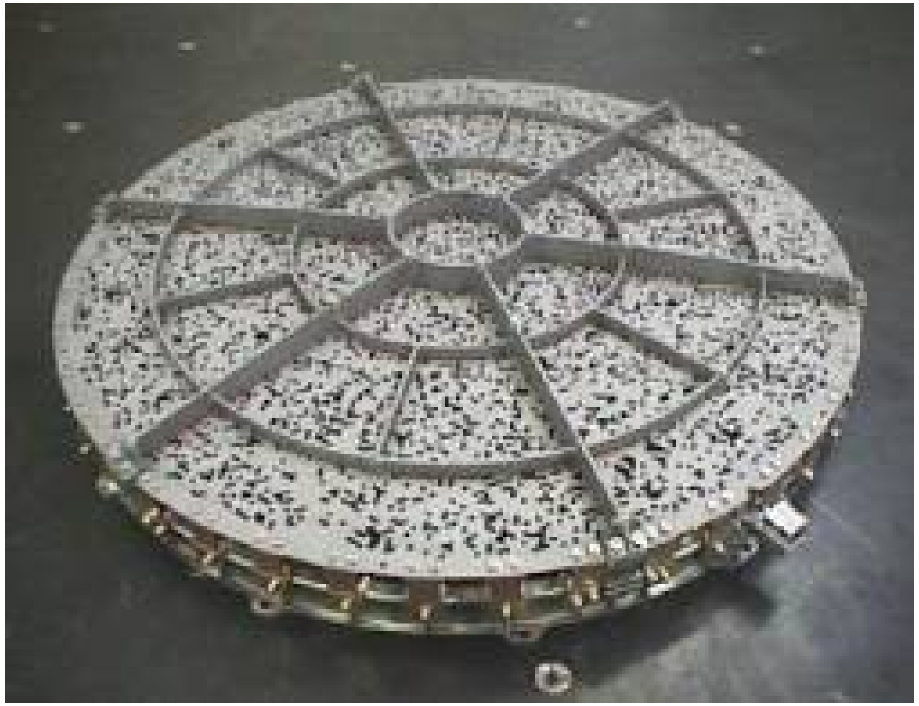
Coded photography



15-463, 15-663, 15-862
Computational Photography
Fall 2020, Lecture 15

http://graphics.cs.cmu.edu/courses/15-463

Course announcements

- Homework assignment 4 due November 2nd.
  - Generally shorter to accommodate final project proposals.
  - Two bonus parts.
- Project logistics on Piazza and the course website.
  - Feedback on project ideas posted.
  - Project proposals due on Gradescope on October 30th.
- Office hour logistics for this week:
  - Yannis will have extra office hours on Friday (time TBD).
- Please take the mid-semester survey:

https://docs.google.com/forms/d/e/1FAIpQLScogA89LrEgwGiCDZ2DfzRhwZpFz5SdC_fci_LejbirrLT5hw/viewform

Overview of today's lecture

- Leftover from deconvolution lecture.
- The coded photography paradigm.
- Dealing with depth blur: coded aperture.
- Dealing with depth blur: focal sweep.
- Dealing with depth blur: generalized optics.
- Dealing with motion blur: coded exposure.
- Dealing with motion blur: parabolic sweep.

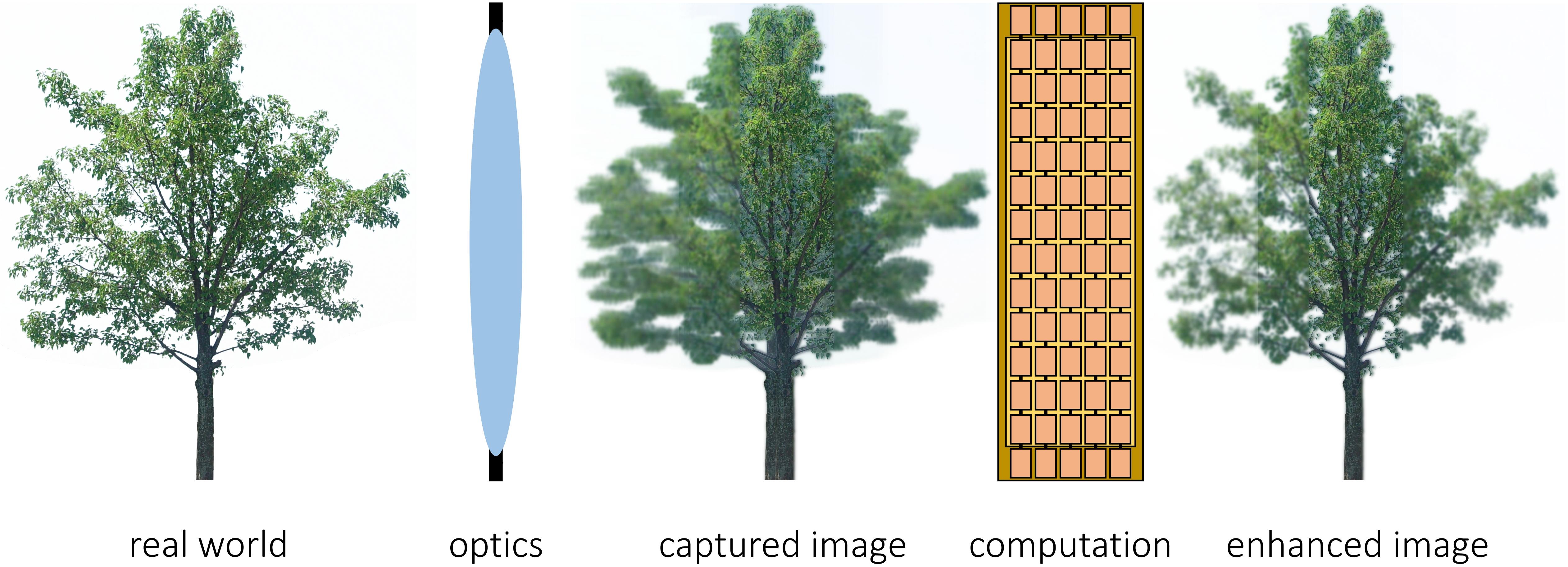
Slide credits

Most of these slides were adapted from:

- Fredo Durand (MIT).
- Anat Levin (Technion).
- Gordon Wetzstein (Stanford).

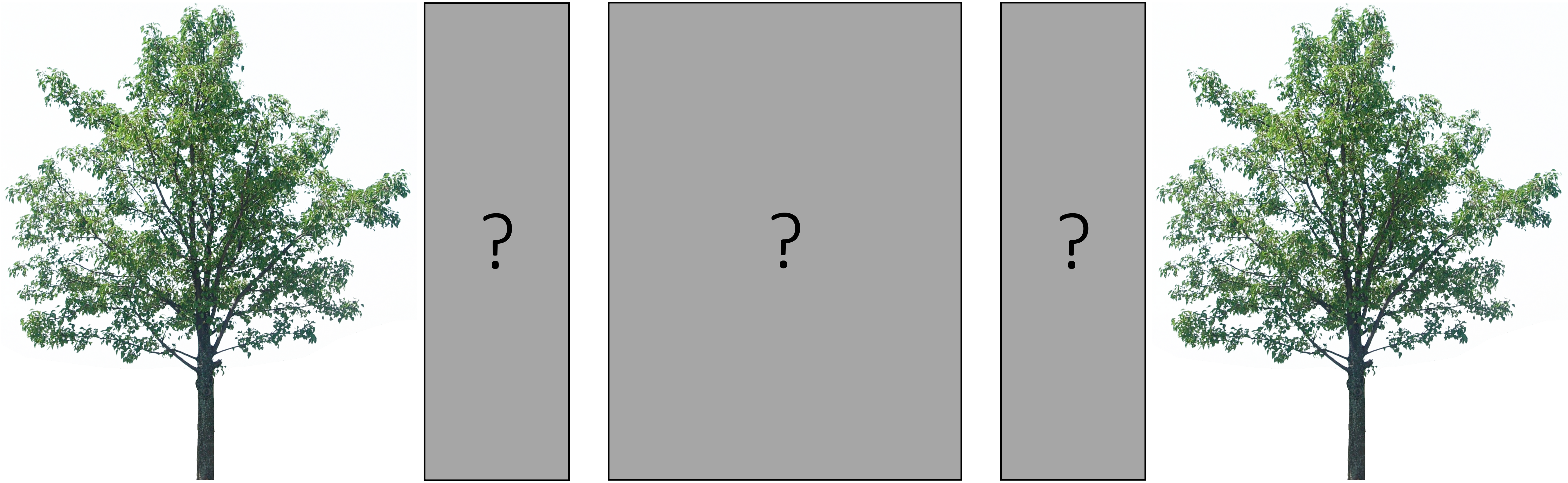
The coded photography paradigm

Conventional photography



- 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.

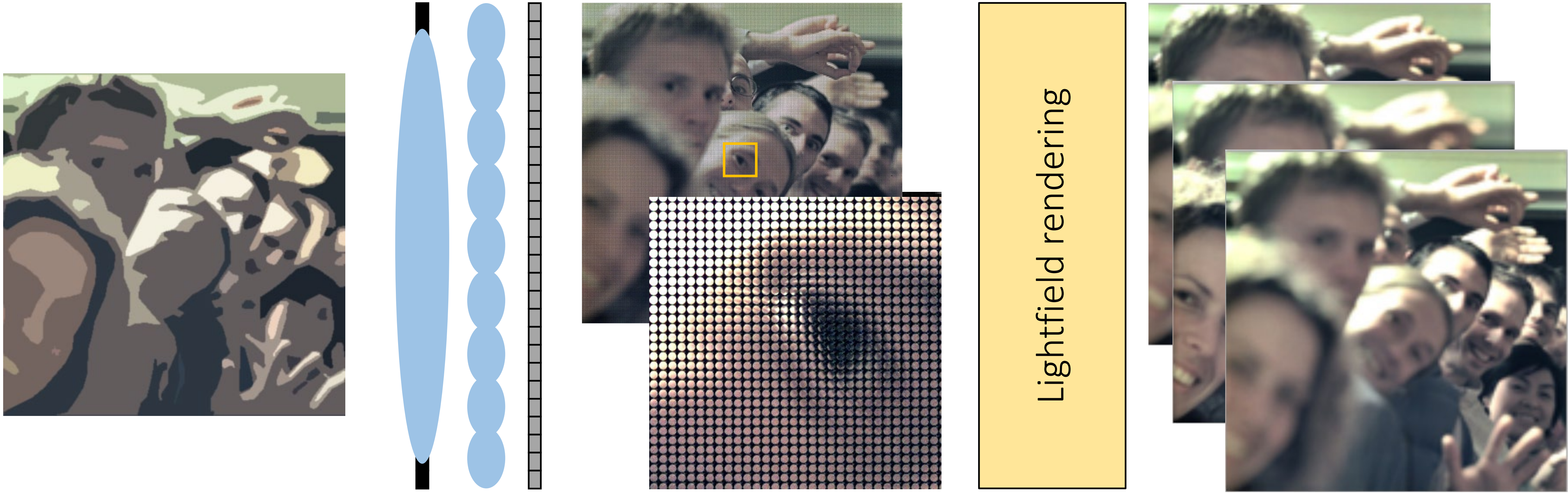
Can you think of any examples?

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



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

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

Why are our images blurry?

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

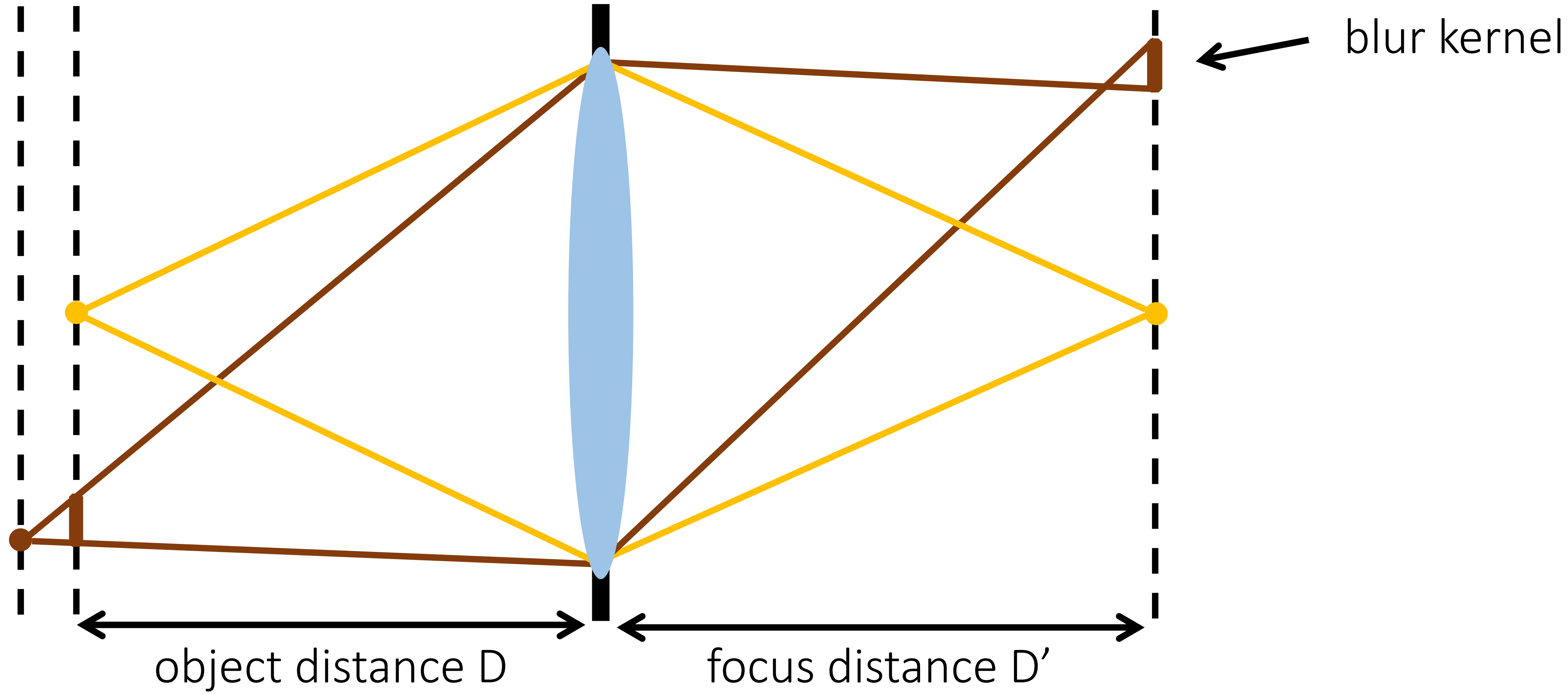
} conventional photography (lens imperfections, camera shake)

} coded photography (scene motion, depth defocus)

Dealing with depth blur: coded aperture

Defocus blur

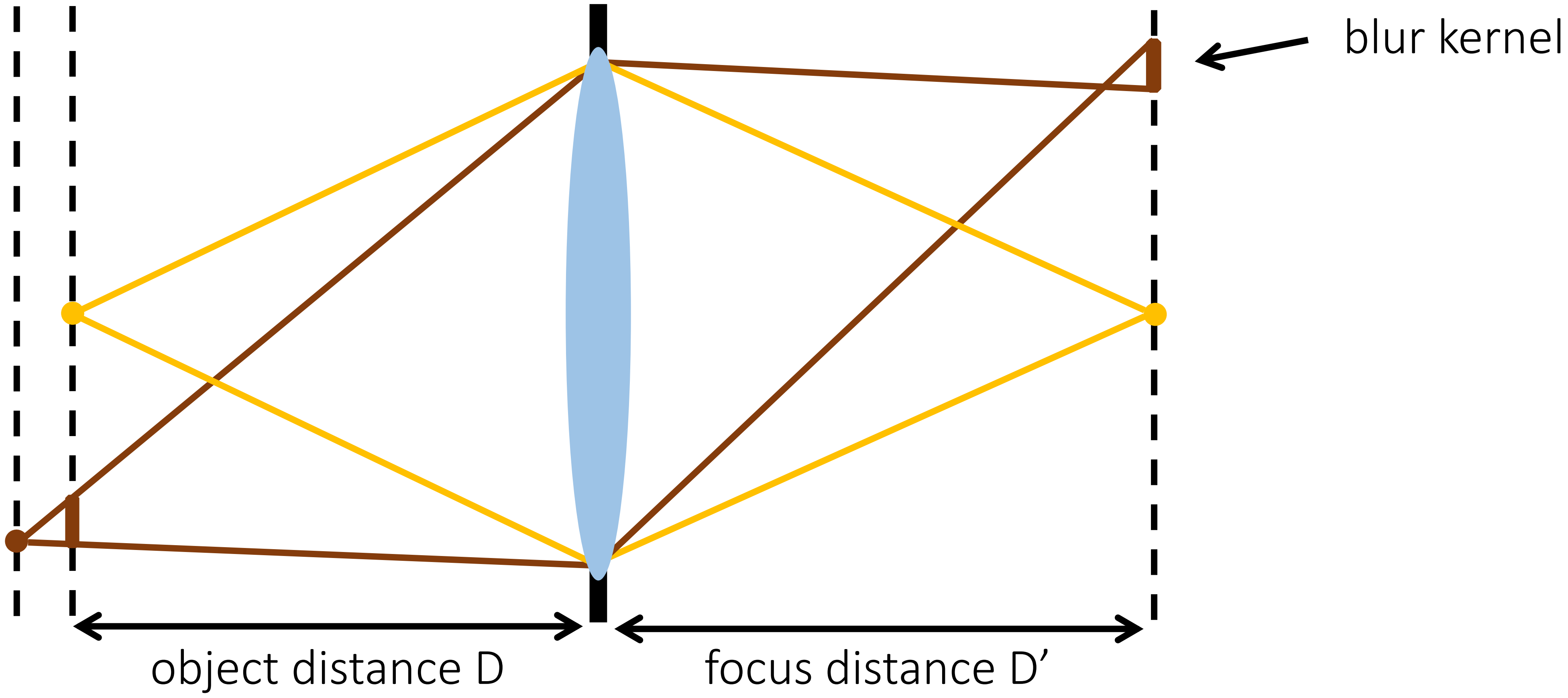
Point spread function (PSF): The blur kernel of a (perfect) lens at some out-of-focus depth.



What does the blur kernel depend on?

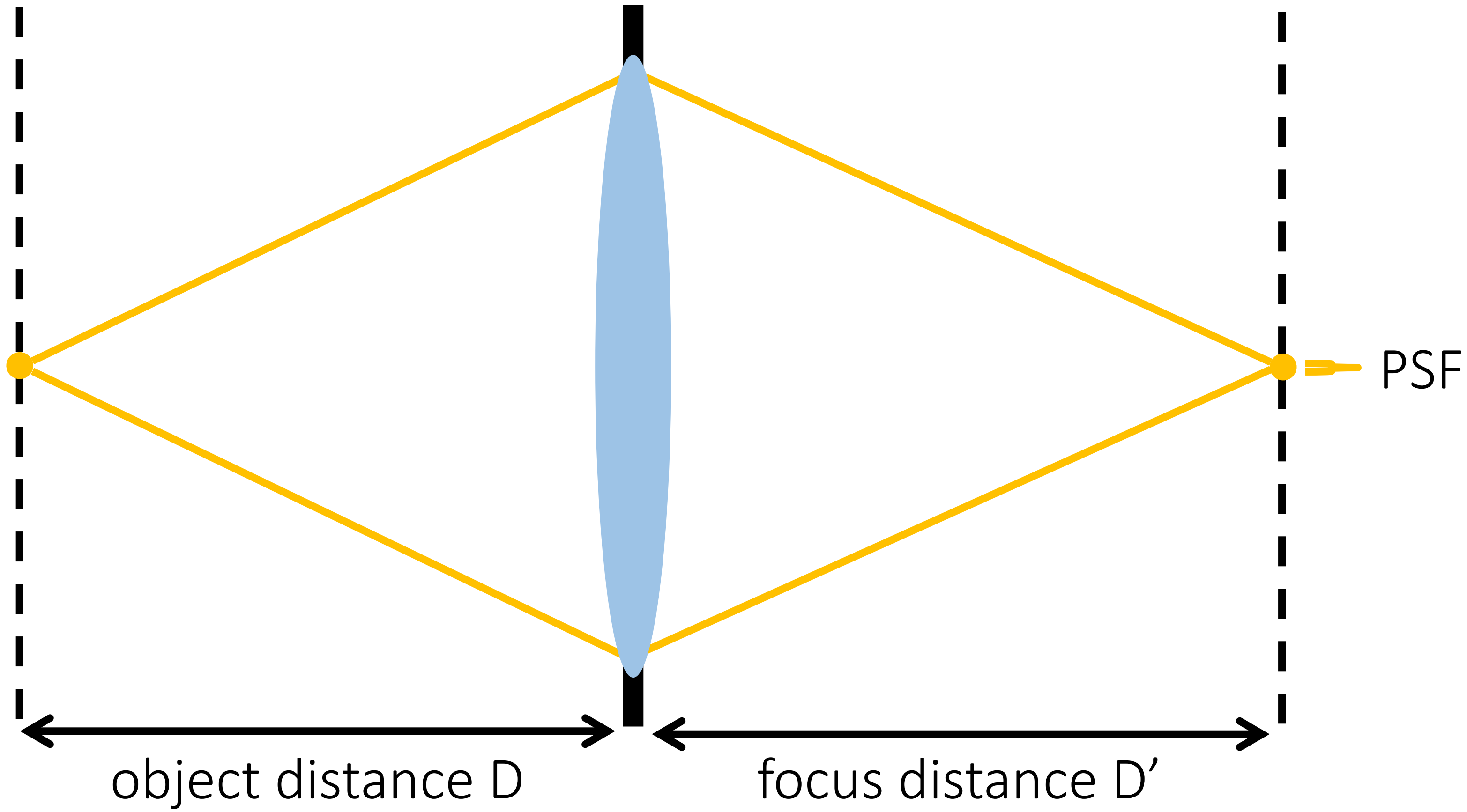
Defocus blur

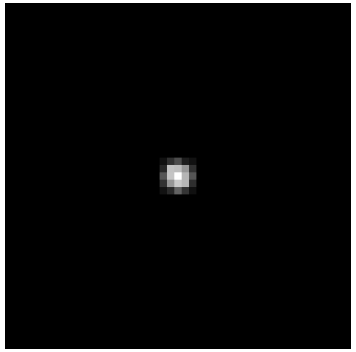
Point spread function (PSF): The blur kernel of a (perfect) lens at some out-of-focus depth.



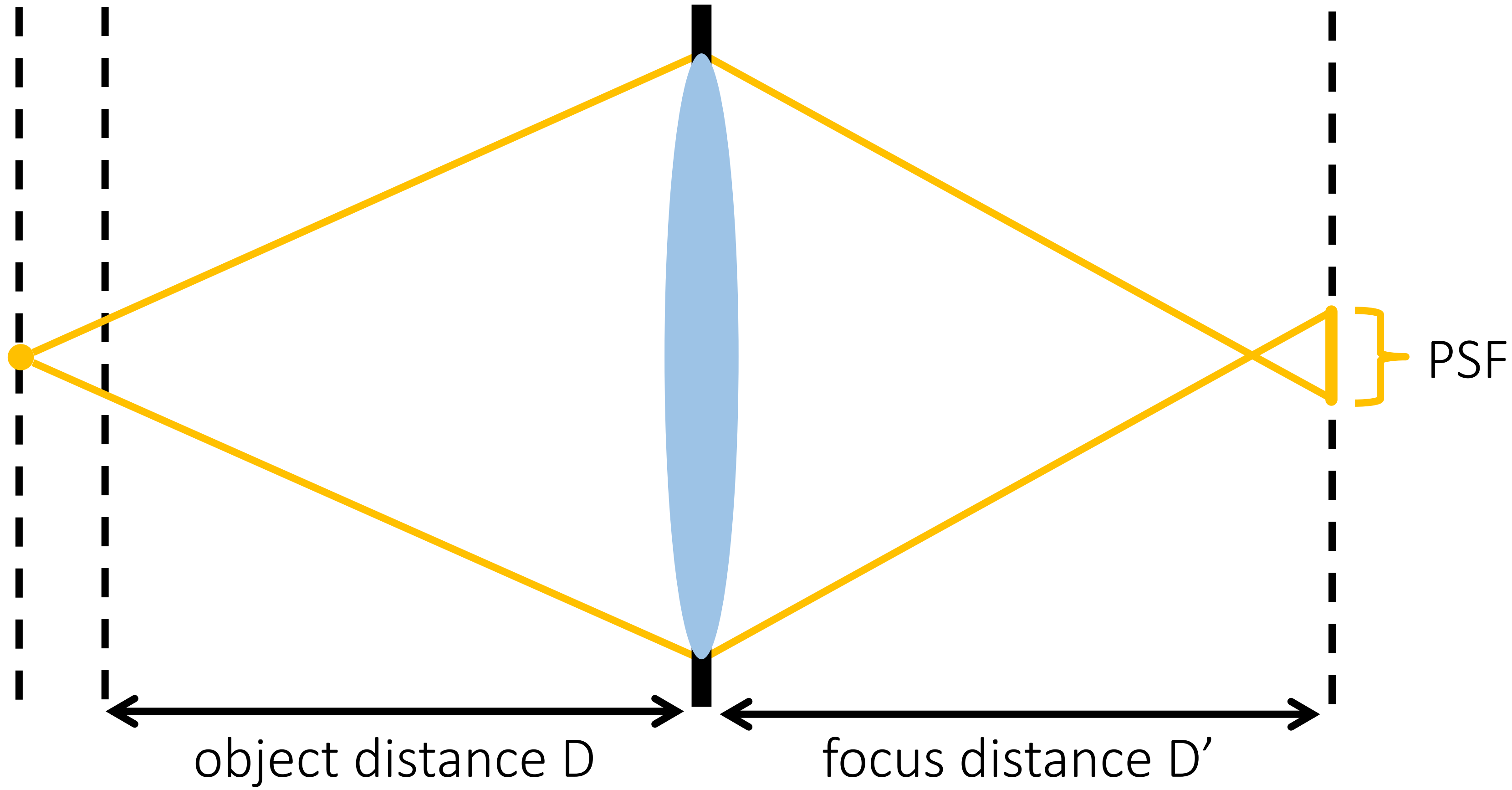
- Aperture determines *shape* of kernel.
- Depth determines *scale* of blur kernel.

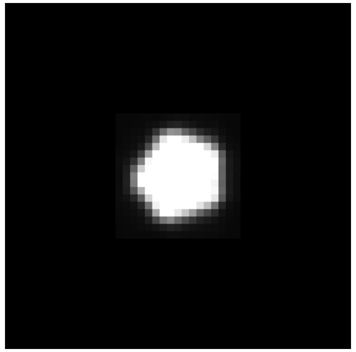
Depth determines scale of blur kernel



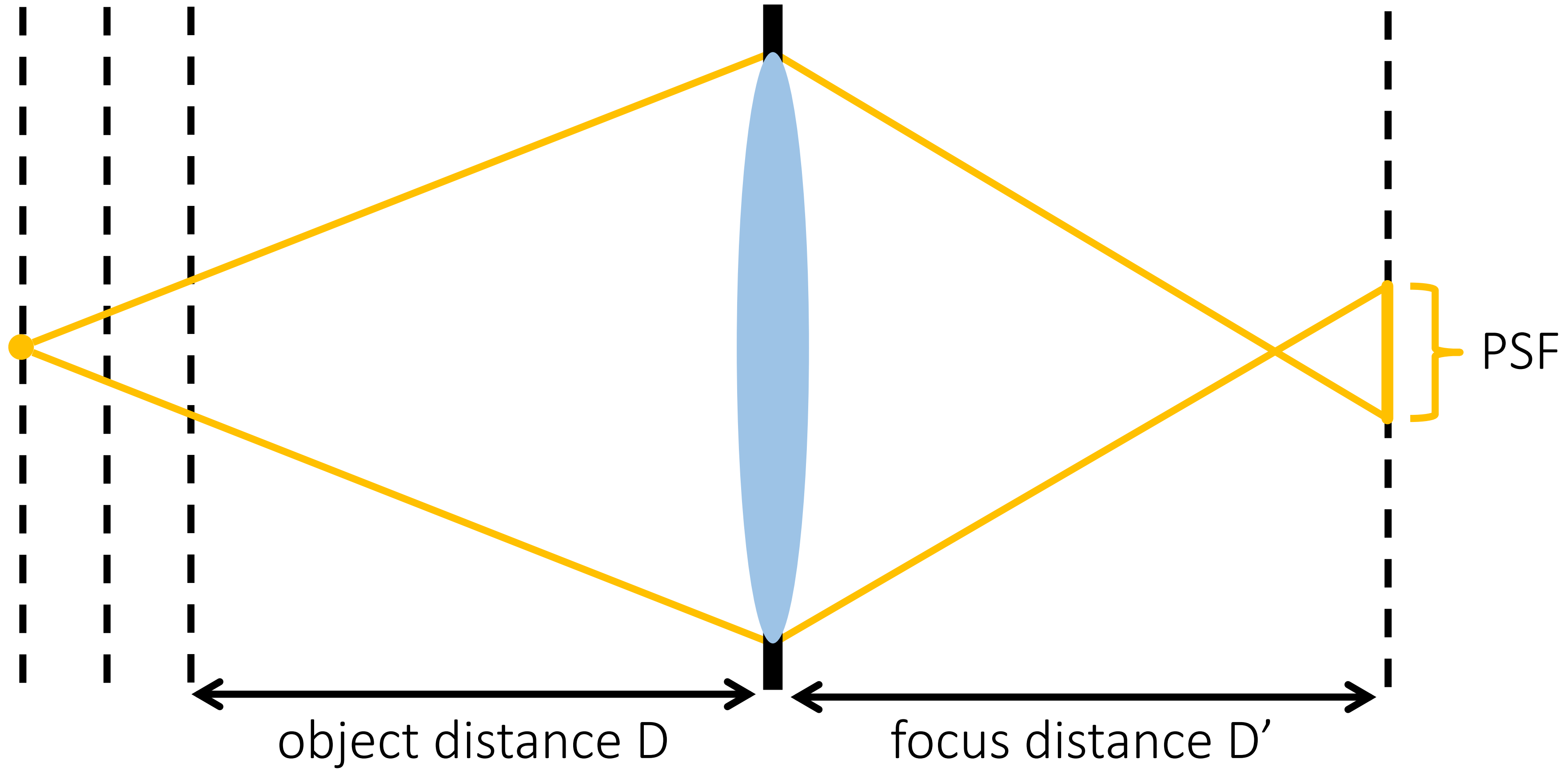


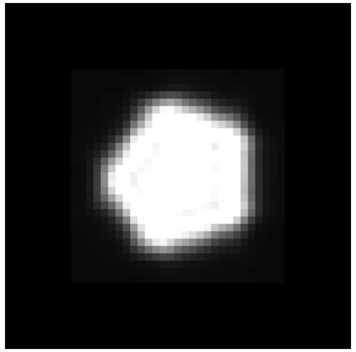
Depth determines scale of blur kernel



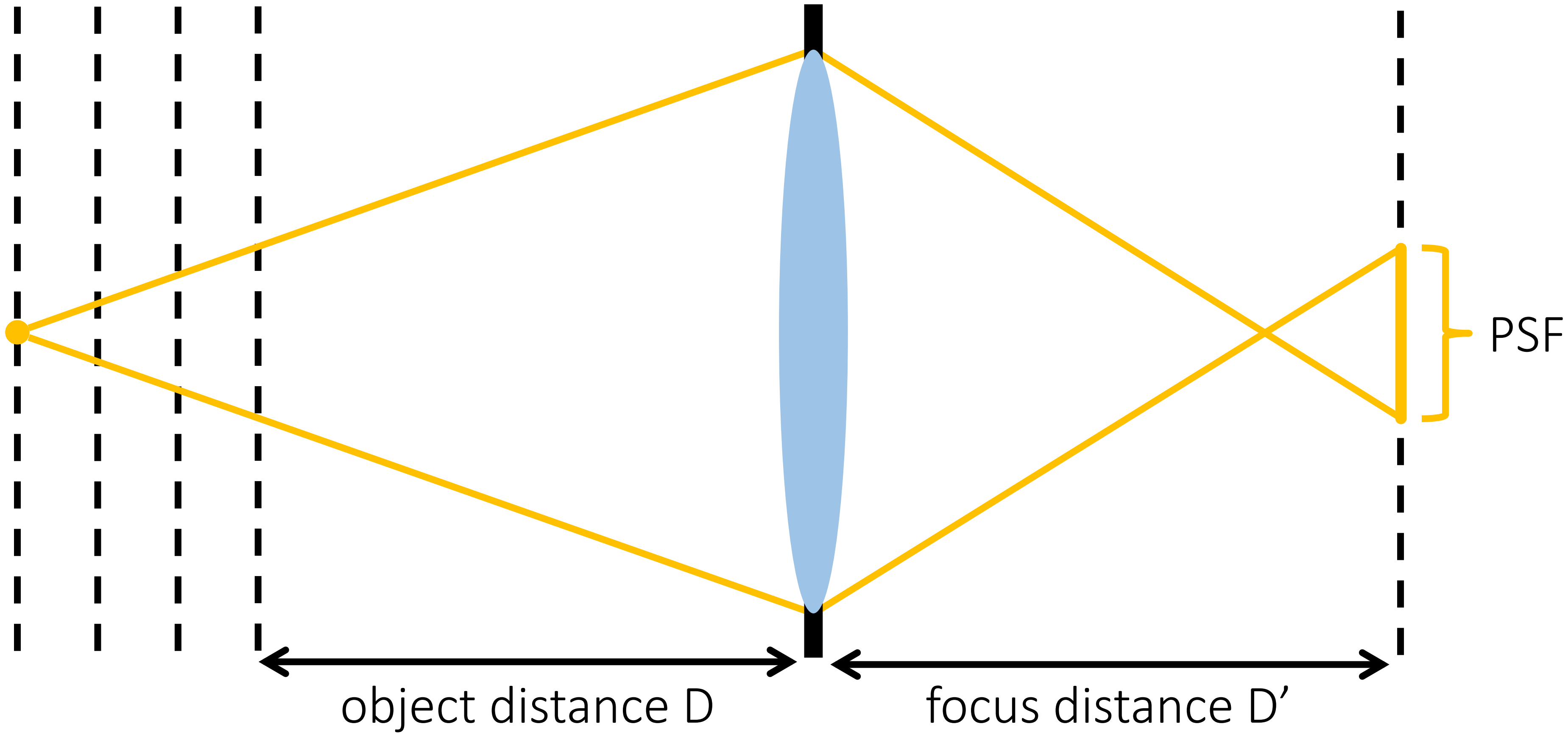


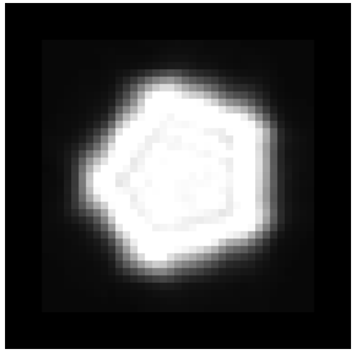
Depth determines scale of blur kernel



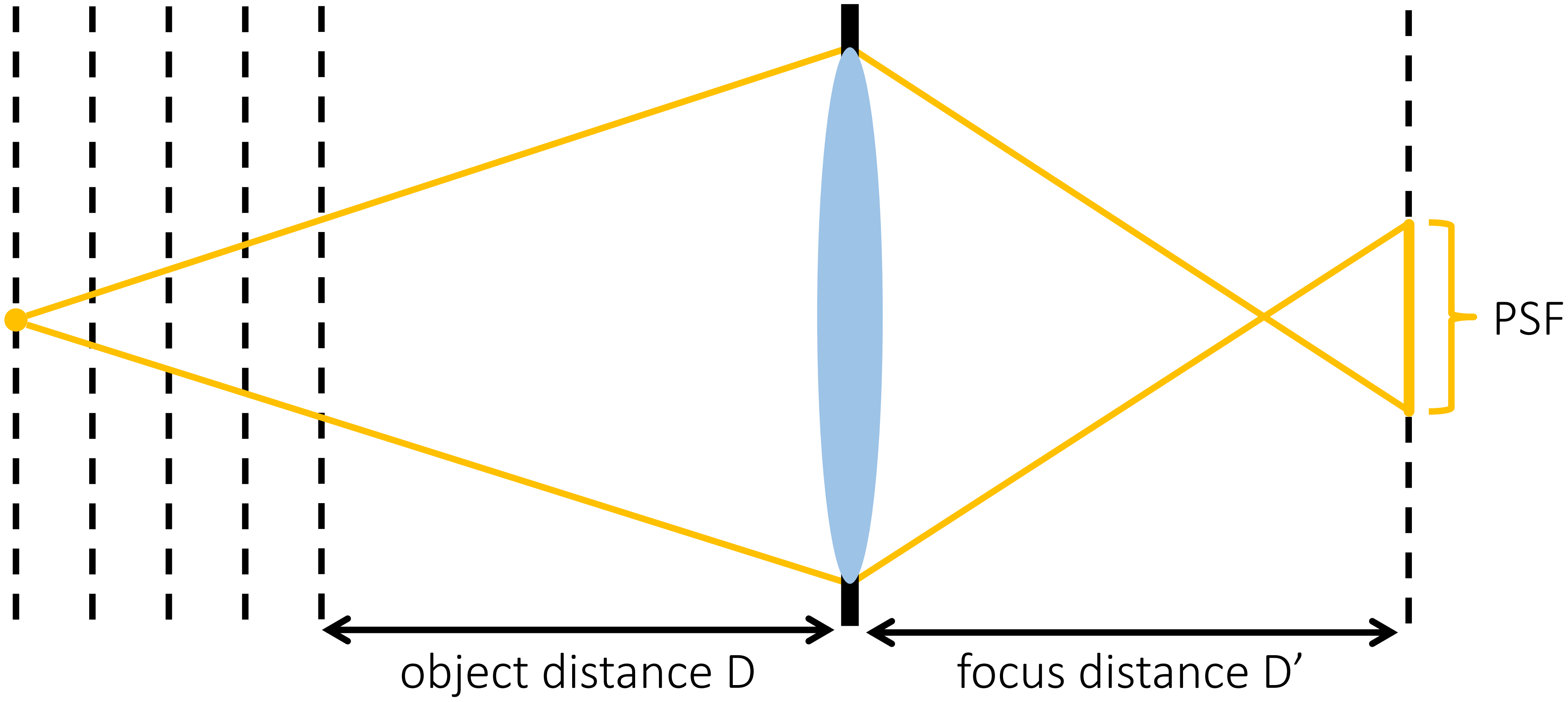


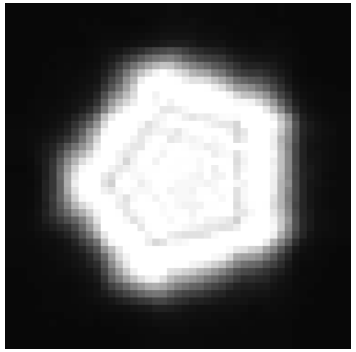
Depth determines scale of blur kernel



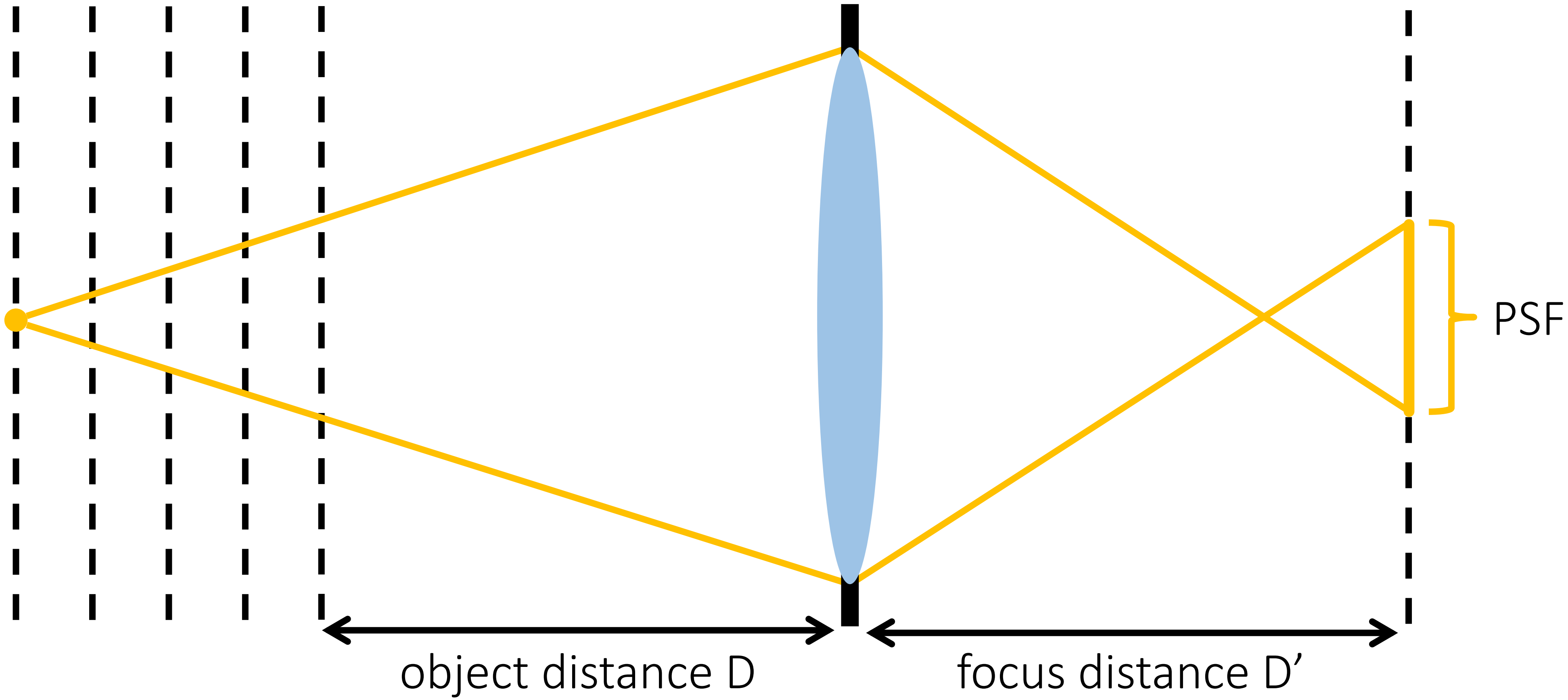


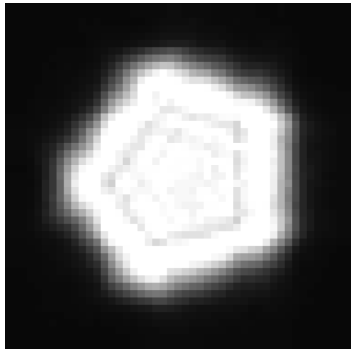
Depth determines scale of blur kernel





Aperture determines shape of blur kernel

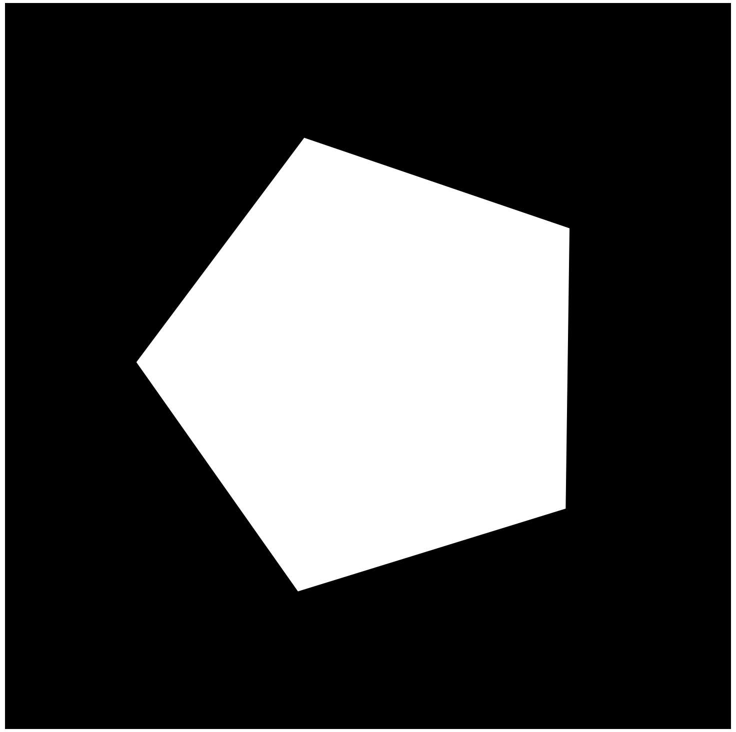




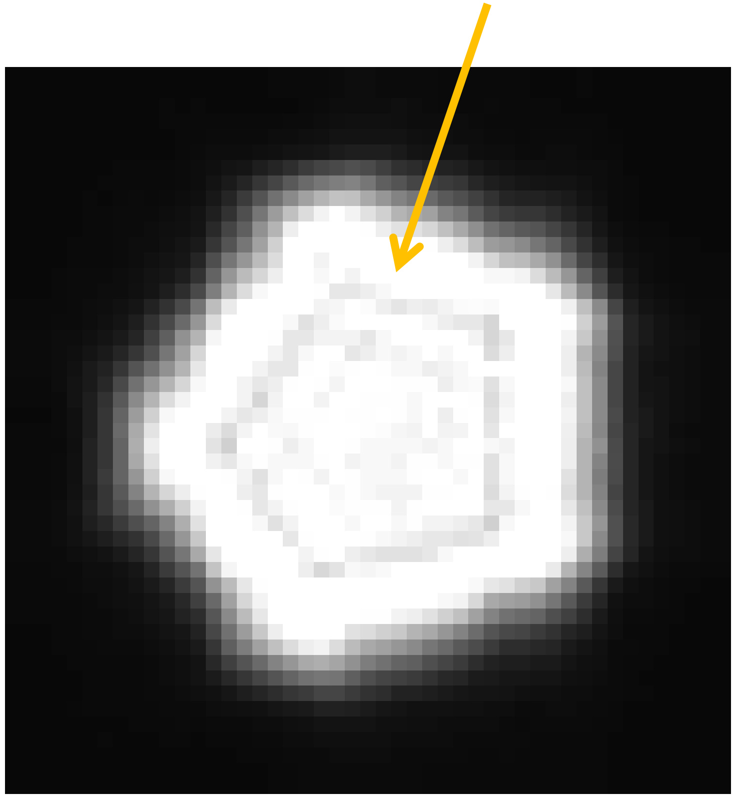
Aperture determines shape of blur kernel

What causes these lines?

photo of aperture



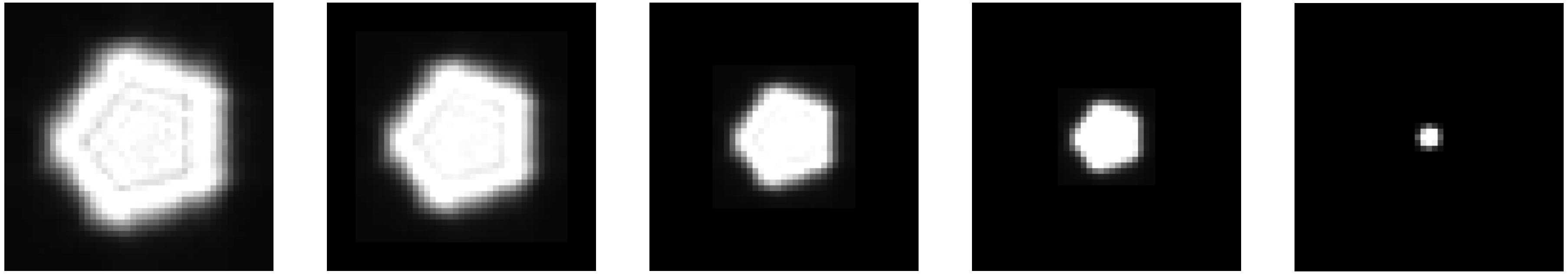
shape of aperture
(optical transfer function, OTF)



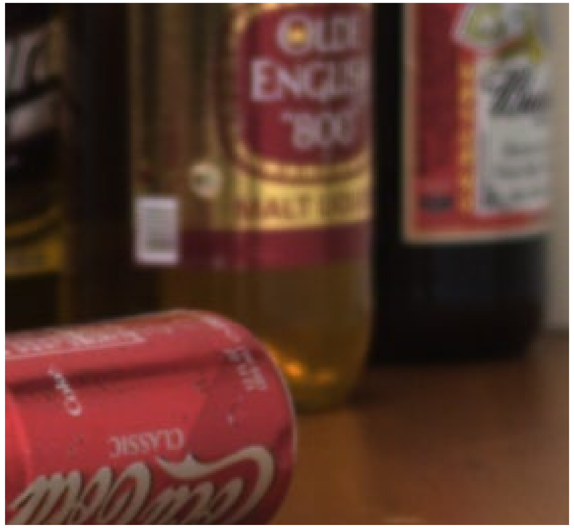
blur kernel
(point spread function, PSF)

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?

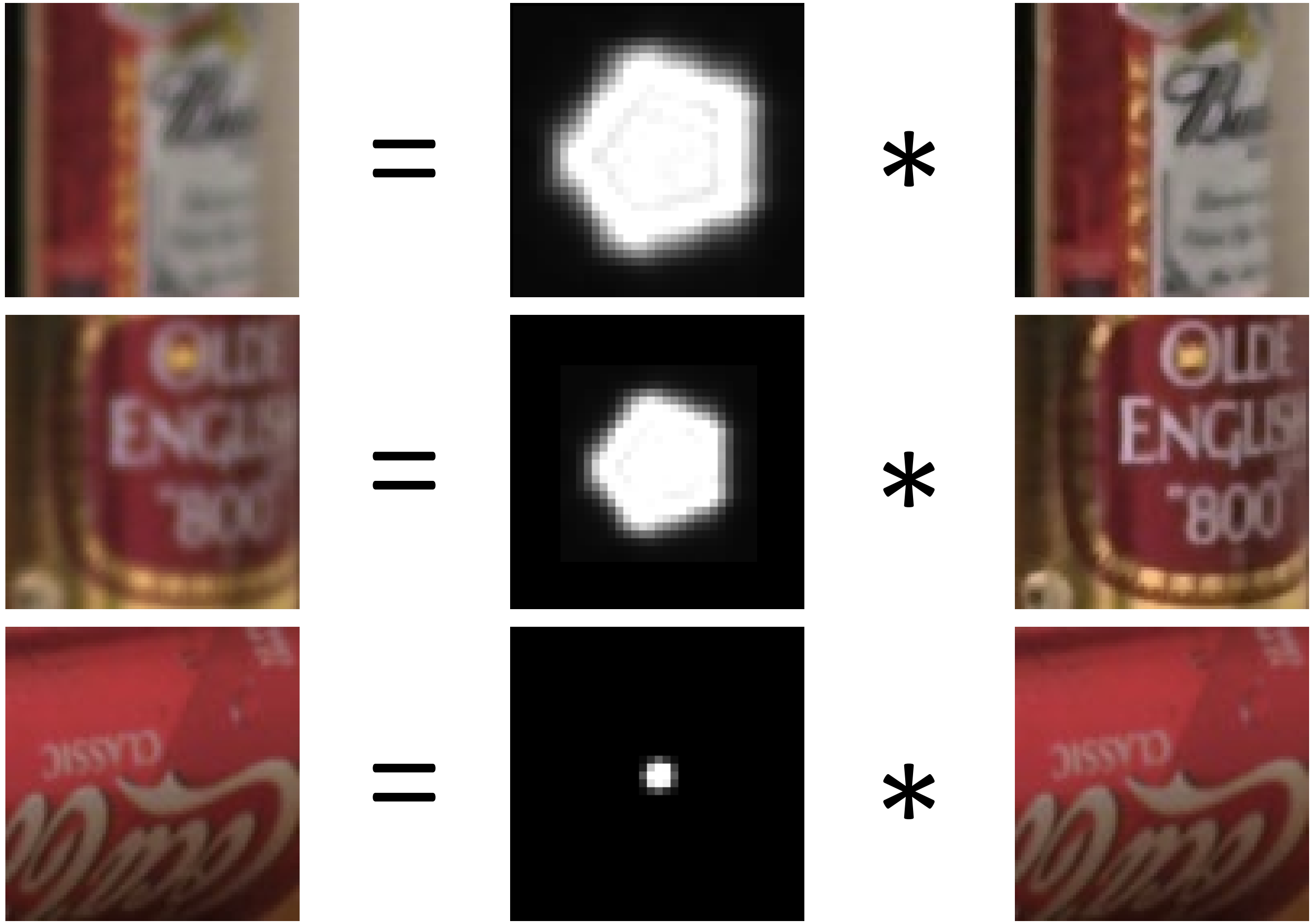
Removing depth defocus

Defocus is *local* convolution with a depth-dependent kernel

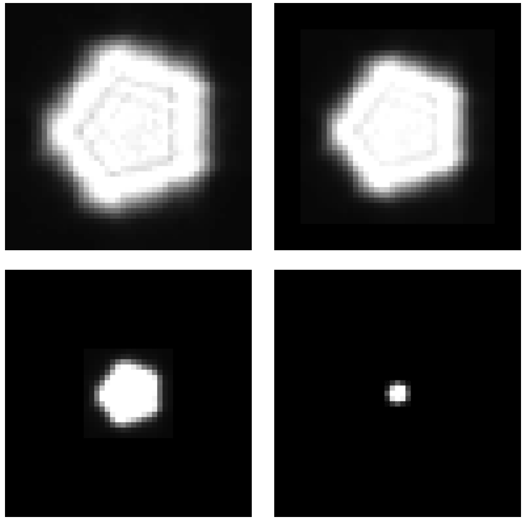
depth 3

depth 2

depth 1



input defocused image

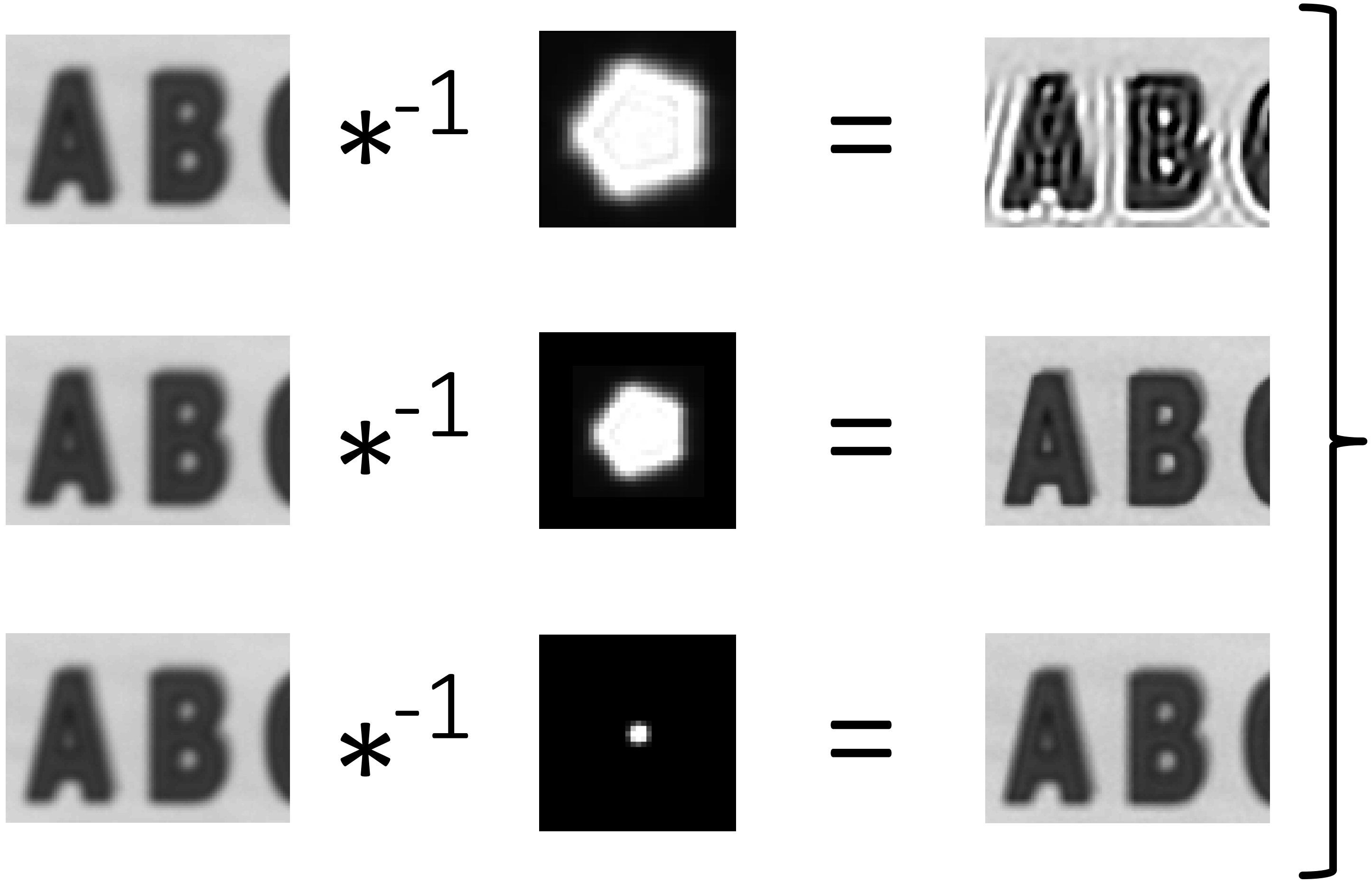


measured PSFs at different depths

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

Removing depth defocus

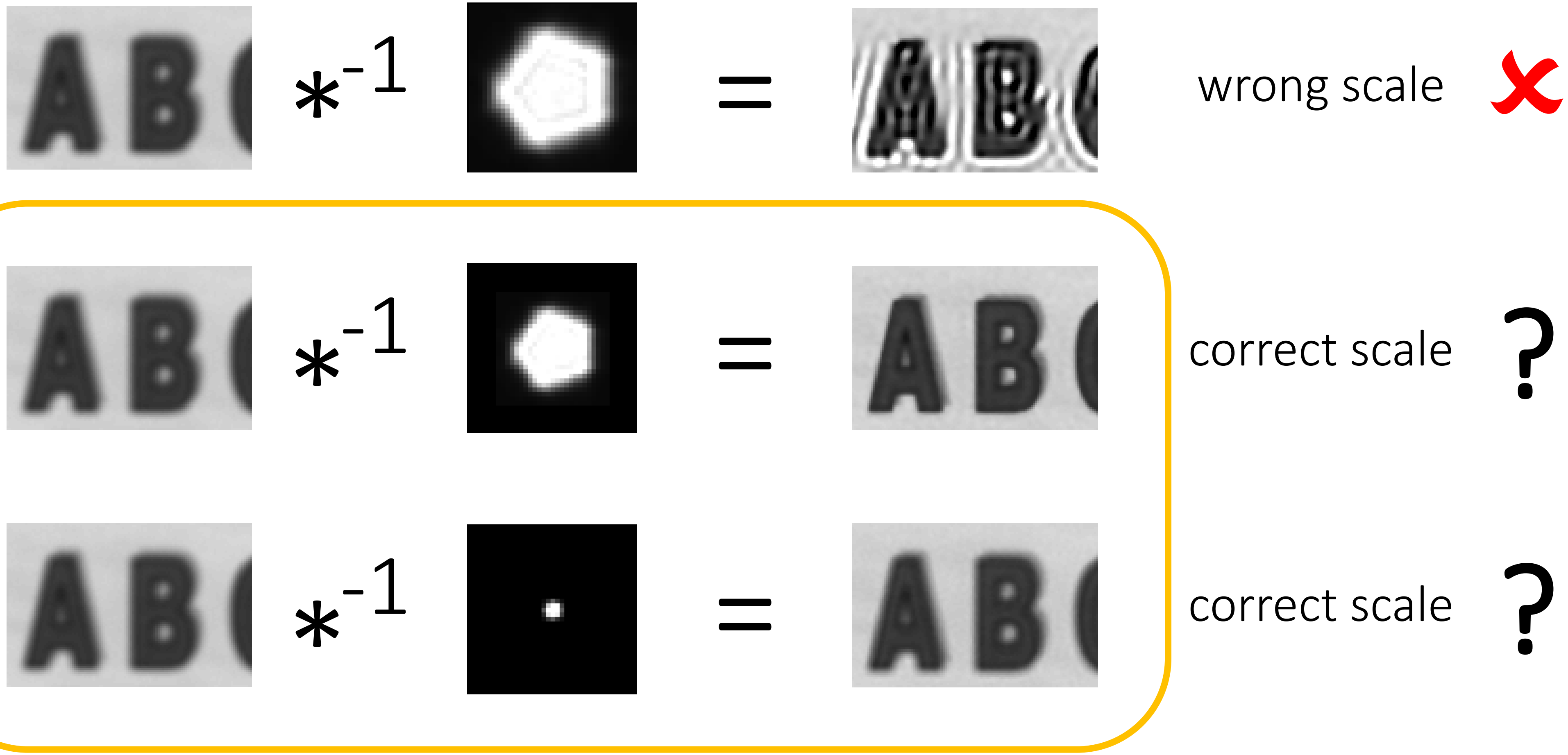
- Deconvolve each image patch with all kernels
- Select the right scale by evaluating the deconvolution results



How do we select the correct scale?

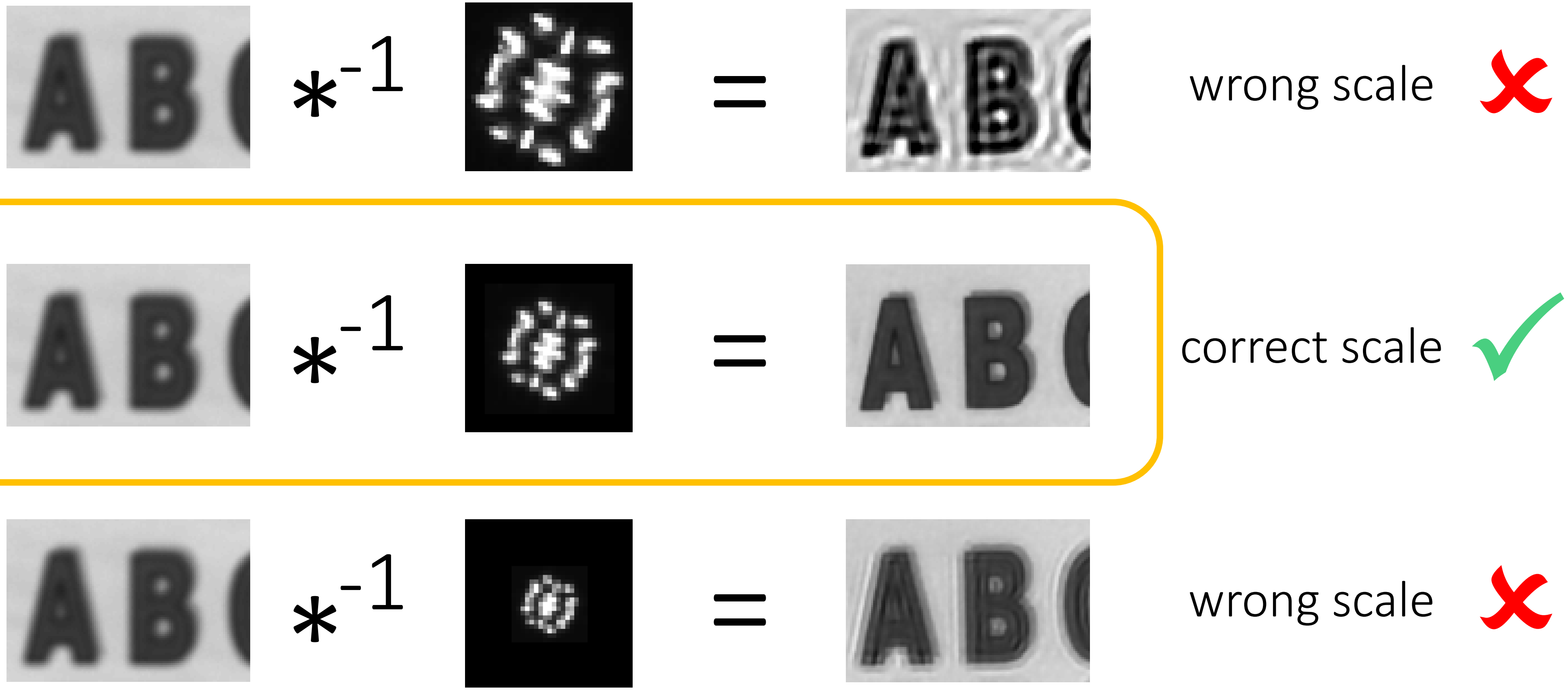
Removing depth defocus

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



Coded aperture

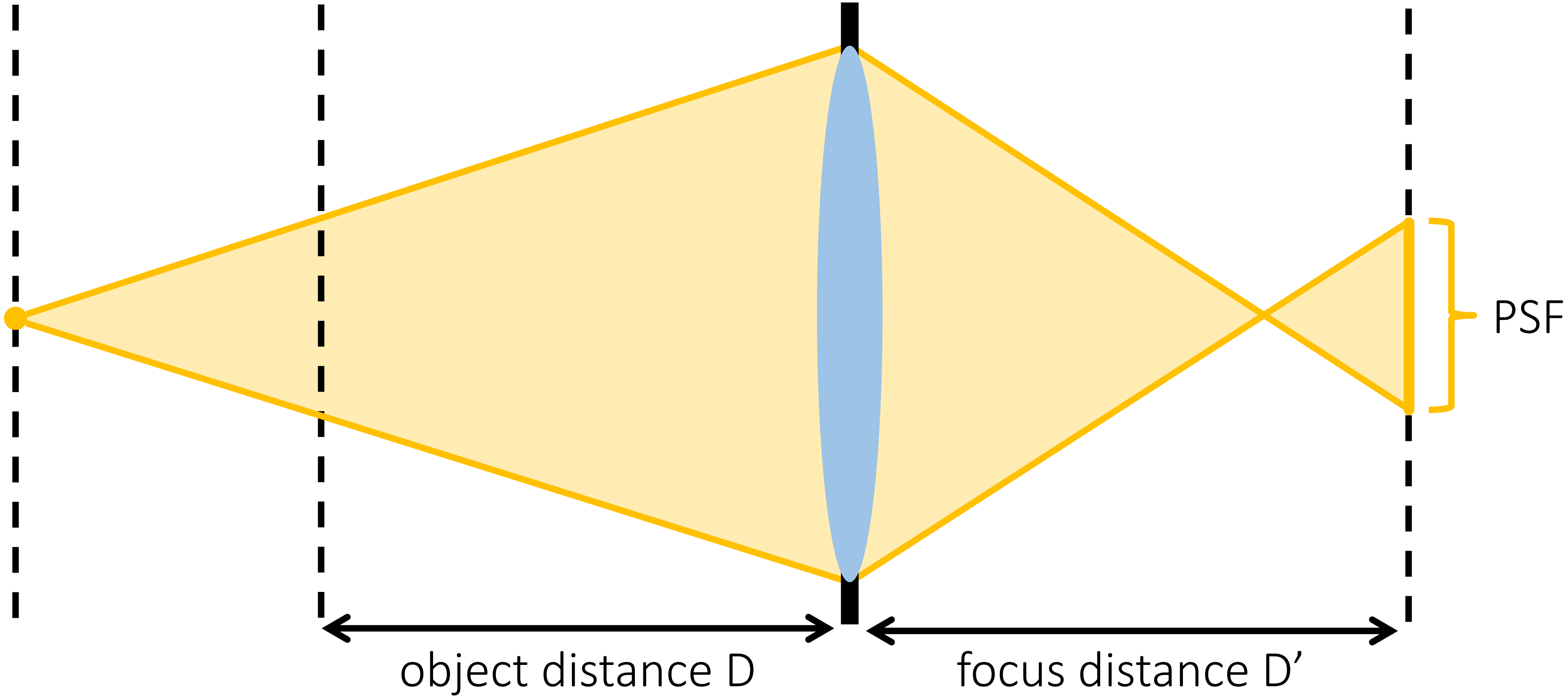
Solution: Change aperture so that it is easier to pick the correct scale

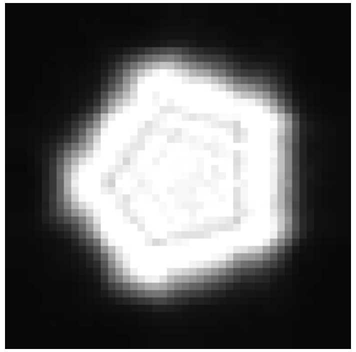


Build your own coded aperture

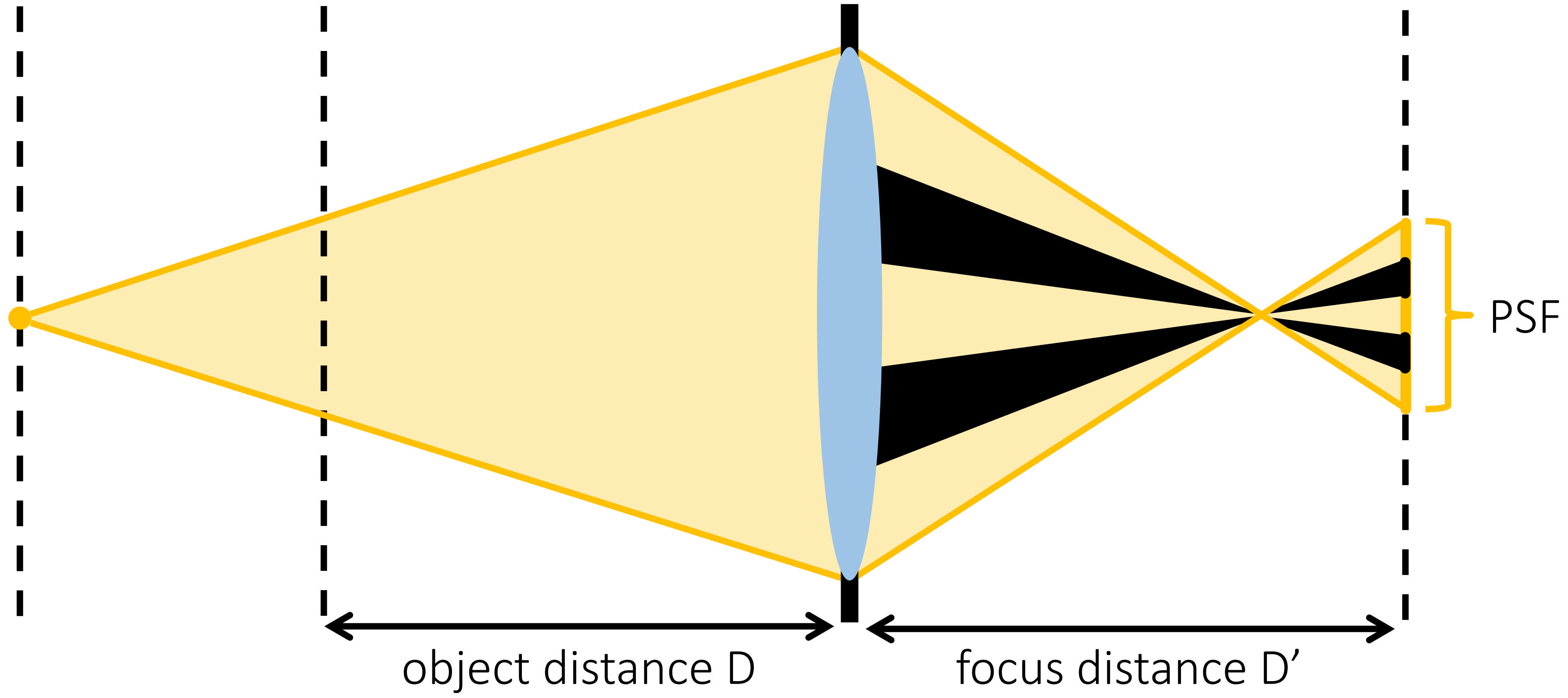
Voila!

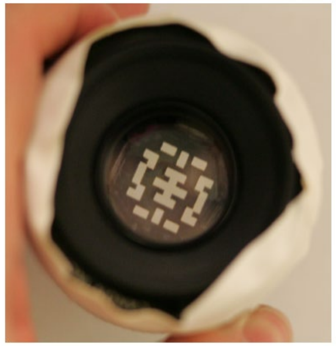
Coded aperture changes shape of kernel

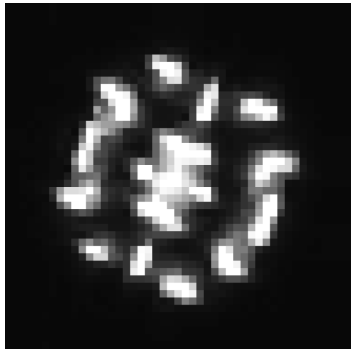




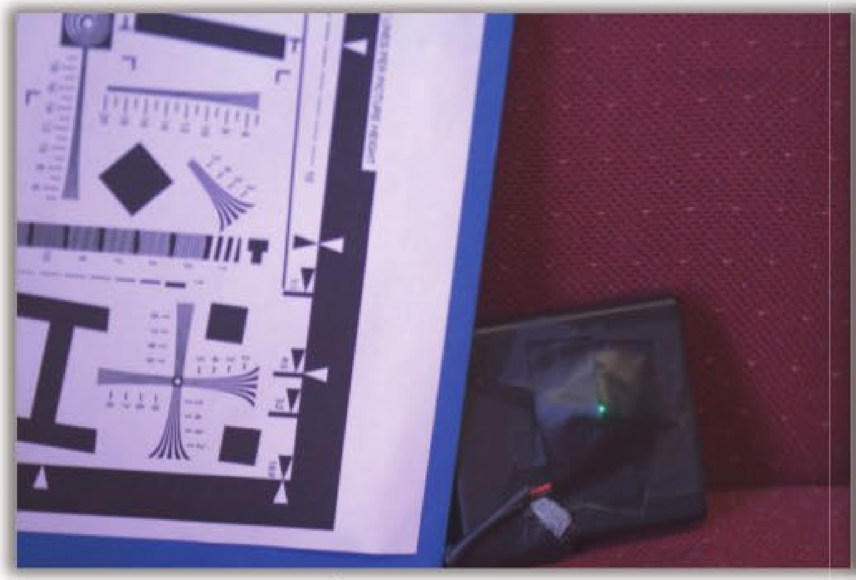
Coded aperture changes shape of kernel



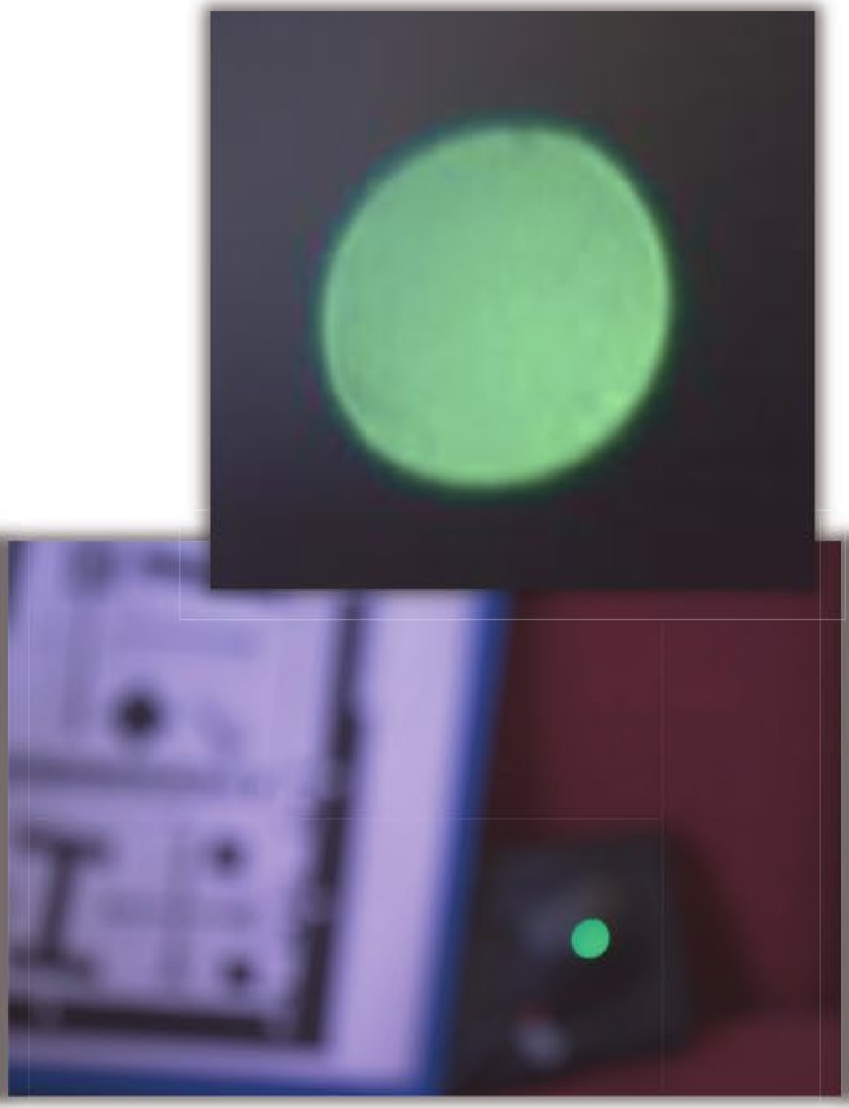




Coded aperture changes shape of PSF



in-focus photo

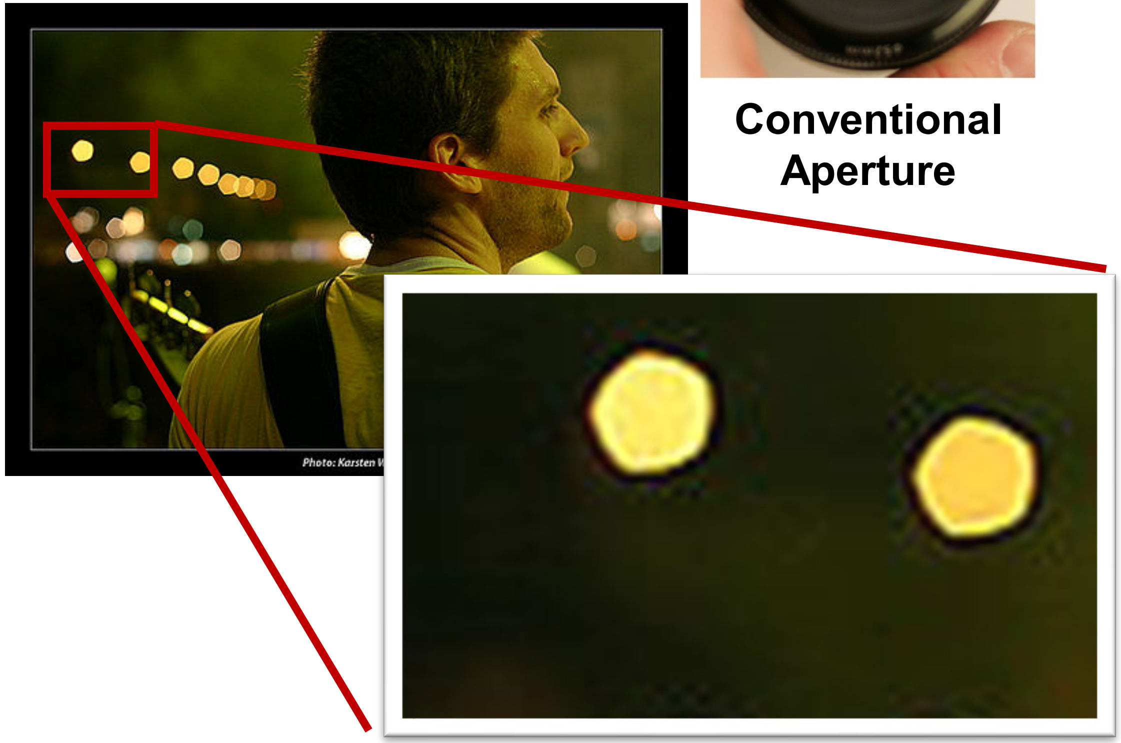


out-of-focus, circular aperture

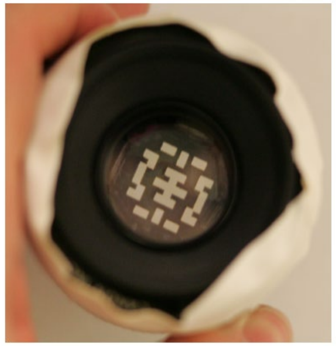
out-of-focus, coded aperture

Image of a point light source

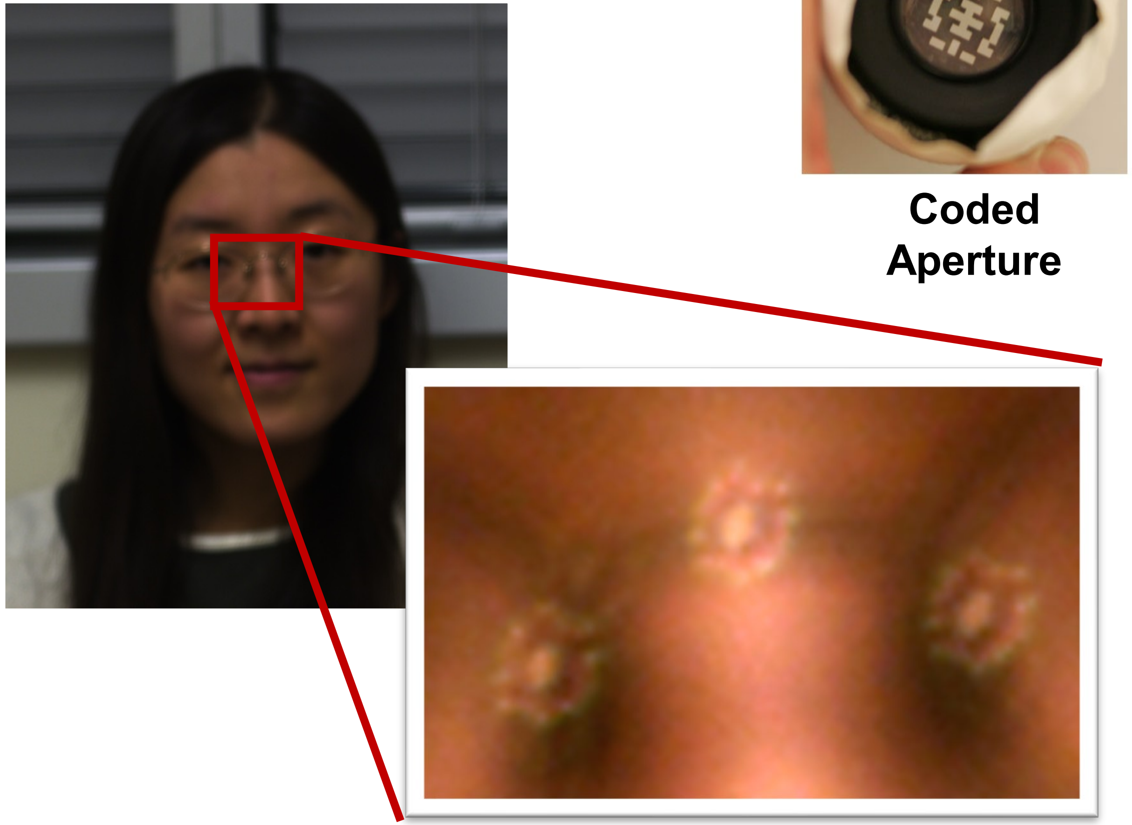
Conventional Aperture



Captured Image



Coded Aperture

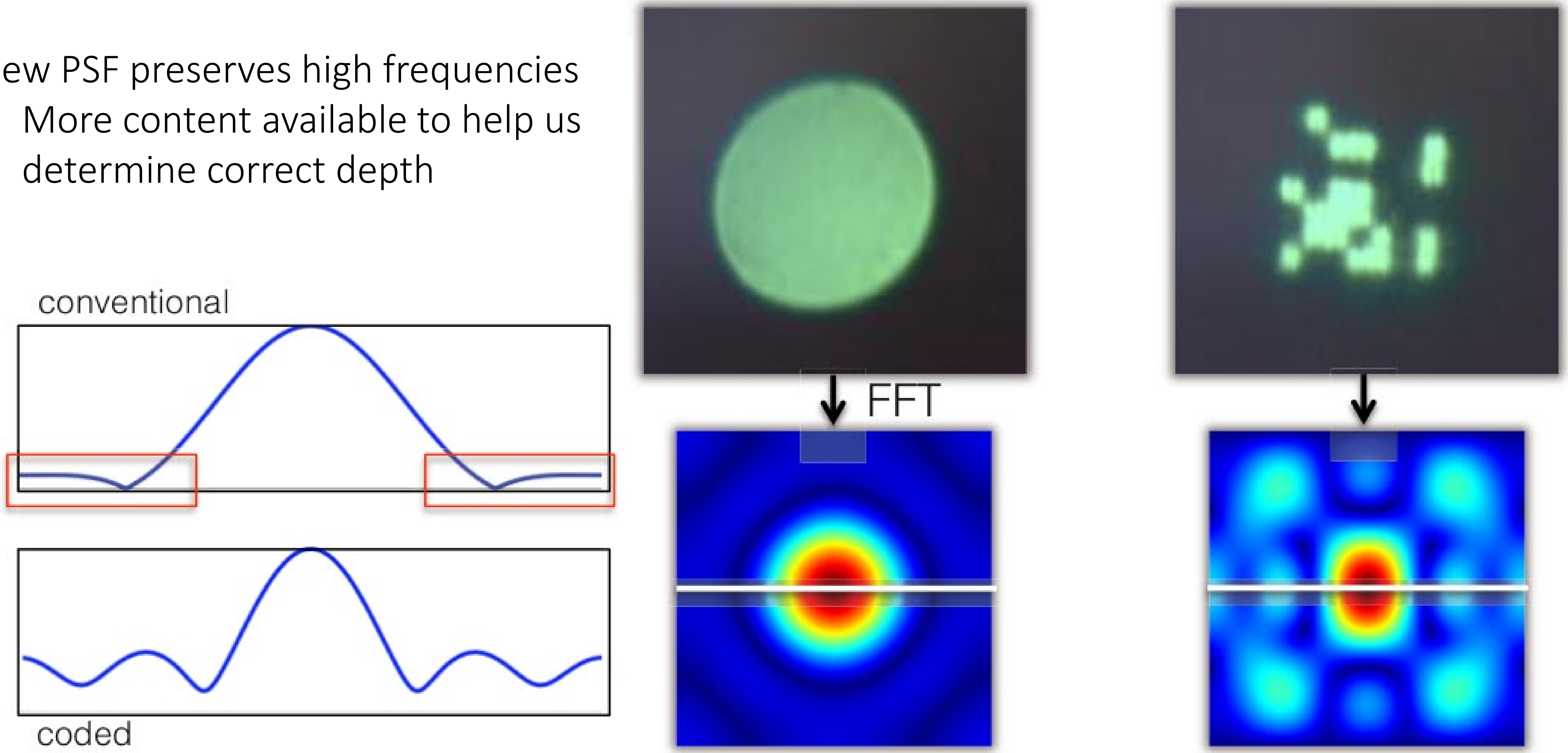


Captured Image

Coded aperture changes shape of PSF

New PSF preserves high frequencies

- More content available to help us determine correct depth

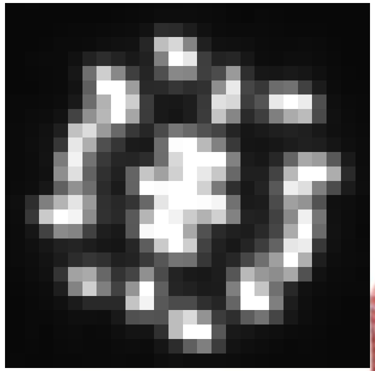


Input

All-focused
(deconvolved)

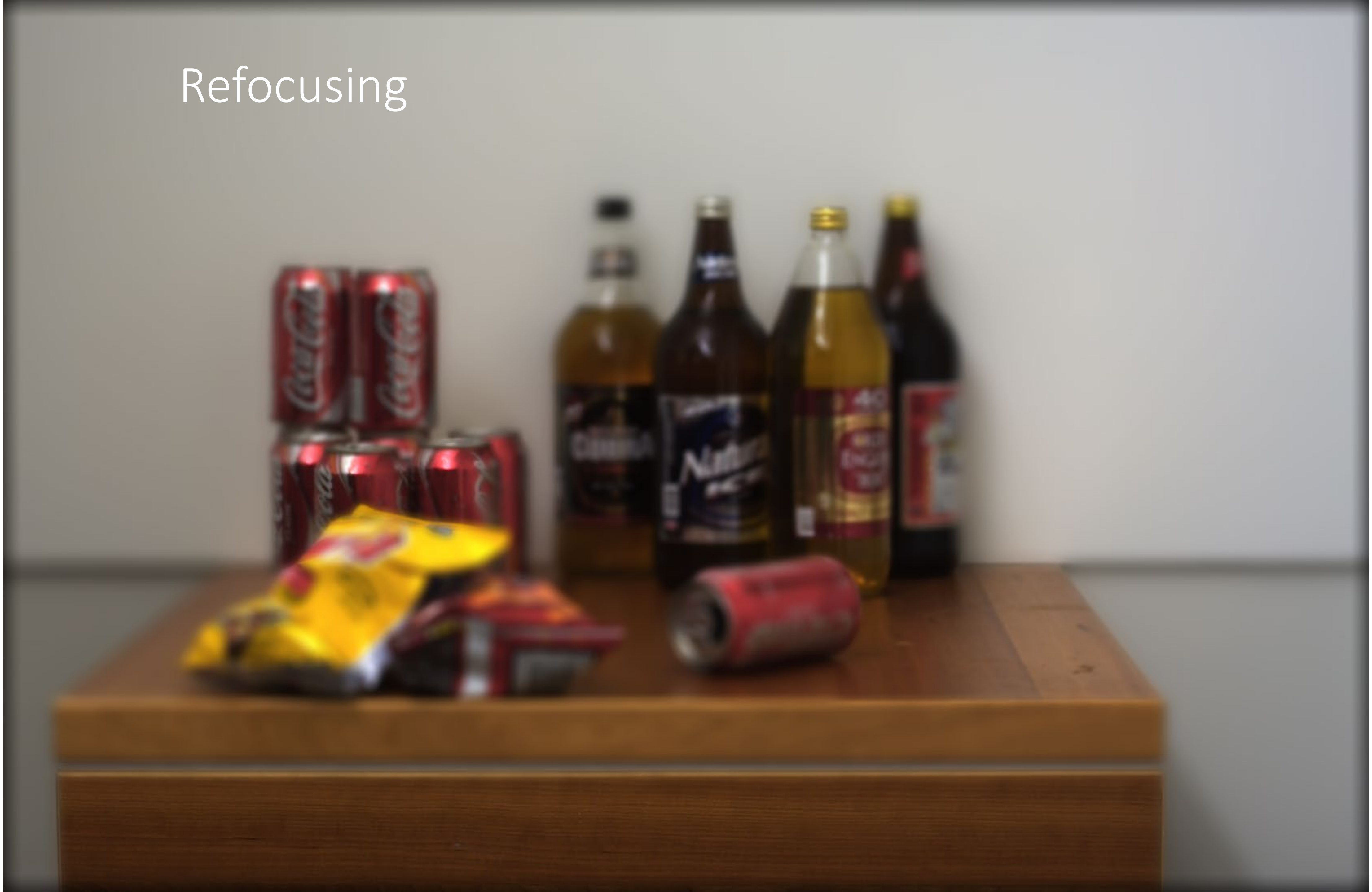
Comparison between standard and coded aperture

Comparison between standard and coded aperture



Refocusing

Refocusing



Refocusing

Depth estimation


Input


All-focused
(deconvolved)

Refocusing

Refocusing

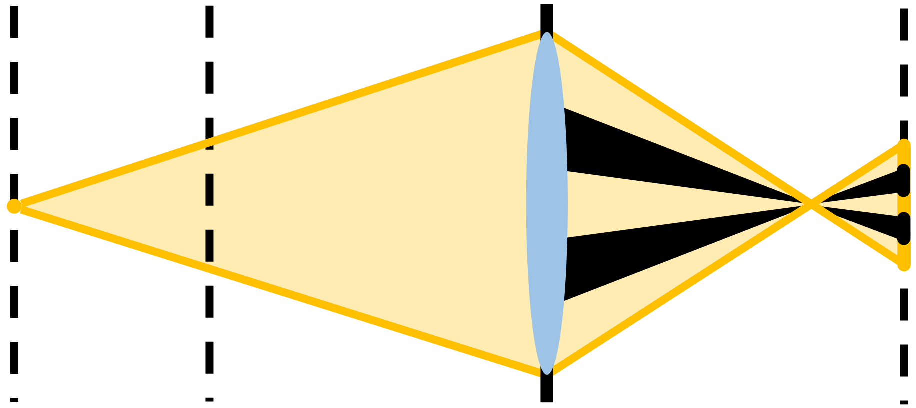
Refocusing

Depth estimation

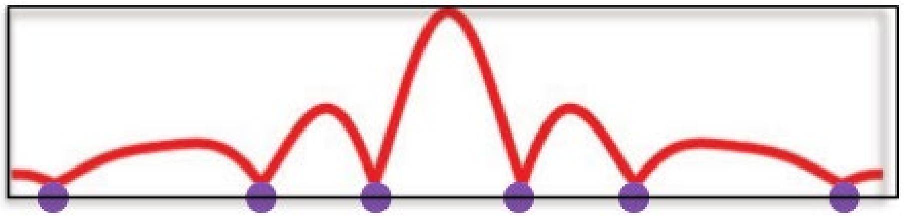
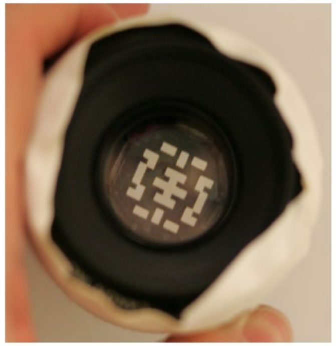
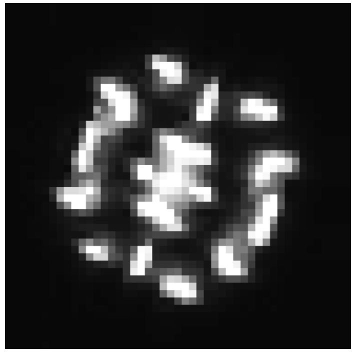
Any problems with using a coded aperture?

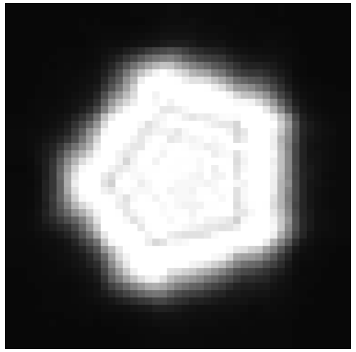
Any problems with using a coded aperture?

- We lose a lot of light due to blocking.



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

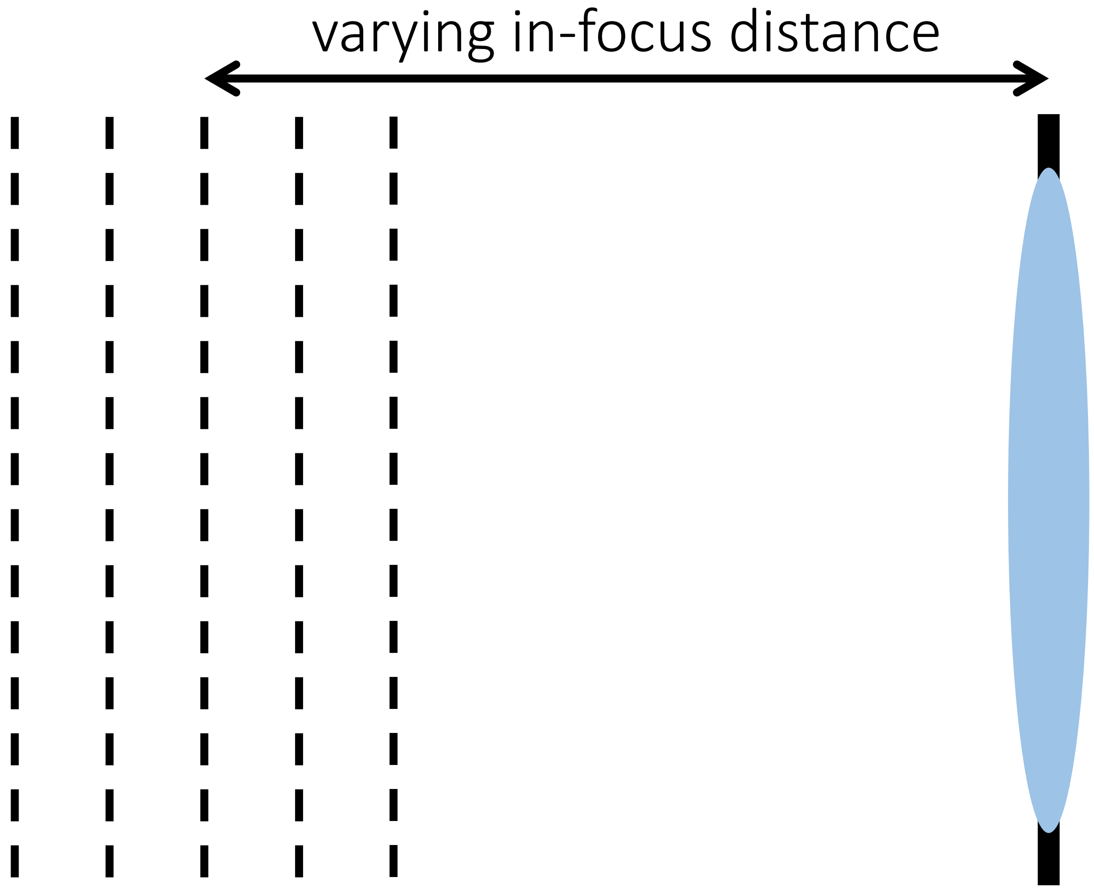






- We still need to select correct scale.

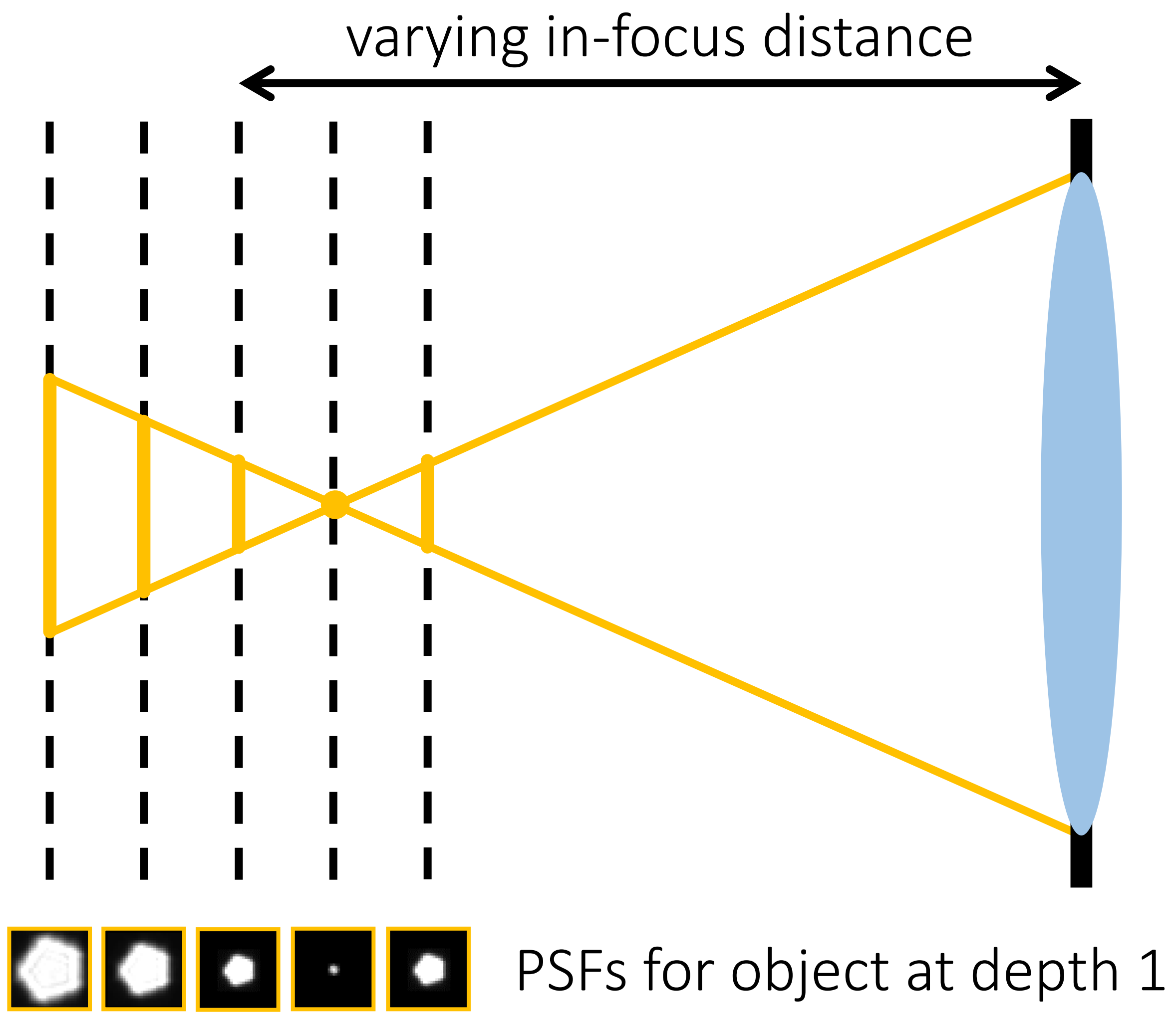
Dealing with depth blur: focal sweep

The difficulty of dealing with depth defocus



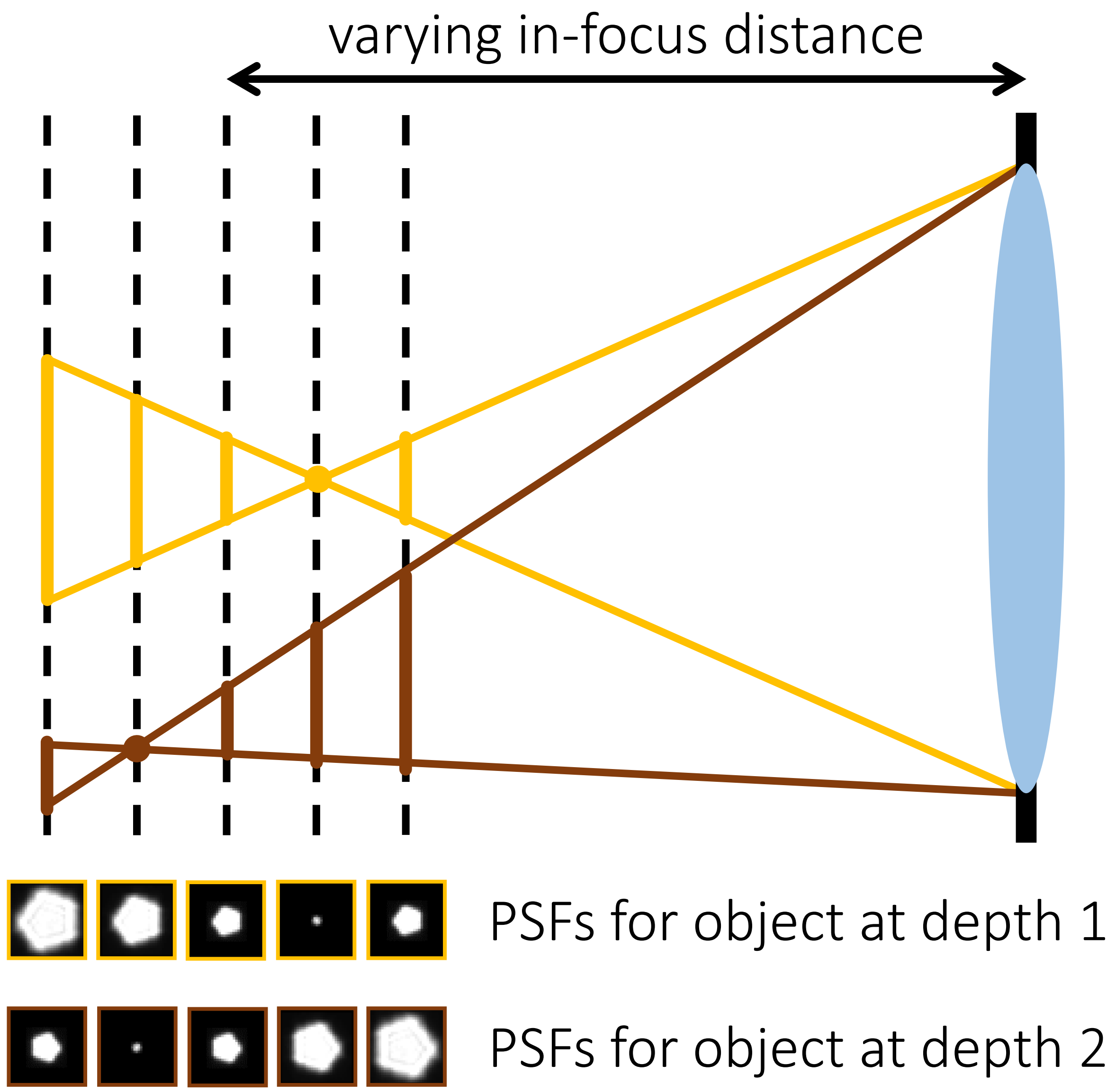
At every focus setting, objects at different depths are blurred by different PSF

The difficulty of dealing with depth defocus



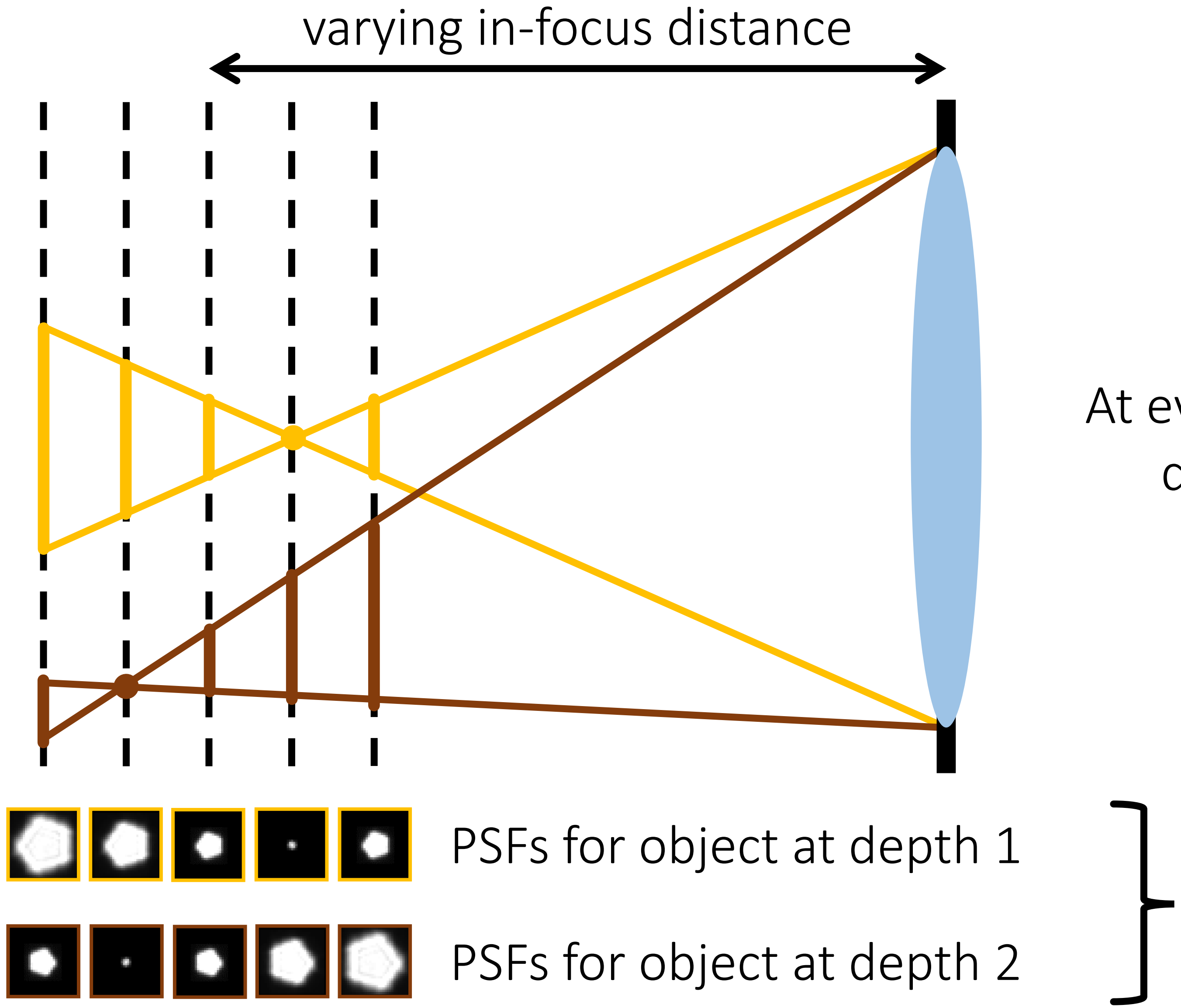
At every focus setting, objects at different depths are blurred by different PSF

The difficulty of dealing with depth defocus



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

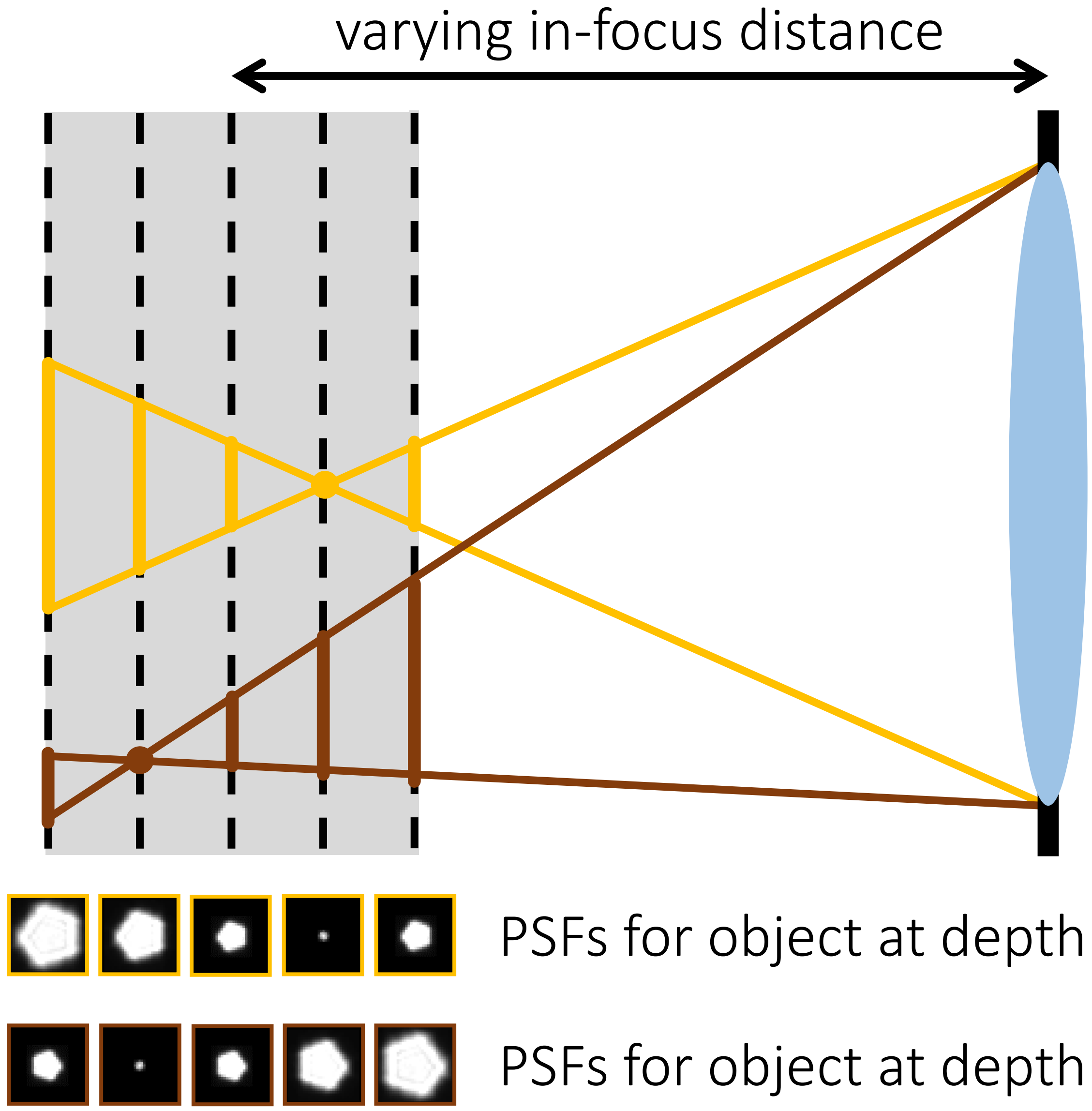
The difficulty of dealing with depth defocus



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

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

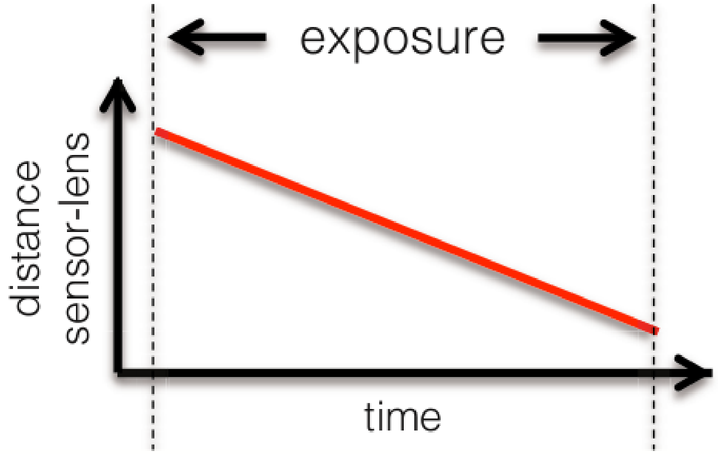
Focal sweep



PSFs for object at depth 1

PSFs for object at depth 2

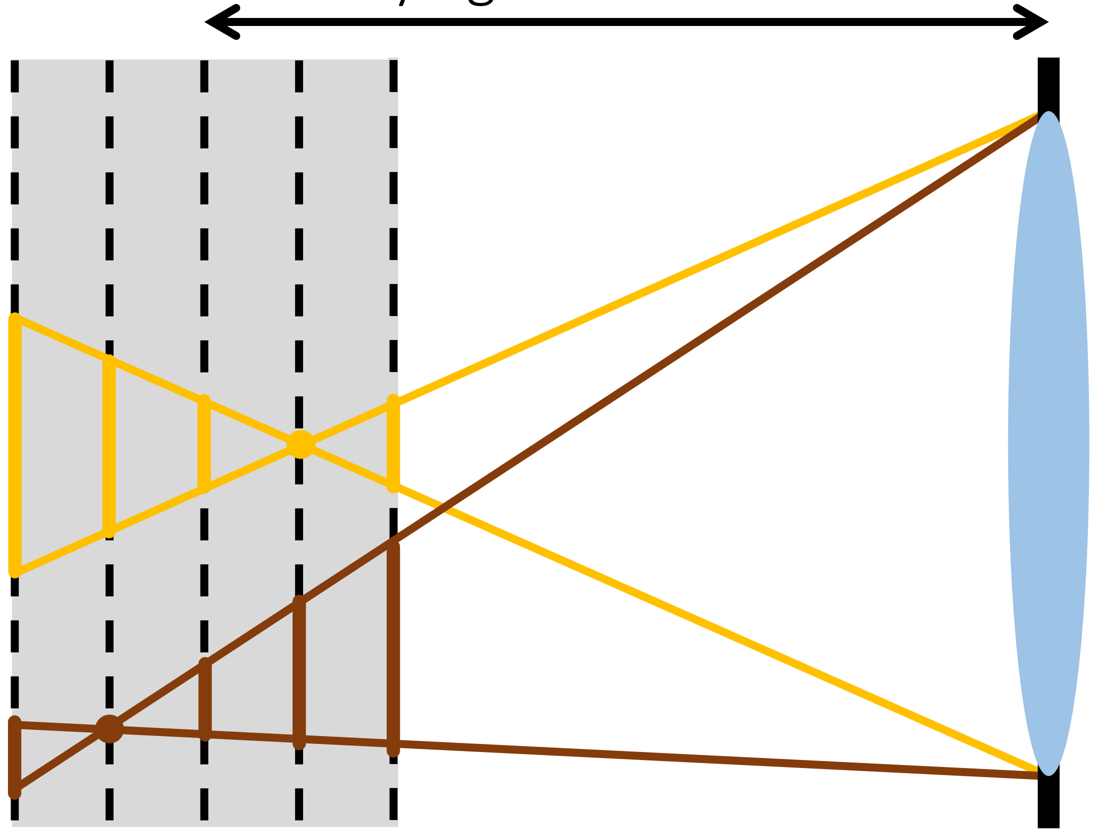
linear motion:



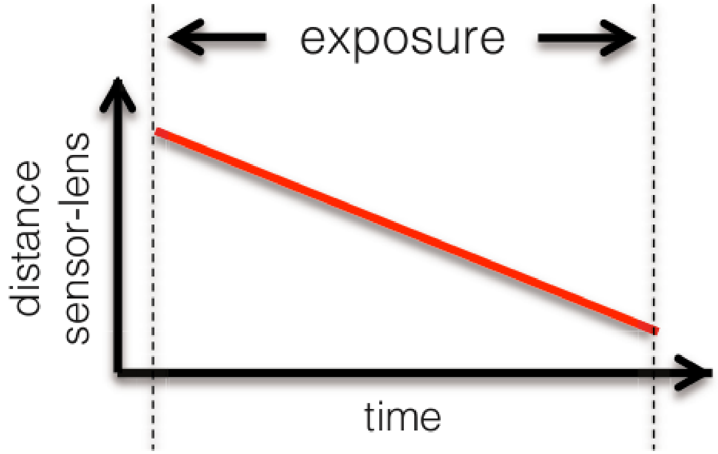
Go through all focus settings
during a single exposure

What is the effective PSF in this case?

Focal sweep



linear motion:



Go through all focus settings
during a single exposure

∫ [PSF images] dt = [PSF image] effective PSF for object at depth 1

∫ [PSF images] dt = [PSF image] effective PSF for object at depth 2

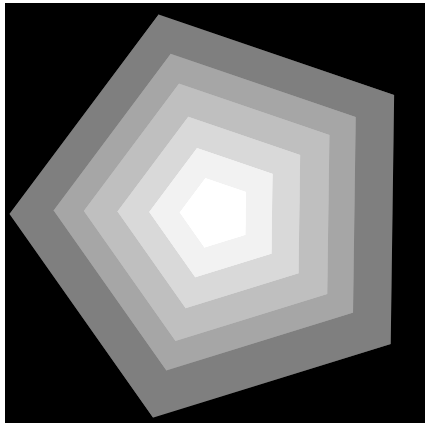
Anything special about these effective PSFs?

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.

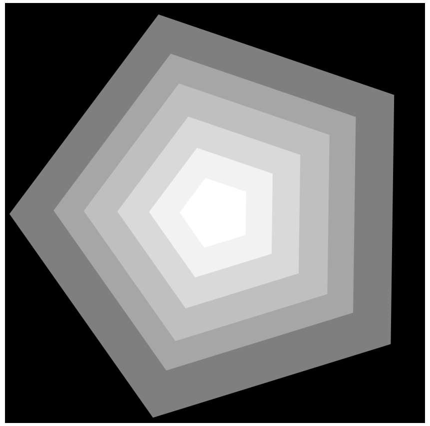
What are the implications of this?



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?

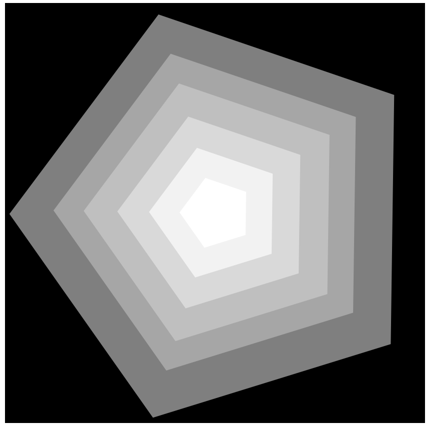
1. The image we capture will not be sharp anywhere; but
2. We can use simple (global) deconvolution to sharpen parts we want

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?

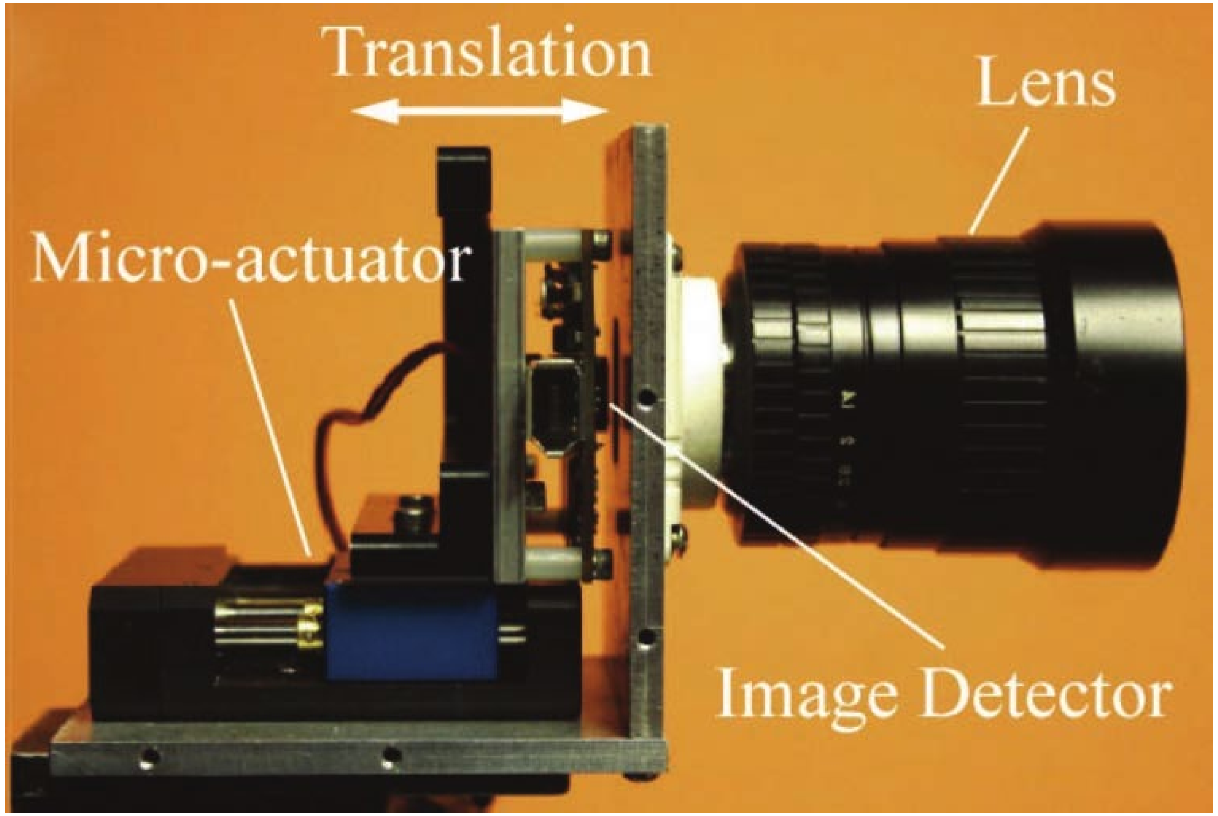
1. The image we capture will not be sharp anywhere; but
2. We can use simple (global) deconvolution to sharpen parts we want

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?

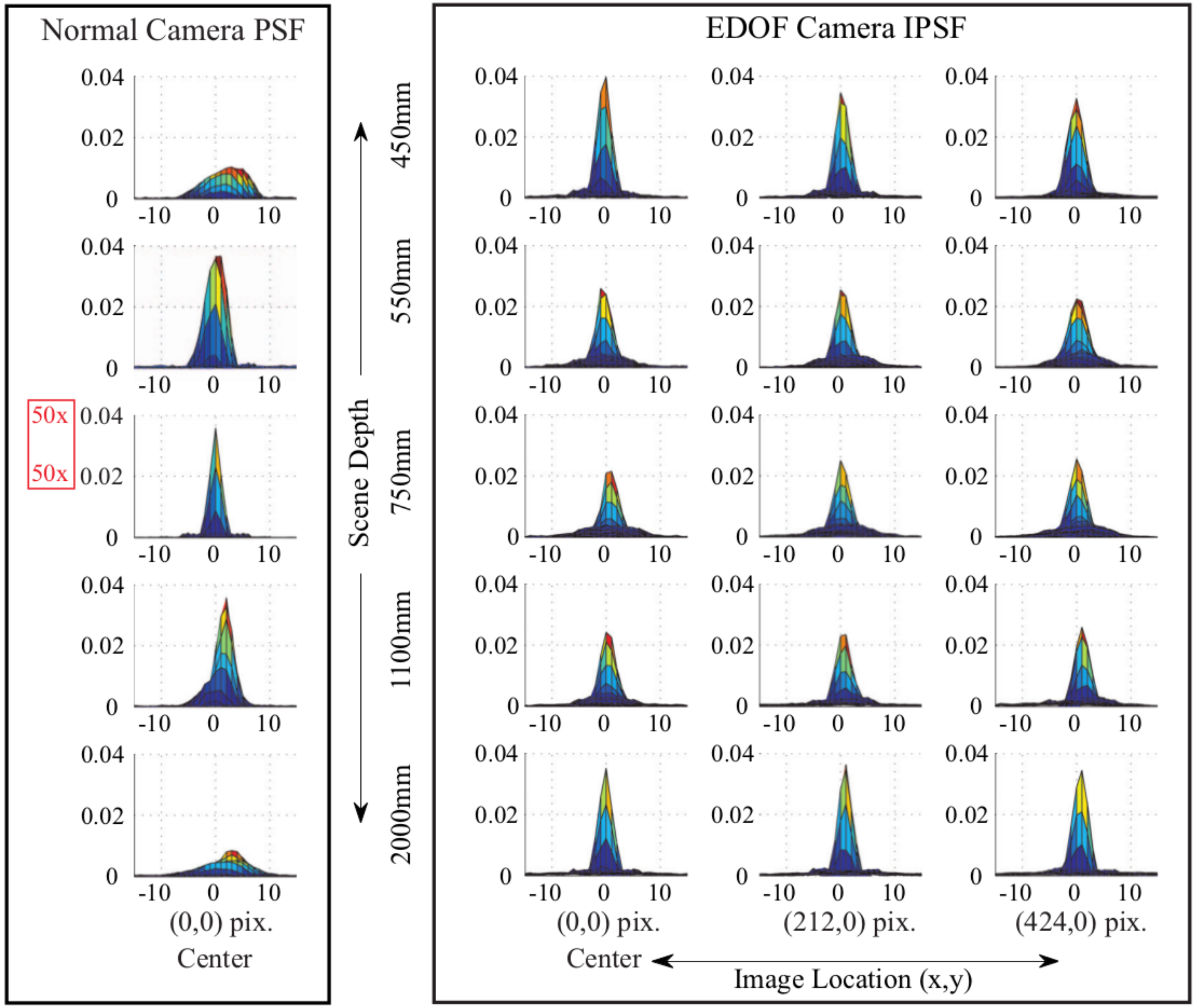
How can you implement focal sweep?



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

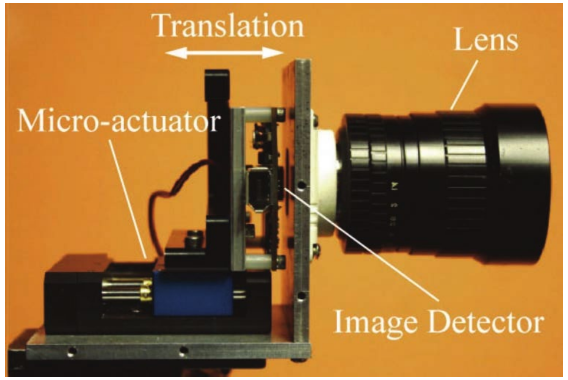
conventional photo (small DOF)

conventional photo (large DOF, noisy)

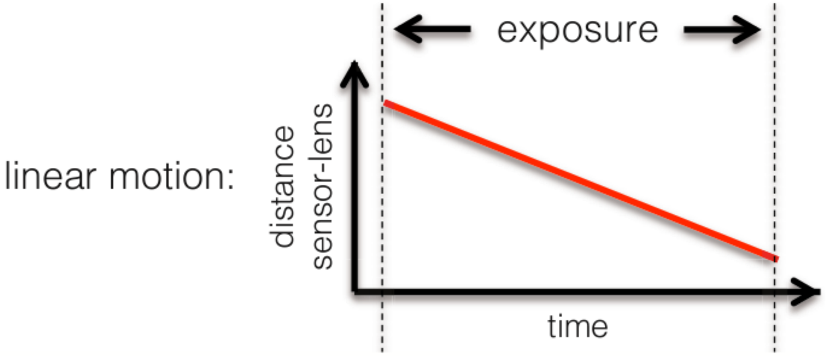
Any problems with using focal sweep?

Any problems with using focal sweep?

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



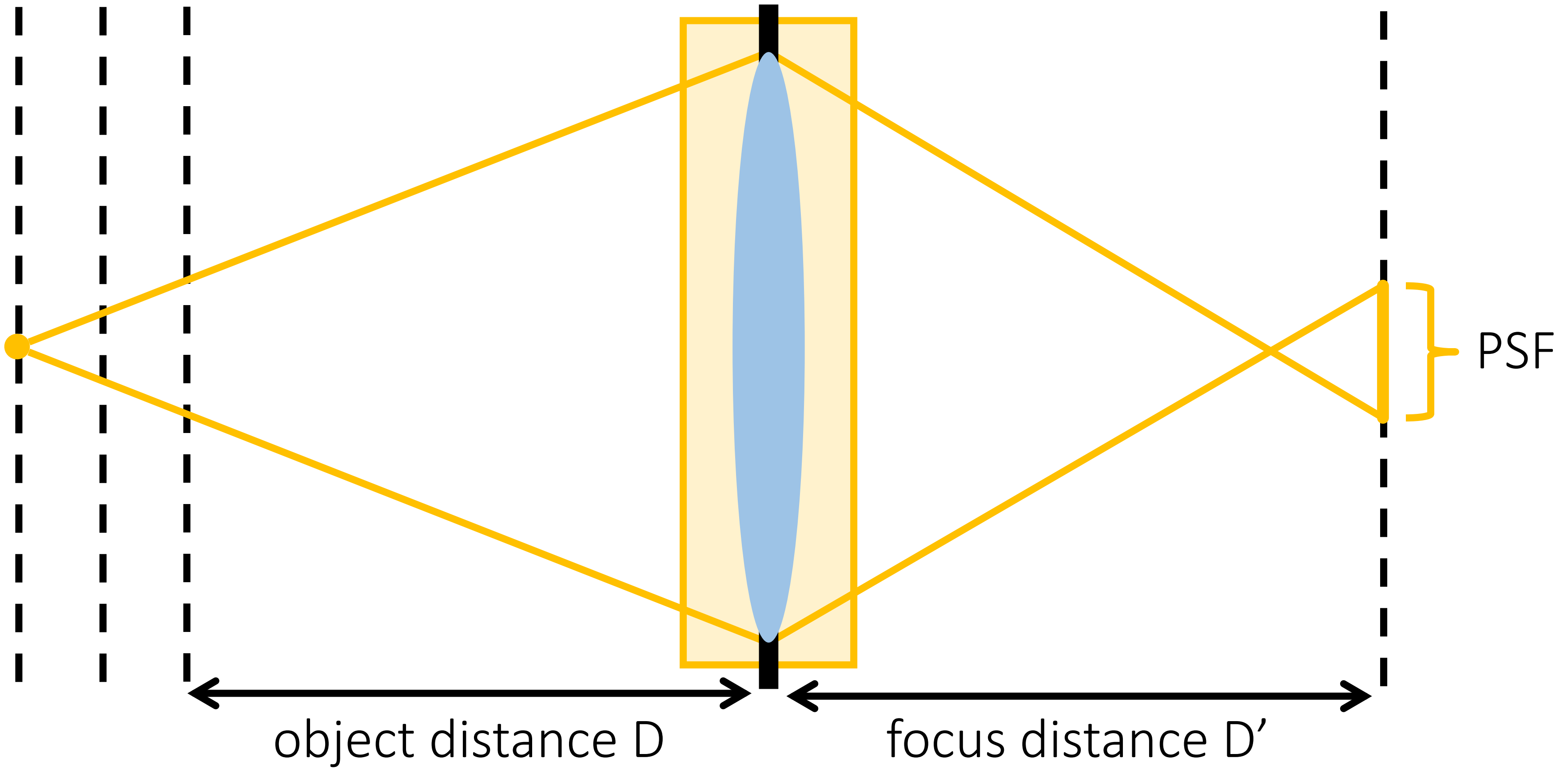
- Perfect depth invariance requires very constant speed.



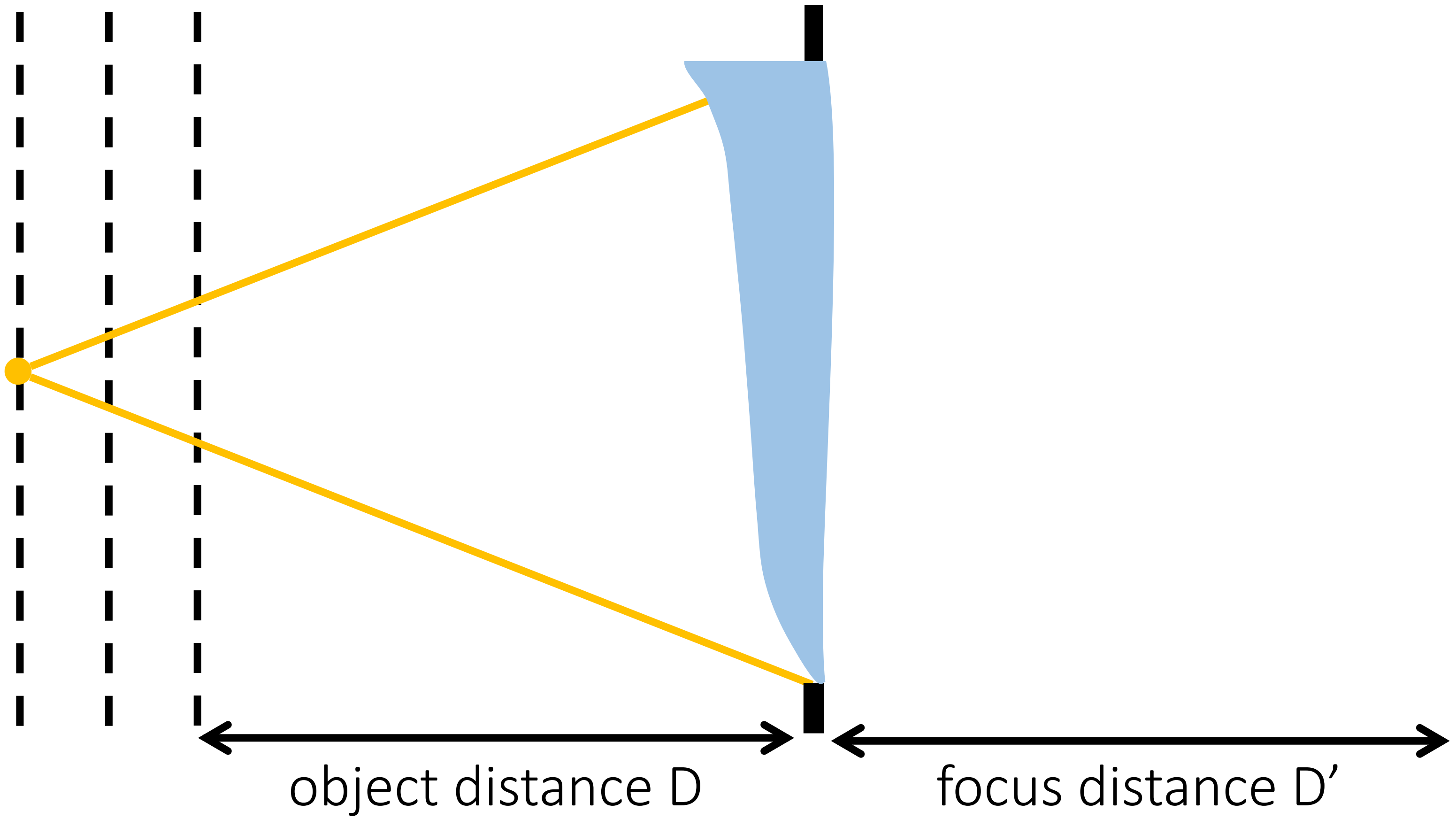
- We lose depth information.

Dealing with depth blur: generalized optics

Change optics, not aperture

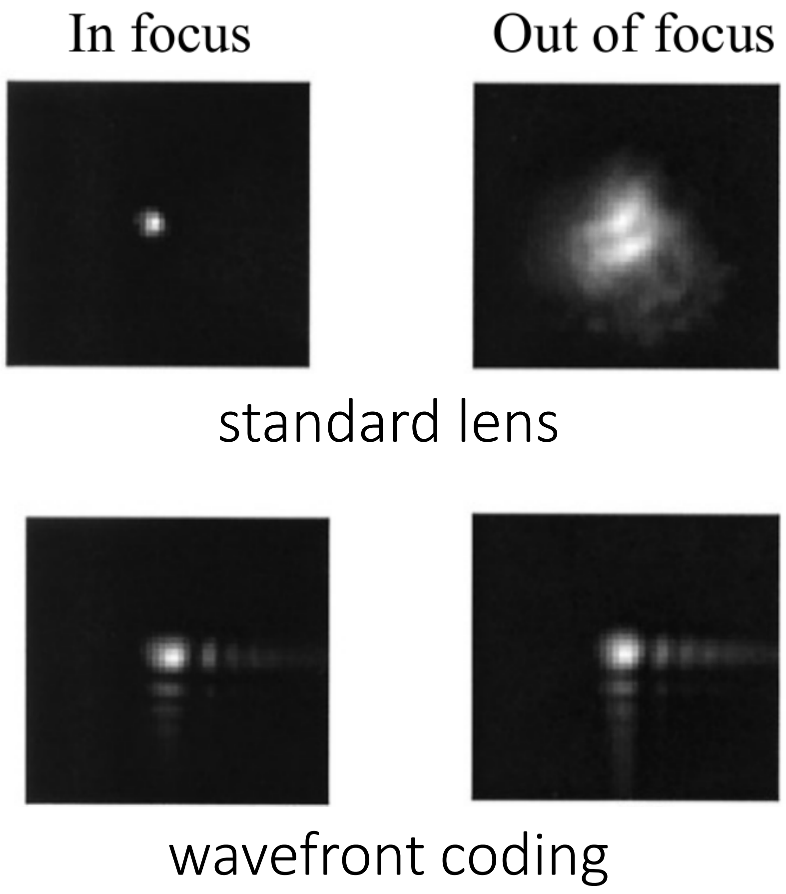


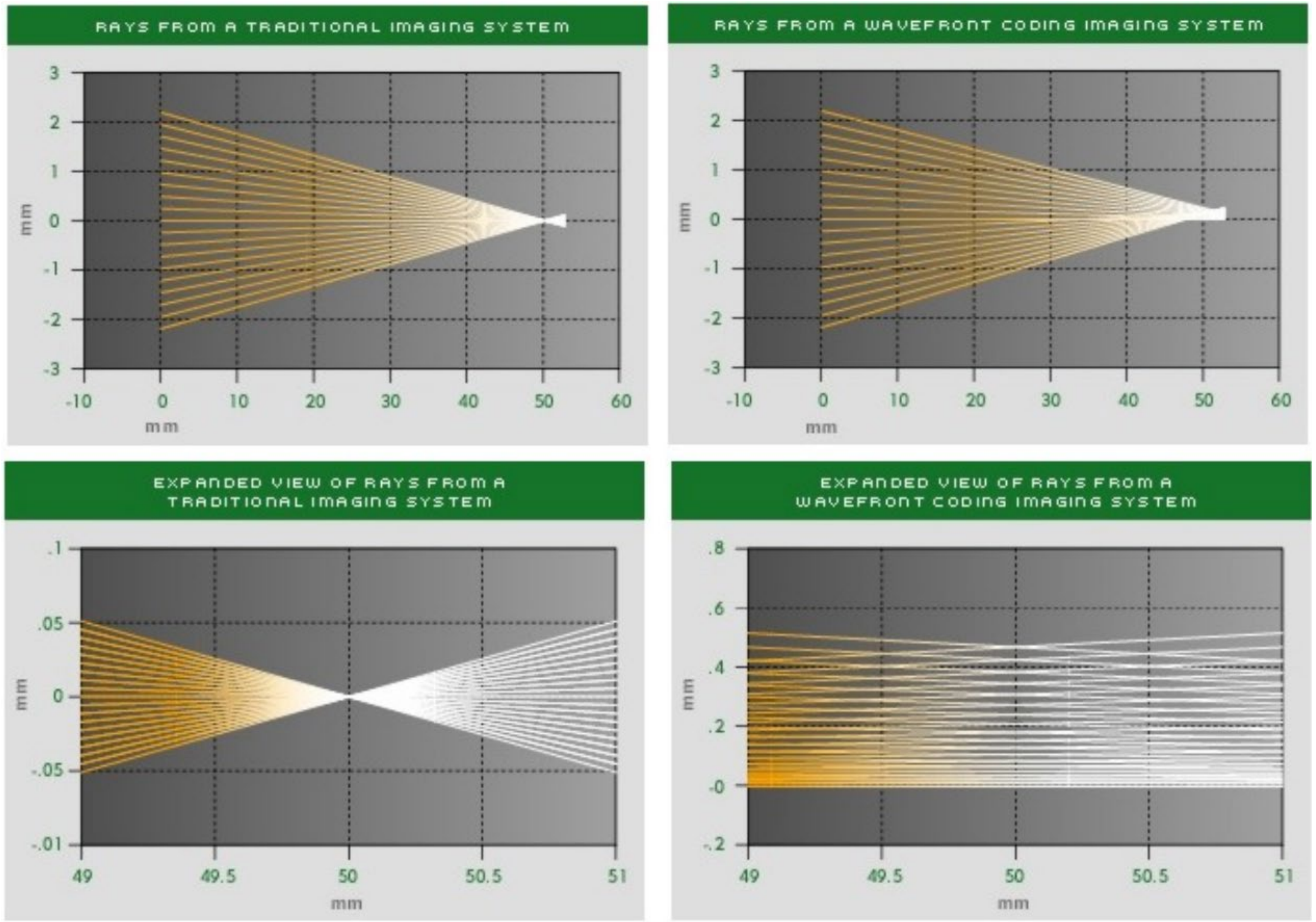
Wavefront coding



Replace lens with a cubic phase plate

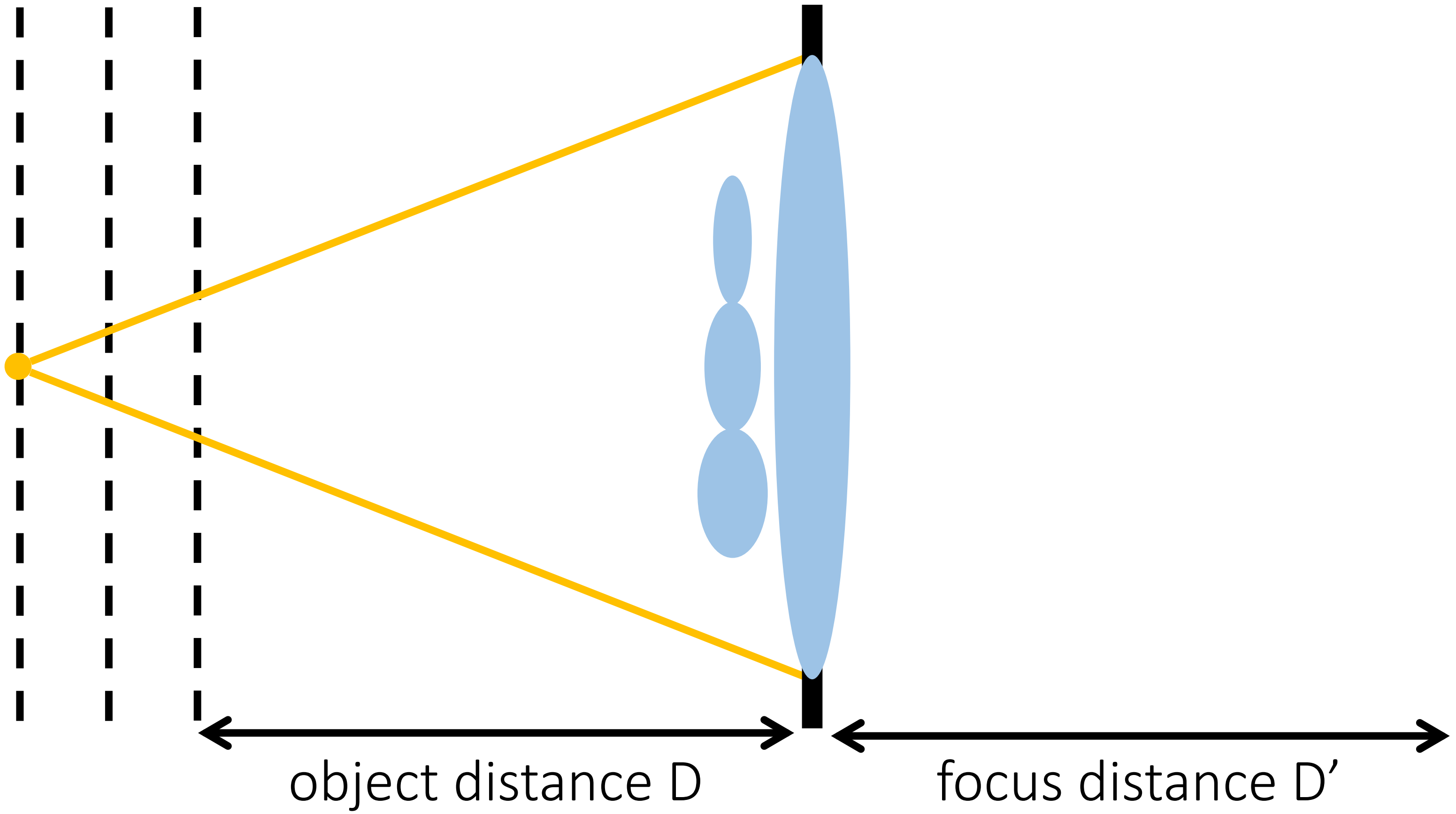
Wavefront coding





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

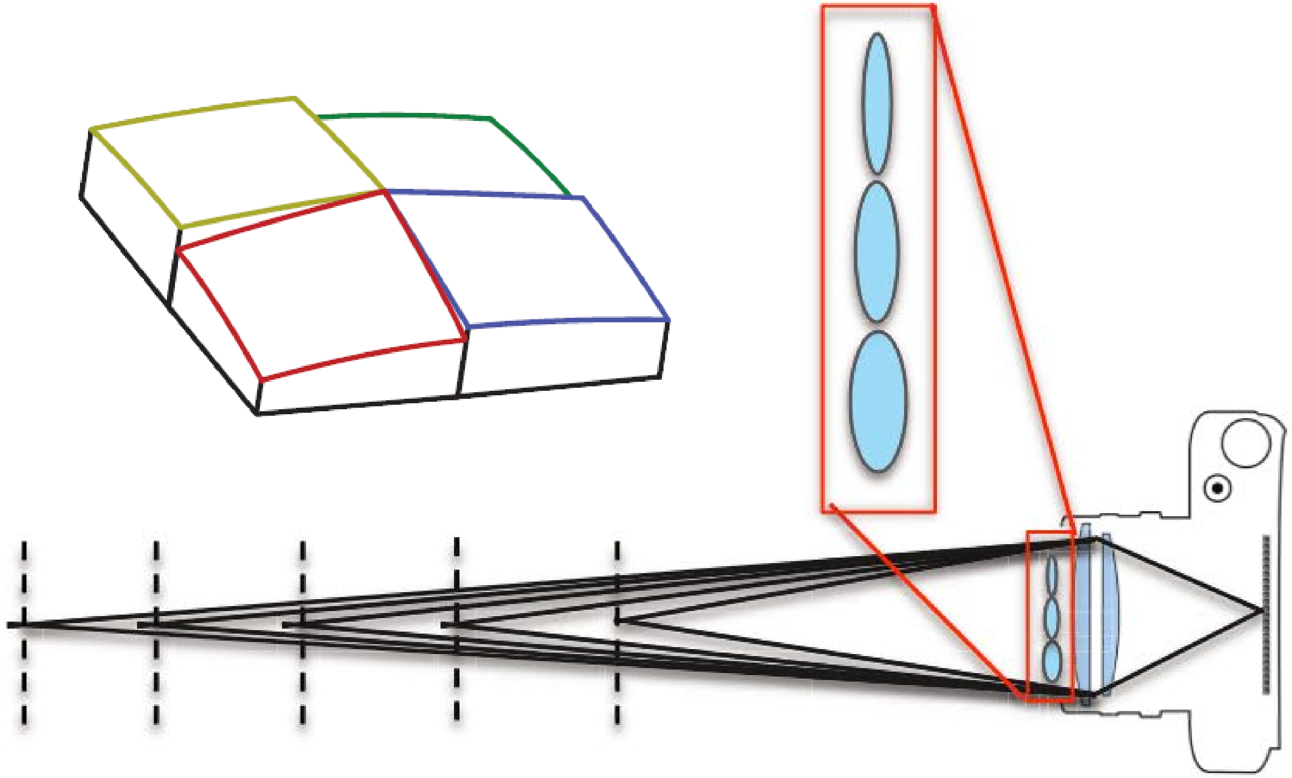
Lattice lens



Add lenslet array with varying focal length in front of lens

Lattice lens

Does this remind you of something?



Lattice lens

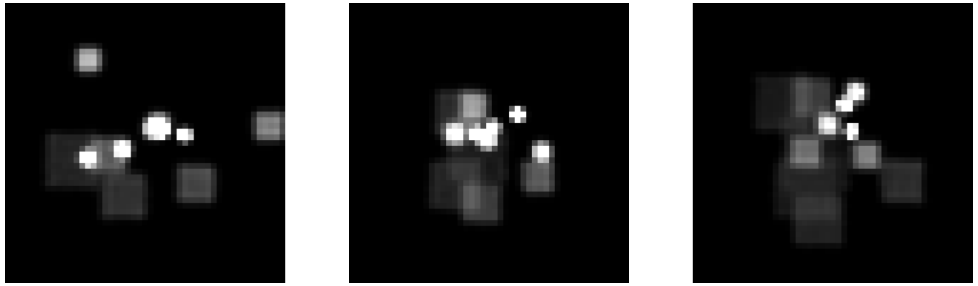
- Effectively captures only the “useful” subset of the 4D lightfield.

Light field spectrum: 4D
Image spectrum: 2D } **3D**
Depth: 1D }
→ 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



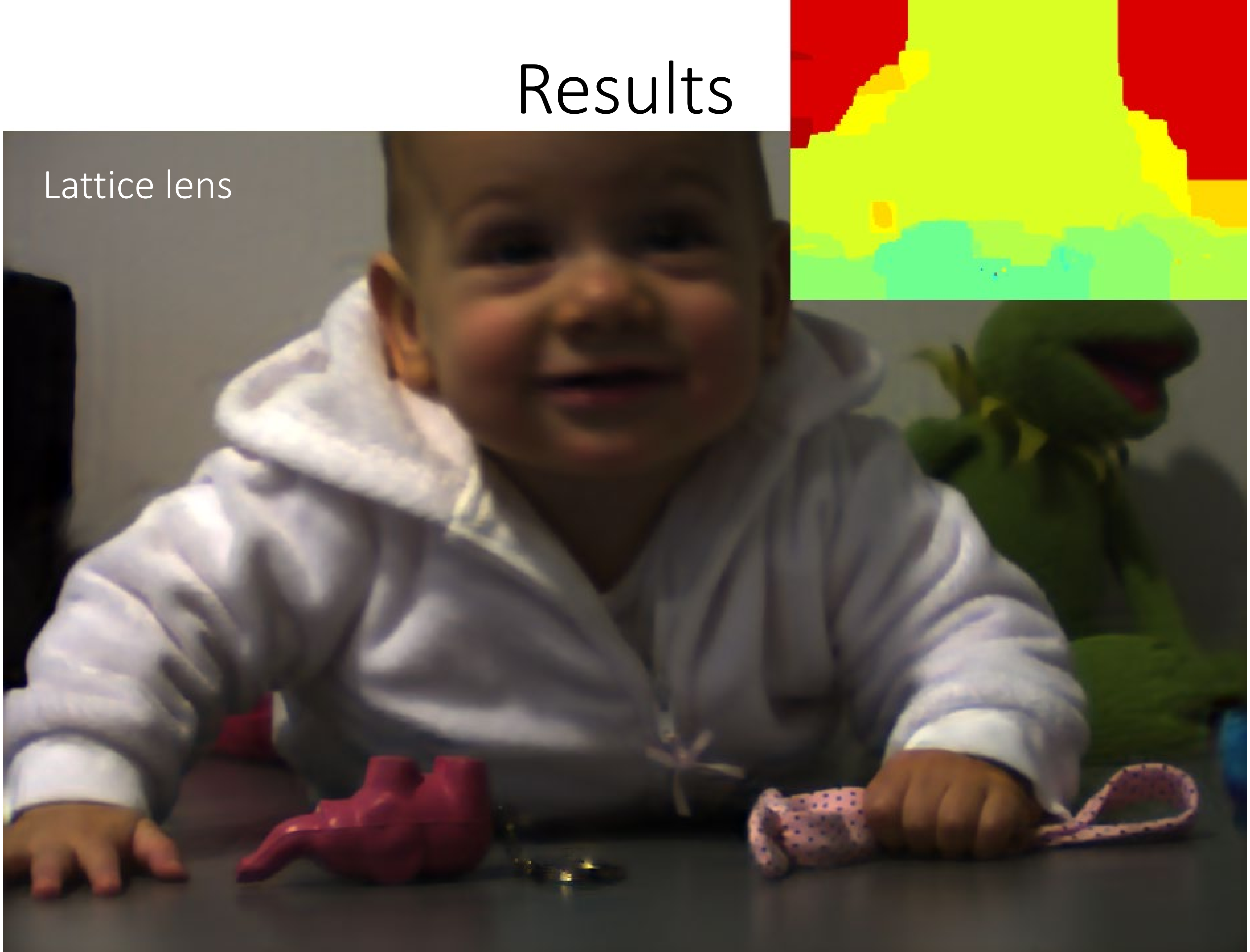
Results

Results


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

Results

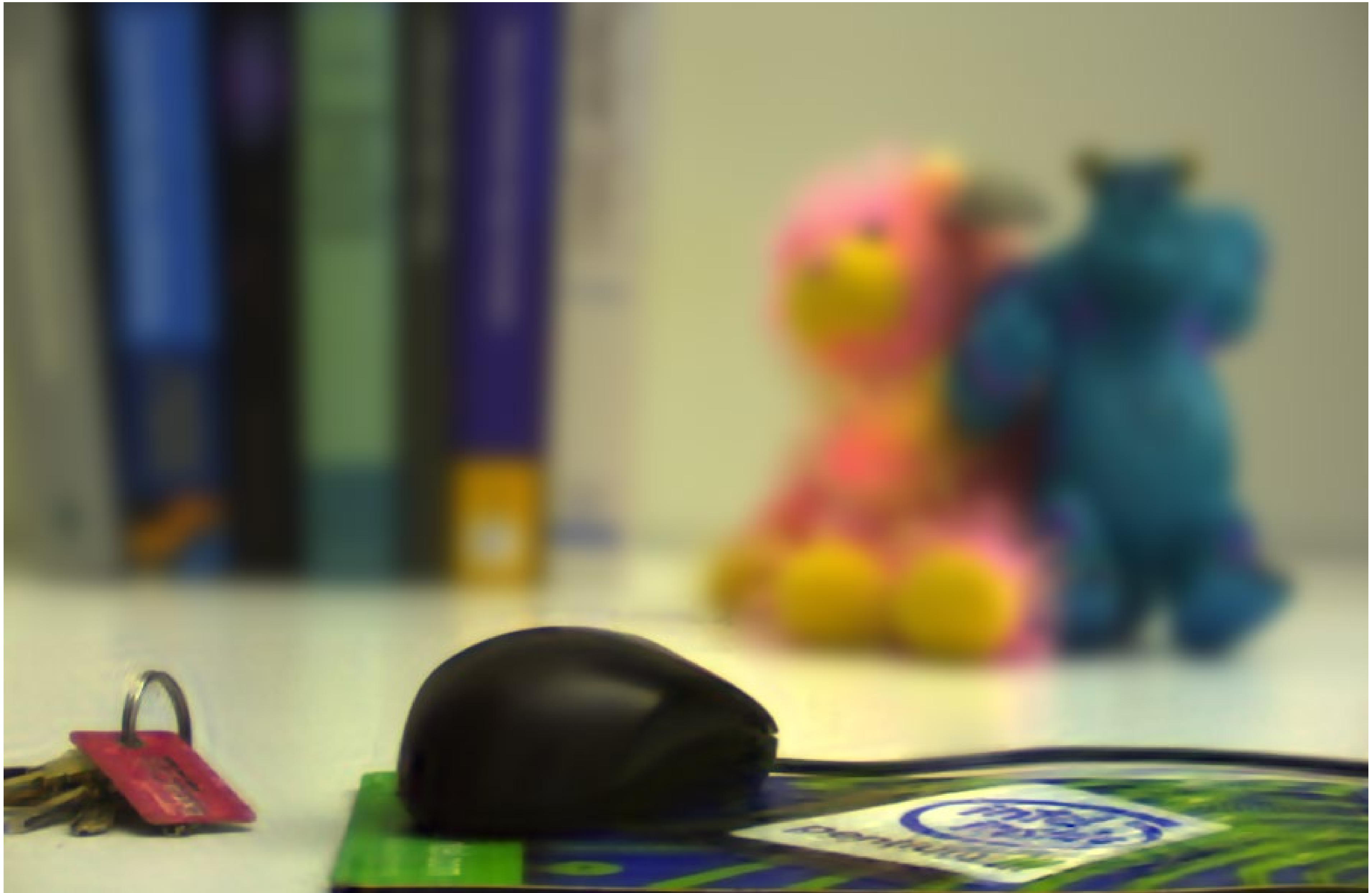
Results



Results

Results

Refocusing example



Refocusing example

Refocusing example

Comparison of different techniques

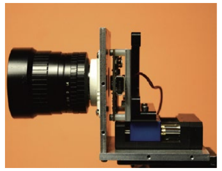
Depth of field comparison:

standard lens

<

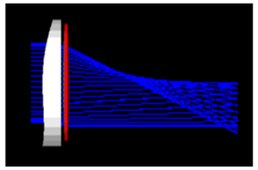
coded aperture

<<



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

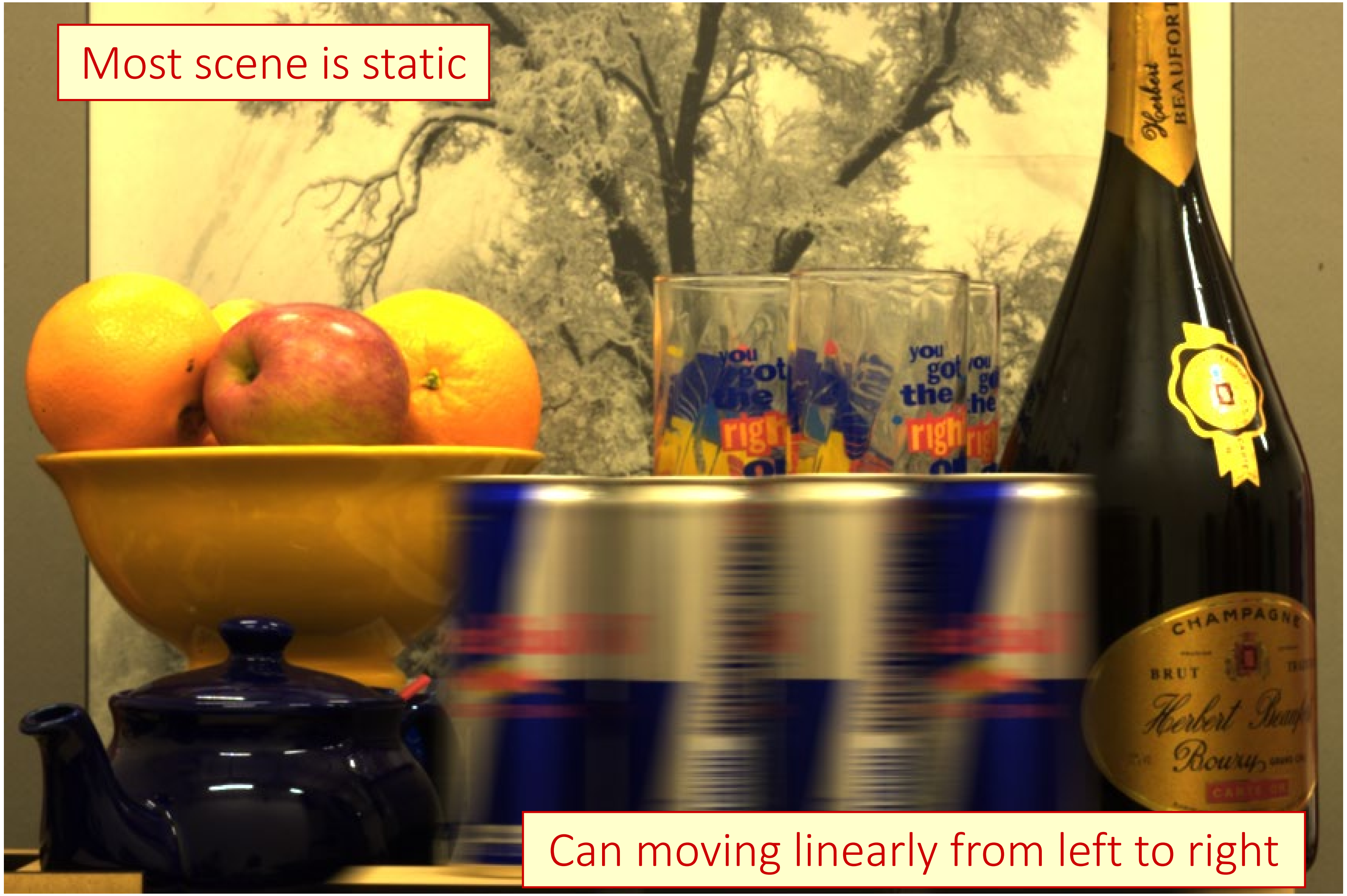
Can you think of any issues?

Dealing with motion blur

Why are our images blurry?

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

Motion blur


Most scene is static
Can moving linearly from left to right
CHAMPAGNE
BRUT
BEAUFORT

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.

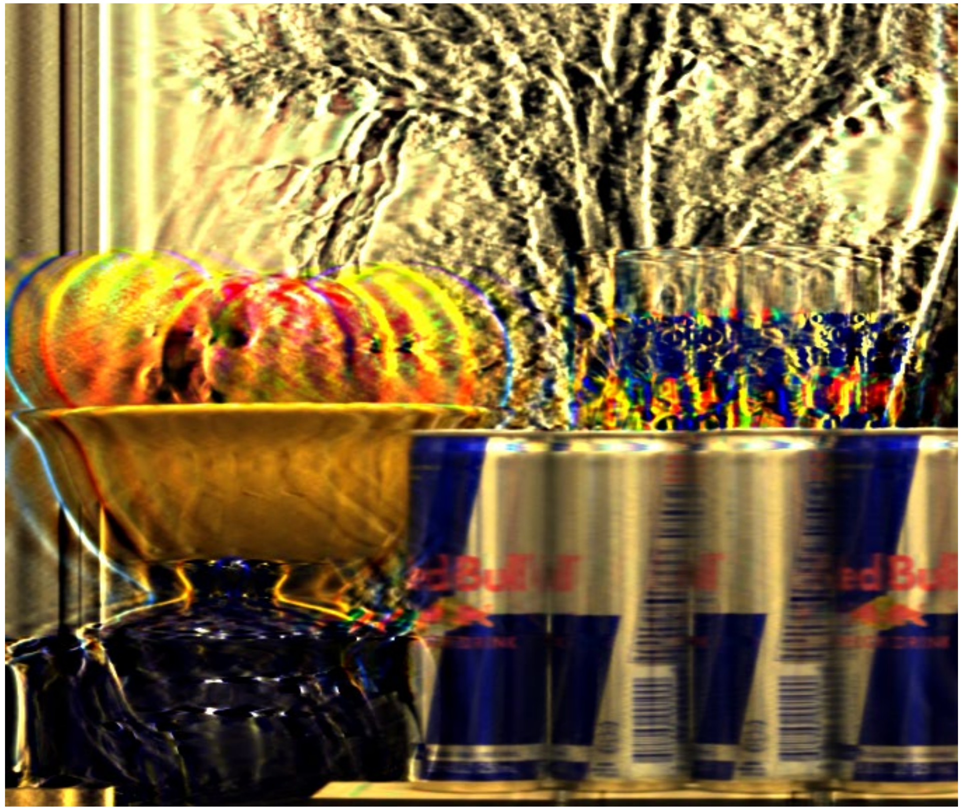
Can we use deconvolution to remove motion blur?

Challenges of motion deblurring

- Blur kernel is not invertible.

- Blur kernel is unknown.

- Blur kernel is different for different objects.

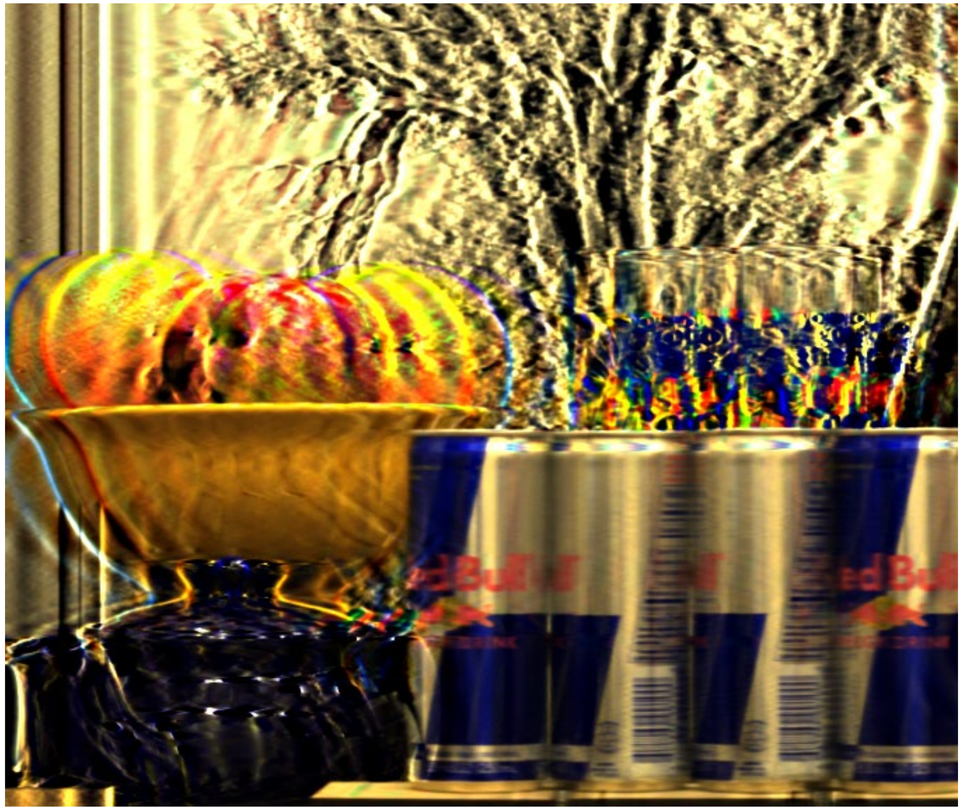


Challenges of motion deblurring

- Blur kernel is not invertible. ← How would you deal with this?

- 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

=

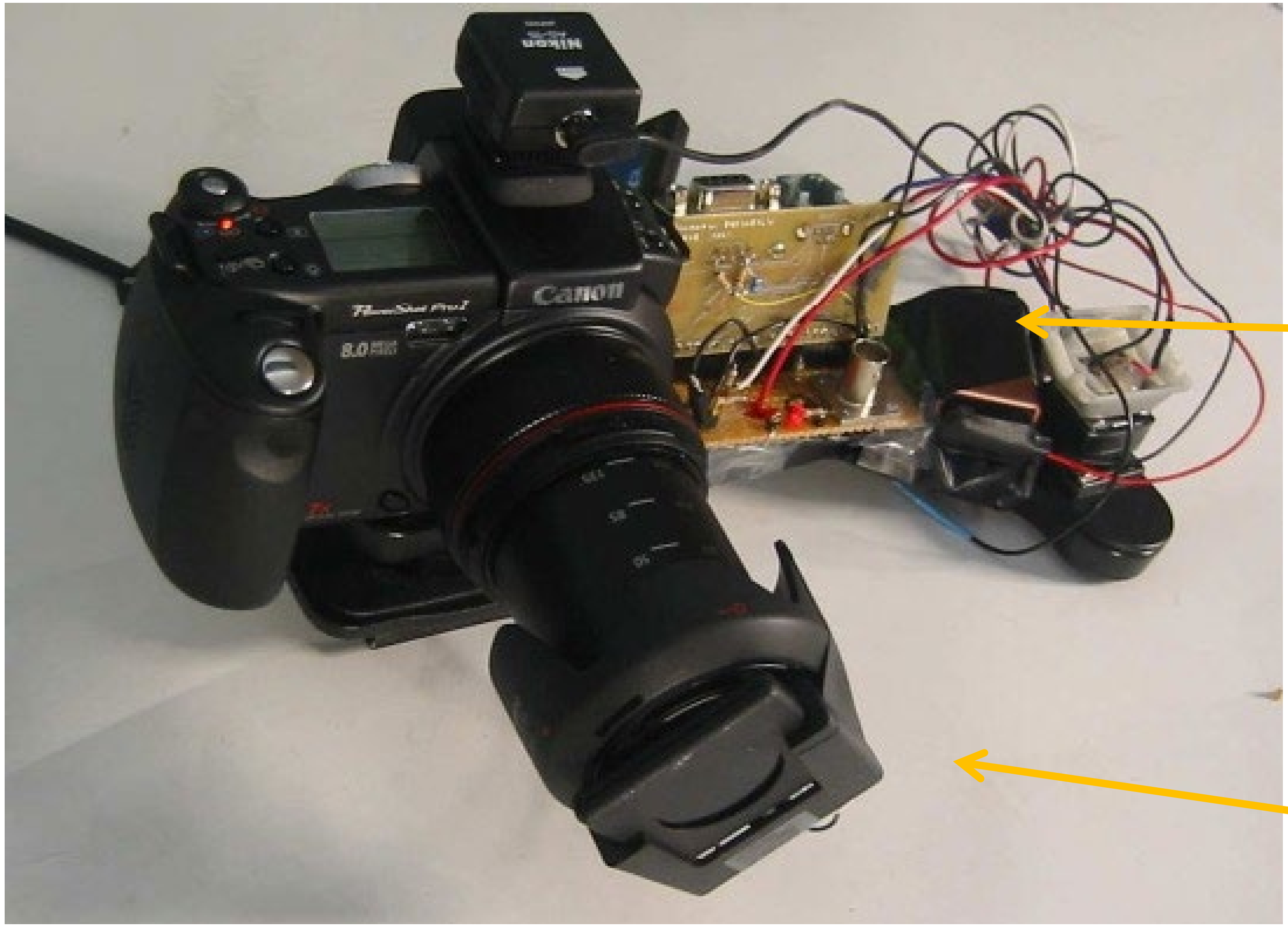
motion blur kernel

*

sharp image of static object

How would you implement coded exposure?

How would you implement coded exposure?



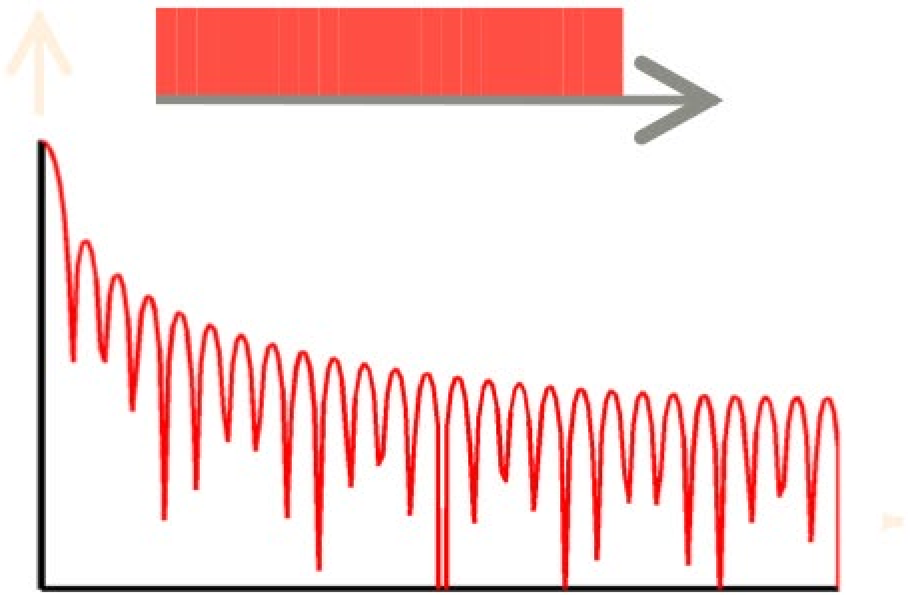
electronics for external shutter control

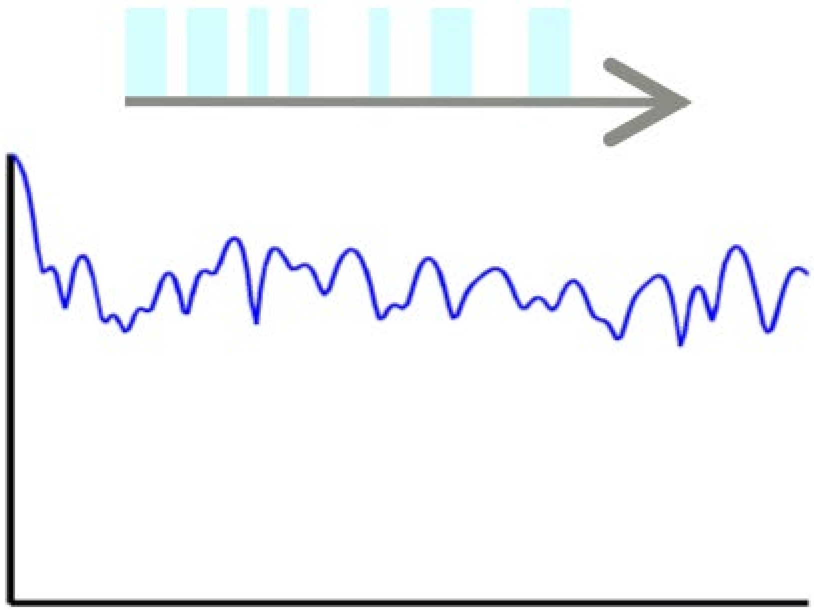
very fast external shutter

Coded exposure a.k.a. flutter shutter

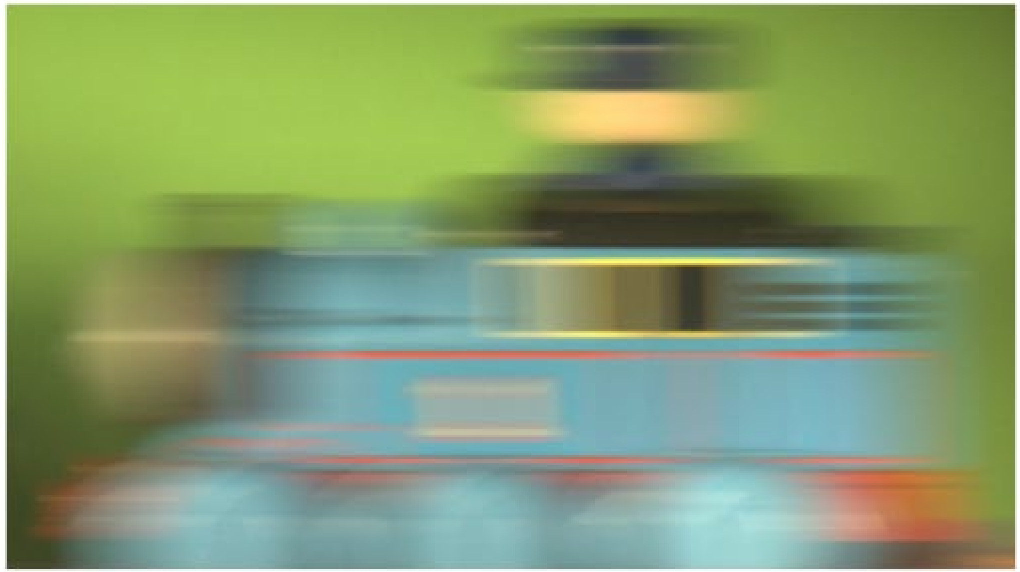
motion blur kernel
in time domain

motion blur kernel
in Fourier domain





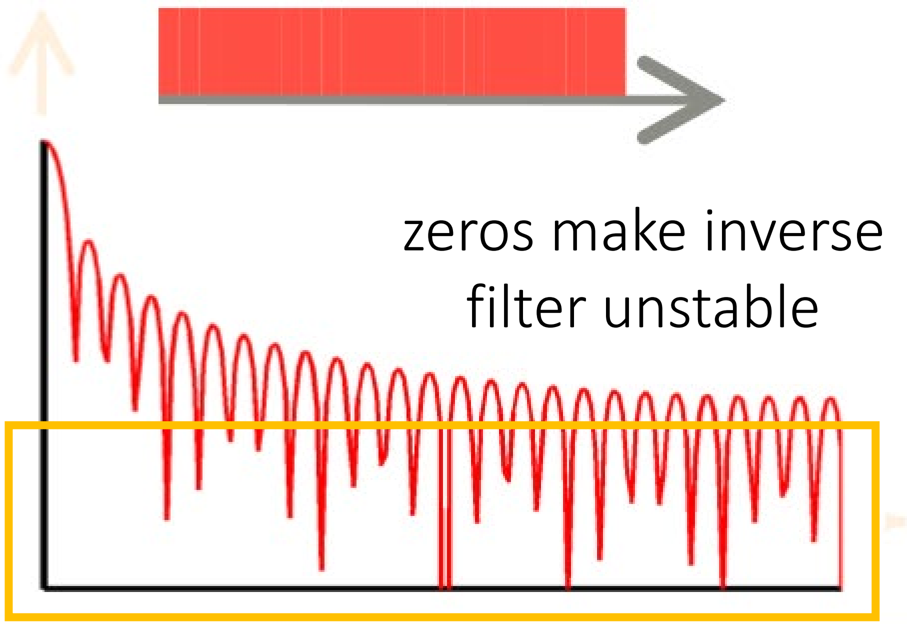
Why is flutter
shutter
better?

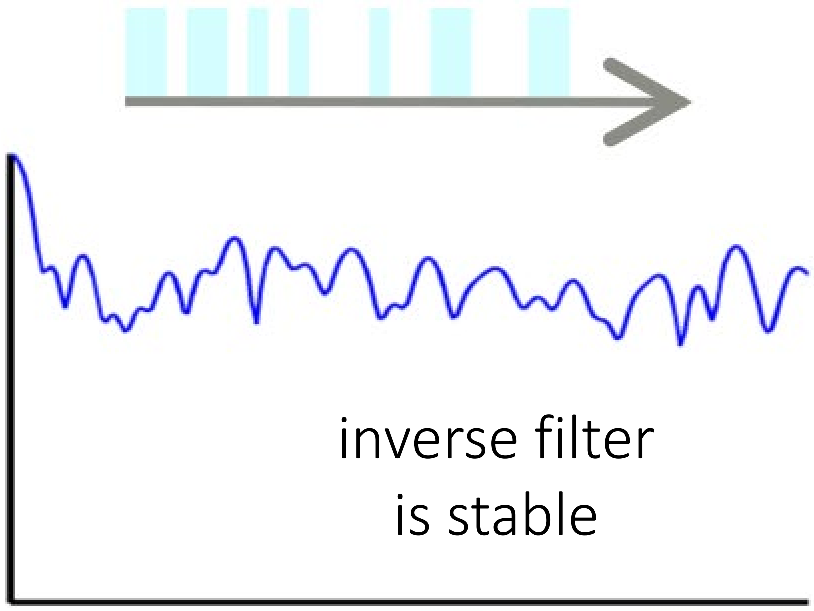


Coded exposure a.k.a. flutter shutter

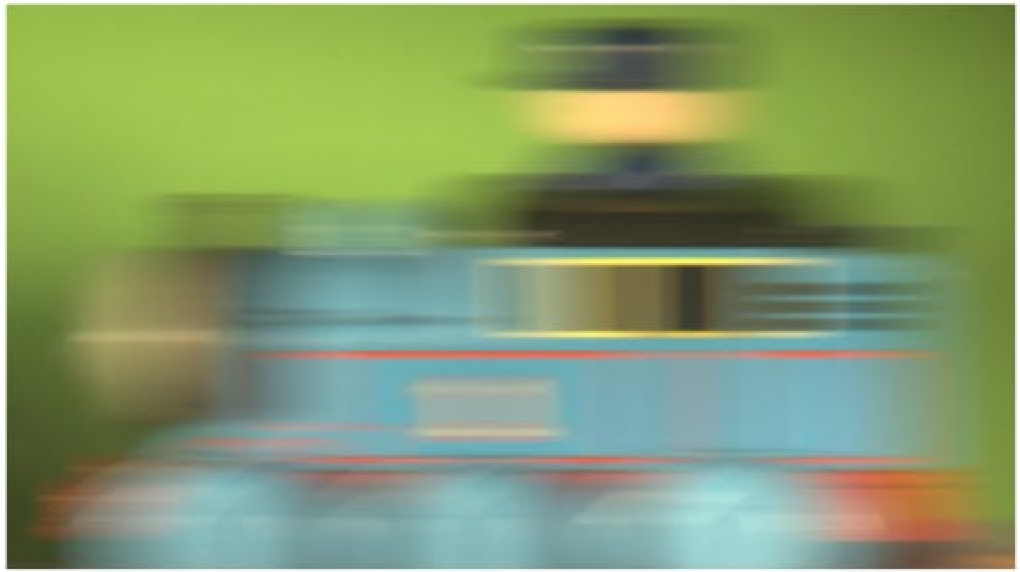
motion blur kernel in time domain

motion blur kernel in Fourier domain





Why is flutter shutter better?



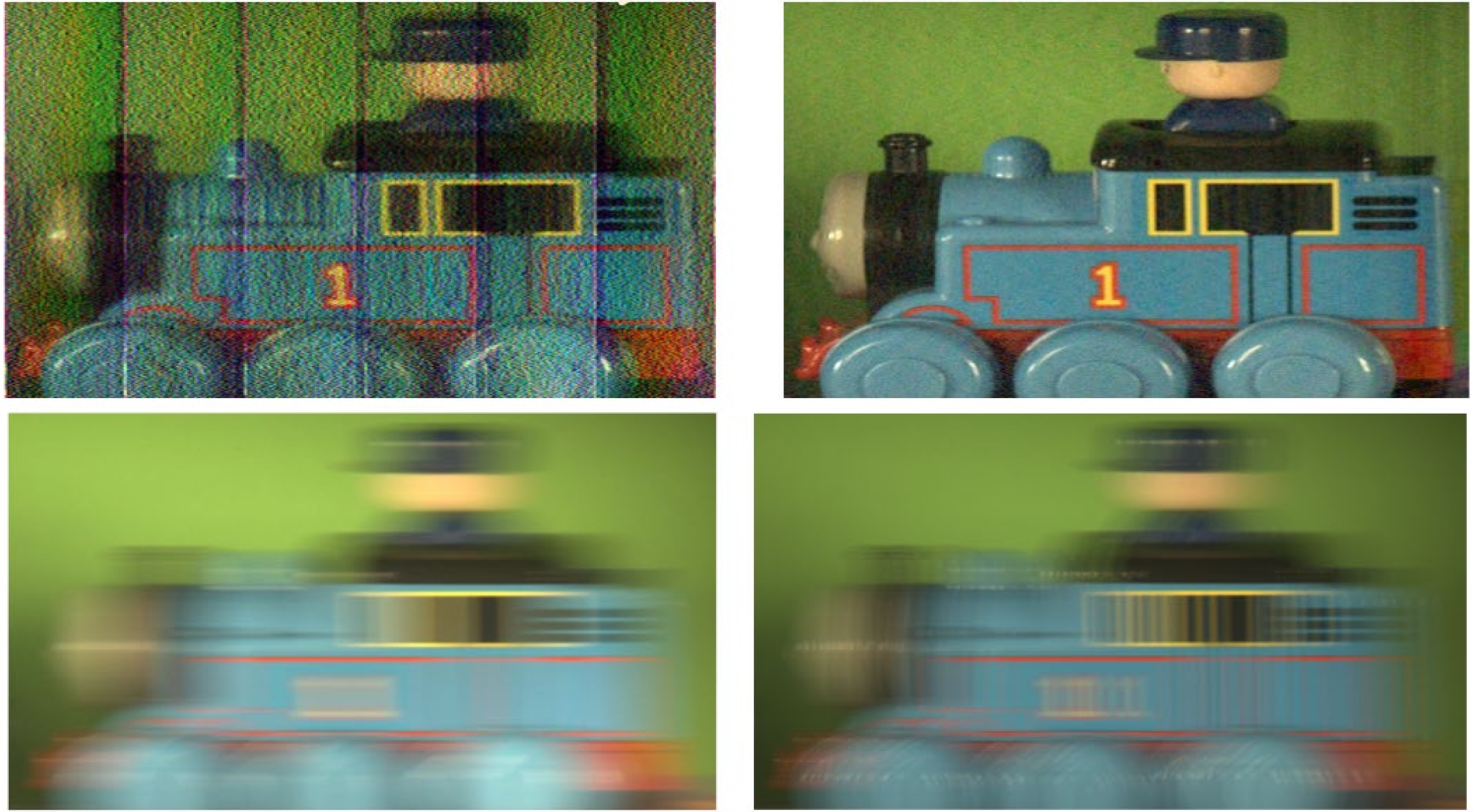
Motion deblurring comparison

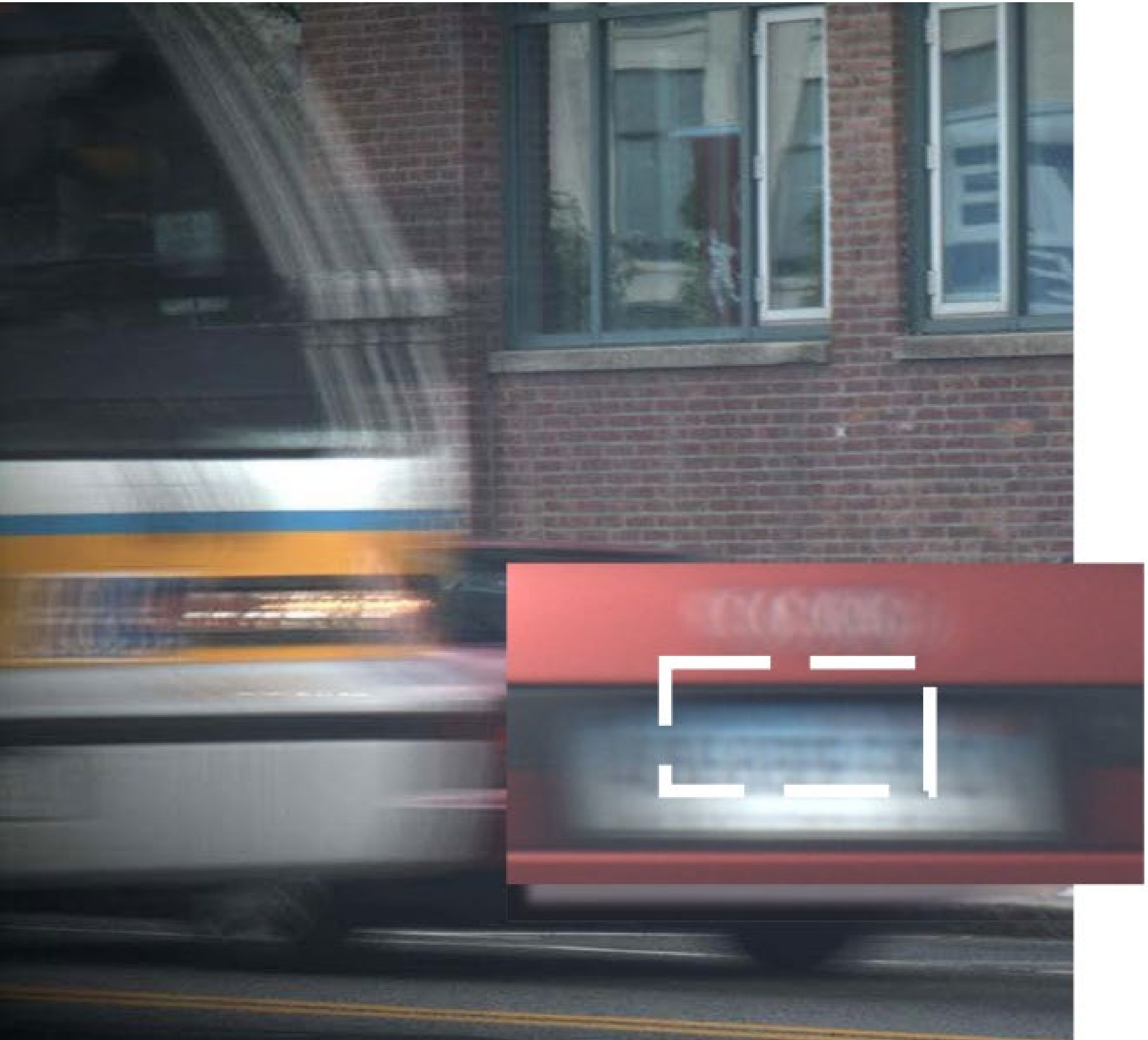
conventional photography

flutter-shutter photography

deconvolved output

blurry input





License Plate Retrieval

License Plate Retrieval

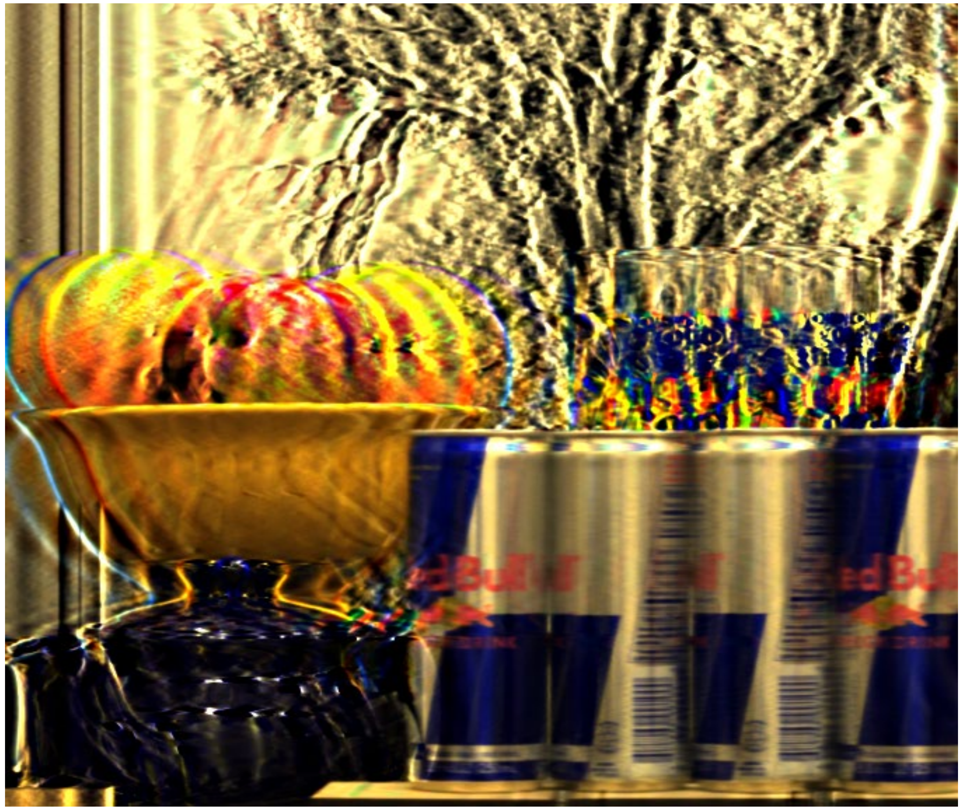
Challenges of motion deblurring

- 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).

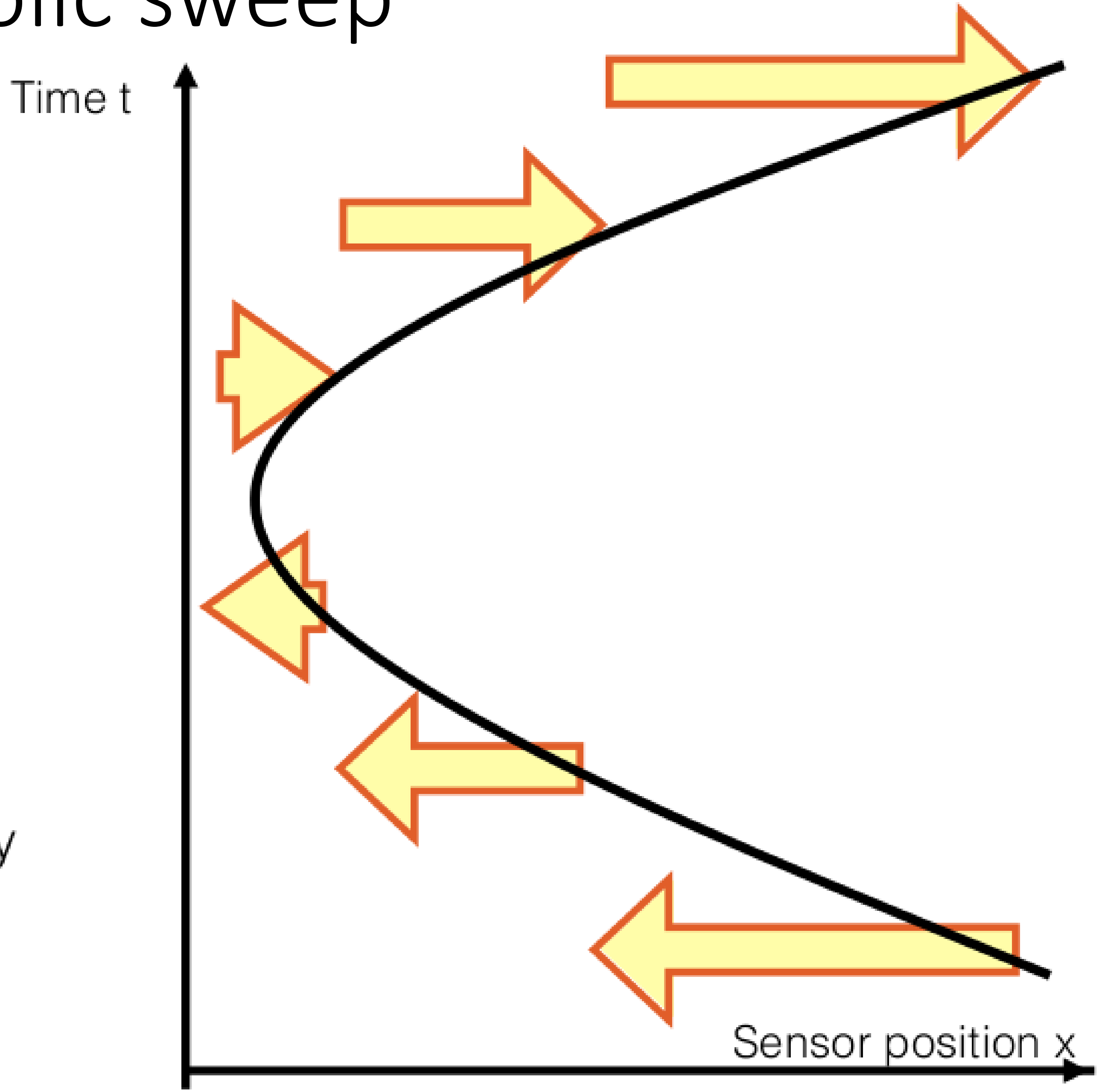
How would you achieve this?

Parabolic sweep

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

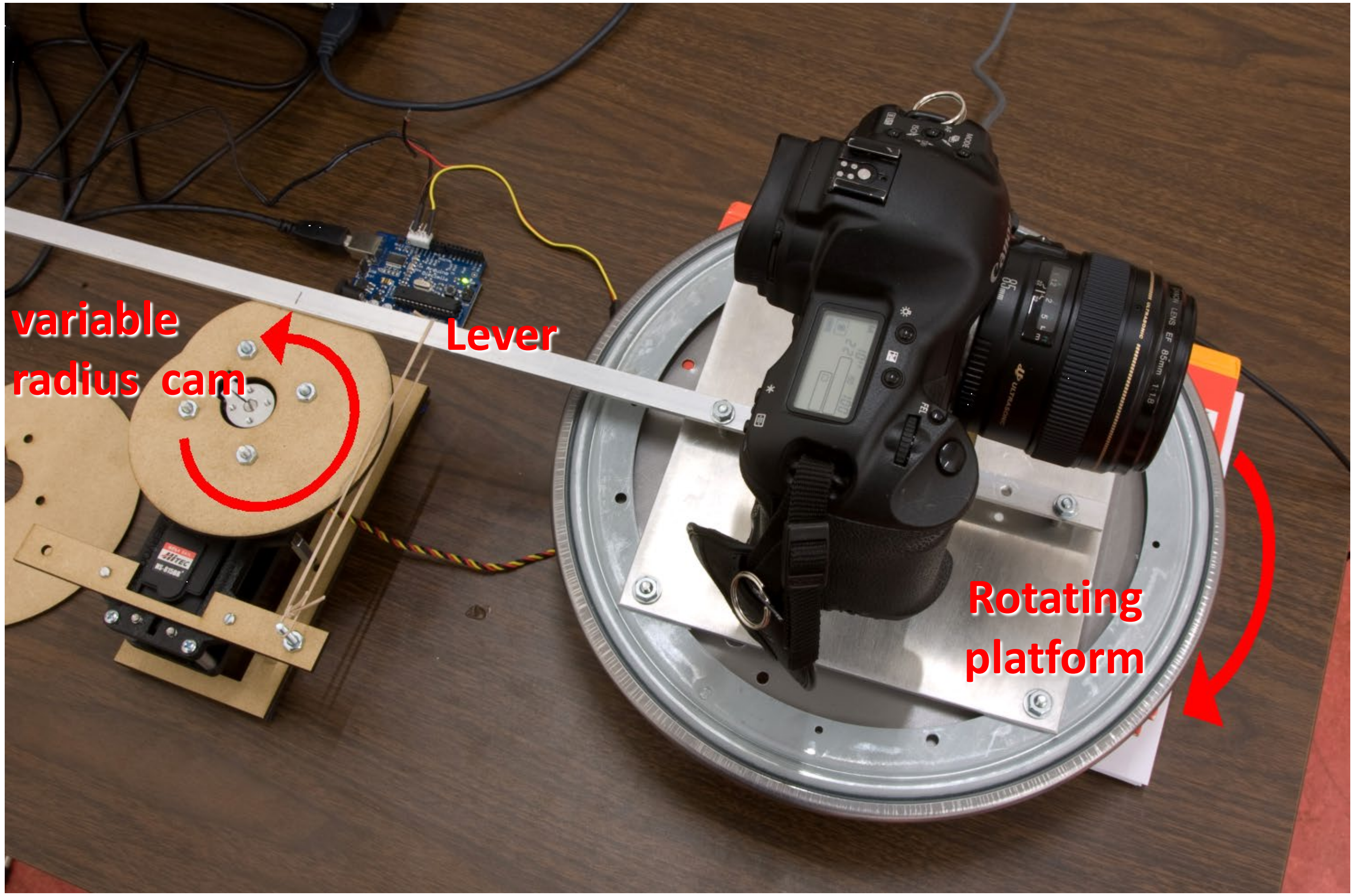
- start by moving very fast to the right
- continuously slow down until stop
- continuously accelerate to the left

- 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

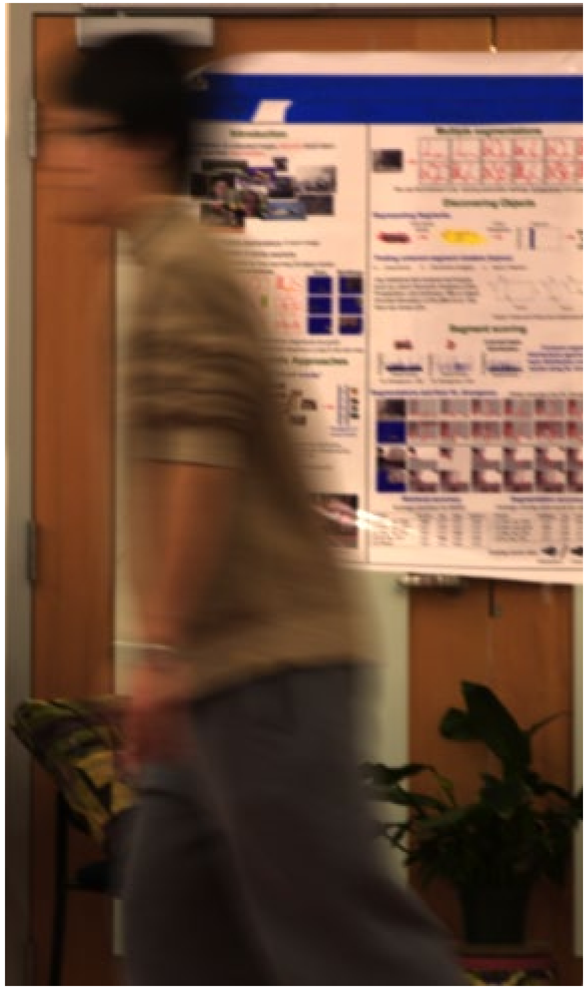
Some results

static camera input - unknown
and variable blur

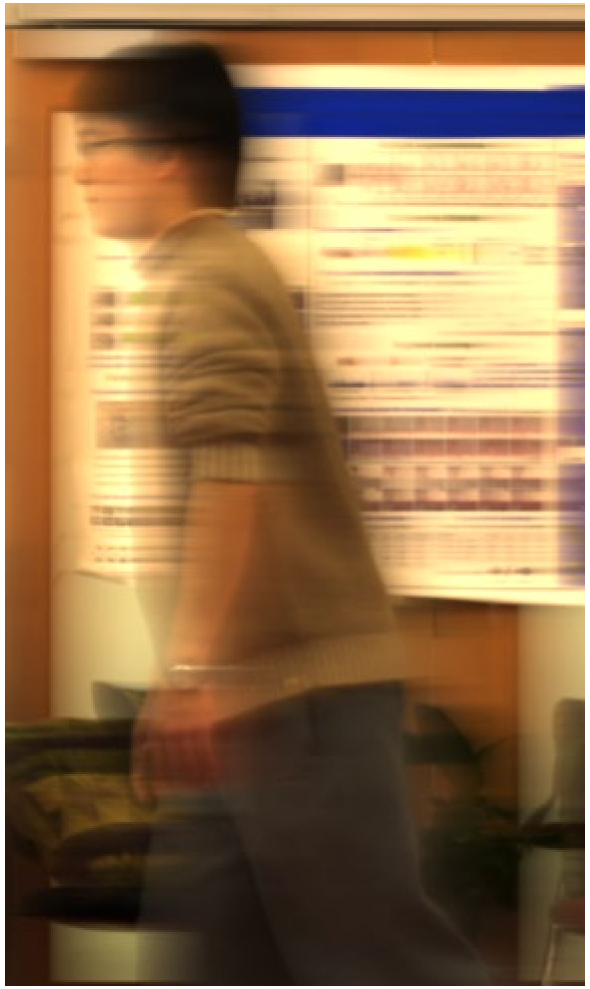
output after deconvolution

Is this blind or non-blind deconvolution?

Some results



static camera input



parabolic camera input

deconvolution output

Some results

static camera input

output after deconvolution

Why does it fail in this case?

References

Basic reading:

- Levin et al., “Image and depth from a conventional camera with a coded aperture,” SIGGRAPH 2007.
- Veeraraghavan et al., “Dappled photography: Mask enhanced cameras for heterodyned light fields and coded aperture refocusing,” SIGGRAPH 2007.
  The two papers introducing coded aperture for depth and refocusing, the first covers deblurring in more detail, whereas the second deals with optimal mask selection and includes very interesting lightfield analysis.
- Nagahara et al., “Flexible depth of field photography,” ECCV 2008 and PAMI 2010.
  The focal sweep paper.
- Dowski and Cathey, “Extended depth of field through wave-front coding,” Applied Optics 1995.
  The wavefront coding paper.
- Levin et al., “4D Frequency Analysis of Computational Cameras for Depth of Field Extension,” SIGGRAPH 2009.
  The lattice focal lens paper, which also includes a discussion of wavefront coding.
- Cossairt et al., “Diffusion Coded Photography for Extended Depth of Field,” SIGGRAPH 2010.
  The diffusion coded photography paper.
- Raskar et al., “Coded Exposure Photography: Motion Deblurring using Fluttered Shutter,” SIGGRAPH 2006.
  The flutter shutter paper.
- Levin et al., “Motion-Invariant Photography,” SIGGRAPH 2008.
  The motion-invariant photography paper.

Additional reading:

- Zhang and Levoy, “Wigner distributions and how they relate to the light field,” ICCP 2009.
  This paper has a nice discussion of wavefront coding, in addition to analysis of lightfields and their relationship to wave optics concepts.
- Gehm et al., “Single-shot compressive spectral imaging with a dual-disperser architecture,” Optics Express 2007.
  This paper introduces the use of coded apertures for hyperspectral imaging, instead of depth and refocusing.