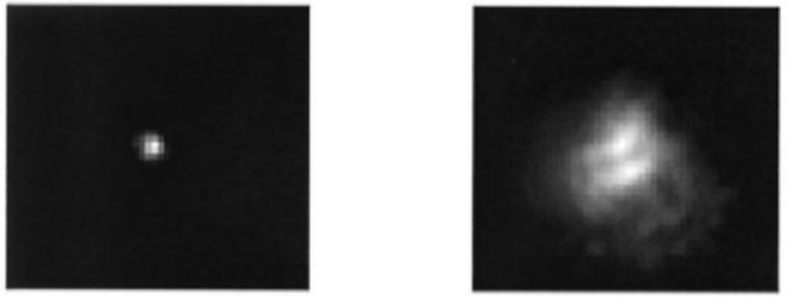
Wavefront coding



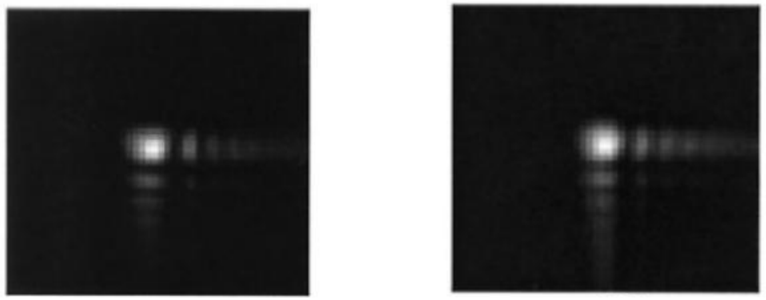
Replace lens with a cubic phase plate

Wavefront coding

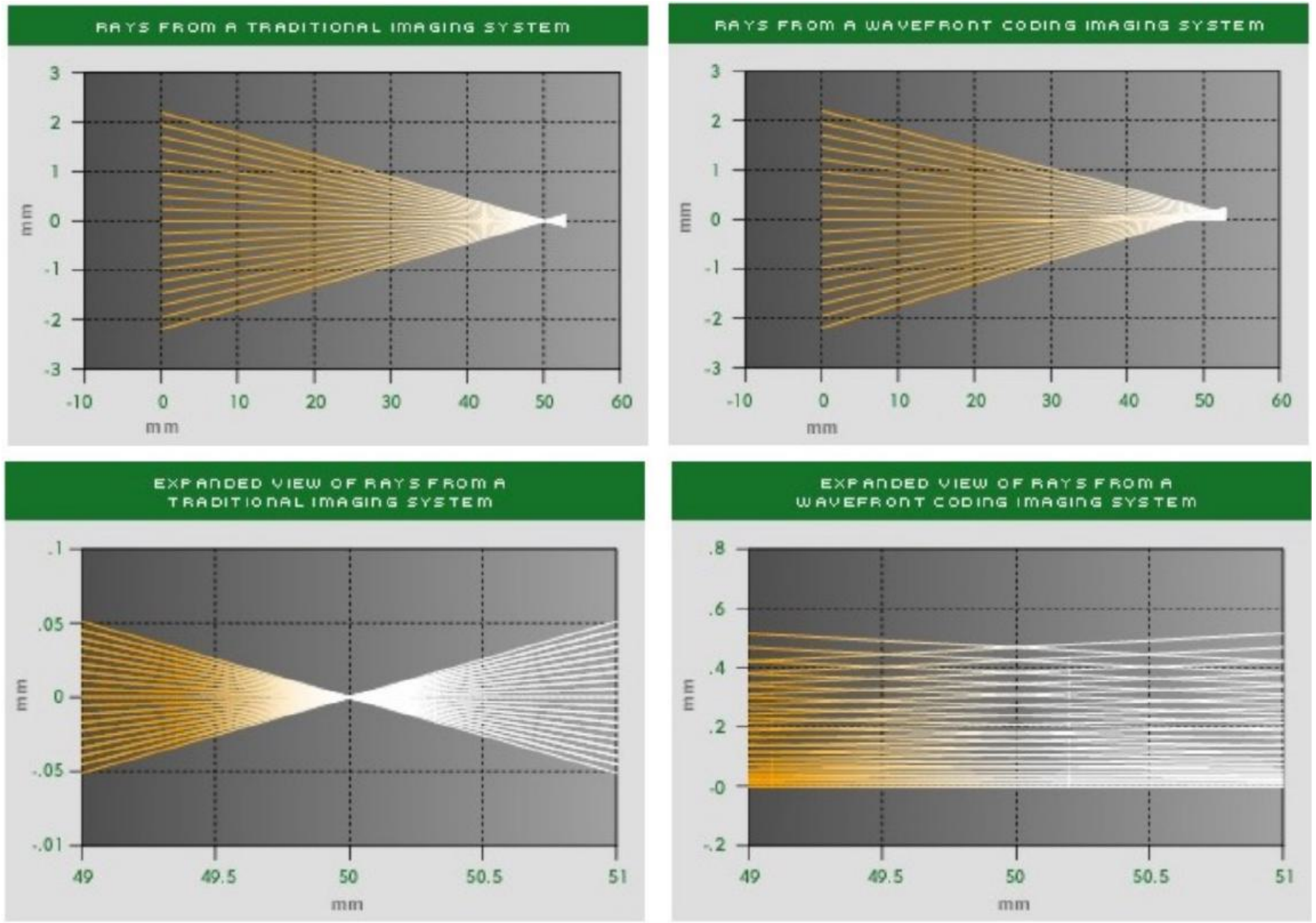
In focus	Out of focus



standard lens

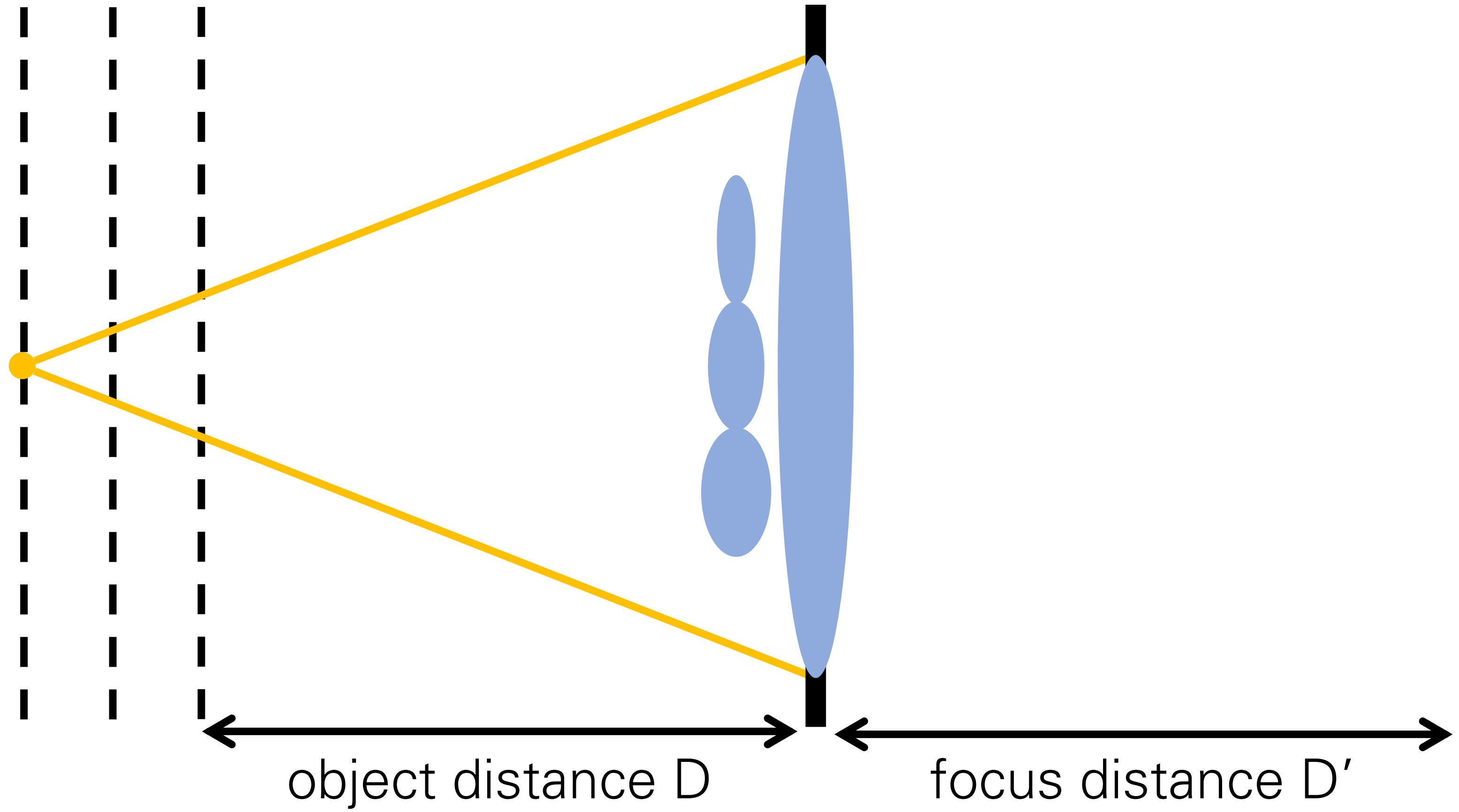


wavefront coding



- Rays no longer converge.
- Approximately depth-invariant PSF for certain range of depths.

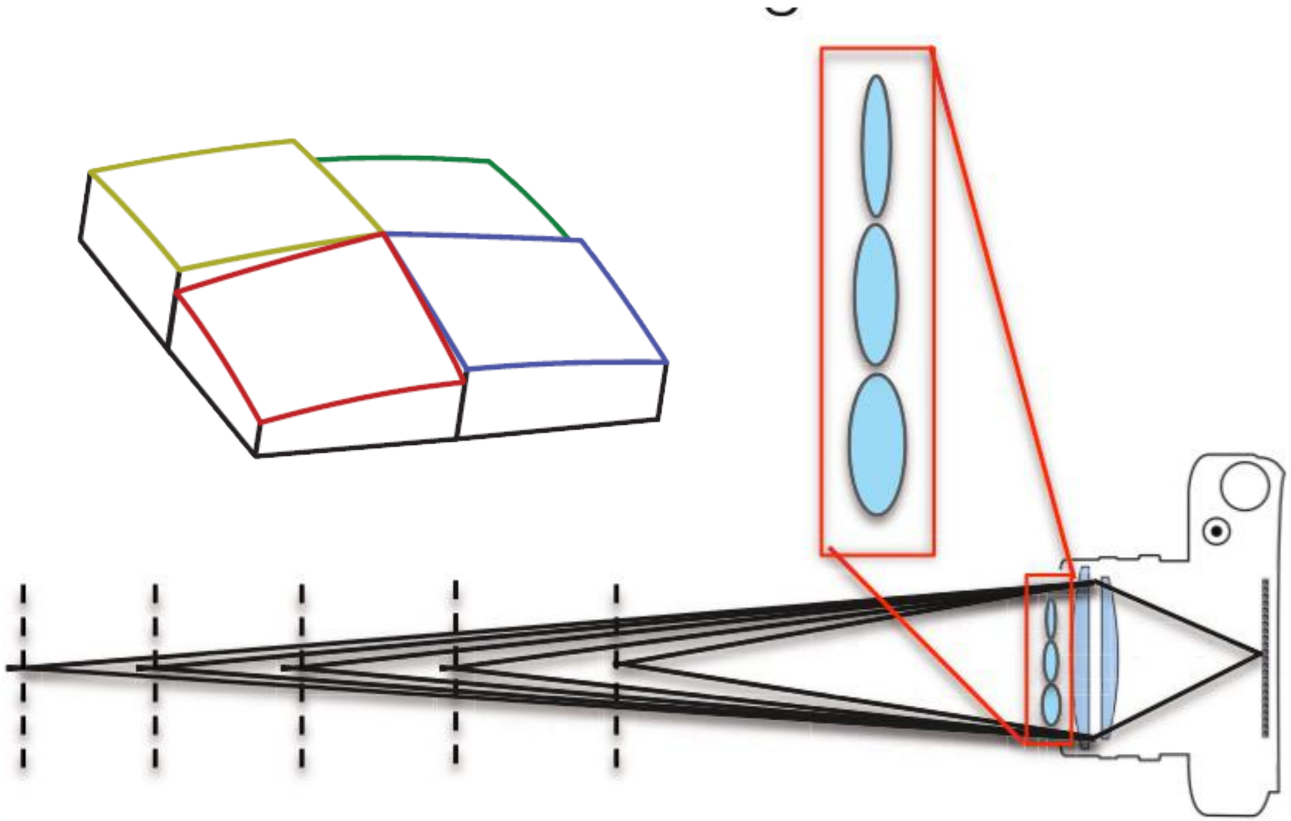
Lattice lens



Add lenslet array with varying focal length in front of lens

Lattice lens

Does this remind you of something?



Lattice lens

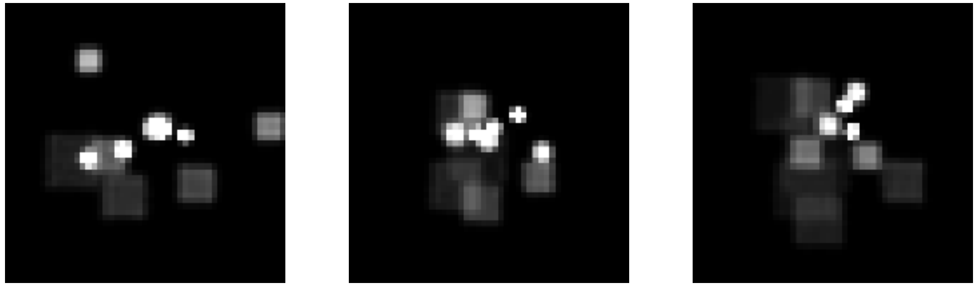
- Effectively captures only the "useful" subset of the 4D lightfield.

Light field spectrum: 4D
Image spectrum: 2D
Depth: 1D
3D
→ Dimensionality gap (Ng 05)

Only the 3D manifold corresponding to physical focusing distance is useful

- PSF is not depth-invariant, so local deconvolution as in coded aperture.

PSFs at different depths



Results

Results

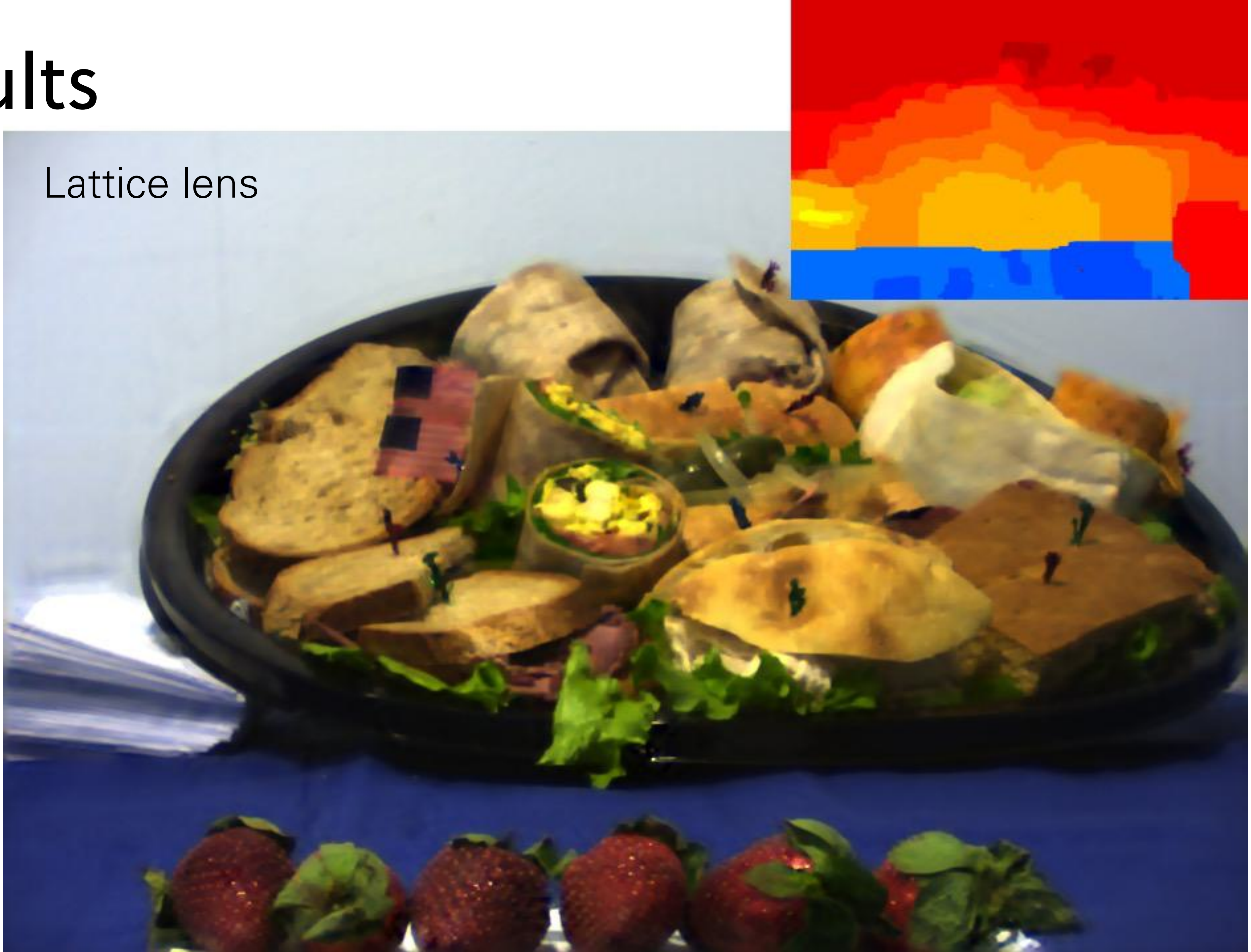

Lattice lens
Numerical Optimization
Multiple View Geometry
Hartley and Zisserman
Optics
Theory of Remote Image Formation
INTRODUCTION TO
Fourier Optics

Results

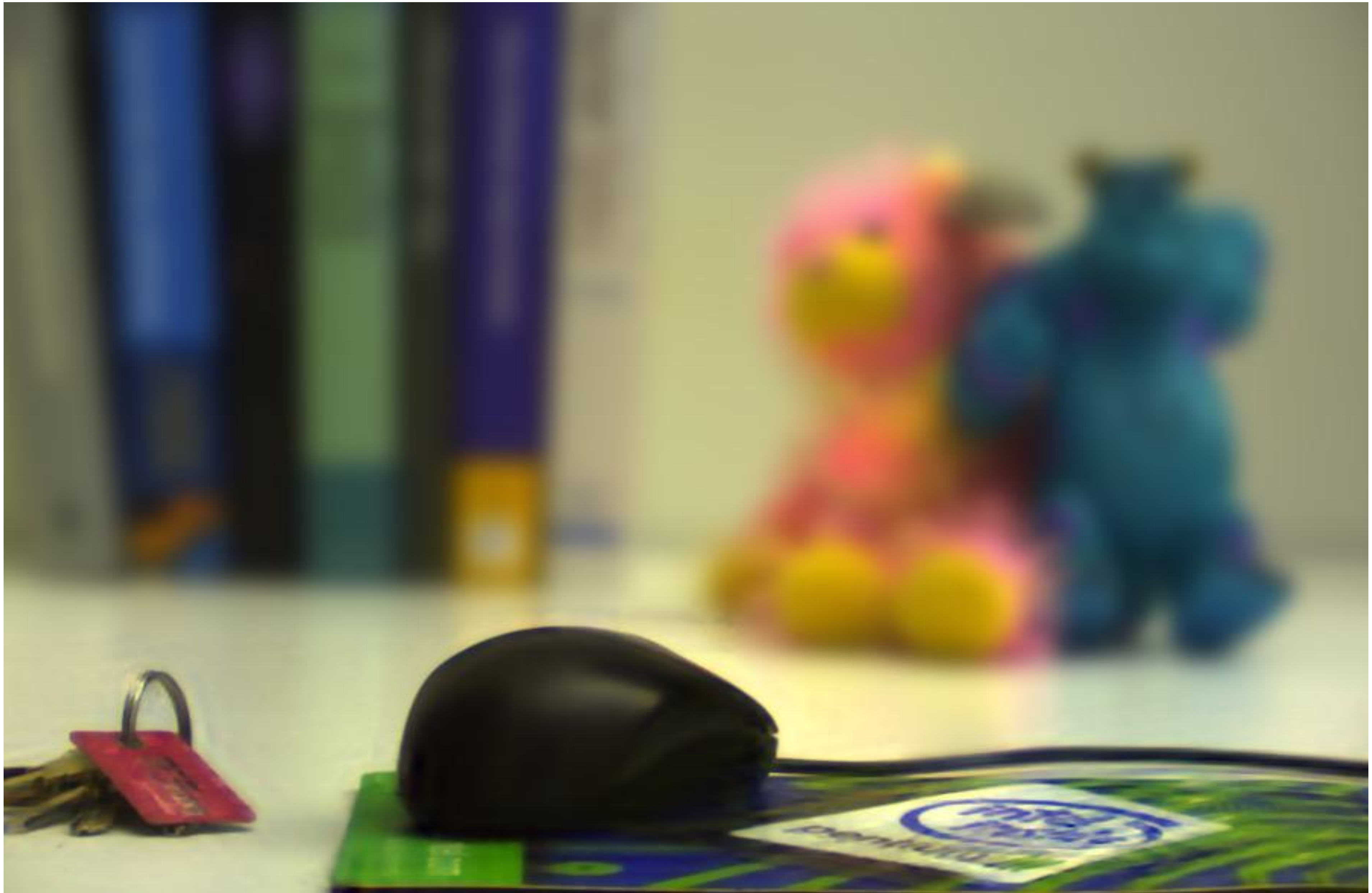
Results

Results

Results



Refocusing example



Refocusing example

Refocusing example

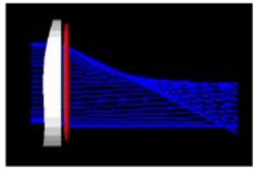
Comparison of different techniques

Depth of field comparison:

<

<<

<



<

standard lens | coded aperture | focal sweep | wavefront coding | lattice lens

Object at in-focus depth

Object at extreme depth

Diffusion coded photography

- can also do EDOF with diffuser as coded aperture, has better inversion characteristics than lattice focal lens

Conventional Camera

Diffusion Coded Camera

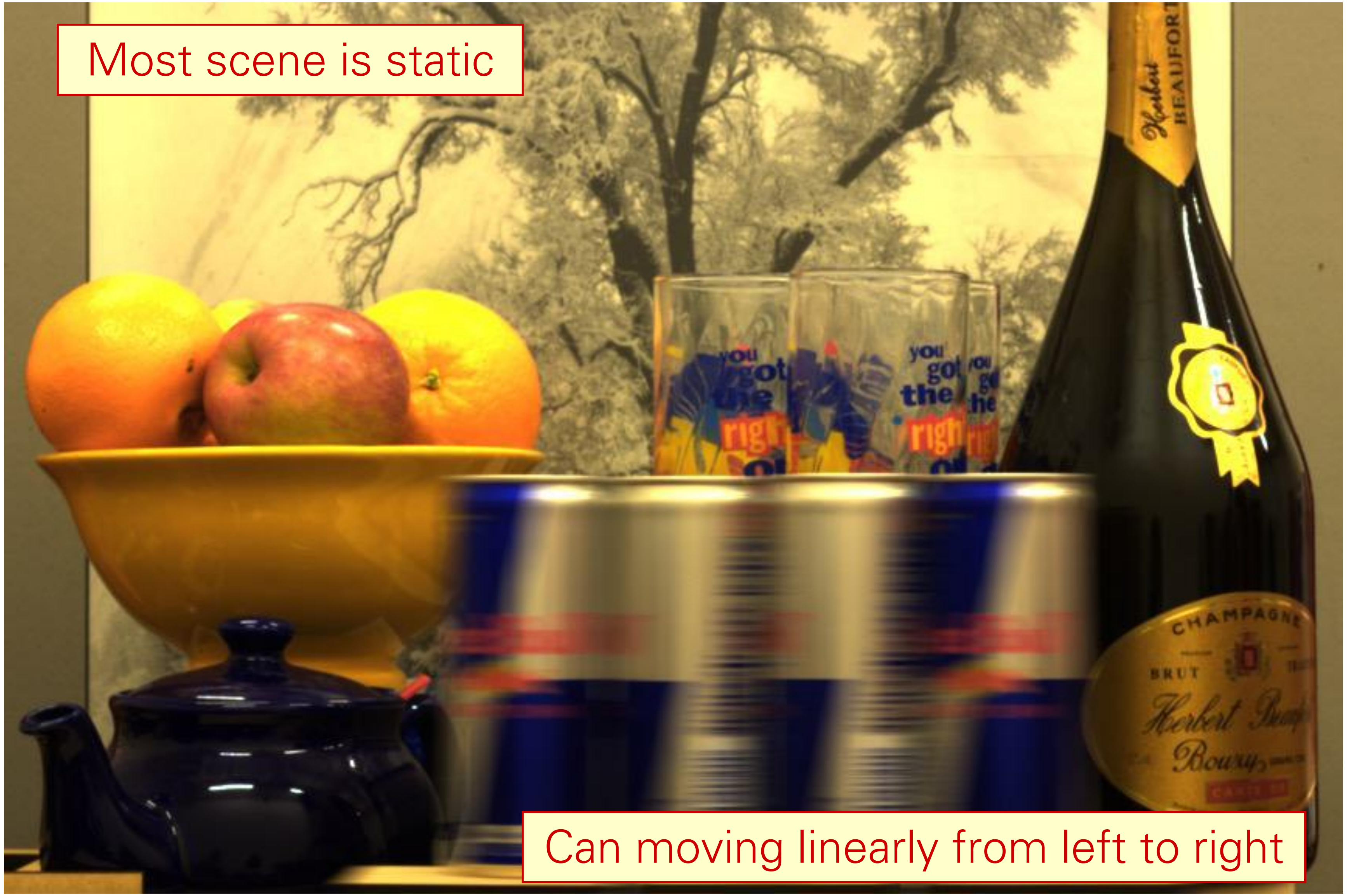
Can you think of any issues?

Dealing with motion blur

Why are our images blurry?

• Lens imperfections.	⟵	non-blind deconvolution	conventional photography
• Camera shake.	⟵	blind deconvolution	
• Scene motion.	⟵	flutter shutter, motion-invariant photo	coded photography
• Depth defocus.	⟵	coded aperture, focal sweep, lattice lens	

Motion blur



Motion blur

blurry image of
moving object

=

motion blur kernel

$*$

sharp image of
static object

What does the motion blur kernel depend on?

Motion blur

blurry image of
moving object

=

motion blur kernel

$*$

sharp image of
static object

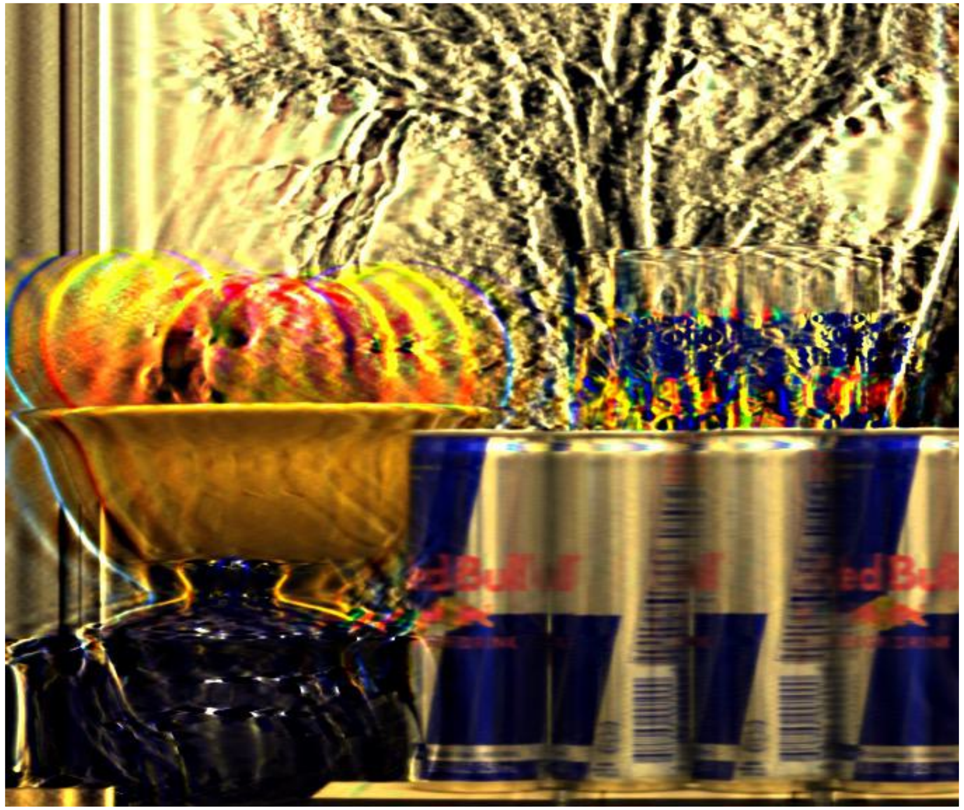
What does the motion blur kernel depend on?

- Motion velocity determines direction of kernel.
- Shutter speed determines width of kernel.

Can we use deconvolution to remove motion blur?

Challenges of motion deblurring

- Blur kernel is not invertible.
- Blur kernel is unknown.
- Blur kernel is different for different objects.



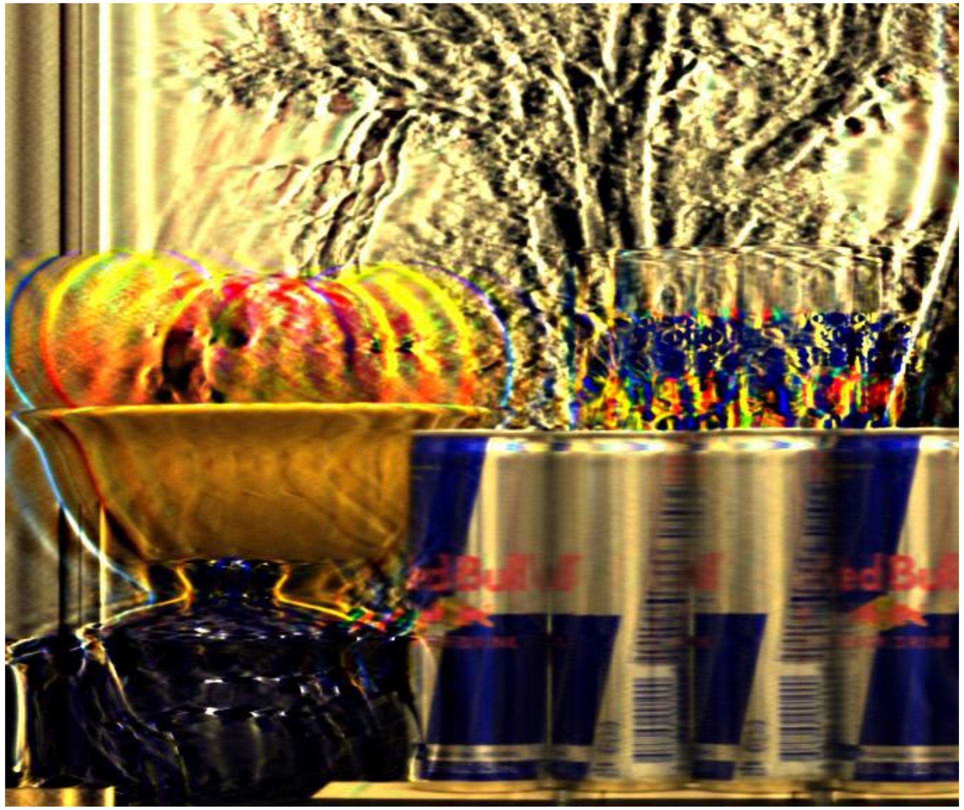
Challenges of motion deblurring

How would you deal with this?

- Blur kernel is not invertible.

- Blur kernel is unknown.

- Blur kernel is different for different objects.



Dealing with motion blur: coded exposure

Coded exposure a.k.a. flutter shutter

Code exposure (i.e., shutter speed) to make motion blur kernel better conditioned.

traditional camera

blurry image of moving object

=

motion blur kernel

*

sharp image of static object

flutter-shutter camera

blurry image of moving object

=

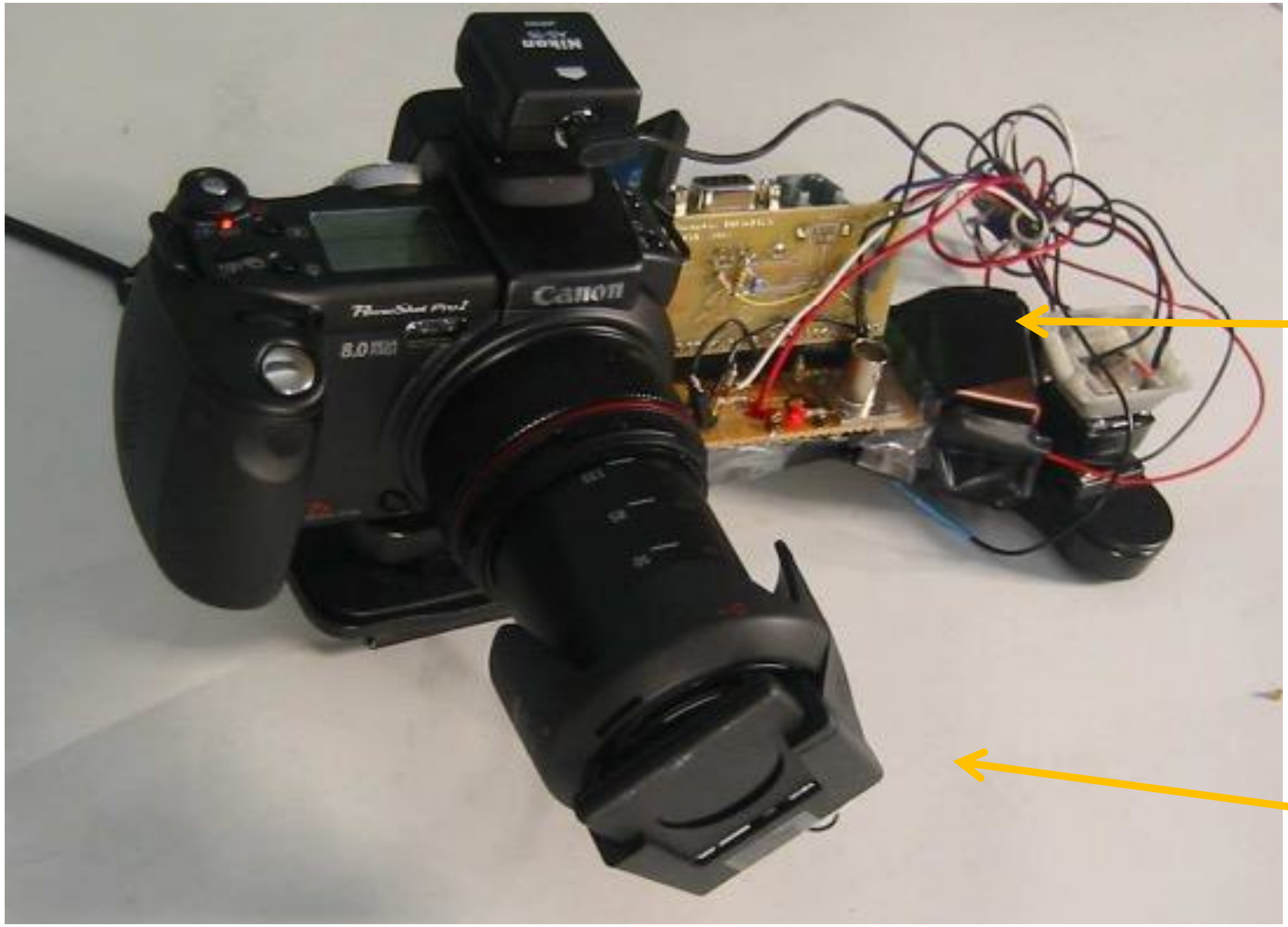
motion blur kernel

*

sharp image of static object

How would you implement coded exposure?

How would you implement coded exposure?



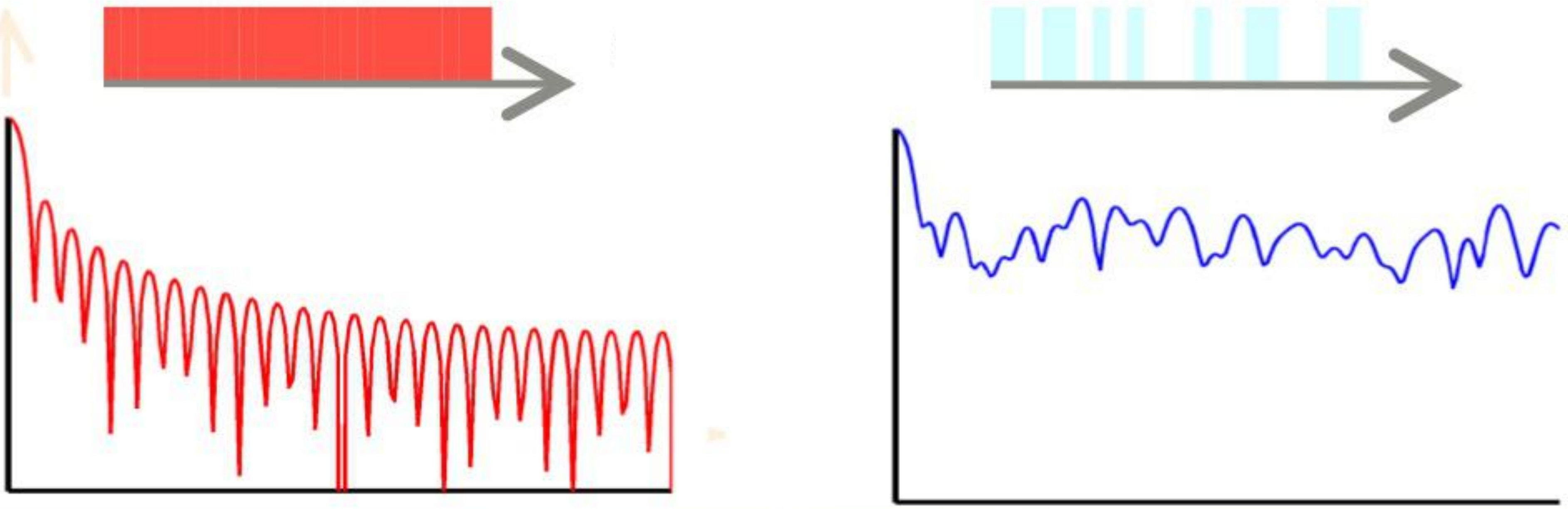
electronics for external shutter control

very fast external shutter

Coded exposure a.k.a. flutter shutter

motion blur kernel
in time domain

motion blur kernel
in Fourier domain

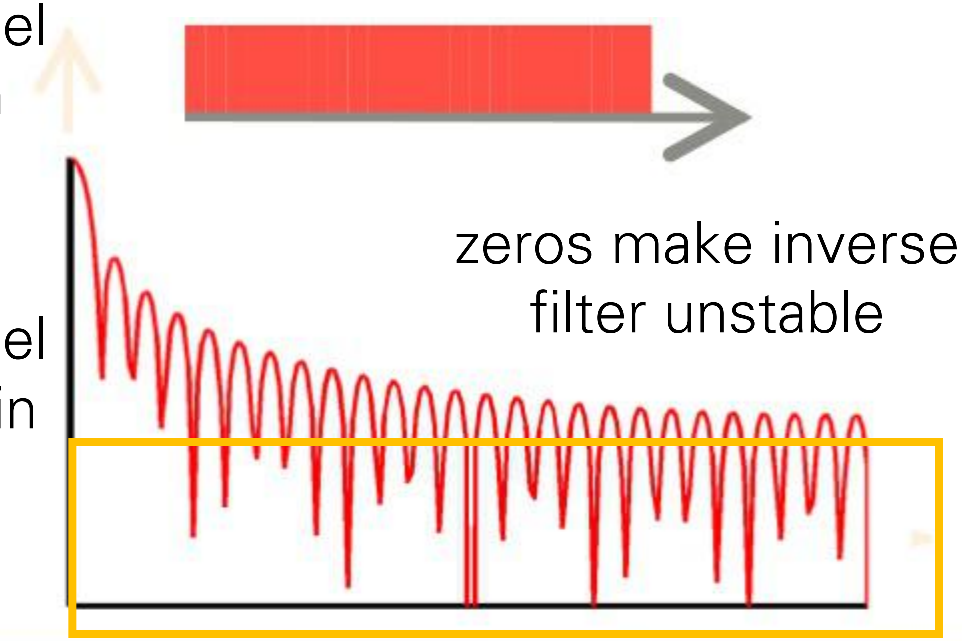


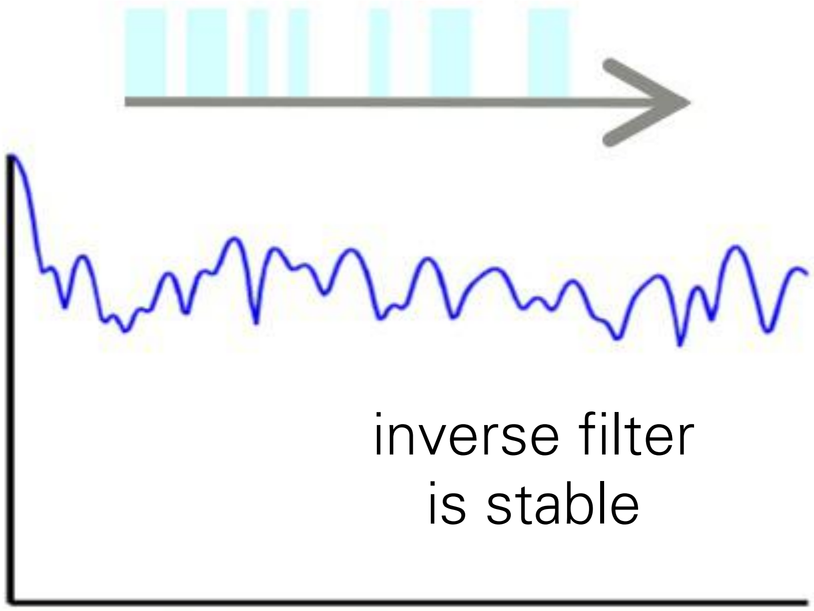
Why is flutter shutter better?

Coded exposure a.k.a. flutter shutter

motion blur kernel
in time domain

motion blur kernel
in Fourier domain





Why is flutter
shutter
better?

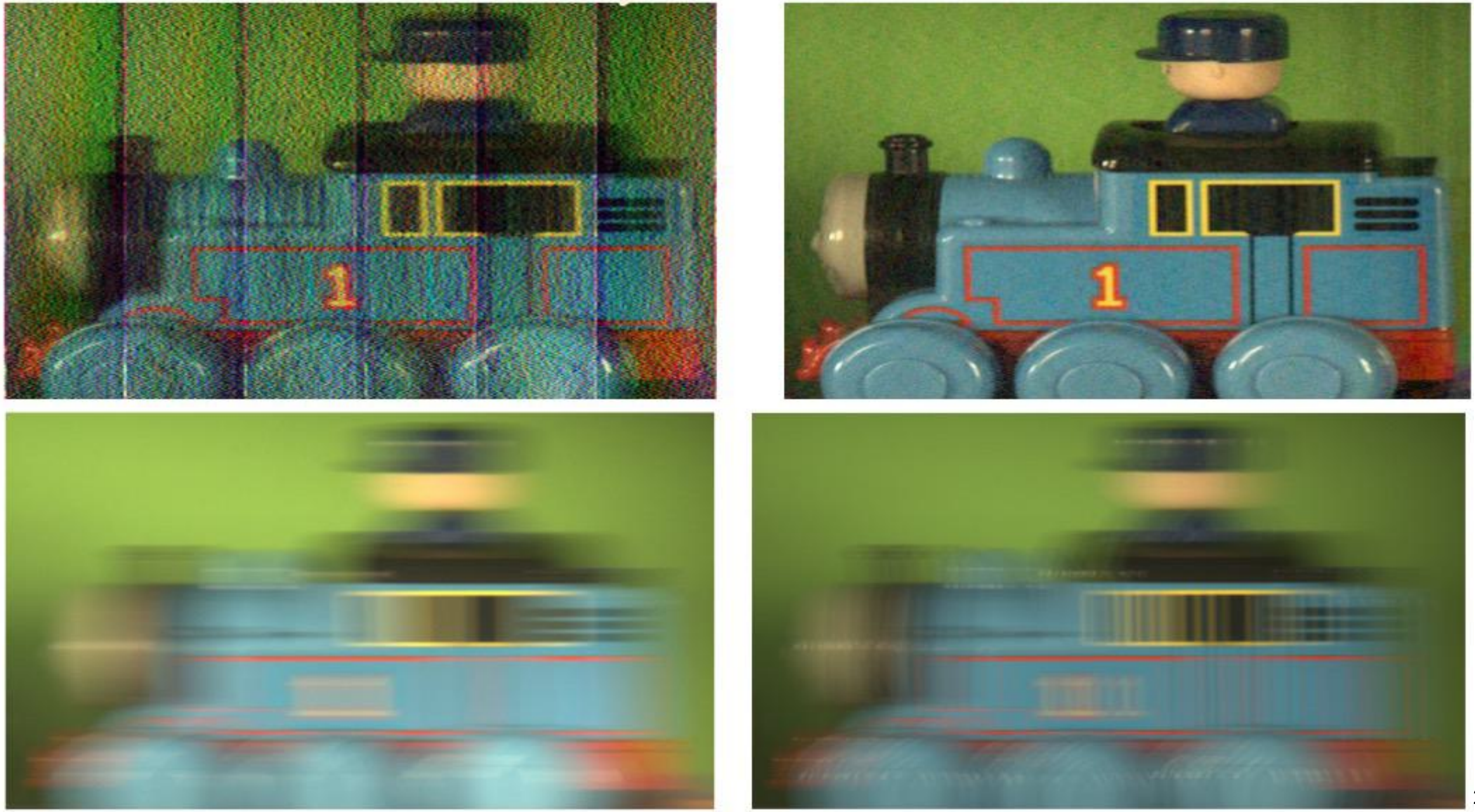
Motion deblurring comparison

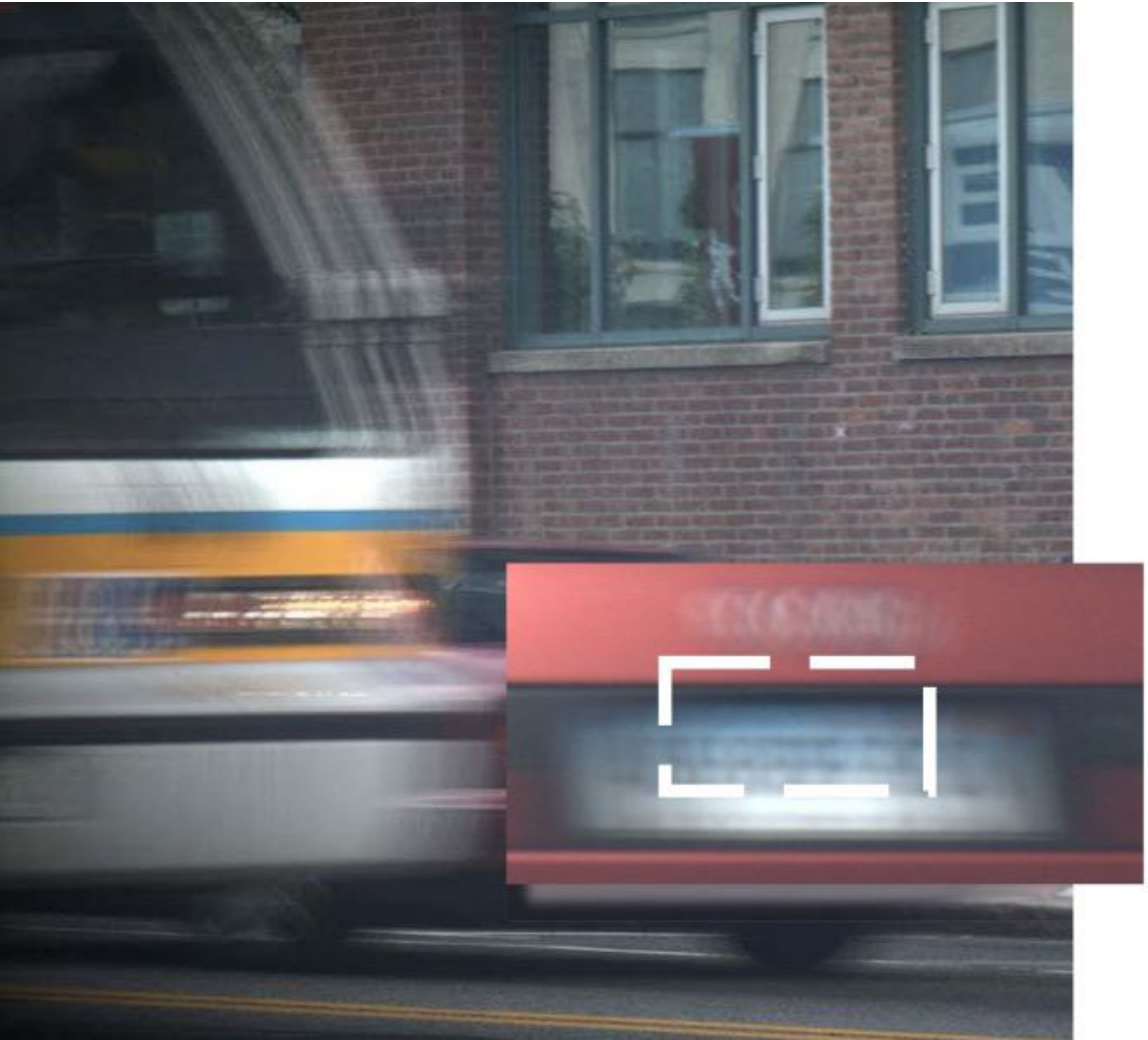
conventional photography

flutter-shutter photography

deconvolved output

blurry input





License Plate Retrieval

License Plate Retrieval

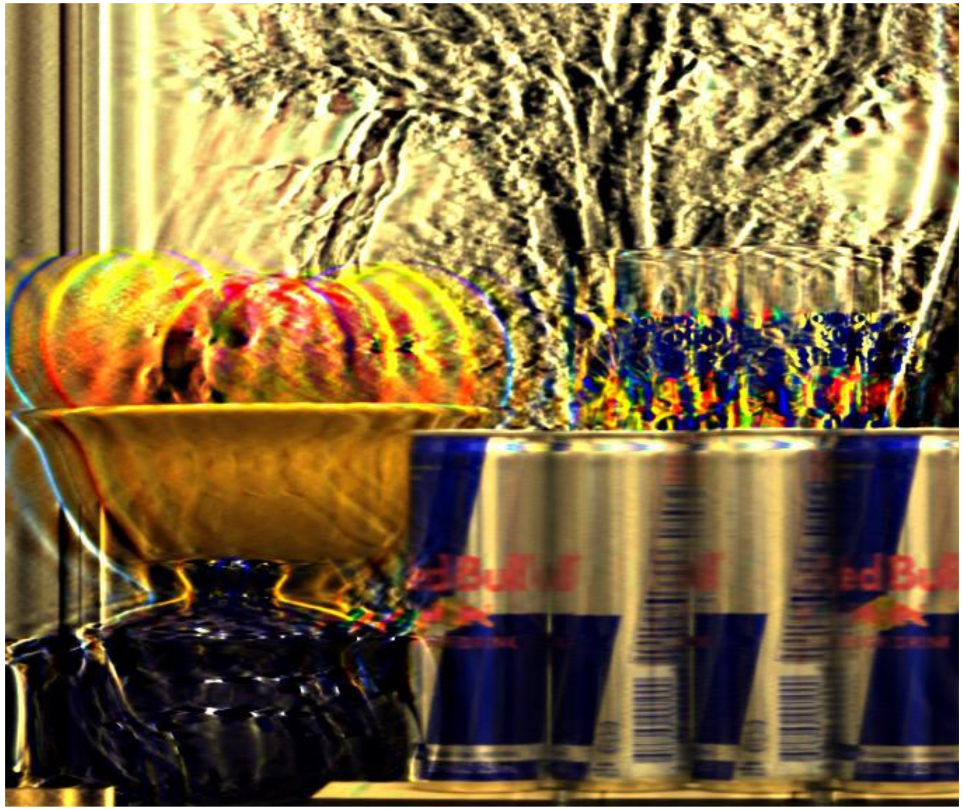
Challenges of motion deblurring

- Blur kernel is not invertible.

- Blur kernel is unknown.

How would you deal with these two?

- Blur kernel is different for different objects.



Dealing with motion blur: parabolic sweep

Motion-invariant photography

Introduce extra motion so that:

- Everything is blurry; and
- The blur kernel is motion invariant (same for all objects).

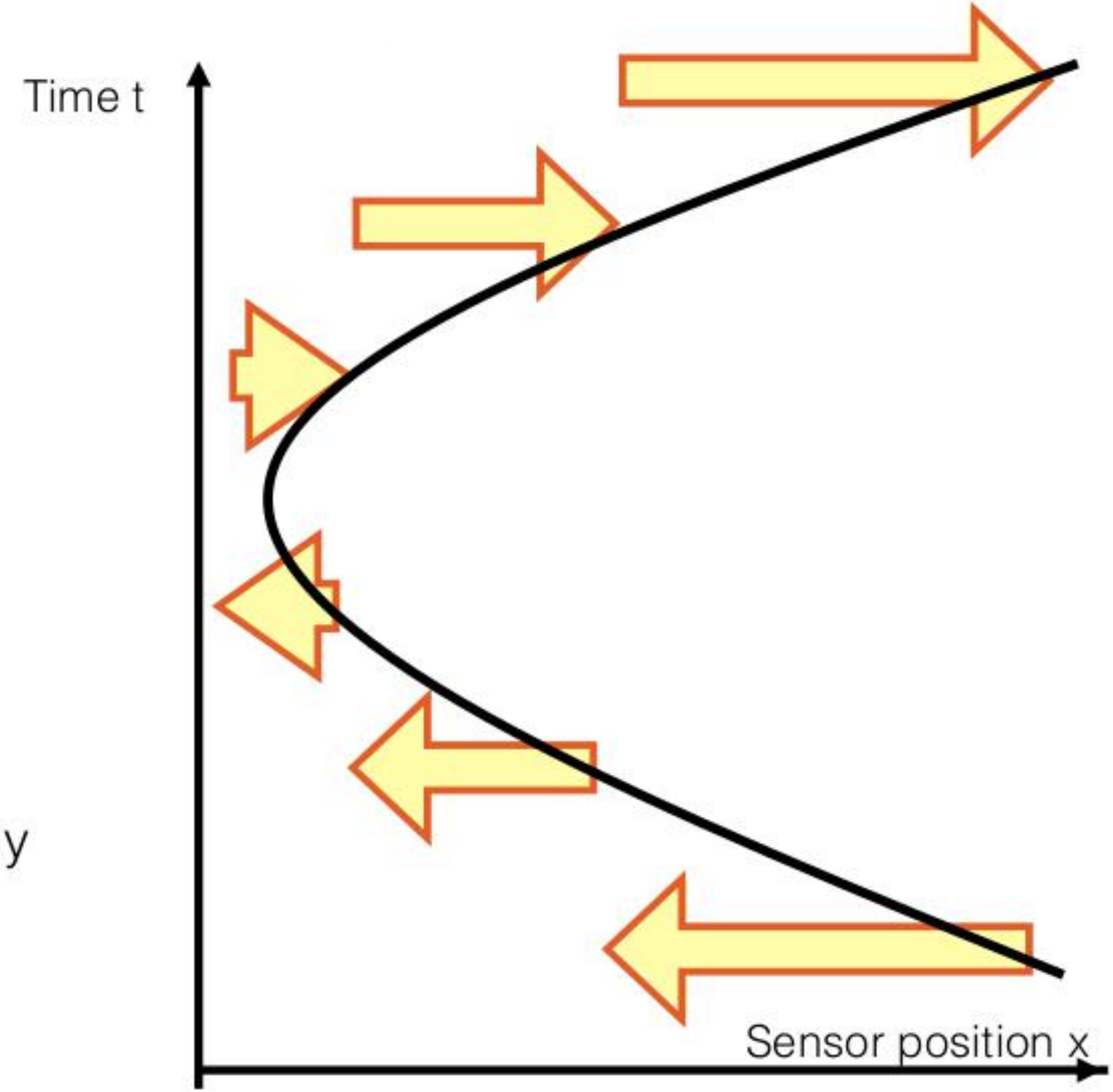
How would you achieve this?

Parabolic sweep

Sensor position $x(t)=a\ t^2$

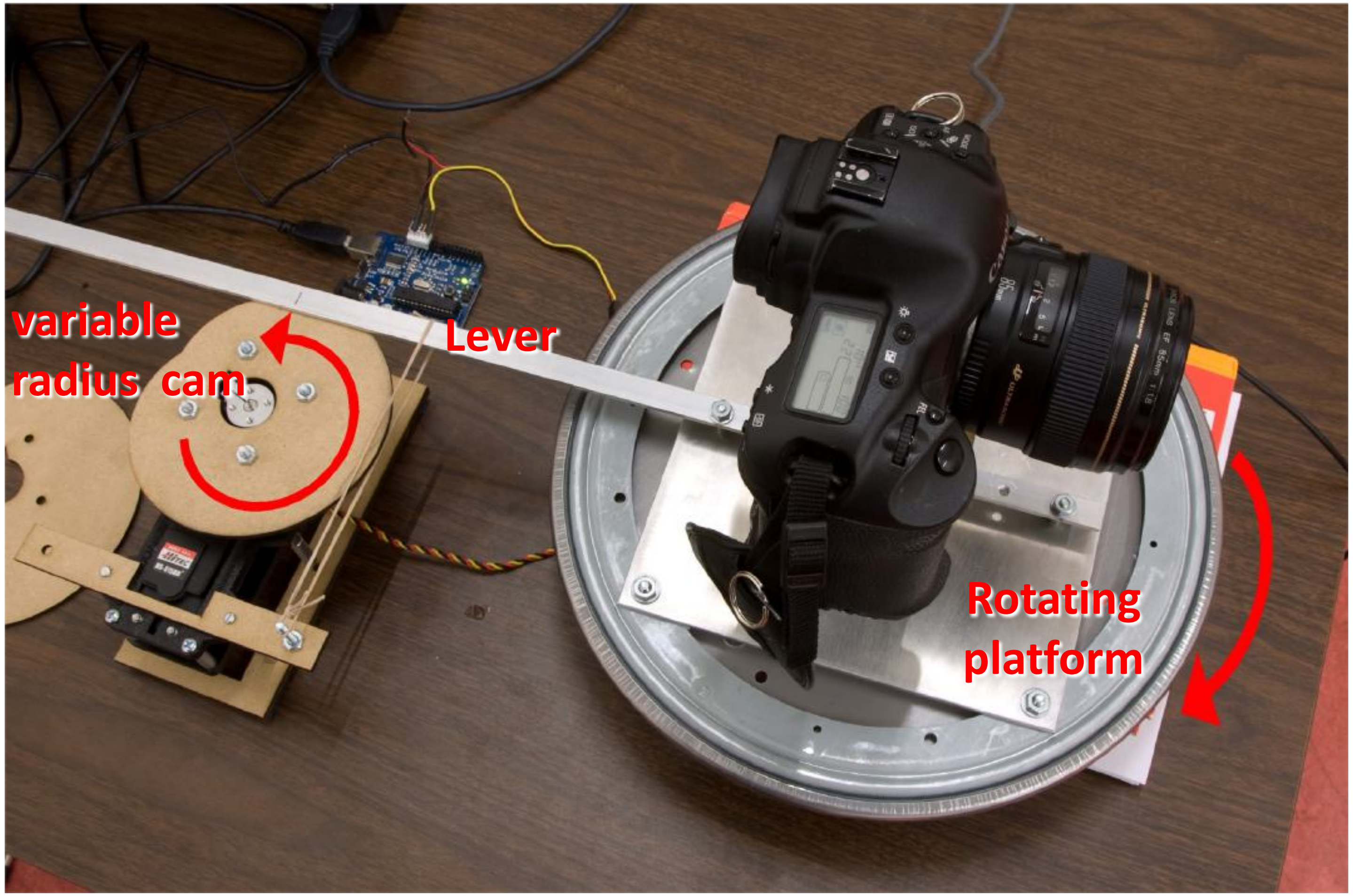
- start by moving very fast to the right
- continuously slow down until stop
- continuously accelerate to the left

- Intuition:
  - for any velocity, there is one instant where we track perfectly
  - all velocities captured same amount of time



Hardware implementation

Approximate small translation by small rotation



Some results

static camera input -
unknown and variable blur

parabolic input - blur is
invariant to velocity

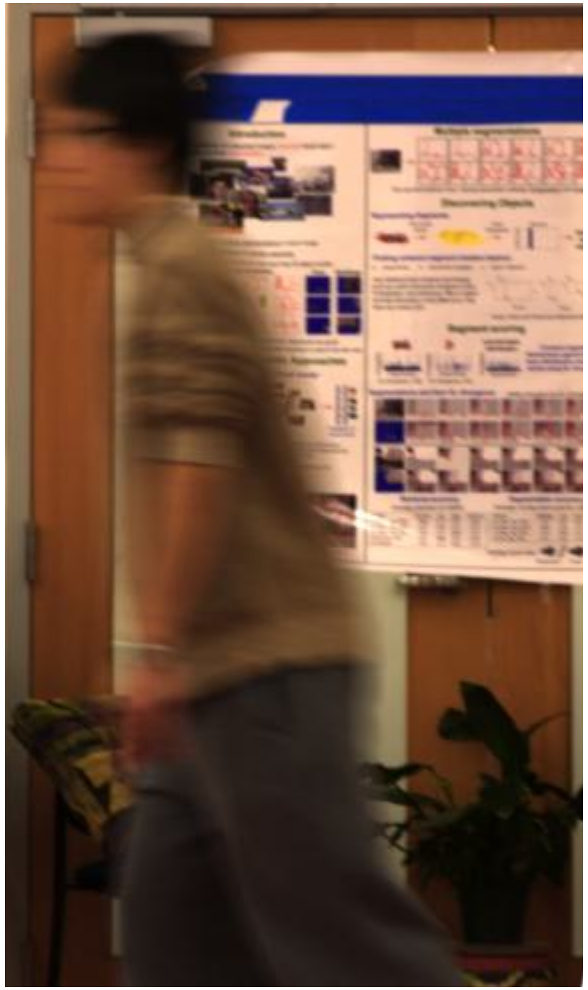
Some results

static camera input -
unknown and variable blur

output after deconvolution

Is this blind or non-blind deconvolution?

Some results



static camera input

parabolic camera input

deconvolution output

Next Lecture:
Convolutional Neural Networks