Computational photography techniques based on deconvolution

CS 178, Spring 2009

Marc Levoy
Computer Science Department
Stanford University

Removing camera shake

- Can you fix a blurry image by sharpening it in Photoshop?

(simulated blurry image)

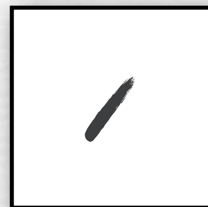
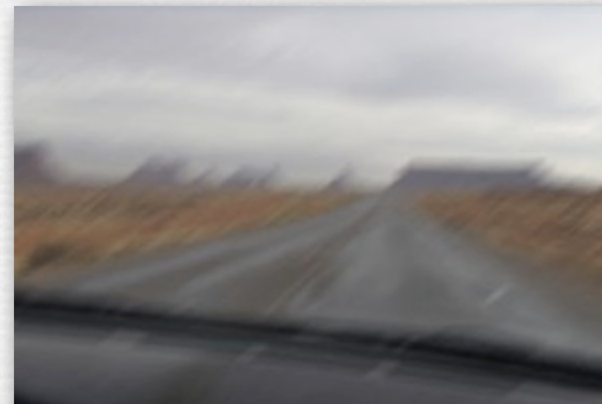
Removing camera shake

- Can you fix a blurry image by sharpening it in Photoshop?

(simulated blurry image)

Removing camera shake, 2nd try

- camera shake can be modeled as a 2D convolution

 $\otimes$  $=$ 

- recall that discrete convolution replaces each pixel with a linear combination of nearby pixels
- in linear algebra, a matrix replaces each element in a vector with a linear combination of all other elements

$\therefore$ convolution can be formulated as matrix multiplication

Convolution as matrix multiplication

- let the sharp scene be represented by a vector

$$\mathbf{f} = [4 \quad 7 \quad 8 \quad 4 \quad 2 \quad 5 \quad 9 \quad 6 \quad 8 \quad 4 \quad 2]$$

- let the filter kernel be represented as a second vector

$$\mathbf{g} = [1 \;\; 2 \;\; 3 \;\; 2 \;\; 1]$$

- the convolution $\mathbf{f} \otimes \mathbf{g}$ becomes the matrix-vector product

$$\mathbf{Ax} = \begin{bmatrix} 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 4 \\ 7 \\ 8 \\ 4 \\ 2 \\ 5 \\ 9 \\ 6 \\ 8 \\ 4 \\ 2 \end{bmatrix}$$

where $\mathbf{x} = \mathbf{f}^T$
and $\mathbf{A}$ is built from $\mathbf{g}$ as shown

Convolution as matrix multiplication

- let the sharp scene be represented by a vector

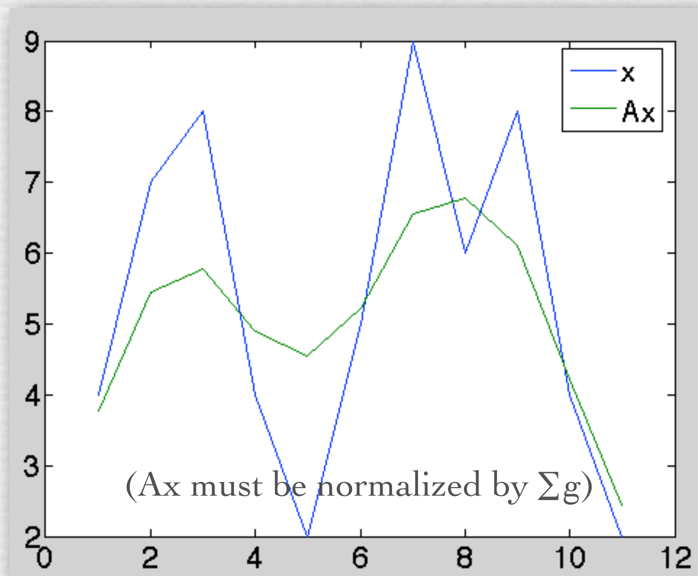
$$\mathbf{f} = [4 \quad 7 \quad 8 \quad 4 \quad 2 \quad 5 \quad 9 \quad 6 \quad 8 \quad 4 \quad 2]$$

- let the filter kernel be represented as a second vector

$$\mathbf{g} = [1 \;\; 2 \;\; 3 \;\; 2 \;\; 1]$$

- the convolution $\mathbf{f} \otimes \mathbf{g}$ becomes the matrix-vector product

$$\mathbf{Ax} = \begin{bmatrix} 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 4 \\ 7 \\ 8 \\ 4 \\ 2 \\ 5 \\ 9 \\ 6 \\ 8 \\ 4 \\ 2 \end{bmatrix}$$



Inverting convolution (*deconvolution*)

- if the blurred image $\mathbf{b}$ is given by

$$\mathbf{A}\mathbf{x} = \mathbf{b}$$

- then the sharp scene $\mathbf{x}$ can be recovered by

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$$

where

$$\mathbf{A}^{-1} = \left[\begin{array}{rrrrrrrrrrr}
0.8571 & -0.7857 & 0.0000 & 0.6429 & -0.5714 & 0 & 0.4286 & -0.3571 & -0.0000 & 0.2143 & -0.1429 \\
-0.7857 & 1.5536 & -0.7500 & -0.5893 & 1.1071 & -0.5000 & -0.3929 & 0.6607 & -0.2500 & -0.1964 & 0.2143 \\
0.0000 & -0.7500 & 1.5000 & -0.7500 & -0.5000 & 1.0000 & -0.5000 & -0.2500 & 0.5000 & -0.2500 & 0.0000 \\
0.6429 & -0.5893 & -0.7500 & 1.9821 & -1.1786 & -0.5000 & 1.3214 & -0.7679 & -0.2500 & 0.6607 & -0.3571 \\
-0.5714 & 1.1071 & -0.5000 & -1.1786 & 2.2143 & -1.0000 & -0.7857 & 1.3214 & -0.5000 & -0.3929 & 0.4286 \\
0 & -0.5000 & 1.0000 & -0.5000 & -1.0000 & 2.0000 & -1.0000 & -0.5000 & 1.0000 & -0.5000 & 0.0000 \\
0.4286 & -0.3929 & -0.5000 & 1.3214 & -0.7857 & -1.0000 & 2.2143 & -1.1786 & -0.5000 & 1.1071 & -0.5714 \\
-0.3571 & 0.6607 & -0.2500 & -0.7679 & 1.3214 & -0.5000 & -1.1786 & 1.9821 & -0.7500 & -0.5893 & 0.6429 \\
-0.0000 & -0.2500 & 0.5000 & -0.2500 & -0.5000 & 1.0000 & -0.5000 & -0.7500 & 1.5000 & -0.7500 & 0.0000 \\
0.2143 & -0.1964 & -0.2500 & 0.6607 & -0.3929 & -0.5000 & 1.1071 & -0.5893 & -0.7500 & 1.5536 & -0.7857 \\
-0.1429 & 0.2143 & 0.0000 & -0.3571 & 0.4286 & 0.0000 & -0.5714 & 0.6429 & 0.0000 & -0.7857 & 0.8571
\end{array}\right]$$

(again omitting normalization by $\sum g$; see http://graphics.stanford.edu/courses/cs178-09/demos/deconvolution.m)

Why is deconvolution hard?

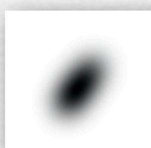
- matrix **A** and blurred image **b** are typically <u>very</u> <u>big</u>
- for a 10 megapixel image
  - **A** has 10 million rows and 10 million columns
  - **b** has 10 million entries
- matrix **A** is typically <u>very</u> <u>sparse</u>
  - mostly zeros
- methods for solving big sparse systems of equations
  - conjugate gradient descent
  - etc.

Another reason deconvolution is hard

- matrix **A** may be *poorly conditioned*
  - a small change (or noise) in **b** causes a large change in **x**

A **x** = **b**

Another reason deconvolution is hard

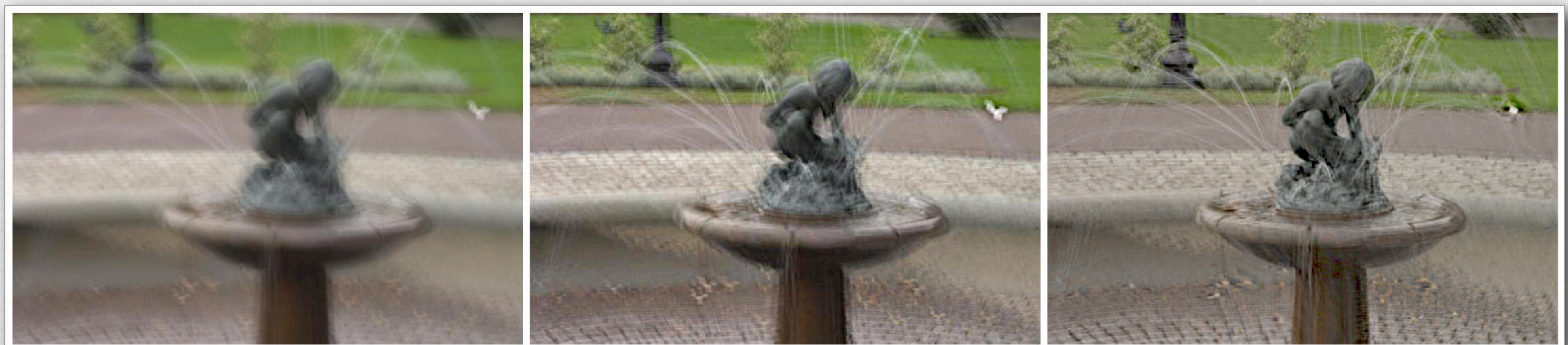
- matrix **A** may be *poorly conditioned*
  - a small change (or noise) in **b** causes a large change in **x**
- equivalently, its Fourier transform may contain zeros
  - sinusoids of some frequencies will be missing from **b**
- to be well conditioned, the filter shouldn't be smooth
  - bad:  better: 
  - convolution by the first throws away detail, creating zeros
  - convolution by the second makes many sharp copies
- inverting an ill-conditioned **A** produces a noisy result

Blind deconvolution

- sometimes you don't know **x** <u>or</u> **A**
  - i.e. you don't know the sharp scene or the filter
- solving blind deconvolution problems
  - use a *prior* assumption about what the unknown sharp scene **x** should look like
- this is hard, and we're not very good at it
  - solutions typically contain ringing, or worse...

Removing camera shake
[Fergus SIGGRAPH 2006]

- deconvolve blurred image,
  using the statistics of natural scenes as a prior

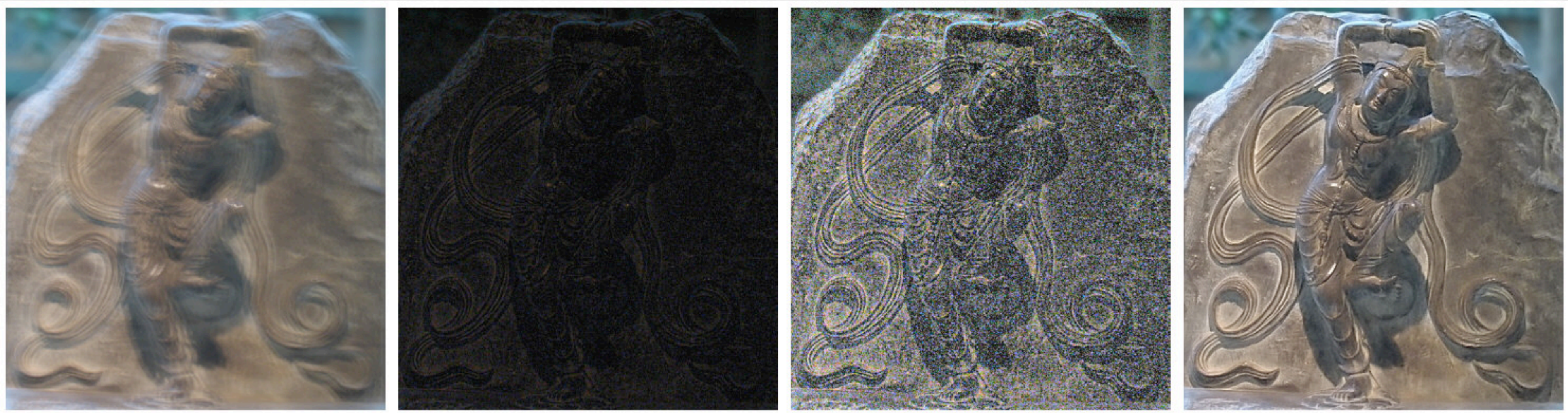


blurred image | Photoshop Unsharp Mask | deconvolution

blur kernel

Removing camera shake
[Yuan SIGGRAPH 2007]

- deconvolve long-exposure (blurred) image, using short-exposure (noisy) image as a prior

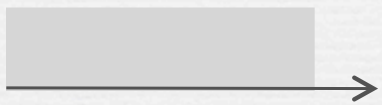


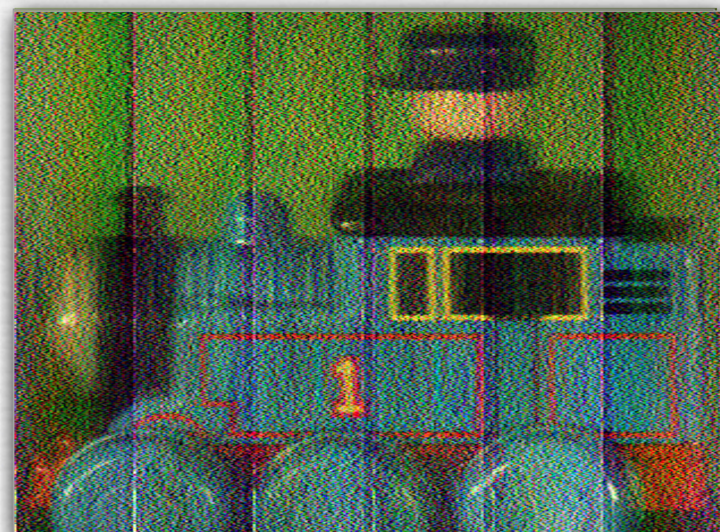
long exposure (blurry) | short exposure (dark) | same, scaled up (noisy) | joint deconvolution

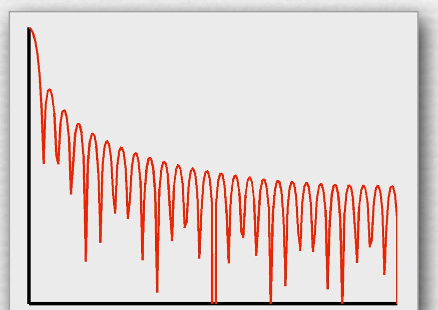
Removing motion blur

[Raskar SIGGRAPH 2006]

continuous shutter



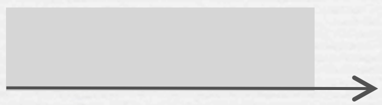




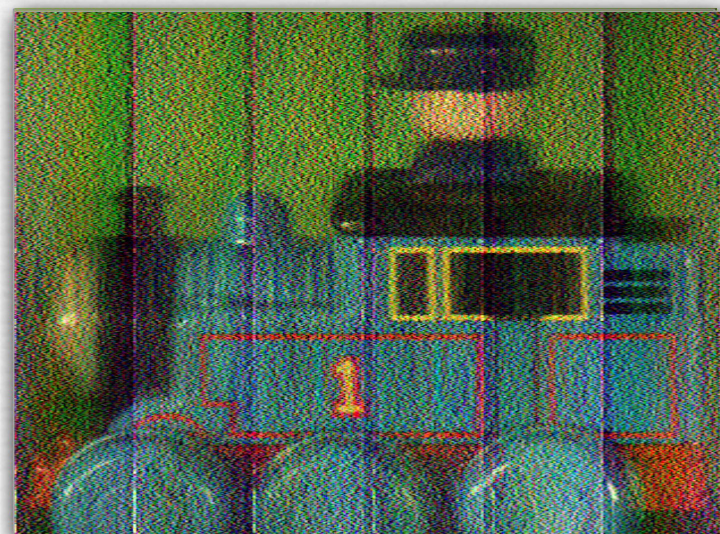
Removing motion blur

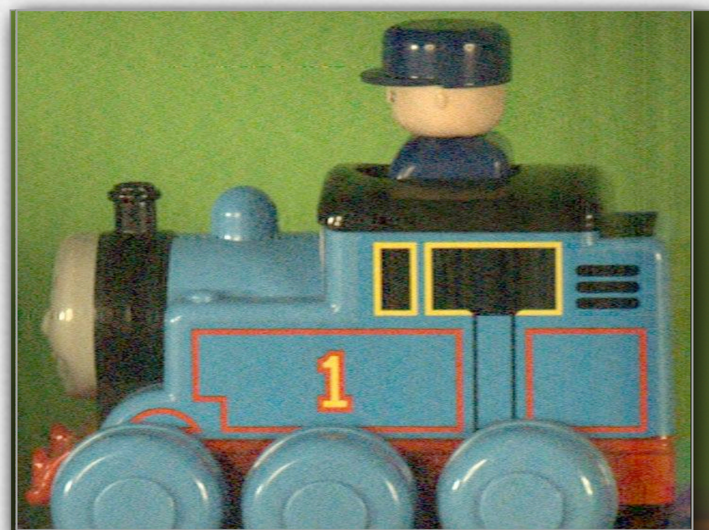
[Raskar SIGGRAPH 2006]

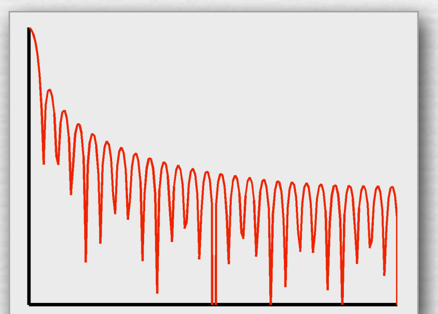
continuous shutter

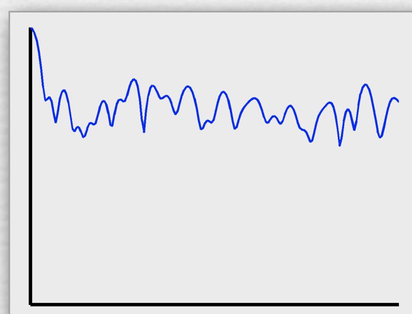


fluttered shutter










TOP CAB

Removing defocus

- a.k.a. extended depth of field (EDOF)

- all-focus algorithm
- wavefront coding + deconvolution
- rubber focus + deconvolution

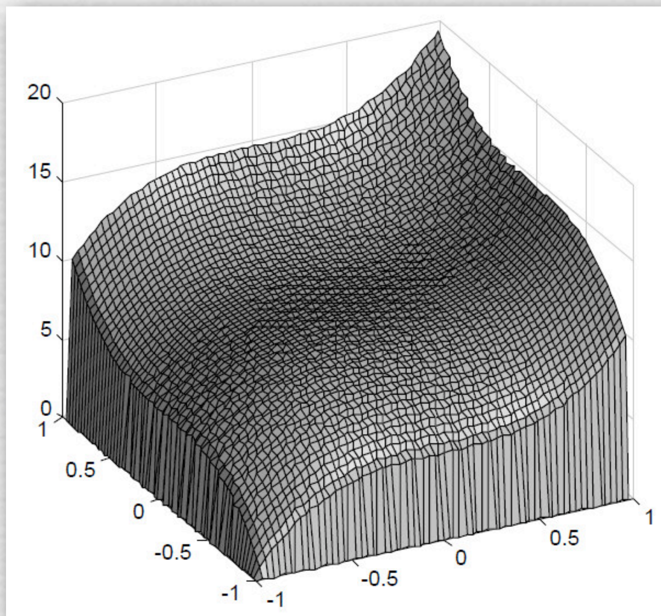
All-focus algorithm

[Agarwala SIGGRAPH 2004]

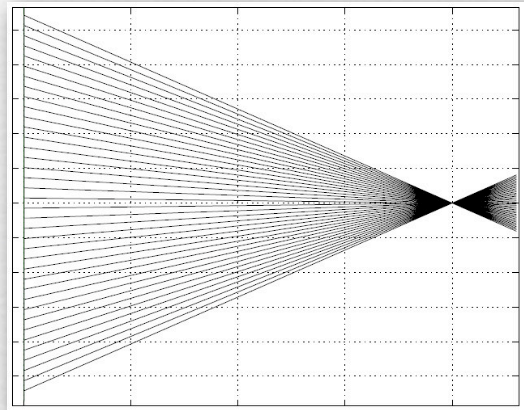
1 2 3 4 all

Wavefront coding

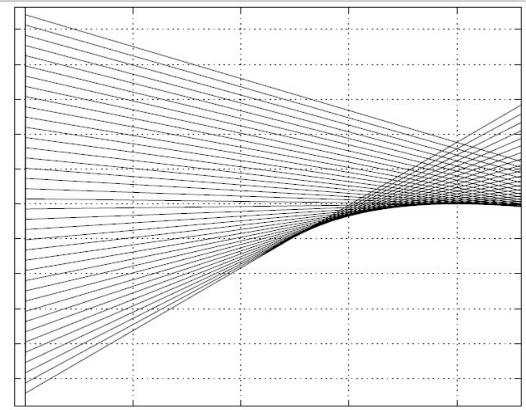
[Dowski 1995]



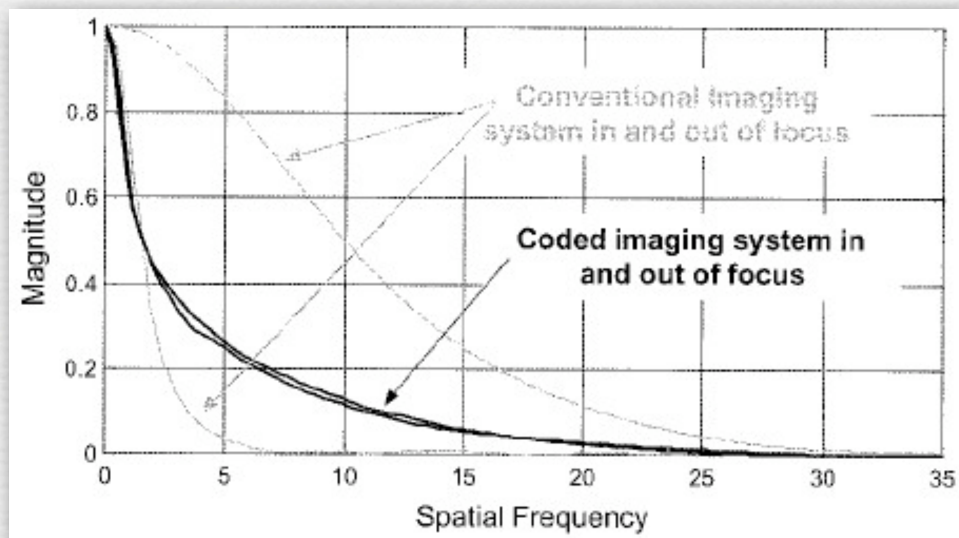
profile of
cubic phase plate



ray trace through
a normal lens



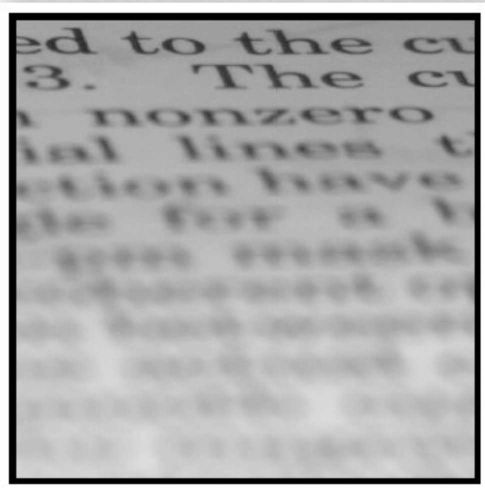
lens plus
cubic phase plate

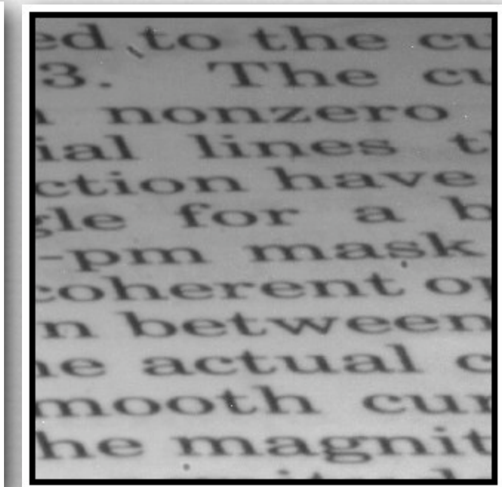


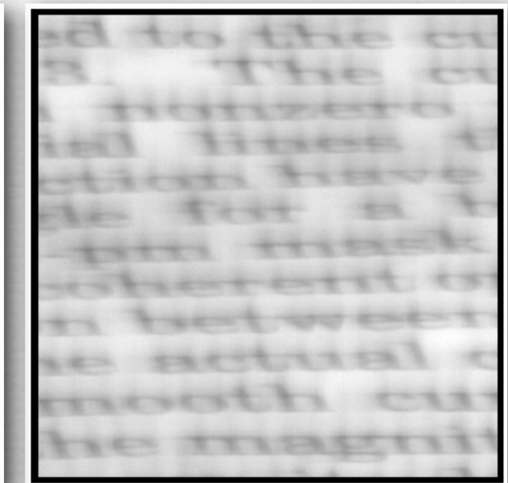
MTFs through lens and cubic phase plate

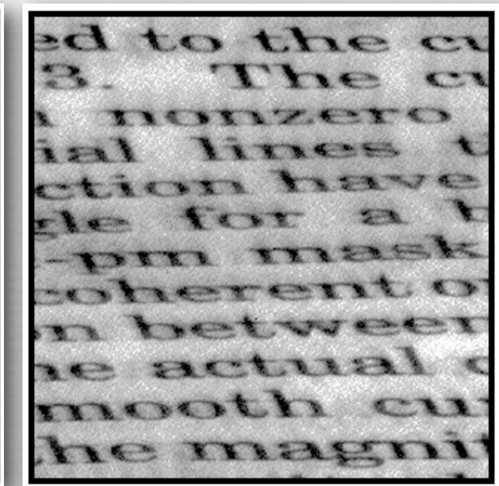
Wavefront coding

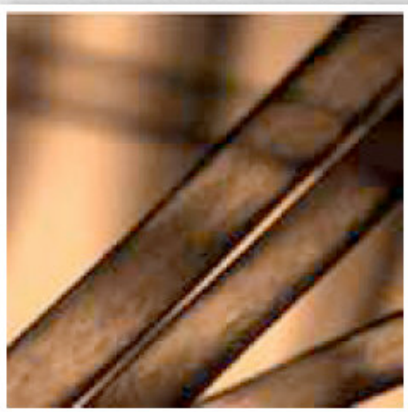
[Dowski 1995]

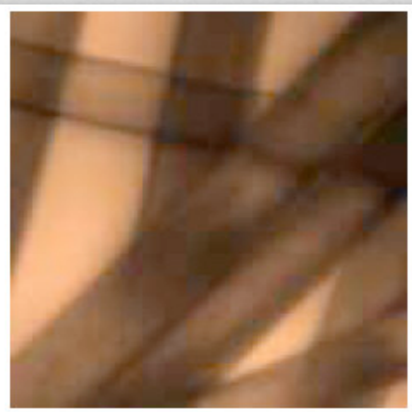

normal lens

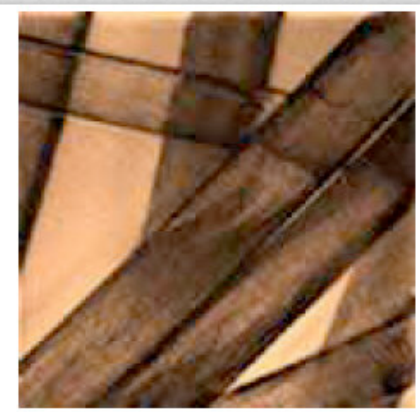

stopped down


wavefront coded


after deconvolution


normal


wavefront


deconvolved

Slide credits

- Andrew Adams